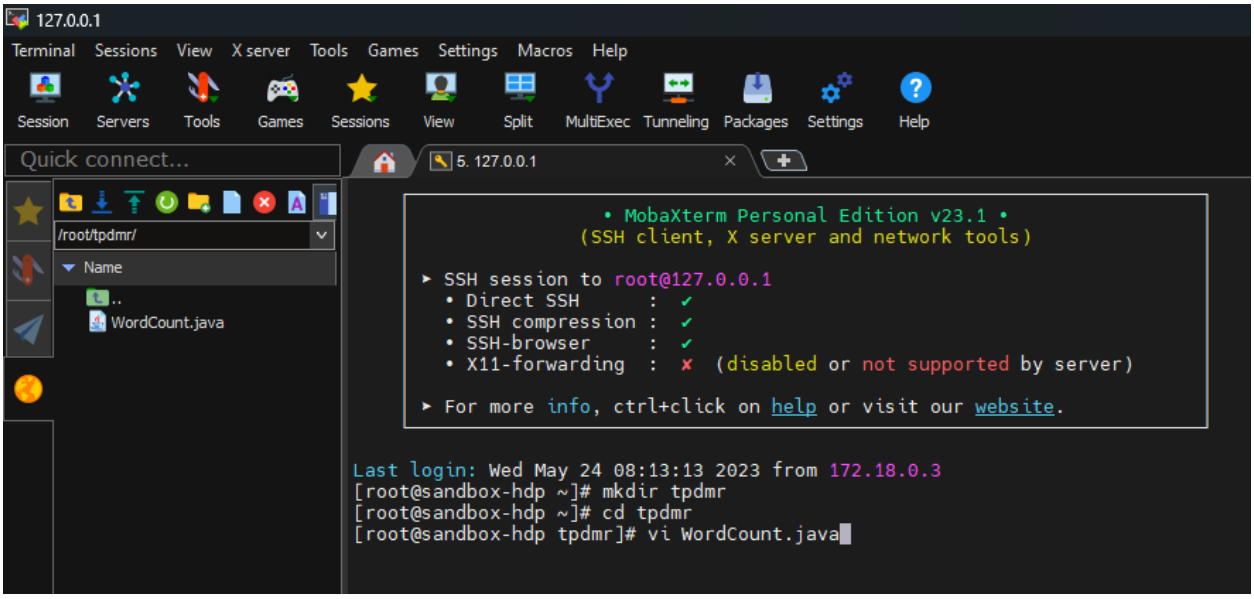
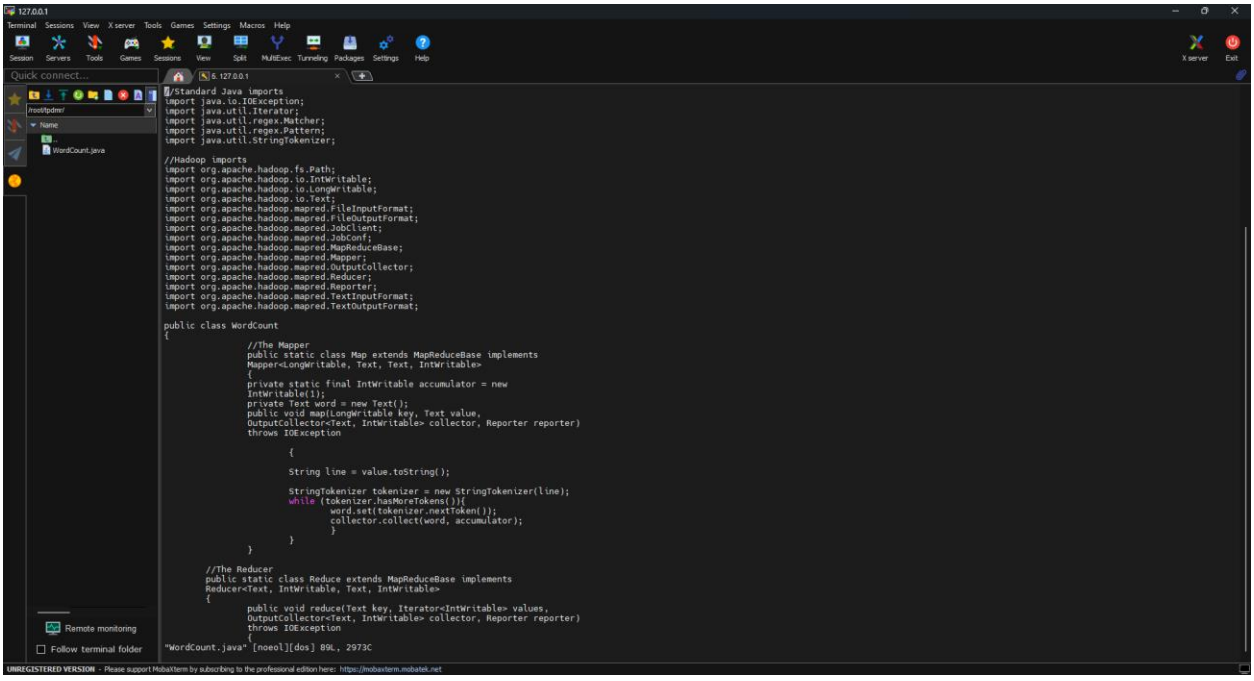
Nama : Komang Niko Romano Prodi
NIM : 222011356
Kelas : 3SD1
No Absen : 06

