Jenkinsfile deploy

1- stage : Check Nexus Tag

A- Description:

- the stage used to check commit id exist in nexus repository or not .
- The response variable used to execute curl to get the artifact url from the Nexus repository.
- if response contains 404 output message commit id not exist else output exist and continue pipeline

B-code :

```
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}'"
        }
      }
    }
```

2- stage : pull code from Nexus

A- Description:

- but of stage is execute curl command to download the artifact from the Nexus repository with the specific name .

B-code :

```
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}"
        }
      }
```

3- stage : unzip

A- Description:

- the stage used to unzip the artifact because it is download file.zip

B-code :

```
stage ('unzip') {
      steps {
        sh "apt-get install unzip"
        sh "unzip -o next.zip"
      }
```

4- stage : build docker image :

A- Description:

- the stage used to build docker image with tag

B-code :

```
stage('Build Docker Image') {
      steps {
        // sh "chown -R jenkins:jenkins ."
        sh "docker build -f dockerfile -t next-js:'${params.COMMIT_ID}' ."
      }
```

5- stage : Push to Nexus

A- Description:

- the stage used to push docker image to nexus repository with specific tag

B-code :

```
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 next-js:'${params.COMMIT_ID}'
172.20.0.10:8085/repository/docker-repo/next-js:'${params.COMMIT_ID}'"
          sh "docker push 172.20.0.10:8085/repository/docker-repo/next-js:'$
{params.COMMIT_ID}'"
        }
      }
}
```

6- stage : run container

A- Description:

- this stage alow to run container with number of port and image that can be buided

B-Code :

```
 stage('run container'){
      steps {
        sh " docker run -dit --name next  -p 9999:80 next-js:'${params.COMMIT_ID}'"
      }
}
```

jenkinsfile-cd

```
def url
def response
def commitId
pipeline {
  agent any
  parameters {
    string(name:'COMMIT_ID', defaultValue: '', description: '')
  }
  environment {
    GROUP_ID = 'quantum.solutions.io'
    ARTIFACT_ID = 'next-gen-radio'
    NEXUS_URL = 'http://172.20.0.10:8081'
    REPOSITORY = 'maven-releases'
    NEXUS_REGISTRY = '172.20.0.10:8082/'
    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/quantum/solutions/io/next-
gen-radio/${params.COMMIT_ID}/next-gen-radio-${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}:${ARTIFACT_ID}'"
         }
       }
     }
     post {
       success {
         slackSend (color: 'good', message: "check to nexus with '$
{params.COMMIT_ID}' succeeded!")
       }
```

```
      failure {
        slackSend (color: 'danger', message:"check to nexus with '$
{params.COMMIT_ID}' failed!")
      }
    }
  }
  stage('pull code from Nexus') {
    steps {
      script {
        url = "${NEXUS_URL}/repository/maven-releases/quantum/solutions/io/next-
gen-radio/${params.COMMIT_ID}/next-gen-radio-${params.COMMIT_ID}.zip"
        sh "curl -L -o project.zip ${url}"
      }
    }
    post {
      success {
        slackSend (color: 'good', message: "pull from nexus  succeeded!")
      }
      failure {
        slackSend (color: 'danger', message: "pull from nexus   failed!")
      }
    }
  }
  stage ('unzip') {
    steps {
      sh "apt-get install unzip"
      sh "unzip -o project.zip"
    }
    post {
      success {
        slackSend (color: 'good', message: "unzip succeeded!")
      }
      failure {
        slackSend (color: 'danger', message: "unzip   failed!")
      }
    }
  }
  stage('Build Docker Image') {
    steps {
      // sh "chown -R jenkins:jenkins ."
      sh "docker build -f dockerfile -t hamza:'${params.COMMIT_ID}' ."
    }
    post {
      success {
        slackSend (color: 'good', message: "Build Docker Image succeeded!")
```

```
        }
        failure {
          slackSend (color: 'danger', message: "Build Docker Image  failed!")
        }
      }
    }
    stage('Push to Nexus') {
      steps {
        withCredentials([usernamePassword(credentialsId: 'nexus-jenkins',
usernameVariable: 'NEXUS_USERNAME', passwordVariable:
'NEXUS_PASSWORD')]) {
          sh "docker login -u wissem  --password wissem
172.20.0.10:8082/repository/dockerhosted-repo"
          sh "docker tag hamza:'${params.COMMIT_ID}'
172.20.0.10:8082/repository/dockerhosted-repo/hamza:'${params.COMMIT_ID}'"
          sh "docker push 172.20.0.10:8082/repository/dockerhosted-repo/hamza:'$
{params.COMMIT_ID}'"
        }
      }
      post {
        success {
          slackSend (color: 'good', message: "Push image to Nexus  succeeded!")
        }
        failure {
          slackSend (color: 'danger', message: "Push image to Nexus failed!")
        }
      }
    }
    stage('run container'){
      steps {
        sh " docker run -dit --name laravel -p 8888:80 hamza:'$
{params.COMMIT_ID}'"
      }
      post {
        success {
          slackSend (color: 'good', message: "run container  succeeded!")
        }
        failure {
          slackSend (color: 'danger', message: "run container failed!")
        }
      }
    }
  }
  post {
    always {
```

```
      cleanWs()
    }
    success {
      slackSend channel: '#jenkins-alerts-pfe-2023', message: " deploy $
{currentBuild.result} for job ${env.JOB_NAME} #${env.BUILD_NUMBER}
(duration: ${currentBuild.durationString}). Check out the deploy at $
{env.BUILD_URL}"
      emailext (
        to: 'wissemsghaier2000@gmail.com',
        subject: "Job '${env.JOB_NAME}'",
        body: "deploy  ${currentBuild.result} for job  ${env.JOB_NAME} at  $
{env.BUILD_URL} and the build number $BUILD_NUMBER"
      )
    }
    failure {
      slackSend channel: '#jenkins-alerts-pfe-2023', message: "deploy $
{currentBuild.result} for ${env.JOB_NAME} #${env.BUILD_NUMBER} (duration:
${currentBuild.durationString}). Check out the deploy at ${env.BUILD_URL}"
      emailext (
        to: 'wissemsghaier2000@gmail.com',
        subject: "Job '${env.JOB_NAME}'",
        body: "deploy  ${currentBuild.result} for job  ${env.JOB_NAME} at  $
{env.BUILD_URL} and the build number $BUILD_NUMBER"
      )
    }
  }
}
```