Practical 7

Implementing Map-reduce program for word count problem

Theory: Apache Hadoop is an open-source framework designed for distributed storage and processing of large datasets using a cluster of computers. It is particularly suited for handling unstructured and semi-structured data. Here's an overview:

Core Components of Hadoop

MapReduce

- **Purpose**: Distributed data processing framework.
- **Features**:
  - Breaks tasks into small chunks, processes them in parallel, and aggregates results.
  - Ideal for batch processing of large datasets.

Steps:

A)Checking the Hadoop

Open the terminal and type the following command

hadoop version

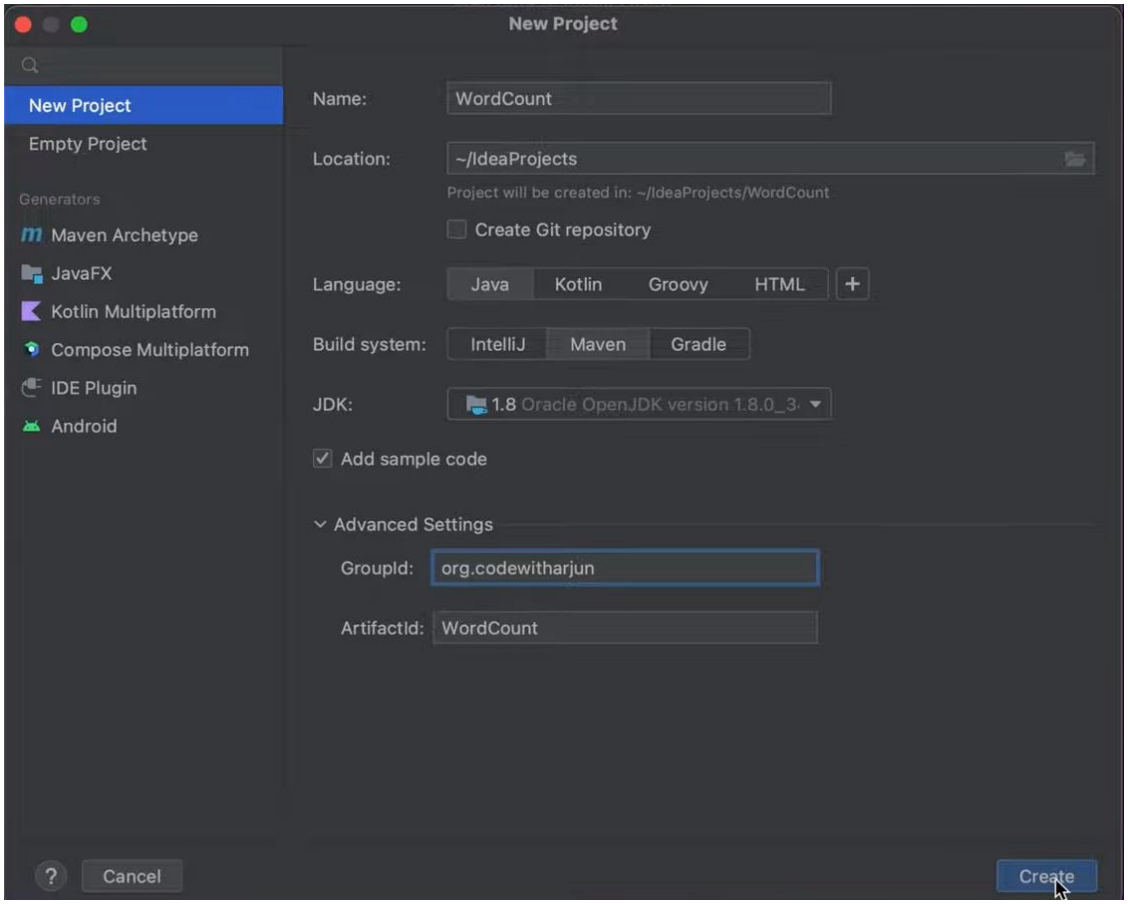
start-all.sh

jps

open the browser and check localhost:9870

B) Setting up environments & files for wordcount

Step:1 – open intellij→new project→Details:

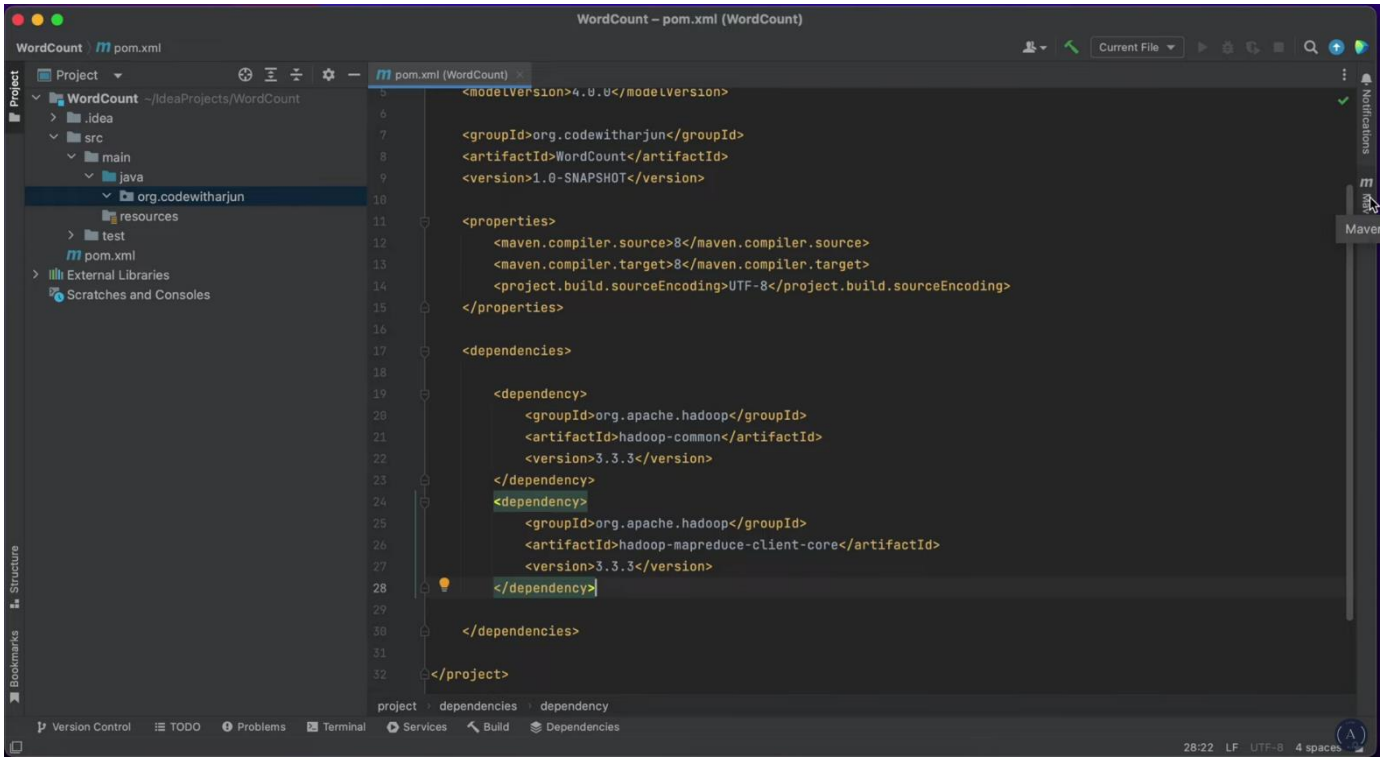


<u>Step:2-</u> On projects→java→delete main class then in org.codewitharjun edit & add the dependicies

These are the dependencies you have add on pom.xml file.

Dependencies :

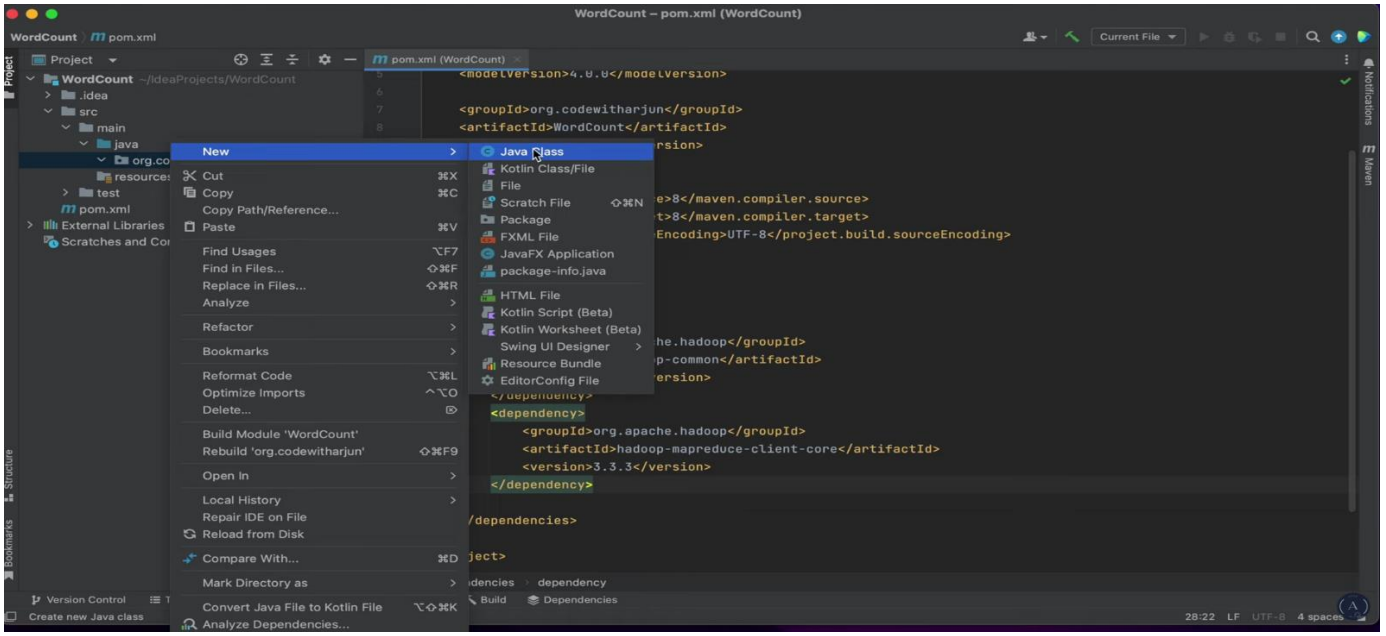
```
<dependency>
    <groupId>org.apache.hadoop</groupId>
    <artifactId>hadoop-common</artifactId>
    <version>3.4.1</version>
</dependency>
<dependency>
    <groupId>org.apache.hadoop</groupId>
    <artifactId>hadoop-mapreduce-client-core</artifactId>
    <version>3.4.1</version>
</dependency>
```



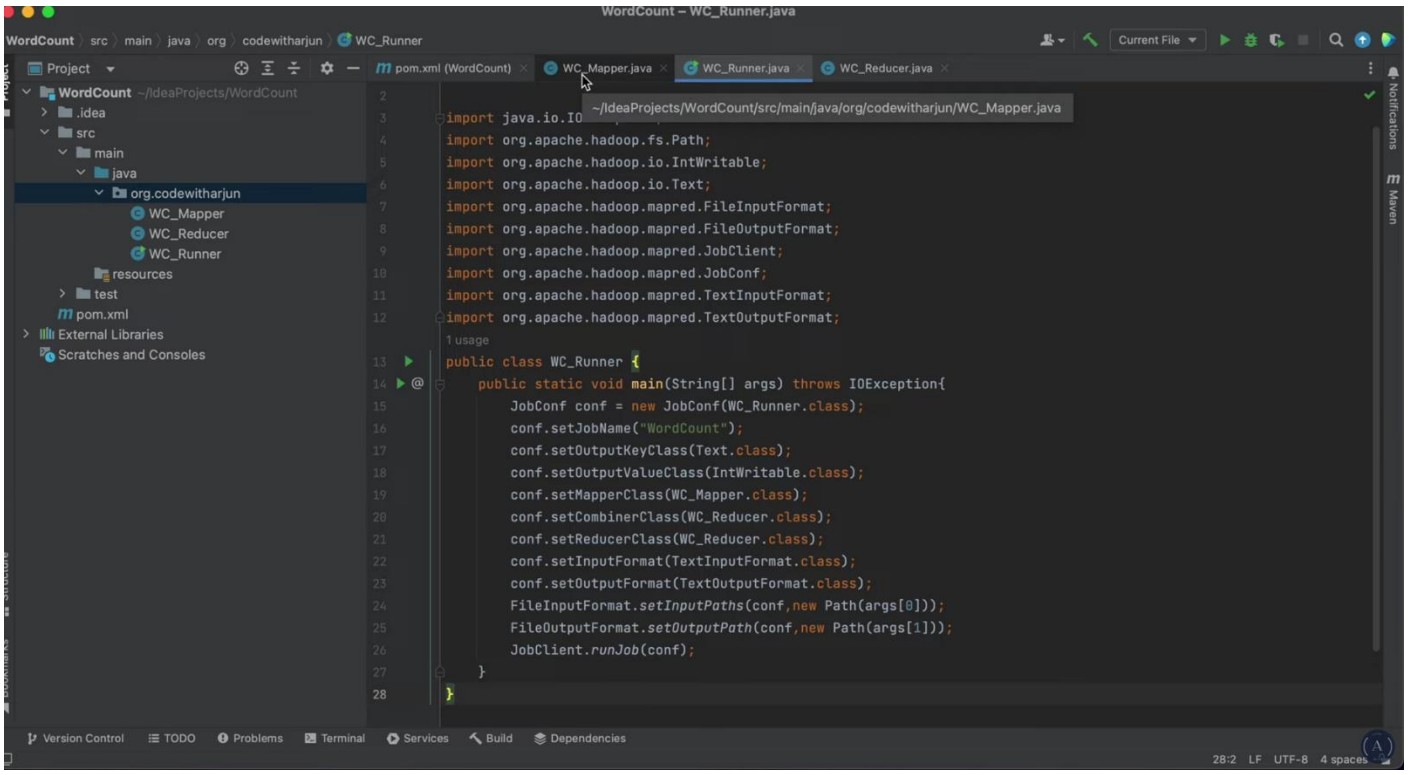
<u>Step:3-</u> right click on org.codewitharjun & create new→java class→WC_Mapper.java

Similarly create WC_Runner.java

WC_Reducer.java



1)WC_Mapper.java file and add the following code on it.



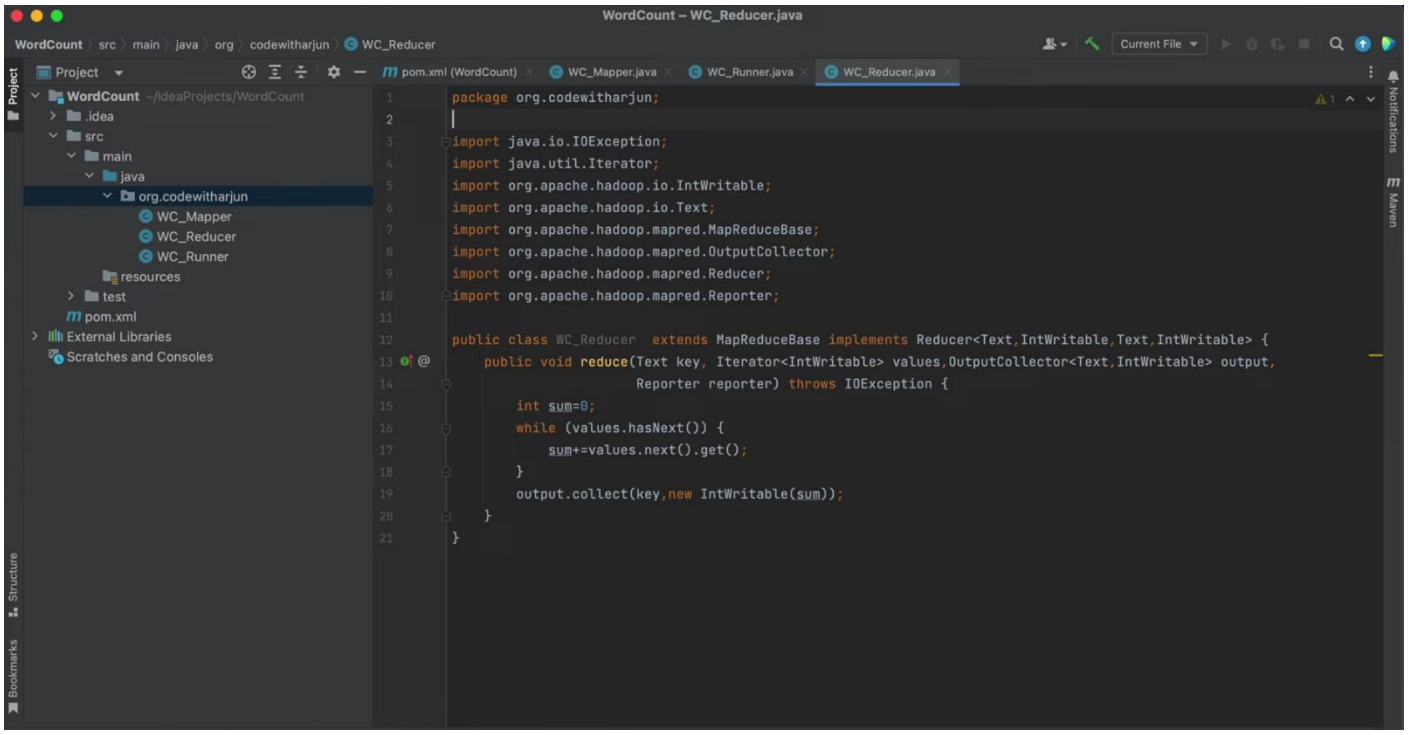
code:

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
```

```
public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();
   public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
                 Reporter reporter) throws IOException{
      String line = value.toString();
      StringTokenizer  tokenizer = new StringTokenizer(line);
      while (tokenizer.hasMoreTokens()){
         word.set(tokenizer.nextToken());
         output.collect(word, one);
      }
   }

}
```

2) WC_Runner.java file and add the following code on it.



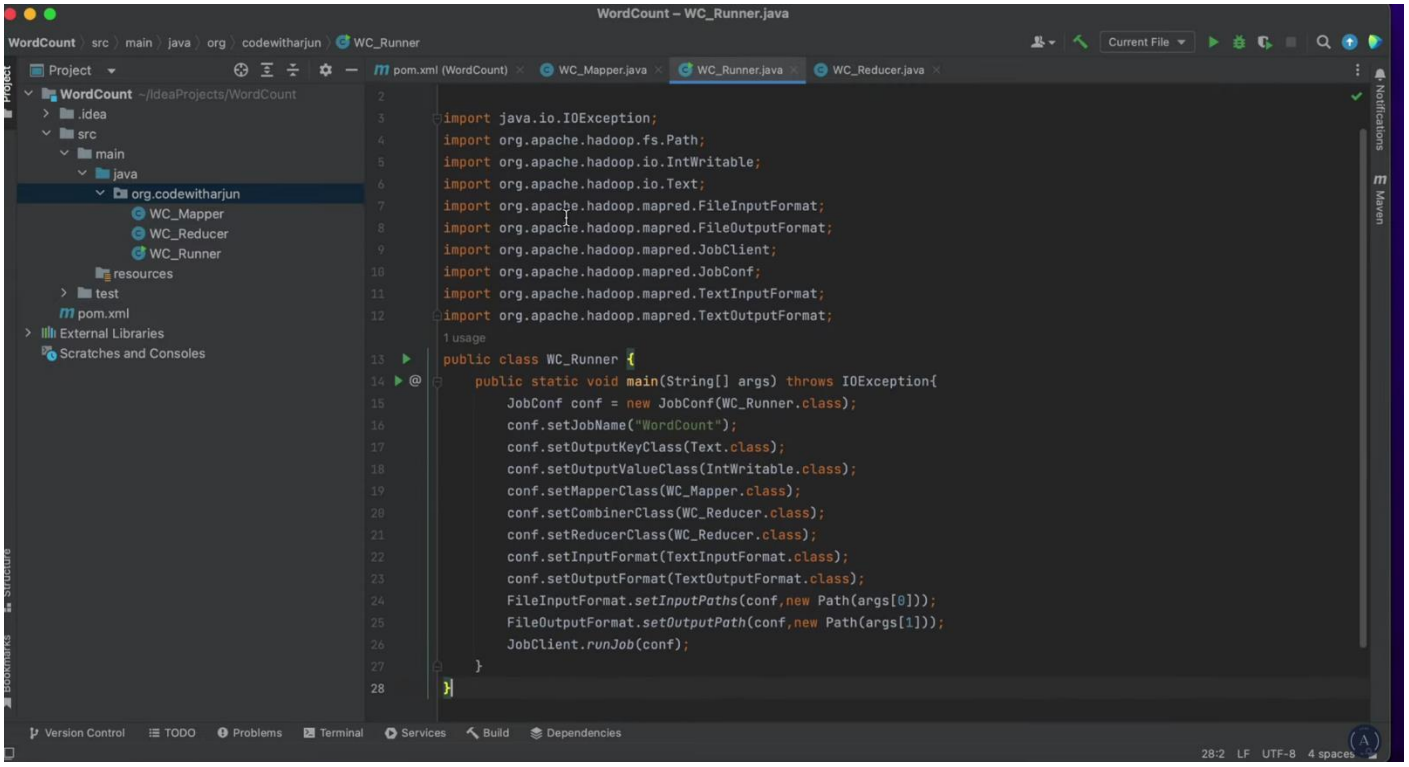
Code:

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
```

```
public class WC_Reducer extends MapReduceBase implements Reducer<Text,IntWritable,Text,IntWritable> {
  public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> output,
                Reporter reporter) throws IOException {
    int sum=0;
    while (values.hasNext()) {
      sum+=values.next().get();
    }
    output.collect(key,new IntWritable(sum));
  }
}
```

3) WC_Reducer.java file and add the following code on it.

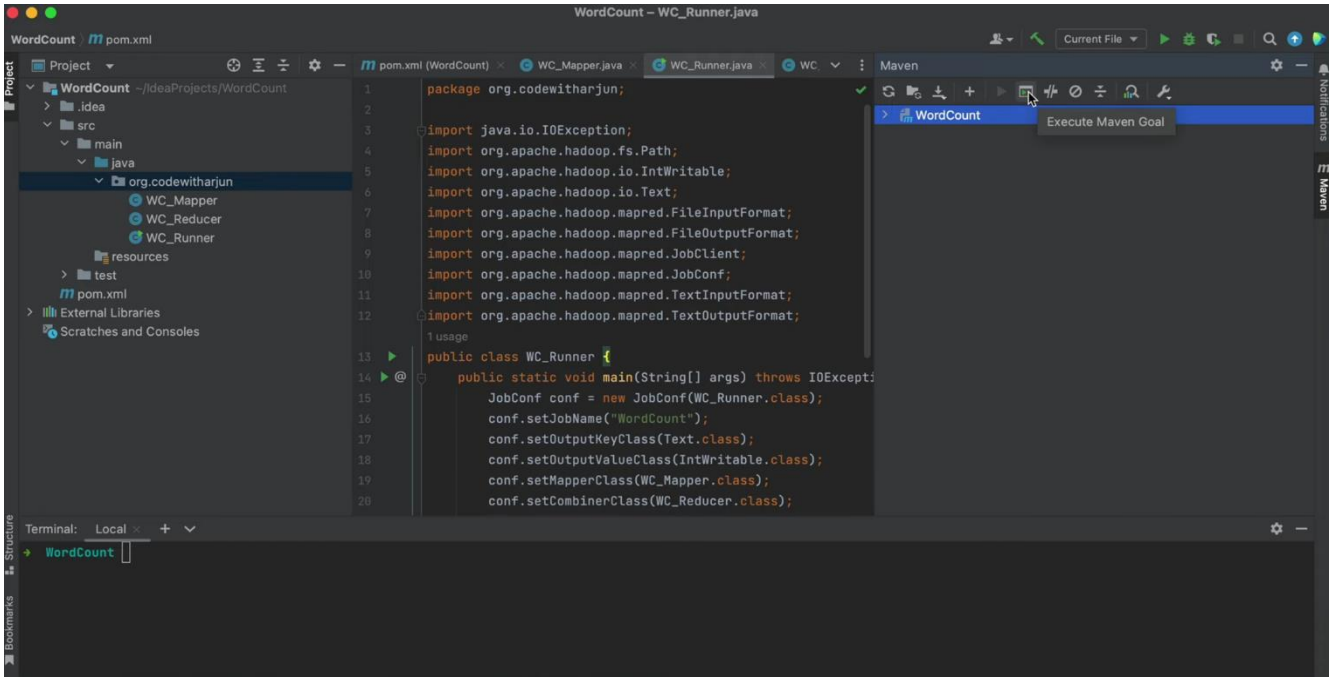
Code:




```
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
public class WC_Runner {
  public static void main(String[] args) throws IOException{
```

```
        JobConf conf = new JobConf(WC_Runner.class);
        conf.setJobName("WordCount");
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(WC_Mapper.class);
        conf.setCombinerClass(WC_Reducer.class);
        conf.setReducerClass(WC_Reducer.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf,new Path(args[0]));
        FileOutputFormat.setOutputPath(conf,new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

Step:4- In right end side in the middle→click on maven→excute maven goal→mvn clean
Again click on maven→excute maven goal→mvn install



<u>Step:5-</u> Create input file & put into Hadoop file system →open terminal →type the following command
Cmd:
cd desktop
nano input.txt

Note: Inside this folder type your own message

<u>Step:6-</u> open terminal →type the following command
Cmd:
cat input.txt

Hadoop fs-mkdir /input

Step:7- **Go to browser →search→localhost:9870→utilities→browse the file system**

Step:8- **open terminal →type the following command**

Cmd:

Hadoop fs-put input.txt /input

Step:9- **open intellij →terminal:local →wordcount**

Cmd:

hadoop jar /target/WordCount-1.0-SNAPSHOT.jar org.codewitharjun.WC_Runner /input/input.txt /output

Step:10- **Go to browser →search→localhost:9870→browse the file→output→part-000**

Step:11- **open terminal →type the following command**

Cmd:

Hadoop fs-cat /output/part-00000