# React Application Deployment using Docker

## Problem Statement:

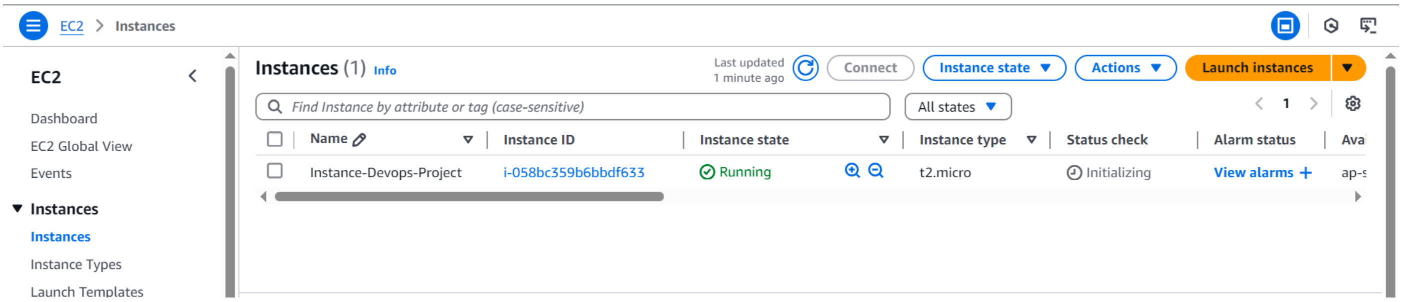
Deployment of React application using docker with multi stage docker build approach.

## Solution:

### Requirement:

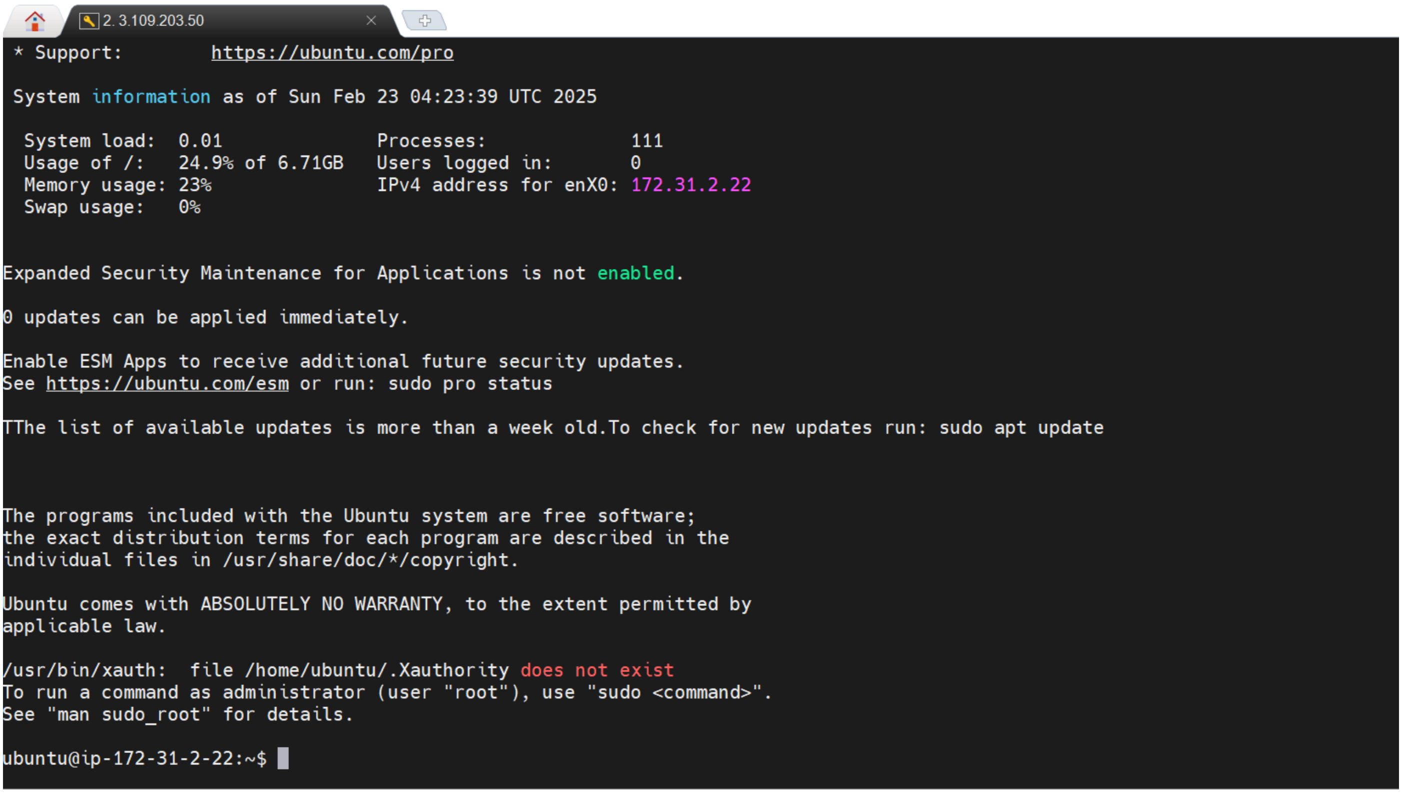
1. AWS Cloud
2. AWS EC2 instance
3. Git & GitHub
4. Docker
5. Java
6. Jenkins

## Step:1 – Launching an EC2 instance:



| □ | Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Ava |
|---|---|---|---|---|---|---|---|
| □ | Instance-Devops-Project | i-058bc359b6bbdf633 | ✓ Running | t2.micro | Initializing | View alarms + | ap-s |

## Step2: Connect to the instance



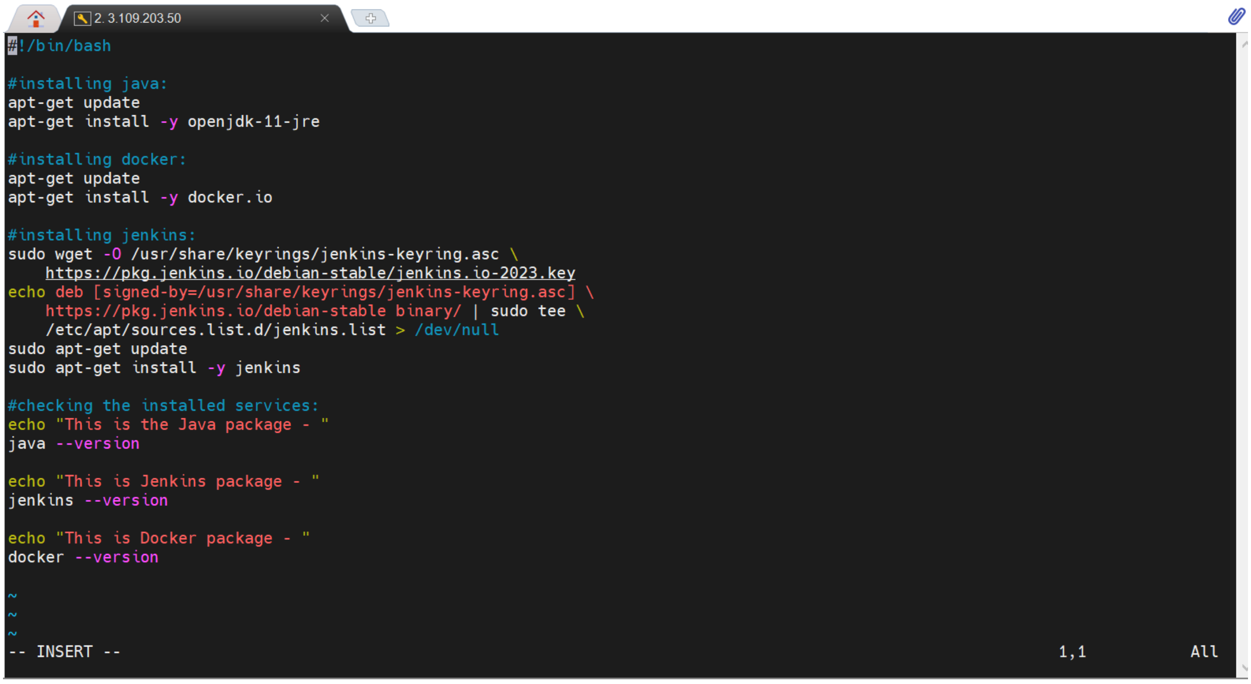
```
 * Support:        https://ubuntu.com/pro

 System information as of Sun Feb 23 04:23:39 UTC 2025

  System load:  0.01              Processes:             111
  Usage of /:   24.9% of 6.71GB   Users logged in:       0
  Memory usage: 23%               IPv4 address for enX0: 172.31.2.22
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

TThe list of available updates is more than a week old.To check for new updates run: sudo apt update



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

/usr/bin/xauth:  file /home/ubuntu/.Xauthority does not exist
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-2-22:~$ 
```

**Present working directory: /home/ubunutu**

# React Application Deployment using Docker

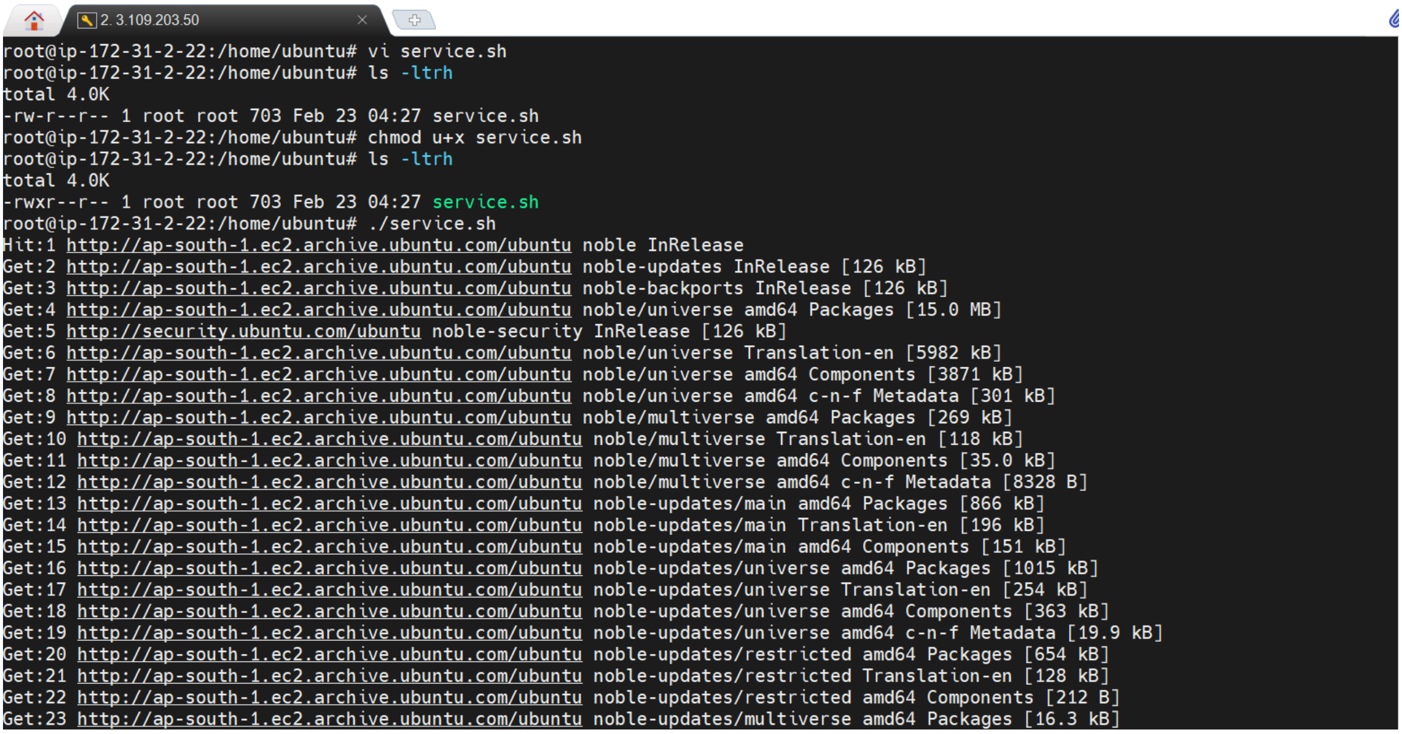
## Step 3: Installing necessary Components using shell script:

1. Docker
2. Java
3. Jenkins



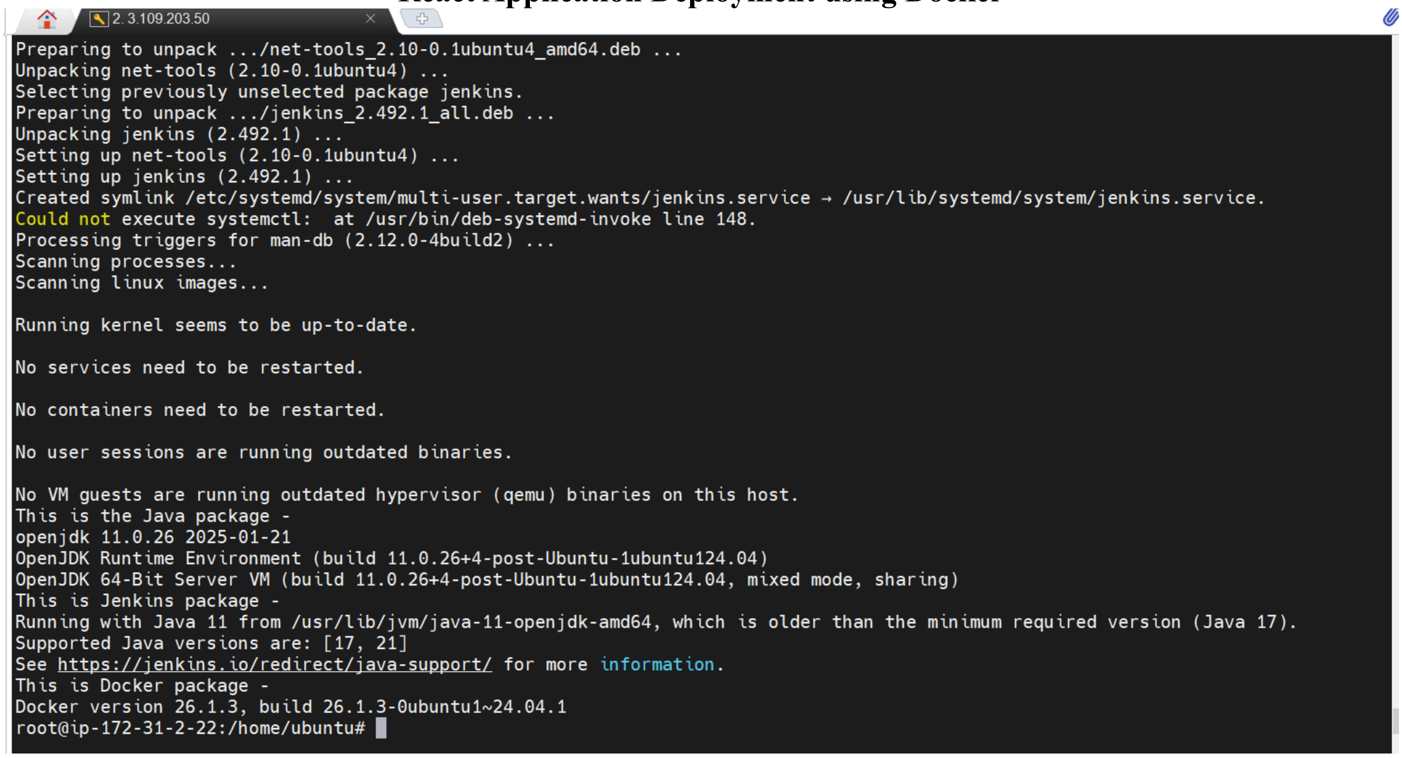
```
#!/bin/bash

#installing java:
apt-get update
apt-get install -y openjdk-11-jre

#installing docker:
apt-get update
apt-get install -y docker.io

#installing jenkins:
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
    https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
    https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
    /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install -y jenkins

#checking the installed services:
echo "This is the Java package - "
java --version

echo "This is Jenkins package - "
jenkins --version

echo "This is Docker package - "
docker --version

~
~
~
-- INSERT --                                                                          1,1           All
```

**Giving execute permission to the owner and executing the script:**



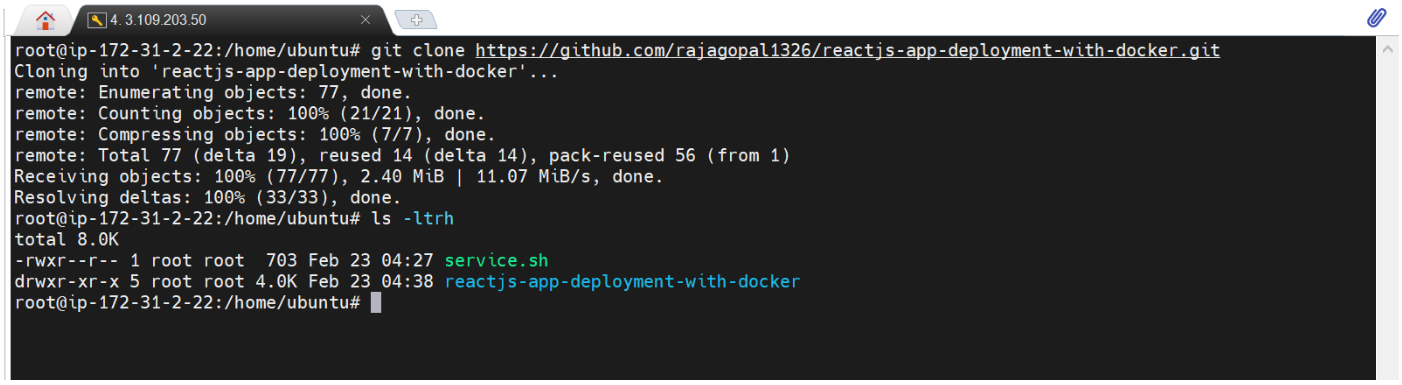
```
root@ip-172-31-2-22:/home/ubuntu# vi service.sh
root@ip-172-31-2-22:/home/ubuntu# ls -ltrh
total 4.0K
-rw-r--r-- 1 root root 703 Feb 23 04:27 service.sh
root@ip-172-31-2-22:/home/ubuntu# chmod u+x service.sh
root@ip-172-31-2-22:/home/ubuntu# ls -ltrh
total 4.0K
-rwxr--r-- 1 root root 703 Feb 23 04:27 service.sh
root@ip-172-31-2-22:/home/ubuntu# ./service.sh
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:9 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:10 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:11 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:12 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:13 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [866 kB]
Get:14 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [196 kB]
Get:15 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
Get:16 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1015 kB]
Get:17 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [254 kB]
Get:18 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [363 kB]
Get:19 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [19.9 kB]
Get:20 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [654 kB]
Get:21 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [128 kB]
Get:22 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:23 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [16.3 kB]
```

# React Application Deployment using Docker



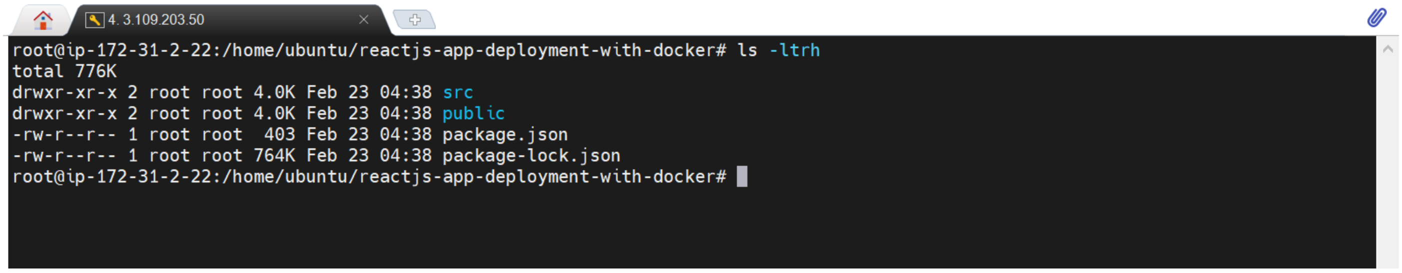
```
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.492.1_all.deb ...
Unpacking jenkins (2.492.1) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.492.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Could not execute systemctl:  at /usr/bin/deb-systemd-invoke line 148.
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
This is the Java package -
openjdk 11.0.26 2025-01-21
OpenJDK Runtime Environment (build 11.0.26+4-post-Ubuntu-1ubuntu124.04)
OpenJDK 64-Bit Server VM (build 11.0.26+4-post-Ubuntu-1ubuntu124.04, mixed mode, sharing)
This is Jenkins package -
Running with Java 11 from /usr/lib/jvm/java-11-openjdk-amd64, which is older than the minimum required version (Java 17).
Supported Java versions are: [17, 21]
See https://jenkins.io/redirect/java-support/ for more information.
This is Docker package -
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
root@ip-172-31-2-22:/home/ubuntu#
```

## Step 4: Cloning the react app git to local system



```
root@ip-172-31-2-22:/home/ubuntu# git clone https://github.com/rajagopal1326/reactjs-app-deployment-with-docker.git
Cloning into 'reactjs-app-deployment-with-docker'...
remote: Enumerating objects: 77, done.
remote: Counting objects: 100% (21/21), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 77 (delta 19), reused 14 (delta 14), pack-reused 56 (from 1)
Receiving objects: 100% (77/77), 2.40 MiB | 11.07 MiB/s, done.
Resolving deltas: 100% (33/33), done.
root@ip-172-31-2-22:/home/ubuntu# ls -ltrh
total 8.0K
-rwxr--r-- 1 root root  703 Feb 23 04:27 service.sh
drwxr-xr-x 5 root root 4.0K Feb 23 04:38 reactjs-app-deployment-with-docker
root@ip-172-31-2-22:/home/ubuntu#
```

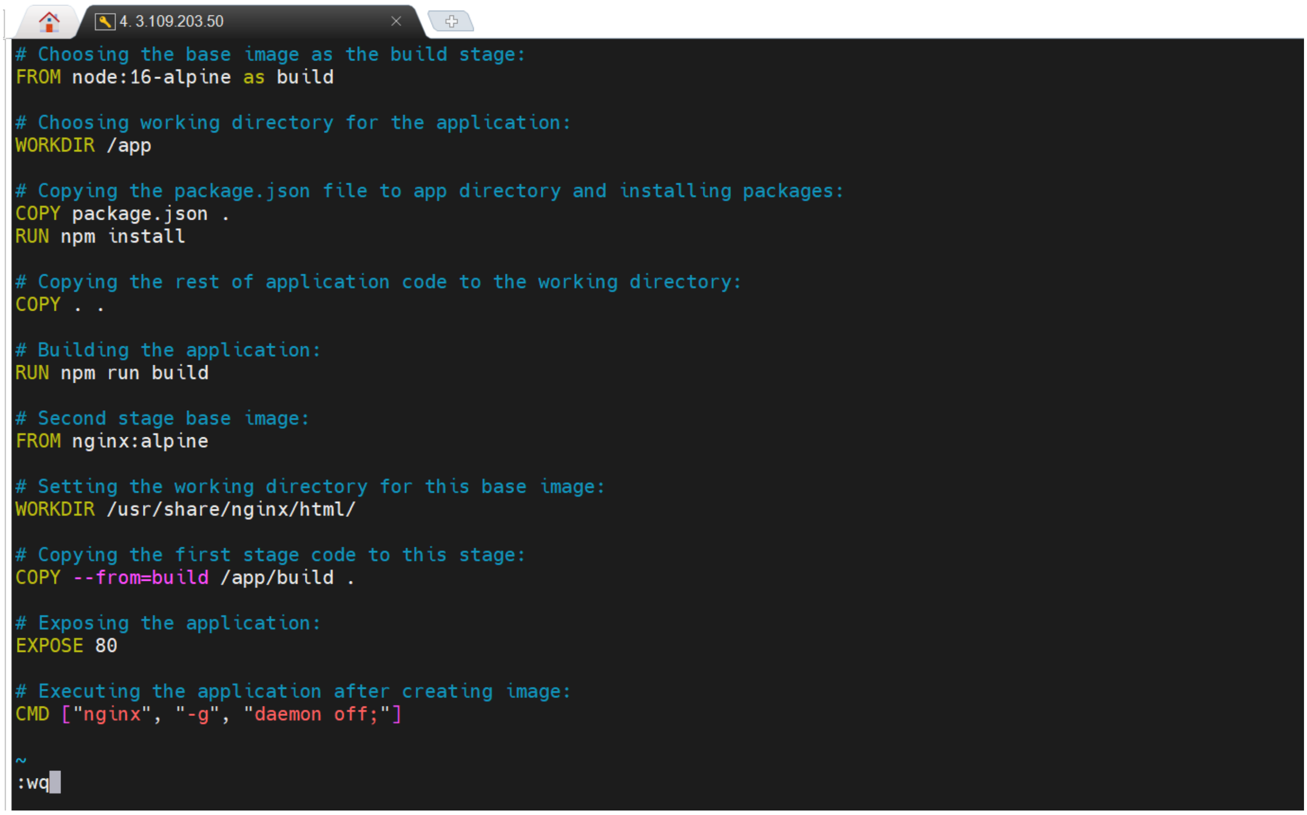
## Inside the directory:



```
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# ls -ltrh
total 776K
drwxr-xr-x 2 root root 4.0K Feb 23 04:38 src
drwxr-xr-x 2 root root 4.0K Feb 23 04:38 public
-rw-r--r-- 1 root root  403 Feb 23 04:38 package.json
-rw-r--r-- 1 root root 764K Feb 23 04:38 package-lock.json
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker#
```

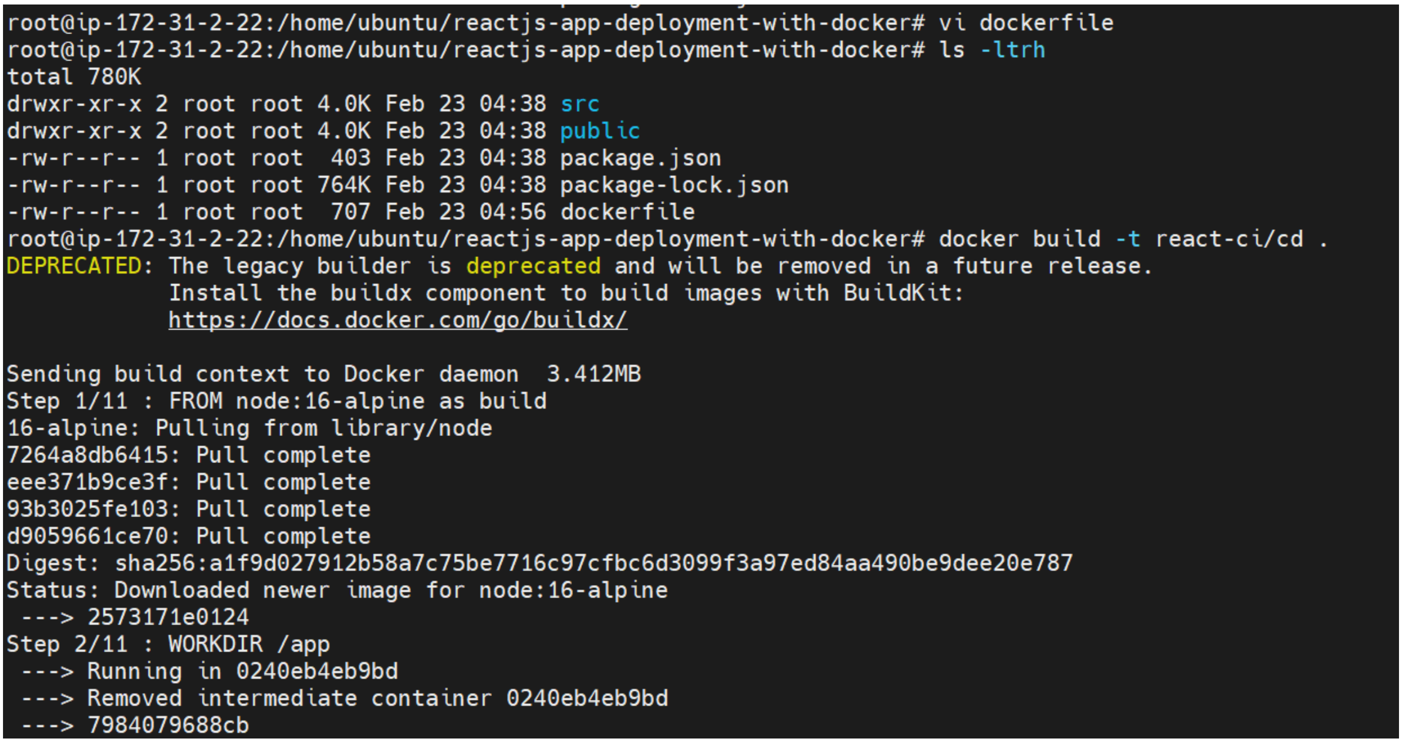
# React Application Deployment using Docker

## Step 5: Creating a Dockerfile for the above react application:



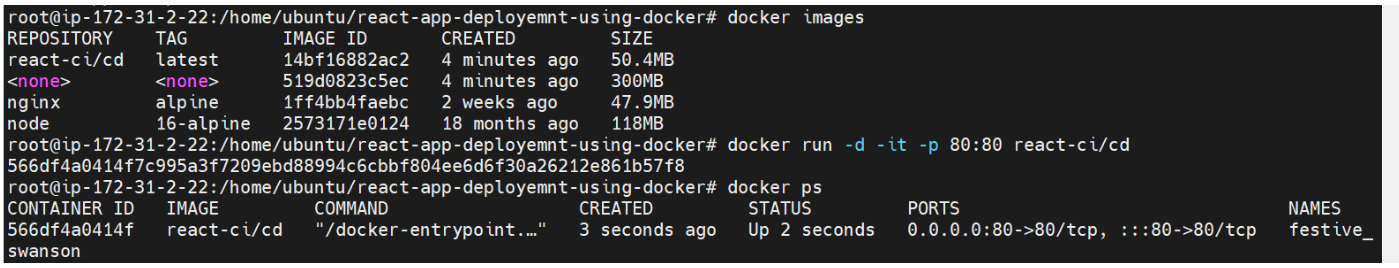
```
# Choosing the base image as the build stage:
FROM node:16-alpine as build

# Choosing working directory for the application:
WORKDIR /app

# Copying the package.json file to app directory and installing packages:
COPY package.json .
RUN npm install

# Copying the rest of application code to the working directory:
COPY . .

# Building the application:
RUN npm run build

# Second stage base image:
FROM nginx:alpine

# Setting the working directory for this base image:
WORKDIR /usr/share/nginx/html/

# Copying the first stage code to this stage:
COPY --from=build /app/build .

# Exposing the application:
EXPOSE 80

# Executing the application after creating image:
CMD ["nginx", "-g", "daemon off;"]

~
:wq
```

## Step 6: Build and run the Dockerfile



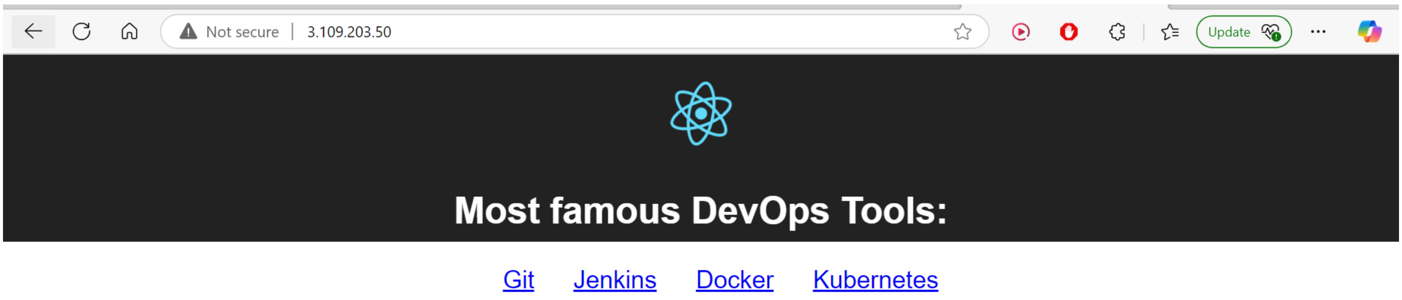
```
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# vi dockerfile
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# ls -ltrh
total 780K
drwxr-xr-x 2 root root 4.0K Feb 23 04:38 src
drwxr-xr-x 2 root root 4.0K Feb 23 04:38 public
-rw-r--r-- 1 root root  403 Feb 23 04:38 package.json
-rw-r--r-- 1 root root 764K Feb 23 04:38 package-lock.json
-rw-r--r-- 1 root root  707 Feb 23 04:56 dockerfile
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# docker build -t react-ci/cd .
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  3.412MB
Step 1/11 : FROM node:16-alpine as build
16-alpine: Pulling from library/node
7264a8db6415: Pull complete
eee371b9ce3f: Pull complete
93b3025fe103: Pull complete
d9059661ce70: Pull complete
Digest: sha256:a1f9d027912b58a7c75be7716c97cfbc6d3099f3a97ed84aa490be9dee20e787
Status: Downloaded newer image for node:16-alpine
 ---> 2573171e0124
Step 2/11 : WORKDIR /app
 ---> Running in 0240eb4eb9bd
 ---> Removed intermediate container 0240eb4eb9bd
 ---> 7984079688cb
```

# React Application Deployment using Docker



```
root@ip-172-31-2-22:/home/ubuntu/react-app-deployemnt-using-docker# docker images
REPOSITORY    TAG         IMAGE ID       CREATED          SIZE
react-ci/cd   latest      14bf16882ac2   4 minutes ago    50.4MB
<none>        <none>      519d0823c5ec   4 minutes ago    300MB
nginx         alpine      1ff4bb4faebc   2 weeks ago      47.9MB
node          16-alpine   2573171e0124   18 months ago    118MB
root@ip-172-31-2-22:/home/ubuntu/react-app-deployemnt-using-docker# docker run -d -it -p 80:80 react-ci/cd
566df4a0414f7c995a3f7209ebd88994c6cbbf804ee6d6f30a26212e861b57f8
root@ip-172-31-2-22:/home/ubuntu/react-app-deployemnt-using-docker# docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED         STATUS         PORTS                                 NAMES
566df4a0414f   react-ci/cd   "/docker-entrypoint.…"   3 seconds ago   Up 2 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp     festive_
swanson
```

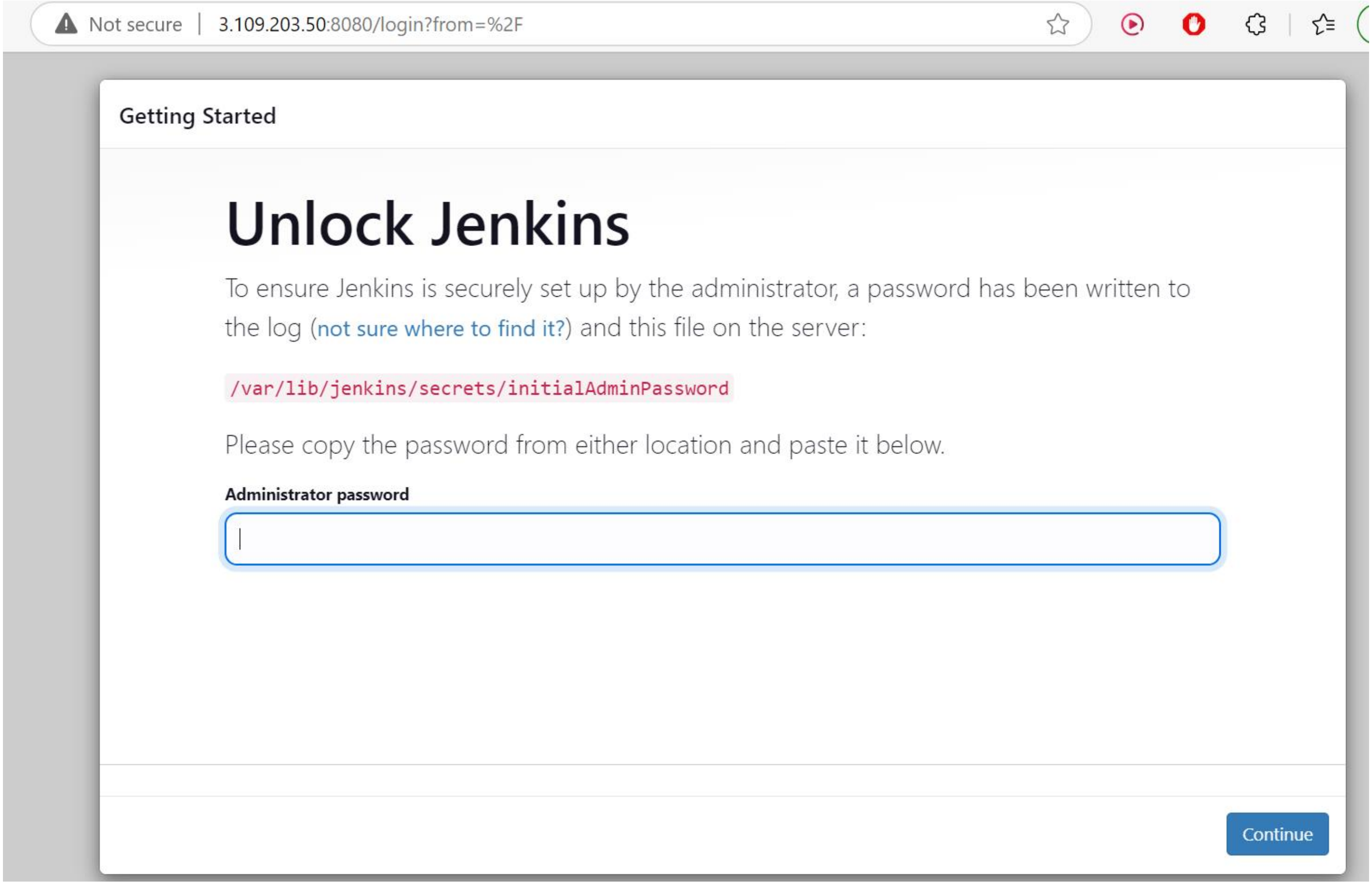
**Reaching the dockerised application over the internet:**



Most famous DevOps Tools:

Git Jenkins Docker Kubernetes

## Step 7: start the Jenkins



Getting Started

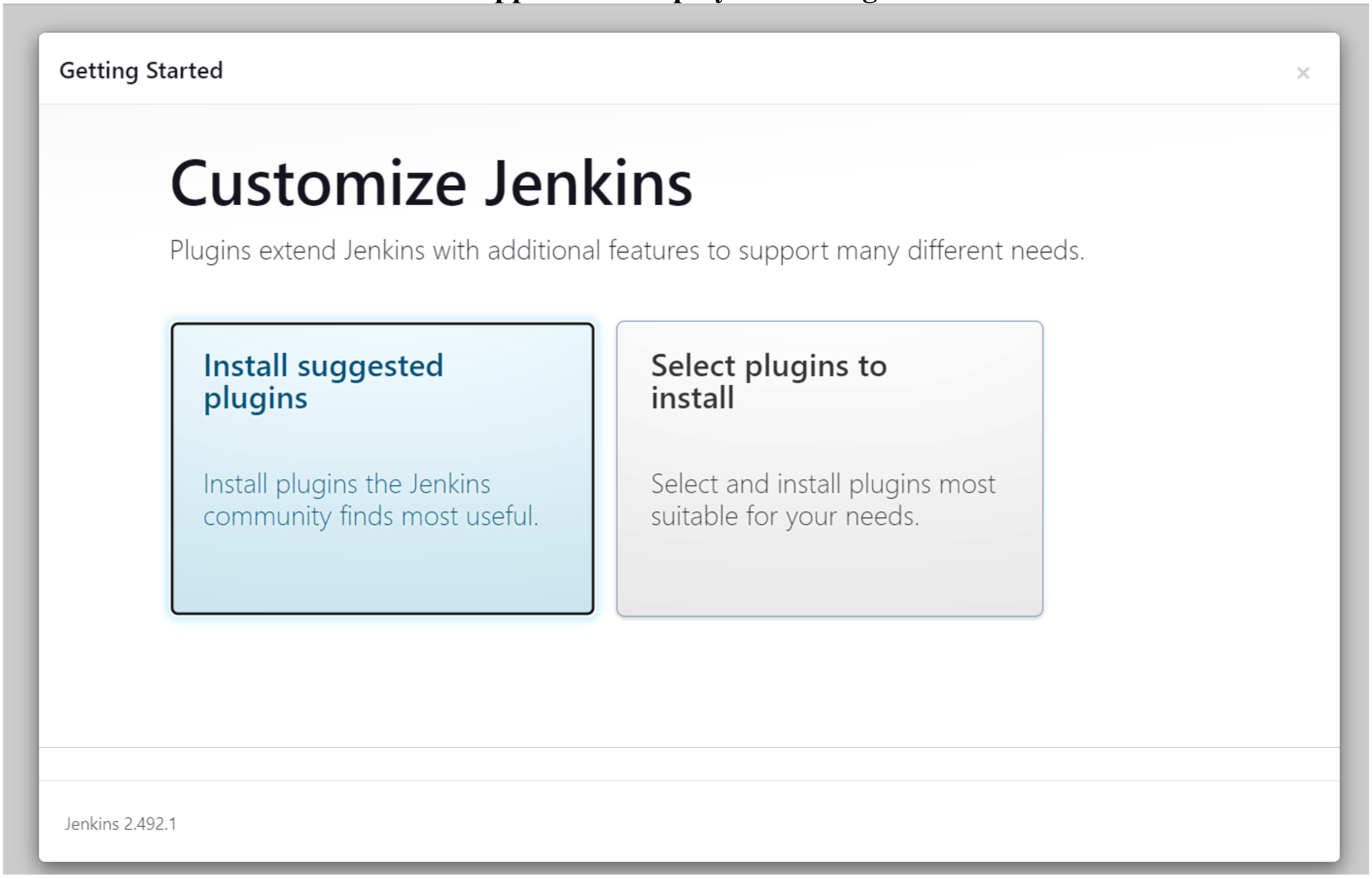
Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

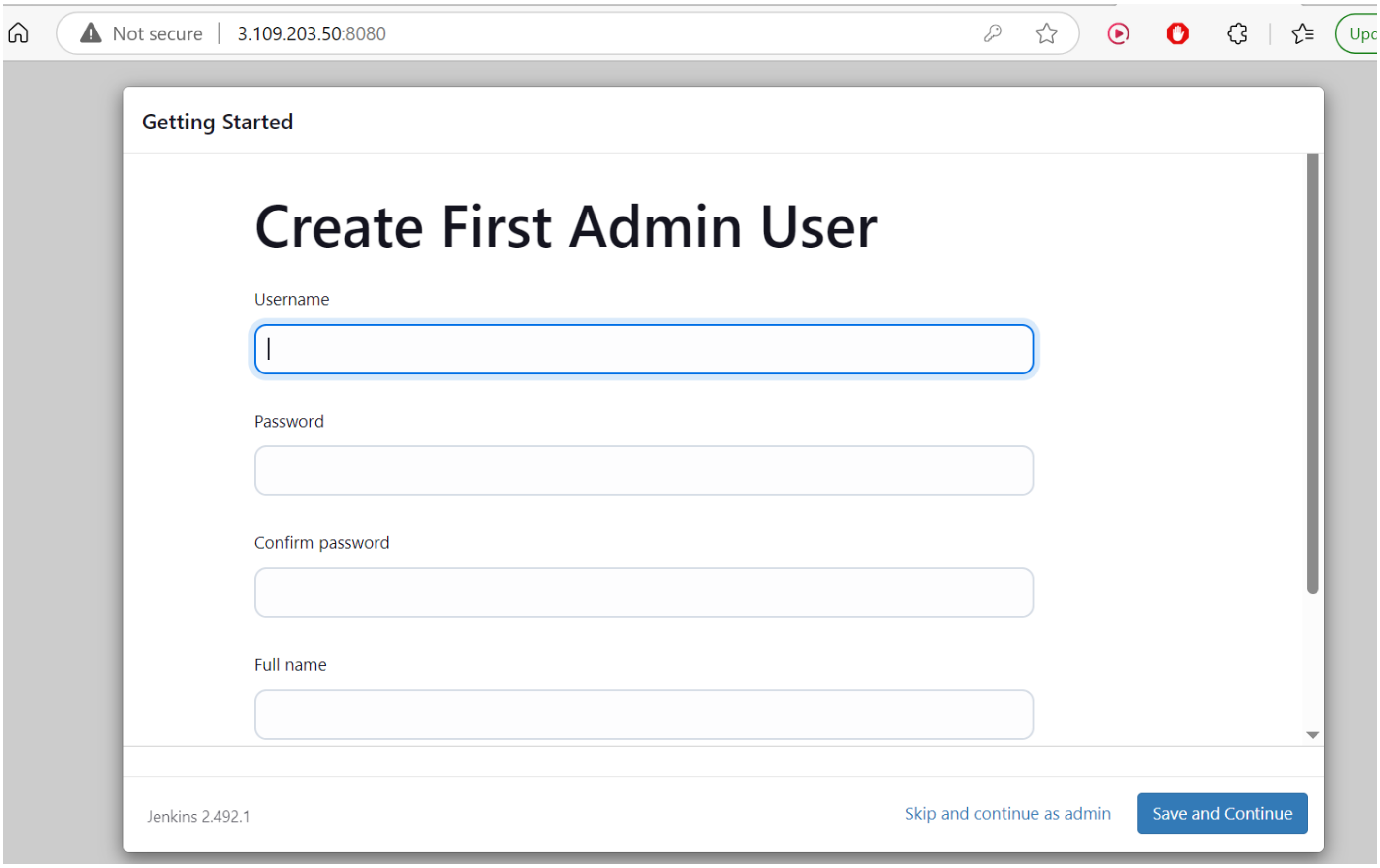


Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins
Install plugins the Jenkins community finds most useful.
Select plugins to install
Select and install plugins most suitable for your needs.
×
Jenkins 2.492.1



Not secure | 3.109.203.50:8080

Getting Started

Create First Admin User

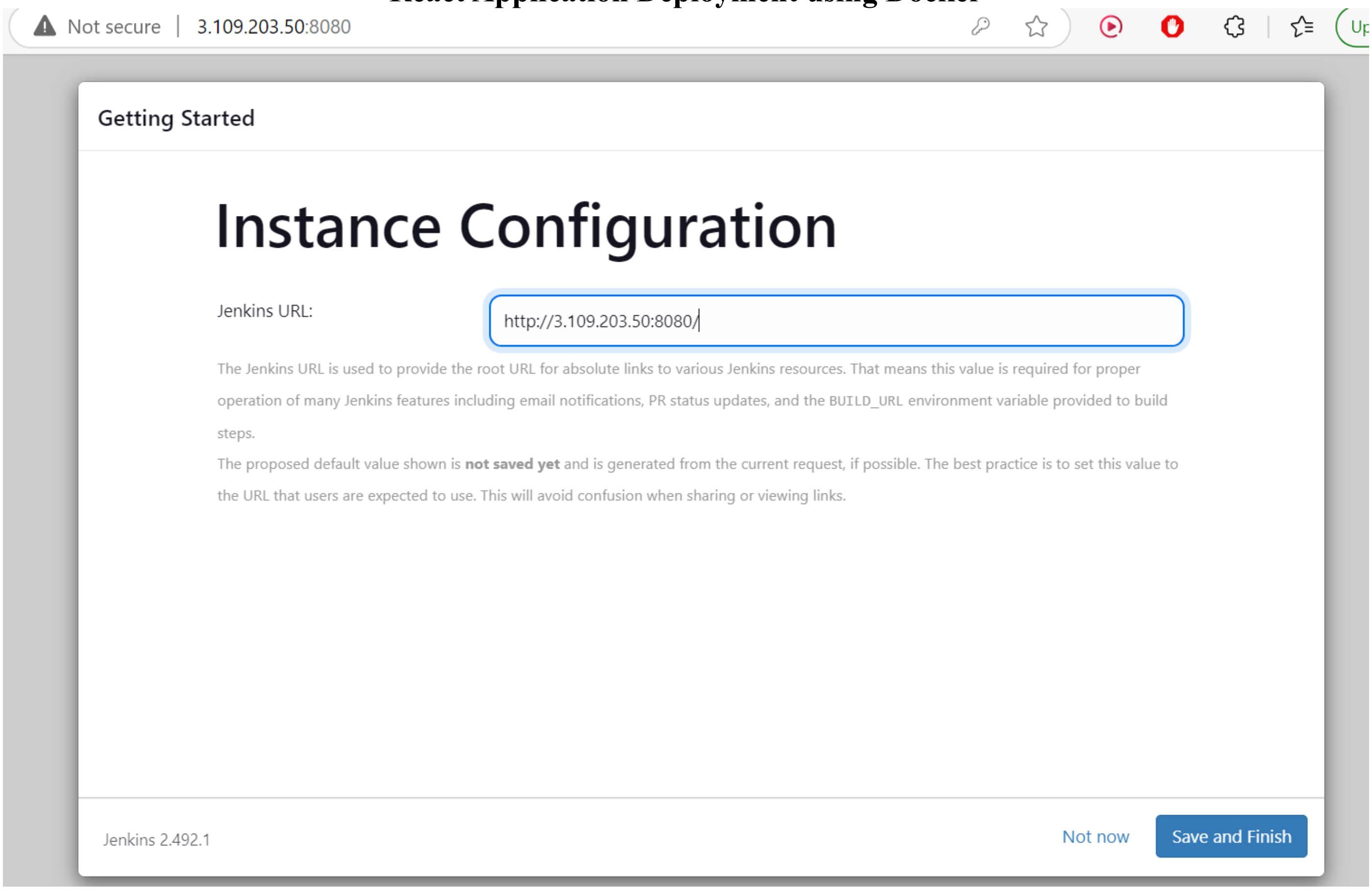
Username

Password

Confirm password

Full name

# React Application Deployment using Docker



Getting Started

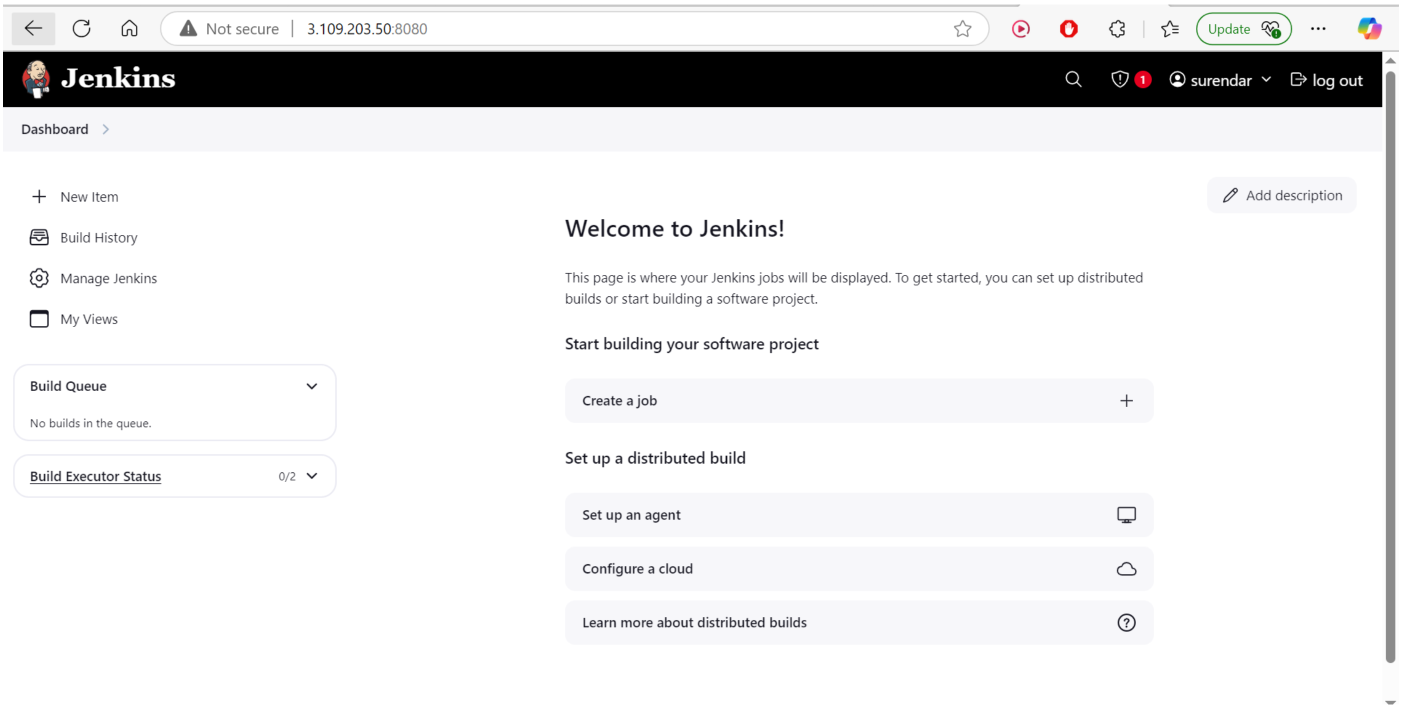
Instance Configuration

Jenkins URL: http://3.109.203.50:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.492.1



Build Queue

No builds in the queue.

Build Executor Status 0/2

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job

Set up a distributed build

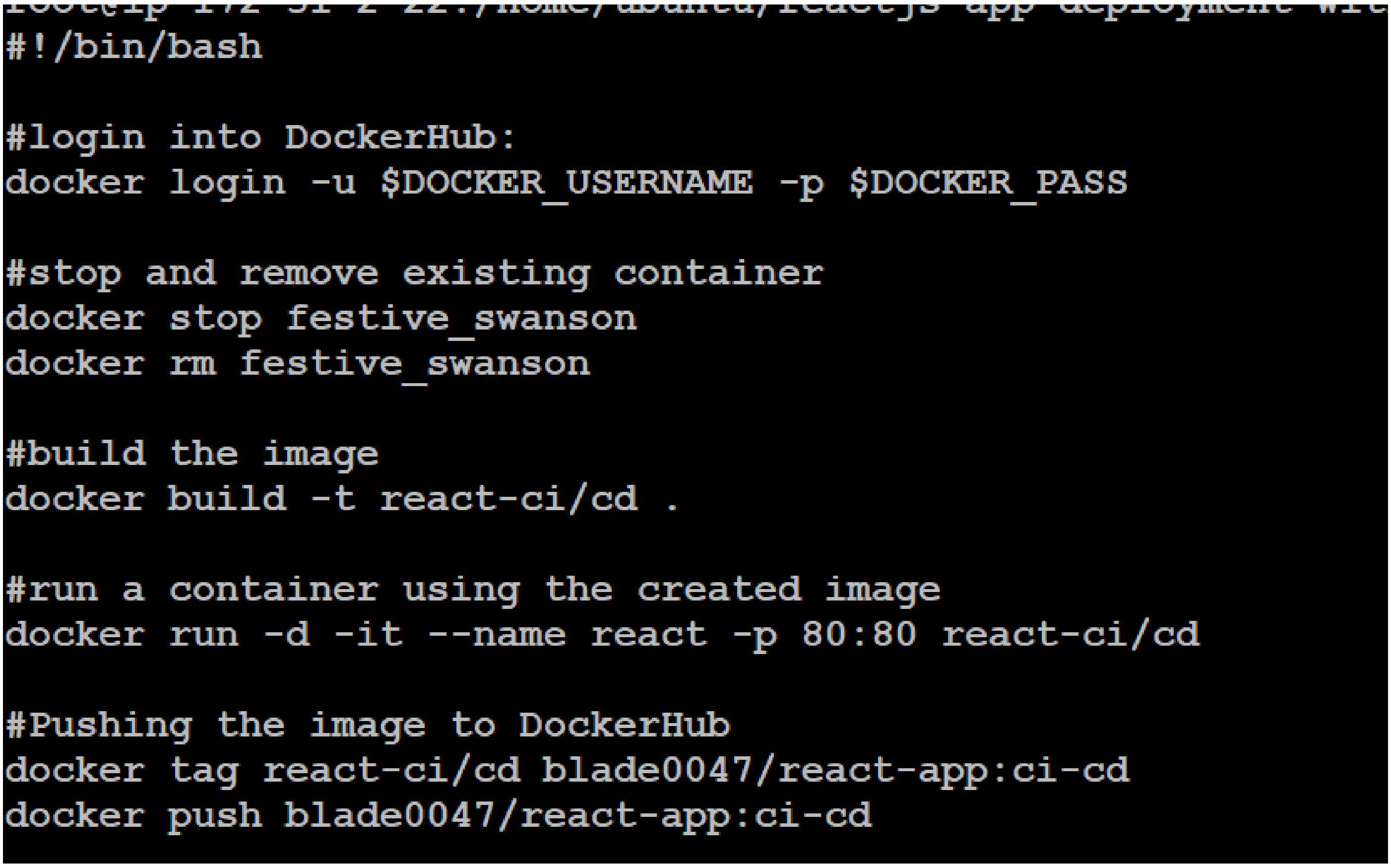
Set up an agent

Configure a cloud

Learn more about distributed builds

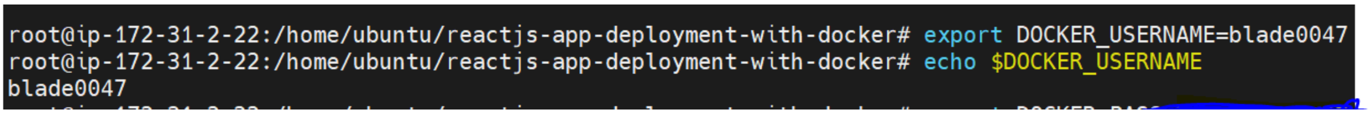
React Application Deployment using Docker

## Step 8 - Creating a script file for building & pushing the image to Docker Hub:

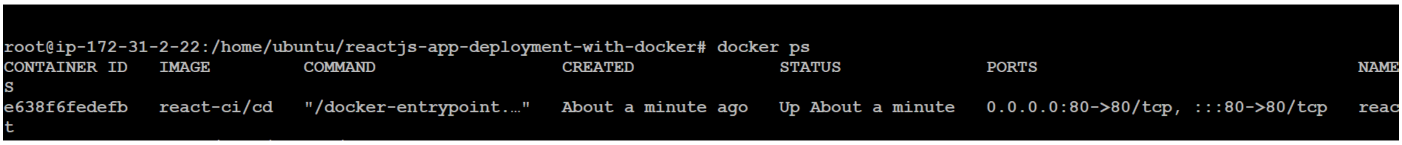


```
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-wit
#!/bin/bash

#login into DockerHub:
docker login -u $DOCKER_USERNAME -p $DOCKER_PASS

#stop and remove existing container
docker stop festive_swanson
docker rm festive_swanson

#build the image
docker build -t react-ci/cd .

#run a container using the created image
docker run -d -it --name react -p 80:80 react-ci/cd

#Pushing the image to DockerHub
docker tag react-ci/cd blade0047/react-app:ci-cd
docker push blade0047/react-app:ci-cd
```

## Step 9: Exporting username and password



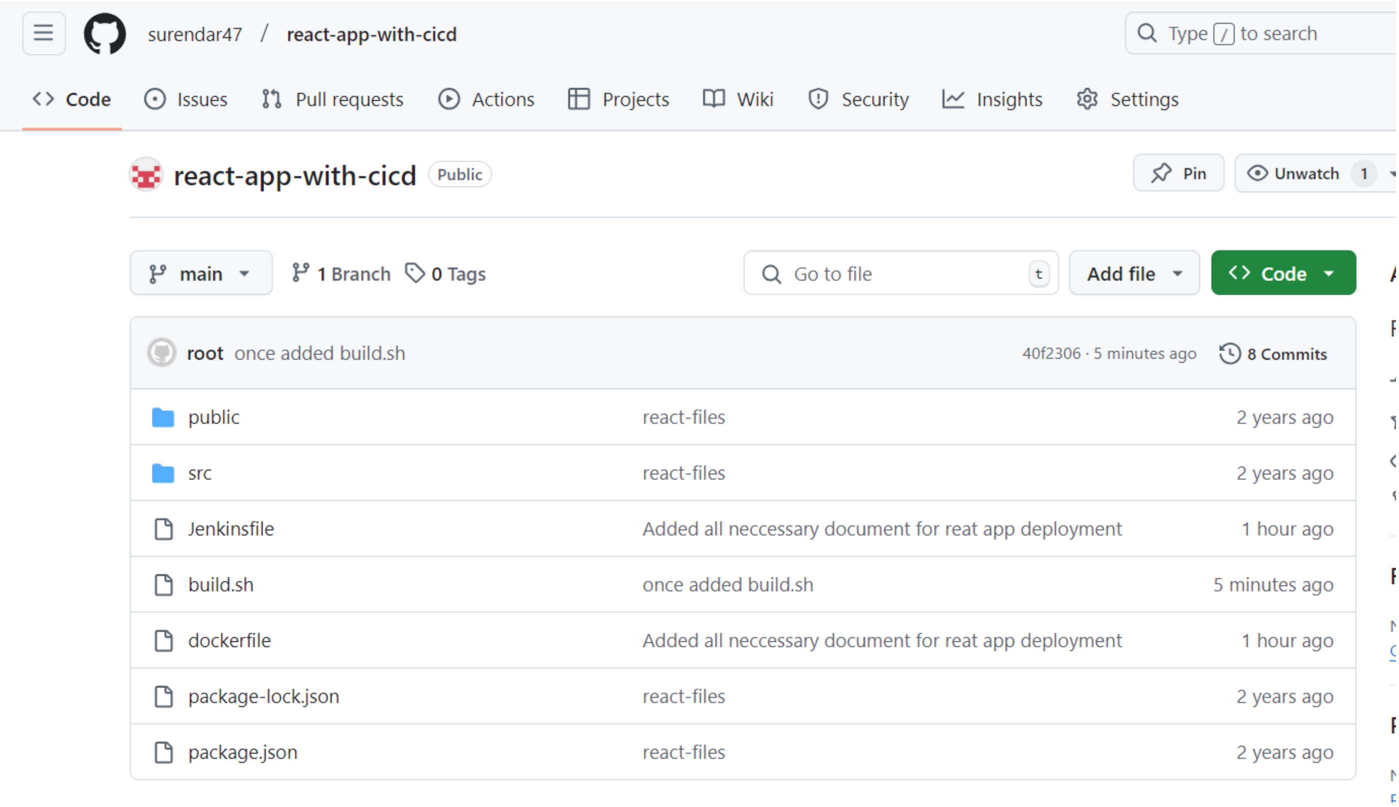
```
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# export DOCKER_USERNAME=blade0047
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# echo $DOCKER_USERNAME
blade0047
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# export DOCKER_PASS[redacted]
```



```
root@ip-172-31-2-22:/home/ubuntu/reactjs-app-deployment-with-docker# docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED              STATUS              PORTS                               NAME
S
e638f6fedefb   react-ci/cd   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:80->80/tcp, :::80->80/tcp   reac
t
```

# React Application Deployment using Docker

## Step 10: pushing the source code into github



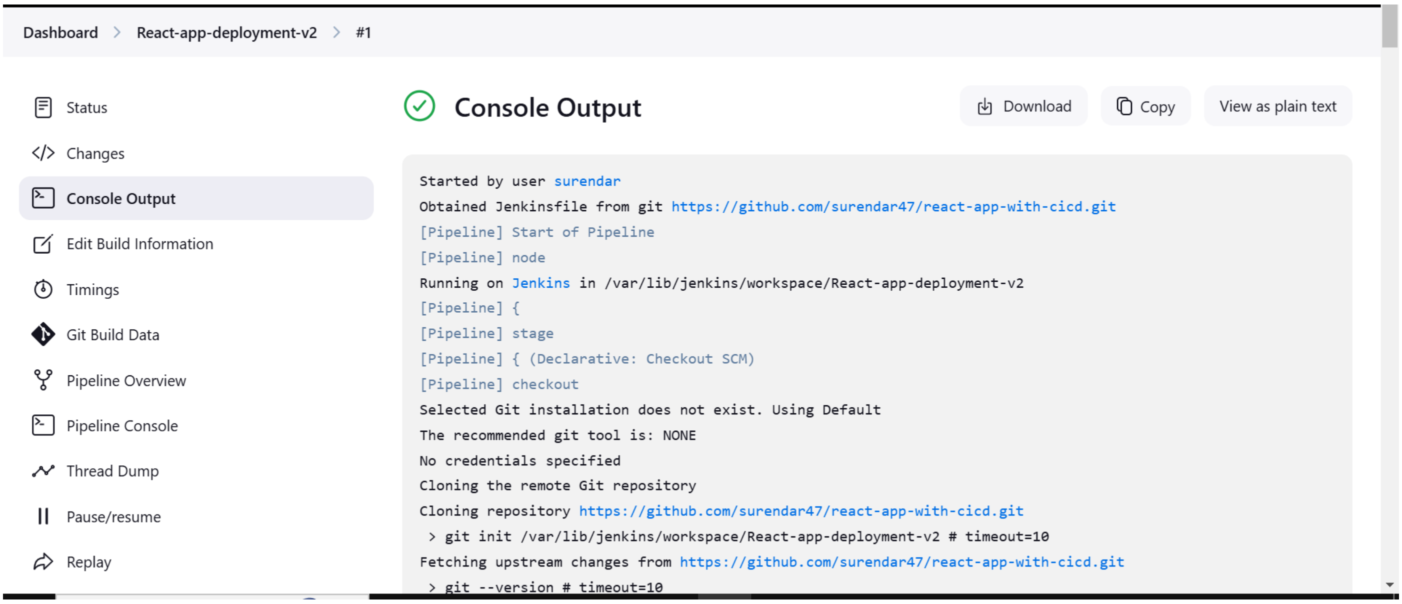
root once added build.sh 40f2306 · 5 minutes ago 8 Commits

| | | |
|---|---|---|
| public | react-files | 2 years ago |
| src | react-files | 2 years ago |
| Jenkinsfile | Added all neccessary document for reat app deployment | 1 hour ago |
| build.sh | once added build.sh | 5 minutes ago |
| dockerfile | Added all neccessary document for reat app deployment | 1 hour ago |
| package-lock.json | react-files | 2 years ago |
| package.json | react-files | 2 years ago |

## Step 11: Setting up Environment variables of DockerHub with Jenkins

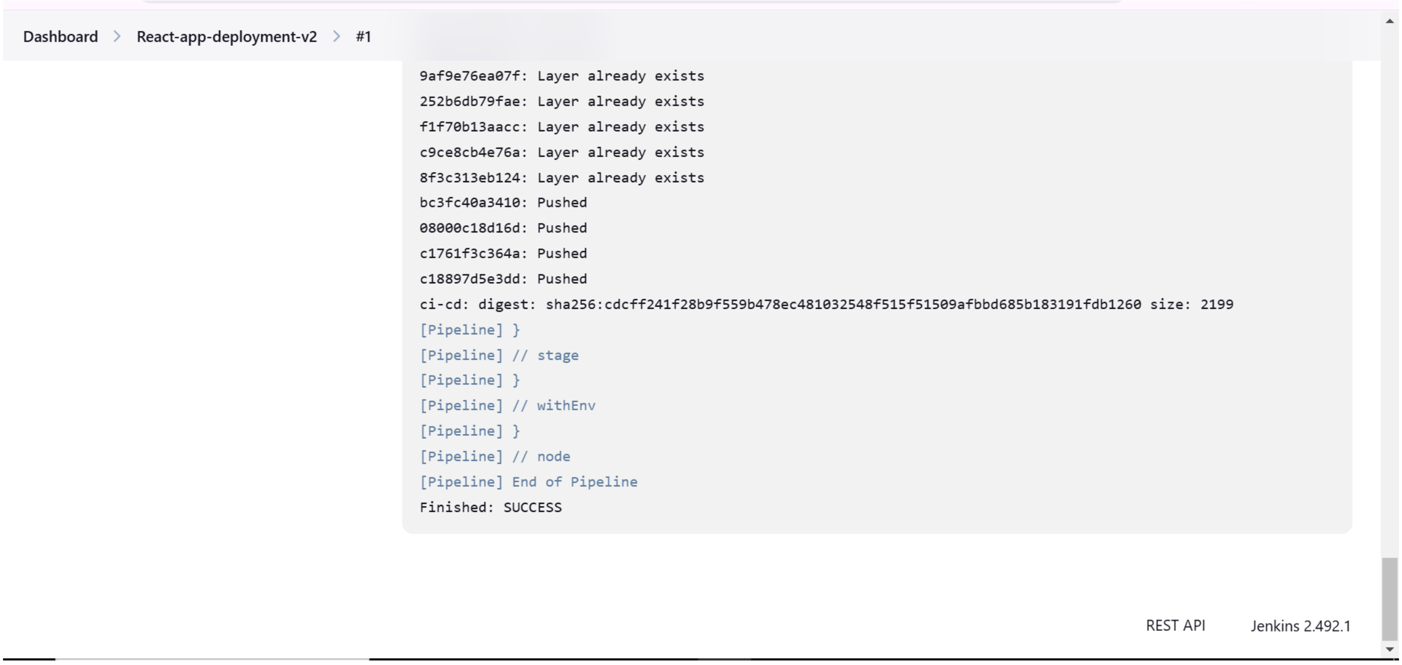
Manage Jenkins → System → Environment variables → Fill DOCKER_USERNAME,DOCKER_PASS & Values → Apply & Save.

## Step 12: Build Jenkis Pipeline using git & github repository



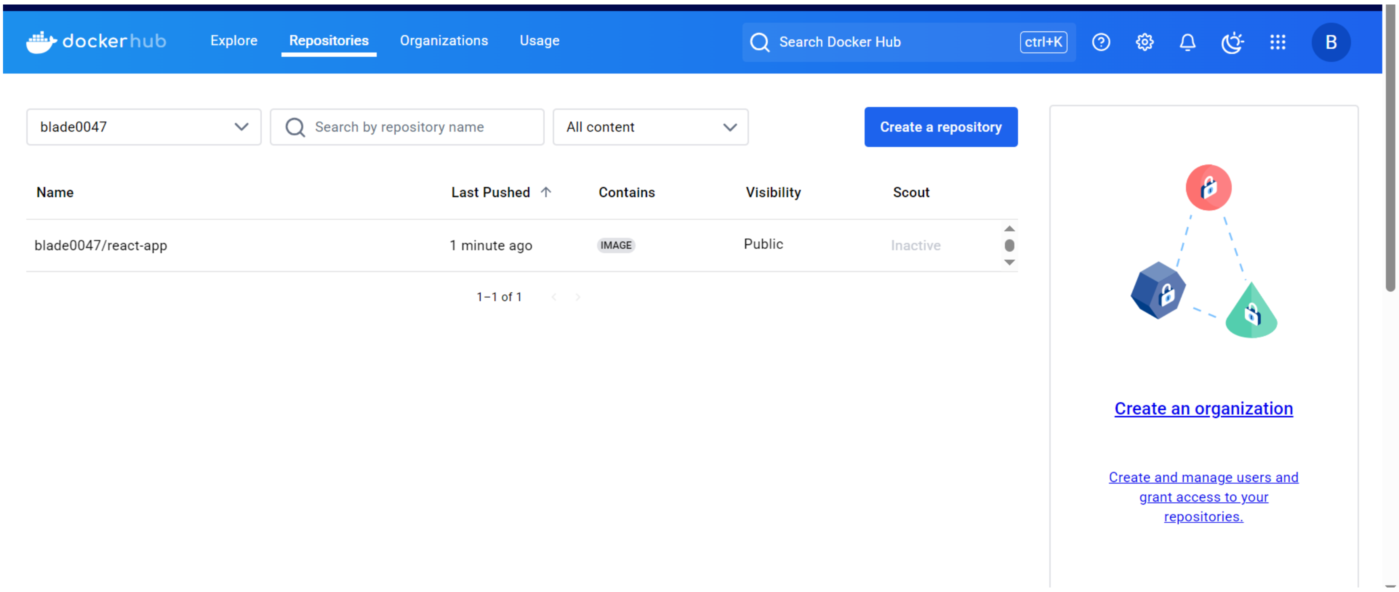
```
Started by user surendar
Obtained Jenkinsfile from git https://github.com/surendar47/react-app-with-cicd.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/React-app-deployment-v2
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/surendar47/react-app-with-cicd.git
 > git init /var/lib/jenkins/workspace/React-app-deployment-v2 # timeout=10
Fetching upstream changes from https://github.com/surendar47/react-app-with-cicd.git
 > git --version # timeout=10
```

# React Application Deployment using Docker



```
9af9e76ea07f: Layer already exists
252b6db79fae: Layer already exists
f1f70b13aacc: Layer already exists
c9ce8cb4e76a: Layer already exists
8f3c313eb124: Layer already exists
bc3fc40a3410: Pushed
08000c18d16d: Pushed
c1761f3c364a: Pushed
c18897d5e3dd: Pushed
ci-cd: digest: sha256:cdcff241f28b9f559b478ec481032548f515f51509afbbd685b183191fdb1260 size: 2199
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

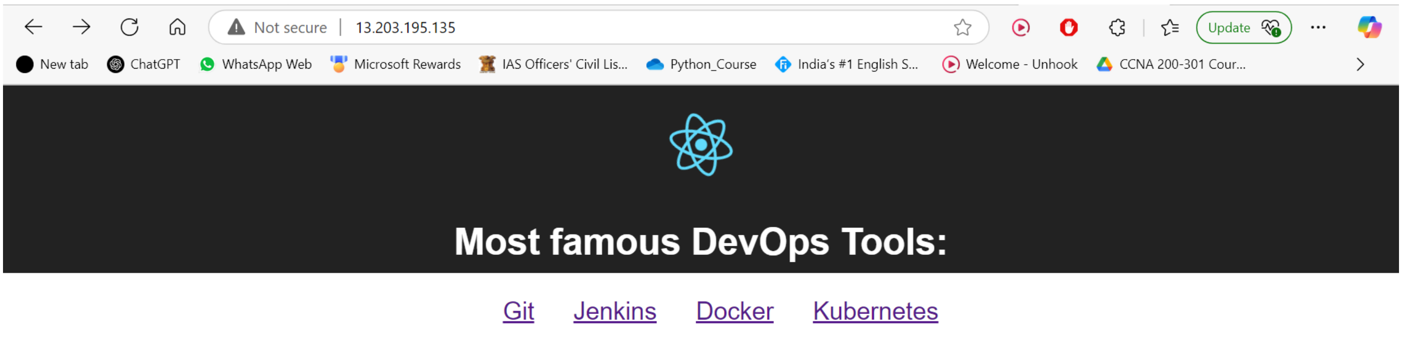
**Once the Build is successful, you can see the docker image will be pushed into your Docker Hub account.**



| Name | Last Pushed ↑ | Contains | Visibility | Scout |
|---|---|---|---|---|
| blade0047/react-app | 1 minute ago | IMAGE | Public | Inactive |

1–1 of 1

Create an organization
Create and manage users and grant access to your repositories.



Most famous DevOps Tools:

Git Jenkins Docker Kubernetes