< JAVA APP DEPLOYMENT ON KUBERNETES CLUSTER >

IMPLEMENTATION PLAN

VERSION 1.0 | 20/06/2022

Purpose

Java app deployment on Kubernetes cluster.

About the Project

- *Containerization of Application*
- *Test it local & Cloud*
- *Local & Cloud*
- *Multi-Tier Web Application Stack*
- *Host for production*

Requirement

- *High Availability*
- *Fault Tolerance*
- *Easily Scalable*
- *Platform Independent*
- *Portable & Flexible*

Tools

- *Minikube – LOCAL, for testing and learning*
- *Kubeadm – Multi Node Cluster Local/Cloud*
- *KOPS – Cloud AWS & GCP*
- *AWS EKS – AWS Cloud Kubernetes Services*

Objective

- *Setup Kubernetes*
- *Test Kubernetes Cluster*
- *Containerized apps (eereeda)*
- *Create EBS volume for DB Pod*
- *LABEL Node with zones names*

1. Minikube for Test Env

Installation Guide: https://kubernetes.io/docs/tasks/tools/

First install Docker:

https://github.com/merajafnan/DevOps_Projects/blob/fd154734f80b307ae5fb5fb750329df17bf68188/Docker-Control-machine.sh

```
#!/bin/bash
#Download the latest release with the command
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

#Validate the binary
curl -LO "https://dl.k8s.io/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"
echo "$(cat kubectl.sha256)  kubectl" | sha256sum --check
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

#Install & verify kubectl
kubectl version --client
kubectl version --client --output=yaml

#Minikube Installation
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-
linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube

minikube start
```

```
ubuntu@ip-172-31-16-30:~$ minikube start
😄  minikube v1.25.2 on Ubuntu 18.04 (xen/amd64)
✨  Automatically selected the docker driver. Other choices: none, ssh
👍  Starting control plane node minikube in cluster minikube
🚜  Pulling base image ...
💾  Downloading Kubernetes v1.23.3 preload ...
    > preloaded-images-k8s-v17-v1...: 505.68 MiB / 505.68 MiB  100.00% 56.48 Mi
    > gcr.io/k8s-minikube/kicbase: 379.06 MiB / 379.06 MiB  100.00% 39.67 MiB p
🔥  Creating docker container (CPUs=2, Memory=2200MB) ...
🐳  Preparing Kubernetes v1.23.3 on Docker 20.10.12 ...
    ▪ kubelet.housekeeping-interval=5m
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
ubuntu@ip-172-31-16-30:~$
```

```
ubuntu@ip-172-31-16-30:~$ kubectl get nodes
NAME       STATUS   ROLES                  AGE     VERSION
minikube   Ready    control-plane,master   3m32s   v1.23.3
ubuntu@ip-172-31-16-30:~$
```

```
ubuntu@ip-172-31-16-30:~$ cat .kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/ubuntu/.minikube/ca.crt
    extensions:
    - extension:
        last-update: Mon, 20 Jun 2022 05:46:02 UTC
        provider: minikube.sigs.k8s.io
        version: v1.25.2
      name: cluster_info
    server: https://192.168.49.2:8443
  name: minikube
contexts:
- context:
    cluster: minikube
    extensions:
    - extension:
        last-update: Mon, 20 Jun 2022 05:46:02 UTC
        provider: minikube.sigs.k8s.io
        version: v1.25.2
      name: context_info
    namespace: default
    user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
- name: minikube
  user:
    client-certificate: /home/ubuntu/.minikube/profiles/minikube/client.crt
    client-key: /home/ubuntu/.minikube/profiles/minikube/client.key
```

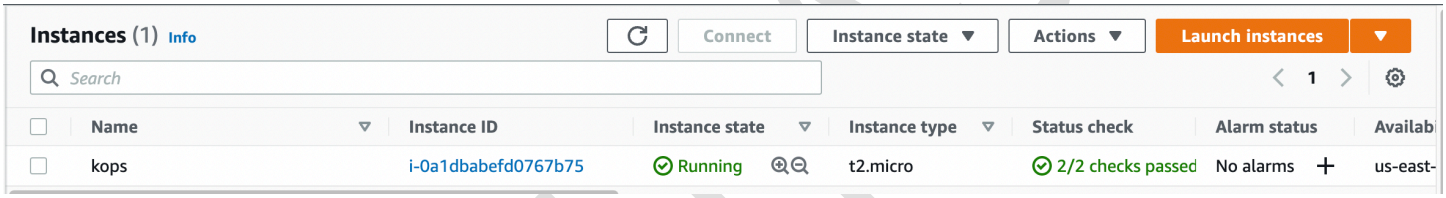
```
ubuntu@ip-172-31-16-30:~$ kubectl create deployment hello-minikube --image=k8s.gcr.io/echoserver:1.4
deployment.apps/hello-minikube created
ubuntu@ip-172-31-16-30:~$ kubectl expose deployment hello-minikube --type=NodePort --port=8080
service/hello-minikube exposed
ubuntu@ip-172-31-16-30:~$ kubectl get pod
NAME                              READY   STATUS    RESTARTS   AGE
hello-minikube-7bc9d7884c-96pnt   1/1     Running   0          49s
ubuntu@ip-172-31-16-30:~$ kubectl get deploy
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
hello-minikube   1/1     1            1           57s
ubuntu@ip-172-31-16-30:~$ minikube service hello-minikube --URL
Error: unknown flag: --URL
See 'minikube service --help' for usage.
ubuntu@ip-172-31-16-30:~$ minikube service hello-minikube --url
http://192.168.49.2:31132


ubuntu@ip-172-31-16-30:~$ kubectl get svc
NAME             TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
hello-minikube   NodePort    10.105.139.102   <none>        8080:31132/TCP   44m
kubernetes       ClusterIP   10.96.0.1        <none>        443/TCP          52m
ubuntu@ip-172-31-16-30:~$ kubectl delete svc hello-minikube
service "hello-minikube" deleted
ubuntu@ip-172-31-16-30:~$ kubectl get deployment
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
hello-minikube   1/1     1            1           45m
ubuntu@ip-172-31-16-30:~$ kubectl delete deployment hello-minikube
deployment.apps "hello-minikube" deleted
ubuntu@ip-172-31-16-30:~$ mikube stop
mikube: command not found
ubuntu@ip-172-31-16-30:~$ minikube stop
✋  Stopping node "minikube"  ...
🛑  Powering off "minikube" via SSH ...
🛑  1 node stopped.
```

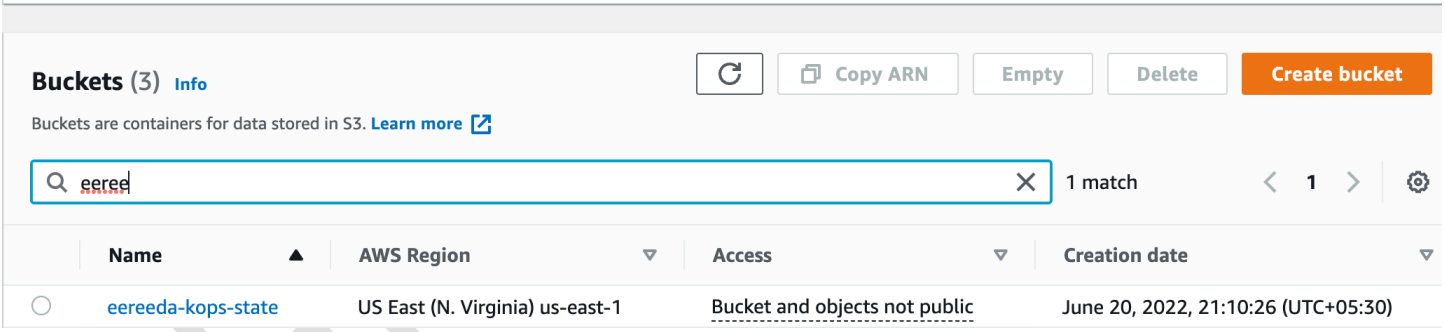
2. Kops for cluster on AWS

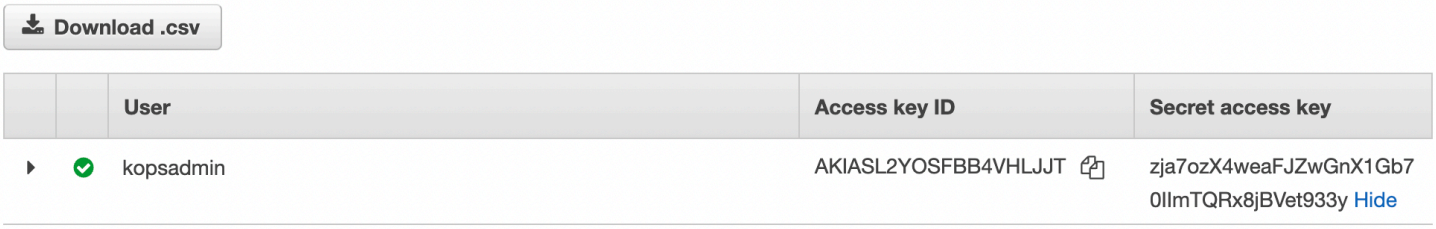
- Domain for Kubernetes DNS records
  - Eg: eereeda.in from GoDady
- Create linux VM and setup
  - Kops,kubectl,ssh key,awscli
- Login to AWS account and setup
  - S3 bucket,IAM user for AWSCLI,Route53 Hosted Zones

Setup

EC2 Instance > Ubuntu 20 | t2 micro
S3 Bucket > eereeda-kops-state
IAM User > kopsadmin > Administrator Access
Route 53 > kube.eereeda.in
Godady > NS Server







Hosted zones (1)

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

View details | Edit | Delete | **Create hosted zone**

Filter hosted zones by property or value

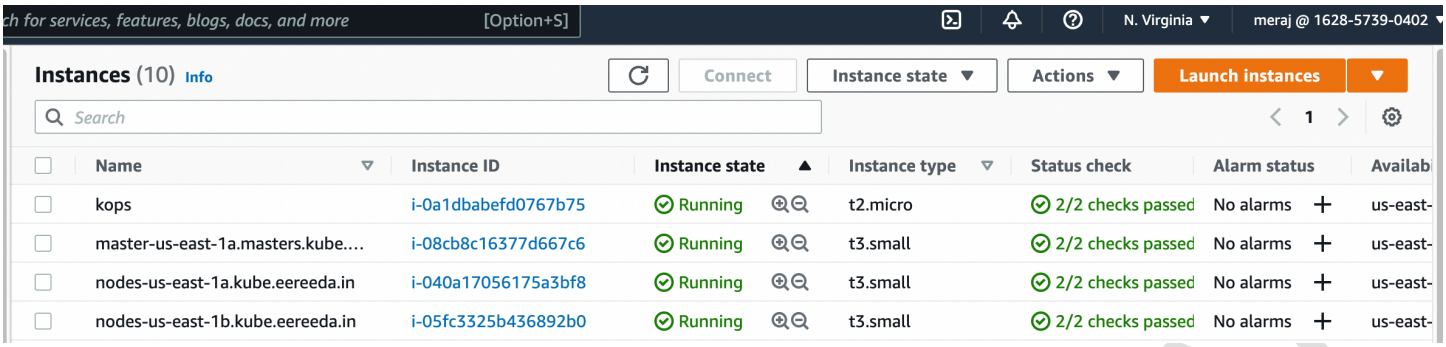
1

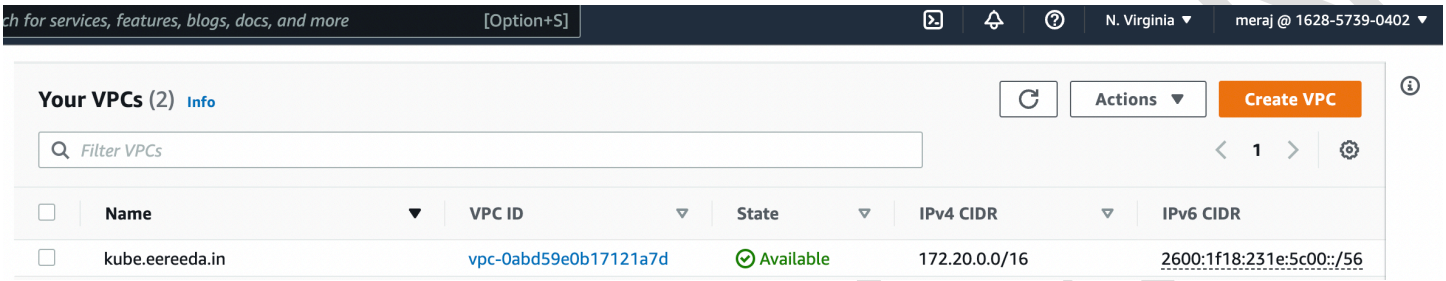
Domain name	Type	Created by	Record count	Description	Hosted zone ID
kube.eereeda.in	Public	Route 53	2	kubeeereeda	Z04006002NWFER...

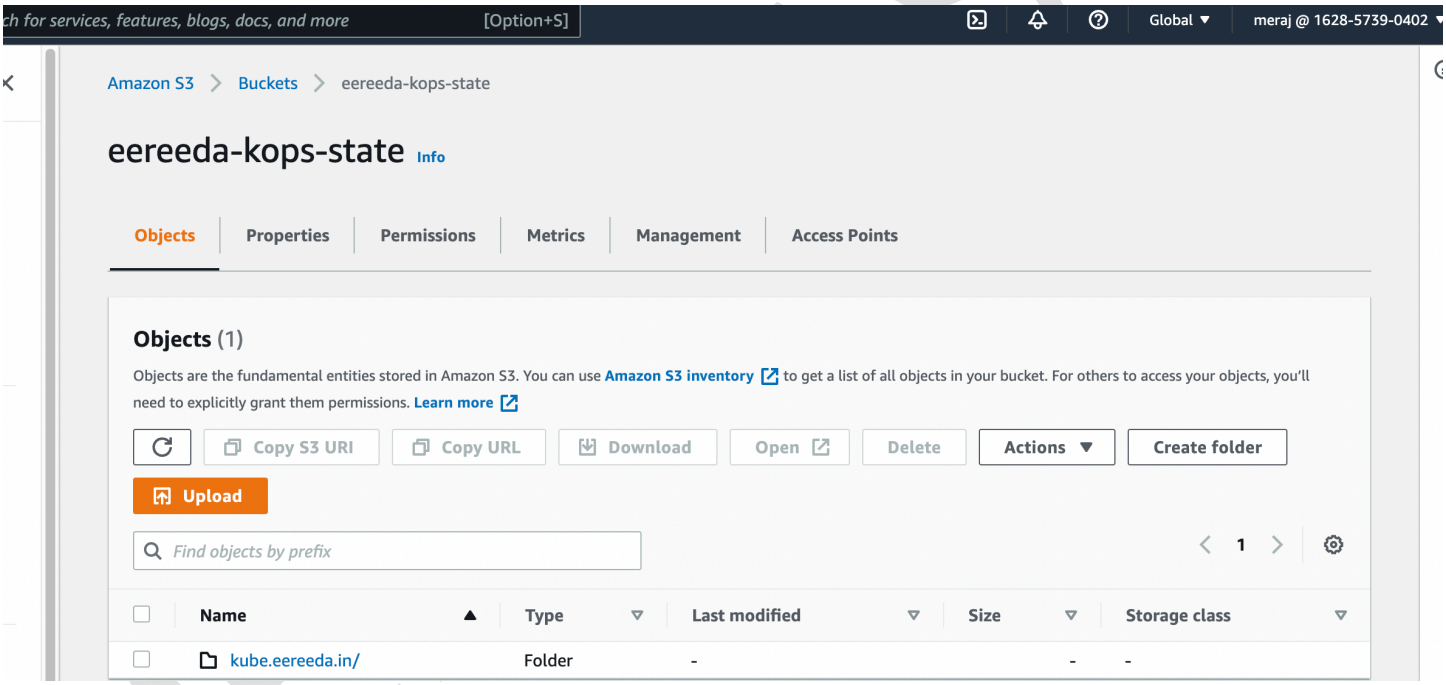
NS	kube	ns-1213.awsdns-23.org.	1 Hour	Delete	Edit
NS	kube	ns-1552.awsdns-02.co.uk.	1 Hour	Delete	Edit
NS	kube	ns-253.awsdns-31.com.	1 Hour	Delete	Edit
NS	kube	ns-641.awsdns-16.net.	1 Hour	Delete	Edit

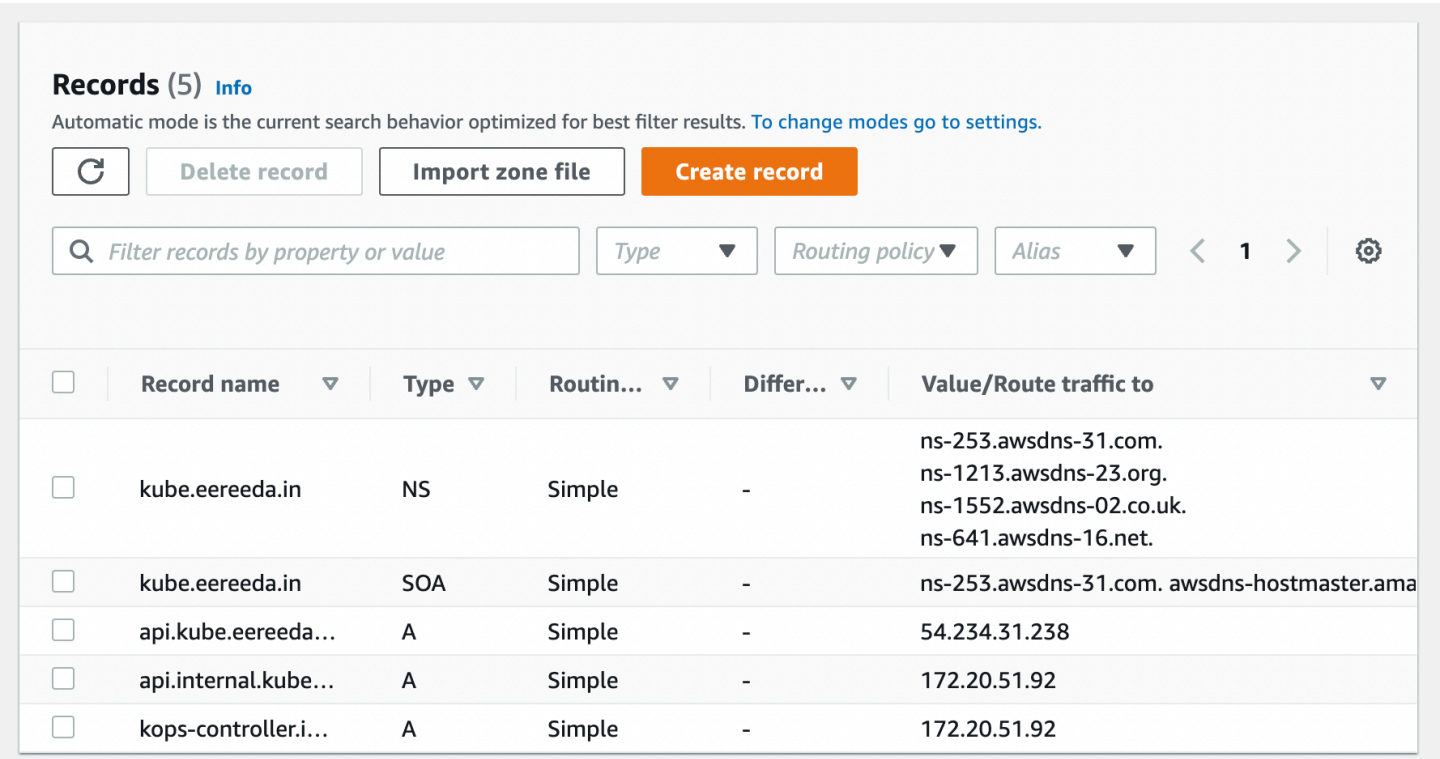
3. Cluster Configuration

```
# Create Key & Install awscli
ssh-keygen
sudo apt update && sudo apt install awscli -y
aws configure

# Install kubectl
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
chmod +x ./kubectl
sudo mv kubectl /usr/local/bin/
kubectl --help

#Install kops
curl -LO https://github.com/kubernetes/kops/releases/download/$(curl -s
https://api.github.com/repos/kubernetes/kops/releases/latest | grep
tag_name | cut -d '"' -f 4)/kops-linux-amd64
chmod +x ./kops-linux-amd64
sudo mv kops-linux-amd64 /usr/local/bin/kops

# check DNS server
nslookup -type=ns kube.eereeda.in

# Create configuration for Kubernetes Cluster in S3 Bucket
kops create cluster --name=kube.eereeda.in --state=s3://eereeda-kops-
state --zones=us-east-1a,us-east-1b --node-count=2 --node-size=t3.small
--master-size=t3.small --dns-zone=kube.eereeda.in

#Create Cluster
kops update cluster --name kube.eereeda.in --state=s3://eereeda-kops-
state --yes
kops export kubecfg --state=s3://eereeda-kops-state --admin

#Validate Cluster
kops validate cluster --state=s3://eereeda-kops-state

cat .kube/config

kubectl get nodes

kops delete cluster --name kube.eereeda.in --state=s3://eereeda-kops-
state -yes
```


ch for services, features, blogs, docs, and more [Option+S]
N. Virginia
meraj @ 1628-5739-0402
Instances (10) Info
Connect
Instance state
Actions
Launch instances
Search
1
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availab
kops
i-0a1dbabefd0767b75
Running
t2.micro
2/2 checks passed
No alarms
us-east-
master-us-east-1a.masters.kube....
i-08cb8c16377d667c6
Running
t3.small
2/2 checks passed
No alarms
us-east-
nodes-us-east-1a.kube.eereeda.in
i-040a17056175a3bf8
Running
t3.small
2/2 checks passed
No alarms
us-east-
nodes-us-east-1b.kube.eereeda.in
i-05fc3325b436892b0
Running
t3.small
2/2 checks passed
No alarms
us-east-


ch for services, features, blogs, docs, and more [Option+S]
N. Virginia
meraj @ 1628-5739-0402
Your VPCs (2) Info
Actions
Create VPC
Filter VPCs
1
Name
VPC ID
State
IPv4 CIDR
IPv6 CIDR
kube.eereeda.in
vpc-0abd59e0b17121a7d
Available
172.20.0.0/16
2600:1f18:231e:5c00::/56


ch for services, features, blogs, docs, and more [Option+S]
Global
meraj @ 1628-5739-0402
Amazon S3 > Buckets > eereeda-kops-state
eereeda-kops-state Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (1)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
kube.eereeda.in/
Folder
-
-
-




```
ubuntu@ip-172-31-85-74:~$ kops validate cluster --state=s3://eereeda-kops-state
Using cluster from kubectl context: kube.eereeda.in

Validating cluster kube.eereeda.in

INSTANCE GROUPS
NAME                    ROLE    MACHINETYPE     MIN     MAX     SUBNETS
master-us-east-1a       Master  t3.small        1       1       us-east-1a
nodes-us-east-1a        Node    t3.small        1       1       us-east-1a
nodes-us-east-1b        Node    t3.small        1       1       us-east-1b

NODE STATUS
NAME                            ROLE    READY
ip-172-20-51-92.ec2.internal    master  True
ip-172-20-58-40.ec2.internal    node    True
ip-172-20-87-109.ec2.internal   node    True

Your cluster kube.eereeda.in is ready
```

```
ubuntu@ip-172-31-85-74:~$ kubectl get nodes
NAME                            STATUS   ROLES                  AGE   VERSION
ip-172-20-51-92.ec2.internal    Ready    control-plane,master   11m   v1.23.7
ip-172-20-58-40.ec2.internal    Ready    node                   10m   v1.23.7
ip-172-20-87-109.ec2.internal   Ready    node                   10m   v1.23.7
```

4. Volume prerequisite for DB Production

Create volume for DB Pod so that it can store the MySQL data that is sorted in var/lib/myql/ebs/volume

$ aws ec2 create-volume --availability-zone=us-east-1a --size=3 --volume-type=gp2

```
ubuntu@ip-172-31-85-74:~$ aws ec2 create-volume --availability-zone=us-east-1a --size=3 --volume-type=gp2
{
    "AvailabilityZone": "us-east-1a",
    "CreateTime": "2022-06-20T17:38:30.000Z",
    "Encrypted": false,
    "Size": 3,
    "SnapshotId": "",
    "State": "creating",
    "VolumeId": "vol-0faca59697131a588",
    "Iops": 100,
    "Tags": [],
    "VolumeType": "gp2",
    "MultiAttachEnabled": false
}
```

Db-Pod must be in the same zone i:e us-east-1a. This is done using node selector in definition file.

$ kubectl get nodes --show-labels

```
ubuntu@ip-172-31-85-74:~$ kubectl get nodes --show-labels
NAME                         STATUS   ROLES                  AGE   VERSION   LABELS
ip-172-20-51-92.ec2.internal    Ready    control-plane,master   30m   v1.23.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=t3.small,beta.
kubernetes.io/os=linux,failure-domain.beta.kubernetes.io/region=us-east-1,failure-domain.beta.kubernetes.io/zone=us-east-1a,kops.k8s.io/instancegroup=master-
us-east-1a,kops.k8s.io/kops-controller-pki=,kubernetes.io/arch=amd64,kubernetes.io/hostname=ip-172-20-51-92.ec2.internal,kubernetes.io/os=linux,kubernetes.io
/role=master,node-role.kubernetes.io/control-plane=,node-role.kubernetes.io/master=,node.kubernetes.io/exclude-from-external-load-balancers=,node.kubernetes.
io/instance-type=t3.small,topology.ebs.csi.aws.com/zone=us-east-1a,topology.kubernetes.io/region=us-east-1,topology.kubernetes.io/zone=us-east-1a
ip-172-20-58-40.ec2.internal    Ready    node                   28m   v1.23.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=t3.small,beta.
kubernetes.io/os=linux,failure-domain.beta.kubernetes.io/region=us-east-1,failure-domain.beta.kubernetes.io/zone=us-east-1a,kops.k8s.io/instancegroup=nodes-u
s-east-1a,kubernetes.io/arch=amd64,kubernetes.io/hostname=ip-172-20-58-40.ec2.internal,kubernetes.io/os=linux,kubernetes.io/role=node,node-role.kubernetes.io
/node=,node.kubernetes.io/instance-type=t3.small,topology.ebs.csi.aws.com/zone=us-east-1a,topology.kubernetes.io/region=us-east-1,topology.kubernetes.io/zone
=us-east-1a
ip-172-20-87-109.ec2.internal   Ready    node                   28m   v1.23.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=t3.small,beta.
kubernetes.io/os=linux,failure-domain.beta.kubernetes.io/region=us-east-1,failure-domain.beta.kubernetes.io/zone=us-east-1b,kops.k8s.io/instancegroup=nodes-u
s-east-1b,kubernetes.io/arch=amd64,kubernetes.io/hostname=ip-172-20-87-109.ec2.internal,kubernetes.io/os=linux,kubernetes.io/role=node,node-role.kubernetes.i
o/node=,node.kubernetes.io/instance-type=t3.small,topology.ebs.csi.aws.com/zone=us-east-1b,topology.kubernetes.io/region=us-east-1,topology.kubernetes.io/zon
e=us-east-1b
```

$ kubectl get nodes

$ kubectl describe node ip-172-20-58-40.ec2.internal | grep us-east-1

$ kubectl label nodes ip-172-20-58-40.ec2.internal zone=us-east-1a

$ kubectl describe node ip-172-20-87-109.ec2.internal | grep us-east-1

$ kubectl label nodes ip-172-20-87-109.ec2.internal zone=us-east-1b

```
ubuntu@ip-172-31-85-74:~$ kubectl get nodes
NAME                         STATUS   ROLES                  AGE   VERSION
ip-172-20-51-92.ec2.internal    Ready    control-plane,master   31m   v1.23.7
ip-172-20-58-40.ec2.internal    Ready    node                   29m   v1.23.7
ip-172-20-87-109.ec2.internal   Ready    node                   29m   v1.23.7
ubuntu@ip-172-31-85-74:~$ kubectl describe node ip-172-20-58-40.ec2.internal | grep us-east-1
                    failure-domain.beta.kubernetes.io/region=us-east-1
                    failure-domain.beta.kubernetes.io/zone=us-east-1a
                    kops.k8s.io/instancegroup=nodes-us-east-1a
                    topology.ebs.csi.aws.com/zone=us-east-1a
                    topology.kubernetes.io/region=us-east-1
                    topology.kubernetes.io/zone=us-east-1a
ProviderID:                  aws:///us-east-1a/i-040a17056175a3bf8
ubuntu@ip-172-31-85-74:~$ kubectl label nodes ip-172-20-58-40.ec2.internal zone=us-east-1a
node/ip-172-20-58-40.ec2.internal labeled
ubuntu@ip-172-31-85-74:~$ kubectl describe node ip-172-20-87-109.ec2.internal | grep us-east-1
                    failure-domain.beta.kubernetes.io/region=us-east-1
                    failure-domain.beta.kubernetes.io/zone=us-east-1b
                    kops.k8s.io/instancegroup=nodes-us-east-1b
                    topology.ebs.csi.aws.com/zone=us-east-1b
                    topology.kubernetes.io/region=us-east-1
                    topology.kubernetes.io/zone=us-east-1b
ProviderID:                  aws:///us-east-1b/i-05fc3325b436892b0
ubuntu@ip-172-31-85-74:~$ kubectl label nodes ip-172-20-87-109.ec2.internal zone=us-east-1b
node/ip-172-20-87-109.ec2.internal labeled
```

5. Kube secret for password

app-secret.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: app-secret
type: Opaque
data:
  db-pass: ZWVyZWVkYXBhc3M=
  rmq-pass: Z3Vlc3Q=
```

$ git clone https://github.com/merajafnan/DevOps_Projects.git

$ git checkout kube-setup

$ kubectl create -f app-secret.yaml

$ kubectl get secret

$ kubectl describe secret

```
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl create -f app-secret.yaml
secret/app-secret created
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl get secret
NAME                  TYPE                                  DATA   AGE
app-secret            Opaque                                2      16s
default-token-s9bkq   kubernetes.io/service-account-token   3      19m
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl describe secret
Name:         app-secret
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
db-pass:   11 bytes
rmq-pass:  5 bytes


Name:         default-token-s9bkq
Namespace:    default
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: default
              kubernetes.io/service-account.uid: a1edffa1-7ace-499d-b2e6-1c88787493f7

Type:  kubernetes.io/service-account-token

Data
====
token:      eyJhbGciOiJSUzI1NiIsImtpZCI6InAwYTFlNUFZZkNsY2F5QVFLNlVSS1FYRDYwc1hlSlpVNWM5ckREVWt3RDAifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6ImRlZmF1bHQtdG9rZW4tczlia3EiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC5uYW1lIjoiZGVmYXVsdCIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50LnVpZCI6ImExZWRmZmExLTdhY2UtNDk5ZC1iMmU2LTFjODg3ODc0OTNmNyIsInN1YiI6InN5c3RlbTpzZXJ2aWNlYWNjb3VudDpkZWZhdWx0OmRlZmF1bHQifQ.WsUB0IxJpYI4hQemX5ljF0uu5Ad2R3q3qsVVmupHE8Of1EkKXLALkGNZdeixkboRX4dWutKYRQlhIVdOLNWPV3qcCJ8jo63BfbhZuwNSATvcDVgBtSGRc_y2vP63-hbGrhBt2Y_U6pOqnBpyeXwE2Xkgn8QLOD74H4cnD93D1-t1-HTY_0ZjcTHpMgtppVxtJFEKg5gumouEVKeHkaxU1N6zy-Gl4gaHG__FeSmjfIwtMmDVo8RxuVzgKWLdi2WM0WRv0AXzhCX6G5KoF_6hNdeHpZvtShH8YcgQRXbNybNYkC1d6F07XRAtqHZljPCYdGGtYBXKJ3nWDmgWnpjlew
ca.crt:     1090 bytes
namespace:  7 bytes
```

6. DB Deployment Definition

EC2 > Volumes > vol-005d1868799f9ba3d > Manage tags

Eereedadbdep.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eereedadb
  labels:
    app: eereedadb
spec:
  selector:
    matchLabels:
      app: eereedadb
  replicas: 1
  template:
    metadata:
      labels:
        app: eereedadb
    spec:
      containers:
        - name: eereedadb
          image: merajafnan/eereeda-db:v1
          args:
            - "--ignore-db-dir=lost+found"
          volumeMounts:
            - mountPath: /var/lib/mysql
              name: eereeda-db-data
          ports:
            - name: eereedadb-port
              containerPort: 3306
          env:
            - name: MYSQL_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: app-secret
                  key: db-pass
      nodeSelector:
        zone: us-east-1a
      volumes:
        - name: eereeda-db-data
          awsElasticBlockStore:
            volumeID: vol-005d1868799f9ba3d
            fsType: ext4
```

$ kubectl create -f eereedadbdep.yaml
$ kubectl get pod
$ kubectl describe pod eereedadb-5d4bb79956-g5bg2

```
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl create -f eereedadbdep.yaml
deployment.apps/eereedadb created
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl get pod
NAME                          READY   STATUS    RESTARTS   AGE
eereedadb-5d4bb79956-g5bg2    1/1     Running   0          6s
```

```
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl describe pod eereedadb-5d4bb79956-g5bg2
Name:         eereedadb-5d4bb79956-g5bg2
Namespace:    default
Priority:     0
Node:         ip-172-20-55-29.ec2.internal/172.20.55.29
Start Time:   Tue, 21 Jun 2022 08:23:47 +0000
Labels:       app=eereedadb
              pod-template-hash=5d4bb79956
Annotations:  kubernetes.io/limit-ranger: LimitRanger plugin set: cpu request for container eereedadb
Status:       Running
IP:           100.96.1.6
IPs:
  IP:           100.96.1.6
Controlled By:  ReplicaSet/eereedadb-5d4bb79956
Containers:
Events:
  Type    Reason                  Age   From                     Message
  ----    ------                  ----  ----                     -------
  Normal  Scheduled               24s   default-scheduler        Successfully assigned default/eereedadb-5d4bb79956-g5bg2 to ip-172-20-55-29.ec2.internal
  Normal  SuccessfulAttachVolume  21s   attachdetach-controller  AttachVolume.Attach succeeded for volume "ebs.csi.aws.com-vol-005d1868799f9ba3d"
  Normal  Pulled                  20s   kubelet                  Container image "merajafnan/eereeda-db:v1" already present on machine
  Normal  Created                 20s   kubelet                  Created container eereedadb
  Normal  Started                 19s   kubelet                  Started container eereedadb
```

DB Service Definition

db-CIP.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: eereedadb
spec:
  ports:
    - port: 3306
      targetPort: eereedadb-port
      protocol: TCP
  selector:
    app: eereedadb
  type: ClusterIP
```

7. Memcached Deployment & Service

mcdep.yaml

```
spec:
  selector:
    matchLabels:
      app: eereedamc
  replicas: 1
  template:
    metadata:
      labels:
        app: eereedamc
    spec:
      containers:
      - name: eereedamc
        image: memcached
        ports:
        - name: eereedamc-port
          containerPort: 11211
```

db-CIP.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: eereedacache01
spec:
  ports:
  - port: 11211
    targetPort: eereedamc-port
    protocol: TCP
  selector:
    app: eereedamc
  type: ClusterIP
```

8. RabbitMQ Deployment & Service

rmqdep.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eereedamq01
  labels:
    app: eereedamq01
spec:
  selector:
    matchLabels:
      app: eereedamq01
  replicas: 1
  template:
    metadata:
      labels:
        app: eereedamq01
    spec:
      containers:
      - name: eereedamq01
        image: rabbitmq
        ports:
        - name: eereedamq01-port
          containerPort: 15672
        env:
         - name: RABBITMQ_DEFAULT_PASS
           valueFrom:
             secretKeyRef:
               name: app-secret
               key: rmq-pass
         - name: RABBITMQ_DEFAULT_USER
           value: "guest"
```

rmq-cip.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: eereedamq01
spec:
  ports:
  - port: 15672
    targetPort: eereedamq01-port
    protocol: TCP
  selector:
    app: eereedamq01
  type: ClusterIP
```

9. RabbitMQ Deployment, Service and Initcontainers

eereeda-appdep.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eereedaapp
  labels:
    app: eereedaapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: eereedaapp
  template:
    metadata:
      labels:
        app: eereedaapp
    spec:
      containers:
      - name: eereedaapp
        image: merajafnan/eereeda-app:V1
        ports:
        - name: eereedaapp-port
          containerPort: 8080
      initContainers:
      - name: init-mydb
        image: busybox
        command: ['sh', '-c', 'until nslookup vprodb; do echo waiting
for mydb; sleep 2; done;']
      - name: init-memcache
        image: busybox
        command: ['sh', '-c', 'until nslookup vprocache01; do echo
waiting for mydb; sleep 2; done;']
```

app-CIP.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: eereedaapp-service
spec:
  ports:
  - port: 80
    targetPort: eereedaapp-port
    protocol: TCP
  selector:
    app: eereedaapp
  type: LoadBalancer
```

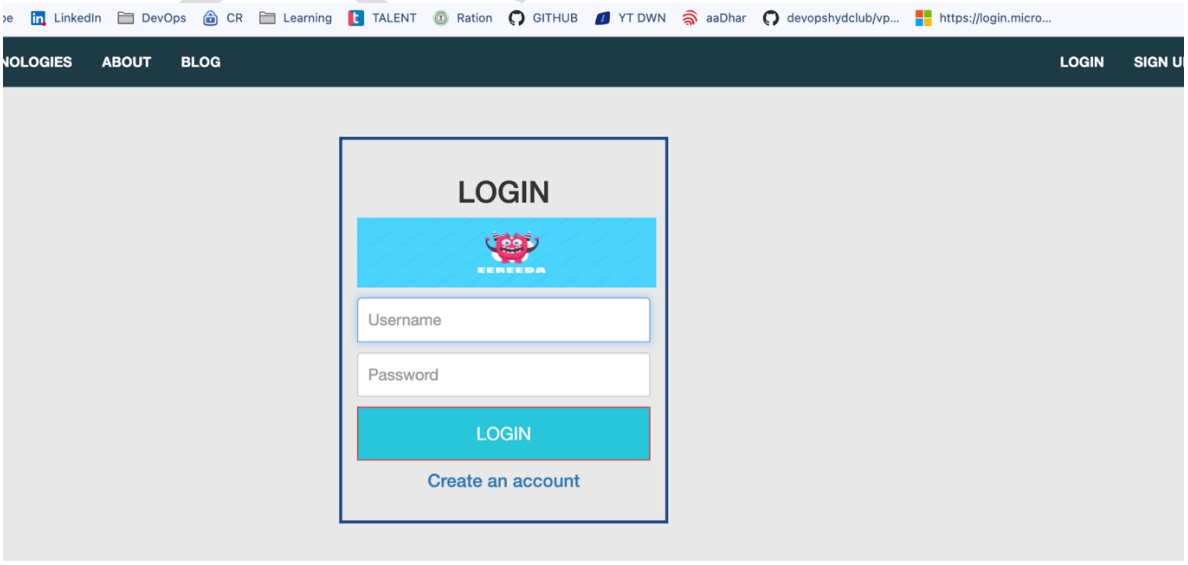
10.Provision Stack on K8s Custer

$ kubectl create -f .
$ kubectl get deploy
$ kubectl get pods
$ kubectl get svc

```
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl create -f .
service/eereedamq01 created
deployment.apps/eereedamq01 created
Error from server (AlreadyExists): error when creating "app-CIP.yaml": services "eereedaapp-service" alr
Error from server (AlreadyExists): error when creating "app-secret.yaml": secrets "app-secret" already e
Error from server (AlreadyExists): error when creating "db-CIP.yaml": services "eereedadb" already exist
Error from server (AlreadyExists): error when creating "eereeda-appdep.yaml": deployments.apps "eereedaa
Error from server (AlreadyExists): error when creating "eereedadbdep.yaml": deployments.apps "eereedadb"
Error from server (AlreadyExists): error when creating "mc-CIP.yaml": services "eereedacache01" already
Error from server (AlreadyExists): error when creating "mcdep.yaml": deployments.apps "eereedamc" alread
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl get depl
error: the server doesn't have a resource type "depl"
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl get deploy
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
eereedaapp     0/1     1            0           2m54s
eereedadb      1/1     1            1           65m
eereedamc      1/1     1            1           2m53s
eereedamq01    1/1     1            1           34s
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl get pods
NAME                            READY   STATUS     RESTARTS   AGE
eereedaapp-54bff5c8c8-dxlb4     0/1     Init:0/2   0          3m6s
eereedadb-5d4bb79956-g5bg2      1/1     Running    0          66m
eereedamc-6f46576575-b626x      1/1     Running    0          3m6s
eereedamq01-7f88f7988-hcm9b     1/1     Running    0          47s
```

```
ubuntu@ip-172-31-85-74:~/DevOps_Projects/kube-app$ kubectl get svc
NAME              TYPE           CLUSTER-IP       EXTERNAL-IP                                                                PORT(S)        AGE
eereedaapp-service   LoadBalancer   100.65.151.86    a8df761a633214ac295cfa512974d4ac-983248082.us-east-1.elb.amazonaws.com   80:31777/TCP   4m14s
eereedacache01    ClusterIP      100.64.160.170   <none>                                                                     11211/TCP      4m13s
eereedadb         ClusterIP      100.70.251.38    <none>                                                                     3306/TCP       4m14s
eereedamq01       ClusterIP      100.67.155.20    <none>                                                                     15672/TCP      114s
kubernetes        ClusterIP      100.64.0.1       <none>                                                                     443/TCP        3h25m
```

Open URL: a8df761a633214ac295cfa512974d4ac-983248082.us-east-1.elb.amazonaws.com



11. References

The following table summarizes the documents referenced in this plan.

DOCUMENT NAME	INSTRUCTOR	LOCATION
DevOps Beginners to Advanced	Imran Teli	https://www.udemy.com/course/decodingdevops/learn/lecture/28273912?start=0#overview