HINDUSTHAN COLLEGE OF ARTS & SCIENCE

(Autonomous)

An Autonomous Institution – Affiliated to Bharathiar University

(ISO 9001 – 2001 Certificate Instituation)

Behind Nava India, Coimbatore – 641028.

DEPARTMENT OF COMPUTER APPLICATIONS (PG)

MASTER OF COMPUTER APPLICATIONS

PRACTICAL RECORD

23MCP24 – PRACTICAL: BIG DATA ANALYTICS

NAME : ______________________________

REGISTER NO : ______________________________

CLASS : ______________________________

SEMESTER : ______________________________

YEAR : ______________________________

HINDUSTHAN COLLEGE OF ARTS & SCIENCE

(Autonomous)

An Autonomous Institution – Affiliated to Bharathiar University

(ISO 9001 – 2001 Certificate Instituation)

Behind Nava India, Coimbatore – 641028.

DEPARTMENT OF COMPUTER APPLICATIONS (PG)

CERTIFICATE

Certificate that this is a bonafide record of **Big Data Analytics (23MCP24)** done by ______________________________ Register No: ________________ during the academic year of 2024-2025.

STAFF-IN CHARGE **DIRECTOR**

Submitted for the Bharathiar University Practical Examination held on ________________ at Hindusthan College of Arts & Science, Coimbatore – 641028.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Date:

Place: Coimbatore

CONTENTS

PROGRAM NO: 01 **DATE:**	**Installation of Hadoop**	**PAEG NO:**

AIM:

SETTING UP AND INSTALLING HADOOP:

Prerequisites:

1. Install Java Development Kit (JDK):

- Hadoop requires Java, so ensure that JDK is installed. Download it from Oracle or OpenJDK.
- Set the JAVA_HOME environment variable.

Steps:

- Download and install JDK 11 or later.
- Set JAVA_HOME:
  - Open Control Panel > System > Advanced system settings.
  - Click Environment Variables.
  - In the System variables section, click New and add:
    - **Variable Name:** JAVA_HOME
    - **Variable Value:** C:\Program Files\Java\jdk-11
  - Also, add Java to Path by editing the Path variable and adding %JAVA_HOME%\bin.

2. Install WinRAR or 7-Zip:

- To extract the Hadoop binary package you will download later.

3. Install SSH (Optional for pseudo/fully distributed mode):

- You will need an SSH client (such as PuTTY) for fully distributed setups if multiple machines are involved. However, it’s optional for single-machine setups.

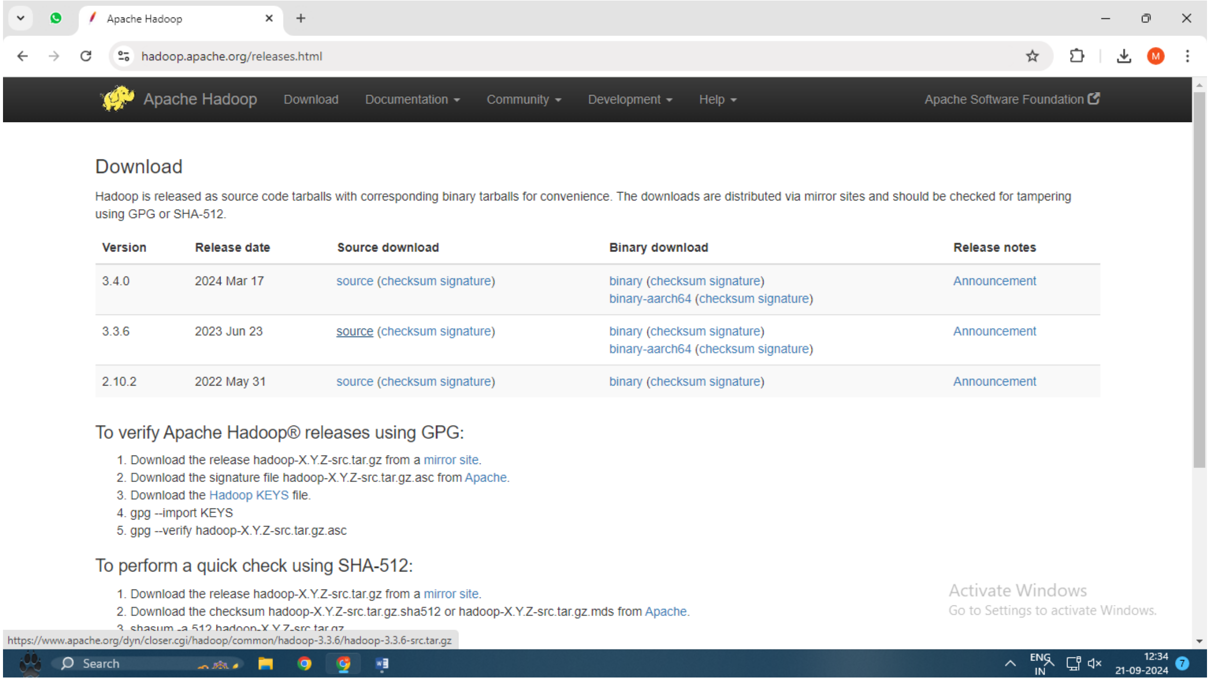
1. Standalone Mode

Standalone mode requires no Hadoop daemon services (such as NameNode or DataNode). It runs directly on the local filesystem.

Steps:

1. Download Hadoop:

- Download the binary release for Hadoop from Apache Hadoop Releases.
- Extract the .tar.gz file to a directory like C:\hadoop.



Download

Hadoop is released as source code tarballs with corresponding binary tarballs for convenience. The downloads are distributed via mirror sites and should be checked for tampering using GPG or SHA-512.

