گزارش این تمرین به شرح زیر است:

نصب و راه‌اندازی Hadoop cluster

ابتدا یک ماشین ubuntu server ایجاد شد و در آن تلاش شد که دانلود و نصب Hadoop و java 8 صورت پذیرد تا دو ماشین را بتوانیم با کمک همین ماشین از طریق clone کردن بدست آوریم.

مراحل نصب همانند لینک موجود در تمرین و اسلایدهای درس انجام شد و ip ماشین‌ها نیز در قسمت hosts هر سه ماشین به صورت زیر قرار گرفت:

```
hadoop@vm1:~$ cat /etc/hosts
# 127.0.0.1 localhost
# 127.0.1.1 vm1

192.168.1.50 vm1
192.168.1.49 vm2
192.168.1.51 vm3

# The following lines are desirable for IPv6 capable hosts
# ::1     ip6-localhost ip6-loopback
# fe00::0 ip6-localnet
# ff00::0 ip6-mcastprefix
# ff02::1 ip6-allnodes
# ff02::2 ip6-allrouters
```

همچنین public-key ماشین اصلی در دو ماشین دیگر قرار گرفت تا ارتباط بین ماشین‌ها نیز بدون مشکل انجام شود.

سپس تنظیمات خود Hadoop بر اساس لینک موجود در تمرین و اسلایدهای درس روی ماشین اصلی صورت گرفت و بر روی دو ماشین دیگر نیز کپی انجام شد.

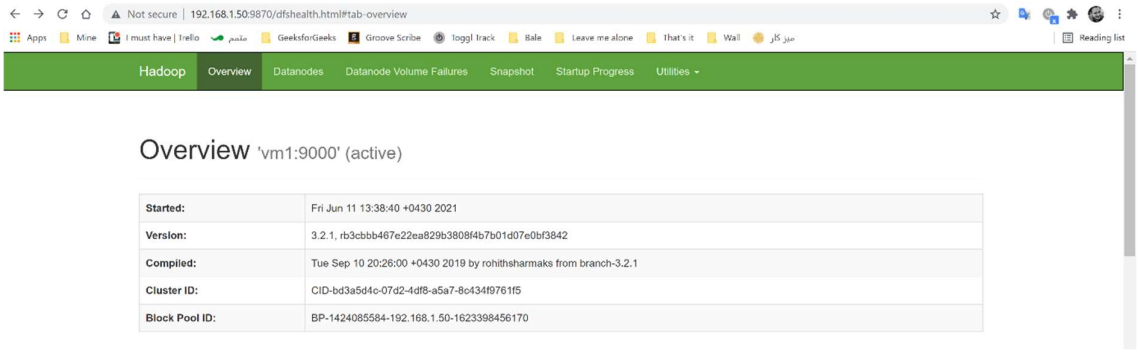
در این‌جا اسکریپت شروع Hadoop و Yarn اجرا شد و موارد مدنظر به درستی در حال اجرا بودند:

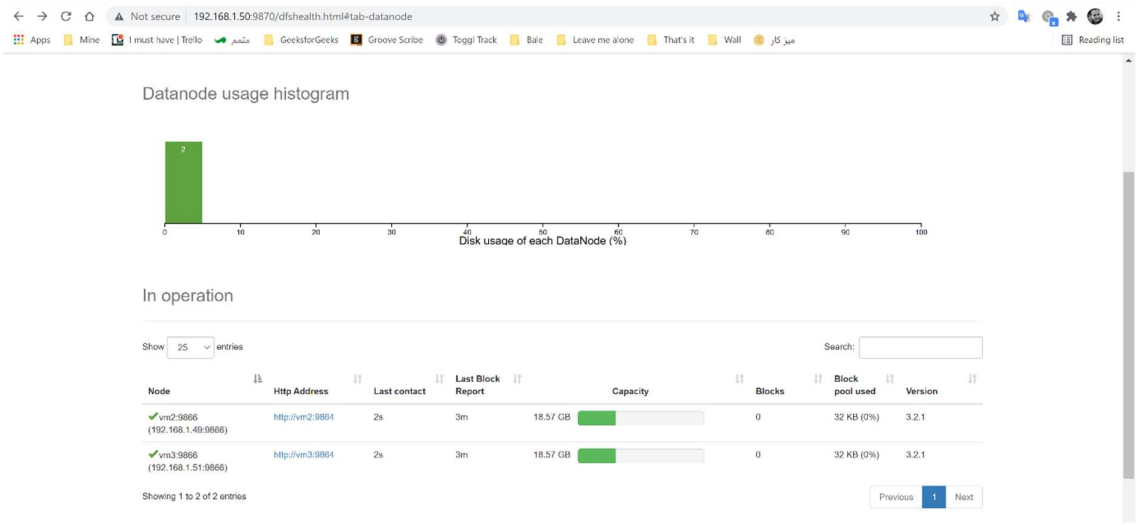
```
hadoop@vm1:~$ start-dfs.sh
Starting namenodes on [vm1]
Starting datanodes
Starting secondary namenodes [vm1]
hadoop@vm1:~$ jps
8549 NameNode
8892 Jps
8798 SecondaryNameNode
hadoop@vm1:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@vm1:~$ jps
8549 NameNode
9113 ResourceManager
8798 SecondaryNameNode
9214 Jps
```

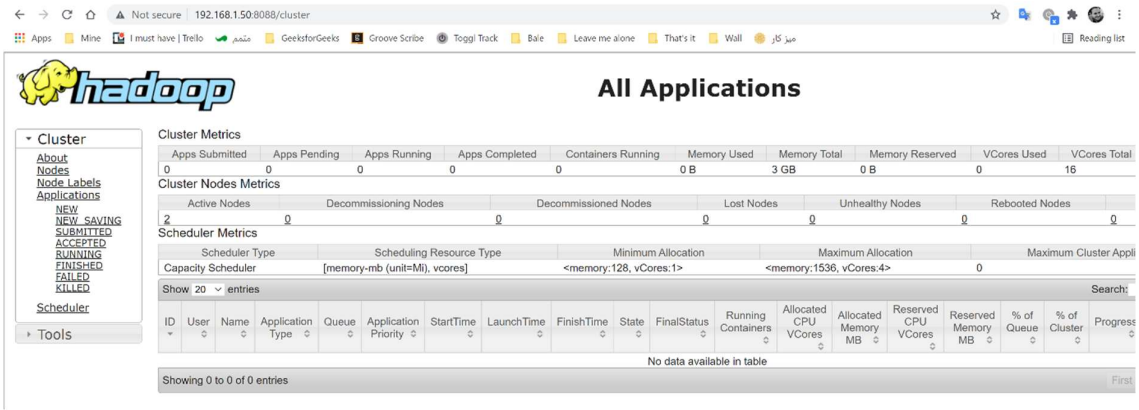
```
hadoop@vm3:~/hadoop$ jps
6624 NodeManager
6720 Jps
6380 DataNode
```

```
hadoop@vm2:~/hadoop$ jps
6697 NodeManager
6793 Jps
6443 DataNode
```

در زیر صحت اجرای این موارد در Web-GUI خود HDFS و Hadoop قابل مشاهده است:







توضیحاتی که در مورد اطلاعاتی که WebGUI در اختیار ما قرار می‌دهد نیز به صورت زیر است:

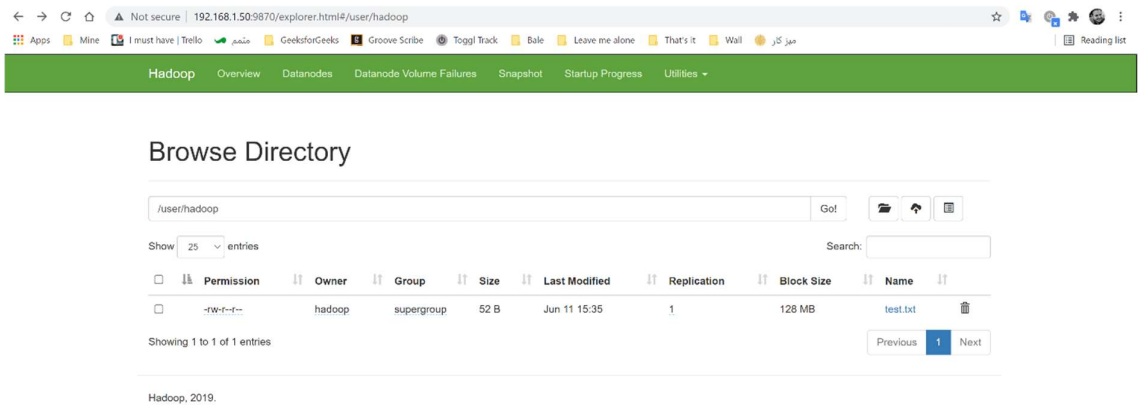
- در قسمت overview اطلاعاتی در مورد زمان شروع به کار سرور، نسخه‌ی آن، زمان کامپایل آن، آیدی cluster داده شده است.
- در قسمت summary نیز اطلاعاتی در مورد تعداد فایل‌های موجود، انواع مموری‌های مورد استفاده و... موجود است
- هم‌چنین در جدولی که در این قسمت وجود دارد اطلاعاتی در مورد کل فضای ذخیره‌سازی‌ای که در دسترس هست و میزانی از آن که توسط dfs مورد استفاده قراره گرفته و میزانی که آزاد است قابل مشاهده است.
- در پایین‌تر تعداد نودهایی که فعال هستند نیز قابل مشاهده است که با کلیک بر روی live node اطلاعات بیشتری در مورد هر کدام از این نودها مانند فضای ذخیره‌سازی آزاد در هر کدام، block pool استفاده شده در هر کدام از آن‌ها، آدرس ماشین آن‌ها و... قابل مشاهده است.

بارگذاری فایل

همان‌طور که در تصویر زیر مشخص است، فایل مورد نظر پس از ساخته شدن و با موفقیت نیز بارگذاری شده است:

```
hadoop@vm1:~$ hdfs dfs -mkdir -p /user/hadoop
hadoop@vm1:~$ nano test.txt
hadoop@vm1:~$ cat test.txt
This is a test by me.
I should finish this HW soon.
hadoop@vm1:~$ hdfs dfs -put test.txt /user/hadoop
2021-06-11 11:05:45,146 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
hadoop@vm1:~$ hdfs dfs -ls /user/hadoop
Found 1 items
-rw-r--r--   1 hadoop supergroup         52 2021-06-11 11:05 /user/hadoop/test.txt
hadoop@vm1:~$
```

که از طریق WebGUI نیز قابل مشاهده است:



تست WordCount

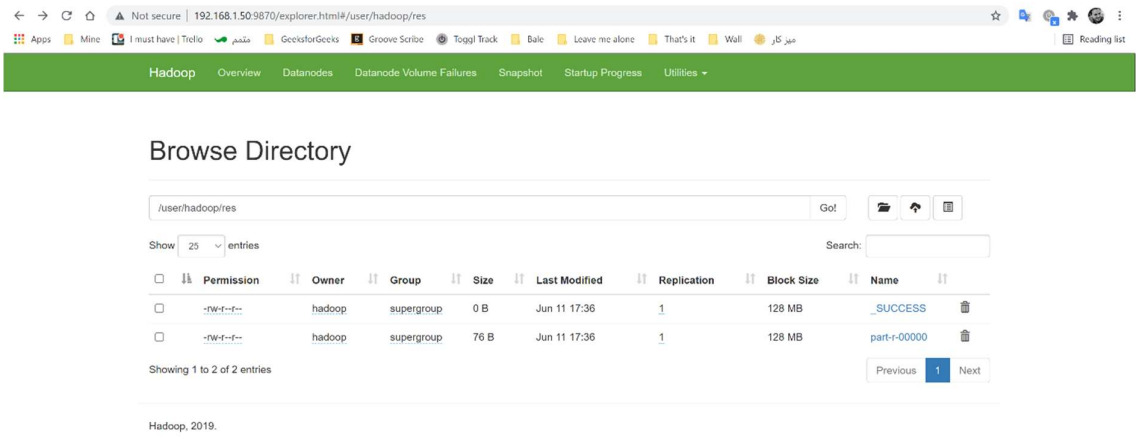
کد WordCount موجود در سایت Hadoop را درون ماشین اصلی قرار می‌دهیم و آن را کامپایل می‌کنیم و بر روی فایل تستی که در قسمت قبل ساختیم اجرا می‌کنیم:

```
hadoop@vm1:~$ hadoop com.sun.tools.javac.Main WordCount.java
/home/hadoop/hadoop/libexec/hadoop-functions.sh: line 2366: HADOOP_COM.SUN.TOOLS.JAVAC.MAIN_USER: invalid variable name
/home/hadoop/hadoop/libexec/hadoop-functions.sh: line 2461: HADOOP_COM.SUN.TOOLS.JAVAC.MAIN_OPTS: invalid variable name
hadoop@vm1:~$ jar cf wc.jar WordCount*.class
hadoop@vm1:~$ hadoop jar wc.jar WordCount /user/hadoop /user/hadoop/res
2021-06-11 13:06:02,560 INFO client.RMProxy: Connecting to ResourceManager at /192.168.1.50:8032
```

hadoop@vm1: ~ | hadoop@vm2: ~ | hadoop@vm3: ~

```
                Peak Map Virtual memory (bytes)=1863757824
                Peak Reduce Physical memory (bytes)=167510016
                Peak Reduce Virtual memory (bytes)=1873408000
        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=52
        File Output Format Counters
                Bytes Written=76
hadoop@vm1:~$ hadoop fs -cat /user/hadoop/res/part-r-00000
2021-06-11 13:07:42,346 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
HW      1
I       1
This    1
a       1
by      1
finish  1
is      1
me.     1
should  1
soon.   1
test    1
this    1
hadoop@vm1:~$
```

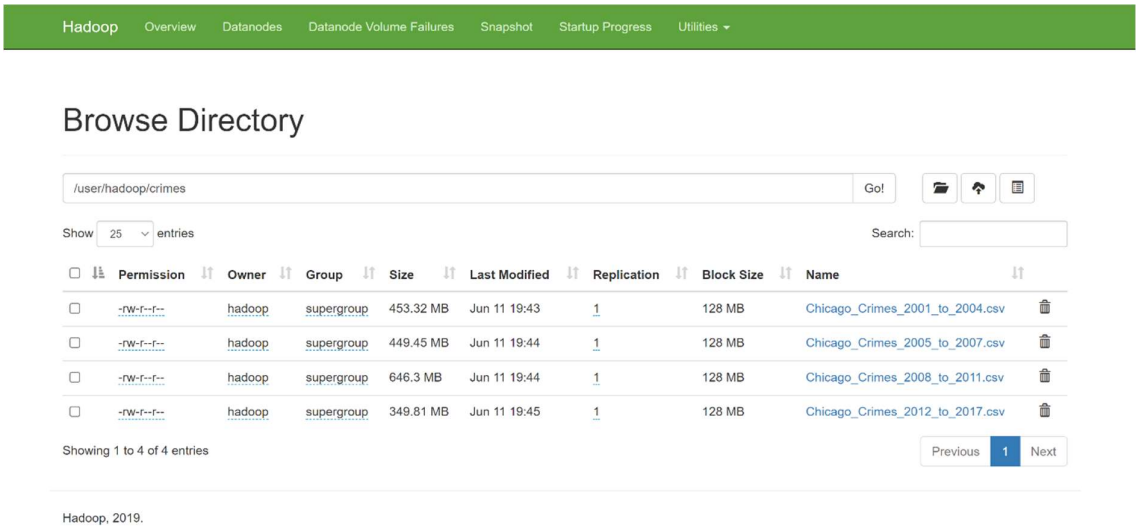
در تصویر بالا نتیجه‌ی اجرای کد بر روی فایل تست نیز مشخص است.



در قسمت بعد به سراغ دیتاست‌های اصلی تمرین می‌رویم.

اجرای MapReduce بر روی دیتاست جرم‌ها

دیتاست‌ها را در hdfs قرار می‌دهیم:



سپس کد WordCount را مقداری تغییر می‌دهیم تا بتواند ورودی جدیدی که از نوع csv است را پشتیبانی کند که به این منظور، tokenization بر اساس comma انجام شده است. اسم فایل مربوط به کد جدید نیز DistrictCount گذاشته شده است. پس از ساخت فایل اجرایی این کد، آن را بر روی دیتاست‌ها اجرا می‌کنیم و خروجی را در یک فایل csv ذخیره می‌کنیم.

```
8.0     550011
11.0    498775
7.0     476524
25.0    463938
6.0     457669
4.0     453896
3.0     407866
9.0     397942
12.0    386274
2.0     378297
5.0     354567
15.0    352585

19.0    351537
18.0    337197
10.0    335491
14.0    314642
1.0     291073
16.0    263443
22.0    263055
24.0    237983
17.0    230955
20.0    137310
31.0    158
District        4
21.0    4
Beat    1
23.0    1
13.0    1
        91
```

که نتیجه‌ی اجرای این کد از طریق WebGUI نیز قابل مشاهده است: