

Laboratory Assignment File
for

Big Data Analytics (CS0552)

Master of Technology
in
Computer Science & Engineering

Submitted by

Yugal
(Roll no. 25903053)

Submitted to

Dr. Shveta Mahajan
(Assistant Professor, NIT Jalandhar)



Department of Computer Science & Engineering
Dr. B R Ambedkar National Institute of Technology Jalandhar
Punjab, India-144008
June, 2026

Contents

Assignment 1: Install Apache Hadoop and Setup Single Node Cluster(27-01-2026)	1
1.1 Step 1: Installing Java	1
1.2 Step 2: Setup SSH	2
1.3 Step 3: Download and Configure Hadoop	3
1.4 Step 4: Environment Variable Configuration	4
1.5 Step 5: Hadoop XML Configuration Files	5
1.6 Step 6: Format NameNode & Start Hadoop Services	10
1.7 Step 7: HDFS Operations	11
1.8 Result	14
1.9 Conclusion	14
Assignment 2: Load large datasets into HDFS and analyze block distribution. (03-02-2026)	15
2.1 Step 1: Collect large Dataset (CSV/TXT)	15
2.2 Step 2: Create HDFS directory.	15
2.3 Step 3: Upload large dataset into HDFS	15
2.4 Step 4: Verify data storage	16
2.5 Step 5: Analyze block distribution	16
2.6 Step 6: Download processed dataset from HDFS to local system	16
2.7 Step 7: Delete data from HDFS	16
2.8 Step 8: Check file block information	17
2.9 Result	17
2.10 Conclusion	17
Assignment 3: Implement MapReduce programs for Data Processing. (10-02-2026)	18
3.1 Step 1: Start Hadoop Services	18
3.2 Step 2: Create Input Directory in HDFS	18
3.3 Step 3: Write MapReduce Program	19
3.4 Step 4: Compile Program and Create JAR	21
3.5 Step 5: Execute MapReduce Job	21
3.6 Step 6: Check Output Directory in HDFS	21
3.7 Step 7: Display Result File	22
3.8 Step 8: Modify Input and Re-run Job	22
3.9 Result	23
3.10 Conclusion	23
Assignment 4: Implement Data Processing using Spark and Compare with MapReduce. (17-02-2026)	24

4.1	Step 1: Start Hadoop Services	24
4.2	Step 2: Verify Input File in HDFS	24
4.3	Step 3: Start Spark Shell	25
4.4	Step 4: Load Data from HDFS into Spark	25
4.5	Step 5: Implement Spark Transformations	26
4.6	Step 6: Apply Actions and Display Output	26
4.7	Step 7: Save Output to HDFS	27
4.8	Step 8: Check Output Directory and Validate Result	27
4.9	Step 9: Stop Spark and Hadoop Services	27
4.10	Step 10: Comparison with MapReduce	28
4.11	Result	28
4.12	Conclusion	28

List of Figures

1	System Update Command	1
2	Java Installation	1
3	Java Version and Path Verification	2
4	Install SSH Server	2
5	Generate SSH Key Pair	3
6	Create Hadoop User	3
7	Download and Extract Hadoop	4
8	Moving Hadoop	4
9	.bashrc Modification	5
10	hadoop-env.sh Modification	6
11	core-site.xml Modification	7
12	hdfs-site.xml Modification	8
13	Creating Folder for NameNode and DataNode	8
14	mapred-site.xml Modification	9
15	yarn-site.xml Modification	9
16	Format NameNode	10
17	Start Hadoop Services	10
18	Start YARN Services	11
19	HDFS Operation 1	12
20	HDFS Operation 2	12
21	HDFS Result 1	13
22	HDFS Result 2	13
23	Start Hadoop Services and Verify with jps	15
24	Created exceldata directory in hdfs and copied dataset	15
25	Upload large dataset into HDFS	15
26	Verify data storage in HDFS	16
27	Analyze block distribution in HDFS	16
28	Download processed dataset from HDFS to local system	16
29	Delete data from HDFS	16
30	Check file block information in HDFS	17
31	Starting Hadoop services and verifying with jps	18
32	Creating input directory and uploading dataset	19
33	Mapper class implementation	19
34	Reducer class implementation	20
35	Driver class implementation	20
36	Compilation and JAR creation	21
37	Executing MapReduce job	21

38	Output directory created in HDFS	21
39	Displaying MapReduce output	22
40	Updating input text file	22
41	Re-running MapReduce with modified dataset	22
42	New output after re-running MapReduce job	23
43	Starting Hadoop services and verifying using jps	24
44	Verifying input dataset in HDFS	25
45	Launching Spark Shell	25
46	Reading input data from HDFS	26
47	Spark transformations for Word Count	26
48	Displaying results using collect()	26
49	Saving Spark output to HDFS	27
50	Validating Spark output in HDFS	27
51	Stopping Spark session and Hadoop services	28

List of Tables

1	Comparison between MapReduce and Spark	28
---	--	----

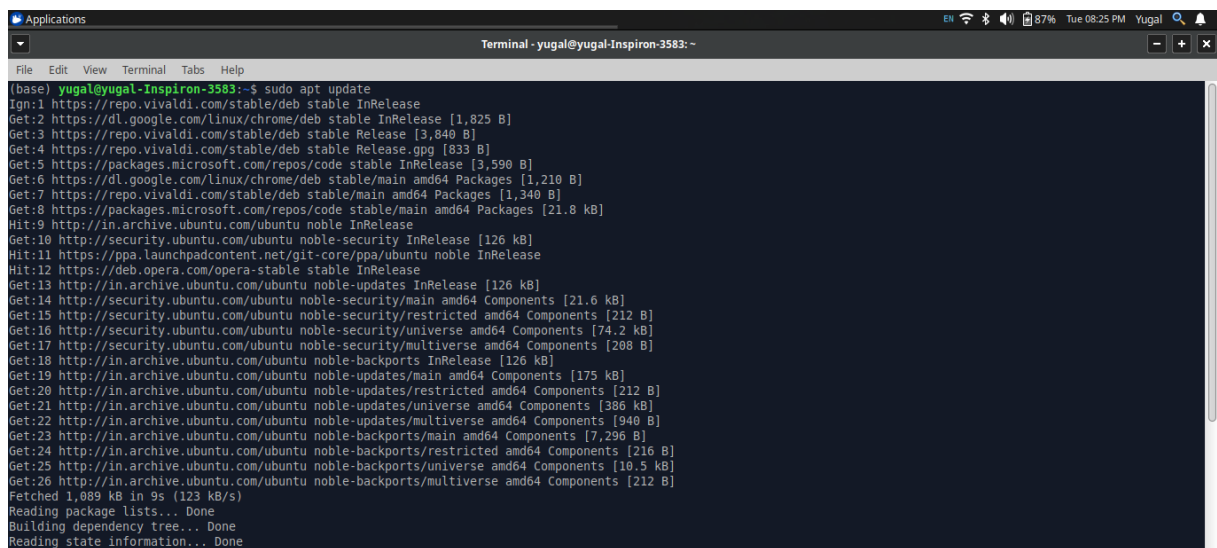
Assignment 1: Install Apache Hadoop and Setup Single Node Cluster

Objective: To install and configure Apache Hadoop in pseudo-distributed mode and perform basic HDFS operations such as upload, delete, replication check, and permission handling.

1.1 Step 1: Installing Java

Updating System

`sudo apt update`

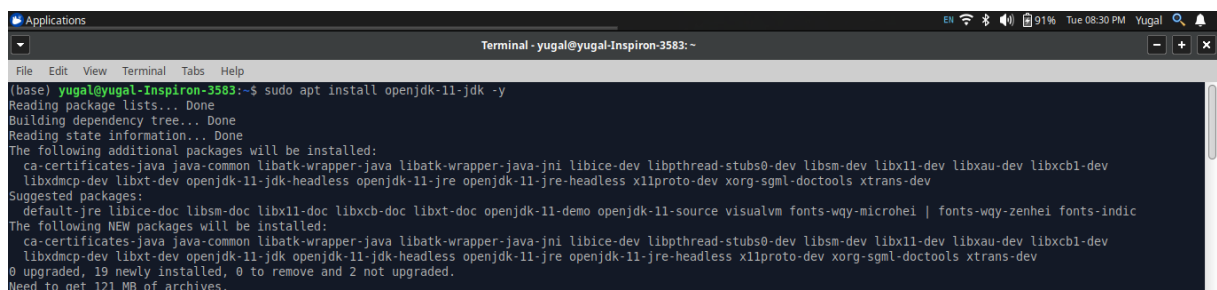


```
(base) yugal@yugal-Inspiron-3583:~$ sudo apt update
Ign:1 https://repo.vivaldi.com/stable/deb stable InRelease
Get:2 https://dl.google.com/linux/chrome/deb stable InRelease [1,825 B]
Get:3 https://repo.vivaldi.com/stable/deb stable Release [3,840 B]
Get:4 https://repo.vivaldi.com/stable/deb stable Release.gpg [833 B]
Get:5 https://packages.microsoft.com/repos/code stable InRelease [3,590 B]
Get:6 https://dl.google.com/linux/chrome/deb stable/main amd64 Packages [1,210 B]
Get:7 https://repo.vivaldi.com/stable/deb stable/main amd64 Packages [1,340 B]
Get:8 https://packages.microsoft.com/repos/code stable/main amd64 Packages [21.8 kB]
Hit:9 http://in.archive.ubuntu.com/ubuntu noble InRelease
Get:10 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:11 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu noble InRelease
Hit:12 https://deb.opera.com/opera-stable stable InRelease
Get:13 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [74.2 kB]
Get:17 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [280 B]
Get:18 http://in.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:19 http://in.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [175 kB]
Get:20 http://in.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:21 http://in.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [386 kB]
Get:22 http://in.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:23 http://in.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7,296 B]
Get:24 http://in.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:25 http://in.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [10.5 kB]
Get:26 http://in.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Fetched 1,089 kB in 9s (123 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Figure 1: System Update Command

Installing Java

`sudo apt install openjdk-11-jdk -y`

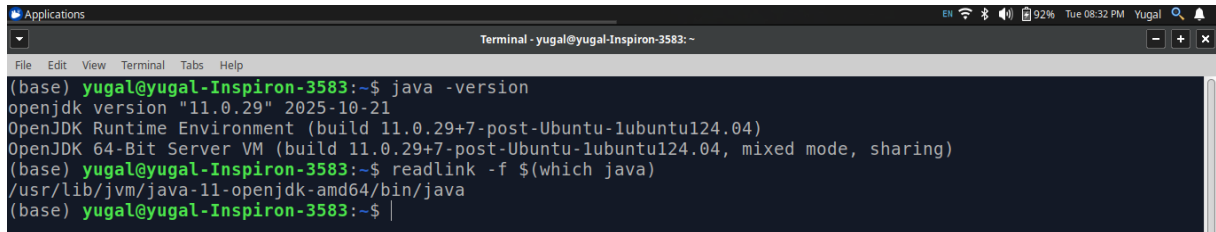


```
(base) yugal@yugal-Inspiron-3583:~$ sudo apt install openjdk-11-jdk -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  default-jre libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source visualvm fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  ca-certificates-java java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 19 newly installed, 0 to remove and 2 not upgraded.
Need to get 121 MB of archives.
After this operation, 478 MB of additional disk space will be used.
```

Figure 2: Java Installation

Java Version and Path Verification

