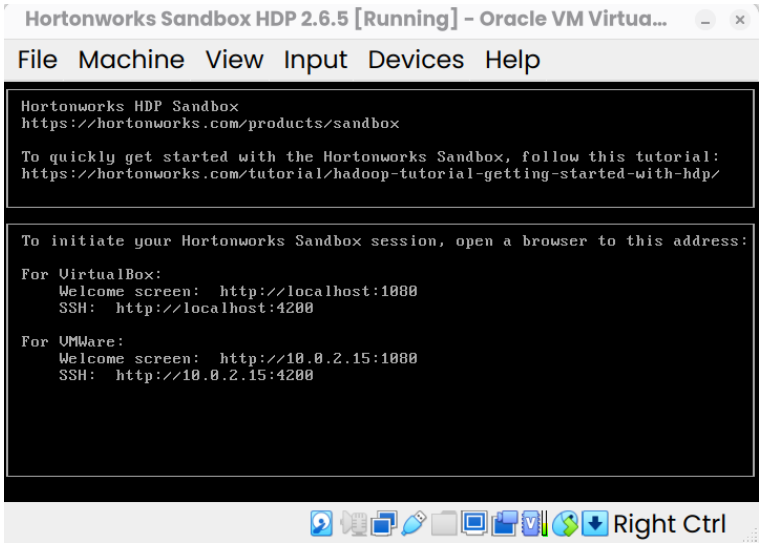
LAB 4

Aim: Run a java program (WordCount) on the concept of MapReduce

Procedure:

1. Start the HortonWorks it shows the IP address of machine.



9. Navigate to http://localhost:4200
10. Use the username and password: maria_dev to login.

11. Execute the following commands one-by-one:

● **Prepare Input Data**

echo "Hadoop is big data Hadoop is Java" > sample.txt
hdfs dfs -mkdir /input
hdfs dfs -put sample.txt /input/
hdfs dfs -ls /input

```
[maria_dev@sandbox-hdp ~]$ echo "Hadoop is big data Hadoop is Java" > sample.txt
[maria_dev@sandbox-hdp ~]$ hdfs dfs -mkdir /input
mkdir: `/input': File exists
[maria_dev@sandbox-hdp ~]$ hdfs dfs -put sample.txt /input/
[maria_dev@sandbox-hdp ~]$ hdfs dfs -ls /input
Found 1 items
-rw-r--r--   1 maria_dev hdfs         34 2025-09-16 04:34 /input/sample.txt
```

● **Create Java Files**

```
[maria_dev@sandbox-hdp ~]$ cat > WordMapper.java
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class WordMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

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

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

        String line = value.toString();

        // Split the line into words using whitespace as delimiter
        for (String token : line.split("\\s+")) {
            word.set(token);
            context.write(word, one);
        }
    }
}

^C

[maria_dev@sandbox-hdp ~]$ cat > WordReducer.java
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class WordReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    @Override
    public void reduce(Text key, Iterable<IntWritable> values, Context context)
            throws IOException, InterruptedException {

        int sum = 0;

        // Sum up all the counts for this word
        for (IntWritable val : values) {
            sum += val.get();
        }

        // Write the word and its total count to context
        context.write(key, new IntWritable(sum));
    }
}

^C

[maria_dev@sandbox-hdp ~]$ cat > WordCountDriver.java
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.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCountDriver {

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

        // Ensure input and output paths are provided
        if (args.length != 2) {
            System.err.println("Usage: WordCountDriver <input path> <output path>");
            System.exit(-1);
        }

        // Create Hadoop job configuration
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "word count");

        job.setJarByClass(WordCountDriver.class);

        // Set Mapper and Reducer classes
        job.setMapperClass(WordMapper.class);
        job.setReducerClass(WordReducer.class);

        // Set output key/value types
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        // Set input and output paths
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        // Exit after job completion
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}

^C
```

● **Compile Java Files**

javac -cp `hadoop classpath` -d wordcount_classes WordMapper.java WordReducer.java WordCountDriver.java

● **Create JAR**

jar -cvf wordcount.jar -C wordcount_classes/ .

```
[maria_dev@sandbox-hdp ~]$ javac -cp `hadoop classpath` -d wordcount_classes WordMapper.java WordReducer.java WordCountDriver.java
[maria_dev@sandbox-hdp ~]$ jar -cvf wordcount.jar -C wordcount_classes/ .
added manifest
adding: WordMapper.class(in = 1867) (out= 776)(deflated 58%)
adding: WordReducer.class(in = 1592) (out= 663)(deflated 58%)
adding: WordCountDriver.class(in = 1535) (out= 851)(deflated 44%)
```

● **Run MapReduceJob**

hdfs dfs -rm -r /output

hadoop jar wordcount.jar WordCountDriver /input /output

```
[maria_dev@sandbox-hdp ~]$ hadoop jar wordcount.jar WordCountDriver /input /output
25/09/16 04:06:06 INFO client.RMProxy: Connecting to ResourceManager at sandbox-hdp.hortonworks.com/172.18.0.2:8032
25/09/16 04:06:06 INFO client.AHSProxy: Connecting to Application History server at sandbox-hdp.hortonworks.com/172.18.0.2:10200
25/09/16 04:06:06 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
25/09/16 04:06:06 INFO input.FileInputFormat: Total input paths to process : 1
25/09/16 04:06:07 INFO mapreduce.JobSubmitter: number of splits:1
25/09/16 04:06:07 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1757994539875_0001
25/09/16 04:06:08 INFO impl.YarnClientImpl: Submitted application application_1757994539875_0001
25/09/16 04:06:08 INFO mapreduce.Job: The url to track the job: http://sandbox-hdp.hortonworks.com:8088/proxy/application_1757994539875_0001/
25/09/16 04:06:08 INFO mapreduce.Job: Running job: job_1757994539875_0001
25/09/16 04:06:20 INFO mapreduce.Job: Job job_1757994539875_0001 running in uber mode : false
25/09/16 04:06:20 INFO mapreduce.Job:  map 0% reduce 0%
25/09/16 04:06:27 INFO mapreduce.Job:  map 100% reduce 0%
25/09/16 04:06:32 INFO mapreduce.Job:  map 100% reduce 100%
25/09/16 04:06:34 INFO mapreduce.Job: Job job_1757994539875_0001 completed successfully
25/09/16 04:06:35 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=82
                FILE: Number of bytes written=305979
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=155
                HDFS: Number of bytes written=34
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
```

● **View Output**

hdfs dfs -ls /output

hdfs dfs -cat /output/part-r-00000

```
[maria_dev@sandbox-hdp ~]$ hdfs dfs -ls /output
Found 2 items
-rw-r--r--   1 maria_dev hdfs          0 2025-09-16 04:06 /output/_SUCCESS
-rw-r--r--   1 maria_dev hdfs         34 2025-09-16 04:06 /output/part-r-00000
[maria_dev@sandbox-hdp ~]$ hdfs dfs -cat /output/part-r-00000
Hadoop  2
Java    1
big     1
data    1
is      2
```

Result: MapReduce has been used to perform Word Count