The following is the content of dev.tfvars in the dev directory:

bucket\_name = "angular-sample-app-dev"

tfstate-dir = "dev/terraform.tfstate"

Given here is the content of prod.tfvars in the prod directory:

bucket\_name = "angular-sample-app-prod"

tfstate-dir = "prod/terraform.tfstate"

This is a var.tf file:

variable "apac\_region" {

default = "YOUR\_REGION"

}

#Keep Empty to initialize it in terraform.tfvars; keep terraform.tfvars in git ignore list so secret information is not available

variable "bucket\_name" {}

variable "tfstate-dir" {}

variable "mime\_types" {

default = {

html = "text/html"

css = "text/css"

js = "application/javascript"

map = "application/javascript"

jpg = "image/jpeg"

ico = "image/ico"

}

}

Given here is a version.tf file to verify the terraform version:

terraform {

required\_version = ">= 0.12"

}

terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "3.40.0"

}

}

backend "s3" {

bucket = "tfstate-bucket0703"

key = "terraform.tfstate"

region = "YOUR\_REGION"

}

}

provider "aws" {

region = var.apac\_region

profile = "default"

}

#Provides an S3 object resource.

resource "aws\_s3\_bucket" "angular\_s3\_bucket" {

bucket = var.bucket\_name

acl = "private"

website {

index\_document = "index.html"

}

tags = {

Name = var.bucket\_name

}

}

# Attaches a policy to an S3 bucket resource.

resource "aws\_s3\_bucket\_policy" "angular\_s3\_bucket\_policy" {

bucket = aws\_s3\_bucket.angular\_s3\_bucket.id

policy = jsonencode({

"Version": "2012-10-17",

"Statement": [

{

"Sid": "PublicReadGetObject",

"Effect": "Allow",

"Principal": "\*",

"Action": [

"s3:GetObject",

"s3:GetObjectVersion"

],

"Resource": "arn:aws:s3:::${var.bucket\_name}/\*"

}

]

})

}

# Provides an S3 object resource.

#content\_type - Standard MIME type describing the format of the object data, e.g., application/octet-stream. All Valid MIME Types are valid for this input.

resource "aws\_s3\_bucket\_object" "object" {

for\_each = fileset("browser/", "\*\*/\*.\*")

bucket = aws\_s3\_bucket.angular\_s3\_bucket.id

key = each.value

content\_type = lookup(var.mime\_types, split(".", each.contenttype)[length(split(".", each.contenttype)) - 1])

source = "browser/${each.contenttype}"

etag = filemd5("browser/${each.contenttype}")

}

Dockerfile:

==========

FROM nginx:1-alpine

COPY dist/browser/ /usr/share/nginx/html

build-agent-pod.yaml:

====================

apiVersion: v1

kind: Pod

spec:

containers: # list of containers that you want present for your build, you can define a default container in the Jenkinsfile

- name: docker

image: docker:latest

command: ["tail", "-f", "/dev/null"]

imagePullPolicy: Always

volumeMounts:

- name: docker

mountPath: /var/run/docker.sock # k8s host docker engine

- name: node

image: node:latest

command: ["tail", "-f", "/dev/null"]

imagePullPolicy: Always

volumes:

- name: docker

hostPath:

path: /var/run/docker.sock

app-deployment.yaml:

===================

apiVersion: apps/v1

kind: Deployment

metadata:

name: angular-demo-deployment

namespace: jenkins

labels:

app: angular-demo-app

spec:

template:

metadata:

name: angular-demo-pod

labels:

app: angular-demo

spec:

serviceAccountName: jenkins

containers:

- name: angular-demo

image: USERNAME/angular-demo:0.1

replicas: 1

selector:

matchLabels:

app: angular-demo

---

apiVersion: v1

kind: Service

metadata:

name: angular-demo-service

namespace: jenkins

labels:

app: angular-demo

spec:

selector:

app: angular-demo

ports:

- protocol: TCP

port: 80

targetPort: 80

type: LoadBalancer

Jenkinsfile:

===========

pipeline {

agent {

kubernetes {

idleMinutes 5 // how long the pod will live after no jobs have run on it

yamlFile 'build-agent-pod.yaml' // path to the pod definition relative to the root of our project

}

}

stages {

stage('Static Code Analysis') {

steps {

container('node') {

echo "Steps to execute SCA"

sh 'wget https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-3.3.0.1492-linux.zip'

sh 'unzip sonar-scanner-cli-3.3.0.1492-linux.zip'

withSonarQubeEnv(installationName: 'SonarQube', credentialsId: 'SonarToken') {

sh 'ls -l'

sh 'sonar-scanner-3.3.0.1492-linux/bin/sonar-scanner -Dsonar.projectVersion=1.0 -Dsonar.projectKey=sample-angular-app -Dsonar.sources=src'

}

waitForQualityGate(abortPipeline: true, credentialsId: 'SonarToken')

}

}

}

stage('UnitTests & Coverage') {

steps {

container('chrome') {

echo "Steps to execute Unit Tests"

catchError(buildResult: 'SUCCESS', stageResult: 'FAILURE') {

sh 'npm install && npm install karma-junit-reporter --save-dev && npm run test --progress false --watch false'

}

}

}

}

stage('Build') {

steps {

container('node') {

echo "Steps to execute Build"

sh 'npm run build'

zip archive: true, dir: 'dist/Demo1', glob: '', zipFile: 'browser.zip'

stash(includes: 'browser.zip', name: 'dist')

}

}

}

stage('IaC & Deploy') {

agent {

label 'terraform'

}

stages {

stage("Download Artifact") {

steps {

unstash 'dist'

unzip dir: 'iac/dist', glob: '', zipFile: 'browser.zip'

}

}

stage('dev-plan') {

steps {

bat '''terraform -version

cd iac

dir

terraform init -var-file=dev/dev.tfvars

terraform workspace new dev

terraform workspace list

terraform workspace select dev

terraform plan --out=mymodule-dev.tfplan --var-file=dev/dev.tfvars'''

}

}

stage('dev-build') {

steps {

bat '''terraform -version

cd iac

terraform workspace select dev

terraform apply -auto-approve mymodule-dev.tfplan'''

}

}

stage('Prod Deployment Approval') {

steps {

input 'Do you want to Deploy in Production environment?'

}

}

stage('prod-plan') {

steps {

bat '''terraform -version

cd iac

dir

terraform init -var-file=prod/prod.tfvars

terraform workspace new prod

terraform workspace list

terraform workspace select prod

terraform plan --out=mymodule-prod.tfplan --var-file=prod/prod.tfvars'''

}

}

stage('prod-build') {

steps {

bat '''terraform -version

cd iac

terraform workspace select prod

terraform apply -auto-approve mymodule-prod.tfplan'''

}

}

}

}

}

}

}

Note: Use any sample Angular Project or React Project.