Praktikum 4 Teknologi Big Data – Distributed Processing

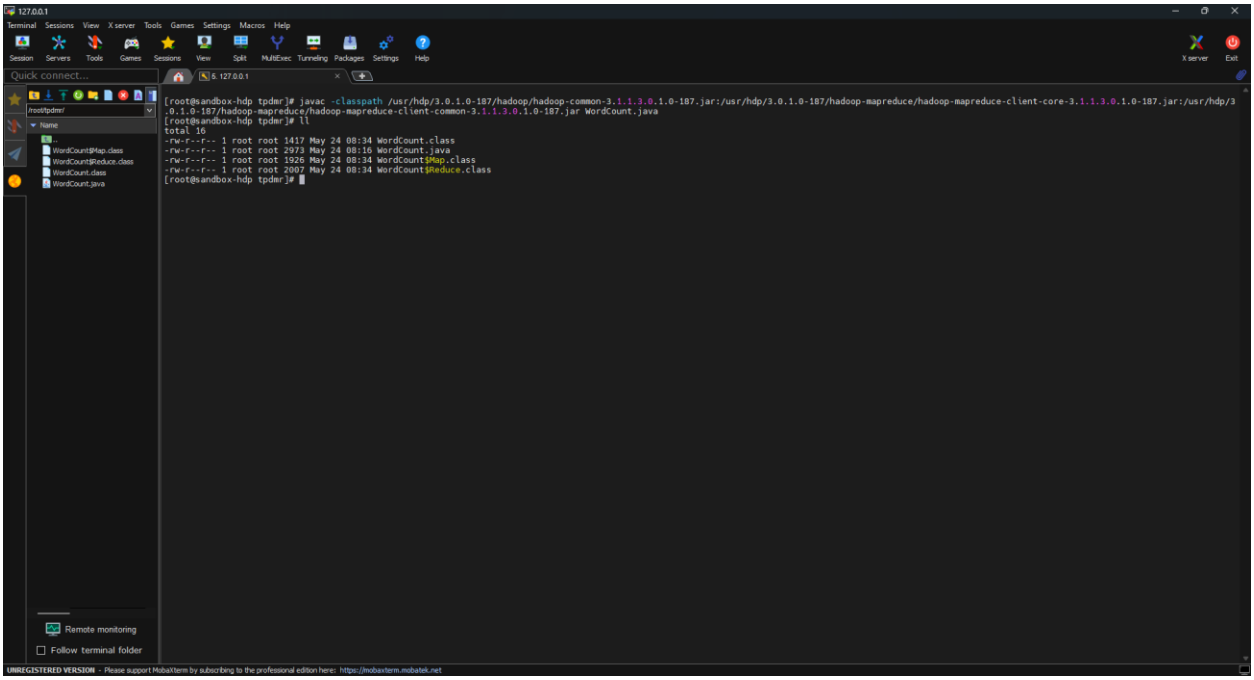
1. **Persiapan (membuat folder tpdmr beserta mengupload file)**



