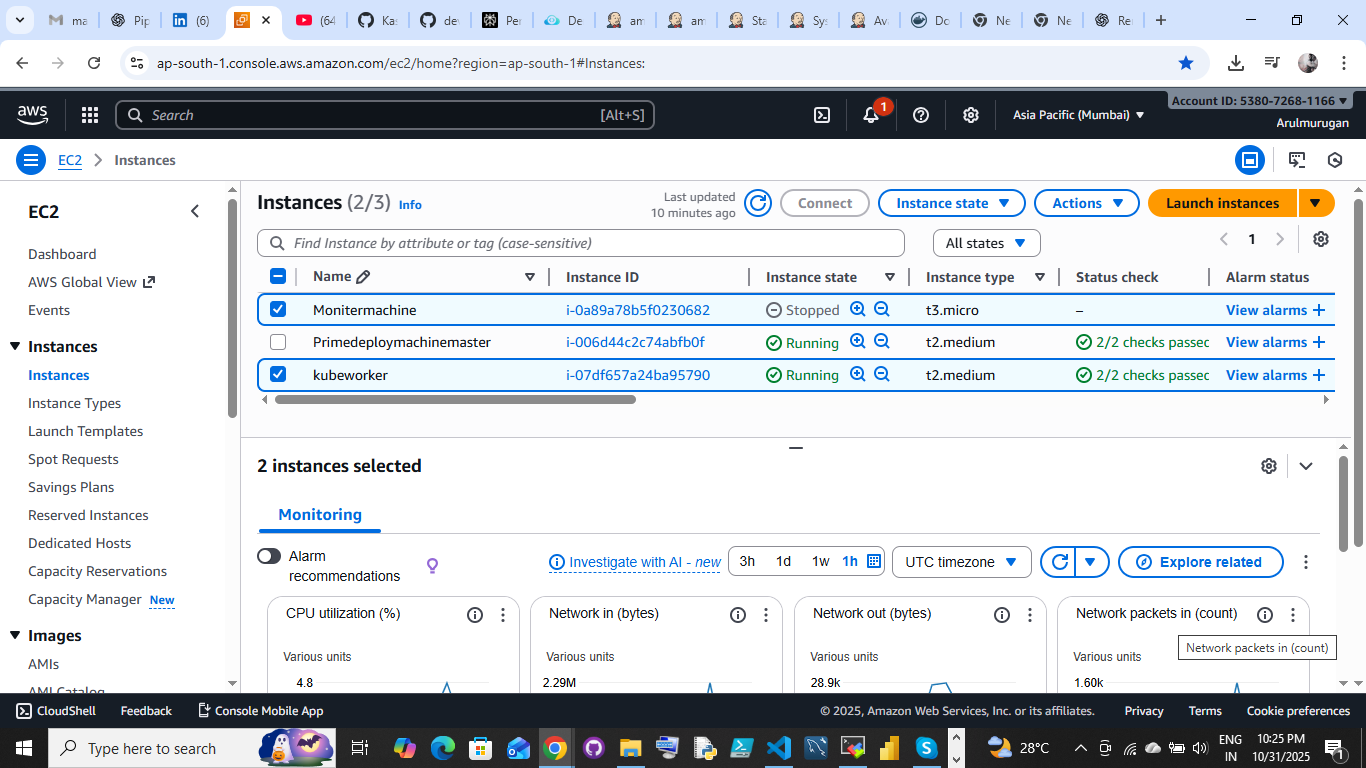
STEPS:

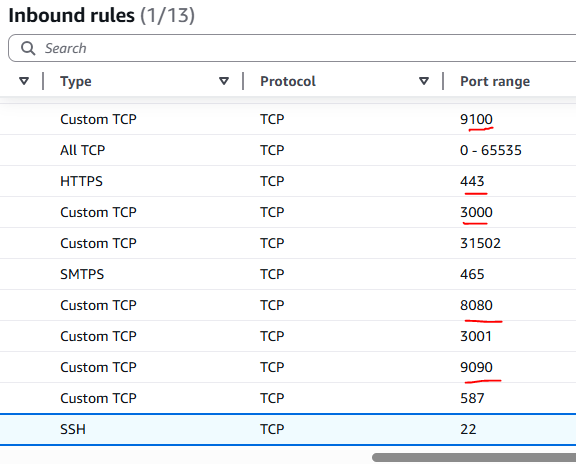
DEVOPS PROJC TS AMAZON PRIME CLONE

**Step1: jenkins Server In AWS Cloud (**EC2 MACHINE)

**CREATE THE EC2 MACHNE AND LOGIN USING MOBEX TOOLS**



**Port allows:**



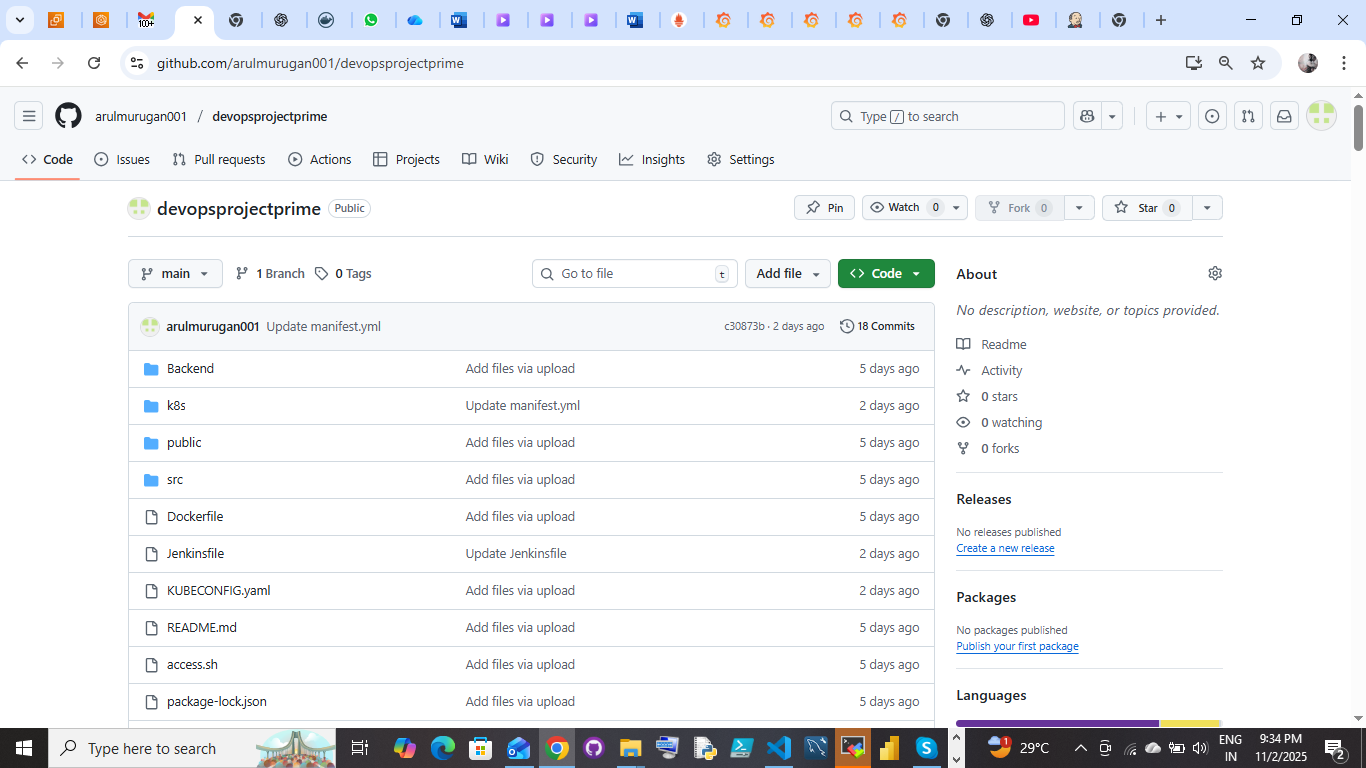
**Step2:**

**INSTALLATIONS AND CONFIGURATION:**

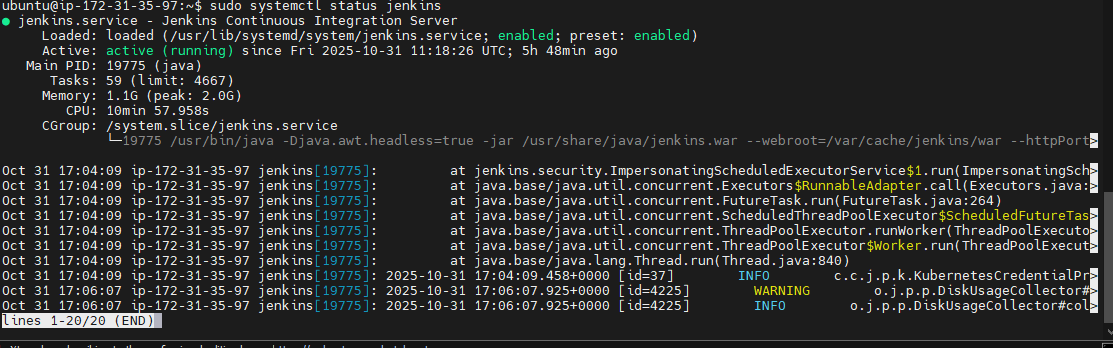
**1.GIT:**



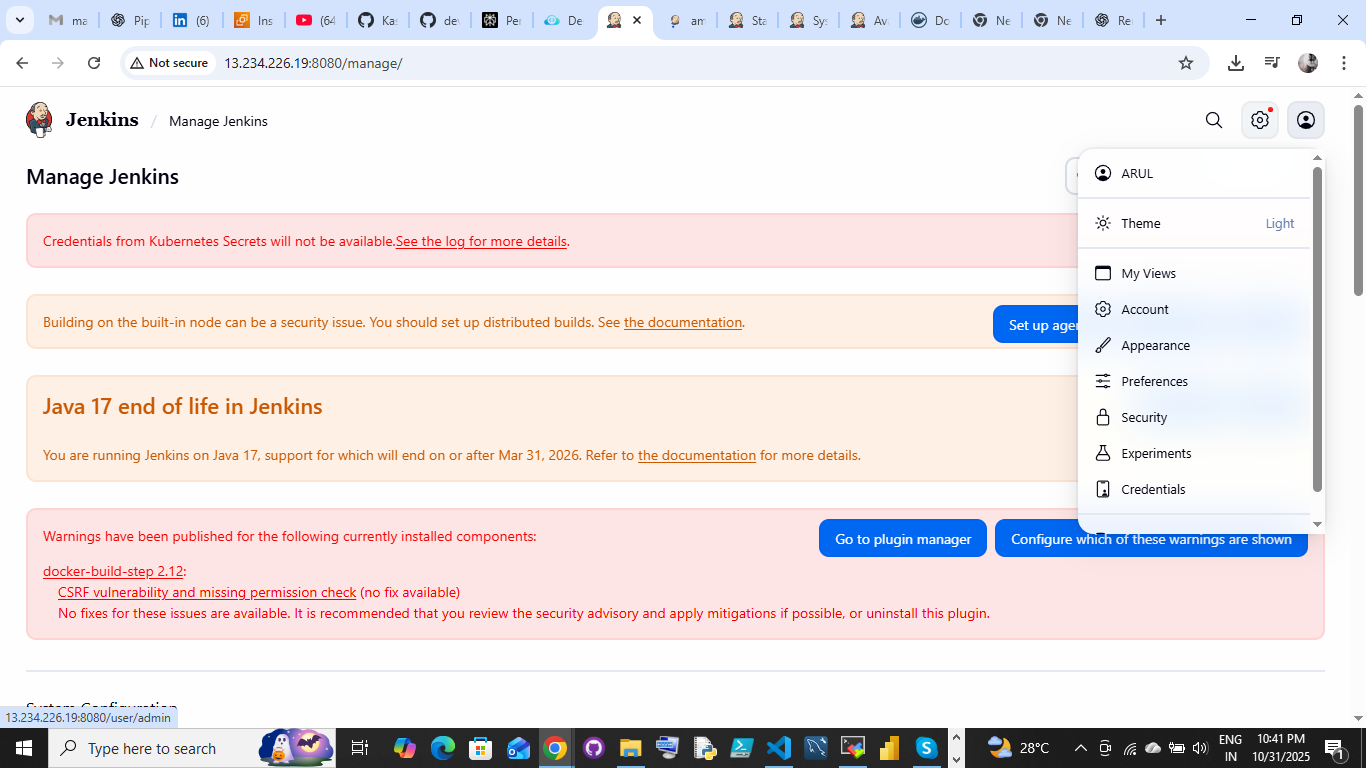
**1.1.GITHub account:**



**2.JENKINS**:



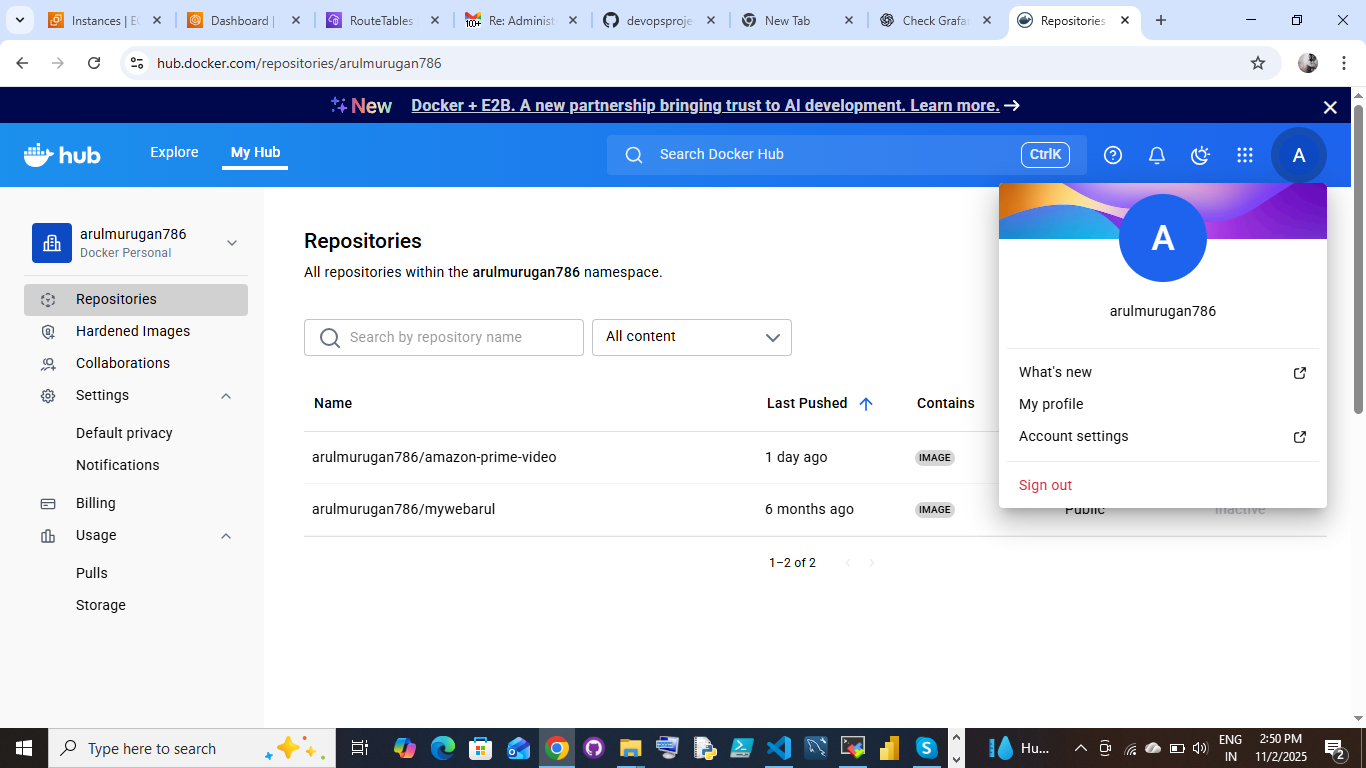
* After installation login and configure the Jenkins
* verify the Jenkins: <http://puplicip:8080>



**3.DOCKER:**

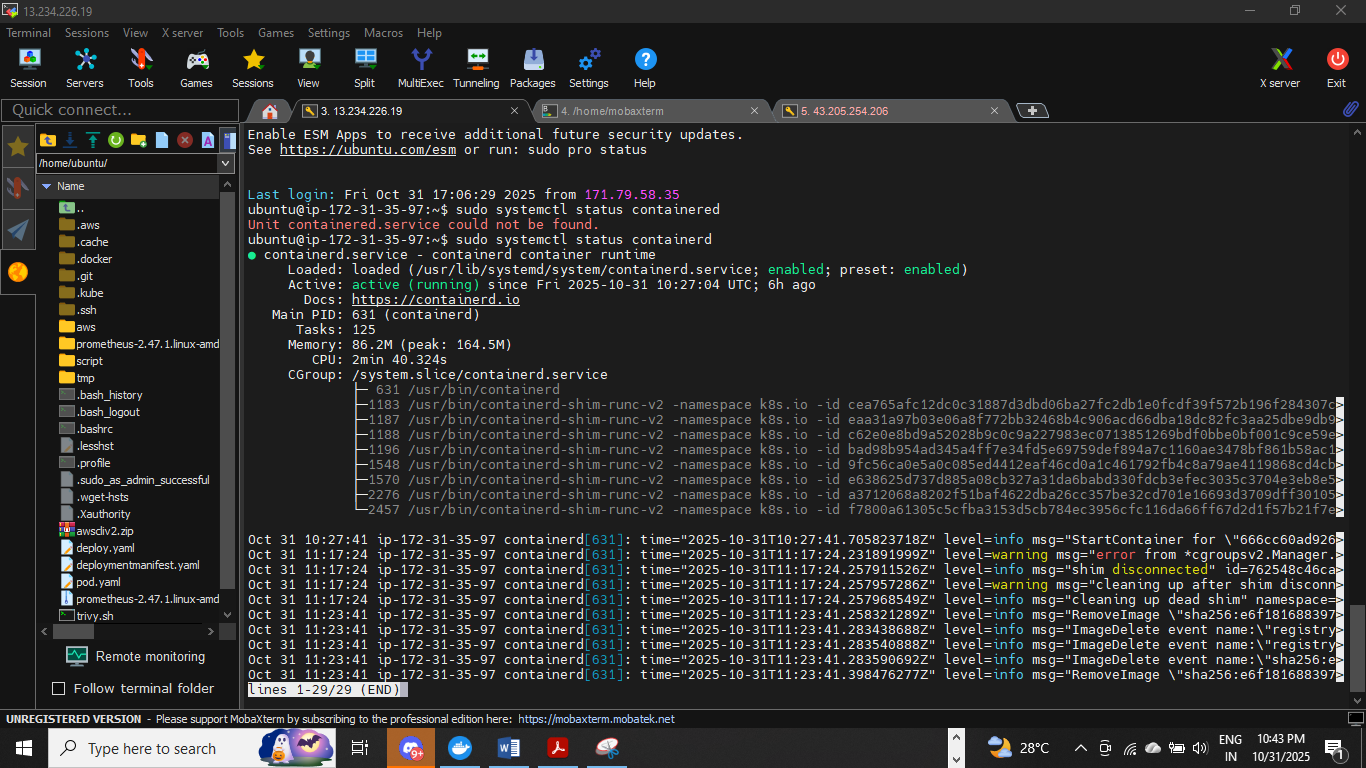


* After installation we have to create the Docker hub account
* Use to pull and push the images & container



4.KUBERNETS:

* Installation done at both machine worker node & Master node



**Configuration:** Connectivity create the master node and worker node - Using kubeadmn

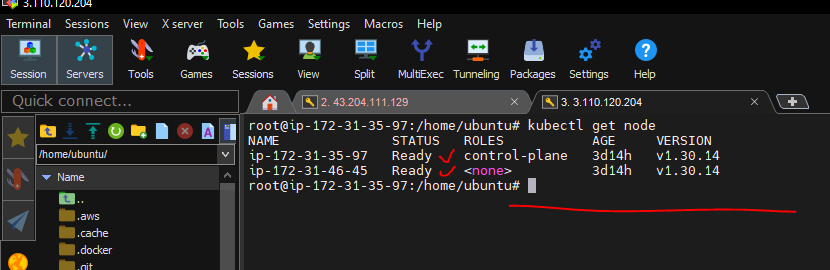
Kubernetes init(masternode):

sudo kubeadm init --apiserver-advertise-address=172.31.35.97(masteprivate) --pod-network-cidr=10.244.0.0/16

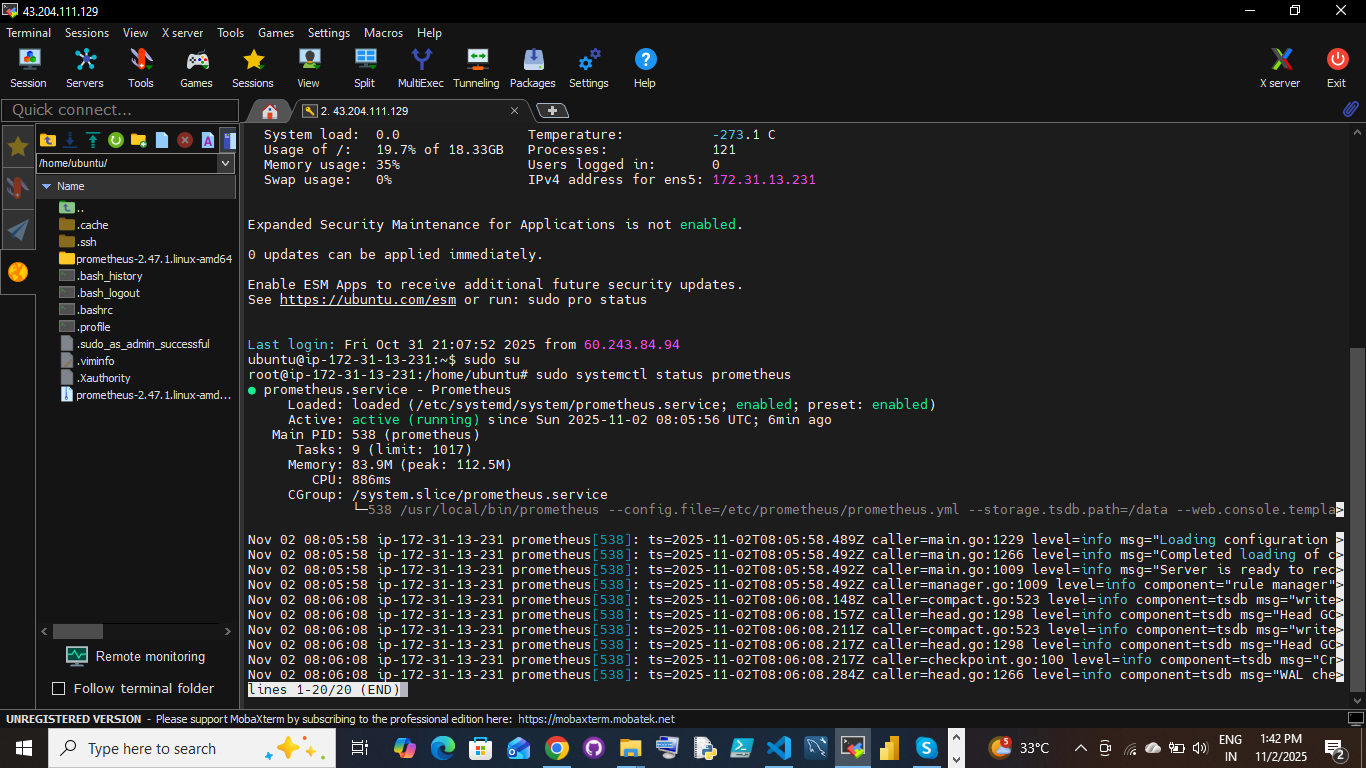
kubernetes join(workernode):

kubeadm join 172.31.35.97:6443 --token nqqwoj.szlz2g3sr357ho9u \ --discovery-token-ca-cert-hash sha256:f7d8f489e35ae59bd8c865a16b1da6edb6c89ceb578a60e65a8b8b32f42fd39b

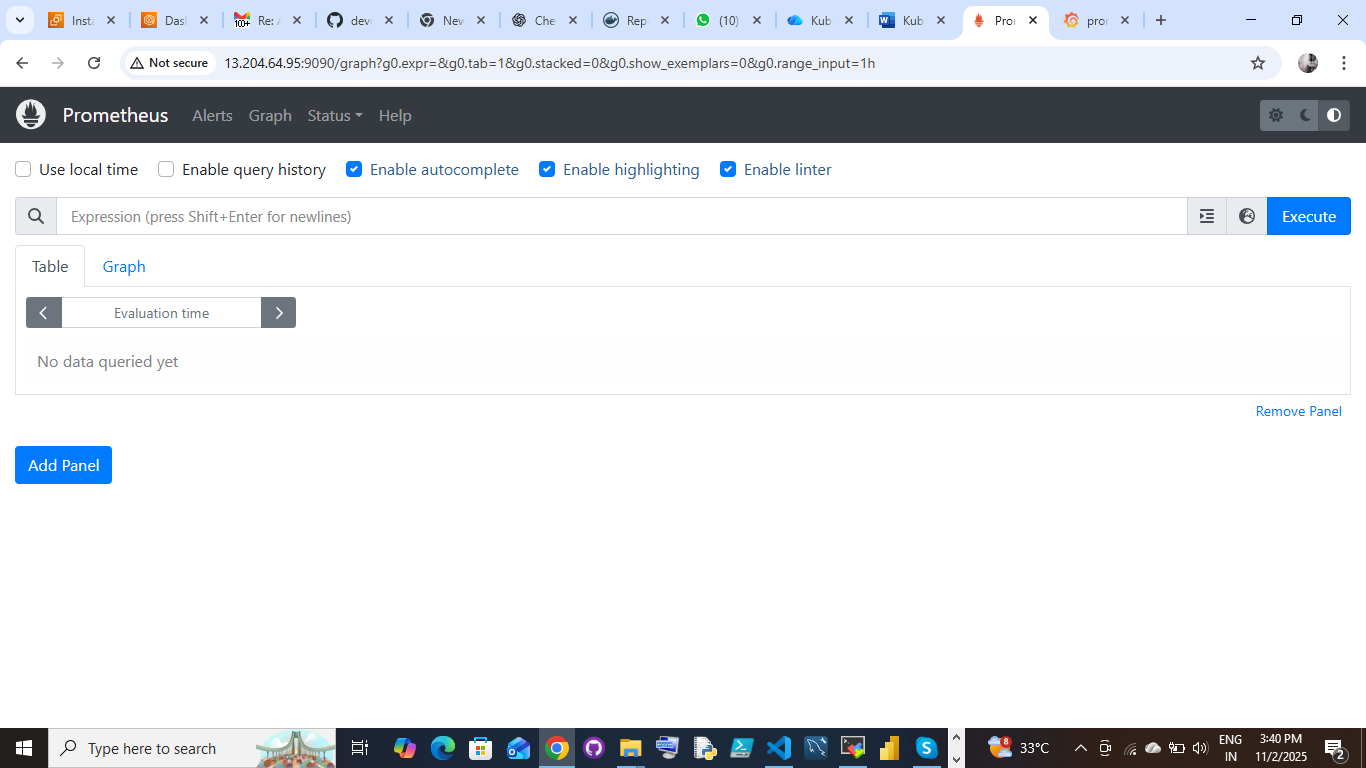
use this token to connect the machines-----verification:



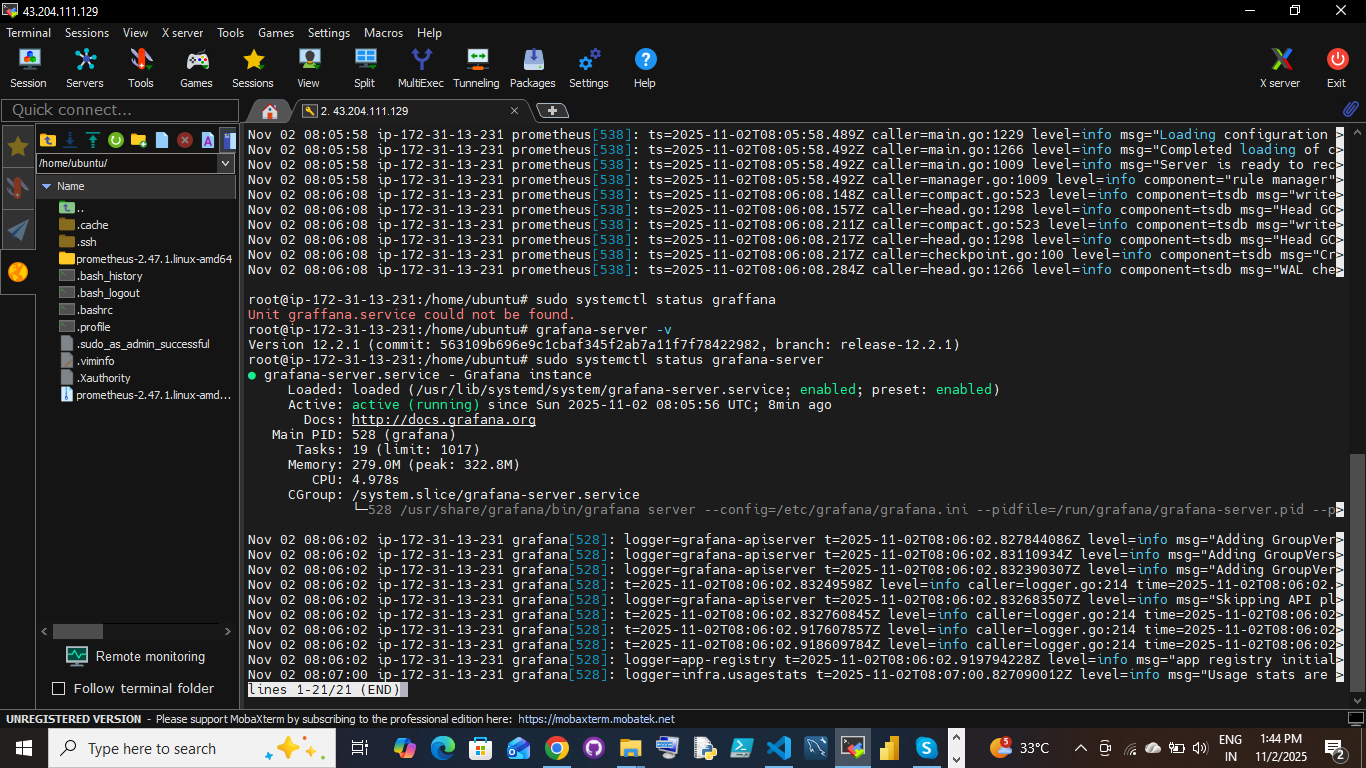
**5.PROMOTHUES:**



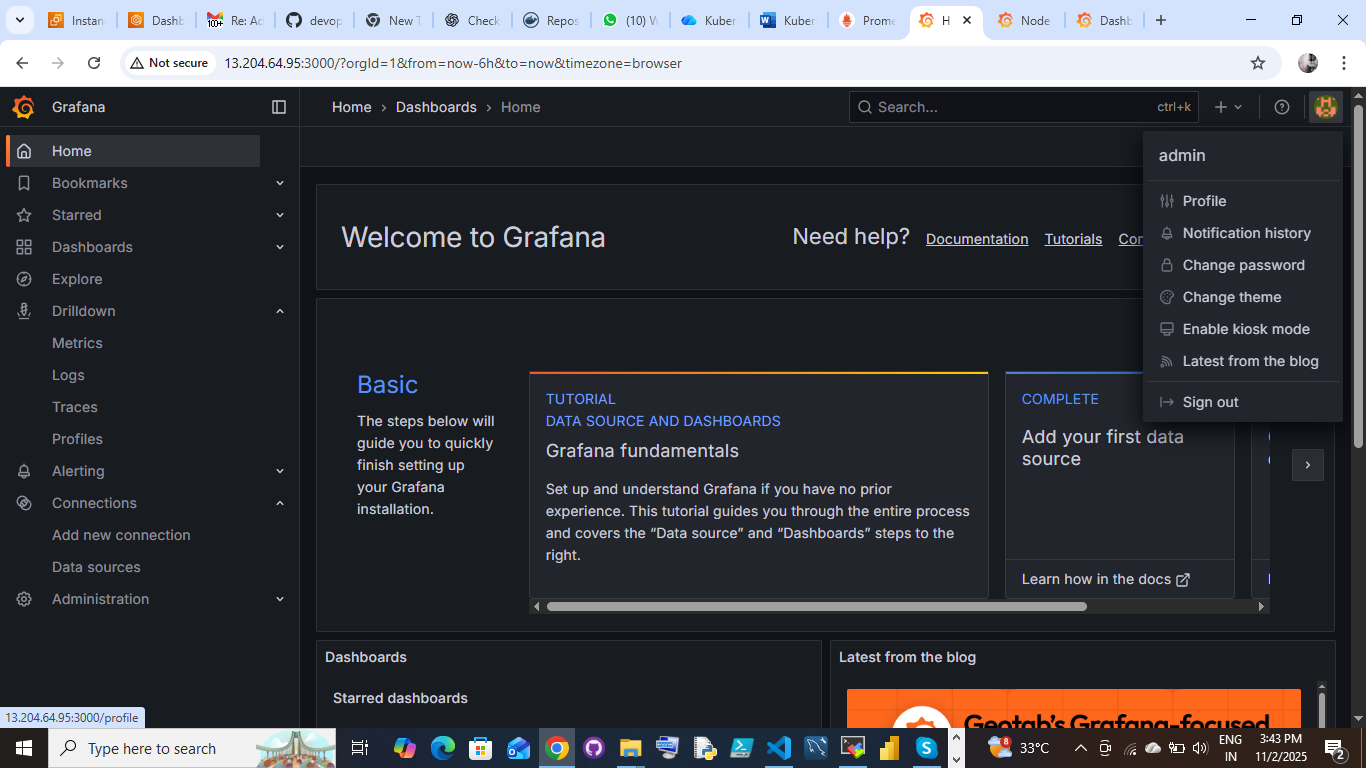
Access http://<your-server-ip>:9090



6.GRAFFANA:



Access: http://<your-server-ip>:3000



Before job creation we have to add the plugin and tools additionally add the credentials

Setup the Plugins form Jenkins Dashboard

Steps:

JENKINS PLUGINS:

Go to manage Jenkins select plugins check available plugins then find the

below mentioned plugins install and restart the Jenkins

1. Eclipse Temurin Installer

2. SonarQube Scanner

3. NodeJs Plugin

4 jdk-java

6.pipeline stage view

5. Docker plugin

6. Docker

7. Docker Commons

8. Docker Pipeline

9. Docker API

10. docker-build-step

11.Email Extension Template,

12.Kubernetes, Kubernetes CLI

13. Kubernetes Client API

14.Kubernetes Credentials

15. Kubernetes Credentials Provider

16.Config File Provider

17.Prometheus metrics

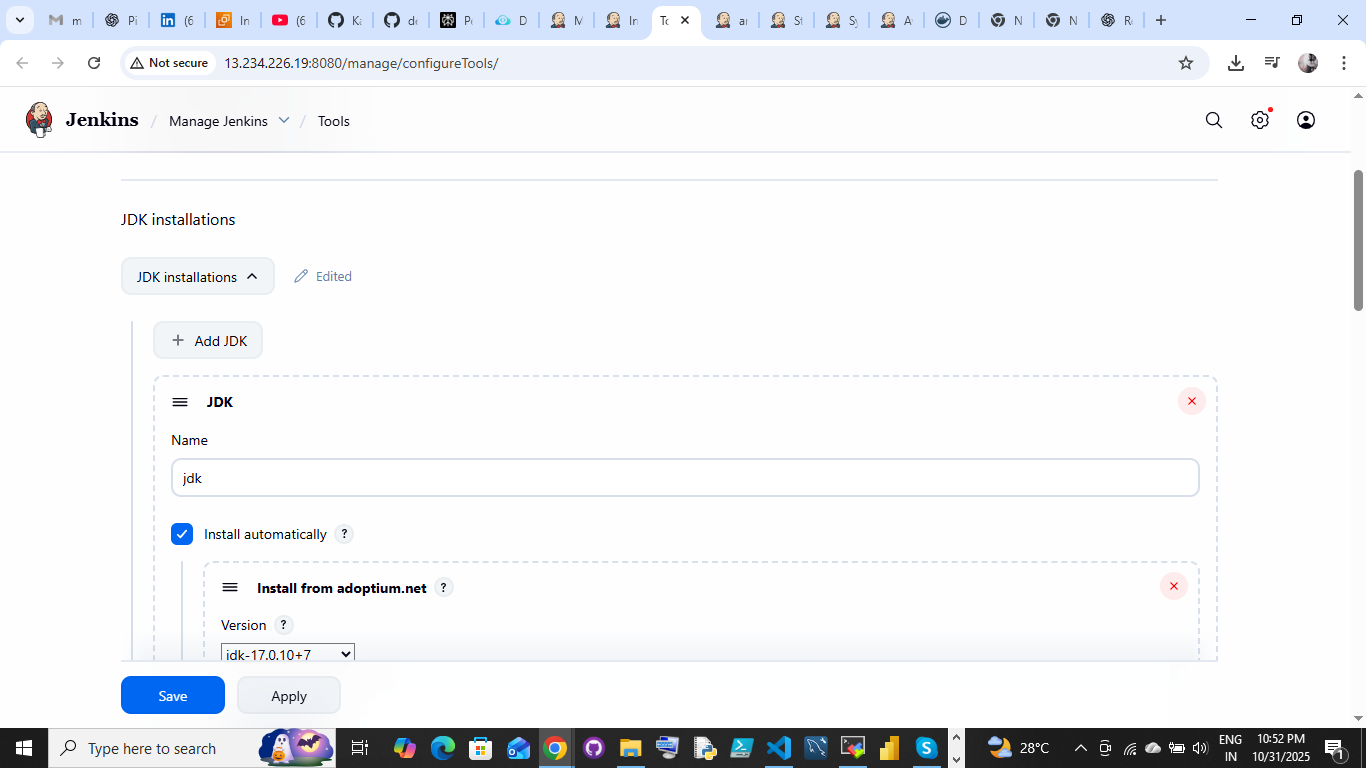
Setup the Tools form Jenkins Dashboard

Steps:

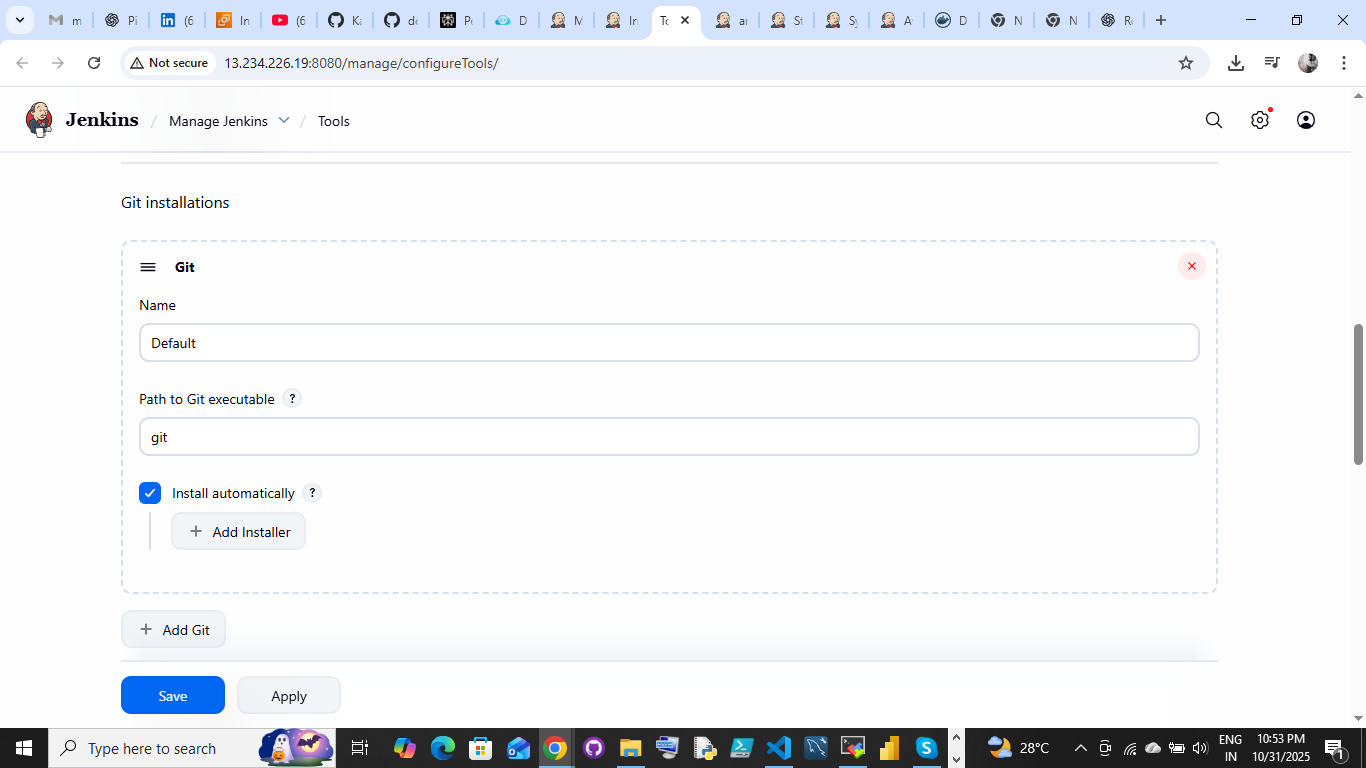
Tools installation in Jenkins:

Go to manage Jenkins select the tools then add the tools

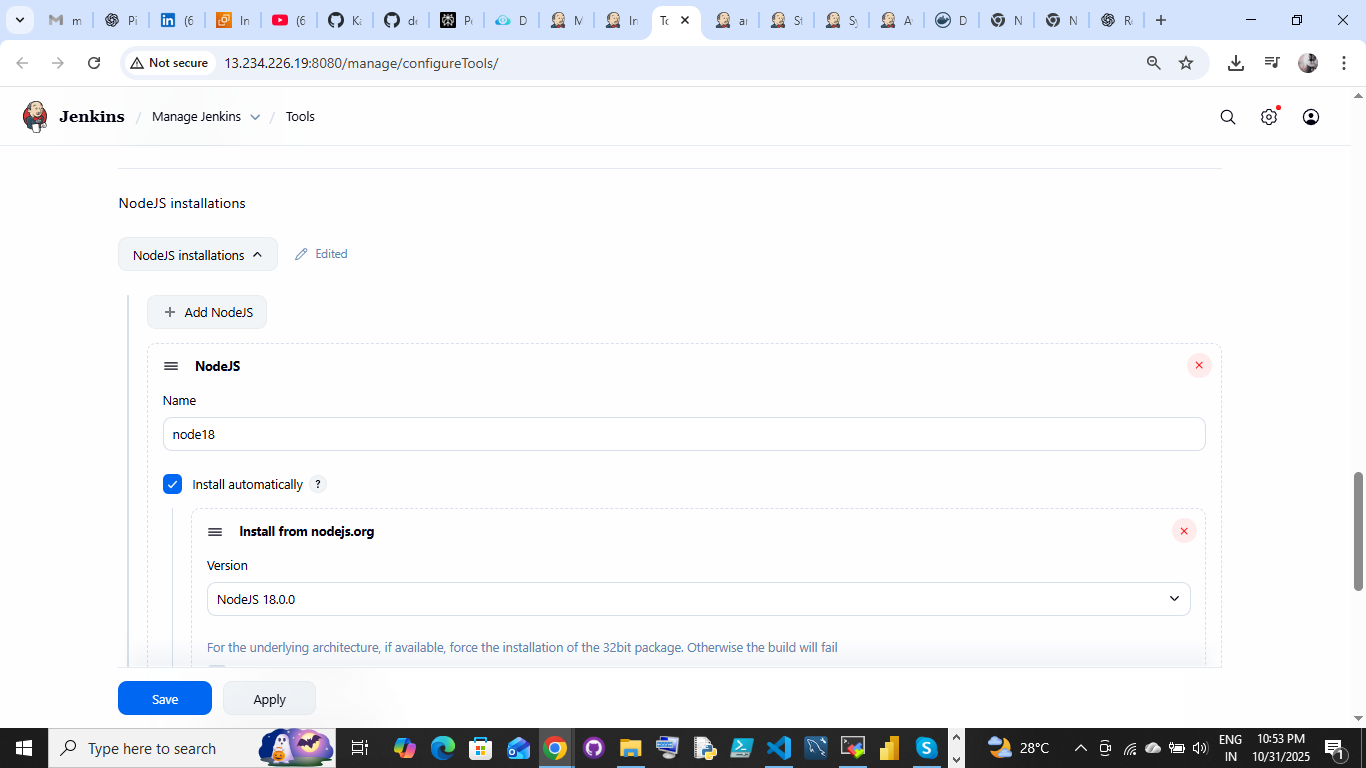
java Jdk:



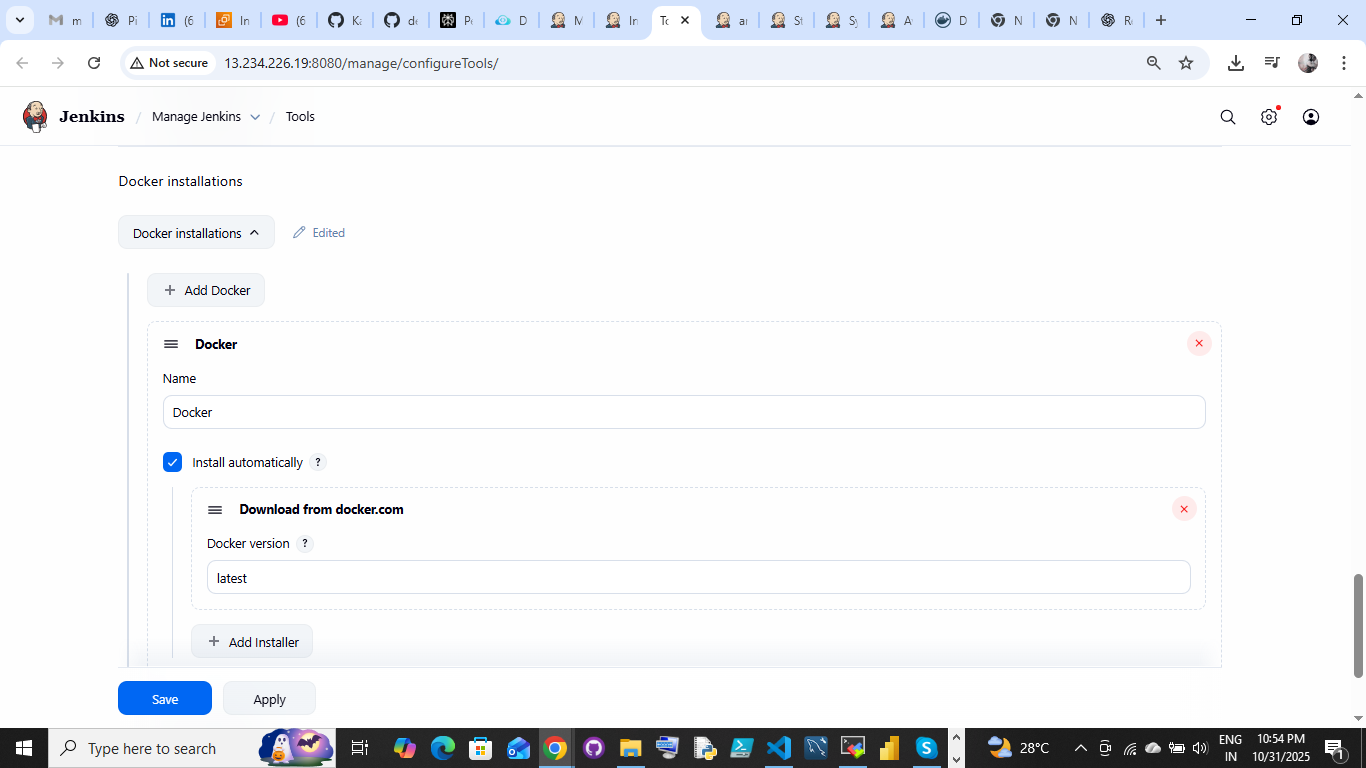
Git



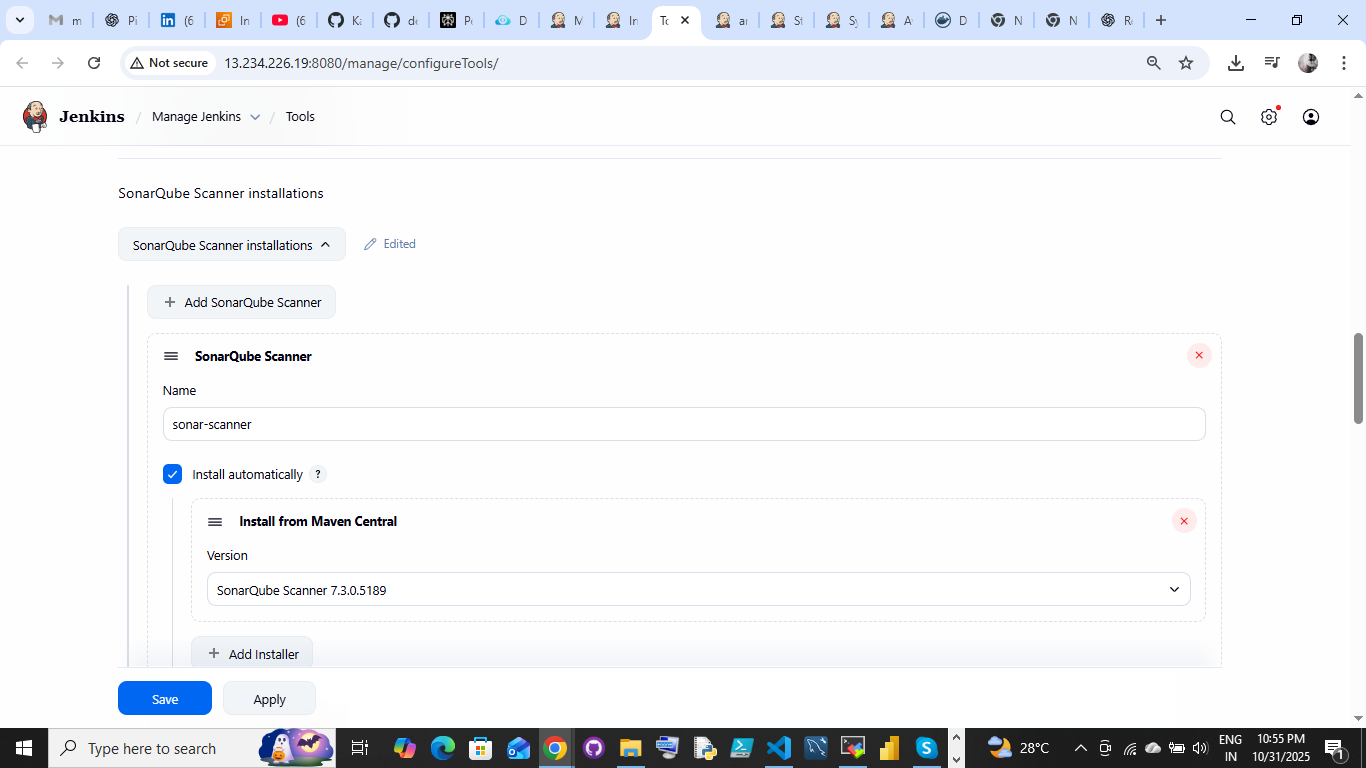
Node js:



Docker:



SonarQube tools installation:



Once complete

Setup the credential in Jenkins

Credential add into jenkins:

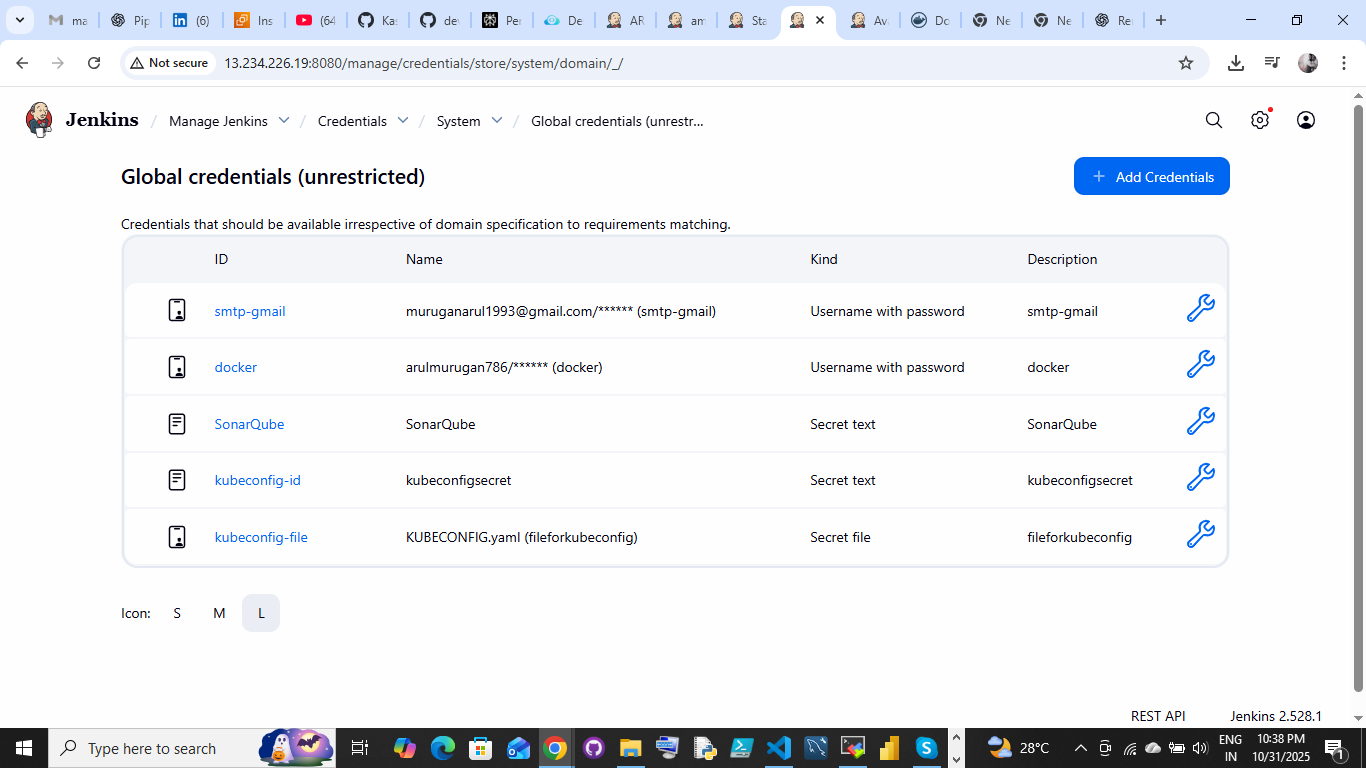
Go to manage Jenkins

Credentials

System

Global tool configuration then add the creaditials

Use to access the file thoose you are using tools like git, docker, etc



STEP4:

JOB CREATION:

Login Jenkins:

Step1: Go to new item

Step2: create a job name

Step3: choose pipeline

then,

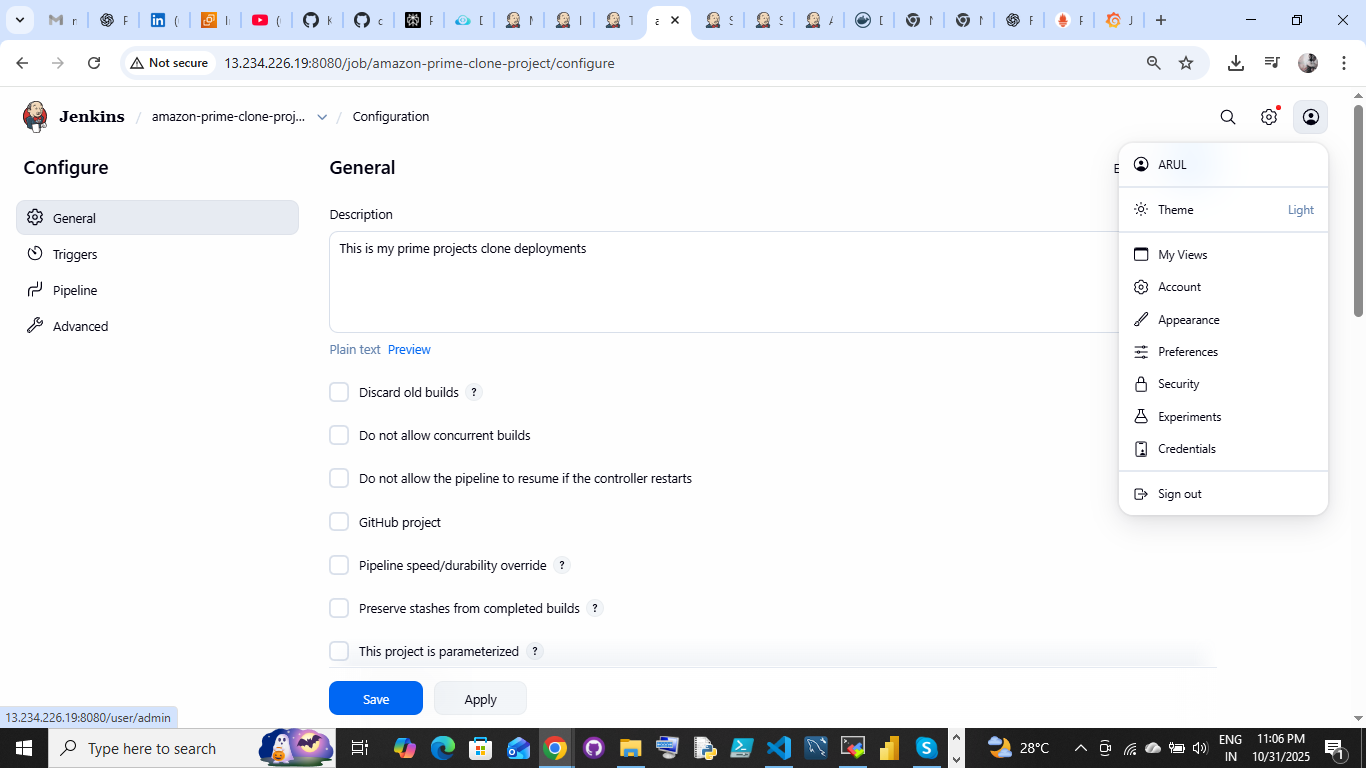
Add the pipeline script into the coding part

save and apply

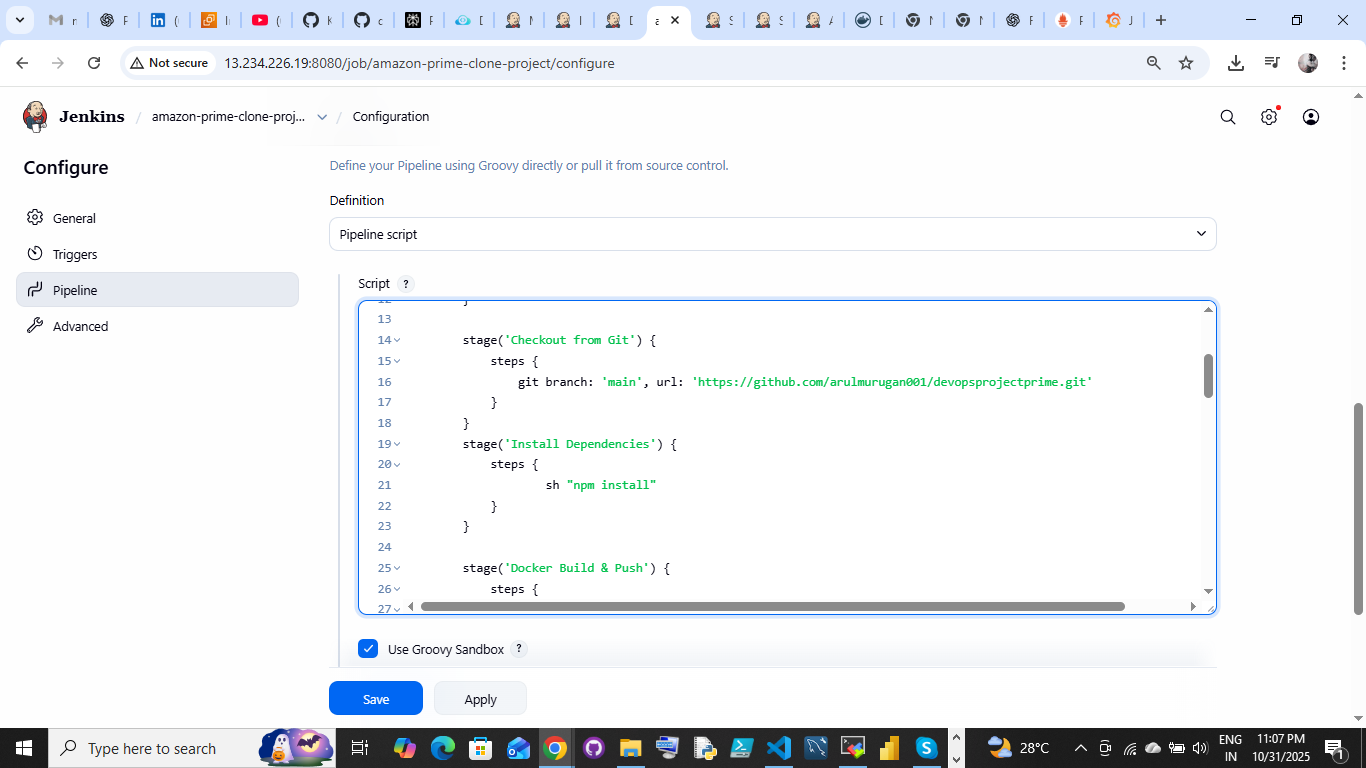
build the job

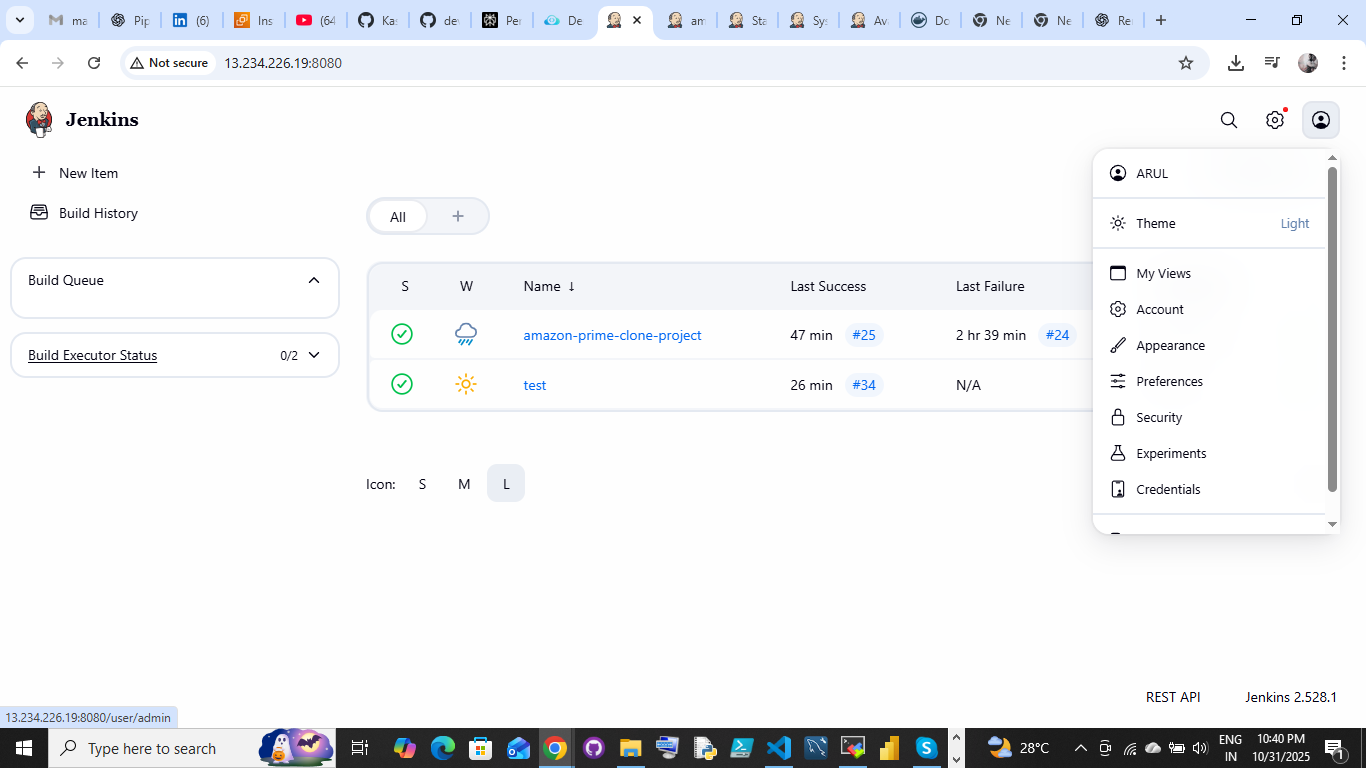
check the stage view everything its going correct or not once complete to verify the deployment

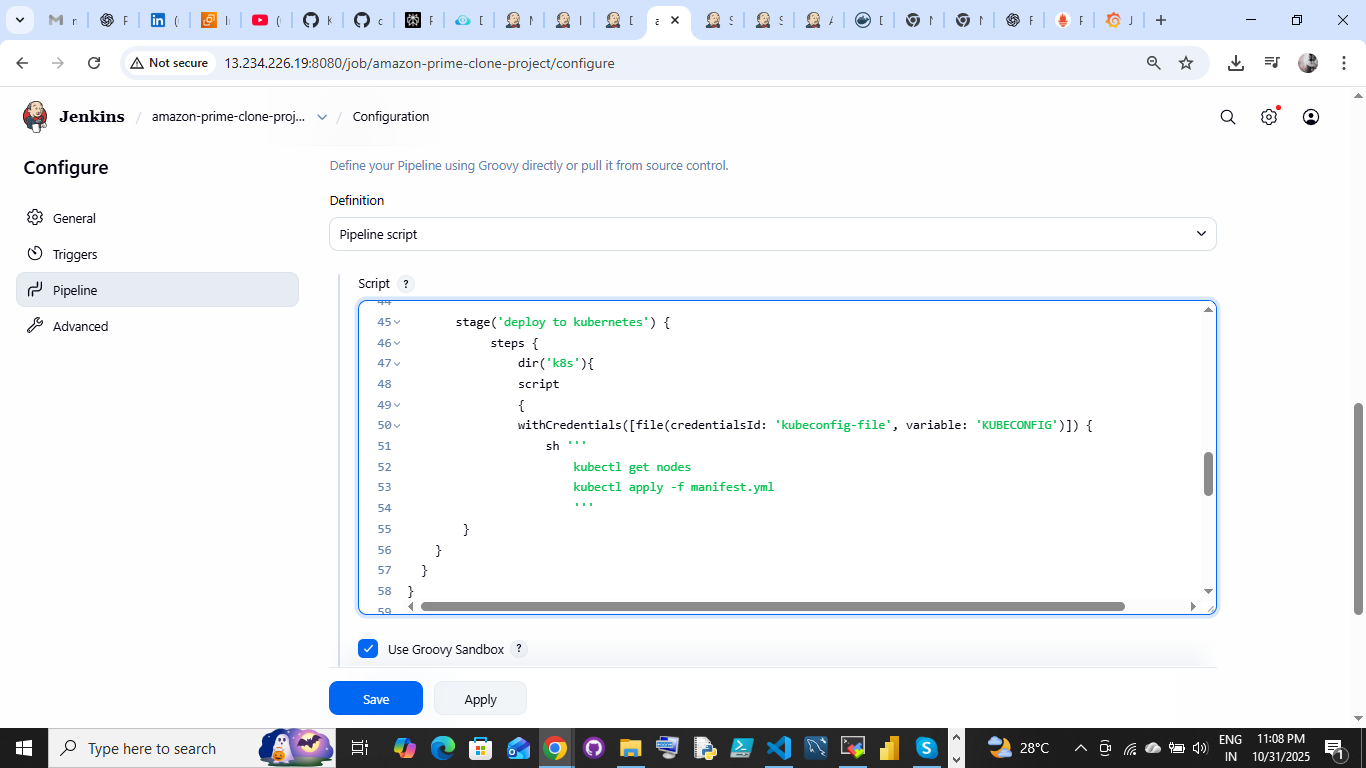
via http://localhost:portnumber



Create the pipeline script

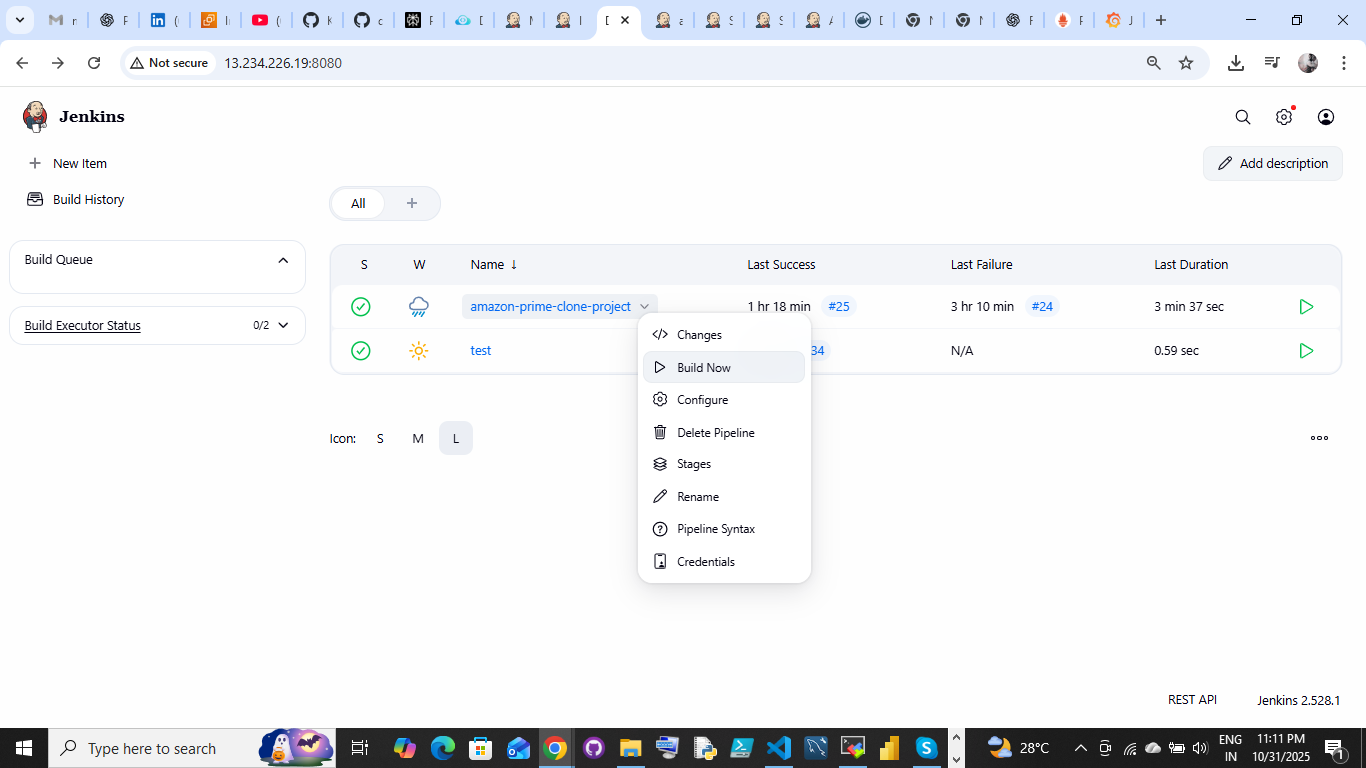




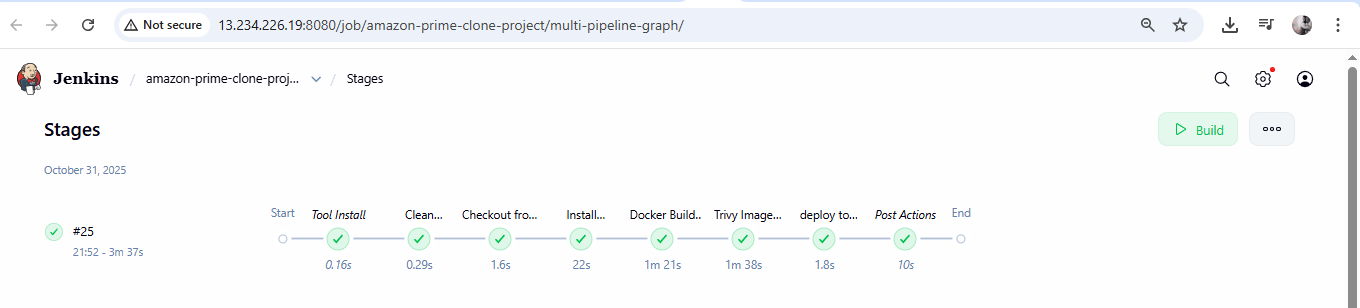


Apply and Save the file

Build the pipeline script:



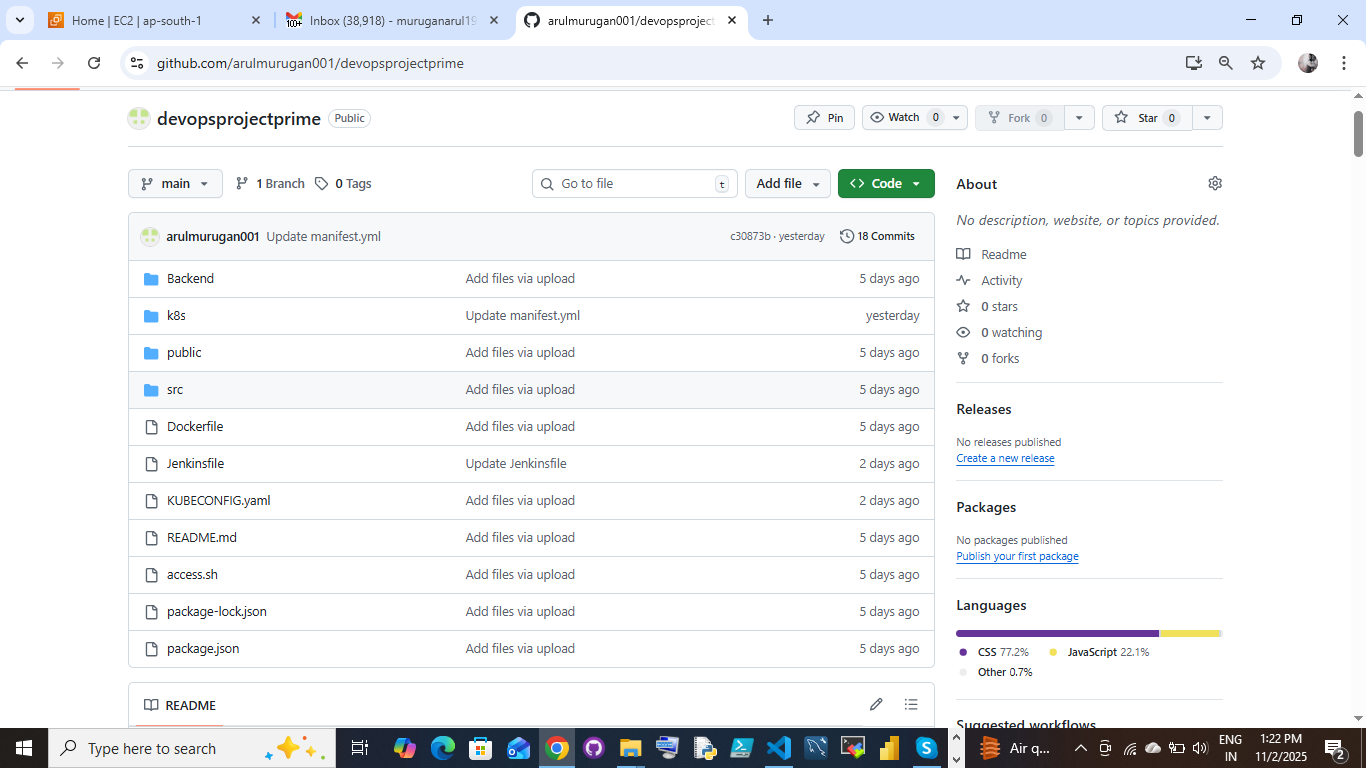
Stage view:



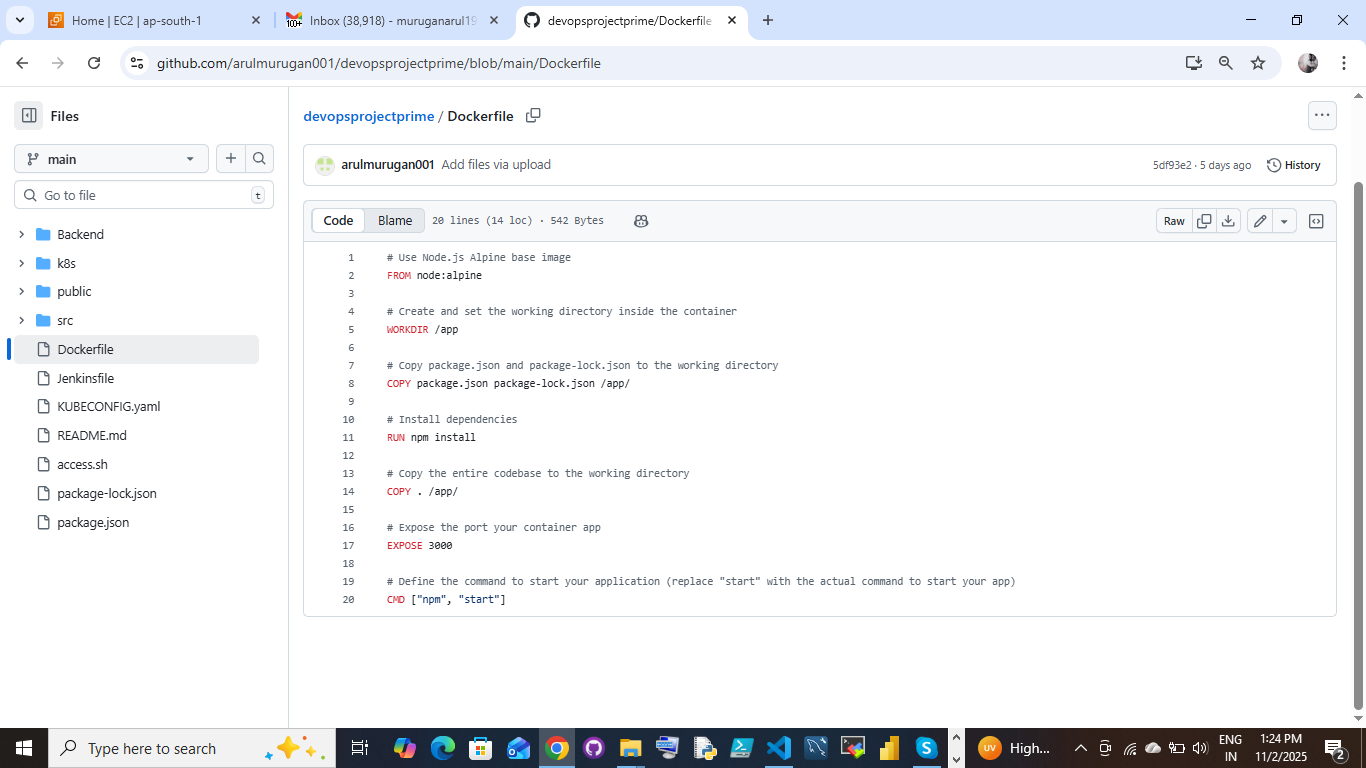
Before write a pipeline script you have verify

Git hub repository:

Folder structure and docker files

Folder structure: 

Docker file:



Pipeline Scripts without kubernetes:

Pipeline Script: deploy to containers:

pipeline{

agent any

tools{

jdk 'jdk'

nodejs 'node18'

}

stages {

stage('clean workspace'){

steps{

cleanWs()

}

}

stage('Checkout from Git'){

steps{

git branch: 'main', url: 'https://github.com/arulmurugan001/devopsprojectprime.git'

}

}

stage('Install Dependencies') {

steps {

sh "npm install"

}

}

stage("Docker Build & Push"){

steps{

script{

withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){

sh "docker build -t amazon-prime-video ."

sh "docker tag amazon-prime-video arulmurugan786/amazon-prime-video:latest "

sh "docker push arulmurugan786/amazon-prime-video:latest "

}

}

}

}

stage('App Deploy to Docker container'){

steps{

sh 'docker rm -f amazon-prime-video || true'

sh 'docker run -d --name amazon-prime-video -p 3001:3000 arulmurugan786/amazon-prime-video:latest'

}

}

}

post {

always {

script {

def buildStatus = currentBuild.currentResult

def buildUser = currentBuild.getBuildCauses('hudson.model.Cause$UserIdCause')[0]?.userId ?: 'Github User'

emailext (

subject: "Pipeline ${buildStatus}: ${env.JOB\_NAME} #${env.BUILD\_NUMBER}",

body: """

<p>This is a Jenkins starbucks CICD pipeline status.</p>

<p>Project: ${env.JOB\_NAME}</p>

<p>Build Number: ${env.BUILD\_NUMBER}</p>

<p>Build Status: ${buildStatus}</p>

<p>Started by: ${buildUser}</p>

<p>Build URL: <a href="${env.BUILD\_URL}">${env.BUILD\_URL}</a></p>

""",

to: 'muruganarul1993@gmail.com',

from: 'muruganarul1993@gmail.com',

replyTo: 'muruganarul1993@gmail.com',

mimeType: 'text/html',

)

}

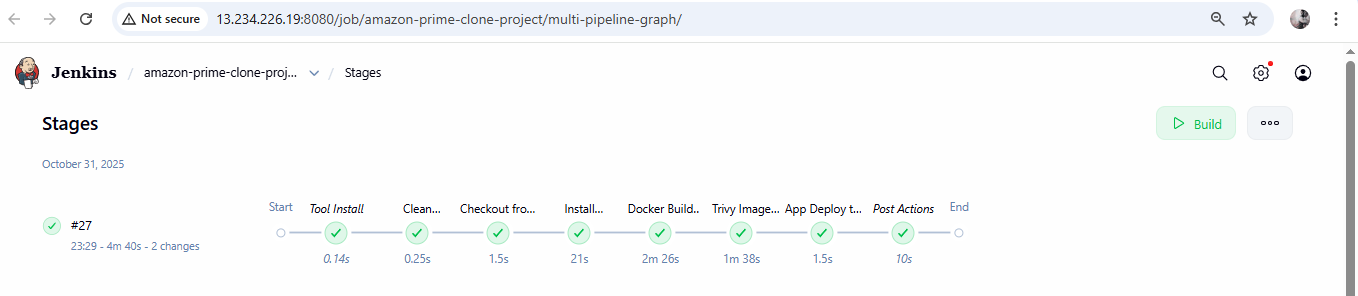
}

}

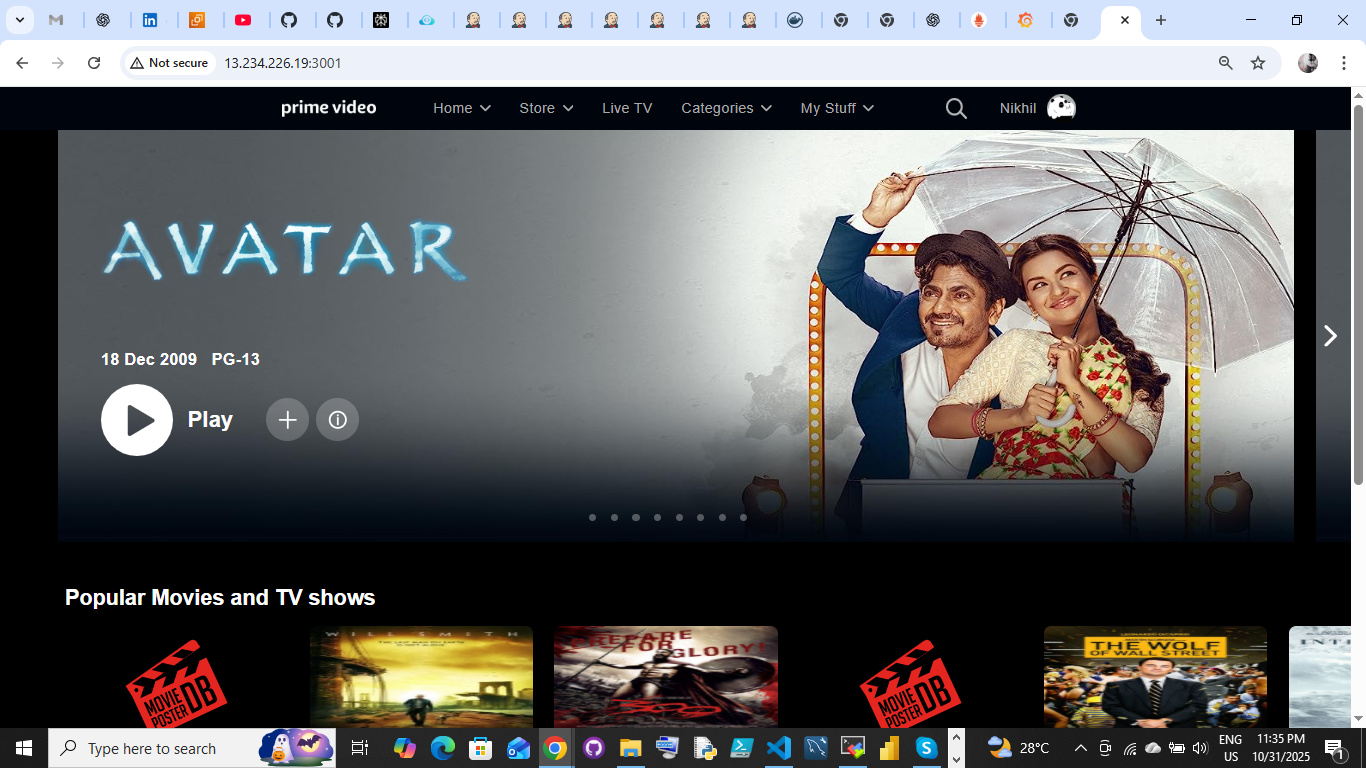
}

**Output:**

**Console stage view:**

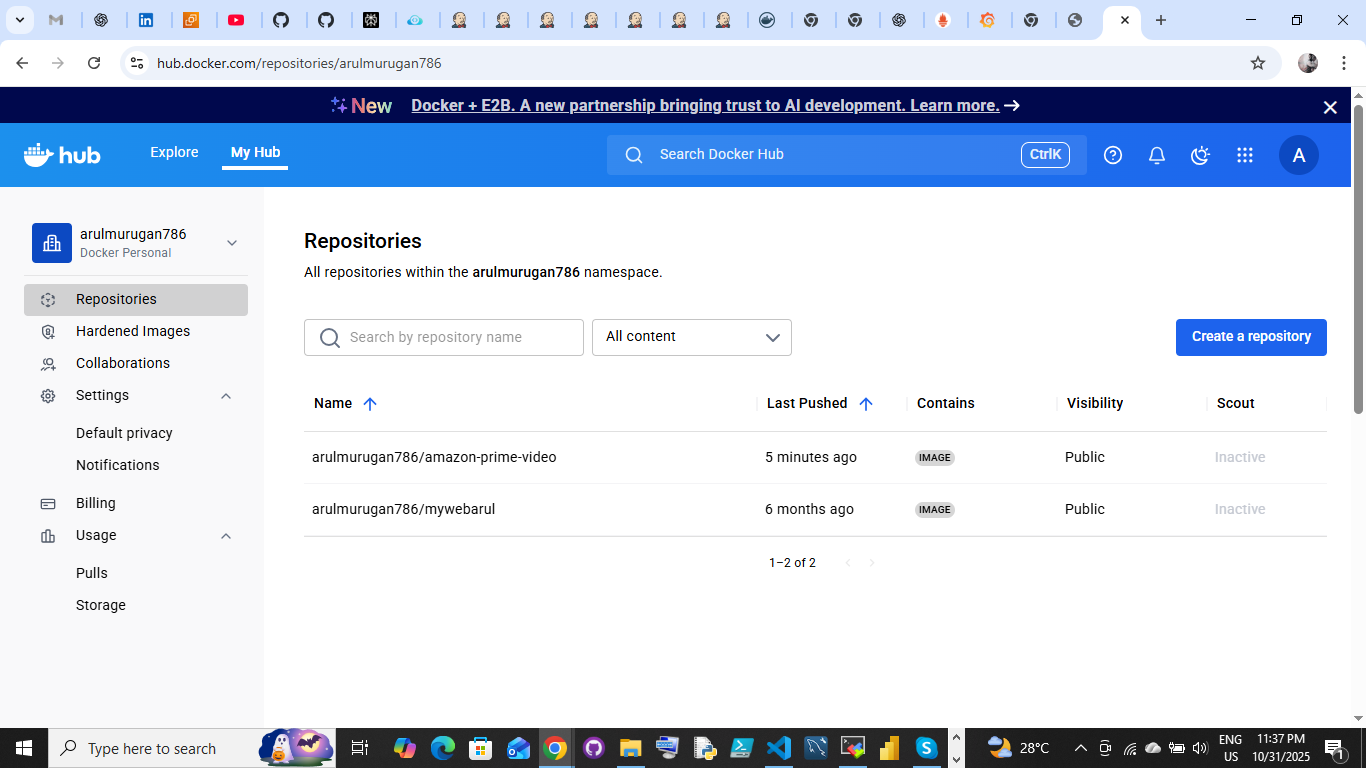


**Browser View:** http://publicip:3000



**Docker hub view:**

Container pushed into docker hub:



Pipeline Scripts with kubernetes cluster and nodeport svc:

Pipeline Script: deploy to kubernetes:

pipeline {

agent any

tools {

jdk 'jdk'

nodejs 'node18'

}

stages {

stage('Clean Workspace') {

steps {

cleanWs()

}

}

stage('Checkout from Git') {

steps {

git branch: 'main', url: 'https://github.com/arulmurugan001/devopsprojectprime.git'

}

}

stage('Install Dependencies') {

steps {

sh "npm install"

}

}

stage('Docker Build & Push') {

steps {

script {

withDockerRegistry(credentialsId: 'docker', toolName: 'docker') {

sh '''

docker build -t amazon-prime-video .

docker tag amazon-prime-video arulmurugan786/amazon-prime-video:latest

docker push arulmurugan786/amazon-prime-video:latest

'''

}

}

}

}

stage('Trivy Image Scan') {

steps {

sh "trivy image arulmurugan786/amazon-prime-video:latest > trivyimage.txt"

}

}

stage('deploy to kubernetes') {

steps {

dir('k8s'){

script

{

withCredentials([file(credentialsId: 'kubeconfig-file', variable: 'KUBECONFIG')]) {

sh '''

kubectl get nodes

kubectl apply -f manifest.yml

'''

}

}

}

}

}

}

post {

always {

script {

def buildStatus = currentBuild.currentResult

def buildUser = currentBuild.getBuildCauses('hudson.model.Cause$UserIdCause')[0]?.userId ?: 'GitHub Trigger'

emailext(

subject: "Pipeline ${buildStatus}: ${env.JOB\_NAME} #${env.BUILD\_NUMBER}",

body: """

<p><b>Jenkins CI/CD Pipeline Report</b></p>

<p>Project: ${env.JOB\_NAME}</p>

<p>Build Number: ${env.BUILD\_NUMBER}</p>

<p>Status: <b>${buildStatus}</b></p>

<p>Triggered By: ${buildUser}</p>

<p><a href="${env.BUILD\_URL}">View Build Details</a></p>

""",

to: 'muruganarul1993@gmail.com',

from: 'muruganarul1993@gmail.com',

replyTo: 'muruganarul1993@gmail.com',

mimeType: 'text/html',

attachmentsPattern: 'trivy\*.txt'

)

}

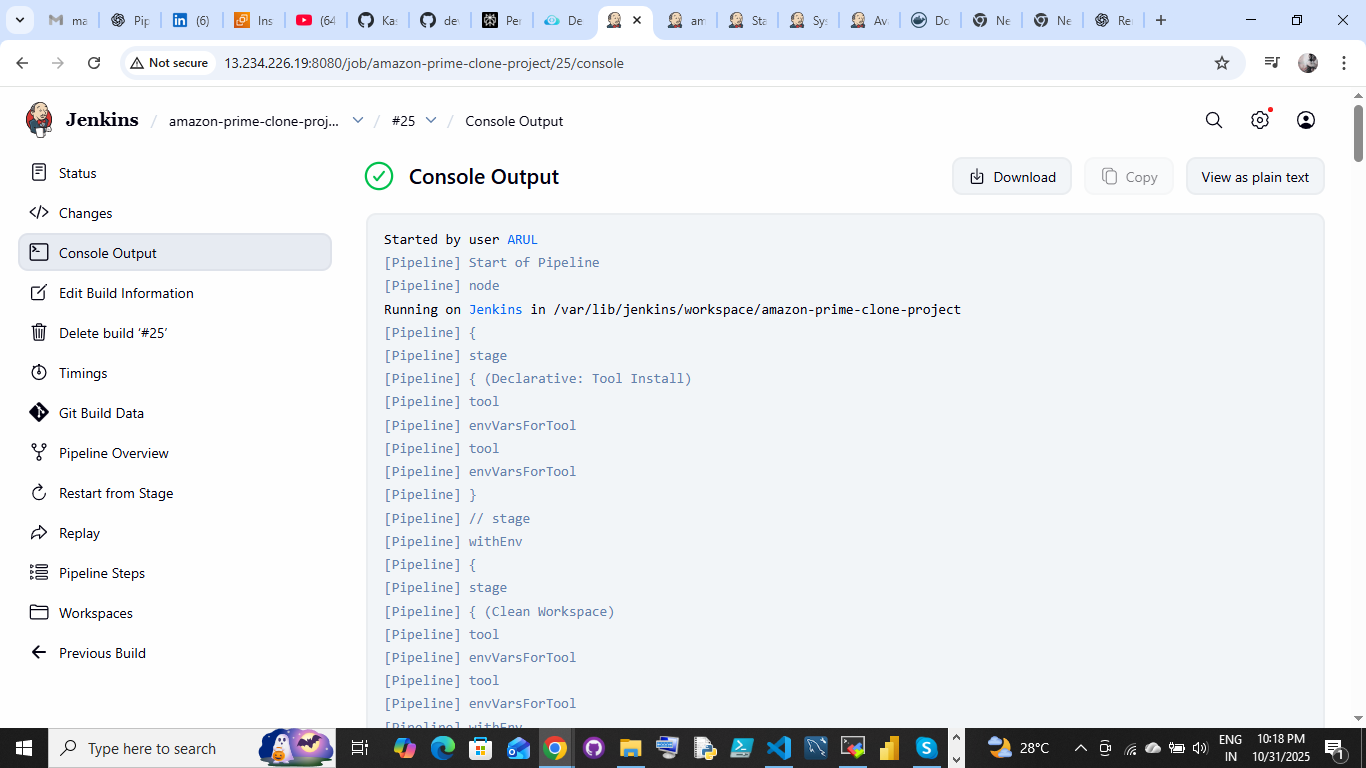
}

}

}

Jenkins output:

Pipeline console output:



Verification:

Started by user [ARUL](http://13.234.226.19:8080/user/admin)

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [Jenkins](http://13.234.226.19:8080/computer/(built-in)/) in /var/lib/jenkins/workspace/amazon-prime-clone-project

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Declarative: Tool Install)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] }

[Pipeline] // stage

[Pipeline] withEnv

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Clean Workspace)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] cleanWs

[WS-CLEANUP] Deleting project workspace...

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

[WS-CLEANUP] done

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Checkout from Git)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] git

The recommended git tool is: NONE

No credentials specified

Cloning the remote Git repository

Cloning repository <https://github.com/arulmurugan001/devopsprojectprime.git>

> git init /var/lib/jenkins/workspace/amazon-prime-clone-project # timeout=10

Fetching upstream changes from <https://github.com/arulmurugan001/devopsprojectprime.git>

> git --version # timeout=10

> git --version # 'git version 2.43.0'

> git fetch --tags --force --progress -- <https://github.com/arulmurugan001/devopsprojectprime.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git config remote.origin.url <https://github.com/arulmurugan001/devopsprojectprime.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/main^{commit} # timeout=10

Checking out Revision 4cccd0bdd9d7ca135bb45803286a6bbae1457654 (refs/remotes/origin/main)

> git config core.sparsecheckout # timeout=10

> git checkout -f 4cccd0bdd9d7ca135bb45803286a6bbae1457654 # timeout=10

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

> git checkout -b main 4cccd0bdd9d7ca135bb45803286a6bbae1457654 # timeout=10

Commit message: "Update and rename service.yaml to service.yaml"

> git rev-list --no-walk 4cccd0bdd9d7ca135bb45803286a6bbae1457654 # timeout=10

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Install Dependencies)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] sh

+ npm install

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>

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

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

npm WARN deprecated sourcemap-codec@1.4.8: Please use @jridgewell/sourcemap-codec instead

npm WARN deprecated workbox-cacheable-response@6.6.0: workbox-background-sync@6.6.0

npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.

added 1490 packages, and audited 1491 packages in 20s

235 packages are looking for funding

run `npm fund` for details

38 vulnerabilities (6 low, 14 moderate, 16 high, 2 critical)

To address issues that do not require attention, run:

npm audit fix

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

npm audit fix --force

Run `npm audit` for details.

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Docker Build & Push)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] script

[Pipeline] {

[Pipeline] withDockerRegistry

$ docker login -u arulmurugan786 -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/amazon-prime-clone-project@tmp/c7f4e9b0-aeb4-4867-86d5-8db7c4886e77/config.json'.

Configure a credential helper to remove this warning. See

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

Login Succeeded

[Pipeline] {

[Pipeline] sh

+ docker build -t amazon-prime-video .

#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile

#1 transferring dockerfile: 581B done

#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/node:alpine

#2 ...

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

#3 DONE 0.0s

#2 [internal] load metadata for docker.io/library/node:alpine

#2 DONE 1.6s

#4 [internal] load .dockerignore

#4 transferring context: 2B done

#4 DONE 0.0s

#5 [1/5] FROM docker.io/library/node:alpine@sha256:7e467cc5aa91c87e94f93c4608cf234ca24aac3ec941f7f3db207367ccccdd11

#5 DONE 0.0s

#6 [internal] load build context

#6 transferring context: 62.22MB 5.0s

#6 transferring context: 109.82MB 10.1s

#6 transferring context: 231.90MB 15.2s

#6 transferring context: 245.24MB 16.5s done

#6 DONE 16.6s

#7 [2/5] WORKDIR /app

#7 CACHED

#8 [3/5] COPY package.json package-lock.json /app/

#8 CACHED

#9 [4/5] RUN npm install

#9 CACHED

#10 [5/5] COPY . /app/

#10 DONE 22.9s

#11 exporting to image

#11 exporting layers

#11 exporting layers 7.6s done

#11 writing image sha256:a852cb546274a916d2e9edc4eff585353a8477440bbda385e96a6e9be8f06125 done

#11 naming to docker.io/library/amazon-prime-video done

#11 DONE 7.6s

+ docker tag amazon-prime-video arulmurugan786/amazon-prime-video:latest

+ docker push arulmurugan786/amazon-prime-video:latest

The push refers to repository [docker.io/arulmurugan786/amazon-prime-video]

7da89350108c: Preparing

e56a2988df76: Preparing

da7e371dcdcf: Preparing

0da39bb25961: Preparing

795e072f0230: Preparing

a2288d9fbb6f: Preparing

29b6fab3a528: Preparing

256f393e029f: Preparing

a2288d9fbb6f: Waiting

29b6fab3a528: Waiting

256f393e029f: Waiting

0da39bb25961: Layer already exists

795e072f0230: Layer already exists

e56a2988df76: Layer already exists

da7e371dcdcf: Layer already exists

a2288d9fbb6f: Layer already exists

256f393e029f: Layer already exists

29b6fab3a528: Layer already exists

7da89350108c: Pushed

latest: digest: sha256:ffeee0498f2ba47912779fac1ab54efb505e67d96a13363746ceac1fdcb88d0c size: 1999

[Pipeline] }

[Pipeline] // withDockerRegistry

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Trivy Image Scan)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] sh

+ trivy image arulmurugan786/amazon-prime-video:latest

2025-10-31T16:24:23Z INFO [vuln] Vulnerability scanning is enabled

2025-10-31T16:24:23Z INFO [secret] Secret scanning is enabled

2025-10-31T16:24:23Z INFO [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning

2025-10-31T16:24:23Z INFO [secret] Please see <https://trivy.dev/v0.67/docs/scanner/secret#recommendation> for faster secret detection

2025-10-31T16:25:54Z INFO Detected OS family="alpine" version="3.22.2"

2025-10-31T16:25:54Z INFO [alpine] Detecting vulnerabilities... os\_version="3.22" repository="3.22" pkg\_num=18

2025-10-31T16:25:54Z INFO Number of language-specific files num=1

2025-10-31T16:25:54Z INFO [node-pkg] Detecting vulnerabilities...

2025-10-31T16:25:55Z INFO Table result includes only package filenames. Use '--format json' option to get the full path to the package file.

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (deploy to kubernetes)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] dir

Running in /var/lib/jenkins/workspace/amazon-prime-clone-project/k8s

[Pipeline] {

[Pipeline] script

[Pipeline] {

[Pipeline] withCredentials

Masking supported pattern matches of $KUBECONFIG

[Pipeline] {

[Pipeline] sh

+ kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-172-31-35-97 Ready control-plane 45h v1.30.14

ip-172-31-46-45 Ready <none> 45h v1.30.14

+ kubectl apply -f manifest.yml

deployment.apps/amazon-prime-video-deployment created

service/amazon-prime-video-service created

[Pipeline] }

[Pipeline] // withCredentials

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // dir

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Declarative: Post Actions)

[Pipeline] script

[Pipeline] {

[Pipeline] emailext

Sending email to: muruganarul1993@gmail.com

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

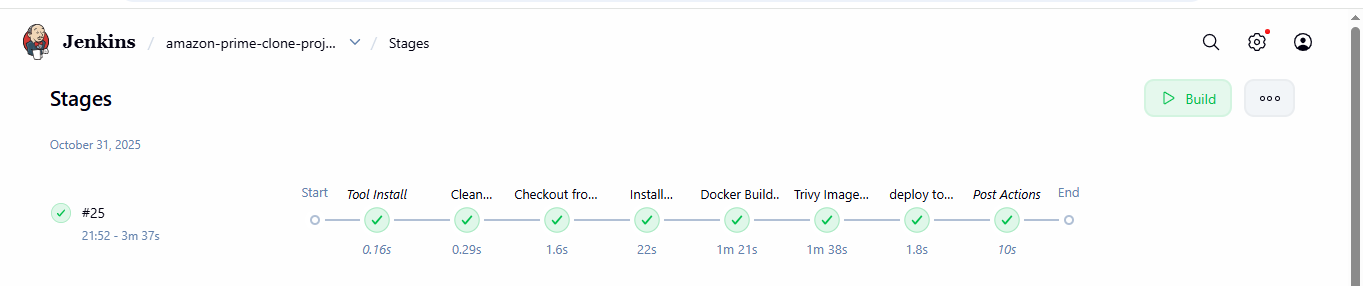
[Pipeline] // withEnv

[Pipeline] }

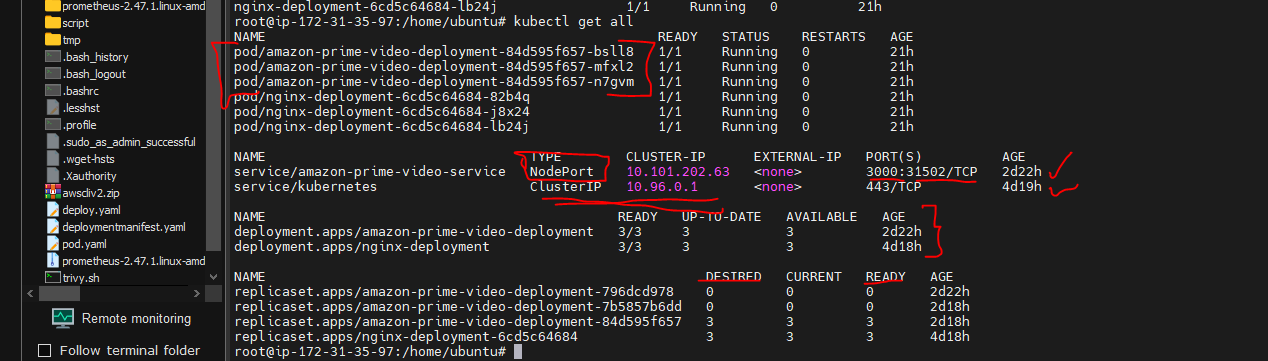
[Pipeline] // node

[Pipeline] End of Pipeline

Finished: SUCCESS

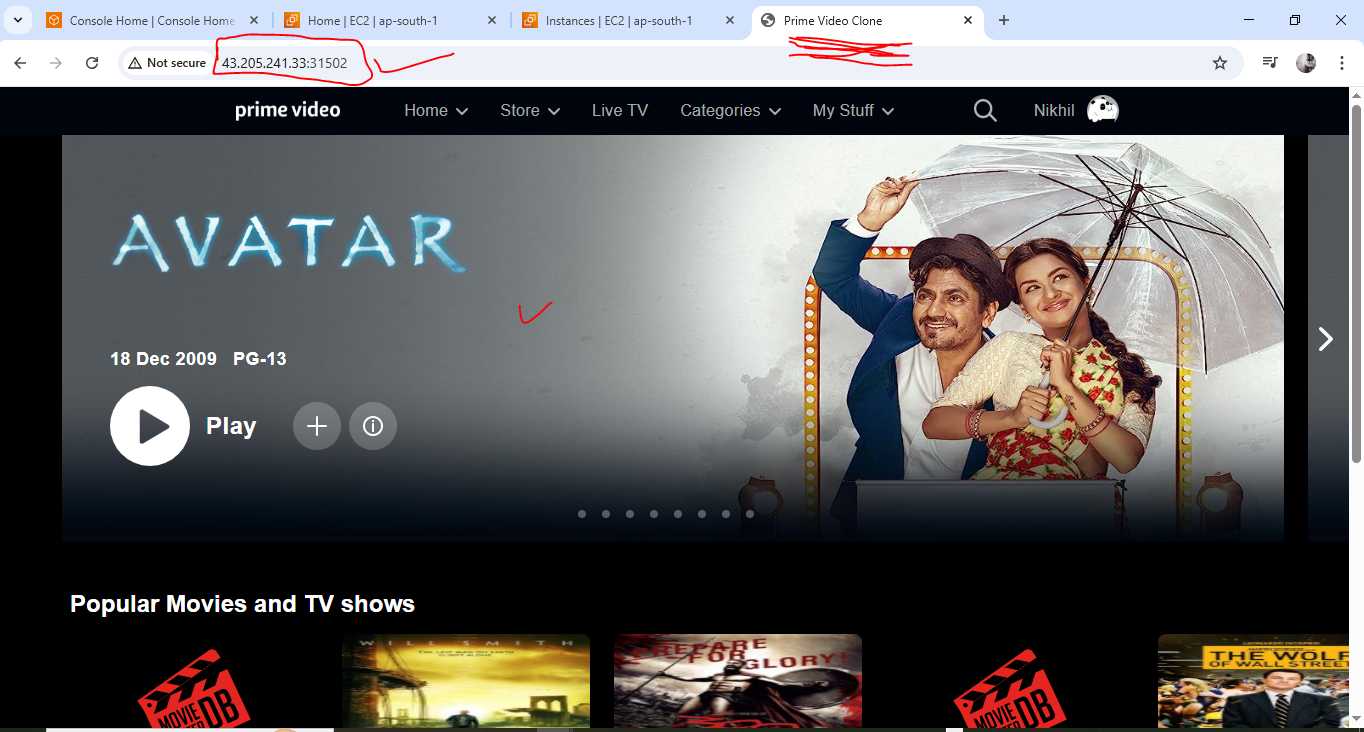


**KUBERNET MACHINE OUT PUT:**



Browser view of kubernetes deployments:

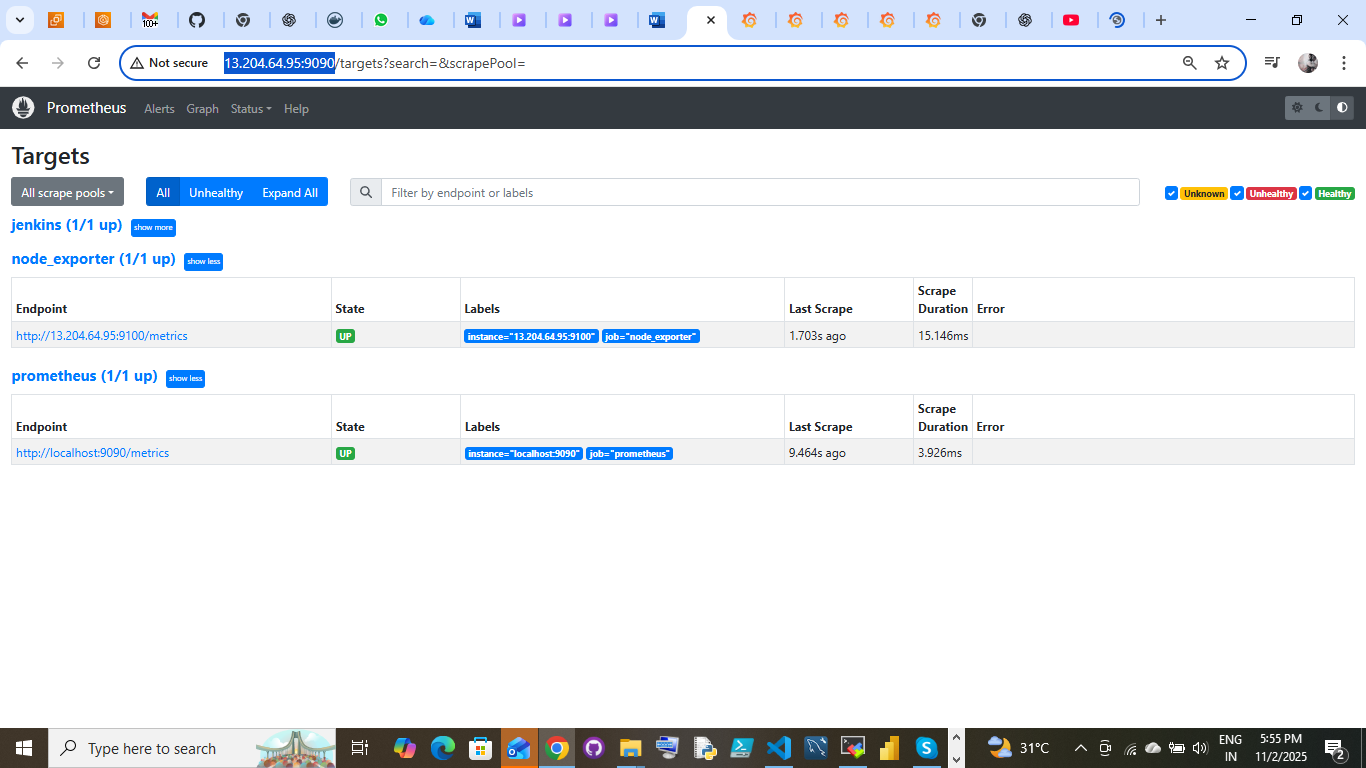
Acceess : <http://publicip:31502> (worker node publicip : nodeport portnumber)



**Promotheus:**

**output:metrics**

**access :** [**http://publicip:9090**](http://publicip:9090) **portnumber**



**Grafana Confguration:**

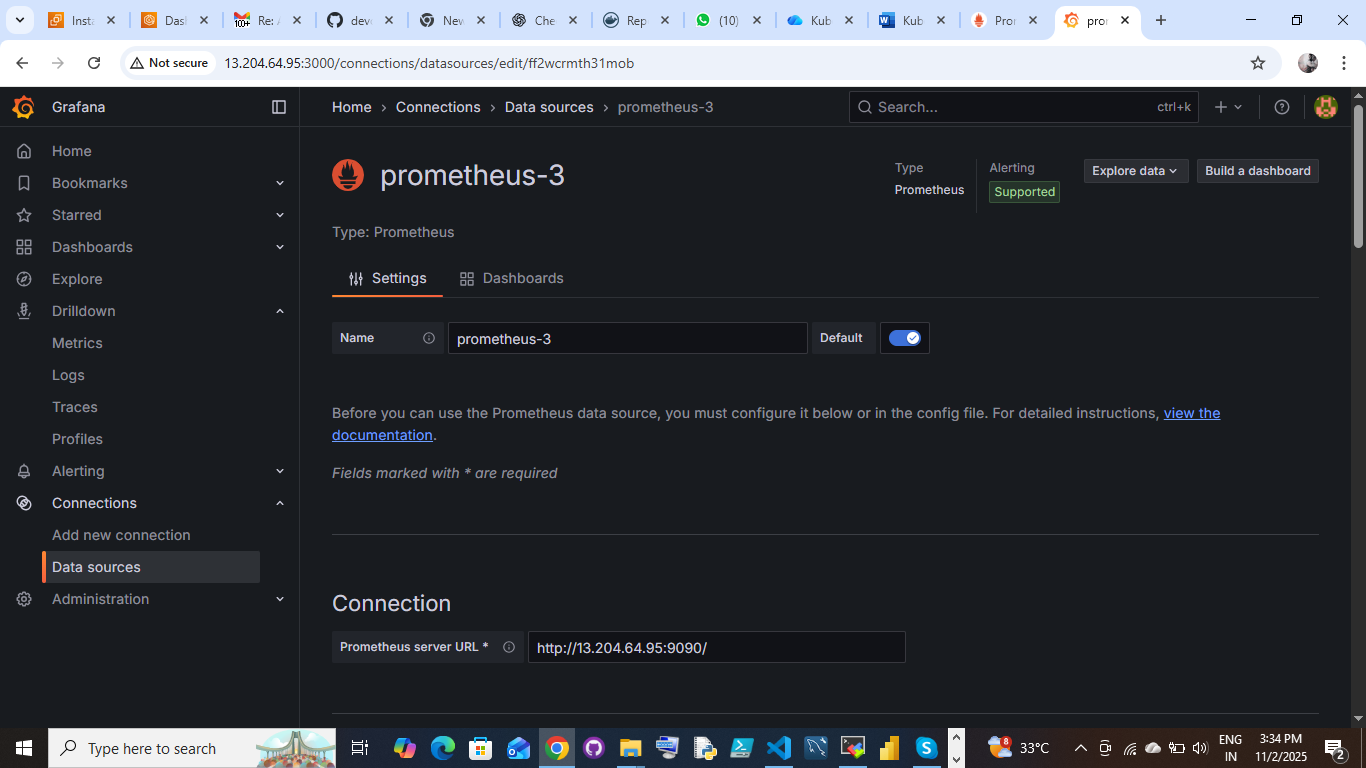
Add Prometheus into Grafana

Login:

Go to connection

add new connection then create datasource

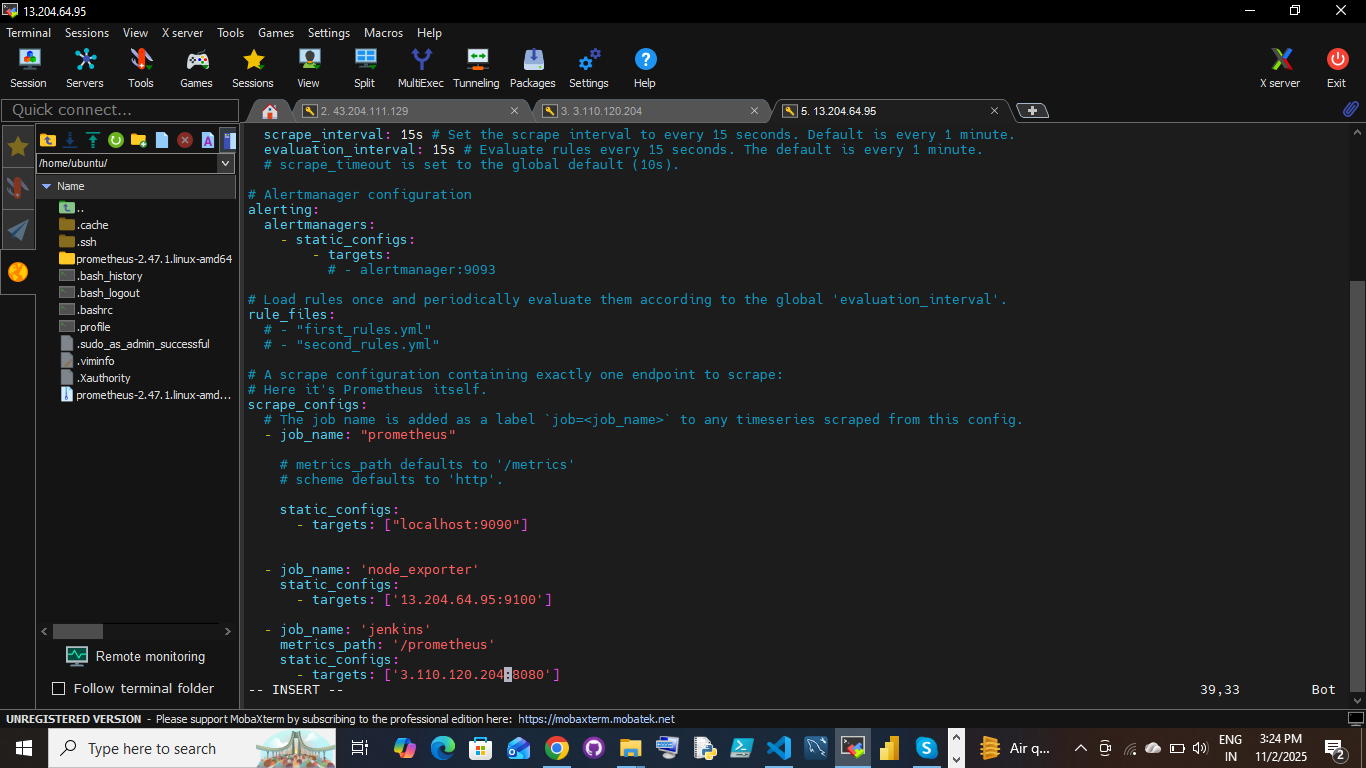
Add new source <http://pu-ip:9090> (of promotheus)



Before we add the node exporter and Jenkins server configuration file into Promethues.yaml file

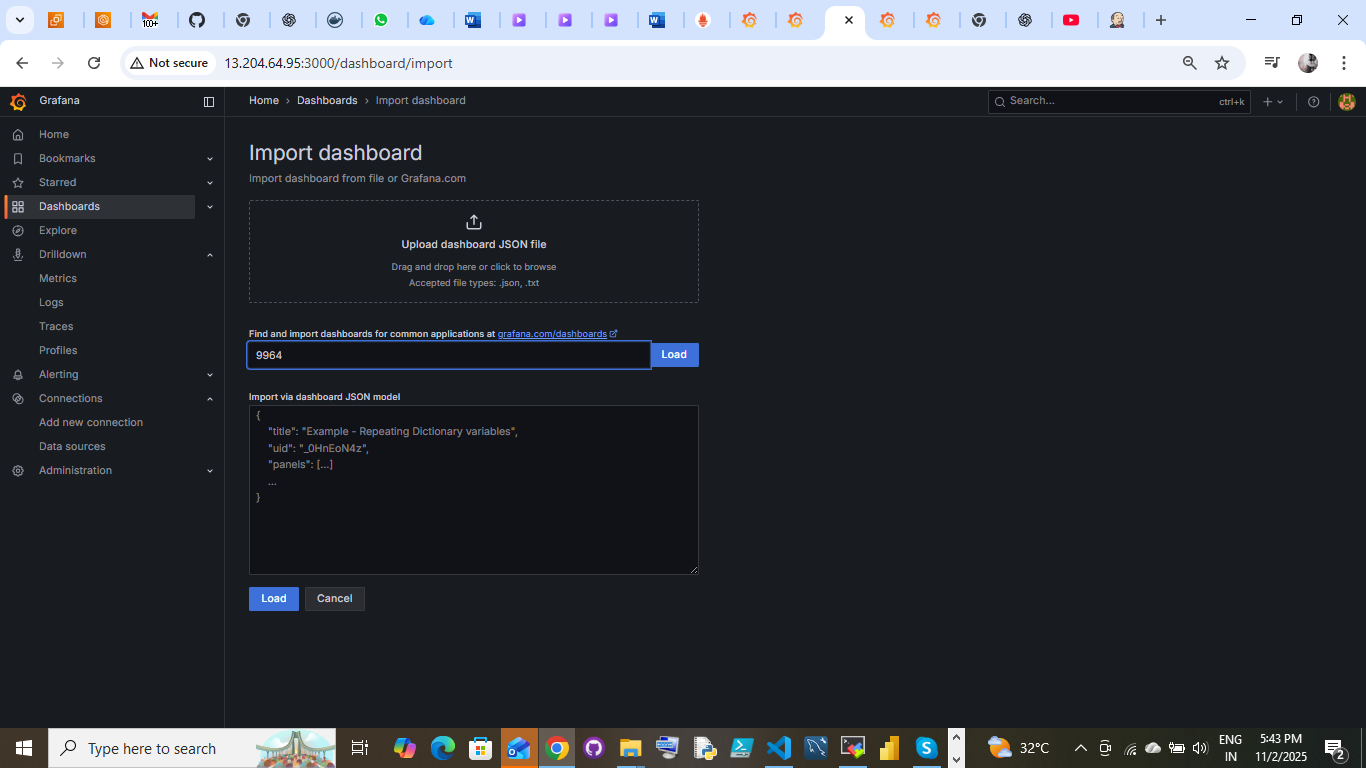
**Configuration:**

Go to machine open Promethues.yaml file then add the script:



Build Dashboard: Jenkins

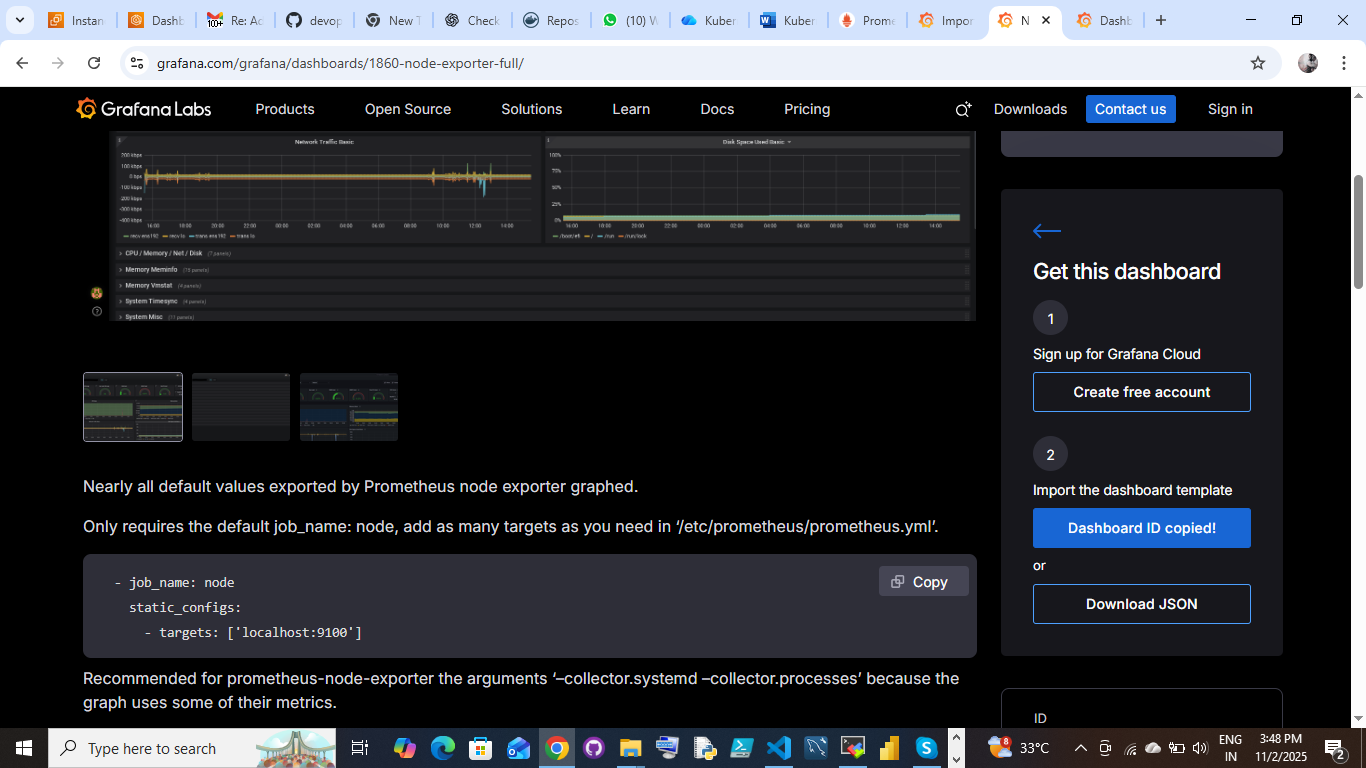
We have to create the Jenkins: Add the Jenkins id 9964 default ID Jenkins Health



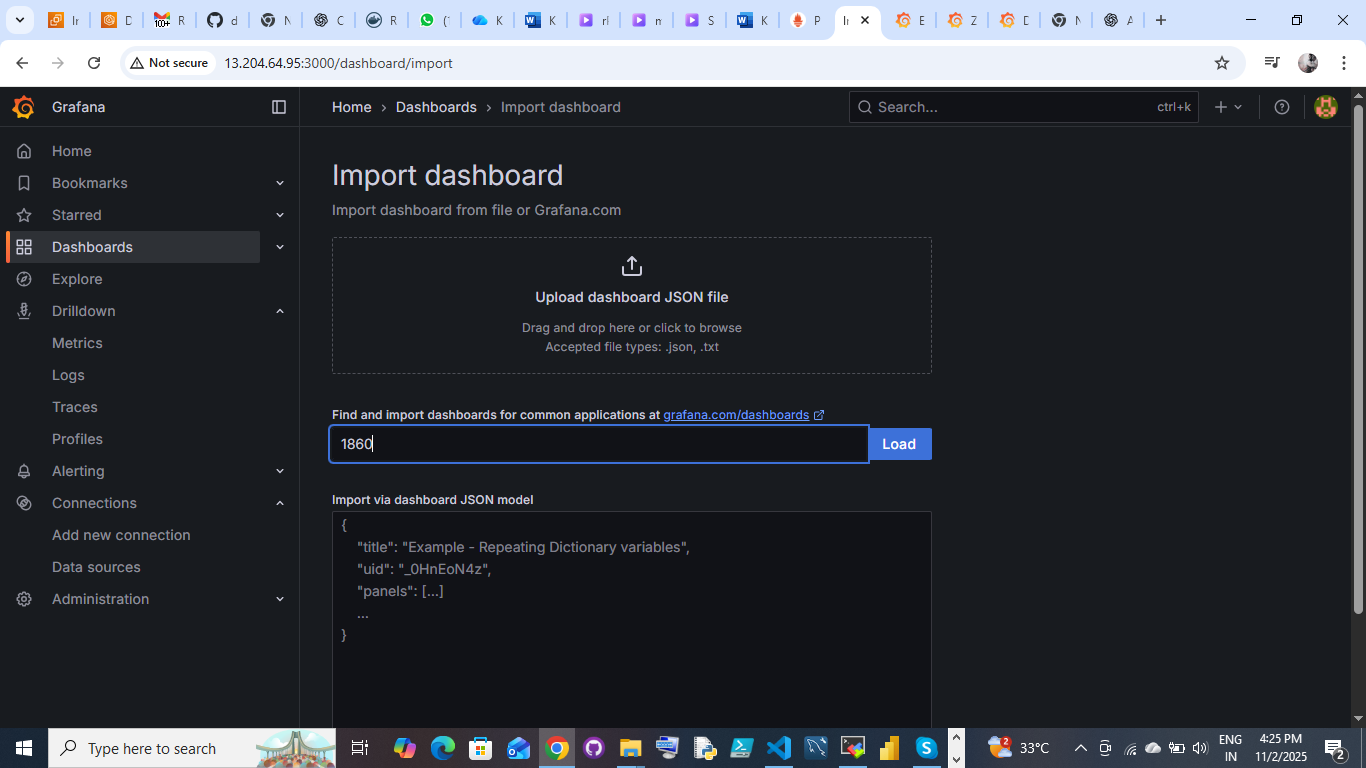
**node exporter:**

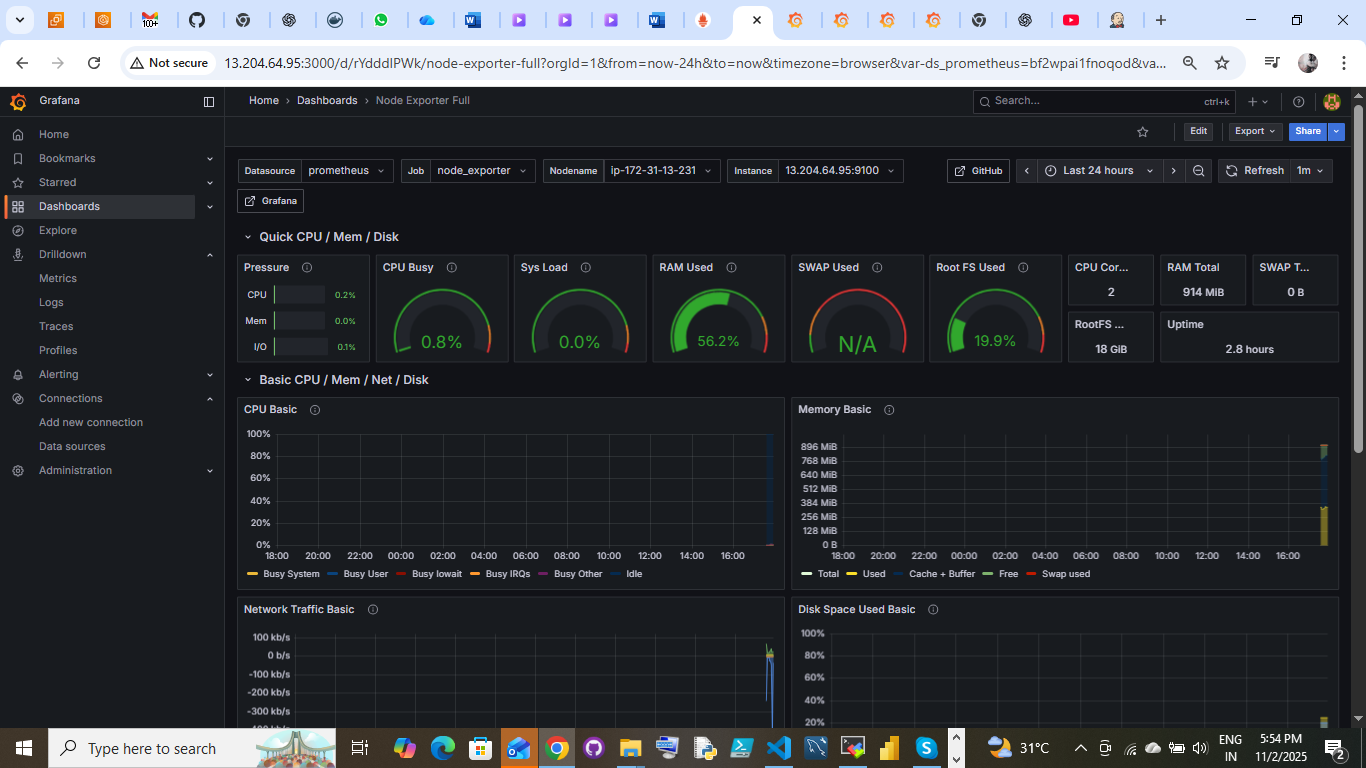
Go to Grafana dashboard <https://grafana.com/grafana/dashboards/> Get the node exporter id:

1860



**Import dashboard:**





Graffana visulaazations:

Jenkins Dashboard: Monitoring Jenkins servers:

