**SENTIMENT ANALYSIS OF TEXT DATA**

***A***

***Project Report***

*Submitted in partial fulfillment of the requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

***in***

**COMPUTER SCIENCE & ENGINEERING**

***by***

|  |  |  |  |
| --- | --- | --- | --- |
| Amresh Garg | Gaurav Porwal | Sudip Maji | Utkarsh Bahuguna |
| 500077157 | 500076356 | 5000676608 | 500075285 |
| Roll no: R110219009 | Roll no: R110219054 | Roll no: R110219140 | Roll no R110219146 |
| Branch: CSE-CCVT | Branch: CSE-CCVT | Branch: CSE-CCVT | Branch: CSE-CCVT |

**Under the guidance of**

**Mr. Amrendra Nath Tripathi**

**Assistant Professor**

**Dept. of Virtualization**

**School of Computer Science**



**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

### Bidholi, Via Prem Nagar, Dehradun, Uttarakhand 2021-2022

**CANDIDATE’S DECLARATION**

Whereby certify that the project work entitled **“*Opinion Analysis of Textual data*”** impartial fulfillment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in **Cloud Computing And Virtualization Technology** and submitted to the School of Computer Science, Department of Virtualization, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from **Jan 2022** to **May 2022** under the supervision of **Mr.Amrendra Nath Tripathi, Assistant Professor, Dept. of virtualization.**

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

|  |  |  |
| --- | --- | --- |
| **Specialization** | **Name** | **Roll Number** |
| CSE CCVT | Utkarsh Bahuguna | R110219146 |
| CSE CCVT | Gaurav Porwal | R110219054 |
| CSE CCVT | Sudip Maji | R110219140 |
| CSE CCVT | Amresh Garg | R110219009 |

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_2022 Mr.Amrendra Nath Tripathi

**Project Guide**

### 

**ABSTRACT**

Opinion Analysis is the computational study of sentiments/opinion expresses in an entity. There are many levels of opinion analysis like feature level, entity level, sentence level, document level. In our project, we will consider opinion analysis as sentence and document level.

This project will take input as a web comment and it will analyze and tell its polarity. This project will help us to choose the type of content we want to read or focus on. It will help in analyzing what type of impact that text is going to leave on the reader. This will help in detecting the emotion in the text.

The goal of our Opinion analysis system is to obtain an output value that shows how positive, negative, or neutral the sentence or document is.

Our project aims to divide or distribute the content according to their sentiments and helps the reader to choose the type of content they want to read.

**Keywords**— opinion analysis, polarity, modules, web comments

# **TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Contents** | **Page no.** |
| **1.** | Introduction | 6 |
| **2.** | Applications | 7 |
| **3.** | Literature review | 8-9 |
| **4.** | Objective | 10 |
| **5.** | Methodology | 11-12 |
| **6.** | Designing phase | 13-15 |
| **7.** | Proposed Algorithm | 16-17 |
| **8.** | Implementation | 18 |
| **9.** | Flow Charts | 19-27 |
| **10.** | Code | 28-44 |
| **11.** | Output Screens | 45-54 |
| **12.** | References | 55-56 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Name of Figure** | **Page no.** |
| **1.** | Waterfall model | 11 |
| **2.** | DFD Level 1 | 13 |
| **3.** | ER- Diagram | 14 |
| **4.** | Use Case Diagram | 15 |
| **5.** | Flow Chart | 16 |
| **6.** | Flow Chart 1 (MessageInterface.class) | 19 |
| **7.** | Flow Chart 2 (MessageHolderInterface. class) | 20 |
| **8.** | Flow Chart 3 (DictionaryInterface.class) | 21 |
| **9.** | Flow Chart 4(AnalysisInterface.class) | 21 |
| **10.** | Flow Chart 5(Message.class) | 22 |
| **11.** | Flow Chart 6(MessageHolder.class) | 23 |
| **12.** | Flow Chart 7(Dictionary.class) | 24 |
| **13.** | Flow Chart 8(Analysis.class) | 25 |
| **14.** | Flow Chart 9(Node.class) | 26 |
| **15.** | Flow Chart 10(MainController.class) | 27 |

# **INTRODUCTION**

Opinion Analysis is the extraction of sentiments from a source using basic language processing and computational analysis. It is the computer process of recognizing and classifying the viewpoint stated over a text to establish if that opinion or attitude expressed is favorable, negative, or neutral. Because views are essential to practically all human activities, sentiment analysis has a wide range of applications. Whenever we need to make a decision, we seek the advice of others. The goal of this project is to check the emotions and polarity in a sentence.

Because of the complex nature of the writings, we intend to perform sentiment analysis on them. With the advent of social media blogs and social media, sentiment analysis is becoming increasingly popular. All internet reviews, ratings, and other forms of feedback have become virtual currency for businesses. Before purchasing anything, people look at the reviews and the thoughts they provide.

We'll look at the user-generated material and categorize the responses as favorable, negative, or neutral. This project will assist users in filtering the information they choose to read or purchase. It will assist people in determining the emotional content of any given piece of content. It will determine the speaker's or writer's attitude toward the subject.

#### For Example, This product is amazing.

**This sentence expresses a positive opinion about the product as a whole.**

Like this example, our project first will determine the sentiment of the sentence and then It will calculate its polarity which will tell that the document is how much percent positive sentiment the document or sentence contains.

Polarity = Positive sentiment

Total Sentiments

# **APPLICATIONS**

The goal of our Opinion analysis system is to obtain an output value that shows that how much the sentence or document is positive, negative, or neutral on the localhost through SpringBoot and AWS.

Sentiment analysis is extremely useful in social network monitoring as it allows us to gain an overview of the wider public opinion behind certain topics. We analyze the user-generated text on the social network and classify the comments as being positive, negative, or neutral.

