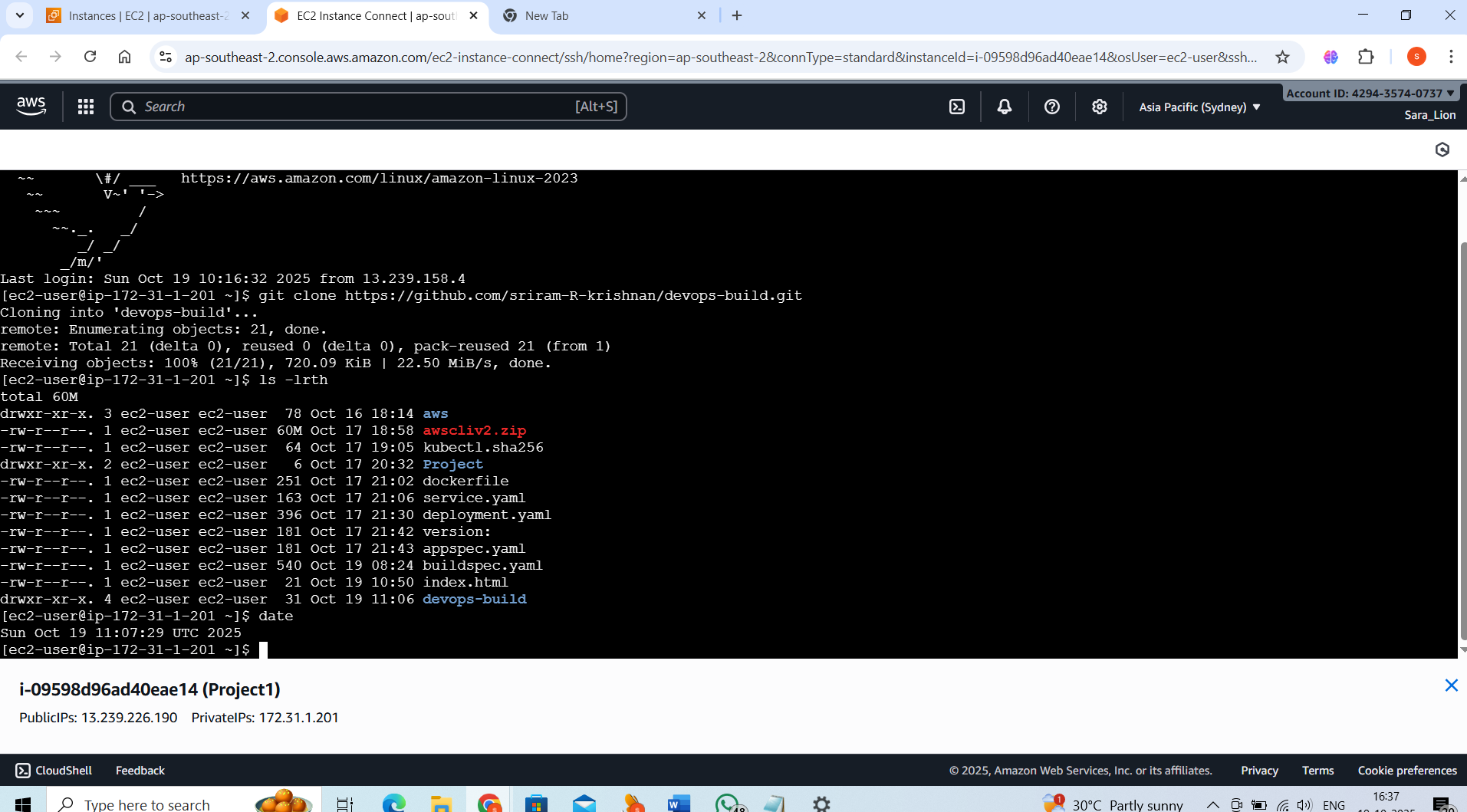
STEP 1:

TASK : Clone the git content into the server to copy the project content.

URL : <https://github.com/sriram-R-krishnan/devops-build>

Commands :

Git clone https://github.com/sriram-R-krishnan/devops-build



STEP 2:

Task : Docker installation completed please check the docker version.

Commands :

# 1. Update system

sudo apt-get update

# 2. Install required packages

sudo apt-get install \

ca-certificates \

curl \

gnupg \

lsb-release

# 3. Add Docker’s GPG key

sudo mkdir -m 0755 -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \

sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

# 4. Set up the Docker repository

echo \

"deb [arch=$(dpkg --print-architecture) \

signed-by=/etc/apt/keyrings/docker.gpg] \

https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# 5. Update package index

sudo apt-get update

# 6. Install Docker Engine

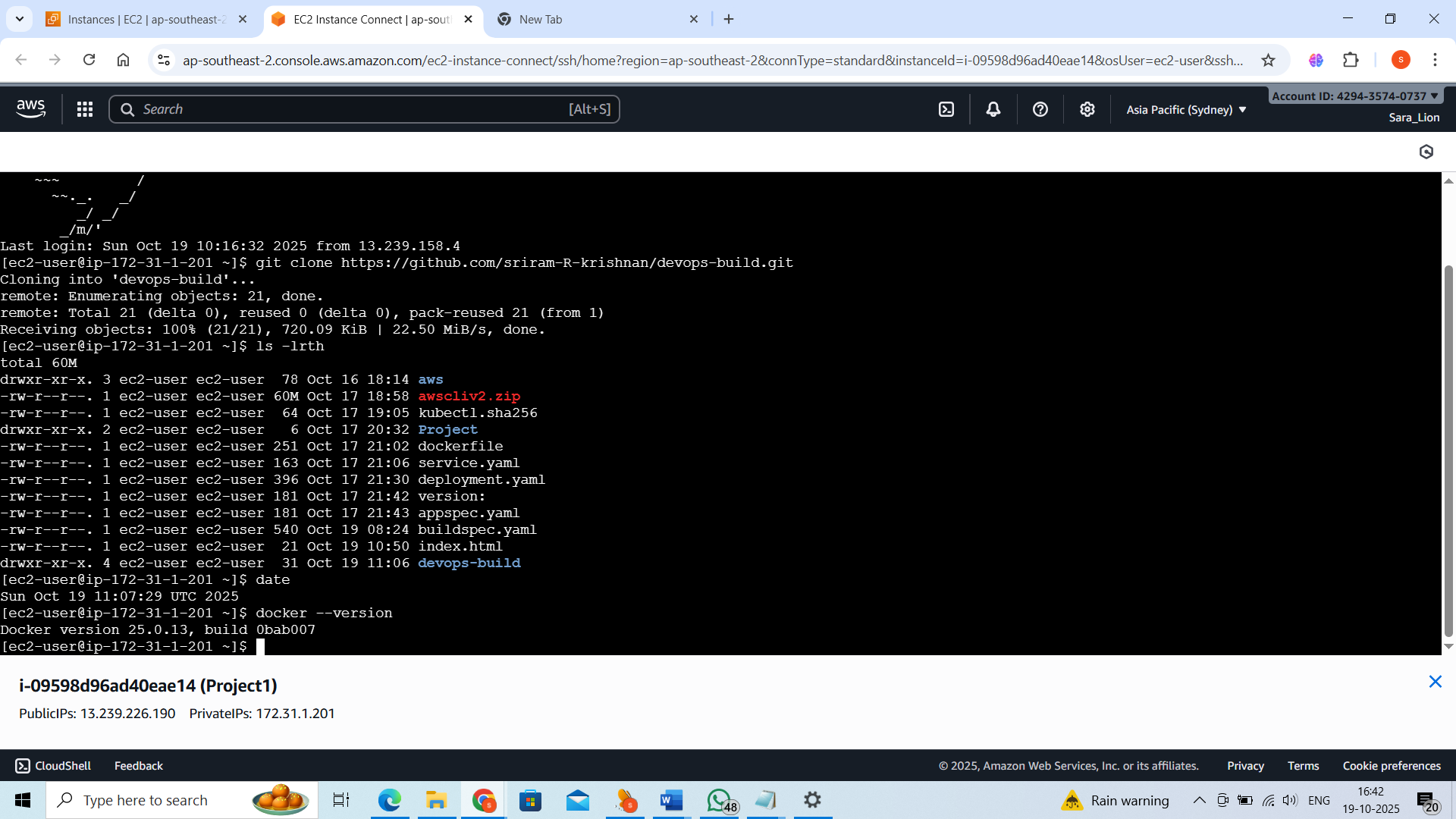
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

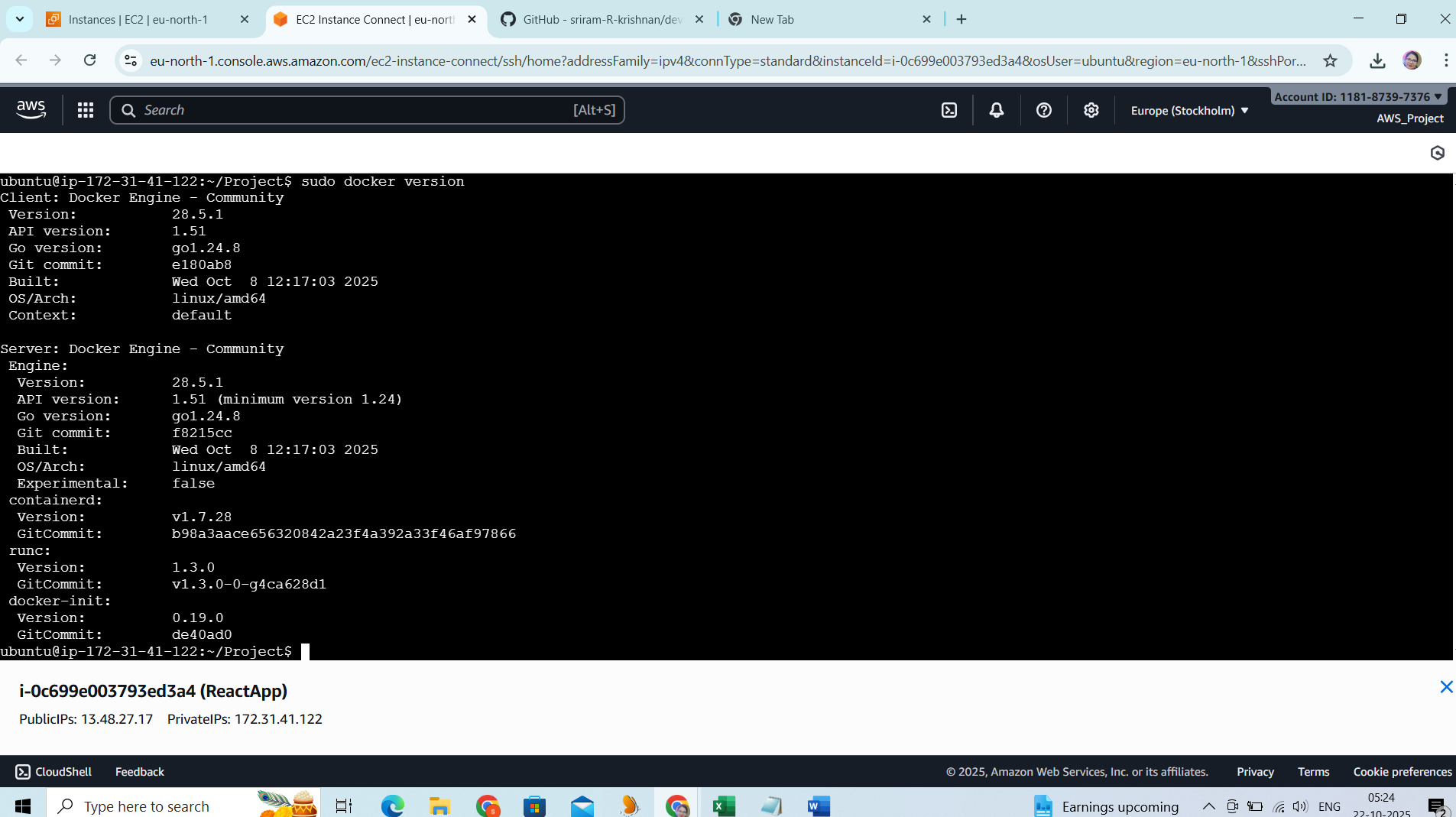
# 7. Optional: Add your user to the docker group (no sudo needed to run docker)

sudo usermod -aG docker $USER

newgrp docker

docker --version





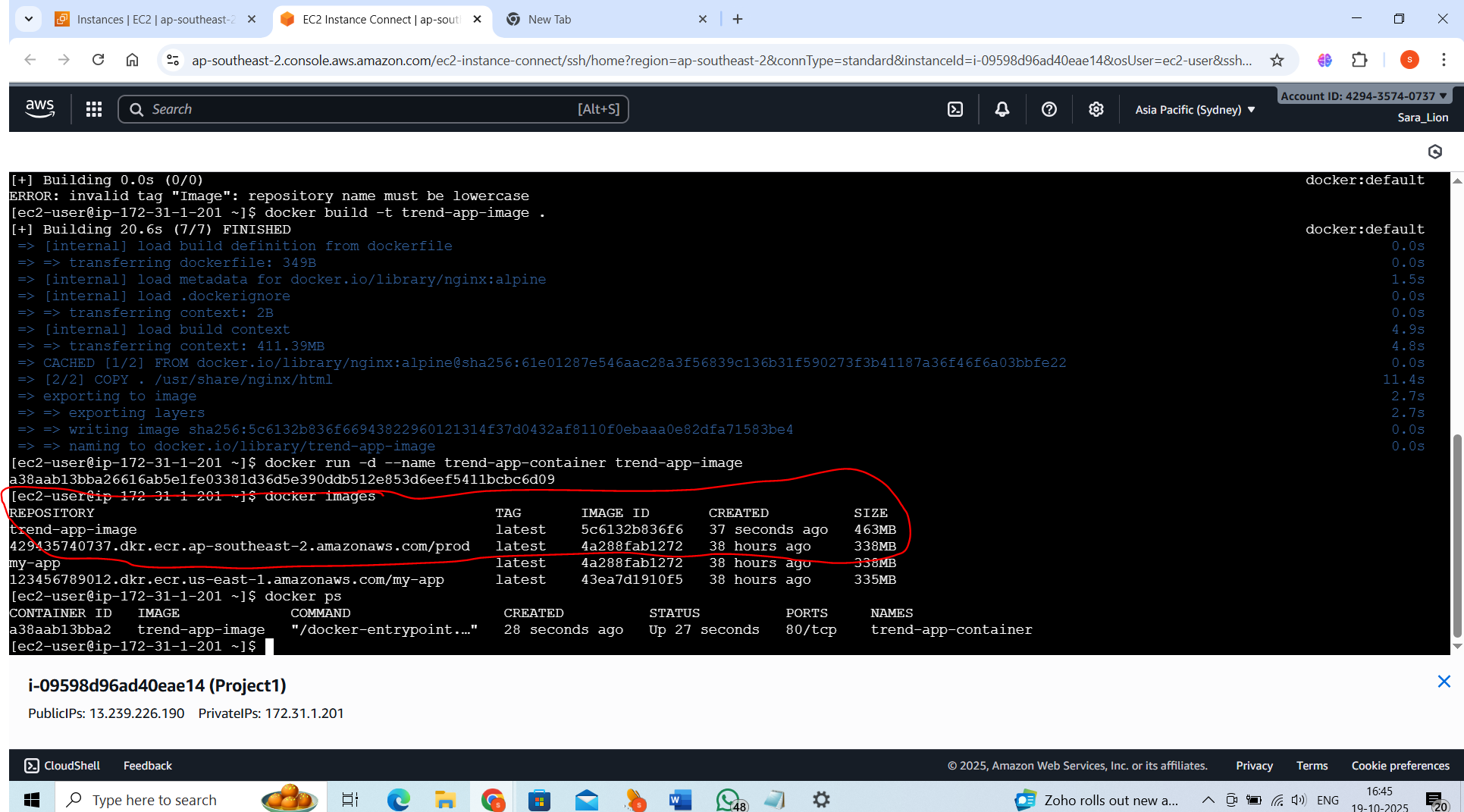
STEP 3:

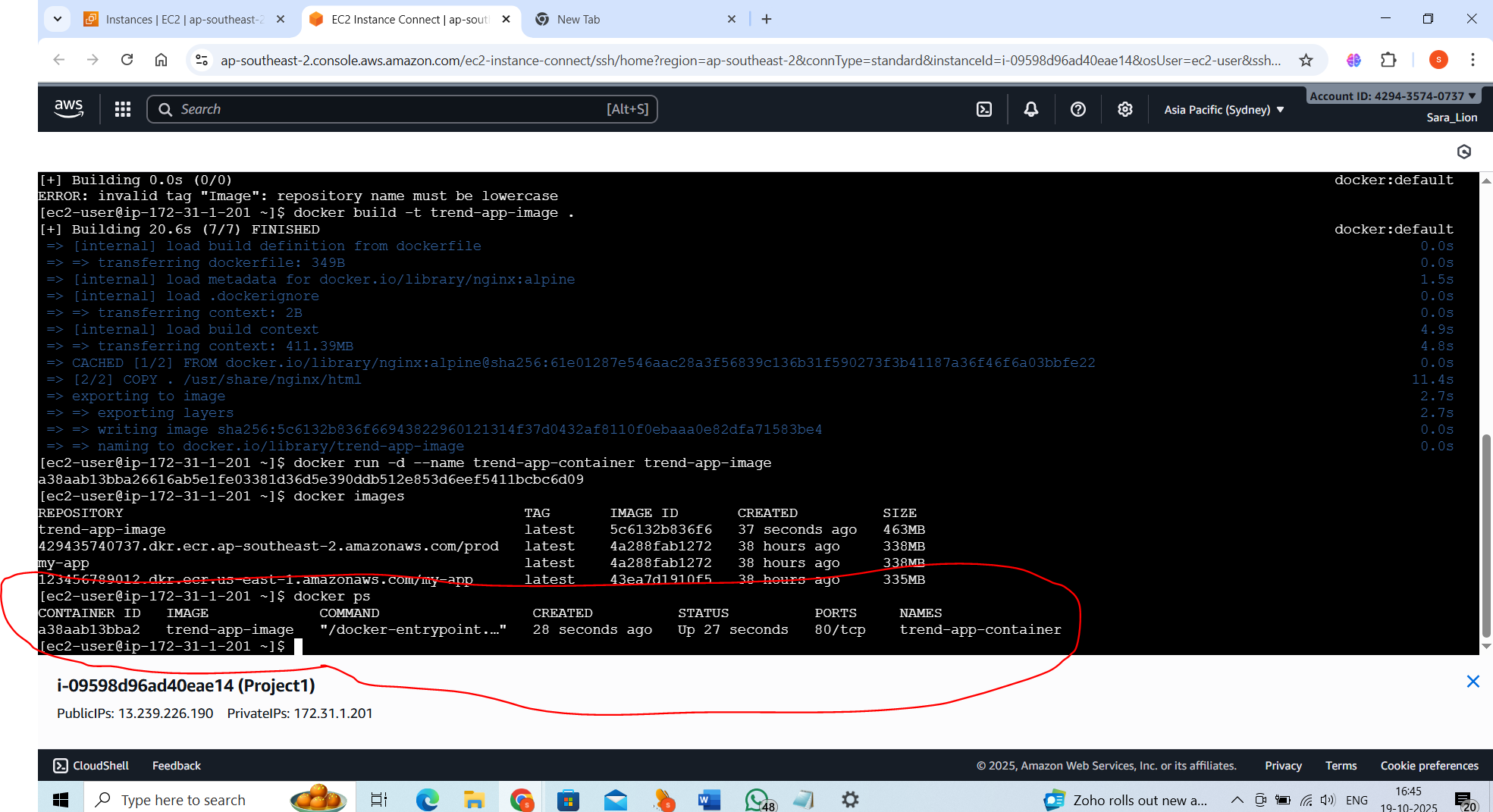
Create the Dockerfile ; build the images and run the images .

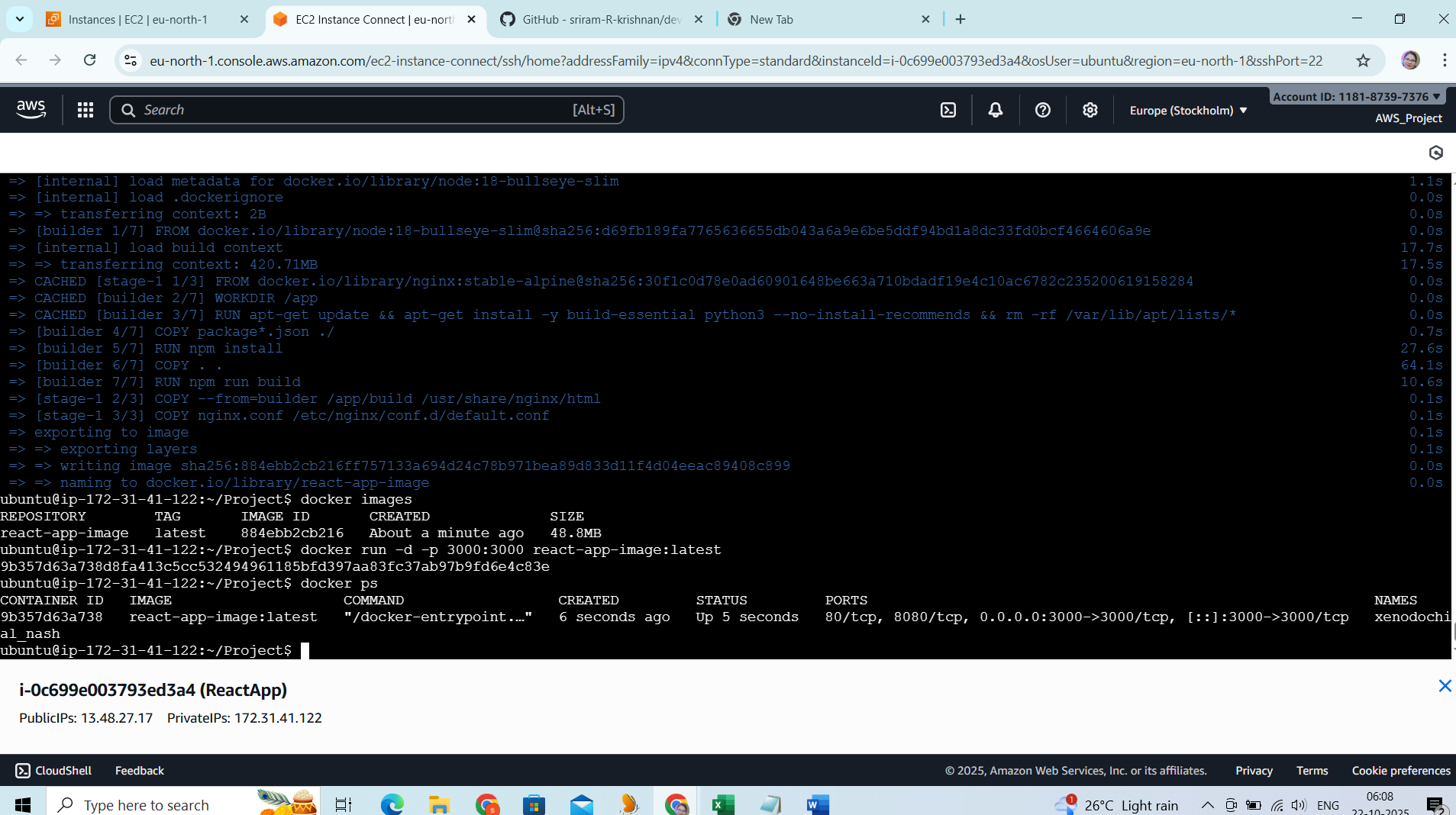
Commands:

docker build -t trend-app-image .

docker run --name trend-app-container trend-app-image







STEP 3:

Task : Docker compose installation completed . Create the docker compose file to use the image

Commands:

docker-compose.yml

version: '3.8'

services:

web:

image: ${IMAGE\_NAME:-<DOCKER\_USERNAME>/<IMAGE\_NAME>:latest}

ports:

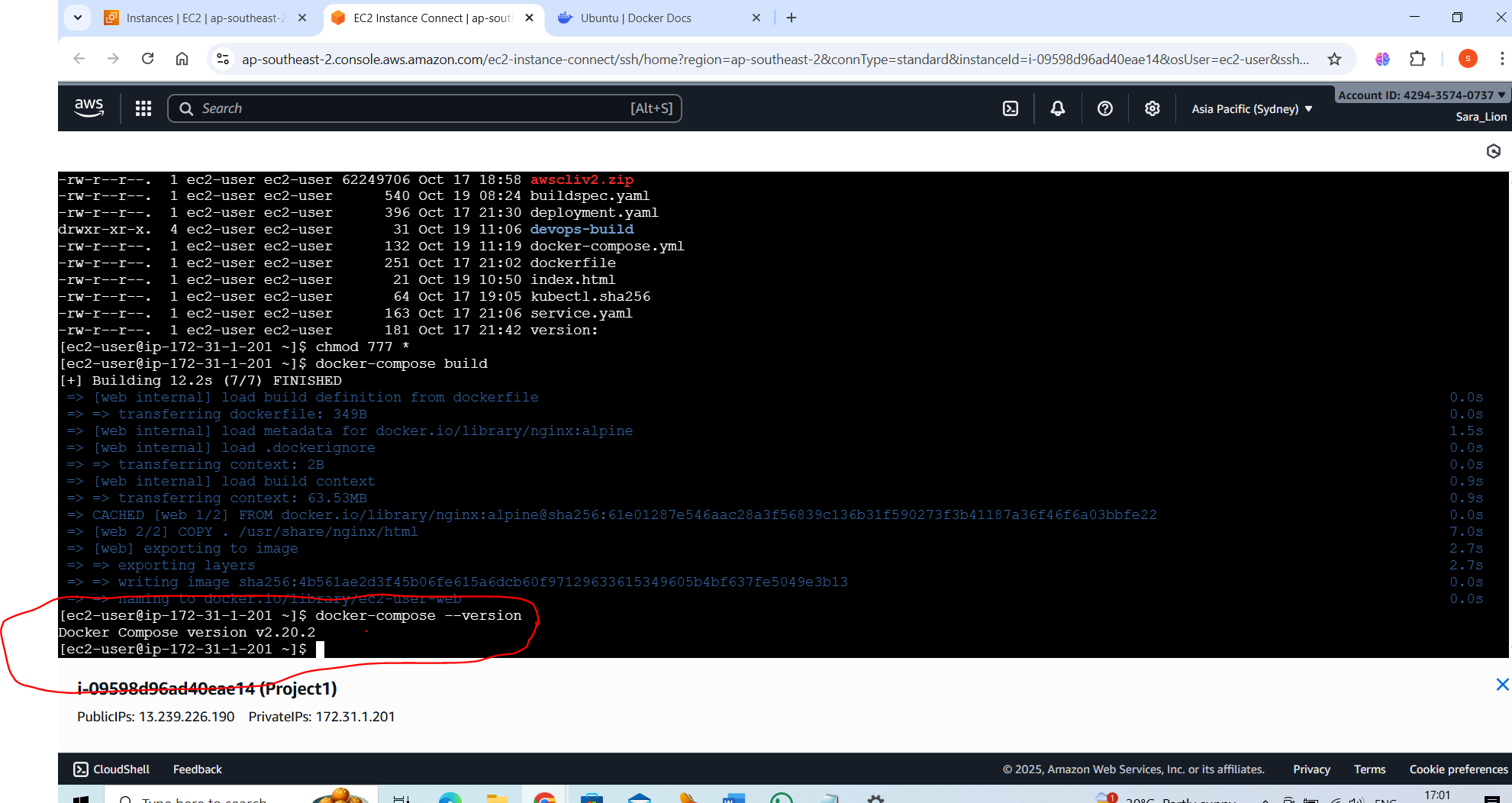
- "80:80"

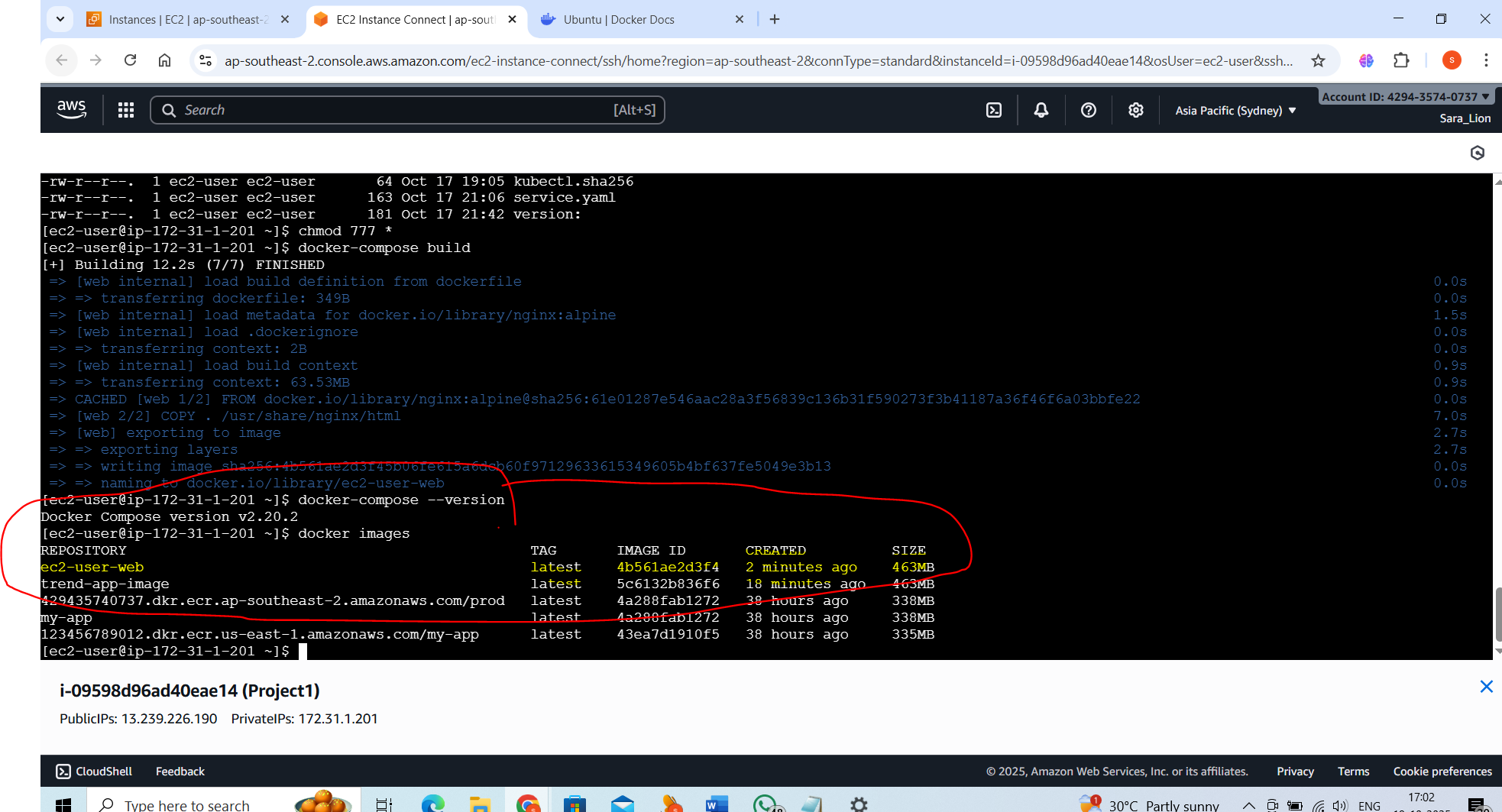
restart: unless-stopped

docker-compose build

docker-compose up

docker-compose ps





**STEP 4:**

Task : created build.sh and deploy.sh

Build.sh :

#!/bin/bash

set -euo pipefail

# replace with your DockerHub username

DOCKER\_USER="${DOCKER\_USER:-saraswathi6}"

IMAGE\_NAME="${DOCKER\_USER}/guvi-react-dev"

echo "Building image ${IMAGE\_NAME}:latest ..."

docker build -t "${IMAGE\_NAME}:latest" .

# tag with build id if provided

TAG="${BUILD\_TAG:-latest}"

docker tag "${IMAGE\_NAME}:latest" "${IMAGE\_NAME}:${TAG}"

echo "Build finished: ${IMAGE\_NAME}:${TAG}"

Deploy.sh:

#!/bin/bash

set -euo pipefail

DOCKER\_USER="${DOCKER\_USER:-saraswathi6}"

IMAGE\_NAME="${DOCKER\_USER}/guvi-react-dev"

TAG="${BUILD\_TAG:-latest}"

echo "Pulling ${IMAGE\_NAME}:${TAG}"

docker pull "${IMAGE\_NAME}:${TAG}"

# write image name into docker-compose env var or override

export IMAGE\_NAME="${IMAGE\_NAME}:${TAG}"

# stop and start

docker compose down || true

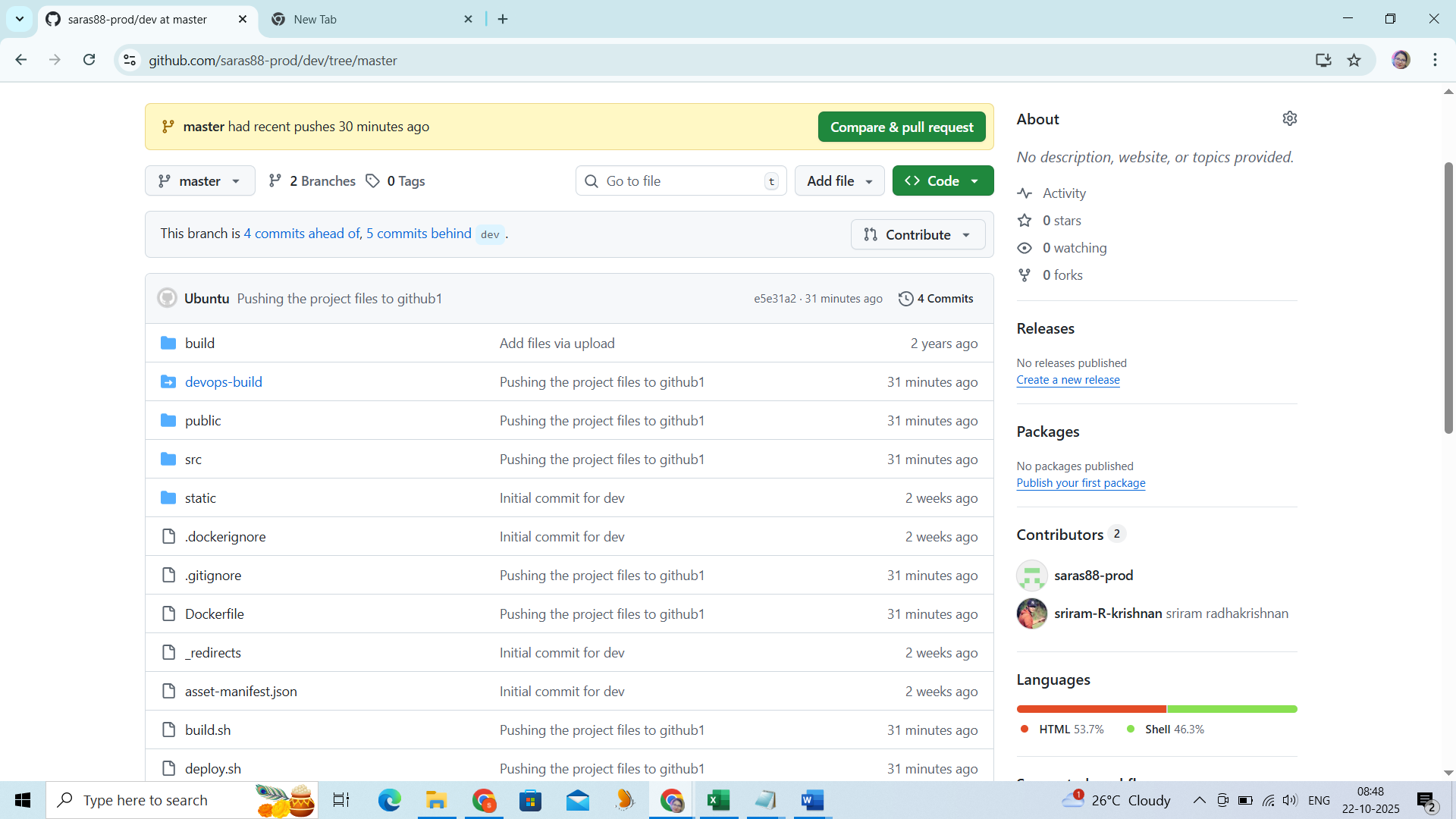
docker compose up -d

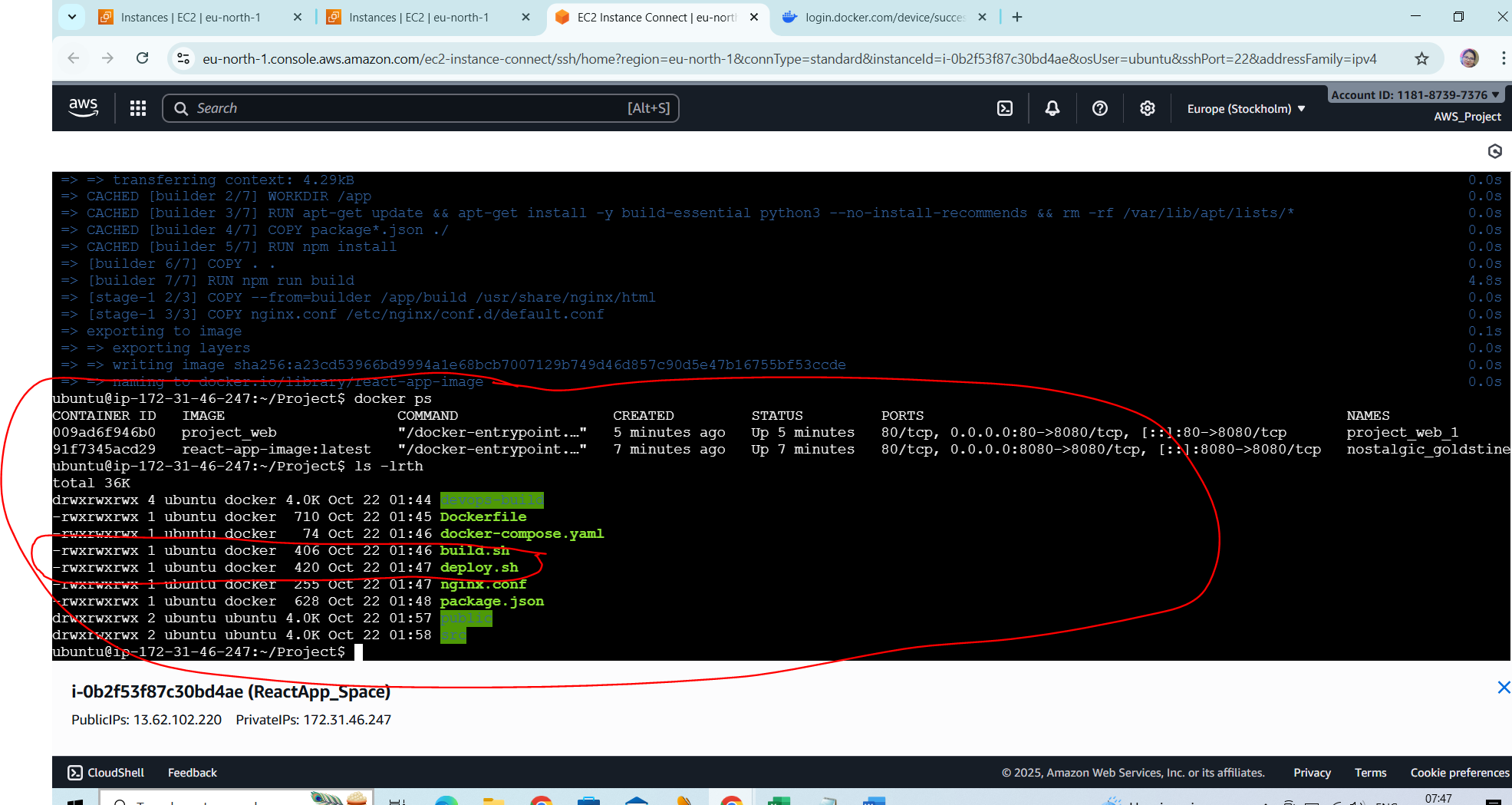
echo "Deployed ${IMAGE\_NAME}:${TAG}"

**Step 5:version control**

Push all the codes from EC2 server to github.

<https://github.com/saras88-prod/dev.git>





Commands:

sudo apt update

sudo apt install git -y

git init

git remote add origin https://github.com/saras88-prod/dev.git

git pull origin master --rebase

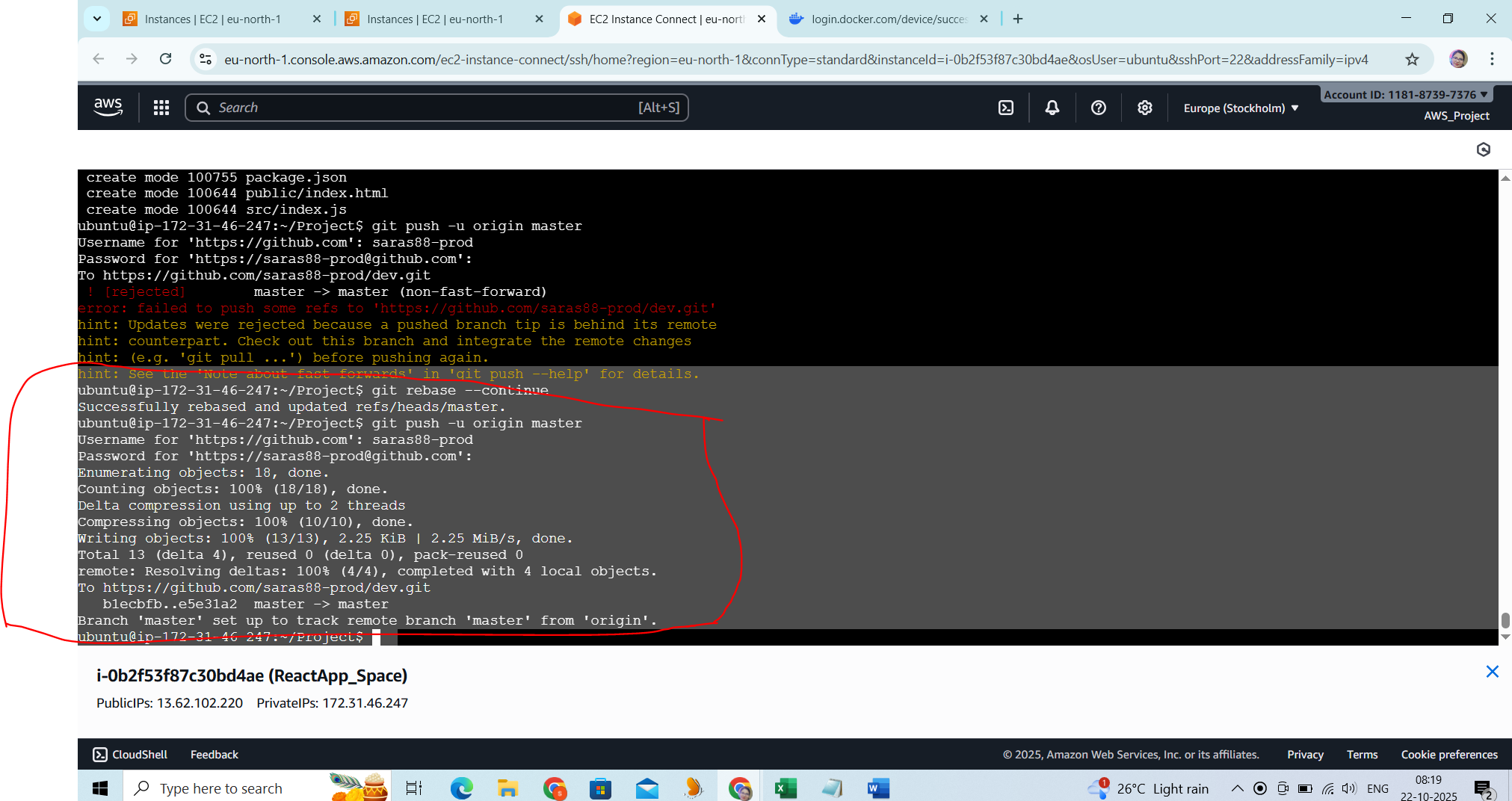
git add .

git commit -m "Pushing the project files to github1"

git push -u origin master

git rebase --continue

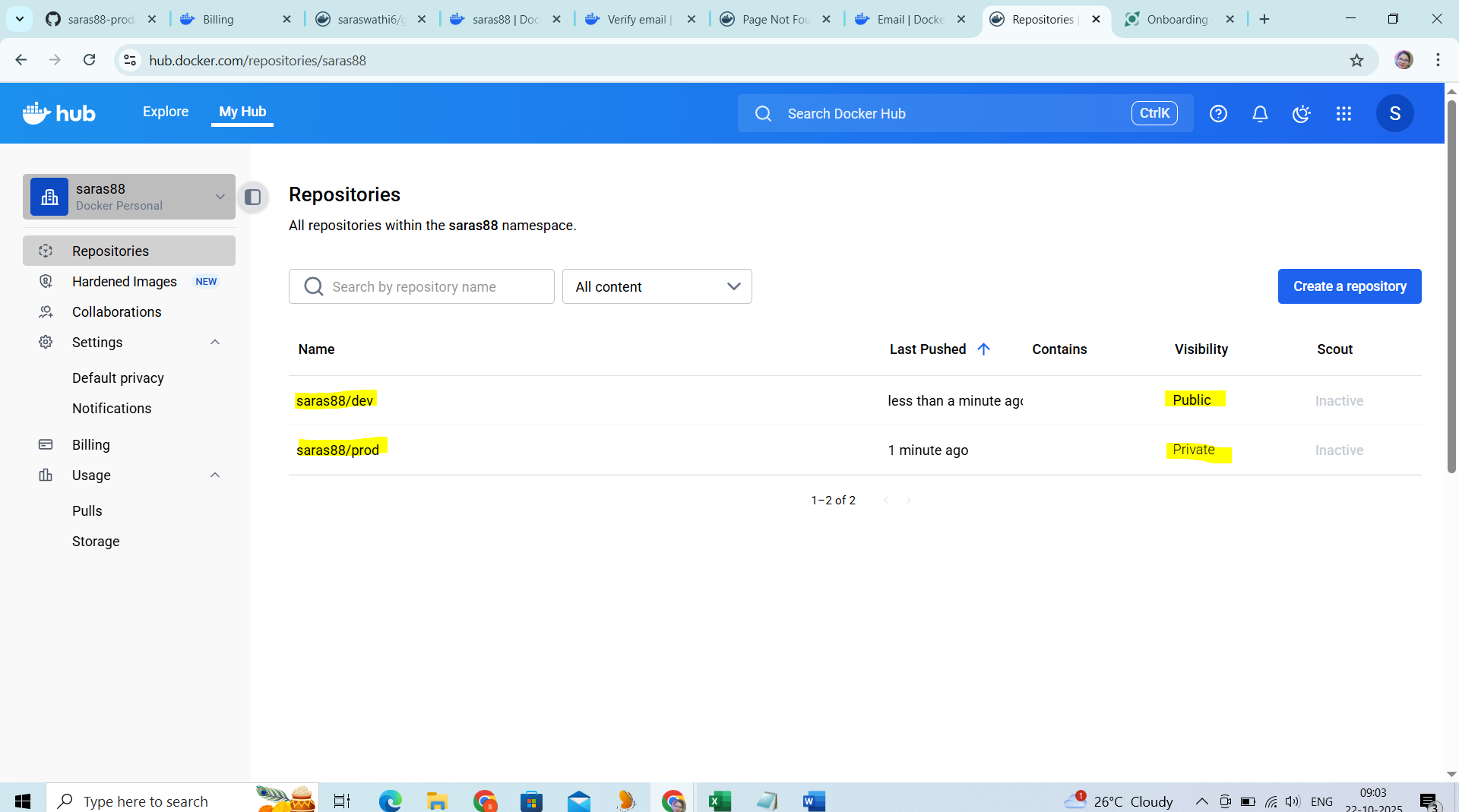
git push -u origin master



Step 6:

Create 2 repositories prod and dev.

<https://hub.docker.com/repositories/saras88>



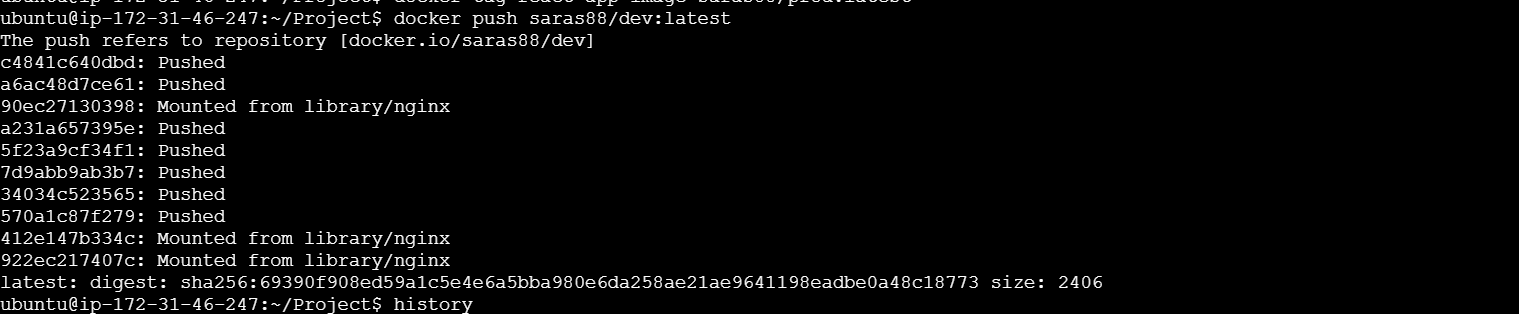
Commands to push the docker images

docker tag react-app-image saras88/dev:latest

docker tag react-app-image saras88/prod:latest

docker push saras88/dev:latest

docker push saras88/prod:latest





Step 7:

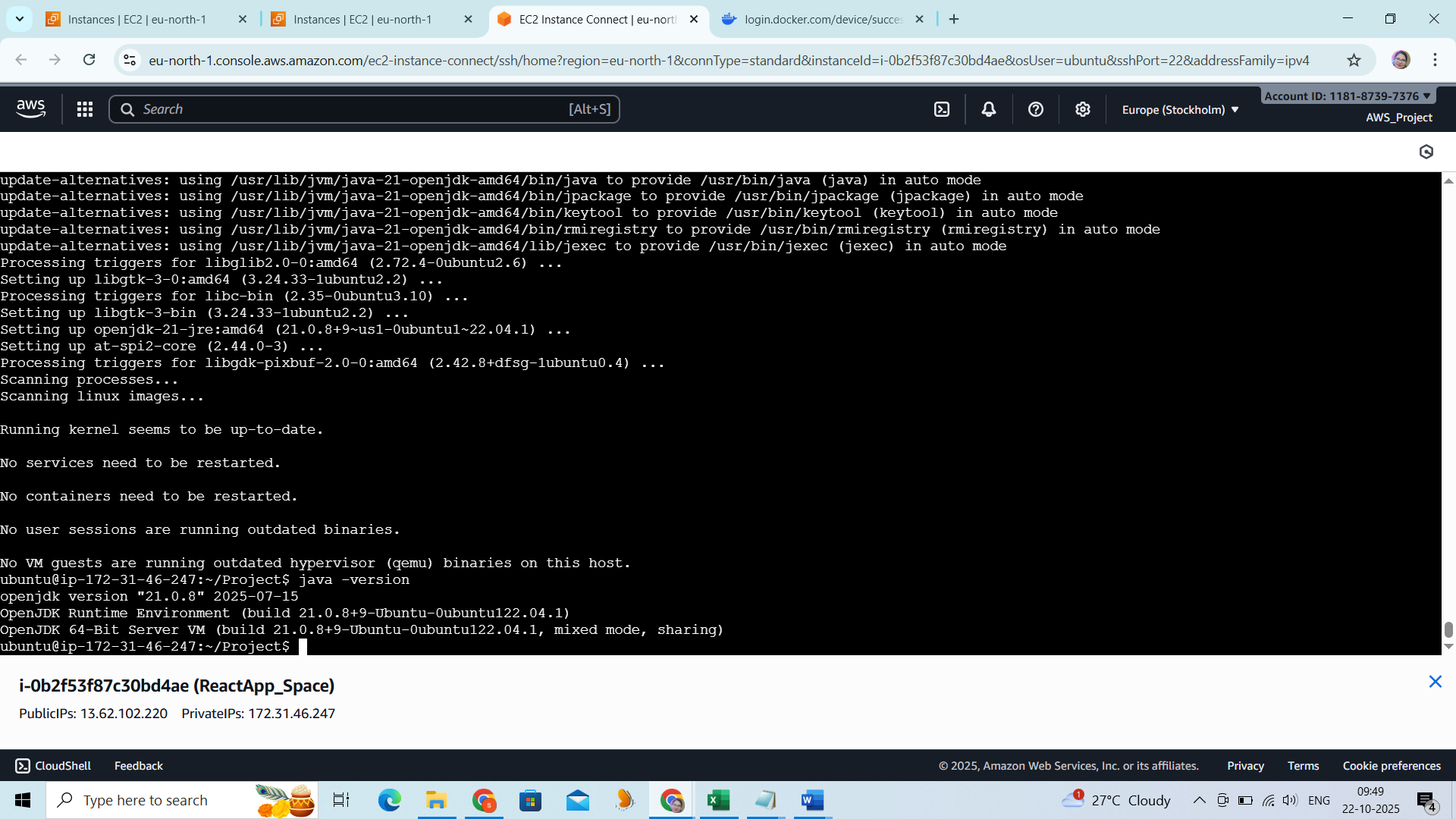
Jenkins installation on the AWS Ubuntu server,First install Java

Commands:

sudo apt update

sudo apt install fontconfig openjdk-21-jre

java -version



Jenkins have been installed and launched successfully:

sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc \

https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

/etc/apt/sources.list.d/jenkins.list **>** /dev/null

sudo apt update

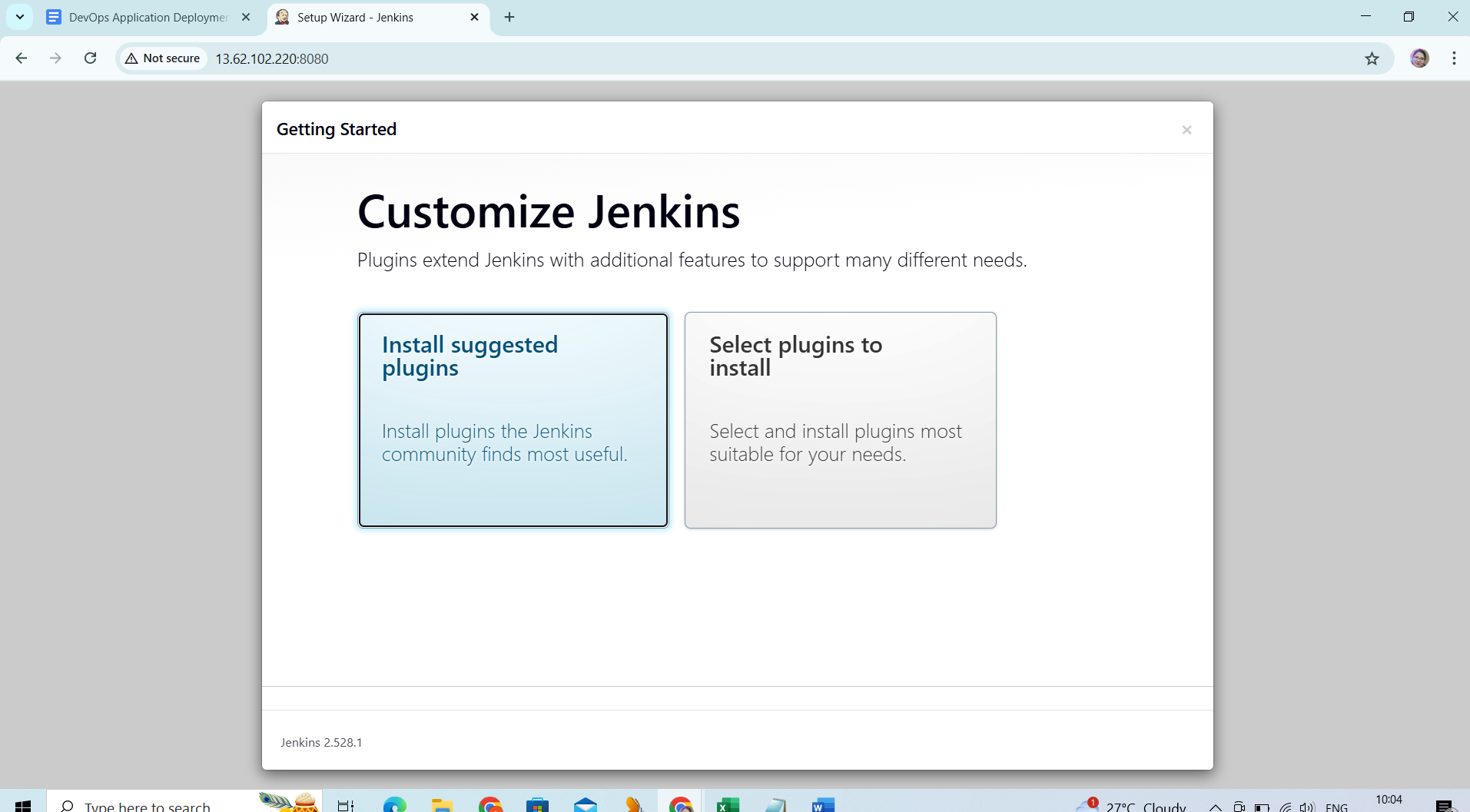
sudo apt install jenkins

sudo systemctl daemon-reexec

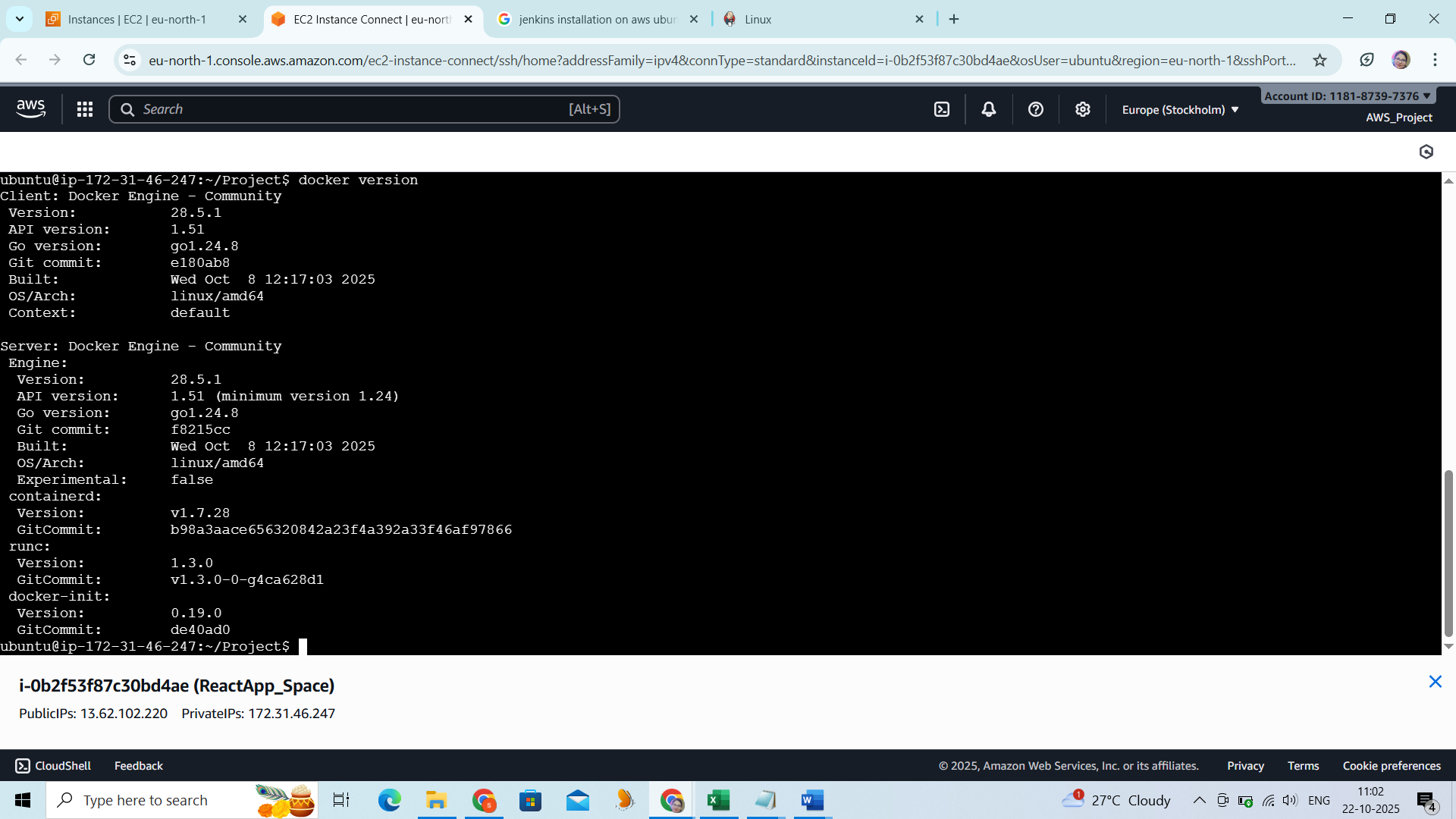
sudo systemctl restart jenkins

sudo systemctl status Jenkins

http://13.62.102.220:8080/



Docker version:



->Jenkins installation on the EC2/Ubuntu server

->Launch the Jenkins using the URL : <http://13.62.102.220:8080/job/Build-and-Push-Docker/7/console>

**(1) Install the following plugins on Jenkins:**

1. Git plugin
2. GitHub plugin
3. Docker Pipeline or Docker plugin
4. Pipeline plugin
5. Restart the Jenkins after the plugins installation

sudo apt install docker.io

sudo usermod -aG docker jenkins

sudo systemctl restart docker

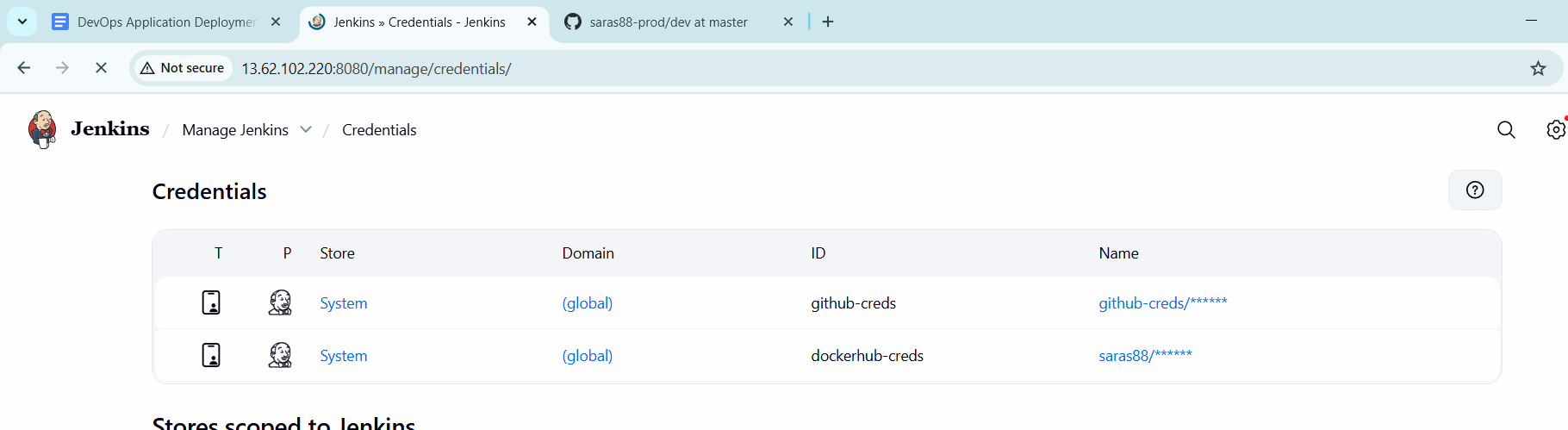
sudo systemctl restart Jenkins

(2) Create the dev and master branches and push our codes on Github

([https://github.com/saras88-prod/dev/](https://github.com/saras88-prod/dev/blob/master/Jenkinsfile)

(3)Configure Docker Hub account and its credentials on the Jenkins.

(4)Jenkins must have access to: GitHub via webhook



**Dockerfile:**

FROM node:18-alpine

WORKDIR /app

COPY package\*.json ./

RUN npm install

COPY . .

RUN npm run build

RUN npm install -g serve

EXPOSE 80

CMD ["serve", "-s", "build", "-l", "80"]

FROM node:18-bullseye-slim AS builder

WORKDIR /app

RUN apt-get update && apt-get install -y build-essential python3 --no-install-recommends && rm -rf /var/lib/apt/lists/\*

COPY package\*.json ./

RUN npm install

COPY . .

RUN npm run build

# production stage (serve with nginx)

FROM nginx:stable-alpine

COPY --from=builder /app/build /usr/share/nginx/html

COPY nginx.conf /etc/nginx/conf.d/default.conf

EXPOSE 8080

CMD ["nginx", "-g", "daemon off;"]

Jenkins job configuration:

**Master**

1. Create Jenkins file to push the docker images from github .
2. Push the dev branch content from github to dockurhub dev image.
3. Push the master mranch content from github to dockhub image prod image

Master Jenkinsfile

pipeline {

agent any

environment {

IMAGE\_NAME = "saras88/prod" // Replace with your DockerHub repo name

IMAGE\_TAG = "latest" // Or use ${BUILD\_NUMBER}, ${GIT\_COMMIT}, etc.

}

stages {

stage('Checkout Code') {

steps {

git credentialsId: 'github-creds', url: 'https://github.com/saras88-prod/dev.git', branch: 'prod'

}

}

stage('Build Docker Image') {

steps {

script {

sh "docker build -t ${IMAGE\_NAME}:${IMAGE\_TAG} ."

}

}

}

stage('Login to Docker Hub') {

steps {

withCredentials([usernamePassword(credentialsId: 'dockerhub-creds', usernameVariable: 'DOCKER\_USER', passwordVariable: 'DOCKER\_PASS')]) {

sh "echo $DOCKER\_PASS | docker login -u $DOCKER\_USER --password-stdin"

}

}

}

stage('Push Docker Image') {

steps {

sh "docker push ${IMAGE\_NAME}:${IMAGE\_TAG}"

}

}

}

post {

always {

cleanWs()

}

success {

echo "✅ Docker image pushed: ${IMAGE\_NAME}:${IMAGE\_TAG}"

}

failure {

echo "❌ Build failed. Image not pushed."

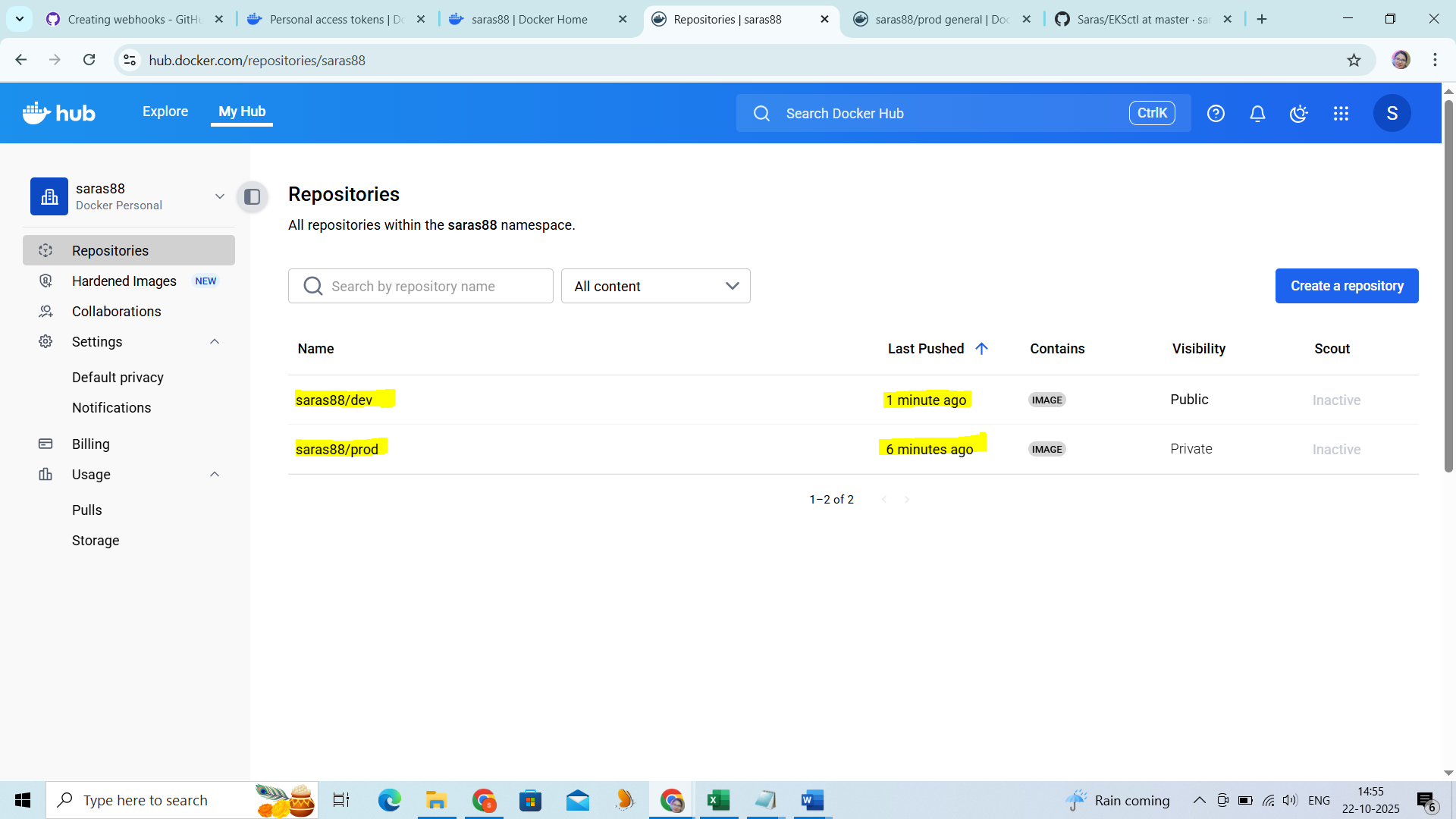
}

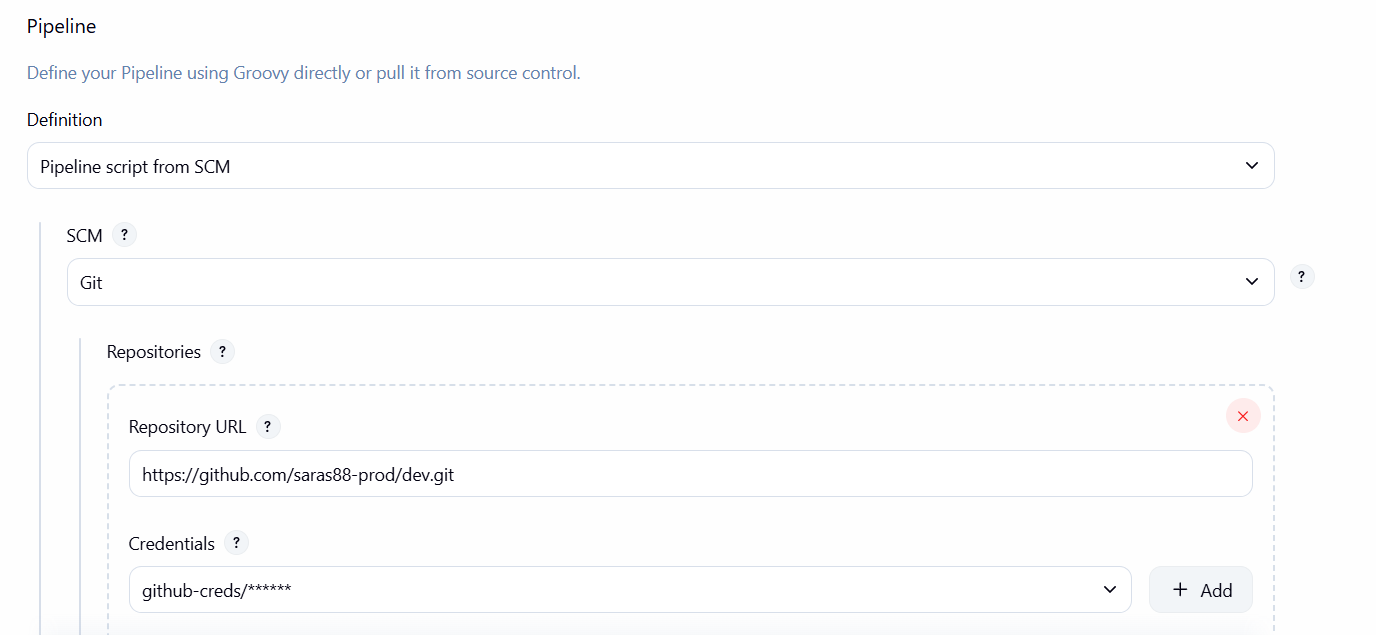
}

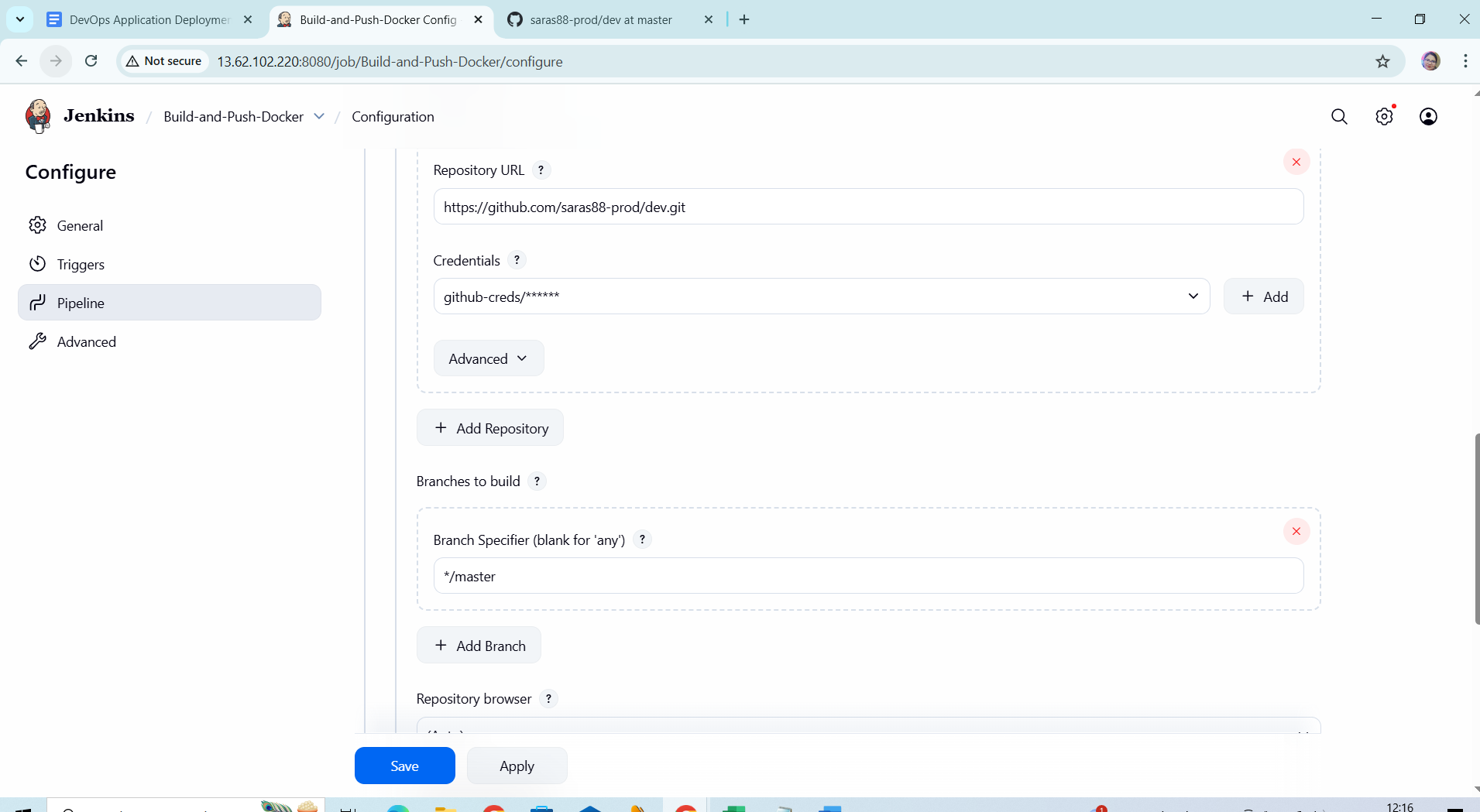
}

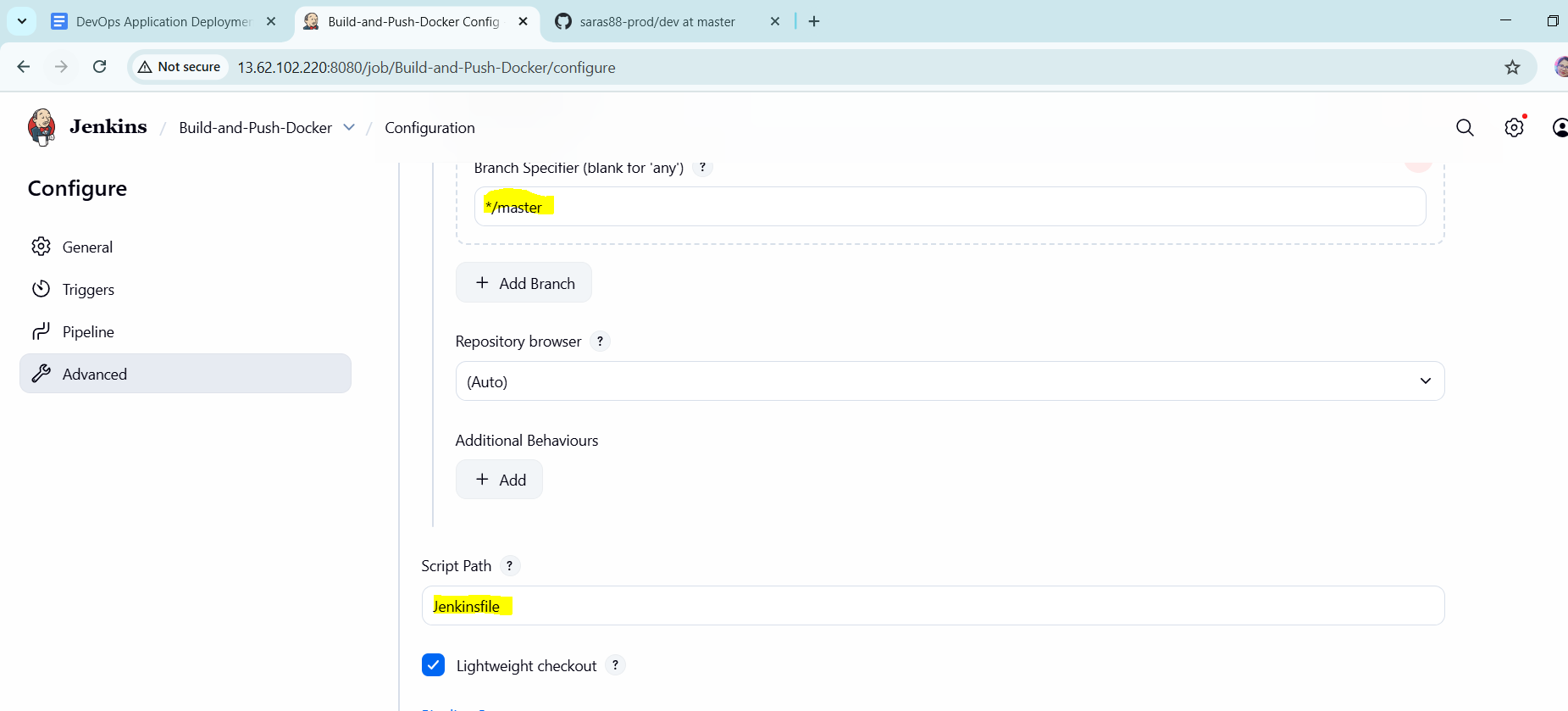
Jenkins config for the pipeline

* Pushed the files from github and built ,deploy the dockers image and pushed them to dockerfile









**Jenkinsfile Console Output:**

Started by user [saraswathi](http://13.62.102.220:8080/user/admin)

Obtained Jenkinsfile from git <https://github.com/saras88-prod/dev.git>

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [Jenkins](http://13.62.102.220:8080/computer/(built-in)/) in /var/lib/jenkins/workspace/Build-and-Push-Docker

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Declarative: Checkout SCM)

[Pipeline] checkout

Selected Git installation does not exist. Using Default

The recommended git tool is: NONE

using credential github-creds

Cloning the remote Git repository

Cloning repository <https://github.com/saras88-prod/dev.git>

> git init /var/lib/jenkins/workspace/Build-and-Push-Docker # timeout=10

Fetching upstream changes from <https://github.com/saras88-prod/dev.git>

> git --version # timeout=10

> git --version # 'git version 2.34.1'

using GIT\_ASKPASS to set credentials

> git fetch --tags --force --progress -- <https://github.com/saras88-prod/dev.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git config remote.origin.url <https://github.com/saras88-prod/dev.git> # timeout=10

> git config --add remote.origin.fetch +refs/heads/\*:refs/remotes/origin/\* # timeout=10

Avoid second fetch

> git rev-parse refs/remotes/origin/master^{commit} # timeout=10

Checking out Revision 34c0297421bb48b81f79bd8f42fe5ff72870635b (refs/remotes/origin/master)

> git config core.sparsecheckout # timeout=10

> git checkout -f 34c0297421bb48b81f79bd8f42fe5ff72870635b # timeout=10

Commit message: "Update docker-compose.yml"

> git rev-list --no-walk f81a18e7ac14ccf2c20b9a275cef1f51956f5432 # timeout=10

[Pipeline] }

[Pipeline] // stage

[Pipeline] withEnv

[Pipeline] {

[Pipeline] withEnv

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Checkout)

[Pipeline] checkout

Selected Git installation does not exist. Using Default

The recommended git tool is: NONE

using credential github-creds

> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Build-and-Push-Docker/.git # timeout=10

Fetching changes from the remote Git repository

> git config remote.origin.url <https://github.com/saras88-prod/dev.git> # timeout=10

Fetching upstream changes from <https://github.com/saras88-prod/dev.git>

> git --version # timeout=10

> git --version # 'git version 2.34.1'

using GIT\_ASKPASS to set credentials

> git fetch --tags --force --progress -- <https://github.com/saras88-prod/dev.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git rev-parse refs/remotes/origin/master^{commit} # timeout=10

Checking out Revision 34c0297421bb48b81f79bd8f42fe5ff72870635b (refs/remotes/origin/master)

> git config core.sparsecheckout # timeout=10

> git checkout -f 34c0297421bb48b81f79bd8f42fe5ff72870635b # timeout=10

Commit message: "Update docker-compose.yml"

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Build Docker Image)

[Pipeline] script

[Pipeline] {

[Pipeline] isUnix

[Pipeline] withEnv

[Pipeline] {

[Pipeline] sh

+ docker build -t your-app:null .

#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile

#1 transferring dockerfile: 1.42kB done

#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/nginx:stable-alpine

#2 DONE 0.9s

#3 [internal] load metadata for docker.io/library/node:18-bullseye-slim

#3 DONE 1.0s

#4 [internal] load .dockerignore

#4 transferring context: 117B done

#4 DONE 0.0s

#5 [builder 1/7] FROM docker.io/library/node:18-bullseye-slim@sha256:d69fb189fa7765636655db043a6a9e6be5ddf94bd1a8dc33fd0bcf4664606a9e

#5 DONE 0.0s

#6 [stage-2 1/3] FROM docker.io/library/nginx:stable-alpine@sha256:30f1c0d78e0ad60901648be663a710bdadf19e4c10ac6782c235200619158284

#6 DONE 0.0s

#7 [internal] load build context

#7 transferring context: 2.61MB 0.0s done

#7 DONE 0.0s

#8 [builder 2/7] WORKDIR /app

#8 CACHED

#9 [builder 3/7] RUN apt-get update && apt-get install -y build-essential python3 --no-install-recommends && rm -rf /var/lib/apt/lists/\*

#9 CACHED

#10 [builder 4/7] COPY package\*.json ./

#10 DONE 0.0s

#11 [builder 5/7] RUN npm install

#11 47.56 npm warn deprecated rollup-plugin-terser@7.0.2: This package has been deprecated and is no longer maintained. Please use @rollup/plugin-terser

#11 47.67 npm warn deprecated sourcemap-codec@1.4.8: Please use @jridgewell/sourcemap-codec instead

#11 48.03 npm warn deprecated workbox-cacheable-response@6.6.0: workbox-background-sync@6.6.0

#11 48.42 npm warn deprecated stable@0.1.8: Modern JS already guarantees Array#sort() is a stable sort, so this library is deprecated. See the compatibility table on MDN: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort#browser_compatibility>

#11 48.51 npm warn deprecated q@1.5.1: You or someone you depend on is using Q, the JavaScript Promise library that gave JavaScript developers strong feelings about promises. They can almost certainly migrate to the native JavaScript promise now. Thank you literally everyone for joining me in this bet against the odds. Be excellent to each other.

#11 48.51 npm warn deprecated

#11 48.51 npm warn deprecated (For a CapTP with native promises, see @endo/eventual-send and @endo/captp)

#11 48.85 npm warn deprecated workbox-google-analytics@6.6.0: It is not compatible with newer versions of GA starting with v4, as long as you are using GAv3 it should be ok, but the package is not longer being maintained

#11 49.18 npm warn deprecated w3c-hr-time@1.0.2: Use your platform's native performance.now() and performance.timeOrigin.

#11 49.21 npm warn deprecated domexception@2.0.1: Use your platform's native DOMException instead

#11 49.21 npm warn deprecated abab@2.0.6: Use your platform's native atob() and btoa() methods instead

#11 49.45 npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.

#11 49.48 npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported

#11 49.85 npm warn deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported

#11 50.38 npm warn deprecated @humanwhocodes/object-schema@2.0.3: Use @eslint/object-schema instead

#11 50.49 npm warn deprecated @humanwhocodes/config-array@0.13.0: Use @eslint/config-array instead

#11 50.70 npm warn deprecated @babel/plugin-proposal-private-methods@7.18.6: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-private-methods instead.

#11 50.71 npm warn deprecated @babel/plugin-proposal-optional-chaining@7.21.0: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-optional-chaining instead.

#11 50.72 npm warn deprecated @babel/plugin-proposal-numeric-separator@7.18.6: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-numeric-separator instead.

#11 50.72 npm warn deprecated @babel/plugin-proposal-nullish-coalescing-operator@7.18.6: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-nullish-coalescing-operator instead.

#11 50.73 npm warn deprecated @babel/plugin-proposal-class-properties@7.18.6: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-class-properties instead.

#11 52.54 npm warn deprecated source-map@0.8.0-beta.0: The work that was done in this beta branch won't be included in future versions

#11 52.55 npm warn deprecated @babel/plugin-proposal-private-property-in-object@7.21.11: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-private-property-in-object instead.

#11 52.85 npm warn deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.

#11 54.03 npm warn deprecated eslint@8.57.1: This version is no longer supported. Please see <https://eslint.org/version-support> for other options.

#11 56.00

#11 56.00 added 1323 packages, and audited 1324 packages in 56s

#11 56.00

#11 56.00 271 packages are looking for funding

#11 56.00 run `npm fund` for details

#11 56.00

#11 56.00 9 vulnerabilities (3 moderate, 6 high)

#11 56.00

#11 56.00 To address all issues (including breaking changes), run:

#11 56.00 npm audit fix --force

#11 56.00

#11 56.00 Run `npm audit` for details.

#11 56.00 npm notice

#11 56.00 npm notice New major version of npm available! 10.8.2 -> 11.6.2

#11 56.00 npm notice Changelog: <https://github.com/npm/cli/releases/tag/v11.6.2>

#11 56.00 npm notice To update run: npm install -g npm@11.6.2

#11 56.00 npm notice

#11 DONE 56.4s

#12 [builder 6/7] COPY . .

#12 DONE 0.1s

#13 [builder 7/7] RUN npm run build

#13 0.326

#13 0.326 > project@1.0.0 build

#13 0.326 > react-scripts build

#13 0.326

#13 1.125 Creating an optimized production build...

#13 4.750 Compiled successfully.

#13 4.750

#13 4.750 File sizes after gzip:

#13 4.750

#13 4.757 45.23 kB build/static/js/main.e078f3b1.js

#13 4.758

#13 4.758 The project was built assuming it is hosted at /.

#13 4.758 You can control this with the homepage field in your package.json.

#13 4.758

#13 4.758 The build folder is ready to be deployed.

#13 4.758 You may serve it with a static server:

#13 4.758

#13 4.758 npm install -g serve

#13 4.758 serve -s build

#13 4.758

#13 4.758 Find out more about deployment here:

#13 4.758

#13 4.758 <https://cra.link/deployment>

#13 4.758

#13 DONE 4.8s

#14 [stage-2 2/3] COPY --from=builder /app/build /usr/share/nginx/html

#14 CACHED

#15 [stage-2 3/3] COPY nginx.conf /etc/nginx/conf.d/default.conf

#15 DONE 0.0s

#16 exporting to image

#16 exporting layers 0.0s done

#16 writing image sha256:5f7eff11894e258e0e2d19ff99ad19d619db55479ffe246eaaadb7a14a790756 done

#16 naming to docker.io/library/your-app:null done

#16 DONE 0.1s

[Pipeline] }

[Pipeline] // withEnv

Did you forget the `def` keyword? WorkflowScript seems to be setting a field named dockerImage (to a value of type Image) which could lead to memory leaks or other issues.

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Push Docker Image)

[Pipeline] script

[Pipeline] {

[Pipeline] withEnv

[Pipeline] {

[Pipeline] withDockerRegistry

$ docker login -u saras88 -p \*\*\*\*\*\*\*\* <https://index.docker.io/v1/>

WARNING! Using --password via the CLI is insecure. Use --password-stdin.

WARNING! Your credentials are stored unencrypted in '/var/lib/jenkins/workspace/Build-and-Push-Docker@tmp/82f98acb-2195-4ec4-9c32-5d412253587b/config.json'.

Configure a credential helper to remove this warning. See

<https://docs.docker.com/go/credential-store/>

Login Succeeded

[Pipeline] {

[Pipeline] }

[Pipeline] // withDockerRegistry

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Declarative: Post Actions)

[Pipeline] cleanWs

[WS-CLEANUP] Deleting project workspace...

[WS-CLEANUP] Deferred wipeout is used...

[WS-CLEANUP] done

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // node

[Pipeline] End of Pipeline

Finished: SUCCESS

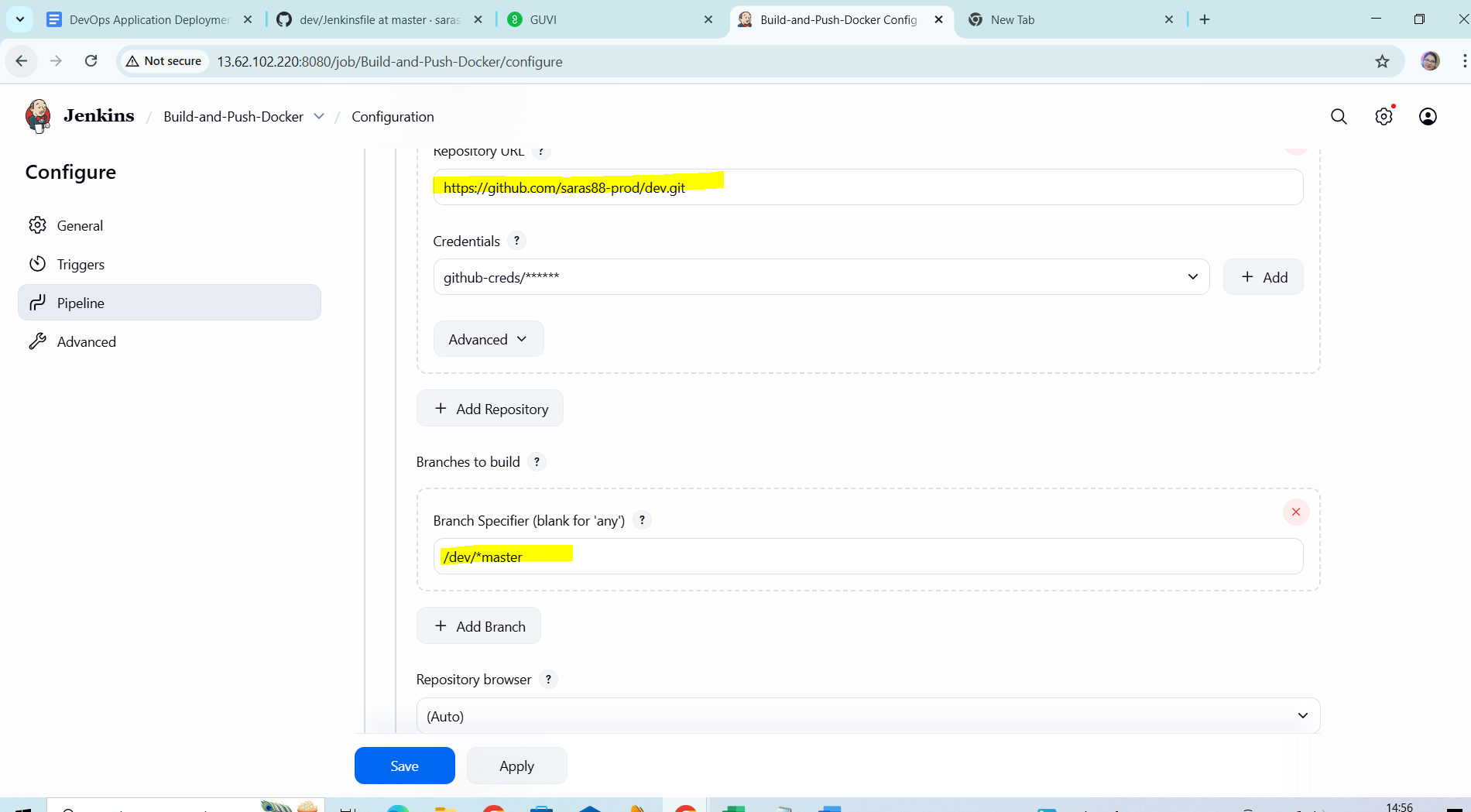
Commands used to install npm:

npm install

sudo apt install npm

Pipeline push from dev repo:

Do Jenkins config to build and deploy code from github and push them to dockerhub dev image.



**Jenkinsfile:**

pipeline {

agent any

environment {

IMAGE\_NAME = "saras88/dev" // Replace with your DockerHub repo name

IMAGE\_TAG = "latest" // Or use ${BUILD\_NUMBER}, ${GIT\_COMMIT}, etc.

}

stages {

stage('Checkout Code') {

steps {

git credentialsId: 'github-creds', url: 'https://github.com/saras88-prod/dev.git', branch: 'dev'

}

}

stage('Build Docker Image') {

steps {

script {

sh "docker build -t ${IMAGE\_NAME}:${IMAGE\_TAG} ."

}

}

}

stage('Login to Docker Hub') {

steps {

withCredentials([usernamePassword(credentialsId: 'dockerhub-creds', usernameVariable: 'DOCKER\_USER', passwordVariable: 'DOCKER\_PASS')]) {

sh "echo $DOCKER\_PASS | docker login -u $DOCKER\_USER --password-stdin"

}

}

}

stage('Push Docker Image') {

steps {

sh "docker push ${IMAGE\_NAME}:${IMAGE\_TAG}"

}

}

}

post {

always {

cleanWs()

}

success {

echo "✅ Docker image pushed: ${IMAGE\_NAME}:${IMAGE\_TAG}"

}

failure {

echo "❌ Build failed. Image not pushed."

}

}

}

**Jenkins console output:**

Started by user [saraswathi](http://13.62.102.220:8080/user/admin)

Obtained Jenkinsfile from git <https://github.com/saras88-prod/dev.git>

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [Jenkins](http://13.62.102.220:8080/computer/(built-in)/) in /var/lib/jenkins/workspace/Build-and-Push-Docker

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Declarative: Checkout SCM)

[Pipeline] checkout

Selected Git installation does not exist. Using Default

The recommended git tool is: NONE

using credential github-creds

Cloning the remote Git repository

Cloning repository <https://github.com/saras88-prod/dev.git>

> git init /var/lib/jenkins/workspace/Build-and-Push-Docker # timeout=10

Fetching upstream changes from <https://github.com/saras88-prod/dev.git>

> git --version # timeout=10

> git --version # 'git version 2.34.1'

using GIT\_ASKPASS to set credentials

> git fetch --tags --force --progress -- <https://github.com/saras88-prod/dev.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git config remote.origin.url <https://github.com/saras88-prod/dev.git> # timeout=10

> git config --add remote.origin.fetch +refs/heads/\*:refs/remotes/origin/\* # timeout=10

Avoid second fetch

> git rev-parse refs/remotes/origin/dev^{commit} # timeout=10

Checking out Revision 785a65230c46e718bdc303efe6fdc15509b71db5 (refs/remotes/origin/dev)

> git config core.sparsecheckout # timeout=10

> git checkout -f 785a65230c46e718bdc303efe6fdc15509b71db5 # timeout=10

Commit message: "Add Jenkins pipeline for Docker image build and push"

> git rev-list --no-walk 785a65230c46e718bdc303efe6fdc15509b71db5 # timeout=10

[Pipeline] }

[Pipeline] // stage

[Pipeline] withEnv

[Pipeline] {

[Pipeline] withEnv

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Checkout Code)

[Pipeline] git

Selected Git installation does not exist. Using Default

The recommended git tool is: NONE

using credential github-creds

> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Build-and-Push-Docker/.git # timeout=10

Fetching changes from the remote Git repository

> git config remote.origin.url <https://github.com/saras88-prod/dev.git> # timeout=10

Fetching upstream changes from <https://github.com/saras88-prod/dev.git>

> git --version # timeout=10

> git --version # 'git version 2.34.1'

using GIT\_ASKPASS to set credentials

> git fetch --tags --force --progress -- <https://github.com/saras88-prod/dev.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git rev-parse refs/remotes/origin/dev^{commit} # timeout=10

Checking out Revision 785a65230c46e718bdc303efe6fdc15509b71db5 (refs/remotes/origin/dev)

> git config core.sparsecheckout # timeout=10

> git checkout -f 785a65230c46e718bdc303efe6fdc15509b71db5 # timeout=10

> git branch -a -v --no-abbrev # timeout=10

> git checkout -b dev 785a65230c46e718bdc303efe6fdc15509b71db5 # timeout=10

Commit message: "Add Jenkins pipeline for Docker image build and push"

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Build Docker Image)

[Pipeline] script

[Pipeline] {

[Pipeline] sh

+ docker build -t saras88/dev:latest .

#0 building with "default" instance using docker driver

#1 [internal] load build definition from dockerfile

#1 transferring dockerfile: 750B done

#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/node:18-bullseye-slim

#2 ...

#3 [auth] library/nginx:pull token for registry-1.docker.io

#3 DONE 0.0s

#4 [auth] library/node:pull token for registry-1.docker.io

#4 DONE 0.0s

#5 [internal] load metadata for docker.io/library/nginx:stable-alpine

#5 ...

#2 [internal] load metadata for docker.io/library/node:18-bullseye-slim

#2 DONE 0.9s

#5 [internal] load metadata for docker.io/library/nginx:stable-alpine

#5 DONE 0.9s

#6 [internal] load .dockerignore

#6 transferring context: 117B done

#6 DONE 0.0s

#7 [builder 1/7] FROM docker.io/library/node:18-bullseye-slim@sha256:d69fb189fa7765636655db043a6a9e6be5ddf94bd1a8dc33fd0bcf4664606a9e

#7 DONE 0.0s

#8 [stage-1 1/3] FROM docker.io/library/nginx:stable-alpine@sha256:30f1c0d78e0ad60901648be663a710bdadf19e4c10ac6782c235200619158284

#8 DONE 0.0s

#9 [internal] load build context

#9 transferring context: 2.61MB 0.0s done

#9 DONE 0.0s

#10 [builder 4/7] COPY package\*.json ./

#10 CACHED

#11 [builder 2/7] WORKDIR /app

#11 CACHED

#12 [builder 3/7] RUN apt-get update && apt-get install -y build-essential python3 --no-install-recommends && rm -rf /var/lib/apt/lists/\*

#12 CACHED

#13 [builder 5/7] RUN npm install

#13 CACHED

#14 [builder 6/7] COPY . .

#14 CACHED

#15 [stage-1 2/3] COPY --from=builder /app/build /usr/share/nginx/html

#15 CACHED

#16 [builder 7/7] RUN npm run build

#16 CACHED

#17 [stage-1 3/3] COPY nginx.conf /etc/nginx/conf.d/default.conf

#17 CACHED

#18 exporting to image

#18 exporting layers done

#18 writing image sha256:ca7cf39f073a5838fca9106b318508f5b1e5b21225dc97fa1bc4b4e96109ce24 done

#18 naming to docker.io/saras88/dev:latest done

#18 DONE 0.0s

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Login to Docker Hub)

[Pipeline] withCredentials

Masking supported pattern matches of $DOCKER\_PASS

[Pipeline] {

[Pipeline] sh

Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.

Affected argument(s) used the following variable(s): [DOCKER\_PASS]

See <https://jenkins.io/redirect/groovy-string-interpolation> for details.

+ echo \*\*\*\*

+ docker login -u saras88 --password-stdin

WARNING! Your credentials are stored unencrypted in '/var/lib/jenkins/.docker/config.json'.

Configure a credential helper to remove this warning. See

<https://docs.docker.com/go/credential-store/>

Login Succeeded

[Pipeline] }

[Pipeline] // withCredentials

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Push Docker Image)

[Pipeline] sh

+ docker push saras88/dev:latest

The push refers to repository [docker.io/saras88/dev]

26f829d6cb57: Preparing

a6ac48d7ce61: Preparing

90ec27130398: Preparing

a231a657395e: Preparing

5f23a9cf34f1: Preparing

7d9abb9ab3b7: Preparing

34034c523565: Preparing

570a1c87f279: Preparing

412e147b334c: Preparing

922ec217407c: Preparing

7d9abb9ab3b7: Waiting

34034c523565: Waiting

570a1c87f279: Waiting

412e147b334c: Waiting

922ec217407c: Waiting

5f23a9cf34f1: Layer already exists

90ec27130398: Layer already exists

a231a657395e: Layer already exists

a6ac48d7ce61: Layer already exists

7d9abb9ab3b7: Layer already exists

34034c523565: Layer already exists

412e147b334c: Layer already exists

570a1c87f279: Layer already exists

922ec217407c: Layer already exists

26f829d6cb57: Pushed

latest: digest: sha256:1bad00d00b9736ed6d7ba0949c8fe3dc7d74e949c4149d7e776481643e2b530d size: 2406

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Declarative: Post Actions)

[Pipeline] cleanWs

[WS-CLEANUP] Deleting project workspace...

[WS-CLEANUP] Deferred wipeout is used...

[WS-CLEANUP] done

[Pipeline] echo

✅ Docker image pushed: saras88/dev:latest

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // withEnv

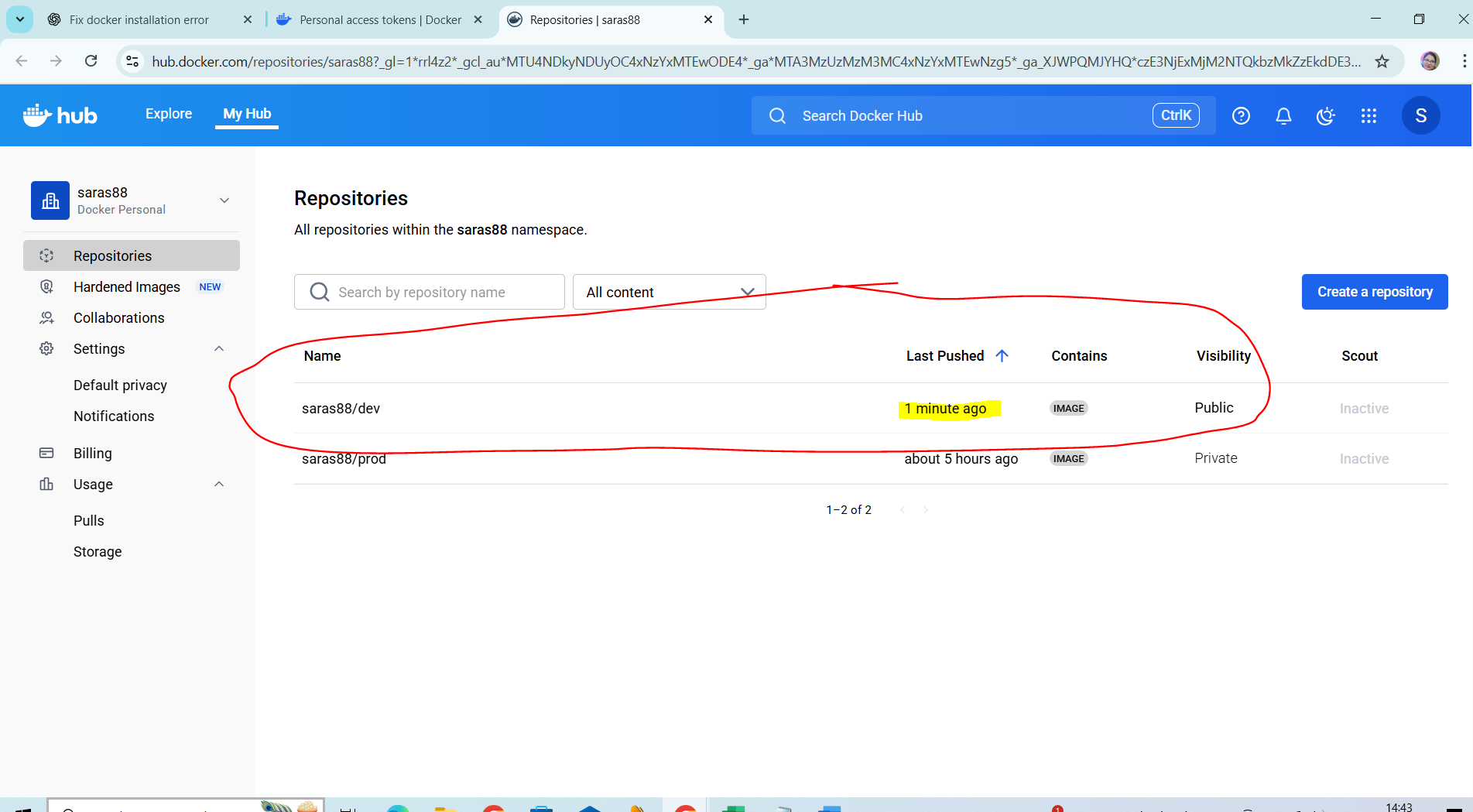
[Pipeline] }

[Pipeline] // node

[Pipeline] End of Pipeline

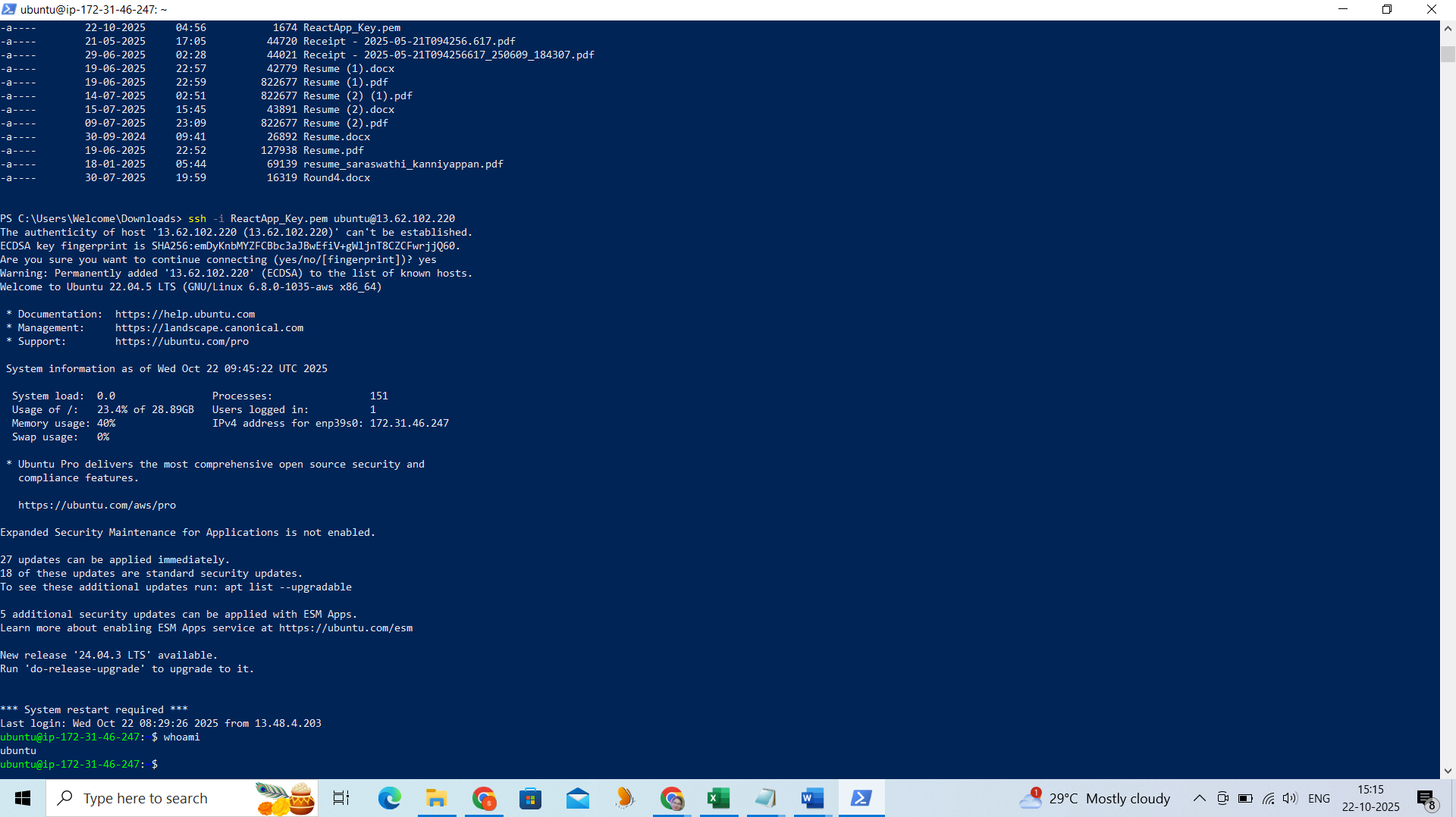
Finished: SUCCESS

**DOCKHUB Image:**



Launched the machine outside the AWS using the key as per the given configuration.

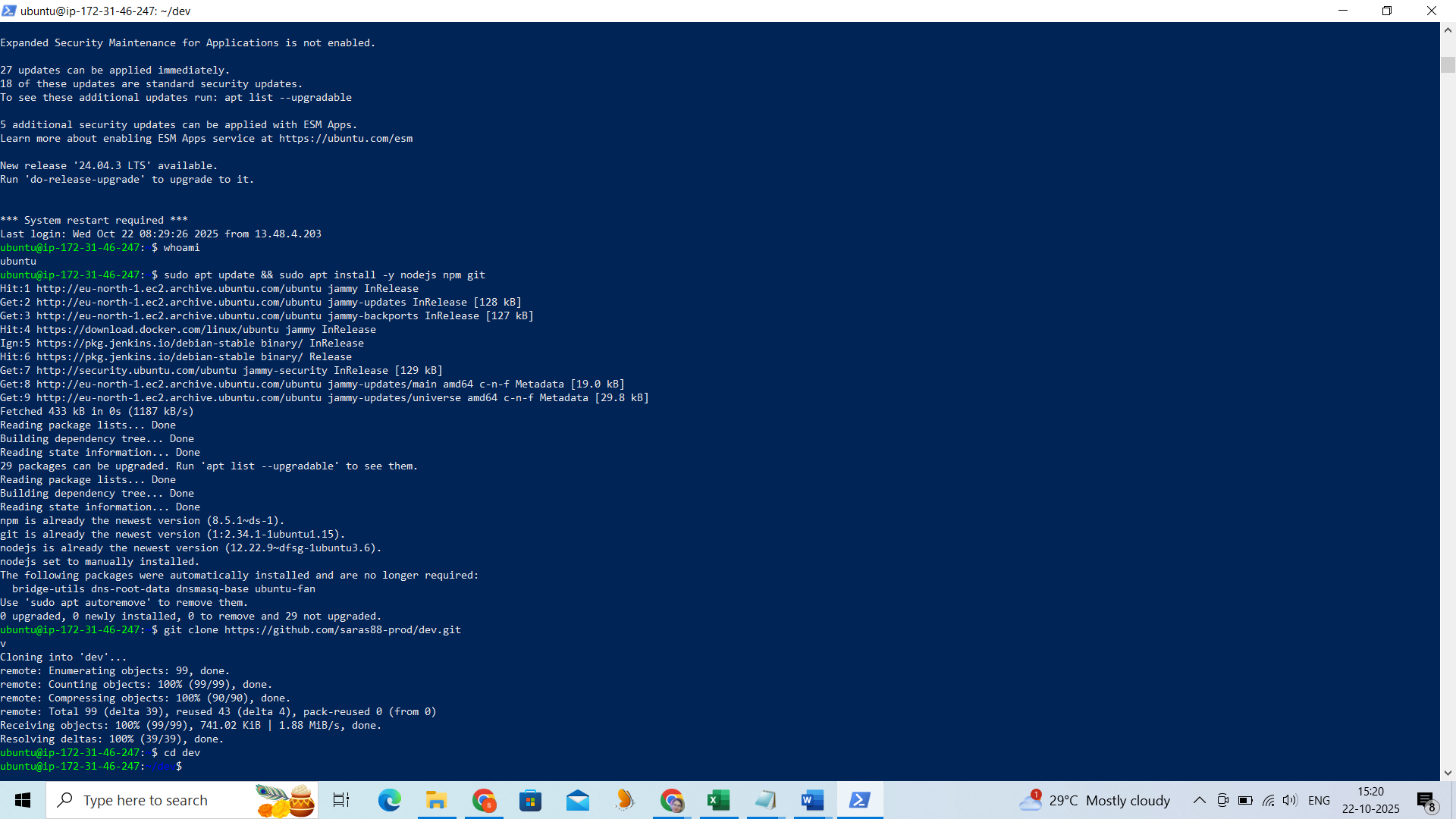
sudo apt update && sudo apt install -y nodejs npm git



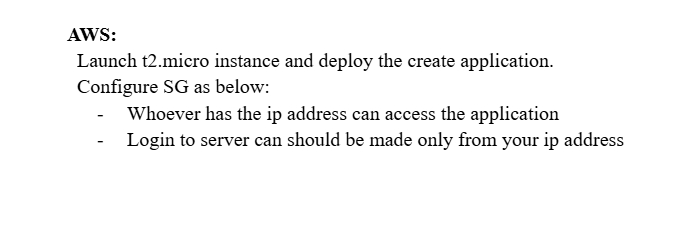
Clone your repo:

git clone https://github.com/saras88-prod/dev.git

cd dev



Install and run:



npm install

npm run build # if applicable

npm start # or serve -s build, based on your app

Install Node.js 18 :



Install Node.js 18

Launched the reach application after the npm installation.

cd Project/

npm install

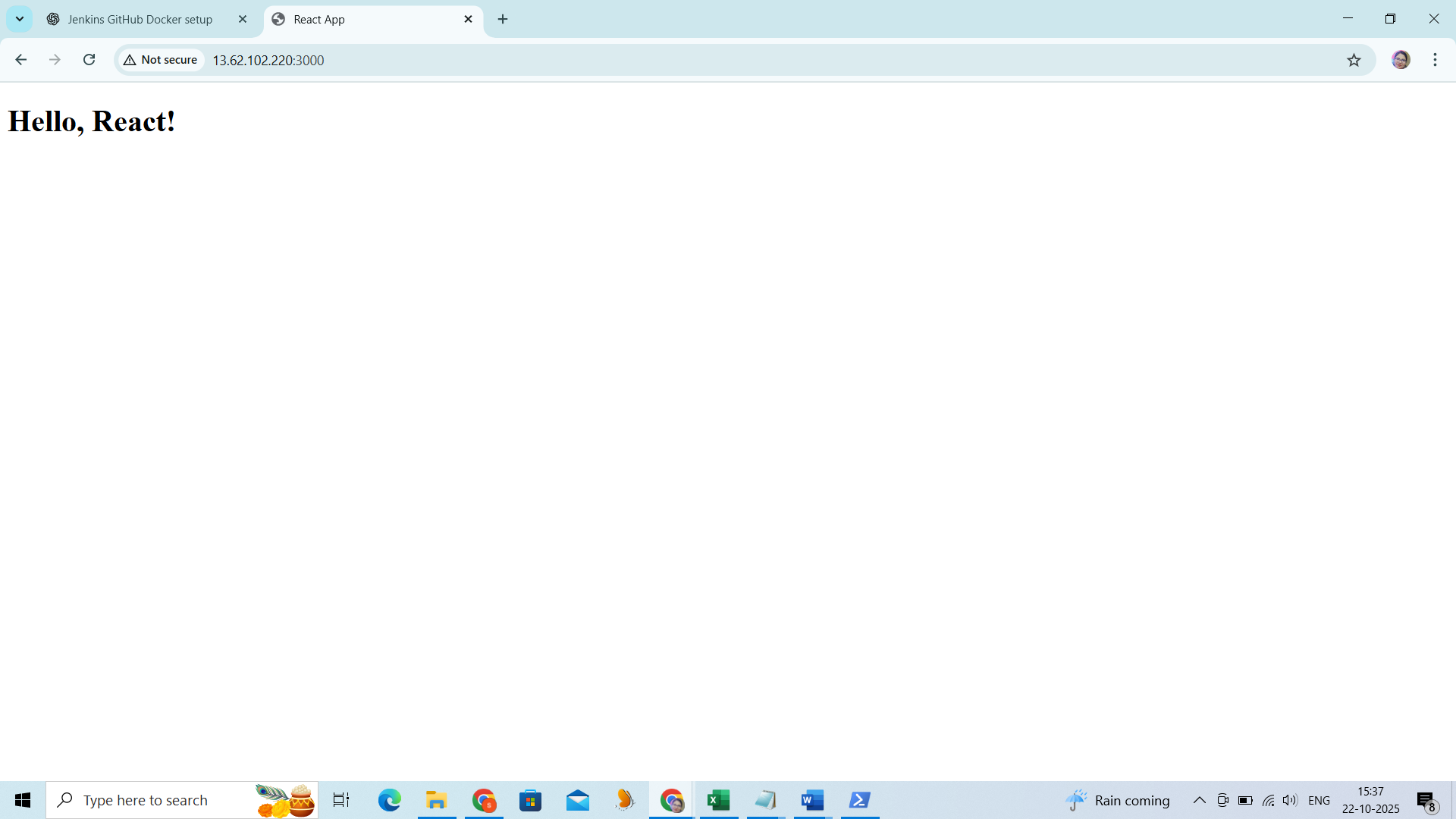
sudo ufw allow 3000/tcp

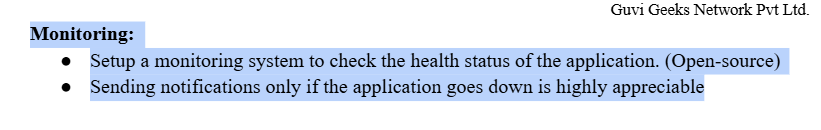
npm install -g serve

npm start

npm run build

npm start





**Set up Prometheus for Health Monitoring**

**Prometheus** is an open-source monitoring system that collects metrics.

wget https://github.com/prometheus/prometheus/releases/download/v2.51.2/prometheus-2.51.2.linux-amd64.tar.gz

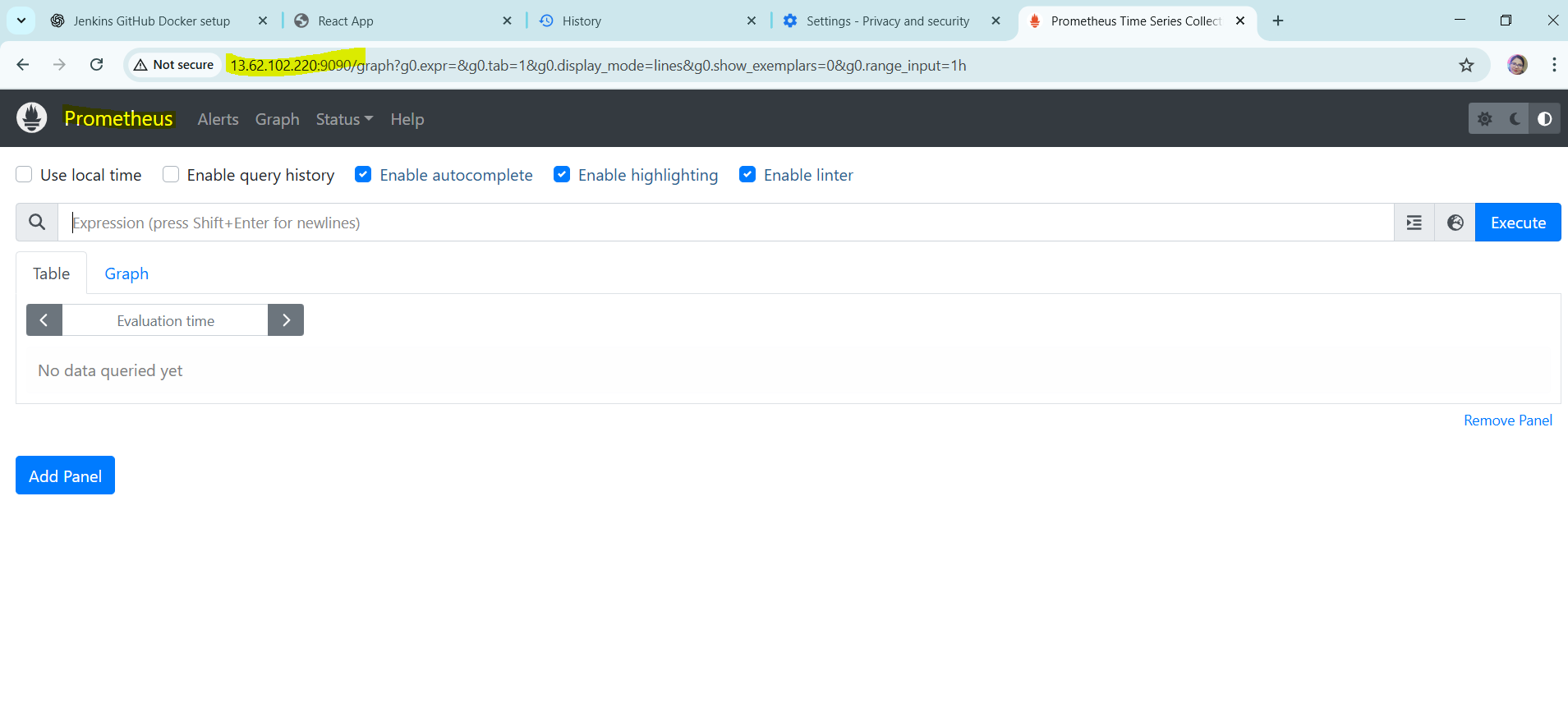
tar xvf prometheus-2.51.2.linux-amd64.tar.gz

cd prometheus-2.51.2.linux-amd64

./prometheus --config.file=prometheus.yml

Prometheus runs on:

<http://13.62.102.220:9090/graph?g0.expr=&g0.tab=1&g0.display_mode=lines&g0.show_exemplars=0&g0.range_input=1h>



**Installation of Grafana:**

<http://13.62.102.220:3000/login>

223 sudo apt-get install -y software-properties-common

224 sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"

225 wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -

226 sudo apt-get update

227 sudo apt-get install grafana

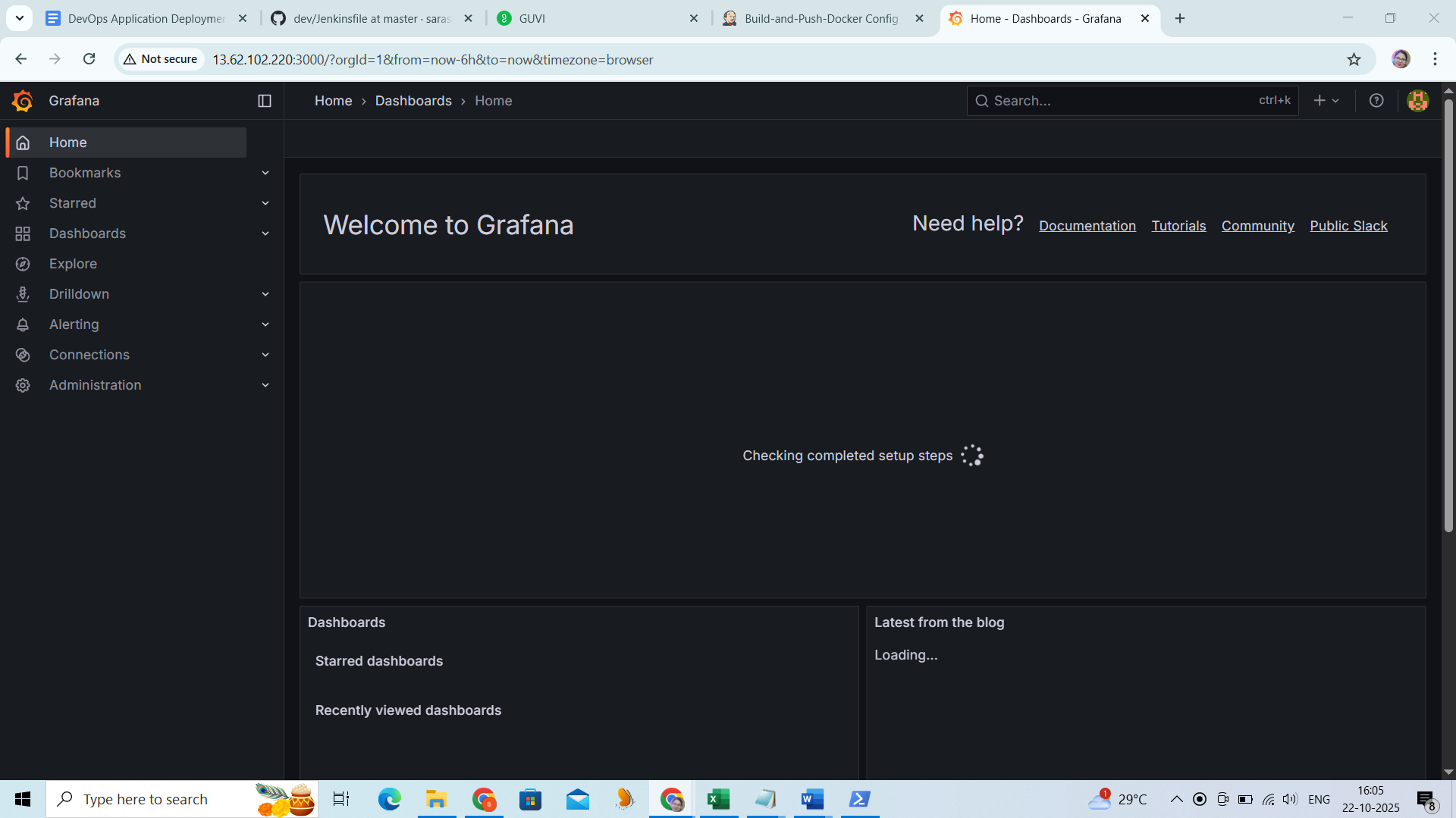
228 sudo systemctl start grafana-server

229 sudo systemctl enable grafana-server

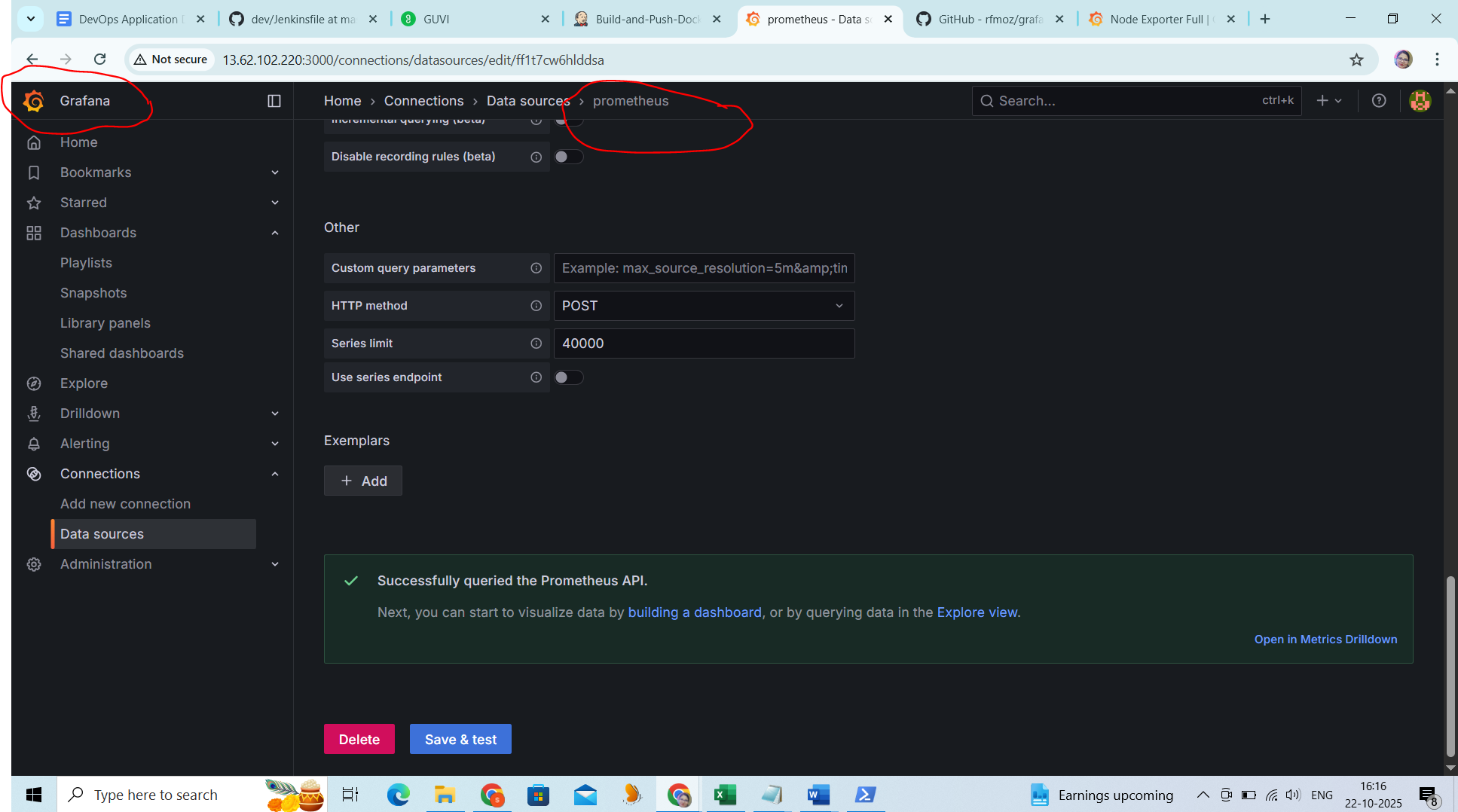


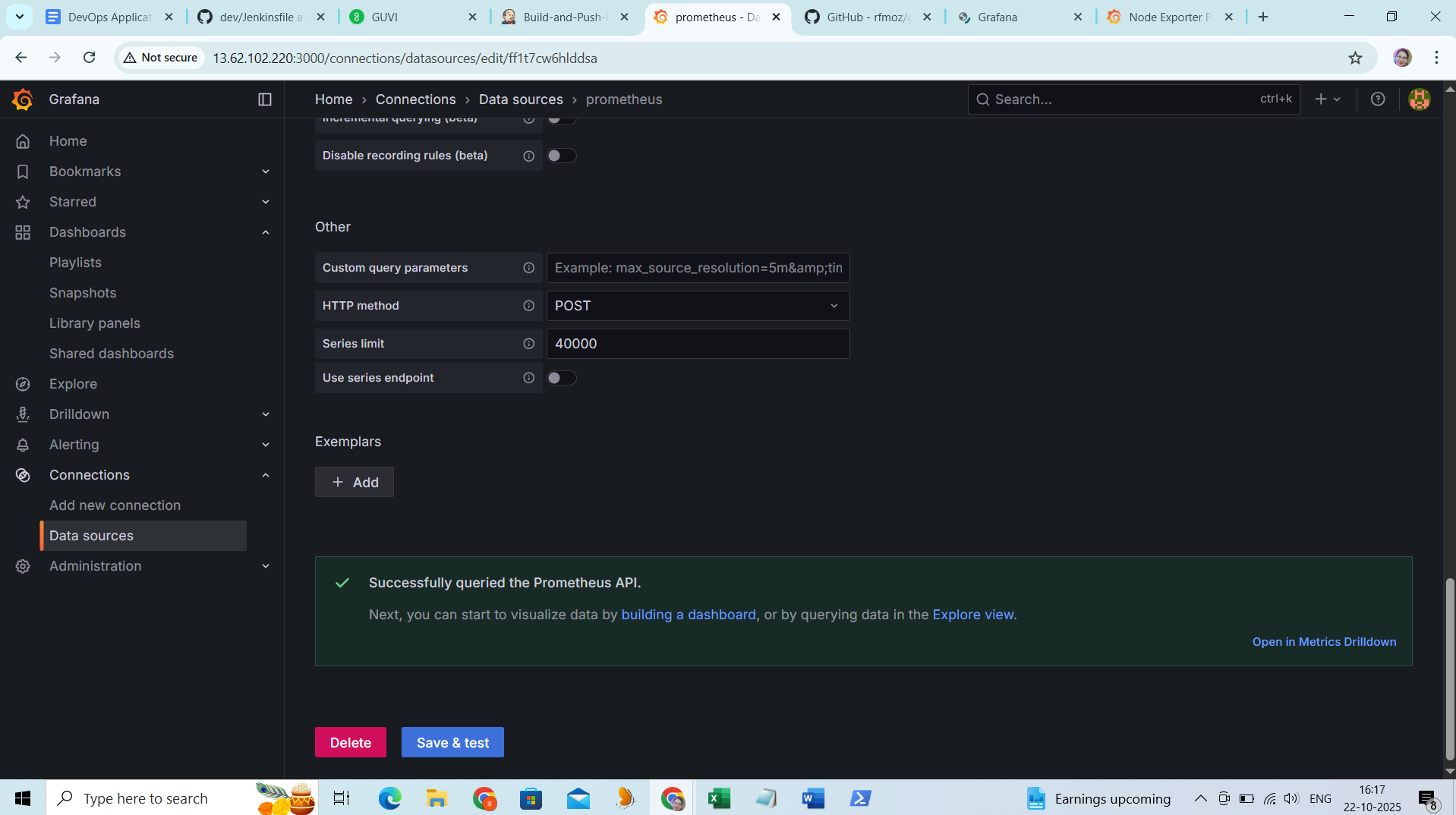
Username – admin

Password-admin



Attach the Prometheus with the Grafana:





Application monitor is running on 8080 port:

