

Assignment 2

Sayali Kudale

Documentation:

TopologyMain.java

In this file we have provided topology definition and configuration.

In this version we are running/debugging Storm on local mode, Hence, cluster type is specified as LocalCluster. To run storm on local model storm dependencies are required and which is mentioned in pom.xml. Maven takes care of all the dependencies.

We create topology using using TopologyBuilder, in which we decide how nodes arranges and how they exchange data.

flightDataReader spout and HubIdentifier bolt are connected using shuffleGrouping, which means storm sends data from spout to bolt in randomly distributed fashion.

HubIdentifier bolt and AirlineSorter bolt are connected using fieldsGrouping, which means storm maintains the order of the words sends between these bolts.

We pass the data to config object which will be then sent to all nodes in prepare method.

Lastly, Create and run the topology using createTopology and submitTopology, sleep for 5 seconds and then stop the topology by shutting down the cluster.

flightsDataReader.java

The flightsDataReader is a spout class which extends BaseRichSpout class. The main responsibility of this class is to read line of flight.txt on by one and send it to the bolt (HubIdentifier.java)

The fields which are read from file are then emitted to the bolt. The spout emits list of defined fields which are defined in the declareOutputFields method.

In the class we are initializing the filereader object in the Open function. Then, parsing the filereader using JsonParser. After parsing only retrieving the Array of States.

After Iterating through each state, we are retrieving the 17 flight parameters and emitting those using collector object. All 17 fields are declared in the declareOutputFields method.

HubIdentifier.java

HubIdentifier.java is a Bolt class extended from BaseBasicBolt and it process the data received from flightsDataReader.

The prepare method in the bolt gets called at the time of initialization and all the data reading and initialization activity happens here. In this program we are preparing the airport list from given airport.txt The execute method in this bolt is very important method and which gets called for every tuple sent by the flightDataReader.

In execute method we are checking which airports are nearest to the flights based on the latitude and longitude of the flight. If latitude and longitude is within 20 miles of any airport then that flight is considered as nearest to the airport.

After determining the proximity of the flight, we are emitting the only the flights nearest to any airport.

AirlineSorter.java

AirlineSorter.java is the next bolt and it is responsible for counting the number of flights of each airline per airport.

In execute method it received the tuple sent by previous bolt and based on the call sign of the flight determines the airline and add that airline and corresponding count to the map. This map is again added as a key to the counters map which keeps track of the flights per airport.

In the cleanup function, which gets executed when topology finishes, we are printing the count of the airlines per airport.

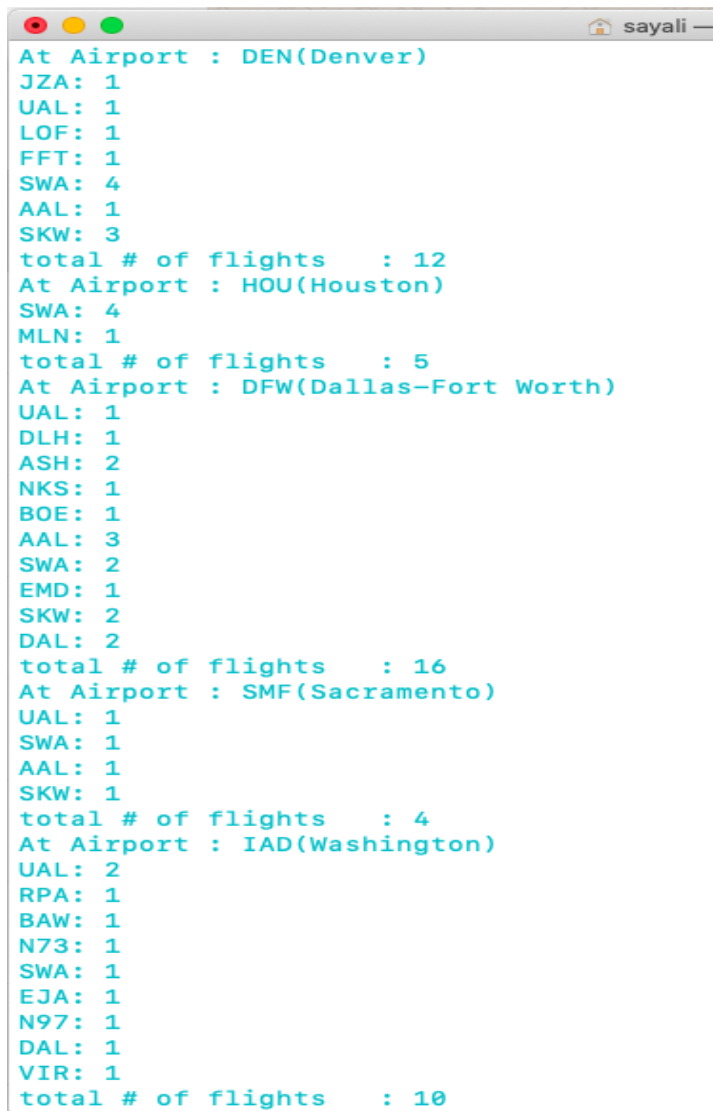
Source Code:

Kindly refer the **src** folder which contains the below files:

TopologyMain.java
FlightsDataReader.java
HubIdentifier.java
AirlineSorter.java

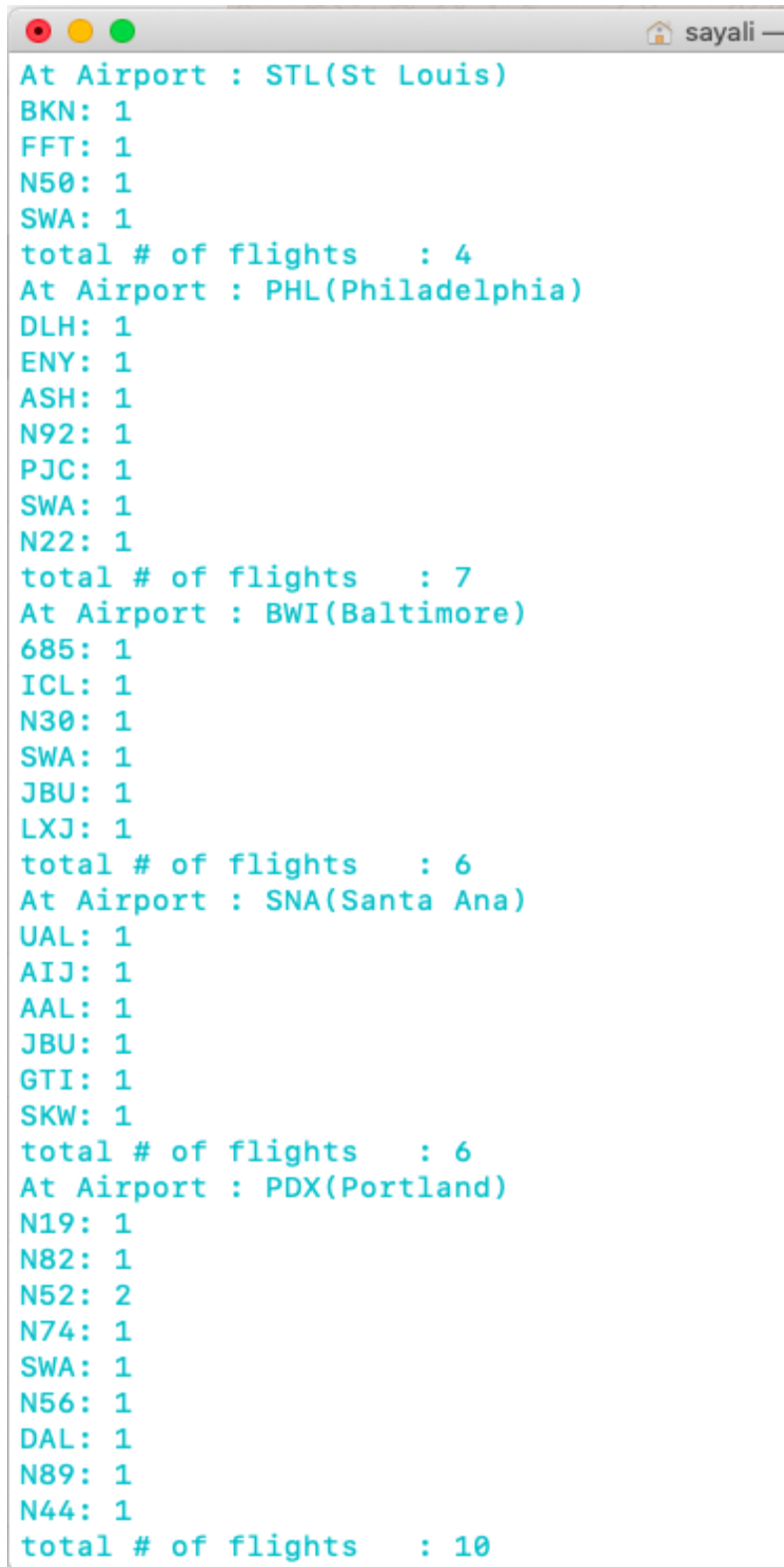
Execution Output:

1. Basic Implementation:



```
At Airport : DEN(Denver)
JZA: 1
UAL: 1
LOF: 1
FFT: 1
SWA: 4
AAL: 1
SKW: 3
total # of flights : 12
At Airport : HOU(Houston)
SWA: 4
MLN: 1
total # of flights : 5
At Airport : DFW(Dallas-Fort Worth)
UAL: 1
DLH: 1
ASH: 2
NKS: 1
BOE: 1
AAL: 3
SWA: 2
EMD: 1
SKW: 2
DAL: 2
total # of flights : 16
At Airport : SMF(Sacramento)
UAL: 1
SWA: 1
AAL: 1
SKW: 1
total # of flights : 4
At Airport : IAD(Washington)
UAL: 2
RPA: 1
BAW: 1
N73: 1
SWA: 1
EJA: 1
N97: 1
DAL: 1
VIR: 1
total # of flights : 10
```

Figure 1 : BasicImp_Part1



```
At Airport : STL(St Louis)
BKN: 1
FFT: 1
N50: 1
SWA: 1
total # of flights : 4
At Airport : PHL(Philadelphia)
DLH: 1
ENY: 1
ASH: 1
N92: 1
PJC: 1
SWA: 1
N22: 1
total # of flights : 7
At Airport : BWI(Baltimore)
685: 1
ICL: 1
N30: 1
SWA: 1
JBU: 1
LXJ: 1
total # of flights : 6
At Airport : SNA(Santa Ana)
UAL: 1
AIJ: 1
AAL: 1
JBU: 1
GTI: 1
SKW: 1
total # of flights : 6
At Airport : PDX(Portland)
N19: 1
N82: 1
N52: 2
N74: 1
SWA: 1
N56: 1
DAL: 1
N89: 1
N44: 1
total # of flights : 10
```

Figure 2: Basic Implementation part 2

Detailed output files are present in ExecutionOutputs folder: BasicOut.txt

2. Multithreaded Implementation:

Detailed output files are present in ExecutionOutputs Folder :

Thread025.txt

Thread026.txt

Thread036.txt

Performance Evaluation Table:

Spout Threads	HubIdentifier Threads	AirlineSorter Threads	Time(msec)
1	1	1	4801
1	1	2	4852
1	1	3	4808
1	1	5	4192
1	1	6	4485
1	2	1	4633
1	2	2	4555
1	2	3	4520
1	2	5	4367
1	2	6	4053
1	3	1	4674
1	3	2	4425
1	3	3	4445
1	3	5	4567
1	3	6	4691
2	3	2	4761
2	2	5	4609
3	1	2	4546
3	3	3	4818

Table 1: Thread Performance Evaluation

Thread Performance Graph (Spout thread: 1 , Bolt 1 thread : 1):

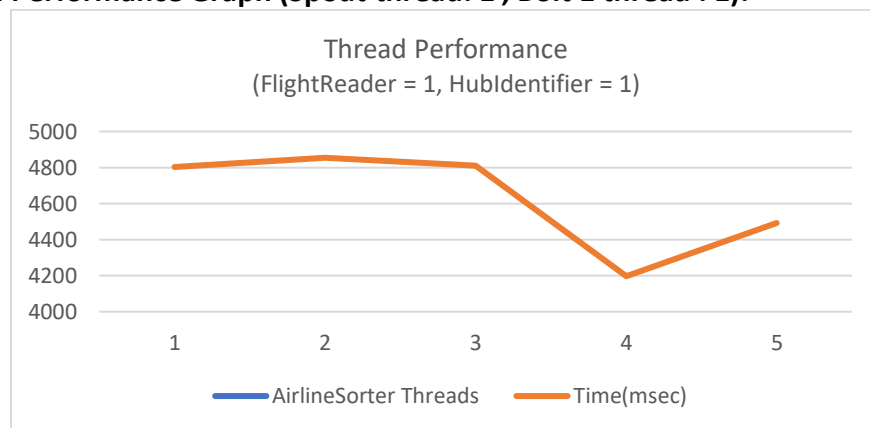


Figure 3 : Graph 1 : Thread perm with spout thread 1 & Bolt1 threads 1

Thread Performance Graph (Spout thread: 1 , Bolt1 threads : 2):

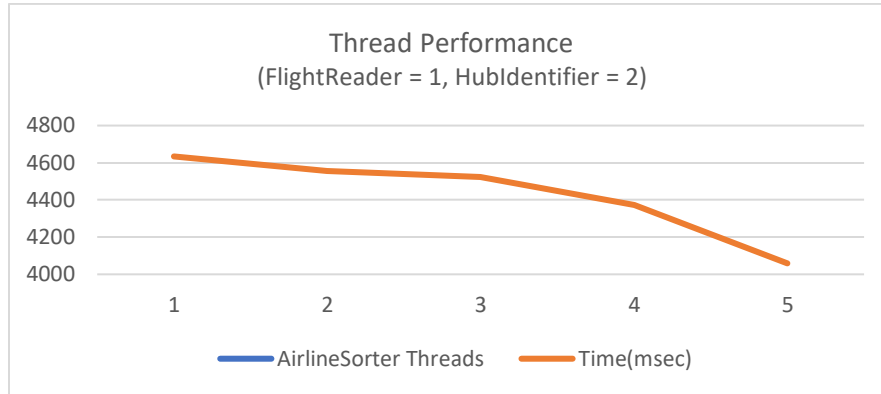


Figure 4 : Thread Perm with Spout thread :1 & bolt1 threads : 2

Thread Performance Graph (Spout thread: 1, Bolt1 threads: 3):

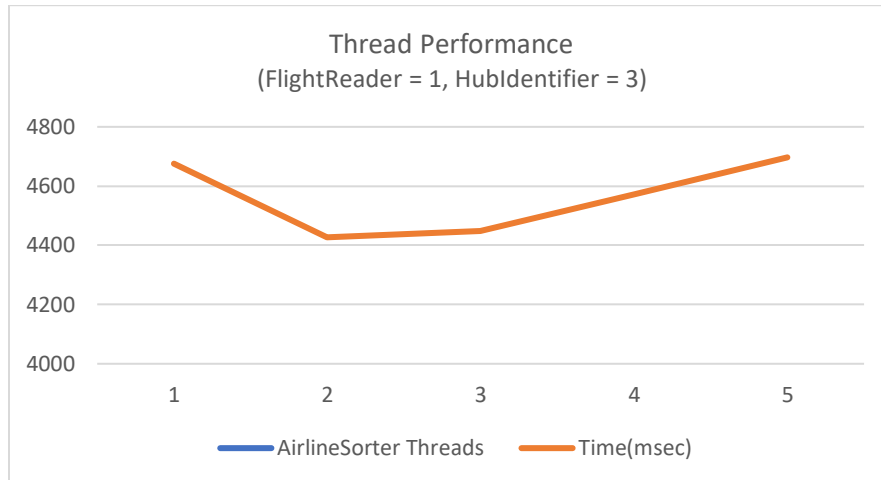


Figure 5: Thread Perm with Spot thread: 1 & Bolt1 threads: 3

3. Implementation of Extra Feature:

a. Feature 1: Detailed information of airlines and sorting

Detailed output files are present in ExecutedOutPutFolder :
feature1Out.txt

```

-- Airline Sorter [airline-sorter-2] --
At Airport : DEN(Denver)
SWA(Southwest):      5
UAL(United):         4
SKW(SkyWest):        3
JZA(Jazz Aviation):  1
LOF(Trans States):   1
FFT(Frontier):       1
N85(Alexandria Field): 1
N52(Jaars-Townsend):  1
AAL(American):       1
total # of flights   : 18

At Airport : HOU(Houston)
SWA(Southwest):      4
N19(aztec municipal): 1
N72(Warwick Municipal): 1
MLN(Melilla):        1
total # of flights   : 7

At Airport : DFW(Dallas-Fort Worth)
AAL(American):       6
ASH(Mesa):           3
SWA(Southwest):      3
SKW(SkyWest):        2
DAL(Delta Air Lines): 2
UAL(United):         1
DLH(Duluth International): 1
NKS(Spirit):         1
BOE(Boeing):         1
EMD(Eaglemed (Ballard Aviation)): 1
QXE(Horizon Air):    1
total # of flights   : 22

```

Figure 6: additionalFeature_part1

```

At Airport : SMF(Sacramento)
UAL(United):         1
SWA(Southwest):      1
AAL(American):       1
SKW(SkyWest):        1
total # of flights   : 4

At Airport : IAD(Washington)
UAL(United):         5
DAL(Delta Air Lines): 3
RPA(Republic):       1
N73(Beech 300 Super King Air): 1
AAL(American):       1
N97(Hiatt):          1
FFT(Frontier):       1
EDV(Endeavor Air):   1
BAW(British Airways): 1
ROU(rouge airline):  1
SWA(Southwest):      1
EJA(NetJets):        1
VIR(Virgin Atlantic): 1
total # of flights   : 19

At Airport : STL(St Louis)
SWA(Southwest):      2
N47(Pottstown Municipal): 1
BKN(Blackwell Tonkawa Municipal): 1
FFT(Frontier):       1
N50(Bridgeton):      1
N72(Warwick Municipal): 1
N95(N95):            1
N20(Ine):            1
total # of flights   : 9

```

Figure 7: Additional Feature part_2

```
sayali — ssh sayalik@cssmpi1h.uw

At Airport : PHL(Philadelphia)
SWA(Southwest):          2
DLH(Duluth International): 1
ENY(Envoy Air):          1
GAJ(Yamagata):           1
RPA(Republic):           1
ASH(Mesa):               1
N92(Laneys):             1
PJC(Zelienople Municipal): 1
N22(GAF Nomad):          1
total # of flights      : 10

At Airport : BWI(Baltimore)
AAL(American):           2
UAL(United):             1
685(Wauchula):           1
ICL(Municipal airport):  1
N30(Cherry Ridge):       1
SWA(Southwest):          1
JBU(JetBlue):            1
LXJ(Bombardier Business Jet Solutions):
total # of flights      : 9

At Airport : SNA(Santa Ana)
UAL(United):             2
AAL(American):           2
SKW(SkyWest):            2
AIJ(Mexico City International): 1
JBU(JetBlue):            1
CPZ(Compass):            1
GTI(Atlas Air):          1
DAL(Delta Air Lines):    1
total # of flights      : 11
```

Figure 8 : Additional Feature part 3

Discussions:

The program 2 was great hands-on Storm Programming and understanding the topology of Storm.

Performance Evaluations:

We first executed the Storm topology on single threaded configuration and then in multithreaded configuration.

To evaluate the performance by increasing or decreasing number of threads, we have added start time(ms) in prepare function of AirlineSorter.Java and endTime(ms) in cleanup Function. This is because, prepare function gets called at the time of topology initialization and cleanup function gets invoked when topology finishes.

Observations:

According to the time captured at the Table 1, below are the observations of multithreaded execution of the Storm:

1. By keeping the number of threads in spout as 1, we tried increasing the thread numbers of bolt 1 and bolt 2.
 - a. By keeping Number of threads in bolt_1 = 1

In this setting we tried to change the values of bolt 2 threads from 1 to 6 and the time captured is plotted in graph as seen in Figure 3.

As per the Graph time doesn't show significant improvement till number of threads increased to 4, However, the execution time was less when number of threads used in bolt 2 was 5.

Also, would like to note that when number of threads increased beyond 5, the execution time again increased.

- b. By keeping Number of threads in bolt_1 = 2

In this setting we tried to change the values of bolt 2 threads from 1 to 6 and the time captured is plotted in graph as seen in Figure 4.

As per the Graph, execution time was decreasing when number of threads increases. The lowest execution time i.e 4053 ms observed when number of threads used 6 for bolt2.

- c. By keeping Number of threads in bolt_1 = 3

In this setting we tried to change the values of bolt 2 threads from 1 to 6 and the time captured is plotted in graph as seen in Figure 5.

From graph, less execution time observed (4424 ms) when number of threads used were 2. However, we it is not a significantly lower value than the other values.

2. When number of threads in spout increases then we have seen the discrepancy in the output. For instance, when number of threads in spout were 2, then all the flight counts in output were exactly doubled than the correct output.

Summary:

1. From the above observations, the optimal performance was observed at below configuration:

	Spout	Bolt 1	Bolt 2
Number of threads	1	2	6

2. However, all the numbers captured in Time were not drastically different than each other and it were fluctuating when program ran with same setting multiple times. Hence, we cannot say at what setting program will give best performance.
3. Additionally, as output results were changing when number of threads increased in spout, we can say that the implementation of the Spout is not thread safe.

Additional Features:

To give the descriptive output, we have performed the below additional tasks:

1. Added the Airline names along with the code
2. Sorted the output so that airline with the highest number of flights will be on top and so on.

To implement the additional feature, we have manually collected the airline names from the code and added all the data in AirlineCode.txt.

Also, we have made changes in the below functions of AirlineSorter.java:

Prepare Function:

In the basic implementation prepare function was not doing any major task, but here we are reading the AirlineCode data and Storing it in the HashMap with key as the AirlineCode and Value as the Airline Name.

Cleanup Function:

Here, we are performing the additional task of sorting the data descending by the count of flights. We have implemented the function `sortByValue` which sorts the inner hashmap collection by the count values.

Additionally, after getting the Airline code we are fetching the associated airline name from the `airlinesWithCode` HashMap and displaying the result.

Improvements in the program:

1. Making the Spout as Thread safe so that if number of threads increases the output will not change.
2. This program is not executed in the distributed environment; hence performance is evaluated only based on the single machine execution. This needs to be performed in distributed environment to validate the performance.
3. Cluster mode execution is not performed, in future all the configuration and installation need to be done to execute this program in cluster mode.
4. Although, we have executed this program by adding the number of workers and by setting the Max task parallelism, the detailed study of the performance is not done with this setting.

Limitations:

1. In Storm Architecture, it is tedious to update the topology, every time we need to kill the older topology and create the new one. This is the problem when organization wants the zero-down time.
2. Storm is more about the Stream processing. Unlike Spark it doesn't perform all type of processing such as Batch Processing, Iterative Processing etc.
3. In this program, even if we increase the number of threads, it doesn't show significant improvement in the performance.

Lab Sessions:

Lab 4 output:

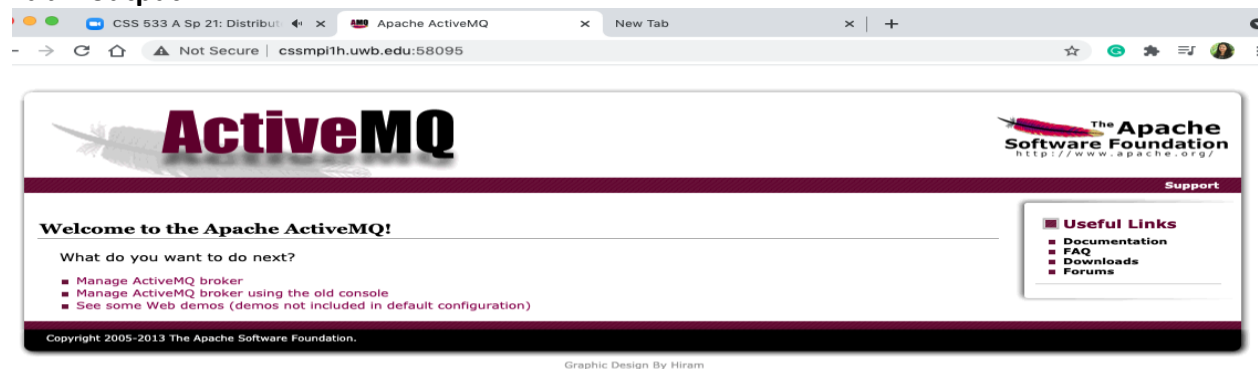


Figure 9 : Lab4 : ActiveMQ

```
sayali — sayalik@cssmpi1h:~/lab4/apache-activemq-5.9.0/examples/lab4 — ssh sayalik@cssmpi1h.uwb.edu — 80x24
[sayalik@cssmpi1h lab4]$ java -cp ../../activemq-all-5.9.0.jar:. Chat TopicCF to
pic1 Mary
log4j:WARN No appenders could be found for logger (org.apache.activemq.transport
.WireFormatNegotiator).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more in
fo.
Hey John
John: Hey Mary!!! How are you doing?
```

Figure 10 : Lab4 : Console output1 part1

```
Q Find
[sayalik@cssmpi1h lab4]$ java -cp ../../activemq-all-5.9.0.jar:. Chat TopicCF topic1 John
log4j:WARN No appenders could be found for logger (org.apache.activemq.transport.WireFormatNegotiator).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Mary: Hey John
Hey Mary!!! How are you doing?
```

Figure 11 : Lab 4 Console output part 2

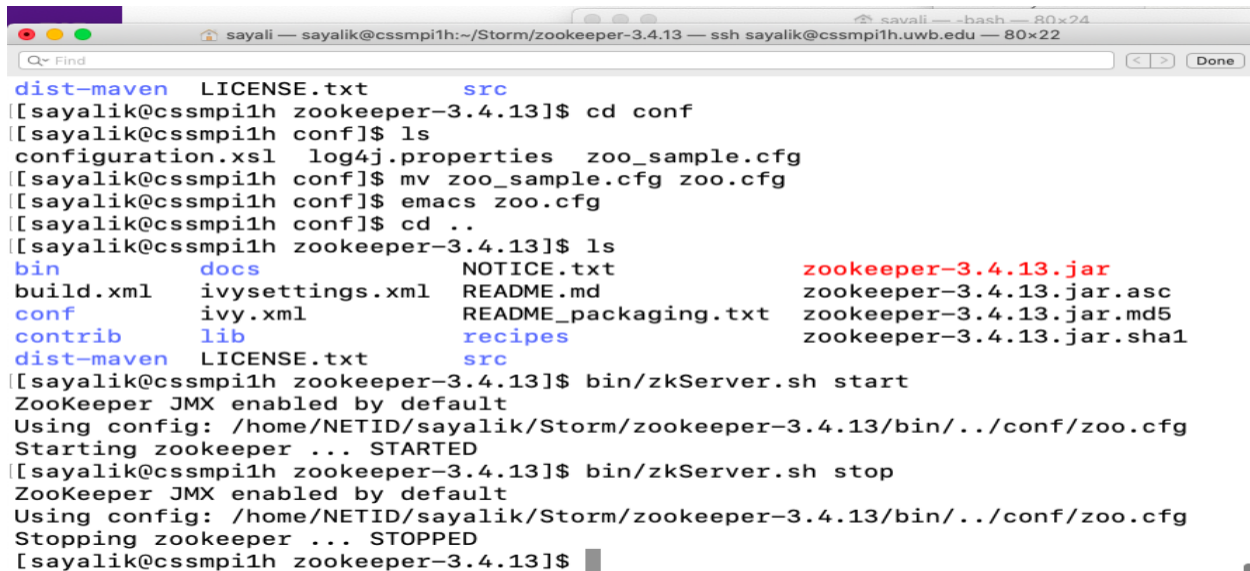
```
[sayalik@cssmpi1h bin]$ ./activemq stop
INFO: Using default configuration
(you can configure options in one of these file: /etc/default/activemq /home/NETID/sayalik/.activemqrc)

INFO: Invoke the following command to create a configuration file
./activemq setup [ /etc/default/activemq | /home/NETID/sayalik/.activemqrc ]

INFO: Using java '/etc/alternatives/jre/bin/java'
INFO: Waiting at least 30 seconds for regular process termination of pid '33748' :
Java Runtime: Red Hat, Inc. 11.0.10 /usr/lib/jvm/java-11-openjdk-11.0.10.9-1.el7_9.x86_64
Heap sizes: current=1048576k free=1046838k max=1048576k
JVM args: -Xms1G -Xmx1G -Djava.util.logging.config.file=logging.properties -Dhawtio.realm=activemq -Dhawtio.role=admins -Dhawt
io.rolePrincipalClasses=org.apache.activemq.jaas.GroupPrincipal -Djava.security.auth.login.config=/home/NETID/sayalik/lab4/apache-
activemq-5.9.0/conf/login.config -Dactivemq.classpath=/home/NETID/sayalik/lab4/apache-activemq-5.9.0/conf; -Dactivemq.home=/home/N
ETID/sayalik/lab4/apache-activemq-5.9.0 -Dactivemq.base=/home/NETID/sayalik/lab4/apache-activemq-5.9.0 -Dactivemq.conf=/home/NETID
/sayalik/lab4/apache-activemq-5.9.0/conf -Dactivemq.data=/home/NETID/sayalik/lab4/apache-activemq-5.9.0/data
Extensions classpath:
[ /home/NETID/sayalik/lab4/apache-activemq-5.9.0/lib, /home/NETID/sayalik/lab4/apache-activemq-5.9.0/lib/camel, /home/NETID/sayalik
/lab4/apache-activemq-5.9.0/lib/optional, /home/NETID/sayalik/lab4/apache-activemq-5.9.0/lib/web, /home/NETID/sayalik/lab4/apache-ac
tivemq-5.9.0/lib/extra ]
ACTIVEMQ_HOME: /home/NETID/sayalik/lab4/apache-activemq-5.9.0
ACTIVEMQ_BASE: /home/NETID/sayalik/lab4/apache-activemq-5.9.0
ACTIVEMQ_CONF: /home/NETID/sayalik/lab4/apache-activemq-5.9.0/conf
ACTIVEMQ_DATA: /home/NETID/sayalik/lab4/apache-activemq-5.9.0/data
Connecting to pid: 33748
Stopping broker: localhost
... FINISHED
[sayalik@cssmpi1h bin]$
```

