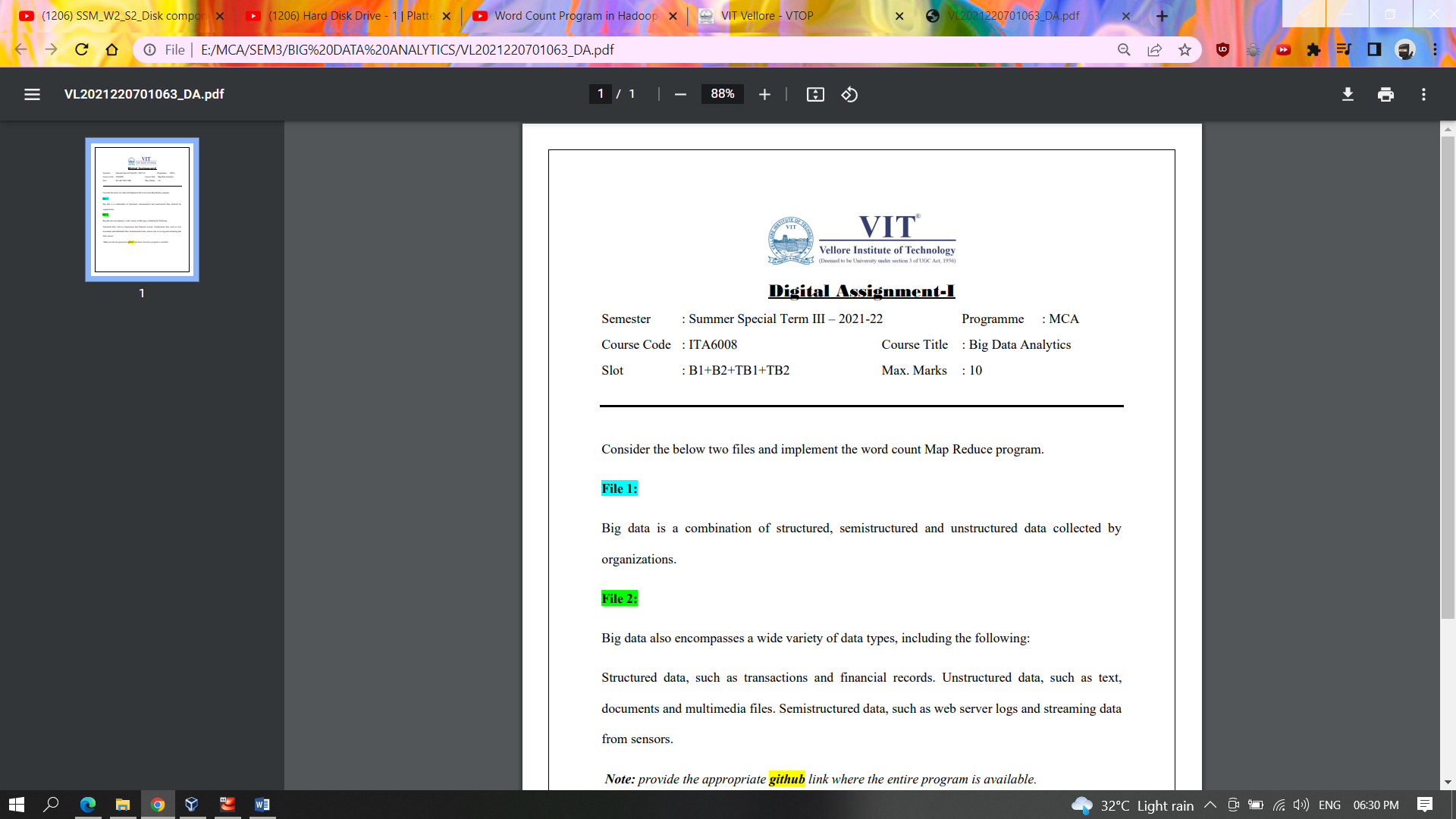
BIG DATA ANALYTICS

Digital Assignment-1

Name:Vivek Kumar Singh Reg. no:21MCA0280



GitHub link:

**Word count** implemention using **HADOOP map reduce:**

Creating a project in Eclipse

a.Creating a package name,class name as WordCount2

b.Writing the word count program

package WordCount2;

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

public static class TokenizerMapper

extends Mapper<Object, Text, Text, IntWritable>{

private final static IntWritable one = new IntWritable(1);

private Text word = new Text();

public void map(Object key, Text value, Context context

) throws IOException, InterruptedException {

StringTokenizer itr = new StringTokenizer(value.toString());

while (itr.hasMoreTokens()) {

word.set(itr.nextToken());

context.write(word, one);

}

}

}

public static class IntSumReducer

extends Reducer<Text,IntWritable,Text,IntWritable> {

private IntWritable result = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values,

Context context

) throws IOException, InterruptedException {

int sum = 0;

for (IntWritable val : values) {

sum += val.get();

}

result.set(sum);

context.write(key, result);

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "word count");

job.setJarByClass(WordCount.class);

job.setMapperClass(TokenizerMapper.class);

job.setCombinerClass(IntSumReducer.class);

job.setReducerClass(IntSumReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

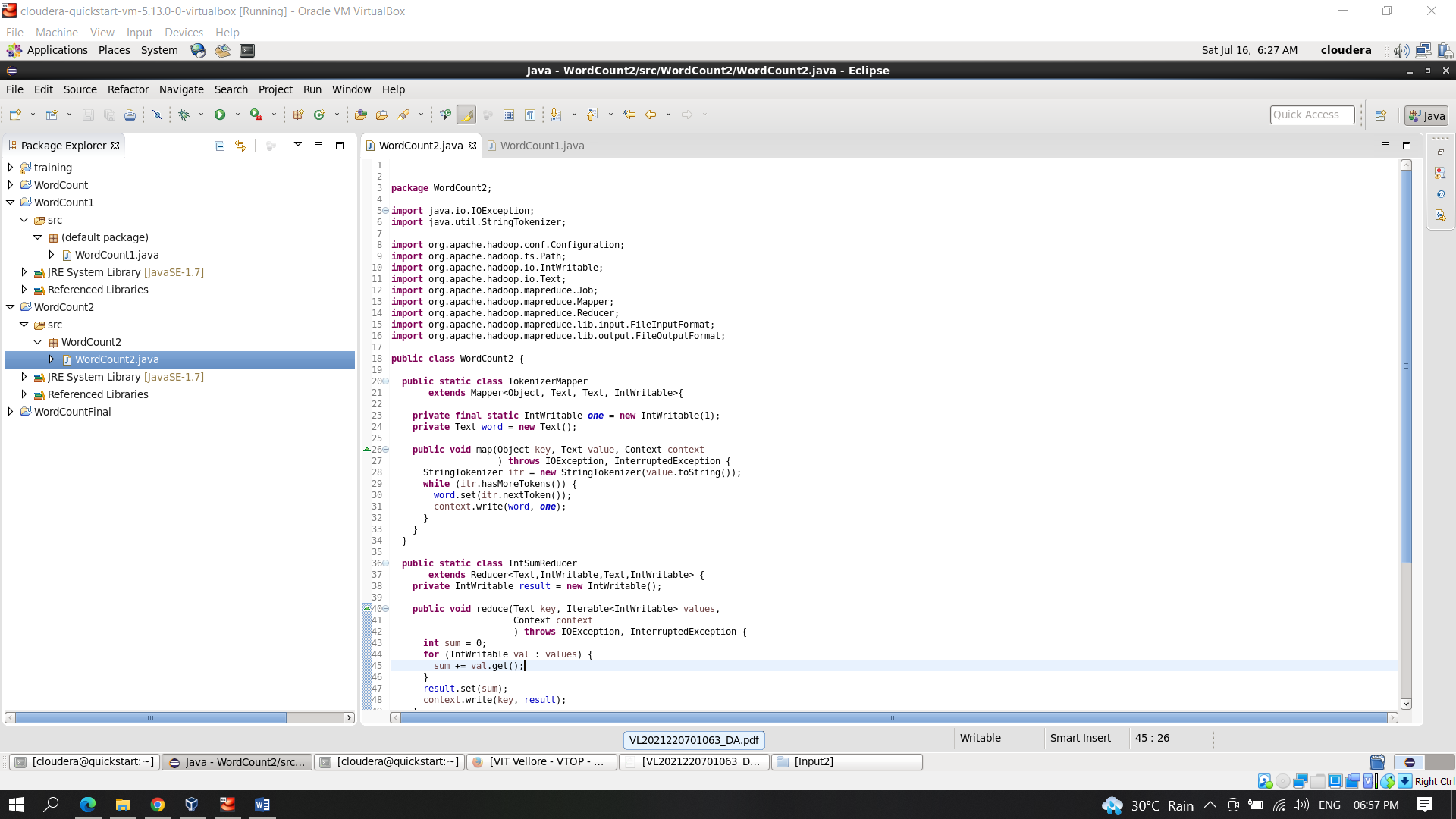
FileOutputFormat.setOutputPath(job, new Path(args[1]));

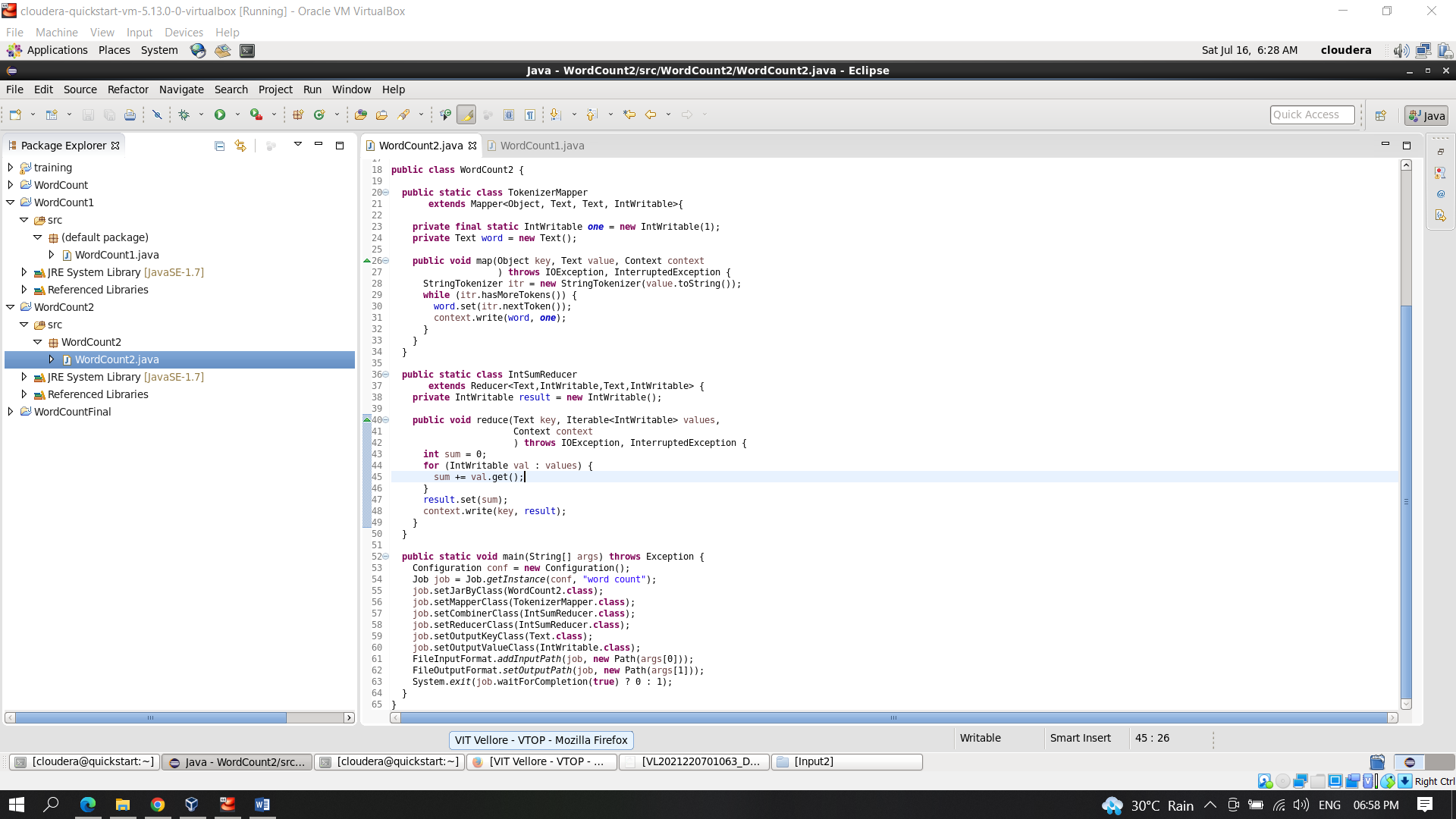
System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}