```
java -version  
readlink -f $(which java)
```

A terminal window titled 'Terminal - yugal@yugal-Inspiron-3583: ~' showing the output of 'java -version' and 'readlink -f \$(which java)'. The output of 'java -version' shows 'openjdk version "11.0.29" 2025-10-21' and 'OpenJDK Runtime Environment (build 11.0.29+7-post-Ubuntu-1ubuntu124.04)'. The output of 'readlink -f \$(which java)' shows '/usr/lib/jvm/java-11-openjdk-amd64/bin/java'.

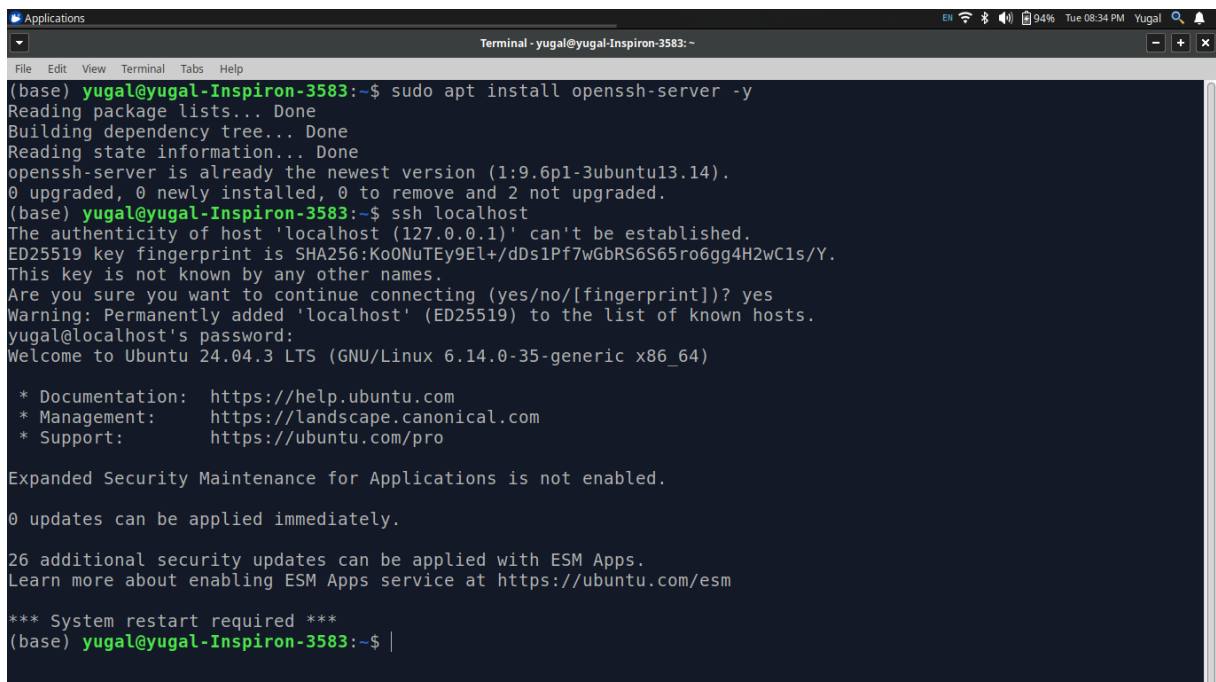
```
(base) yugal@yugal-Inspiron-3583:~$ java -version  
openjdk version "11.0.29" 2025-10-21  
OpenJDK Runtime Environment (build 11.0.29+7-post-Ubuntu-1ubuntu124.04)  
OpenJDK 64-Bit Server VM (build 11.0.29+7-post-Ubuntu-1ubuntu124.04, mixed mode, sharing)  
(base) yugal@yugal-Inspiron-3583:~$ readlink -f $(which java)  
/usr/lib/jvm/java-11-openjdk-amd64/bin/java  
(base) yugal@yugal-Inspiron-3583:~$
```

Figure 3: Java Version and Path Verification

1.2 Step 2: Setup SSH

Install SSH Server

```
sudo apt install openssh-server -y  
ssh localhost
```

A terminal window titled 'Terminal - yugal@yugal-Inspiron-3583: ~' showing the output of 'sudo apt install openssh-server -y' and 'ssh localhost'. The output of 'sudo apt install openssh-server -y' shows 'Reading package lists... Done', 'Building dependency tree... Done', 'Reading state information... Done', and 'openssh-server is already the newest version (1:9.6p1-3ubuntu13.14)'. The output of 'ssh localhost' shows 'The authenticity of host 'localhost (127.0.0.1)' can't be established.', 'ED25519 key fingerprint is SHA256:Ko0NuTEy9EL+/dDs1Pf7wGbRS6S6Sro6gg4H2wC1s/Y.', 'This key is not known by any other names.', 'Are you sure you want to continue connecting (yes/no/[fingerprint])? yes', 'Warning: Permanently added 'localhost' (ED25519) to the list of known hosts.', 'yugal@localhost's password:', 'Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-35-generic x86_64)', and '*** System restart required ***'.

```
(base) yugal@yugal-Inspiron-3583:~$ sudo apt install openssh-server -y  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
openssh-server is already the newest version (1:9.6p1-3ubuntu13.14).  
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.  
(base) yugal@yugal-Inspiron-3583:~$ ssh localhost  
The authenticity of host 'localhost (127.0.0.1)' can't be established.  
ED25519 key fingerprint is SHA256:Ko0NuTEy9EL+/dDs1Pf7wGbRS6S6Sro6gg4H2wC1s/Y.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'localhost' (ED25519) to the list of known hosts.  
yugal@localhost's password:  
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-35-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/pro  
  
Expanded Security Maintenance for Applications is not enabled.  
  
0 updates can be applied immediately.  
  
26 additional security updates can be applied with ESM Apps.  
Learn more about enabling ESM Apps service at https://ubuntu.com/esm  
  
*** System restart required ***  
(base) yugal@yugal-Inspiron-3583:~$
```

Figure 4: Install SSH Server

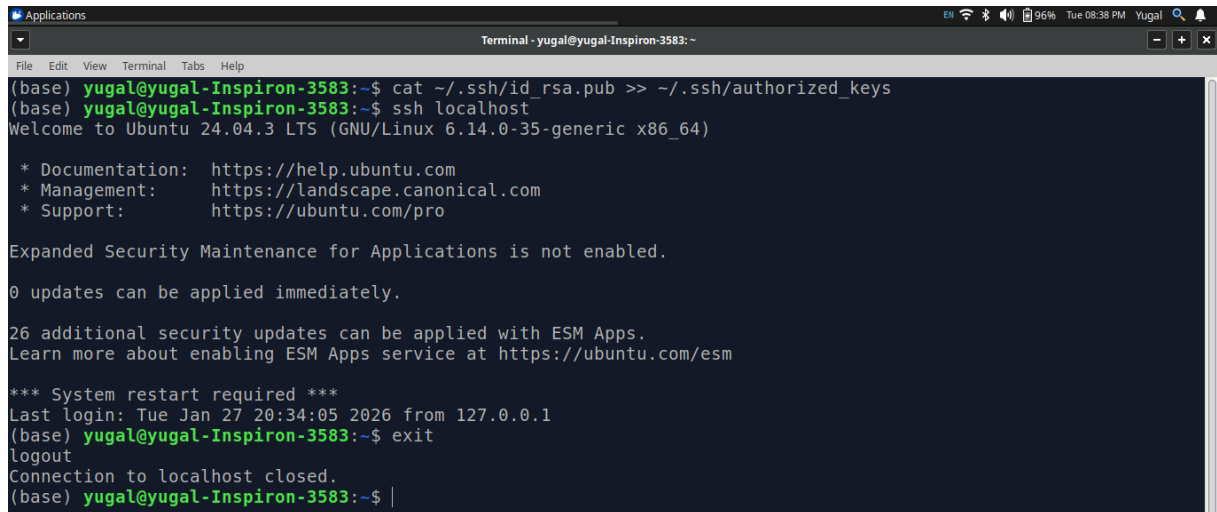
Generate SSH Key Pair

```
ssh-keygen -t rsa -P ""  
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```



```
ssh localhost
```

```
exit
```

A terminal window titled 'Terminal - yugal@yugal-Inspiron-3583: ~' showing the process of generating an SSH key pair and connecting to localhost. The user runs 'cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys', then 'ssh localhost'. The terminal displays the Ubuntu 24.04.3 LTS welcome message, system updates, and the user's login session. The session ends with 'exit', logging out, and closing the connection to localhost.

```
(base) yugal@yugal-Inspiron-3583:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
(base) yugal@yugal-Inspiron-3583:~$ ssh localhost
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-35-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

26 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
Last login: Tue Jan 27 20:34:05 2026 from 127.0.0.1
(base) yugal@yugal-Inspiron-3583:~$ exit
logout
Connection to localhost closed.
(base) yugal@yugal-Inspiron-3583:~$ |
```

Figure 5: Generate SSH Key Pair

1.3 Step 3: Download and Configure Hadoop

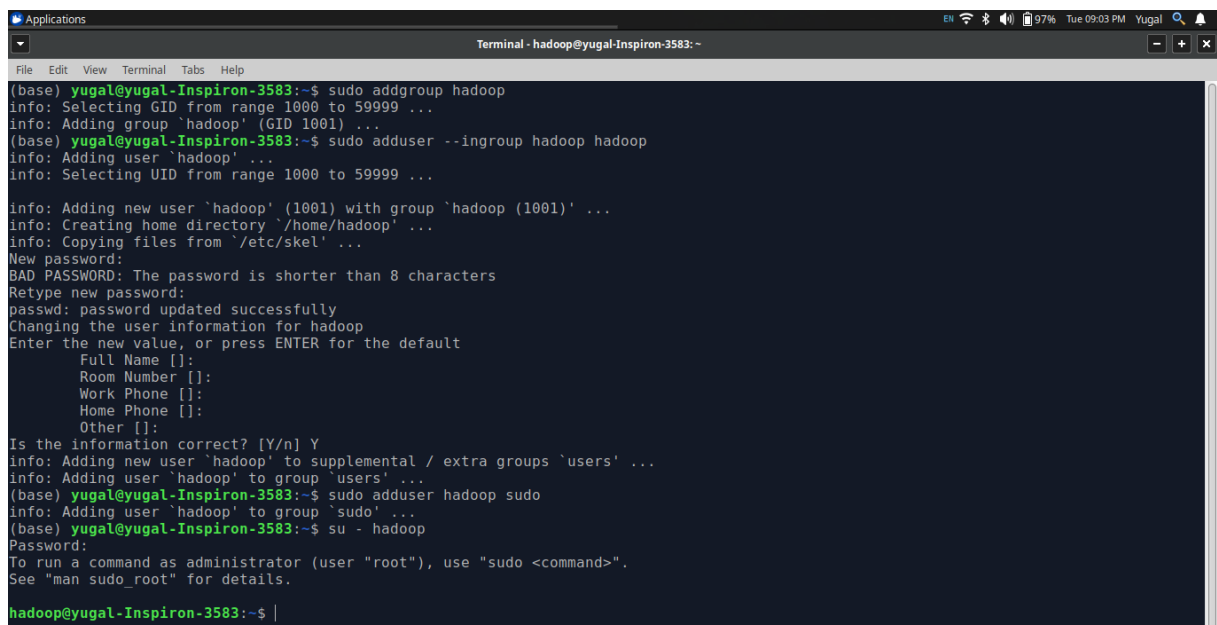
Create Hadoop User

```
sudo addgroup hadoop
```

```
sudo adduser --ingroup hadoop hadoop
```

```
sudo adduser hadoop sudo
```

```
su - hadoop
```

A terminal window titled 'Terminal - hadoop@yugal-Inspiron-3583: ~' showing the execution of commands to create a new user named 'hadoop'. The user runs 'sudo addgroup hadoop' and 'sudo adduser --ingroup hadoop hadoop'. The terminal shows the group being added, the user being added, and the user's password being set. The user then runs 'su - hadoop' to switch to the 'hadoop' user.

```
(base) yugal@yugal-Inspiron-3583:~$ sudo addgroup hadoop
info: Selecting GID from range 1000 to 59999 ...
info: Adding group `hadoop' (GID 1001) ...
(base) yugal@yugal-Inspiron-3583:~$ sudo adduser --ingroup hadoop hadoop
info: Adding user `hadoop' ...
info: Selecting UID from range 1000 to 59999 ...

info: Adding new user `hadoop' (1001) with group `hadoop (1001)' ...
info: Creating home directory `/home/hadoop' ...
info: Copying files from `/etc/skel' ...
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
Changing the user information for hadoop
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] Y
info: Adding new user `hadoop' to supplemental / extra groups `users' ...
info: Adding user `hadoop' to group `users' ...
(base) yugal@yugal-Inspiron-3583:~$ sudo adduser hadoop sudo
info: Adding user `hadoop' to group `sudo' ...
(base) yugal@yugal-Inspiron-3583:~$ su - hadoop
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
hadoop@yugal-Inspiron-3583:~$ |
```

Figure 6: Create Hadoop User

Download and Extract Hadoop

exit

```
sudo cp /home/yugal/Downloads/hadoop-3.3.0.tar.gz /home/hadoop/
```

```
sudo chown hadoop:hadoop /home/hadoop/hadoop-3.3.0.tar.gz
```

```
su - hadoop
```

```
ls
```

```
tar -xvzf hadoop-3.3.0.tar.gz
```

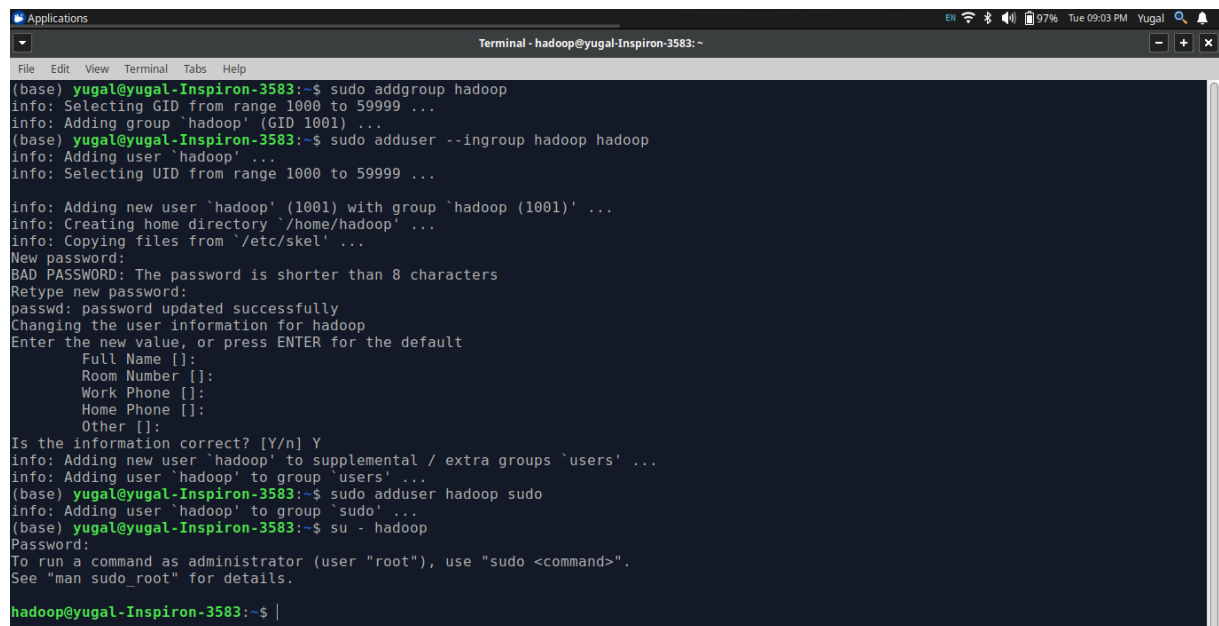
A terminal window titled 'Terminal - hadoop@yugal-Inspiron-3583: ~' showing the process of creating a new user and group. The user 'yugal@yugal-Inspiron-3583' runs 'sudo addgroup hadoop', which selects GID 1001. Then, 'sudo adduser --ingroup hadoop hadoop' is run, which prompts for a password (it's rejected as too short, then accepted), sets the home directory to /home/hadoop, and copies files from /etc/skel. The user is then prompted for additional information (Full Name, Room Number, Work Phone, Home Phone, Other) and asked if the information is correct (Y/n). Finally, 'sudo adduser hadoop sudo' is run to add the user to the sudo group. The terminal ends with 'hadoop@yugal-Inspiron-3583:~\$'.

Figure 7: Download and Extract Hadoop

Moving Hadoop

```
mv hadoop-3.3.0 hadoop
```

```
ls
```

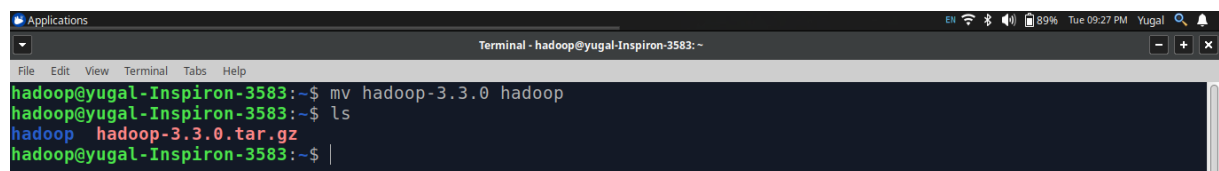
A terminal window titled 'Terminal - hadoop@yugal-Inspiron-3583: ~' showing the movement of the hadoop directory. The user 'hadoop@yugal-Inspiron-3583' runs 'mv hadoop-3.3.0 hadoop'. Then, 'ls' is run, showing the directory 'hadoop-3.3.0.tar.gz'. The terminal ends with 'hadoop@yugal-Inspiron-3583:~\$'.

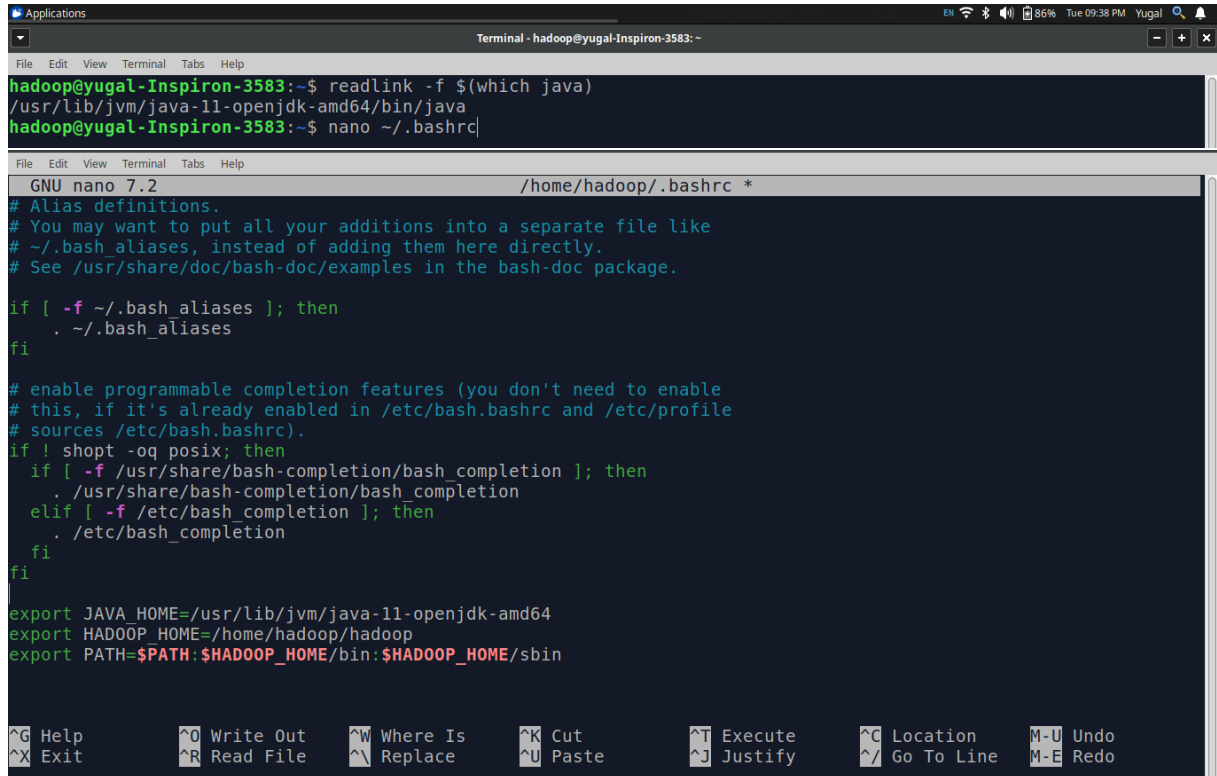
Figure 8: Moving Hadoop

1.4 Step 4: Environment Variable Configuration

.bashrc Modification

```
nano ~/.bashrc
```

```
% Add the following lines at the end of the file
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_HOME=/home/hadoop/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```



```
hadoop@yugal-Inspiron-3583:~$ readlink -f $(which java)
/usr/lib/jvm/java-11-openjdk-amd64/bin/java
hadoop@yugal-Inspiron-3583:~$ nano ~/.bashrc

GNU nano 7.2 /home/hadoop/.bashrc *
# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.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

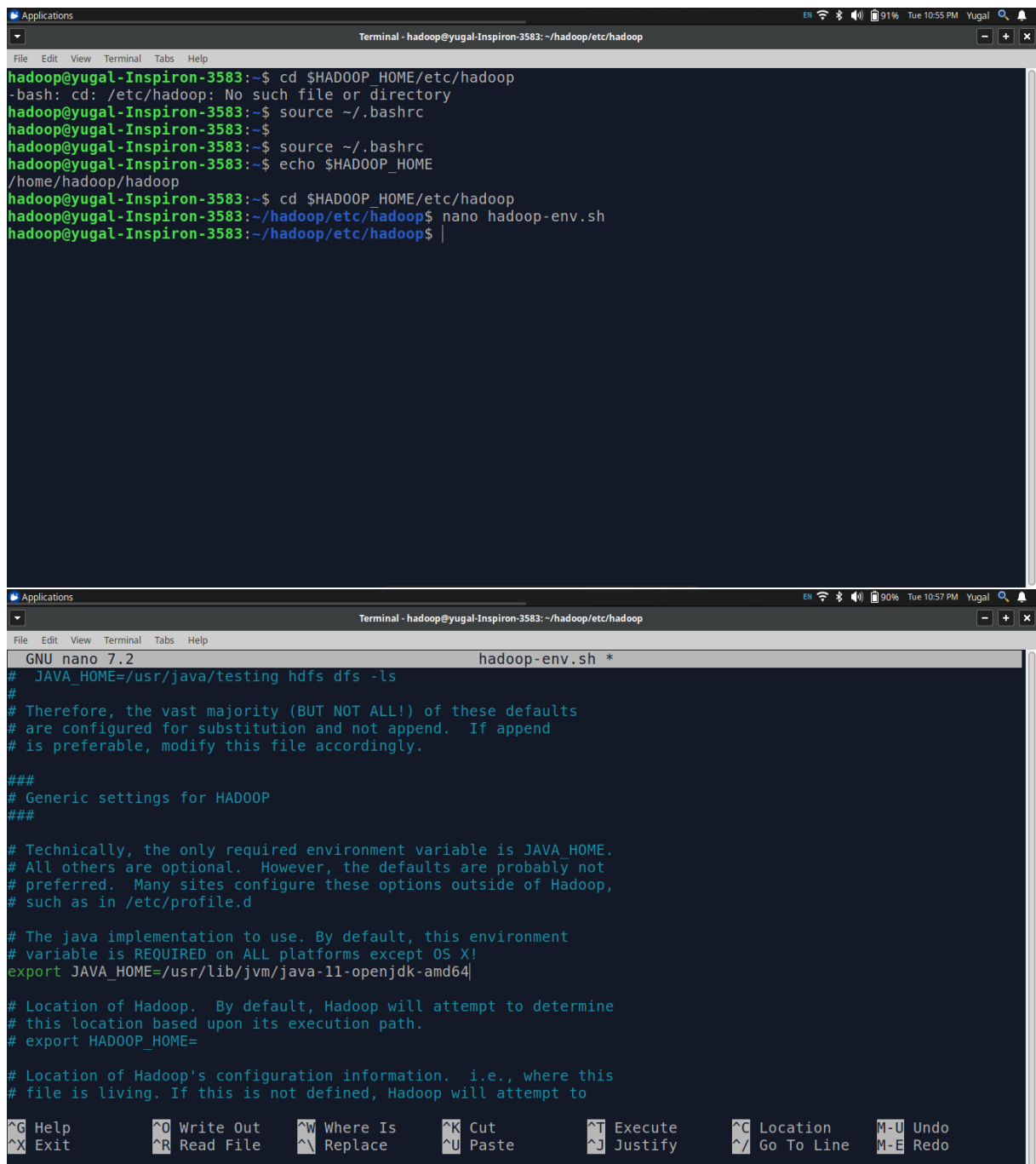
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_HOME=/home/hadoop/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

Figure 9: .bashrc Modification

1.5 Step 5: Hadoop XML Configuration Files

hadoop-env.sh Modification

```
source ~/.bashrc
echo $HADOOP_HOME
/home/hadoop/hadoop
cd $HADOOP_HOME/etc/hadoop
ls
nano hadoop-env.sh
% Add the following lines at the path
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```



```
hadoop@yugal-Inspiron-3583:~$ cd $HADOOP_HOME/etc/hadoop
-bash: cd: /etc/hadoop: No such file or directory
hadoop@yugal-Inspiron-3583:~$ source ~/.bashrc
hadoop@yugal-Inspiron-3583:~$ source ~/.bashrc
hadoop@yugal-Inspiron-3583:~$ echo $HADOOP_HOME
/home/hadoop/hadoop
hadoop@yugal-Inspiron-3583:~$ cd $HADOOP_HOME/etc/hadoop
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ nano hadoop-env.sh
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$

GNU nano 7.2 hadoop-env.sh *
# JAVA_HOME=/usr/java/testing hdfs dfs -ls
#
# 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.
# All others are optional. However, the defaults are probably not
# preferred. Many sites configure these options outside of Hadoop,
# such as in /etc/profile.d
#
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
#
# Location of Hadoop. By default, Hadoop will attempt to determine
# this location based upon its execution path.
# export HADOOP_HOME=
#
# Location of Hadoop's configuration information. i.e., where this
# file is living. If this is not defined, Hadoop will attempt to
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```

Figure 10: hadoop-env.sh Modification

core-site.xml Modification

nano core-site.xml

% Add the following property inside <configuration> tag

<property>

<name>fs.defaultFS</name>

<value>hdfs://localhost:9000</value>

</property>

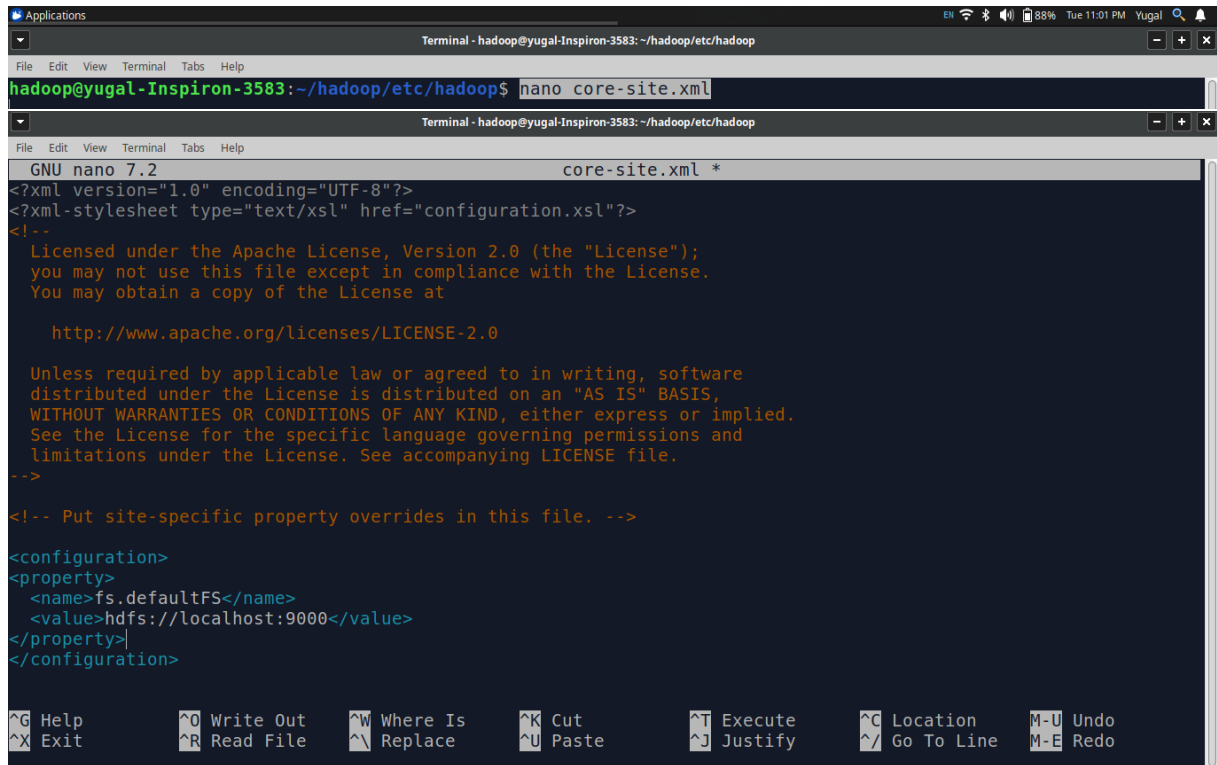


Figure 11: core-site.xml Modification

hdfs-site.xml Modification

nano hdfs-site.xml

% Add the following property inside <configuration> tag

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:///home/hadoop/hdfs/namenode</value>

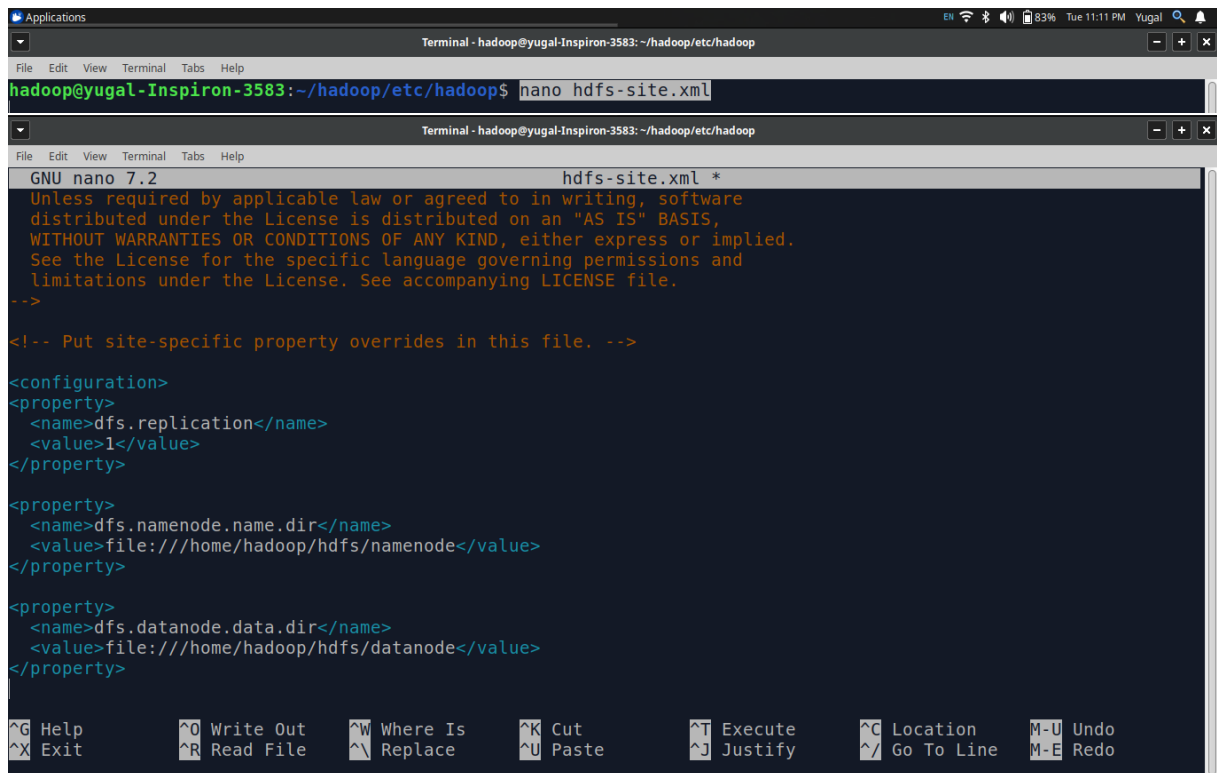
</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>file:///home/hadoop/hdfs/datanode</value>

</property>



```
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ nano hdfs-site.xml

GNU nano 7.2 hdfs-site.xml *
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>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>

<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:///home/hadoop/hdfs/namenode</value>
</property>

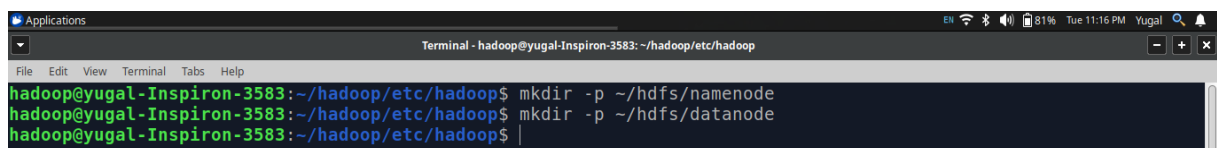
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:///home/hadoop/hdfs/datanode</value>
</property>

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

Figure 12: hdfs-site.xml Modification

Creating Folder for NameNode and DataNode

```
mkdir -p ~/hdfs/namenode
mkdir -p ~/hdfs/datanode
```

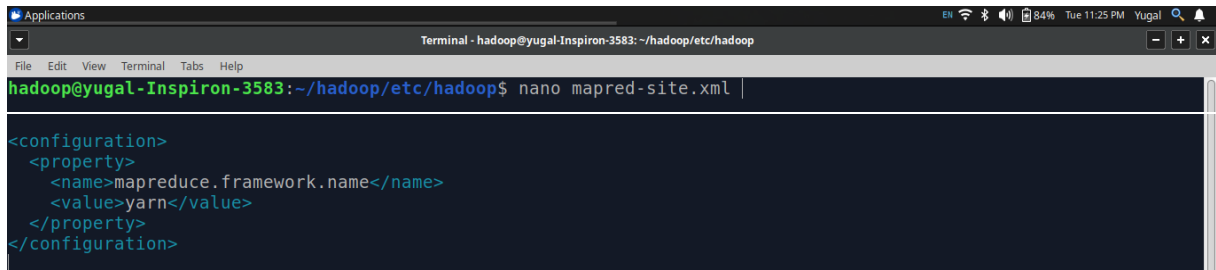


```
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ mkdir -p ~/hdfs/namenode
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ mkdir -p ~/hdfs/datanode
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$
```

Figure 13: Creating Folder for NameNode and DataNode

mapred-site.xml Modification

```
nano mapred-site.xml
% Add the following property inside <configuration> tag
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
```



```
Applications
Terminal - hadoop@yugal-Inspiron-3583: ~/hadoop/etc/hadoop
File Edit View Terminal Tabs Help
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ nano mapred-site.xml

<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

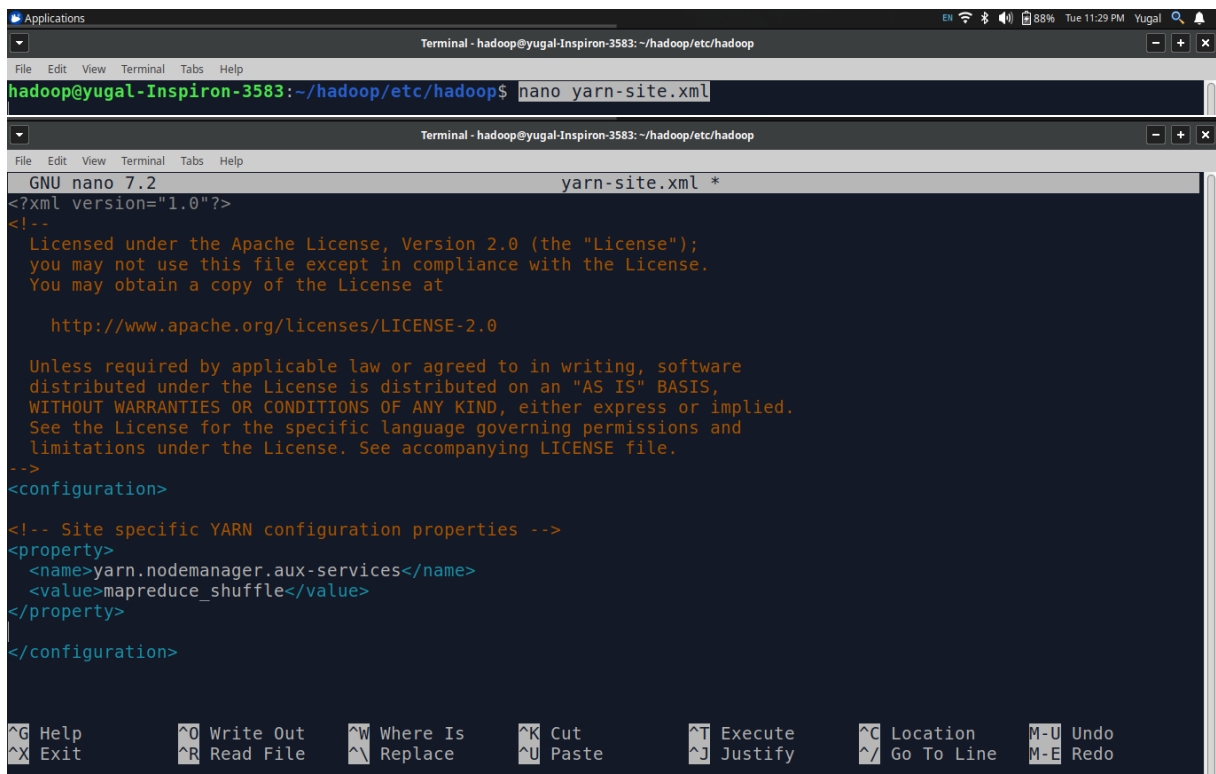
Figure 14: mapred-site.xml Modification

yarn-site.xml Modification

nano yarn-site.xml

% Add the following property inside <configuration> tag

```
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
```



```
Applications
Terminal - hadoop@yugal-Inspiron-3583: ~/hadoop/etc/hadoop
File Edit View Terminal Tabs Help
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ nano yarn-site.xml

GNU nano 7.2 yarn-site.xml *
<?xml version="1.0"?>
<!--
Licensed 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. See accompanying LICENSE file.
-->
<configuration>

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

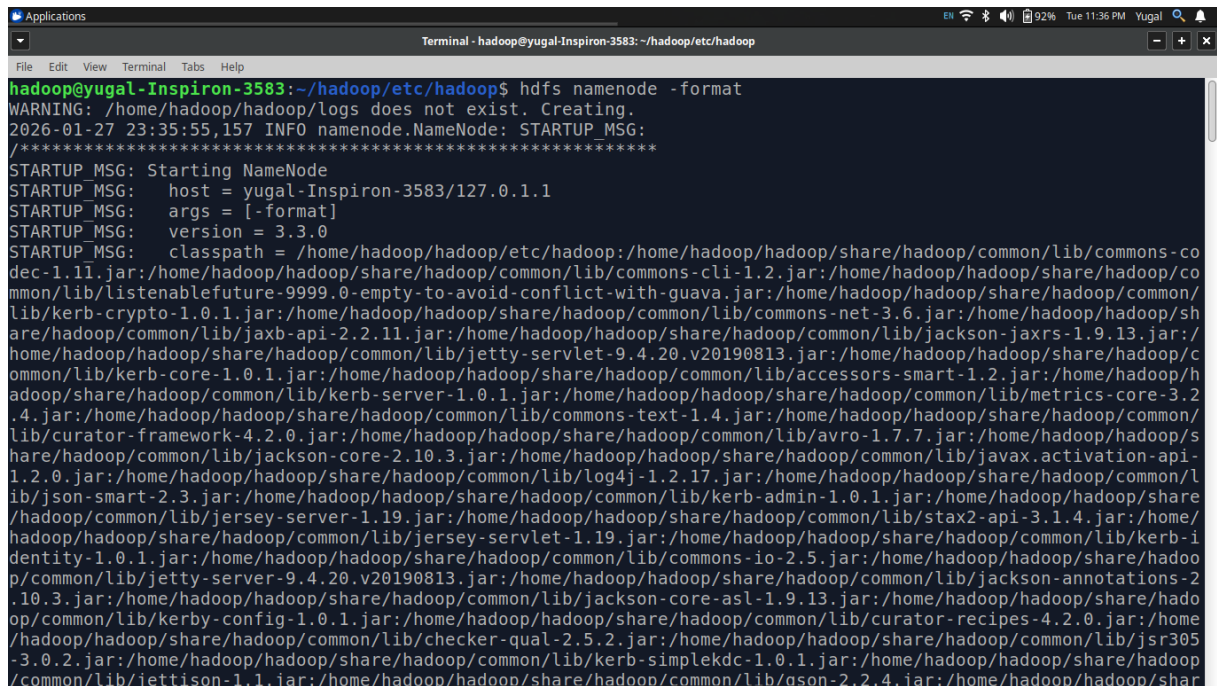
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_ Go To Line M-E Redo
```

Figure 15: yarn-site.xml Modification

1.6 Step 6: Format NameNode & Start Hadoop Services

Format NameNode

```
hdfs namenode -format
```



```
Applications
Terminal - hadoop@yugal-Inspiron-3583: ~/hadoop/etc/hadoop
File Edit View Terminal Tabs Help
hadoop@yugal-Inspiron-3583:~/hadoop/etc/hadoop$ hdfs namenode -format
WARNING: /home/hadoop/hadoop/logs does not exist. Creating.
2026-01-27 23:35:55,157 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = yugal-Inspiron-3583/127.0.1.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.3.0
STARTUP_MSG:   classpath = /home/hadoop/hadoop/etc/hadoop:/home/hadoop/hadoop/share/hadoop/common/lib/commons-codec-1.11.jar:/home/hadoop/hadoop/share/hadoop/common/lib/commons-cli-1.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/listenablefuture-9999.0-empty-to-avoid-conflict-with-guava.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-crypto-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/commons-net-3.6.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jaxb-api-2.2.11.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-jaxrs-1.9.13.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jetty-servlet-9.4.20.v20190813.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-core-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/accessors-smart-1.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-server-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/metrics-core-3.2.4.jar:/home/hadoop/hadoop/share/hadoop/common/lib/commons-text-1.4.jar:/home/hadoop/hadoop/share/hadoop/common/lib/curator-framework-4.2.0.jar:/home/hadoop/hadoop/share/hadoop/common/lib/avro-1.7.7.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-core-2.10.3.jar:/home/hadoop/hadoop/share/hadoop/common/lib/javax.activation-api-1.2.0.jar:/home/hadoop/hadoop/share/hadoop/common/lib/log4j-1.2.17.jar:/home/hadoop/hadoop/share/hadoop/common/lib/json-smart-2.3.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-admin-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jersey-server-1.19.jar:/home/hadoop/hadoop/share/hadoop/common/lib/stax2-api-3.1.4.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jersey-servlet-1.19.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-identity-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/commons-io-2.5.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jetty-server-9.4.20.v20190813.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-annotations-2.10.3.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-core-asl-1.9.13.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerby-config-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/curator-recipes-4.2.0.jar:/home/hadoop/hadoop/share/hadoop/common/lib/checker-qual-2.5.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jsr305-3.0.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-simplekdc-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jettison-1.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/gson-2.2.4.jar:/home/hadoop/hadoop/share
```

Figure 16: Format NameNode

Start Hadoop Services

```
start-dfs.sh
```

```
jps
```



```
hadoop@yugal-Inspiron-3583:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [yugal-Inspiron-3583]
hadoop@yugal-Inspiron-3583:~$ jps
81351 Jps
81062 SecondaryNameNode
80713 NameNode
80847 DataNode
```

Figure 17: Start Hadoop Services

Start YARN Services

```
start-yarn.sh
```

```
jps
```


The image shows a terminal window and a web browser. The terminal window displays the following commands and output:

```
hadoop@yugal-Inspiron-3583:~$ jps
81351 Jps
81062 SecondaryNameNode
80713 NameNode
80847 DataNode
hadoop@yugal-Inspiron-3583:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@yugal-Inspiron-3583:~$ jps
81792 NodeManager
82355 Jps
81670 ResourceManager
81062 SecondaryNameNode
80713 NameNode
80847 DataNode
hadoop@yugal-Inspiron-3583:~$ |
```

The web browser shows the Hadoop Overview page for 'localhost:9000' (active). The page includes a table with the following information:

Started:	Tue Jan 27 23:49:34 +0530 2026
Version:	3.3.0, raa96f1871bfd858f9bac59cf2a81ec470da649af
Compiled:	Tue Jul 07 00:14:00 +0530 2020 by brahma from branch-3.3.0
Cluster ID:	CID-7d9aa263-5957-4992-9da1-32c648d8341c
Block Pool ID:	BP-1685072063-127.0.1.1-1769537156943

The Summary section shows the following information:

- Security is off.
- Safemode is off.
- 1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).
- Heap Memory used 119.44 MB of 616 MB Heap Memory. Max Heap Memory is 3.88 GB.
- Non Heap Memory used 53.86 MB of 56.5 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Figure 18: Start YARN Services

1.7 Step 7: HDFS Operations

```
% Create a test file
echo "Hello Big Data" > file.txt

% Make directory in HDFS
hdfs dfs -mkdir /data
```

```

% Upload file to HDFS
hdfs dfs -put file.txt /data
% List files
hdfs dfs -ls /data
% Check blocks & replication
hdfs fsck /data/file.txt -files -blocks
% Change permission
hdfs dfs -chmod 777 /data/file.txt
% Delete file
hdfs dfs -rm /data/file.txt
hdfs dfs -ls /data

```

```

Applications
Terminal - hadoop@yugal-Inspiron-3583: ~
File Edit View Terminal Tabs Help
hadoop@yugal-Inspiron-3583:~$ echo "Hello Big Data" > file.txt
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -mkdir /data
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -put file.txt /data
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /data
Found 1 items
-rw-r--r-- 1 hadoop supergroup      15 2026-01-28 00:50 /data/file.txt
hadoop@yugal-Inspiron-3583:~$ hdfs fsck /data/file.txt -files -blocks
Connecting to namenode via http://localhost:9870/fsck?ugi=hadoop&files=1&blocks=1&path=%2Fdata%2Ffile.txt
FSCK started by hadoop (auth:SIMPLE) from /127.0.0.1 for path /data/file.txt at Wed Jan 28 00:51:19 IST 2026

/data/file.txt 15 bytes, replicated: replication=1, 1 block(s): OK
0. BP-1685072063-127.0.1.1-1769537156943:blk_1073741825_1001 len=15 Live_repl=1

Status: HEALTHY
Number of data-nodes: 1
Number of racks:      1
Total dirs:           0
Total symlinks:       0

```

Figure 19: HDFS Operation 1

```

Erasure Coded Block Groups:
Total size:      0 B
Total files:     0
Total block groups (validated): 0
Minimally erasure-coded block groups: 0
Over-erasure-coded block groups: 0
Under-erasure-coded block groups: 0
Unsatisfactory placement block groups: 0
Average block group size: 0.0
Missing block groups: 0
Corrupt block groups: 0
Missing internal blocks: 0
Blocks queued for replication: 0
FSCK ended at Wed Jan 28 00:51:19 IST 2026 in 37 milliseconds

The filesystem under path '/data/file.txt' is HEALTHY
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -chmod 777 /data/file/txt
chmod: '/data/file/txt': No such file or directory
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -rm /data/file.txt

```

Figure 20: HDFS Operation 2

Browse Directory

/data

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rwxrwxrwx	hadoop	supergroup	15 B	Jan 28 00:50	1	128 MB	file.txt

Showing 1 to 1 of 1 entries

Previous 1 Next

Hadoop, 2020.

Figure 21: HDFS Result 1

hadoop SUBMITTED Applications

Cluster

- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW_SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler
- Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used
0	0	0	0	0	0 B

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes
1	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>

Show 20 entries

ID	User	Name	Application Type	Application Tags	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus
No data available in table											

Showing 0 to 0 of 0 entries

Figure 22: HDFS Result 2

1.8 Result

Hadoop was successfully installed and configured in pseudo-distributed mode, and various HDFS operations were performed successfully.

1.9 Conclusion

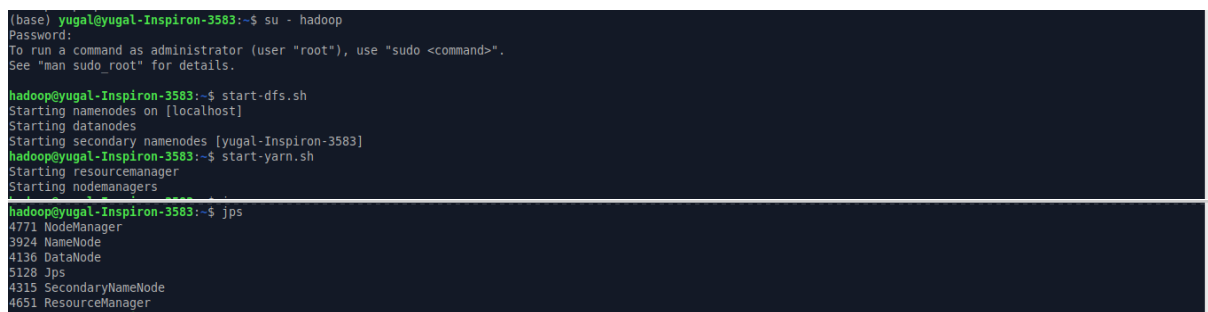
This experiment provided hands-on experience in setting up a Big Data environment using Hadoop and understanding the working of HDFS through practical command execution.

Assignment 2: Load large datasets into HDFS and analyze block distribution.

Objective: Understand how Apache Hadoop HDFS stores large files by splitting them into fixed-size blocks and placing them on DataNodes.

2.1 Step 1: Collect large Dataset (CSV/TXT)

```
su - hadoop
start-dfs.sh
start-yarn.sh
jps
```



```
(base) yugal@yugal-Inspiron-3583:~$ su - hadoop
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

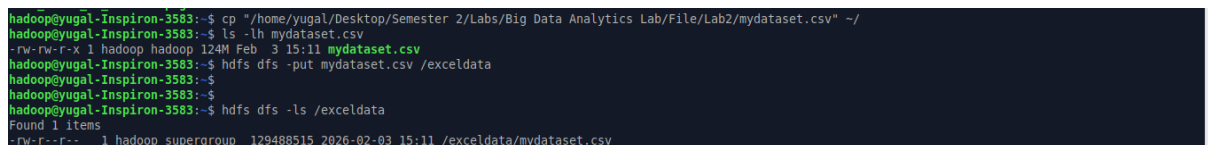
hadoop@yugal-Inspiron-3583:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [yugal-Inspiron-3583]
hadoop@yugal-Inspiron-3583:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers

hadoop@yugal-Inspiron-3583:~$ jps
4771 NodeManager
3924 NameNode
4136 DataNode
5128 Jps
4315 SecondaryNameNode
4651 ResourceManager
```

Figure 23: Start Hadoop Services and Verify with jps

2.2 Step 2: Create HDFS directory.

```
% Created 352 MB csv file using:excel and uploaded to hdfs:
% Created exceldata directory in hdfs:
hdfs dfs -mkdir /exceldata
ls -lh mydataset.csv
hdfs dfs -put mydataset.csv /exceldata
hdfs dfs -ls /exceldata
```



```
hadoop@yugal-Inspiron-3583:~$ cp "/home/yugal/Desktop/Semester 2/Labs/Big Data Analytics Lab/File/Lab2/mydataset.csv" ~/
hadoop@yugal-Inspiron-3583:~$ ls -lh mydataset.csv
-rw-rw-r--x 1 hadoop hadoop 124M Feb  3 15:11 mydataset.csv
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -put mydataset.csv /exceldata
hadoop@yugal-Inspiron-3583:~$
hadoop@yugal-Inspiron-3583:~$
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /exceldata
Found 1 items
-rw-r--r-- 1 hadoop supergroup 129488515 2026-02-03 15:11 /exceldata/mydataset.csv
```

Figure 24: Created exceldata directory in hdfs and copied dataset

2.3 Step 3: Upload large dataset into HDFS

```
hdfs dfs -put mydataset.csv /exceldata
```



```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -put mydataset.csv /exceldata
```

Figure 25: Upload large dataset into HDFS

2.4 Step 4: Verify data storage

```
hdfs dfs -ls /exceldata
```

```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /exceldata
Found 1 items
-rw-r--r-- 1 hadoop supergroup 129488515 2026-02-03 15:11 /exceldata/mydataset.csv
```

Figure 26: Verify data storage in HDFS

2.5 Step 5: Analyze block distribution

```
hdfs fsck /exceldata/mydataset.csv -files -blocks -locations
```

```
hadoop@yugal-Inspiron-3583:~$ hdfs fsck /exceldata/mydataset.csv -files -blocks -locations
Connecting to namenode via http://localhost:9870/fsck?ugi=hadoop&files=1&blocks=1&locations=1&path=%2Fexceldata%2Fmydataset.csv
FSCK started by hadoop (auth:SIMPLE) from /127.0.0.1 for path /exceldata/mydataset.csv at Tue Feb 03 15:18:34 IST 2026

/exceldata/mydataset.csv 369487717 bytes, replicated: replication=1, 3 block(s): OK
0. BP-1685072063-127.0.1.1-1769537156943:blk_1073741830_1006 len=134217728 Live_repl=1 [DatanodeInfoWithStorage[127.0.0.1:9866,DS-47acaae1-e863-462a-9aec-290782a597df,DISK]]
1. BP-1685072063-127.0.1.1-1769537156943:blk_1073741831_1007 len=134217728 Live_repl=1 [DatanodeInfoWithStorage[127.0.0.1:9866,DS-47acaae1-e863-462a-9aec-290782a597df,DISK]]
2. BP-1685072063-127.0.1.1-1769537156943:blk_1073741832_1008 len=101052261 Live_repl=1 [DatanodeInfoWithStorage[127.0.0.1:9866,DS-47acaae1-e863-462a-9aec-290782a597df,DISK]]

Status: HEALTHY
Number of data-nodes: 1
Number of racks: 1
Total dirs: 0
Total symlinks: 0

Replicated Blocks:
Total size: 369487717 B
Total files: 1
Total blocks (validated): 3 (avg. block size 123162572 B)
Minimally replicated blocks: 3 (100.0 %)
Over-replicated blocks: 0 (0.0 %)
Under-replicated blocks: 0 (0.0 %)
Mis-replicated blocks: 0 (0.0 %)
Default replication factor: 1
Average block replication: 1.0
Missing blocks: 0
Corrupt blocks: 0
Missing replicas: 0 (0.0 %)
Blocks queued for replication: 0

Erasure Coded Block Groups:
Total size: 0 B
Total files: 0
Total block groups (validated): 0
Minimally erasure-coded block groups: 0
Over-erasure-coded block groups: 0
```

Figure 27: Analyze block distribution in HDFS

2.6 Step 6: Download processed dataset from HDFS to local system

```
hdfs dfs -get /exceldata/mydataset.csv mydataset_from_hdfs.csv
```

```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -get /exceldata/mydataset.csv mydataset_from_hdfs.csv
```

Figure 28: Download processed dataset from HDFS to local system

2.7 Step 7: Delete data from HDFS

```
hdfs dfs -rm -r /exceldata
```

```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -get /exceldata/mydataset.csv mydataset_from_hdfs.csv
hdfs dfs -rm -r /exceldata
get: 'mydataset_from_hdfs.csv': File exists
Deleted /exceldata
hadoop@yugal-Inspiron-3583:~$
```

Figure 29: Delete data from HDFS

2.8 Step 8: Check file block information

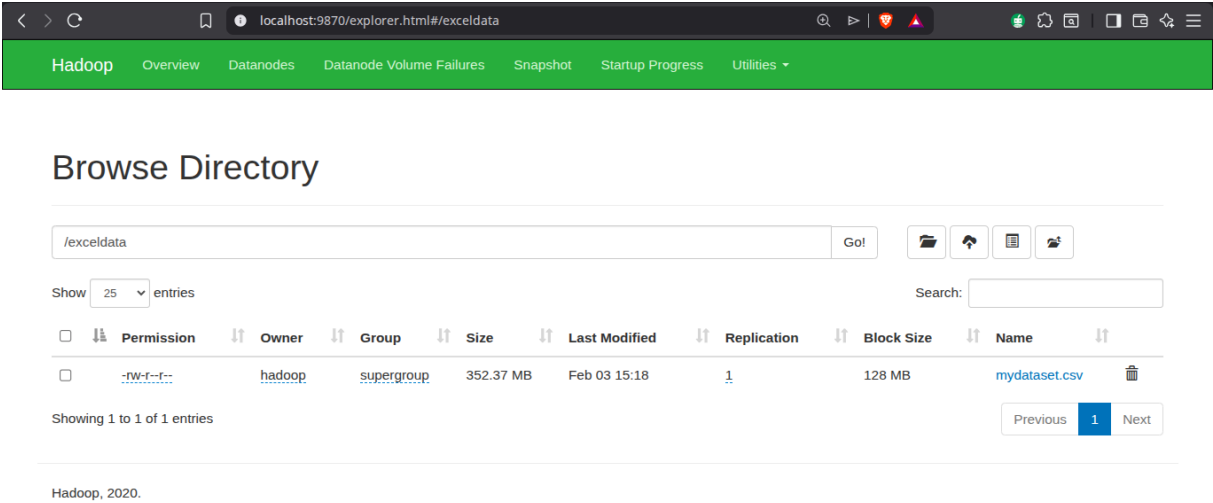


Figure 30: Check file block information in HDFS

2.9 Result

The large dataset (mydataset.csv) was successfully uploaded to HDFS under the /exceldata directory. The file was split into blocks and distributed across DataNodes, as verified by the block distribution analysis. The dataset was also downloaded back to the local system, confirming that data retrieval from HDFS works correctly. Finally, the dataset was deleted from HDFS, demonstrating proper data management and cleanup.

2.10 Conclusion

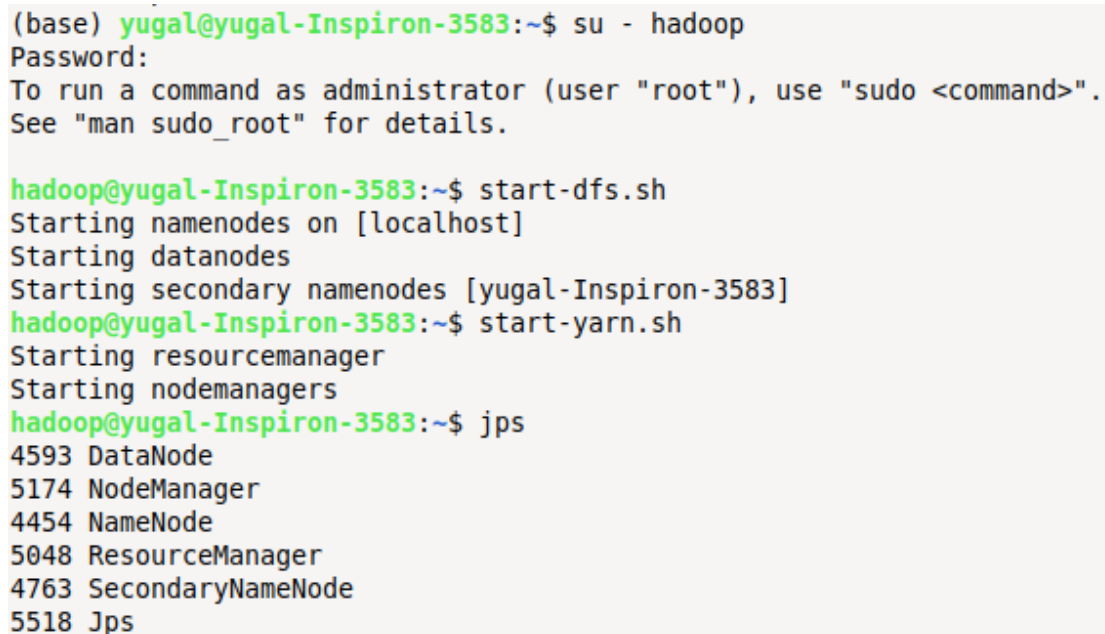
This assignment provided hands-on experience with Apache Hadoop HDFS, demonstrating how to upload large datasets, analyze block distribution, and manage data within the HDFS environment.

Assignment 3: Implement MapReduce programs for Data Processing.

Objective: Apply distributed computing concepts by implementing a MapReduce program in Hadoop to process data efficiently across distributed nodes.

3.1 Step 1: Start Hadoop Services

```
su - hadoop
start-dfs.sh
start-yarn.sh
jps
```



```
(base) yugal@yugal-Inspiron-3583:~$ su - hadoop
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

hadoop@yugal-Inspiron-3583:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [yugal-Inspiron-3583]
hadoop@yugal-Inspiron-3583:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@yugal-Inspiron-3583:~$ jps
4593 DataNode
5174 NodeManager
4454 NameNode
5048 ResourceManager
4763 SecondaryNameNode
5518 Jps
```

Figure 31: Starting Hadoop services and verifying with jps

3.2 Step 2: Create Input Directory in HDFS

```
nano input.txt
ls
cat input.txt
hdfs dfs -mkdir /input
hdfs dfs -put input.txt /input
hdfs dfs -ls /input
```



```

hadoop@yugal-Inspiron-3583:~$ nano input.txt
hadoop@yugal-Inspiron-3583:~$ ls
bigfile.txt  hadoop          hdfs          mydataset.csv  snap
file.txt     hadoop-3.3.0.tar.gz  input.txt     mydataset_from_hdfs.csv
hadoop@yugal-Inspiron-3583:~$ cat input.txt
hadoop mapreduce hadoop
big data hadoop mapreduce
mapreduce big data
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -mkdir /input
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -put input.txt /input
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /input
Found 1 items
-rw-r--r--  1 hadoop supergroup          69 2026-02-10 13:54 /input/input.txt

```

Figure 32: Creating input directory and uploading dataset

3.3 Step 3: Write MapReduce Program

Mapper Class: Processes input data and generates intermediate key-value pairs.

nano Mapper.java

```

File Edit View Terminal Tabs Help
GNU nano 7.2 MapperClass.java *
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class MapperClass extends Mapper<Object, Text, Text, IntWritable> {

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

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

        StringTokenizer itr = new StringTokenizer(value.toString());

        while (itr.hasMoreTokens()) {
            word.set(itr.nextToken());
            context.write(word, one);
        }
    }
}

```

Figure 33: Mapper class implementation

Reducer Class: Aggregates values for each key produced by the mapper.

nano Reducer.java

```

GNU nano 7.2 ReducerClass.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 ReducerClass 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));
    }
}

```

Figure 34: Reducer class implementation

Driver Class: Configures job parameters such as input/output paths, mapper, reducer, and execution settings.

nano Driver.java

```

GNU nano 7.2 Driver.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 Driver{

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

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Word Count");

        job.setJarByClass(Driver.class);

        job.setMapperClass(MapperClass.class);
        job.setReducerClass(ReducerClass.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}

```

Figure 35: Driver class implementation

3.4 Step 4: Compile Program and Create JAR

```
javac -classpath 'hadoop classpath' -d . Mapper.java Reducer.java Driver.java
jar -cvf wordcount.jar *.class
```

```
hadoop@yugal-Inspiron-3583:~$ javac -classpath `hadoop classpath` -d . MapperClass.java ReducerClass.java Driver.java
hadoop@yugal-Inspiron-3583:~$ jar -cvf wordcount.jar *.class
added manifest
adding: Driver.class(in = 1342) (out= 741)(deflated 44%)
adding: MapperClass.class(in = 1680) (out= 728)(deflated 56%)
adding: ReducerClass.class(in = 1591) (out= 661)(deflated 58%)
```

Figure 36: Compilation and JAR creation

3.5 Step 5: Execute MapReduce Job

```
hadoop jar wordcount.jar Driver /input /output
```

```
hadoop@yugal-Inspiron-3583:~$ hadoop jar wordcount.jar Driver /input /output
2026-02-10 14:49:26,254 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2026-02-10 14:49:26,823 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2026-02-10 14:49:26,869 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hadoop/.staging/job_1770715104976_0002
2026-02-10 14:49:27,964 INFO input.FileInputFormat: Total input files to process : 1
2026-02-10 14:49:28,492 INFO mapreduce.JobSubmitter: number of splits:1
2026-02-10 14:49:29,226 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1770715104976_0002
2026-02-10 14:49:29,226 INFO mapreduce.JobSubmitter: Executing with tokens: []
2026-02-10 14:49:29,497 INFO conf.Configuration: resource-types.xml not found
2026-02-10 14:49:29,498 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2026-02-10 14:49:29,924 INFO impl.YarnClientImpl: Submitted application application_1770715104976_0002
2026-02-10 14:49:30,143 INFO mapreduce.Job: The url to track the job: http://yugal-Inspiron-3583:8088/proxy/application_1770715104976_0002/
2026-02-10 14:49:30,145 INFO mapreduce.Job: Running job: job_1770715104976_0002
2026-02-10 14:49:42,586 INFO mapreduce.Job: Job job_1770715104976_0002 running in uber mode : false
2026-02-10 14:49:42,588 INFO mapreduce.Job:  map 0% reduce 0%
2026-02-10 14:49:49,823 INFO mapreduce.Job:  map 100% reduce 0%
2026-02-10 14:49:57,932 INFO mapreduce.Job:  map 100% reduce 100%
2026-02-10 14:49:58,977 INFO mapreduce.Job: Job job_1770715104976_0002 completed successfully
2026-02-10 14:49:59,128 INFO mapreduce.Job: Counters: 54
  File System Counters
    FILE: Number of bytes read=135
    FILE: Number of bytes written=527713
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=171
    HDFS: Number of bytes written=34
    HDFS: Number of read operations=8
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
    HDFS: Number of bytes read erasure-coded=0
  Job Counters
    Launched map tasks=1
    Launched reduce tasks=1
    Data-local map tasks=1
    Total time spent by all maps in occupied slots (ms)=4923
    Total time spent by all map tasks (ms)=4923
    Total time spent by all reduce tasks (ms)=4834
```

Figure 37: Executing MapReduce job

3.6 Step 6: Check Output Directory in HDFS

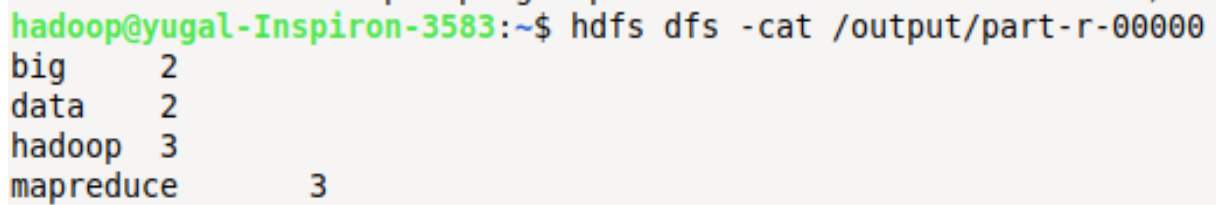
```
hdfs dfs -ls /output
```

```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /output
Found 2 items
-rw-r--r-- 1 hadoop supergroup 0 2026-02-10 14:49 /output/_SUCCESS
-rw-r--r-- 1 hadoop supergroup 34 2026-02-10 14:49 /output/part-r-00000
```

Figure 38: Output directory created in HDFS

3.7 Step 7: Display Result File

```
hdfs dfs -cat /mapreduce_output/part-r-00000
```



```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -cat /output/part-r-00000
big      2
data     2
hadoop   3
mapreduce 3
```

Figure 39: Displaying MapReduce output

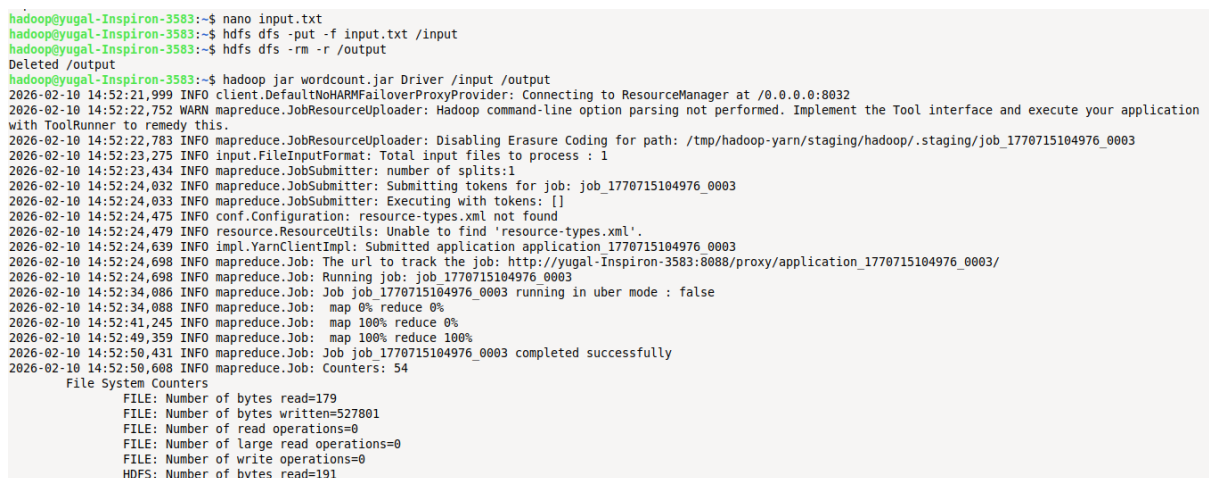
3.8 Step 8: Modify Input and Re-run Job

```
nano input.txt
hdfs dfs -put input.txt /input
hdfs dfs -rm -r /output
hadoop jar wordcount.jar Driver /input /output
hdfs dfs -cat /output/part-r-00000
```



```
GNU nano 7.2 input.txt *
hadoop mapreduce hadoop
big data hadoop mapreduce
mapreduce big data
hadoop big big data
```

Figure 40: Updating input text file



```
hadoop@yugal-Inspiron-3583:~$ nano input.txt
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -put -f input.txt /input
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -rm -r /output
Deleted /output
hadoop@yugal-Inspiron-3583:~$ hadoop jar wordcount.jar Driver /input /output
2026-02-10 14:52:21,999 INFO client.DefaultHARMPFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2026-02-10 14:52:22,752 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application
with ToolRunner to remedy this.
2026-02-10 14:52:22,783 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hadoop/.staging/job_1770715104976_0003
2026-02-10 14:52:23,275 INFO input.FileInputFormat: Total input files to process : 1
2026-02-10 14:52:23,434 INFO mapreduce.JobSubmitter: number of splits:1
2026-02-10 14:52:24,032 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1770715104976_0003
2026-02-10 14:52:24,033 INFO mapreduce.JobSubmitter: Executing with tokens: []
2026-02-10 14:52:24,475 INFO conf.Configuration: resource-types.xml not found
2026-02-10 14:52:24,479 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2026-02-10 14:52:24,639 INFO impl.YarnClientImpl: Submitted application application_1770715104976_0003
2026-02-10 14:52:24,698 INFO mapreduce.Job: The url to track the job: http://yugal-Inspiron-3583:8088/proxy/application_1770715104976_0003/
2026-02-10 14:52:24,698 INFO mapreduce.Job: Running job: job_1770715104976_0003
2026-02-10 14:52:34,086 INFO mapreduce.Job: Job job_1770715104976_0003 running in uber mode : false
2026-02-10 14:52:34,088 INFO mapreduce.Job: map 0% reduce 0%
2026-02-10 14:52:41,245 INFO mapreduce.Job: map 100% reduce 0%
2026-02-10 14:52:49,359 INFO mapreduce.Job: map 100% reduce 100%
2026-02-10 14:52:50,431 INFO mapreduce.Job: Job job_1770715104976_0003 completed successfully
2026-02-10 14:52:50,608 INFO mapreduce.Job: Counters: 54
File System Counters
FILE: Number of bytes read=179
FILE: Number of bytes written=527801
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=191
```

Figure 41: Re-running MapReduce with modified dataset

```
Map output records=14
Map output bytes=145
Map output materialized bytes=179
Input split bytes=102
Combine input records=0
Combine output records=0
Reduce input groups=4
Reduce shuffle bytes=179
Reduce input records=14
Reduce output records=4
Spilled Records=28
Shuffled Maps =1
Failed Shuffles=0
Merged Map outputs=1
GC time elapsed (ms)=100
CPU time spent (ms)=1950
Physical memory (bytes) snapshot=473153536
Virtual memory (bytes) snapshot=5457133568
Total committed heap usage (bytes)=524288000
Peak Map Physical memory (bytes)=283176960
Peak Map Virtual memory (bytes)=2720944128
Peak Reduce Physical memory (bytes)=189976576
Peak Reduce Virtual memory (bytes)=2736189440
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=89
File Output Format Counters
Bytes Written=34
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -cat /output/part-r-00000
big      4
data    3
hadoop   4
mapreduce 3
hadoop@yugal-Inspiron-3583:~$ |
```

Figure 42: New output after re-running MapReduce job

3.9 Result

The MapReduce job was successfully executed on Hadoop. Input data stored in HDFS was processed using Mapper and Reducer classes, and the output was generated in the HDFS output directory. Re-running the job with modified input demonstrated changes in output based on data size and content.

3.10 Conclusion

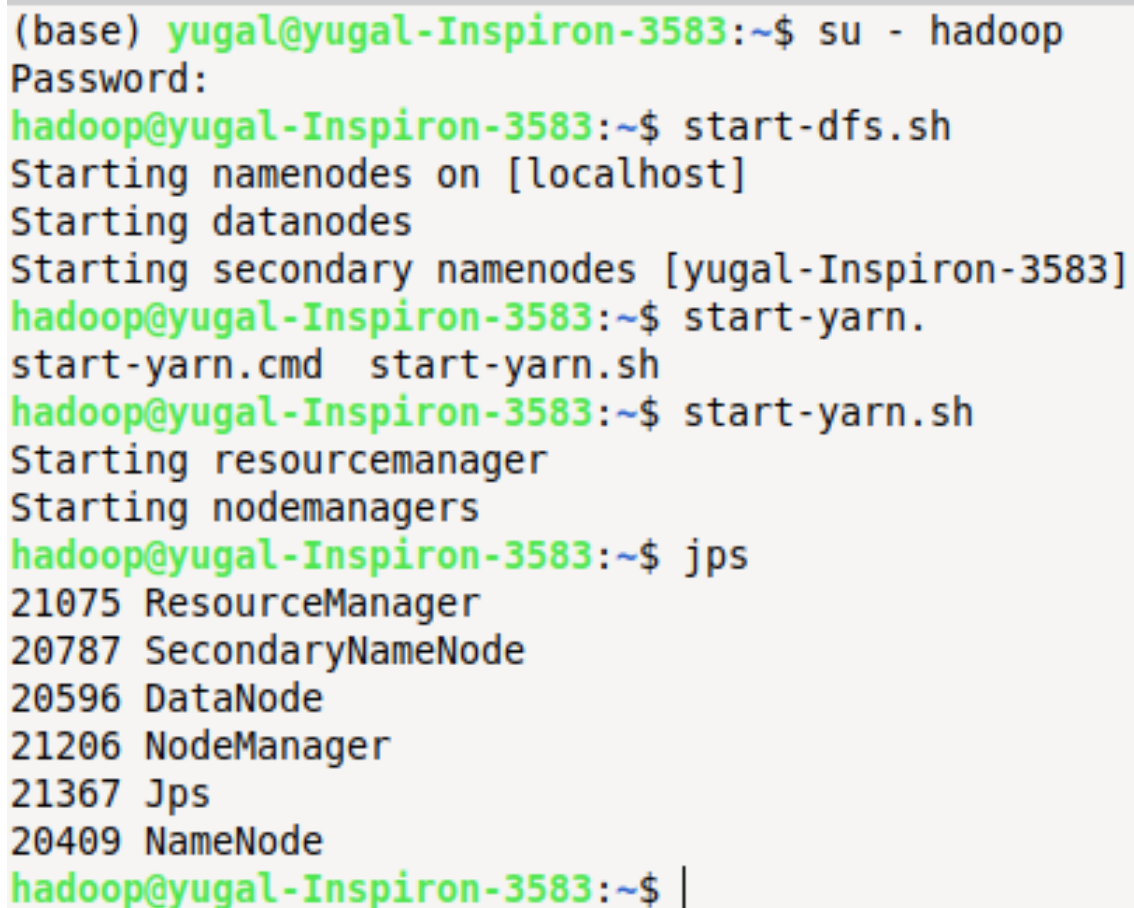
This assignment demonstrated the practical implementation of the MapReduce programming model in Hadoop. It showed how distributed computing enables scalable data processing by dividing tasks into mapping and reducing phases, improving efficiency for large datasets.

Assignment 4: Implement Data Processing using Spark and Compare with MapReduce

Objective: Develop in-memory analytics capability by implementing the same data processing task using Apache Spark and comparing performance with MapReduce.

4.1 Step 1: Start Hadoop Services

```
su - hadoop
start-dfs.sh
start-yarn.sh
jps
```

A terminal window screenshot showing the process of starting Hadoop services. The user switches to the 'hadoop' user and runs 'start-dfs.sh', which starts namenodes and datanodes. Then 'start-yarn.sh' is run, starting the ResourceManager and NodeManagers. Finally, 'jps' is used to verify the running processes, listing ResourceManager, SecondaryNameNode, DataNode, NodeManager, Jps, and NameNode.

```
(base) yugal@yugal-Inspiron-3583:~$ su - hadoop
Password:
hadoop@yugal-Inspiron-3583:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [yugal-Inspiron-3583]
hadoop@yugal-Inspiron-3583:~$ start-yarn.
start-yarn.cmd start-yarn.sh
hadoop@yugal-Inspiron-3583:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@yugal-Inspiron-3583:~$ jps
21075 ResourceManager
20787 SecondaryNameNode
20596 DataNode
21206 NodeManager
21367 Jps
20409 NameNode
hadoop@yugal-Inspiron-3583:~$ |
```

Figure 43: Starting Hadoop services and verifying using jps

4.2 Step 2: Verify Input File in HDFS

The same dataset used in MapReduce is reused for Spark processing.

```
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /
Found 4 items
drwxr-xr-x   - hadoop supergroup          0 2026-01-28 01:01 /data
drwxr-xr-x   - hadoop supergroup          0 2026-02-10 14:52 /input
drwxr-xr-x   - hadoop supergroup          0 2026-02-10 14:52 /output
drwx-----  - hadoop supergroup          0 2026-02-10 14:44 /tmp
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -ls /input
Found 1 items
-rw-r--r--   1 hadoop supergroup          89 2026-02-10 14:52 /input/input.txt
hadoop@yugal-Inspiron-3583:~$ hdfs dfs -cat /input/input.txt
hadoop mapreduce hadoop
big data hadoop mapreduce
mapreduce big data
hadoop big big data
hadoop@yugal-Inspiron-3583:~$ |
```

4.3 Step 3: Start Spark Shell

Spark shell starts successfully and provides the Scala prompt for interactive execution.

Figure 45: Launching Spark Shell

The dataset stored in HDFS is loaded into an RDD for processing.


```
scala> val lines = sc.textFile("hdfs://localhost:9000/input/input.txt")
lines: org.apache.spark.rdd.RDD[String] = hdfs://localhost:9000/input/input.txt MapPartitionsRDD[1] at textFile at <console>:23
scala>
```

Figure 46: Reading input data from HDFS

4.5 Step 5: Implement Spark Transformations

Word Count is implemented using Spark transformations.

```
val words = lines.flatMap(line => line.split(" "))
val wordPairs = words.map(word => (word, 1))
val counts = wordPairs.reduceByKey((a,b) => a+b)
```

flatMap() splits lines into words, **map()** creates key-value pairs, and **reduceByKey()** aggregates counts.

```
scala> val words = lines.flatMap(line => line.split(" "))
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at flatMap at <console>:23

scala> val wordPairs = words.map(word => (word, 1))
wordPairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at map at <console>:23

scala> val counts = wordPairs.reduceByKey((a,b) => a+b)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:23
```

Figure 47: Spark transformations for Word Count

4.6 Step 6: Apply Actions and Display Output

```
counts.collect()
```

Output obtained:

```
(big,4)
(data,3)
(mapreduce,3)
(hadoop,4)
```

```
scala> counts.collect()
res2: Array[(String, Int)] = Array((big,4), (data,3), (mapreduce,3), (hadoop,4))
```

Figure 48: Displaying results using collect()

4.7 Step 7: Save Output to HDFS

```
counts.saveAsTextFile("hdfs://localhost:9000/output_spark")
```

The processed output is stored in HDFS similar to MapReduce output.

```
scala> counts.saveAsTextFile("hdfs://localhost:9000/output_spark")
scala> :quit|
```

Figure 49: Saving Spark output to HDFS

4.8 Step 8: Check Output Directory and Validate Result

```
hdfs dfs -ls /
hdfs dfs -cat /output_spark/part-*
```

The output matches the MapReduce results, validating correctness.

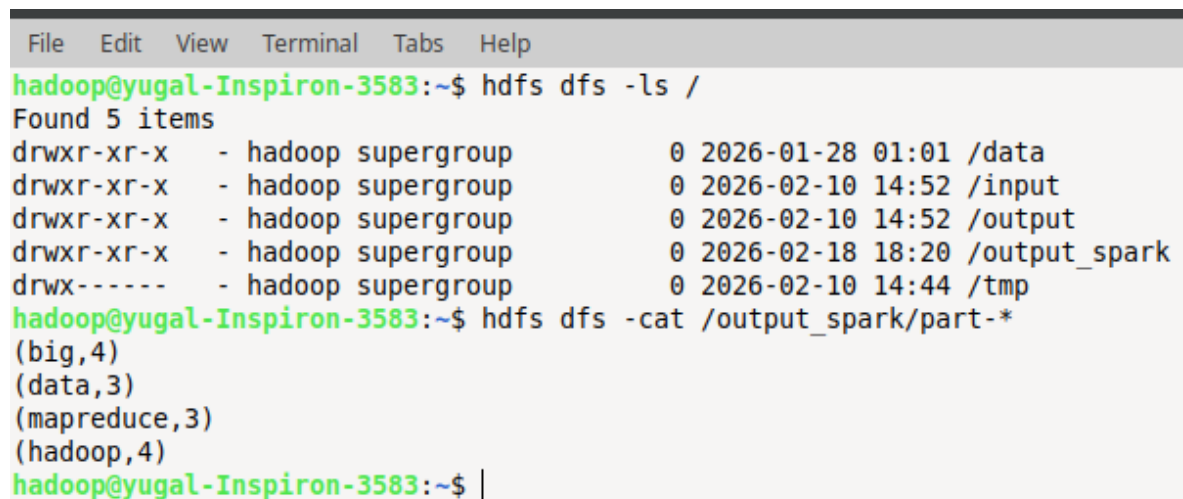
A terminal window with a menu bar (File, Edit, View, Terminal, Tabs, Help) and a title bar (hadoop@yugal-Inspiron-3583:~\$). The terminal shows the command 'hdfs dfs -ls /' and its output, which lists five items: /data, /input, /output, /output_spark, and /tmp. The permissions for /data, /input, /output, and /output_spark are 'drwxr-xr-x', while for /tmp it is 'drwx-----'. The user is 'hadoop' and the group is 'supergroup'. The timestamps are 2026-01-28 01:01 for /data, 2026-02-10 14:52 for /input, /output, and /tmp, and 2026-02-18 18:20 for /output_spark. The next command is 'hdfs dfs -cat /output_spark/part-*', and the output is '(big,4)', '(data,3)', '(mapreduce,3)', and '(hadoop,4)'. The prompt returns to 'hadoop@yugal-Inspiron-3583:~\$ |'.

Figure 50: Validating Spark output in HDFS

4.9 Step 9: Stop Spark and Hadoop Services

```
:quit
stop-yarn.sh
stop-dfs.sh
```

```

hadoop@yugal-Inspiron-3583:~$ stop-yarn.sh
stop-dfs.sh
Stopping nodemanagers
Stopping resourcemanager
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [yugal-Inspiron-3583]
hadoop@yugal-Inspiron-3583:~$ |

```

Figure 51: Stopping Spark session and Hadoop services

4.10 Step 10: Comparison with MapReduce

Feature	MapReduce	Spark
Processing Model	Disk-based	In-memory
Execution Speed	Slower	Faster
Code Complexity	More lines of code	Simple and concise
Iterative Processing	Inefficient	Efficient
Performance	Higher latency	Low latency

Table 1: Comparison between MapReduce and Spark

4.11 Result

The Spark application successfully processed the input dataset using in-memory computation. Word count results were generated and stored in HDFS, matching the output produced by the MapReduce implementation.

4.12 Conclusion

This assignment demonstrated in-memory analytics using Apache Spark. Compared to MapReduce, Spark required less code and executed faster due to memory-based processing. The experiment highlights Spark's suitability for high-performance analytics and iterative big data processing tasks.