**DEVOPS EXERCISE**

**Devops CICD**

Problem: as we are progressing with CICD pipeline creation using maven as an language.

Where we have define multiple stages

1. Connecting to a pvt repo
2. Building the code using maven
3. Testing the code using maven
4. Code quality and code coverage testing.
5. Integration with sonar quebe
6. Conver the application into an docker container.
7. Create an ansible pipeline to install docker.
8. And finally deploy the container using ansible

Task: build the same pipeline in Jenkins using grunt or nodejs.

**-** ansible playbook for installing docker (docker.yaml)

---

- name: Install docker

  hosts: client

  become: true

  tasks:

    - name: Update packages

      apt:

        upgrade: yes

        update\_cache: yes

    - name: Install required system packages

      apt:

        pkg:

          - apt-transport-https

          - ca-certificates

          - curl

          - software-properties-common

          - python3-pip

          - virtualenv

          - python3-setuptools

        state: latest

        update\_cache: true

    - name: ensure repository key is installed

      apt\_key:

        url: https://download.docker.com/linux/ubuntu/gpg

        state: present

    - name: ensure docker registry is available

      apt\_repository:

        repo: 'deb https://download.docker.com/linux/ubuntu bionic stable'

        state: present

    - name: ensure docker and dependencies are installed

      apt:

        name: docker-ce

        update\_cache: yes

    - name: Install Docker Module for Python

      pip:

        name: docker

    - name: ensure docker can use insecure registries

      copy:

        content: |-

          {

            "insecure-registries" : ["172.173.154.5:8085"]

          }

        dest: /etc/docker/daemon.json

    - name: restart service docker

      service:

        name: docker

        state: restarted

**-** ansible playbook for deploying image on docker (deploy-image.yaml)

---

- name: Deploy Docker Image

  hosts: client

  become: true

  vars\_files:

  - docker-cred.yml

  tasks:

    - name: Log into docker hub

      docker\_login:

        username: "{{ username }}"

        password: "{{ password }}"

        reauthorize: yes

    - name: Run Docker Conatiner

      docker\_container:

        name: node-demo

        image: asatyam78/node-demo:latest

        state: started

        pull: true

        ports:

        - "3005:3005"

    - name: Connect to docker container on port 8080 and check status 200 (Try 5 times)

      tags: test

      uri:

        url: http://localhost:3005

      register: result

      until: "result.status == 200"

      retries: 5

      delay: 5

- Jenkinsfile for this pipeline

pipeline {

    agent any

environment {

      dockerImage = ''

    }

    stages {

        stage('Poll Code Repository') {

            steps {

                git credentialsId: 'git', url: 'git@github.com:asatyam78/node-demo.git'

            }

        }

        stage('npm install') {

            steps {

                sh 'npm install'

            }

        }

        stage('test') {

            steps {

                sh 'npm run test'

            }

        }

        stage('Sonarqube'){

            steps {

                sh '''

                sudo npm install -g sonarqube-scanner

                sonar-scanner \

                    -Dsonar.projectKey=node-demo \

                    -Dsonar.sources=. \

                    -Dsonar.host.url=http://20.172.204.197:9000 \

                    -Dsonar.login=sqp\_cdf138821f3c0cd7ee232c4cbcbff97028690233

                '''

            }

        }

        stage('docker build') {

            steps{

                script{

                    dockerImage = docker.build("asatyam78/node-demo:latest")

                }

            }

        }

        stage('docker push') {

            steps {

                script {

                    withDockerRegistry(credentialsId: 'docker-hub', url: "") {

                        dockerImage.push()

                    }

                }

            }

        }

        stage('Ansible install docker'){

            steps{

                ansiblePlaybook credentialsId: 'git', disableHostKeyChecking: true, inventory: 'ansible/dev.inv', playbook: 'ansible/docker.yml'

            }

        }

        stage('Ansible deploy  image'){

            steps{

                ansiblePlaybook credentialsId: 'git', disableHostKeyChecking: true, inventory: 'ansible/dev.inv', playbook: 'ansible/deploy-image.yml', vaultCredentialsId: 'ansible-vault'

            }

        }

    }

}