Screenshot

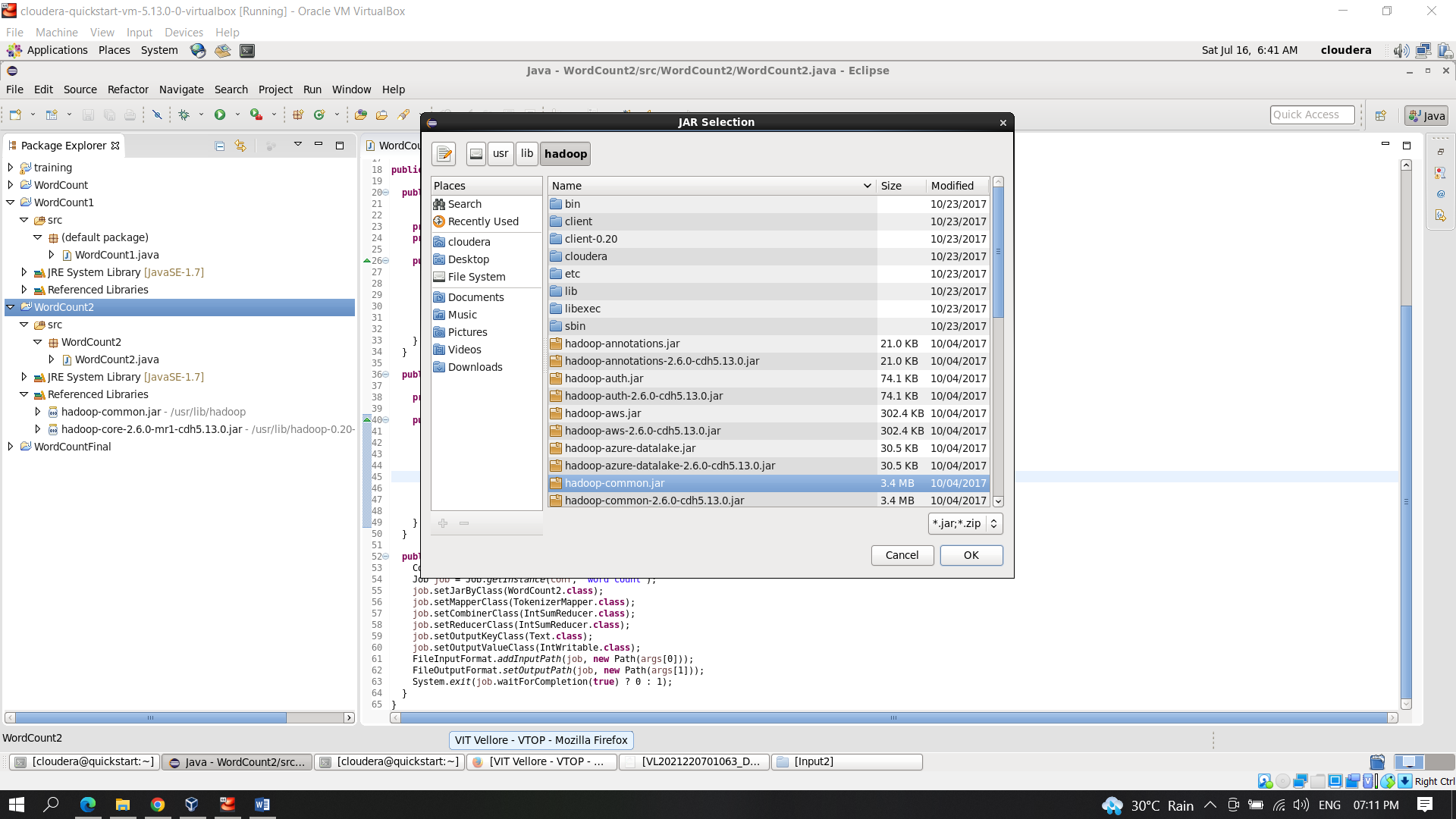




Add External Archives

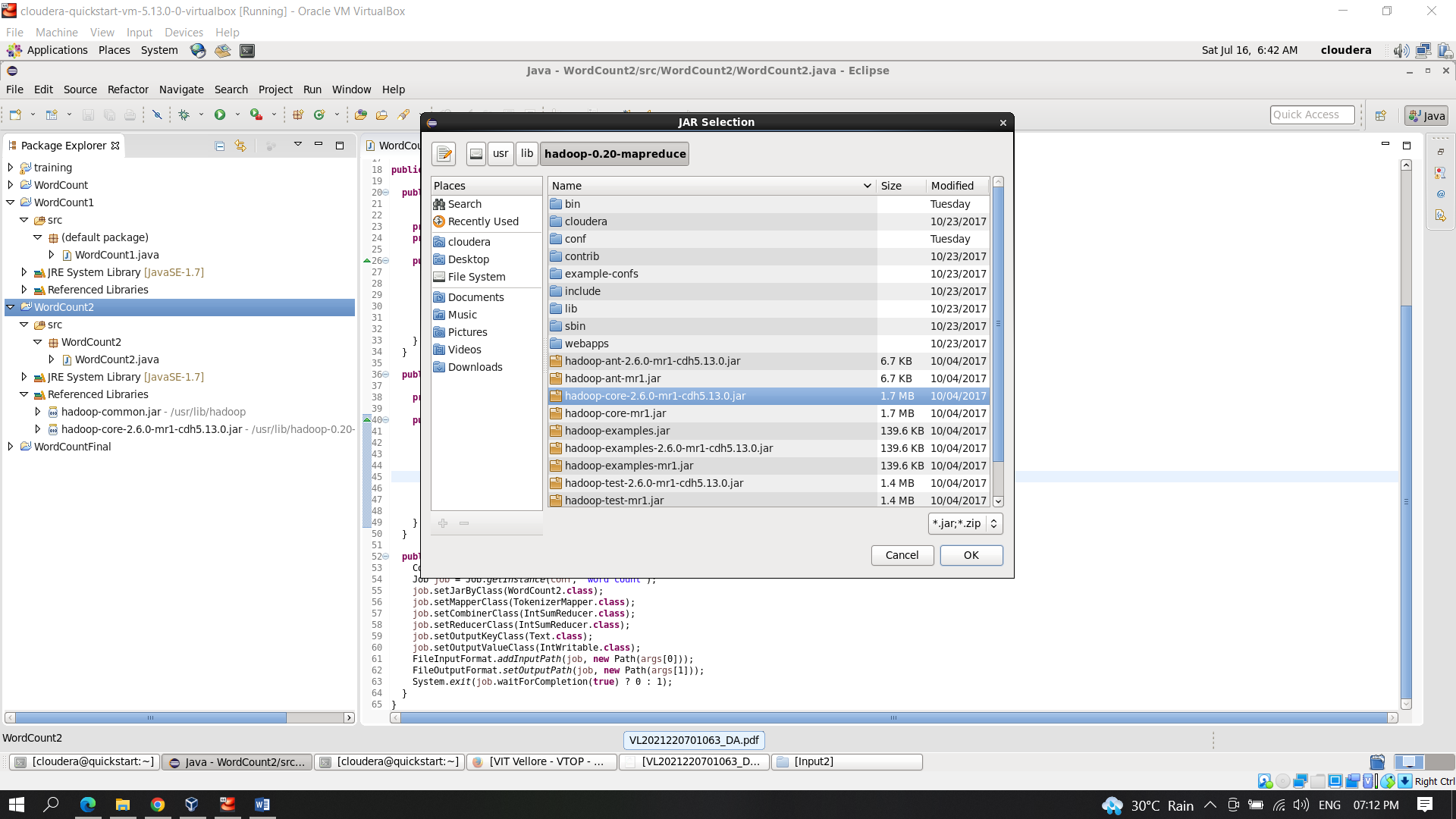
c.Include hadoop-common.jar

Write click on project file > go to the build path > add external archives > go to File System > go to usr > go to lib > go to hadoop > select hadoop-common-jar > click ok



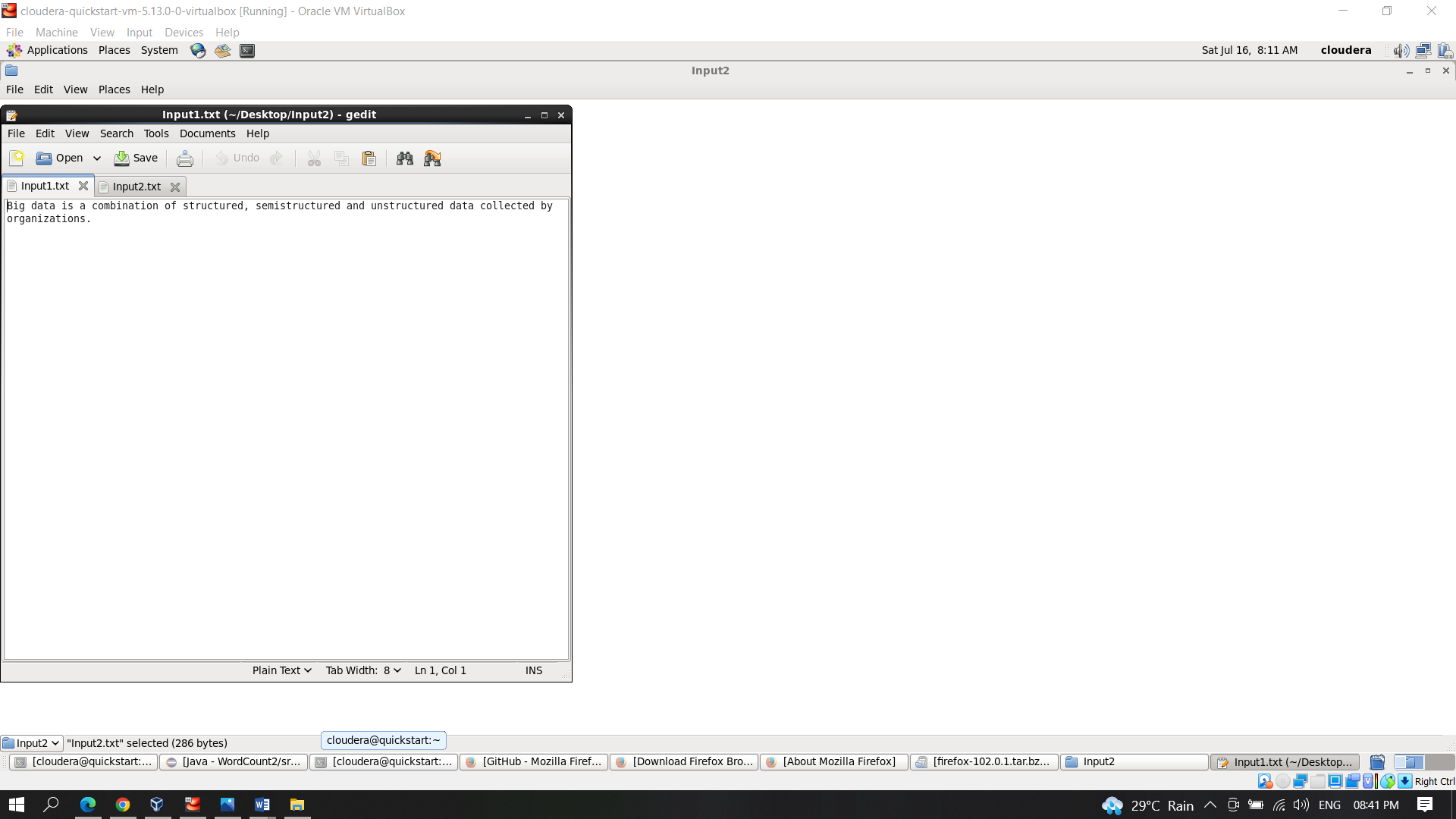
d. Include hadoop-core-2.6.0-mr1-cdh5.4.2.jar

Write click on project file > go to the build path > add external archives > go to File System > go to usr > go to lib > go to hadoop-0.20-mapreduce > select hadoop-core-2.6.0-mr1-cdh5.4.2.jar > click ok

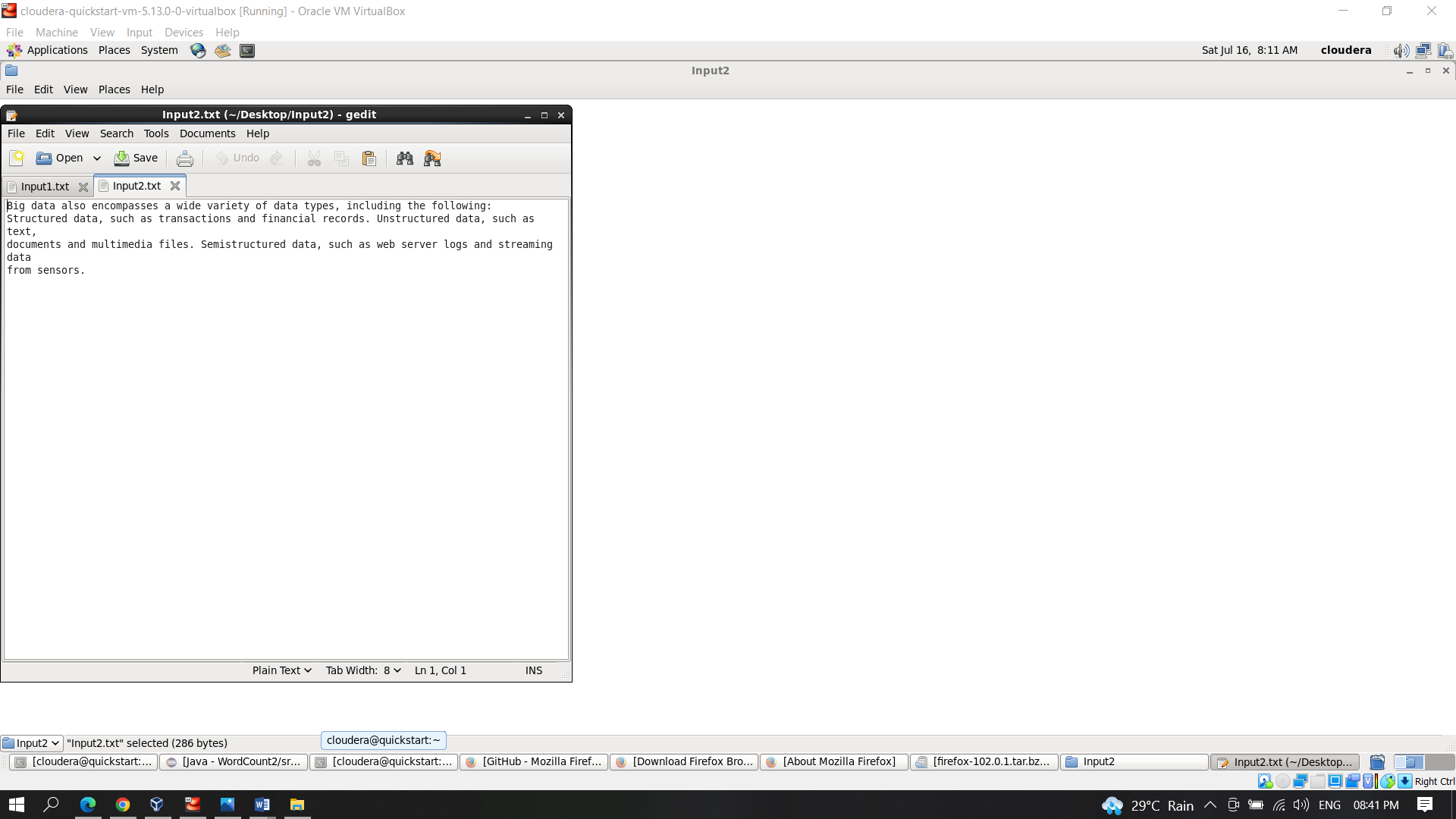


e.Create the text files inside the folder “Input2”

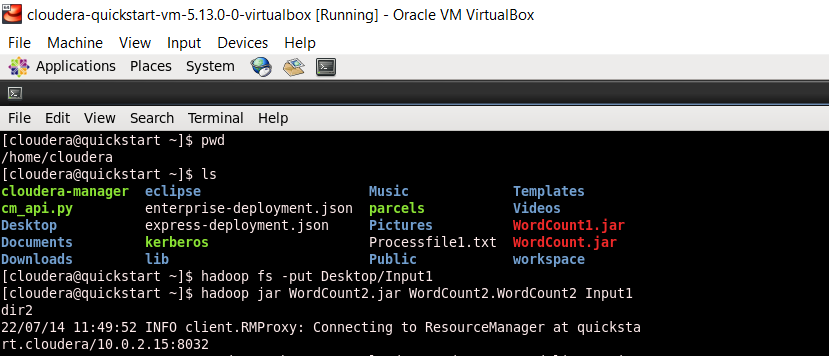
Input1.txt



Input2.txt



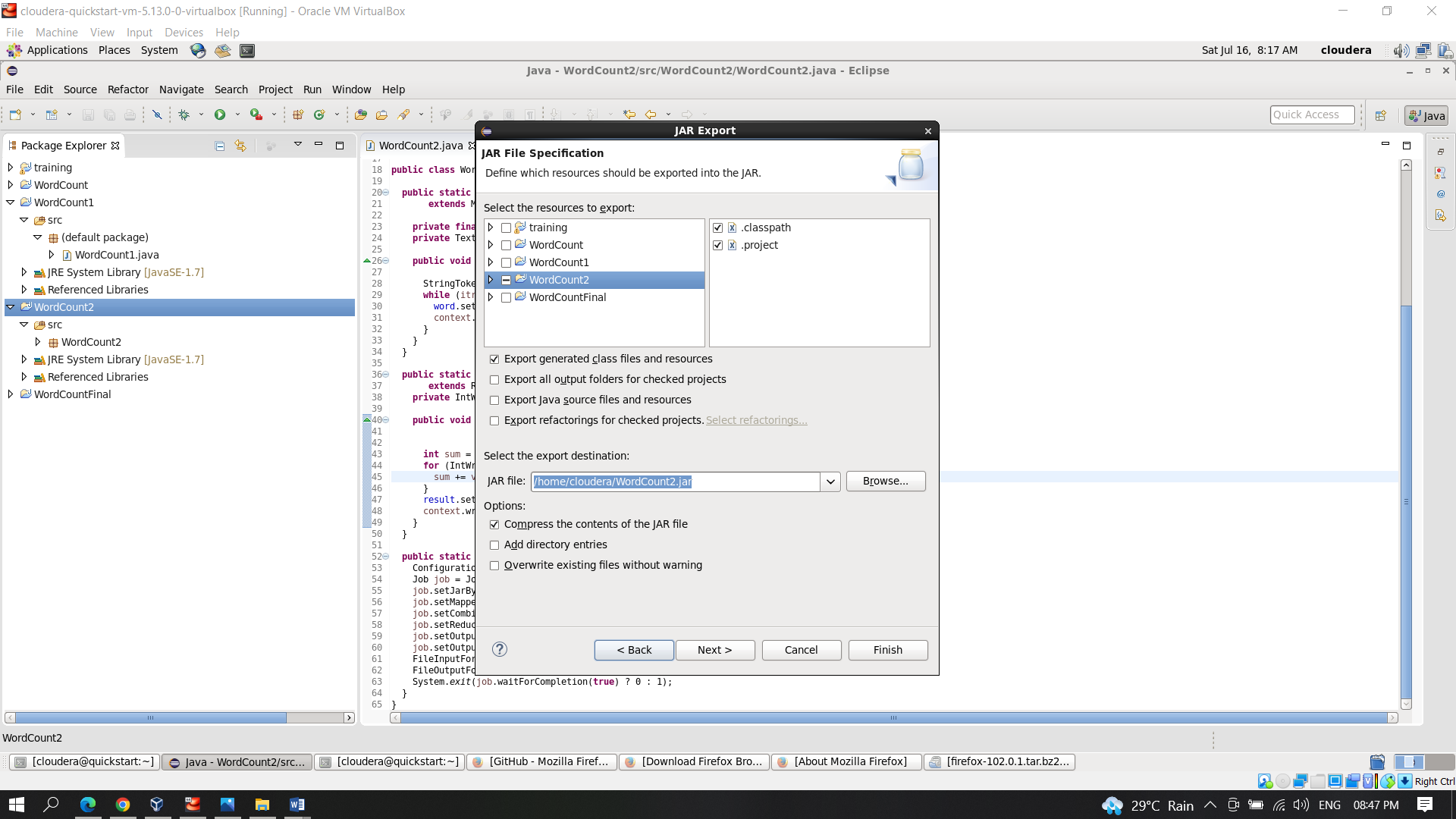
f.Moving the folder to hadoop environment



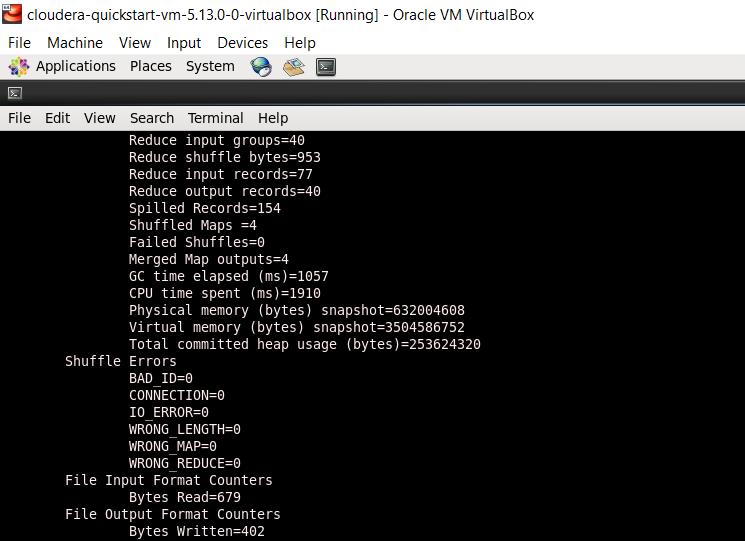
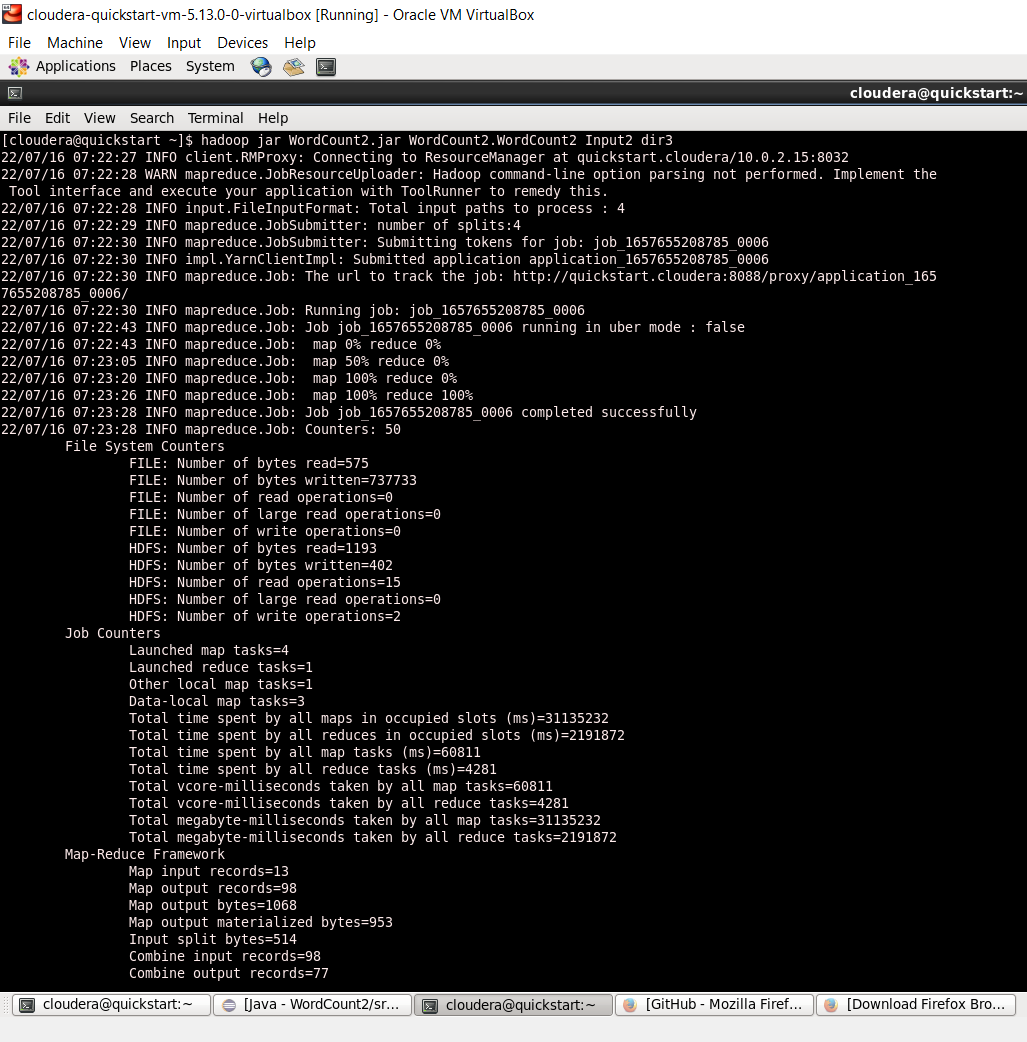
g.Now to execute our WordCount programe we need a jar file

For this go to your program > right click on the program file <

Click export > click jar file > click next > save it as WordCount2.jar in “cloudera” folder



f.Implementing the mapreduce programe in hadoop



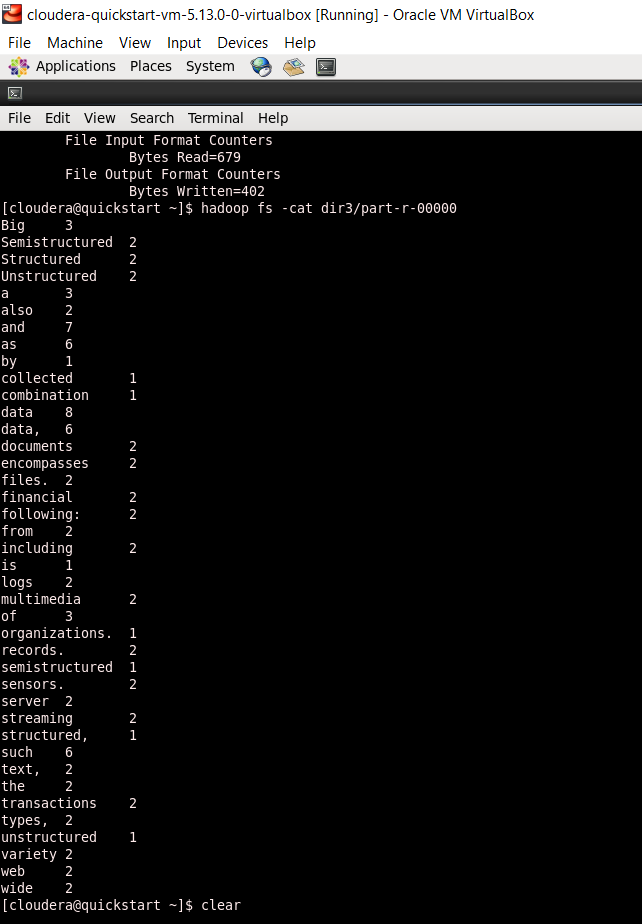
Conclusion

No. of Splits=4

No. of Mappers=4

No. of Reducers=1

Execute the programe



**output**

