

SMOKE TEST

HDF 3.3.0

Date Prepared: July 2019

Document Information

Project Name	HDF Smoke Test Document		
Project Owner		Document Version No	1.0
Quality Review Method	By email/HPE SharePoint		
Prepared By		Preparation Date	July 2019
Reviewed By	Refer to version history	Review Date	

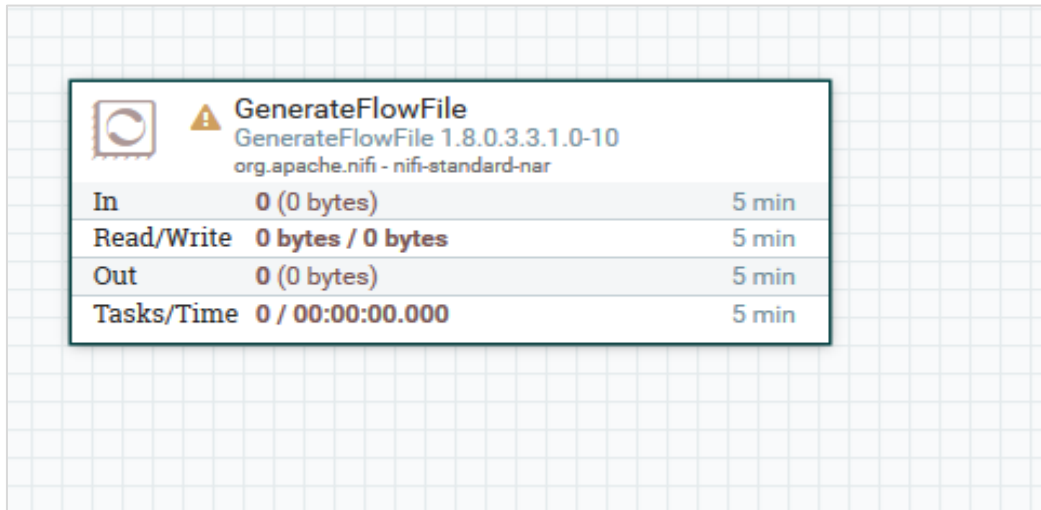
Table of Contents

1	TESTING NIFI	4
1.1	SELECT A GENERATEFLOWFILE PROCESSOR IN NIFI UI	4
1.2	CONFIGURE GENERATEFLOWFILE PROCESSOR TO GENERATE THE DATA.....	4
1.3	SELECT A PUTFILE PROCESSOR IN NIFI UI	5
1.4	CONFIGURE PUTFILE PROCESSOR TO STORE GENERATED DATA FROM GENERATEFLOWFILE PROCESSOR.....	5
1.5	CONNECT AND RUN THE PROCESSORS	6
1.6	VERIFY THE DATA	6
2	TESTING NIFI REGISTRY.....	7
2.1	CREATE HELLOWORLD BUCKET IN NIFI REGISTRY UI	7
2.2	CREATE A PROCESS GROUP	7
2.3	CONFIGURE A PROCESS GROUP	8
2.4	VERIFY VERSION CONTROL	8
3	TESTING GRAFANA	10
3.1	SELECTION OF COMPONENT	10
4	TESTING KAFKA	11
4.1	CREATE A NEW KAFKA TOPIC.....	11
4.2	CREATE A NEW TOPIC	11
5	TESTING STREAMING ANALYTICS MANAGER.....	12
5.1	GO TO THE STREAMLINE UI	12
5.2	CREATE A SERVICE POOL	12
5.3	CREATE A NEW ENVIRONMENT FOR HDF CLUSTER FROM SERVICE POOL	13
5.4	CREATE A NEW APPLICATION FOR YOUR ENVIRONMENT	13
6	TESTING STORM.....	15
6.1	DOWNLOAD STORM WORDCOUNT EXAMPLE	15
6.2	CHANGE DIRECTORY TO THE STORM-EXAMPLE	15
6.3	BUILDING JAR.....	15
6.4	CREATE TOPOLOGY FOR STORM WORDCOUNT EXAMPLE.....	16
6.5	VERIFY WORDCOUNT TOPOLOGY FROM STORM UI	17

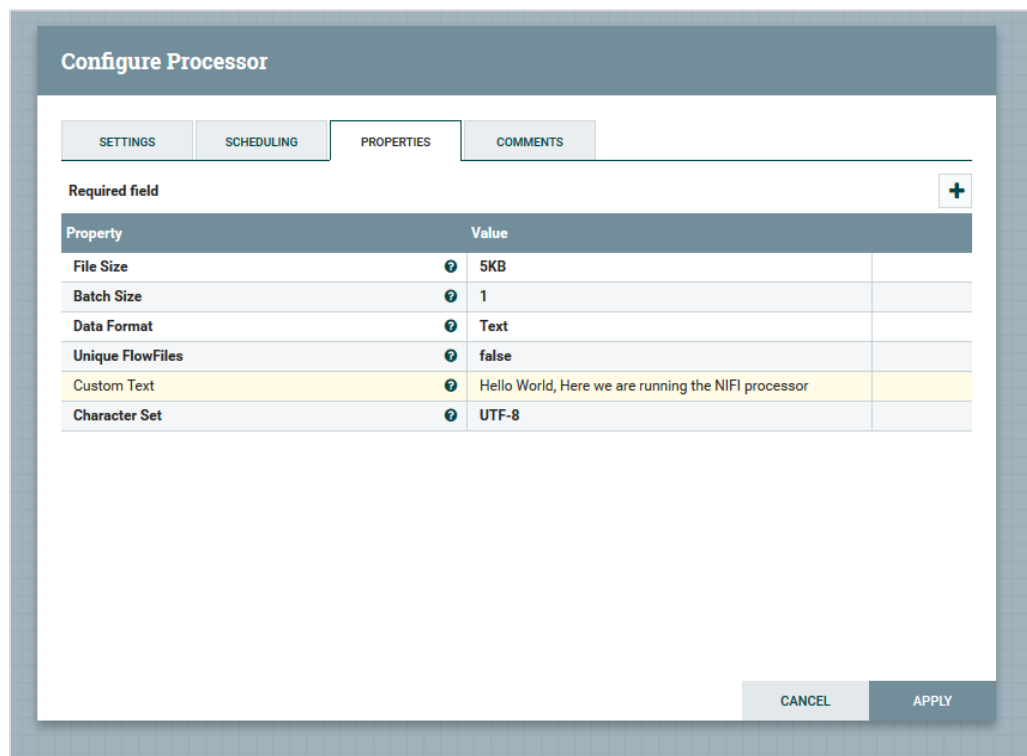
1 TESTING NIFI

We will use NIFI UI to create a HelloWorld dataflow.

1.1 Select a GenerateFlowFile processor in NIFI UI



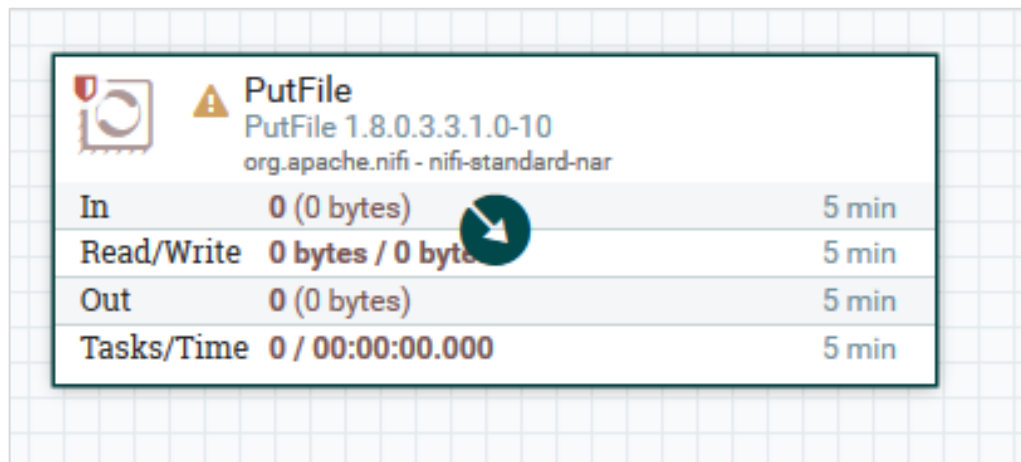
1.2 Configure GenerateFlowFile processor to generate the data



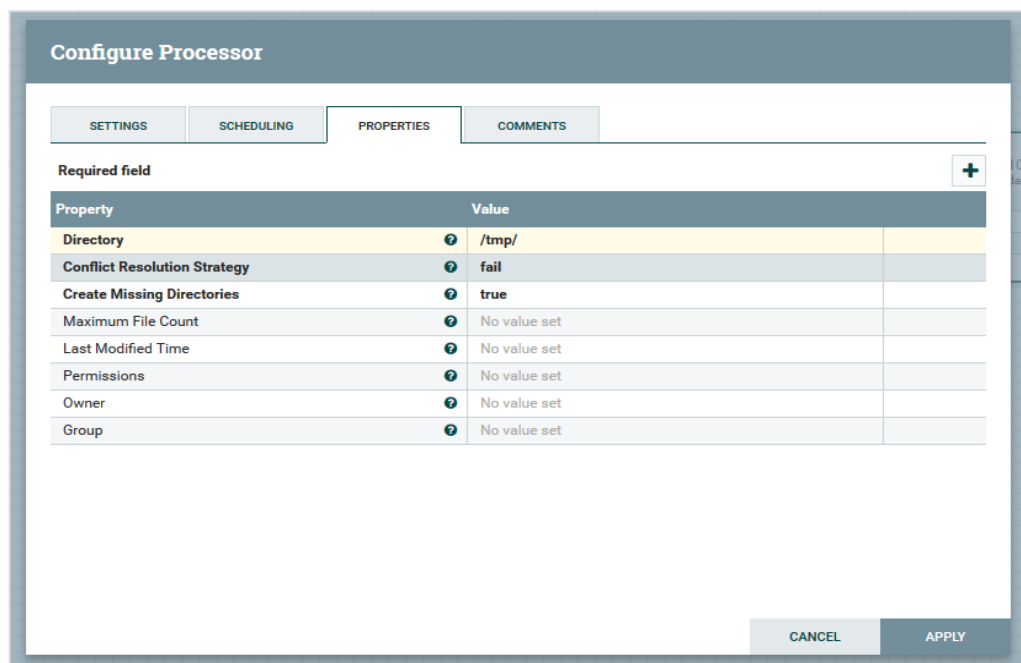
Following are the configuration you need to perform for GenerateFlowFile processor:

- Set the file size
- Enter custom text in Custom Text section

1.3 Select a PutFile processor in NIFI UI





1.4 Configure PutFile processor to store generated data from GenerateFlowFile processor



You need to perform following configuration for PutFile processor:

- Enter a location where you want to store generated data from GenerateFlowFile processor.
(Ex. /tmp/)

1.5 Connect and run the processors

 GenerateFlowFile GenerateFlowFile 1.8.0.3.3.1.0-10 org.apache.nifi - nifi-standard-nar	 PutFile PutFile 1.8.0.3.3.1.0-10 org.apache.nifi - nifi-standard-nar
In 0 (0 bytes) 5 min	In 6 (306 bytes) 5 min
Read/Write 0 bytes / 306 bytes 5 min	Read/Write 306 bytes / 306 bytes 5 min
Out 6 (306 bytes) 5 min	Out 0 (0 bytes) 5 min
Tasks/Time 6 / 00:00:00.025 5 min	Tasks/Time 6 / 00:00:00.032 5 min

1.6 Verify the data

Go to NIFI container and check into **/tmp** directory, if the data is stored or not.

```
ls /tmp/
```

```
[bluedata@bluedata-6670 ~]$ ls /tmp/
0f415ed5-b405-422c-89a6-69bd68f17bcf  6ae13641-8aef-4eab-9e0a-bc53e70a3f1d  bds-20190616225559.log  eella6ce-974d-4276-bd1f-5b4b54495e66
1c96755c-209b-4690-b8b5-71e9186d50fc  6f5eef59-f810-4a31-a838-ced3e4f86650  bd_vagent_bundle.517.log  efc2534f-e25f-412d-a0f5-309bc920b95a
2019466d-1b12-44f9-83f2-033252867b9f  7852fa6c-d820-41f1-bb38-34158882a2d0  cfa81c94-579d-48fb-917b-abb48bfd6e04  fe448eb1-b30b-493a-af79-4cb05730d89c
23994186-4bef-4f26-9638-074eb4b9e20b  787f475d-c6a4-4f2a-ab98-e786f0cd5212  d6c6e001-57c2-41f5-aabc-ebdc894122f0  hsperrdata_mynifiuser
3540a303-9e91-42ad-a66c-ada17c3f1673  87111caa-0238-4323-867b-9602378bbc5c  dbbab28a-5666-4f93-b992-904a7265fe63  hsperrdata_registry
358ef550-c2cc-42fb-837e-e71b75422b2e  8c0c012e-ed1c-4eab-9027-903d5b49f2b7  de538e75-172d-4631-9684-cda97eb5d315  hsperrdata_root
4f44eda8-5801-43ae-965d-fded68baa078  8d970764-d334-4f9d-b964-3c2f6c1e428d  e3a9c207-d4dc-44df-88a5-90f30e6ad659  jna-551163445
647c88ff-ea0f-4536-9612-2ce45f79992c  a2c32c88-76bb-4210-a9ad-f1f43d3c44b6  e9e1a5f6-915e-4438-aa4d-c25ea7985a37  snappy-1.0.5-libsnapppyjava.so
65b8d7ee-9635-4de4-9d2e-8622f8504fb1  a477f930-f8b8-4873-8cbd-a2e73a10e849  ec394c58-beda-469a-a17f-fd83a262ef70
```

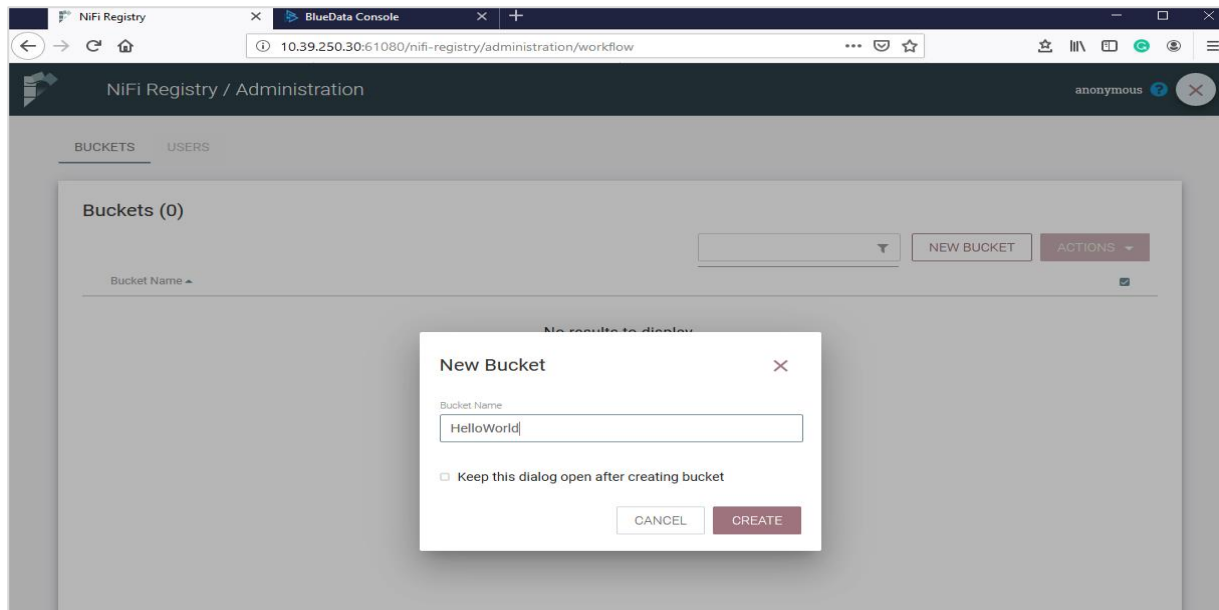
Use **cat** command to check the stored data.

```
[bluedata@bluedata-6670 ~]$ cat /tmp/0f415ed5-b405-422c-89a6-69bd68f17bcf
Hello World, Here we are running the NIFI processor[bluedata@bluedata-6670 ~]$
[bluedata@bluedata-6670 ~]$
```

2 TESTING NIFI REGISTRY

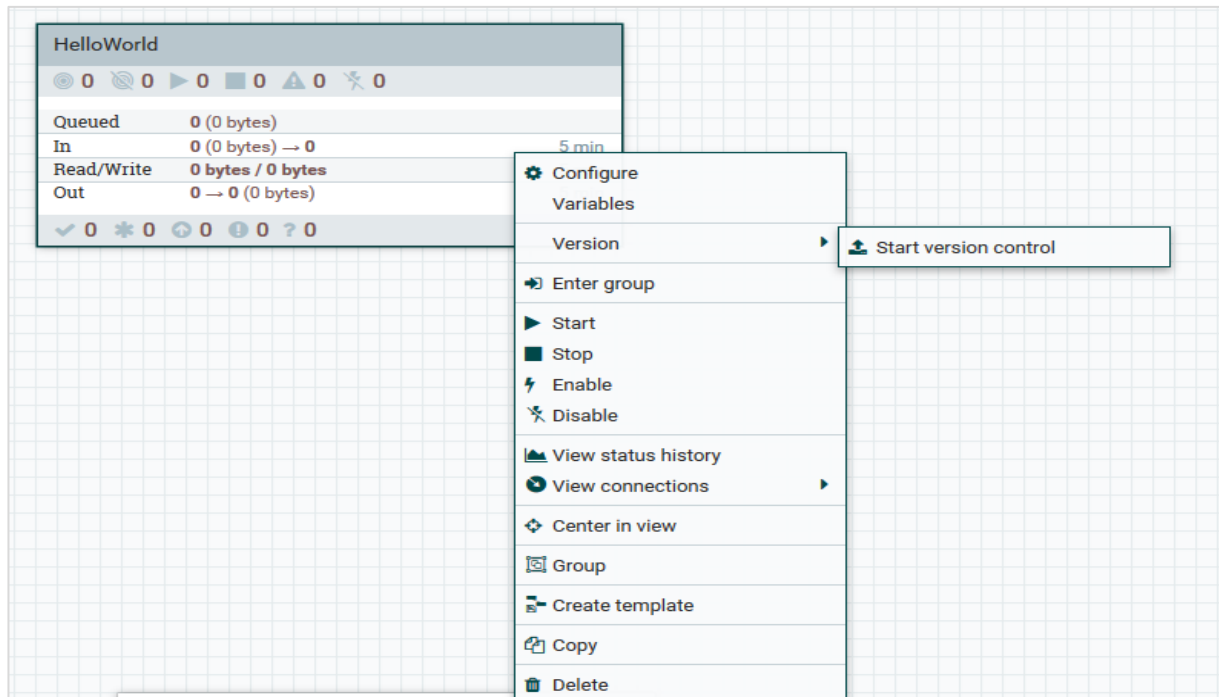
Here we will test the NIFI Registry service. Go to the NIFI Registry UI and create a bucket called HelloWorld.

2.1 Create HelloWorld bucket in NIFI Registry UI



2.2 Create a process group

Create a HelloWorld process group in NIFI UI and start version control.



2.3 Configure a process group

Configure HelloWorld process group for storing it into bucket HelloWorld in NIFI Registry.

Save Flow Version

Registry

bluedata-6700.bdlocal

Bucket

HelloWorld

Flow Name

HelloWorld

1

Flow Description

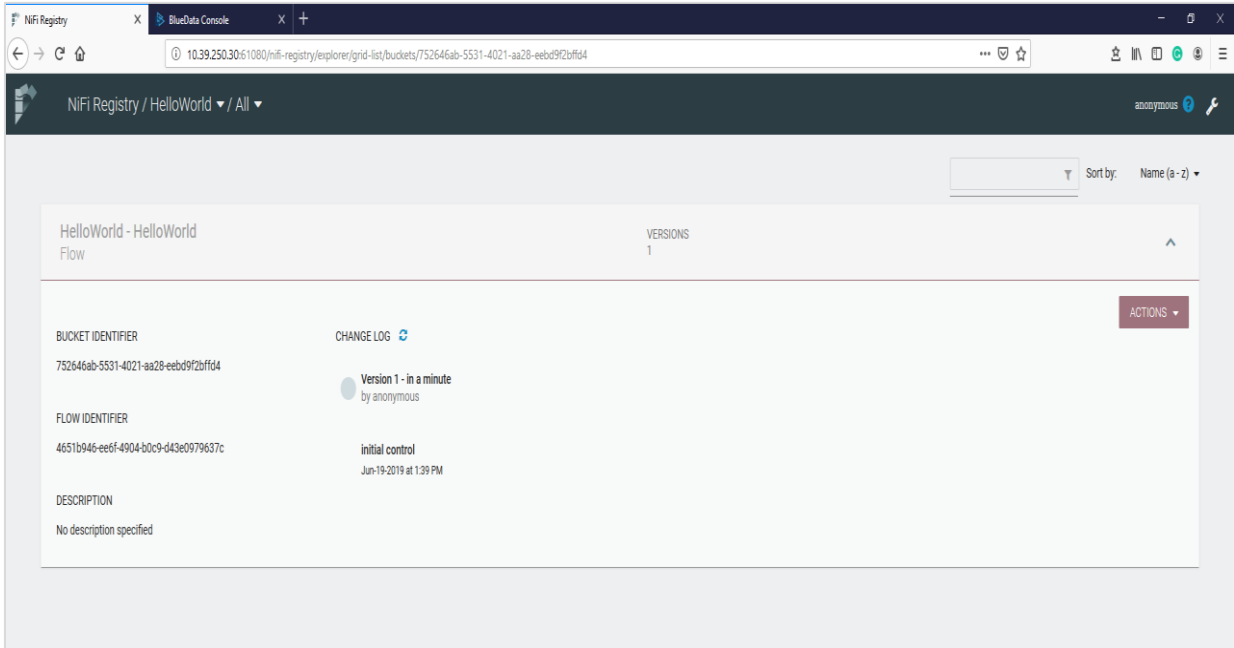
Version Comments

CANCEL

SAVE

2.4 Verify version control

Go to the NIFI Registry UI. Select HelloWorld bucket and check if version control started for process group HelloWorld.



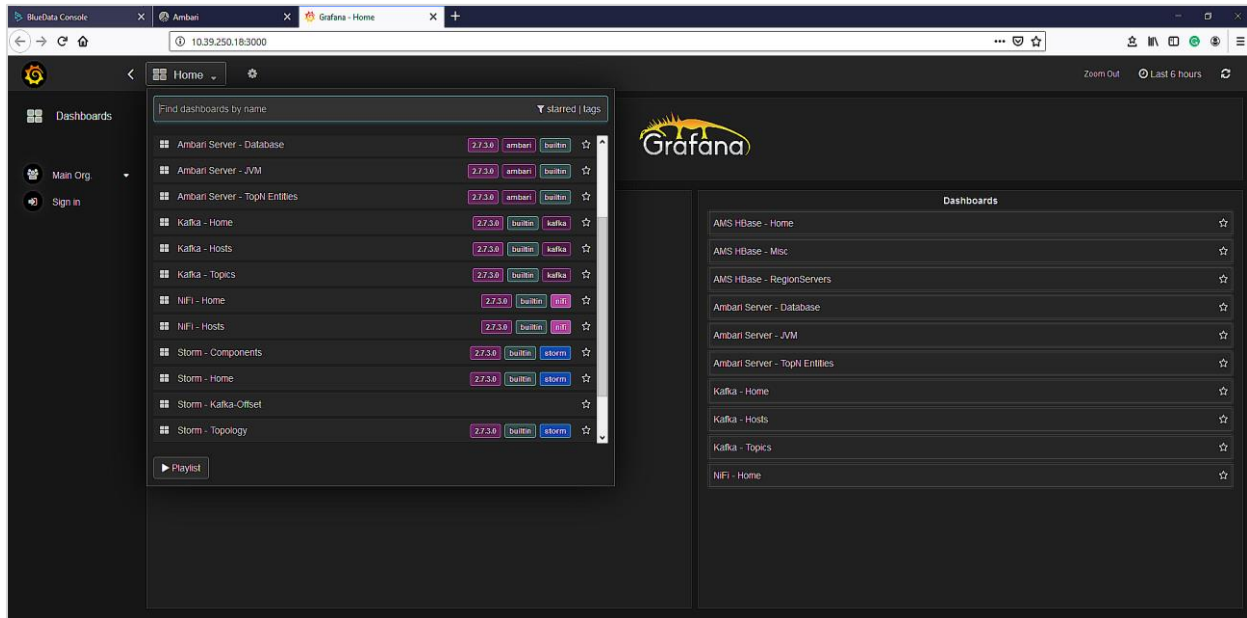
The screenshot displays the NiFi Registry web application. The browser's address bar shows the URL: `10.39.250.30:1080/nifi-registry/explorer/grid-list/buckets/752646ab-5531-4021-aa28-eebd9f2bffd4`. The page title is "NiFi Registry / HelloWorld / All". The main content area shows details for a flow named "HelloWorld - HelloWorld Flow".

HelloWorld - HelloWorld Flow		VERSIONS 1
BUCKET IDENTIFIER	752646ab-5531-4021-aa28-eebd9f2bffd4	<div>Version 1 - in a minute by anonymous</div> <div>initial control Jun-19-2019 at 1:39 PM</div>
FLOW IDENTIFIER	4651b946-ee6f-4904-b0c9-d43e0979637c	
DESCRIPTION	No description specified	

Additional UI elements include a "CHANGE LOG" link, an "ACTIONS" dropdown menu, and a "Sort by: Name (a-z)" filter.

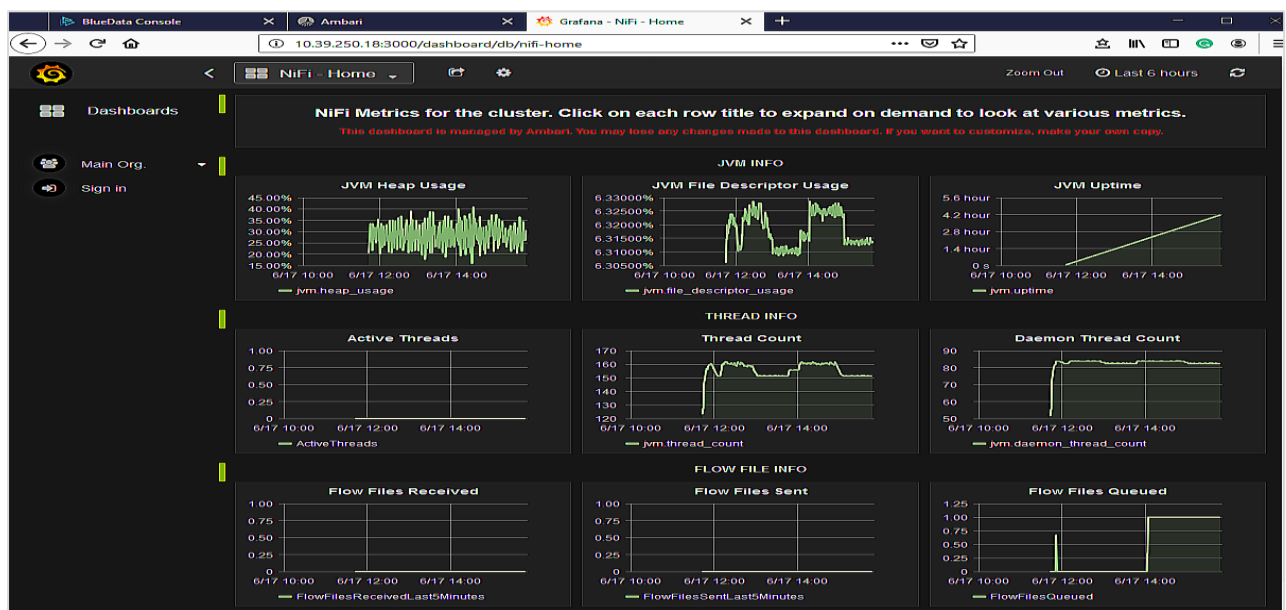
3 TESTING GRAFANA

Go to the Grafana UI and click on the search icon. Here you can see the components which is deployed into the HDF cluster.



3.1 Selection of component

NIFI – Home component is selected in Grafana UI, and you can see the resources usages by the NIFI service.



4 TESTING KAFKA

Go inside the Kafka container and follow the below steps.

4.1 Create a new Kafka topic

Create new Kafka topic inside the Kafka container.

4.2 Create a new topic

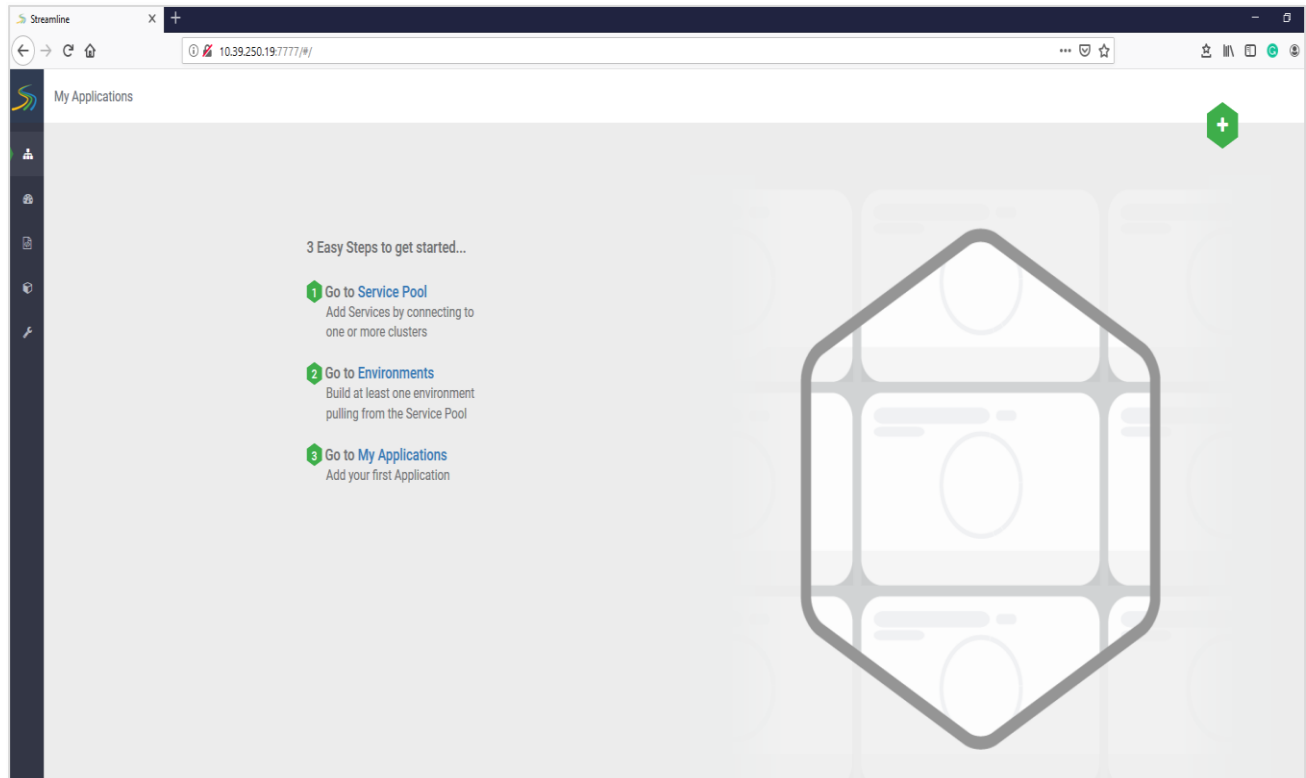
Execute the below command to create a new topic "SDS"

```
bin/Kafka-topics --create --zookeeper <ip address of zookeeper>:2181 --  
replication-factor 1 --partitions 1 --topic SDS
```

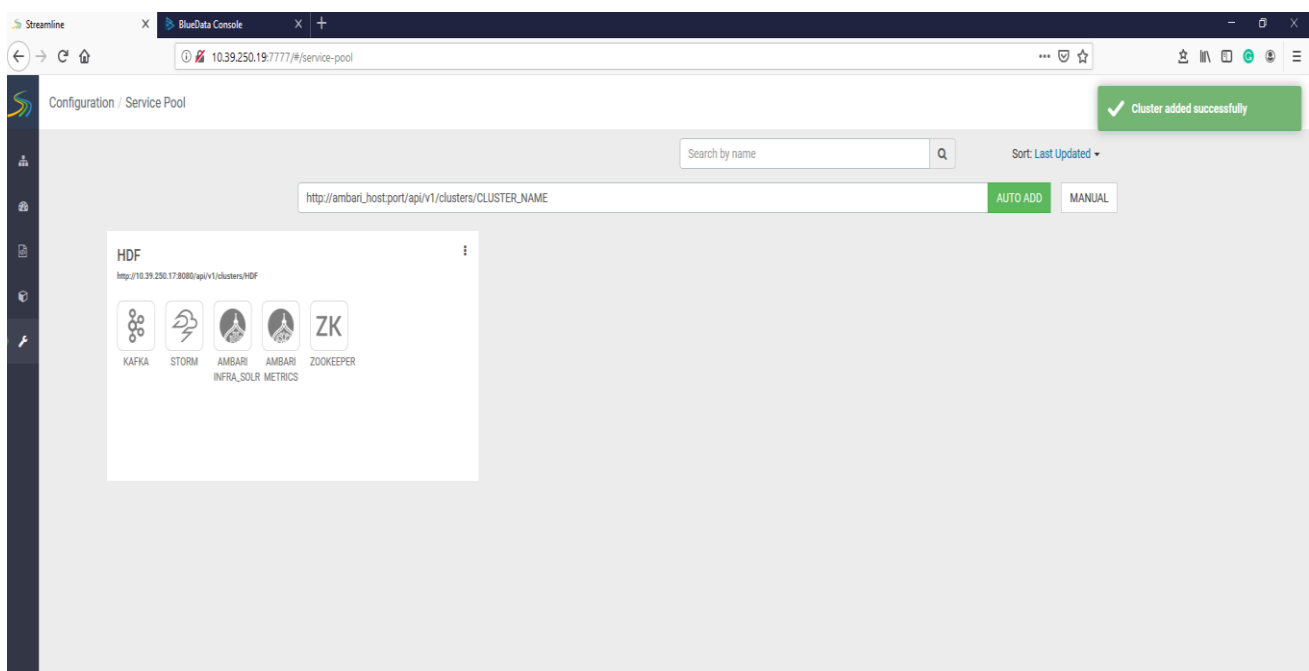
```
[bluedata@bluedata-6711 kafka]$ ls bin/  
connect-distributed.sh    kafka-configs.sh          kafka-delegation-tokens.sh  kafka-preferred-replica-elec  
connect-standalone.sh    kafka-console-consumer.sh  kafka-delete-records.sh     kafka-producer-perf-test.sh  
kafka                    kafka-console-producer.sh  kafka-dump-log.sh           kafka-reassign-partitions.sh  
kafka-acls.sh            kafka-consumer-groups.sh   kafka-log-dirs.sh           kafka-replica-verification.s  
kafka-broker-api-versions.sh  kafka-consumer-perf-test.sh kafka-mirror-maker.sh       kafka-run-class.sh  
[bluedata@bluedata-6711 kafka]$  
[bluedata@bluedata-6711 kafka]$ ./bin/kafka-topics.sh --create --zookeeper 10.39.250.17,10.39.250.30,10.39.250.28:2  
181 --replication-factor 1 --partitions 1 --topic SDS  
Created topic "SDS".  
[bluedata@bluedata-6711 kafka]$
```

5 TESTING STREAMING ANALYTICS MANAGER

5.1 Go to the Streamline UI



5.2 Create a service pool

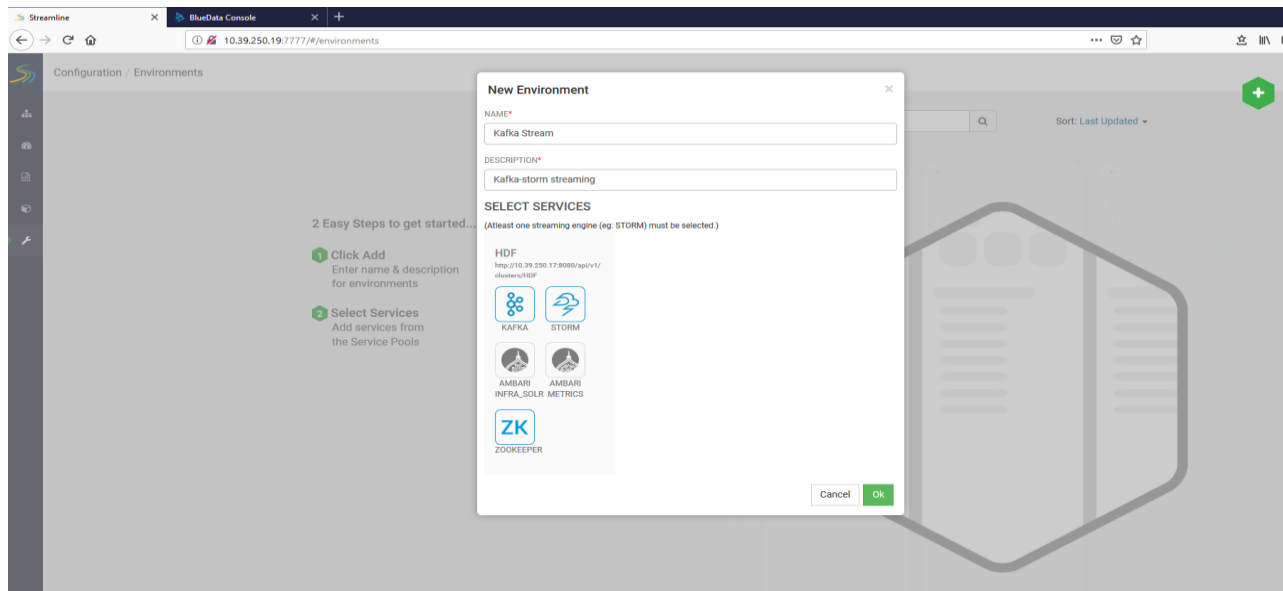


Perform below steps to create a service pool for HDF Cluster:

Enter url for HDF cluster into url bar and click on AUTO ADD button.

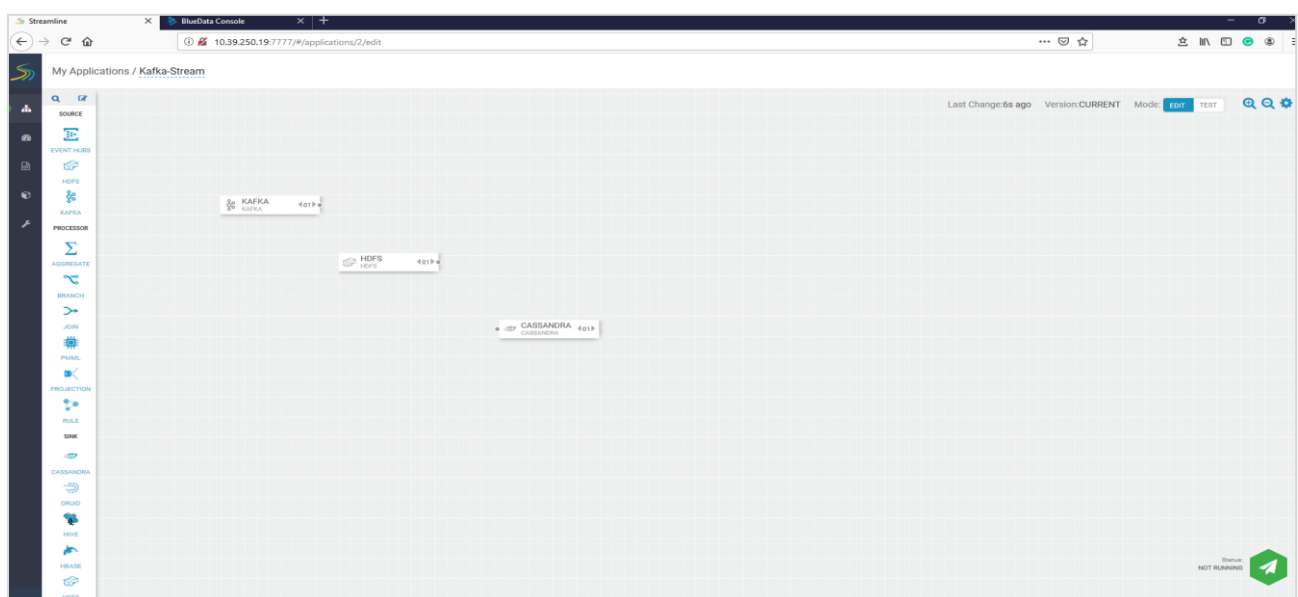
(Ex.: <http://<ip address of ambari server>:8080/api/v1/clusters/HDF>)

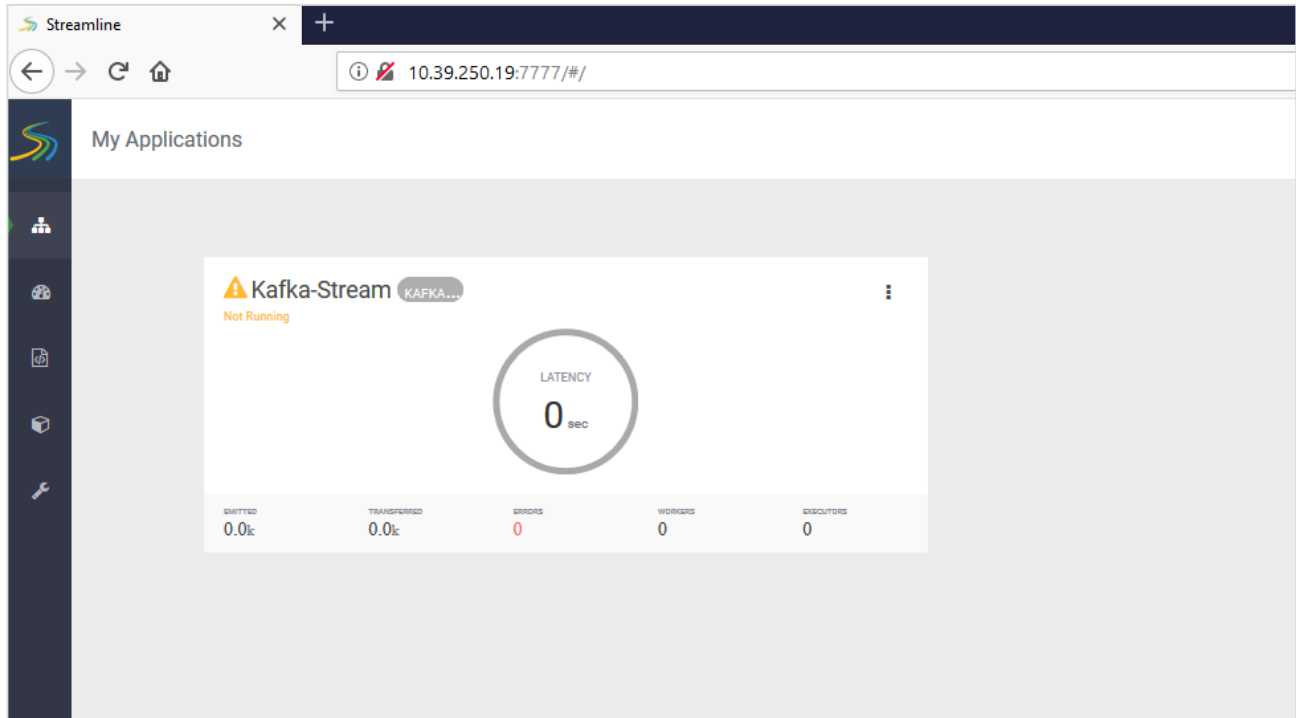
5.3 Create a new environment for HDF cluster from service pool



5.4 Create a new application for your environment

Here you can create and design your application. After running your application you can verify your application under application section.





6 TESTING STORM

Here we will create a WordCount topology for Storm. We will download a WordCount example and create a WordCount topology.

Note: Install git and maven (if not installed), by executing the following.

```
yum install git -y
```

```
yum install maven -y
```

6.1 Download Storm WordCount example

Execute the below command to download Storm WordCount example

```
git clone https://github.com/ADMICloud/examples.git
```

```
[bluedata@bluedata-6896 ~]$  
[bluedata@bluedata-6896 ~]$ git clone https://github.com/ADMICloud/examples.git  
Cloning into 'examples'...  
remote: Enumerating objects: 152, done.  
remote: Total 152 (delta 0), reused 0 (delta 0), pack-reused 152  
Receiving objects: 100% (152/152), 34.88 KiB | 0 bytes/s, done.  
Resolving deltas: 100% (34/34), done.  
[bluedata@bluedata-6896 ~]$  
[bluedata@bluedata-6896 ~]$
```

6.2 Change directory to the storm-example

```
cd examples/storm-example/
```

```
[bluedata@bluedata-6896 ~]$ ls  
examples  vagent.bin  
[bluedata@bluedata-6896 ~]$ cd examples/storm-example/  
[bluedata@bluedata-6896 storm-example]$  
[bluedata@bluedata-6896 storm-example]$  
[bluedata@bluedata-6896 storm-example]$
```

6.3 Building jar

Execute the below command to build jar for Storm WordCount example.

```
mvn clean install
```

```
[bluedata@bluedata-6896 storm-example]$  
[bluedata@bluedata-6896 storm-example]$ mvn clean install  
[INFO] Scanning for projects...  
[INFO]  
[INFO] -----  
[INFO] Building storm-example 1.0  
[INFO] -----  
[INFO]  
[INFO] --- maven-clean-plugin:2.4.1:clean (default-clean) @ storm-example ---  
[INFO]
```

Note: After executing this command a new directory target will be created where you can find Storm WordCount example jar file. Use ls command to verify.

```
[bluedata@bluedata-6896 storm-example]$ ls  
pom.xml src target  
[bluedata@bluedata-6896 storm-example]$  
[bluedata@bluedata-6896 storm-example]$  
[bluedata@bluedata-6896 storm-example]$ ls target/  
archive-tmp classes generated-sources maven-archiver maven-status storm-example-1.0.jar storm-example-1.0-jar-with-dependencies.jar surefire  
[bluedata@bluedata-6896 storm-example]$
```

