Jenkinsfile-cd

```
def url
def response
def commitId
pipeline {
  agent any
  parameters {
    string(name:'COMMIT_ID', defaultValue: '', description: '')
  }
  environment {
    GROUP_ID = 'next.quantum.solutions'
    ARTIFACT_ID = 'next-js'
    NEXUS_URL = 'http://172.20.0.10:8081'
    REPOSITORY = 'maven-releases'
    NEXUS_REGISTRY = '172.20.0.10:8085/'
    NEXUS_CREDENTIALS = credentials('nexus-jenkins')
    NEXUS_USERNAME = 'wissem'
    NEXUS_PASSWORD = 'wissem'
  }
  stages {
    stage('Check Nexus Tag') {
      steps {
        script {
          commitId = 'params.COMMIT_ID'
          echo "Commit ID: ${params.COMMIT_ID}"
          url = "${NEXUS_URL}/repository/maven-releases/next/quantum/soluttions/next-js/$
{params.COMMIT_ID}/next-js-${params.COMMIT_ID}.zip"
          response = sh(script: "curl -sS -X GET ${url}", returnStdout: true)
          echo "response: ${response}"
          if (response.contains("404 - Nexus Repository Manager")) {
            error "Version tag '${params.COMMIT_ID}' does not exist in Nexus for artifact '$
{GROUP_ID}:${ARTIFACT_ID}'"
          }
          else {
            echo "Version tag '${params.COMMIT_ID}' exists in Nexus for artifact '${GROUP_ID}:$
{ARTIFACT_ID}'"
          }
        }
      }
    }
    stage('pull code from Nexus') {
      steps {
        script {
          url = "${NEXUS_URL}/repository/maven-releases/next/quantum/soluttions/next-js/$
{params.COMMIT_ID}/next-js-${params.COMMIT_ID}.zip"
            sh "curl -L -o next.zip ${url}"
        }
      }
    }
```

```
    stage ('unzip') {
      steps {
        sh "apt-get install unzip"
        sh "unzip -o next.zip"
      }
    }
    stage('Build Docker Image') {
      steps {
        // sh "chown -R jenkins:jenkins ."
        sh "docker build -f dockerfile -t hamza:'${params.COMMIT_ID}' ."
      }
    }
    stage('Push to Nexus') {
      steps {
        withCredentials([usernamePassword(credentialsId: 'nexus-jenkins', usernameVariable:
'NEXUS_USERNAME', passwordVariable: 'NEXUS_PASSWORD')]) {
          sh "docker login -u wissem  --password wissem 172.20.0.10:8085/repository/docker-repo"
          sh "docker tag hamza:'${params.COMMIT_ID}'
172.20.0.10:8085/repository/docker-repo/hamza:'${params.COMMIT_ID}'"
          sh "docker push 172.20.0.10:8085/repository/docker-repo/hamza:'${params.COMMIT_ID}'"
        }
      }
    }
    stage('run container'){
      steps {
        sh " docker run -dit --name nextjs  -p 8888:80 hamza:'${params.COMMIT_ID}'"
      }
    }
  }
  post {
    always {
      cleanWs()
    }
  }
}
```

dockerfile

```
FROM node:16.19.1
USER root


# Install Nginx
RUN apt-get update && \
    apt-get install -y nginx

# Remove default Nginx configuration
RUN rm /etc/nginx/sites-enabled/default

# Copy custom Nginx configuration
COPY ./nginx.conf /etc/nginx/conf.d/default.conf

# Create a directory for the app
WORKDIR /var/www/html

# Copy package.json and package-lock.json
COPY ["package.json", "package-lock.json*", "./"]

# Install app dependencies
RUN npm install

# Copy app file
ADD ./  /var/www/html
COPY ./node_modules /var/www/html


RUN chmod -R 777 /var/www/html
RUN chown -R node:node /var/www/html
EXPOSE 80

# Start Nginx and the app
CMD service nginx start && npm start
```

jenkinsfile-ci

```
def COMMIT_ID
pipeline {
  agent any
  tools {nodejs "NodeJS"}
  stages {
    stage('Checkout') {
      steps {
        script {
          checkout([$class: 'GitSCM',
            branches: [[name: "main"]],
            doGenerateSubmoduleConfigurations: false,
            extensions: [],
            submoduleCfg: [],
            userRemoteConfigs: [[credentialsId: 'wiss-git', url:
'https://gitlab.com/wissemsghaier2000/project_next.git']]
          ])
          sh "git rev-parse --short HEAD > commit_hash.txt"
          COMMIT_ID = readFile('commit_hash.txt').trim()
          echo "Commit Hash: ${COMMIT_ID}"
        }
      }
    }
    stage ('istall package ') {
      steps {
        nodejs(nodeJSInstallationName: 'NodeJS'){
          sh ' npm install'
        }
      }
    }
    stage ('build ') {
      steps {
        nodejs(nodeJSInstallationName: 'NodeJS'){
          sh " npm run build"
        }
      }
    }
    /*stage ('test') {
      steps{
        nodejs(nodeJSInstallationName: 'NodeJS'){
          sh " npm run test "
        }
      }
    }
    */
    stage('Code Quality Check via SonarQube') {
      steps {
        script {
          def scannerHome = tool 'sonarqube-scanner';
```

```
        withSonarQubeEnv("SonarQube") {
          sh "${tool("sonarqube-scanner")}/bin/sonar-scanner \
          -Dsonar.projectName=test-app \
          -Dsonar.projectKey=test-app\
          -Dsonar.sources=. \
          -Dsonar.host.url=http://172.20.0.1:9001/  \
          -Dsonar.language=js \
          -Dsonar.login=squ_3aef020e30f6b636447ce330b479af53e3c72423"
        }
       }
      }
    }
    stage("Quality Gate") {
      steps {
        sleep 60
        waitForQualityGate abortPipeline: true
      }
    }
    stage(" Publish to Nexus Repository Manager") {
      steps {
        sh "apt-get install zip"
        sh "zip -r project_next.zip ."
        sh "curl -v -u wissem:wissem --upload-file project_next.zip
http://172.20.0.10:8081/repository/maven-releases/next/quantum/soluttions/next-js/$
{COMMIT_ID}/next-js-${COMMIT_ID}.zip"
      }
    }
  }
  post {
    always {
      cleanWs()
    }
  }
}
```