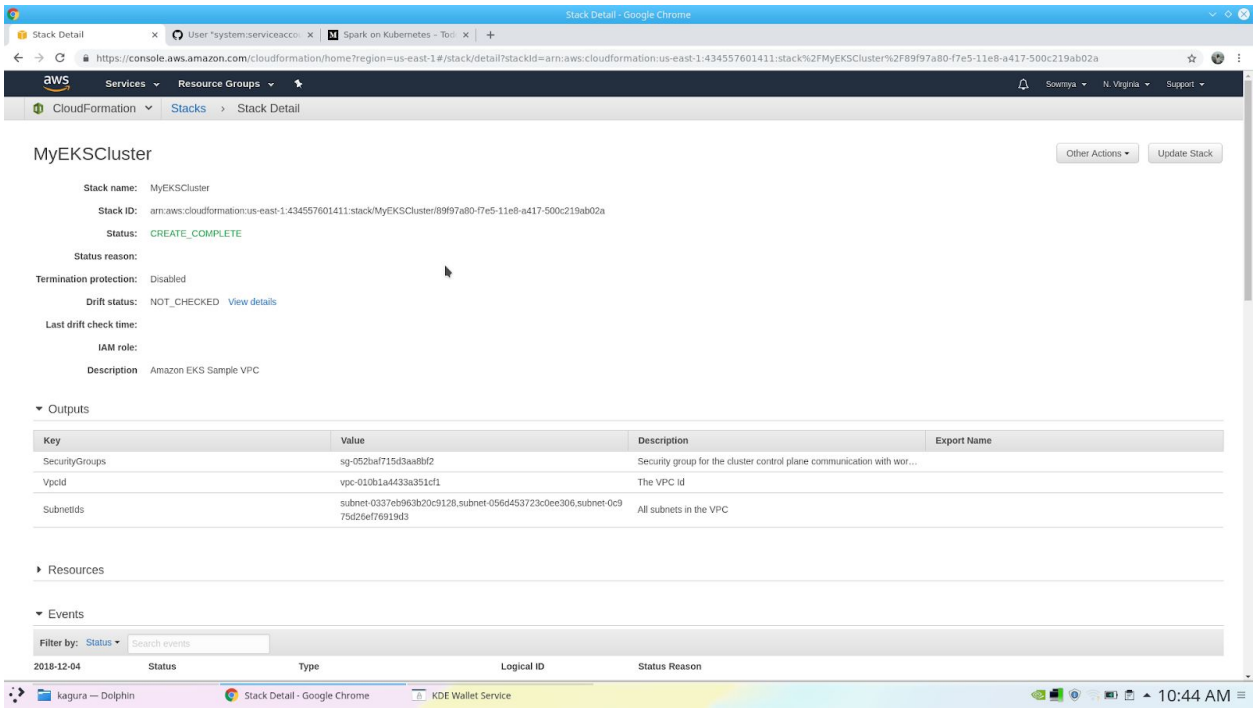
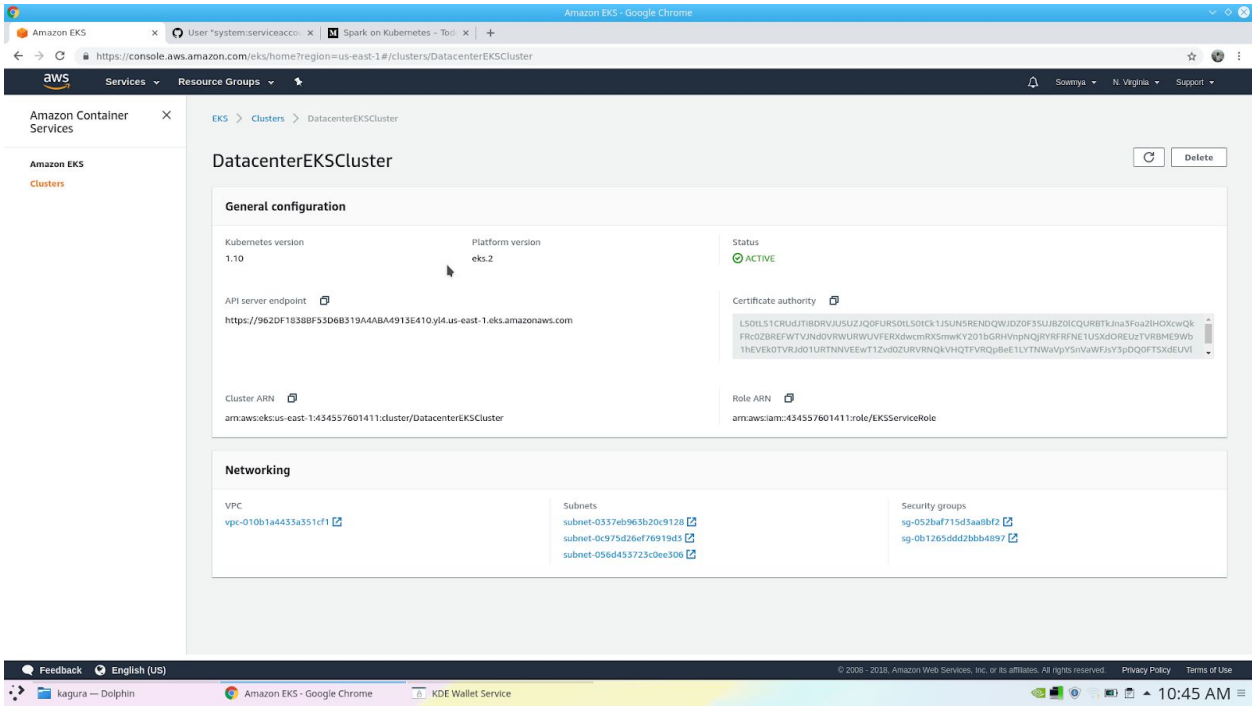
SPARK ON KUBERNETES - USING EKS AND MINIKUBE

EKS (Amazon Web Services' Managed Kubernetes Service)

Cluster VPC (Created using CloudFormation and an S3 Template URL)

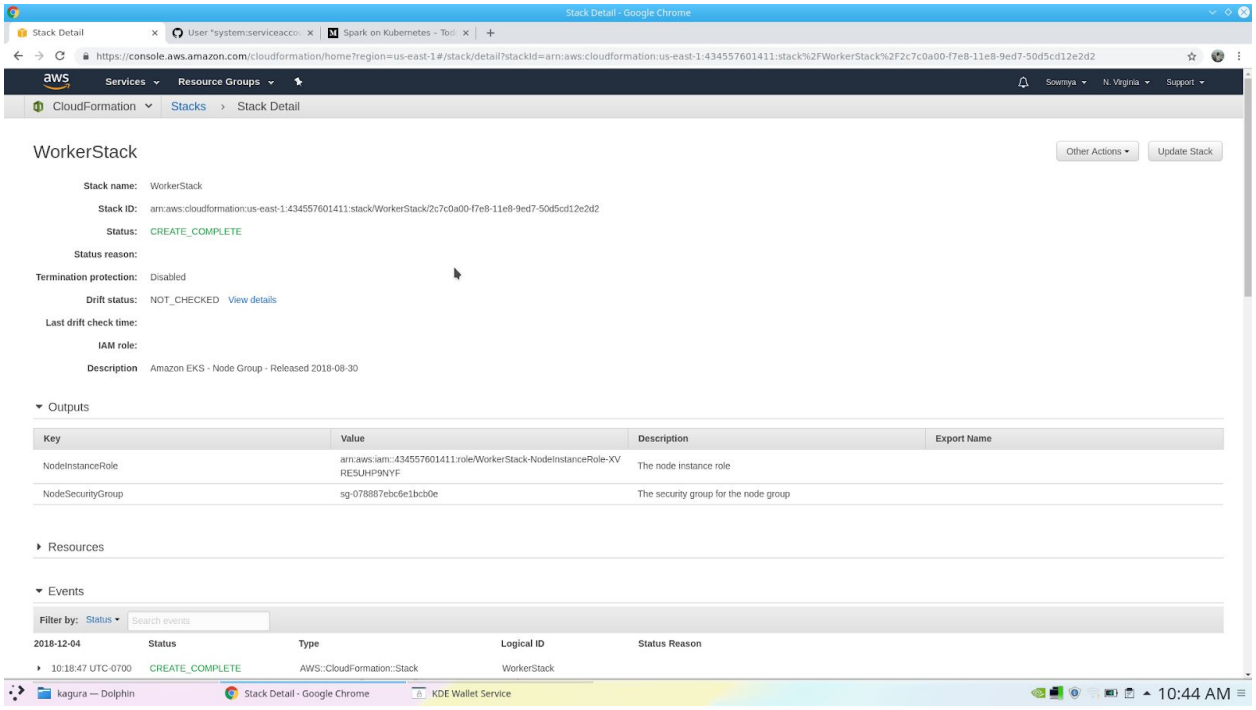


AWS EKS Cluster Plane (Created using EKS)

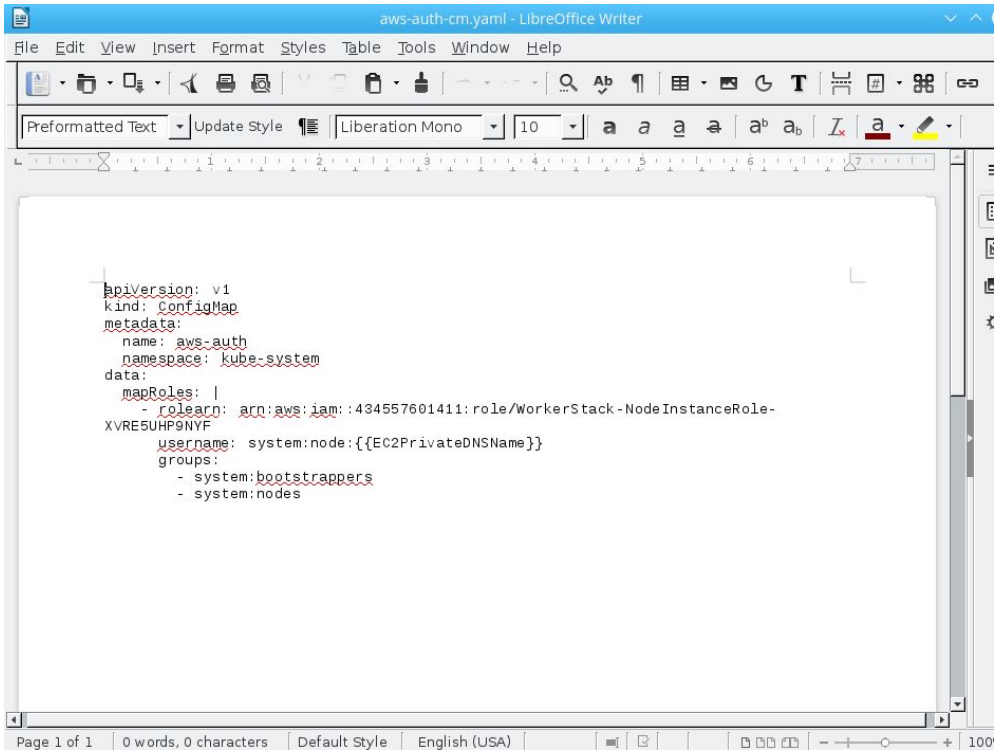


- **Here, care should be taken to select appropriate VPC (created earlier through CloudFormation template) and thus, appropriate security groups.**
- **The console automatically shows the default VPC and its associated security groups, and creating the cluster with these will not serve up a running Kube-DNS pod and hence show TCP Connection Issues when the spark job is run. This is because, the default security groups do not give the cluster access to worker nodes. This is uniquely what the ControlPlaneSecurityGroup does, which is part of the Cluster VPC created.**

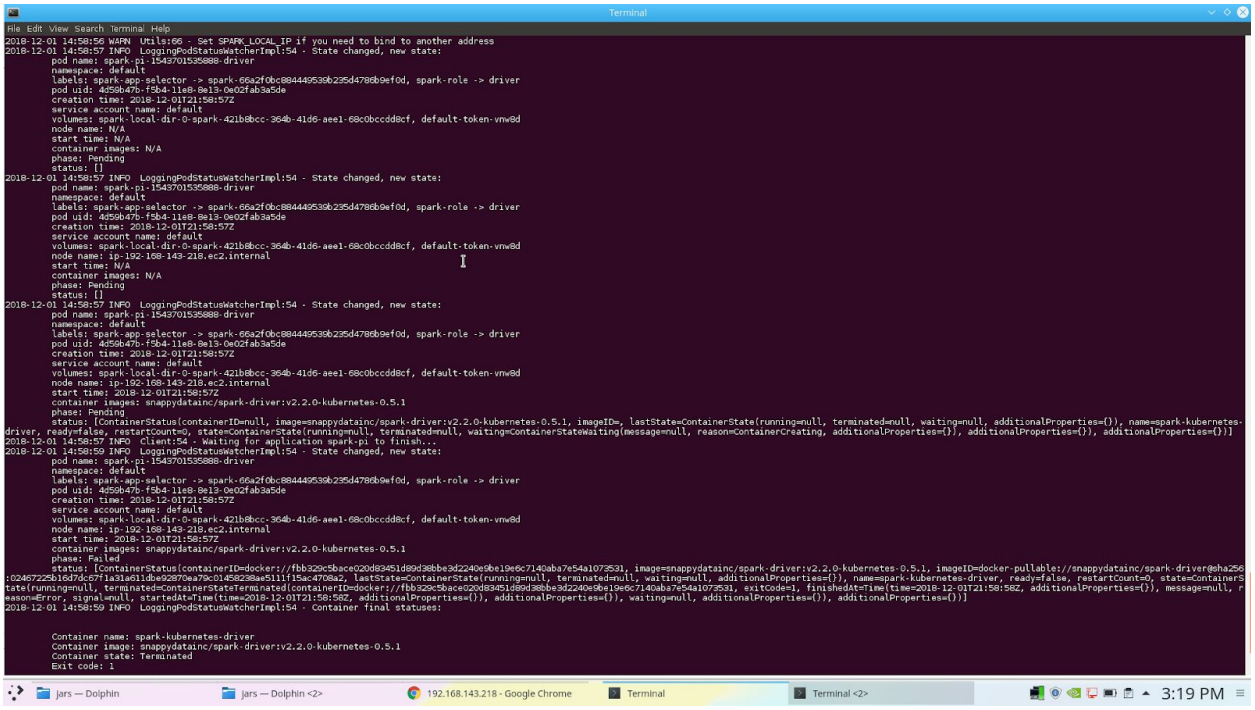
Worker Nodes



The Instance Role ARN is copied into the aws-auth-cm.yaml file.



Failure of EKS Spark Job



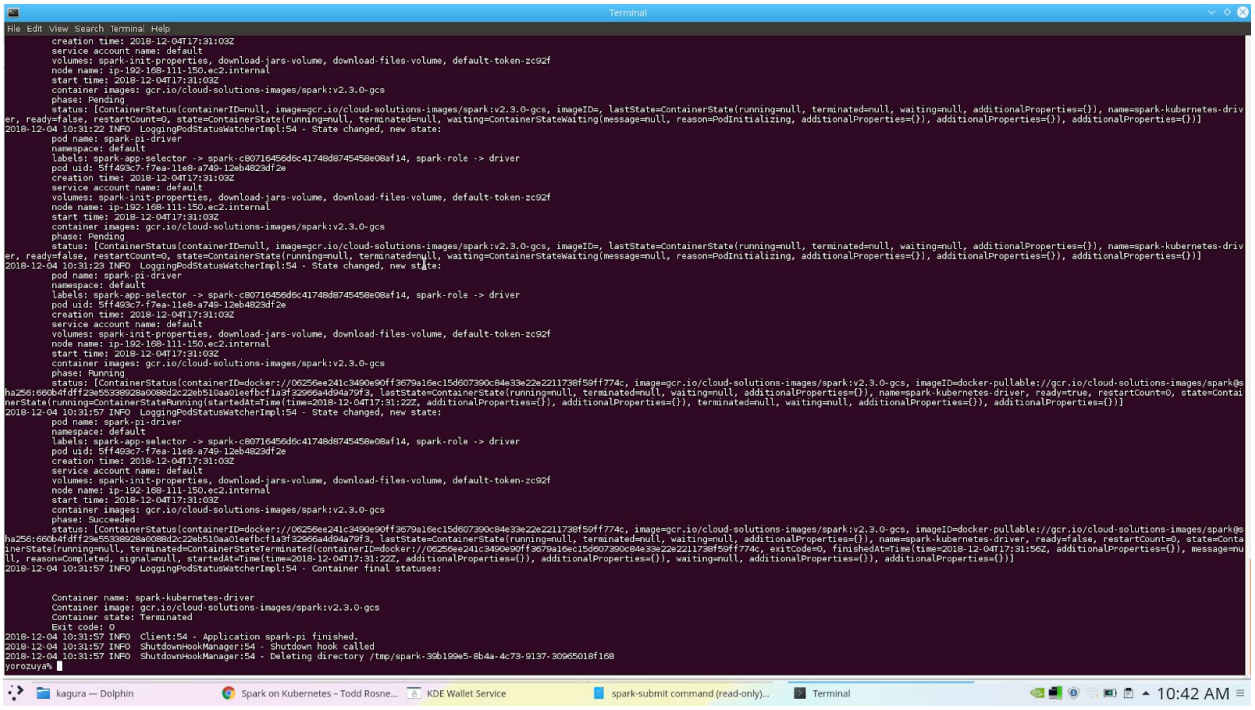
On observing the logs in the Kubernetes Dashboard, the following issues were observed and debugged:

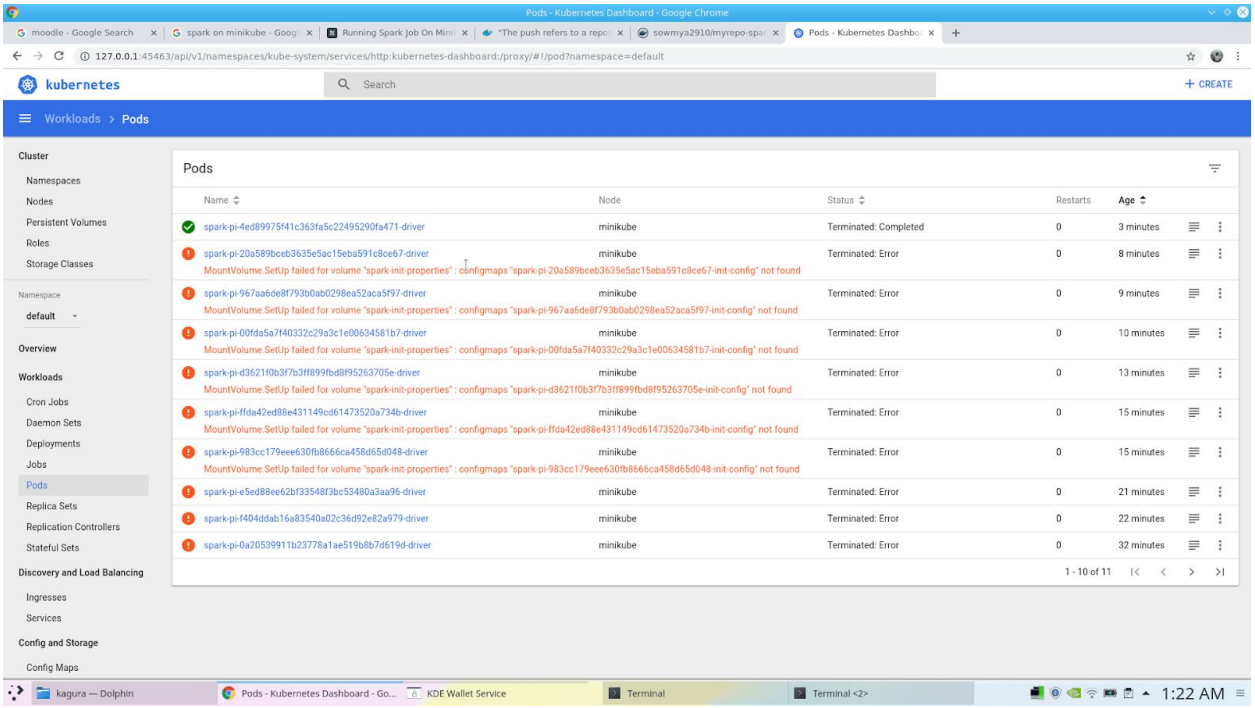
- **Error: Could not find or load main class SparkPi - This was resolved by uploading the jar file to a public S3 bucket and using the link in the spark-submit command.**
- **MountVolume.SetUp failed for volume "spark-init-properties" : configmaps "spark-pi-f35e896061393afebbb78dc7f3e40a7e-init-config" not found - This was resolved by creating cluster role bindings for default namespace.**

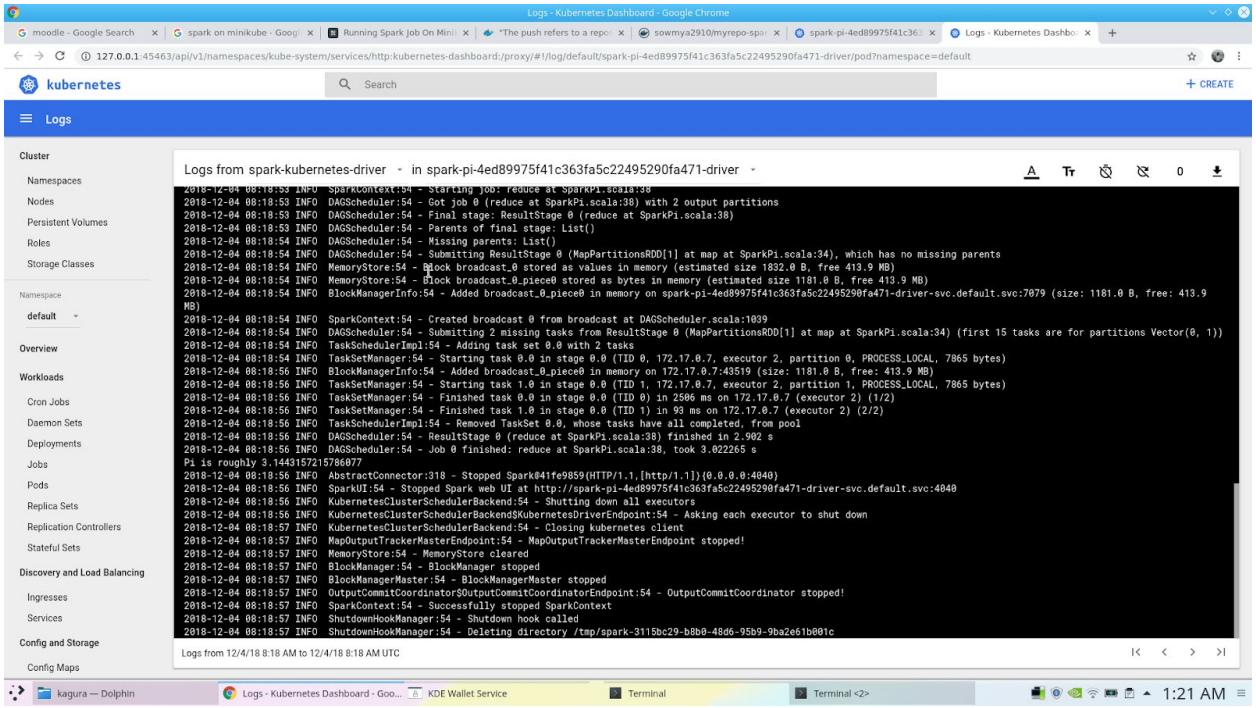
Also, we tried the steps given in this github repo :
https://github.com/SnappyDataInc/spark-on-k8s

We used EKS, and in the Setup Helm Charts step, faced a major issue, which, till date, we haven't been able to rectify: After installing appropriate Helm version, and doing helm init (which installs Tiller), and after confirming that tiller is installed, when we tried helm ls or helm version, the client (helm) successfully showed up but the server (Tiller pod) stayed in ContainerCreating state and it hence said 'Could not find a ready tiller pod', which was weird. Anyhow, this approach was then abandoned and we were able to successfully create cluster and run the spark application the easier way!

Screenshots showing successful run:





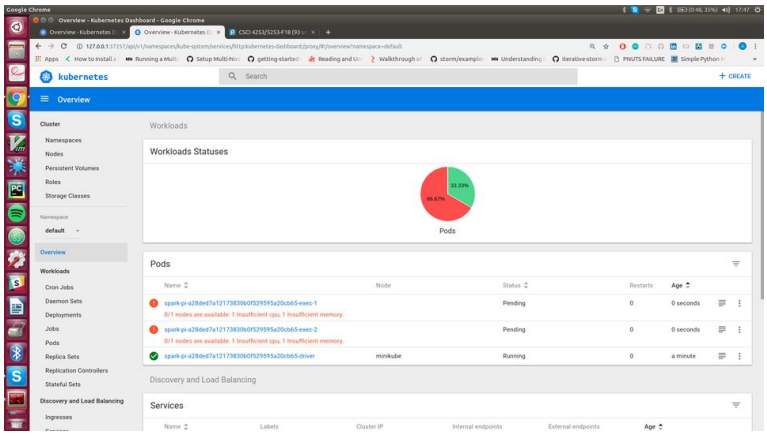


MINIKUBE

The issues we faced were similar to those observed during EKS implementation - Main class and configmaps, and were resolved the same way - use of S3 and creating cluster role bindings.

One other significant issue we faced was that of Insufficient CPU, Memory.

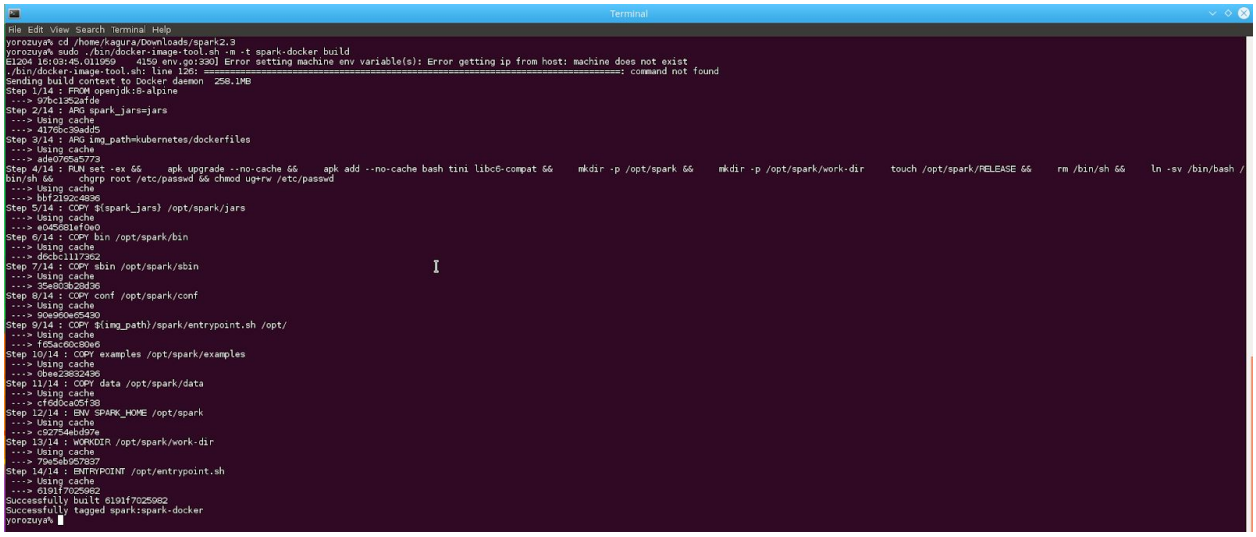
- **Insufficient CPU, Insufficient Memory:**

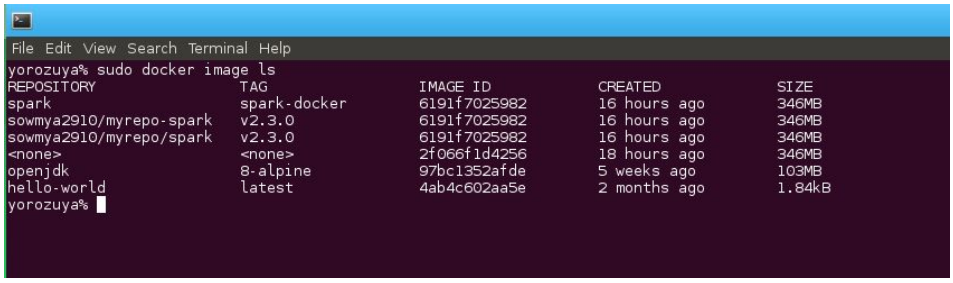


This was resolved by restarting kubectl pods (stopping and deleting minikube and starting it back up) - minikube stop ; minikube delete ; minikube start --memory 8192 --cpus 4

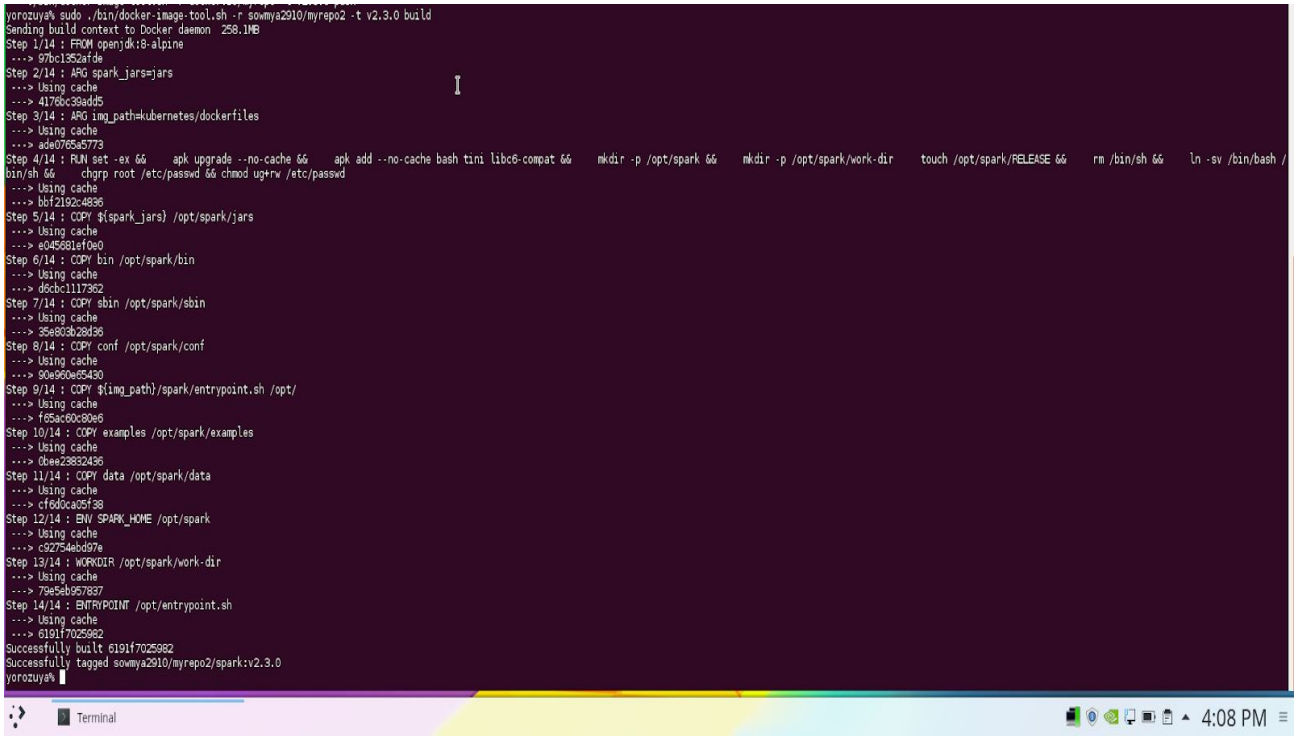
Minikube start

Creation of Spark Docker Image





Building the image for push to docker hub repository



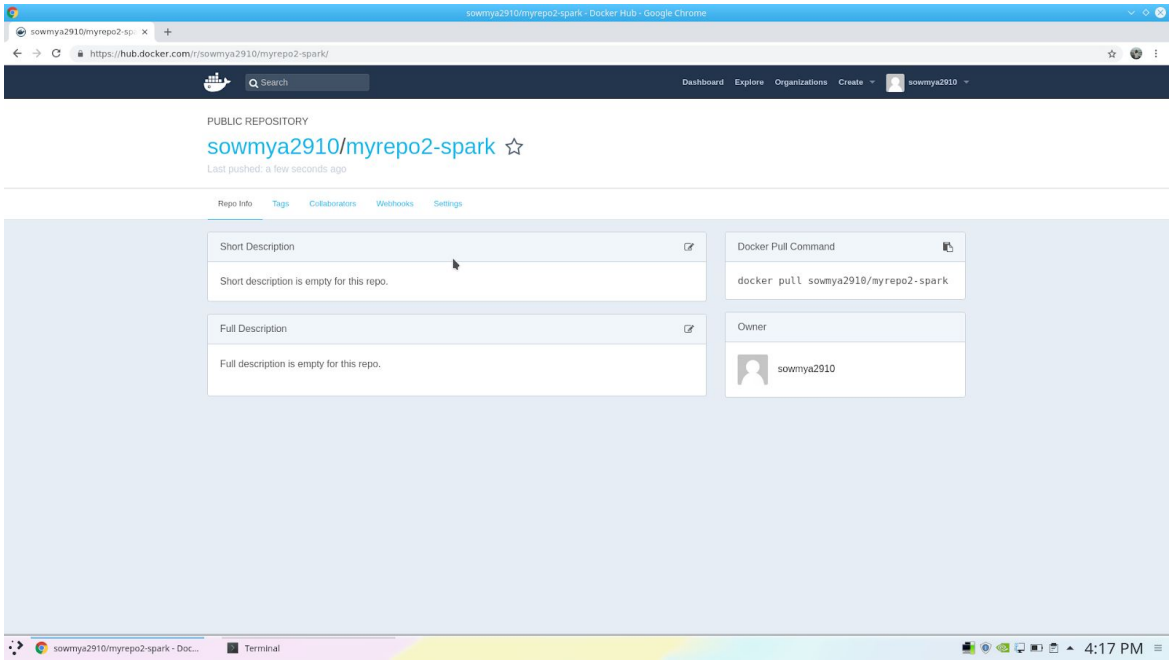
Docker hub login (for access to repository)

Renaming repo and pushing image

```
yorozuya% sudo docker tag sowmya2910/myrepo2/spark sowmya2910/myrepo2-spark
Error response from daemon: No such image: sowmya2910/myrepo2/spark:latest
yorozuya% sudo docker tag sowmya2910/myrepo2/spark:v2.3.0 sowmya2910/myrepo2-spark:v2.3.0
yorozuya% sudo docker push sowmya2910/myrepo2-spark:v2.3.0
The push refers to a repository [docker.io/sowmya2910/myrepo2-spark]
fd99da662d66: Pushed
52f485c165ba: Pushed
781dedd9877c: Pushed
7fa21420da76: Pushed
f22a61e86d49: Pushed
fd4e89e918a4: Pushed
6e97b177306c: Pushed
4933ed0cc7e7: Pushed
ed6f0bd39121: Pushed
0c3170905795: Pushed
df64d3292fd6: Pushed
v2.3.0: digest: sha256:2fd2b4fbca902d9f87f120f6db857e967c28daf19a67eec7b62532762ebbcd7d size: 2624
yorozuya%
```

The need to rename was found as a solution to an issue, during which the push didn't happen since the repo could not be accessed (even after login). A workaround suggested that we rename the repo, (it apparently doesn't work if it has more than two parameters! (sowmya2910/myrepo2/spark) along with the image tag) and hence it was renamed as sowmya2910/myrepo2-spark, and it pushed.!

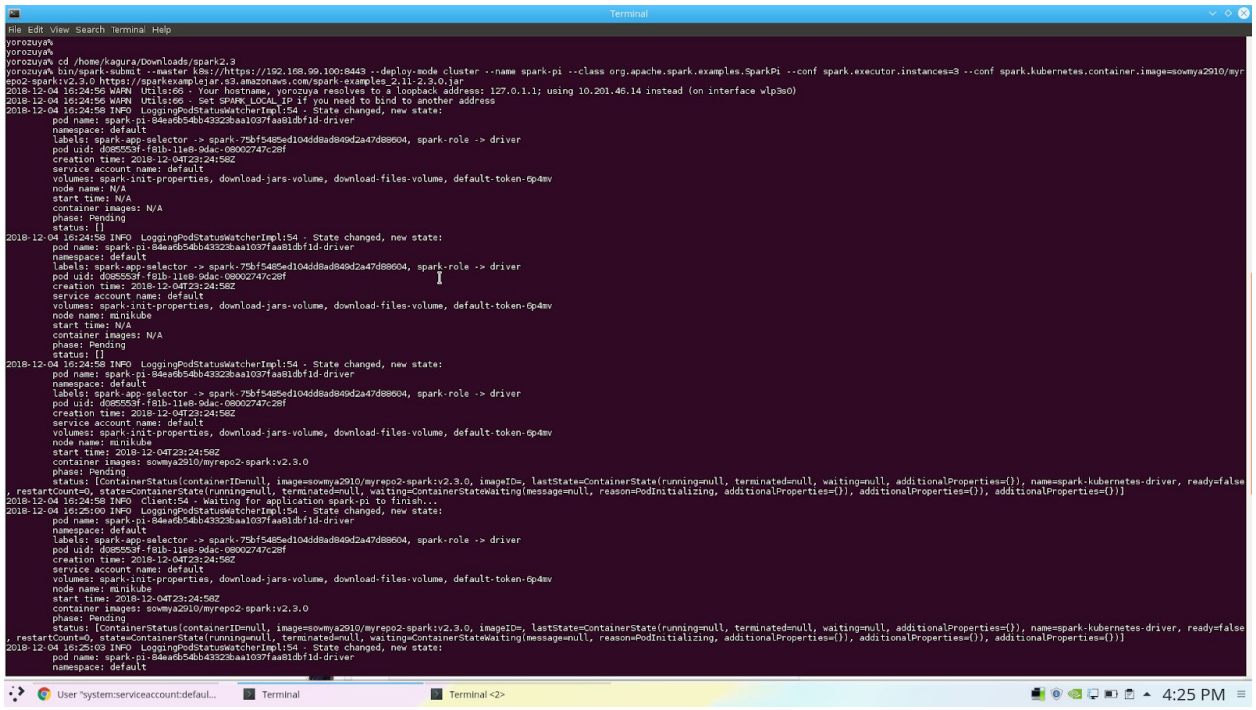
Screenshot showing repo creation and last push notification:

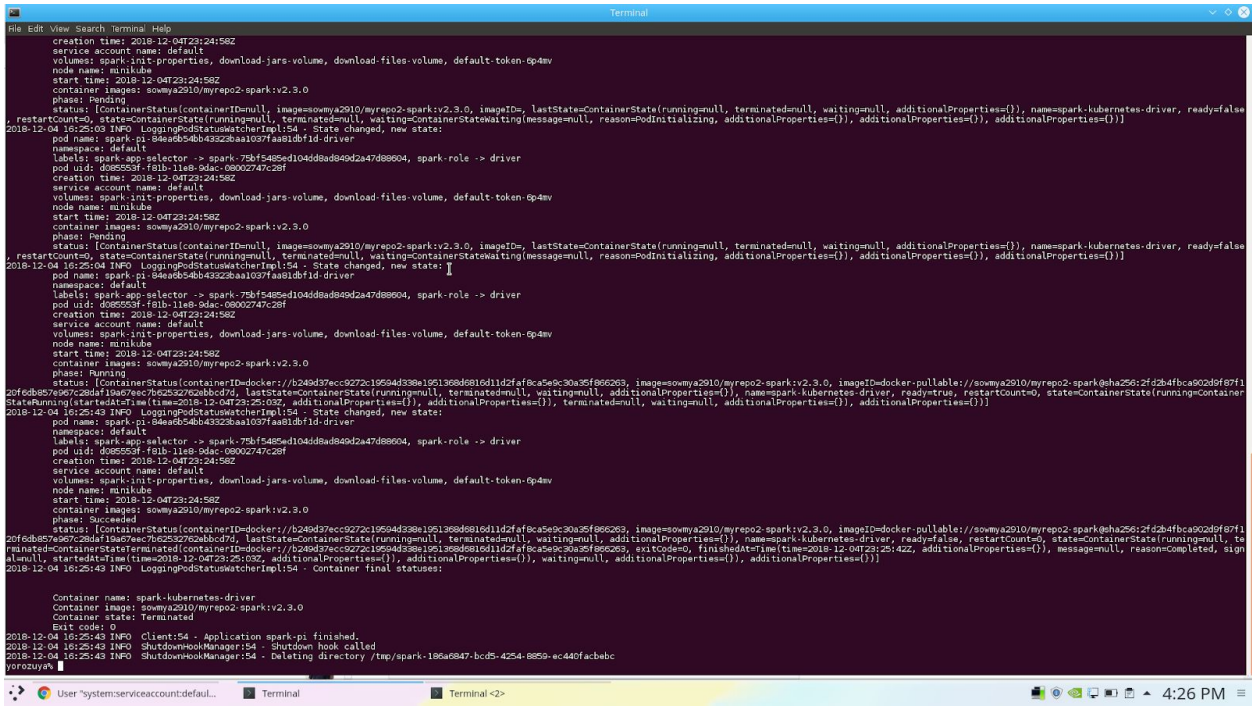


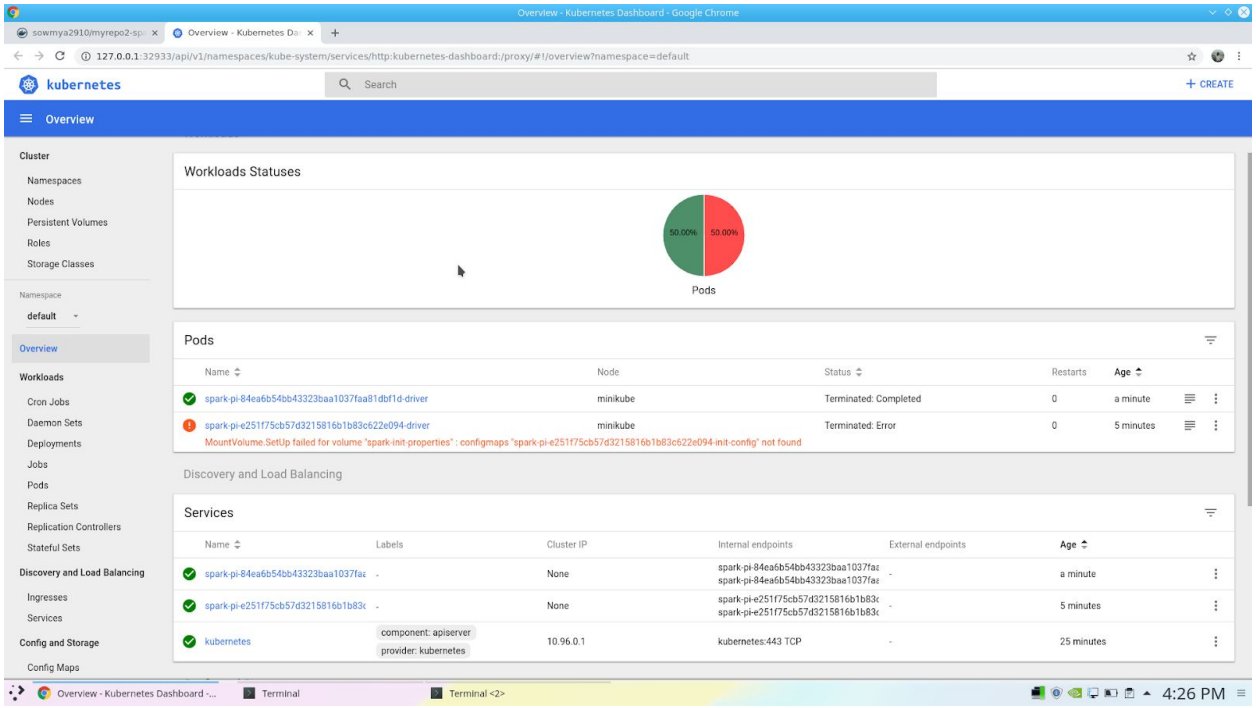
Creation of Cluster role bindings

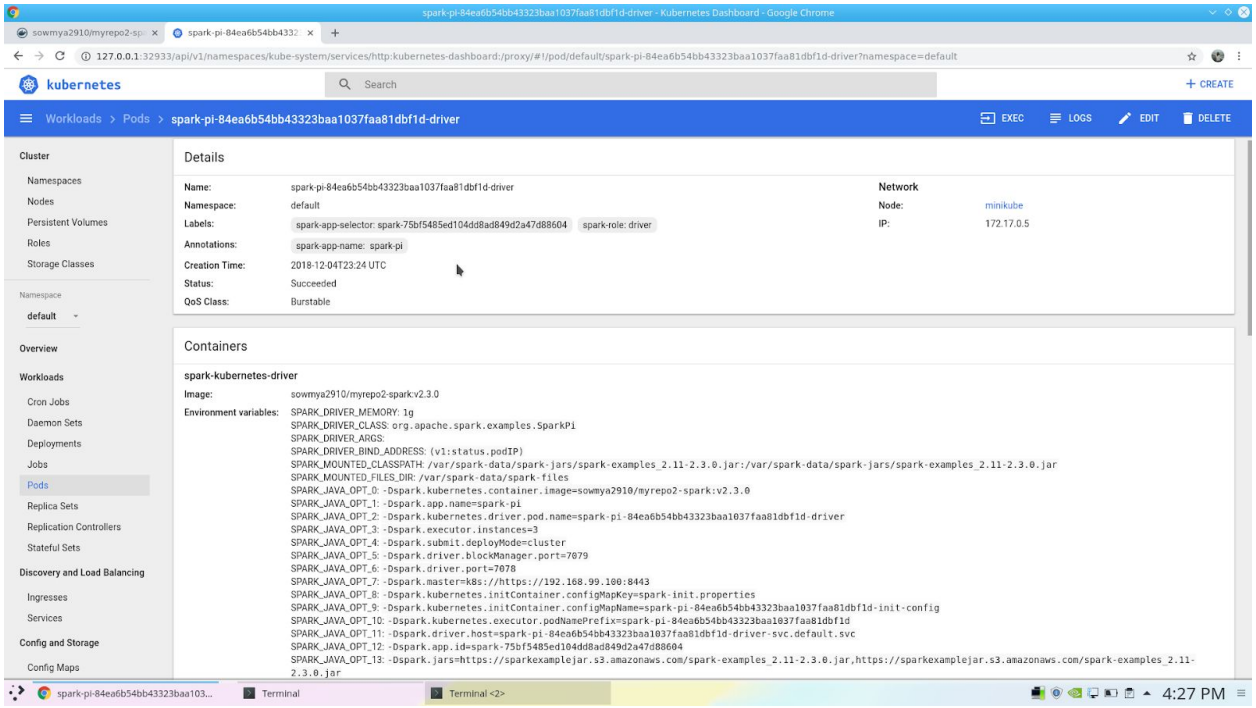
```
yorozuya% kubectl create clusterrolebinding cluster-system-anonymous --clusterrole=cluster-admin --user=system:anonymous
clusterrolebinding.rbac.authorization.k8s.io/cluster-system-anonymous created
yorozuya% kubectl create clusterrolebinding default-admin --clusterrole cluster-admin --serviceaccount=default:default
clusterrolebinding.rbac.authorization.k8s.io/default-admin created
yorozuya%
yorozuya%
yorozuya%
```

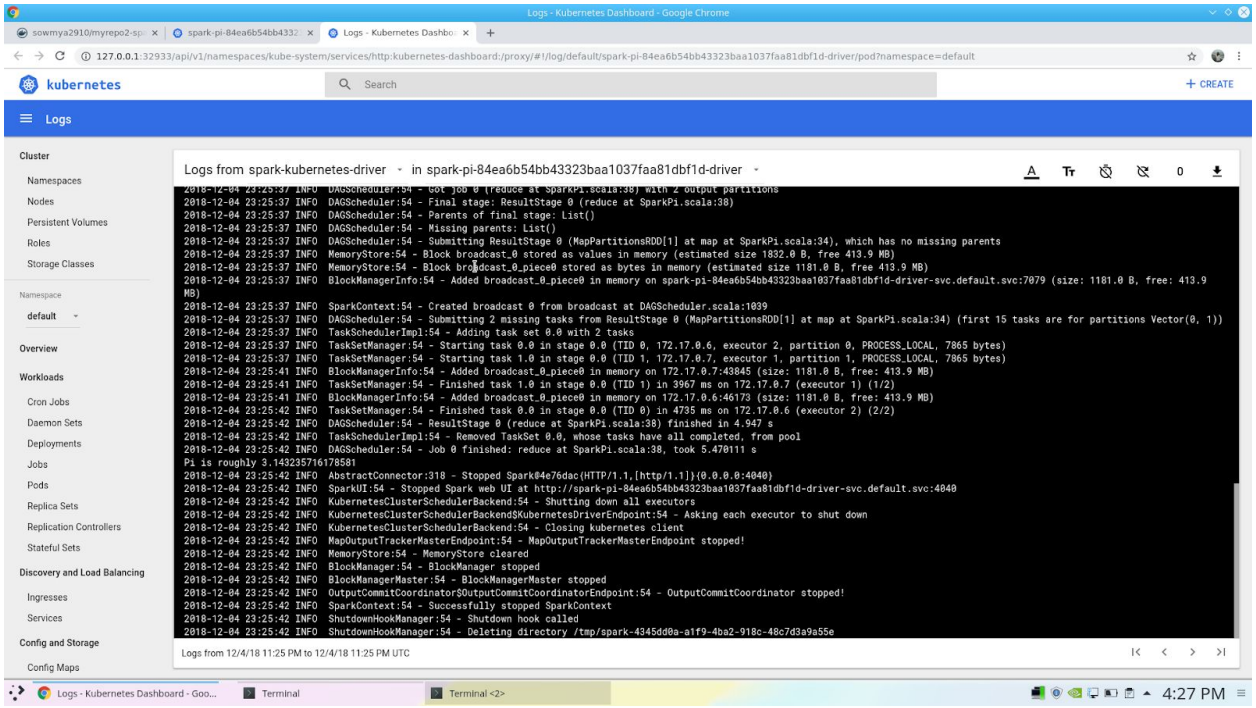
Screenshots showing successful run:




Terminal
File Edit View Search Terminal Help
creation time: 2018-12-04T23:24:58Z
service account name: default
volumes: spark-init-properties, download-jars-volume, download-files-volume, default-token-6p4mv
node name: minikube
start time: 2018-12-04T23:24:58Z
container images: sowmya2910/myrepo2-spark:v2.3.0
phase: Pending
status: [ContainerStatus(containerID=null, image=sowmya2910/myrepo2-spark:v2.3.0, imageID=, lastState=ContainerState(running=null, terminated=null, waiting=null, additionalProperties={}), name=spark-kubernetes-driver, ready=false, restartCount=0, state=ContainerState(running=null, terminated=null, waiting=ContainerStateWaiting(message=null, reason=PodInitializing, additionalProperties={}), additionalProperties={}), additionalProperties={})]
2018-12-04 16:25:03 INFO LoggingPodStatusWatcherImpl:54 - State changed, new state:
pod name: spark-pi-84ea6b54bb43323baa1037faa81dbf1d-driver
namespace: default
labels: spark-app-selector -> spark-75bf5485ed104dd8ad849d2a47d88604, spark-role -> driver
pod uid: d085553f-f81b-11e8-9dac-08002747c28f
creation time: 2018-12-04T23:24:58Z
service account name: default
volumes: spark-init-properties, download-jars-volume, download-files-volume, default-token-6p4mv
node name: minikube
start time: 2018-12-04T23:24:58Z
container images: sowmya2910/myrepo2-spark:v2.3.0
phase: Pending
status: [ContainerStatus(containerID=null, image=sowmya2910/myrepo2-spark:v2.3.0, imageID=, lastState=ContainerState(running=null, terminated=null, waiting=null, additionalProperties={}), name=spark-kubernetes-driver, ready=false, restartCount=0, state=ContainerState(running=null, terminated=null, waiting=ContainerStateWaiting(message=null, reason=PodInitializing, additionalProperties={}), additionalProperties={}), additionalProperties={})]
2018-12-04 16:25:04 INFO LoggingPodStatusWatcherImpl:54 - State changed, new state:
pod name: spark-pi-84ea6b54bb43323baa1037faa81dbf1d-driver
namespace: default
labels: spark-app-selector -> spark-75bf5485ed104dd8ad849d2a47d88604, spark-role -> driver
pod uid: d085553f-f81b-11e8-9dac-08002747c28f
creation time: 2018-12-04T23:24:58Z
service account name: default
volumes: spark-init-properties, download-jars-volume, download-files-volume, default-token-6p4mv
node name: minikube
start time: 2018-12-04T23:24:58Z
container images: sowmya2910/myrepo2-spark:v2.3.0
phase: Running
status: [ContainerStatus(containerID=docker://b249d37ecc9272c19594d338e1951368d8816d11d2faf8ca5e9c30a35f866263, image=sowmya2910/myrepo2-spark:v2.3.0, imageID=docker-pullable://sowmya2910/myrepo2-spark@sha256:2fd2b4fbca902d9f87f120f6db857e967c28daf19a67eec7b62532762ebbcd7d, lastState=ContainerState(running=null, terminated=null, waiting=null, additionalProperties={}), name=spark-kubernetes-driver, ready=true, restartCount=0, state=ContainerState(running=ContainerStateRunning(startedAt=Time(time=2018-12-04T23:25:03Z, additionalProperties={}), additionalProperties={}), terminated=null, waiting=null, additionalProperties={}), additionalProperties={})]
2018-12-04 16:25:43 INFO LoggingPodStatusWatcherImpl:54 - State changed, new state:
pod name: spark-pi-84ea6b54bb43323baa1037faa81dbf1d-driver
namespace: default
labels: spark-app-selector -> spark-75bf5485ed104dd8ad849d2a47d88604, spark-role -> driver
pod uid: d085553f-f81b-11e8-9dac-08002747c28f
creation time: 2018-12-04T23:24:58Z
service account name: default
volumes: spark-init-properties, download-jars-volume, download-files-volume, default-token-6p4mv
node name: minikube
start time: 2018-12-04T23:24:58Z
container images: sowmya2910/myrepo2-spark:v2.3.0
phase: Succeeded
status: [ContainerStatus(containerID=docker://b249d37ecc9272c19594d338e1951368d8816d11d2faf8ca5e9c30a35f866263, image=sowmya2910/myrepo2-spark:v2.3.0, imageID=docker-pullable://sowmya2910/myrepo2-spark@sha256:2fd2b4fbca902d9f87f120f6db857e967c28daf19a67eec7b62532762ebbcd7d, lastState=ContainerState(running=null, terminated=null, waiting=null, additionalProperties={}), name=spark-kubernetes-driver, ready=false, restartCount=0, state=ContainerState(running=null, terminated=ContainerStateTerminated{containerID=docker://b249d37ecc9272c19594d338e1951368d8816d11d2faf8ca5e9c30a35f866263, exitCode=0, finishedAt=Time(time=2018-12-04T23:25:42Z, additionalProperties={}), message=null, reason=Completed, signal=null, startedAt=Time(time=2018-12-04T23:25:03Z, additionalProperties={}), additionalProperties={}), waiting=null, additionalProperties={}), additionalProperties={})]
2018-12-04 16:25:43 INFO LoggingPodStatusWatcherImpl:54 - Container final statuses:
Container name: spark-kubernetes-driver
Container image: sowmya2910/myrepo2-spark:v2.3.0
Container state: Terminated
Exit code: 0
2018-12-04 16:25:43 INFO Client:54 - Application spark-pi finished.
2018-12-04 16:25:43 INFO ShutdownHookManager:54 - Shutdown hook called
2018-12-04 16:25:43 INFO ShutdownHookManager:54 - Deleting directory /tmp/spark-186a6847-bcd5-4254-8859-ec440facbebc
yorozuya%
User "system:serviceaccount:defaul...
Terminal
Terminal <2>
4:26 PM


Overview - Kubernetes Dashboard - Google Chrome
127.0.0.1:32933/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/#!/overview?namespace=default
kubernetes
Search
+ CREATE
Overview
Cluster
Namespaces
Nodes
Persistent Volumes
Roles
Storage Classes
Namespace
default
Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Discovery and Load Balancing
Ingresses
Services
Config and Storage
Config Maps
Workloads Statuses
50.00%
50.00%
Pods
Pods
Name
Node
Status
Restarts
Age
spark-pi-84ea6b54bb43323baa1037faa81dbf1d-driver
minikube
Terminated: Completed
0
a minute
spark-pi-e251f75cb57d3215816b1b83c622e094-driver
minikube
Terminated: Error
0
5 minutes
MountVolume.SetUp failed for volume "spark-init-properties" : configmaps "spark-pi-e251f75cb57d3215816b1b83c622e094-init-config" not found
Discovery and Load Balancing
Services
Name
Labels
Cluster IP
Internal endpoints
External endpoints
Age
spark-pi-84ea6b54bb43323baa1037faa
None
spark-pi-84ea6b54bb43323baa1037faa
spark-pi-84ea6b54bb43323baa1037faa
a minute
spark-pi-e251f75cb57d3215816b1b83c
None
spark-pi-e251f75cb57d3215816b1b83c
spark-pi-e251f75cb57d3215816b1b83c
5 minutes
kubernetes
component: apiserver
provider: kubernetes
10.96.0.1
kubernetes:443 TCP
25 minutes
Overview - Kubernetes Dashboard -...
Terminal
Terminal <2>
4:26 PM





Thus, we were successfully able to run spark jobs on clusters created by both EKS as well as Minikube. The process was a little herculean, with it taking more time than we expected and (too) many errors encountered along the way, but lots of support made sure we crossed the line! Lots of learning too. Next steps are to integrate MongoDB

database to Kubernetes and to find a way to transfer spark query results to the Django App!