# PRINCIPLES OF BIG DATA MANAGEMENT

# PROJECT – III

# SPRING 2017

Submitted by: TEAM 13

Sri Sai Anusha Gandu (sgr43)

Sushma Mitta (smgp6)

Sadanand Kallakuri (sk789)

Abhilash Reddy Gaddam (aggg6)

# CONTENTS


| Title | Pg. No. |
|---|---|
| 1. Introduction | 1 |
| 1.1 About Twitter | 1 |
| 1.2 About the Project | 1 |
| 2. Requirements | 1 |
| 2.1 Languages | 1 |
| 2.2 Software | 1 |
| 3. Tasks | 2 |
| 4. Query 1 (Using RDD) | 2 |
| 5. Query 2 | 4 |
| 6. Query 3 (Using Trends File) | 5 |
| 7. Extra Requirement | 6 |
| 7.1 Pie Chart Representation of Query 1 | 7 |
| 7.2 Bar Graph Representation of Query 2 | 8 |
| 7.3 Line Chart Representation of Query 3 | 9 |
| References | 10 |

# 1. INTRODUCTION

## 1.1 About Twitter

Twitter is an online news and social networking service where users post and interact with messages, "tweets," restricted to 140 characters. Registered users can post tweets, but those who are unregistered can only read them. Twitter Inc. is based in San Francisco, California, United States, and has more than 25 offices around the world. Twitter was created in March 2006 by Jack Dorsey, Noah Glass, Biz Stone, and Evan Williams and launched in July, whereby the service rapidly gained worldwide popularity. As of 2016, Twitter had more than 319 million monthly active users.

## 1.2 About the Project

Here, we have collected the tweets using twitter API through tweepy using the keywords Python, JavaScript and Ruby in JSON (JavaScript Object Notation) format. The tweets then collected have been analyzed and different SQL queries are written to obtain the result.

# 2. REQUIREMENTS

## 2.1 Languages

1. Scala
2. SQL
3. Java
4. Javascript
5. HTML

## 2.2 Software

1. IntelliJ IDEA 3.4 (IDE)
2. JDK 1.8
3. Scala 2.12.1
4. Spark 2.1
5. Virtual Box (Cloudera)
6. WebStorm 2016.2.4

## 3. TASKS

- Implement below three analytical tasks using a collection of tweets.

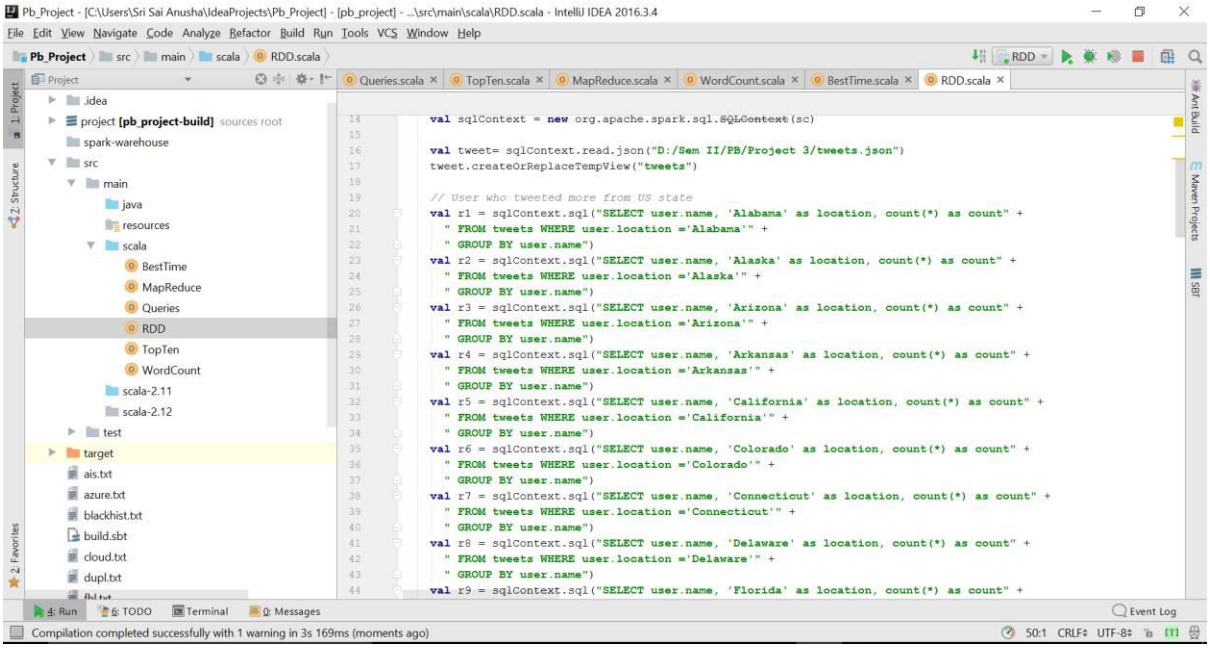
1. One using RDD transformations and actions.
2. The other two using Spark SQL and DFs.
3. One of the query including the input file trends.txt

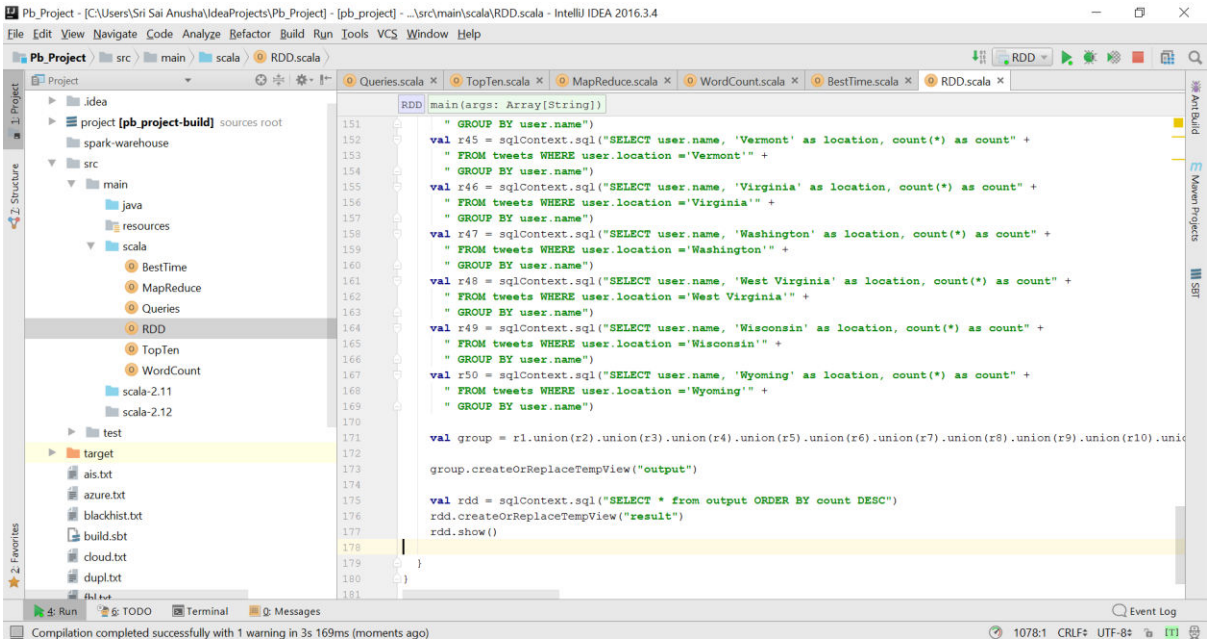
### EXTRA REQUIREMENT

- To provide the visual implementation of the queries.

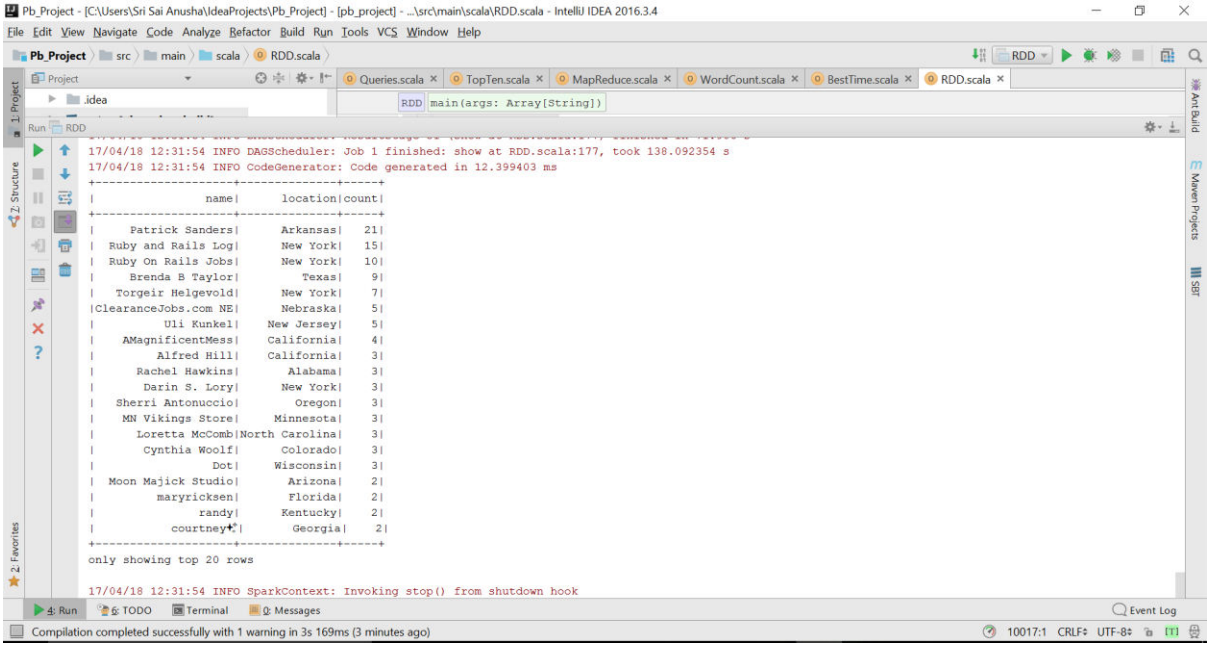
## 4. QUERY 1 (USING RDD)

