Experiment 7: Spark Word Count Program

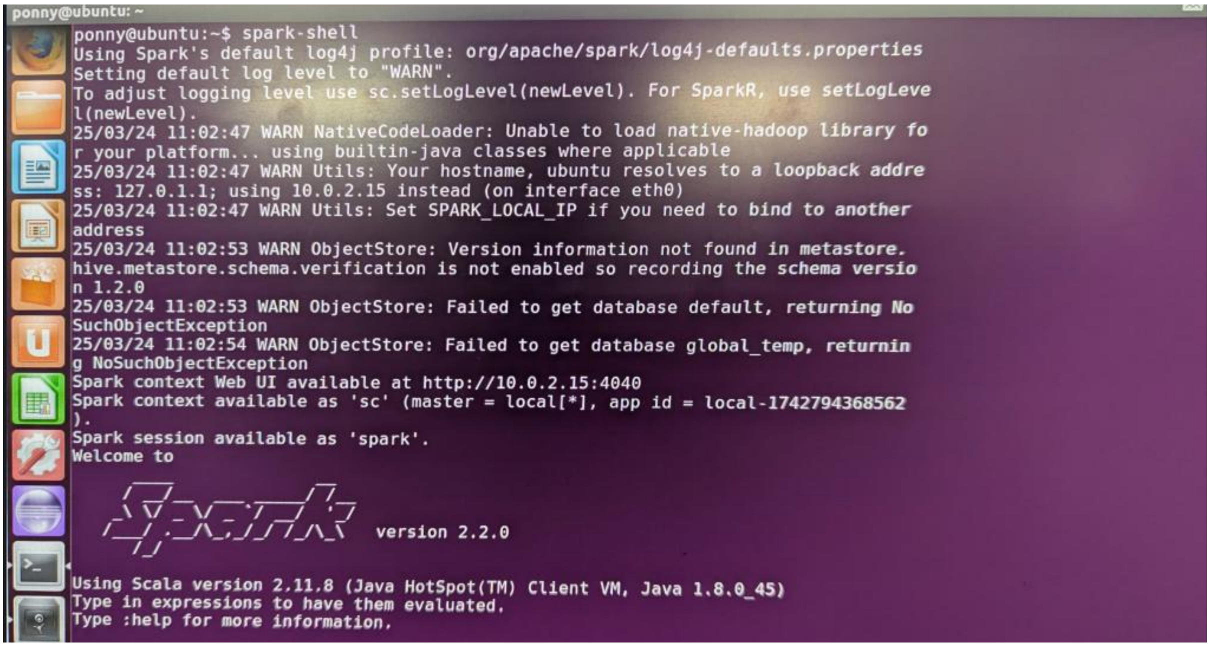
Namansh Singh Maurya
22MIA1034

Aim:

To count number of words in a Text File using spark.

Algorithm/Procedure:

1. Checking if Spark is present in the machine or not by using **spark-shell** command if it is present, we will spark version displayed on the screen, otherwise error.



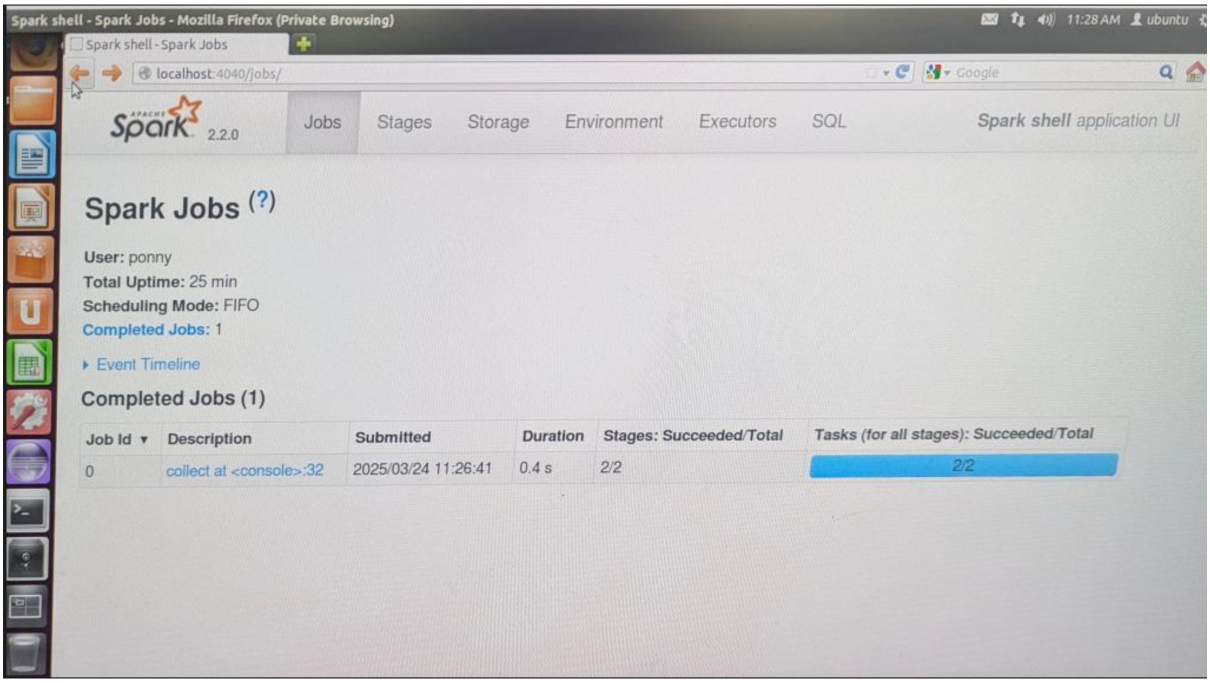
```
ponny@ubuntu: ~
ponny@ubuntu:~$ spark-shell
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLeve
l(newLevel).
25/03/24 11:02:47 WARN NativeCodeLoader: Unable to load native-hadoop library fo
r your platform... using builtin-java classes where applicable
25/03/24 11:02:47 WARN Utils: Your hostname, ubuntu resolves to a loopback addre
ss: 127.0.1.1; using 10.0.2.15 instead (on interface eth0)
25/03/24 11:02:47 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another
address
25/03/24 11:02:53 WARN ObjectStore: Version information not found in metastore.
hive.metastore.schema.verification is not enabled so recording the schema versio
n 1.2.0
25/03/24 11:02:53 WARN ObjectStore: Failed to get database default, returning No
SuchObjectException
25/03/24 11:02:54 WARN ObjectStore: Failed to get database global_temp, returnin
g NoSuchObjectException
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[*], app id = local-1742794368562
).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.2.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) Client VM, Java 1.8.0_45)
Type in expressions to have them evaluated.
Type :help for more information.
```

2. After Reviewing the version of Spark, we can move on for making a text file named as **test.txt** in that we will write something.