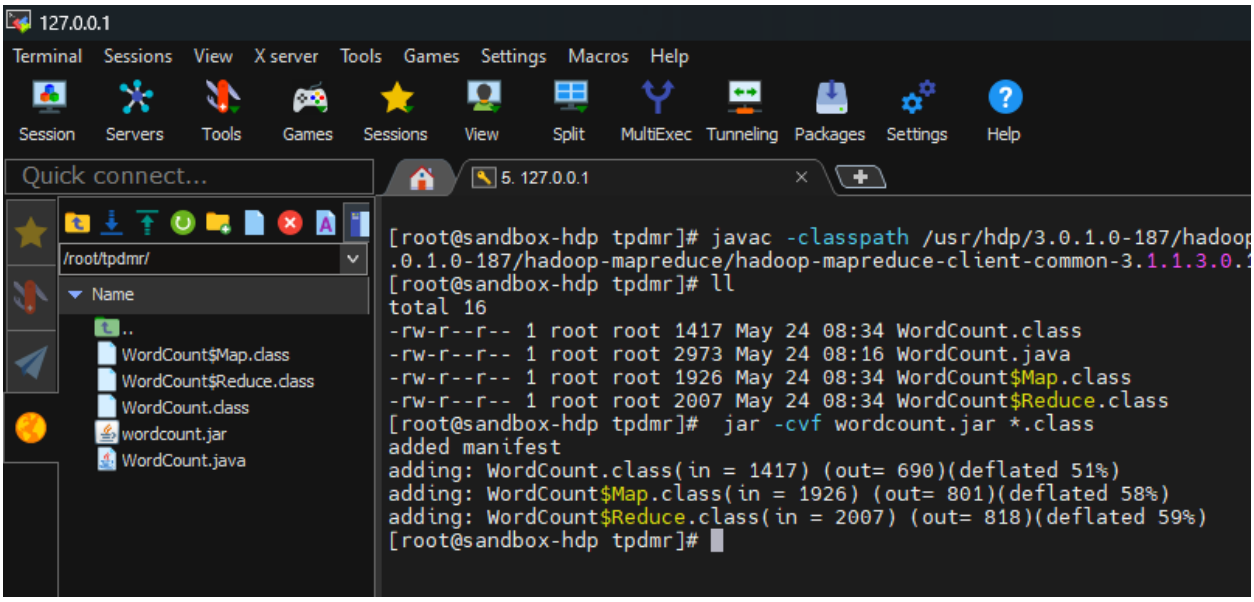
2. **View File**



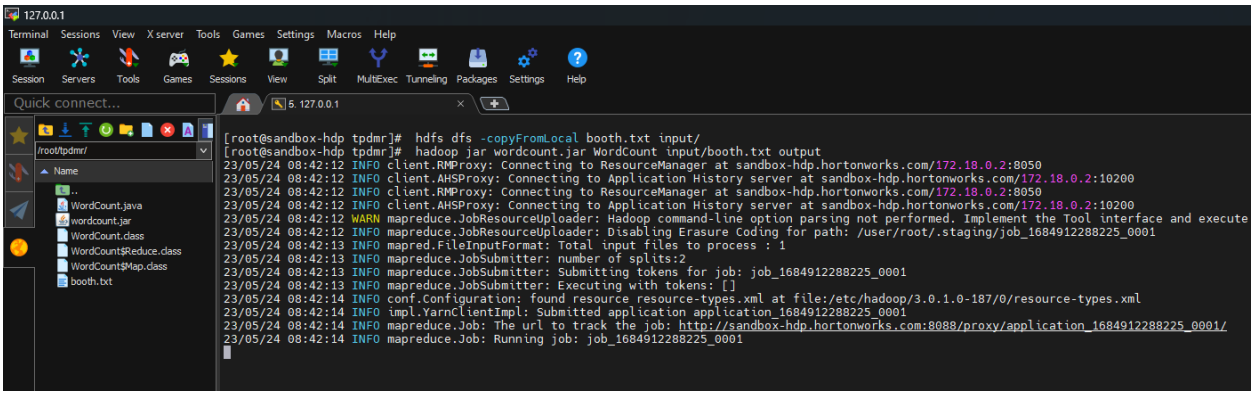
3. Compile Java File

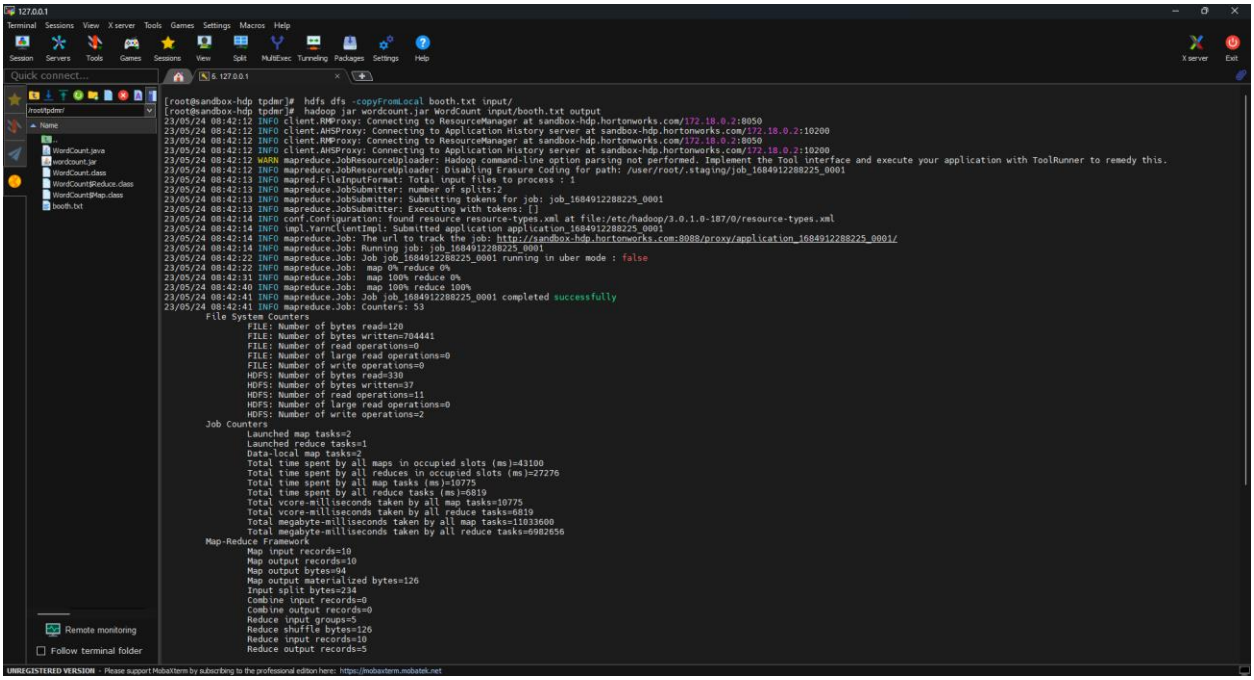