This project may help in:

**Social media monitoring:** As social is growing day by day, its horizons are becoming wider. So, social media, blogging platforms are the platform to share opinions. So, it is very important to recognize what type of opinion or emotions that person trying to convey or that content is even healthy or not. Social blogs we read have an impact on our minds that can be positive or negative.

**Customer support (Analyzing reviews):** We check others' reviews and opinions before taking any decision. So, it’s difficult to check what type of responses are they?

If we are purchasing something online there are types of responses that are difficult to judge. So this can help us by telling the polarity of reviews.

# **LITERATURE REVIEW**

The main research on sentiment analysis so far has mainly focused on two things:

* Identifying whether a given textual entity is subjective or objective.
* Identifying the polarity of that subjective textual entity after removing the objective content.

1. Pang, Lee, and Vaithyananthan (2002) focus on the principles and basic applications of sentiment analysis in this research work. It was the first paper to use Sentiment Classification Using Supervised Learning to divide movie reviews into positive and negative categories. Pang and Lee took the basic task of categorizing a movie review as good or negative and applied it to predicting star ratings on a 3- or 4-star scale. It was discovered that employing unigrams (a collection of words) as classification characteristics worked well using either naive Bayes or SVM. It was also the first paper in Scopus and Google Scholar to receive the most citations per year.
2. This research paper (N. Godbole, M. Srinivasaiah, and S. Skiena-"Large Scale Sentiment Analysis for News and Blogs," in ICWSM 07,2007.Godbole et al) developed techniques to algorithmically identify a large number (hundreds) of adjectives, each with an assigned polarity score, from a dozen or so seed adjectives. By recursively querying the synonyms and antonyms from WordNet, their algorithms expand two clusters of adjectives (positive and negative word groups). They took many precautions, such as providing weights that decrease exponentially as the number of hops grows, because recursive search quickly joins words from the two clusters. They confirm that the adjectives generated by the algorithm are extremely accurate when compared to the results of human-selected word lists.
3. The authors of this research paper (Arijit Chatterjee and Dr. William Perrizo-Investor Classification and Sentiment Analysis) discuss the impact of investors' bias on market volatility, sentiment analysis of potential investors' tweets, and why Microsoft Azure was chosen over other sentiment analysis tools. With over 280 million active users and about 500 million tweets sent every day, Twitter is one of the most popular social networking sites. Every day, some investors use Twitter to offer their thoughts on certain ticker symbols; this article examines how these individuals' views affect the stock market. Investors are said to be emotion-driven. To ensure that a stock is not overvalued or undervalued by investors, a top-down strategy is adopted.

The strategy is founded on two basic behaviors:

1. Sentiment analysis is one of them. Sentiment analysis is performed on tweets culled from the Twitter feeds of a few select investors. By recognizing "poor," "not good," and "wonderful" words in the tweets, they assign positive, negative, and neutral emotion scores to the ticker symbols. A user can utilize sentiment analysis to get the social sentiment score of a ticker symbol based on investor discussion and make an informed decision about which stock to invest in.
2. This research paper (Xing Fang and Justin Zhan-Sentiment Analysis Using Product Review Data) attempts to address the most fundamental problem in sentiment analysis, polarity categorization, by using a dataset containing over 5.1 million product reviews from Amazon.com for products in four categories: beauty, books, electronics, and home. Previous research in this area advised that to undertake sentiment analysis, all objective content be removed, however here the subjective content is extracted instead for future analysis. Reviews are used as inputs, and they include customer information, a review, helpfulness, and a rating. For a more accurate interpretation of the review's emotion, the rating is used as the ground truth. To classify the words of the poem, a maximum entropy POS tagger is utilized to classify the words of the sentence into 46 tags. An additional python program is particularly used to speed up this process. As a result, a total of 25 million adjectives,22 million adverbs, and 56 million verbs are identified, which usually tend to determine the sentiment. The algorithm also makes a list of phrases based on occurrence. The following are the various classification models which are selected for categorization-Naïve Baysian, Random Forest, and support VectorMachine.

# **OBJECTIVES**

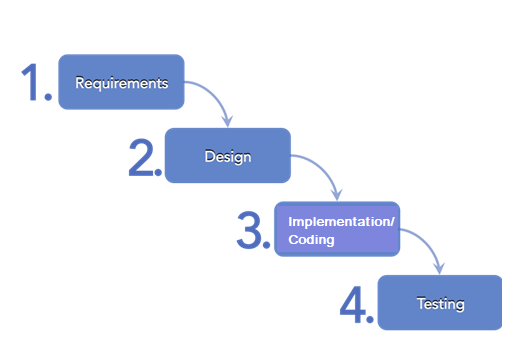
Its main objective is to determine the emotional tone behind a series of words, used to gain an understanding of the attitudes, opinions, and emotions expressed within an online mention.

* + It also helps in understanding public opinion, companies use sentiment analysis in doing market research and figuring out if their customers like a particular product or not.
  + It can lead to more accurate tools for extracting semantic information and provides means for empirically studying the properties of social interactions.

**Sub objectives:**

* To evaluate the person's opinion in certain cases
* To determine the emotional tone behind the series of words.
* To teach the machine to analyze the various grammatical nuances
* To implement an algorithm for automatic classification of text into positive or negative

**METHODOLOGY**

**

*Fig. 1: Waterfall Model*

According to Waterfall Model:

1. **Requirements Analysis:**

For this project, we have searched various research papers to find the drawbacks of the current system so that we can update it.

**Project requirements:**

#### Hardware Requirements-

* An average computer system with basic specifications:
* 2/4 GB RAM.
* Intel Dual Core or Higher Processor.

#### Software Requirements-

* Spring Boot 1.5.10. RELEASE requires Java 7
* Spring Framework 4.3.14. release or above
* PuTTY
* Eclipse

1. **Design:**

This phase includes designing of DFD, Use case diagram, ER Diagram, Flow charts

1. **Implementation:**

This phase will show the algorithm to use, Implementation of code.

1. **Testing:**

This phase includes the test cases by which we can check our project is correctly telling the sentiments or not. In this phase, we will check our code using test cases.

Example:

(1) I’ve just registered for the competition.

(2) This movie is amazing.

(3) The reports are very good.

(4) Their blog is so informative.

(5)It is very expensive.

• Sentence 1 is neutral, as it doesn’t offer any sentiment.

Output : 0

• Sentence 2 expresses a positive opinion about the movie as a whole.

Output: 1

• Sentence 3 expresses a positive opinion about the movie’s reports.

Output: 1

• Sentence 4 expresses a positive opinion about the company’s blog.

Output: 1

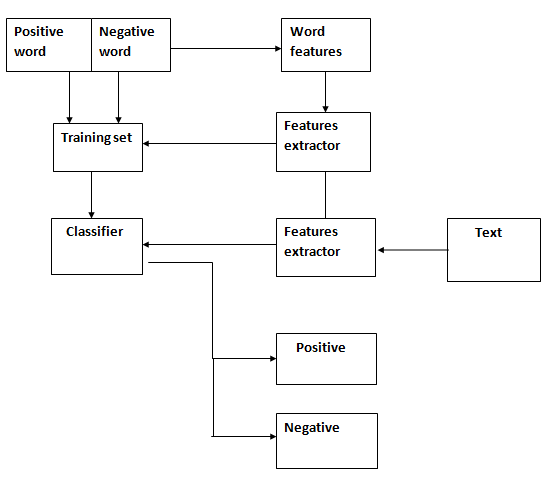
• Sentence 5 expresses a negative opinion about a product’s pricing.

Output: -1

**DESIGNING PHASE**

**Data Flow Diagram**

**DFD Level 1**

****

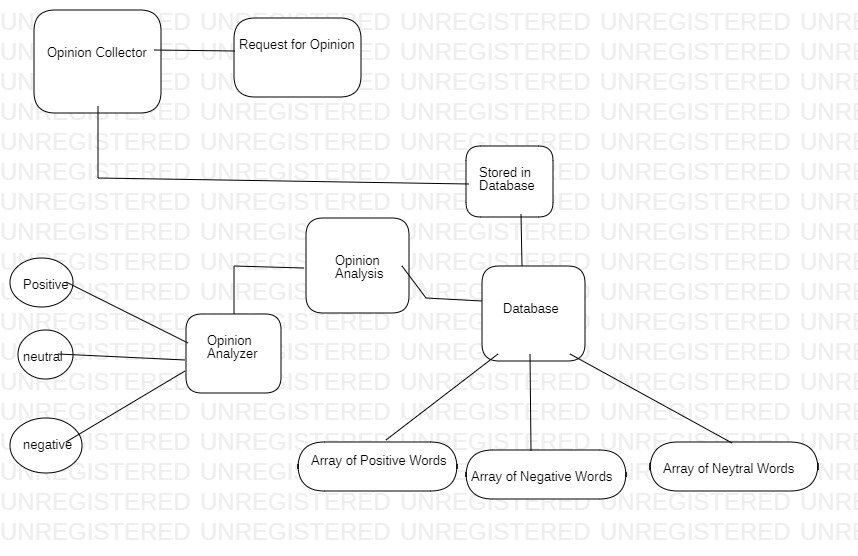
*Fig. 2: DFD Level 1*

*Figure 2* Describes the data flow of our project using level 1 DFD. This shows more explanation of how to analyze text (documents, online reviews, social media posts, etc.) and determine the sentiment of opinions on a scale from negative to positive.

**Entities :**

* Word features
* Training set
* Features extractor
* Classifier

**ER DIAGRAM**

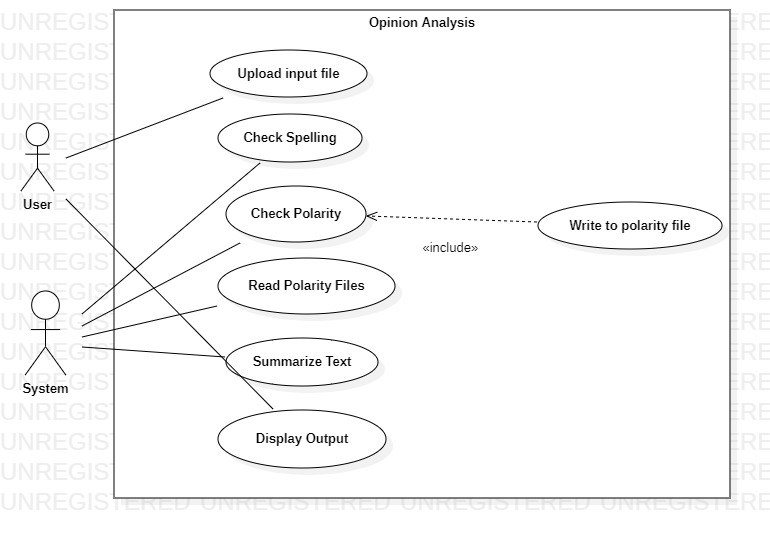
****

*Fig. 3: ER Diagram*

The figure shows 5 entities (opinion collector, request for opinion, database, opinion analysis, opinion analyzer). All the entities are connected by the straight line according to their relationship with each other.

The degree of their relationships could be one to one, one to many, many to many, and many to one.

**USE CASE DIAGRAM:**

****

*Fig.4.5: USE Case Diagram*

The above fig. 4 describes the use cases of our project. This includes two actors acting upon six modules:

**User**: The user has to upload the input File and check the output opinion and polarity in the end.

**Admin**: The system will check the spellings in the input file then check its polarity by reading from the polarity files. then summarize the polarity and compile the output.

**Proposed Algorithm:**

The Proposed algorithm contains the following features:

• Function to extract comments.

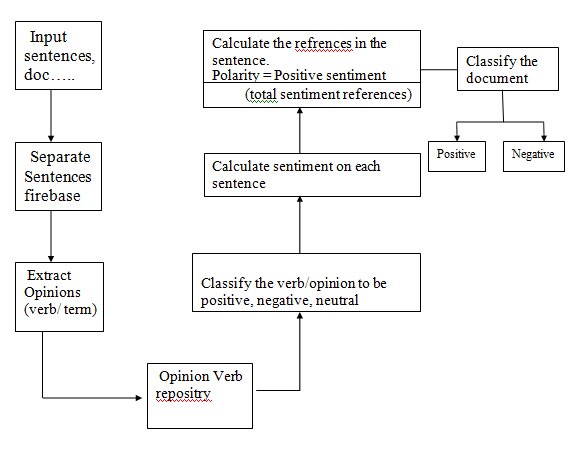
• Function to divide and categorize the text into positive or negative.

• Function that matches words (from comments) to the built list of words.

• Function to calculate the polarity of statements.

1. **Architecture:**

The system architecture is as shown below:



1. **Modules:**

This algorithm can be divided into 5 modules:

1. Creation of vocabulary and data sets:

* Build two separate lists namely
* Positive/Good words. Ex: good, awesome, useful, pretty
* Negative/Bad words. Ex: rude, dirty, worst, horrible

1. Input: Extract comments from web pages.
2. Matching:

* After extracting the context

-divide it into a sentence

-perform word-by-word analysis of each sentence.

-categorize each word to be positive, negative, or neutral.

1. Calculate Polarity:

• Polarity indicates the percentage of positive sentiment references among total sentiment references. **Polarity = (positive\_sentiment)**

**(total\_sentiment\_references)**

1. Output:

Display the calculated polarity to the end-user

. Ex: Total words, Positive and Negative words, Polarity.

**IMPLEMENTATION PHASE**

**Pseudo Code:**

Step1: Define the files containing the words with their sentiment values.

Step2: Extract/Input the comment from user i.e. text[ ]

Step3: Divide each sentence in the text to form an array of words i.e. input[ ]

Step4: For each word in the string input[ ], check whether it is present in the list of goodWords[ ] and badWords[ ] arrays.

1. If the word is found in goodWords[ ] list, increment the count of positive\_sentiment and also the total\_sentiment\_references
2. If the word is found in badWords[ ] list, increment the count of negative\_sentiment and also the total\_sentiment\_references

Step5: Calculate the general sentiment count

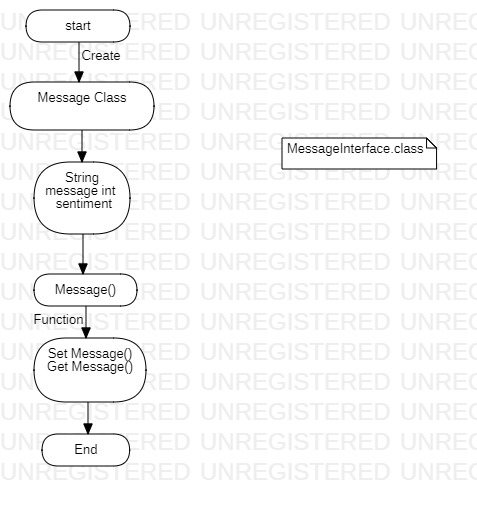
Step6: Display the calculated general sentiment count and the general sentiment of the text.

**Algorithm:**

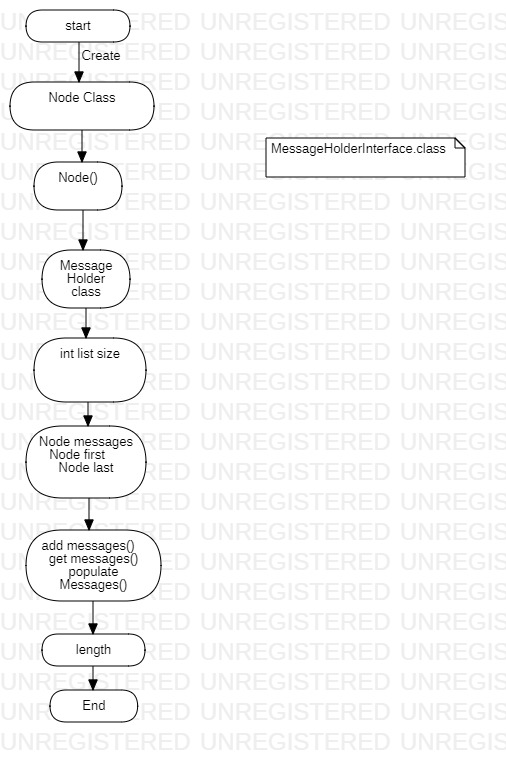
1. Start
2. Import Message file.
3. Message files take input from the user
4. Import MessageHolder
5. MessageHolder file stores the messages till the user presses SUBMIT
6. Calls PopulateMessage() function
7. Messages populated.
8. An import Dictionary file.
9. It checks sentiments.
10. Add the value of sentiments.
11. Open AF1NN file.
12. Search for input words and set sentiment values accordingly.
13. Import Analysis class
14. Check values of sentiments.
15. Differentiate positive, negative, and neutral sentiments.
16. Calculate overall sentiment. percentage of positive, negative, and neutral.
17. Delete the message holder.
18. Show output general sentiment on the output screen.
19. End