Version	Release date	Source download	Binary download	Release notes
3.4.0	2024 Mar 17	source (checksum signature)	binary (checksum signature) binary-aarch64 (checksum signature)	Announcement
3.3.6	2023 Jun 23	source (checksum signature)	binary (checksum signature) binary-aarch64 (checksum signature)	Announcement
2.10.2	2022 May 31	source (checksum signature)	binary (checksum signature)	Announcement

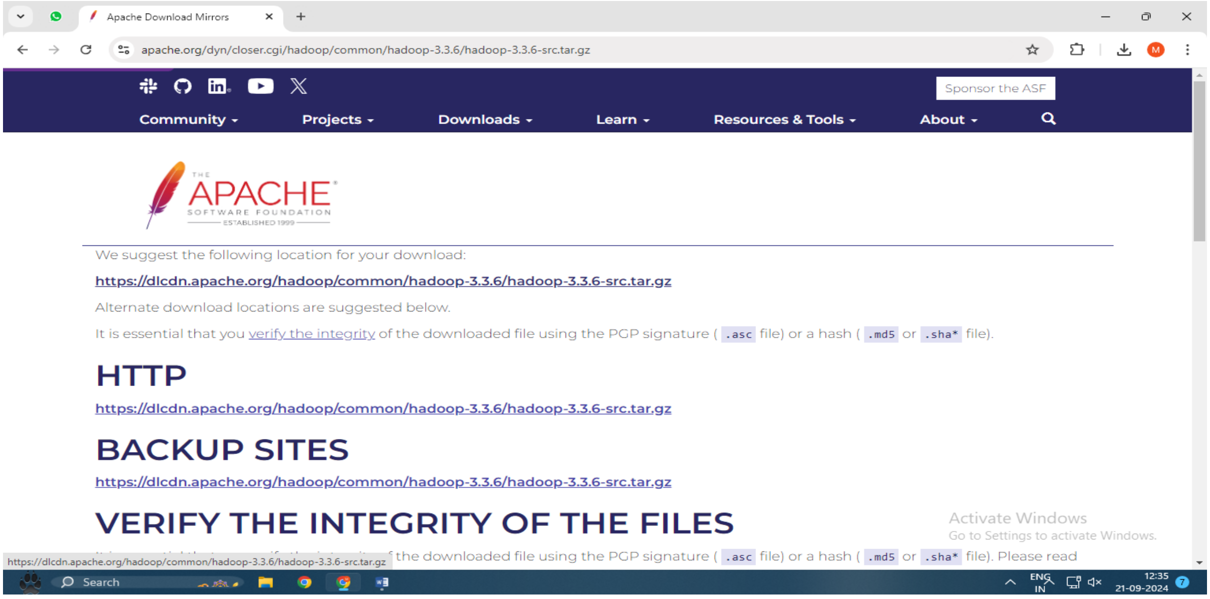
To verify Apache Hadoop® releases using GPG:

1. Download the release hadoop-X.Y.Z-src.tar.gz from a mirror site.
2. Download the signature file hadoop-X.Y.Z-src.tar.gz.asc from Apache.
3. Download the Hadoop KEYS file.
4. gpg --import KEYS
5. gpg --verify hadoop-X.Y.Z-src.tar.gz.asc

To perform a quick check using SHA-512:

1. Download the release hadoop-X.Y.Z-src.tar.gz from a mirror site.
2. Download the checksum hadoop-X.Y.Z-src.tar.gz.sha512 or hadoop-X.Y.Z-src.tar.gz.mds from Apache.
3. shasum -a 512 hadoop-X.Y.Z-src.tar.gz

https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-3.3.6/hadoop-3.3.6-src.tar.gz



We suggest the following location for your download:

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

Alternate download locations are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file).

HTTP

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

BACKUP SITES

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

VERIFY THE INTEGRITY OF THE FILES

the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file). Please read

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

2. Configure Environment Variables: Add Hadoop to the system's PATH.

- Open Control Panel > System > Advanced system settings > Environment Variables.
- Add the following variables:
  - **HADOOP_HOME:** C:\hadoop
  - Edit the Path variable and add %HADOOP_HOME%\bin.

3. Test Installation: Open a new command prompt and run:

```
hadoop version
```

4. Running a MapReduce Job: You can run a sample MapReduce job in standalone mode:

```
hadoop jar %HADOOP_HOME%

\share\hadoop\mapreduce\hadoop-mapreduce-examples-*.jar wordcount input output
```

2. Pseudo-Distributed Mode

Pseudo-distributed mode runs all of Hadoop's daemons (NameNode, DataNode, ResourceManager, NodeManager) on a single machine, but simulates a distributed cluster.

Steps:

1. Configure Hadoop: You'll need to modify several XML configuration files located in the C:\hadoop\etc\hadoop directory.

2. Configure core-site.xml: Modify the file C:\hadoop\etc\hadoop\core-site.xml:

```
<configuration>

  <property>

    <name>fs.defaultFS</name>

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

  </property>

</configuration>
```

3. Configure hdfs-site.xml: Modify the file C:\hadoop\etc\hadoop\hdfs-site.xml:

```
<configuration>
```

```
  <property>
    <name>dfs.replication</name>
    <value>1</value> <!-- Since it's running on a single machine -->
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/C:/hadoop_data/hdfs/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/C:/hadoop_data/hdfs/datanode</value>
  </property>
</configuration>
```

4. Configure mapred-site.xml: Modify the file C:\hadoop\etc\hadoop\mapred-site.xml (first copy it from the template):

```
cp C:\hadoop\etc\hadoop\mapred-site.xml.template C:\hadoop\etc\hadoop\mapred-site.xml
```

Then, edit the following:

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

5. Configure yarn-site.xml: Modify the file C:\hadoop\etc\hadoop\yarn-site.xml:

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

6. Format the NameNode: Open a command prompt and run:

```
hdfs namenode -format
```

7. Start Hadoop Daemons: Run the following commands to start the Hadoop daemons:

```
start-dfs.cmd
start-yarn.cmd
```

8. Test HDFS: Verify that HDFS is running correctly:

```
hdfs dfs -mkdir /user
hdfs dfs -ls /
```

3. Fully Distributed Mode

Fully distributed mode is where Hadoop runs on multiple machines (master and worker nodes). For this, you'll need to configure Hadoop on each machine and ensure proper communication between them.

Steps:

1. Master-Slave Setup:

- Set up Master Node (NameNode) on one machine and Slave Nodes (DataNodes) on other machines.
- SSH setup for passwordless login between master and slave nodes may be required (for cross-machine communication).

2. Configure core-site.xml on all machines: On the master node and all slave nodes, configure **C:\hadoop\etc\hadoop\core-site.xml:**

```
<configuration>

  <property>

    <name>fs.defaultFS</name>

    <value>hdfs://master-node-ip:9000</value>

  </property>

</configuration>
```

3. Configure hdfs-site.xml: On all machines, configure **C:\hadoop\etc\hadoop\hdfs-site.xml:**

```
<configuration>

  <property>

    <name>dfs.replication</name>

    <value>3</value>

  </property>

</configuration>
```

4. Configure workers file on Master Node: In the file **C:\hadoop\etc\hadoop\workers,** list all the slave nodes:

```
slave-node1

slave-node2
```

5. Set Up SSH and Communication: Set up SSH for passwordless communication between the master and the slave nodes.

6. Format the NameNode: On the master node, run:

```
hdfs namenode -format
```

7. Start Hadoop Daemons on Master and Slave Nodes: On the master node, start the services:

```
start-dfs.cmd

start-yarn.cmd
```