Figure 12 : Lab 4 : Console output part 3

Lab5 output:



```
sayali — sayalik@cssmpi1h:~/Storm/zookeeper-3.4.13 — ssh sayalik@cssmpi1h.uwb.edu — 80x22
dist-maven LICENSE.txt src
[sayalik@cssmpi1h zookeeper-3.4.13]$ cd conf
[sayalik@cssmpi1h conf]$ ls
configuration.xml log4j.properties zoo_sample.cfg
[sayalik@cssmpi1h conf]$ mv zoo_sample.cfg zoo.cfg
[sayalik@cssmpi1h conf]$ emacs zoo.cfg
[sayalik@cssmpi1h conf]$ cd ..
[sayalik@cssmpi1h zookeeper-3.4.13]$ ls
bin docs NOTICE.txt zookeeper-3.4.13.jar
build.xml ivysettings.xml README.md zookeeper-3.4.13.jar.asc
conf ivy.xml README_packaging.txt zookeeper-3.4.13.jar.md5
contrib lib recipes zookeeper-3.4.13.jar.sha1
dist-maven LICENSE.txt src
[sayalik@cssmpi1h zookeeper-3.4.13]$ bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /home/NETID/sayalik/Storm/zookeeper-3.4.13/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[sayalik@cssmpi1h zookeeper-3.4.13]$ bin/zkServer.sh stop
ZooKeeper JMX enabled by default
Using config: /home/NETID/sayalik/Storm/zookeeper-3.4.13/bin/../conf/zoo.cfg
Stopping zookeeper ... STOPPED
[sayalik@cssmpi1h zookeeper-3.4.13]$
```

Figure 13 : Storm Installation

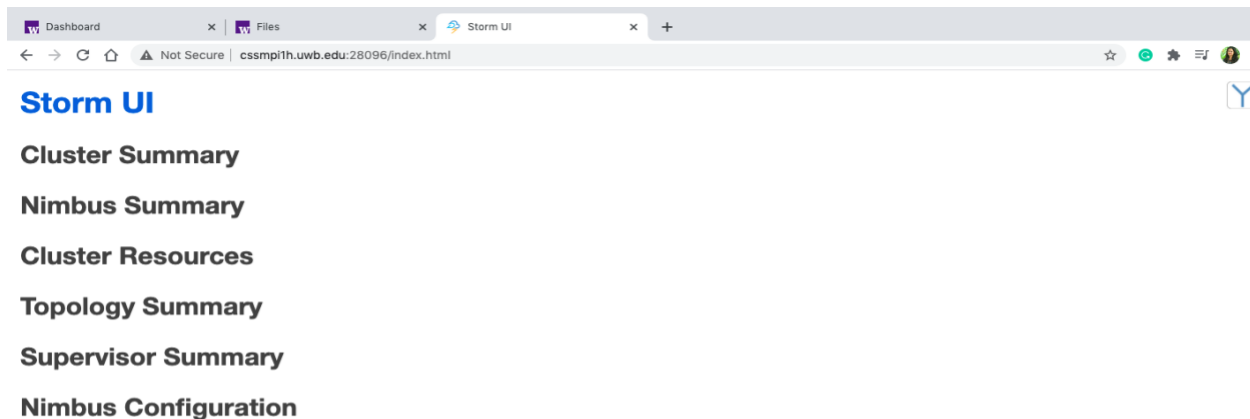


Figure 14 : Storm UI