**Flow chart**

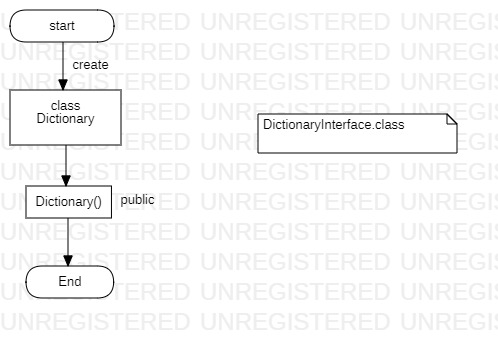
1. **MessageInterface.class**

****

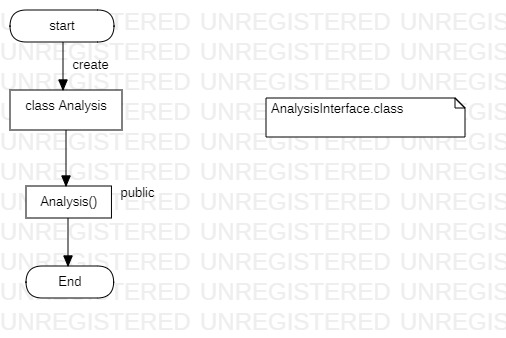
1. **MessageHolderInterface.class**

****

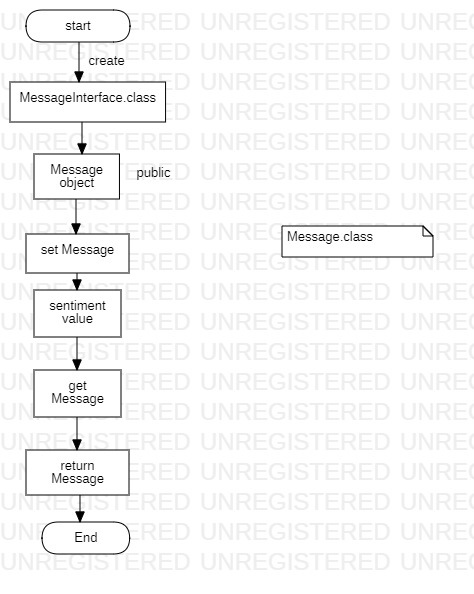
1. **DictionaryInterface.class**

****

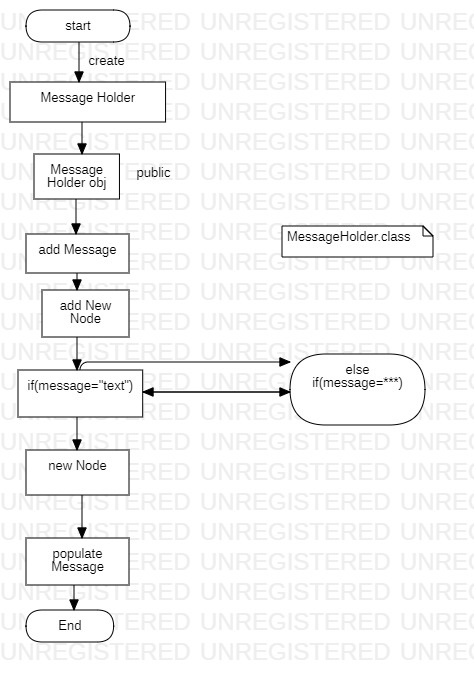
1. **AnalysisInterface.class**

****

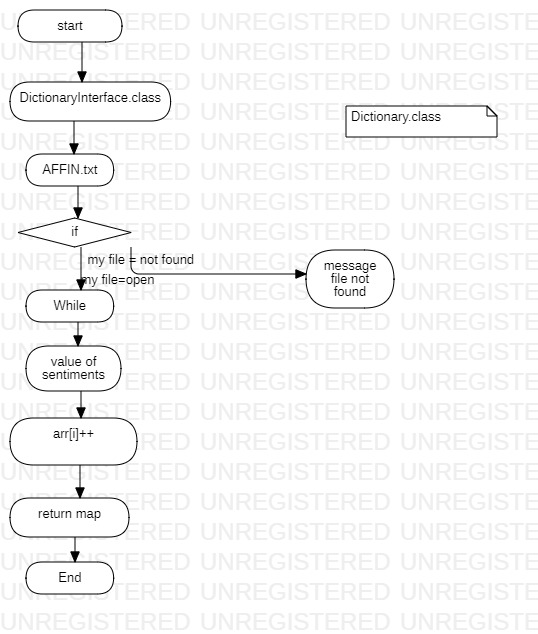
1. **Message.class**

****

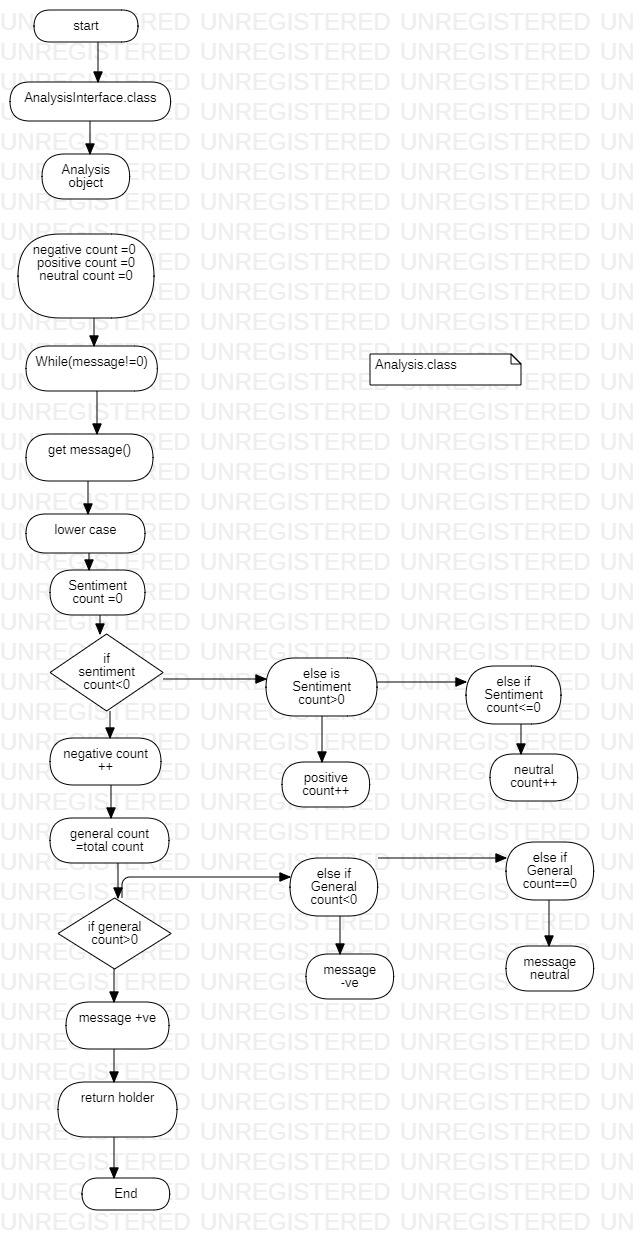
1. **MessageHolder.class**

****

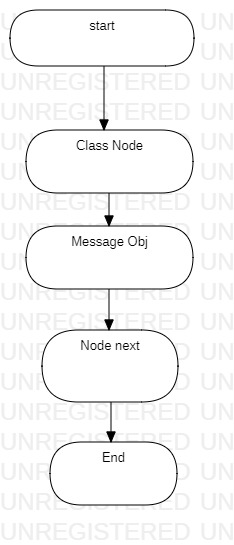
1. **Dictionary.class**

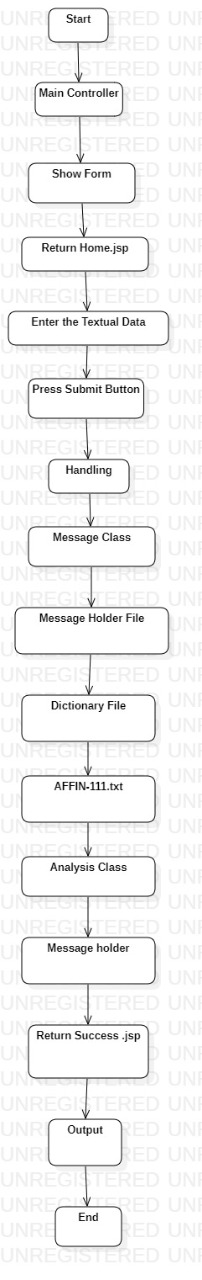
****

1. **Analysis.class**

****

1. **Node.class**



**10. MainController.class**

**Code:**

1. **MessageInterface.java**

package OpinionAnalysis;

public interface MessageInterface {

public void setMessage(String message);

public void setSentiment(int sentiment);

public String getMessage();

public int getSentiment();

}

1. **MessageHolderInterface.java**

package OpinionAnalysis;

public interface MessageHolderInterface {

public void addMessage(Message message);

public String populateMessages(String string);

public Node getAllMessages();

public int size();

}

1. **DictionaryInterface.java**

package OpinionAnalysis;

import java.util.Map;

public interface DictionaryInterface {

Map<String, Integer> getDictionary(String fileLocation);

}

1. **AnalysisInterface.java**

package OpinionAnalysis;

import java.util.Map;

import org.springframework.ui.Model;

public interface AnalysisInterface {

MessageHolder analyseMessages(MessageHolder holder, Map<String, Integer> map, Model model);

}

1. **Node.java**

package OpinionAnalysis;

class Node{

public Message message;

public Node next;

public Node(){}

}

1. **Message.java**

package OpinionAnalysis;

class Message implements MessageInterface{

private String message;

private int sentiment;

public Message(String message){