3. Then we will use spark’s commands to do the word counting the codes are given below for the same.

4. We can check **localhost** also for Spark just like we used to do in **Hadoop**.



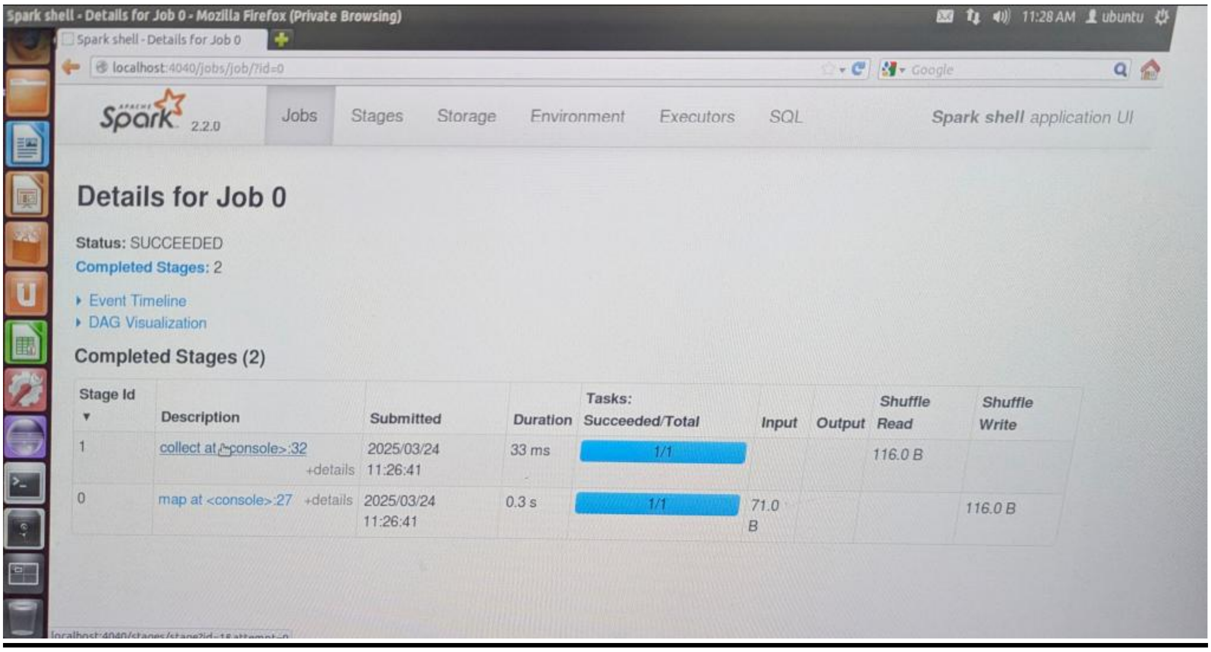
Spark Jobs (?)

User: ponny
Total Uptime: 25 min
Scheduling Mode: FIFO
Completed Jobs: 1

▸ Event Timeline

Completed Jobs (1)

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	collect at <console>:32	2025/03/24 11:26:41	0.4 s	2/2	2/2


Spark shell - Details for Job 0 - Mozilla Firefox (Private Browsing)
11:28 AM ubuntu
Spark shell - Details for Job 0
localhost:4040/jobs/job/?id=0
Google
Spark 2.2.0
Jobs
Stages
Storage
Environment
Executors
SQL
Spark shell application UI
localhost:4040/stages/stage?id=1&attempt=0

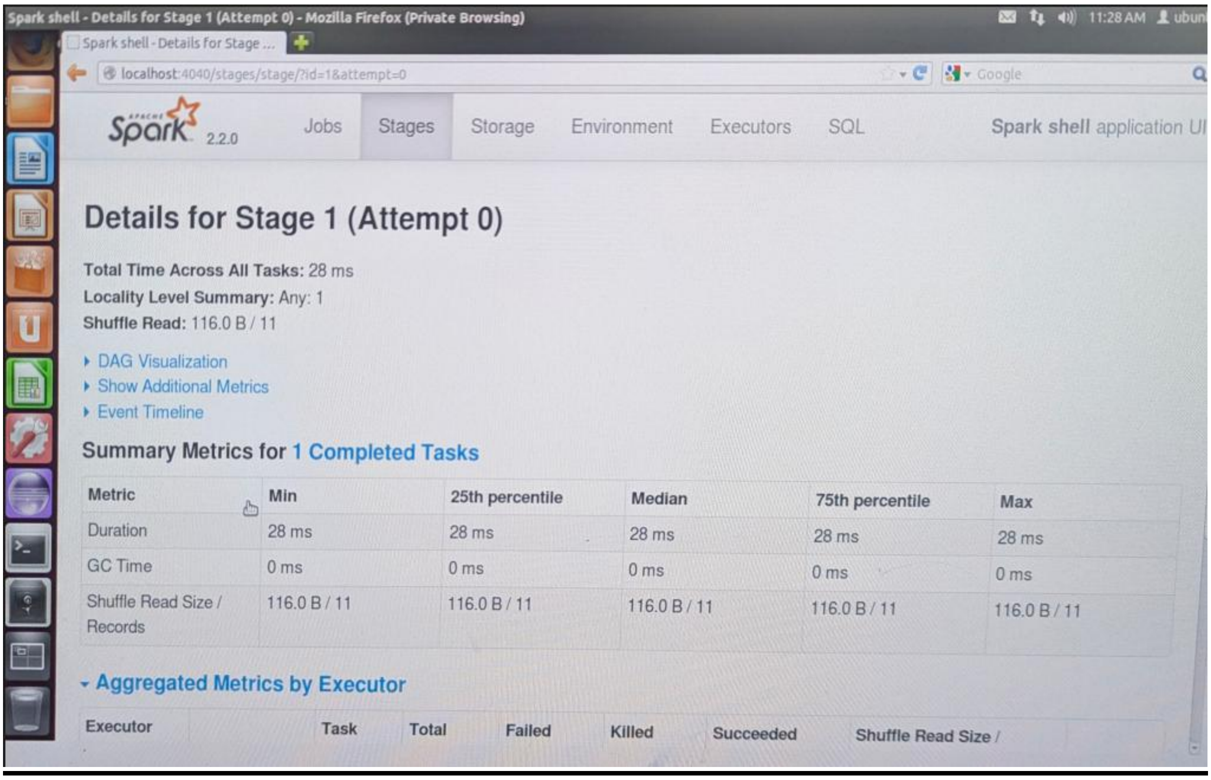
Details for Job 0

Status: SUCCEEDED
Completed Stages: 2

▸ Event Timeline
▸ DAG Visualization

Completed Stages (2)

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
1	collect at <console>:32 +details	2025/03/24 11:26:41	33 ms	1/1			116.0 B	
0	map at <console>:27 +details	2025/03/24 11:26:41	0.3 s	1/1	71.0 B			116.0 B


Spark shell - Details for Stage 1 (Attempt 0) - Mozilla Firefox (Private Browsing)
11:28 AM
Spark shell - Details for Stage ...
localhost:4040/stages/stage/?id=1&attempt=0
Google
Spark 2.2.0
Jobs
Stages
Storage
Environment
Executors
SQL
Spark shell application UI

Details for Stage 1 (Attempt 0)

Total Time Across All Tasks: 28 ms
Locality Level Summary: Any: 1
Shuffle Read: 116.0 B / 11

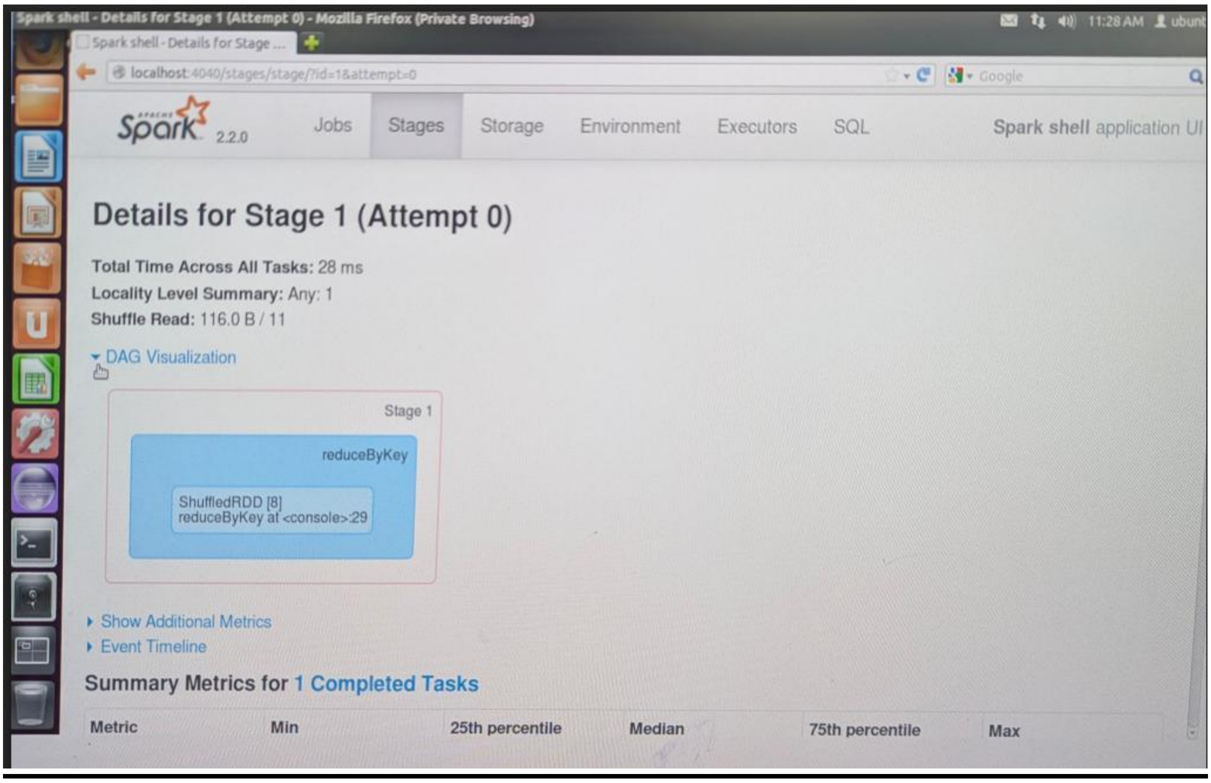
- DAG Visualization
- Show Additional Metrics
- Event Timeline

Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	28 ms	28 ms	28 ms	28 ms	28 ms
GC Time	0 ms	0 ms	0 ms	0 ms	0 ms
Shuffle Read Size / Records	116.0 B / 11	116.0 B / 11	116.0 B / 11	116.0 B / 11	116.0 B / 11

Aggregated Metrics by Executor

Executor	Task	Total	Failed	Killed	Succeeded	Shuffle Read Size /


Spark shell - Details for Stage 1 (Attempt 0) - Mozilla Firefox (Private Browsing)
11:28 AM
Spark shell - Details for Stage ...
localhost:4040/stages/stage/?id=1&attempt=0
Google
Spark 2.2.0
Jobs
Stages
Storage
Environment
Executors
SQL
Spark shell application UI

Details for Stage 1 (Attempt 0)

Total Time Across All Tasks: 28 ms
Locality Level Summary: Any: 1
Shuffle Read: 116.0 B / 11

- DAG Visualization

Stage 1
reduceByKey
ShuffledRDD [8]
reduceByKey at <console>:29

- Show Additional Metrics
- Event Timeline

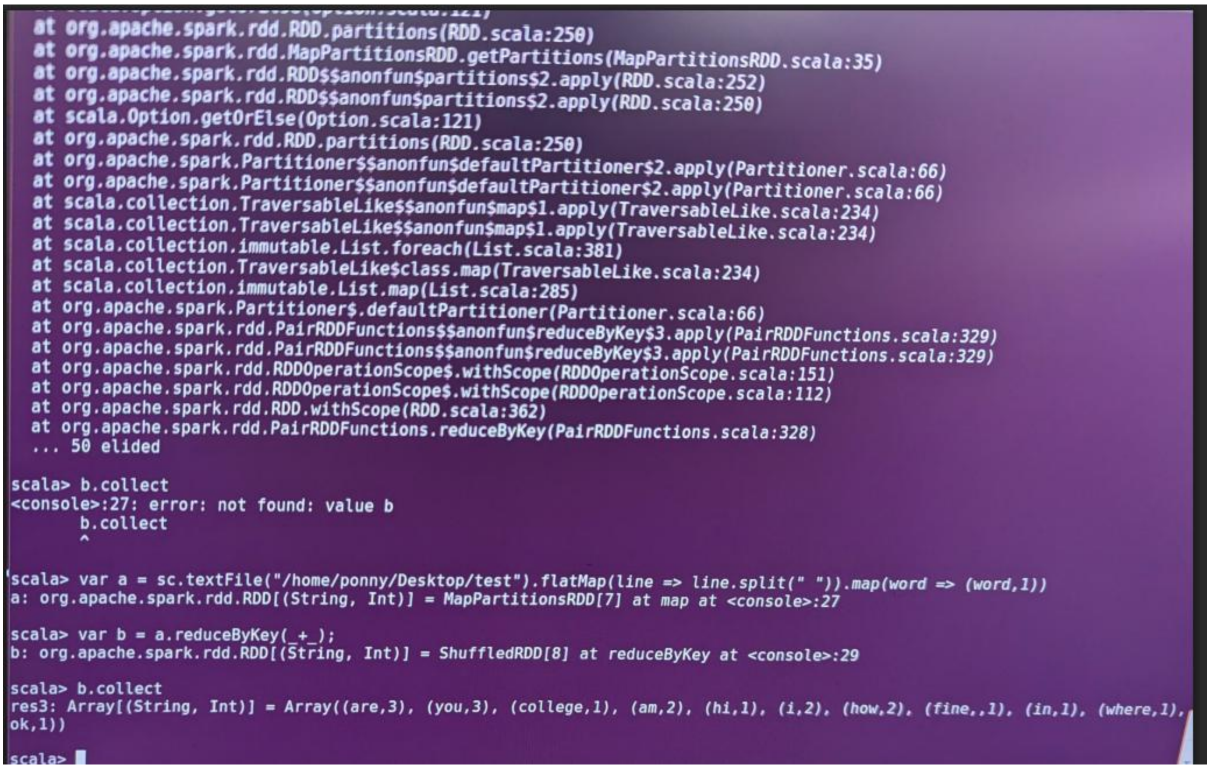
Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max

Program:

```
var a = sc.textFile("/home/ponny/Desktop/test").flatMap(line =? line.split("
")).map(word => (word,1))
var b = a.reduceByKey(_+_);
b.collect
```

Output:



```
  at scala.Option.getOrElse(Option.scala:121)
  at org.apache.spark.rdd.RDD.partitions(RDD.scala:250)
  at org.apache.spark.rdd.MapPartitionsRDD.getPartitions(MapPartitionsRDD.scala:35)
  at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:252)
  at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:250)
  at scala.Option.getOrElse(Option.scala:121)
  at org.apache.spark.rdd.RDD.partitions(RDD.scala:250)
  at org.apache.spark.Partitioner$$anonfun$defaultPartitioner$2.apply(Partitioner.scala:66)
  at org.apache.spark.Partitioner$$anonfun$defaultPartitioner$2.apply(Partitioner.scala:66)
  at scala.collection.TraversableLike$$anonfun$map$1.apply(TraversableLike.scala:234)
  at scala.collection.TraversableLike$$anonfun$map$1.apply(TraversableLike.scala:234)
  at scala.collection.immutable.List.foreach(List.scala:381)
  at scala.collection.TraversableLike$class.map(TraversableLike.scala:234)
  at scala.collection.immutable.List.map(List.scala:285)
  at org.apache.spark.Partitioner$.defaultPartitioner(Partitioner.scala:66)
  at org.apache.spark.rdd.PairRDDFunctions$$anonfun$reduceByKey$3.apply(PairRDDFunctions.scala:329)
  at org.apache.spark.rdd.PairRDDFunctions$$anonfun$reduceByKey$3.apply(PairRDDFunctions.scala:329)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:362)
  at org.apache.spark.rdd.PairRDDFunctions.reduceByKey(PairRDDFunctions.scala:328)
  ... 50 elided

scala> b.collect
<console>:27: error: not found: value b
       b.collect
       ^

scala> var a = sc.textFile("/home/ponny/Desktop/test").flatMap(line => line.split(" ")).map(word => (word,1))
a: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[7] at map at <console>:27

scala> var b = a.reduceByKey(_+_);
b: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[8] at reduceByKey at <console>:29

scala> b.collect
res3: Array[(String, Int)] = Array((are,3), (you,3), (college,1), (am,2), (hi,1), (i,2), (how,2), (fine,,1), (in,1), (where,1),
ok,1))

scala> 
```

Result:

Hence we used spark to count number of words in a given text file using Spark’s commands.