

ECC Lab-01

Name: Dev Jatinbhai Patel

Vraj Parekh

1.Create a cluster of 1 tiny instance from the virtual image called “E516 Ubuntu22 UFW Enabled”.

The screenshot shows the Jetstream2 interface for creating a new instance. The instance name is set to 'lab1'. The image selected is 'E516_Ubuntu22_UFW_Enabled'. Under the 'Flavor' section, the 'General-purpose' tab is selected, showing options for m3.x series. The 'm3.tiny' flavor is chosen, which has 1 CPU, 3 GB RAM, 20 GB Root Disk, and no ephemeral disk. Under the 'GPU' section, the 'g3.x' series is selected, with 'g3.small' chosen, which has 4 CPUs, 15 GB RAM, 60 GB Root Disk, and no ephemeral disk. The 'Create' button is visible at the bottom right.

The screenshot shows the Jetstream2 dashboard for the instance 'Lab1'. The instance status is 'Ready'. Key details include: created 5 days ago by user vparekh1@access-ci.org, from image E516_Ubuntu22_UFW_Enabled, flavor m3.small, and burn rate 2.00 SUs/hour. The 'Resource Usage' section displays three line graphs: CPU usage (of 2 total cores), RAM usage (of 6 total GB), and Root Disk usage (of 20 total GB). The CPU and RAM graphs show minimal usage, while the Root Disk graph shows a sharp increase starting around 9:53. The 'Interactions' section lists available connections: Web Shell, Web Desktop, Native SSH (to exouser@149.165.175.131), and Console. The 'Credentials' section shows the Public IP Address (149.165.175.131), Username (exouser), Passphrase (hidden), and SSH Public Key Name (lab1). The 'Volumes' section indicates no volumes are attached.

2.Configure the instance by the following steps:

a) Check if your UFW is working or not (super important!).

E.g., `sudo ufw status verbose.`

(see <https://linuxize.com/post/how-to-setup-a-firewall-with-ufw-on-ubuntu-20-04/>)

Warning: Disabling firewall or allowing unknown IP addresses to access your cluster will

lead to a 5-10 points deduction (Jetstream2 has a security monitor program checking all

instances' firewall constantly). In addition, the Jetstream2

Administrators will delete all

your instances.

b) Configure SSH logins using SSH key-pair so that your instances will not ask for pass-words

Answer 2a.

```
[vrajparekh@Vrajs-MacBook-Pro-3 ~ % ssh exouser@149.165.175.131

System information as of Fri Feb 14 03:06:44 UTC 2025

System load: 1.91          Processes:           258
Usage of /: 94.7% of 19.20GB  Users logged in:      1
Memory usage: 23%          IPv4 address for ens3: 10.3.34.194
Swap usage:  0%

=> / is using 94.7% of 19.20GB

===== https://jetstream.status.io/ =====

Overall Jetstream2 Status: Operational

=====

Last login: Fri Feb 14 02:57:50 2025 from 68.58.34.226
exouser@lab1:~$ ]
```

After this, we run the command to authorise the PC so that it can get through firewall.

```
vrajparekh — exouser@lab1: ~ — ssh exouser@149.165.175.131 — 94x26
=> / is using 94.7% of 19.20GB
https://jetstream.status.io/
Overall Jetstream2 Status: Operational

Last login: Fri Feb 14 02:57:50 2025 from 68.58.34.226
[exouser@lab1:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

To           Action      From
--           ----       ---
22/tcp (OpenSSH) ALLOW IN  Anywhere
22/tcp        ALLOW IN  Anywhere
Anywhere      ALLOW IN  68.50.8.118
22/tcp (OpenSSH (v6)) ALLOW IN  Anywhere (v6)
22/tcp (v6)   ALLOW IN  Anywhere (v6)

exouser@lab1:~$
```

2b.

Then, we generated the ssh keys and below is generated key which will enable us to log-in without credentials, the snapshot of which is attached below:

```
vrajparekh — exouser@lab1: ~ — -zsh — 94x26
[vrajparekh@Vrajs-MacBook-Pro-3 ~ % cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQACQC0173fW0bEJLmptMGtwnniPhlUXAuLGtBBpZ/GAmG+cWs2IMPoSGkV5A
sj2sUg4sYFmLBuew2RhHYfPwa4pfj9r0+mFRZ3g1LyksAG1TGqgYSV7s/gg+ebN5h1AI304P4FjgcXDsgtdeh9hSudQL
ekZxLbTyPf/D+3ArMJBw3rqtJOffzcuVjXEW3kqqN7ghXHFEUvrFo5l9RfcgjMXeWi/2oI+2oI02lzMYYitixE2h0F1r5s
FPSEDCMi5hDKwGR04le02ikPB2024iQl2ohYvaxg5dgKwfHQ1p5IEBEbjAddtKQaaETdgKS3p24eGqdVLCKSj3Drx2g1VW
JTKSETcEYgxeAIovDnf03DYKeguFW9QrY5rQQLSdpeG10PNw09SW1yc1cuIf3dzL4oLPc1Z6tHKoG5AltehMuo51PZ24Qz
aLedJFTchJ8NfziwmQhoQjNQffgO2j0E95UuVjLab/06qFb2jCK0gLxZQMfR/L0t0W3njwCHzY2jN1v8mUhLqJxtCwasr8
9DhF7zzjCan80V7Quq9NZiXQJr4v11IVQ/JI5/B6jzCk8vDAGEb97d7NUqEmXu6cecu34XVGLa1omPncb6ppCXu63DHqWx
YquV6agbpqhaMk1nFDIy1kQBH4gJ35pUYND+fRLOn0ueA5KRVs9GUewAfeWm8klQ== vraparekh@iu.edu
[vrajparekh@Vrajs-MacBook-Pro-3 ~ %
[vrajparekh@Vrajs-MacBook-Pro-3 ~ %
[vrajparekh@Vrajs-MacBook-Pro-3 ~ % ssh-copy-id exouser@149.165.175.131
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/Users/vrajparekh/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the remote system.
(if you think this is a mistake, you may want to use -f option)

vrajparekh@Vrajs-MacBook-Pro-3 ~ %
```

```
vrajparekh — exouser@lab1: ~ — ssh exouser@149.165.175.131 — 94x26
[vrajparekh@Vrajs-MacBook-Pro-3 ~ % ssh exouser@149.165.175.131

System information as of Fri Feb 14 03:14:44 UTC 2025

System load: 1.86          Processes: 266
Usage of /: 95.1% of 19.20GB  Users logged in: 1
Memory usage: 24%          IPv4 address for ens3: 10.3.34.194
Swap usage: 0%

=> / is using 95.1% of 19.20GB
===== https://jetstream.status.io =====

Overall Jetstream2 Status: Operational

=====

Last login: Fri Feb 14 03:13:44 2025 from 68.58.34.226
exouser@lab1:~$
```

3. Install Apache Hadoop, and test if your HDFS is working well or not. Test commands like

hdfs dfs -mkdir. Please refer to
<https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html> to see more hdfs
dfs commands

We were having the Java already installed, so that we directly created hadoop user through this command sudo adduser hadoop

```
vrajparekh — hadoop@lab1: ~ — ssh exouser@149.165.175.131 — 95x23
[exouser@lab1:~$ java -version
openjdk version "11.0.26" 2025-01-21
OpenJDK Runtime Environment (build 11.0.26+4-post-Ubuntu-1ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.26+4-post-Ubuntu-1ubuntu122.04, mixed mode, sharing)
[exouser@lab1:~$ 
[exouser@lab1:~$ 
[exouser@lab1:~$ su - hadoop
[Password:
hadoop@lab1:~$ ]
```

Then we configured the ssh keys to bypass authentication.

```
vrajparekh — hadoop@lab1: ~ — ssh exouser@149.165.175.131 — 95x23
[hadoop@lab1:~$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQgQDNJ73aWws36RdzfSGNEKVMNgQTSgmiWcF41V341MpimxbRumxQSvR
ssnTaeQz+QIvhJYExeOj3V+51l/hzVkgv2PiuE2MIsKo5vUqqp+dg9jneoP6mKLut10qfrEpQYtt4acwzNc53aWzMgfDSh
Mpe19TAAPgxm/EJR/dY74CzZPAYhg7uxL9go5hmQ0L5GEObNqxdcdUoOdcEBV0QFU+hUGcigKk//wmZW5oZ8ffrLshW9j
1cA5YB4PgoOS3ncCm28L6vWhtdTNpfwNckRbOsIpG+DbYHKS/wf70NRXU81kLEV0vdGRP+zSrcy56gSHNzbRN3eYdMUBBy2
fphP1K0u0vtW/j9jCTCkDrykoCawj517To4b0ZdcDHevjJ3gpqQIU1fZoJDrqgLN7kyeLuELPWvmGk1BpphkBexsz0LjSKG
T0df4JB8LW8d7Imv7Vid7LiNNNc2pSjGDMapmk5U07LZFEMtBvZVciIic/7EPmmPqVSMEeyNdEE= hadoop@lab1
hadoop@lab1:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
chmod 640 ~/.ssh/authorized_keys
[ ]
```

Then, we installed the hadoop in that user (named hadoop) using this command:

<https://dlcdn.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz>

```
tar -xvzf hadoop-3.3.6.tar.gz
```

```

vrajparekh — hadoop@lab1: ~ — ssh exouser@149.165.175.131 — 95x23
[hadoop@lab1:~$ hadoop version
Hadoop 3.3.6
Source code repository https://github.com/apache/hadoop.git -r 1be78238728da9266a4f88195058f08fd012bf9c
Compiled by ubuntu on 2023-06-18T08:22Z
Compiled on platform linux-x86_64
Compiled with protoc 3.7.1
From source with checksum 5652179ad55f76cb287d9c633bb53bbd
This command was run using /home/hadoop/hadoop/share/hadoop/common/hadoop-common-3.3.6.jar
[hadoop@lab1:~$ 
[hadoop@lab1:~$ 
[hadoop@lab1:~$ 
[hadoop@lab1:~$ ls
WordStandardDeviation.java  hadoop  hadoop-3.3.6.tar.gz  hadoopdata
hadoop@lab1:~$ 

```

4. Configure the MapReduce programming framework properly.

Answer:

After installing the hadoop, we made changes in the bashrc and env.sh files which are below:

```

GNU nano 6.2                               /home/hadoop/.bashrc
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_HOME=/home/hadoop/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HADOOP_YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="--Djava.library.path=$HADOOP_HOME/lib/native"

```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
 ^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line

```

GNU nano 6.2          /home/hadoop/hadoop/etc/hadoop/hadoop-env.sh
##
## {YARN_xyz|HDFS_xyz} > HADOOP_xyz > hard-coded defaults
##

# Many of the options here are built from the perspective that users
# may want to provide OVERWRITING values on the command line.
# For example:
#
#   JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
#
# Therefore, the vast majority (BUT NOT ALL!) of these defaults
# are configured for substitution and not append. If append
# is preferable, modify this file accordingly.

#####
# Generic settings for HADOOP
#####

Technically, the only required environment variable is JAVA_HOME.

^G Help      ^O Write Out    ^W Where Is     ^K Cut        ^T Execute     ^C Location
^X Exit      ^R Read File    ^\ Replace      ^U Paste      ^J Justify     ^/ Go To Line

```

After this, we configures hadoop by creating the namenode and datanode directory.

```

[hadoop@lab1:~$ cd hadoop
[hadoop@lab1:~/hadoop$ ls
LICENSE-binary      classes      lib          share
LICENSE.txt         etc          libexec      test.txt
NOTICE-binary       hadoop-3.3.6  licenses-binary wordcountj.jar
NOTICE.txt          hadoopdata   logs        wordstandarddeviation.jar
README.txt          include      org         wordstdddev.jar
WordStandardDeviation.java input      output
bin                 input.txt    sbin
[hadoop@lab1:~/hadoop$ cd hadoopdata
[hadoop@lab1:~/hadoop/hadoopdata$ ls
datanode  namenode
[hadoop@lab1:~/hadoop/hadoopdata/hdfs$ 

