Apache Spark on Kubernetes

WordCount + PageRank + GKE

Yubraj Niraula
07/03/2024

Table of Contents

Introduction

- Platform used: Google Cloud Platform
- Technologies used:
  - Google Kubernetes Engine
  - Pyspark
- Functions:
  - Word Count
  - Page Rank

Design

- Apache Spark + Python = PySpark
- Apache Spark: An open-source tool for large-scale data processing, speeding up tasks like counting words or ranking webpages.
- Advantages of Apache Spark:
  - Speed: Spark excels at processing big data thanks to its in-memory computing, making it significantly faster than traditional systems.
  - Ease of use: Spark offers user-friendly APIs for working with large datasets, simplifying complex tasks for developers.
  - Versatility: Spark is a one-stop shop for various data tasks, including real-time analytics, machine learning, and traditional batch processing.

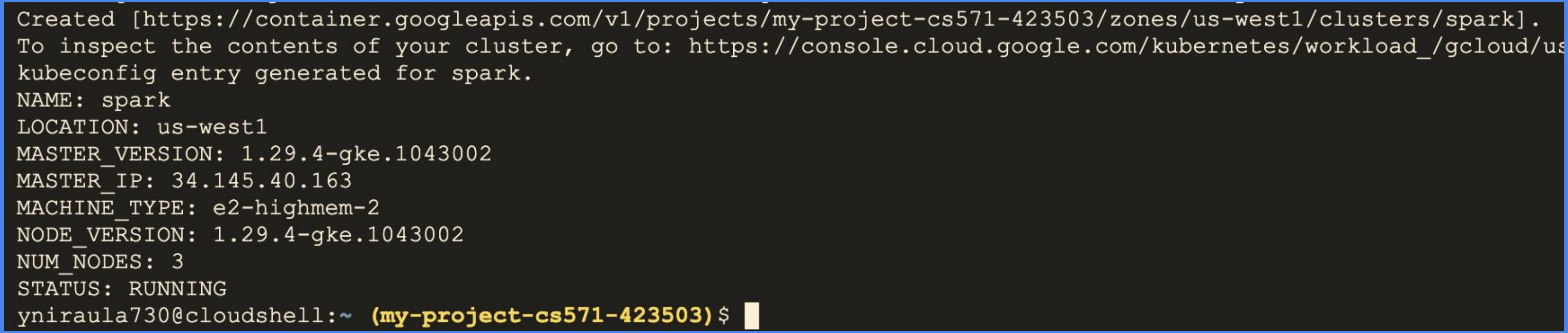
Mechanism

This project uses PySpark which is run on a system called Apache Spark, which itself is running on another system called Kubernetes. When we start a PySpark program, it talks directly to Kubernetes, which assigns resources for the main program (driver). The main program then works with Kubernetes to launch additional programs (executors) to help it with the work. These extra programs each run in their own isolated container. Kubernetes can automatically add or remove helper programs (executors) based on how much work there is to do, or we can set a fixed number of helpers to use.

Implementation

1. Create a cluster on GKE

```
$ gcloud container clusters create spark --num-nodes=1 --machine-type=e2-highmem-2 --region=us-west1
```

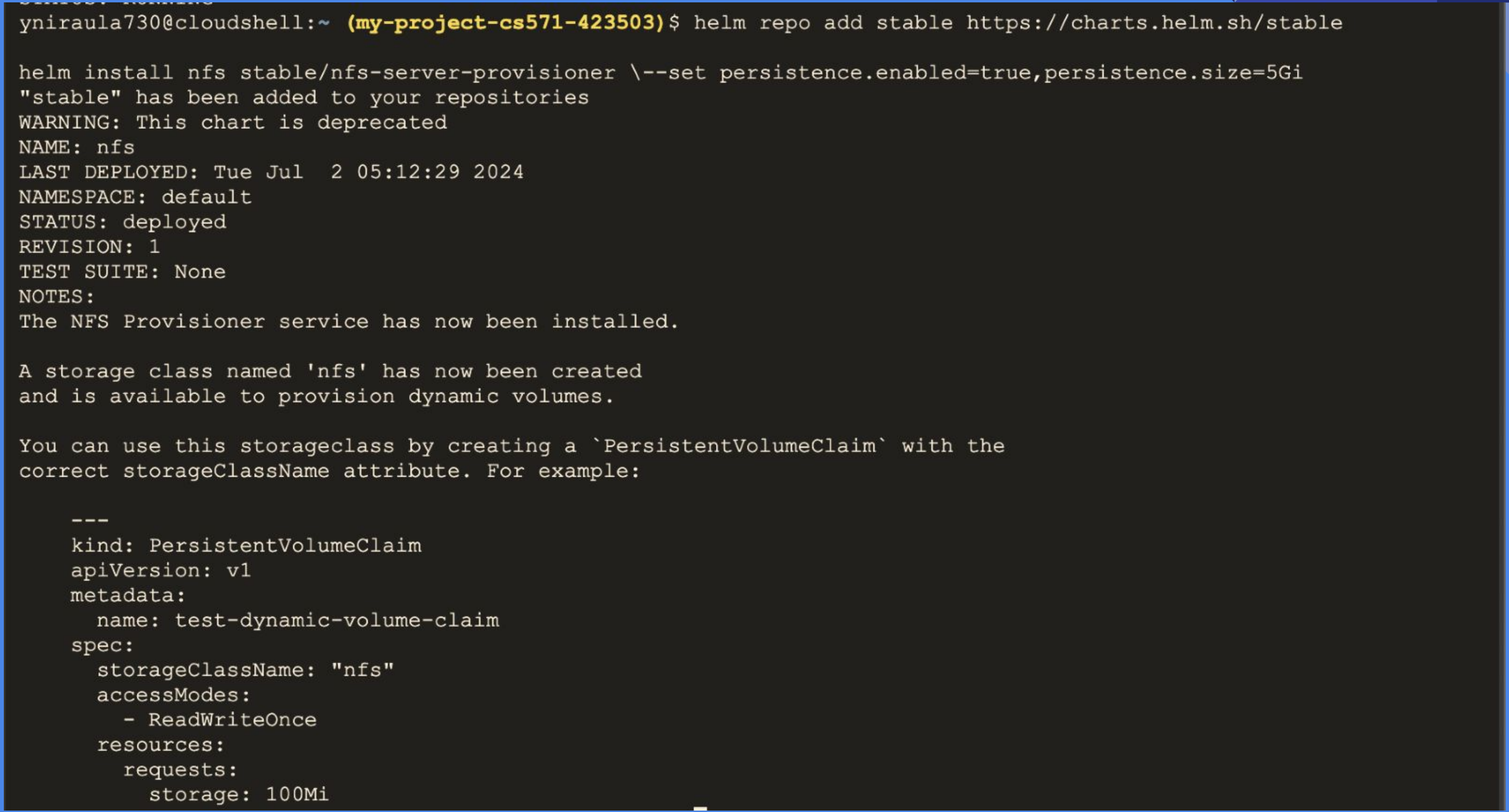


```
Created [https://container.googleapis.com/v1/projects/my-project-cs571-423503/zones/us-west1/clusters/spark].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us
kubeconfig entry generated for spark.
NAME: spark
LOCATION: us-west1
MASTER_VERSION: 1.29.4-gke.1043002
MASTER_IP: 34.145.40.163
MACHINE_TYPE: e2-highmem-2
NODE_VERSION: 1.29.4-gke.1043002
NUM_NODES: 3
STATUS: RUNNING
yniraula730@cloudshell:~ (my-project-cs571-423503)$
```

2. Install the NFS Server Provisioner

```
$ helm repo add stable https://charts.helm.sh/stable
$ helm install nfs stable/nfs-server-provisioner \--set persistence.enabled=true,persistence.size=5Gi
```

Implementation



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ helm repo add stable https://charts.helm.sh/stable

helm install nfs stable/nfs-server-provisioner \--set persistence.enabled=true,persistence.size=5Gi
"stable" has been added to your repositories
WARNING: This chart is deprecated
NAME: nfs
LAST DEPLOYED: Tue Jul  2 05:12:29 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The NFS Provisioner service has now been installed.

A storage class named 'nfs' has now been created
and is available to provision dynamic volumes.

You can use this storageclass by creating a `PersistentVolumeClaim` with the
correct storageClassName attribute. For example:

    ---
    kind: PersistentVolumeClaim
    apiVersion: v1
    metadata:
      name: test-dynamic-volume-claim
    spec:
      storageClassName: "nfs"
      accessModes:
        - ReadWriteOnce
      resources:
        requests:
          storage: 100Mi
```

Implementation

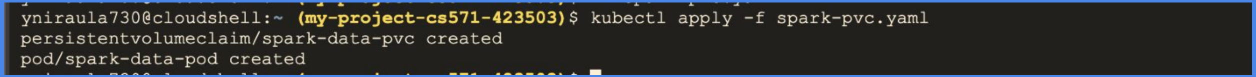
3. Create a persistent disk volume and a pod to use NFS

```
$ vi spark-pvc.yaml

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: spark-data-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 2Gi
  storageClassName: nfs
---
apiVersion: v1
kind: Pod
metadata:
  name: spark-data-pod
spec:
  volumes:
    - name: spark-data-pv
      persistentVolumeClaim:
        claimName: spark-data-pvc
  containers:
    - name: inspector
      image: bitnami/minideb
      command:
        - sleep
        - infinity
      volumeMounts:
        - mountPath: "/data"
          name: spark-data-pv
```

Implementation

4. Apply the yaml descriptor

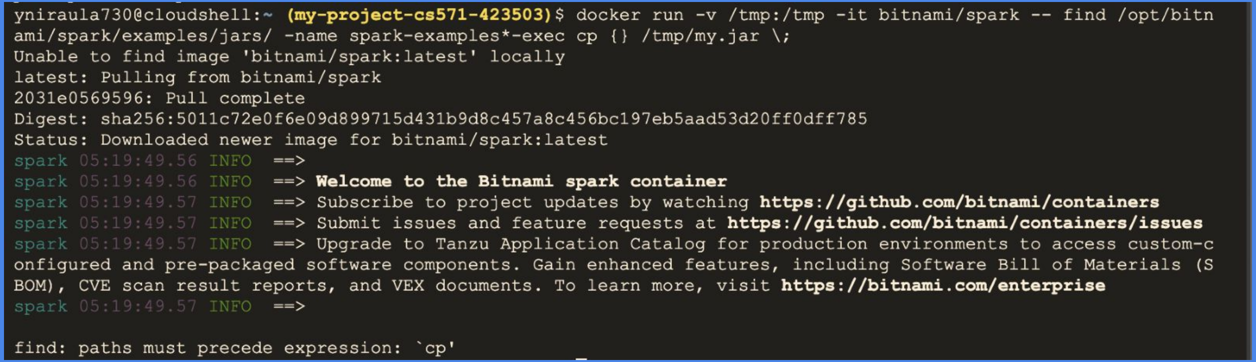
$ kubectl apply -f spark-pvc.yaml



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl apply -f spark-pvc.yaml
persistentvolumeclaim/spark-data-pvc created
pod/spark-data-pod created
```

4. Create and prepare your application JAR file

$ docker run -v /tmp:/tmp -it bitnami/spark -- find /opt/bitnami/spark/examples/jars/ -name spark-examples* -exec cp {} /tmp/my.jar \;

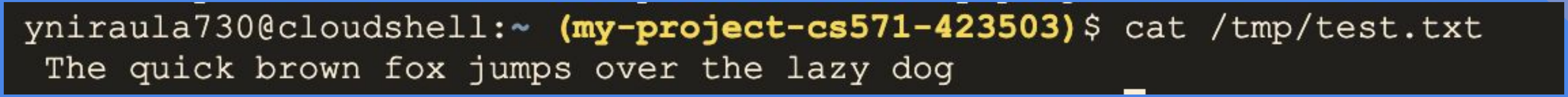


```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ docker run -v /tmp:/tmp -it bitnami/spark -- find /opt/bitn
ami/spark/examples/jars/ -name spark-examples*-exec cp {} /tmp/my.jar \;
Unable to find image 'bitnami/spark:latest' locally
latest: Pulling from bitnami/spark
2031e0569596: Pull complete
Digest: sha256:5011c72e0f6e09d899715d431b9d8c457a8c456bc197eb5aad53d20ff0dff785
Status: Downloaded newer image for bitnami/spark:latest
spark 05:19:49.56 INFO  ==>
spark 05:19:49.56 INFO  ==> Welcome to the Bitnami spark container
spark 05:19:49.57 INFO  ==> Subscribe to project updates by watching https://github.com/bitnami/containers
spark 05:19:49.57 INFO  ==> Submit issues and feature requests at https://github.com/bitnami/containers/issues
spark 05:19:49.57 INFO  ==> Upgrade to Tanzu Application Catalog for production environments to access custom-c
onfigured and pre-packaged software components. Gain enhanced features, including Software Bill of Materials (S
BOM), CVE scan result reports, and VEX documents. To learn more, visit https://bitnami.com/enterprise
spark 05:19:49.57 INFO  ==>

find: paths must precede expression: `cp'
```

Implementation

5. Add a test file with a line of words that we will be using later for the word count test

$ echo "The quick brown fox jumps over the lazy dog" > /tmp/test.txt

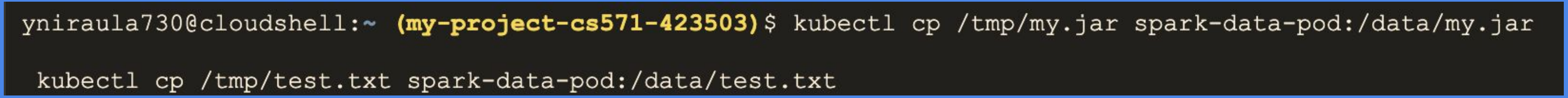


```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ cat /tmp/test.txt
 The quick brown fox jumps over the lazy dog
```

6. Copy the JAR file containing the application, and any other required files, to the PVC using the mount point

$ kubectl cp /tmp/my.jar spark-data-pod:/data/my.jar

$ kubectl cp /tmp/test.txt spark-data-pod:/data/test.txt

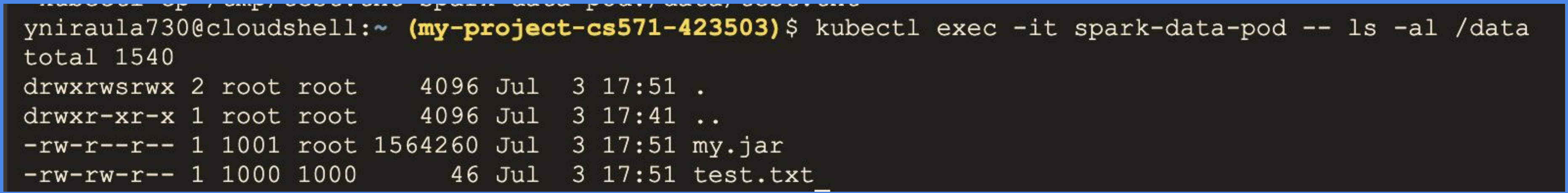


```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl cp /tmp/my.jar spark-data-pod:/data/my.jar

 kubectl cp /tmp/test.txt spark-data-pod:/data/test.txt
```

Implementation

7. Make sure the files are inside the persistent volume

$ kubectl exec -it spark-data-pod -- ls -al /data



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl exec -it spark-data-pod -- ls -al /data
total 1540
drwxrwsrwx 2 root root    4096 Jul  3 17:51 .
drwxr-xr-x 1 root root    4096 Jul  3 17:41 ..
-rw-r--r-- 1 1001 root 1564260 Jul  3 17:51 my.jar
-rw-rw-r-- 1 1000 1000      46 Jul  3 17:51 test.txt
```

Implementation

9. Deploy Apache Spark on Kubernetes using the shared volume spark-chart.yaml

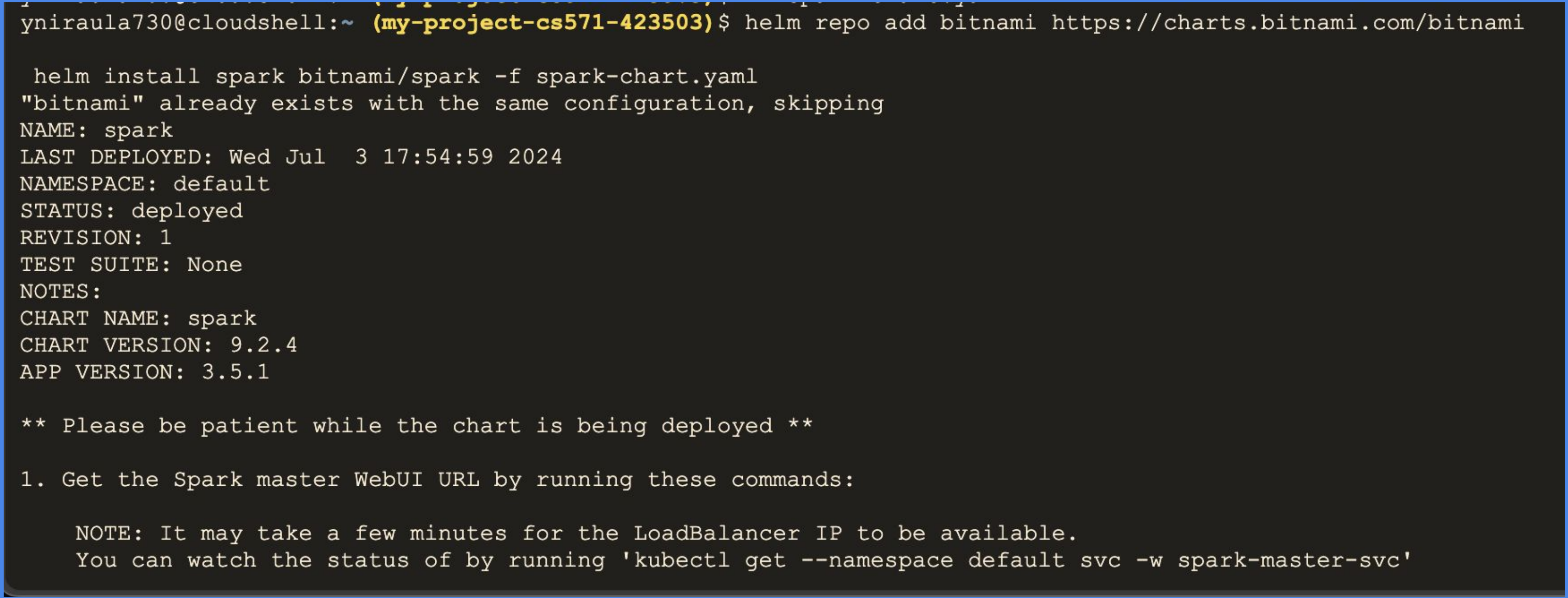
```
$ vi spark-chart.yaml
service:
  type: LoadBalancer

worker:
  replicaCount: 3
extraVolumes:
  - name: spark-data
    persistentVolumeClaim:
      claimName: spark-data-pvc
extraVolumeMounts:
  - name: spark-data
    mountPath: /data
```

Implementation

10. Deploy Apache Spark on the Kubernetes cluster using the Bitnami Apache Spark Helm chart and supply it with the configuration file above.

$ helm repo add bitnami https://charts.bitnami.com/bitnami

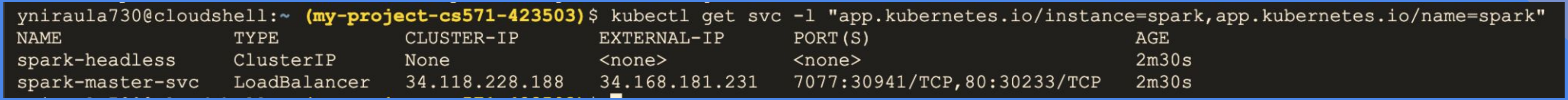
$ helm install spark bitnami/spark -f spark-chart.yaml



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ helm repo add bitnami https://charts.bitnami.com/bitnami

 helm install spark bitnami/spark -f spark-chart.yaml
"bitnami" already exists with the same configuration, skipping
NAME: spark
LAST DEPLOYED: Wed Jul  3 17:54:59 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: spark
CHART VERSION: 9.2.4
APP VERSION: 3.5.1

** Please be patient while the chart is being deployed **

1. Get the Spark master WebUI URL by running these commands:

    NOTE: It may take a few minutes for the LoadBalancer IP to be available.
    You can watch the status of by running 'kubectl get --namespace default svc -w spark-master-svc'
```

Implementation

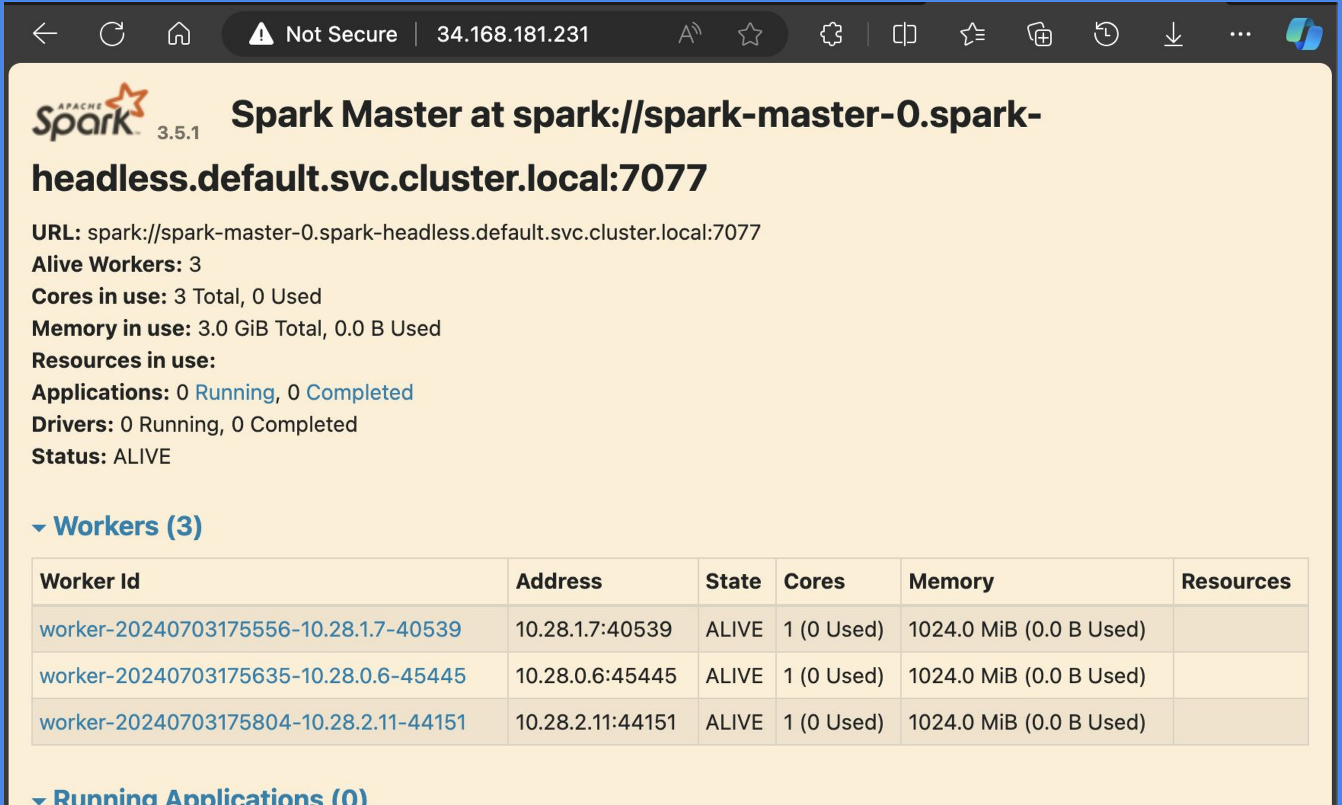
11. Get the external IP of the running pod



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl get svc -l "app.kubernetes.io/instance=spark,app.kubernetes.io/name=spark"
NAME                 TYPE           CLUSTER-IP       EXTERNAL-IP      PORT(S)                        AGE
spark-headless       ClusterIP      None             <none>           <none>                         2m30s
spark-master-svc     LoadBalancer   34.118.228.188   34.168.181.231   7077:30941/TCP,80:30233/TCP    2m30s
```

Implementation

12. Open the external ip on your browser.



3.5.1 **Spark Master at spark://spark-master-0.spark-headless.default.svc.cluster.local:7077**

URL: spark://spark-master-0.spark-headless.default.svc.cluster.local:7077
Alive Workers: 3
Cores in use: 3 Total, 0 Used
Memory in use: 3.0 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (3)

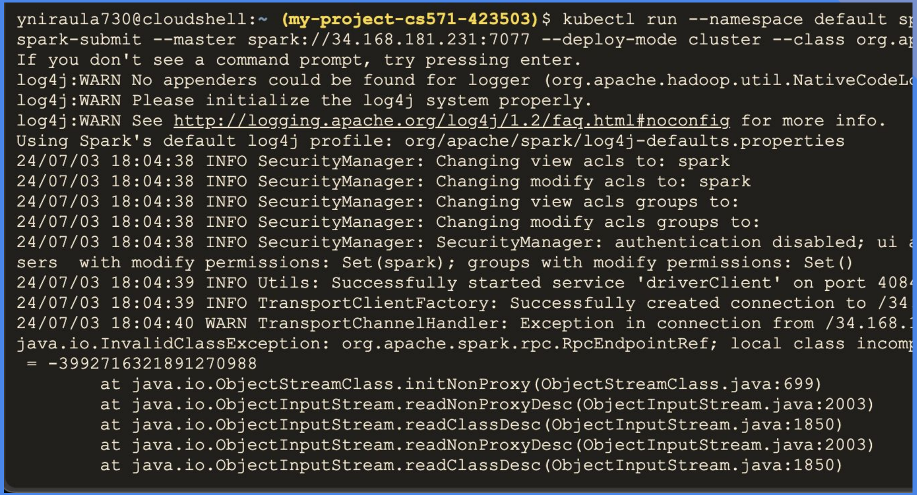
Worker Id	Address	State	Cores	Memory	Resources
worker-20240703175556-10.28.1.7-40539	10.28.1.7:40539	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20240703175635-10.28.0.6-45445	10.28.0.6:45445	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20240703175804-10.28.2.11-44151	10.28.2.11:44151	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	

Running Applications (0)

Testing

1. Submit a word count task

```
$ kubectl run --namespace default
spark-client --rm -it --restart='Never' \
  --image
docker.io/bitnami/spark:3.0.1-debian-10-r1
15 \
  -- spark-submit --master
spark://LOAD-BALANCER-External-ip-ADD
RESS:7077 \
  --deploy-mode cluster \
  --class
org.apache.spark.examples.JavaWordCou
nt \
  /data/my.jar /data/test.txt
```

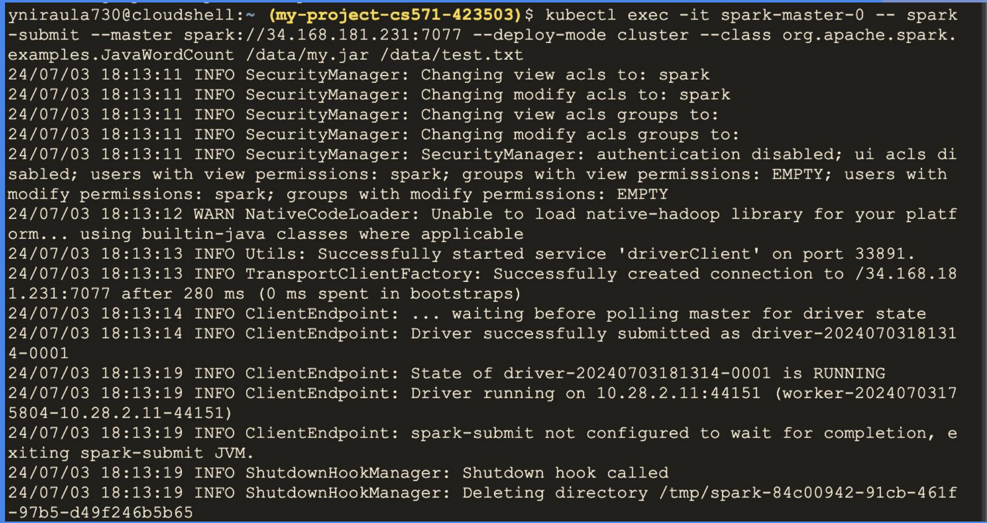


```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl run --namespace default sp
spark-submit --master spark://34.168.181.231:7077 --deploy-mode cluster --class org.ap
If you don't see a command prompt, try pressing enter.
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.NativeCodeLo
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
24/07/03 18:04:38 INFO SecurityManager: Changing view acls to: spark
24/07/03 18:04:38 INFO SecurityManager: Changing modify acls to: spark
24/07/03 18:04:38 INFO SecurityManager: Changing view acls groups to:
24/07/03 18:04:38 INFO SecurityManager: Changing modify acls groups to:
24/07/03 18:04:38 INFO SecurityManager: SecurityManager: authentication disabled; ui a
sers  with modify permissions: Set(spark); groups with modify permissions: Set()
24/07/03 18:04:39 INFO Utils: Successfully started service 'driverClient' on port 408
24/07/03 18:04:39 INFO TransportClientFactory: Successfully created connection to /34
24/07/03 18:04:40 WARN TransportChannelHandler: Exception in connection from /34.168.1
java.io.InvalidClassException: org.apache.spark.rpc.RpcEndpointRef; local class incomp
 = -3992716321891270988
        at java.io.ObjectStreamClass.initNonProxy(ObjectStreamClass.java:699)
        at java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:2003)
        at java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1850)
        at java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:2003)
        at java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1850)
```

Testing

2. Encountered an error with that command. Let’s switch to kubectl exec from the Spark master node to run the job

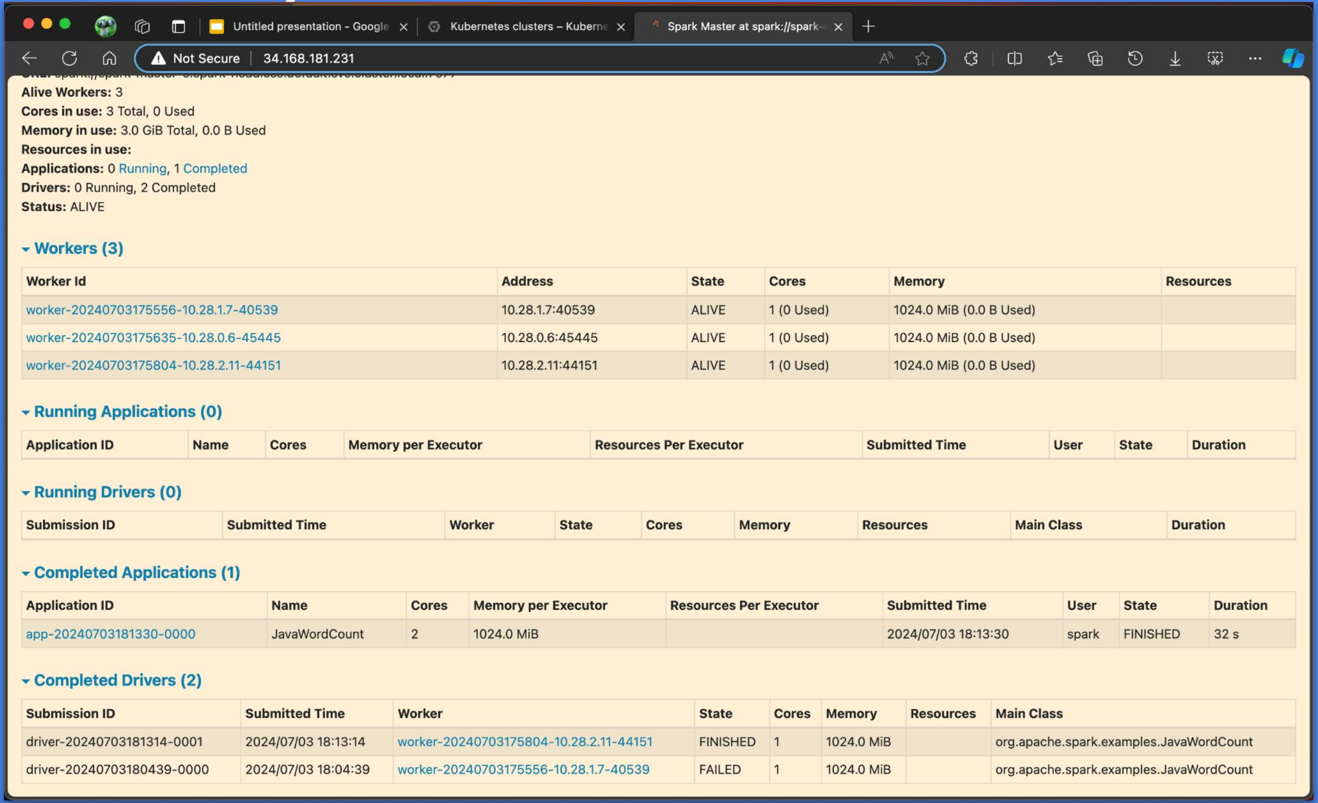
$ kubectl exec -it spark-master-0 --
spark-submit
master
spark://34.168.181.231:7077
-deploy-mode cluster --class
org.apache.spark.examples.JavaWo
rdCount /data/my.jar /data/test.txt



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl exec -it spark-master-0 -- spark
-submit --master spark://34.168.181.231:7077 --deploy-mode cluster --class org.apache.spark.
examples.JavaWordCount /data/my.jar /data/test.txt
24/07/03 18:13:11 INFO SecurityManager: Changing view acls to: spark
24/07/03 18:13:11 INFO SecurityManager: Changing modify acls to: spark
24/07/03 18:13:11 INFO SecurityManager: Changing view acls groups to:
24/07/03 18:13:11 INFO SecurityManager: Changing modify acls groups to:
24/07/03 18:13:11 INFO SecurityManager: SecurityManager: authentication disabled; ui acls di
sabled; users with view permissions: spark; groups with view permissions: EMPTY; users with
modify permissions: spark; groups with modify permissions: EMPTY
24/07/03 18:13:12 WARN NativeCodeLoader: Unable to load native-hadoop library for your platf
orm... using builtin-java classes where applicable
24/07/03 18:13:13 INFO Utils: Successfully started service 'driverClient' on port 33891.
24/07/03 18:13:13 INFO TransportClientFactory: Successfully created connection to /34.168.18
1.231:7077 after 280 ms (0 ms spent in bootstraps)
24/07/03 18:13:14 INFO ClientEndpoint: ... waiting before polling master for driver state
24/07/03 18:13:14 INFO ClientEndpoint: Driver successfully submitted as driver-2024070318131
4-0001
24/07/03 18:13:19 INFO ClientEndpoint: State of driver-20240703181314-0001 is RUNNING
24/07/03 18:13:19 INFO ClientEndpoint: Driver running on 10.28.2.11:44151 (worker-2024070317
5804-10.28.2.11-44151)
24/07/03 18:13:19 INFO ClientEndpoint: spark-submit not configured to wait for completion, e
xiting spark-submit JVM.
24/07/03 18:13:19 INFO ShutdownHookManager: Shutdown hook called
24/07/03 18:13:19 INFO ShutdownHookManager: Deleting directory /tmp/spark-84c00942-91cb-461f
-97b5-d49f246b5b65
```

Testing

3. On the browser, you should be able to see the task finished.


Untitled presentation - Google
Kubernetes clusters – Kuberne
Spark Master at spark://spark-
Not Secure
34.168.181.231

Alive Workers: 3
Cores in use: 3 Total, 0 Used
Memory in use: 3.0 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 1 Completed
Drivers: 0 Running, 2 Completed
Status: ALIVE

Workers (3)

Worker Id	Address	State	Cores	Memory	Resources
worker-20240703175556-10.28.1.7-40539	10.28.1.7:40539	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20240703175635-10.28.0.6-45445	10.28.0.6:45445	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20240703175804-10.28.2.11-44151	10.28.2.11:44151	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Running Drivers (0)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class	Duration

Completed Applications (1)

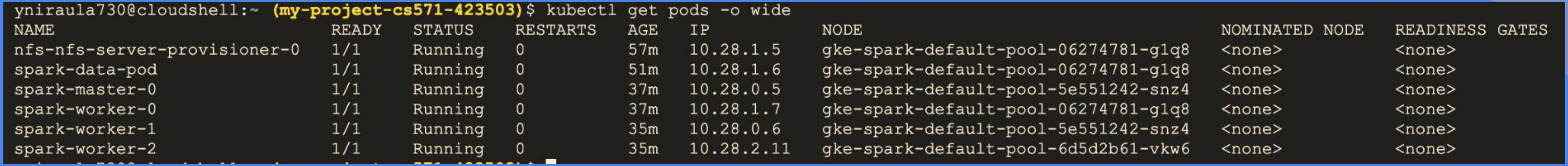
Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20240703181330-0000	JavaWordCount	2	1024.0 MiB		2024/07/03 18:13:30	spark	FINISHED	32 s

Completed Drivers (2)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class
driver-20240703181314-0001	2024/07/03 18:13:14	worker-20240703175804-10.28.2.11-44151	FINISHED	1	1024.0 MiB		org.apache.spark.examples.JavaWordCount
driver-20240703180439-0000	2024/07/03 18:04:39	worker-20240703175556-10.28.1.7-40539	FAILED	1	1024.0 MiB		org.apache.spark.examples.JavaWordCount

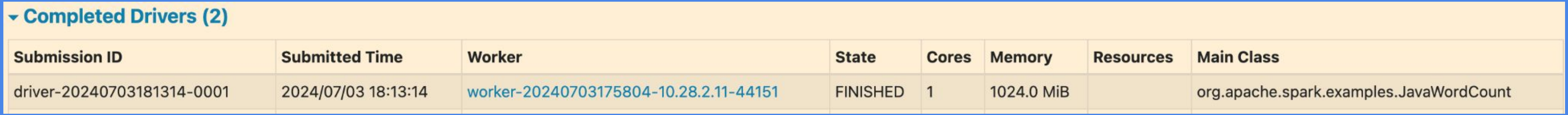
Testing

4. Get the name of the worker node
   $ kubectl get pods -o wide



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl get pods -o wide
NAME                            READY   STATUS    RESTARTS   AGE   IP           NODE                                      NOMINATED NODE   READINESS GATES
nfs-nfs-server-provisioner-0    1/1     Running   0          57m   10.28.1.5    gke-spark-default-pool-06274781-g1q8      <none>           <none>
spark-data-pod                  1/1     Running   0          51m   10.28.1.6    gke-spark-default-pool-06274781-g1q8      <none>           <none>
spark-master-0                  1/1     Running   0          37m   10.28.0.5    gke-spark-default-pool-5e551242-snz4      <none>           <none>
spark-worker-0                  1/1     Running   0          37m   10.28.1.7    gke-spark-default-pool-06274781-g1q8      <none>           <none>
spark-worker-1                  1/1     Running   0          35m   10.28.0.6    gke-spark-default-pool-5e551242-snz4      <none>           <none>
spark-worker-2                  1/1     Running   0          35m   10.28.2.11   gke-spark-default-pool-6d5d2b61-vkw6      <none>           <none>
```

We can verify that the IP address of the worker node that processed the word count task is 10.28.2.11, which we can see in the browser as well and the name is spark-worker-2.

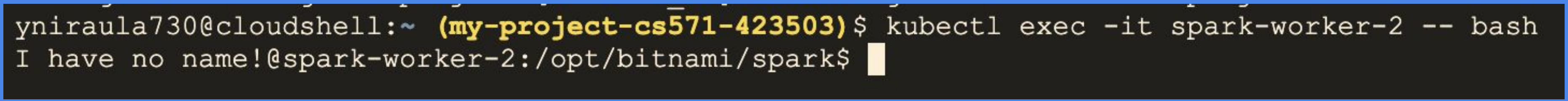


▾ Completed Drivers (2)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class
driver-20240703181314-0001	2024/07/03 18:13:14	worker-20240703175804-10.28.2.11-44151	FINISHED	1	1024.0 MiB		org.apache.spark.examples.JavaWordCount

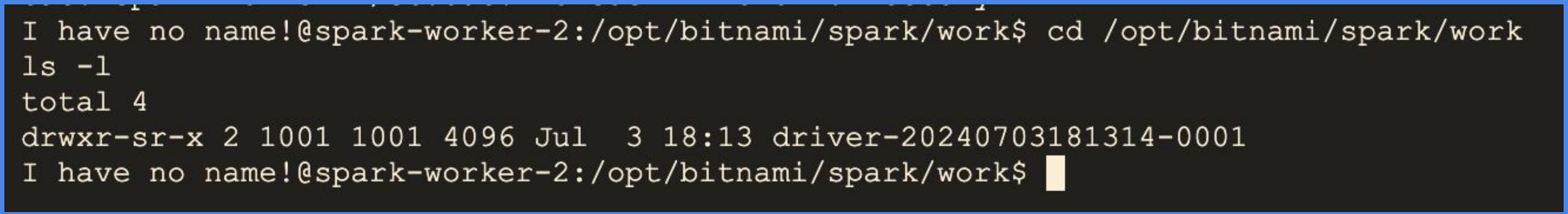
Testing

5. Execute the pod to see the result of the task
$ kubectl exec -it <worker node name> -- bash



```
yniraula730@cloudshell:~ (my-project-cs571-423503)$ kubectl exec -it spark-worker-2 -- bash
I have no name!@spark-worker-2:/opt/bitnami/spark$
```

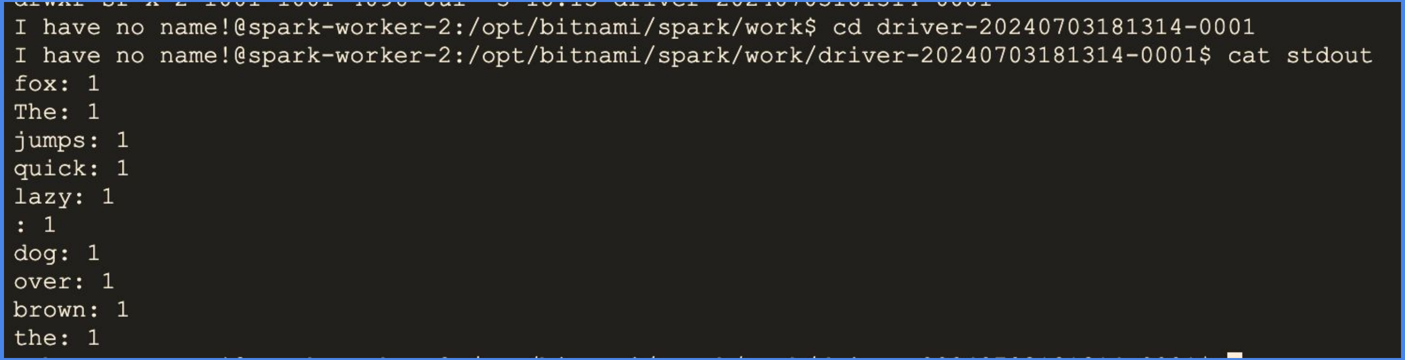
cd /opt/bitnami/spark/work
ls -l



```
I have no name!@spark-worker-2:/opt/bitnami/spark/work$ cd /opt/bitnami/spark/work
ls -l
total 4
drwxr-sr-x 2 1001 1001 4096 Jul  3 18:13 driver-20240703181314-0001
I have no name!@spark-worker-2:/opt/bitnami/spark/work$
```

Testing

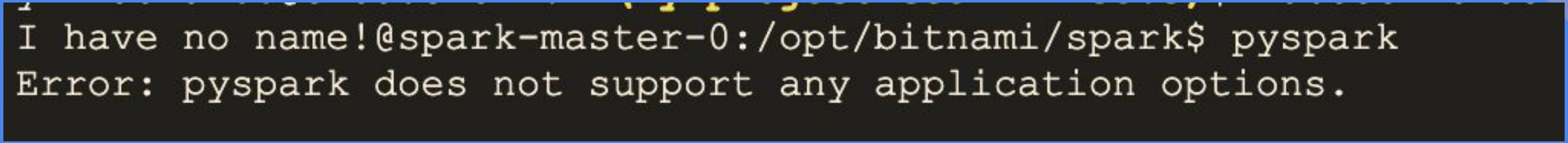
cd driver-20240703181314-0001
cd stdout



```
I have no name!@spark-worker-2:/opt/bitnami/spark/work$ cd driver-20240703181314-0001
I have no name!@spark-worker-2:/opt/bitnami/spark/work/driver-20240703181314-0001$ cat stdout
fox: 1
The: 1
jumps: 1
quick: 1
lazy: 1
: 1
dog: 1
over: 1
brown: 1
the: 1
```

Testing

6. Running python PageRank onPySpark on the pods. Execute the Spark master node
$ kubectl exec -it spark-master-0 -- bash

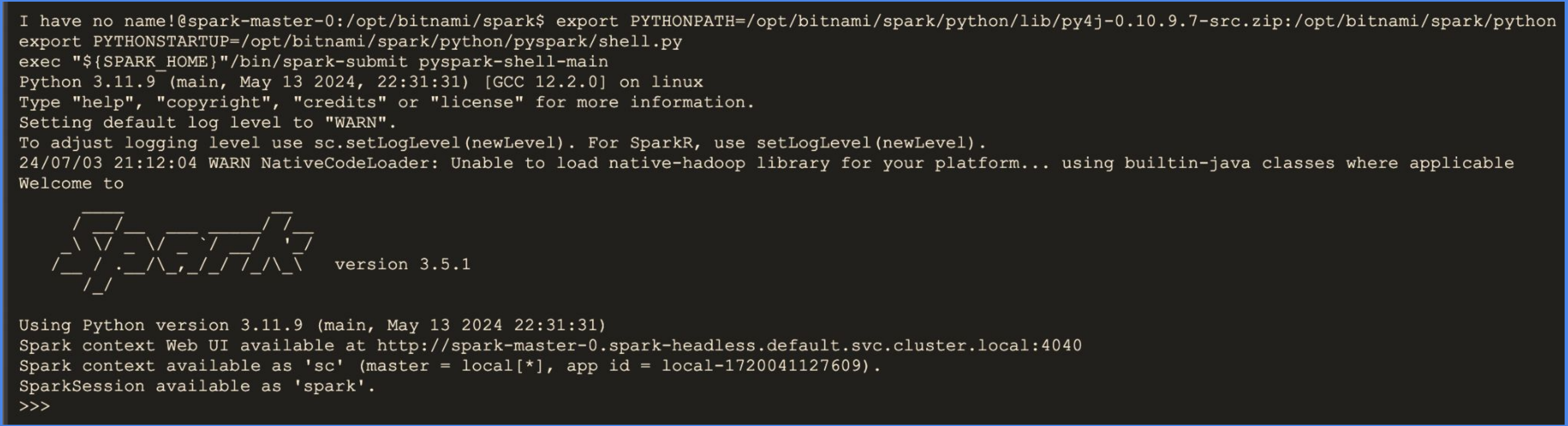


```
I have no name!@spark-master-0:/opt/bitnami/spark$ pyspark
Error: pyspark does not support any application options.
```

It seems to be the --name argument is causing the issue in script.

Testing

Solution:
export PYTHONPATH=/opt/bitnami/spark/python/lib/py4j-0.10.9.7-src.zip:/opt/bitnami/spark/python
export PYTHONSTARTUP=/opt/bitnami/spark/python/pyspark/shell.py
exec "${SPARK_HOME}"/bin/spark-submit pyspark-shell-main



```
I have no name!@spark-master-0:/opt/bitnami/spark$ export PYTHONPATH=/opt/bitnami/spark/python/lib/py4j-0.10.9.7-src.zip:/opt/bitnami/spark/python
export PYTHONSTARTUP=/opt/bitnami/spark/python/pyspark/shell.py
exec "${SPARK_HOME}"/bin/spark-submit pyspark-shell-main
Python 3.11.9 (main, May 13 2024, 22:31:31) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/07/03 21:12:04 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.1
      /_/

Using Python version 3.11.9 (main, May 13 2024 22:31:31)
Spark context Web UI available at http://spark-master-0.spark-headless.default.svc.cluster.local:4040
Spark context available as 'sc' (master = local[*], app id = local-1720041127609).
SparkSession available as 'spark'.
>>>
```

Conclusion

- PySpark is a powerful tool used for scalable data analysis, building machine learning pipelines, and creating ETLs for data platforms.

- Efficiency: PySpark excels in performance due to its lazy execution model.
  - Lazy execution means that operations are not executed until they are actually needed, optimizing the overall workflow and resource utilization.

References

- https://towardsdatascience.com/how-to-guide-set-up-manage-monitor-spark-on-kubernetes-with-code-examples-c5364ad3aba2
- https://www.datamechanics.co/apache-spark-on-kubernetes
- https://spark.apache.org/docs/latest/running-on-kubernetes.html
- https://npu85.npu.edu/~henry/npu/classes/master_apache_spark/kubernetes/slide/exercise_kubernetes.html
- https://hc.labnet.sfbu.edu/~henry/npu/classes/cloud_computing/week1/syllabus.html

Appendix

- Github link:

  https://github.com/yubrajniraula/Cloud-Computing/tree/main/Word%20Count%20%2B%20PageRank

- Google Slides link:

  https://docs.google.com/presentation/d/15PUgzjryIPaYZMa5aQ2JYceWnCtKYDoHb0sX7C9Aa54/edit#slide=id.g275165550b3_0_928