

Module - Capstone Project

Health Care Project

Submitted by:- Jonna Padmarao

Submission Date:- 22/05/2025

Resubmission Date:-

Step1:- On the desktop create a new folder (star-agile-health-care-Pro) and enter into that folder and open the git bash in that folder

Step2:- Now give git clone

<https://github.com/StarAgileDevOpsTraining/star-agile-health-care.git> to get the project code in to that folder

MINGW64:/c/Users/Pj/Desktop/Star-agile-health-care-Pro

```
Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro
$ git clone https://github.com/StarAgileDevOpsTraining/star-agile-health-care.git
Cloning into 'star-agile-health-care'...
remote: Enumerating objects: 174, done.
remote: Total 174 (delta 0), reused 0 (delta 0), pack-reused 174 (from 1)
Receiving objects: 100% (174/174), 1.81 MiB | 1.21 MiB/s, done.
Resolving deltas: 100% (52/52), done.

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro
$ ls -al
total 8
drwxr-xr-x 1 Pj 197121 0 May 19 12:33 ./
drwxr-xr-x 1 Pj 197121 0 May 19 12:30 ../
drwxr-xr-x 1 Pj 197121 0 May 19 12:34 star-agile-health-care/

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro
$ |
```

This PC > Desktop > Star-agile-health-care-Pro > star-agile-health-care >					
	Name	Date modified	Type	Size	
	.mvn	5/19/2025 12:34 PM	File folder		
	src	5/19/2025 12:34 PM	File folder		
	.DS_Store	5/19/2025 12:34 PM	DS_STORE File	7 KB	
Personal	.gitignore	5/19/2025 12:34 PM	Git Ignore Source File	1 KB	
	ansible-playbook	5/19/2025 12:34 PM	Yaml Source File	1 KB	
	Dockerfile	5/19/2025 12:34 PM	File	1 KB	
	mvnw	5/19/2025 12:34 PM	File	11 KB	
	mvnw	5/19/2025 12:34 PM	Windows Comman...	7 KB	
	pom	5/19/2025 12:34 PM	XML Document	2 KB	

Step3:- Now go to the folder that we get from git clone and again open git bash there and check the origin and remove that origin

git remote -v --> To get origin list

git remote remove origin to remove the origin

```

MINGW64:~/Desktop/Star-agile-Finance-Pro/star-agile-banking-finance
Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-Finance-Pro/star-agile-banking-finance (master)
$ git remote -v
origin https://github.com/StarAgileDevOpsTraining/star-agile-banking-finance (fetch)
origin https://github.com/StarAgileDevOpsTraining/star-agile-banking-finance (push)

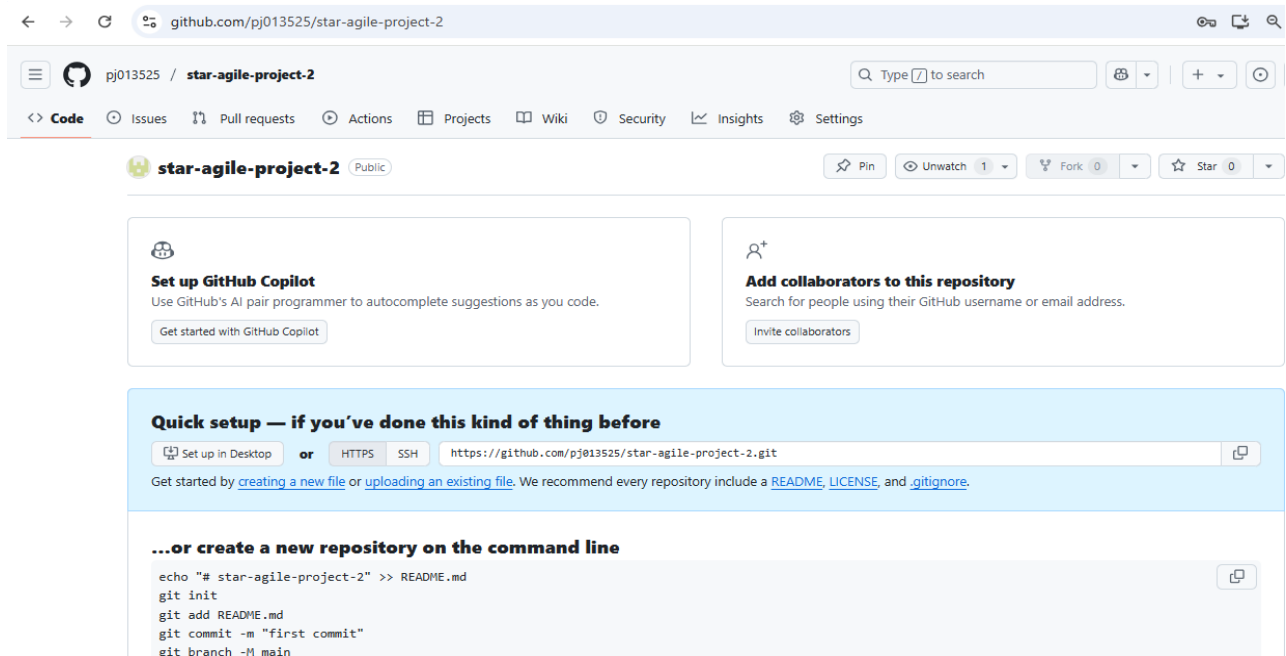
Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-Finance-Pro/star-agile-banking-finance (master)
$ git remote remove origin

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-Finance-Pro/star-agile-banking-finance (master)
$ git remote -v

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-Finance-Pro/star-agile-banking-finance (master)
$

```

Step4:- Now go to github and create a new repo and copy the url in the gitbash



Step5:- Now again go to the gitbash and add this git repo url in the project by using `git remote add origin <git-repo-url>` and verify

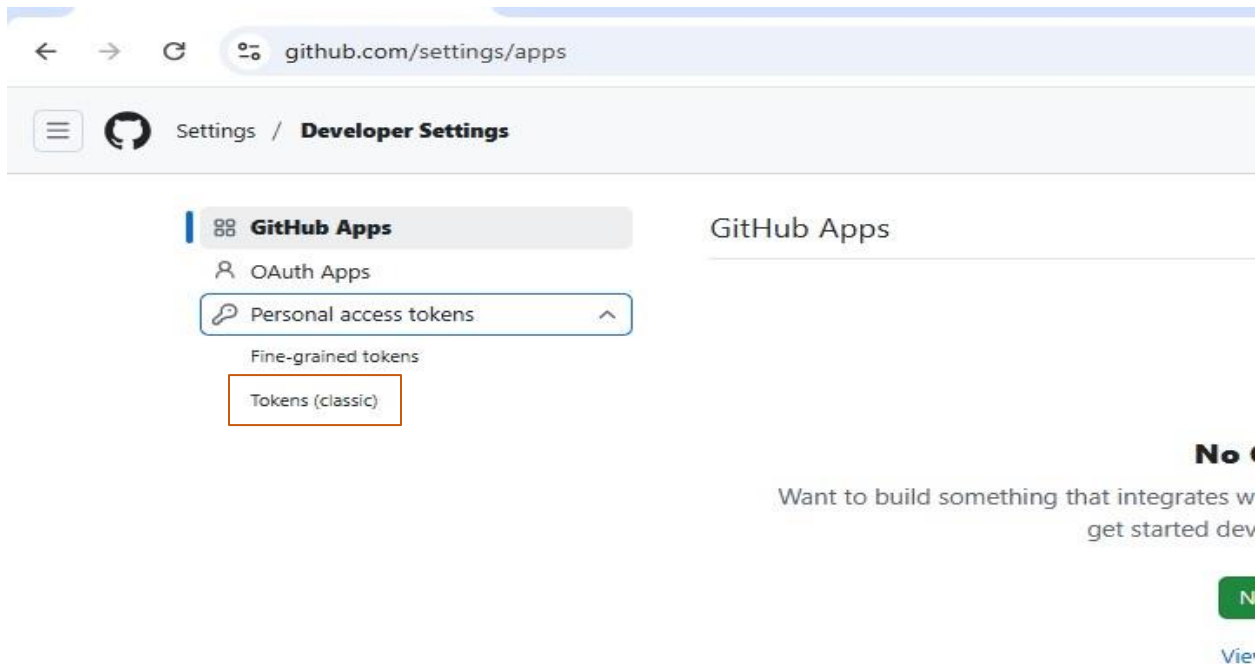
```
MINGW64/c/Users/Pj/Desktop/Star-agile-health-care-Pro/star-agile-health-care

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro/star-agile-health-care (master)
$ git remote add oigin https://github.com/pj013525/star-agile-project-2.git

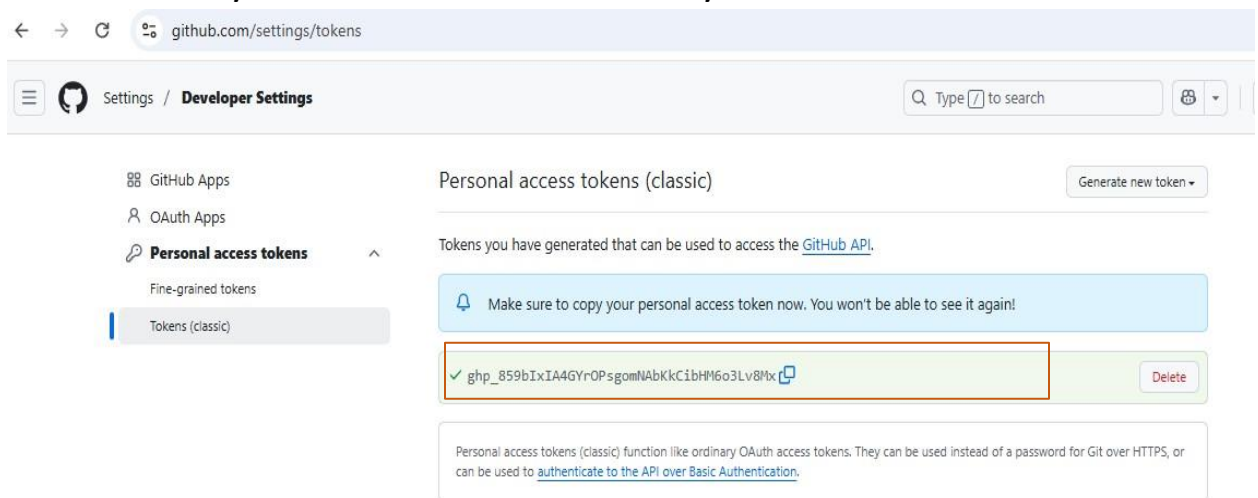
Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro/star-agile-health-care (master)
$ git remote -v
oigin    https://github.com/pj013525/star-agile-project-2.git (fetch)
oigin    https://github.com/pj013525/star-agile-project-2.git (push)

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro/star-agile-health-care (master)
$ |
```

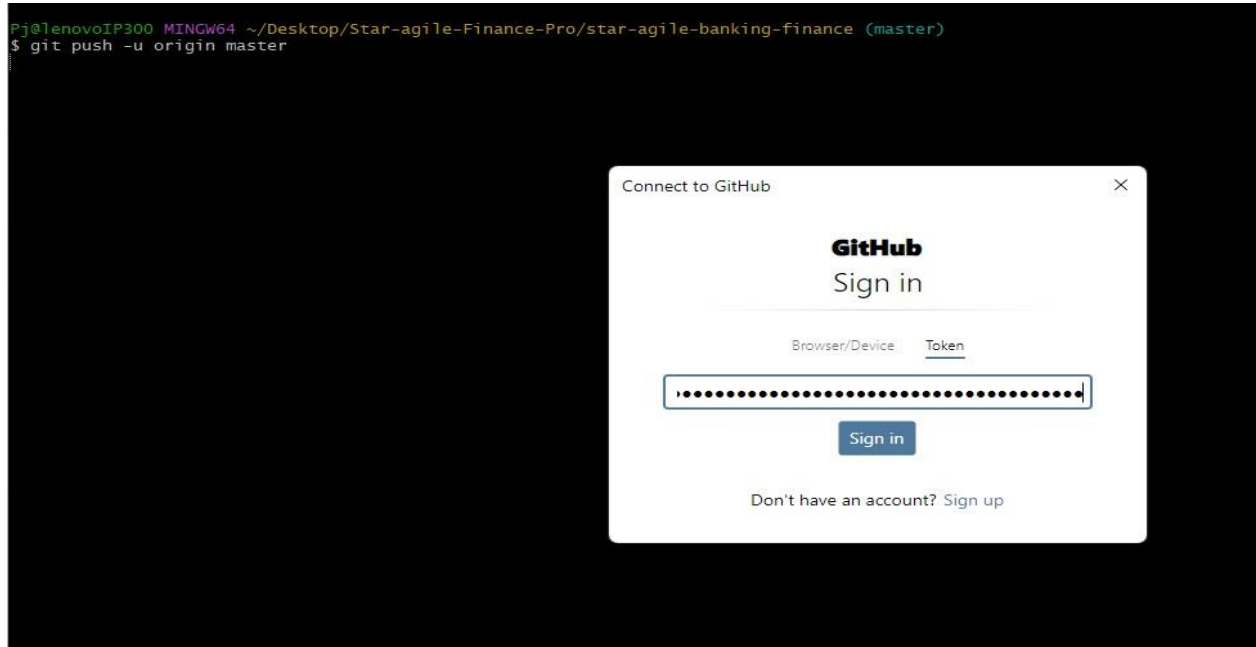
Step6:- Now again go to github ⑦ Profile setting ⑦ Developer settings
⑦ Personal access token Tokens(classic) ⑦ Generate new token



Step7:- Now a token will be generated , copy this token that generated since it is only available for one time only



Step8:- Now give link this the remote repo with gitbash using this token
git push -u origin master and paste the token the copied from the
github and press sign in

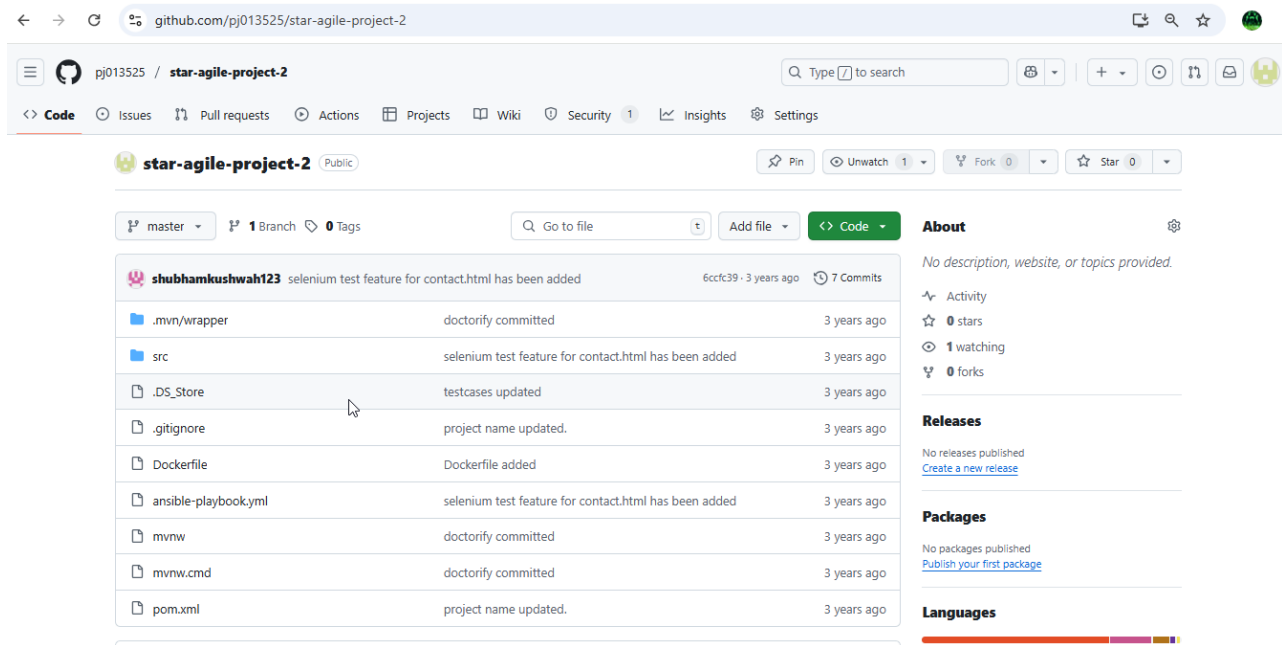


Step9:- Now the master branch will be set to our repo by default

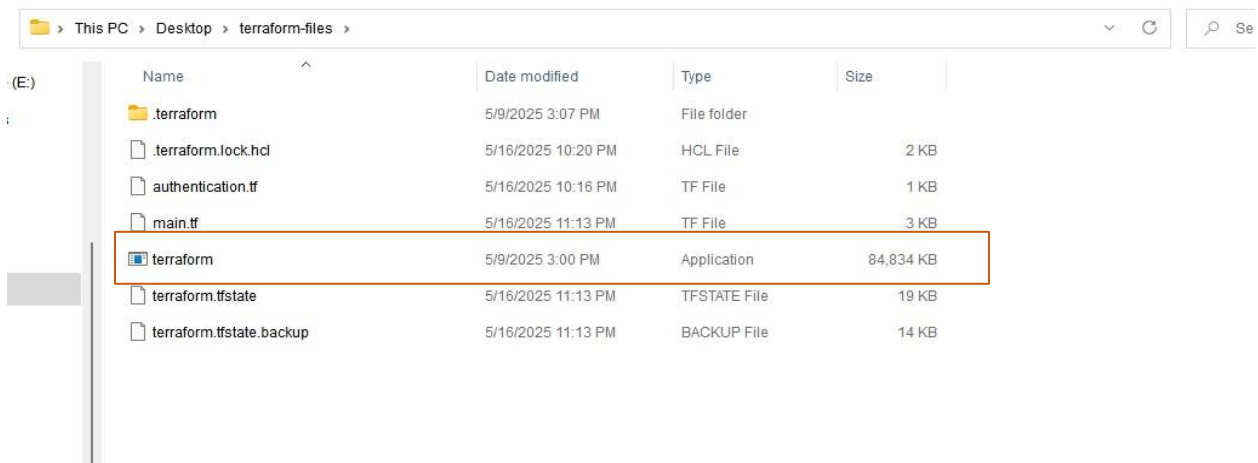
```
Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro/star-agile-health-care (master)
$ git push -u origin master
Enumerating objects: 174, done.
Counting objects: 100% (174/174), done.
Delta compression using up to 4 threads
Compressing objects: 100% (84/84), done.
Writing objects: 100% (174/174), 1.81 MiB | 1.30 MiB/s, done.
Total 174 (delta 52), reused 174 (delta 52), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (52/52), done.
To https://github.com/pj013525/star-agile-project-2.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

Pj@lenovoIP300 MINGW64 ~/Desktop/Star-agile-health-care-Pro/star-agile-health-care (master)
$ |
```

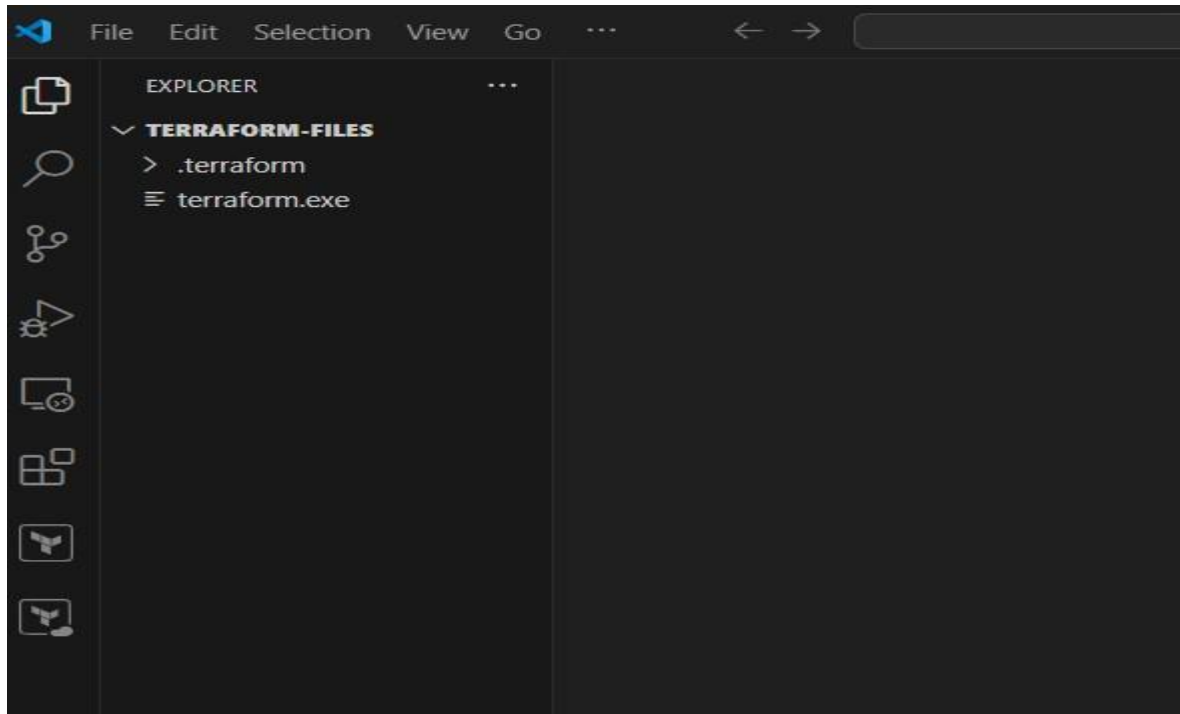
Step10:- Now go to the github repo and you will see the source code in that repo



