\*\*Deploying mule application on cloudhub through CI/CD pipeline in Jenkins.

pipeline

{

agent any

stages{

stage('SonarQube analysis') {

steps {

withSonarQubeEnv('SonarQube') {

bat 'mvn sonar:sonar -Dsonar.sources=src/ -Dsonar.host.url=http://localhost:9000 -Dsonar.login=c91a76d35e003f1d1af99856195a65522c327514'

}

}

}

stage("Quality gate") {

steps {

waitForQualityGate abortPipeline: true

}

}

stage('Build Application'){

steps{

bat 'mvn clean package -DskipTests'

}

}

stage('Munit Testing'){

steps{

bat 'mvn clean test'

}

}

stage('Deploy Application To Mulesoft'){

steps{

bat 'mvn package deploy -DmuleDeploy -Danypoint.userName=OssomVictory5 -Danypoint.password=\*\*\*\*\*\*\*'

}

}

}

}

\*\*Creating Docker Image on EC2 instance:

steps

1) EC2 -> ubuntu 20.4 ->vpc ->20gb storage

2) security -> all traffic ->anywhere

3) launch instance

4) puttygen -> upkoad pem -> save private key

5) putty -> auth -> browse ppk

6) putty -> host ->ubuntu@ip

/////////////////////////////////////////////

1)open puty

2) $ sudo su

3) # apt update -y

4) # apt install openjdk-11-jdk-headless (y)

# apt update -y

5) # apt install docker.io (y)

6) # systemctl status docker (press q ) // or active(running) → # systemctl enable --now Docker

7) # apt install maven (y)

8) # apt install unzip

9) # mvn --version

/////////////////////////////////////////////////

1) open springstarter

2) create springboot proj

3) maven , java 11 ,2.4.5 ...etc

4) add spring web dependency

5)downlaod zip

6) create s3 bkt

7) upload zip of spring proj

8) make public

///////////////////////////////////////////////////

1) # wget S3-Object-Url.zip

2) # unzip dsb.zip

3) # ls

4) # cd dsb

5) # ls

6) # cd src/main/java/com/nubeera/dsb

7) # mkdir resource

8) # cd resource

9) # nano HelloResource.java

10) copy code

package com.nubeera.dsb.resource;

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

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

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

@RestController

@RequestMapping("/rest/docker/hello")

public class HelloResource {

@GetMapping

public String hello() {

return "Welcome to NubeEra";

}

}

ctrl+X y

enter

11) cd ..

(till you get main .... then cd resources java resources )

12) cd resource

13) vi application.properties

14) server.port=8085

(esc) :wq

15) cd .. till dsb

16) vi pom.xml

(press i)

go down in <build > <plugins> after first </plugins > and before </build>

17) <finalName>dsb</fianlName>

(esc) :wq

18) cat > Dockerfile

//FROM openjdk:11 → Download OS, Install OS, Run OS,Runtimes

//ADD target/docker-spring-boot.jar docker-spring-boot.jar

//EXPOSE 8085= Container(Running-OS,App2)→ Local Machine → Public

//ENTRYPOINT ["java","-jar","docker-spring-boot.jar"]

FROM openjdk:11

ADD target/dsb.jar dsb.jar

EXPOSE 8085

ENTRYPOINT ["java","-jar","dsb.jar"]

ctrl+z

19) mvn clean install

20) docker build -f Dockerfile -t dsb .

//docker build -f Dockerfile -t dsb (above was not working)

21) docker images

22) docker run -p 8085:8085 dsb

23) Duplicate session

curl http://localhost:8085/rest/docker/hello

24) Browser

public ipv4:port/rest/docker/hello

<http://184.73.77.10:8095/rest/docker/hello>

\*\*Deploying Java application through azure devOps on azure app service.

Graphical user interface, application, email

Description automatically generated

A picture containing graphical user interface

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

\*\*Building Docker Image and pushing it into ACR

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, application, email

Description automatically generated

\*\*Deploying Image to AKS:

Deployment.yml

apiVersion: apps/v1

kind: Deployment

metadata:

name: myapp-deployment

labels:

app: myapp

spec:

replicas: 1

selector:

matchLabels:

app: myapp

template:

metadata:

labels:

app: myapp

spec:

containers:

- name: myapp

image: appregistry10002313.azurecr.io/myapp:latest

ports:

- containerPort: 80

Service.yml

apiVersion: v1

kind: Service

metadata:

name: app-service

spec:

type: LoadBalancer

ports:

- port: 80

selector:

app: myapp

\*\*Creating VM in Azure using ARM template.

Vm.json

{

"$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",

"contentVersion": "1.0.0.0",

"parameters": {},

"functions": [],

"variables": {},

"resources": [

{

"name": "vmstore5677676",

"type": "Microsoft.Storage/storageAccounts",

"apiVersion": "2021-04-01",

"location": "[resourceGroup().location]",

"sku": {

"name": "Standard\_LRS"

},

"kind": "StorageV2"

},

{

"name": "app-nsg",

"type": "Microsoft.Network/networkSecurityGroups",

"apiVersion": "2020-11-01",

"location": "[resourceGroup().location]",

"properties": {

"securityRules": [

{

"name": "nsgRule1",

"properties": {

"description": "description",

"protocol": "Tcp",

"sourcePortRange": "\*",

"destinationPortRange": "3389",

"sourceAddressPrefix": "\*",

"destinationAddressPrefix": "\*",

"access": "Allow",

"priority": 100,

"direction": "Inbound"

}

}

]

}

},

{

"name": "appvm",

"type": "Microsoft.Compute/virtualMachines",

"apiVersion": "2021-03-01",

"location": "[resourceGroup().location]",

"dependsOn": [

"[resourceId('Microsoft.Storage/storageAccounts', toLower('vmstore5677676'))]"

],

"properties": {

"hardwareProfile": {

"vmSize": "Standard\_A2\_v2"

},

"osProfile": {

"computerName": "appvm",

"adminUsername": "demousr",

"adminPassword": "Azure@123"

},

"storageProfile": {

"imageReference": {

"publisher": "MicrosoftWindowsServer",

"offer": "WindowsServer",

"sku": "2019-Datacenter",

"version": "latest"

},

"osDisk": {

"name": "windowsVM1OSDisk",

"caching": "ReadWrite",

"createOption": "FromImage"

},

"dataDisks": [

{

"name":"vm-data-disk",

"diskSizeGB":16,

"createOption": "Empty",

"lun":0

}

]

},

"networkProfile": {

"networkInterfaces": [

{

"id": "[resourceId('Microsoft.Network/networkInterfaces', 'app-interface')]"

}

]

},

"diagnosticsProfile": {

"bootDiagnostics": {

"enabled": true,

"storageUri": "[reference(resourceId('Microsoft.Storage/storageAccounts/', toLower('vmstore5677676'))).primaryEndpoints.blob]"

}

}

}

}

],

"outputs": {}

}