

ABSTRACT

Analysis of public information from social media could yield interesting results and insights into the world of public opinions about almost any product, service or personality. Social network data is one of the most effective and accurate indicators of public sentiment. As a result there has been an eruption of interest in people to mine these vast resources of data for opinions. Developing a program for sentiment analysis is an approach to be used to computationally measure customer's perceptions.

Nowadays, people from all around the world use social media sites to share information. Twitter for example is a platform in which users send, read posts known as 'tweets' and interact with different communities. Users share their daily lives, post their opinions on everything such as brands and places. Companies can benefit from this massive platform by collecting data related to opinions on them. The aim of this paper is to present a model that can perform sentiment analysis of real data collected from Twitter. Data in Twitter is highly unstructured which makes it difficult to analyze. However, our proposed model is different from prior work in this field because it combined the use of supervised and unsupervised machine learning algorithms. The process of performing sentiment analysis as follows: Tweet extracted directly from Twitter API, then cleaning and discovery of data performed. After that the data were fed into several models for the purpose of training. Each tweet extracted classified based on its sentiment whether it is a positive, negative or neutral. Data were collected on two subjects McDonalds and KFC to show which restaurant has more popularity. Different machine learning algorithms were used. The result from these models were tested using various testing metrics like cross validation and f-score. Moreover, our model demonstrates strong performance on mining texts extracted directly from Twitter.

Keywords— Sentiment analysis, social media, Twitter, tweets.

Project Hardware requirements:

Processor : Any Update Processor
Ram : Min 1 GB
Hard Disk : Min 100 GB

Project Software requirements:

Operating System : Windows family
Technology : Java (1.7/1.8)
Front-End Technologies : Html, Html-5, JavaScript, CSS.
Web Server : Tomcat 7/8
Database (Back – End) : My SQL5.5
IDE : EditPlus

-

1. Introduction to Project

Emotions play a major role in expressing the feeling among human beings. Sentiment analysis or opinion mining is the technique that is used to study people's emotions, attitudes, sentiments, evaluations, moods, and opinions. Sentiment analysis is the stem of natural language processing and machine learning methods, which is the current trending research area in the text mining. It is the important source of decision making and it can be extracted, identified, evaluated from the on line sentimental reviews. The content of user generated opinions in the social media such as facebook, twitter, review sites, etc are growing in large volume. These opinions can be tapped and used as business intelligence for various uses such as marketing, prediction, etc. Thus there is a need for efficient algorithms on sentimental analysis. Generally sentiment analysis is used for finding out the attitude of the author considering some topic. Social media is a platform that gives a chance to express the opinion of the people. The sentiment analysis can be applied on the social media platforms such as twitter, instagram, facebook, etc

The impression or opinion of the people with respect to some particular topic varies from person to person, thus it is necessary to collect all kinds of the opinions for finding the exact sentiment of the topic. The basic tasks of sentiment analysis are emotion recognition and polarity detection. The area of sentimental analysis has been extensively used in applications such as sentiment mining for analyzing reviews and opinion about products and also in trending topics such as political analysis, entertainment, etc.

word representation attempts to represent aspects of word meanings. For example, the representation of "cellphone" may capture the facts that cellphones are electronic products, that they include battery and screen, that they can be used to chat with others, and so on. Word representation is a critical component of many natural language processing systems [4], [5] as word is usually the basic computational unit of texts. A straight forward way is to represent each word as a one-hot vector, whose length is vocabulary size and only one dimension is 1, with all others being 0. However, one-hot word representation only encodes the indices of words in a vocabulary, but fails to capture rich relational structure of the lexicon. To solve this problem, many studies represent each word as a continuous, low-dimensional and real-valued vector, also known as word embeddings [6], [7], [8]. Existing embedding learning approaches are mostly on the basis of distributional hypothesis [9], which states that

the representations of words are reflected by their contexts. As a result, words with similar grammatical usages and semantic meanings, such as “hotel” and “motel”, are mapped into neighboring vectors in the embedding space. Since word embeddings capture semantic similarities between words, they have been leveraged as inputs or extra word features for a variety of natural language processing tasks, including machine translation [10], syntactic parsing [11], question answering [12], discourse parsing [13], etc. Despite the success of the context-based word embeddings in many NLP tasks [14], we argue that they are not effective enough if directly applied to sentiment analysis [15], [16], [17], which is the research area targeting at extracting, analyzing and organizing the sentiment/opinion (e.g. thumbs up or thumbs down) of texts. The most serious problem of context-based embedding learning algorithms is that they only model the contexts of words but ignore the sentiment information of text. As a result, words with opposite polarity, such as good and bad, are mapped into close vectors in the embedding space. This is meaningful for some tasks such as pos-tagging [18] because the two words have similar usages and grammatical roles. However, it becomes a disaster for sentiment analysis as they have opposite sentiment polarity labels.

Literature survey

1. Survey on Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification

Abstract:

We present a method that learns word embedding for Twitter sentiment classification in this paper. Most existing algorithms for learning continuous word representations typically only model the syntactic context of words but ignore the sentiment of text. This is problematic for sentiment analysis as they usually map words with similar syntactic context but opposite sentiment polarity, such as good and bad, to neighboring word vectors. We address this issue by learning sentiment specific word embedding (SSWE), which encodes sentiment information in the continuous representation of words. Specifically, we develop three neural networks to effectively incorporate the supervision from sentiment polarity of text (e.g. sentences or tweets) in their loss functions. To obtain large scale training corpora, we learn the sentiment-specific word embedding from massive distant-supervised tweets collected by positive and negative emoticons. Experiments on applying SSWE to a benchmark Twitter sentiment classification dataset in SemEval 2013 show that (1) the SSWE feature performs comparably with hand-crafted features in the top-performed system; (2) the performance is further improved by concatenating SWE with existing feature set.

Conclusion:

In this paper, we propose learning continuous word representations as features for Twitter sentiment classification under a supervised learning framework. We show that the word embedding learned by traditional neural networks are not effective enough for Twitter sentiment classification. These methods typically only model the context information of words so that they cannot distinguish words with similar context but opposite sentiment polarity (e.g. good and bad). We learn sentiment-specific word embedding (SSWE) by integrating the sentiment information into the loss functions of three neural networks. We train SSWE with massive distant-supervised tweets selected by positive and negative emoticons. The effectiveness of SSWE has been implicitly evaluated by using it as features in sentiment classification on the benchmark dataset in SemEval 2013, and explicitly verified by measuring word similarity in the embedding space for sentiment lexicons. Our unified model combining syntactic context of words and sentiment information of sentences yields the best performance in both experiments.

2. Survey on Coooolll: A Deep Learning System for Twitter Sentiment Classification

Abstract:

In this paper, we develop a deep learning system for message-level Twitter sentiment classification. Among the 45 submitted systems including the SemEval 2013 participants, our system (Coooolll) is ranked 2nd on the Twitter2014 test set of SemEval 2014 Task 9. Coooolll is built in a supervised learning framework by concatenating the sentiment-specific word embedding (SSWE) features with the state-of-the-art hand-crafted features. We develop a neural network with hybrid loss function 1 to learn SSWE, which encodes the sentiment information of tweets in the continuous representation of words. To obtain large-scale training corpora, we train SSWE from 10M tweets collected by positive and negative emoticons, without any manual annotation. Our system can be easily re-implemented with the publicly available sentiment-specific word embedding.

Conclusion:

We develop a deep learning system (Coooolll) for message-level Twitter sentiment classification in this paper. The feature representation of Coooolll is composed of two parts, a state-of-the-art hand-crafted features and the sentiment-specific word embedding (SSWE) features. The SSWE is learned from 10M tweets collected by positive and negative emoticons, without any manual annotation. The effectiveness of Coooolll has been verified in both positive/negative/neutral and positive/negative classification of tweets. Among 45 systems of SemEval 2014 Task 9 subtask(b), Coooolll yields Rank 2 on the Twitter2014 test set, along with the SemEval 2013 participants owning larger training data.

3. Survey on Building Large-Scale Twitter-Specific Sentiment Lexicon : A Representation Learning Approach

Abstract:

In this paper, we propose to build large-scale sentiment lexicon from Twitter with a representation learning approach. We cast sentiment lexicon learning as a phrase-level sentiment classification task. The challenges are developing effective feature representation of phrases and obtaining training data with minor manual annotations for building the sentiment classifier. Specifically, we develop a dedicated neural architecture and integrate the sentiment information of text (e.g. sentences or tweets) into its hybrid loss function for learning sentiment-specific phrase embedding (SSPE). The neural network is trained from massive tweets collected with positive and negative emoticons, without any manual annotation. Furthermore, we introduce the Urban Dictionary to expand a small number of sentiment seeds to obtain more training data for building the phrase-level sentiment classifier. We evaluate our sentiment lexicon (TS-Lex) by applying it in a supervised learning framework for Twitter sentiment classification. Experiment results on the benchmark dataset of SemEval 2013 show that, TS-Lex yields better performance than previously introduced sentiment lexicons.

Conclusion:

In this paper, we propose building large-scale Twitter-specific sentiment lexicon with a representation learning approach. Our method contains two parts: (1) a representation learning algorithm to effectively learn the embedding of phrases, which are used as features for classification, (2) a seed expansion algorithm that enlarge a small list of sentiment seeds to obtain training data for building the phrase-level sentiment classifier. We introduce a tailored neural architecture and integrate the sentiment information of tweets into its hybrid loss function for learning sentiment-specific phrase embedding (SSPE). We learn SSPE from the tweets collected by positive and negative emoticons, without any manual annotation. To collect more training data for building the phrase-level classifier, we utilize the similar words from Urban Dictionary to expand a small list of sentiment seeds. The effectiveness of our sentiment lexicon (TS-Lex) has been verified through applied in the supervised learning framework for Twitter sentiment classification. Experiment results on the benchmark dataset of SemEval 2013 show that, TSLex outperforms previously introduced sentiment lexicons and further improves the top-perform system in SemEval 2013 with feature combination. In

future work, we plan to apply TS-Lex into the unsupervised learning framework for Twitter sentiment classification.

4.Survey on Opinion Mining and Sentiment Analysis

Abstract:

An important part of our information-gathering behavior has always been to find out what other people think. With the growing availability and popularity of opinion-rich resources such as online review sites and personal blogs, new opportunities and challenges arise as people now can, and do, actively use information technologies to seek out and understand the opinions of others. The sudden eruption of activity in the area of opinion mining and sentiment analysis, which deals with the computational treatment of opinion, sentiment, and subjectivity in text, has thus occurred at least in part as a direct response to the surge of interest in new systems that deal directly with opinions as a first-class object. This survey covers techniques and approaches that promise to directly enable opinion-oriented information-seeking systems. Our focus is on methods that seek to address the new challenges raised by sentiment-aware applications, as compared to those that are already present in more traditional fact-based analysis. We include material on summarization of evaluative text and on broader issues regarding privacy, manipulation, and economic impact that the development of opinion-oriented information-access services gives rise to. To facilitate future work, a discussion of available resources, benchmark datasets, and evaluation campaigns is also provided.

Conclusion:

Our goal in this survey has been to cover techniques and approaches that promise to directly enable opinion-oriented information-seeking systems, and to convey to the reader a sense of our excitement about the intellectual richness and breadth of the area. We very much encourage the reader to take up the many open challenges that remain, and hope we have provided some resources that will prove helpful in this regard. On the topic of resources: we have already indicated above that the bibliographic database used in this survey is publicly available. In fact, the URL mentioned above, <http://www.cs.cornell.edu/home/lee/opinion-mining-sentiment-analysis-survey.html>, is our personally maintained homepage for this survey. Any subsequent editions or versions of this survey that may be produced, or related news, will be announced there. Speaking of resources, we have drawn considerably on those of many others during the course of this work. We thus have a number of sincere

acknowledgments to make. This survey is based upon work supported in part by the National Science Foundation under grant no. IIS-0329064, a Cornell University Provost's Award for Distinguished Scholarship, a Yahoo! Research Alliance gift, and an Alfred P. Sloan Research Fellowship. Any opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of any sponsoring institutions, the US government, or any other entity. We would like to wholeheartedly thank the anonymous referees, who provided outstanding feedback astonishingly quickly. Their insights contributed immensely to the final form of this survey on many levels. It is hard to describe our level of gratitude to them for their time and their wisdom, except to say this: we have, in various capacities, seen many examples of reviewing in the community, but this is the best we have ever encountered. We also thank Eric Breck for his careful reading of and commentary on portions of this survey. All remaining errors and faults are, of course, our own. We are also very thankful to Fabrizio Sebastiani, for all of his editorial guidance and care. We owe him a great debt. We also greatly appreciate the help we received from Jamie Callan, who, along with Fabrizio, serves as Editor in Chief of the Foundations and Trends in Information Retrieval series, and James Finlay, of Now Publishers, the publisher of this series. Finally, a number of unexpected health problems arose in our families during the writing of this survey. Despite this, it was our families who sustained us with their cheerful and unlimited support (on many levels), not the other way around. Thus — to end on a sentimental note — this work is dedicated to them.

5. Survey on Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis

Abstract:

This paper presents a new approach to phrase-level sentiment analysis that first determines whether an expression is neutral or polar and then disambiguates the polarity of the polar expressions. With this approach, the system is able to automatically identify the contextual polarity for a large subset of sentiment expressions, achieving results that are significantly better

than baseline.

Conclusions:

In this paper, we present a new approach to phrase-level sentiment analysis that first determines whether an expression is neutral or polar and then disambiguates the polarity of

the polar expressions. With this approach, we are able to automatically identify the contextual polarity for a large subset of sentiment expressions, achieving results that are significantly better than baseline.

2. SYSTEM ANALYSIS

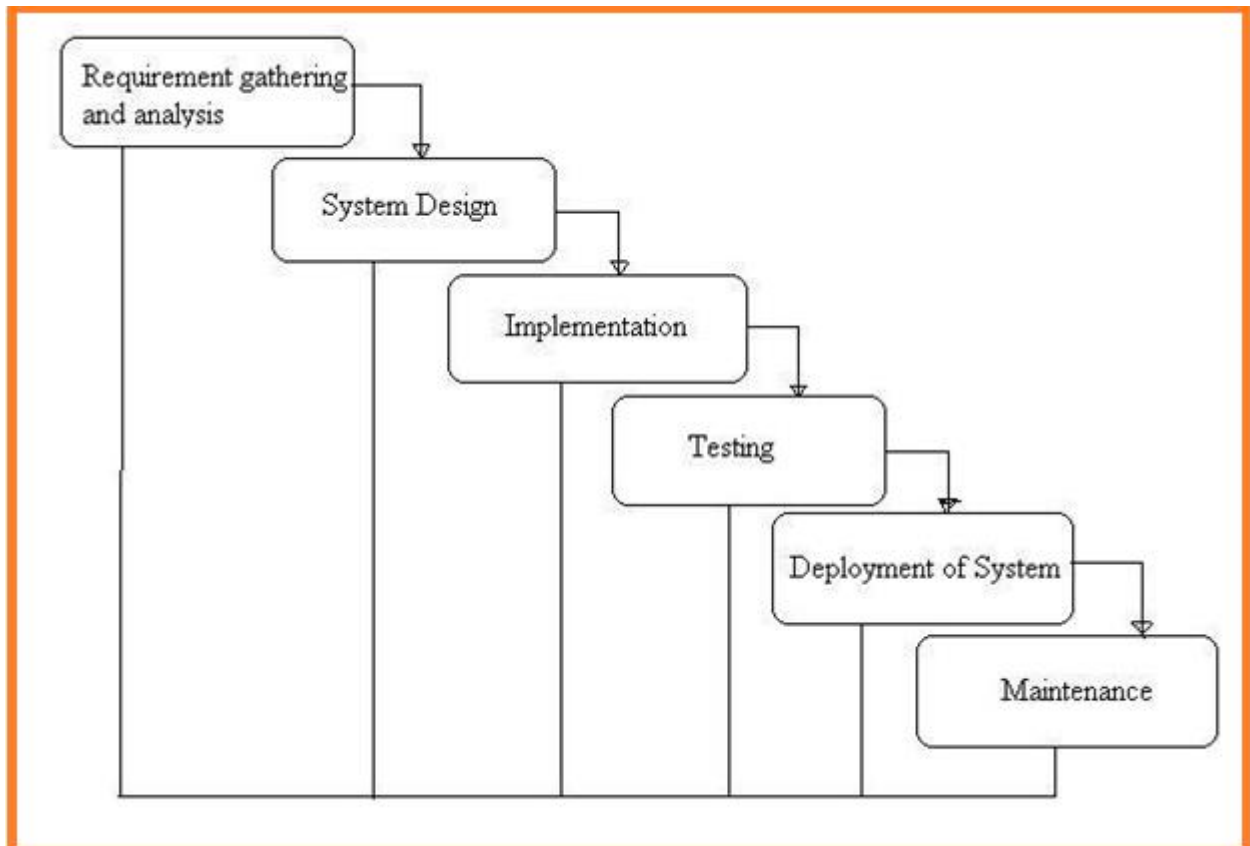


Fig:-1 Project SDLC

- Project Requisites Accumulating and Analysis
- Application System Design
- Practical Implementation
- Manual Testing of My Application
- Application Deployment of System
- Maintenance of the Project

Requisites Accumulating and Analysis

It's the first and foremost stage of the any project as our is a an academic leave for requisites amassing we followed of IEEE Journals and Amassed so many IEEE Relegated papers and final culled a Paper designated by setting and substance importance input and for analysis stage we took referees from the paper and did literature survey of some papers and amassed all the Requisites of the project in this stage

System Design

In System Design has divided into three types like GUI Designing, UML Designing with avails in development of project in facile way with different actor and its utilizer case by

utilizer case diagram, flow of the project utilizing sequence, Class diagram gives information about different class in the project with methods that have to be utilized in the project if comes to our project our UML Will utilizable in this way The third and post import for the project in system design is Data base design where we endeavor to design data base predicated on the number of modules in our project

Implementation

The Implementation is Phase where we endeavor to give the practical output of the work done in designing stage and most of Coding in Business logic lay coms into action in this stage its main and crucial part of the project

Testing

Unit Testing

It is done by the developer itself in every stage of the project and fine-tuning the bug and module predicated additionally done by the developer only here we are going to solve all the runtime errors

Manual Testing

As our Project is academic Leave we can do any automatic testing so we follow manual testing by endeavor and error methods

Deployment of System

Once the project is total yare we will come to deployment of client system in genuinely world as its academic leave we did deployment i our college lab only with all need Software's with having Windows OS

Maintenance

The Maintenance of our Project is one time process only

Functional Requirements

- **Admin**
- **Topic Creation**
- **Sentiment Analysis**

Application needs Non-Functional Requisites

Expanded System admin security: oversee to eschew the abuse of the application by PC ought to be exceptionally secured and available.

Compactness: The Presentation of this application is facile to utilize so it is looks simple for the using client to comprehend and react to identically tantamount.

Unwavering quality: and the functionalities accessible in the application thissubstructure has high probability to convey us the required inquiries.

Time take for Reaction: The time taken by the application to culminate an undertaking given by the client is very fast.

Multifariousness: Our application can be stretched out to incorporate the vicissitudes done by applications present now to enhance the performance of the item. This is implicatively insinuated for the future works that will be done on the application.

Vigor: The project is blame tolerant concerning illicit client/beneficiary sources of info. Blunder checking has been worked in the platforms to avert platforms disappointment.

3. IMPLEMENTATION

Sentiment Analysis

Sentiment Analysis is a process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral. While Sentiment Analysis we perform pre-processing steps like stop words removal after we perform Sentiment Analysis based on user's statements.

User

Here the user is end user of application, In this User module User authenticate his/her own username and password. User after his/her authenticating he/she can submit some statements of a topic and search a topic of statements. Based on these data only can perform Sentiment Analysis. User performs Sentiment Analysis for a particular topic.

Admin

Admin is a main authority of this application, he maintains all the data means user's data. Admin will filter positive and negative statements. And perform evolution measure of our application.

Algorithm for Sentiment Analysis

Input: Data T;

Output: Result R;

Initialization:

i. Let $T = \{ t_1, t_2, t_3, \dots, t_n \}$; total(n) data records. //Collection of tweets

ii. Let $D^* = \{ d_1^*, d_2^*, d_3^*, \dots, d_k^* \}$; total(k) Sentiment dictionary data records.

for each data T

//Pre-processing

$T_w = T.split("\\s+");$

for each T_w

compare with D^ ;*

if match is found

$$pos = \sum_{i=1}^k \binom{k}{k_i} ps // ps- positive score$$

$$neg = \sum_{i=1}^k \binom{k}{k_i} ns // ns- negative score$$

$R = [pos, neg];$

end if

end for;

end for;

return R;

End

Sample Code

Databasecon.java

```
package databaseconnection;
```

```
import java.sql.*;
```

```
public class databasecon
```

```
{
```

```
    static Connection con;
```

```
    public static Connection getConnection()
```

```
    {
```

```
        try
```

```
        {
```

```
            Class.forName("com.mysql.jdbc.Driver");
```

```
            con
```

```
=
```

```
            DriverManager.getConnection("jdbc:mysql://localhost:3306/sentiment_tweets","root",  
            "root");
```

```
        }
```

```
        catch(Exception e)
```

```
        {
```

```
            System.out.println("class error"+e);
```

```
        }
```

```
        return con;
```

```
    }
```

```
}
```


SearchTweets.java

package CT;

import twitter4j.*;

import java.io.*;

import twitter4j.auth.OAuth2Token;

import databaseconnection.databasecon;

import twitter4j.conf.ConfigurationBuilder;

import java.util.Map;

import java.util.Vector;

import java.sql.*;

public class SearchTweets {

private static PrintWriter pw=null;

**private static final String CONSUMER_KEY =
"3H8VEdM8tooU4iNgZbZNL0E1S";**

**private static final String CONSUMER_SECRET =
"MovNMWZWbj59F0SsP0IeNs2DKkQwlaca3VXi0o95VFVmoNsRcu";**

private static final int TWEETS_PER_QUERY = 100;

private static final int MAX_QUERIES = 1;

private static String SEARCH_TERM = "";

public static String cleanText(String text)

```

{
    text = text.replace("\n", "\\n");
    text = text.replace("\t", "\\t");
    return text;
}

```

```

public static OAuth2Token getOAuth2Token()
{
    OAuth2Token token = null;
    ConfigurationBuilder cb;

    cb = new ConfigurationBuilder();
    cb.setApplicationOnlyAuthEnabled(true);

    cb.setOAuthConsumerKey(CONSUMER_KEY).setOAuthConsumerSecret(CONSUMER_SECRET);

    try
    {
        token = new
TwitterFactory(cb.build()).getInstance().getOAuth2Token();
    }
    catch (Exception e)
    {
        System.out.println("Could not get OAuth2 token");
        e.printStackTrace();
    }
}

```

```

        System.exit(0);
    }

    return token;
}

public static Twitter getTwitter()
{
    OAuth2Token token;

    token = getOAuth2Token();

    ConfigurationBuilder cb = new ConfigurationBuilder();
    cb.setApplicationOnlyAuthEnabled(true);
    cb.setOAuthConsumerKey(CONSUMER_KEY);
    cb.setOAuthConsumerSecret(CONSUMER_SECRET);
    cb.setOAuth2TokenType(token.getTokenType());
    cb.setOAuth2AccessToken(token.getAccessToken());

    return new TwitterFactory(cb.build()).getInstance();
}

public static Vector<String> main(String args)throws Exception
{
    Vector<String> v=new Vector<String>();

```

```

Connection con=databasecon.getConnection();

PreparedStatementps=con.prepareStatement("delete from tweets");
// ps.executeUpdate();

ps=con.prepareStatement("insert into
tweets(sno,topic,user_,tweet,pos,neg,result) values(?,?,?,?,?,?,?)");

SEARCH_TERM=args;

int totalTweets = 0;

longmaxID = -1;

Twitter twitter = getTwitter();

int count=1;

try
{
    Map<String, RateLimitStatus>rateLimitStatus =
twitter.getRateLimitStatus("search");

    RateLimitStatussearchTweetsRateLimit =
rateLimitStatus.get("/search/tweets");

    System.out.printf("You have %d calls remaining out of %d, Limit
resets in %d seconds\n",

searchTweetsRateLimit.getRemaining(),

searchTweetsRateLimit.getLimit(),

searchTweetsRateLimit.getSecondsUntilReset());

```

```

        for (int queryNumber=0; queryNumber < MAX_QUERIES;
queryNumber++)
        {

            //                pw=new
PrintWriter("log"+queryNumber+".txt");

            System.out.printf("\n\n!!! Starting loop %d\n\n",
queryNumber);

            if (searchTweetsRateLimit.getRemaining() == 0)
            {

                System.out.printf("!!! Sleeping for %d seconds due
to rate limits\n", searchTweetsRateLimit.getSecondsUntilReset());

                Thread.sleep((searchTweetsRateLimit.getSecondsUntilReset()+2) * 1000l);

            }

            Query q = new Query(SEARCH_TERM);                //
Search for tweets that contains this term

            q.setCount(TWEETS_PER_QUERY);
            // How many tweets, max, to retrieve

            //q.ResultType.values();
            // Get all tweets

            q.setLang("en");
            // English language tweets, please

            if (maxID != -1)
            {

                q.setMaxId(maxID - 1);

            }

            QueryResult r = twitter.search(q);

```

```

        if (r.getTweets().size() == 0)
        {
            break;                // Nothing? We must be done
        }

        for (Status s: r.getTweets())                // Loop
through all the tweets...
        {
            totalTweets++;
            if (maxID == -1 || s.getId() < maxID)
            {
                maxID = s.getId();
            }

            pw.println(cleanText(s.getText()));
            ps.setInt(1, totalTweets);
            ps.setString(2, args);
            ps.setString(3, s.getUser().getScreenName());
            ps.setString(4, cleanText(s.getText()));
            ps.setDouble(5, 0.0);
            ps.setDouble(6, 0.0);
            ps.setString(7, "non");
            try{

                ps.executeUpdate();

                v.add((count++)+" "+s.getUser().getScreenName()+" ">> "+cleanText(s.getText()));

```

```

        }

        catch(Exception e){}

//        pw.println("At %s, @%-20s said: %s\n"+
//
//        s.getCreatedAt().toString()+s.getUser().getScreenName()+cleanText(s.getText()))
;

        }

//        pw.close();

        searchTweetsRateLimit = r.getRateLimitStatus();

    }

}

catch (Exception e)
{

    System.out.println("That didn't work well...wonder why?");

    e.printStackTrace();

}

System.out.printf("\n\nA total of %d tweets retrieved\n", totalTweets);

return v;

}

public static void main(String[] args)throws Exception
{

    main(args[0]);

}

```

}

SentiWordNetDemoCode.java

package CT;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

public class SentiWordNetDemoCode {

private Map<String, Double>pos;

private Map<String, Double>neg;

public SentiWordNetDemoCode(String pathToSWN) throws IOException {

// This is our main dictionary representation

pos = new HashMap<String, Double>();

neg = new HashMap<String, Double>();

// From String to list of doubles.

