**Deploy Application with Helm**

**Pre-requisites:**

* Java
* Git
* Install Maven
* Install Docker
* Install Helm3
* Install mysql
* EKS-Cluster

**Installations:**

**Java:**

yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel –y

**Docker:**

amazon-linux-extras install docker

service docker start

usermod -a -G docker ec2-user

chmod 666 /var/run/docker.sock

chown ec2-user:docker /var/run/docker.sock

**Maven:**

wget <http://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz>

tar xvzf apache-maven-3.6.3-bin.tar.gz

vi /etc/profile.d/maven.sh

----------------------------------------------------------------------------

export MAVEN\_HOME=/opt/apache-maven-3.6.3

export PATH=$PATH:$MAVEN\_HOME/bin

----------------------------------------------------------------------------

source /etc/profile.d/maven.sh

mvn -version

**Kubectl:**

curl -o kubectl https://amazon-eks.s3-us-west-2.amazonaws.com/1.14.6/2019-08-22/bin/linux/amd64/kubectl

chmod +x ./kubectl

mkdir -p $HOME/bin

cp ./kubectl $HOME/bin/kubectl

export PATH=$HOME/bin:$PATH

echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc

source $HOME/.bashrc

kubectl version --short –client

**aws-iam-authenticator:**

curl -o aws-iam-authenticator https://amazon-eks.s3-us-west-2.amazonaws.com/1.14.6/2019-08-22/bin/linux/amd64/aws-iam-authenticator

chmod +x ./aws-iam-authenticator

cp ./aws-iam-authenticator $HOME/bin/aws-iam-authenticator

export PATH=$HOME/bin:$PATH

echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc

source ~/.bashrc

aws-iam-authenticator –help

**Helm3:**

wget https://get.helm.sh/helm-v3.0.2-linux-amd64.tar.gz

tar xvzf helm-v3.0.2-linux-amd64.tar.gz

cd linux-amd64/

mv helm /usr/local/bin

export PATH=$PATH:/usr/local/bin

**mysql:**

wget https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm

yum localinstall mysql57-community-release-el7-11.noarch.rpm

yum install mysql-community-server

systemctl start mysqld.service

cat /var/log/mysqld.log # Here we get pwd

mysql -u root -p # Mysql Login

ALTER USER 'root'@'localhost' IDENTIFIED BY 'Naresh#240'; # To change pwd for "root" user

GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY 'Naresh#240' WITH GRANT OPTION; # Enable Remote Access (Grant)

**Get code from github:** To get code

git clone <https://github.com/Naresh240/employee-jdbc.git>

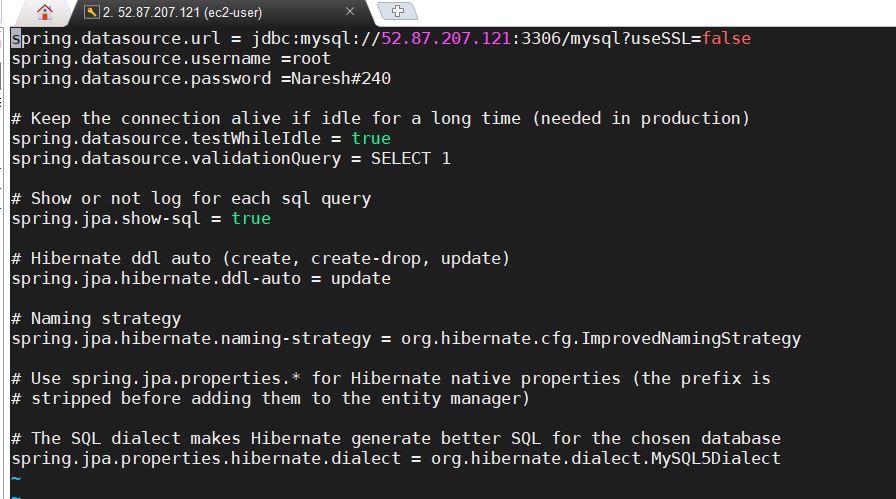
**Goto project**:

cd employee-jdbc

Open file **src/main/resources/application.properties** and change IP address as ur master Machine

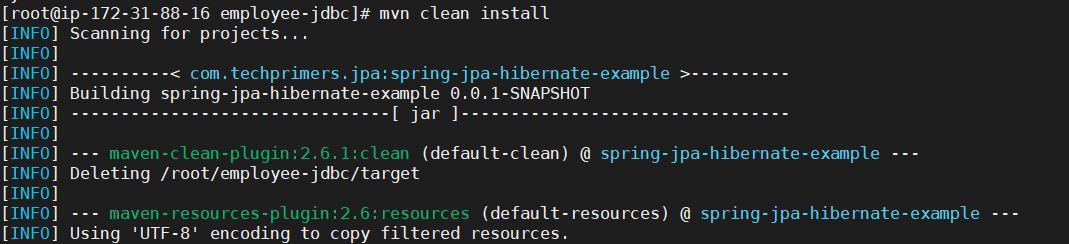
vi src/main/resources/application.properties

spring.datasource.url = jdbc:mysql://<ur-ip-address>:3306/mysql?useSSL=false



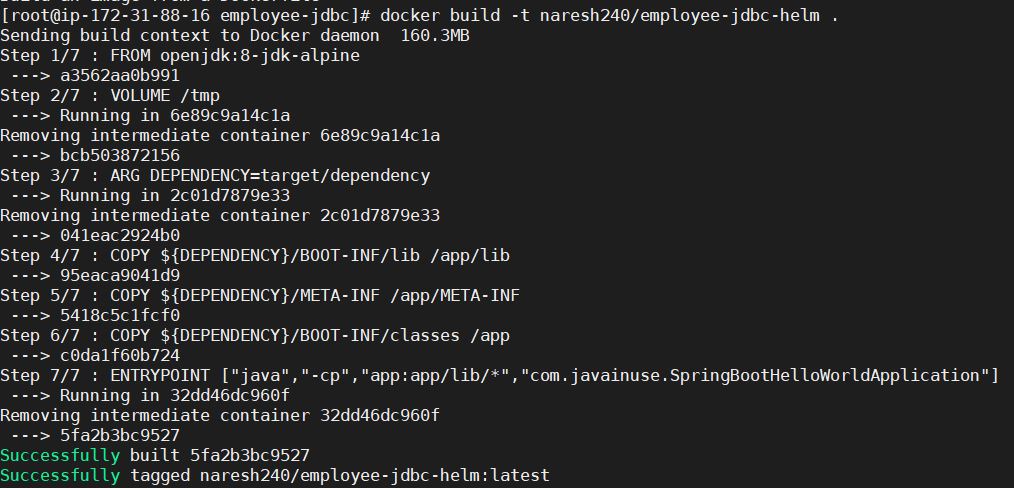
**Build war/jar file:**

mvn clean install



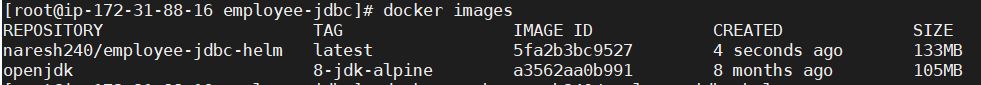
**Build Docker Image:**

docker build –t naresh240/employee-jdbc-helm .



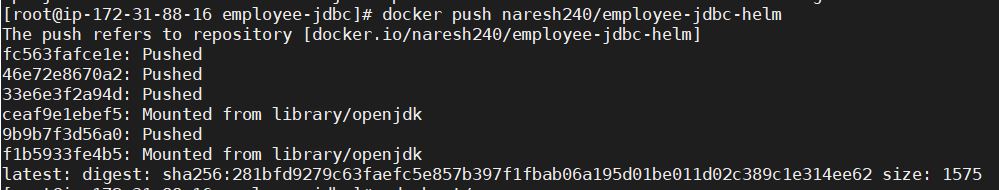
**To check docker images:**

docker images



**Push Docker Image to Our Repository:**

docker push naresh240/employee-jdbc-helm



**To push docker image to our docker hub, we need to login first into docker**

docker login

**Deploy Application with Helm using Docker Imange:**

**Step1:**

if chart is already exit leave this step

( In our code chart already there, so Just check step3 and step4 data)

helm create chart

**step2:**

Goto values.yaml file and check these data: Image name and type of Service

Path: vi employee-jdbc/chart/values.yaml

replicaCount: 2

image:

repository: naresh240/employee-jdbc-helm

tag: latest

pullPolicy: IfNotPresent

nameOverride: ""

fullnameOverride: ""

service:

type: LoadBalancer

port: 8080

ingress:

enabled: false

annotations: {}

paths: []

hosts:

- chart-example.local

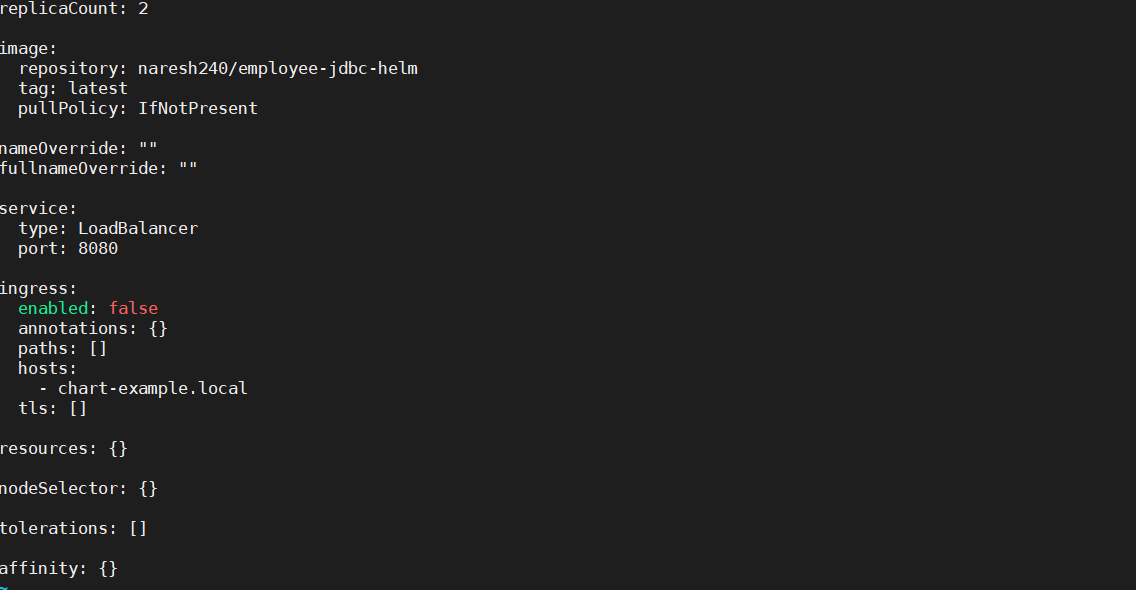
tls: []

resources: {}

nodeSelector: {}

tolerations: []

affinity: {}



**Step3:**

Goto deployment.yaml and delete “livenessProbe” and “readinessProbe”:

Path: vi employee-jdbc/chart/ templates/ deployment.yaml

kind: Deployment

metadata:

name: {{ include "chart.fullname" . }}

labels:

app.kubernetes.io/name: {{ include "chart.name" . }}

helm.sh/chart: {{ include "chart.chart" . }}

app.kubernetes.io/instance: {{ .Release.Name }}

app.kubernetes.io/managed-by: {{ .Release.Service }}

spec:

replicas: {{ .Values.replicaCount }}

selector:

matchLabels:

app.kubernetes.io/name: {{ include "chart.name" . }}

app.kubernetes.io/instance: {{ .Release.Name }}

template:

metadata:

labels:

app.kubernetes.io/name: {{ include "chart.name" . }}

app.kubernetes.io/instance: {{ .Release.Name }}

spec:

containers:

- name: {{ .Chart.Name }}

image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"

imagePullPolicy: {{ .Values.image.pullPolicy }}

ports:

- name: http

containerPort: 8080

protocol: TCP

resources:

{{- toYaml .Values.resources | nindent 12 }}

{{- with .Values.nodeSelector }}

nodeSelector:

{{- toYaml . | nindent 8 }}

{{- end }}

{{- with .Values.affinity }}

affinity:

{{- toYaml . | nindent 8 }}

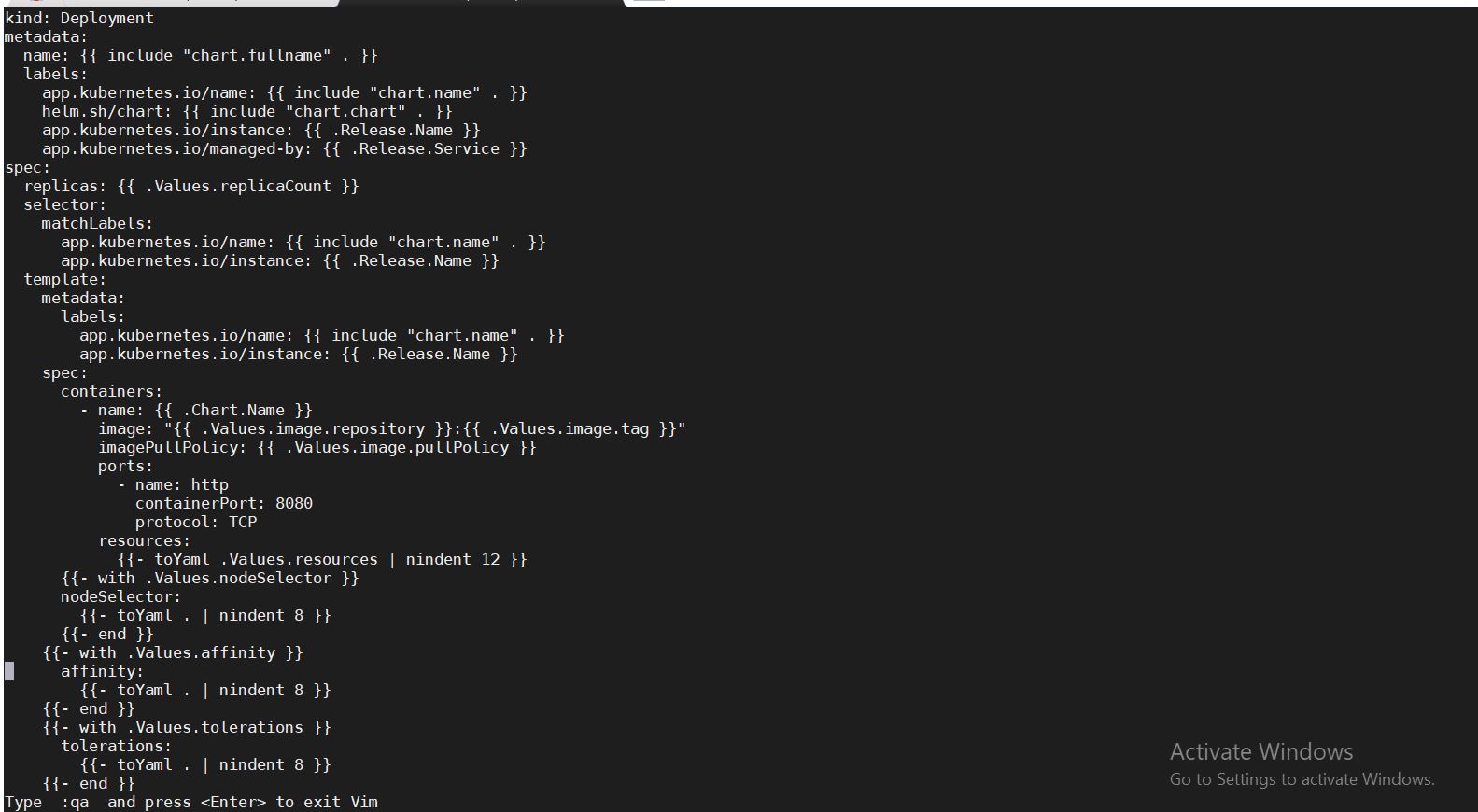
{{- end }}

{{- with .Values.tolerations }}

tolerations:

{{- toYaml . | nindent 8 }}

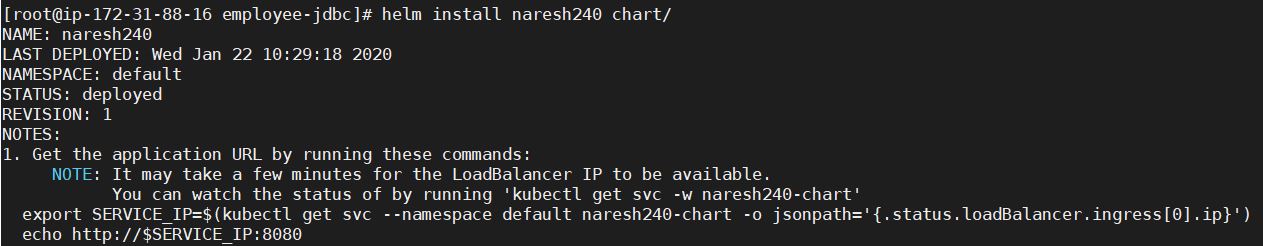
{{- end }}



**Step4:**

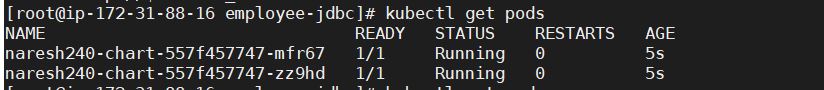
Deploy application using below command

helm install naresh240 chart/

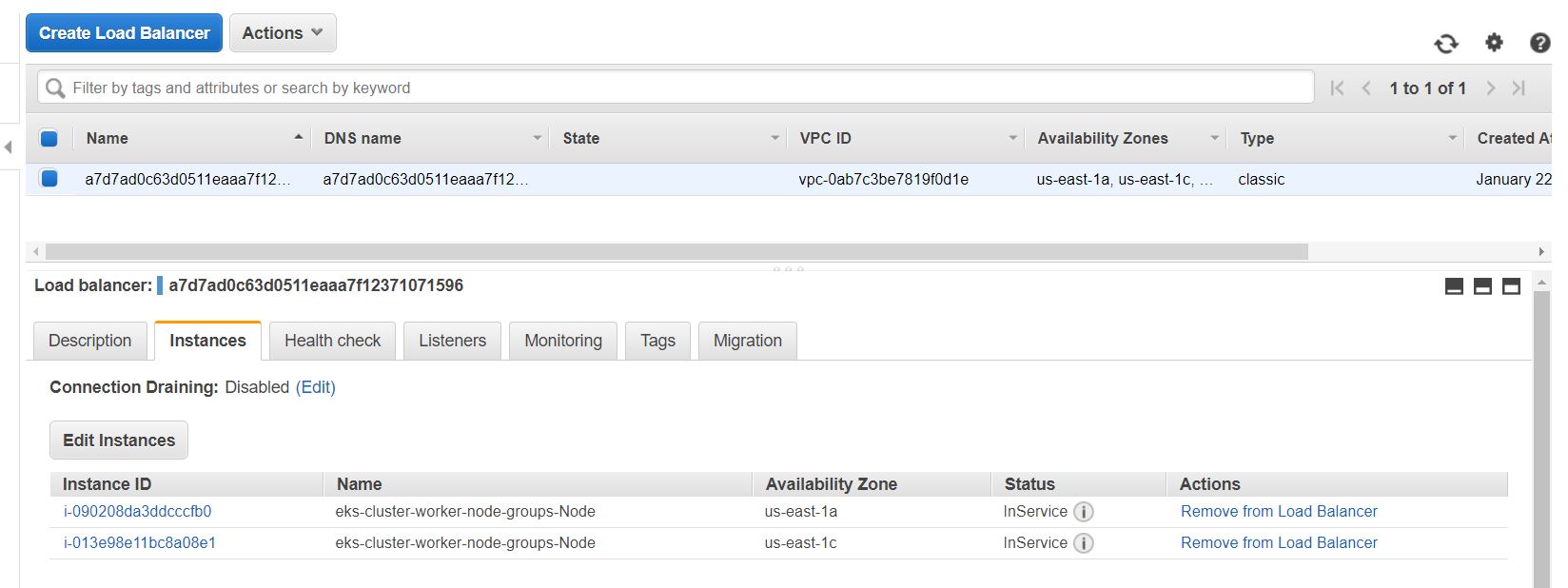


Check whether the pods are Running or not:

kubectl get pods



Open loadbalancer and check whether the nodes comes InService or not:



Connect to Mysql with mysql database:

mysql –u root –p

Connect to Database:

use mysql;

Create table:

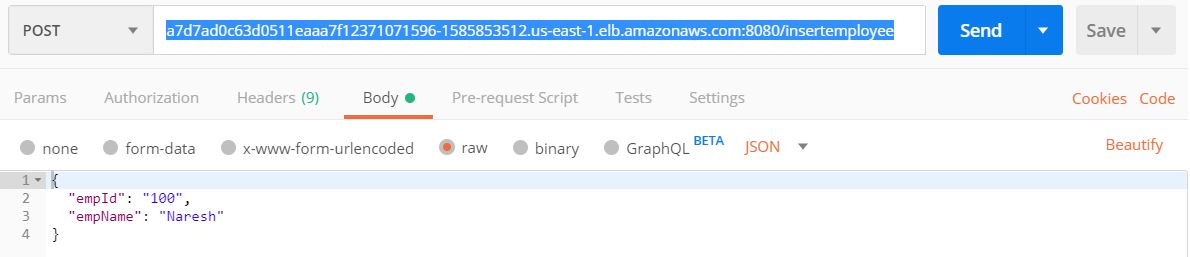
create table employee(empId varchar(40), empName varchar(40));

D:\Naresh\snipping\Capture.JPG

Open Postman app:

Keep “post” method and give url as:

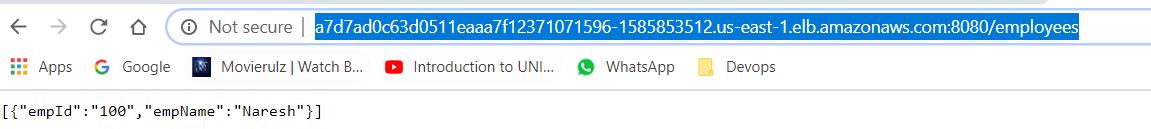
a7d7ad0c63d0511eaaa7f12371071596-1585853512.us-east-1.elb.amazonaws.com:8080/insertemployee



Check output url for “/employees” API:

Goto crome and give url as:

<http://a7d7ad0c63d0511eaaa7f12371071596-1585853512.us-east-1.elb.amazonaws.com:8080/employees>



**Delete Deployment:**

helm del naresh240

D:\Naresh\snipping\Capture.JPG