4. **Lihat file di dalam folder, kemudian buat jar file**

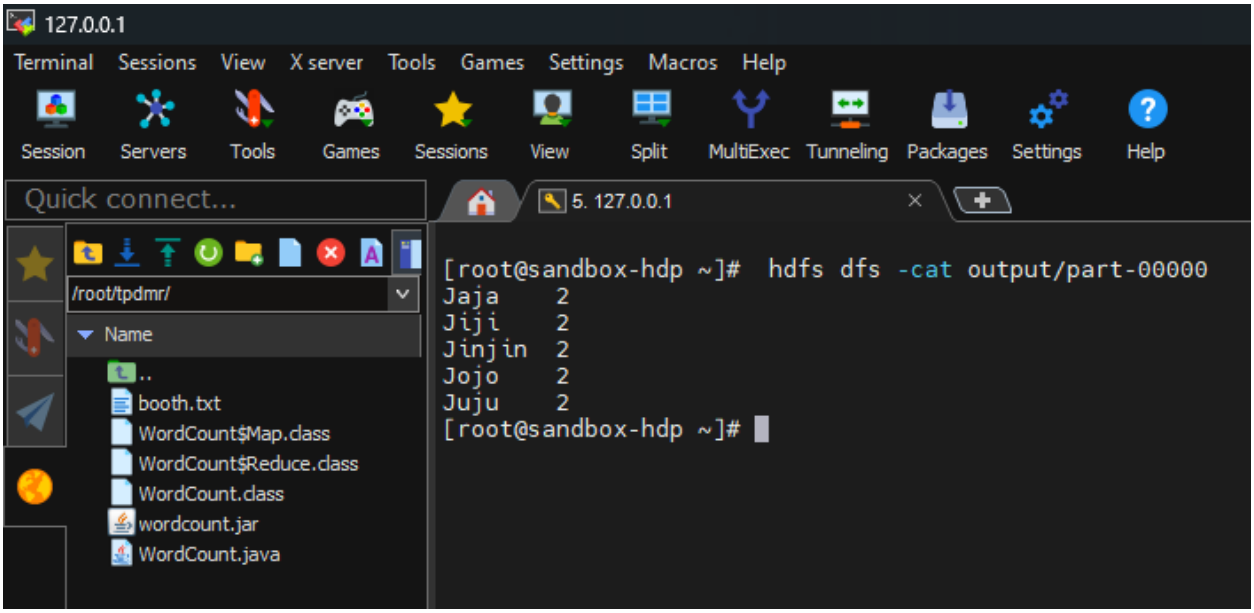


5. **Buat folder hdfs baru untuk menyimpan input, kemudian upload file input dan transfer ke hdfs**

