PySpark on Kubernetes: Word Count+ PageRank

PROJECT

— By Lidia Bereketeab

TABLE OF CONTENT

INTRODUCTION

- Kubernetes: A flexible, open-source platform for managing containerized workloads and services, supporting automated scaling and resilient deployment.

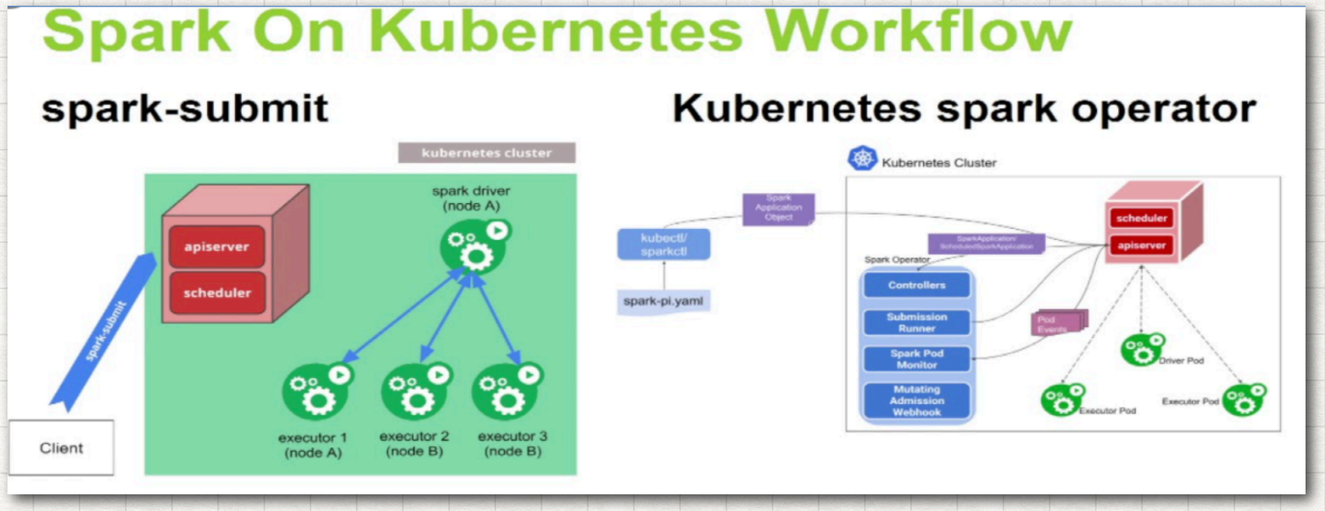
- Apache Spark on Kubernetes: Integrates the strengths of Apache Spark, a powerful data processing framework, with Kubernetes' orchestration capabilities.

- PySpark: Python interface for Apache Spark, enabling development and interactive data analysis using Python APIs in distributed environments.

INTRODUCTION

Spark on Kubernetes:

- Submitting a Spark application communicates directly with Kubernetes' API server.
- Kubernetes schedules the Spark driver pod and container.
- The Spark driver interacts with the Kubernetes cluster to request and launch Spark executors.
- Each Spark executor is scheduled in its own pod for efficient execution



INTRODUCTION

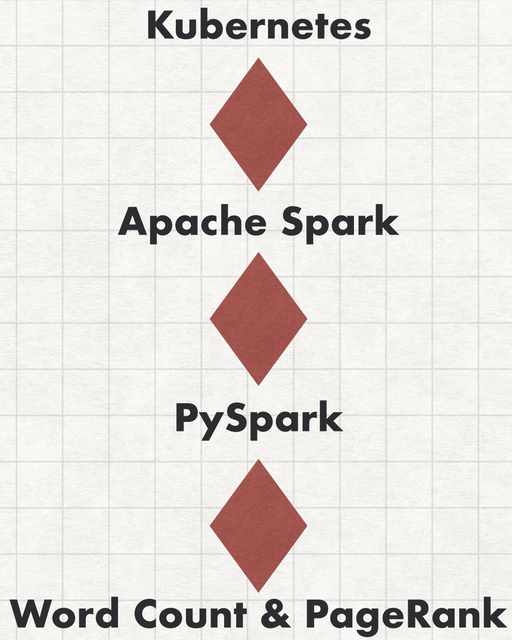
Understanding these technologies is crucial for:

- **Optimizing Infrastructure:** Kubernetes ensures efficient resource utilization and scalability for containerized applications.
- **Enhancing Data Processing**: Apache Spark on Kubernetes enables efficient management of big data processing tasks, improving performance and reliability.
- **Boosting Development Efficiency**: PySpark simplifies data analysis tasks with Python's rich ecosystem, accelerating development cycles and enabling rapid prototyping in data-driven applications.

DESIGN

PROJECT DESCRIPTION: WORK COUNT + PAGERANK

- Step 1: Implementing Word Count and PageRank using PySpark on Apache Spark deployed on Kubernetes.
- Step 2: Enhance your portfolio by linking your Google Slides presentation to your GitHub repository.
- Step 3: Include the URLs of your Google Slides and GitHub in your homework submission.



DESIGN

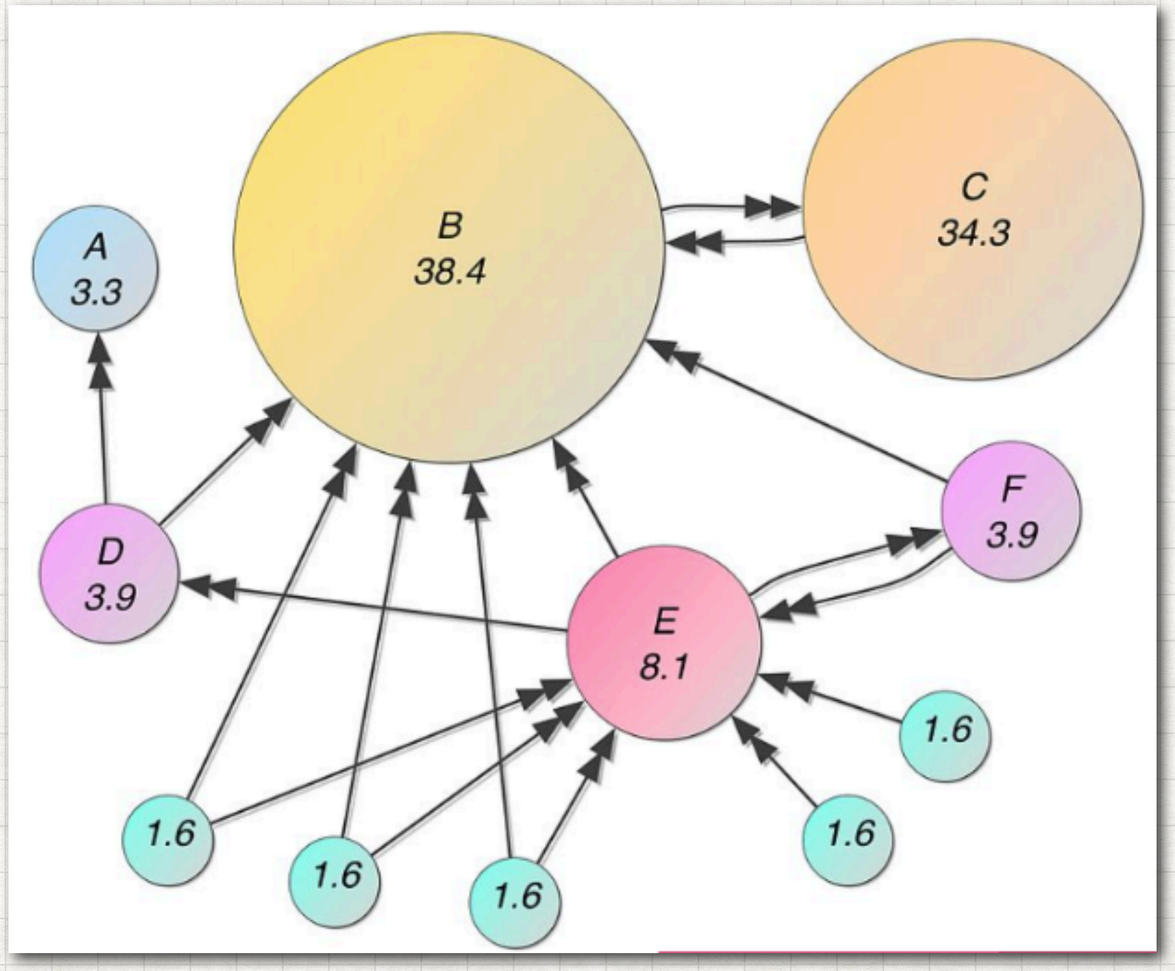
PROJECT DESCRIPTION: WORK COUNT

- **WordCount:** Counts word frequencies in text documents using Spark.
- **Scalability:** Frameworks capable of parallel processing at scale handle large computational tasks effectively.
- **MapReduce**: Distributed computing model in Java with key tasks: Map (transforms data into key/value pairs) and Reduce (processes and summarizes data).

DESIGN

Project Description: PageRank

- **PageRank**: Algorithm used by Google to rank search results based on importance.
- Origins: Developed by Sergey Brin and Larry Page at Stanford University in 1996.
- **Impact:** Higher PageRank boosts a webpage's search engine ranking, attracting more visitors and potential customers.
- **Calculation:** Uses a formula involving link probabilities and damping factor (d) for iterative score calculation.
  PR(A) = (1 – d) + d (PR(t1) / C(t1) + ... + PR(tn)/C(tn)
- **Iteration:** Iterative process repeated multiple times to converge on final PageRank scores for each webpage.



IMPLEMENTATION

Wordcount running on spark, deploying to Kubernetes on GKE

1. Create a cluster on GKE with

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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ gcloud container clusters create spark --num-nodes=1 --machine-type=e2-highmem-2 --region=us-west1
Default change: VPC-native is the default mode during cluster creation for versions greater than 1.21.0-gke.1500. To create advanced routes based clusters, please pass the `--no-ena
ble-ip-alias` flag
Note: The Kubelet readonly port (10255) is now deprecated. Please update your workloads to use the recommended alternatives. See https://cloud.google.com/kubernetes-engine/docs/how-
to/disable-kubelet-readonly-port for ways to check usage and for migration instructions.
Note: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
Creating cluster spark in us-west1... Cluster is being health-checked (master is healthy)...done.
Created [https://container.googleapis.com/v1/projects/cs-570-big-data-424802/zones/us-west1/clusters/spark].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-west1/spark?project=cs-570-big-data-424802
kubeconfig entry generated for spark.
NAME: spark
LOCATION: us-west1
MASTER_VERSION: 1.29.4-gke.1043004
MASTER_IP: 35.203.172.196
MACHINE_TYPE: e2-highmem-2
NODE_VERSION: 1.29.4-gke.1043004
NUM_NODES: 3
STATUS: RUNNING
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ helm repo add stable https://charts.helm.sh/stable
"stable" already exists with the same configuration, skipping
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ helm repo update
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!⎈
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

3. Install the NFS Server Provisioner

helm install nfs stable/nfs-server-provisioner \
set persistence.enabled=true,persistence.size=5Gi

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ helm install nfs stable/nfs-server-provisioner --set persistence.enabled=true,persistence.size=5Gi
WARNING: This chart is deprecated
NAME: nfs
LAST DEPLOYED: Fri Jul  5 05:57:22 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
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

4. Create a persistent disk volume and a pod to use NFS spark-pvc.yaml:

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ vim spark-pvc.yaml
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ cat spark-pvc.yaml
---
kind: PersistentVolumeClaim
apiVersion: v1
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

lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

5. Apply the above yaml descriptor

    - kubectl apply -f spark-pvc.yaml

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl apply -f spark-pvc.yaml
persistentvolumeclaim/spark-data-pvc created
pod/spark-data-pod created
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

6. 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 \;

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ docker run -v /tmp:/tmp -it bitnami/spark sh -c 'find /opt/bitnami/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 06:04:30.35 INFO  ==>
spark 06:04:30.35 INFO  ==> Welcome to the Bitnami spark container
spark 06:04:30.35 INFO  ==> Subscribe to project updates by watching https://github.com/bitnami/containers
spark 06:04:30.35 INFO  ==> Submit issues and feature requests at https://github.com/bitnami/containers/issues
spark 06:04:30.36 INFO  ==> Upgrade to Tanzu Application Catalog for production environments to access custom-configured and pre-packaged software components. Gain enhanced features
, including Software Bill of Materials (SBOM), CVE scan result reports, and VEX documents. To learn more, visit https://bitnami.com/enterprise
spark 06:04:30.36 INFO  ==>

lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

7.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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ echo "how much wood could a woodpecker chuck if a woodpecker could chuck wood" > /tmp/test.txt
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

8-Copy the JAR file containing the application, and any other required files, to the PVC using t 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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl cp /tmp/my.jar spark-data-pod:/data/my.jar
lbereket625@cloudshell:~ (cs-570-big-data-424802)$  kubectl cp /tmp/test.txt spark-data-pod:/data/test.txt
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

9. Make sure the files a inside the persistent volume

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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl exec -it spark-data-pod -- ls -al /data
total 1540
drwxrwsrwx 2 root root    4096 Jul  5 06:06 .
drwxr-xr-x 1 root root    4096 Jul  5 06:01 ..
-rw-r--r-- 1 1001 root 1564260 Jul  5 06:06 my.jar
-rw-rw-r-- 1 1000 1000      72 Jul  5 06:06 test.txt
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ cat spark-chart.yaml
# spark-chart.yaml

service:
  type: LoadBalancer

worker:
  replicaCount: 3

  extraVolumes:
    - name: spark-data
      persistentVolumeClaim:
        claimName: spark-data-pvc

  extraVolumeMounts:
    - name: spark-data
      mountPath: /data

lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

11. Check the pods is running:

- kubectl get pods

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
nfs-nfs-server-provisioner-0      1/1     Running   0          13m
spark-data-pod                    1/1     Running   0          9m37s
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

12. 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

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" already exists with the same configuration, skipping
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

IMPLEMENTATION

Create image and deploy spark to kubernetes

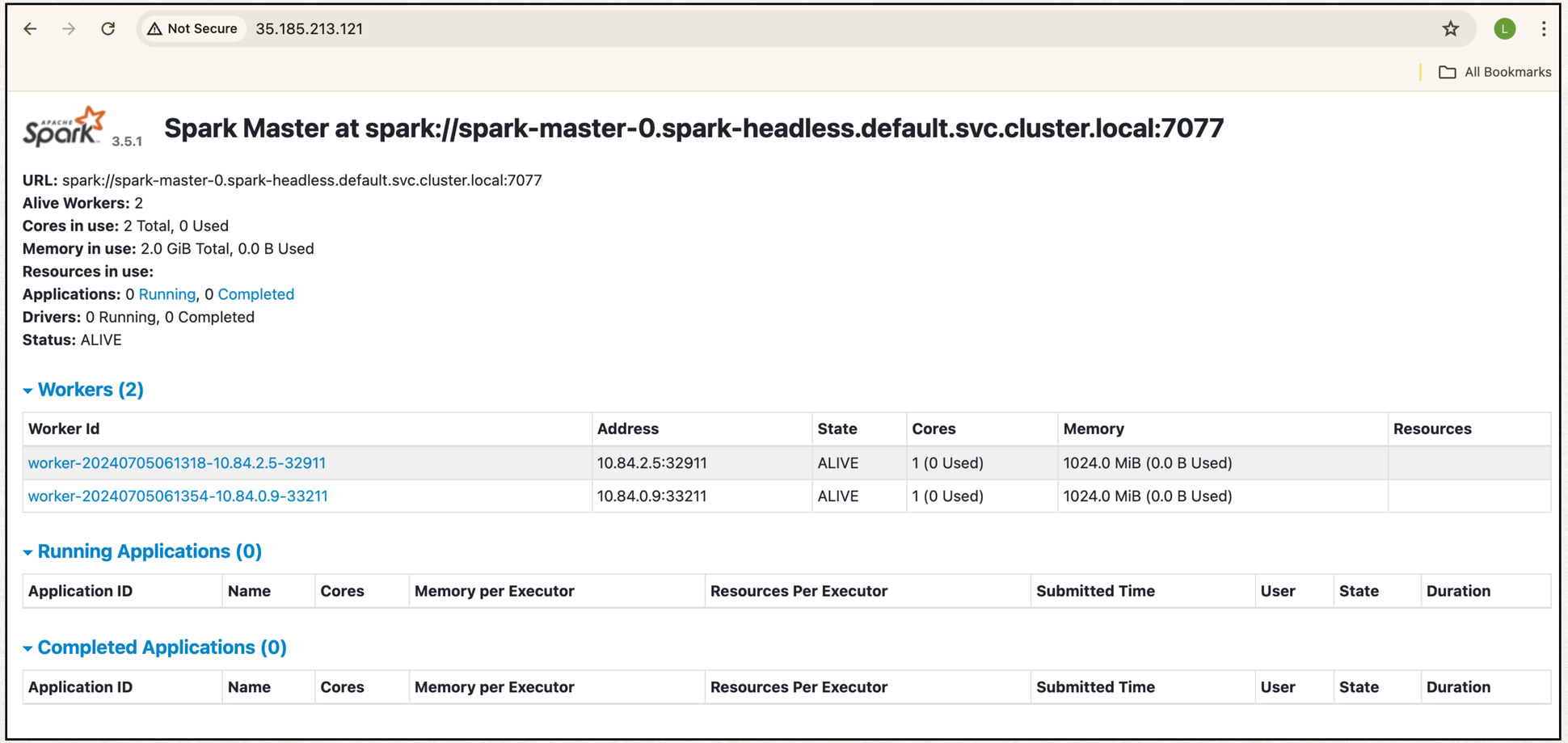
13. Deploy Apache Spark on the Kubernetes cluster using the Bitnami Apache Spark Helm chart and supply it with the configuration file above(cont)
- helm install spark bitnami/spark -f spark-chart.yaml

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ helm install spark bitnami/spark -f spark-chart.yaml
NAME: spark
LAST DEPLOYED: Fri Jul  5 06:12:16 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: spark
CHART VERSION: 9.2.5
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=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loadBalancer.ingress[0]['ip', 'hostname'] }")
  echo http://$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
  obtained at https://github.com/apache/spark/tree/master/bin. Also you can use kubectl run.

  Run the commands below to obtain the master IP and submit your application.

  export EXAMPLE_JAR=$(kubectl exec -ti --namespace default spark-worker-0 -- find examples/jars/ -name 'spark-example*\.jar' | tr -d '\r')
  export SUBMIT_IP=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loadBalancer.ingress[0]['ip', 'hostname'] }")

  kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
    --image docker.io/bitnami/spark:3.5.1-debian-12-r8 \
    -- spark-submit --master spark://$SUBMIT_IP:7077 \
    --deploy-mode cluster \
    --class org.apache.spark.examples.SparkPi \
    $EXAMPLE_JAR 1000

** IMPORTANT: When submit an application the --master parameter should be set to the service IP, if not, the application will not resolve the master. **
```

TEST

Results

. Open the external ip on your browser



TEST

Results

Word Count on Spark
Submit a word count task :

```
kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \

--image docker.io/bitnami/spark:3.0.1-debian-10-r115 \

-- spark-submit --master spark://LOAD-BALANCER-External-ipADDRESS:7077 \ --deploy-mode cluster \

--class org.apache.spark.examples.JavaWordCount \
/data/my.jar /data/test.txt
```

TEST

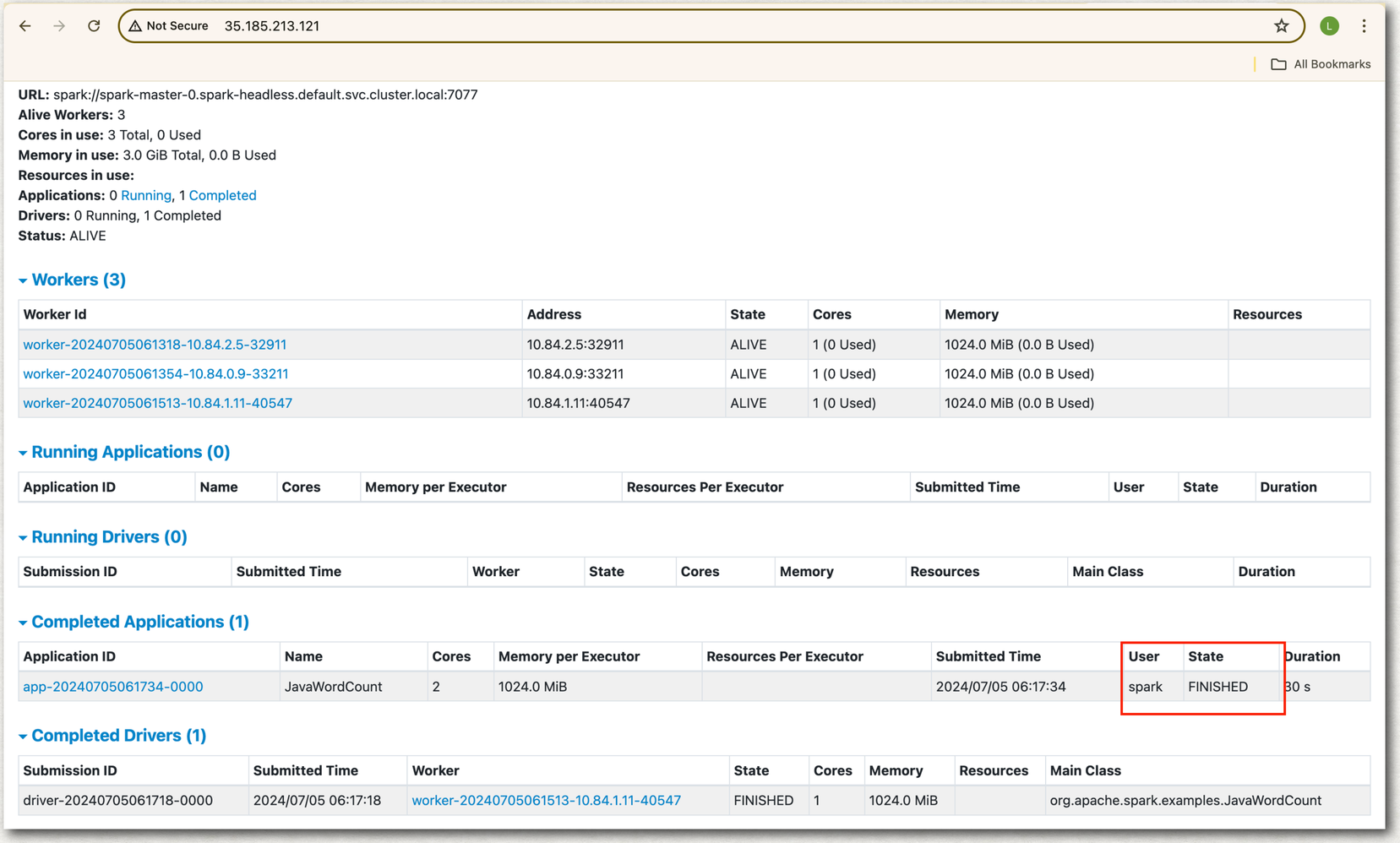
Results

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
    --image docker.io/bitnami/spark:3.5.1-debian-12-r8 \
    -- spark-submit \
    --master spark://35.185.213.121:7077 \
    --deploy-mode cluster \
    --class org.apache.spark.examples.JavaWordCount \
    /data/my.jar /data/test.txt
If you don't see a command prompt, try pressing enter.
24/07/05 06:17:17 INFO SecurityManager: Changing view acls to: spark
24/07/05 06:17:17 INFO SecurityManager: Changing modify acls to: spark
24/07/05 06:17:17 INFO SecurityManager: Changing view acls groups to:
24/07/05 06:17:17 INFO SecurityManager: Changing modify acls groups to:
24/07/05 06:17:17 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: spark; groups with view permissions: EMPTY; users wi
th modify permissions: spark; groups with modify permissions: EMPTY
24/07/05 06:17:17 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/07/05 06:17:18 INFO Utils: Successfully started service 'driverClient' on port 45949.
24/07/05 06:17:18 INFO TransportClientFactory: Successfully created connection to /35.185.213.121:7077 after 72 ms (0 ms spent in bootstraps)
24/07/05 06:17:18 INFO ClientEndpoint: ... waiting before polling master for driver state
24/07/05 06:17:19 INFO ClientEndpoint: Driver successfully submitted as driver-20240705061718-0000
24/07/05 06:17:23 INFO ClientEndpoint: State of driver-20240705061718-0000 is RUNNING
24/07/05 06:17:23 INFO ClientEndpoint: Driver running on 10.84.1.11:40547 (worker-20240705061513-10.84.1.11-40547)
24/07/05 06:17:23 INFO ClientEndpoint: spark-submit not configured to wait for completion, exiting spark-submit JVM.
24/07/05 06:17:24 INFO ShutdownHookManager: Shutdown hook called
24/07/05 06:17:24 INFO ShutdownHookManager: Deleting directory /tmp/spark-f143fb57-1301-4abf-a623-23b5ee7ac7b4
pod "spark-client" deleted
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

TEST

Results

And on your browser, you should see this task finished



TEST

Results

2. Get the name of the worker node

kubectl get pods -o wide | grep WORKER-NODE-ADDRESS

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl get pods -o wide | grep 10.84.2.5
spark-worker-0                 1/1     Running   0          8m54s   10.84.2.5    gke-spark-default-pool-114923bf-mb2z   <none>           <none>
lbereket625@cloudshell:~ (cs-570-big-data-424802)$
```

TEST

Results

3. Execute this pod and see the result of the finished tasks
kubectl exec -it spark-worker-0 -- bash

cd /opt/bitnami/spark/work

cat <taskname>/stdout

```
lbereket625@cloudshell:~ (cs-570-big-data-424802)$ kubectl exec -it spark-worker-2 -- bash
I have no name!@spark-worker-2:/opt/bitnami/spark$ ls
LICENSE  NOTICE  R  README.md  RELEASE  bin  conf  conf.default  data  examples  jars  kubernetes  licenses  logs  python  sbin  tmp  venv  work  yarn
I have no name!@spark-worker-2:/opt/bitnami/spark$ cd /opt/bitnami/spark/work
I have no name!@spark-worker-2:/opt/bitnami/spark/work$ ls
driver-20240705061718-0000
I have no name!@spark-worker-2:/opt/bitnami/spark/work$ cat driver-20240705061718-0000/stdout
if: 1
a: 2
how: 1
could: 2
wood: 2
woodpecker: 2
much: 1
chuck: 2
I have no name!@spark-worker-2:/opt/bitnami/spark/work$
```

TEST

Results

Running python PageRank onPySpark on the pods

1. Execute the spark master pod - kubectl exec -it spark-master-0 -- bash

2. Start pyspark - pyspark

```
lbereket625@cloudshell:/ (cs-570-big-data-424802)$ kubectl exec -it spark-master-0 -- bash
I have no name!@spark-master-0:/opt/bitnami/spark$ pyspark
Python 3.6.13 (default, Apr 19 2021, 18:12:00)
[GCC 8.3.01 on linux
Type "help", "copyright",
"credits" or "license" for more information.
21/04/23 20:51:05 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN"
To adjust logging level use sc. setLogLevel (newLevel). For Spark, use setLogLevel (newLevel).
Welcome to
```

TEST

Results

3. Exit pyspark with
   exit()

```
I have no name!@spark-master-0:/opt/bitnami/spark$ exit
exit
command terminated with exit code 2
lbereket625@cloudshell:/ (cs-570-big-data-424802)$
```

4. Go to the directory where pagerank.py located
   cd /opt/bitnami/spark/examples/src/main/python

5. Run the page rank using pyspark
   spark-submit pagerank.py /opt

```
    at org.apache.spark.sql.execution.datasources.DataSource.resolveRelation(DataSource.scala:418)
    at org.apache.spark.sql.DataFrameReader.loadV1Source(DataFrameReader.scala:326)
    at org.apache.spark.sql.DataFrameReader.$anonfun$load$3(DataFrameReader.scala:308)
    at scala.Option.getOrElse(Option.scala:189)
    at org.apache.spark.sql.DataFrameReader.load(DataFrameReader.scala:308)
    at org.apache.spark.sql.DataFrameReader.text(DataFrameReader.scala:945)
21/04/23 20:52:44 INFO SparkContext: Invoking stop() from shutdown hook
21/04/23 20:52:44 INFO SparkUI: Stopped Spark web UI at http://spark-master-0.spark-headless.default.svc.cluster.local:4040
21/04/23 20:52:44 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
21/04/23 20:52:44 INFO MemoryStore: MemoryStore cleared
21/04/23 20:52:44 INFO BlockManager: BlockManager stopped
21/04/23 20:52:44 INFO BlockManagerMaster: BlockManagerMaster stopped
21/04/23 20:52:44 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
21/04/23 20:52:44 INFO SparkContext: Successfully stopped SparkContext
21/04/23 20:52:44 INFO ShutdownHookManager: Shutdown hook called
21/04/23 20:52:44 INFO ShutdownHookManager: Deleting directory /tmp/spark-65e26f5e-996e-40ab-a11e-2f753a90b940
21/04/23 20:52:44 INFO ShutdownHookManager: Deleting directory /tmp/spark-65e265e-996e-40ab-alle-2f753a90b940/pyspark-16725e8a-5068-4bdc-af9d-C
21/04/23 20:52:44 INFO ShutdownHookManager: Deleting directory /tmp/spark-df67652d-97af-46a9-99df-30c667da65e3
```

CONCLUSION

This project leveraged PySpark to execute Word Count and Page Rank algorithms on Apache Spark deployed on Kubernetes. Deploying Spark on Kubernetes offers compelling advantages. Firstly, containerization enhances portability and simplifies dependency management, facilitating consistent and reliable build processes. Secondly, efficient resource utilization results in significant cost efficiencies. Lastly, Kubernetes integration enriches the ecosystem, providing cloud-agnostic flexibility and reducing vendor lock-in, thereby supporting scalable and adaptable data processing solutions

REFERENCE

- https://hc.labnet.sfbu.edu/~henry/sfbu/course/master_apache_spark/kubernetes/slide/Wordcount_PageRank_running_on_spark_delpoying_to_kubernetes_on_GKE.pdf

- https://github.com/MikaelaMontaos/Cloud-Computing-Infrastructure/blob/main/PySpark/Word%20Count%20and%20Page%20Rank%20using%20PySpark%20(Google%20Slides%20Version)

MY GITHUB LINK

Project Github Link : https://github.com/LidiaYon/Cloud-Computing-Infrastructure-/tree/main/WorkCount_PageRank