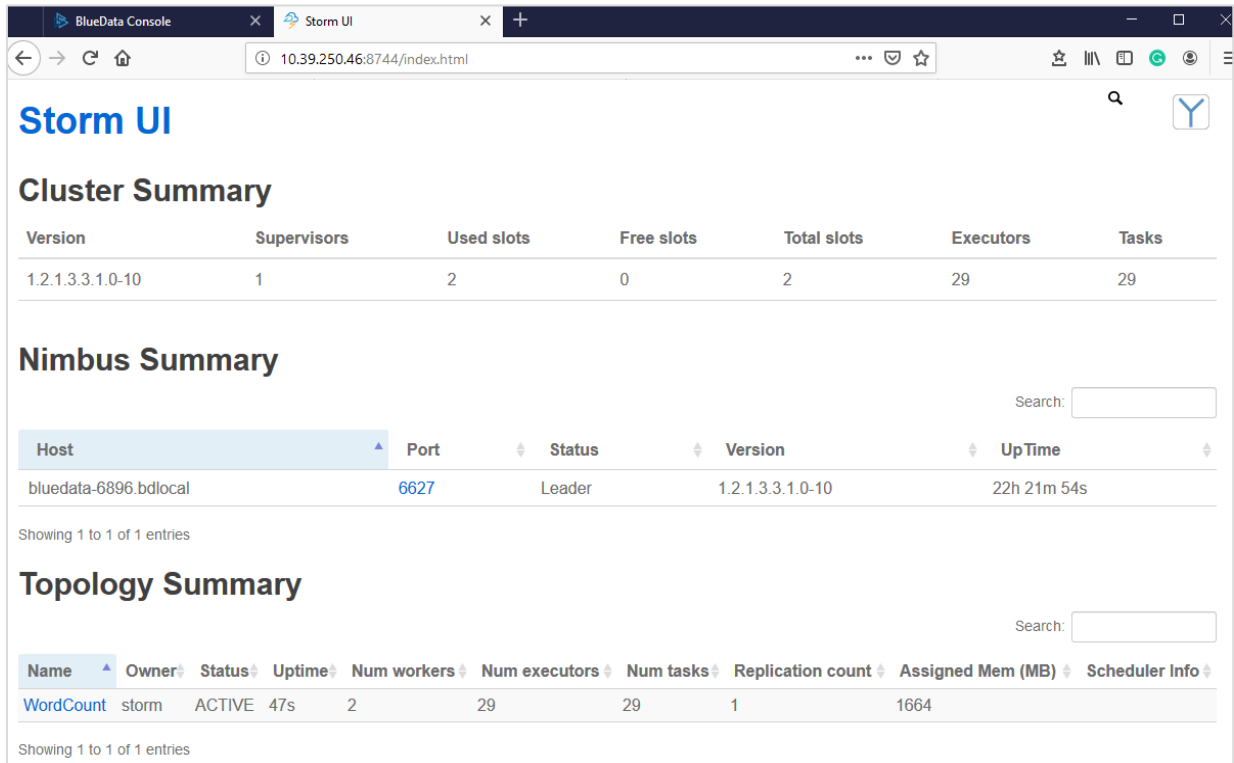
6.4 Create topology for Storm WordCount example

Execute the below command to create topology for Storm WordCount example. Navigate to `/usr/hdf/3.3.0.0-165/storm` directory to execute the command.

```
./bin/storm jar /home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar  
admicloud.storm.wordcount.WordCountTopology WordCount
```

```
[bluedata@bluedata-6896 storm]$ ./bin/storm jar /home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar admicloud.s  
torm.wordcount.WordCountTopology WordCount  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/usr/hdf/3.3.1.0-10/storm/lib/log4j-slf4j-impl-2.8.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: Found binding in [jar:file:/home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar!/org/slf4j/impl/StaticLog  
gerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]  
Running: /usr/java/default/bin/java -Ddaemon.name= -Dstorm.options= -Dstorm.home=/usr/hdf/3.3.1.0-10/storm -Dstorm.log.dir=/var/log/storm -Djava.lib  
rary.path=/usr/local/lib:/opt/local/lib:/usr/lib -Dstorm.conf.file= -cp /usr/hdf/3.3.1.0-10/storm/*:/usr/hdf/3.3.1.0-10/storm/lib/*:/usr/hdf/3.3.1.0  
-10/storm/extlib/*:/home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar:/usr/hdf/current/storm-supervisor/conf:/  
usr/hdf/3.3.1.0-10/storm/bin -Dstorm.jar=/home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar -Dstorm.dependency  
.jars= -Dstorm.dependency.artifacts={} admicloud.storm.wordcount.WordCountTopology WordCount  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/usr/hdf/3.3.1.0-10/storm/lib/log4j-slf4j-impl-2.8.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: Found binding in [jar:file:/home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar!/org/slf4j/impl/StaticLog  
gerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]  
741 [main] WARN o.a.s.u.Utils - STORM-VERSION new 1.2.1.3.3.1.0-10 old null  
777 [main] INFO o.a.s.StormSubmitter - Generated ZooKeeper secret payload for MD5-digest: -6269264375147352907:-8793272926567057743  
894 [main] INFO o.a.s.u.NimbusClient - Found leader nimbus : bluedata-6896.bdlocal:6627  
918 [main] INFO o.a.s.s.a.AuthUtils - Got AutoCreds []  
921 [main] INFO o.a.s.u.NimbusClient - Found leader nimbus : bluedata-6896.bdlocal:6627  
941 [main] INFO o.a.s.StormSubmitter - Uploading dependencies - jars...  
942 [main] INFO o.a.s.StormSubmitter - Uploading dependencies - artifacts...  
942 [main] INFO o.a.s.StormSubmitter - Dependency Blob keys - jars : [] / artifacts : []  
947 [main] INFO o.a.s.StormSubmitter - Uploading topology jar /home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies  
.jar to assigned location: /hadoop/storm/nimbus/inbox/stormjar-a9a53b08-926d-4bb0-a096-08654df6cbad.jar  
Start uploading file '/home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar' to '/hadoop/storm/nimbus/inbox/storm  
jar-a9a53b08-926d-4bb0-a096-08654df6cbad.jar' (165697 bytes)  
[=====] 165697 / 165697  
File '/home/bluedata/examples/storm-example/target/storm-example-1.0-jar-with-dependencies.jar' uploaded to '/hadoop/storm/nimbus/inbox/stormjar-a9a  
53b08-926d-4bb0-a096-08654df6cbad.jar' (165697 bytes)  
967 [main] INFO o.a.s.StormSubmitter - Successfully uploaded topology jar to assigned location: /hadoop/storm/nimbus/inbox/stormjar-a9a53b08-926d-  
4bb0-a096-08654df6cbad.jar  
967 [main] INFO o.a.s.StormSubmitter - Submitting topology WordCount in distributed mode with conf {"storm.zookeeper.topology.auth.scheme":"digest  
","storm.zookeeper.topology.auth.payload":"-6269264375147352907:-8793272926567057743","topology.workers":3,"topology.debug":true}  
967 [main] WARN o.a.s.u.Utils - STORM-VERSION new 1.2.1.3.3.1.0-10 old 1.2.1.3.3.1.0-10  
1201 [main] INFO o.a.s.StormSubmitter - Finished submitting topology: WordCount
```


6.5 Verify WordCount topology from Storm UI



The screenshot shows the Storm UI interface in a web browser. The browser tabs include 'BlueData Console' and 'Storm UI'. The address bar shows '10.39.250.46:8744/index.html'. The page title is 'Storm UI'. The 'Cluster Summary' section displays the following data:

Version	Supervisors	Used slots	Free slots	Total slots	Executors	Tasks
1.2.1.3.3.1.0-10	1	2	0	2	29	29

The 'Nimbus Summary' section shows a search bar and a table with the following data:

Host	Port	Status	Version	UpTime
bluedata-6896.bdlocal	6627	Leader	1.2.1.3.3.1.0-10	22h 21m 54s

Below the Nimbus Summary table, it says 'Showing 1 to 1 of 1 entries'.

The 'Topology Summary' section shows a search bar and a table with the following data:

Name	Owner	Status	Uptime	Num workers	Num executors	Num tasks	Replication count	Assigned Mem (MB)	Scheduler Info
WordCount	storm	ACTIVE	47s	2	29	29	1	1664	

Below the Topology Summary table, it says 'Showing 1 to 1 of 1 entries'.

Note: Under Topology Summary WordCount topology is created.