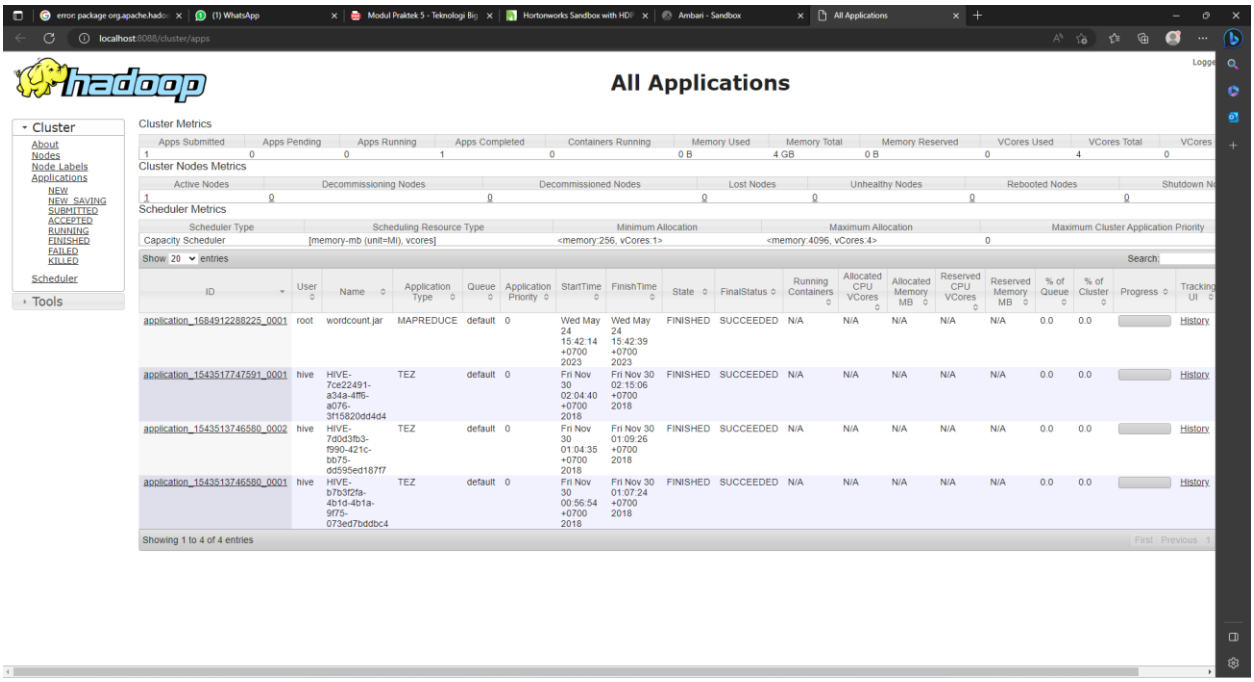




6. Baca File Output



7. Monitoring Jobs yang sedang berjalan di cluster



8. Penugasan

Pada penugasan menggunakan file salaryinfo.txt dan file MaxSalaryByCountry.java yaitu sebagai berikut:

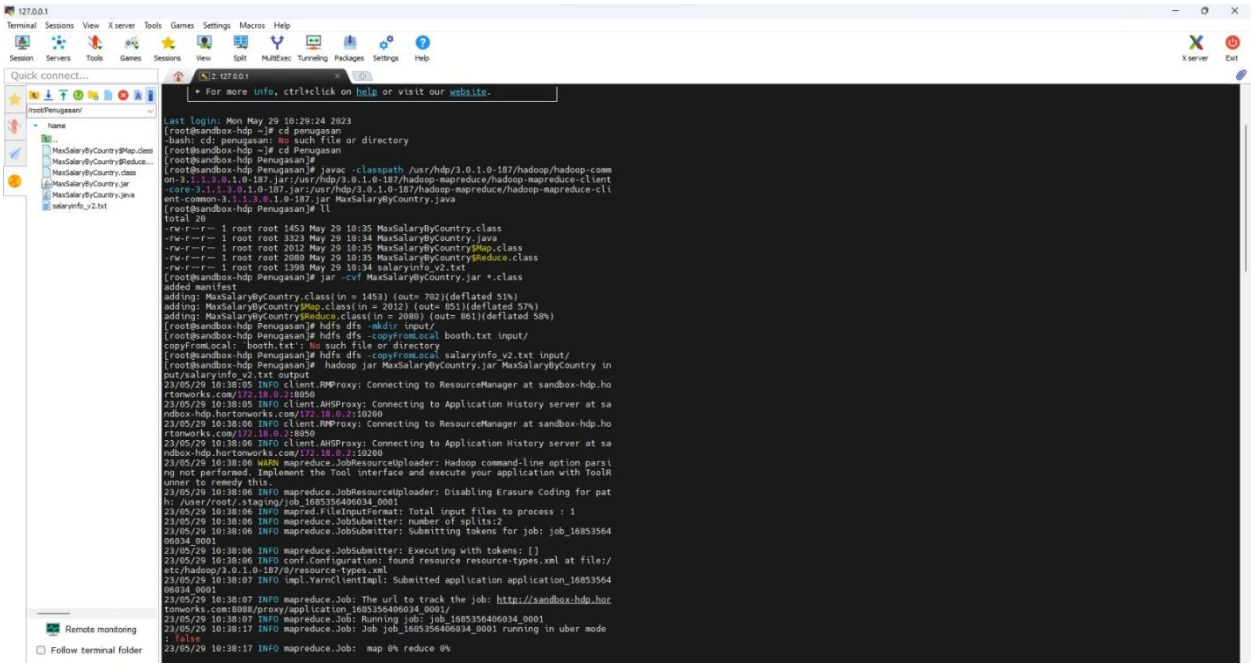
```
//Standard Java imports
import java.io.IOException;
import java.util.Iterator;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.util.StringTokenizer;

//Hadoop imports
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
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.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
```

```
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;


public class MaxSalaryByCountry
{
   //The Mapper
   public static class Map extends MapReduceBase implements
Mapper<LongWritable, Text, Text, IntWritable>
   {
      private Text word = new Text();

      public void map(LongWritable key, Text value,
OutputCollector<Text, IntWritable> collector, Reporter
reporter) throws IOException
      {
         String line = value.toString();
         StringTokenizer tokenizer = new
StringTokenizer(line);

         while (tokenizer.hasMoreTokens()){
            // 1st Step : kita misahin ini ininya
            StringTokenizer str = new
StringTokenizer(tokenizer.nextToken(), ",");
            String str0 = str.nextToken();
            String str1 = str.nextToken();
            String str2 = str.nextToken();
            // now you have 3 array , in which idx 0, 1, 2

            Text str11 = new Text(str1);
            IntWritable str22 = new
IntWritable(Integer.parseInt(str2));

            collector.collect(str11, str22);
         }
      }
   }

   //The Reducer
   public static class Reduce extends MapReduceBase implements
Reducer<Text, IntWritable, Text, IntWritable>
   {
```