8. Verify Hadoop Cluster: On the master node, check the cluster status:

```
hdfs dfsadmin -report
```

RESULT:

PROGRAM NO: 02 **DATE:**	**File Management in Hadoop**	**PAEG NO:**

AIM:

ALGORITHM:

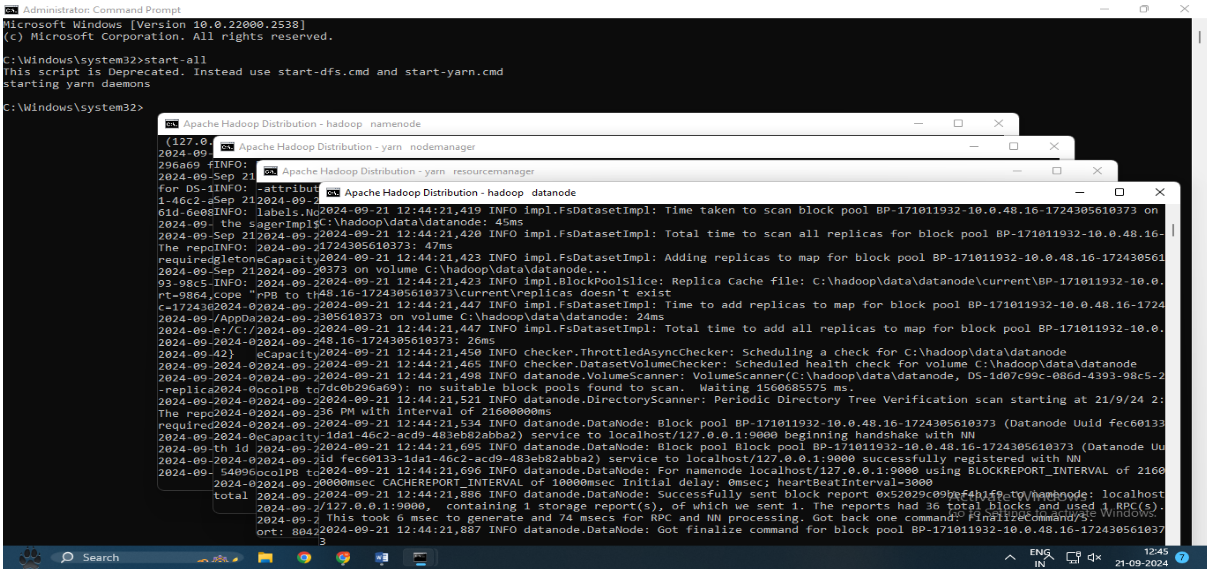
PROGRAM:

Start Hadoop Daemons:

C:\Windows\system32>start-all

This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd

starting yarn daemons



Administrator: Command Prompt
Apache Hadoop Distribution - hadoop namenode
Apache Hadoop Distribution - yarn nodemanager
Apache Hadoop Distribution - yarn resourcemanager
Apache Hadoop Distribution - hadoop datanode
Search
ENG IN
12:45
21-09-2024

```
Microsoft Windows [Version 10.0.22000.2538]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>start-all
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\Windows\system32>
```

```
2024-09-21 12:44:21,419 INFO impl.FsDatasetImpl: Time taken to scan block pool BP-171011932-10.0.48.16-1724305610373 on
C:\hadoop\data\datanode: 45ms
2024-09-21 12:44:21,420 INFO impl.FsDatasetImpl: Total time to scan all replicas for block pool BP-171011932-10.0.48.16-
1724305610373: 47ms
2024-09-21 12:44:21,423 INFO impl.FsDatasetImpl: Adding replicas to map for block pool BP-171011932-10.0.48.16-172430561
0373 on volume C:\hadoop\data\datanode...
2024-09-21 12:44:21,423 INFO impl.BlockPoolSlice: Replica Cache file: C:\hadoop\data\datanode\current\BP-171011932-10.0.
48.16-1724305610373\current\replicas doesn't exist
2024-09-21 12:44:21,447 INFO impl.FsDatasetImpl: Time to add replicas to map for block pool BP-171011932-10.0.48.16-1724
305610373 on volume C:\hadoop\data\datanode: 24ms
2024-09-21 12:44:21,447 INFO impl.FsDatasetImpl: Total time to add all replicas to map for block pool BP-171011932-10.0.
48.16-1724305610373: 26ms
2024-09-21 12:44:21,450 INFO checker.ThrottledAsyncChecker: Scheduling a check for C:\hadoop\data\datanode
2024-09-21 12:44:21,465 INFO checker.DatasetVolumeChecker: Scheduled health check for volume C:\hadoop\data\datanode
2024-09-21 12:44:21,498 INFO datanode.VolumeScanner: VolumeScanner(C:\hadoop\data\datanode, DS-1d07c99c-086d-4393-98c5-2
7dc0b296a69): no suitable block pools found to scan.  Waiting 1560685575 ms.
2024-09-21 12:44:21,521 INFO datanode.DirectoryScanner: Periodic Directory Tree Verification scan starting at 21/9/24 2:
36 PM with interval of 21600000ms
2024-09-21 12:44:21,534 INFO datanode.DataNode: Block pool BP-171011932-10.0.48.16-1724305610373 (Datanode Uuid fec60133
-1da1-46c2-acd9-483eb82abba2) service to localhost/127.0.0.1:9000 beginning handshake with NN
2024-09-21 12:44:21,695 INFO datanode.DataNode: Block pool Block pool BP-171011932-10.0.48.16-1724305610373 (Datanode Uu
id fec60133-1da1-46c2-acd9-483eb82abba2) service to localhost/127.0.0.1:9000 successfully registered with NN
2024-09-21 12:44:21,696 INFO datanode.DataNode: For namenode localhost/127.0.0.1:9000 using BLOCKREPORT_INTERVAL of 2160
0000msec CACHEREPORT_INTERVAL of 10000msec Initial delay: 0msec; heartBeatInterval=3000
2024-09-21 12:44:21,886 INFO datanode.DataNode: Successfully sent block report 0x52029c09bef4b1f9 to namenode: localhost
/127.0.0.1:9000,  containing 1 storage report(s), of which we sent 1. The reports had 36 total blocks and used 1 RPC(s).
 This took 6 msec to generate and 74 msecs for RPC and NN processing. Got back one command: FinalizeCommand/5.
2024-09-21 12:44:21,887 INFO datanode.DataNode: Got finalize command for block pool BP-171011932-10.0.48.16-17243056103
3
```

