Run Python App on K8 Cluster

SSH to your AWS Workstation
ssh devops@<public-ip-addr> of your Workstation
Password is : **Dev0p$!!/**
Replace <your-name> with your name throughout the lab.

1. Run the below commands on your AWS-Workstation.

```
$ sudo su
# cd application/
# curl -f https://pastebin.com/raw/SdZPXvXK >  python-app-<your-name>.yaml
```

2. **Edit the python-app-<your-name>.yaml script.**

```
# vim python-app-<your-name>.yaml
```

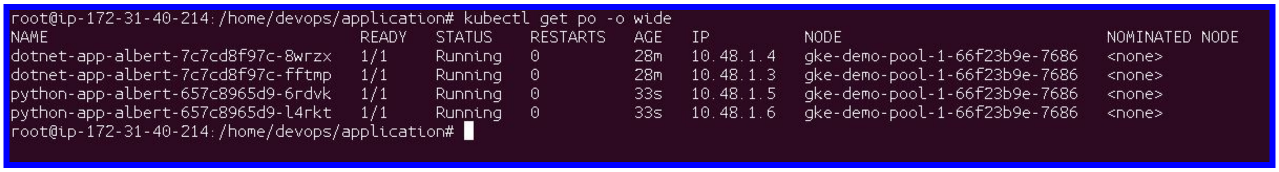
Update the image: **lovescloud/docker-python:latest** with your dockerhub image name that you uploaded to docker hub in docker lab Pushing images to docker Hub for python. Update <your-name> with your name.
Save and exit by pressing the ESC key and type wq to save and quit by pressing enter

3. Run the below commands to deploy the python application on your Kubernetes Cluster

```
# kubectl apply -f  python-app-<your-name>.yaml
```

4. Check the NODE where your app has been deployed.

```
# kubectl get po -o wide
```

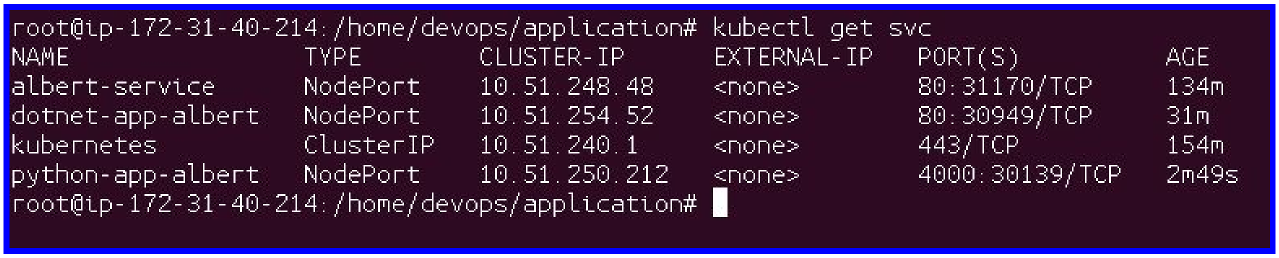


```
root@ip-172-31-40-214:/home/devops/application# kubectl get po -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP          NODE                                NOMINATED NODE
dotnet-app-albert-7c7cd8f97c-8wrzx   1/1     Running   0          28m   10.48.1.4   gke-demo-pool-1-66f23b9e-7686   <none>
dotnet-app-albert-7c7cd8f97c-fftmp   1/1     Running   0          28m   10.48.1.3   gke-demo-pool-1-66f23b9e-7686   <none>
python-app-albert-657c8965d9-6rdvk   1/1     Running   0          33s   10.48.1.5   gke-demo-pool-1-66f23b9e-7686   <none>
python-app-albert-657c8965d9-l4rkt   1/1     Running   0          33s   10.48.1.6   gke-demo-pool-1-66f23b9e-7686   <none>
root@ip-172-31-40-214:/home/devops/application#
```

In this example the python app has been deployed to the NODE **gke-demo-pool-1-66f23b9e-7686**

5. Check the NODEPORT of the application

```
# kubectl get svc
```

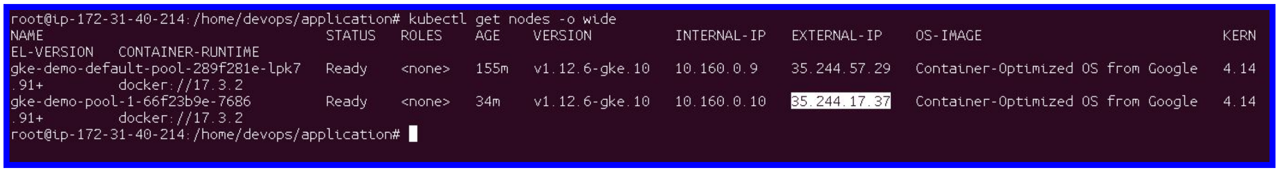


```
root@ip-172-31-40-214:/home/devops/application# kubectl get svc
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
albert-service      NodePort    10.51.248.48    <none>        80:31170/TCP     134m
dotnet-app-albert   NodePort    10.51.254.52    <none>        80:30949/TCP     31m
kubernetes          ClusterIP   10.51.240.1     <none>        443/TCP          154m
python-app-albert   NodePort    10.51.250.212   <none>        4000:30139/TCP   2m49s
root@ip-172-31-40-214:/home/devops/application#
```

In this example the Python application has been exposed on port **30139** as shown in the below screenshot.

6. Check the public IP of the NODE (**gke-demo-pool-1-66f23b9e-7686**) to access the Python application web page from the NODE Public IP address and Node Port on which it is exposed at.

# kubectl get nodes -o wide



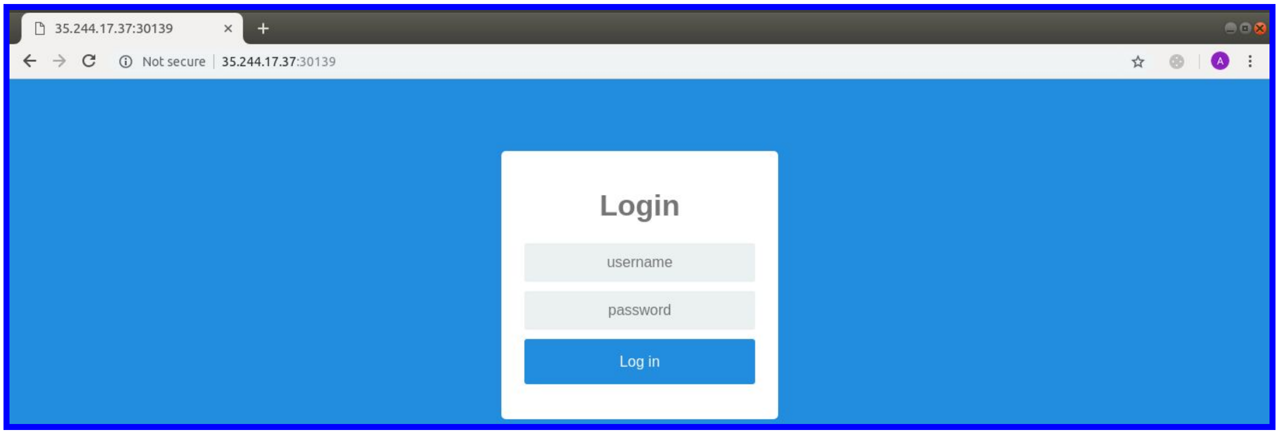
```
root@ip-172-31-40-214:/home/devops/application# kubectl get nodes -o wide
NAME                                 STATUS   ROLES    AGE    VERSION          INTERNAL-IP   EXTERNAL-IP    OS-IMAGE                              KERN
EL-VERSION   CONTAINER-RUNTIME
gke-demo-default-pool-289f281e-lpk7  Ready    <none>   155m   v1.12.6-gke.10   10.160.0.9    35.244.57.29   Container-Optimized OS from Google    4.14
.91+         docker://17.3.2
gke-demo-pool-1-66f23b9e-7686        Ready    <none>   34m    v1.12.6-gke.10   10.160.0.10   35.244.17.37   Container-Optimized OS from Google    4.14
.91+         docker://17.3.2
root@ip-172-31-40-214:/home/devops/application# 
```

In this Example the Pubic-IP of NODE **gke-demo-pool-1-66f23b9e-7686** is **35.244.17.37** and NodePort is **30139**

http://<NODE-PUBLIC-IP>:NODEPORT

7. Access the application from the public IP of the NODE and the NodePort as shown below

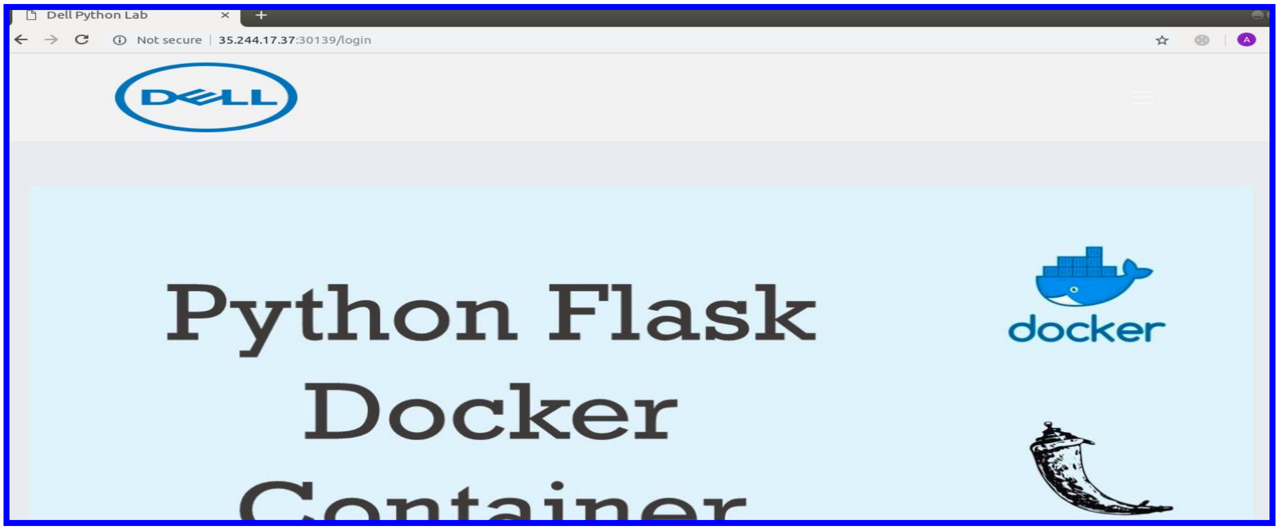
Example.



Login with the below credentials.

Username - admin

Password - password


Dell Python Lab
Not secure | 35.244.17.37:30139/login
DELL
Python Flask
Docker
Container
docker