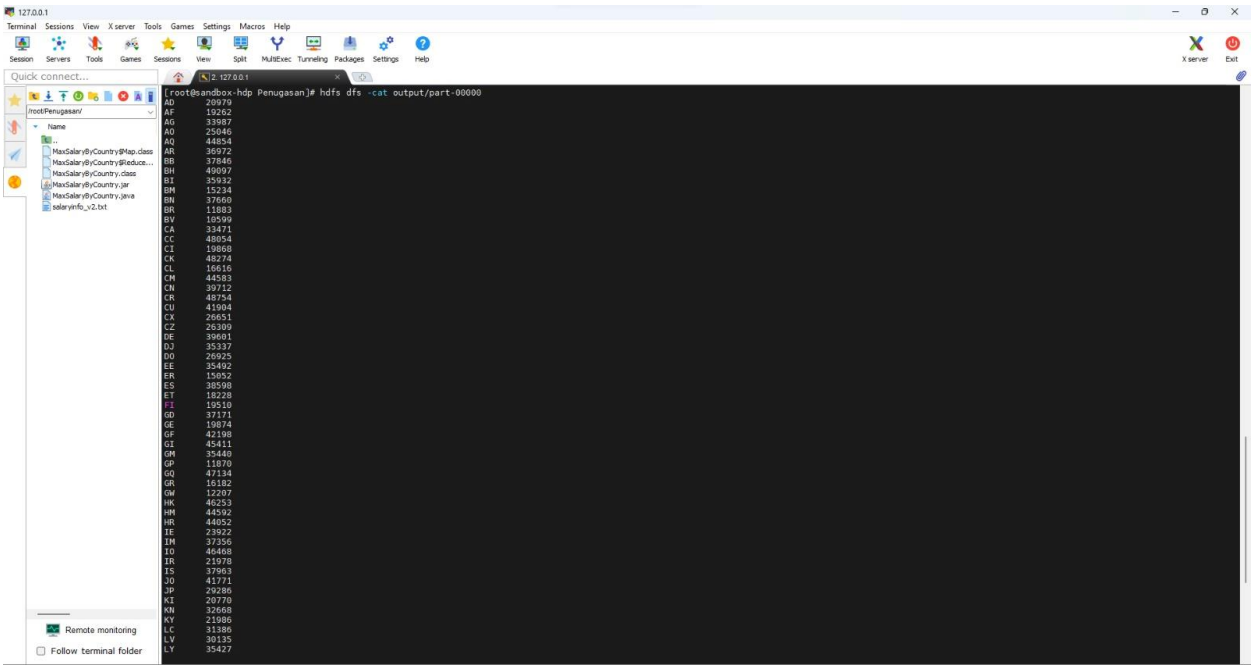
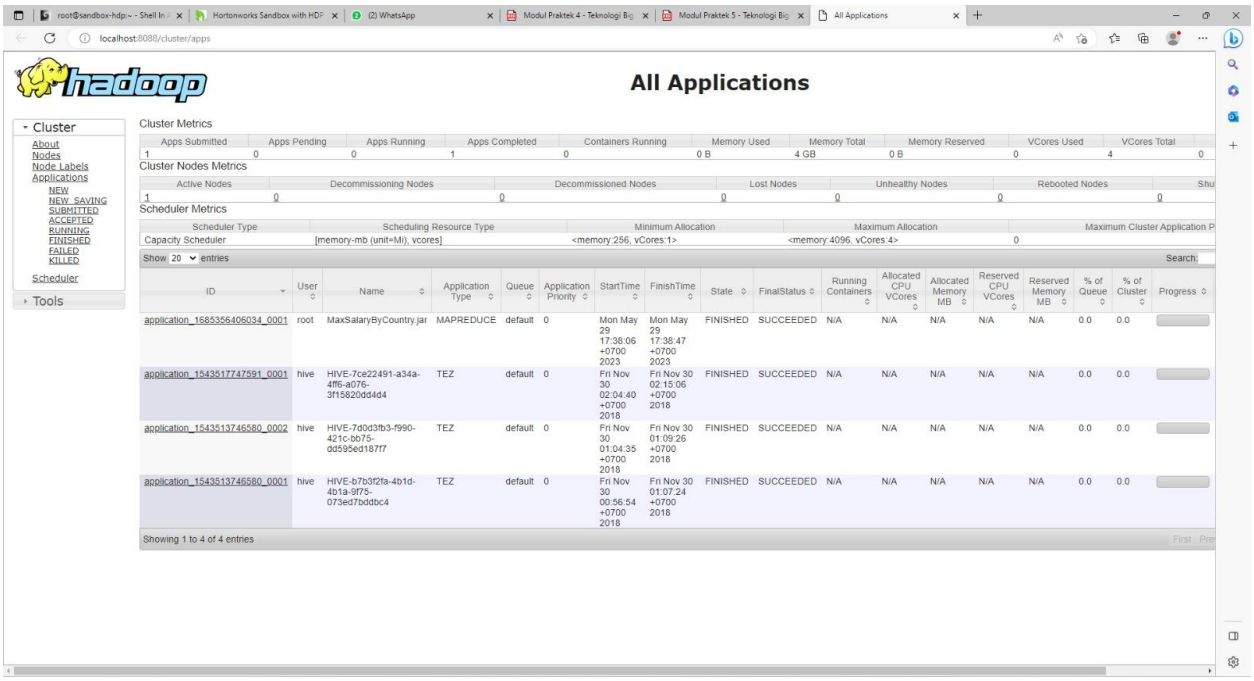
```
     public void reduce(Text key, Iterator<IntWritable>
values, OutputCollector<Text, IntWritable> collector, Reporter
reporter) throws IOException
     {
        int count = 0;

        //code to find the bring the horizon maximum the
occurrence
        while(values.hasNext())
        {
           // check if bigger, its better
           count = Math.max(count,values.next().get());
        }

        System.out.println(key + "\t" + count);
        collector.collect(key, new IntWritable(count));
     }
   }

   //The java main method to execute the MapReduce job
   public static void main(String[] args) throws Exception
   {
      //Code to create a new Job specifying the MapReduce class
      final JobConf conf = new
JobConf(MaxSalaryByCountry.class);
      conf.setOutputKeyClass(Text.class);
      conf.setOutputValueClass(IntWritable.class);
      conf.setMapperClass(Map.class);
      // Combiner is commented out – to be used in bonus
activity
      //conf.setCombinerClass(Reduce.class);
      conf.setReducerClass(Reduce.class);
      conf.setInputFormat(TextInputFormat.class);
      conf.setOutputFormat(TextOutputFormat.class);
      //File Input argument passed as a command line argument
      FileInputFormat.setInputPaths(conf, new Path(args[0]));
      //File Output argument passed as a command line argument
      FileOutputFormat.setOutputPath(conf, new Path(args[1]));
      //statement to execute the job
      JobClient.runJob(conf);
   }
}
```

Tahapan-tahapan yang dilakukan yaitu:

Membuat folder baru yaitu “penugasan”, lakukan upload file java dan file txt. Kemudian lakukan compile pada file java dan buat jar file.



Setelah itu baca output dari folder output



Lakukan monitoring jobs