C:\Windows\system32>jps

2832 ResourceManager

10328 DataNode

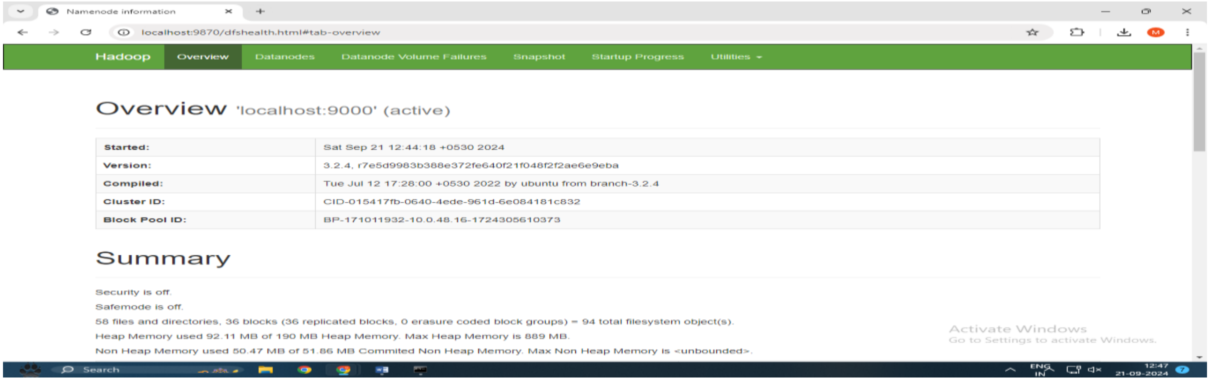
12936 NodeManager

6504 NameNode

10012 Jps

ENABLE HADOOP INTERFACE

Open Browser and enable “**localhost:9870**”



Overview 'localhost:9000' (active)

Started:	Sat Sep 21 12:44:18 +0530 2024
Version:	3.2.4, r7e5d9983b388e372fe640f21f048f2f2ae6e9eba
Compiled:	Tue Jul 12 17:28:00 +0530 2022 by ubuntu from branch-3.2.4
Cluster ID:	CID-015417fb-0640-4ede-961d-6e084181c832
Block Pool ID:	BP-171011932-10.0.48.16-1724305610373

Summary

Security is off.

Safemode is off.

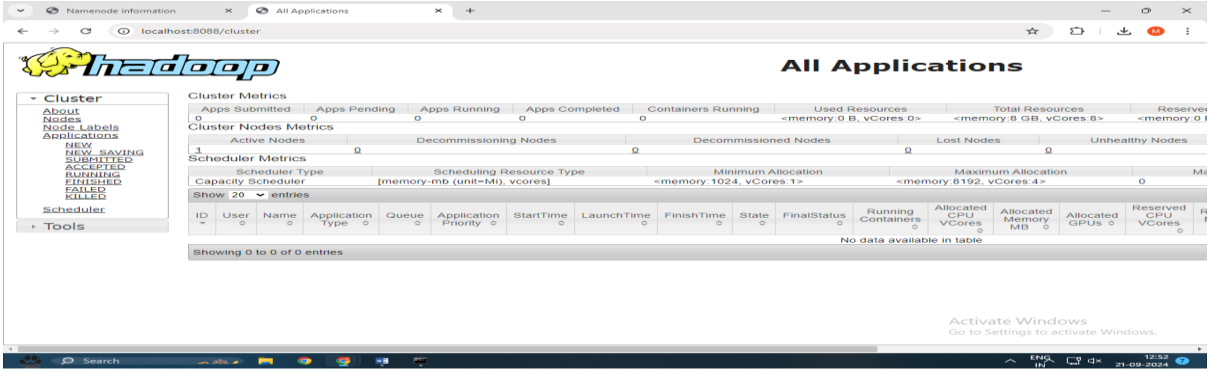
58 files and directories, 36 blocks (36 replicated blocks, 0 erasure coded block groups) = 94 total filesystem object(s).

Heap Memory used 92.11 MB of 190 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 50.47 MB of 51.86 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

ENABLE HADOOP ALL APPLICATION

Open Browser and enable “**localhost:8088**”



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserve
0	0	0	0	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 I

Cluster Nodes Metrics

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

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Ma
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allocated GPUs	Reserved CPU VCores	R
No data available in table																

Showing 0 to 0 of 0 entries

Adding files and directories:

1. **Directory**

C:\Windows\system32>hdfs dfs -mkdir /HADOOP

2. **Files**

C:\Windows\system32>hdfs dfs -put D:\example.txt /HADOOP

Retrieving files:

C:\Windows\system32>hdfs dfs -get /HADOOP /example.txt D:\

get: `/example.txt': No such file or directory

View Data from the file:

C:\Windows\system32>hdfs dfs -cat /HADOOP/example.txt

hi

hello everyone

how are you?

Deleting files and directories:

1. **File**

C:\Windows\system32>hdfs dfs -rm -r /HADOOP/example.txt

Deleted /HADOOP/example.txt

2. **Directory**

C:\Windows\system32>hdfs dfs -rm -r /HADOOP

Deleted /HADOOP

RESULT:

PROGRAM NO: 03 **DATE:**	**Word Count using MapReduce Paradigm**	**PAEG NO:**

AIM:

ALGORITHM:

PROGRAM:

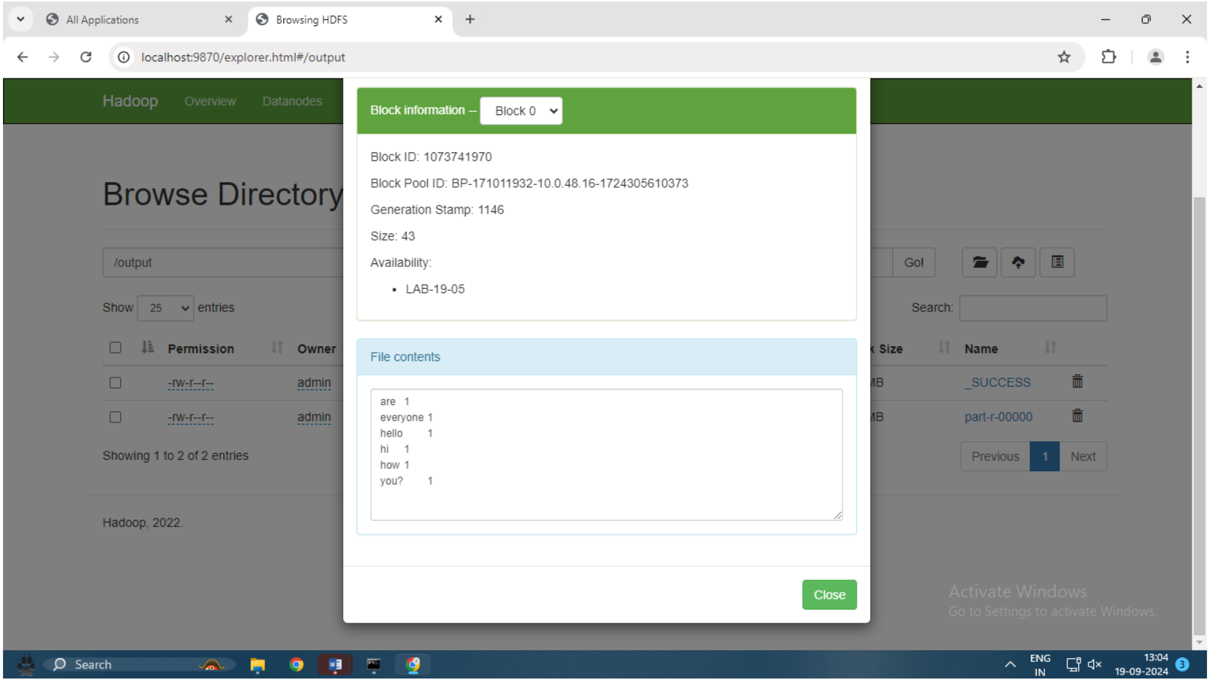
example.txt

```
hi
hello everyone
how are you?
```

Command:

```
C:\Windows\system32>hdfs dfs -mkdir /ex3
C:\Windows\system32>hdfs dfs -put D:\example.txt /ex3
C:\Windows\system32>hadoop jar C:\hadoop\share\hadoop\mapreduce\hadoop-mapreduce-examples-3.2.4.jar wordcount /ex3/example.txt /output
```

OUTPUT:



Block ID: 1073741970

Block Pool ID: BP-171011932-10.0.48.16-1724305610373

Generation Stamp: 1146

Size: 43

Availability:

- LAB-19-05

```
are	1
everyone	1
hello	1
hi	1
how	1
you?	1
```

RESULT:

PROGRAM NO: 04 **DATE:**	**Weather Report using MapReduce**	**PAEG NO:**

AIM:

ALGORITHM:

PROGRAM:

weather_data.txt

```
2024-08-01,Rainy

2024-08-02,Sunny

2024-08-03,Rainy

2024-08-04,Cloudy
```

weather_mapper.py

```
import sys
for line in sys.stdin:
    line=line.strip()
    date, weather=line.split(',')
    print(f"{weather}\t1")
```

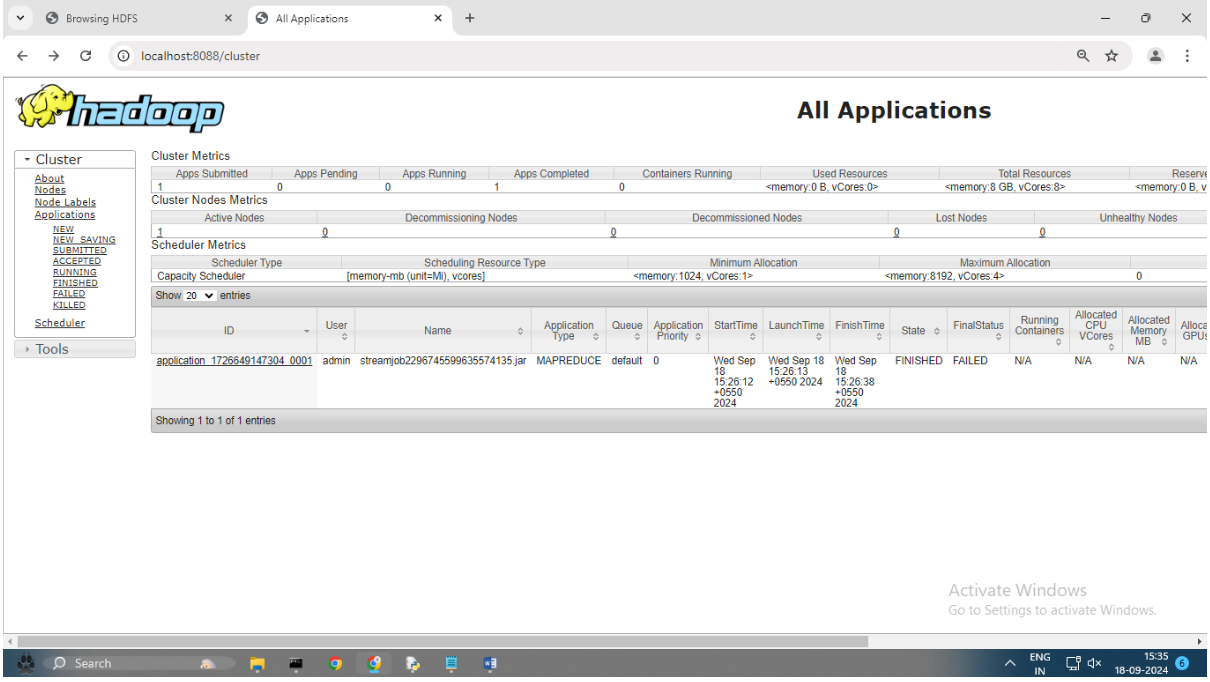
weather_reducere.py

```
import sys
current_weather=None
current_count=0

for line in sys.stdin:
    line=line.strip()
    weather, count=line.split('\t',1)
    count=int(count)

    if current_weather == weather:
        count_count += count

    else:
        if current_weather:
            print(f"{current_weather}\t{current_count}")
        current_weather = weather
        current_count = count

if current_weather == weather:
    print(f"{current_weather}\t{current_count}")
```

Command:

```
C:\Windows\system32>hdfs dfs -mkdir /wc
C:\Windows\system32>hdfs dfs -put D:\weather_data.txt /wc
C:\Windows\system32>hdfs dfs -put D:\weather_mapper.py /wc
C:\Windows\system32>hdfs dfs -put D:\weather_reducer.py /wc
C:\Windows\system32>hadoop jar C:\hadoop\share\hadoop\tools\lib\hadoop-streaming-3.2.4.jar ^
More?   -input /wc/weather_data.txt ^
More?   -output /weather_output ^
More?   -mapper /weather_mapper.py ^
More?   -reducer /weather_reducer.py
```

OUTPUT:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserve
1	0	0	1	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 B, v

Cluster Nodes Metrics

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

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Alloca GPUs
application_1726649147304_0001	admin	streamjob2296745599635574135.jar	MAPREDUCE	default	0	Wed Sep 18 15:26:12 +0550 2024	Wed Sep 18 15:26:13 +0550 2024	Wed Sep 18 15:26:38 +0550 2024	FINISHED	FAILED	N/A	N/A	N/A	N/A