```

We also made changes in the core-site.xml which is shown below:

```

GNU nano 6.2          /home/hadoop/hadoop/etc/hadoop/core-site.xml
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<name>fs.defaultFS</name>
  <value>hdfs://lab1:9000</value>
</configuration>

```

Key Bindings:

- ^G Help
- ^O Write Out
- ^W Where Is
- ^K Cut
- ^T Execute
- ^C Location
- ^X Exit
- ^R Read File
- ^\\ Replace
- ^U Paste
- ^J Justify
- ^/ Go To Line

After completion of the changes in core-site, we made changes in hdfs-site.xml:

```

GNU nano 6.2          /home/hadoop/hadoop/etc/hadoop/hdfs-site.xml
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:///home/hadoop/hadoopdata/hdfs/namenode</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:///home/hadoop/hadoopdata/hdfs/datanode</value>
</property>
</configuration>

```

Key Bindings:

- ^G Help
- ^O Write Out
- ^W Where Is
- ^K Cut
- ^T Execute
- ^C Location
- ^X Exit
- ^R Read File
- ^\\ Replace
- ^U Paste
- ^J Justify
- ^/ Go To Line

Following that, we also changed yarn.xml and mapred.xml:

```
vrajparekh — hadoop@lab1: ~/hadoop/hadoopdata/hdfs — ssh exouser@149.165.175.131...
GNU nano 6.2 /home/hadoop/hadoop/etc/hadoop/mapred-site.xml

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
    <name>yarn.app.mapreduce.am.env</name>
    <value>HADOOP_MAPRED_HOME=$HADOOP_HOME/home/hadoop/hadoop/bin/hadoop</value>
</property>
<property>
    <name>mapreduce.map.env</name>
    <value>HADOOP_MAPRED_HOME=$HADOOP_HOME/home/hadoop/hadoop/bin/hadoop</value>
</property>
<property>
    <name>mapreduce.reduce.env</name>
    <value>HADOOP_MAPRED_HOME=$HADOOP_HOME/home/hadoop/hadoop/bin/hadoop</value>
</property>

</configuration>

```

**^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line**

```
vrajparekh — hadoop@lab1: ~/hadoop/hadoopdata/hdfs — ssh exouser@149.165.175.131...
GNU nano 6.2 /home/hadoop/hadoop/etc/hadoop/yarn-site.xml

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->
<configuration>

<!-- Site specific YARN configuration properties -->
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
</property>

</configuration>

```

**^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line**

Finally, we are starting the hadoop cluster:

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 95x23
2025-02-14 03:43:07,341 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1433288555-10.3.34
.194-1739504587324
2025-02-14 03:43:07,343 INFO common.Storage: Will remove files: [/home/hadoop/hadoopdata/hdfs/n
amenode/current/fsimage_00000000000000000000, /home/hadoop/hadoopdata/hdfs/namenode/current/fsim
age_00000000000000000000.md5, /home/hadoop/hadoopdata/hdfs/namenode/current/seen_txid, /home/had
oop/hadoopdata/hdfs/namenode/current/VERSION]
2025-02-14 03:43:07,378 INFO common.Storage: Storage directory /home/hadoop/hadoopdata/hdfs/nam
enode has been successfully formatted.
2025-02-14 03:43:07,442 INFO namenode.FSImageFormatProtobuf: Saving image file /home/hadoop/had
oopdata/hdfs/namenode/current/fsimage.ckpt_00000000000000000000 using no compression
2025-02-14 03:43:07,551 INFO namenode.FSImageFormatProtobuf: Image file /home/hadoop/hadoopdata
/hdfs/namenode/current/fsimage.ckpt_00000000000000000000 of size 401 bytes saved in 0 seconds .
2025-02-14 03:43:07,562 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with
txid >= 0
2025-02-14 03:43:07,605 INFO namenode.FSNamesystem: Stopping services started for active state
2025-02-14 03:43:07,605 INFO namenode.FSNamesystem: Stopping services started for standby state
2025-02-14 03:43:07,613 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when meet
shutdown.
2025-02-14 03:43:07,627 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****SHUTDOWN_MSG: Shutting down NameNode at lab1/10.3.34.194
*****/
hadoop@lab1:~/hadoop$
```

After that, we start the hadoop cluster and also checked all status of hadoop services:

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 95x23
[hadoop@lab1:~/hadoop$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [lab1]
Starting datanodes
Starting secondary namenodes [lab1]
Starting resourcemanager
Starting nodemanagers
[hadoop@lab1:~/hadoop$]
[hadoop@lab1:~/hadoop$]
[hadoop@lab1:~/hadoop$]
[hadoop@lab1:~/hadoop$]
[hadoop@lab1:~/hadoop$ jps
164661 Jps
163672 ResourceManager
163881 NodeManager
hadoop@lab1:~/hadoop$]
```

Atlast, we tested commands like mkdir to check the working status:

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 84x25
[hadoop@lab1:~/hadoop$ ls
LICENSE-binary         .hadoop-3.3.6      org
LICENSE.txt             .hadoopdata       output
NOTICE-binary           .include         sbin
NOTICE.txt              .input          share
README.txt              .input.txt        test.txt
WordStandardDeviation.java.lib           wordcountj.jar
bin                     .libexec        wordstandarddeviation.jar
classes                 .licenses-binary wordstddev.jar
etc                     .logs          logs
[hadoop@lab1:~/hadoop$ 
[hadoop@lab1:~/hadoop$ 
[hadoop@lab1:~/hadoop$ hdfs dfs -mkdir dev_vraj
[hadoop@lab1:~/hadoop$ ls
LICENSE-binary          etc             logs
LICENSE.txt             .hadoop-3.3.6      org
NOTICE-binary           .hadoopdata       output
NOTICE.txt              .include         sbin
README.txt              .input          share
WordStandardDeviation.java.input.txt      test.txt
bin                     .lib           wordcountj.jar
classes                 .libexec        wordstandarddeviation.jar
dev_vraj                .licenses-binary wordstddev.jar
[hadoop@lab1:~/hadoop$ ]
```

5 Compile and run the MapReduce example of WordStandardDeviation.java code.

To save your time to find the file, the example's source code is posted on Canvas now.

Also, to help you learn how to compile and create a jar file, please refer to <https://engineering.tomtom.com/create-java-jar/>.

We uploaded the file through sftp and below is the snapshot of it:

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 85x48
[hadoop@lab1:~/hadoop$ cat WordStandardDeviation.java
/**
 * Licensed to the Apache Software Foundation (ASF) under one
 * or more contributor license agreements. See the NOTICE file
 * distributed with this work for additional information
 * regarding copyright ownership. The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License. You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package org.apache.hadoop.examples;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.nio.charset.StandardCharsets;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class WordStandardDeviation extends Configured implements Tool {
    private double stddev = 0;

    private final static Text LENGTH = new Text("length");
    private final static Text SQUARE = new Text("square");
```

After that, we compiled the given .java file and then converted the created .class file into the .jar file using this command:

```
jar -cvf wordstandarddeviation.jar -C ~/hadoop/classes/
```

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 128x25
[hadoop@lab1:~/hadoop$ javac -classpath $(hadoop classpath) -d ~/hadoop/classes WordStandardDeviation.java
[hadoop@lab1:~/hadoop$ jar -cvf wordstandarddeviation.jar -C ~/hadoop/classes/ .
[added manifest
adding: org/(in = 0) (out= 0)(stored 0%)
adding: org/apache/(in = 0) (out= 0)(stored 0%)
adding: org/apache/hadoop/(in = 0) (out= 0)(stored 0%)
adding: org/apache/hadoop/examples/(in = 0) (out= 0)(stored 0%)
adding: org/apache/hadoop/examples/WordStandardDeviation$WordStandardDeviationReducer.class(in = 1882) (out= 775)(deflated 58%)
adding: org/apache/hadoop/examples/WordStandardDeviation$WordStandardDeviationMapper.class(in = 2066) (out= 926)(deflated 55%)
adding: org/apache/hadoop/examples/WordStandardDeviation.class(in = 4895) (out= 2423)(deflated 50%
[hadoop@lab1:~/hadoop$ jar -tf wordstandarddeviation.jar
META-INF/
META-INF/MANIFEST.MF
org/
org/apache/
org/apache/hadoop/
org/apache/hadoop/examples/
org/apache/hadoop/examples/WordStandardDeviation$WordStandardDeviationReducer.class
org/apache/hadoop/examples/WordStandardDeviation$WordStandardDeviationMapper.class
org/apache/hadoop/examples/WordStandardDeviation.class
[hadoop@lab1:~/hadoop$ ]
```

Then after, we created the input and output named directory using the command:

```
hdfs dfs -mkdir /input
hdfs dfs -mkdir /output
```

```
vrajparekh — hadoop@lab1: ~/hadoop/input — ssh exouser@149.165.175.131 — 128x25
[hadoop@lab1:~/hadoop$ ls
LICENSE-binary README.txt      dev_vraj      include      libexec      output      wordcountj.jar
LICENSE.txt      WordStandardDeviation.java  etc          input       licenses-binary sbin      wordstandarddeviation.jar
NOTICE-binary    bin           hadoop-3.3.6   input.txt   logs        share      wordstddev.jar
NOTICE.txt       classes       hadoopdata    lib         org        test.txt
[hadoop@lab1:~/hadoop$ [hadoop@lab1:~/hadoop$ [hadoop@lab1:~/hadoop$ cd input
[hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ ls
[hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ echo -e "hello world\nhello hadoop" > input.txt
[hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ [hadoop@lab1:~/hadoop/input$ cat input.txt
hello world
hello hadoop
[hadoop@lab1:~/hadoop/input$ ]
```

Then after, we run hadoop with mapreduce:

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 141x40
[hadoop@lab1:~/hadoop$ hadoop jar wordcount.jar org.apache.hadoop.examples.WordCount /input /output
2025-02-14 04:10:34,313 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2025-02-14 04:10:34,617 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2025-02-14 04:10:34,617 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2025-02-14 04:10:34,803 INFO mapreduce.JobSubmitter: Cleaning up the staging area file:/tmp/hadoop/mapred/staging/hadoop644837637/.staging/job_local1644837637_0001
Exception in thread "main" org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input path does not exist: file:/input
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:340)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.listStatus(FileInputFormat.java:279)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.getSplits(FileInputFormat.java:404)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeNewSplits(JobSubmitter.java:310)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeSplits(JobSubmitter.java:327)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:200)
        at org.apache.hadoop.mapreduce.Job.run(Job.java:1678)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1675)
        at java.base/java.security.AccessController.doPrivileged(Native Method)
        at java.base/javax.security.auth.Subject.doAs(Subject.java:423)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1899)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1675)
        at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1696)
        at org.apache.hadoop.examples.WordCount.main(WordCount.java:87)
        at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.base/java.lang.reflect.Method.invoke(Method.java:566)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:328)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:241)
Caused by: java.io.IOException: Input path does not exist: file:/input
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:313)
        ... 19 more
hadoop@lab1:~/hadoop$
```

Here, is the final output result:

```
vrajparekh — hadoop@lab1: ~/hadoop — ssh exouser@149.165.175.131 — 99x22
[hadoop@lab1:~/hadoop$ hdfs dfs -cat output/part-r-00000
hadoop 1
hello 2
world 1
hadoop@lab1:~/hadoop$
```