

DevOps Advanced Assessment

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

1. Create node.js project repository
2. Create dockerfile

```
FROM node:14.17.5

WORKDIR /app
COPY package.json ./
RUN npm install
COPY . .
CMD ["node", "app.js"]
EXPOSE 3000
```

3. Create ansible playbooks for installing docker

```
---
- name: Install Docker in Ubuntu system
```

```
hosts: all
become: true
tasks:
  - name: Update and upgrade apt 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: Add Docker GPG apt Key
    apt_key:
      url: https://download.docker.com/linux/ubuntu/gpg
      state: present

  - name: Add Docker Repository
    apt_repository:
      repo: 'deb https://download.docker.com/linux/ubuntu
bionic stable'
      state: present

  - name: Update apt and install docker-ce
    apt:
      name: docker-ce
      state: latest
      update_cache: yes

  - name: Install Docker Module for Python
```

```

    pip:
      name: docker

- name: Allow insecure registries
  copy:
    dest: /etc/docker/daemon.json
    content: |-
      {
        "insecure-registries" : ["40.117.186.85:8085"]
      }

- name: Enable the docker service and start
  service:
    name: docker
    enabled: yes
    state: restarted

- name: Check if docker is Installed
  shell:
    docker -v

```

4. Create ansible playbook for running docker container

```

---
- name: Deploy Docker Image
  hosts: all
  become: true
  vars_files:
    - cred.yaml
  tasks:
    - name: Log into private registry and force re-
      authorization
      docker_login:
        username: "{{ username }}"
        password: "{{ password }}"
        reauthorize: yes

```

```

- name: Running Docker Container
  docker_container:
    name: chatapp
    image: arnabs10/container-node-project:latest
    state: started
    pull: yes
    ports:
      - "3000:3000"
- name: Connect to app server on port 3000 and check
  status 200 - Try 5 times
  tags: test
  uri:
    url: http://localhost:3000
  register: result
  until: "result.status == 200"
  retries: 5
  delay: 10

```

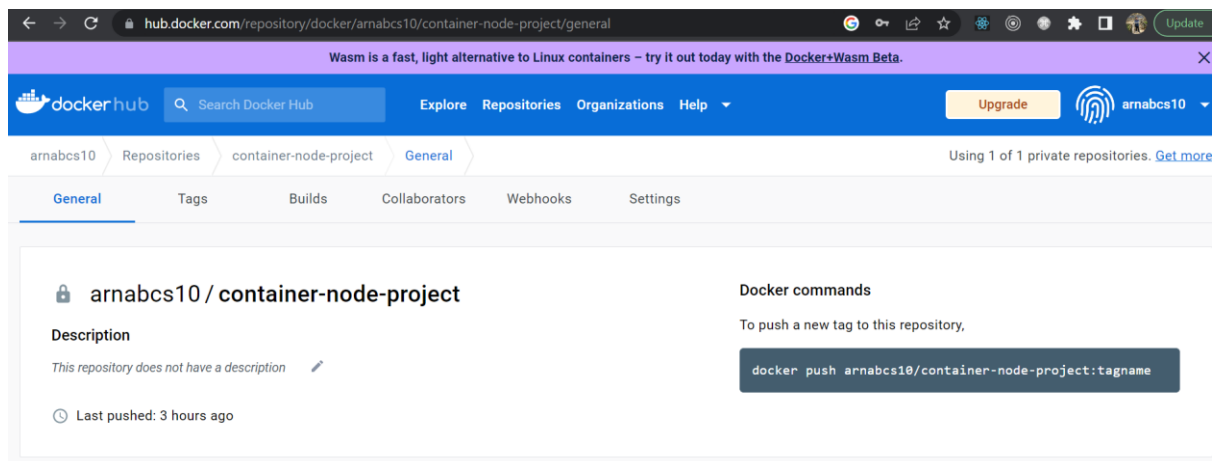
5. Encrypt docker hub credentials in cred.yaml file

```

root@ansible:~# ansible-vault create cred.yaml
New Vault password:
Confirm New Vault password:
root@ansible:~# cat cred.yaml
$ANSIBLE_VAULT;1.1;AES256
62383035346664663964623930393539323262303635643566653661356632346337656165616430
6134396434373033633266356462343130373164633636630a646632663736663737663238393165
34323639346432363133663965656233616339306539626339643230343232336361323363613732
3461653830303834660a343138343133653064623639616335643331383035616539363161323638
62623436303131303566303166376131303839393533613566653264353232613662666330323362
61613730663366626538653665636339326437323462656337323938653435383938623663353632
33353561353337373431636365343066386235626163363762663838393861333263323031363235
34653462613831313831
root@ansible:~#

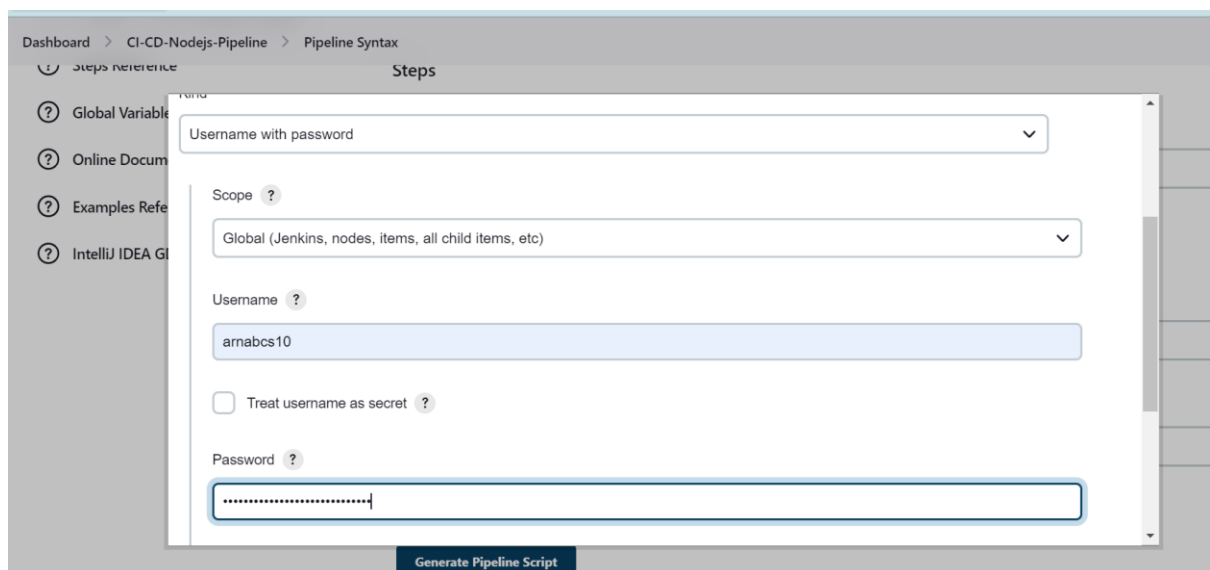
```

6. Create private repository in docker hub and create access token



Configuring Environment in Jenkins

7. Generate credentials for docker login in Jenkins. Enter username and token



8. Generate credentials for vault and enter password



9. Install sonar scanner

```
root@ansible:~# sudo npm install -g sonarqube-scanner
added 62 packages in 2s
4 packages are looking for funding
  run `npm fund` for details
root@ansible:~# ls
```

10. Create project in sonarqube server and generate key

11. Prepare Jenkinfile

```
pipeline {
    agent any
    environment {
        dockerImage = ''
    }
    stages {
        stage('Git Checkout') {
            steps {
                git credentialsId: 'git-key', url:
'git@github.com:arnabcs10/container-node-
project.git'
            }
        }
        stage('Build') {
            steps {
                sh 'npm install'
            }
        }
        stage('Test') {
            steps {
                sh 'npm run test'
            }
        }
        stage('Sonar Analysis'){
            steps {
                //withSonarQubeEnv('sonarqube-server'){
                sh '''
                sonar-scanner \
```

```

        -Dsonar.projectKey=container-node-
project \
        -Dsonar.sources=. \
        -
Dsonar.host.url=http://20.185.62.113:9000 \
        -
Dsonar.login=sqp_d9637f8158325f5abfa98ab591c9b98fd620ce27

        ...
    //}

    }
}
stage('Docker Build') {
    steps{
        script{
            dockerImage =
docker.build("arnabcs10/container-node-project:latest")
        }
    }
}
stage('Docker Push') {
    steps {
        script {
            withDockerRegistry(credentialsId:
'docker-cred', url: "") {
                dockerImage.push()
            }
        }
    }
}
stage('Ansible: Install Docker'){
    steps {
        ansiblePlaybook colored: true,
credentialsId: 'git-key', disableHostKeyChecking: true,
inventory: 'ansible/dev.inv', playbook: 'ansible/docker-
install.yaml'
    }
}

```

```

    }
    stage('Ansible: Run Docker Image'){
        steps {
            ansiblePlaybook colored: true,
credentialsId: 'git-key', disableHostKeyChecking: true,
inventory: 'ansible/dev.inv', playbook: 'ansible/deploy-
docker-image.yaml', vaultCredentialsId: 'vault-pass'
        }
    }
}

```

12. Create Jenkins pipeline

Configure

- General
- Advanced Project Options
- Pipeline**

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

git@github.com:arnabcs10/container-node-project.git

Credentials ?

root

+ Add

Advanced...

Save Apply

Dashboard > CI-CD-Nodejs-Pipeline > Configuration

Configure

- General
- Advanced Project Options
- Pipeline**

Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

Add Branch

Repository browser ?

(Auto)

Additional Behaviours

Add

Script Path ?

Jenkinsfile

☒ Lightweight checkout ?

Save Apply

13. Run Jenkins Pipeline

The screenshot shows the Jenkins Pipeline CI-CD-Nodejs-Pipeline dashboard. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, and Pipeline Syntax. The main area displays the 'Stage View' for the pipeline. A table shows the duration of each stage for two builds. Build #7 (Jan 15, 2023, 6:33 AM) is green, indicating success. Build #6 (Jan 15, 2023, 6:26 AM) is red, indicating failure. The 'Permalink' section shows the last build (#7) from 3 hours and 6 minutes ago.

	Git Checkout	Build	Test	Sonar Analysis	Docker Build	Docker Push	Ansible: Install Docker	Ansible: Run Docker Image
Average stage times: (Average full run time: ~3min 12s)	480ms	1s	1s	22s	34s	5s	57s	20s
#7 Jan 15 06:33 No Changes	484ms	1s	1s	21s	7s	3s	1min 53s	41s
#6 Jan 15 06:26 No Changes	476ms	1s	1s	24s	1min 0s	7s	1s failed	68ms failed

14. SonarQube Dashboard

The screenshot shows the SonarQube dashboard with two projects listed: 'container-node-project' and 'websocket-demo'. Both projects are marked as 'Passed'. The dashboard displays various quality metrics for each project, including Bugs, Vulnerabilities, Hotspots Reviewed, Code Smells, Coverage, Duplications, and Lines of code. The 'container-node-project' has 270 lines of code, while 'websocket-demo' has 249 lines of code.

Project	Status	Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
container-node-project	Passed	D	A	A	A	0.0%	0.0%	270
websocket-demo	Passed	C	A	A	A	25.0%	0.0%	249

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

container-node-project master

Last analysis of this Branch had 1 warning January 15, 2023 at 12:04 PM Version 1.0.0

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

Passed
All conditions passed.

New Code
Since January 15, 2023
Started 3 hours ago

Overall Code

3 Bugs Reliability D

0 Vulnerabilities Security A

0 Security Hotspots Reviewed Security Review A

15min Debt 3 Code Smells Maintainability A

0.0% Coverage on 15 Lines to cover Unit Tests -

0.0% Duplications on 270 Lines Duplicated Blocks 0

15. Application running on client machine on port 3000

```
Last login: Sun Jan 15 06:32:16 2023 from 10.0.0.4
arnab@client:~$ sudo -i
root@client:~# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
NAMES
ca7b5b40ff25   arnabcs10/container-node-project:latest "docker-entrypoint.s..." 3 hours ago    Up 3 hours    0.0.0.0:3000->3000/tcp
chatapp
root@client:~# docker images
REPOSITORY              TAG         IMAGE ID      CREATED        SIZE
arnabcs10/container-node-project   latest     1f31b85e53f3   3 hours ago    1GB
root@client:~# curl localhost:3000
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-scale=1.0">
    <title>Spring Boot WebSocket Chat Application | CalliCoder</title>
    <link rel="stylesheet" href="/styles/styles.css" />
  </head>
  <body>
    <noscript>
      <h2>Sorry! Your browser doesn't support Javascript</h2>
    </noscript>
  </body>
</html>
```