BufferedReader csv = null;

try {

csv = new BufferedReader(new FileReader(pathToSWN));

int lineNumber = 0;

int i = 0;

String line;

```

while ((line = csv.readLine()) != null) {

    lineNumber++;

    if (!line.trim().startsWith("#")) {

        String[] data = line.split("\t");

        String wordTypeMarker = data[0];

        if (data.length != 6) {

            throw new IllegalArgumentException(

                "Incorrect tabulation format in

file, line: "

                + lineNumber);

        }

        // System.out.println(data[4]+"");

        String data1[]=data[4].split("\\s+");

        String data2[]=null;

        String data3[]=null;

        data2=data[4].split("#");

        // System.out.println(data2[0]+"-----"+data[2]+"-----"+data[3]);

```

```

                                Object
o=pos.put(data2[0],Double.parseDouble(data[2]));

                                if(o==null){

                                }else{

                                double d1=(double)o;

                                if(d1>Double.parseDouble(data[2]))

                                    pos.put(data2[0],d1);

                                }

//                                neg.put(data2[0],Double.parseDouble(data[3]));

                                Object
o2=neg.put(data2[0],Double.parseDouble(data[3]));

                                if(o2==null){

                                }else{

                                double d1=(double)o2;

                                if(d1>Double.parseDouble(data[3]))

                                    neg.put(data2[0],d1);

                                }

                                }

                                }} catch (Exception e) {

                                e.printStackTrace();

                                } finally {

```

```

        if (csv != null) {
            csv.close();
        }
    }
}

public double posextract(String word) {
    return pos.get(word);
}

public double negextract(String word) {
    return neg.get(word);
}

public static void main(String [] args) throws IOException {
    if(args.length<1) {
        System.err.println("Usage:
SentiWordNetDemoCode<pathToSentiWordNetFile>");
        return;
    }

    String pathToSWN = args[0];

    SentiWordNetDemoCode sentiwordnet = new
SentiWordNetDemoCode(pathToSWN);

    // System.out.println(sentiwordnet.posextract("good"));
    System.out.println(sentiwordnet.negextract("not bad"));

    // System.out.println("good#a "+sentiwordnet.extract("good", "a"));

```

```
        //      System.out.println("bad#a "+sentiwordnet.extract("spare", "a"));
    }
}
```

WordsReader.java

package Tag;

import wordcram.Word;

import java.io.*;

import java.util.ArrayList;

/**

*** Helper to read list of Words with initial weights from a simple Textfile.**

*** Supported text format:**

*** WORD1,WEIGHT1**

*** WORD2,WEIGHT2**

*** WORD must not contain a comma, but can include a space character.**

*** WEIGHT will be parsed as a Float**

***/**

public class WordsReader {

private static final String UTF8 = "UTF-8";

private String file;

public static final String SEPARATOR = ",";

/**

*** Constructs a reader**

```

    * @param file to be parsed
    */

public WordsReader(String file){
this.file = file;

}

/**
    * Read the file and parse a list of words.
    * @return an array of weighted words.
    */

public Word[] getWords(){
    ArrayList<Word> words = new ArrayList<Word>();

    BufferedReader reader;

    try {

        reader = new BufferedReader(new InputStreamReader(new FileInputStream(file),
        UTF8));

        } catch (FileNotFoundException e) {
e.printStackTrace();
return new Word[0];

        } catch (UnsupportedEncodingException e) {
e.printStackTrace();
return new Word[0];

        }

    String line;

    try {

```

```

while((line = reader.readLine()) != null){
    Word word = parse(line);
    if(word != null)
        words.add(word);
    }
    } catch (IOException e) {
e.printStackTrace();
return new Word[0];
    }

return words.toArray(new Word[words.size()]);
    }

```

```

private Word parse(String line){
String[] values = line.split(SEPARATOR);
if(values.length == 2){
return new Word(values[0].trim(), Float.parseFloat(values[1].trim()));
    }
else {
return null;
    }
    }
}

```

About Project Software's

JAVA, Apache Server, MSQL, EDIT ++

In our web Application Development we are using one tier architecture as total applicant will be developed in single system with all the three layers of application development like presentation layer where we use our web technologies to make of GUI of the application like HTML, HTML-5, CSS, JS Etc. and in second layer we have to make our business logic or called as implementation of application where we are using java, J2EE and also we use JDBC to connect from our Business layer to data base layer and final our data base layer where we develop the Data structure of the application

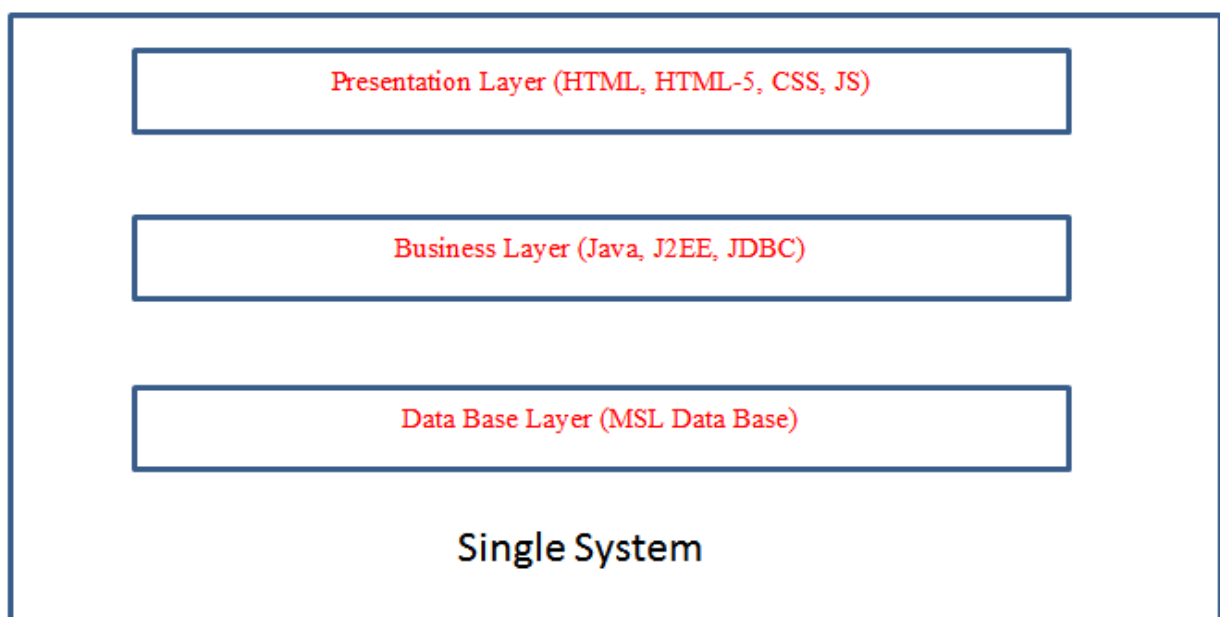


Fig:-3 Single tire Architecture Project Development

How we used java in our Project Development

Installation and Setup in our system

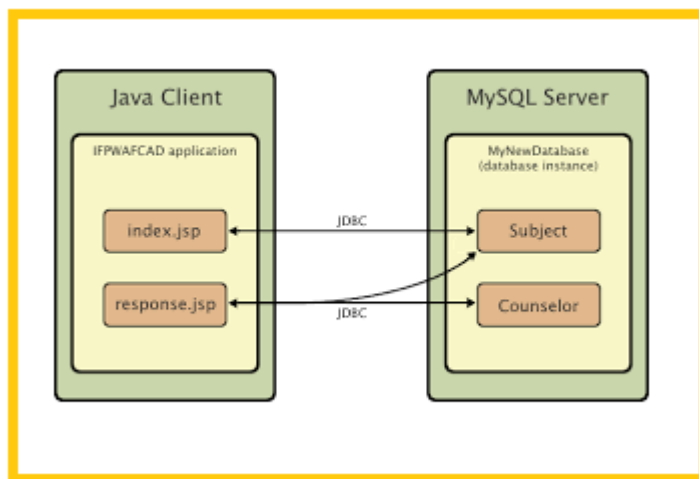
The Software we download from the oracle website as it's an open source as per the software we have installed it in our system and for we have set the system path of java in our OS location We have used the main logic of our algorithm by core java concepts only for web application we have used all JSP concepts and to connect data base we have used JDBC with all this concept we have done the application in Single tire Architecture Project

Data Storage in MYSQL

We have taken open source software MYSQL from the provide website and run in our system we used for creating our project data base related tables as per project requirement's even for user friendly access of my sql we used Software called SQL Yog where we can do all the operation of mysql by click & use

About the role of apache tomcat webserver

As our project is a web applicant we need webserver so for that we used again open sour software where our total project source code will be in webapps of the server form that location the application run into web browser where users can see the implementation of the total project



Application Development Structure

Client Server

Over view:

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired

a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine revealed that 76% of its readers were actively looking at the client server solution. The growth in the client server development tools from \$200 million in 1992 to more than \$1.2 billion in 1996.

Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE.

The typical client either a PC or a Work Station connected through a network to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request.

The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging heterogeneous table joins.

What is a Client Server

Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison differs there! The file server simply provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access, Record modifying, Insert, Delete with full relational integrity backup/ restore performance for high volume of transactions, etc. the client server middleware provides a flexible interface between client and server, who does what, when and to whom.

Why Client Server

Client server has evolved to solve a problem that has been around since the earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental and enterprise wide data processing. During mainframe era choices were quite limited. A central machine housed both the CPU

and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions therefore could best be characterized as “SLAVE-MASTER”.

Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with non-programmer query languages, online users could formulate adhoc queries and produce local reports with out adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

Front end or User Interface Design

The entire user interface is planned to be developed in browser specific environment with a touch of Intranet-Based Architecture for achieving the Distributed Concept.

The browser specific components are designed by using the HTML standards, and the dynamism of the designed by concentrating on the constructs of the Java Server Pages.

Communication or Database Connectivity Tier

The Communication architecture is designed by concentrating on the Standards of Servlets and Enterprise Java Beans. The database connectivity is established by using the Java Data Base Connectivity.

The standards of three-tire architecture are given major concentration to keep the standards of higher cohesion and limited coupling for effectiveness of the operations.

Features of The Language Used

In my project, I have chosen *Java* language for developing the code.

About Java

Initially the language was called as “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer’s language.
- Java is cohesive and consistent.
- Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

Finally, Java is to Internet programming where C was to system programming.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability. But, Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Java can be used to create two types of programs

Applications and Applets: An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java's ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a Java –compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

Features Of Java

Security

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scanned them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer.

When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

Portability

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java's solution to these two problems is both elegant and efficient.

The Byte code

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code.

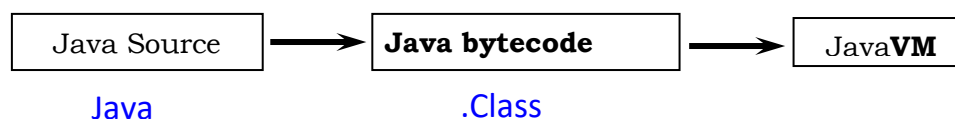
Translating a Java program into byte code helps makes it much easier to run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

Java, Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

Overall Description



Picture showing the development process of JAVA Program

Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a . Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a . class file, which contains the byte code. The . Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

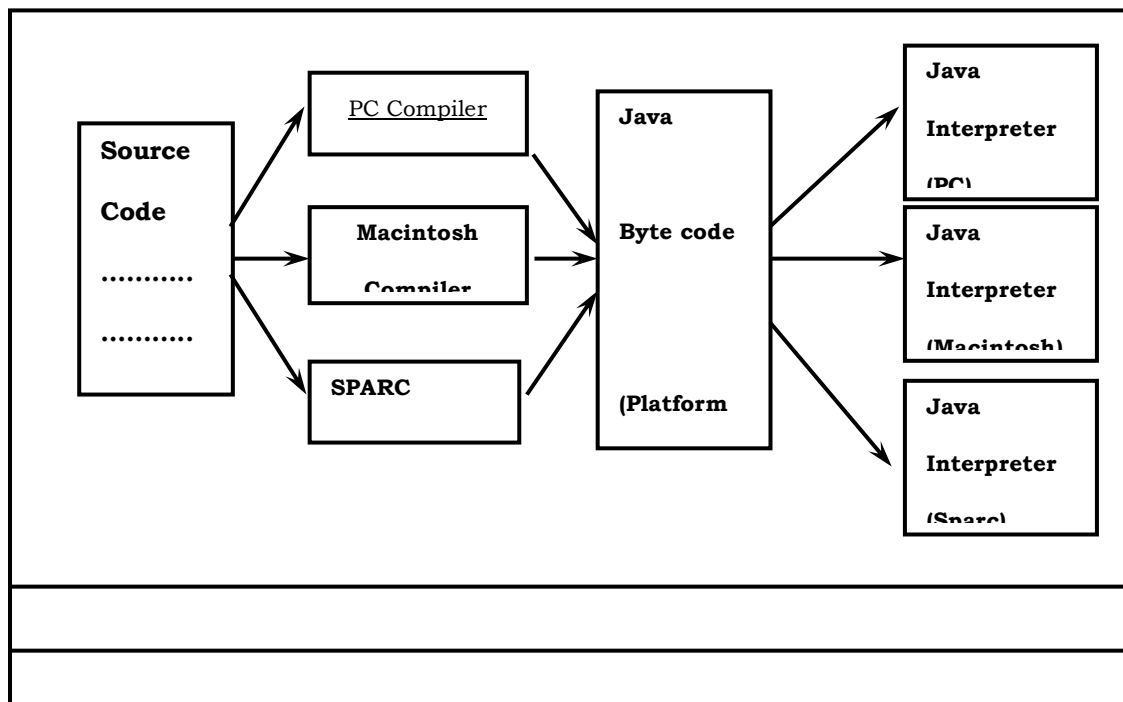
Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

Compiling and interpreting Java Source Code



During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or Sun SARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object-Oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

Java virtually eliminates the problems of memory management and deallocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

JAVASCRIPT

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java. JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then updates the browser's display accordingly. Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

```
<SCRIPTS>..  
</SCRIPT>.
```

```
<SCRIPT LANGUAGE = "JavaScript">
```

```
JavaScript statements
```

```
</SCRIPT>
```

Here are a few things we can do with JavaScript :

- Validate the contents of a form and make calculations.
- Add scrolling or changing messages to the Browser's status line.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

J a v a S c r i p t V s J a v a

JavaScript and Java are entirely different languages. A few of the most glaring differences are:

- Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.
- While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

A D V A N T A G E S

- JavaScript can be used for Sever-side and Client-side scripting.
- It is ~~more~~ flexible than VBScript.
- JavaScript is the default scripting languages at Client-side since all the browsers supports it.

Hyper Text Markup Language

Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

Basic HTML Tags :

<!-- -->	Specifies comments
<A>.....	Creates hypertext links
.....	Formats text as bold
<BIG>.....</BIG>	Formats text in large font.
<BODY>...</BODY>	Contains all tags and text in the HTML document
<CENTER>...</CENTER>	Creates text
<DD>...</DD>	Definition of a term
<DL>...</DL>	Creates definition list
...	Formats text with a particular font
<FORM>...</FORM>	Encloses a fill-out form
<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>...</H#>	Creates headings of different levels
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HR>...</HR>	Creates a horizontal rule
<HTML>...</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>...</SCRIPT>	Contains client-side or server-side script
<TABLE>...</TABLE>	Creates a table
<TD>...</TD>	Indicates table data in a table

<code><TR>...</TR></code>	Designates a table row
<code><TH>...</TH></code>	Creates a heading in a table

ADVANTAGES

- A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- HTML is platform independent.
- HTML tags are not case-sensitive.

Java Database Connectivity

What Is JDBC?

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere.

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things:

- Establish a connection with a database
- Send SQL statements
- Process the results.

JDBC versus ODBC and other APIs

At this point, Microsoft's ODBC (Open Database Connectivity) API is that probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms.

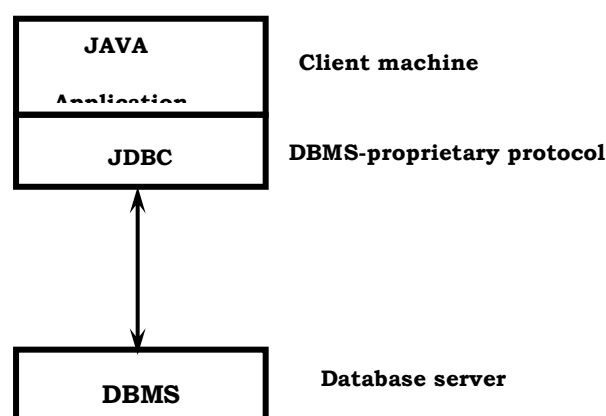
So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes "Why do you need JDBC?" There are several answers to this question:

1. ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.
2. A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void *". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.
3. ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.
4. A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

Two-tier and Three-tier Models

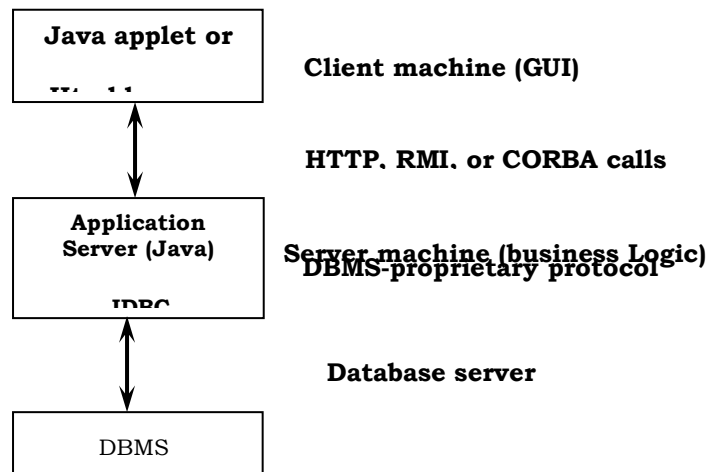
The JDBC API supports both two-tier and three-tier models for database access.

In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or



it can be the Internet.

In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control



over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages.

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.

JDBC Driver Types

The JDBC drivers that we are aware of at this time fit into one of four categories:

- JDBC-ODBC bridge plus ODBC driver
- Native-API partly-Java driver
- JDBC-Net pure Java driver
- Native-protocol pure Java driver

JDBC-ODBC Bridge

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

What Is the JDBC- ODBC Bridge?

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the

sun.jdbc.odbc Java package and contains a native library used to access ODBC. The Bridge is a joint development of Intersolv and JavaSoft.

Java Server Pages (JSP)

Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and a mature re-usable component model .The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches, it also allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout, and web application designers on programming, with minimal concern about impacting each other's work.

Features of JSP

Portability:

Java Server Pages files can be run on any web server or web-enabled application server that provides support for them. Dubbed the JSP engine, this support involves recognition, translation, and management of the Java Server Page lifecycle and its interaction components.

Components

It was mentioned earlier that the Java Server Pages architecture can include reusable Java components. The architecture also allows for the embedding of a scripting language directly into the Java Server Pages file. The components currently supported include Java Beans, and Servlets.

Processing

A Java Server Pages file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a JSP extension to the server as a Java Server Pages file. Before the page is served, the Java Server Pages syntax is parsed and processed into a Servlet on the server side. The Servlet that is generated outputs real content in straight HTML for responding to the client.

Access Models:

A Java Server Pages file may be accessed in at least two different ways. A client's request comes directly into a Java Server Page. In this scenario, suppose the page accesses reusable Java Bean components that perform particular well-defined computations like accessing a database. The result of the Beans computations, called result sets is stored within the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.

In both of the above cases, the page could also contain any valid Java code. Java Server Pages architecture encourages separation of content from presentation.

Steps in the execution of a JSP Application:

1. The client sends a request to the web server for a JSP file by giving the name of the JSP file within the form tag of a HTML page.
2. This request is transferred to the JavaWebServer. At the server side JavaWebServer receives the request and if it is a request for a jsp file server gives this request to the JSP engine.
3. JSP engine is program which can understand the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the JavaWebServer and then it is transferred back to the result is given back to the JavaWebServer and then it is transferred back to the client.

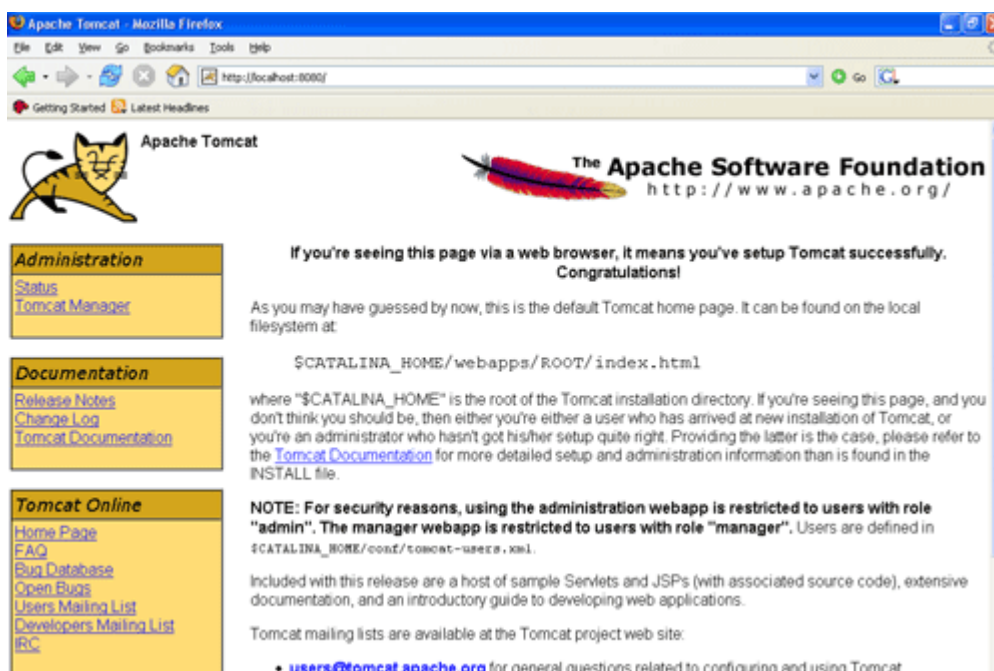
JDBC connectivity

The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

- Perform connection and authentication to a database server
- Manager transactions
- Move SQL statements to a database engine for preprocessing and execution
- Execute stored procedures
- Inspect and modify the results from Select statements.

Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server). To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.



Bibliography:

References for the Project Development were taken from the following Books and Web Sites.

Oracle

PL/SQL Programming by Scott Urman

SQL complete reference by Livion

JAVA Technologies

JAVA Complete Reference

Java Script Programming by Yehuda Shiran

Mastering JAVA Security

JAVA2 Networking by Pistoria

JAVA Security by Scotl oaks

Head First EJB Sierra Bates

J2EE Professional by Shadabsiddiqui

JAVA server pages by Larne Pekowsley

JAVA Server pages by Nick Todd

HTML

HTML Black Book by Holzner

JDBC

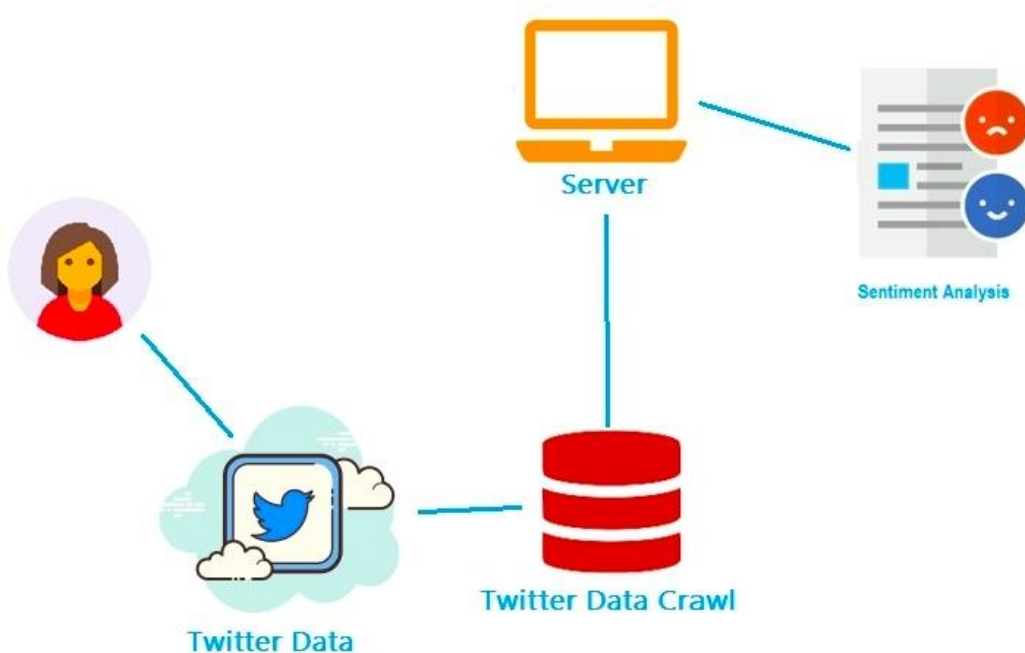
Java Database Programming with JDBC by Patel moss.

Software Engineering by Roger Pressman

SYSTEM DESIGN

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

SYSTEM ARCHITECTURE:



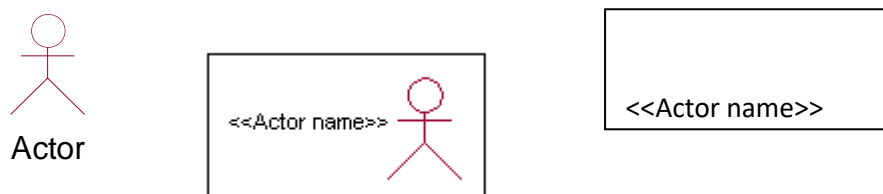
UML DIAGRAMS

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:



An actor is someone or something that:

Interacts with or uses the system.

- Provides input to and receives information from the system.
- Is external to the system and has no control over the use cases.

Actors are discovered by examining:

- Who directly uses the system?
- Who is responsible for maintaining the system?
- External hardware used by the system.
- Other systems that need to interact with the system.

Questions to identify actors:

- Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
- Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
- Which external hardware or systems (if any) use the system to perform tasks?
- What problems does this application solve (that is, for whom)?
- And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

- a. System Administrator**
- b. Customer**
- c. Customer Care**

Identification of usecases:

Usecase: A use case can be described as a specific way of using the system from a user's (actor's) perspective.

Graphical representation:



A more detailed description might characterize a use case as:

- Pattern of behavior the system exhibits
- A sequence of related transactions performed by an actor and the system
- Delivering something of value to the actor

Use cases provide a means to:

- capture system requirements
- communicate with the end users and domain experts
- test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

- For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
- Name the use cases.
- Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases:

- What are the tasks of each actor?
- Will any actor create, store, change, remove or read information in the system?
- What use case will store, change, remove or read this information?
- Will any actor need to inform the system about sudden external changes?
- Does any actor need to inform about certain occurrences in the system?
- What usecases will support and maintains the system?

1.2 Flow of Events

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

Construction of Usecase diagrams:

Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may depict all or some of the use cases of a system.

A use-case diagram can contain:

- actors ("things" outside the system)
- use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

Relationships in usecases:

1. Communication:

The communication relationship of an actor in a usecase is shown by connecting the actor symbol to the usecase symbol with a solid path. The actor is said to communicate with the usecase.

2. Uses:

A Uses relationship between the usecases is shown by generalization arrow from the usecase.

3. Extends:

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence it is like subclass.

SEQUENCE DIAGRAMS

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

Object:

An object has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance.

The object icon is similar to a class icon except that the name is underlined:

An object's concurrency is defined by the concurrency of its class.

Message:

A message is the communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

Link:

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If an object links to itself, use the loop version of the icon.

CLASS DIAGRAM:

Identification of analysis classes:

A class is a set of objects that share a common structure and common behavior (the same attributes, operations, relationships and semantics). A class is an abstraction of real-world items.

There are 4 approaches for identifying classes:

- a. Noun phrase approach:
- b. Common class pattern approach.
- c. Use case Driven Sequence or Collaboration approach.
- d. Classes , Responsibilities and collaborators Approach

1. Noun Phrase Approach:

The guidelines for identifying the classes:

- Look for nouns and noun phrases in the usecases.
- Some classes are implicit or taken from general knowledge.
- All classes must make sense in the application domain; Avoid computer implementation classes – defer them to the design stage.
- Carefully choose and define the class names After identifying the classes we have to eliminate the following types of classes:
 - Adjective classes.

2. Common class pattern approach:

The following are the patterns for finding the candidate classes:

- Concept class.
- Events class.
- Organization class
- Peoples class
- Places class
- Tangible things and devices class.

3. Use case driven approach:

We have to draw the sequence diagram or collaboration diagram. If there is need for some classes to represent some functionality then add new classes which perform those functionalities.

4. CRC approach:

The process consists of the following steps:

- Identify classes' responsibilities (and identify the classes)
- Assign the responsibilities
- Identify the collaborators.

Identification of responsibilities of each class:

The questions that should be answered to identify the attributes and methods of a class respectively are:

- a. What information about an object should we keep track of?
- b. What services must a class provide?

Identification of relationships among the classes:

Three types of relationships among the objects are:

Association: How objects are associated?

Super-sub structure: How are objects organized into super classes and sub classes?

Aggregation: What is the composition of the complex classes?

Association:

The **questions** that will help us to identify the associations are:

- a. Is the class capable of fulfilling the required task by itself?
- b. If not, what does it need?
- c. From what other classes can it acquire what it needs?

Guidelines for identifying the tentative associations:

- A dependency between two or more classes may be an association. Association often corresponds to a verb or prepositional phrase.
- A reference from one class to another is an association. Some associations are implicit or taken from general knowledge.

Some common association patterns are:

Location association like part of, next to, contained in.....

Communication association like talk to, order to

We have to eliminate the unnecessary association like implementation associations, ternary or n-ary associations and derived associations.

Super-sub class relationships:

Super-sub class hierarchy is a relationship between classes where one class is the parent class of another class (derived class). This is based on inheritance.

Guidelines for identifying the super-sub relationship, a generalization are

1. **Top-down:**

Look for noun phrases composed of various adjectives in a class name. Avoid excessive refinement. Specialize only when the sub classes have significant behavior.

2.Bottom-up:

Look for classes with similar attributes or methods. Group them by moving the common attributes and methods to an abstract class. You may have to alter the definitions a bit.

3.Reusability:

Move the attributes and methods as high as possible in the hierarchy.

4. Multiple inheritances:

Avoid excessive use of multiple inheritances. One way of getting benefits of multiple inheritances is to inherit from the most appropriate class and add an object of another class as an attribute.

Aggregation or a-part-of relationship:

It represents the situation where a class consists of several component classes. A class that is composed of other classes doesn't behave like its parts. It behaves very differently. The major properties of this relationship are transitivity and anti symmetry.

The **questions** whose answers will determine the distinction between the part and whole relationships are:

- Does the part class belong to the problem domain?
- Is the part class within the system's responsibilities?
- Does the part class capture more than a single value?(If not then simply include it as an attribute of the whole class)
- Does it provide a useful abstraction in dealing with the problem domain?

There are three types of aggregation relationships. They are:

Assembly:

It is constructed from its parts and an assembly-partsituation physically exists.

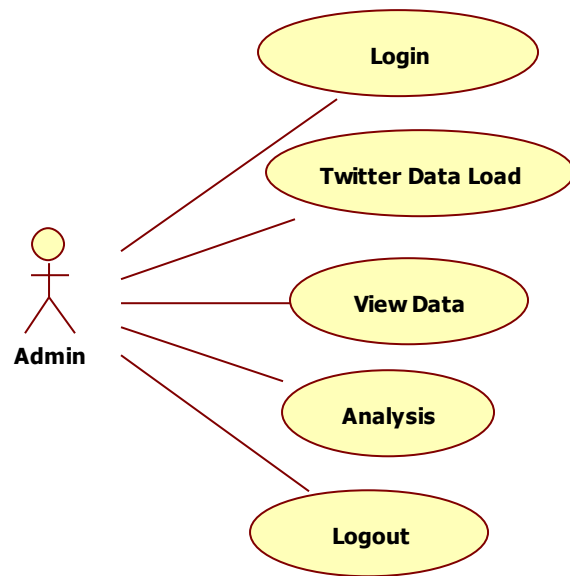
Container:

A physical whole encompasses but is not constructed from physical parts.

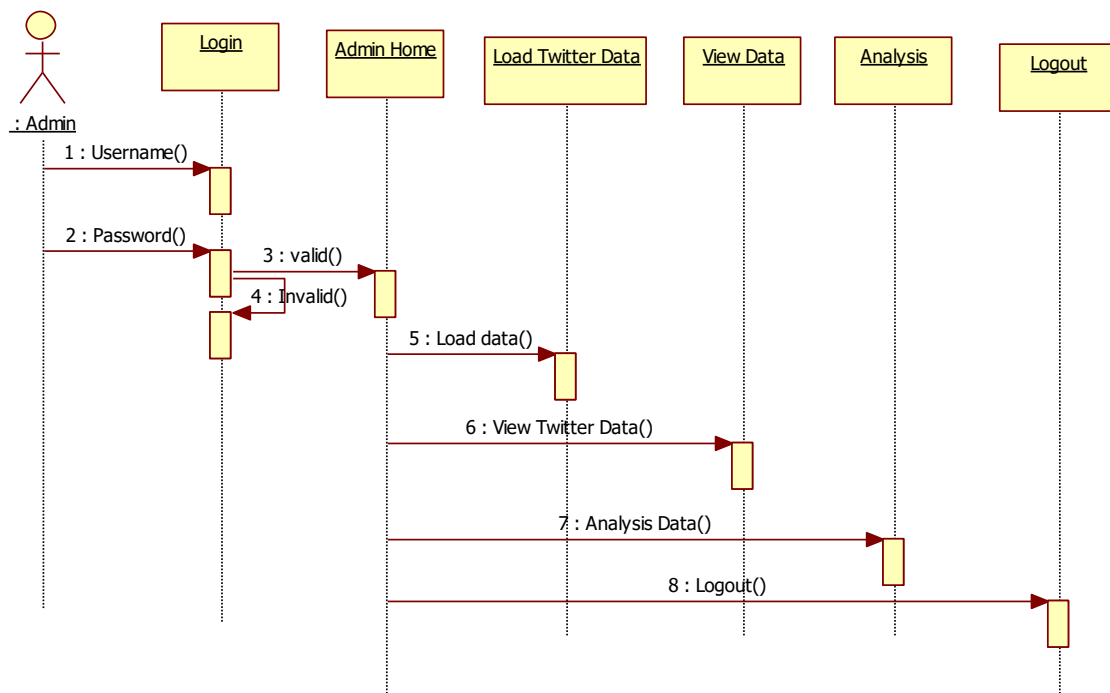
Collection member:

A conceptual whole encompasses parts that may be physical or conceptual. The container and collection are represented by hollow diamonds but composition is represented by solid diamond.

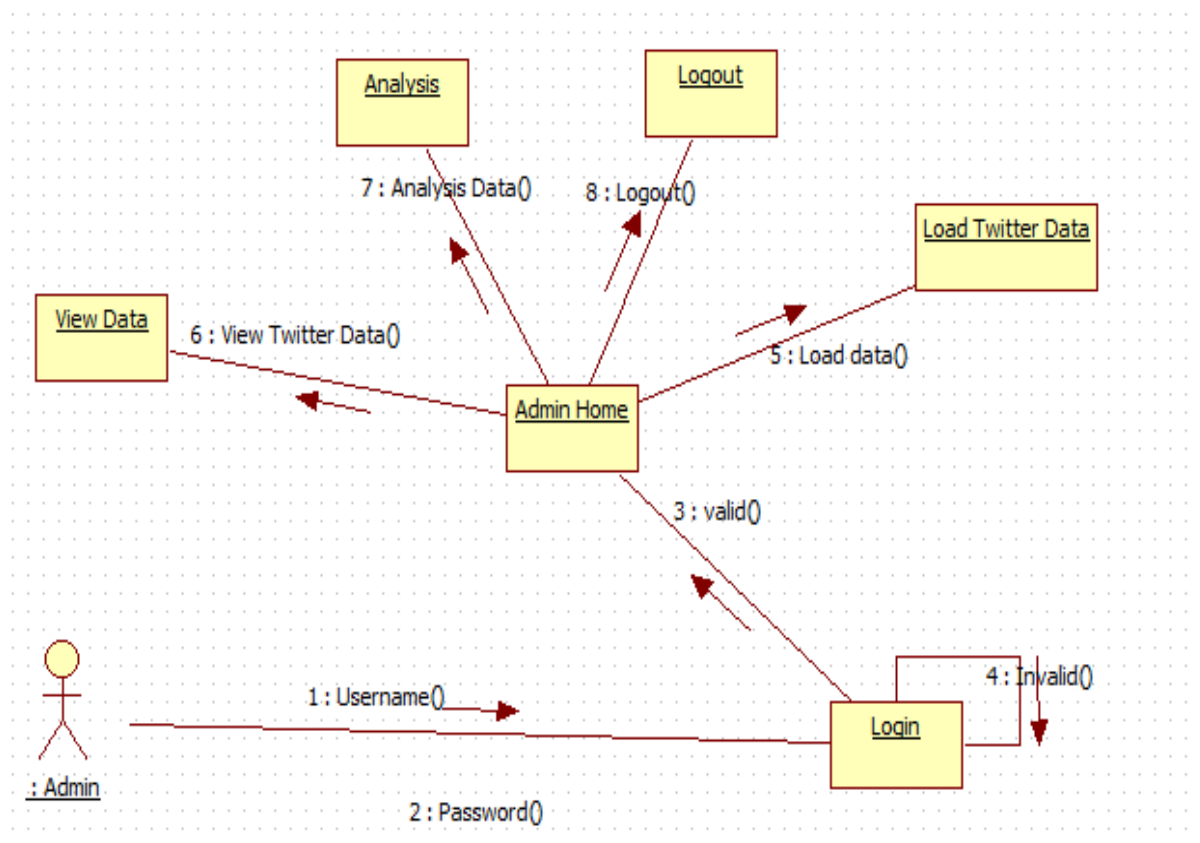
UseCase Diagram



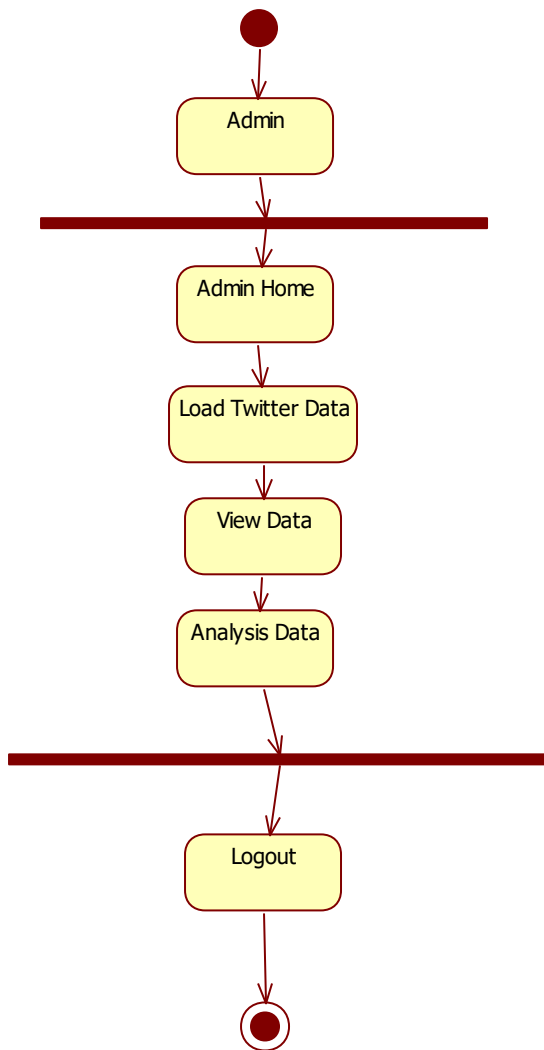
Sequence Diagram:



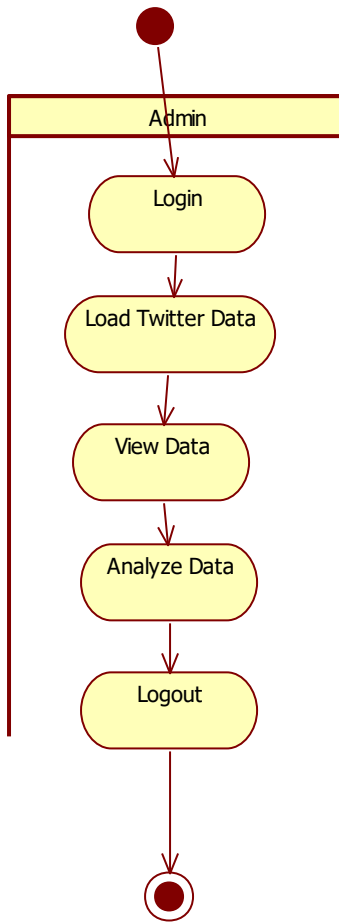
Collaboration Diagram:



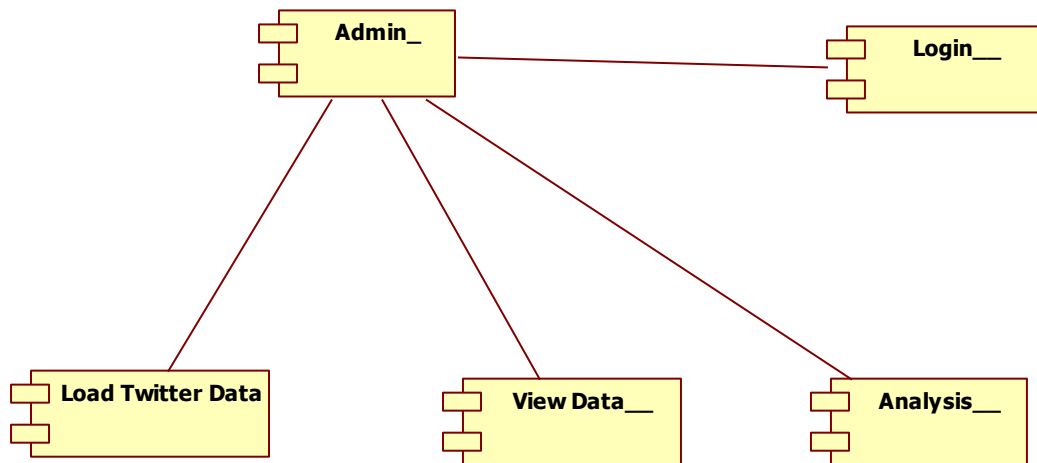
StateChart Diagram:



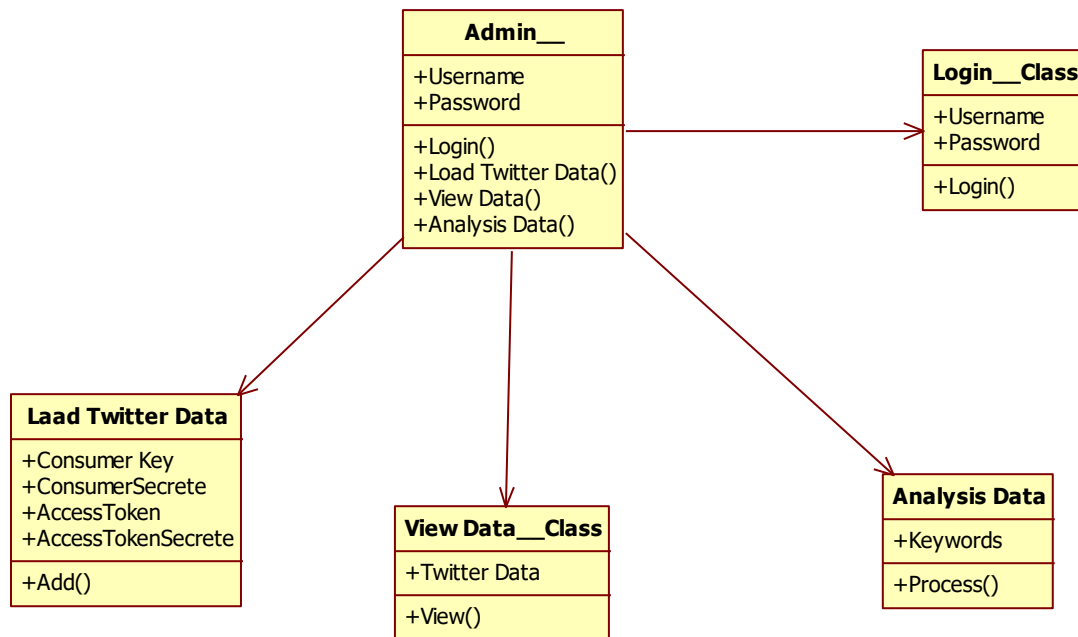
ActivityDiagram:



ComponentDiagram:



Class



Deployment Diagram:

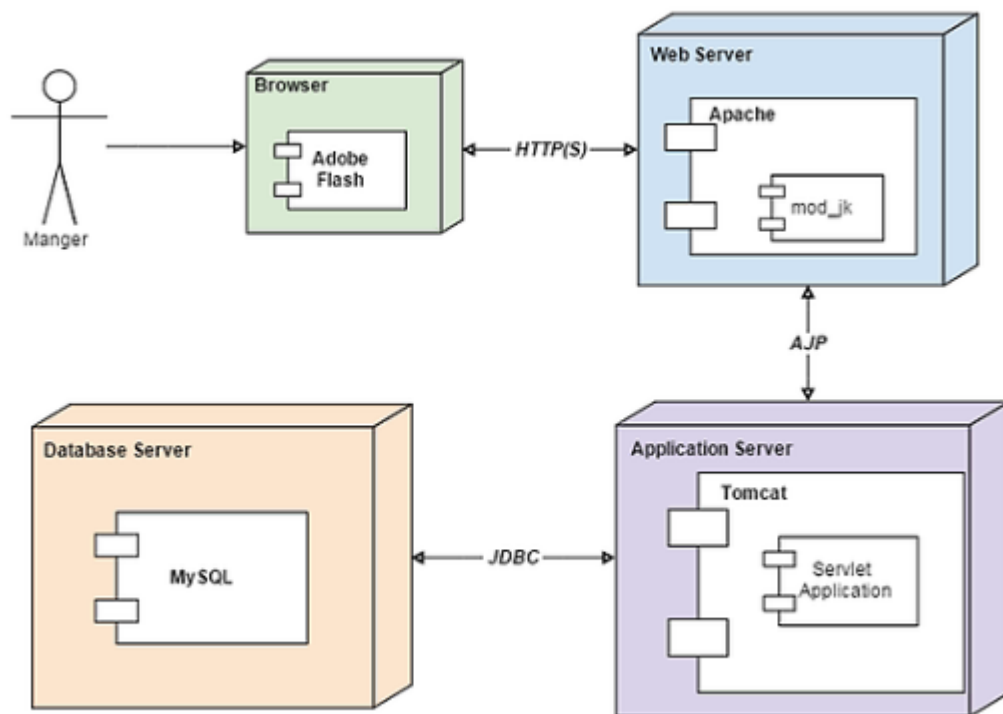
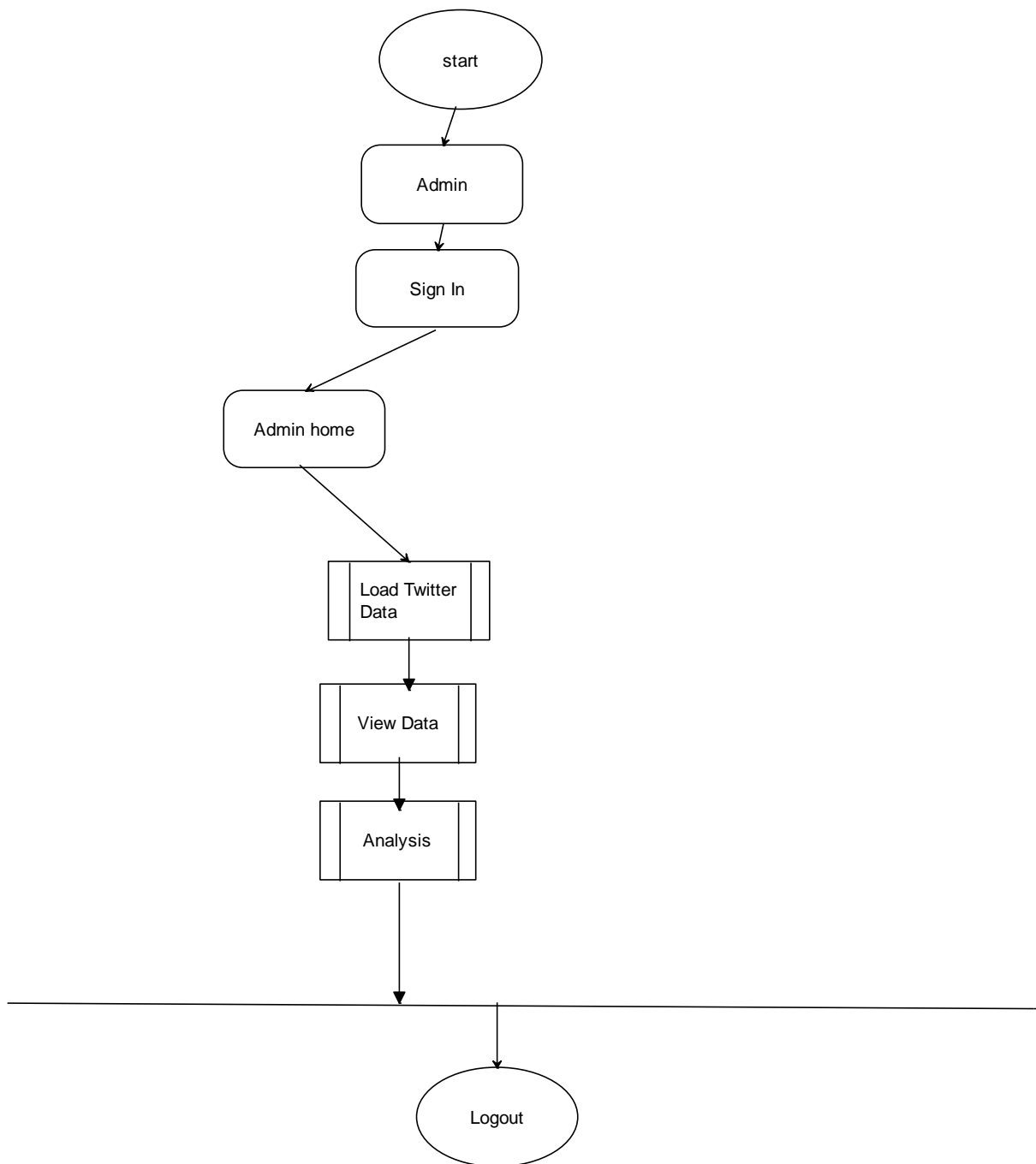


Fig: - Java Application Deployment Diagram

DFD:



Data Base design

Entity-relationship model

A relationship is how the data is shared between entities. There are three types of relationships between entities:

1. One-to-One

One instance of an entity (A) is associated with one other instance of another entity (B). For example, in a database of employees, each employee name (A) is associated with only one social security number (B).

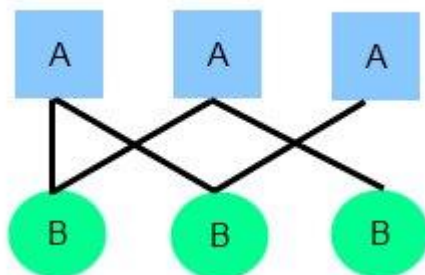


2. One-to-Many

One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entity A. For example, for a company with all employees working in one building, the building name (A) is associated with many different employees (B), but those employees all share the same singular association with entity A.

3. Many-to-Many

One instance of an entity (A) is associated with one, zero or many instances of another entity (B), and one instance of entity B is associated with one, zero or many instances of entity A. For example, for a company in which all of its employees work on multiple projects, each instance of an employee (A) is associated with many instances of a project (B), and at the same time, each instance of a project (B) has multiple employees (A) associated with it.



ER-Diagram

ER

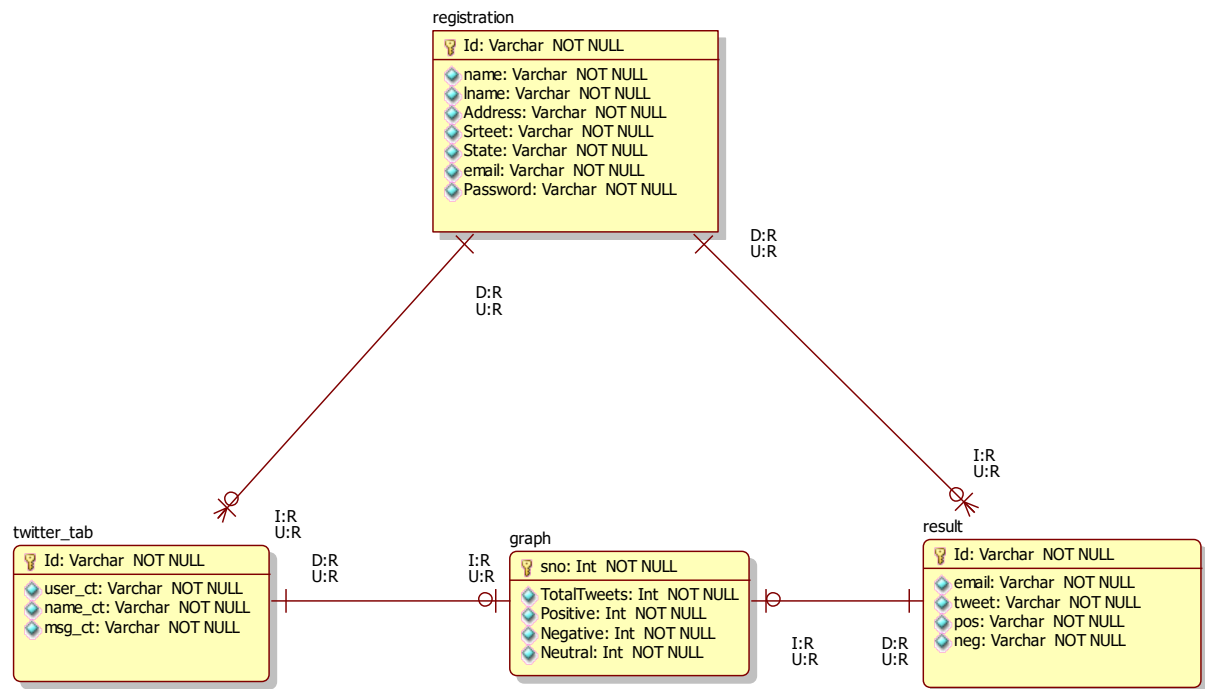


Fig: Project ER-Diagram

Data Base Tables

```
CREATE TABLE `analysis` (  
  `topic` varchar(100) NOT NULL,  
  `subtopic` varchar(100) NOT NULL,  
  `tot` varchar(100) DEFAULT NULL,  
  `pos` varchar(100) DEFAULT NULL,  
  `neg` varchar(100) DEFAULT NULL,  
  `nue` varchar(100) DEFAULT NULL,  
  `fileid` varchar(100) DEFAULT NULL,  
  PRIMARY KEY (`topic`,`subtopic`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

/*Table structure for table `temp` */

```
DROP TABLE IF EXISTS `temp`;
```

```
CREATE TABLE `temp` (  
  `tag_img` longblob  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

/*Table structure for table `topic` */

```
DROP TABLE IF EXISTS `topic`;
```

```
CREATE TABLE `topic` (  
  `topic` varchar(300) NOT NULL,  
  PRIMARY KEY (`topic`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
/*Table structure for table `tweets` */
```

```
DROP TABLE IF EXISTS `tweets`;
```

```
CREATE TABLE `tweets` (  
  `sno` int(11) DEFAULT NULL,  
  `topic` varchar(110) DEFAULT NULL,  
  `user_` varchar(110) NOT NULL DEFAULT "",  
  `tweet` varchar(500) NOT NULL,  
  `pos` double DEFAULT NULL,  
  `neg` double DEFAULT NULL,  
  `result` varchar(20) DEFAULT NULL,  
  PRIMARY KEY (`user_`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
```

```
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
```

SOFTWARE TESTING

Software testing is one of the main stages of project development life cycle to provide our cessation utilizer with information about the quality of the application and ours, in our Project we have under gone some stages of testing like unit testing where it's done in development stage of the project when we are in implementation of the application after the Project is yare we have done manual testing with different Case of all the different modules in the application we have even done browser compatibility testing in different web browsers in market, even we have done Client side validation testing on our application

Unit testing

The unit testing is done in the stage of implementation of the project only the error are solved in development stage some of the error we come across in development are given below

Testing done when application is in development stage

Class version Error in our application

HTTP Status 500 - An exception occurred processing JSP page /addactivity.jsp at line 24

```
type Exception report
message An exception occurred processing JSP page /addactivity.jsp at line 24
description The server encountered an internal error that prevented it from fulfilling this request.
exception
org.apache.jasper.JasperException: An exception occurred processing JSP page /addactivity.jsp at line 24

21: <hr>
22: <%
23: int count=0;
24: Connection con1=databasecon.getConnection();
25: //System.out.println(con1);
26: Statement st = con1.createStatement();
27: ResultSet rs=st.executeQuery("select * from child_1");

Stacktrace:
org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:568)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:455)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)

root cause
javax.servlet.ServletException: java.lang.UnsupportedClassVersionError: databaseconnection/databasecon : Unsupported major.minor version 52.0 (unable to load class databaseconnection
org.apache.jasper.runtime.PageContextImpl.doHandlePageException(PageContextImpl.java:912)
org.apache.jasper.runtime.PageContextImpl.handlePageException(PageContextImpl.java:841)
org.apache.jsp.addactivity_jsp._jspService(addactivity_jsp.java:485)
org.apache.jasper.runtime.HttpJspBase.service(HttpJspBase.java:70)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:432)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
```

This Error Come when we move our application from one system to other and mainly when we version issues in the software's we us

Path related error in our application

HTTP Status 500 - An exception occurred processing JSP page /graph.jsp at line 27

type Exception report

message An exception occurred processing JSP page /graph.jsp at line 27

description The server encountered an internal error that prevented it from fulfilling this request.

exception

org.apache.jasper.JasperException: An exception occurred processing JSP page /graph.jsp at line 27

```
24: dataset.executeQuery( query);
25: JFreeChart chart = ChartFactory .createBarChart3D("Evaluation Graph", "", "",dataset, PlotOrientation.VERTICAL, true, true, true);
26:
27: ChartUtilities.saveChartAsJPEG(new File("E://Apache Tomcat/webapps/Web Revisitation//images//graph.jpg"), chart, 800, 400);
28: %>
29:
30: <center></center>
```

Stacktrace:

```
org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:568)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:460)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
```

root cause

```
java.io.FileNotFoundException: E:\Apache Tomcat\webapps\Web Revisitation\images\graph.jpg (The system cannot find the path specified)
java.io.FileOutputStream.open(Native Method)
java.io.FileOutputStream.<init>(Unknown Source)
java.io.FileOutputStream.<init>(Unknown Source)
org.jfree.chart.ChartUtilities.saveChartAsJPEG(ChartUtilities.java:506)
org.jfree.chart.ChartUtilities.saveChartAsJPEG(ChartUtilities.java:460)
org.apache.jsp.graph_jsp._jspService(graph_jsp.java:215)
org.apache.jasper.runtime.HttpJspBase.service(HttpJspBase.java:70)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:432)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
```

This Error Came when I have Performance Metrics to show in graph when I missed my server directly path in the system so we got this error in the applicant in development stage

Server Connection Error

< > 88 | localhost:2015/Web%20Revisitation/u_home.jsp

♡ ↴

HTTP Status 500 - An exception occurred processing JSP page /u_home.jsp at line 11

type Exception report

message An exception occurred processing JSP page /u_home.jsp at line 11

description The server encountered an internal error that prevented it from fulfilling this request.

exception

org.apache.jasper.JasperException: An exception occurred processing JSP page /u_home.jsp at line 11

```
8:
9: <font size="" color=""><h2>Welcome <%=session.getAttribute("name")%></h2></font>
10: <%
11: String ip=GetMyIPAddress.main();
12: String loc="";
13: Connection con1 = databasecon.getConnection();
14: Statement st1 = con1.createStatement();
```

Stacktrace:

```
org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:568)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:455)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
```

root cause

```
javax.servlet.ServletException: java.lang.UnsupportedClassVersionError: CT/GetMyIPAddress : Unsupported major.minor version 52.0 (unable to load class CT/GetMyIPAddress)
org.apache.jasper.runtime.PageContextImpl.doHandlePageException(PageContextImpl.java:912)
org.apache.jasper.runtime.PageContextImpl.handlePageException(PageContextImpl.java:841)
org.apache.jsp.u_005fhome_jsp._jspService(u_005fhome_jsp.java:364)
org.apache.jasper.runtime.HttpJspBase.service(HttpJspBase.java:70)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:432)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
```

root cause

Manual testing on project application

TEST CASES

Table 1 Login test case:

Test Case ID #1		Test Case Description - Validations in Login Form		
S#	Prerequisites	S#	Test Data Requirement	
1	Admin should have an email id	1	Data should be valid	
Test Condition				
Entering data in login form				
Step #	Step Details	Expected Results	Actual Results	Pass/Fail/Not Executed/Suspended
1	Admin gives a email or password of <6 characters	Admin logged in	Enter valid email/password	Fail
2	Submitting the form without entering any details	Admin logged in	Enter email /password	Fail
3	Admin enters wrong Email and (or) password	Admin logged in	Enter correct email /password	Fail

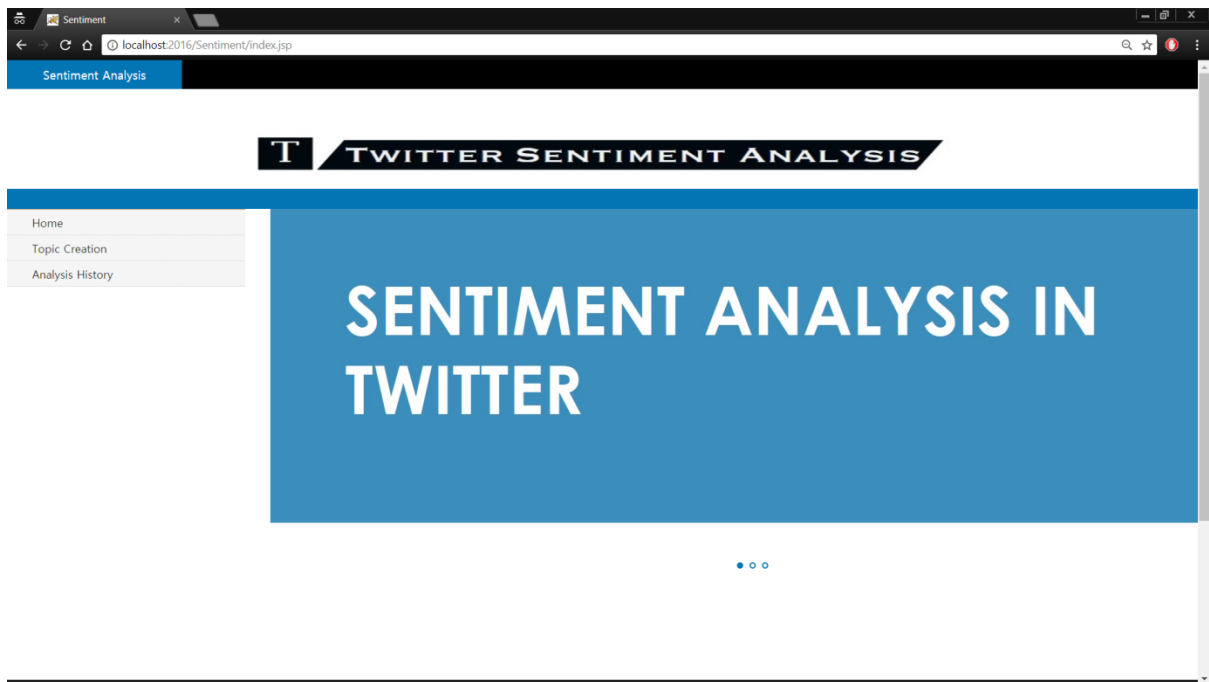
Compatibility testing

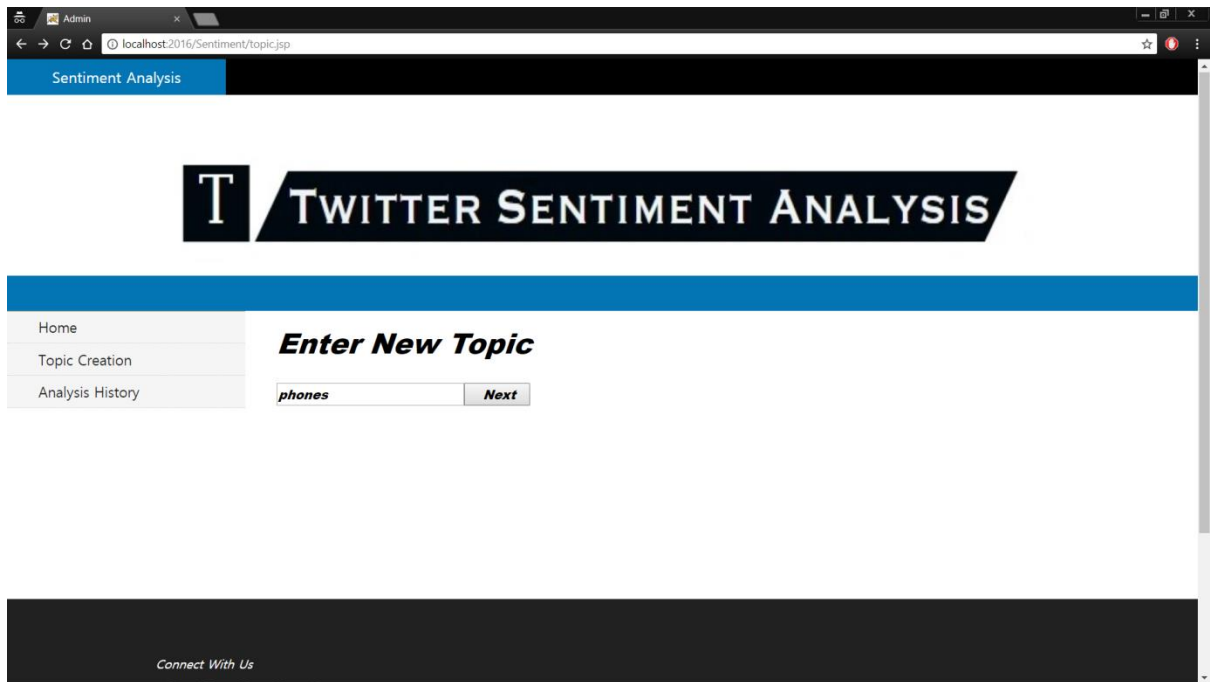


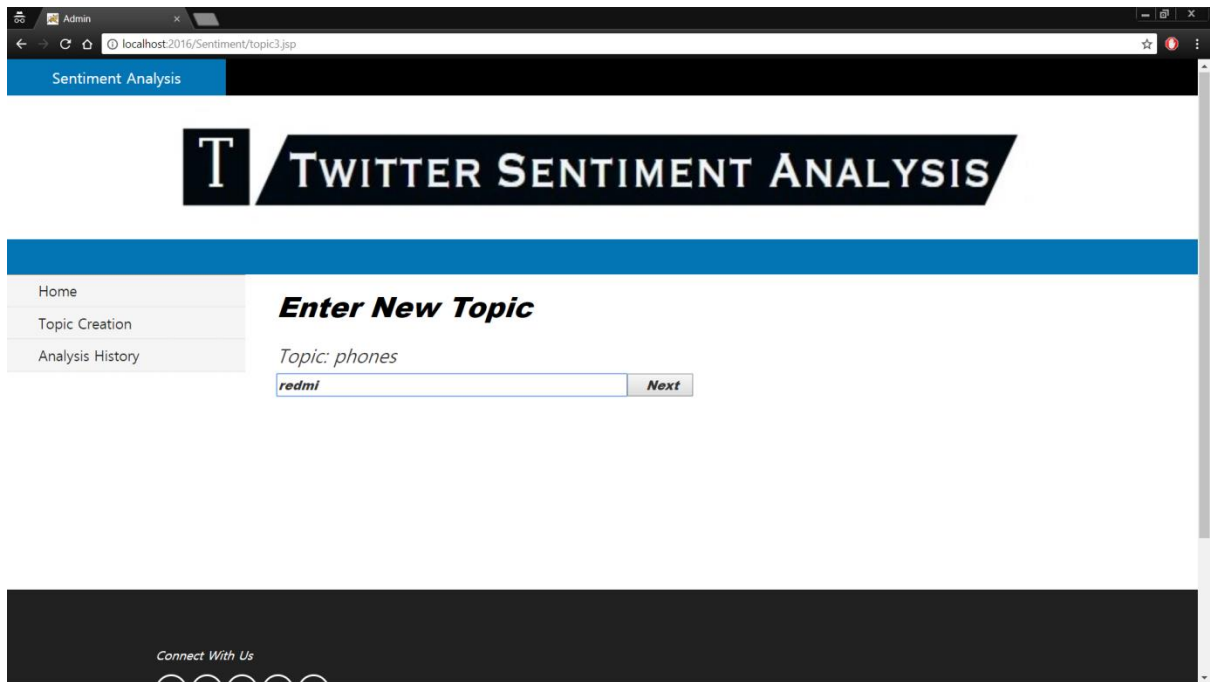
Compatibility testing is a type of software testing used to ensure compatibility of the system/application/website built with various other objects such as other web browsers, hardware platforms, users (in case if it's very specific type of requirement, such as a user who speaks and can read only a particular language), operating systems etc. This type of testing helps find out how well a system performs in a particular environment that includes hardware, network, operating system and other software etc.

Result of My Application on opera browser

SCREEN SHORTS







T TWITTER SENTIMENT ANALYSIS

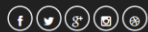
HomeTopic CreationAnalysis History

REDMI

1) Magger96 >> Help me win a Xiaomi Redmi 5 Plus Smartphone from Redskull <https://t.co/UuzaafnbPs>
2) reigunner90 >> RT @DennisCNLew: Redmi Note 5 PRO Bend Test-Glass Scratch (Durability Video) Kinked this time- Not as strong as RN5? <https://t.co/AHOCYRH7>
3) bestoffershops1 >> Redmi Note 5 (Gold, 64 GB) (4 GB RAM)#n4.4 728,809 Ratings & 4,358 Reviews#nMRP 11,999#nNo Cost EMIs from 500/month w? <https://t.co/HsmAN8foMW>
4) nitanshr >> RT @91mobiles: [Review] Xiaomi Redmi 5 review: a solid smartphone that's eclipsed by its own brethren <https://t.co/hLDg47gay5> <https://t.co/>
5) vvaibhav >> RT @techdroider: Using iPhone X-Like Swipe Gestures on Redmi Note 5 Pro: <https://t.co/elzpsydHvx> via @YouTube
6) shantanukapoor >> @RedmiIndia @Flipkart Fake sale.Making fool of customers.#nTrying from first sale but still not? <https://t.co/MZKZKCvHAJ>

User Duplicate Filter

Connect With Us



T TWITTER SENTIMENT ANALYSIS

Home

Topic Creation

Analysis History

REDMI

1) Magger96>> Help me win a Xiaomi Redmi 5 Plus Smartphone from Redskull <https://t.co/UuzaafnbPs>
2) reigunner90>> RT @DennisCNLew: Redmi Note 5 PRO Bend Test-Glass Scratch (Durability Video) Kinked this time- Not as strong as RN5?: <https://t.co/AHhOCCRH7>
3) bestoffershops1>> Redmi Note 5 (Gold, 64 GB) (4 GB RAM)/Wn4.4 128,809 Ratings & 4,358 Reviews/WnMRP 11,999/WnNo Cost EMIs from 500/month w? <https://t.co/HmAN8foMW>
4) nitanshr>> RT @91mobiles: [Review] Xiaomi Redmi 5 review: a solid smartphone that's eclipsed by its own brethren <https://t.co/hLDg47gy5> <https://t.co/?>
5) vvaibhav>> RT @techroider: Using iPhone X-Like Swipe Gestures on Redmi Note 5 Pro: <https://t.co/elzpSydHvx> via @YouTube
6) shantanukapoor>> @RedmiIndia @Flipkart Fake sale.Making fool of customers.WnTrying from first sale but still not? <https://t.co/MZKZKCvHAJ>

Sentiment Analysis

Connect With Us



© Made in VNBRVJET

Admin

localhost:2016/Sentiment/topic6.jsp

Sentiment Analysis

T

TWITTER SENTIMENT ANALYSIS

Home

Topic Creation

Analysis History

redmi


Sno	User	Tweet	
1)	Magger96	help[0.5,0.25] win[0.25,0.0] a[0.0,0.0] plus[0.625,0.0]	Positive Score= 1.375 Negative Score= 0.25 Result = Positive
2)	reigunner90	note[0.375,0.25] pro[0.0,0.0] bend[0.0,0.0] scratch[0.0,0.25] not[0.0,0.625] strong[0.5,0.875]	Positive Score= 0.875 Negative Score= 2.0 Result = Negative
3)	bestoffershops1	note[0.375,0.25] cost[0.0,0.125]	Positive Score= 0.375 Negative Score= 0.375 Result = Neutral
4)	nitanshr	review[0.125,0.0] a[0.0,0.0] solid[0.875,0.75] by[0.0,0.5] own[0.0,0.0] brethren[0.0,0.0]	Positive Score= 1.0 Negative Score= 1.25 Result = Negative
5)	vvaiibhav	swipe[0.0,0.125] on[0.125,0.0] note[0.375,0.25] pro[0.0,0.0]	Positive Score= 0.5 Negative Score= 0.375 Result = Positive

Analysis

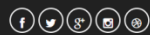
T TWITTER SENTIMENT ANALYSIS

Home
Topic Creation
Analysis History

Analysis Result

Topic	Sub Topic	Pos Score	Neg Score	Nue Score	Graph	Result
phones	redmi	55%	37%	7%	Graph	

Connect With Us



Admin

localhost:2016/Sentiment/viewanalysis.jsp

Sentiment Analysis

Home

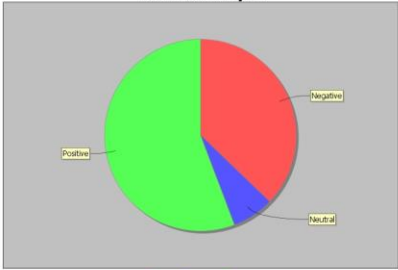
Topic Creation

Analysis History

localhost:2016/Sentiment/piegraph.jsp?pos=87&dneg=58&dnue=11 - Google Chrome

localhost:2016/Sentiment/piegraph.jsp?pos=87&dneg=58&dnue=11

Sentiment Analysis







Positive Negative Neutral

phones	redmi	55%	37%	7%	Click here for Pie Chart	
--------	-------	-----	-----	----	--------------------------	--

Pie Chart	Result	Delete
Click here		
Click here		
Click here		
Click here		
Click here		
Click here		

NOTE:
CLICK ON score percentages to view their respective WordCloud

localhost:2016/Sentiment/viewanalysis.jsp#

Sentiment Analysis							
TWITTER SENTIMENT ANALYSIS							
Analysis History							
Home	Topic	Sub Topic	Pos Score	Neg Score	Nue Score	Pie Chart	Result
Topic Creation							
Analysis History							
	1	oneplus	61%	19%	18%	Click here	
	2	#modi	28%	55%	17%	Click here	
	3	microsoft	54%	26%	19%	Click here	
	4	vnrvjiet	57%	34%	7%	Click here	

7. CONCLUSIONS

We learn sentiment-specific word embeddings (named assentiment embeddings) in this paper. Different from majority of exiting studies that only encode word contexts in word embeddings, we factor in sentiment of texts to facilitate the ability of word embeddings in capturing word similarities in terms of sentiment semantics. As a result, the words with similar contexts but opposite sentiment polarity labels like “good” and “bad” can be separated in the sentiment embedding space. We introduce several neural networks to effectively encode context and sentiment level informations simultaneously into word embeddings in a unified way. The effectiveness of sentiment embeddings are verified empirically on three sentiment analysis tasks. On word level sentiment analysis, we show that sentiment embeddings are useful for discovering similarities between sentiment words. On sentence level sentiment classification, sentiment embeddings are helpful in capturing discriminative features for predicting the sentiment of sentences. On lexical level task like building sentiment lexicon, sentiment embeddings are shown to be useful for measuring the similarities between words. Hybrid models that capture both context and sentiment information are the best performer on all three tasks.

8. REFERENCES

- [1] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, “Learning Sentiment-specific word embedding for twitter sentiment classification,” in Proc. 52th Annu. Meeting Assoc. Comput. Linguistics., 2014, pp. 1555–1565.
- [2] D. Tang, F. Wei, B. Qin, M. Zhou, and T. Liu, “Building large-scale twitter-specific sentiment lexicon: A representation learning approach,” in Proc. 25th Int. Conf. Comput. Linguistics, 2014, pp. 172–182.
- [3] D. Tang, F. Wei, B. Qin, T. Liu, and M. Zhou, “Coooooll: A deep learning system for twitter sentiment classification,” in Proc. 8th Int. Workshop Semantic Eval., 2014, pp. 208–212.
- [4] C. D. Manning and H. Schütze, Foundations of Statistical Natural Language Processing. Cambridge, MA, USA: MIT Press, 1999.
- [5] D. Jurafsky and H. James, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Englewood Cliffs, NJ, USA: Prentice-Hall, 2000.
- [6] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, “A neural probabilistic language model,” J. Mach. Learning Res., vol. 3, pp. 1137–1155, 2003.

- [7] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Proc. Conf. Neural Inf. Process. Syst., 2013, pp. 3111–3119.
- [8] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2014, pp. 1532–1543.
- [9] Z. S. Harris, “Distributional structure,” *Word*, vol. 10, pp. 146–162, 1954.
- [10] N. Yang, S. Liu, M. Li, M. Zhou, and N. Yu, “Word alignment modeling with context dependent deep neural network,” in Proc. 51st Annu. Meeting Assoc. Comput. Linguistics, 2013, pp. 166–175.