BT demo project

-----------------

Required tools

--------------

VScode,Putty,Github,gitbash,Jenkins,maven,sonarqube,nexus,docker,eks(kubernetes),argocd

AWS Services-- ec2,securitygroup,application load balancer,route 53,

For starting the demo first we require a github account and repository

----------------------------------------------------------------------

1) use the following link to create a github account https://github.com/join

2) once the required details provided your account will be created now sign in to github account on the right hand side click on + button select new repository

3) Give the repository name, description and repository type private or public

To connect remote repository to your VScode require gitbash installation in your Local follow the below steps for installation and configure local repo to remote repo

---------------------------------------------------------------------------------------------------------------------------------------------------------------------

1)First download the Vs code using the following link https://code.visualstudio.com/Download

2)once download the VScode copy the setup path and then go on this PC select properties and select Environment variables select new and paste the path

3)once path setup is done you can start installation of gitbash use the following link to download gitbash https://git-scm.com/downloads

4)once git bash downloaded select run and install the git bash

5)Now open your command line or power shell type git --version, it will show up your gitbash version

6)Now create a Folder named Buchanan\_CICD\_Demo and select open with vscode once open click on file symbol which will create a file now name the file as jenkins file

7)once the file is created open the terminal in VScode run the following commands to create a local git repository

git init

8) this command will initialize local git repo now add some content in jenkins file, press ctrl+s to save now it will show M symbol means modified now your local git repo is tracking the file changes

9) Now on terminal run the following commands to add your file changes to local repository and commit the file to local repository

git commit -m "first commit"

10) Now your file changes are commited to local repository but not the remote repository to commit these changes to remote repository run the follwing commands

git branch -M main

git remote add origin repourlXXXXXXXX

git push -u origin main

11) Now when you try to push it will ask for credentials for this go github select profile there is dropdown menu now select settings in settings select developer settings

12) In developer settings select PERSONAL ACESS TOKEN (PAT) select fined grained acess, select genearate token now copy token and enter the token in gitbash

13) Now all your local commits are pushed to remote repository, now local repo integration with remote repository completed

Steps For Setting up Jenkins server

-----------------------------------

1) For setting up Jenkins server require Aws instance with ubuntu flavour of type t2.medium, setup the security inbound rules to 8080 which is default for jenkins server, setup a newkey pair and download the key pair

2) Then launch the server and connect server server with putty session using the pem key (key pair) use the default user as "ubuntu"

3) once login into server change the user to super user(root) and run the following commands for installing java and jenkins

commands for installing java jdk 11 and jenkins

--------------------------------------------------

sudo apt update -y

sudo apt install default-jre -y

java -version

sudo apt update -y

wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -

sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

sudo apt update -y

sudo add-apt-repository universe -y

sudo apt-get install jenkins -y

sudo service jenkins start

sudo service jenkins enable

sudo service status jenkins

Once the service jenkins status shows active and running then use the below command to get the Initial Admin password

cat /var/lib/jenkins/secrets/initialAdminPassword

1) Now take the public Ip of jenkins server and paste on browser using port 8080

2) It will ask for intial admin password use the password which we get earlier and login to jenkins server and setup the admin credentials by selecting username and password

3) Now it will ask for default plugins to install, Select install

4) once the plugins installation is sucessful Login to server using admin credentials which we have earlier setup

STEP--1

Creating a Pipeline project with github webhook

-----------------------------------------------

1) Now navigate to Jenkins select Dashboard, select new item, give name select pipeline project

2) once pipeline project is created select configure once open navigate to build triggers, select Github webhook trigger for scm polling

3) now on your github repository select the repository and select settings of repository

4) select webhooks in the settings and then add webhook copy the Jenkins url in payload url paste the jenkins url /github-webhook then in content type select json and select add webhook

5) Now Naviagate to pipeline section select defination in drop down select pipeline script from SCM

6) In SCM select git in repositories paste the github repository URL

7) select brain \*/main

8) select script path jenkinsfile now click on apply and save

9) Now navigate to your VSCODE add the following groovy script in jenkins file, then commit and push

10) once push is done now your jenkins pipeline will trigger with the help of webhook

Errors and Debugging

----------------------

1) Now once the jenkins file is commited the webhook trigger will happen due to this pipeline job executed but result is failed

2) Error is due to not installation of git, now login to jenkins server install the git by following command

sudo apt install get update

sudo apt install git

sudo git --version

sudo which git

GIT,JAVA,Path configure in jenkins

------------------------------------

1) once installation of git is done run which git command select the output open the jenkins navigate to globaltools

2) select git, give default name gitbt and then in path paste the output of which git command

3) now select jdk give the default name as jdkbt

4) for jdk path setting you need to login to jenkins server and then execute which java and execute the followings commands to set the path of jdk

sudo whereis java and then output is /usr/bin/java

sudo ls -l /usr/bin/java

sudo ls -l /etc/alternatives/java output will be /usr/lib/jvm/java-11-openjdk-amd64

Now setting Java\_Home variable

-------------------------------

execute the following commands to set java\_home variable

export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64

echo $JAVA\_HOME

cp ~/.bashrc ~/.bashrc.bak

echo "export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64" >> ~/.bashrc

Now verify the java\_home path using the command

tail -3 ~/.bashrc

Now javapath is set in jenkins server now copy the past now navigate to jenkins dashboard, select global tools and then select the jdk and paste the java\_home path variable in path

Jenkins Job Sucess for pulling source code from the SCM

--------------------------------------------------------

1) Now as we configure all the required tools and path of tools in jenkins server your pipeline will be sucessful

STEP--2

Building source code

--------------------

1) For building the source code we require a build tool in this case we are using a Java project will build with Maven

2) Now naviagate to jenkins server run the following commands to install maven

sudo apt update -y

sudo apt install maven -y

mvn -version

3) now you have added the build tool in jenkins server add the build stage in jenkinsfile and push the code

4) once push is done webhook will trigger job is build sucessfully check in jenkins workspace for war file

5) Now the next stage is sonarqube code analysis for this we need to setup the sonarqube server

STEP--3

Sonarqube server setup

----------------------

1) For Installing Sonarqube require t2.medium instance type, port no 9000 open in security groups,then launch the server

2) Once server is launched login to server changed to super user by using following commands

sudo su -

Install Java ( Java is the Pre Requisite Software)

--------------------------------------------------------------

yum install java-11-openjdk-devel -y

#Download the SonarqQube Server software.

cd /opt

yum install wget unzip -y

wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.6.1.59531.zip

unzip sonarqube-9.6.1.59531.zip

mv sonarqube-9.6.1.59531 sonarqube

As a good security practice, SonarQuber Server is not advised to run sonar service as a root user, so create a new user called sonar user and grant sudo access to manage nexus services as follows.

useradd sonar

Give the sudo access to sonar user

visudo

sonar ALL=(ALL) NOPASSWD: ALL

Change the owner and group permissions to /opt/sonarqube/ directory.

chown -R sonar:sonar /opt/sonarqube/

chmod -R 775 /opt/sonarqube/

su - sonar

cd /opt/sonarqube/bin/linux-x86-64/

./sonar.sh start

3) Acess the server using the public ip and then port number 9000 ip:9000

4) login to server using the default credentials username: admin password: admin and then change the credentials

STEP--4

Installing Required plugins for sonarqube

------------------------------------------

1) For integrating sonarqube with jenkins server we require some plugins in jenkins server

2) Navigate to jenkins dashboard select manage jenkins select manage plugins

3) Download and install the following plugins

qualitygate plugin,sonar geritt plugin,sonar qualitygate plugin,sonarqube scanner for jenkins,sonarqube generic coverage plugin

4) Once install the following plugins now restart the jenkins server

5) Now navigate to manage jenkins, select configure system Now select Sonarqube server

6) Add the sonarqube server url and then generate a token in sonarqube server by navigating to adminstration tab then select security tab then select token

7) On token click it then open a new window where we can generate token give the token name and select generate now copy token

8) Add the token as secret text and define token id and description click apply and save

9) Now as the sonarqube server and token defined now add the stage for sonarqube analysis in jenkins file and push the code

10) Once the code is Pushed then webhook will trigger and sonarqube code analysis will be completed

11) Now you can navigate to Sonarqube server and your code scan will be reflected in projects

STEP--5

Configuring webhook for Quality Gate Analysis

----------------------------------------------

1) For getting Quality gate analysis status to jenkins server we require to configure a webhook between jenkins server and sonarqube server

2) Copy the jenkins server Url and then navigate to sonarqube server

3) Select Administration tab in that select configuration then select webhook

4) now create a webhook giving the webhook url name and then copying the jenkins url and then select create

5) Now your jenkins is ready to get the details of quality gate analysis

6) Now write the stage for quality gate analysis in jenkins file and push the code

7) Once code is pushed webhook will trigger and then quality gate analysis stage is completed sucessfull

STEP--6

Configuring nexus repository

--------------------------------

1) For configuring nexus repo require instance t2.medium type instance with port open on 8081 in security groups

2) Launch the server with the required configuration and then install Java with the followings commands

3) Login into the nexus server as root user

#Login as a root user

sudo su -

#Install JRE 1.8

yum install java-1.8.0-openjdk -y

java -version

3) once java installation is done now install nexus with the following commands

Login as a root user

sudo su -

cd /opt

yum install tar wget -y

wget http://download.sonatype.com/nexus/3/nexus-3.15.2-01-unix.tar.gz

tar -zxvf nexus-3.15.2-01-unix.tar.gz

mv /opt/nexus-3.15.2-01 /opt/nexus

As a good security practice, Nexus is not advised to run nexus service as a root user, so create a new user called nexus and grant sudo access to manage nexus services as follows.

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

useradd nexus

Give the sudo access to nexus user

----------------------------------

visudo

nexus ALL=(ALL) NOPASSWD: ALL

Change the owner and group permissions to /opt/nexus and /opt/sonatype-work directories.

-----------------------------------------------------------------------------------------

chown -R nexus:nexus /opt/nexus-3.15.2-01

chown -R nexus:nexus /opt/sonatype-work

chmod -R 775 /opt/nexus-3.15.2-01

chmod -R 775 /opt/sonatype-work

Open /opt/nexus/bin/nexus.rc file and uncomment run\_as\_user parameter and set as nexus user.

---------------------------------------------------------------------------------------------

vi /opt/nexus/bin/nexus.rc

run\_as\_user="nexus"