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

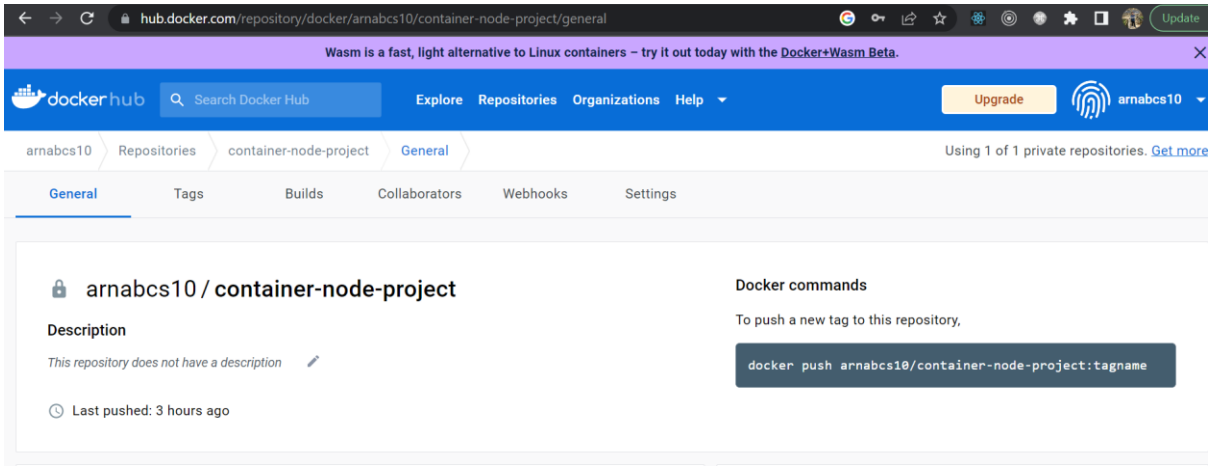
```
---
- name: Deploy Docker Image
  hosts: all
  become: true
  vars_files:
    - cred.yaml
  tasks:
    - name: Log into private registry and force re-
authorization
      docker_login:
        username: "{{ username }}"
        password: "{{ password }}"
        reauthorize: yes
```

```
    - name: Running Docker Conatiner
      docker_container:
        name: chatapp
        image: arnabcs10/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

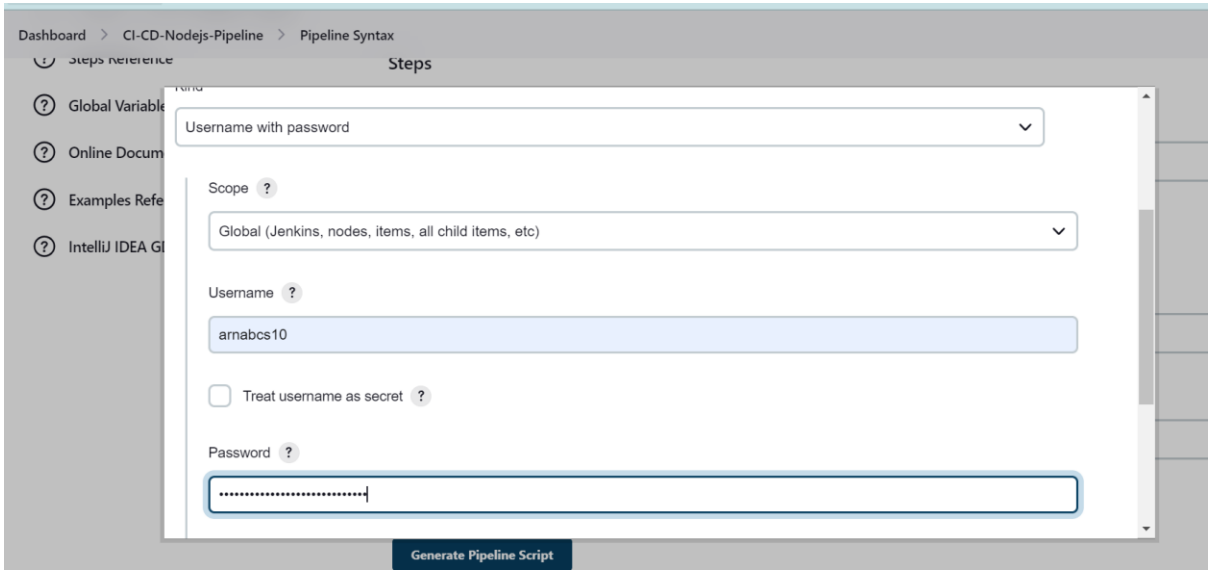
```
root@ansible:~# ansible-vault create cred.yaml
New Vault password:
Confirm New Vault password:
root@ansible:~# cat cred.yaml
$ANSIBLE_VAULT;1.1;AES256
62383035346664663964623930393539323262303635643566653661356632346337656165616430
61343964343730336332663564623431303731646336636630a64663266373666373766323839316 5
34323639346432363133663965656233616339306539626339643230343232336361323363613732
34616538303038346 60a34313834313365306462363961633564333138303561653936316132363 8
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

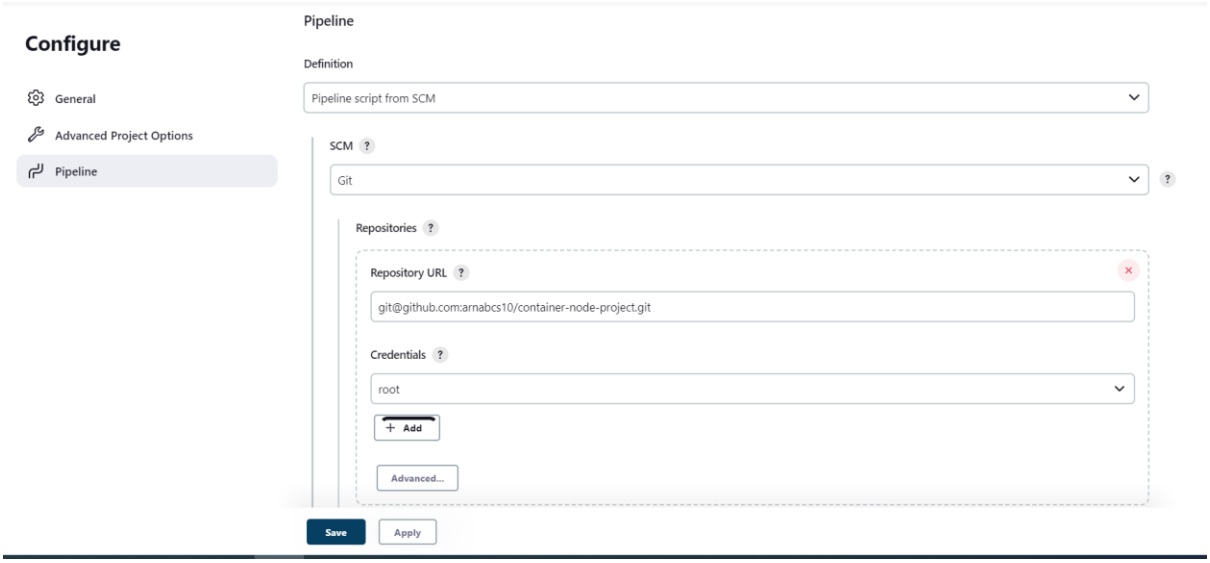
11. Prepare Jenkinfile

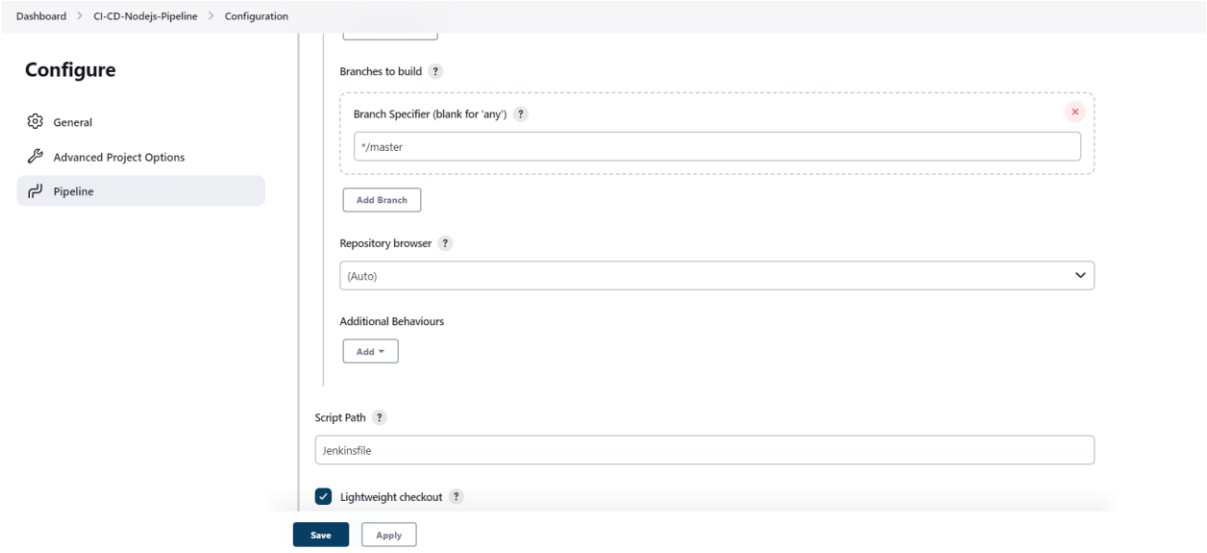
```
pipeline {
    agent any
    environment {
        dockerImage = ''
    }
    stages {
        stage('Git Checkout') {
            steps {
                git credentialsId: 'git-key', url:
'git@github.com:arnabcs10/container-node-
project.git'
            }
        }
        stage('Build') {
            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 colorized: true,
credentialsId: 'git-key', disableHostKeyChecking: true,
inventory: 'ansible/dev.inv', playbook: 'ansible/docker-
install.yaml'
            }
```

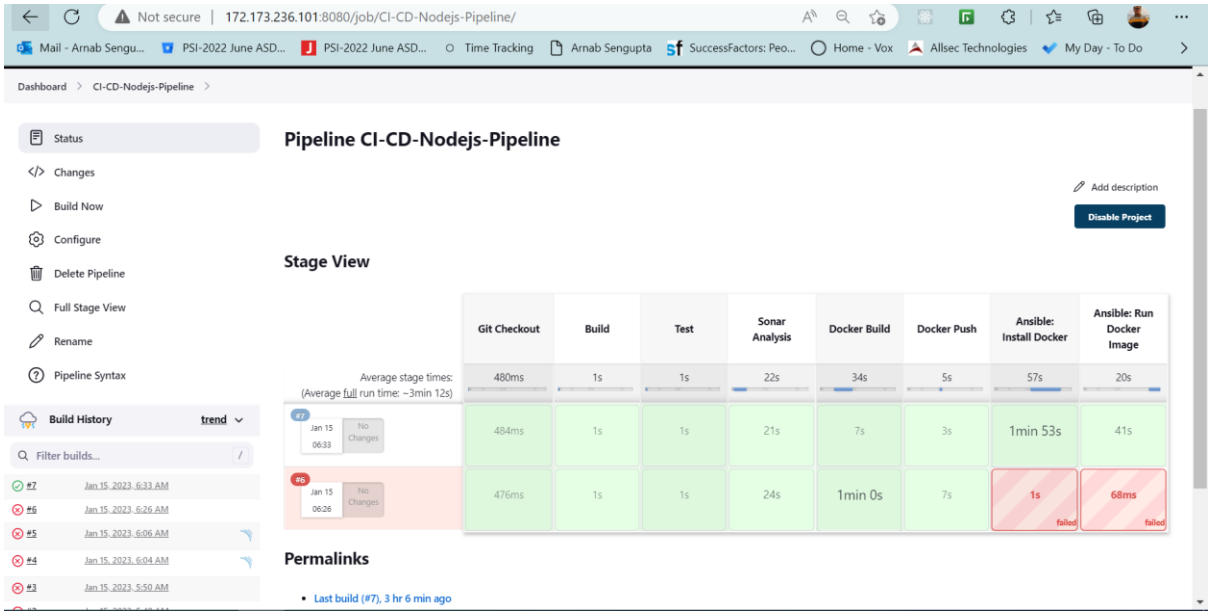
```
        }
        stage('Ansible: Run Docker Image'){
            steps {
                ansiblePlaybook colorized: true,
credentialsId: 'git-key', disableHostKeyChecking: true,
inventory: 'ansible/dev.inv', playbook: 'ansible/deploy-
docker-image.yaml', vaultCredentialsId: 'vault-pass'
       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

12. Create Jenkins pipeline

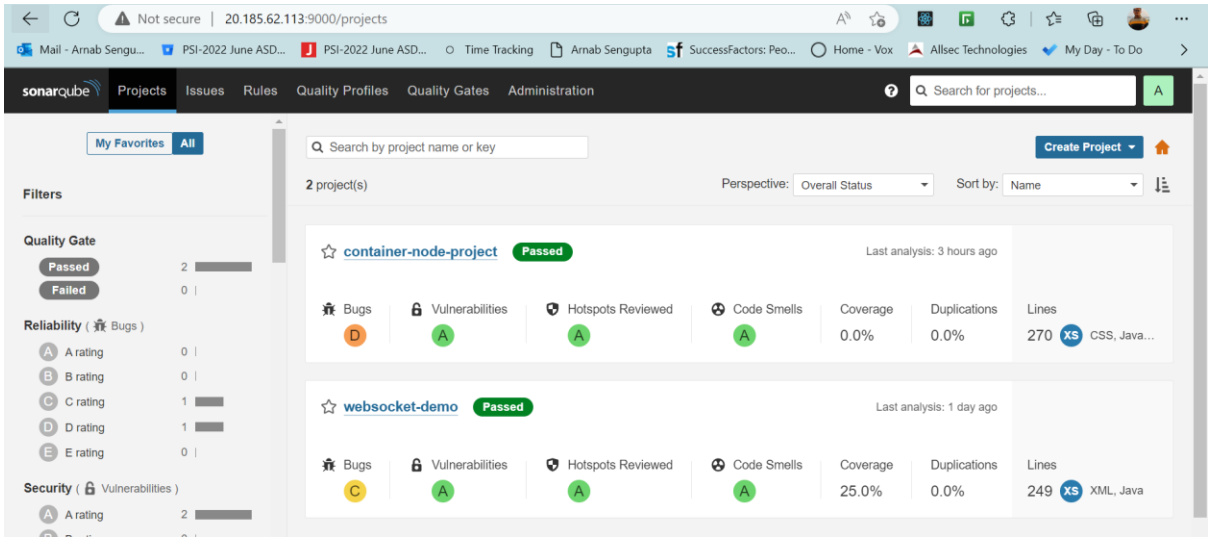


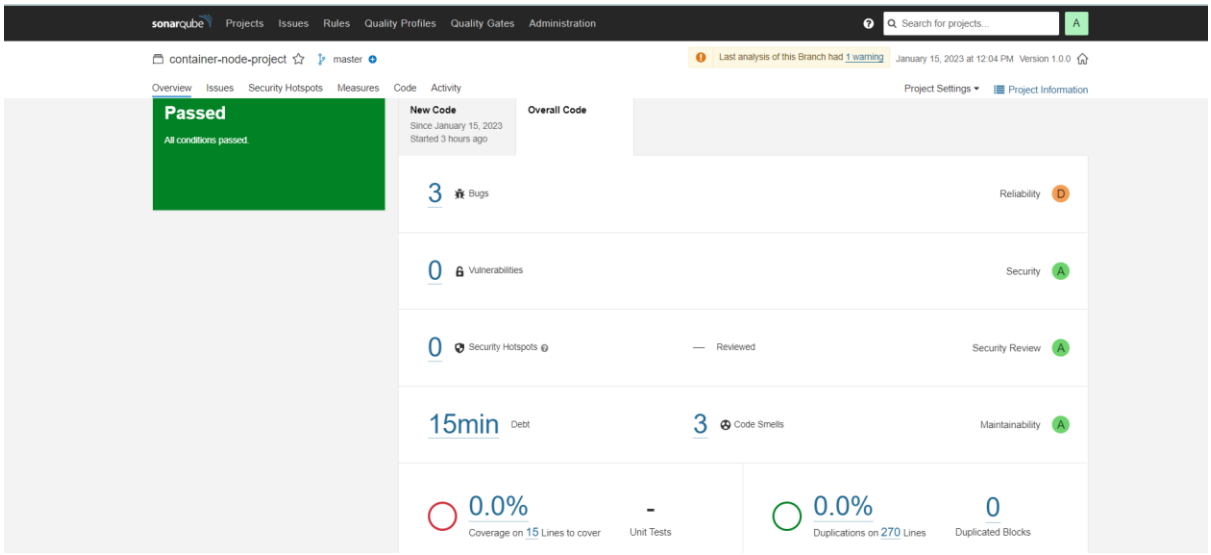


13. Run Jenkins Pipeline



14. SonarQube Dashboard





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>
```