Showing 1 to 1 of 1 entries

RESULT:

PROGRAM NO: 05 **DATE:**	**Matrix Multiplication using MapReduce**	**PAEG NO:**

AIM:

ALGORITHM:

PROGRAM:

MatrixA.txt

```
1,2,3
4,5,6
```

MatrixB.txt

```
7,8
9,10
11,12
```

matrix_mapper.py

```
import sys
for line in sys.stdin:
    line = line.strip()
    elements = line.split()
    if elements[0] == "A":
        print(f"{elements[1]}\t{elements[2]}\t{elements[3]}\tA")
    else:
        print(f"{elements[2]}\t{elements[1]}\t{elements[3]}\tB")
```

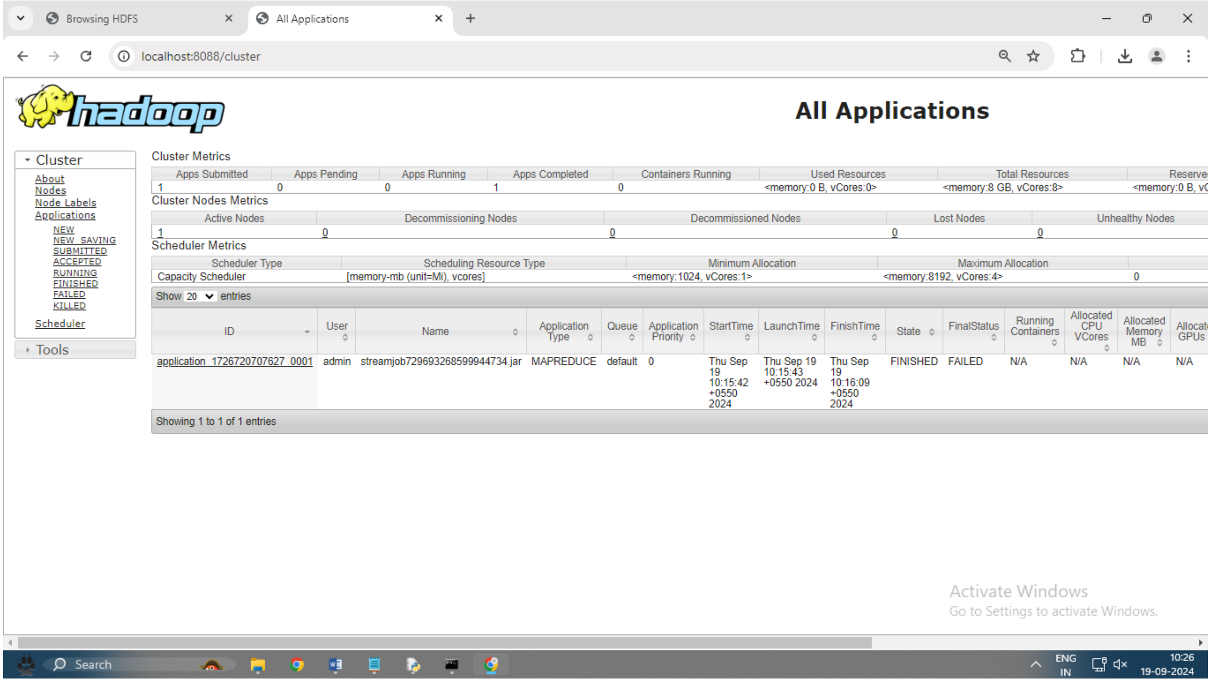
matrix_reducer.py

```
import sys
from collections import defaultdict

MatrixA = defaultdict(list)
MatrixB = defaultdict(list)s

for line in sys.stdin:
    line = line.strip()
    i, j, value, Matrix = line.split()
    if Matrix == "A":
        MatrixA[int(i)].append((int(j), int(value)))
    else:
        MatrixB[int(j)].append((int(i), int(value)))

for i in MatrixA:
    for a_col, a_value in MatrixA[i]:
        for b_row, b_value in MatrixB[a_col]:
            print(f"{i},{b_row}\t{a_value * b_value}")
```

Command:

```
C:\Windows\system32>hdfs dfs -mkdir /ex5

C:\Windows\system32>hdfs dfs -put D:\MatrixA.txt /ex5

C:\Windows\system32>hdfs dfs -put D:\MatrixB.txt /ex5

C:\Windows\system32>hdfs dfs -put D:\matrix_mapper.py /ex5

C:\Windows\system32>hdfs dfs -put D:\matrix_reducer.py /ex5

C:\Windows\system32>hadoop jar C:\hadoop\share\hadoop\tools\lib\hadoop-streaming-3.2.4.jar ^

More?   -input /ex5/MatrixA.txt ^

More?   -input /ex5/MatrixB.txt ^

More?   -output /Matrix_output ^

More?   -mapper /matrix_mapper.py ^

More?   -reducer /matrix_reducer.py
```

OUTPUT:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserve
1	0	0	1	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 B, v

Cluster Nodes Metrics

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

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allocate GPUs
application_1726720707627_0001	admin	streamjob7296932685999447344.jar	MAPREDUCE	default	0	Thu Sep 19 10:15:42 +0550 2024	Thu Sep 19 10:15:43 +0550 2024	Thu Sep 19 10:16:09 +0550 2024	FINISHED	FAILED	N/A	N/A	N/A	N/A

Showing 1 to 1 of 1 entries

RESULT:

PROGRAM NO: 06 **DATE:**	**Sales Data Report using MapReduce**	**PAEG NO:**

AIM:

ALGORITHM:

PROGRAM:

sales_data.txt

```
USA,100
India,200
USA,150
UK,50
```

sales_mapper.py

```python
import sys
for line in sys.stdin:
    line = line.strip()
    country, sales = line.split(',')
    print(f"{country}\t{sales}")
```

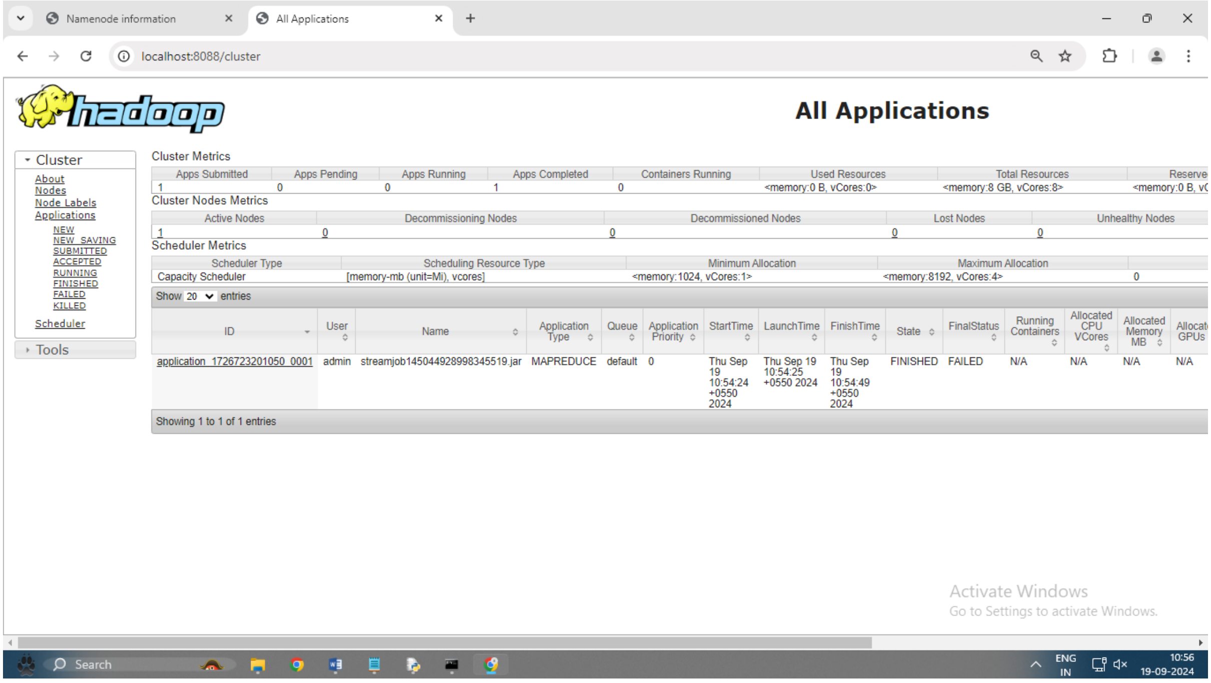
sales_reducer.py

```python
import sys
current_country = None
current_sales = 0

for line in sys.stdin:
    line = line.strip()
    country, sales = line.split('\t')
    sales = int(sales)

    if current_country == country:
        current_sales += sales
    else:
        if current_country:
            print(f"{current_country}\t{current_sales}")
        current_country = country
        current_sales = sales

if current_country == country:
    print(f"{current_country}\t{current_sales}")
```

Command:

```
C:\Windows\system32>hdfs dfs -mkdir /ex6

C:\Windows\system32>hdfs dfs -put D:\sales_data.txt /ex6

C:\Windows\system32>hdfs dfs -put D:\sales_mapper.py /ex6

C:\Windows\system32>hdfs dfs -put D:\sales_reducer.py /ex6

C:\Windows\system32>hadoop jar C:\hadoop\share\hadoop\tools\lib\hadoop-streaming-3.2.4.jar ^

More?   -input /ex6/sales_data.txt ^

More?   -output /sales_output ^

More?   -mapper /sales_mapper.py ^

More?   -reducer /sales_reducer.py
```

OUTPUT:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserve
1	0	0	1	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 B, vC

Cluster Nodes Metrics

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

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allocat GPUs
application_1726723201050_0001	admin	streamjob14504492899834519.jar	MAPREDUCE	default	0	Thu Sep 19 10:54:24 +0550 2024	Thu Sep 19 10:54:25 +0550 2024	Thu Sep 19 10:54:49 +0550 2024	FINISHED	FAILED	N/A	N/A	N/A	N/A

Showing 1 to 1 of 1 entries

RESULT:

PROGRAM NO: 07 **DATE:**	**Electrical Consumption Report using MapReduce**	**PAEG NO:**

AIM:

ALGORITHM:

PROGRAM:

electricity.txt

```
2020,5000
2020,6000
2021,5500
2021,7000
```

electricity_mapper.py

```
import sys
for line in sys.stdin:
    line = line.strip()
    year, consumption = line.split(',')
    print(f"{year}\t{consumption}")
```

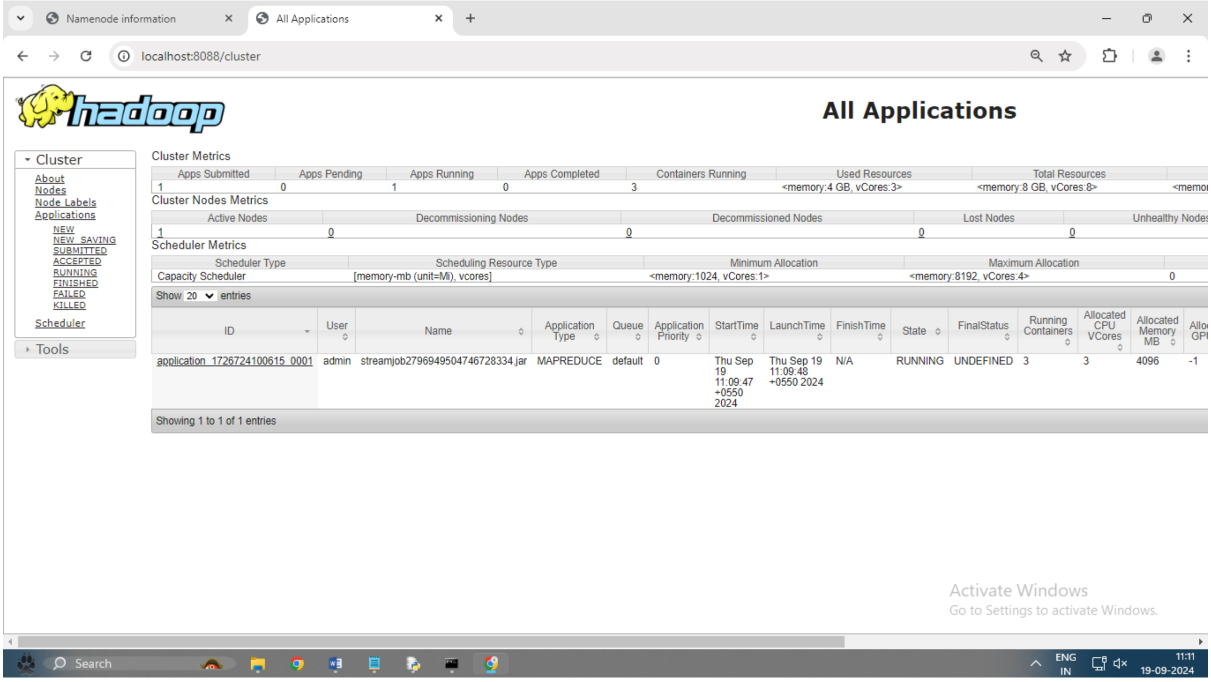
electricity_reducer.py

```
import sys
current_year = None
max_consumption = 0

for line in sys.stdin:
    line = line.strip()
    year, consumption = line.split('\t')
    consumption = int(consumption)

    if current_year == year:
        if consumption > max_consumption:
            max_consumption = consumption
    else:
        if current_year:
            print(f"{current_year}\t{max_consumption}")
        current_year = year
        max_consumption = consumption

if current_year == year:
    print(f"{current_year}\t{max_consumption}")
```

Command:

```
C:\Windows\system32>hdfs dfs -mkdir /ex7
C:\Windows\system32>hdfs dfs -put D:\electricity.txt /ex7
C:\Windows\system32>hdfs dfs -put D:\electricity_mapper.py /ex7
C:\Windows\system32>hdfs dfs -put D:\electricity_reducer.py /ex7
C:\Windows\system32>hadoop jar C:\hadoop\share\hadoop\tools\lib\hadoop-streaming-3.2.4.jar ^
More?   -input /ex7/electricity.txt ^
More?   -output /electricity_output ^
More?   -mapper /electricity_mapper.py ^
More?   -reducer /electricity_reducer.py
```

OUTPUT:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	
1	0	1	0	3	<memory:4 GB, vCores:3>	<memory:8 GB, vCores:8>	<memor

Cluster Nodes Metrics

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

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allo GPU
application_1726724100615_0001	admin	streamjob2796949504746728334.jar	MAPREDUCE	default	0	Thu Sep 19 11:09:47 +0550 2024	Thu Sep 19 11:09:48 +0550 2024	N/A	RUNNING	UNDEFINED	3	3	4096	-1

Showing 1 to 1 of 1 entries

RESULT:

PROGRAM NO: 08 **DATE:**	**Real-time data Analysis using MapReduce**	**PAEG NO:**

AIM:

ALGORITHM:

PROGRAM:

twitter_data.txt

```
2024-08-01,#hadoop
2024-08-01,#bigdata
2024-08-02,#hadoop
2024-08-03,#ai
```

twitter_mapper.py

```
import sys
for line in sys.stdin:
    line = line.strip()
    date, hashtag = line.split(',')
    print(f"{hashtag}\t1")
```

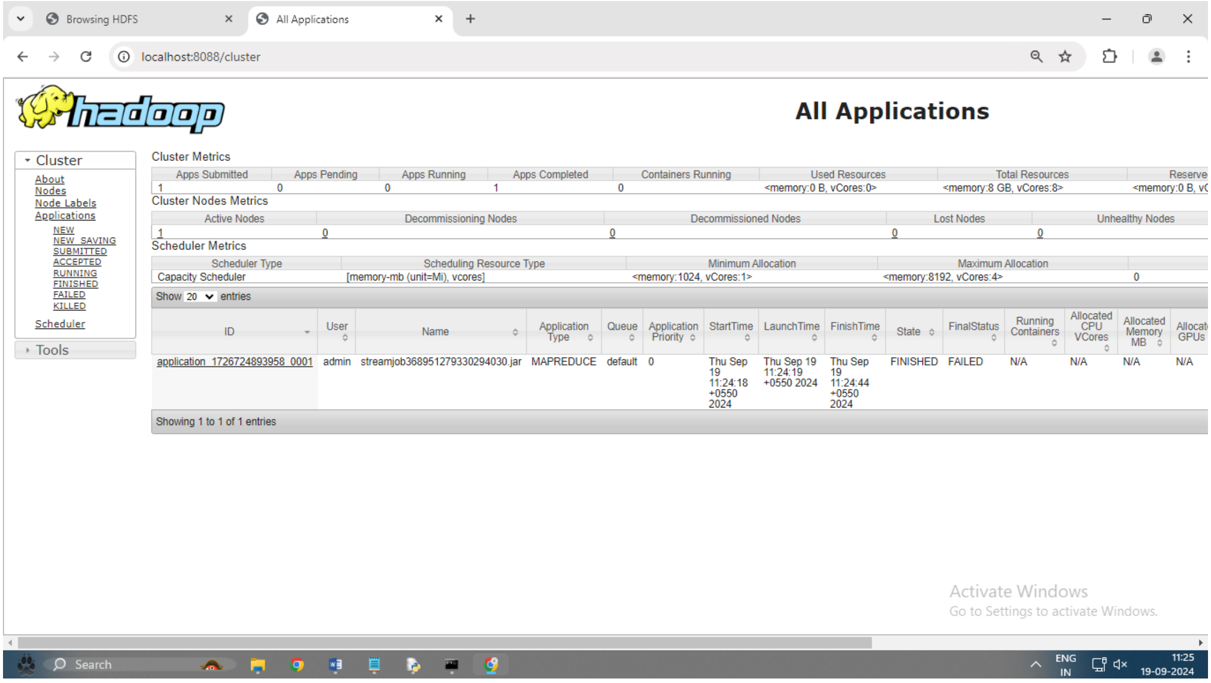
twitter_reducer.py

```
import sys
current_hashtag = None
current_count = 0

for line in sys.stdin:
    line = line.strip()
    hashtag, count = line.split('\t')
    count = int(count)

    if current_hashtag == hashtag:
        current_count += count
    else:
        if current_hashtag:
            print(f"{current_hashtag}\t{current_count}")
        current_hashtag = hashtag
        current_count = count

if current_hashtag == hashtag:
    print(f"{current_hashtag}\t{current_count}")
```

Command:

```
C:\Windows\system32>hdfs dfs -mkdir /ex8

C:\Windows\system32>hdfs dfs -put D:\twitter_data.txt /ex8

C:\Windows\system32>hdfs dfs -put D:\twitter_mapper.py /ex8

C:\Windows\system32>hdfs dfs -put D:\twitter_reducer.py /ex8

C:\Windows\system32>hadoop jar C:\hadoop\share\hadoop\tools\lib\hadoop-streaming-3.2.4.jar ^

More?   -input /ex8/twitter_data.txt ^

More?   -output /twitter_output ^

More?   -mapper /twitter_mapper.py ^

More?   -reducer /twitter_reducer.py
```

OUTPUT:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserve
1	0	0	1	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 B, vC

Cluster Nodes Metrics

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

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allocat GPUs
application_1726724893958_0001	admin	streamjob368951279330294030.jar	MAPREDUCE	default	0	Thu Sep 19 11:24:18 +0550 2024	Thu Sep 19 11:24:19 +0550 2024	Thu Sep 19 11:24:44 +0550 2024	FINISHED	FAILED	N/A	N/A	N/A	N/A

Showing 1 to 1 of 1 entries

RESULT: