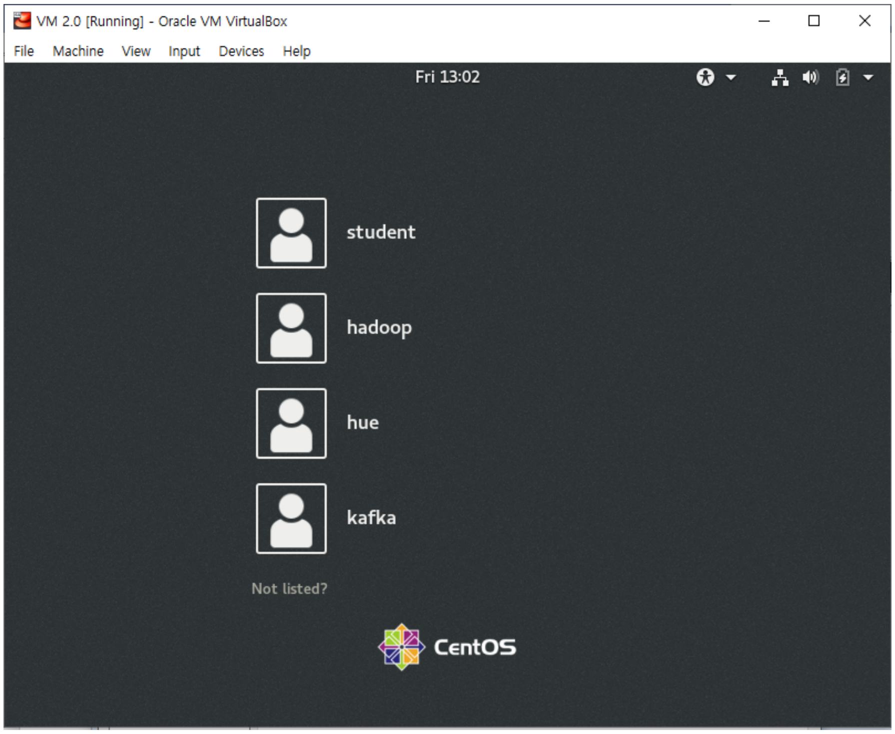
Lab 1: Working with HDFS

In this lab, you will explore and work with HDFS using command and Web UI.

1. **From the login screen, select student. The login password is "student"**



2. **Working with Linux and HDFS Home directories.**

    2.1. From a terminal as user **student**, enter the following command:

```
hdfs dfs -ls /
```

Notice that there are several directories. There is a /tmp directory where all users can read and write files. There is also a /user directory. This is the where the user home directories for HDFS reside. When using HDFS, each user can have a local Linux home directory as well as a HDFS home directory. In CentOS, the local Linux home directory is usually /home/*username*. For HDFS, the home directory is located at /user/*username.*

    2.2. Explore further, the user home directories in HDFS.

```
hdfs dfs -ls /user
hdfs dfs -ls /user/student
```

Notice that the HDFS home directory for student is empty for now.

2.3. Explore user home directories for Linux.

```
cd /home
ls -l
cd /home/student
ls -l
cd /home/
```

3. Exploring and working with HDFS directories.

3.1. Create a subdirectory in the student HDFS home directory and name is “MRtest”

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

3.2. Explore a book located in a subdirectory under your home directory.

There is a copy of the book “Alice’s Adventures in Wonderland” by Lewis Carrol in /home/student/data directory. Navigate to the directory and explore the file. The **~** (tilde) is a shortcut for the path to your Linux home directory. In our case, this will be /home/student. So, cd ~/data is a shortcut for cd /home/student/data

```
cd ~/data
less alice_in_wonderland.txt
```

The **less** command allows us to view a long file without it scrolling too far. Hit enter to continue scrolling. Press “**q**” to quit and stop viewing.

3.3. Put the book file under the MRtest directory

```
hdfs dfs -put \
/home/student/data/alice_in_wonderland.txt \
/user/student/MRtest/
```

Our command to "put" the book is quite long. In Linux, we can break up long commands into multiple lines using the \ at the end of the line. This tells Linux that the command isn't finished and there is more coming.

Recall from the lectures the syntax for the put subcommand is **hdfs dfs -put** ***source destination***. The source and destination can be actual files or directories. So, in the above command, we are requesting HDFS to put the alice_in_wonderland.txt file into the HDFS /user/student/MRtest directory.

Verify that the file has been copied to the HDFS destination.

```
hdfs dfs -ls /user/student/MRtest
```

4. **Add directories and files using HDFS CLI**

4.1. Run the command that make the /testdir in hdfs

```
hdfs dfs -mkdir /testdir
```

4.2. Verify the directory was created

```
hdfs dfs -ls /
```

4.3. Create three test files on the localhost and copy them into HDFS.

```
touch testfile1 testfile2 testfile3
hdfs dfs -put testfile* /testdir/
```

4.4. Verify the files were placed in HDFS

```
hdfs dfs -ls /testdir
```

4.5. Delete one of the files HDFS. Note the info message saying that the file was moved to the trash.

```
hdfs dfs -rm /testdir/testfile1
```

4.6. Delete another one of the files, this time skipping the trash

```
hdfs dfs -rm -skipTrash /testdir/testfile2
```

4.7. Try to delete the directory.

```
hdfs dfs -rmdir /testdir
```

The command did not work, because there is still a file in the directory.

4.8. Delete the directory recursively, which deletes the directory and all its contents.

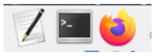
```
hdfs dfs -rm -r /testdir
```

4.9. Empty the user's trash

```
hdfs dfs -expunge
```

5. Explore HDFS with the Web UI

5.1. Open Firefox Web browser by selecting is from the panel on the bottom left.



Open the Namenode Web UI at http://localhost:9870. You can also select the bookmark from the browser.

5.2. Explore Overview tab

From this tab, we can see the overview information for the configured HDFS. We also can get a summary of the disk use statistics and the health of the datanodes.

5.3. Explore Datanodes and Datanode Volume Failures tab

From these tabs, we can view health and statistics of the datanodes. Because our Lab environment is not actually a cluster, we will only see 1 datanode.

5.4. Explore Snapshot tab

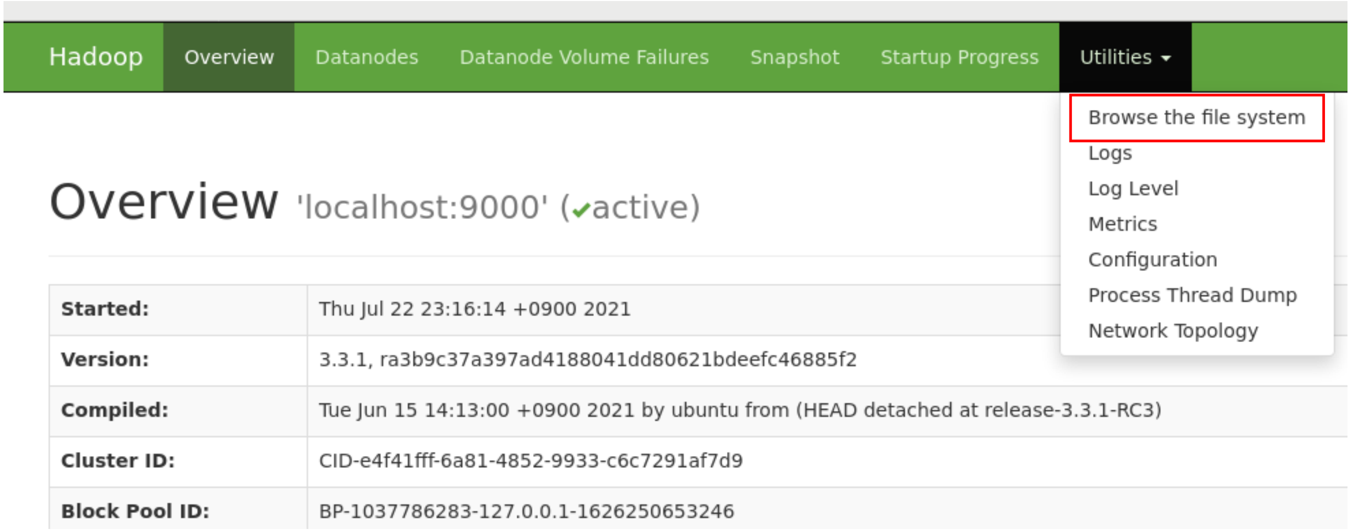
It is possible to take snapshots of directories in HDFS. If there are any snapshots saved, we can review information regarding them from this tab.

5.5. Explore Startup Progress tab

We can review the startup information for HDFS services, including the loading of the fsimage and edits file.

BONUS: Note the directory and filename of the fsimage and edits file. Open a terminal and login has hadoop. Navigate to the directory where the fsimage and edits file is saved. There are many versions of fsimage and edits file. These files were created and checkpointed.

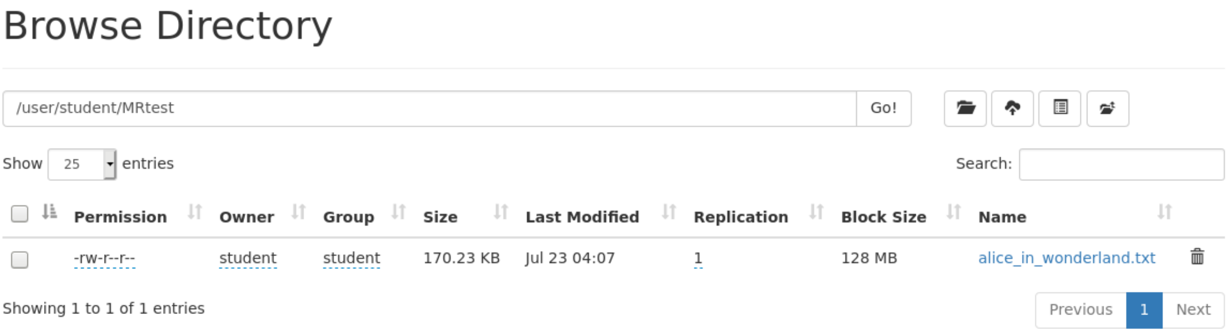
5.6. From Utilities, Browse the file system



Overview 'localhost:9000' (✓active)

Started:	Thu Jul 22 23:16:14 +0900 2021
Version:	3.3.1, ra3b9c37a397ad4188041dd80621bdeefc46885f2
Compiled:	Tue Jun 15 14:13:00 +0900 2021 by ubuntu from (HEAD detached at release-3.3.1-RC3)
Cluster ID:	CID-e4f41fff-6a81-4852-9933-c6c7291af7d9
Block Pool ID:	BP-1037786283-127.0.0.1-1626250653246

5.6.1. Navigate to the HDFS directory where you previously copied the alice_in_wonderland.txt file



Browse Directory

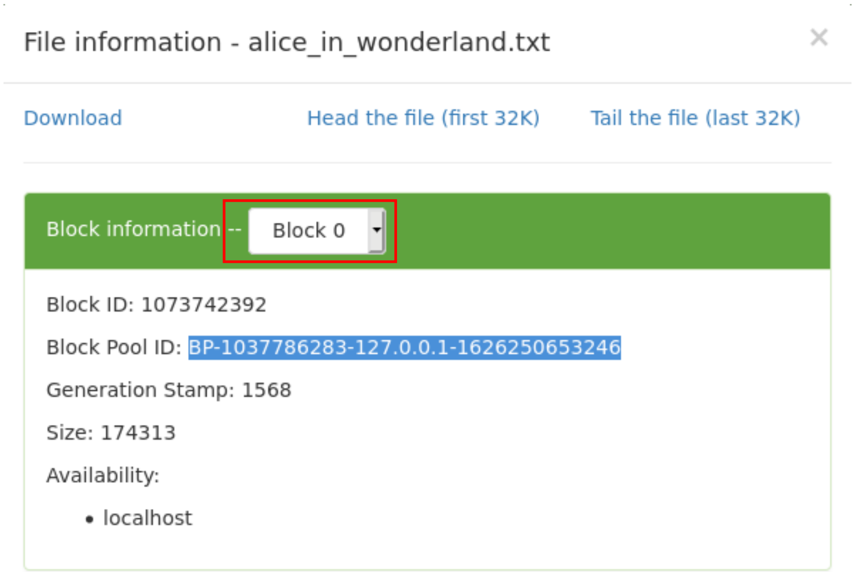
/user/student/MRtest

□	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	-rw-r--r--	student	student	170.23 KB	Jul 23 04:07	1	128 MB	alice_in_wonderland.txt

Showing 1 to 1 of 1 entries

5.6.2. Click on the file to bring up the block information pop-up window.

In our pseudo-distributed Hadoop environment, we have set the replication factor to 1 since there is only on machine in the cluster. When you click on the Block Information, you will only see one block. In a real cluster, the replication factor is normally set to 3 and we would be able to see information regarding all 3 blocks including the location of those blocks. Take note of the Block ID and the Block Pool ID. We will navigate to the directory where HDFS actually stores these blocks.


File information - alice_in_wonderland.txt

Download

Head the file (first 32K)

Tail the file (last 32K)

Block information -- Block 0

Block ID: 1073742392

Block Pool ID: BP-1037786283-127.0.0.1-1626250653246

Generation Stamp: 1568

Size: 174313

Availability:

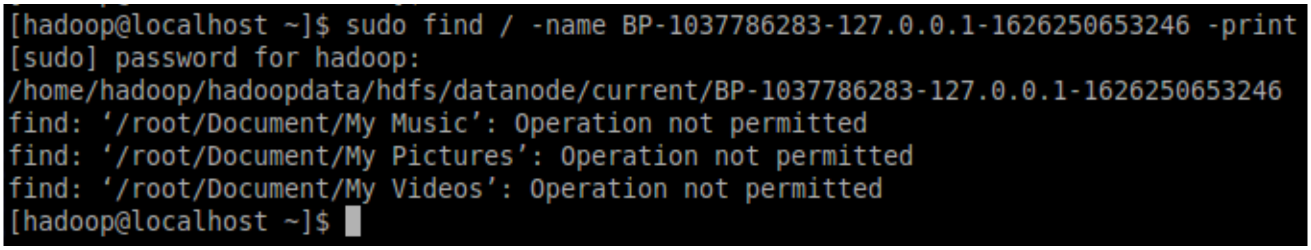
- localhost

5.6.3. Open a terminal and login as user **hadoop**

5.6.4. Use the Linux command find to search for the Block Pool ID

```
sudo find / -name <name of the Block Pool ID> -print
```

The above command instructs Linux to start at the / (root) directory and look for a file or directory with the given name and print out the information. The command will go to every subdirectory from / root. Your output will look similar to below.


```
[hadoop@localhost ~]$ sudo find / -name BP-1037786283-127.0.0.1-1626250653246 -print
[sudo] password for hadoop:
/home/hadoop/hadoopdata/hdfs/datanode/current/BP-1037786283-127.0.0.1-1626250653246
find: '/root/Document/My Music': Operation not permitted
find: '/root/Document/My Pictures': Operation not permitted
find: '/root/Document/My Videos': Operation not permitted
[hadoop@localhost ~]$ █
```

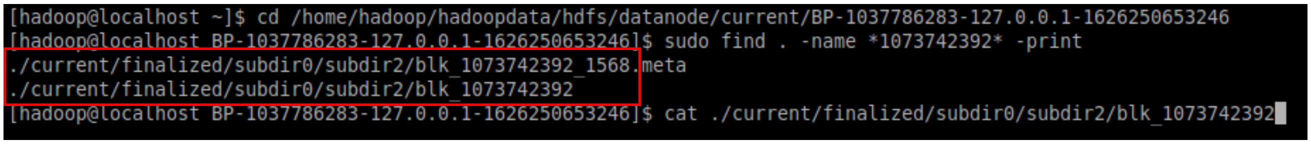
Don't worry about the 3 directories that Operation was not permitted. Those directories are mounted directories connecting your PC folders to the Linux system by VirtualBox.

5.6.5. Navigate to the directory where the Bock Pool ID was found and explore.

```
cd <result of your find from above step>
sudo find . -name *<BLOCK ID>* -print
```

We will use the **find** command again to look for the data blocks. This time, we will tell **find** to start looking from the current directory with the dot notation(.). You should see two files found. One of the files will contain the meta data. This is a binary file and you won't be able to see its content. However, the other file is a text file and the actuals contents that Hadoop has saved for the alice_in_wonderland.txt file save in HDFS.

The output of the above command will be similar to below.



```
[hadoop@localhost ~]$ cd /home/hadoop/hadoopdata/hdfs/datanode/current/BP-1037786283-127.0.0.1-1626250653246
[hadoop@localhost BP-1037786283-127.0.0.1-1626250653246]$ sudo find . -name *1073742392* -print
./current/finalized/subdir0/subdir2/blk_1073742392_1568.meta
./current/finalized/subdir0/subdir2/blk_1073742392
[hadoop@localhost BP-1037786283-127.0.0.1-1626250653246]$ cat ./current/finalized/subdir0/subdir2/blk_1073742392
```

Use the **less** or **cat** command to view the text file.

```
less <path to text file similar to ./current/finalized………>
```

6. **Increase the replication for selected existing files**

6.1. Increase the replication factor of the files in the weblogs directory to 5.

```
hdfs dfs -setrep -R 5 /weblogs
```

Lab 2: Working with YARN/MapReduce

We will use the alice_in_wonderland.txt file that has been saved to HDFS to run a wordcount MapReduce program.

1. Run the wordcount program from the MapReduce examples jar

    1.1. Open a new terminal as **student**.

    1.2. Navigate to the following directory.

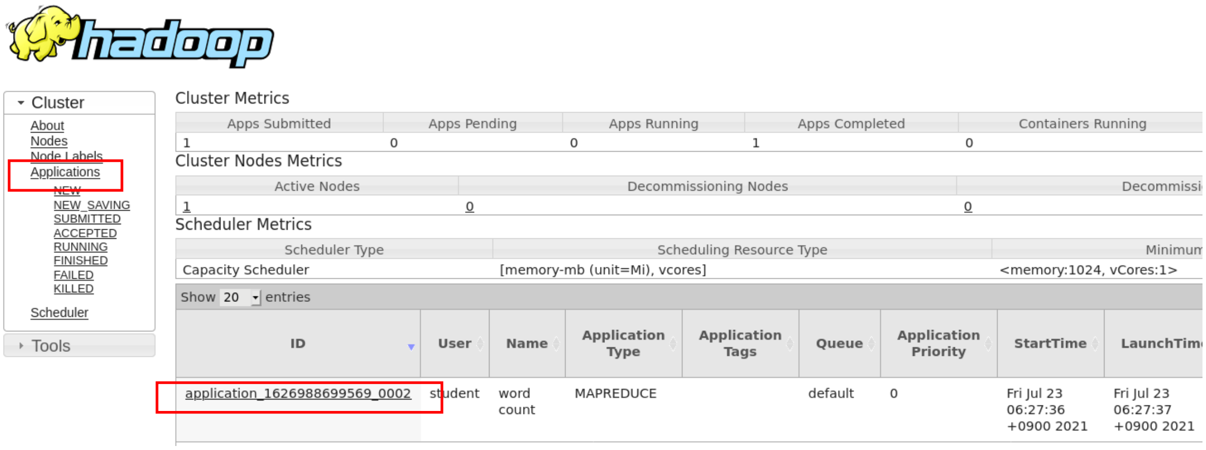
```
cd $HADOOP_HOME/share/hadoop/mapreduce
```

    1.3. Execute the hadoop-mapreduce-examples-3.3.1.jar with the wordcount option

```
hadoop jar ./hadoop-mapreduce-examples-3.3.1.jar wordcount \
MRtest WC_Output
```

    1.4. While the program is running open Firefox and navigate to the YARN Web UI at http://localhost:8088

Click on Applications on the left tab menu. You should see your application running. An application id has been assigned to the job. Click on the link for the application-id



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running
1	0	0	1	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissi
1	0	0

