6 e abril - sábado

Construir uma **Pipeline** (script from SCM) utilizando *Jenkinsfile* com (pelo menos) os passos:

“build”,

“teste**S**” e

“deliver”

• Pode usar um app OpenSource.

• Configure um Webhook do Gitlab p/ o Jenkins (Pipeline)

• Use docker (DooD) para executar pelo menos um passo.

• Use TODOS os comandos *groovy* estudados!

• Faça um deploy no S3, **se** tiver conta na AWS

*(ponto extra)*

Façam uma pipeline ÚNICA executando todos os estágios (build, teste unitário, teste de integração, ... , delivery), ok?

Usem o Jenkinsfile com sintaxe "***scripted***" e não a declarativa.

• Entrega:

- Dockerfile do Jenkins - se a imagem não tiver no Docker Hub

- Caminho para o projeto com Jenkinsfile em repositório PÚBLICO (preciso de acesso) Github

- "Save as PDF" da configuração do job e da execução (console output).

- Qualquer configuração extra que preciso fazer no job (print to PDF da página de configuração do job) e orientações.

<https://www.concrete.com.br/2016/07/18/pipeline-como-codigo-no-jenkins/>

Critérios de Aceitação

* OK Sintaxe Declarativa x “Scripted”
* OK Pipeline definida no repositório
* deleteDir(), Checkout, sh

node {

stage('Build') {

echo 'Building..'

**deleteDir()**

**checkout scm**

**sh 'cat README.md' ------------- faltou readme**

}

stage('Test') {

echo 'Testing..'

}

stage('Deploy') {

echo 'Deploying....'

}

}

* Variáveis (def)

node {

stage('Build') {

echo 'Building..'

**echo "Hello Mr. ${username}"**

deleteDir()

checkout scm

sh 'cat README.md'

}

* Variáveis de Ambiente (env.)

**echo "Running ${env.JOB\_NAME} (${env.BUILD\_ID}) at ${env.JENKINS\_URL}"**

**BUILD\_ID**

ID da build atual (mesmo que BUILD\_NUMBER para Jenkins a partir da v1.597)

**JOB\_NAME**

Nome do projeto dessa build, por exemplo "devops" ou “devops/exercicio1".

**JENKINS\_URL**

URL completa do Jenkins, por exemplo jenkins:8080/jenkins/

def username = 'Jenkins'

**env.CC = 'clang'**

node {

stage('Build') {

**env.DEBUG\_FLAGS = '-g'**

echo 'Building..'

echo "Hello Mr. ${username}"

echo "Running ${env.JOB\_NAME} (${env.BUILD\_ID}) at ${env.JENKINS\_URL}"

deleteDir()

checkout scm

sh 'cat README.md'

**sh 'printenv'**

}

stage('Test') {

echo 'Testing..'

}

stage('Deploy') {

echo 'Deploying....'

}

}

* Entrada de Dados (input)

def ambiente = input id: 'test', message: 'Please Provide Parameters', ok: 'Next',

parameters: [

choice(name: 'ENVIRONMENT',

choices: ['dev','qa'].join('\n'),

description: 'Please select the Environment'),

string(name: 'TAG',

defaultValue: 'tag01',

description: 'Please enter the tag')

]

echo “${ambiente}"

* Execução Paralela

node {

stage('Build') {

echo 'Building..'

}

stage('Test') {

echo 'Testing..'

parallel FrontendTests: { echo 'Testing Frontend..' },

BackendTests: { echo 'Testing Backend..' }

}

stage('Deploy') {

echo 'Deploying....'

}

}

* Node, Stash, Unstash

node {

**stage 'Build'**

**node() {**

echo 'Building..'

deleteDir()

checkout scm

**sh 'echo "res" > result'**

**stash includes: '\*\*/result', name: 'app'**

**}**

**stage 'Test'**

**node("php") {**

echo 'Testing..'

**deleteDir()**

**unstash 'app'**

**sh 'cat result'**

**}**

**stage ‘Deploy'**

**node() {**

echo 'Deploying....'

**}**

}

Jenkins

* Artefatos (archiveArtifacts)

node {

stage 'Build'

node() {

echo 'Building..'

deleteDir()

checkout scm

sh ‘echo “res" > result'

stash includes: ‘\*\*/result', name: 'app'

}

stage 'Test'

node() {

echo 'Testing..'

deleteDir()

unstash 'app'

sh 'cat result’

**archiveArtifacts artifacts: ‘\*\*/result’, fingerprint: true**

}

stage 'Deploy'

node() {

echo 'Deploying....'

}

}

* If, Else

node {

stage 'Build'

node() {

echo 'Building..'

}

stage 'Test'

node() {

echo 'Testing..'

**if (env.BUILD\_ID.toInteger() % 2 == 0) {**

**echo 'Execução PAR'**

**} else {**

**echo 'Execução ÍMPAR'**

**}**

}

stage 'Deploy'

node() {

echo 'Deploying....'

}

}

* Retry, Timeout

node {

stage 'Build'

node() {

echo 'Building..'

}

stage 'Test'

node() {

echo 'Testing..'

**retry(3) {**

**sh './flakey-deploy.sh'**

**}**

**timeout(time: 3, unit: 'MINUTES') {**

**sh './health-check.sh'**

**}**

}

stage 'Deploy'

node() {

echo ‘Deploying....'

}

}

* Try, Catch, Finally

node {

stage 'Build'

node() {

echo 'Building..'

}

stage 'Test'

node() {

echo 'Testing..'

**try {**

**sh 'exit 0'**

**echo 'Sucesso!'**

**}**

**catch (e) {**

**echo 'Falhou.'**

**throw e**

**}**

**finally {**

**echo 'Executa sempre.'**

**}**

}

stage 'Deploy'

node() {

echo ‘Deploying....'

}

}

* Segredos

node {

withCredentials([string(credentialsId: 'mytoken', variable: 'TOKEN')]) {

sh '''

set +x

curl -H "Token: $TOKEN" https://some.api/

'''

}

}

* Deploy no S3