Q1:

1.This is my Jenkins file.. for over view I will give it in vscode and I will show

```
pipeline {
    agent any
    environment{
        IMAGENAME='jithu145/nodejs1'
        TAG='latest'
    }

    stages {
        stage('git') {
            steps {
                git url: 'https://github.com/devopsexamus/nodejsapp', branch: 'main'
            }
        }
        stage('test') {
```

```
stage('test') {
    steps {
        sh'npm test'
    }
}
stage('docker login&push&logout') {
    steps {
        withCredentials([usernamePassword(credentialsId:'docker-creds',usernameVariable:'D
            script{
              sh'''
              echo '$DOCKERPASS' | docker login -u '$DOCKER_USER' --password-stdin
              docker build -t $IMAGENAME:$TAG .
              docker push $IMAGENAME:$TAG
              docker logout
              '''
```

```
14
15
16
17
18
19
20
21  lentialsId:'docker-creds',usernameVariable:'DOCKER_USER',passwordVariable:'DOCKER_PASS')]){
22
23
24  gin -u '$DOCKER_USER' --password-stdin
25  G .
26
27
28
```

```
    }
}
stage('kubernetes') {
    steps {
        withCredentials([file(credentialsId: 'kube-creds', variable: 'kube-creden')]) {
            script{
              sh'''
              echo $KUBECONFIG=$kube_creden
              kubectl create deployment jithendra --image="$IMAGENAME":"$TAG" --dry-run=cl
              kubectl apply -f deployment.yaml
              kubectl expose deployment jithendra --port=80 --target-port=3000 --type=Node
              kubectl apply -f >service.yaml
```

```
32
33
34
35
36  edentialsId: 'kube-creds', variable: 'kube-creden')]) {
37
38
39  kube_creden
40  loyment jithendra --image="$IMAGENAME":"$TAG" --dry-run=client -o yaml > deployment.yaml
41  deployment.yaml
42  loyment jithendra --port=80 --target-port=3000 --type=NodePort -o yaml >service.yaml
43  service.yaml
44  lout deployment/jithendra
45
46
```

```
pipeline {
    agent any
    environment{
        IMAGENAME='jithu145/nodejs1'
        TAG='latest'
    }

    stages {
        stage('git') {
            steps {
                git url: 'https://github.com/devopsexamus/nodejsapp', branch: 'main'
            }
        }
        stage('test') {
            steps {
                sh'npm test'
            }
        }
        stage('docker login&push&logout') {
            steps {
                withCredentials([usernamePassword(credentialsId:'docker-creds',usernameVariable:'DOCKER_USER',passwordVariable:'D
                    script{
                      sh'''
                      echo 'Jithendra@123' | docker login -u 'jithu145' --password-stdin
                      docker build -t $IMAGENAME:$TAG .
                      docker push $IMAGENAME:$TAG
                      docker logout
                      '''
    }
```



```
        stage('docker login&push&logout') {
            steps {
                withCredentials([usernamePassword(credentialsId:'docker-creds',usernameVariable:'DOCKER_USER',passwordVariable:'D
                    script{
                      sh'''
                      echo 'DOCKER_PASSWORD' | docker login -u 'DOCKER_USER' --password-stdin
                      docker build -t $IMAGENAME:$TAG .
                      docker push $IMAGENAME:$TAG
                      docker logout
                      '''
    }
    }

        }
    }
        stage('kubernetes') {
            steps {
                withCredentials([file(credentialsId: 'kube-creds', variable: 'kube-creden')]) {
                    script{
                      sh'''
                      echo $KUBECONFIG=$kube_creden
                      kubectl create deployment jithendra --image="$IMAGENAME":"$TAG" --dry-run=client -o yaml > deployment.yaml
                      kubectl apply -f deployment.yaml
                      kubectl expose deployment jithendra --port=80 --target-port=3000 --type=NodePort -o yaml >service.yaml
                      kubectl apply -f service.yaml
                      kubectl status rollout deployment/jithendra
                      '''
    }}
```

JENKINS LOGS:

```
Started by user admin
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/jithu
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (git)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/jithu/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/devopsexamus/nodejsapp # timeout=10
Fetching upstream changes from https://github.com/devopsexamus/nodejsapp
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- https://github.com/devopsexamus/nodejsapp
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 39641ae2c4928bead3775ac2f4072add7160e9bf (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 39641ae2c4928bead3775ac2f4072add7160e9bf # timeout=10
```



```
[Pipeline] {
[Pipeline] sh
+ docker login -u jithu145 --password-stdin
+ echo ****
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
+ docker build -t jithu145/nodejs1:latest .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  79.87kB

Step 1/7 : FROM node:current-alpine
 ---> 70c307a00bc8
Step 2/7 : WORKDIR /app
 ---> Using cache
 ---> 756319229cff
Step 3/7 : COPY package*.json ./
 ---> Using cache
```

```
Step 2/7 : WORKDIR /app
 ---> Using cache
 ---> 756319229cff
Step 3/7 : COPY package*.json ./
 ---> Using cache
 ---> 627106802476
Step 4/7 : RUN npm install --omit=dev && npm install express
 ---> Using cache
 ---> 5dd9bc0e0ea7
Step 5/7 : COPY . .
 ---> 03886bfb84eb
Step 6/7 : EXPOSE 3000
 ---> Running in a9fcefbeddee
 ---> Removed intermediate container a9fcefbeddee
 ---> a78b59b6c6e2
Step 7/7 : CMD [ "node", "app.js" ]
 ---> Running in 56a1661584d1
 ---> Removed intermediate container 56a1661584d1
 ---> d46381e01e04
Successfully built d46381e01e04
Successfully tagged jithu145/nodejs1:latest
+ docker push jithu145/nodejs1:latest
The push refers to repository [docker.io/jithu145/nodejs1]
 ---> d46381e01e04
Successfully built d46381e01e04
Successfully tagged jithu145/nodejs1:latest
+ docker push jithu145/nodejs1:latest
The push refers to repository [docker.io/jithu145/nodejs1]
6f291bc972a7: Preparing
20797d3116bf: Preparing
4efadf090719: Preparing
b03edecd63c1: Preparing
0b03a9872866: Preparing
d6d1618ae0ad: Preparing
3be867aca878: Preparing
fd2758d7a50e: Preparing
d6d1618ae0ad: Waiting
3be867aca878: Waiting
fd2758d7a50e: Waiting
6f291bc972a7: Retrying in 5 seconds
6f291bc972a7: Retrying in 4 seconds
6f291bc972a7: Retrying in 3 seconds
b03edecd63c1: Layer already exists
4efadf090719: Layer already exists
0b03a9872866: Layer already exists
```

```
3be867aca878: Waiting
fd2758d7a50e: Waiting
6f291bc972a7: Retrying in 5 seconds
6f291bc972a7: Retrying in 4 seconds
6f291bc972a7: Retrying in 3 seconds
b03edecd63c1: Layer already exists
4efadf090719: Layer already exists
0b03a9872866: Layer already exists
20797d3116bf: Layer already exists
6f291bc972a7: Retrying in 2 seconds
d6d1618ae0ad: Layer already exists
3be867aca878: Layer already exists
6f291bc972a7: Retrying in 1 second
fd2758d7a50e: Layer already exists
6f291bc972a7: Pushed
latest: digest: sha256:b48ae17bb1b1d7a72c2c03580e180b7cf847e4bb82c9672451536b6ccfc10fbb size: 1991
+ docker logout
Removing login credentials for https://index.docker.io/v1/
[Pipeline] }
```


Running into ratelimit :

```
Events:
  Type     Reason     Age                  From               Message
  ----     ------     ----                 ----               -------
  Normal   Scheduled  2m26s                default-scheduler  Successfully assigned default/jithendra-dc44dc8dd-5njkf to kubeworker-vm
  Normal   BackOff    98s (x2 over 2m16s)  kubelet            Back-off pulling image "jithu145/nodejs1:latest"
  Warning  Failed     98s (x2 over 2m16s)  kubelet            Error: ImagePullBackOff
  Normal   Pulling    87s (x3 over 2m25s)  kubelet            Pulling image "jithu145/nodejs1:latest"
  Warning  Failed     9s (x3 over 2m16s)   kubelet            Failed to pull image "jithu145/nodejs1:latest": failed to pull and unpack image "docker.io/jith
u145/nodejs1:latest": failed to copy: httpReadSeeker: failed open: unexpected status code https://registry-1.docker.io/v2/jithu145/nodejs1/manifests/sha256:b
48ae17bb1b1d7a72c2c03580e180b7cf847e4bb82c9672451536b6ccfc10fbb: 429 Too Many Requests - Server message: toomanyrequests: You have reached your unauthenticat
ed pull rate limit. https://www.docker.com/increase-rate-limit
  Warning  Failed     9s (x3 over 2m16s)   kubelet            Error: ErrImagePull
```


As dockerfile got build and the Kubernetes stage is stopped because of the pull rate limit:

See the Kubernetes stage which is correct which will definitely run kindly check..

So to show the ouput that the image is working I did it with docker container please check with that:

Pulling the image that I gave in the Jenkins file:

```
root@kubemaster-vm:/# docker pull jithu145/nodejs1:latest
latest: Pulling from jithu145/nodejs1
Digest: sha256:b48ae17bb1b1d7a72c2c03580e180b7cf847e4bb82c9672451536b6ccfc10fbb
Status: Downloaded newer image for jithu145/nodejs1:latest
docker.io/jithu145/nodejs1:latest
```

Running container with that image:

```
root@kubemaster-vm:/# docker run -d -p 80:3000 --name nodejs jithu145/nodejs1:latest
ae1c0fcf4a26093122364afc31416c4dc40318dee2d9f2cad3271b8e10bb3b91
```

Went in to the docker checked with the localhost:3000 because it is node.js location its working

```
oot@kubemaster-vm:/# docker exec -it ae1c /sh
CI runtime exec failed: exec failed: unable to start container process: exec: "/sh": stat /sh: no such file or directory: unknown
oot@kubemaster-vm:/# docker exec -it ae1c /bin/sh
app # ls
EADME.md          app.js            dockerfile        node_modules      package-lock.json  package.json
app # curl localhost:3000
bin/sh: curl: not found
app # wget localhost:3000
onnecting to localhost:3000 ([::1]:3000)
aving to 'index.html'
ndex.html           100% |********************************************************************************************************************
index.html' saved
```

Tried with the port mapped to 80 also that too worked.

```
root@kubemaster-vm:/# curl localhost:80
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Vote for Your Favorite Programming Language</title>
  <style>
    body {
      display: flex;
      flex-direction: column;
      align-items: center;
      justify-content: center;
      height: 100vh;
      margin: 0;
      padding: 0;
    }

    form {
      text-align: center;
    }

    ul {
      text-align: center;
      list-style-type: none;
      padding: 0;
    }
  </style>
</head>
<body>
  <h1>Vote for Your Favorite Programming Language</h1>
  <form action="/vote" method="POST">
    <label>
      <input type="radio" name="language" value="JavaScript"> JavaScript
    </label>
```

To show the files of kubernetes:

Deployment.yaml file:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: jithendra
  name: jithendra
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jithendra
  template:
    metadata:
      labels:
        app: jithendra
    spec:
      containers:
      - image: jithu145/nodejs1:latest
        name: nodejs1
```

Service.yaml:

```
  GNU nano 4.8
apiVersion: v1
kind: Service
metadata:
  name: jithendra
spec:
  selector:
    app: jithendra
  ports:
  - port: 80
    targetPort: 3000
```