Lab 3 Run Dockerized .NET 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/4ymZFiyp > dotnet_app-<your-name>.yaml
```

2. Edit the dotnet_app-<your-name>.yaml

```
# vim dotnet_app-<your-name>.yaml
```

Update the image: **lovescloud/docker-dotnet:latest** with your Docker Hub image name that you uploaded to docker hub in Docker Lab Pushing images to docker Hub for dotnet. Replace <your-name> with your name in the above script.

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 .NET application on your Kubernetes Cluster

```
# kubectl apply -f dotnet_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
dotnet-app-albert-7c7cd8f97c-8wrzx 1/1 Running 0 40s 10.48.1.4 gke-demo-pool-1-66f23b9e-7686 <none>
dotnet-app-albert-7c7cd8f97c-fftmp 1/1 Running 0 40s 10.48.1.3 gke-demo-pool-1-66f23b9e-7686 <none>
root@ip-172-31-40-214:/home/devops/application#
```

In this example the app has been deployed to the NODE gke-demo-default-pool-289f281e-lpk7

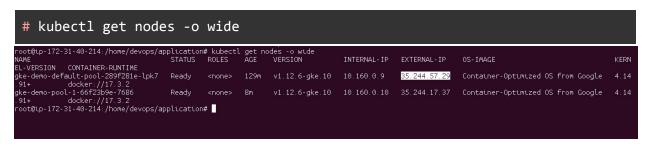
5. Check the **NODEPORT** on which your dotnet app has been exposed.

```
# 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
                                                                            104m
                    NodePort
                                10.51.254.52
                                                             80:30949/TCP
dotnet-app-albert
                                               <none>
                                                                            845
                    ClusterIP
                               10.51.240.1
                                                             443/TCP
                                                                            124m
kubernetes
                                               <none>
root@ip-172-31-40-214:/home/devops/application#
```

Example

In this example the albert dotnet application has been exposed on port **30949** as shown above.

6. Check the public IP of the NODE (**gke-demo-default-pool-289f281e-lpk7**) to access the dotnet application web page from the NODE Public IP address and Node Port on which it is exposed at.



The public IP for the NODE **gke-demo-default-pool-289f281e-lpk7** is **35.244.57.29** and the NodePort on which the app has been exposed is **30949** as obtained in step 5.

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

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

http://35.244.57.29:30949/