this.sentiment = 2;

this.message = message;

}

public void setMessage(String message) {

this.message = message;

}

public void setSentiment(int sentiment) {

this.sentiment = sentiment;

}

public String getMessage(){

return this.message;

}

public int getSentiment(){

return this.sentiment;

}

}

1. **MessageHolder.java**

package OpinionAnalysis;

class MessageHolder implements MessageHolderInterface{

private Node messages;

private Node first;

private Node last;

private int listSize;

public MessageHolder(){

this.first = null;

this.last = null;

this.messages = null;

this.listSize = 0;

}

public void addMessage(Message message) {

// System.out.println("Add message functionality is working");

if(messages == null) {

messages = new Node();

messages.message = message;

messages.next = null;

first = messages;

last = messages;

}

else if(first==last){

first.next = new Node();

last = first.next;

last.message = message;

last.next = null;

}

else {

last.next = new Node();

last = last.next;

last.message = message;

last.next = null;

}

listSize++;

}

public String populateMessages(String userMessage) {

String []userinput = userMessage.split("\n");

for(String i:userinput) {

System.out.println(i);

Message m = new Message(i);

addMessage(m);

}

return "success";

}

public Node getAllMessages() {

// System.out.println("getAllMessages is working");

return this.messages;

}

public int size() {

// System.out.println("size is working");

return listSize;

}

}

1. **Dictionary.java**

package OpinionAnalysis;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.util.HashMap;

import java.util.Map;

class Dictionary {

public Dictionary(){

}

public Map<String, Integer> getDictionary(String fileLocation) throws Exception{

//System.out.println("get dictionary in action!");

final String filePath = fileLocation;

Map<String, Integer> map = new HashMap<String, Integer>();

BufferedReader br = null;

try {

// create file object

File file = new File(filePath);

br = new BufferedReader(new FileReader(file));

String line = null;

// read file line by line

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

// split the line by :

String[] parts = line.split("\\s+");

// first part is name, second is number

String name = parts[0].trim();

String number1 = parts[1].trim();

Integer number = Integer.parseInt(number1);

// put name, number in HashMap if they are not empty

if (!name.equals("") && !number.equals(""))

map.put(name, number);

}

}

catch (Exception e) {

e.printStackTrace();

}

finally {

if (br != null) {

try {

br.close();

} catch (Exception e) {

}

;

}

}

return map;

}

}

1. **Analysis.java**

package OpinionAnalysis;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Iterator;

import java.util.List;

import java.util.Map;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

@Controller

class Analysis implements AnalysisInterface{

public Analysis() {}

public MessageHolder analyseMessages(MessageHolder holder, Map<String, Integer> map, Model model){

Node messages = holder.getAllMessages();

double negativeCount = 0;

double positiveCount = 0;

double neutralCount = 0;

double generalCount = 0;

while(messages != null)

{

String message = messages.message.getMessage().toLowerCase();

int sentimentCount= 0;

String[] elements = message.split("\\s+");

List<String> fixedLenghtList = Arrays.asList(elements);

List<String> wordList = new ArrayList<String>(fixedLenghtList);

Iterator<String> it = wordList.iterator();

while(it.hasNext()) {

String cum=it.next();

if(cum != null) {

if (map.get(cum) != null)

sentimentCount += map.get(cum).intValue();

}

else {

sentimentCount += 0;

}

}

if (sentimentCount < 0){

messages.message.setSentiment(0);

negativeCount++;

}

else if (sentimentCount > 0){

messages.message.setSentiment(1);

positiveCount++;

}

else if (sentimentCount == 0){

messages.message.setSentiment(2);

neutralCount++;

}

generalCount += sentimentCount;

messages= messages.next;

}

System.out.println("Opinion Analysis is as follows: " );

double total = (negativeCount + positiveCount + neutralCount);

model.addAttribute("Total", total);

System.out.println("Total Messages: " + total);

double positivePercentageOriginal = (positiveCount / total)\*100;

double positivePercentage = Math.round(positivePercentageOriginal \* 100.0) / 100.0;

double negativePercentageOriginal = (negativeCount / total)\*100;

double negativePercentage = Math.round(negativePercentageOriginal \* 100.0) / 100.0;

double neutralPercentageOriginal = (neutralCount / total)\*100;

double neutralPercentage = Math.round(neutralPercentageOriginal \* 100.0) / 100.0;

model.addAttribute("Positive\_Percentage", positivePercentage);

model.addAttribute("Negative\_Percentage", negativePercentage);

model.addAttribute("Neutral\_Percentage", neutralPercentage);

System.out.println("Positive %: " + ((positiveCount / total)\*100));

System.out.println("Negative %: " + ((negativeCount / total)\*100));

System.out.println("Neutral %: " + ((neutralCount / total)\*100));

String consensus = "";

if (generalCount>0)

consensus ="General Consensus is Positive!";

else if (generalCount<0)

consensus ="General Consensus is Negative!";

else if (generalCount==0)

consensus ="General Consensus is Neutral!";

model.addAttribute("General\_Consensus", consensus);

return holder;

}

}

1. **MainController.java**

package OpinionAnalysis;

import java.util.Map;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

@Controller

public class MainController {

@RequestMapping("/")

public String showForm() {

return "home";

}

@RequestMapping(path="/processform", method = RequestMethod.POST)

public String handling(@RequestParam("user\_Message") String userMessage, Model model) throws Exception {

MessageHolder holder = new MessageHolder();

holder.populateMessages(userMessage);

Dictionary dictionary = new Dictionary();

Map<String,Integer> map = dictionary.getDictionary("C:\\AFFIN-111.txt");

// Map<String,Integer> map = dictionary.getDictionary("/root/AFFIN-111.txt");

Analysis analysis = new Analysis();

holder = analysis.analyseMessages(holder,map,model);

return "success";

}

}

1. **Home.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!doctype html>

<html lang="en">

<head>

<!-- Required meta tags -->

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<!-- Bootstrap CSS -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<title>Sentiment Analysis</title>

<style>

body{

background:#ffc2c2;

}

</style>

</head>

<body>

<h1 class="text-center mt-5" style="color:#f7450c;font-weight: 900;" >Welcome To Sentiment Analysis!</h1>

<div class="container mt-5">

<form action="processform" method="post">

<div class="mb-3">

<label for="exampleInputData1" style="color: #7e4800;font-weight: 900;font-size: x-large;" class="form-label ">Enter The Text to be Analyzed</label>

<textarea class="form-control" style="background:#ffd3d3;color:#162de4" name="user\_Message" id="exampleInputData1" rows="10"></textarea>

</div>

<div class="container text-center">

<button type="submit" class="btn btn-success">Submit</button>

</div>

</form>

</div>

<!-- Option 1: Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

</body>

</html>

12.**success.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@page isELIgnored="false" %>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Analysis Results</title>

<style>

\*

{

margin:0;

padding:0;

font-family:'Roboto',sans-serif;

}

body

{

background:#ffc2c2;

/\* background:#e8f5ca; \*/

align-items:center;

justify-content:center;

min-height:100vh;

display:flex;

}

.boxMain {

display: flex;

justify-content: space-between;

}

.box

{

/\* background:#fff; \*/

background:#f20404;

width:300px;

height:400px;

display:flex;

justify-content:center;

align-items:center;

position:relative;

flex-direction:column;

box-shadow:0 30px 60px rgba(0,0,0,.4);

transition: transform .2s;

}

.box .percent

{

width:150px;

height:150px;

position:relative;

}

.box .percent svg

{

width:150px;

height:150px;

position:relative;

}

.box .percent svg circle

{

width:150px;

height:150px;

fill:none;

stroke-width:10;

stroke:#000;

transform:translate(5px,5px);

stroke-dasharray:440;

stroke-dashoffset:440;

stroke-linecap:round;

}

.box .percent svg circle:nth-child(1)

{

stroke-dashoffset:0;

stroke:#f3f3f3;

}

/\* .box .percent svg circle:nth-child(2)

{

stroke-dashoffset:calc(440 - (440 \* ${Positive\_Percentage }) / 100);

/\* stroke-dashoffset:calc(440 - (440 \* 12) / 100); \*/

stroke:#ff6600;

}

\*/

.custom svg circle:nth-child(2)

{

/\* stroke-dashoffset:calc(440 - (440 \* ${Positive\_Percentage }) / 100); \*/

/\* stroke-dashoffset:calc(440 - (440 \* 12) / 100); \*/

/\* stroke:#ff6600; \*/

}

.custom0 svg circle:nth-child(2)

{

stroke-dashoffset:calc(440 - (440 \* ${Positive\_Percentage }) / 100);

/\* stroke-dashoffset:calc(440 - (440 \* 12) / 100); \*/

stroke:#ff6600;

}

.custom1 svg circle:nth-child(2)

{

stroke-dashoffset:calc(440 - (440 \* ${Negative\_Percentage }) / 100);

/\* stroke-dashoffset:calc(440 - (440 \* 12) / 100); \*/

stroke:#ff6600;

}

.custom2 svg circle:nth-child(2)

{

stroke-dashoffset:calc(440 - (440 \* ${Neutral\_Percentage }) / 100);

/\* stroke-dashoffset:calc(440 - (440 \* 12) / 100); \*/

stroke:#ff6600;

}

.box .percent .num

{

top:0;

left:0;

width:100%;

height:100%;

display:flex;

justify-content:center;

align-items:center;

position:absolute;

color:#111;

}

.box .percent .num h2

{

font-size:40px;

}

.box .percent .num h2 span

{

font-size:24px;

}

.box .text

{

padding 10px 0 0;

color:#999;

font-weight:700;

letter-spacing:1px;

}

.centering{

text-align: center;

}

</style>

</head>

<body>

<div>

<div class="centering">

<h1 style="color:#0913eb;">

Sentiment Analysis Results <br> <br>

</h1>

<h2 style="color:#0d32a2;">Total Messages: ${Total } </h2> <br>

</div>

<div class="boxMain">

<div class="box">

<div class="percent custom0">

<svg>

<circle cx="70" cy="70" r="70"></circle>

<circle cx="70" cy="70" r="70"></circle>

</svg>

<div class="num">

<h2> ${Positive\_Percentage }<span>%</span></h2>

</div>

</div>

<h2 class="text">Positive</h2>

</div>

<div class="box">

<div class="percent custom1">

<svg>

<circle cx="70" cy="70" r="70"></circle>

<circle cx="70" cy="70" r="70"></circle>

</svg>

<div class="num">

<h2> ${Negative\_Percentage }<span>%</span></h2>

</div>

</div>

<h2 class="text">Negative</h2>

</div>

<div class="box">

<div class="percent custom2">

<svg>

<circle cx="70" cy="70" r="70"></circle>

<circle cx="70" cy="70" r="70"></circle>

</svg>

<div class="num">

<h2> ${Neutral\_Percentage }<span>%</span></h2>

</div>

</div>

<h2 class="text">Neutral</h2>

</div>

</div>

<div class="centering">

<h2 style="color:green;">

<br>${General\_Consensus }

</h2>

</div>

</div>

</h2>

</body>

</html>

1. OpinionAnalysisApplication.Java

package OpinionAnalysis;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class OpinionAnalysisApplication {

public static void main(String[] args) {

SpringApplication.run(OpinionAnalysisApplication.class, args);

}

}

**Output Screens**

**Words sentiment check:**

**Sentences sentiment check:**

**Paragraph sentiment check:**

**Online Shopping reviews sentiment check:**

**Blogs review sentiment check:**

**Movie reviews sentiment check:**

**Book review sentiment check:**

**App reviews sentiment check:**

**Social media comments sentiment check:**

**Article Reveiws sentiment check:**

**References**

* [**Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. EMNLP Proceedings.**](https://aclanthology.org/W02-1011.pdf)

* [**Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. ACL Proceedings.**](https://aclanthology.org/P04-1035.pdf)

* [**Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. ACL Proceedings.**](https://aclanthology.org/P05-1015.pdf)
* [**P. Turney, “Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews,” Proceedings of the Association for Computational Linguistics (ACL), pp. 417–424, 2002**](https://arxiv.org/abs/cs/0212032)

* [**https://github.com/amreshgarg21/Opinion-Analysis-Web-Using-Spring-MVC-Boot**](https://github.com/amreshgarg21/Opinion-Analysis-Web-Using-Spring-MVC-Boot)

1. **AFINN-111.txt**

abandon -2

abandoned -2

abandons -2

abducted -2

abduction -2

abductions -2

abhor -3

abhorred -3

abhorrent -3

abhors -3

abilities 2

ability 2

aboard 1

absentee -1

absentees -1

absolve 2

absolved 2

absolves 2

absolving 2

absorbed 1

abuse -3

abused -3

abuses -3

abusive -3

accept 1

accepted 1

accepting 1

accepts 1

accident -2

accidental -2

**----------------**