

Implementasi MapReduce pada Program Penghitung Kata (Word Count) pada Ubuntu Distribution DigitalOcean.

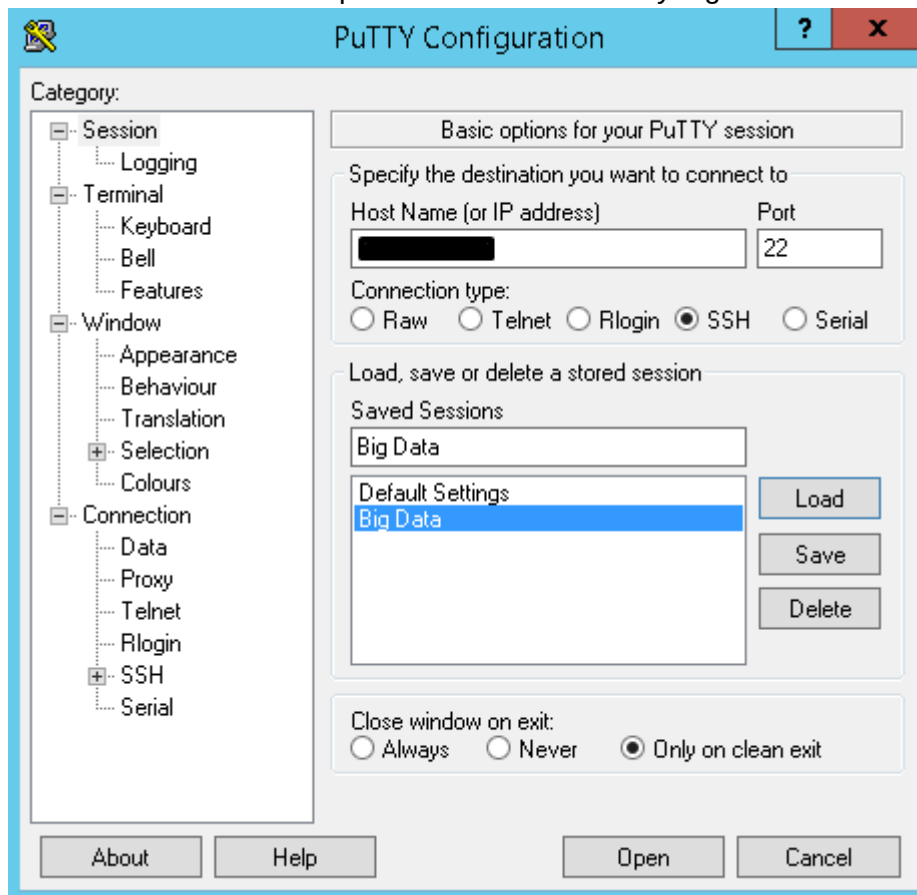
A. Instalasi Java dan Hadoop

Persiapan yang perlu dilakukan :

1. Linux Ubuntu 18.04 (DigitalOcean)
2. Java Development Kit (telah terinstal dan terkonfigurasi)
3. Apache Hadoop (telah terinstal dan terkonfigurasi)

B. Implementasi MapReduce

1. Akses terlebih dahulu droplet Linux Ubuntu 18.04 yang anda miliki terlebih dahulu.



2. Kemudian login sesuai dengan user dan password yang telah anda tetapkan ketika membuat droplet tersebut.

```
root@Jangandidestroy: ~  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
System information as of Tue Sep 29 15:34:51 UTC 2020  
  
System load:  0.0                Processes:            91  
Usage of /:   4.1% of 77.36GB    Users logged in:     0  
Memory usage: 4%                IP address for eth0: 139.59.244.89  
Swap usage:   0%  
  
* Canonical Livepatch is available for installation.  
- Reduce system reboots and improve kernel security. Activate at:  
  https://ubuntu.com/livepatch  
  
0 packages can be updated.  
0 updates are security updates.  
  
New release '20.04.1 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Tue Sep 29 15:08:00 2020 from 128.199.139.143  
root@Jangandidestroy:~#
```

3. Tuliskan command line ls untuk melihat isi direktori.

```
root@Jangandidestroy:~# ls  
Hitler-in-Central-America.txt  grep_example          input    masuk  
WordCount                    hadoop-2.10.1.tar.gz.sha512  jar  
WordCount.zip                 hadoop-3.1.1.tar.gz    keluar  
root@Jangandidestroy:~#
```

4. Kemudian buat dua buah folder yang bertujuan untuk meletakkan file input dan outputnya, dalam hal ini saya membuat folder dengan nama masuk dan keluar.

```
root@Jangandidestroy:~# mkdir masuk  
mkdir: cannot create directory 'masuk': File exists  
root@Jangandidestroy:~# mkdir keluar  
mkdir: cannot create directory 'keluar': File exists
```

5. Buat sebuah file yang berformat txt pada folder masuk, dalam hal ini saya membuat file dengan nama input.txt

```
root@Jangandidestroy:~# nano masuk/input.txt
```

6. Setelah file input telah ada, anda tinggal mengisi file tersebut sesuai dengan apa yang anda inginkan. Dalam hal ini saya mengisinya dengan sebuah buku novel.

```
root@Jangandidestry: ~
GNU nano 2.9.3      masuk/input.txt

Annas Al Farisi | 064001800001

HITLER IN CENTRAL AMERICA

A non-kosher story

Jacobso Schifter Sikora

1

INTRODUCTION

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify    ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell   ^_ Go To Line
```

7. Setelah melakukan edit pada file input.txt tekan Ctrl+X dilanjut dengan tekan Y lalu Enter untuk menyimpan file perubahan.
8. Kemudian buat beberapa folder yang akan diperlukan dalam pembuatan program Word Count. Sama seperti langkah awal gunakan command line mkdir [nama folder]. Berikut tampilan direktorinya:

```
root@Jangandidestry:~# ls WordCount/
WordCount.jar  bin  src
root@Jangandidestry:~# ls WordCount/bin/
'WordCount$Map.class' 'WordCount$Reduce.class'  WordCount.class
root@Jangandidestry:~# ls WordCount/src/
WordCount.java
root@Jangandidestry:~#
```

9. Pada tahap ini tuliskan perintah nano WordCount/src/WordCount.java untuk membuat sebuah file format java yang akan digunakan untuk membuat program.
10. Setelah menjalankan perintah nano WordCount/src/WordCount.java kemudian isi dengan source code berikut.

```
import java.io.IOException;
import java.util.*;

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 class Map extends 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,Context context) throws
IOException,
                InterruptedException{
                    String line = value.toString();
                    StringTokenizer tokenizer = new StringTokenizer(line);
                    while (tokenizer.hasMoreTokens()){
                        word.set(tokenizer.nextToken());
                        context.write(word,one);
                    }
                }
            }

        public static class Reduce extends Reducer<Text, IntWritable, Text,
IntWritable>{
            public void reduce(Text key, Iterable<IntWritable>values, Context context)

            throws IOException, InterruptedException{
                int sum = 0;
                for (IntWritable val : values){
                    sum += val.get();
                }
                context.write(key, new IntWritable(sum));
            }
        }

        public static void main (String[] args) throws Exception {
            Configuration conf = new Configuration();

            @SuppressWarnings("deprecation")
            Job job = new Job(conf, "wordcount");
            job.setJarByClass(WordCount.class);
            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);
            job.setMapperClass(Map.class);
            job.setReducerClass(Reduce.class);
            job.setInputFormatClass(TextInputFormat.class);
            job.setOutputFormatClass(TextOutputFormat.class);

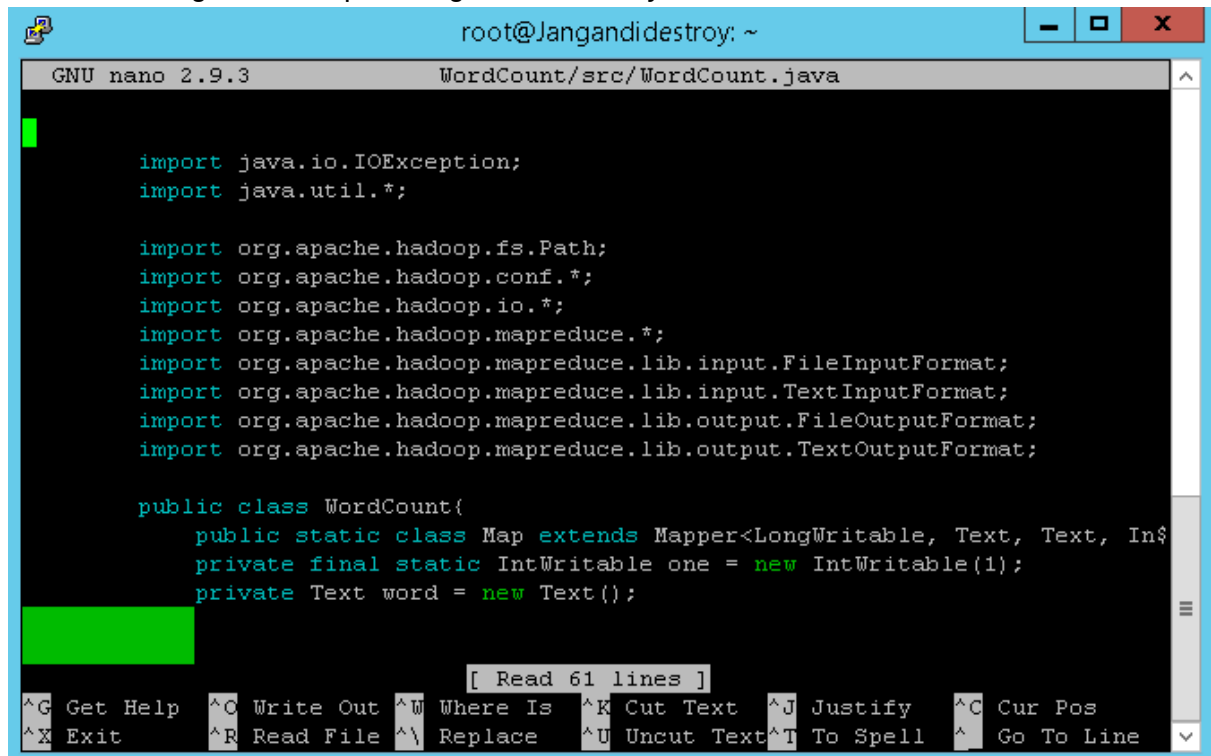
            FileInputFormat.addInputPath(job, new Path(args[0]));

```

```
FileOutputStream.setOutputPath(job, new Path(args[1]));
job.waitForCompletion(true);
```

```
}
}
```

11. Lalu save dengan cara seperti langkah sebelumnya.



```
root@Jangandideestroy: ~
GNU nano 2.9.3 WordCount/src/WordCount.java

import java.io.IOException;
import java.util.*;

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 class Map extends Mapper<LongWritable, Text, Text, In$
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

[ Read 61 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace  ^U Uncut Text ^T To Spell ^_ Go To Line
```

12. Setelah membuat file WordCount.java, selanjutnya masuk ke direktori folder WordCount dengan perintah dibawah dan tuliskan perintah chmod -R 777 bin untuk mengubah permission dari folder bin yang ada di folder WordCount.

```
root@Jangandideestroy:~# cd WordCount
root@Jangandideestroy:~/WordCount# chmod -R 777 bin
```

13. Lakukan perintah cd ke src dan tuliskan perintah javac -classpath \$HADOOP_CLASSPATH -d bin/ WordCount.java

```
root@Jangandideestroy:~/WordCount# cd src
root@Jangandideestroy:~/WordCount/src# javac -classpath $HADOOP_CLASSPATH -d bin/
WordCount.java
```

14. Lihat hasil compiler pada folder bin, tapi tuliskan perintah cd untuk kembali ke direktori awal dan dilanjut dengan perintah ls WordCount/bin/

```
root@Jangandideestroy:~/WordCount/src# cd
root@Jangandideestroy:~# ls WordCount/bin/
'WordCount$Map.class' 'WordCount$Reduce.class' WordCount.class
root@Jangandideestroy:~#
```

15. Ketik perintah cd dan masuk ke direktori WordCount lalu jalankan perintah berikut untuk mengubah ke file executable format jar.

```
root@Jangandideestroy:~# cd WordCount/
root@Jangandideestroy:~/WordCount# jar -cvf WordCount.jar -C bin/ .
```

16. Kemudian tuliskan perintah tersebut untuk menjalankan program WordCount.

```
root@Jangandideestroy:~# /usr/local/hadoop/bin/hadoop jar ~/WordCount/WordCount.jar WordCount ~/masuk/input.txt ~/keluar/output_3
```

17. Jika berhasil maka akan seperti ini tampilannya.

```
20/09/29 16:37:39 INFO mapred.LocalJobRunner: Finishing task: attempt_local49983
9054_0001_r_000000_0
20/09/29 16:37:39 INFO mapred.LocalJobRunner: reduce task executor complete.
20/09/29 16:37:39 INFO mapreduce.Job: map 100% reduce 100%
20/09/29 16:37:39 INFO mapreduce.Job: Job job_local499839054_0001 completed succ
essfully
20/09/29 16:37:39 INFO mapreduce.Job: Counters: 30
    File System Counters
        FILE: Number of bytes read=4931584
        FILE: Number of bytes written=6117688
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
    Map-Reduce Framework
        Map input records=10526
        Map output records=139944
        Map output bytes=1362045
        Map output materialized bytes=1641939
        Input split bytes=91
        Combine input records=0
        Combine output records=0
        Reduce input groups=20250
        Reduce shuffle bytes=1641939
        Reduce input records=139944
        Reduce output records=20250
        Spilled Records=279888
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=23
        Total committed heap usage (bytes)=471859200
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=820338
    File Output Format Counters
        Bytes Written=216669
```

18. Untuk melihat file hasil MapReduce, ketik perintah berikut.

```
root@Jangandideestroy:~# ls
Hitler-in-Central-America.txt  grep_example      input  masuk
WordCount                    hadoop-2.10.1.tar.gz.sha512  jar
WordCount.zip                hadoop-3.1.1.tar.gz      keluar
root@Jangandideestroy:~# ls keluar/output_3
_SUCCESS part-r-000000
```

19. Hasil komputasi dari algoritme MapReduce disimpan di file part-r-000000, untuk melihat isinya ketik perintah cat keluar/output_3/part-r-000000 dan enter. Maka akan muncul jumlah kata dari tulisan yang kita masukan dalam file input.txt(bersifat case sensitive).

```
-old 1
-only 1
-opposites 1
-other 1
-paralysis□ 1
-pay 1
-prices 1
-project□ 1
-quite 1
-redistribution.□ 1
-remember 1
-restaurants□ 1
-righteous,□ 1
-shower□ 1
-so 2
-some 1
-special□ 1
-that 4
-the 10
-theatrical 1
-then 1
-there 2
-this 1
-unable 1
-under 1
-until 1
-was 2
-we 3
-we're 1
-when 1
-where 1
-while 1
-why 2
-wise.□ 1
-you 2
-you've 1
-your 1
□ 2
"Don't 1
"Please, 1
"You 1
"a 1
.. 2
root@Jangandideestroy:~# cat keluar/output_3/part-r-00000
```