1. Create a cluster with:

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

```
nhaile96456@cloudshell:~ (cs570-big-data-analytics)$ gcloud container clusters list --regi
on=us-west1
NAME: spark
LOCATION: us-west1
MASTER_VERSION: 1.29.4-gke.1043002
MASTER_IP: 35.199.181.0
MACHINE_TYPE: e2-highmem-2
NODE_VERSION: 1.29.4-gke.1043002
NUM_NODES: 3
STATUS: RUNNING
nhaile96456@cloudshell:~ (cs570-big-data-analytics)$
```

Create image and deploy spark to Kubernetes

2. Install the NFS Server Provisioner.

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

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ helm repo add stable https:
//charts.helm.sh/stable
"stable" has been added to your repositories
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "stable" chart repository
Update Complete. *Happy Helming!*                          helm install nfs stable/nfs
-server-provisioner --set persistence.enabled=true,persistence.size=5Gitall nfs stable/nfs
WARNING: This chart is deprecatedence.enabled=true,persistence.size=5Gi
NAME: nfs
LAST DEPLOYED: Sun Jun 30 19:01:28 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
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

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

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ vi spark-pvc.yaml
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ ls
spark-pvc.yaml
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ cat 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
```

4. Apply the above yaml descriptor.

- *kubectl apply -f spark-pvc.yaml*

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ kubectl apply -f spark-pvc.yaml
persistentvolumeclaim/spark-data-pvc created
pod/spark-data-pod created
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

5. Create and prepare your application JAR file.

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

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ docker run -v /tmp:/tmp -it bitnami/s
park /bin/bash -c "find /opt/bitnami/spark/examples/jars/ -name 'spark-examples*' -exec cp {} /tmp/m
y.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 19:17:52.44 INFO  ==>
spark 19:17:52.45 INFO  ==> Welcome to the Bitnami spark container
spark 19:17:52.45 INFO  ==> Subscribe to project updates by watching https://github.com/bitnami/cont
ainers
spark 19:17:52.45 INFO  ==> Submit issues and feature requests at https://github.com/bitnami/contain
ers/issues
spark 19:17:52.45 INFO  ==> Upgrade to Tanzu Application Catalog for production environments to acce
ss custom-configured and pre-packaged software components. Gain enhanced features, including Softwar
e Bill of Materials (SBOM), CVE scan result reports, and VEX documents. To learn more, visit https:/
/bitnami.com/enterprise
spark 19:17:52.46 INFO  ==>

nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

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

- *echo "how much wood could a woodpecker chuck if a woodpecker could chuck wood" > /tmp/test.txt*

7. 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*

8. Make sure the files are inside the persistent volume.

- *kubectl exec -it spark-data-pod -- ls -la /data*

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ kubectl exec -it spark-data-pod -- ls
 -la /data
total 1540
drwxrwsrwx 2 root root    4096 Jun 30 19:21 .
drwxr-xr-x 1 root root    4096 Jun 30 19:13 ..
-rw-r--r-- 1 1001 root 1564260 Jun 30 19:21 my.jar
-rw-rw-r-- 1 1000 1000      72 Jun 30 19:21 test.txt
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

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

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ vi spark-chart.yaml
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ cat spark-chart.yaml
service:
  type: LoadBalancer

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

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*

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ helm repo add bitnami https://charts.
bitnami.com/bitnami
helm repo update
"bitnami" has been added to your repositories
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "stable" chart repository
...Successfully got an update from the "bitnami" chart repository
Update Complete. *Happy Helming!*
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ helm install spark bitnami/spark -f s
park-chart.yaml
NAME: spark
LAST DEPLOYED: Sun Jun 30 19:29:28 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'

  export SERVICE_IP=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loa
dBalancer.ingress[0]['ip', 'hostname'] }")
  echo http://$SERVICE_IP:80

2. Submit an application to the cluster:

  To submit an application to the cluster the spark-submit script must be used. That script can be
```

Make sure the deployment was successful.

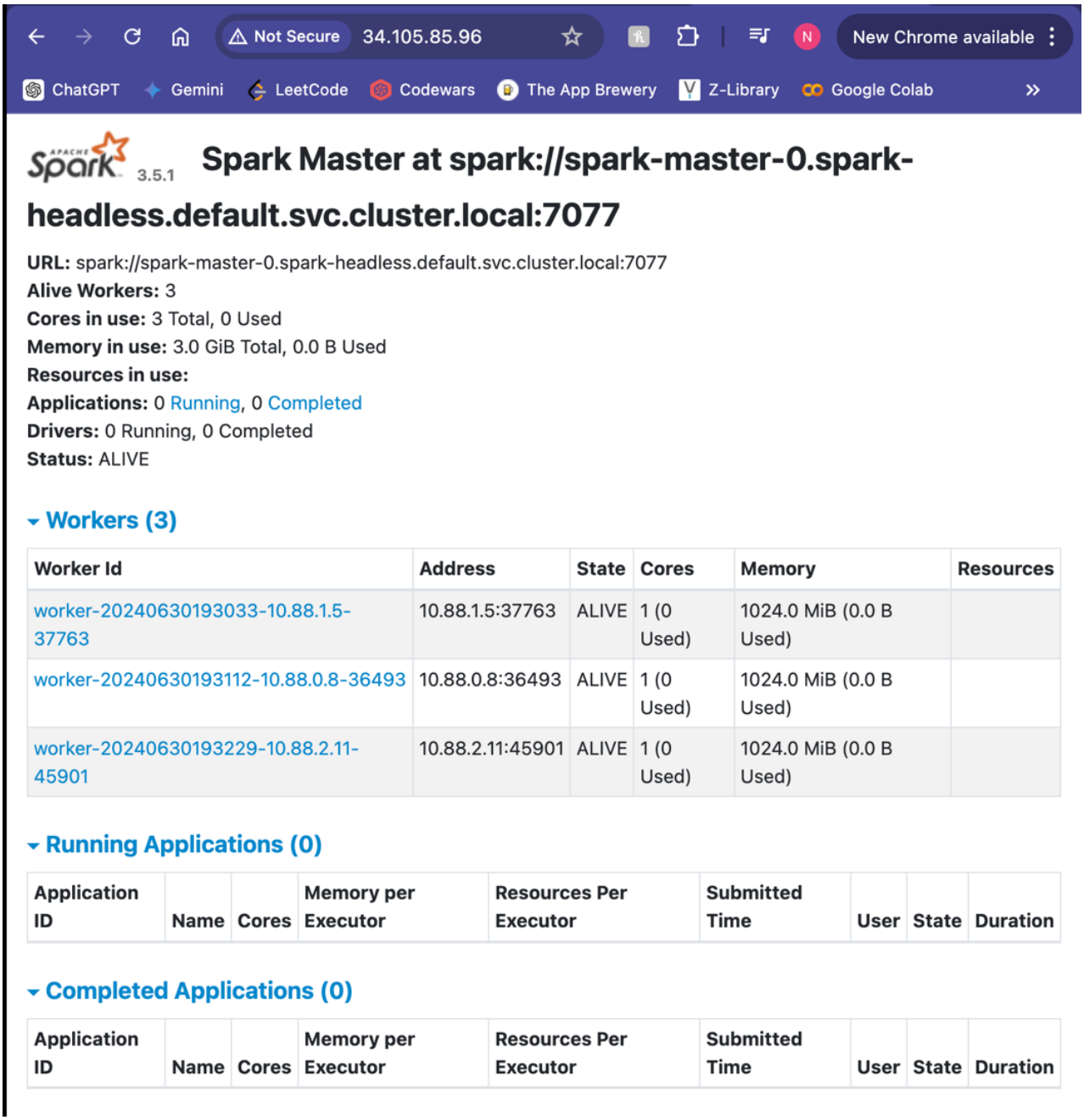
```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ helm list
NAME    NAMESPACE       REVISION        UPDATED                                   STATUS          CHAR
T                               APP VERSION
nfs     default         1               2024-06-30 19:01:28.028978257 +0000 UTC deployed        nfs-
server-provisioner-1.1.3        2.3.0
spark   default         1               2024-06-30 19:29:28.506007397 +0000 UTC deployed        spar
k-9.2.4                         3.5.1
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ kubectl get services
NAME                        TYPE           CLUSTER-IP       EXTERNAL-IP    PORT(S)
                                                                                    AGE
kubernetes                  ClusterIP      34.118.224.1     <none>         443/TCP
                                                                                    58m
nfs-nfs-server-provisioner  ClusterIP      34.118.232.94    <none>         2049/TCP,2049/UDP,32803/
TCP,32803/UDP,20048/TCP,20048/UDP,875/TCP,875/UDP,111/TCP,111/UDP,662/TCP,662/UDP   29m
spark-headless              ClusterIP      None             <none>         <none>
                                                                                    107s
spark-master-svc            LoadBalancer   34.118.239.116   34.105.85.96   7077:32100/TCP,80:30661/
TCP                                                                                 107s
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
nfs-nfs-server-provisioner-0  1/1     Running   0          29m
spark-data-pod                1/1     Running   0          17m
spark-master-0                1/1     Running   0          119s
spark-worker-0                1/1     Running   0          119s
spark-worker-1                0/1     Running   0          27s
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

11. Get the external IP of the running pod.

- *kubectl get svc -l "app.kubernetes.io/instance=spark,app.kubernetes.io/name=spark"*

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ 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>                       7m43
s
spark-master-svc  LoadBalancer   34.118.239.116   34.105.85.96   7077:32100/TCP,80:30661/TCP  7m43
s
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

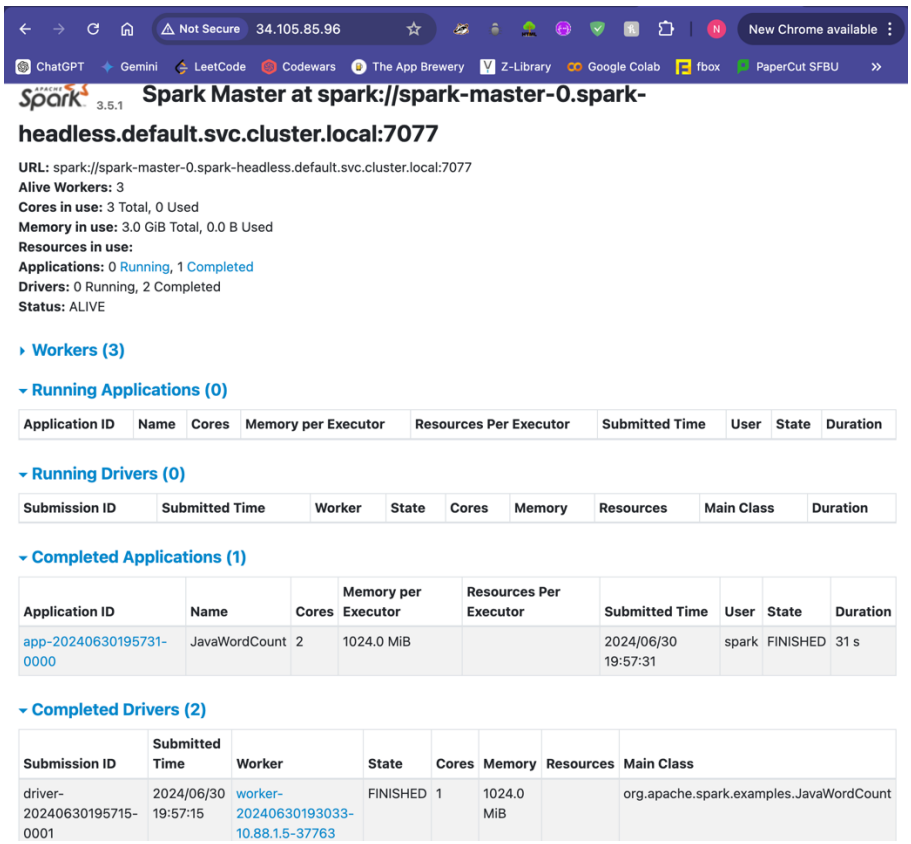
12. Browse to the external IP on a browser.



Word count on Spark

13. Submit a word count task:

- *kubectl exec -it spark-master-0 -- spark-submit --master spark://34.105.85.96:7077 --deploy-mode cluster --class org.apache.spark.examples.JavaWordCount /data/my.jar /data/test.txt*



View the output of the completed job

14. On the browser, you should see the worker node IP address of the finished task.

Completed Drivers (2)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class
driver-20240630195715-0001	2024/06/30 19:57:15	worker-20240630193033-10.88.1.5-37763	FINISHED	1	1024.0 MiB		org.apache.spark.examples.JavaWordCount

For example, my worker node IP address is 10.88.1.5.

15. Get the name of the worker node

- *kubectl get pods -o wide | grep 10.88.1.5*

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ kubectl get pods -o wide | g
rep 10.88.1.5
spark-worker-0                 1/1     Running   0          34m   10.88.1.5    gke-spark-de
fault-pool-740bc581-5pts   <none>           <none>
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$
```

16. Execute this pod and see the result of the finished tasks.

- *kubectl exec -it spark-worker-0 – bash*
- *cd /opt/bitnami/spark/work*
- *cat driver-20240630195715-0001/stdout*

```
nhaile96456@cloudshell:~/wordcount (cs570-big-data-analytics)$ kubectl exec -it spark-worke
r-0 -- bash
I have no name!@spark-worker-0:/opt/bitnami/spark$ cd /opt/bitnami/spark/work
I have no name!@spark-worker-0:/opt/bitnami/spark/work$ ls
driver-20240630195715-0001
I have no name!@spark-worker-0:/opt/bitnami/spark/work$ cat driver-20240630195715-0001/stdo
ut
if: 1
a: 2
how: 1
could: 2
wood: 2
woodpecker: 2
much: 1
chuck: 2
I have no name!@spark-worker-0:/opt/bitnami/spark/work$
```

Running python PageRank on PySpark on the pods

17. Execute the spark master pods.

- *kubectl exec -it spark-master-0 – bash*

```
nhaile96456@cloudshell:~ (cs570-big-data-analytics)$ kubectl exec -it spark-master-0 -- bas
h
I have no name!@spark-master-0:/opt/bitnami/spark$
```

Error:

```
nhaile96456@cloudshell:~ (cs570-big-data-analytics)$ kubectl exec -it spark-master-0 -- bas
h
I have no name!@spark-master-0:/opt/bitnami/spark$ pyspark
Error: pyspark does not support any application options.

Usage: ./bin/pyspark [options]

Options:
  --master MASTER_URL         spark://host:port, mesos://host:port, yarn,
                              k8s://https://host:port, or local (Default: local[*]).
  --deploy-mode DEPLOY_MODE   Whether to launch the driver program locally ("client") or
                              on one of the worker machines inside the cluster ("cluster")
                              (Default: client).
  --class CLASS_NAME          Your application's main class (for Java / Scala apps).
  --name NAME                 A name of your application.
  --jars JARS                 Comma-separated list of jars to include on the driver
                              and executor classpaths.
  --packages                  Comma-separated list of maven coordinates of jars to include
                              on the driver and executor classpaths. Will search the local
                              maven repo, then maven central and any additional remote
                              repositories given by --repositories. The format for the
                              coordinates should be groupId:artifactId:version.
```

Solution: Run the following script from a github repo and the issue with environment variables will be solved.

Link to repo: https://github.com/bitnami/containers/issues/38139

export PYTHONPATH=/opt/bitnami/spark/python/lib/py4j-0.10.9.7-src.zip:/opt/bitnami/spark/python/:/opt/bitnami/spark/python/:

export PYTHONSTARTUP=/opt/bitnami/spark/python/pyspark/shell.py

exec "${SPARK_HOME}"/bin/spark-submit pyspark-shell-main

exit()

```
I have no name!@spark-master-0:/opt/bitnami/spark$ export PYTHONPATH=/opt/bitnami/spark/pyt
hon/lib/py4j-0.10.9.7-src.zip:/opt/bitnami/spark/python/:/opt/bitnami/spark/python/:
export PYTHONSTARTUP=/opt/bitnami/spark/python/pyspark/shell.py
exec "${SPARK_HOME}"/bin/spark-submit pyspark-shell-main
exit()
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)
.
24/06/30 22:35:55 WARN NativeCodeLoader: Unable to load native-hadoop library for your plat
form... 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-1719786958247).
SparkSession available as 'spark'.
>>>
```

18. Exit pyspark.

- *exit()*

19. Go to the directory where pagerank.py located.

- *cd /opt/bitnami/spark/examples/src/main/python*

20. Run the pagerank using pyspark

- *spark-submit pagerank.py /opt 2*

Notice, /opt is an example directory, you can enter any directory you like, and 2 is the number of iterations you want the pagerank to run.

```
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/comprehendmedical/2018-10-
30
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/numpy/core/include/numpy/random
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/mediastore
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/pandas/tests/indexes/multi
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/numpy/f2py/tests/src/module_data
        file:/opt/bitnami/python/lib/python3.11/test/test_warnings/data
        file:/opt/bitnami/python/lib/python3.11/site-packages/virtualenv/activation
        file:/opt/bitnami/python/lib/python3.11/venv/scripts/posix
        file:/opt/bitnami/spark/examples/src/main/python/ml
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/glacier/2012-06-01
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/migrationhubstrategy/2020-
02-19
        file:/opt/bitnami/spark/python/docs/source/reference/pyspark.sql
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/support/2013-04-15
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/pandas/tests/window/moments
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/devicefarm
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/efs
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/dynamodbstreams/2012-08-10
        file:/opt/bitnami/java/jmods
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/medialive/2017-10-14
        file:/opt/bitnami/common/licenses
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/arc-zonal-shift/2022-10-30
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/apprunner/2020-05-15
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/launch-wizard/2018-05-10
        file:/opt/bitnami/spark/python/pyspark/sql/connect/proto
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/connect/2017-08-08
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/textract
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/application-autoscaling
        file:/opt/bitnami/java/legal/jdk.localedata
        file:/opt/bitnami/python/lib/python3.11/site-packages
        file:/opt/bitnami/spark/python/pyspark/sql/connect/protobuf
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/ssm
        file:/opt/bitnami/python/lib/python3.11/test/test_importlib/namespace_pkgs/project2/parent/child
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/pandas/tests/indexes/datetimes
        file:/opt/bitnami/spark/python/pyspark/resource/tests
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/awscli/examples/configure
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/rds/2014-09-01
        file:/opt/bitnami/python/lib/python3.11/site-packages/virtualenv/activation/nushell
        file:/opt/bitnami/python/lib/python3.11/email/mime
        file:/opt/bitnami/spark/venv/lib/python3.11/site-packages/botocore/data/kinesis/2013-12-02
```

```
24/06/30 23:13:28 INFO SparkContext: Invoking stop() from shutdown hook
24/06/30 23:13:28 INFO SparkContext: SparkContext is stopping with exitCode 0.
24/06/30 23:13:28 INFO SparkUI: Stopped Spark web UI at http://spark-master-0.spark-headless.default.svc.c
luster.local:4040
24/06/30 23:13:28 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/06/30 23:13:28 INFO MemoryStore: MemoryStore cleared
24/06/30 23:13:28 INFO BlockManager: BlockManager stopped
24/06/30 23:13:28 INFO BlockManagerMaster: BlockManagerMaster stopped
24/06/30 23:13:28 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator st
opped!
24/06/30 23:13:28 INFO SparkContext: Successfully stopped SparkContext
24/06/30 23:13:28 INFO ShutdownHookManager: Shutdown hook called
24/06/30 23:13:28 INFO ShutdownHookManager: Deleting directory /tmp/spark-2adeec7e-c417-4b1c-b70f-06a05176
6af5
24/06/30 23:13:28 INFO ShutdownHookManager: Deleting directory /tmp/spark-9fb13338-9ba8-4f3b-8e5b-a01ea860
61cc
24/06/30 23:13:28 INFO ShutdownHookManager: Deleting directory /tmp/spark-9fb13338-9ba8-4f3b-8e5b-a01ea860
61cc/pyspark-782e3f31-7ba2-4336-9594-5779c56e6dd5
I have no name!@spark-master-0:/opt/bitnami/spark/examples/src/main/python$
```