**DOCKER FILE**

FROM ubuntu:18.04

ENV DEBIAN\_FRONTEND=non-interactive

# Install dependencies

RUN apt-get update -y

RUN apt-get install -y git curl apache2 php libapache2-mod-php php-mysql

# Install app

RUN rm -rf /var/www/html/\*

ADD src /var/www/html/

# Configure apache

RUN a2enmod rewrite

RUN chown -R www-data:www-data /var/www/html

ENV APACHE\_RUN\_DIR /var/www/html

ENV APACHE\_RUN\_USER www-data

ENV APACHE\_RUN\_GROUP www-data

ENV APACHE\_LOG\_DIR /var/log/apache2

EXPOSE 80

CMD ["/usr/sbin/apache2", "-D", "FOREGROUND"]

**JENKINS FILE**

node {

stage('Checkout Source Code') {

checkout scm

}

stage('Create Docker Image') {

docker.build("docker\_image:${env.BUILD\_NUMBER}")

}

stage ('Run Application') {

try {

// Stop existing Container

sh 'docker rm docker\_container -f'

// Start database container here

sh "docker run -d --name docker\_container docker\_image:${env.BUILD\_NUMBER}"

}

catch (error) {

} finally {

// Stop and remove database container here

}

}

stage ('Notifications') {

mail body: "Project Execution Completed with status : " + currentBuild.result ,

subject: 'Project Execution Notification',

to: 'abc@abc.com'

}

}

**KUBERNETS FILE**

apt-get update && apt-get upgrade

apt-get update && apt-get install -y curl apt-transport-https docker.io

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -

echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" >/etc/apt/sources.list.d/kubernetes.list

apt-get update

apt-get install -y kubelet kubeadm kubectl

kubeadm init

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

export kubever=$(kubectl version | base64 | tr -d '\n')

kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"

kubectl get node

kubectl get pods --all-namespaces

kubectl taint nodes --all node-role.kubernetes.io/master-

kubectl create namespace application

kubectl get pods --all-namespaces

kubectl run kubernetes-bootcamp --image=docker.io/jocatalin/kubernetes-bootcamp:v1 --port=8080

kubectl get services

kubectl expose deployment/kubernetes-bootcamp --port=8080 --target-port=8080 --type=NodePort

kubectl expose deployment/kubernetes-bootcamp --port=8080 --target-port=8080 --type=LoadBalancer

kubectl describe services kubernetes-bootcamp

kubectl exec -ti $POD\_NAME curl localhost:8080

kubectl scale deployments/kubernetes-bootcamp --replicas=2

kubectl get pods -o wide

kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=jocatalin/kubernetes-bootcamp:v2

kubectl rollout undo deployments/kubernetes-bootcamp

kubectl rollout status deployments/kubernetes-bootcamp

**JSON FILE**

{

"family": "console-sample-app",

"volumes": [

{

"name": "my-vol",

"host": {}

}

],

"containerDefinitions": [

{

"environment": [],

"name": "simple-app",

"image": "amazon/amazon-ecs-sample",

"cpu": 10,

"memory": 500,

"portMappings": [

{

"containerPort": 80,

"hostPort": 80

}

],

"mountPoints": [

{

"sourceVolume": "my-vol",

"containerPath": "/var/www/my-vol"

}

],

"entryPoint": [

"/usr/sbin/apache2",

"-D",

"FOREGROUND"

],

"essential": true

},

{

"name": "busybox",

"image": "busybox",

"cpu": 10,

"memory": 500,

"volumesFrom": [

{

"sourceContainer": "simple-app"

}

],

"entryPoint": [

"sh",

"-c"

],

"command": [

"/bin/sh -c \"while true; do /bin/date > /var/www/my-vol/date; sleep 1; done\""

],

"essential": false

}

]

}

**APPLICATION TEST FILE**

package com.org;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class ApplicationTests {

@Test

void contextLoads() {

}

}

**Application.java Class**

package com.org;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan(basePackages = "com.org")

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

**Controller.java Class**

package com.org;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping(path="/aws")

public class Controller {

@GetMapping("/")

public String Hello()

{

return "Hello from AWS";

}

}