The following RDD query extracts the users who have tweeted the most from US states.






The output thus obtained from the query is shown below.



# 5. QUERY 2

This query extracts the tweet count obtained from different locations.

```scala
import org.apache.spark.{SparkConf, SparkContext}


/**...*/

object Spark {
  def main(args: Array[String]): Unit = {
    System.setProperty("hadoop.home.dir", "C:\\Users\\Sri Sai Anusha\\Desktop\\Winutils")
    val conf = new SparkConf().setAppName("Queries").setMaster("local[*]")
    val sc = new SparkContext(conf)
    val sqlContext = new org.apache.spark.sql.SQLContext(sc)

    //Tweets obtained from different locations
    val text = sc.textFile("D:/Sem II/PB/Project 3/tweets.txt")
    val tweet = sqlContext.read.json("D:/Sem II/PB/Project 3/tweets.json")
    tweet.registerTempTable("table")
    tweet.createOrReplaceTempView("table")

    val que = sqlContext.sql("SELECT user.location, count(*) as count FROM table " +
      "GROUP BY user.location ORDER BY count DESC")
    que.show(25)

    //que.write.option("collection", "Spark").mode("overwrite").format("com.mongodb.spark.sql").save()
  }
}
```

The following is the result obtained by the query.

```
17/05/01 18:33:51 INFO CodeGenerator: Code generated in 10.736206 ms
+-----------------+-----+
|         location|count|
+-----------------+-----+
|             null|19955|
|               UK| 1024|
|    United States|  993|
| Bengaluru, India|  793|
|       Yogyakarta|  614|
|    Paris, France|  511|
| London, England|  369|
|              USA|  368|
|           London|  287|
|     Your Console|  277|
|     The Internet|  277|
|            India|  237|
|San Francisco, CA|  203|
|      Everywhere |  203|
|           France|  199|
|    The interwebs|  198|
|              CN|  194|
|      Somerset UK|  189|
|              日本|  187|
|       Texas, USA|  150|
|        Worldwide|  149|
|           Tokyo|  140|
| Los Angeles, CA|  139|
|           india|  138|
|         Internet|  134|
+-----------------+-----+
only showing top 25 rows
```

# 6. QUERY 3 (USING TRENDS FILE)

This query searches through the names of the trends file and gives the output as the text and hashtags that are associated with the common names.

```
import org.apache.spark.sql.functions._
import org.apache.spark.{SparkConf, SparkContext}

/**...*/

//Top 10 users based on tweet_volume
object DataFrame {
  def main(args: Array[String]): Unit = {
    System.setProperty("hadoop.home.dir", "C:\\Users\\Sri Sai Anusha\\Desktop\\Winutils")
    val conf = new SparkConf().setAppName("Queries").setMaster("local[*]")
    val sc = new SparkContext(conf)
    val sqlContext = new org.apache.spark.sql.SQLContext(sc)
    import sqlContext.implicits._

    val tweets = sqlContext.read.json("D:/Sem II/PB/Project 3/trends.txt").toDF()
    val trends = tweets.select(explode($"details.trends").as("trend"))
    trends.printSchema()

    val tweet = sqlContext.read.json("D:\\Sem II\\PB\\Project 1\\PB\\tweets.json")
    tweet.createOrReplaceTempView("tweets")

    val names = trends.select(explode($"trend.name").as("names"))
    names.printSchema()
    names.createOrReplaceTempView("names")

    val output = sqlContext.sql("SELECT t.text as text,d.names as hashtags from tweets t " +
      "JOIN names d on t.text like CONCAT('%',d.names,'%')")
    output.show()

  }
}
```

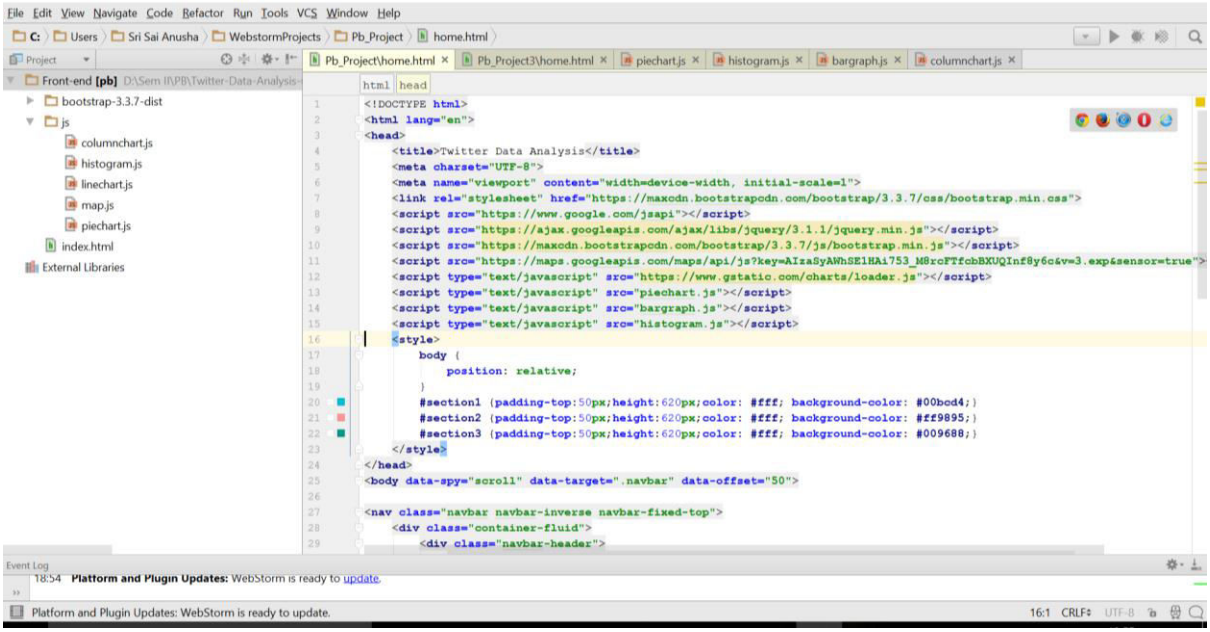
The output of the above query is shown below.

```
+--------------------+--------+
|                text|hashtags|
+--------------------+--------+
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|Marii kita ngejui...|    Ibra|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|RT @TheEaglesNest...|  Eagles|
|#Football 2015 Pa...|   Larry|
|#Football 2015 Pa...|   Larry|
```

# 7. EXTRA REQUIREMENT

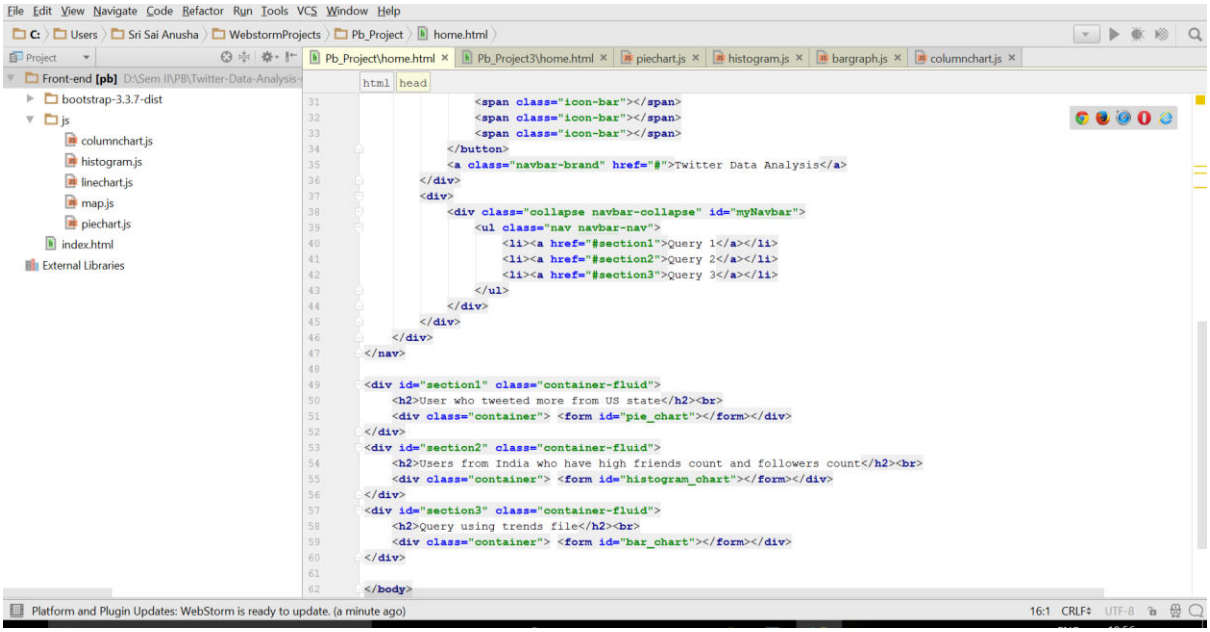
The visualization of the above queries is done for the dynamic representation of the queries.

The HTML code combining all the queries is shown below.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Twitter Data Analysis</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
    <script src="https://www.google.com/jsapi"></script>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyAWhSE1HAi753_M8rcFTfcbBXUQInf8y6c&v=3.exp&sensor=true">
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
    <script type="text/javascript" src="piechart.js"></script>
    <script type="text/javascript" src="bargraph.js"></script>
    <script type="text/javascript" src="histogram.js"></script>
    <style>
        body {
            position: relative;
        }
        #section1 {padding-top:50px;height:620px;color: #fff; background-color: #00bcd4;}
        #section2 {padding-top:50px;height:620px;color: #fff; background-color: #ff9895;}
        #section3 {padding-top:50px;height:620px;color: #fff; background-color: #009688;}
    </style>
</head>
<body data-spy="scroll" data-target=".navbar" data-offset="50">

<nav class="navbar navbar-inverse navbar-fixed-top">
    <div class="container-fluid">
        <div class="navbar-header">
```

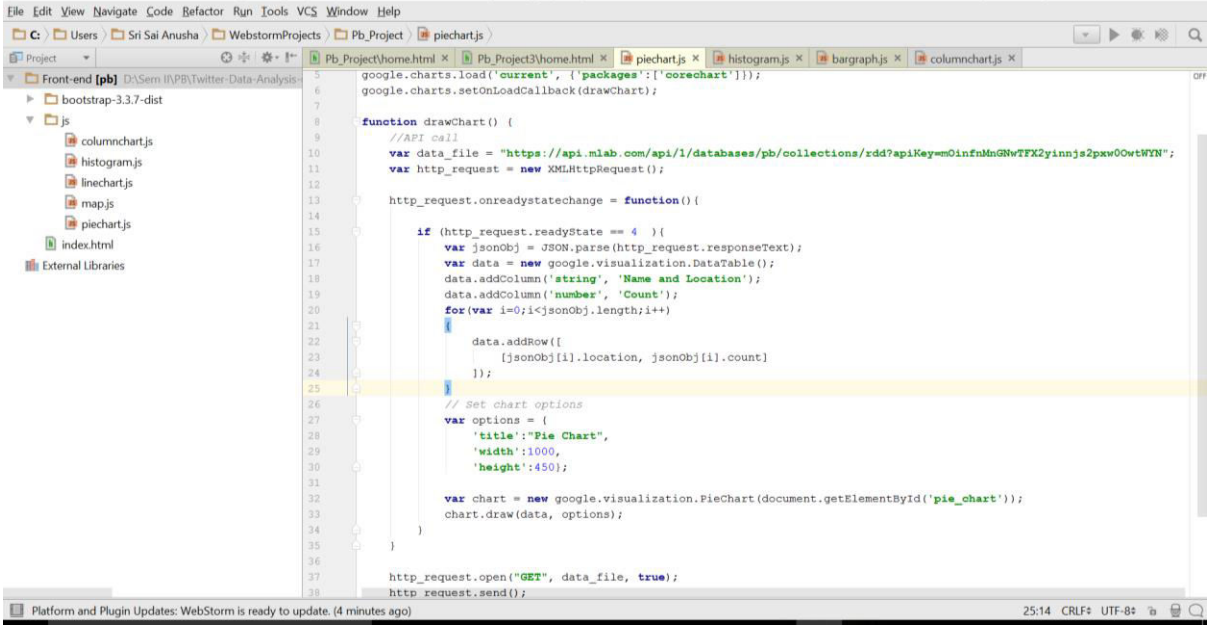


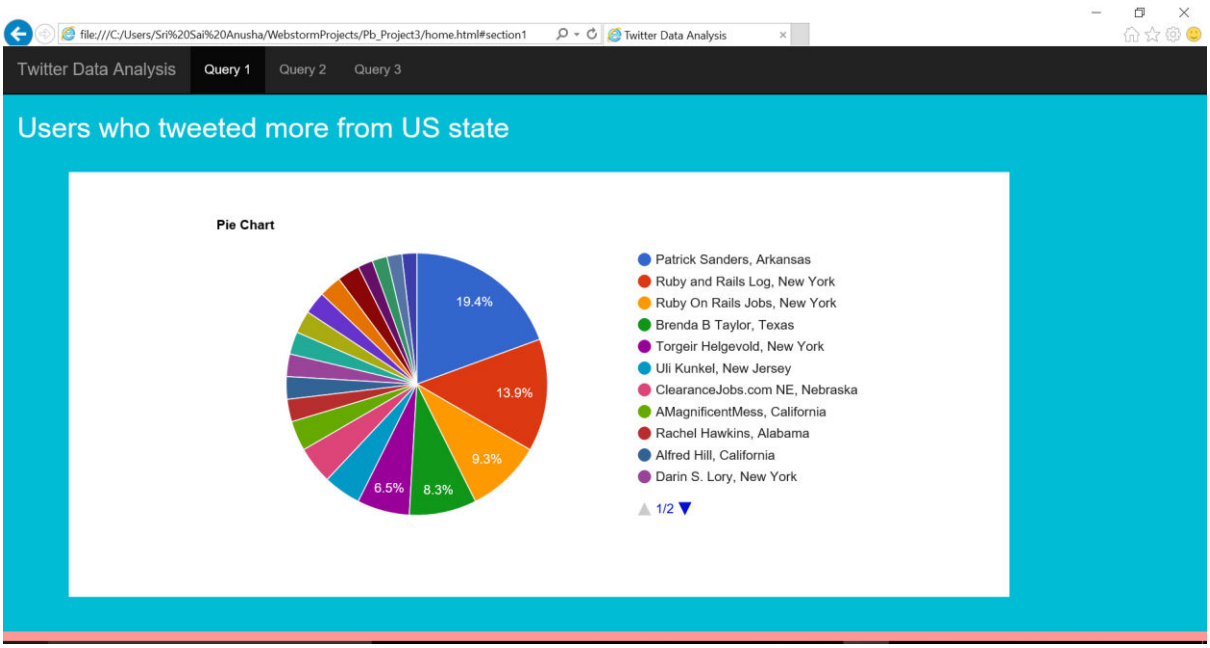
```html
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="#">Twitter Data Analysis</a>
        </div>
        <div>
            <div class="collapse navbar-collapse" id="myNavbar">
                <ul class="nav navbar-nav">
                    <li><a href="#section1">Query 1</a></li>
                    <li><a href="#section2">Query 2</a></li>
                    <li><a href="#section3">Query 3</a></li>
                </ul>
            </div>
        </div>
    </div>
</nav>

<div id="section1" class="container-fluid">
    <h2>User who tweeted more from US state</h2><br>
    <div class="container"> <form id="pie_chart"></form></div>
</div>
<div id="section2" class="container-fluid">
    <h2>Users from India who have high friends count and followers count</h2><br>
    <div class="container"> <form id="histogram_chart"></form></div>
</div>
<div id="section3" class="container-fluid">
    <h2>Query using trends file</h2><br>
    <div class="container"> <form id="bar_chart"></form></div>
</div>

</body>
```

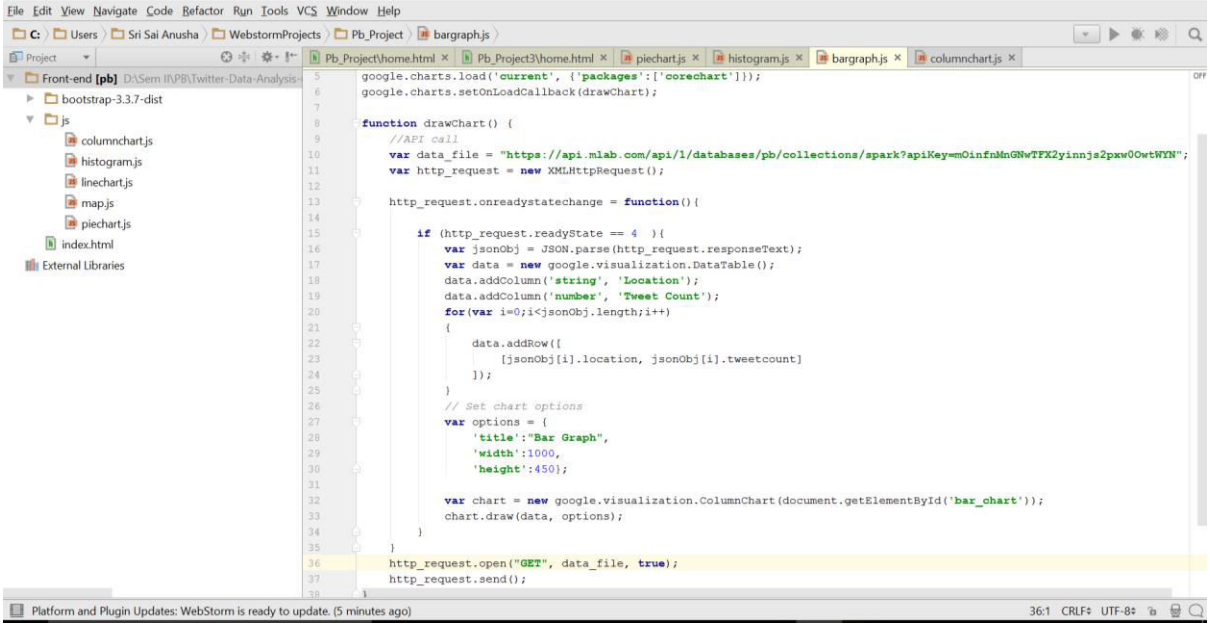
## 7.1 Pie Chart Representation of Query 1

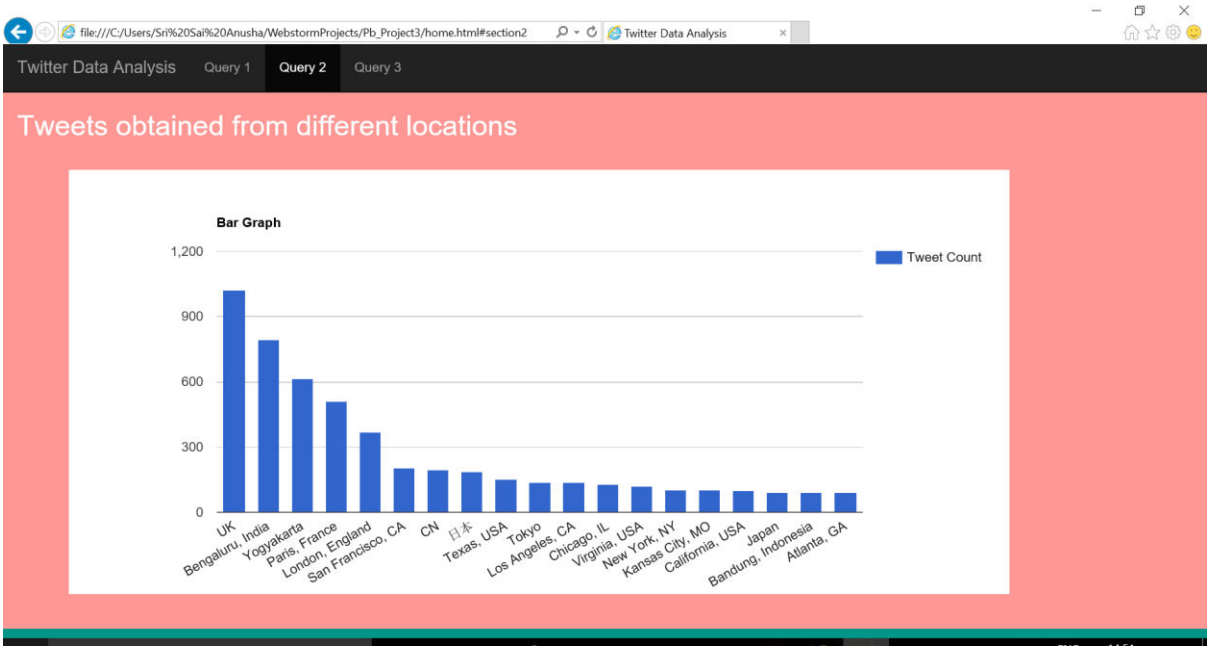



The dynamic representation of query 1 shows the users who tweeted the most from US state.

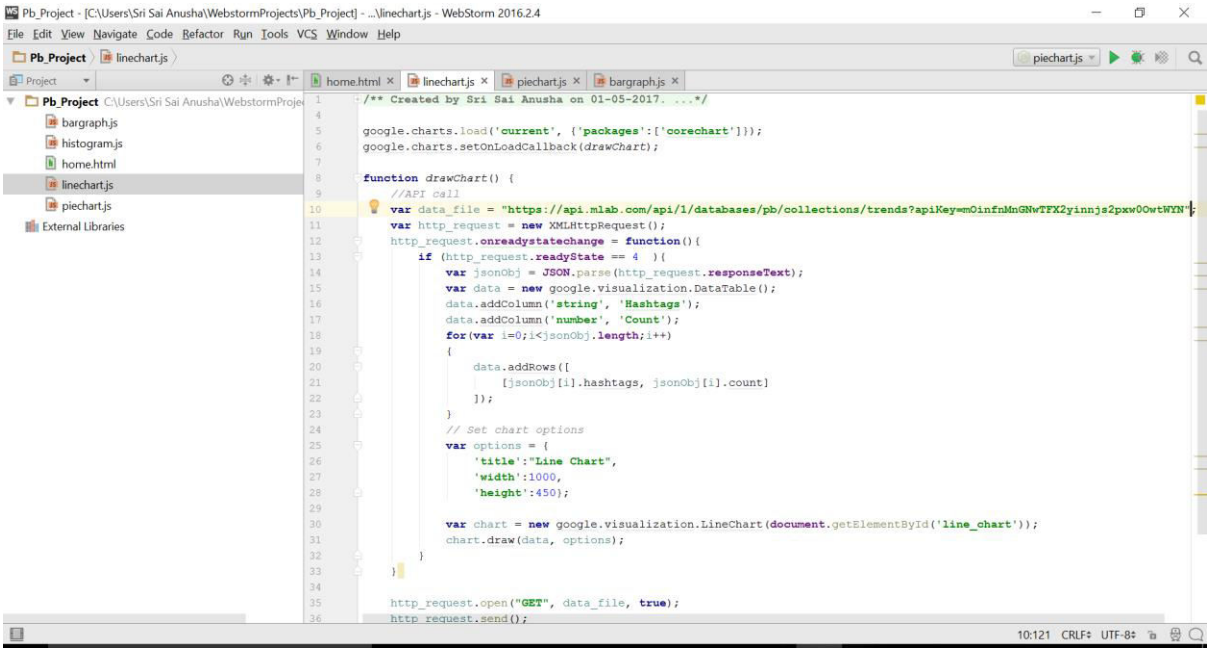
## 7.2 Bar Graph Representation of Query 2

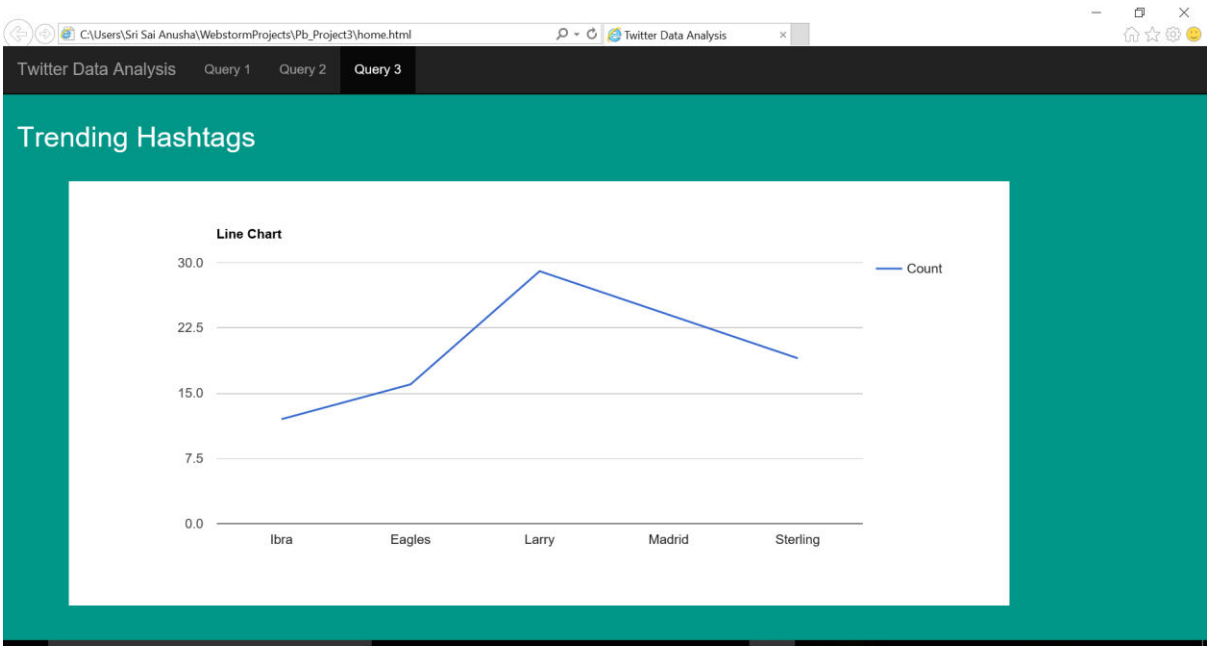



The dynamic representation of query 2 shows the tweets that have been obtained from different locations.

## 7.3 Line Chart Representation of Query 3




The dynamic representation of query 3 shows trending hashtags that are common in the tweets file and trends file.

**REFERENCES**


- www.github.com
- https://mlab.com/databases/pb
- https://developers.google.com/chart/interactive/docs/examples
- https://ikanow.com/how-to-visualize-json-documents-from-mongodb/
- https://mvnrepository.com/artifact/org.mongodb.spark/mongo-spark-connector_2.11/0.1