**SESSION - 79**

--> After deployment we have to configure.

--> if deployment success pipeline will end. if in-case failure deployment we need to check whether first time or not

--> if in-case first time if we able rollback -- no we can't rollback since no previous version error.

--> if not first time we can rollback.

--> if rollback success but that is overall deployment error.

--> every time we have to update record.

--> before deployment will see some commands.

--> connect the server.

--> aws configure

--> aws eks update-kubeconfig--region us-east-1 --name roboshop-dev

--> kubectl get nodes

--> connect the Jenkins

1. allow traffic from agent to eks master

2. create namespace

--> Security group Jenkins-agent-server

--> kubectl create ns roboshop

--> kubectl get ns

--> kubens roboshop

--> kubectl rollout status deployment/catalogue

--> in shell command i want to run a shell command and get this.

check rollout status, keep it in variable, write a condition using variable

--> kubectl rollout status deployment/catalogue --request-timeout=30s

--> if 30min before response not came that is failure.

catalogue-cd/Jenkinsfile

// Build

stages {

stage('Check Status'){

steps{

script{

withAWS(credentials: 'aws-creds', region: 'us-east-1') {

def deploymentStatus = sh(returnStdout: true, script: "kubectl rollout status deployment/catalogue --timeout=30s -n $PROJECT || echo FAILED").trim()

if (deploymentStatus.contains("successfully rolled out")) {

echo "Deployment is success"

} else {

sh """

helm rollback $COMPONENT -n $PROJECT

sleep 20

"""

def rollbackStatus = sh(returnStdout: true, script: "kubectl rollout status deployment/catalogue --timeout=30s -n $PROJECT || echo FAILED").trim()

if (rollbackStatus.contains("successfully rolled out")) {

error "Deployment is Failure, Rollback Success"

}

else{

error "Deployment is Failure, Rollback Failure. Application is not running"

}

}

}

--> Click rebuild

--> expected token line no 42.

--> it should be in the codes line.

--> Console Output.

--> kubectl nodes

--> with AWS is mandatory otherwise it won't get credentials

--> push the code

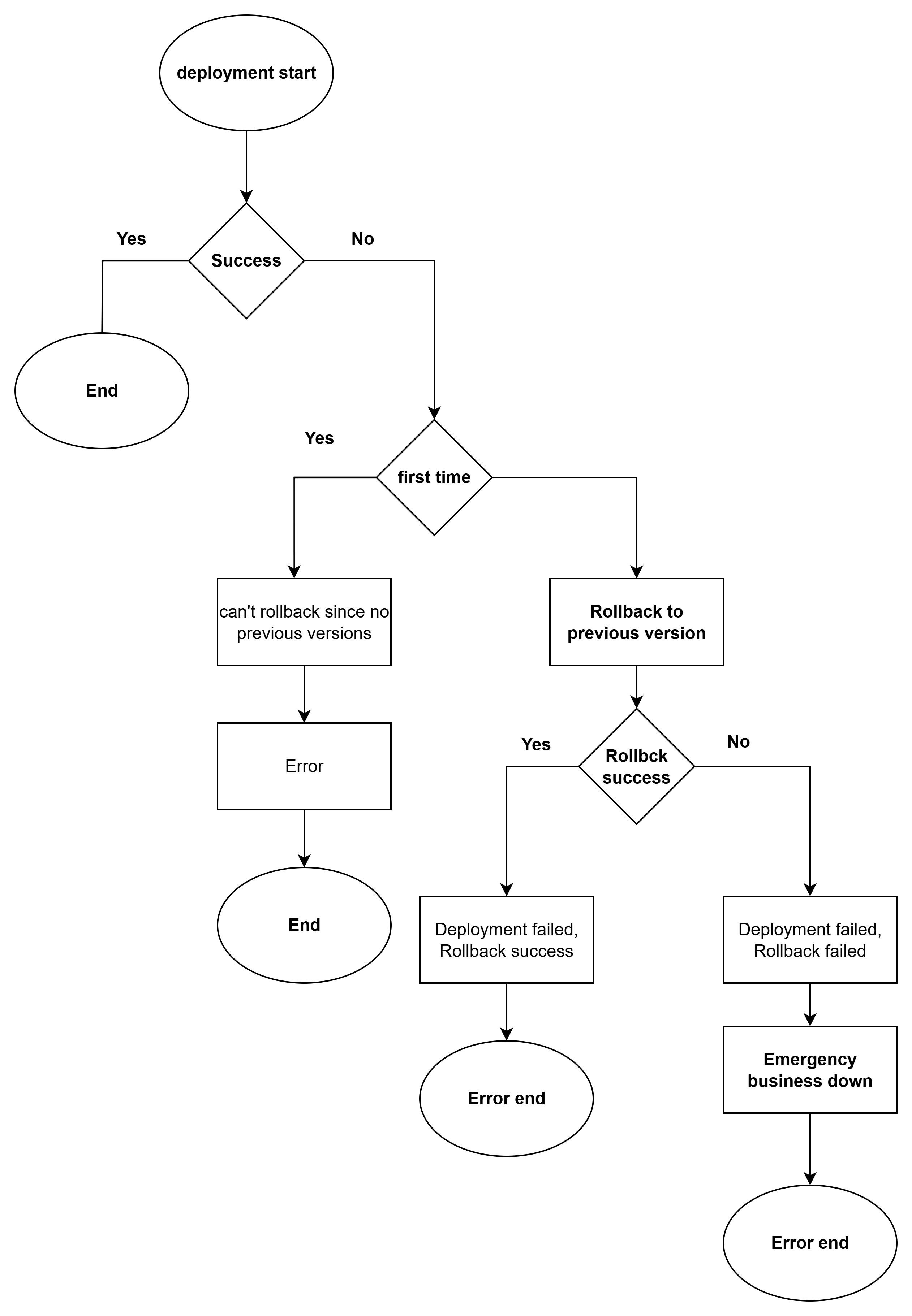
--> check the Jenkins

--> helm list

--> helm uninstall roboshop

--> helm uninstall catalogue -n roboshop

--> kubectl rollout status deployment/catalogue --request-timeout=30s -n roboshop



catalogue-cd/templates/deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: catalogue

namespace: roboshop

# deployment labels

labels:

component: catalogue

project: roboshop

tier: app

spec:

replicas: 1

# These are the labels replica set use to create pod replicas, this should match pod labels

selector:

matchLabels:

component: catalogue

project: roboshop

tier: app

# This is pod definition

template:

metadata:

labels:

component: catalogue

project: roboshop

tier: app

spec:

containers:

- name: catalogue

image: "{{ .Values.deployment.imageURL}}:{{ .Values.deployment.imageVersion }}"

command: ["sh", "-c", "echo 'failure' && exit 1"]

resources:

requests:

cpu: "50m"

memory: "128Mi"

limits:

cpu: "100m"

memory: "256Mi"

envFrom:

- configMapRef:

name: catalogue

--> git add . ; git commit -m "Jenkins"; git push origin main

--> Rebuild

--> will see after failure whether this one will rollback or not

--> it will wait for 10min

--> if fails it will come like this

--> kubectl rollout status deployment/catalogue --request-timeout=30s -n roboshop

waiting for deployment "catalogue" roll out to finish: 0 of 1 updated replicas are available (this is error)

--> Rebuild again (build means running)

--> actually 4 should be failure

--> kubectl get pods

--> helm uninstall catalogue -n roboshop

--> kubectl rollout status deployment/catalogue --request-timeout=30s -n roboshop

1. first time failure, helm can't rollback because there will be no 0 version

2. 2nd deployment also failure, helm rollback attempts, rollback also will be failure. deployment failed and rollback failed

3. 3rd deployment success, deployment is success

4. 4th revision, failure. rollback is success, deployment is failure

**scans**

**=======**

**shift left -->** bringing the testing and scanning to the early stages like DEV environment is shift left instead of doing in higher environments

**build once in DEV and run anywhere -->** We only build the application in DEV environment, we don’t build in multiple environments, we promote the application to multiple environments with different configuration

**static source code analysis -->** sonarqube

**static application secuirty testing -->** sonarqube, fortify scan, github

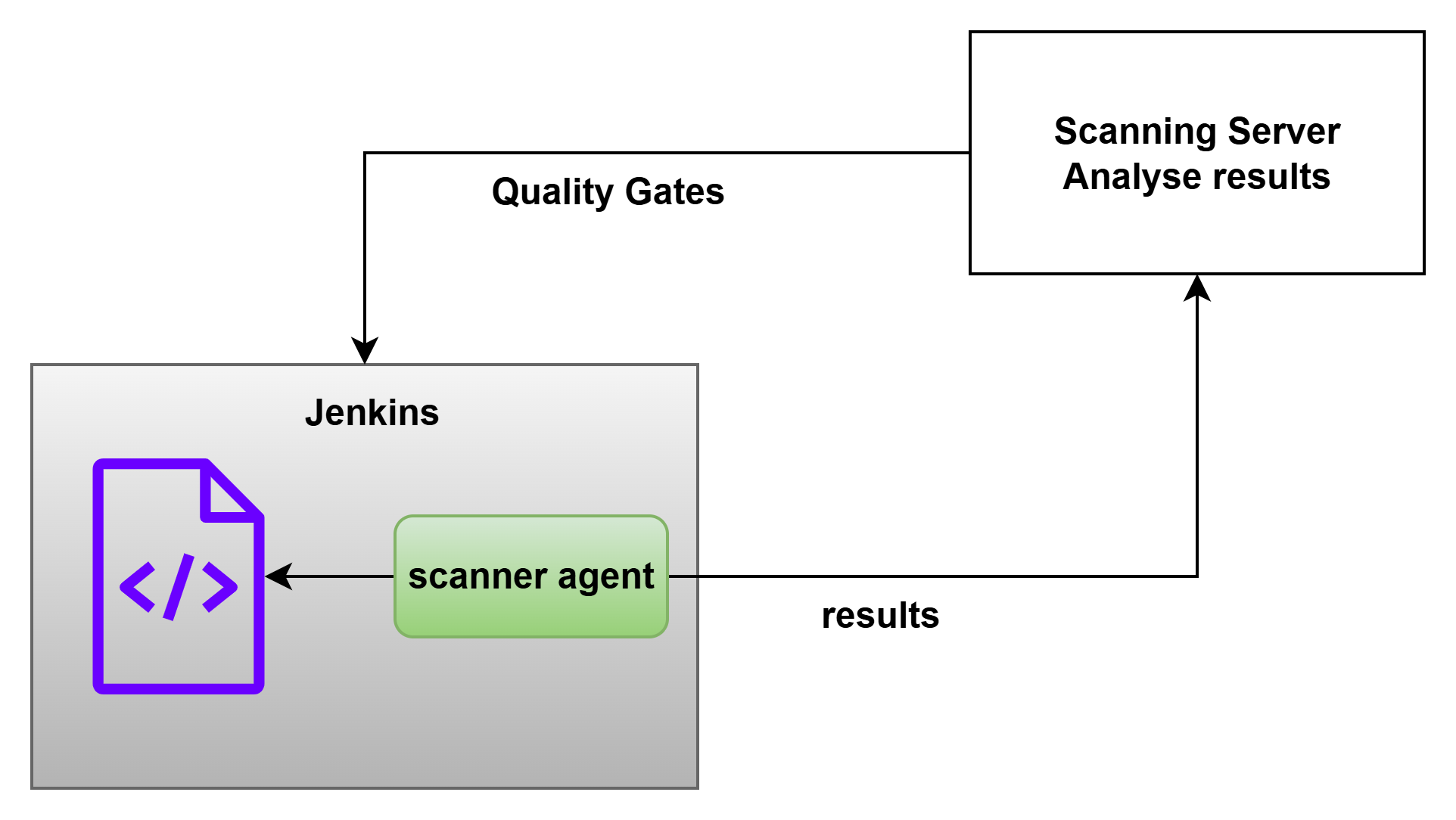
**open source library scan -->** nexus iq, github dependabot

**dynamic application secuirty testing -->** attacks on running application.. fortify **webinspect, veracode -->** Limited freetrail

**docker image scan -->** vulnerabilities in the images

--> Go to AWS accont --> Launch instances --> aws market AMI --> Sonarqube -- 0.09dollers per hour

--> create sonarqube instances



--> <https://ip> number/9000 (port number = 9000)

--> username:admin

--> password: there in instance inside

--> directly give connect it will connect.

--> Connect the jenkins

-->