SetUp Kubernetes Cluster on AWS with Kops

1. Generate SSH Key
```
ssh-keygen -f .ssh/id_rsa
```

2. Rename kops-linux-amd64 to kops for user easy.

```
sudo mv /usr/local/bin/kops-linux-amd64 /usr/local/bin/kops
```

******* Create Cluster ********

Command if you are using your Domain Name:
```
kops create cluster --yes --state=<s3://<Define S3 Bucket Name>> --
zones=<One or more Zones> --node-count=<Number of Nodes> --node-
size=<Define Machine Size> --master-size=<Master Node Size>
--name=<Define DNS Name>
```

Like :
```
kops create cluster --yes --state=s3://kops-storage-b345987 --
zones=ap-south-1a,ap-southeast-1b,ap-southeast-2c --node-count=2 --
node-size=t2.micro --master-size=t2.micro --name=test.easybix.com
```

For Non DNS Base Cluster, work with .k8s.local
```
kops create cluster --yes --state=s3://kops-storage-b345987 --
zones=ap-south-1a,ap-southeast-1b,ap-southeast-2c --node-count=2 --
node-size=t2.micro --master-size=t2.micro --name=test.k8s.local
```

4. Verify Node Status
```
kubectl get node
```

5. Validate Cluster
```
kops validate cluster
```

6. Let’s create a Kubernetes Deployment using an existing image named echoserver, which is a simple HTTP server and expose it on port 8080 using --port.
```
kubectl run hello-minikube --image=k8s.gcr.io/echoserver:1.10 --
port=8080
```

7. In order to access the hello-minikube service, we must first expose the deployment to an external IP via the command:
```
kubectl expose deployment hello-minikube --type=NodePort
```

8. Check if the service was exposed
```
kubectl get services
```

9. Modify Security Group Of Nodes to access the Service

10. Delete Kubernetes Cluster form AWS
```
kops delete cluster --name ${NAME} --yes
kops delete cluster --name test.k8s.local --yes
```