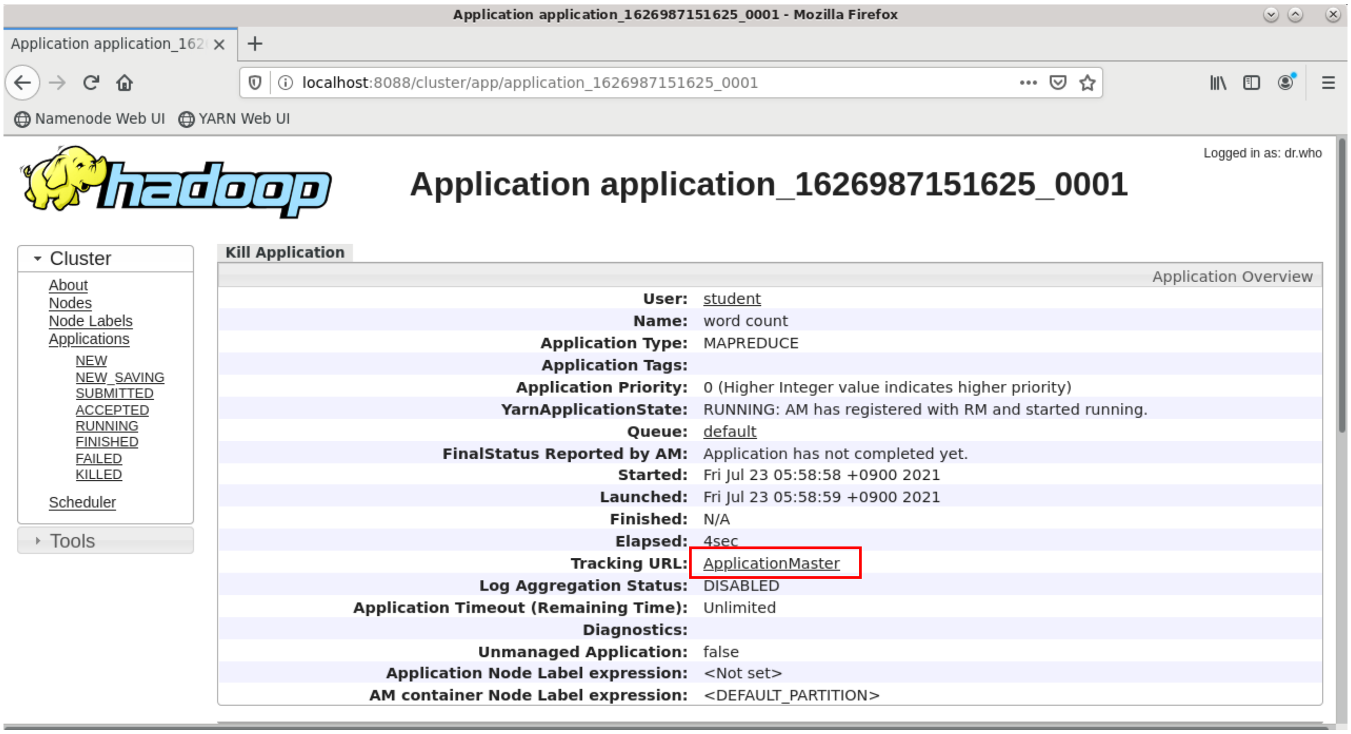
Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum
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	LaunchTim
application_1626988699569_0002	student	word count	MAPREDUCE		default	0	Fri Jul 23 06:27:36 +0900 2021	Fri Jul 23 06:27:37 +0900 2021

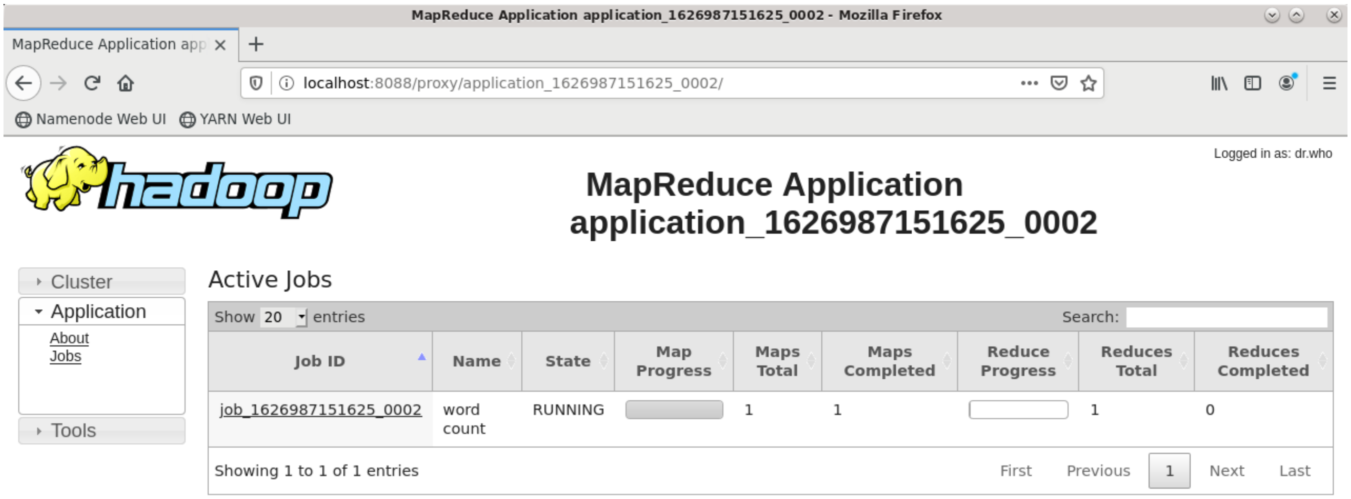
You should get a screen similar to below. Explore the details of the job from this screen. When finished, follow the links to explore the Application Master



Application application_1626987151625_0001

User:	student
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	RUNNING: AM has registered with RM and started running.
Queue:	default
FinalStatus Reported by AM:	Application has not completed yet.
Started:	Fri Jul 23 05:58:58 +0900 2021
Launched:	Fri Jul 23 05:58:59 +0900 2021
Finished:	N/A
Elapsed:	4sec
Tracking URL:	ApplicationMaster
Log Aggregation Status:	DISABLED
Application Timeout (Remaining Time):	Unlimited
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

Screen from the Application Master while job is still running.


MapReduce Application application_1626987151625_0002 - Mozilla Firefox
MapReduce Application app
localhost:8088/proxy/application_1626987151625_0002/
Namenode Web UI
YARN Web UI
Logged in as: dr.who
hadoop
Cluster
Application
About
Jobs
Tools
Show 20 entries
Search:
First
Previous
1
Next
Last

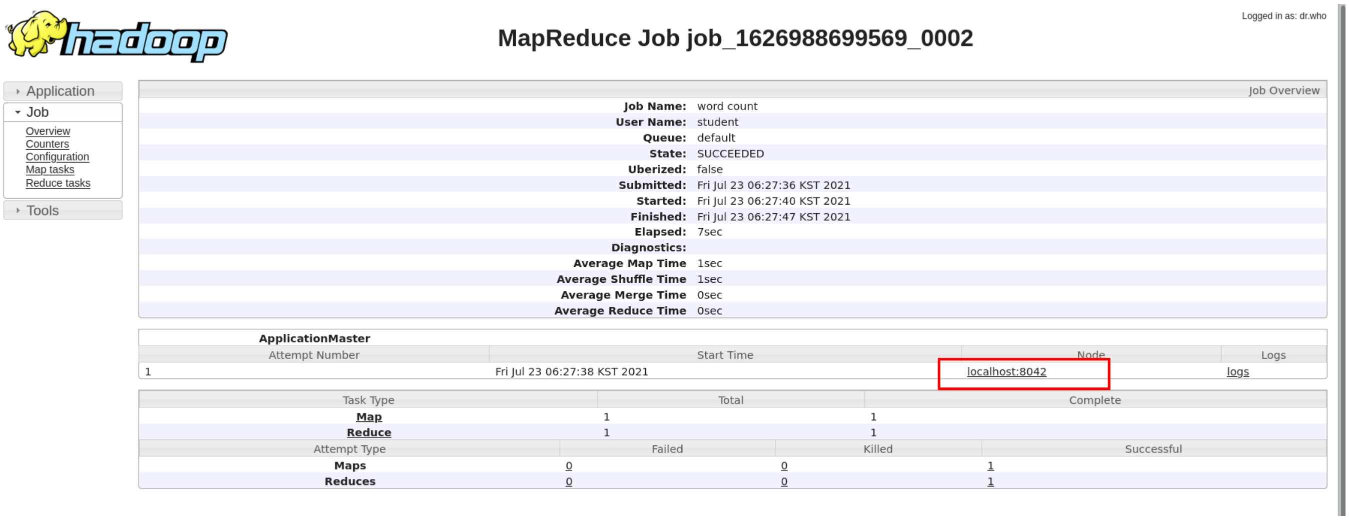
MapReduce Application application_1626987151625_0002

Active Jobs

Job ID	Name	State	Map Progress	Maps Total	Maps Completed	Reduce Progress	Reduces Total	Reduces Completed
job_1626987151625_0002	word count	RUNNING		1	1		1	0

Showing 1 to 1 of 1 entries

If the job is still running, you will see a screen similar to above. If the job has already completed, you will see a screen similar to below. Click on the link to the node that executed the job.



Logged in as: dr.who

MapReduce Job job_1626988699569_0002

Job Overview

Job Name:	word count
User Name:	student
Queue:	default
State:	SUCCEEDED
Uberized:	false
Submitted:	Fri Jul 23 06:27:36 KST 2021
Started:	Fri Jul 23 06:27:40 KST 2021
Finished:	Fri Jul 23 06:27:47 KST 2021
Elapsed:	7sec
Diagnostics:	
Average Map Time	1sec
Average Shuffle Time	1sec
Average Merge Time	0sec
Average Reduce Time	0sec

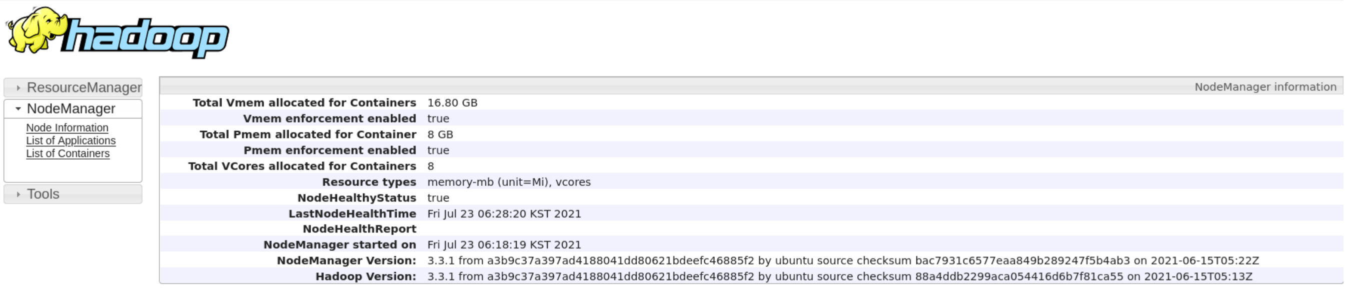
ApplicationMaster

Attempt Number	Start Time	Node	Logs
1	Fri Jul 23 06:27:38 KST 2021	localhost:8042	logs

Task Type	Total	Complete
Map	1	1
Reduce	1	1

Attempt Type	Failed	Killed	Successful
Maps	0	0	1
Reduces	0	0	1

Each node will show the container resources allocated to the job.



NodeManager information

Total Vmem allocated for Containers	16.80 GB
Vmem enforcement enabled	true
Total Pmem allocated for Container	8 GB
Pmem enforcement enabled	true
Total VCores allocated for Containers	8
Resource types	memory-mb (unit=Mi), vcores
NodeHealthyStatus	true
LastNodeHealthTime	Fri Jul 23 06:28:20 KST 2021
NodeHealthReport	
NodeManager started on	Fri Jul 23 06:18:19 KST 2021
NodeManager Version:	3.3.1 from a3b9c37a397ad4188041dd80621bdeefc46885f2 by ubuntu source checksum bac7931c6577eaa849b289247f5b4ab3 on 2021-06-15T05:22Z
Hadoop Version:	3.3.1 from a3b9c37a397ad4188041dd80621bdeefc46885f2 by ubuntu source checksum 88a4ddb2299aca054416d6b7f81ca55 on 2021-06-15T05:13Z

You may wish to re-run the job. If so, you will have to remove the output directory. The -r option of the rm (remove) command tells Hadoop to recursively delete any subfolders as well.

```
hdfs dfs -rm -r WC_Output
```

END OF LAB