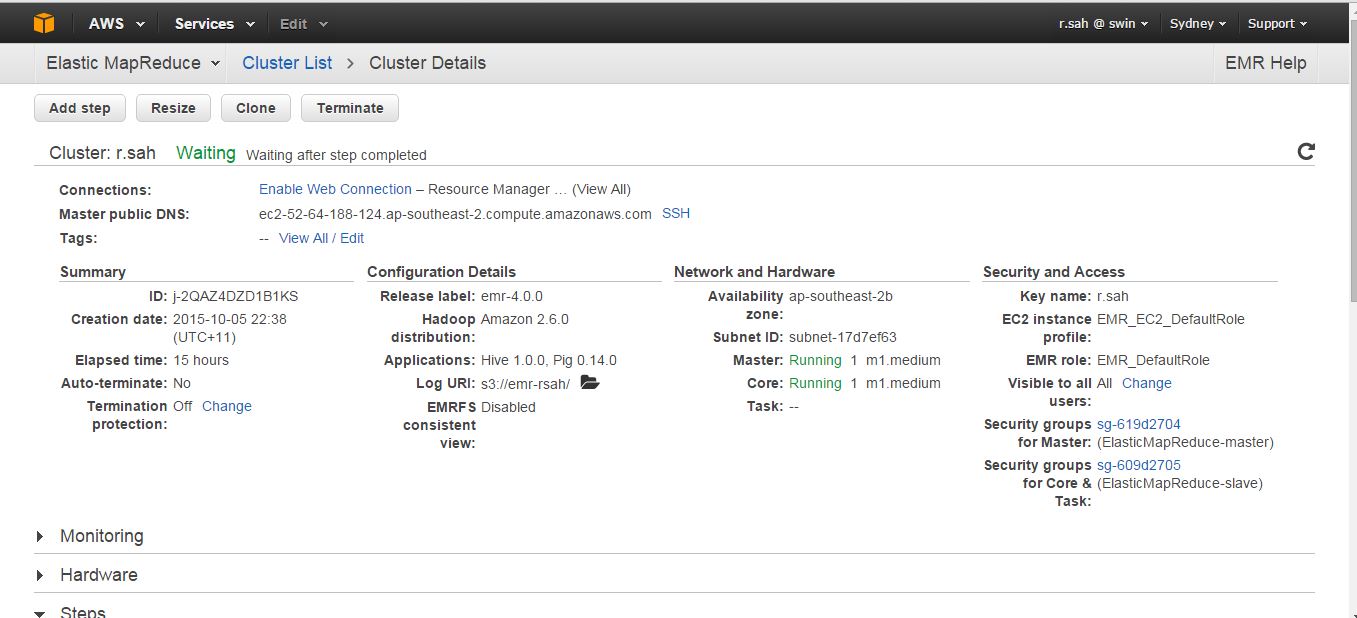
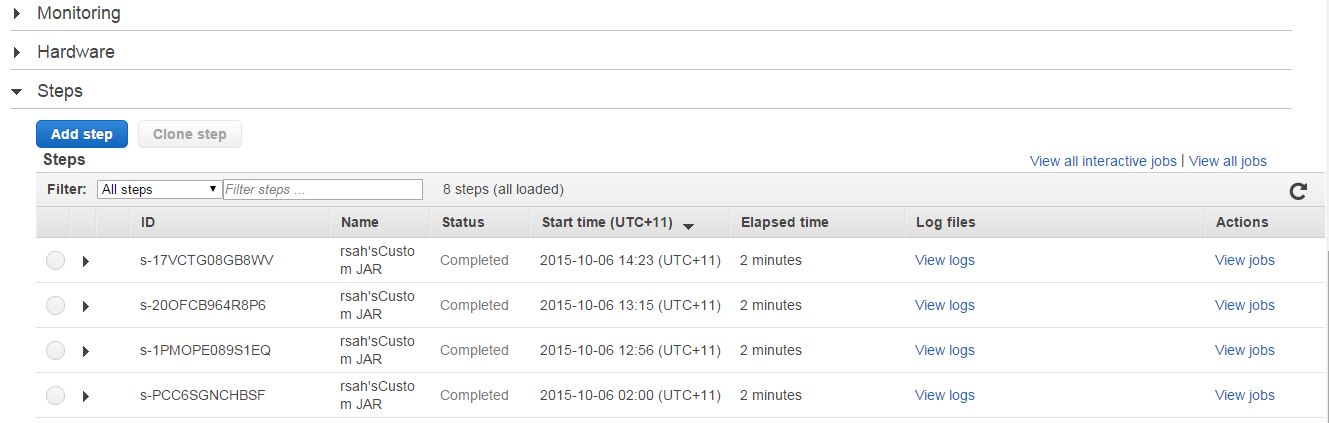
**Rabindra Sah  
ID-4963571  
Unit code-COS80001  
Title-Submission for Assignment 4**

**Software Development for Cloud Computing**

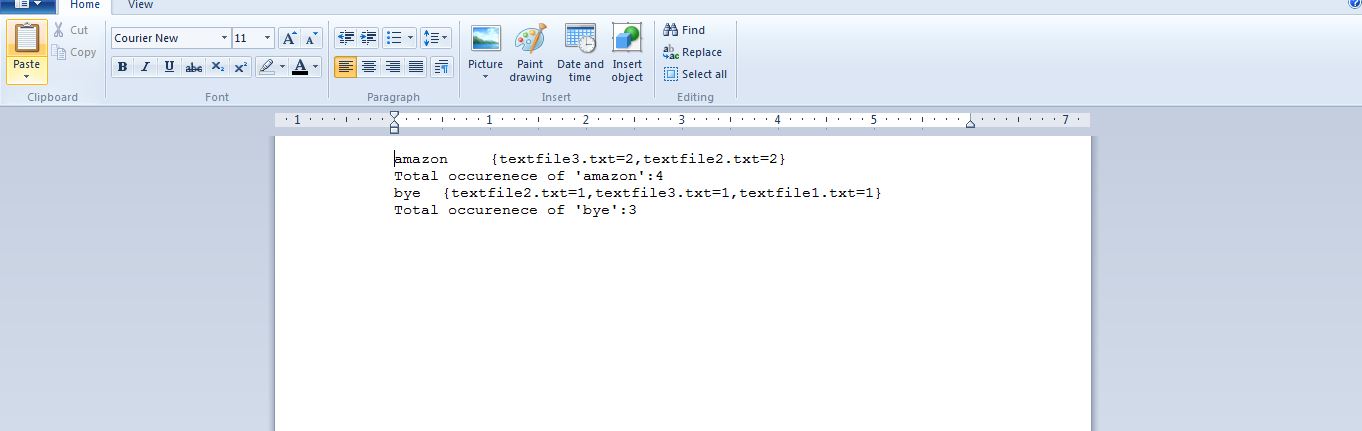
**Assignment 4**

****

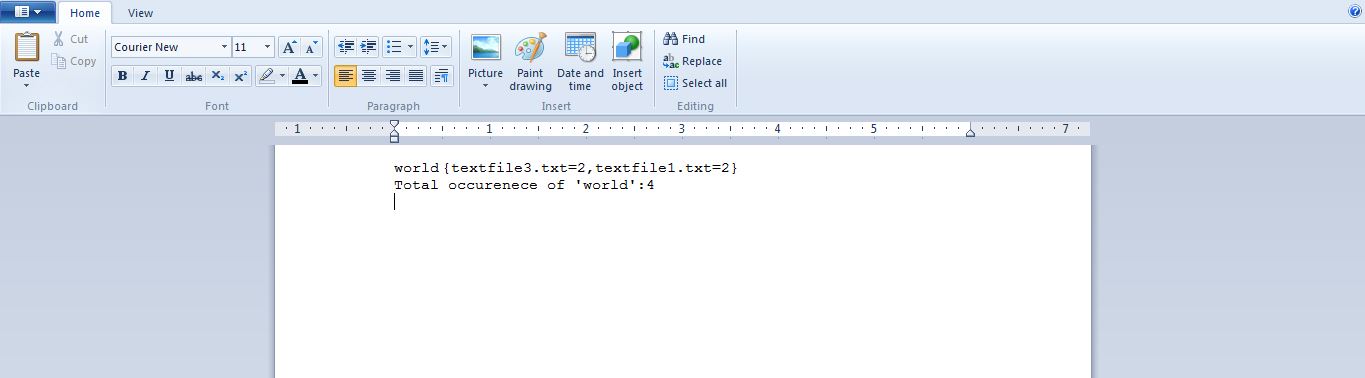
**Fig 1:** Clusture Details



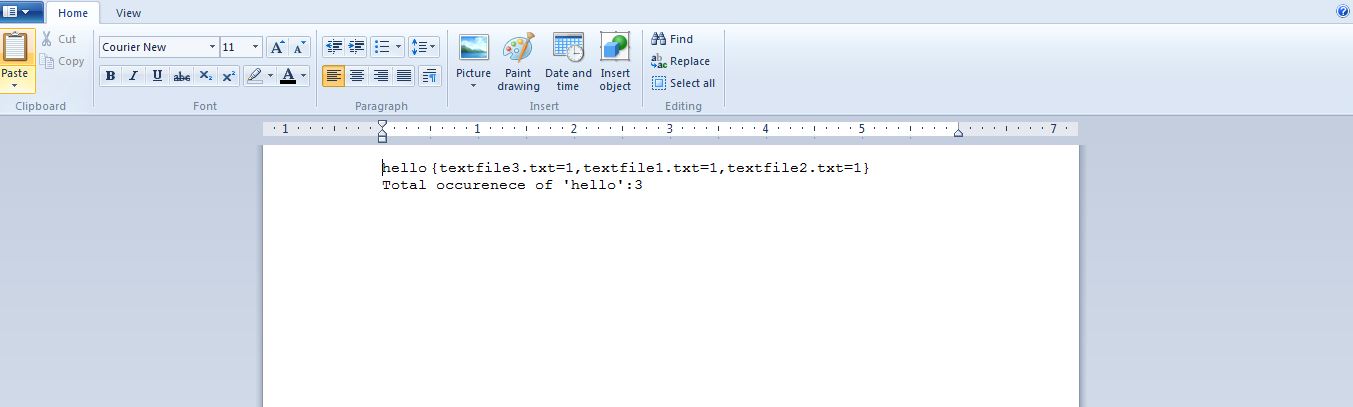
**Fig 2:** Step page to upload jar file and define input and output paths.



**Fig 3:** Displays the output of MapReduce.



**Fig 4:** Displays the output of MapReduce.



**Fig 5:** Displays the output of MapReduce.

**Map.java**

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapreduce.\*;

import org.apache.hadoop.mapreduce.lib.input.\*;

public class Map extends Mapper<LongWritable, Text, Text, Text> {

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

private Text word = new Text();

private String pattern= "^[a-z][a-z0-9]\*$";

public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

InputSplit splitInput = context.getInputSplit();

String fileName = ((FileSplit)splitInput).getPath().getName();

String line = value.toString();

StringTokenizer tokenizer = new StringTokenizer(line);

while (tokenizer.hasMoreTokens()) {

word.set(tokenizer.nextToken());

String stringWord = word.toString().toLowerCase();

if (stringWord.matches(pattern))

{

context.write(new Text(stringWord), new Text(fileName));

//context.write(new Text(stringWord), one);

}

}

}

}

**Reduce.java**

import java.io.IOException;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapreduce.\*;

public class Reduce extends Reducer<Text, Text, Text, Text> {

public void reduce(Text key, Iterable<Text> values, Context context)

throws IOException, InterruptedException {

/\*int sum = 0;

for (IntWritable val : values) {

sum += val.get();

}

context.write(key, new IntWritable(sum));\*/

int sum = 0;

int totalCount =0;

boolean bolCheck =false;

String fileName = "";

String strTxtFormat = "{";

for(Text value:values)

{

if(!bolCheck)

{

fileName = value.toString();

bolCheck= true;

}

if(fileName.equalsIgnoreCase(value.toString()))

{

sum = sum+1;

totalCount+=1;

}

else

{

strTxtFormat+= fileName +"="+ sum+",";

fileName = value.toString();

sum=1;

totalCount+=1;

}

//totalCount+=sum;

}

//totalCount+=sum;

strTxtFormat+= fileName +"="+ sum +"}\n";

context.write(key, new Text(strTxtFormat+"Total occurenece of "+"'"+key+"'"+":"+totalCount));

//context.write(key, new Text("The occurrence of"+key + "is:"+ totalCount));

//context.write("The occurrence of"+key+"is", new Text(":"+totalCount));

}

}

**WordCount.java**

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.\*;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapreduce.\*;

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

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

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

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

public class WordCount {

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

Configuration conf = new Configuration();

Job job = new Job(conf, "WordCount");

job.setJarByClass(WordCount.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

job.setNumReduceTasks(3);

job.setMapperClass(Map.class);

job.setReducerClass(Reduce.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

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

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

job.waitForCompletion(true);

}

}