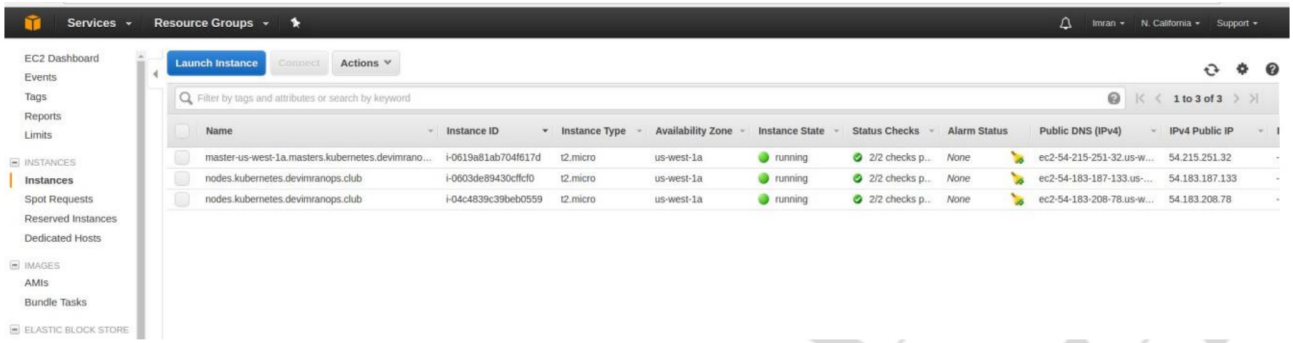
```
Must specify --yes to apply changes

Cluster configuration has been created.

Suggestions:
 * list clusters with: kops get cluster
 * edit this cluster with: kops edit cluster kubernetes.devimranops.club
 * edit your node instance group: kops edit ig --name=kubernetes.devimranops.club nodes
 * edit your master instance group: kops edit ig --name=kubernetes.devimranops.club master-us-west-1a

Finally configure your cluster with: kops update cluster kubernetes.devimranops.club --yes

root@DevImranOps:~#
```

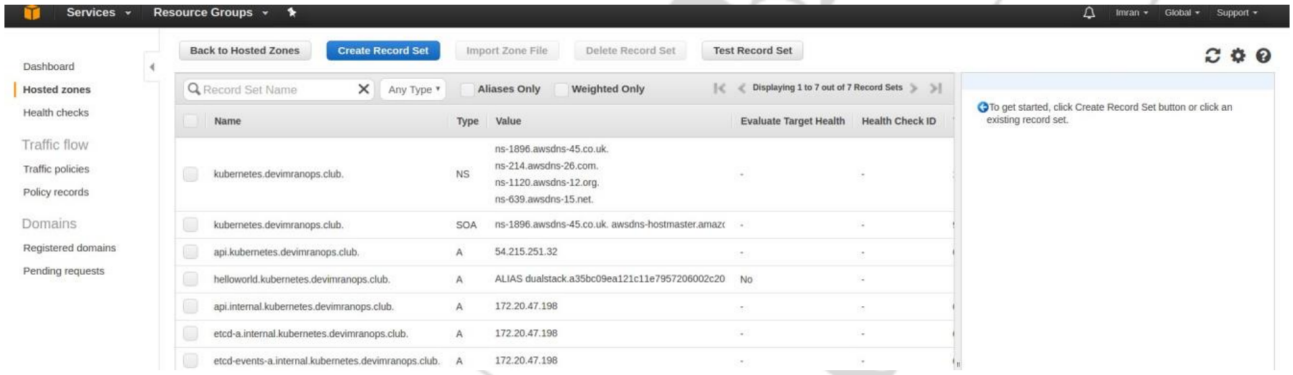
- Kops update cluster to create the cluster.

```
File Edit View Search Terminal Help
A s3 bucket is required to store cluster state information.
root@DevImranOps:~# kops update cluster kubernetes.devimranops.club --yes --state=s3://kops-state-86
I0330 18:12:19.022553    2871 executor.go:91] Tasks: 0 done / 56 total; 27 can run
I0330 18:12:20.430752    2871 vfs_castore.go:422] Issuing new certificate: "kubecfg"
I0330 18:12:20.590799    2871 vfs_castore.go:422] Issuing new certificate: "master"
I0330 18:12:20.771600    2871 vfs_castore.go:422] Issuing new certificate: "kubelet"
I0330 18:12:26.910766    2871 executor.go:91] Tasks: 27 done / 56 total; 12 can run
I0330 18:12:29.447253    2871 executor.go:91] Tasks: 39 done / 56 total; 15 can run
I0330 18:12:33.262695    2871 launchconfiguration.go:310] waiting for IAM instance profile "masters.kubernet
es.devimranops.club" to be ready
I0330 18:12:33.263549    2871 launchconfiguration.go:310] waiting for IAM instance profile "nodes.kubernetes
.devimranops.club" to be ready
I0330 18:12:45.146373    2871 executor.go:91] Tasks: 54 done / 56 total; 2 can run
I0330 18:12:46.170759    2871 executor.go:91] Tasks: 56 done / 56 total; 0 can run
I0330 18:12:46.170816    2871 dns.go:141] Pre-creating DNS records
I0330 18:12:50.787335    2871 update_cluster.go:208] Exporting kubecfg for cluster
Kops has set your kubectl context to kubernetes.devimranops.club

Cluster is starting.  It should be ready in a few minutes.

Suggestions:
 * validate cluster: kops validate cluster
 * list nodes: kubectl get nodes --show-labels
 * ssh to the master: ssh -i ~/.ssh/id_rsa admin@api.kubernetes.devimranops.club
 * read about installing addons: https://github.com/kubernetes/kops/blob/master/docs/addons.md

root@DevImranOps:~#
root@DevImranOps:~#
root@DevImranOps:~#
root@DevImranOps:~#
```

- Cluster setup would have been initiated, verify it by from AWS.



- Verify Route 53 subdomains.



- Validate cluster with kops command.

```
root@DevImranOps:~# kops validate cluster --state=s3://kops-state-86
Using cluster from kubectl context: kubernetes.devimranops.club

Validating cluster kubernetes.devimranops.club

INSTANCE GROUPS
NAME                    ROLE    MACHINETYPE     MIN     MAX     SUBNETS
master-us-west-1a       Master  t2.micro        1       1       us-west-1a
nodes                   Node    t2.micro        2       2       us-west-1a

NODE STATUS
NAME                                            ROLE    READY
ip-172-20-46-167.us-west-1.compute.internal     node    True
ip-172-20-47-198.us-west-1.compute.internal     master  True
ip-172-20-53-5.us-west-1.compute.internal       node    True

Your cluster kubernetes.devimranops.club is ready
root@DevImranOps:~#
```

- Kops creates ~/.kube/config file where it writes all the config details for KUBECTL.

```
File  Edit  View  Search  Terminal  Help

root@DevImranOps:~# cat ~/.kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUMxekNDQWIrZ0F3SUJBZ0lRRUZFU0tSQThyUk1zaFpGbElISDdLekFOQmdrcWhraUc5d
zBCQVFzRkFEQVYKTVJNd0VRWURWUVFERXdwcmRXSmxjbTVsZEdWek1CNFhEVEUzTURNeU9ERTRNVEl5TUZvWERUSTNNRE15T0RFNApNVEl5TUZvd0ZURVRNQkVHQTFVRUF4TUthM1ZpWl
hKdVpYUmxjekNDQVNJd0RRWUpLb1pJaHZjTkFRRUJCUUFECmdnRVBBRENDQVFvQ2dnRUJBTHIvWW93ZUtCbHFMRGpXRVhMak5GTmF5T2tUTFUyekJIWE5GczVhcldGb3dKRzEKcXpvS3R
YMG1Qd3c3aW5GdkFuQk96OGtGTCszVjIxaFFXeG1pWGpZcGptWDNpQTcvVHhwVGJTQ3gzdm9JSEY1SgprSnk3TThCSDRyMVdhaDYwVURQRCtQV2VWak9xcmZUUnFBMVJBMldiUzBTZTdT
VS9MVWJpUitvbmJUalRVNzRzCk5ZOUhvUVJKd3BJa0VTeHhQUG5tL25iSTNzbmVzQ3A0ZGNqdkZBUzEzV0xXNkozSVZacGlZNXV6REExsYUZyMDQKTlJEM2hjZjQ1eUllWEdwOGdmQVhPS
2Y3cVpqZHdjRU5wV3NBNnVNUElLZW1UNkkrR1RkZTNnWk8zRG9ZenhKdgp0TUladzFOaVJDc0lYZzAwdXRwelp1WnY2M2lsSjBEOXdpUGhEM2tDQXdFQUFhTWpNQ0V3RGdZRFZSMFBBUU
gvCkJBUURBZ0VHTUE4R0ExVWRFd0VCL3dRRk1BTUJBZjh3RFFZSktvWklodmNOQVFFTEJRQURnZ0VCQUR6Yk41MjUKd3NDSy9HMHh1MDgwMUYKwYzduVGJJSW5VSURiNTRvdXZnOHZkdEp
UWUthcEhDb1hQSHMwZm44MHk3SUpXd3dZWQpveTZSMndWN0pNandlN2cvbVFweHAvSmdkVlk0UzJPRmpjbTZtTzVtV1N1WFpZME9sU1RtSHY4OXdLM25uUmR6CmJwdHdTUDFRaGI3ZXNJ
bW5oS3FDYU54N1F4b3JoRVpLN1VqK0xOaEkxQlRLdGM4WWpNdGRoL1FSRTZKVVpScGcKa21hbFJzMXlWUEhuTXpoZ3RKMHdIajJTcnlBajJzSHJYcmdtQjVsTkNjZkFQbUEwemRjOFgvb
00xZjNoOGhZcApDMWQ1RUN6K01MQyt2VkxZeGdjb1FSRlFUWDVmU1JTeVN3d1c3Sk01NllFT0pVM0tPZFk1bEIvTjk0cUkraTFJCm8wZUltR3ZYdDlZa01iRT0KLS0tLS1FTkQgQ0VSVE
lGSUNBVEUtLS0tLQo=
    server: https://api.kubernetes.devimranops.club
  name: kubernetes.devimranops.club
contexts:
- context:
    cluster: kubernetes.devimranops.club
    user: kubernetes.devimranops.club
  name: kubernetes.devimranops.club
current-context: kubernetes.devimranops.club
kind: Config
preferences: {}
users:
- name: kubernetes.devimranops.club
  user:
    client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUM0akNDQWNxZ0F3SUJBZ0lNRkxDN2FlaXFZVnovYi9MNU1BMEdDU3FHU0liM0RRRUJDd1VB
TUJVeEV6QVIKQmdOVkJBTVRDbXQxWW1WeWJtVjBaWE13SGhjTk1UY3dNekk0TVRneE1qSXpXaGNOTWpjd016STRNVGd4TWpJegpXakFTTVJBd0RnWURWUVFERXdkcmRXSmxZMlpuTUlJQ
klqQU5CZ2txaGtpRzl3MEJBUUVGQUFPQ0FROEFNSUlCCkNnS0NBUUVBb0M2N2x0cEZkNlNPNjZlNjY2elVrSHpQWWtOS3dWTHVXU3ZoT0tpV0Q4VENPNEFMNTJzY2x4bnAKU3Vzc1N4V3
VCRjIrR1IranF4NWRDR09BQ3FvT0VaemdpQTJxWHpxRWh5UUNaL1FLZ3NzYlkwdVd2cC83dzVHcAoxWWY1U2pWTGJOWk4wdVNsTEUwbU1EanV3UDBhUC92bjZKY3hhMGMyb3d5RTZRSkJ
kTGdmK3VCNlBZRXJJYVE5CmpoVnFldktHc0RJVUxNdUZtLzBhRGxvemxuR05xdFFzSFg3R3F0T0VvVFBHaDV3NDdZZWJHaU9SK0VJUDU0L1EKcVhUYmhkUU5MeG5oQ1RDRUlvVUorSGQ3
QXhQZWpXN1hXVnJCbGlqazBJRjRxa2NZYzU2UEx4WnUybG5ibDF2RQpVeWxkY0Q2cS9FVjFuZS9sS0NFSmxvbHJvbFZ1eHdJREFRQUJvelV3TXpBT0JnTlZIUThCQWY4RUJBTUNCNEF3C
kV3WURWUjBsQkF3d0NnWUlLd1lCQlFVSEF3SXdEQVlEVlIwVEFRSC9CQUl3QURBTkJna3Foa2lHOXcwQkFRc0YKQUFPQ0FRRUFBQ01KaHdzdER3a3oyTEcrVlJGUTk1RnArbmNCZnJOWj
l1djhtZjlPZ1M2eVR4K1dwM0ZNOTUrYQpDekt2SWp3TzJMczJUUkFuWlhNN2NiNXNiVFNFS0V4M09XNU4vcVJISXF1SWtWcTF0d3JwNWM5YXFOb0JWVEE0CnNGMk5OaHpQVmkrZXp4SEx
ZUnBSWGRqQW5oOTZHc0hNcS9PQ05DaWk0V1p6SFZLdmsxZ2EwM0RBajc1enNXOHAKczFudnBwY2lQL3JoTHFzcnJCOHFlVDdTUmVjek9sNWxncWVCRE9QMzk1UUlWMDBHTnhWUDNRajVu
dFpnWDZwaQpRdVJrUFV5THFTN3RCakVkdmh0MENsa2poeVFFNXJERm9VYXdENGlXRU53MU5ldit1cVNTbXlHdCtWVUZVb2dqClhpTmFjVU1hUlZVNnJkREhLTW1HUmNOMlM1K2FHQT09C
i0tLS0tRU5EIENFUlRJRklDQVRFLS0tLS0K
    client-key-data: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFb3dJQkFBS0NBUUVBb0M2N2x0cEZkNlNPNjZlNjY2elVrSHpQWWtOS3dWTHVXU3ZoT0tpV0Q4
```

- We can now start using kubectl command to interact with our kubernetes cluster.

Check the all the nodes in cluster.

```
root@DevImranOps:~# kubectl get nodes
NAME                                        STATUS         AGE
ip-172-20-46-167.us-west-1.compute.internal   Ready          14m
ip-172-20-47-198.us-west-1.compute.internal   Ready,master   15m
ip-172-20-53-5.us-west-1.compute.internal     Ready          14m
root@DevImranOps:~#
```

- Let’s run a container on this cluster.

```
File  Edit  View  Search  Terminal  Help
root@DevImranOps:~# kubectl run hello-minikube --image=gcr.io/google_containers/echoserver:1.4 --port=8080
deployment "hello-minikube" created
root@DevImranOps:~# kubectl expose deployment hello-minikube --type=NodePort
service "hello-minikube" exposed
root@DevImranOps:~# kubectl get service
NAME             CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
hello-minikube   100.64.74.8   <nodes>       8080:31496/TCP   14s
kubernetes       100.64.0.1    <none>        443/TCP          22m
root@DevImranOps:~#
```

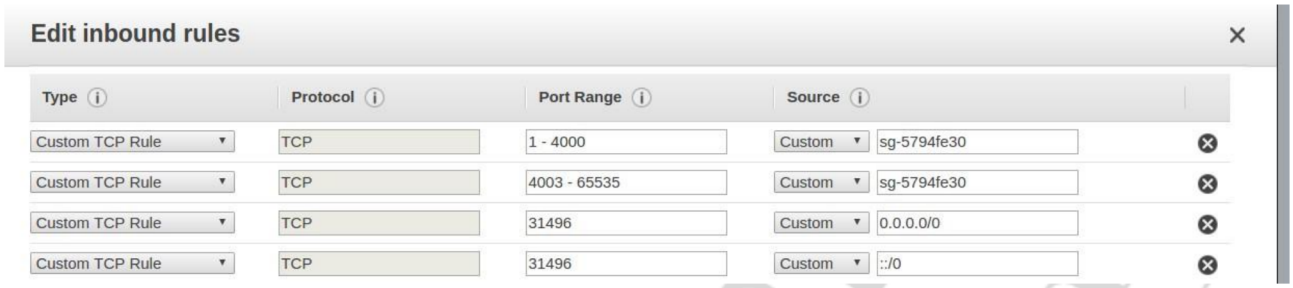
- Deployment and service created, service is exposed on port 31496.

We must allow port 31496 in Security group of Master node to access this service.

- Go to AWS Ec2 => Select Master node => Security groups =>masters.kubernetes.

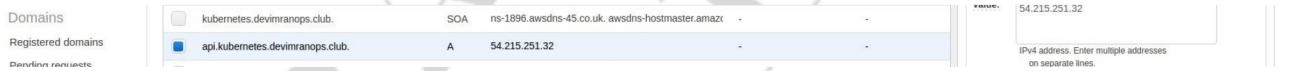
Inbound => Edit.

Add the port number and source from anywhere/MyIP.



- Get the master node Public IP and access it from browser on the exposed port.

- Instead of Public IP of master node we can also use names assigned by Route53.



- If you are not using the cluster you can delete it.

```
root@DevImranOps:~# kops delete cluster kubernetes.devimranops.club --state=s3://kops-state-86 --yes
```

10. Containers & Images

In this exercise, we will build a docker image for nodejs app.

Will use dockerhub as registry and will run that image on our kubernetes cluster.

Check the docker-demo directory. Verify three files shown below.

- Dockerfile

```
# cat Dockerfile
FROM node:4.6
WORKDIR /app
ADD . /app
RUN npm install
EXPOSE 3000
CMD npm start
```

- index.js

```
# cat index.js
var express = require('express');
var app = express();

app.get('/', function (req, res) {
  res.send('Hello World!');
});

var server = app.listen(3000, function () {
  var host = server.address().address;
  var port = server.address().port;

  console.log('Example app listening at http://%s:%s', host, port);
});
```

- package.json

```
# cat package.json
{
  "name": "myapp",
  "version": "0.0.1",
  "private": true,
```

```
  "scripts": {
    "start": "node index.js"
  },
  "engines": {
    "node": "^4.6.1"
  },
  "dependencies": {
    "express": "^4.14.0",
    "mysql": "^2.10.2"
  }
}
```

- Install docker.

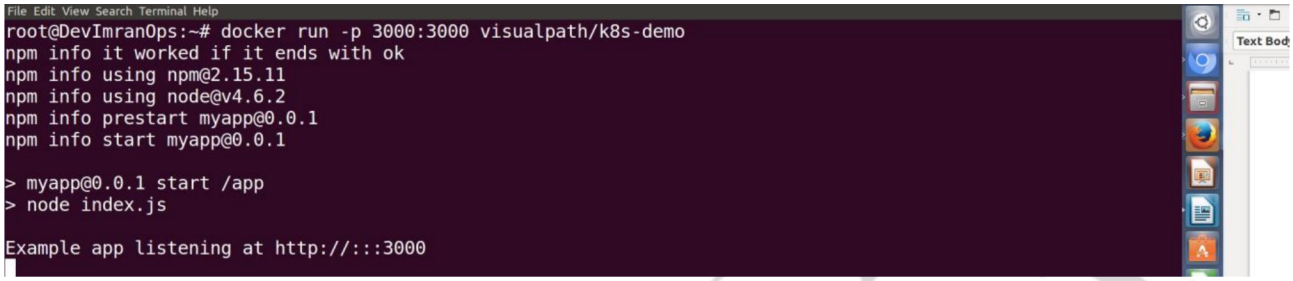
```
root@DevImranOps:~/kube/docker-demo# apt-get install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker.io is already the newest version (1.12.6-0ubuntu1~16.04.1).
```

- - Build Image.

Images name should match with your dockerhub account/reponame. For example my account name in dockerhub is “visualpath”, so my image name is visualpath/k8s-demo.

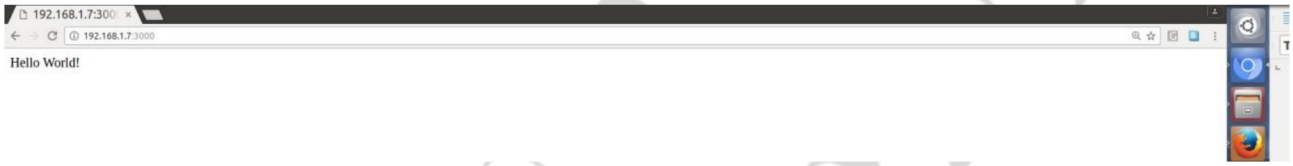
```
File  Edit  View  Search  Terminal  Help

root@DevImranOps:~/kube# cd docker-demo/
root@DevImranOps:~/kube/docker-demo# cat Dockerfile
FROM node:4.6
WORKDIR /app
ADD . /app
RUN npm install
EXPOSE 3000
CMD npm start
root@DevImranOps:~/kube/docker-demo# docker build -t visualpath/k8s-demo  .
Sending build context to Docker daemon 90.11 kB
Step 1 : FROM node:4.6
 ---> e834398209c1
Step 2 : WORKDIR /app
 ---> Using cache
 ---> d808ac4780ac
Step 3 : ADD . /app
 ---> Using cache
 ---> 304268d58a39
Step 4 : RUN npm install
 ---> Using cache
 ---> 8573156bdb29
Step 5 : EXPOSE 3000
 ---> Using cache
 ---> ac8a61ac3689
Step 6 : CMD npm start
 ---> Using cache
 ---> 03be3f739a76
Successfully built 03be3f739a76
root@DevImranOps:~/kube/docker-demo#
```

- Run image locally and test.

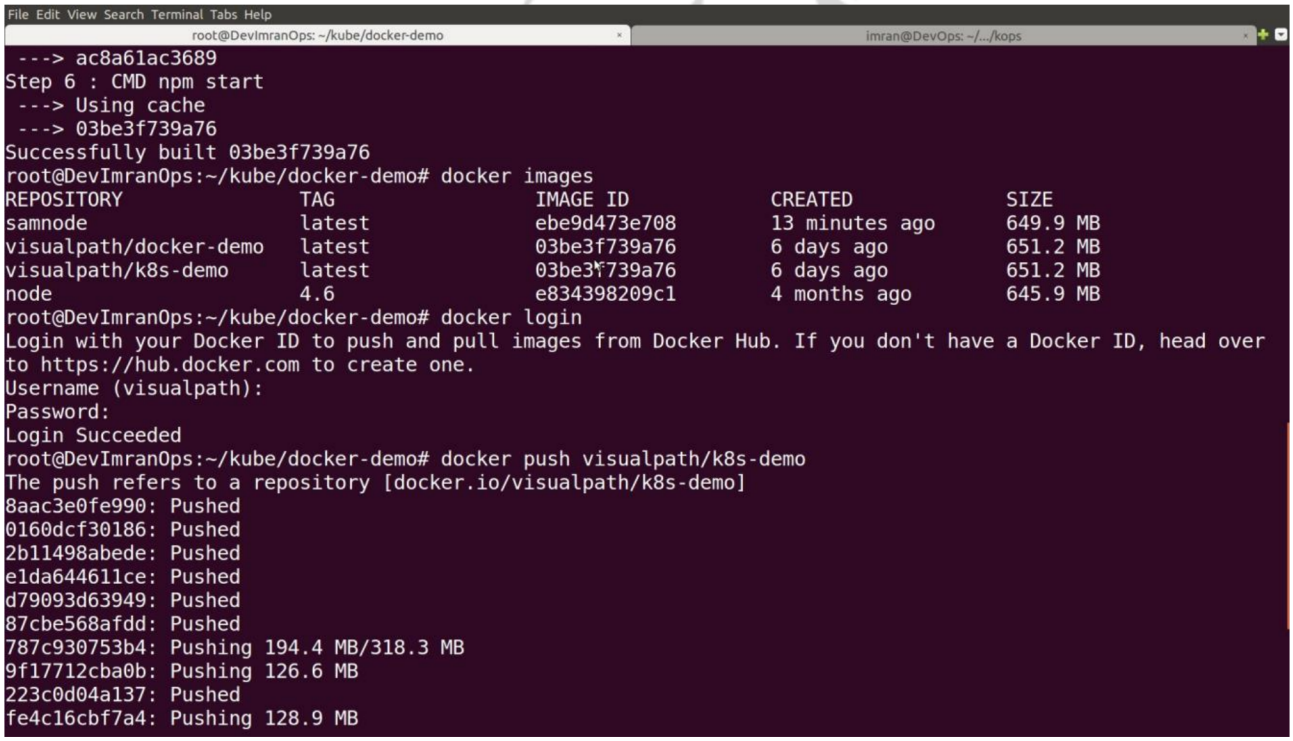


- Test it from browser

Enter IP address of your vm(kops vm) where your container is running and port number.



- Push Image to Dockerhub



11. First App on Kubernetes.

- Let’s run our newly built application on the Kubernetes cluster
- Before we can launch a container, we need to create a POD definition.
  - A Pod describes an application running on Kubernetes.
  - A can contain one or more tightly coupled containers, that make up the app.
    - Those apps can easily communicate with each other using their local port numbers.

  Our app only has one container.

- Read pod definition.

```
root@DevImranOps:~/kube# cat first-app/helloworld.yml
apiVersion: v1
kind: Pod
metadata:
  name: nodehelloworld.example.com
  labels:
    app: helloworld
spec:
  containers:
  - name: k8s-demo
    image: visualpath/k8s-demo
    ports:
    - name: nodejs-port
      containerPort: 3000
root@DevImranOps:~/kube#
```

- Use kubectl to create pod on cluster.

```
root@DevImranOps:~/kube# kubectl create -f first-app/helloworld.yml
pod "nodehelloworld.example.com" created
root@DevImranOps:~/kube# kubectl get pod
NAME                        READY     STATUS    RESTARTS   AGE
nodehelloworld.example.com   1/1       Running   0          3m
```

```
root@DevImranOps:~/kube# kubectl describe pod nodehelloworld.example.com
Name:           nodehelloworld.example.com
Namespace:      default
Node:           ip-172-20-34-251.us-west-1.compute.internal/172.20.34.251
Start Time:     Sat, 01 Apr 2017 19:18:20 +0000
Labels:         app=helloworld
Status:         Running
IP:             100.96.2.2
Controllers:    <none>
```

- Useful Kubectl commands.

Command	Description
kubectl get pod	Get information about all running pods
kubectl describe pod <pod>	Describe one pod
kubectl expose pod <pod> --port=444 --name=frontend	Expose the port of a pod (creates a new service)
kubectl port-forward <pod> 8080	Port forward the exposed pod port to your local machine
kubectl attach <podname> -i	Attach to the pod
kubectl exec <pod> -- command	Execute a command on the pod
kubectl label pods <pod> mylabel=awesome	Add a new label to a pod
kubectl run -i --tty busybox --image=busybox --restart=Never -- sh	Run a shell in a pod - very useful for debugging

- Executing commands on container.

```
File  Edit  View  Search  Terminal  Help
root@DevImranOps:~/kube# kubectl exec nodehelloworld.example.com -- ls /app
Dockerfile
docker-compose.yml
index-db.js
index.js
misc
node_modules
package.json
root@DevImranOps:~/kube# kubectl exec nodehelloworld.example.com -- touch  /app/testfile.txt
root@DevImranOps:~/kube# kubectl exec nodehelloworld.example.com -- ls /app
Dockerfile
docker-compose.yml
index-db.js
index.js
misc
node_modules
package.json
testfile.txt
root@DevImranOps:~/kube#
```

12. Services

- Creating service for our app.

```
File Edit View Search Terminal Help
root@DevImranOps:~/kube# cat first-app/helloworld-nodeport-service.yml
apiVersion: v1
kind: Service
metadata:
  name: helloworld-service
spec:
  ports:
  - port: 31001
    nodePort: 31001
    targetPort: nodejs-port
    protocol: TCP
  selector:
    app: helloworld
  type: NodePort
root@DevImranOps:~/kube# kubectl create -f first-app/helloworld-nodeport-service.yml
service "helloworld-service" created
root@DevImranOps:~/kube# kubectl get service
NAME                CLUSTER-IP     EXTERNAL-IP   PORT(S)           AGE
helloworld-service  100.68.5.83    <nodes>       31001:31001/TCP   5s
kubernetes          100.64.0.1     <none>        443/TCP           35m
root@DevImranOps:~/kube#
```

- Accessing service from outside world.
  - Open MasterNode Security group from AWS.
  - Allow port 31001 from MyIP/Anywhere. **Port could be different**, please check your service port(kubectl get service).
  - Open browser enter your master node public IP and 31001 port.