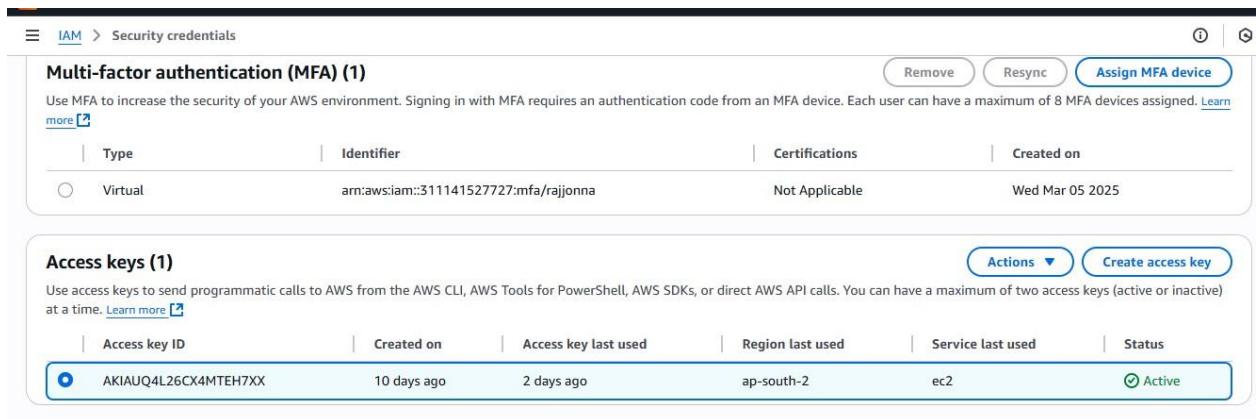
Step11:- Now create an instance using terraform as laac , and for that create a folder on desktop and go to browser download terraform for windows then a terraform application will be generated , now copy this application in to that folder and save



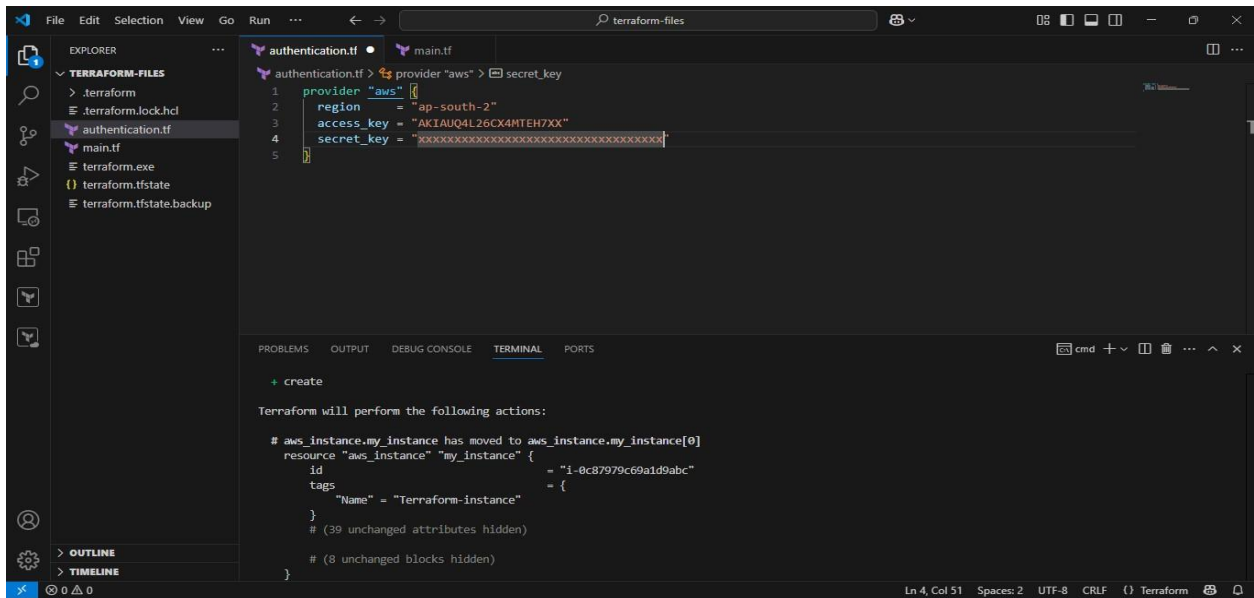
Step12:- open visual studio code and go to terraform folder



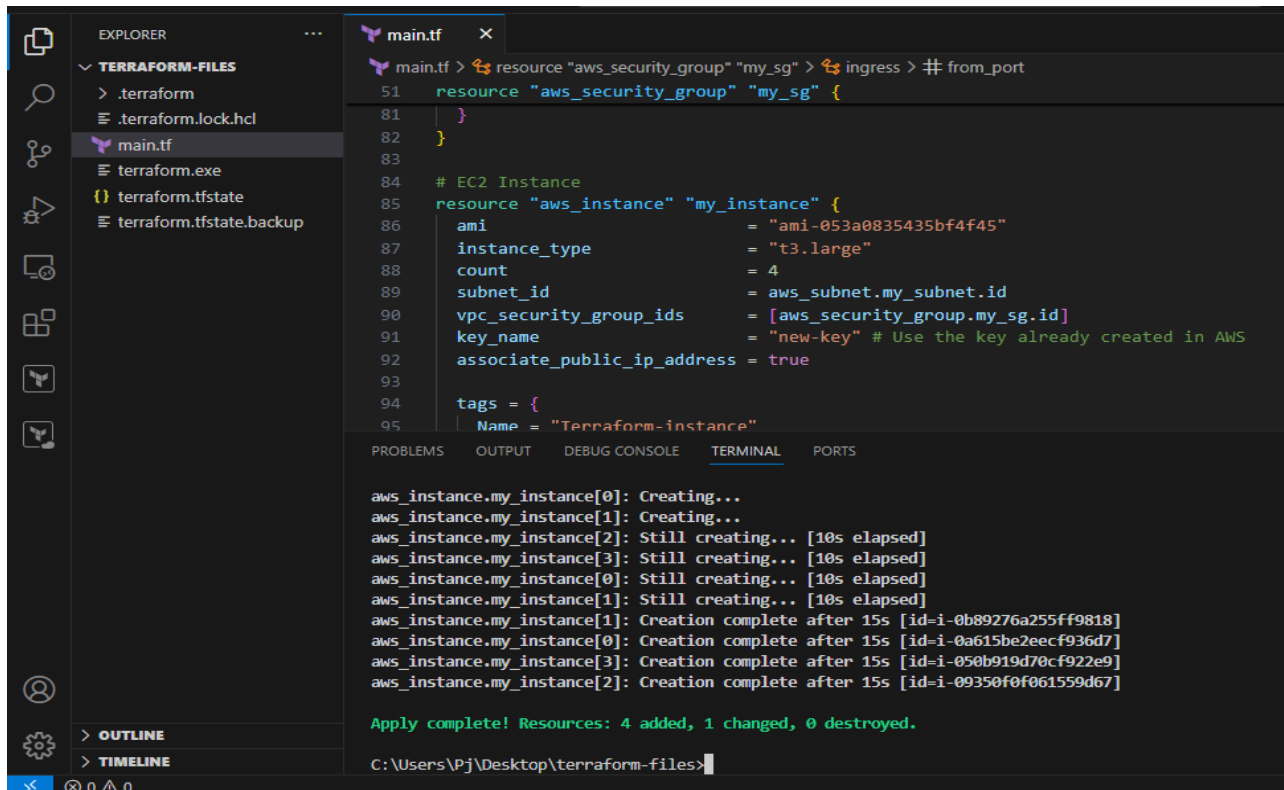
Step13:- Now create a file authentication.tf and give the provider and for that select the region in which you want to launch the server and go to aws account and go to profile ⑦ credentials and go to access keys



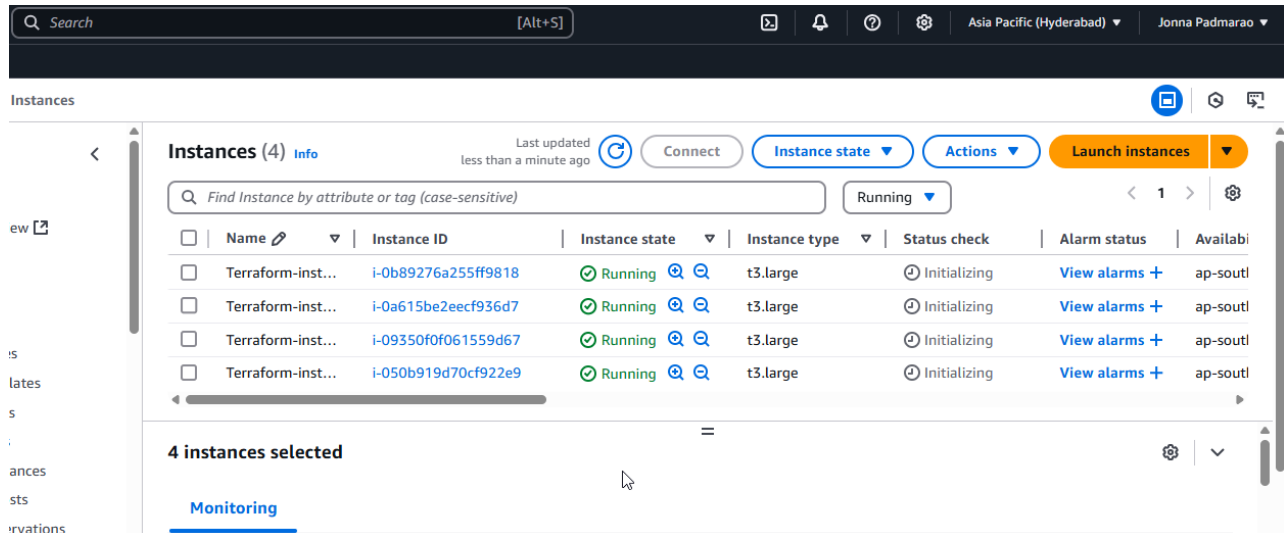
Step14:- now copy this access key details and paste it in this authentication.tf file and initialize it



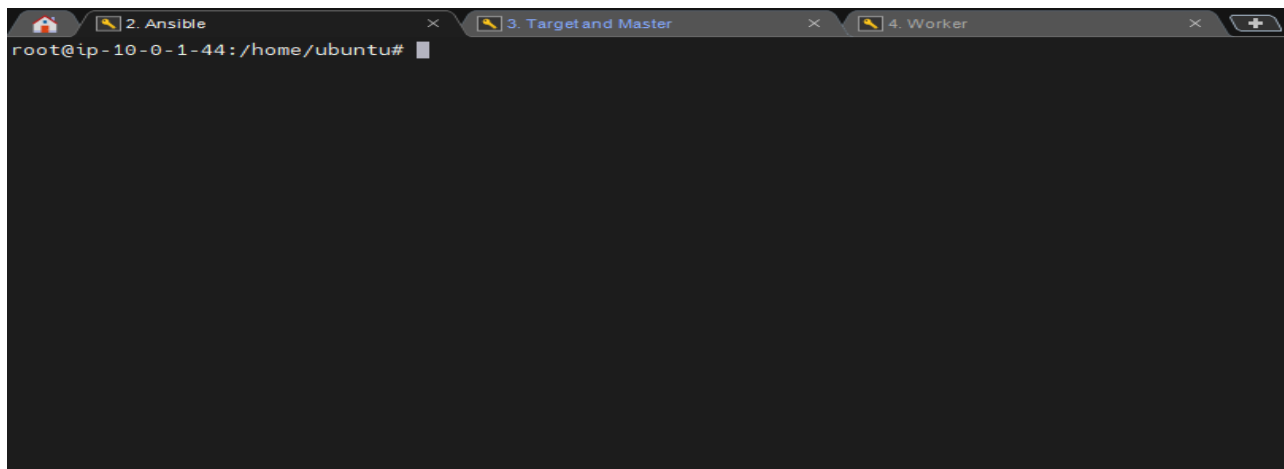
Step15:- After it is successful now create a new file main.tf and give resources details to create instance



Step16:- After it is successful go and check the aws console and rename them as Ansible , Target and Master, Worker and Grafana instances



Step17:- Now connect to Ansible and target and master and worker servers using MobaXterm agent and launch an instance



Step18:- Now install [Ansible in Ansible server](#) and connect this server with the Target and master sever and enable All traffic in the security group of this server

```
2. Ansible 3. Target and Master 4. Worker
root@ip-10-0-1-44:/home/ubuntu# ansible --version
ansible [core 2.18.6]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Feb  4 2025, 14:48:35) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
root@ip-10-0-1-44:/home/ubuntu#
```

```
root@ip-10-0-1-44:/home/ubuntu# ansible all -m ping
[WARNING]: Platform linux on host 10.0.1.242 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of
another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.18/reference_appendices/interpreter_discovery.html for more information.
10.0.1.242 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3.12"
  },
  "changed": false,
  "ping": "pong"
}
root@ip-10-0-1-44:/home/ubuntu#
```

Step19:- Now install java, maven, docker, jenkins in the target and master server using Ansible sever

```
2. Ansible 3. Target and Master 4. Worker
$ cat installation.yml
---
- name: Install Java 17, Maven, Git, Jenkins, and Docker
  hosts: target
  become: yes
  tasks:
    - name: Install dependencies
      apt:
        name:
          - apt-transport-https
          - ca-certificates
          - curl
          - software-properties-common
          - gnupg
        update_cache: yes
        state: present

    - name: Install OpenJDK 17
      apt:
        name: openjdk-17-jdk
        state: present

    - name: Install Maven
      apt:
        name: maven
        state: present

    - name: Install Git
      apt:
        name: git
        state: present

    - name: Add Jenkins GPG key
      apt_key:
        url: https://pkg.jenkins.io/debian/jenkins.io.key
        state: present
```

```
state: present

- name: Add Jenkins repository
  apt_repository:
    repo: 'deb https://pkg.jenkins.io/debian binary/'
    state: present

- name: Install Jenkins
  apt:
    name: jenkins
    update_cache: yes
    state: present

- name: Enable and start Jenkins
  systemd:
    name: jenkins
    enabled: yes
    state: started

- name: Add Docker GPG key
  apt_key:
    url: https://download.docker.com/linux/ubuntu/gpg
    state: present

- name: Add Docker repository
  apt_repository:
    repo: "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"
    state: present

- name: Install Docker
  apt:
    name:
      - docker-ce
      - docker-ce-cli
      - containerd.io
    update_cache: yes
    state: present
```

Step20:- Now go to target and master node and verify the packages

```
TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 10.0.1.242 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of
another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference\_appendices/interpreter\_discovery.html for more information.
ok: [10.0.1.242]

TASK [Update apt cache] *****
changed: [10.0.1.242]

TASK [Install dependencies] *****
ok: [10.0.1.242]

TASK [Install Java 17] *****
ok: [10.0.1.242]

TASK [Install Maven] *****
ok: [10.0.1.242]

TASK [Install Git] *****
ok: [10.0.1.242]

TASK [Add Docker GPG key] *****
ok: [10.0.1.242]

TASK [Add Docker repository] *****
ok: [10.0.1.242]

TASK [Install Docker] *****
ok: [10.0.1.242]

TASK [Enable and start Docker service] *****
ok: [10.0.1.242]

PLAY RECAP *****
10.0.1.242 : ok=10 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

root@ip-10-0-1-44: /home/ubuntu#
```

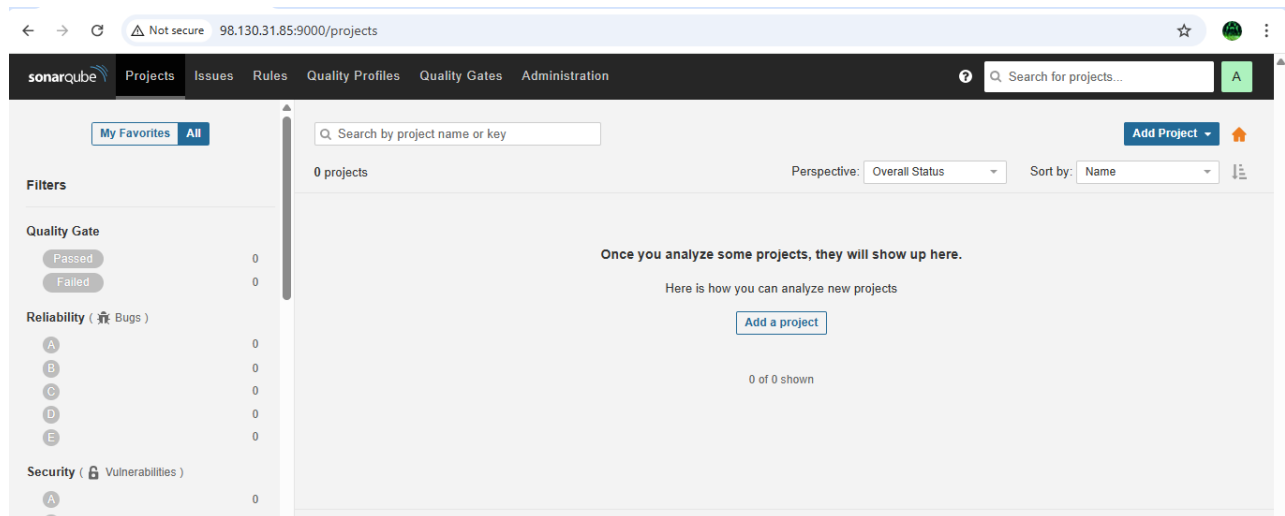
```
root@ip-10-0-1-242:/home/ubuntu# jenkins --version
2.511
root@ip-10-0-1-242:/home/ubuntu# docker --version
Docker version 28.1.1, build 4eba377
root@ip-10-0-1-242:/home/ubuntu# git --version
git version 2.43.0
root@ip-10-0-1-242:/home/ubuntu# mvn --version
Apache Maven 3.8.7
Maven home: /usr/share/maven
Java version: 17.0.15, vendor: Ubuntu, runtime: /usr/lib/jvm/java-17-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.8.0-1024-aws", arch: "amd64", family: "unix"
root@ip-10-0-1-242:/home/ubuntu# java --version
openjdk 17.0.15 2025-04-15
OpenJDK Runtime Environment (build 17.0.15+6-Ubuntu-0ubuntu124.04)
OpenJDK 64-Bit Server VM (build 17.0.15+6-Ubuntu-0ubuntu124.04, mixed mode, sharing)
root@ip-10-0-1-242:/home/ubuntu#
```

Step21:- Now add Jenkins group to docker and give root permissions to the Jenkins user in the sudoers file as under root give

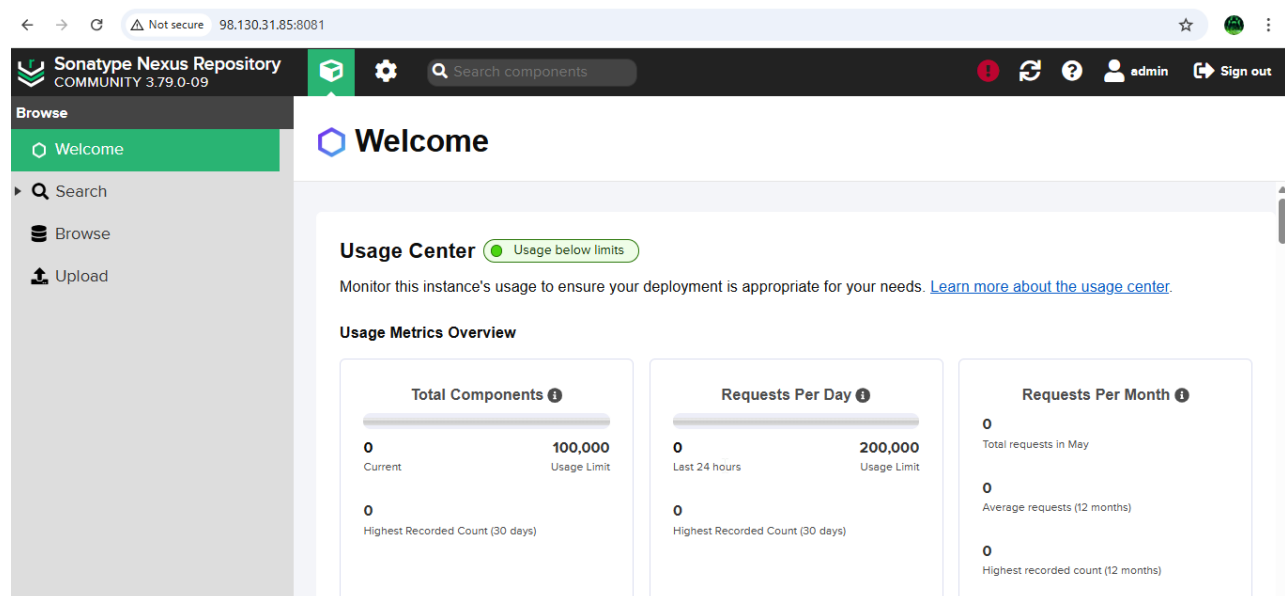
`jenkins ALL=(ALL:ALL) NOPASSWD: ALL` restart the jenkins

```
root@ip-10-0-1-242:/home/ubuntu# sudo usermod -aG docker jenkins
root@ip-10-0-1-242:/home/ubuntu# sudo newgrp docker
sudo: newgrp: command not found
root@ip-10-0-1-242:/home/ubuntu# sudo newgrp docker
root@ip-10-0-1-242:/home/ubuntu# vim /etc/sudoers
root@ip-10-0-1-242:/home/ubuntu# service restart jenkins
restart: unrecognized service
root@ip-10-0-1-242:/home/ubuntu# systemctl jenkins restart
Unknown command verb 'jenkins', did you mean 'kill'?
root@ip-10-0-1-242:/home/ubuntu# systemctl restart jenkins
root@ip-10-0-1-242:/home/ubuntu#
```

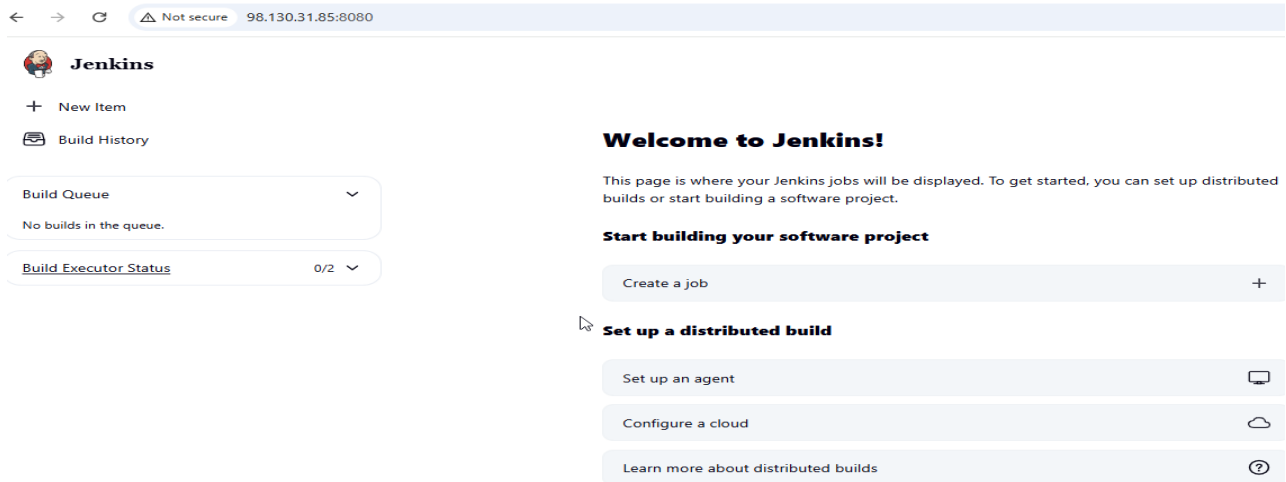
Step22:- Now I am installing sonarqube in the target for code Quality analysis for checking any type of bugs , vulnerabilities in the code and logging into the sonarqube home page using browser with ip:9000



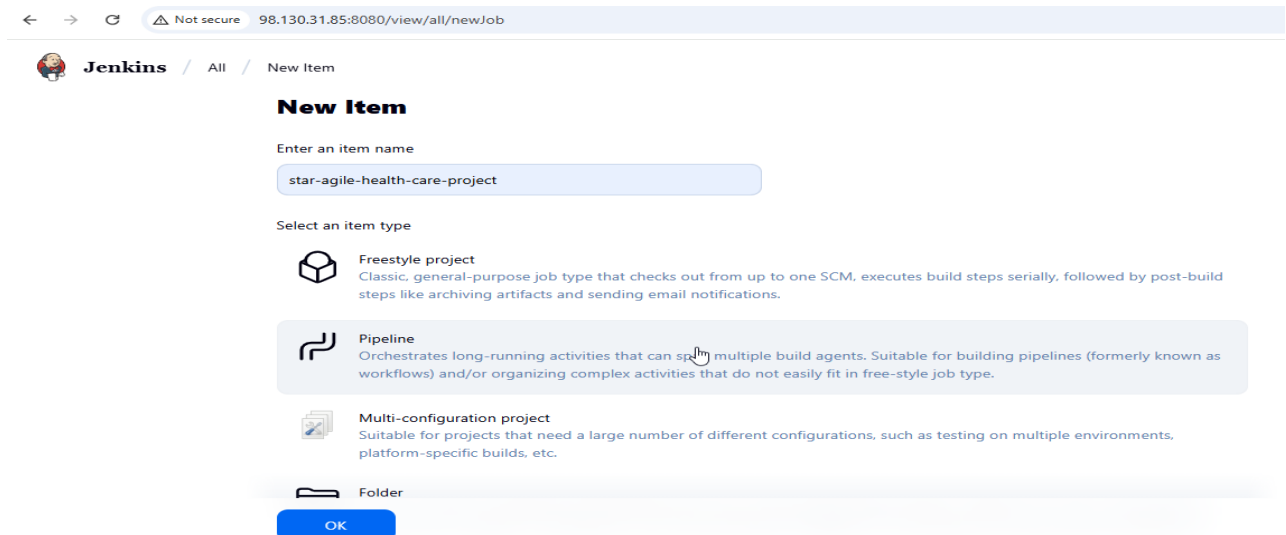
Step23:- Now i am installing nexus tool in the target node to store the artifacts that generated from the maven build



Step24:- Go to the any browser and give the details and click on recommended plugins and login to the Jenkins



Step25:- Now in the Jenkins dashboard click on new item and give any name and select pipeline project as type and click on ok



Step26:- Now install docker and other required plugins in the Jenkins

Pipeline stage view

Git Plugin

Docker Pipeline Plugin

Credentials Binding Plugin

Docker Commons Plugin

Pipeline: GitHub

Maven Integration Plugin

nexus artifact uploader

SonarQube scanner

Sonar quality gates

← → ↻ ⚠ Not secure 98.130.31.85:8080/manage/pluginManager/updates/

Jenkins / Manage Jenkins / Plugins

Plugins

- Updates
- Available plugins
- Installed plugins
- Advanced settings
- Download progress

Email Extension	✓ Success
Mailer	✓ Success
Theme Manager	✓ Success
Dark Theme	✓ Success
Loading plugin extensions	✓ Success
Pipeline: REST API	✓ Success
Pipeline: Stage View	✓ Success
Javadoc	✓ Success
Dev Tools Symbols API	✓ Success
jsoup API	✓ Success
JSch dependency	✓ Success
Maven Integration	✓ Success
Authentication Tokens API	✓ Success
Docker Commons	✓ Success
Docker Pipeline	✓ Success
JavaMail API	✓ Success
Commons HttpClient 3.x API	✓ Success
Nexus Artifact Uploader	✓ Success
SonarQube Scanner	✓ Success
Apache HttpComponents Client 5.x API	✓ Success
Sonar Quality Gates	✓ Success
Loading plugin extensions	✓ Success

Step27:- Now in dash board ⑦ manage Jenkins ⑦ credentials ⑦ global ⑦ add credentials and give dockerhub user name and password and click on create

← → ↻ ⚠ Not secure 98.130.31.85:8080/manage/credentials/store/system/domain/_/credential/dockerhub-creds/update

Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrestricted) / pj013525/***** (dockerhub)

Update credentials

- Update
- Delete
- Move

Scope ?
Global (Jenkins, nodes, items, all child items, etc) ▼

Username ?
pj013525

☐ Treat username as secret ?

Password ?
Concealed Change Password

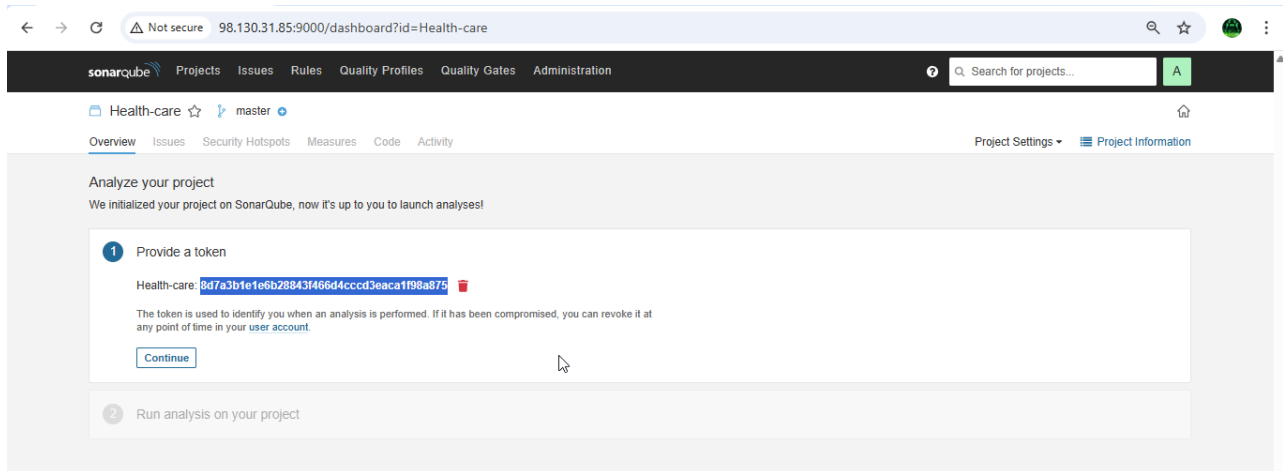
ID ?
dockerhub-creds

Description ?
dockerhub

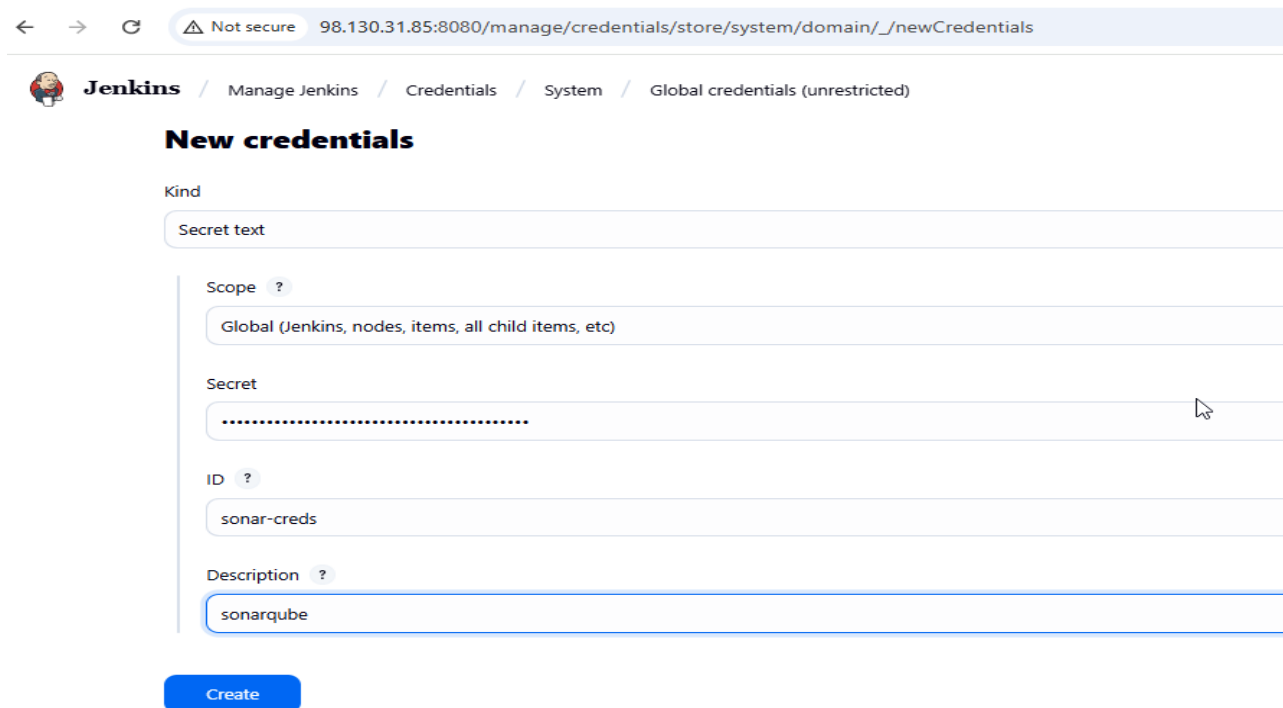
Save

Step28:- Now in jenkins configure the sonarqube and nexus tools

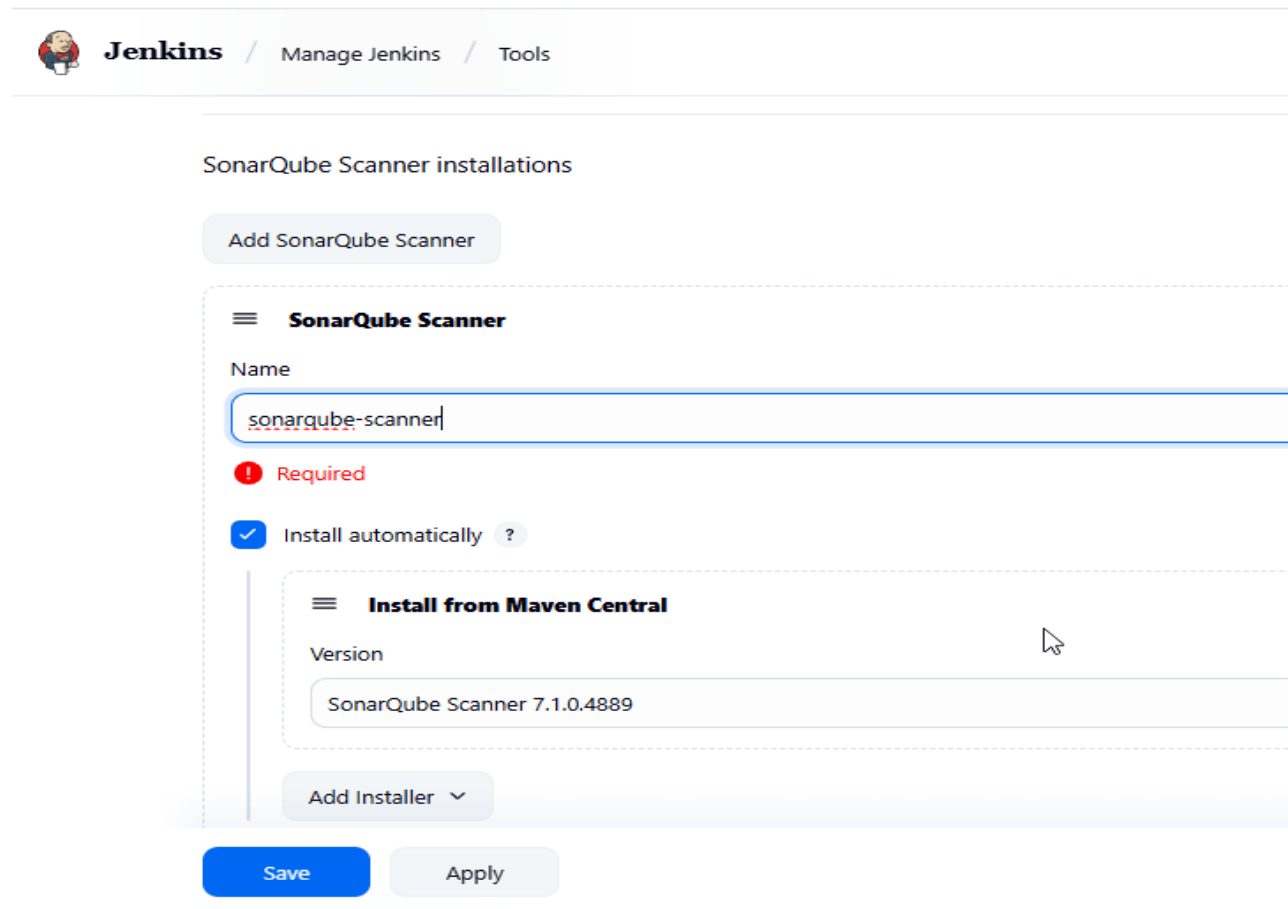
Go to SonarQube ----> Projects ----> Add project ----> Manually ----> Project key: Health-care, Display name: Health-care ----> Setup ----> Generate a token: Health-care ----> Generate ----> Copy the token



Step29:- Lets configure the SonarQube credentials in Jenkins and for that manage-jenkins ==> Credentials ==> global ==> Add credentials==> select secret text and give the text that copied form the sonarqube home page



Step30:- Similarly add sonarqube tool also as
jenkins ----> Manage Jenkins ----> System Configuration ----> Tools ---->
Scroll down to 'sonarqube Scanner Installations' ----> Add SonarQube
scanner ----> Name: sonarscanner, 'Check' Install automatically, Version:
7.1.0.4889 ----> Apply ----> Save.



The screenshot shows the Jenkins 'SonarQube Scanner installations' configuration page. At the top, there's a breadcrumb trail: Jenkins / Manage Jenkins / Tools. Below this, the page title is 'SonarQube Scanner installations'. A button 'Add SonarQube Scanner' is visible. The main configuration area is titled 'SonarQube Scanner'. It has a 'Name' field with the value 'sonarqube-scanner'. Below this, there's a red error message 'Required'. A checkbox 'Install automatically' is checked. Below this, there's a section titled 'Install from Maven Central' with a 'Version' field containing 'SonarQube Scanner 7.1.0.4889'. At the bottom of this section is an 'Add Installer' dropdown. At the very bottom of the page are two buttons: 'Save' and 'Apply'.

Step31:- Now configure the sonarqube in the system of the jenkins and for that jenkins ----> Manage Jenkins ----> System Configuration ----> System ----> Scroll down to 'SonarQube servers' ----> 'Check' environment variables, Add SonarQube ----> Name: sonarqube, Server URL: <SonarQube URL> [only upto 9000] ----> Server authentication token: Select 'sonarqube' from dropdown ----> Apply ----> Save

Jenkins / Manage Jenkins / System

SonarQube installations

List of SonarQube installations

Name

Server URL

Default is <http://localhost:9000>

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

+ Add

Advanced ▾

Save Apply

Step32:- Now configure the nexus in the jenkins for that first
Go to Nexus tab in browser --> Click on settings icon ----> Click on
'repositories' ----> Create repo ----> Scroll down and Click on 'maven2
hosted' ----> Name: Health-care, Version policy: Snapshot, Layout
policy: Strict, Content disposition: Inline, Blob store:default,
Deployment policy: Allow redeploy ----> Create repository ----> You will
see 'Health-care' repo

← → ↺ Not secure 98.130.31.85:8081/#admin/repository/repositories

Sonatype Nexus Repository
COMMUNITY 3.79.0-09

Administration

- Repository
 - Repositories**
 - Blob Stores
 - Data Store
 - Proprietary Repositories
 - Content Selectors
 - Cleanup Policies
 - Routing Rules
- Security
 - Privileges
 - Roles
 - Users
 - Anonymous Access

Repositories / Select Recipe / Create Repository: maven2 (hosted)

Name: A unique identifier for this repository

Online: ☒ If checked, the repository accepts incoming requests

Maven 2

Version policy:
What type of artifacts does this repository store?

Layout policy:
Validate that all paths are maven artifact or metadata paths

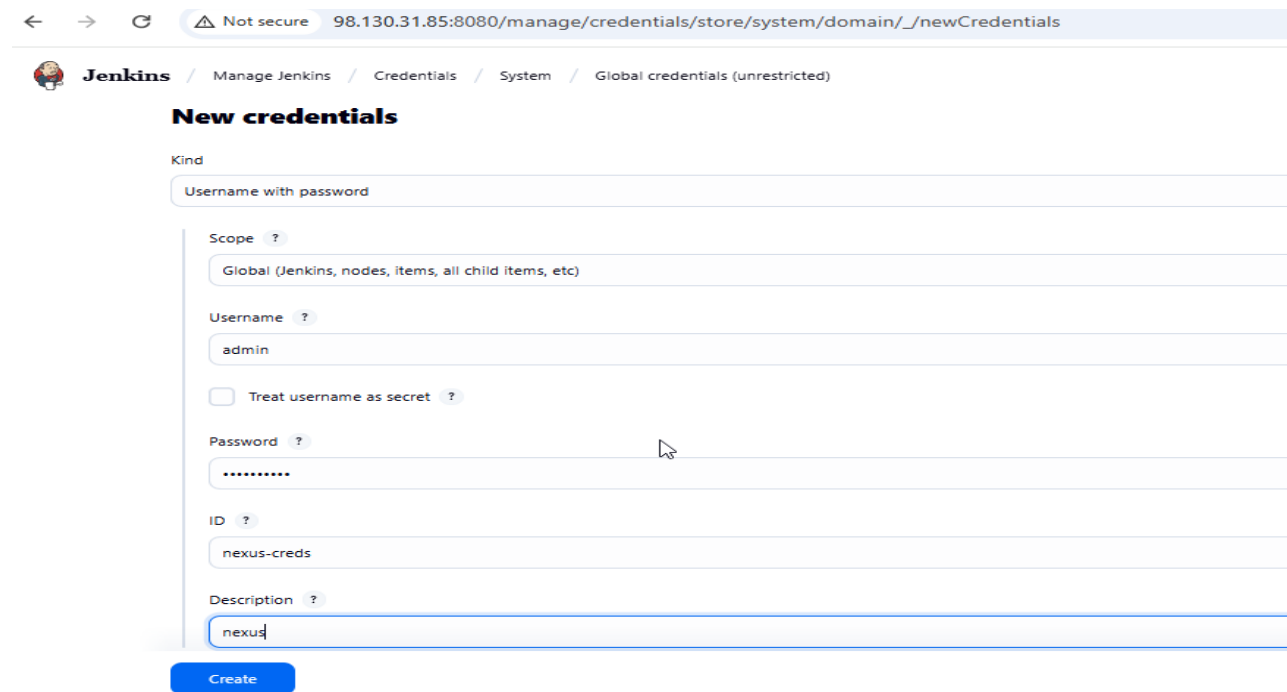
Content Disposition:
Add Content-Disposition header as 'Attachment' to disable some content from being inline in a browser.

Storage

Blob store:
Blob store used to store repository contents

Strict Content Type Validation:

Step33:- Now configure the nexus credentials in the jenkins dash board through Jenkins ----> Manage Jenkins ----> Security ----> Credentials ----> Click on 'global' ----> Add creds ----> Kind: Username with Password, Scope: Global, Username: admin, Password: <Enter The Nexus> , ID: nexus-creds, Description: nexus-creds ----> Create



← → ↺ 98.130.31.85:8080/manage/credentials/store/system/domain/_/newCredentials

Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrestricted)

New credentials

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

admin

☐ Treat username as secret ?

Password ?

.....

ID ?

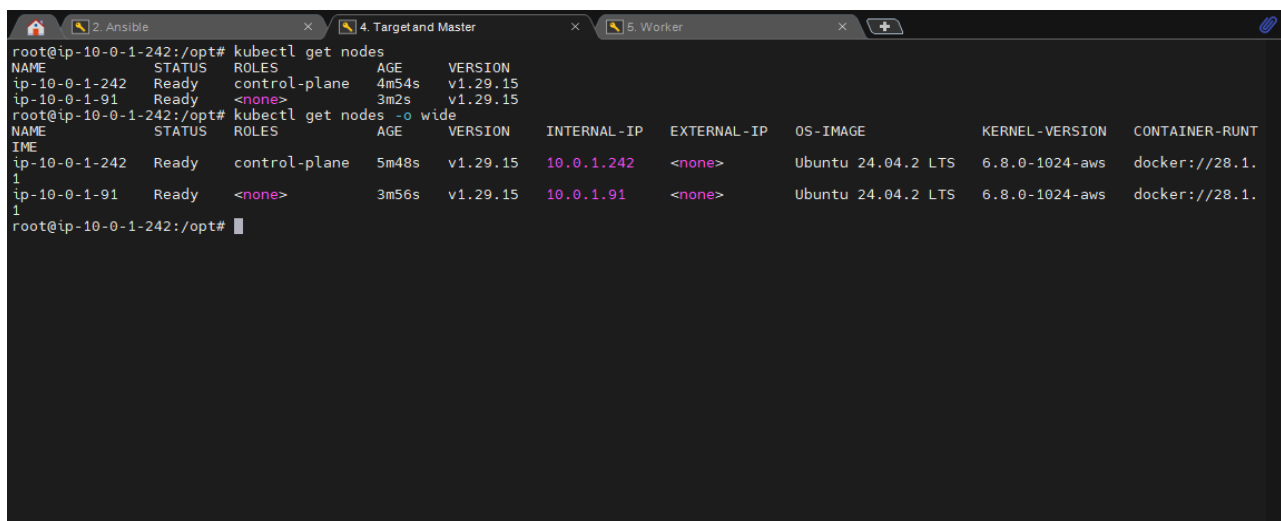
nexus-creds

Description ?

nexus

Create

Step34:- Now go to Target and Master instance and install Kubernetes and configure with Worker node and verify



```
root@ip-10-0-1-242:/opt# kubectl get nodes
NAME                STATUS    ROLES    AGE     VERSION
ip-10-0-1-242       Ready    control-plane  4m54s   v1.29.15
ip-10-0-1-91        Ready    <none>      3m2s    v1.29.15
root@ip-10-0-1-242:/opt# kubectl get nodes -o wide
NAME                STATUS    ROLES    AGE     VERSION    INTERNAL-IP    EXTERNAL-IP    OS-IMAGE             KERNEL-VERSION    CONTAINER-RUNT
ip-10-0-1-242       Ready    control-plane  5m48s   v1.29.15    10.0.1.242     <none>         Ubuntu 24.04.2 LTS   6.8.0-1024-aws    docker://28.1.
ip-10-0-1-91        Ready    <none>      3m56s   v1.29.15    10.0.1.91      <none>         Ubuntu 24.04.2 LTS   6.8.0-1024-aws    docker://28.1.
root@ip-10-0-1-242:/opt#
```

Step35:- Now go to Project repo in the github and there add deployment.yml and jenkinsfile to perform the pipeline in the Jenkins.

Update deployment.yml

```
1  ---
2  apiVersion: apps/v1
3  kind: Deployment
4  metadata:
5    name: health-care-deployment
6    labels:
7      app: health-care
8  spec:
9    replicas: 2
10   selector:
11     matchLabels:
12       app: health-care
13   template:
14     metadata:
15       labels:
16         app: health-care
17     spec:
18       containers:
19         - name: health-care-container
20           image: pj013525/health-care:v1
21           ports:
22             - containerPort: 8082
23           imagePullPolicy: Always
24   ---
25   apiVersion: v1
26   kind: Service
27   metadata:
28     name: health-care-service
29   spec:
30     selector:
31       app: health-care
32     ports:
33       - protocol: TCP
34         port: 80
35         targetPort: 8082
36     type: NodePort
37
```

```
1  pipeline {
2    agent any
3
4    stages {
5      stage('Git checkout') {
6        steps {
7          git branch: 'master', url: 'https://github.com/pj013525/star-agile-project-2.git'
8        }
9      }
10
11      stage('Compilation') {
12        steps {
13          sh 'mvn compile'
14        }
15      }
16
17      stage('Testing') {
18        steps {
19          sh 'mvn test'
20        }
21      }
22
23      stage('SonarQube Scanner') {
24        steps {
25          withSonarQubeEnv('sonarqube-server') {
26            sh 'mvn org.sonarsource.scanner.maven:sonar-maven-plugin:5.0.0.4389:sonar'
27          }
28        }
29      }
30
31      stage('Packing') {
32        steps {
33          sh 'mvn clean package'
34        }
35      }
36    }
37  }
```

```
Code Blame 68 lines (59 loc) · 1.94 KB Code 55% faster with GitHub Copilot
35     }
36
37     stage('Pushing Artifact to Nexus') {
38         steps {
39             nexusArtifactUploader artifacts: [[artifactId: 'medicure', classifier: '', file: 'target/medicure-0.0.1-SNAPSHOT.jar', type: 'jar']], credentialsId: 'nexus-creds', groupId:
40         }
41     }
42
43     stage('Build the Docker Image') {
44         steps {
45             script {
46                 sh 'docker build -t pj013525/health-care:v1 .'
47                 sh 'docker images'
48             }
49         }
50     }
51
52     stage('Push Image to DockerHub') {
53         steps {
54             withCredentials([string(credentialsId: 'dockerhub-details', variable: 'dockerhubPassword')]) {
55                 sh "docker login -u pj013525 -p ${dockerhubPassword}"
56                 sh 'docker push pj013525/health-care:v1'
57             }
58         }
59     }
60     stage('Deploy using k8s') {
61         steps {
62             sh 'sudo kubectl apply -f deployment.yml'
63             sh 'sudo kubectl get all'
64         }
65     }
66 }
67 }
```

Step36:- Now again go back to Jenkins project and click on Build now to check the status of the build and as you can see that the build is successful and a 2 pods are also created in the ec2 Worker node.

← → ↺ ⚠ Not secure 98.130.31.85:8080/job/star-agile-health-care-project/7/pipeline-overview/ 🔍 ☆ ⚙️ 👤

Jenkins / star-agile-health-care-project / #7 / Pipeline Overview 🔍 ⚙️ 👤

Start

Checkout SCM

Git checkout

Compilation

Testing

SonarQube Scanner

Packing

Pushing Artifact to...

Build the Docker Image

Push Image to...

Deploy using k8s

End

0.91 sec

0.69 sec

2.2 sec

9.2 sec

12 sec

17 sec

1.1 sec

2.2 sec

18 sec

0.83 sec

🔍 Search

📄

✓ Deploy using k8s 0.83 sec ⌚ Started 39 sec ago 🗨 Jenkins ⋮

✓ sudo kubectl apply -f deployment.yml 0.53 sec 📄

✓ sudo kubectl get all 0.27 sec 📄

0 + sudo kubectl get all

1	NAME	READY	STATUS	RESTARTS	AGE
2	pod/health-care-deployment-67789b7f98-bcp7t	1/1	Running	0	22m
3	pod/health-care-deployment-67789b7f98-tzcrw	1/1	Running	0	22m
4	pod/health-care-deployment-6f5878b76-gptdw	0/1	ContainerCreating	0	1s

5

6	NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
7	service/health-care-service	NodePort	10.100.65.188	<none>	80:32287/TCP	22m
8	service/kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	51m

9

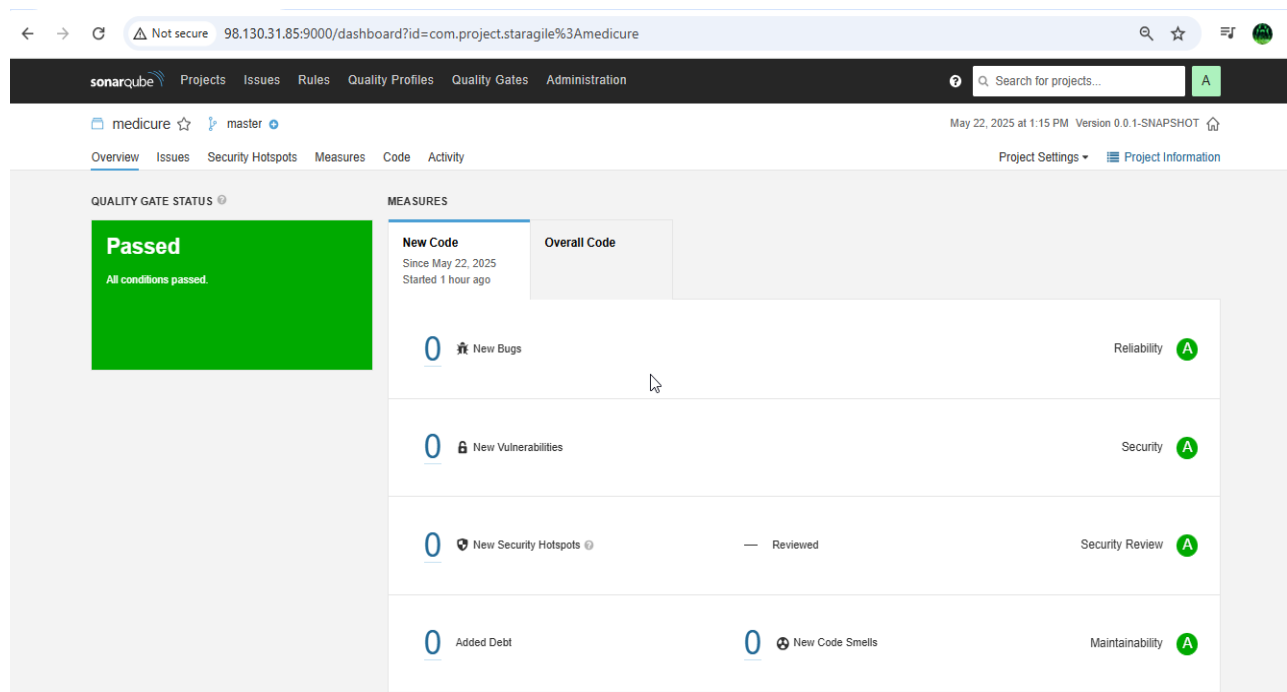
10	NAME	READY	UP-TO-DATE	AVAILABLE	AGE
11	deployment.apps/health-care-deployment	2/2	1	2	22m

12

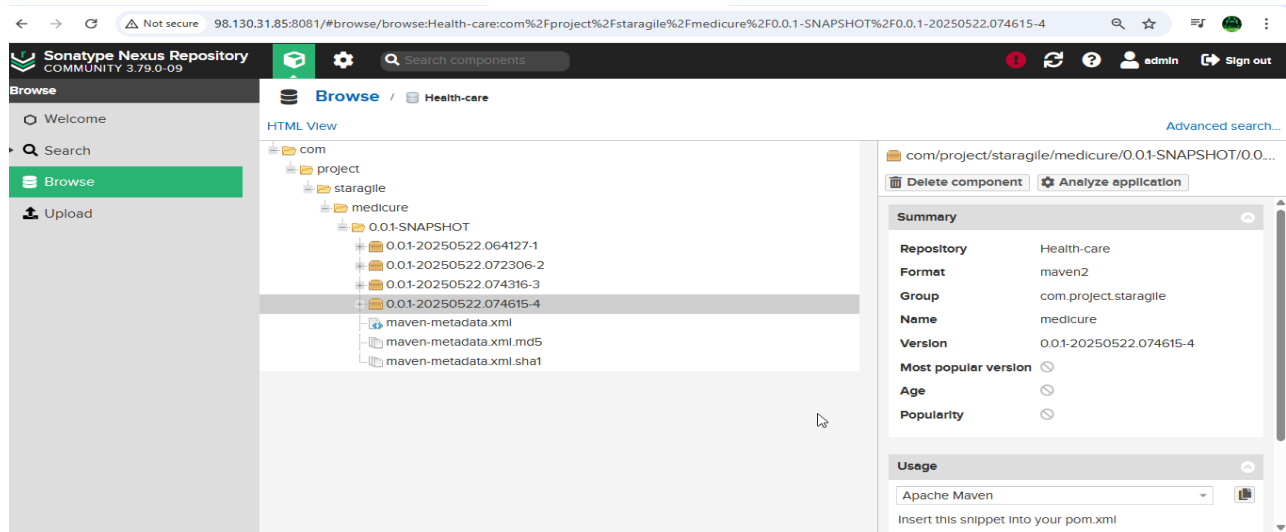
13 NAME READY RESTARTS AGE

```
root@ip-10-0-1-242:/home/ubuntu# kubectl get pods -o wide
NAME                                READY    STATUS    RESTARTS   AGE    IP              NODE             NOMINATED NODE    READINESS GATE
health-care-deployment-6f5878b76-gptdw 1/1      Running   0           10m    192.168.206.195 ip-10-0-1-91     <none>             <none>
health-care-deployment-6f5878b76-nsgpt 1/1      Running   0           10m    192.168.206.196 ip-10-0-1-91     <none>             <none>
root@ip-10-0-1-242:/home/ubuntu# kubectl get svc
NAME                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
health-care-service NodePort    10.100.65.188 <none>         80:32207/TCP     35m
kubernetes           ClusterIP   10.96.0.1     <none>         443/TCP          64m
root@ip-10-0-1-242:/home/ubuntu#
```

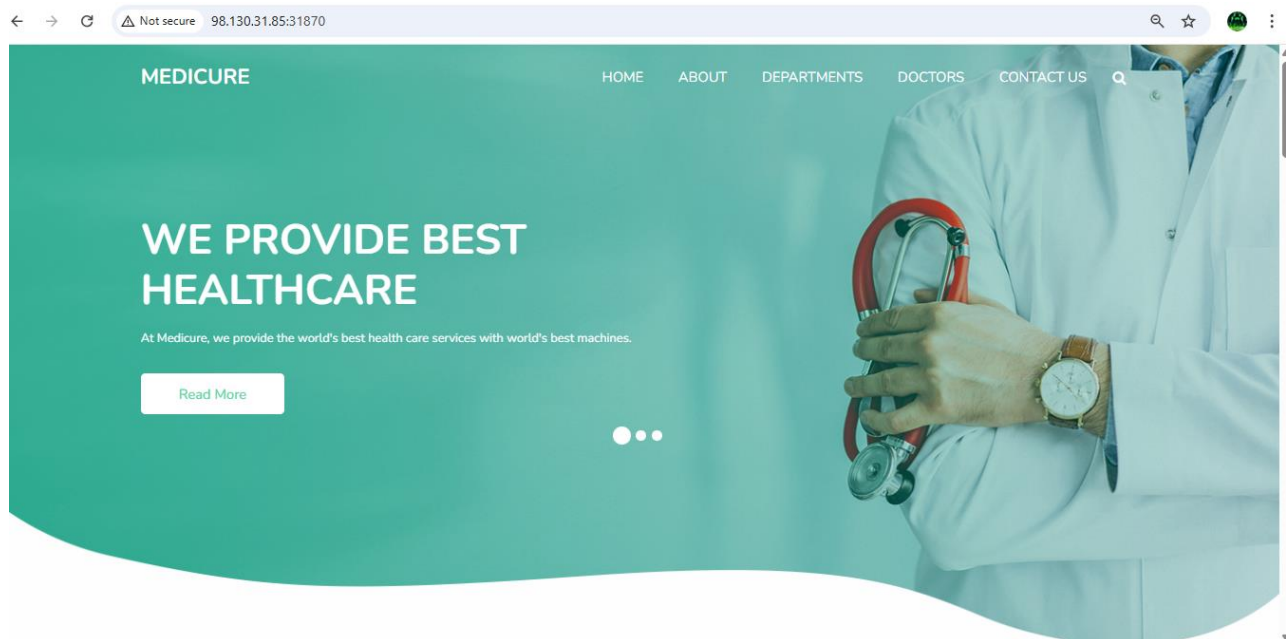
Step37:- Now go to sonarqube page and check the the project details



Step38:- Now check the test results if it is passed then the code is correct and now check the Artifact that created in the build in Nexus



Step39:- Now go to any browser and give the Workernode-IPaddress:31870 (since port is auto allocated) and click enter the you will see the home page of the project and thus the project deployment is successful using kubernetes.



Step40:- Now monitor the pods using Prometheus and Grafana , for that install Prometheus in Jenkins-Docker server and Grafana in another server

```

root@ip-10-0-1-205:/home/ubuntu# wget https://dl.grafana.com/enterprise/release/grafana-enterprise-8.4.4.linux-amd64.tar.gz
ls
--2025-05-17 12:28:16-- https://dl.grafana.com/enterprise/release/grafana-enterprise-8.4.4.linux-amd64.tar.gz
Resolving dl.grafana.com (dl.grafana.com)... 151.101.38.217, 2a04:4e42:9::729
Connecting to dl.grafana.com (dl.grafana.com)|151.101.38.217|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 84007981 (80M) [application/x-tar]
Saving to: 'grafana-enterprise-8.4.4.linux-amd64.tar.gz'

grafana-enterprise-8.4.4.linux-amd 100%[=====] 80.12M 3.07MB/s in 10s

2025-05-17 12:28:30 (7.85 MB/s) - 'grafana-enterprise-8.4.4.linux-amd64.tar.gz' saved [84007981/84007981]

grafana-enterprise-8.4.4.linux-amd64.tar.gz
root@ip-10-0-1-205:/home/ubuntu#

```

WhatsApp

```

ntent-disposition=attachment%3B%20filename%3Dprometheus-2.34.0.linux-amd64.tar.gz&response-content-type=application%
wing]
--2025-05-17 12:30:52-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/6838921/408c8e
a51d5db857X-Amz-Algorithm=AWS4-HMAC-SHA256X-Amz-Credential=releaseassetproduction%2F20250517%2Fus-east-1%2Fs3%2Faws
20250517T123052Z6X-Amz-Expires=3006X-Amz-Signature=234a94e8a223abb35f896d6f6ba36edaff9e13c7b447a16285fd0d3871c427ff8
ost&response-content-disposition=attachment%3B%20filename%3Dprometheus-2.34.0.linux-amd64.tar.gz&response-content-ty
t-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 76299772 (73M) [application/octet-stream]
Saving to: 'prometheus-2.34.0.linux-amd64.tar.gz'

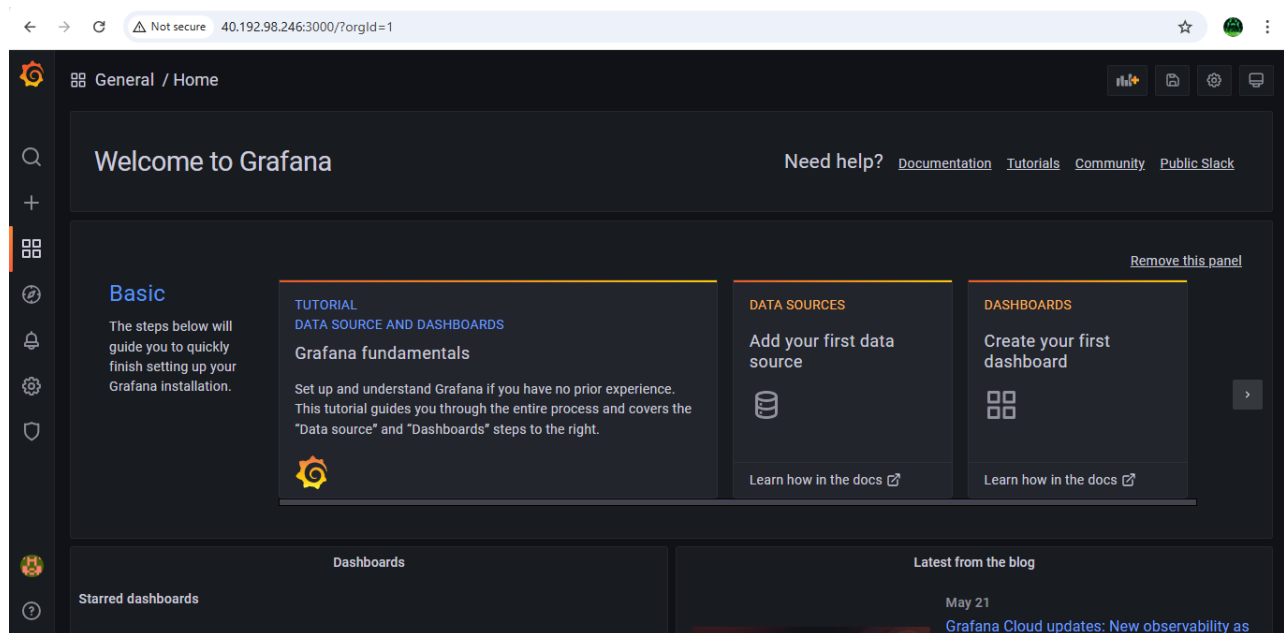
prometheus-2.34.0.linux-amd64.tar. 100%[=====] 72.76M 9

2025-05-17 12:31:00 (11.0 MB/s) - 'prometheus-2.34.0.linux-amd64.tar.gz' saved [76299772/76299772]

root@ip-10-0-1-230:/home/ubuntu# ls
jenkins.sh jenkins.sh.1 prometheus-2.34.0.linux-amd64.tar.gz
root@ip-10-0-1-230:/home/ubuntu# tar zxvf prometheus-2.34.0.linux-amd64.tar.gz
prometheus-2.34.0.linux-amd64/
prometheus-2.34.0.linux-amd64/consoles/
prometheus-2.34.0.linux-amd64/consoles/index.html.example
prometheus-2.34.0.linux-amd64/consoles/node-cpu.html
prometheus-2.34.0.linux-amd64/consoles/node-disk.html
prometheus-2.34.0.linux-amd64/consoles/node-overview.html
prometheus-2.34.0.linux-amd64/consoles/node.html
prometheus-2.34.0.linux-amd64/consoles/prometheus-overview.html
prometheus-2.34.0.linux-amd64/consoles/prometheus.html
prometheus-2.34.0.linux-amd64/console_libraries/
prometheus-2.34.0.linux-amd64/console_libraries/menu.lib
prometheus-2.34.0.linux-amd64/console_libraries/prom.lib
prometheus-2.34.0.linux-amd64/prometheus.yml
prometheus-2.34.0.linux-amd64/LICENSE
prometheus-2.34.0.linux-amd64/NOTICE
prometheus-2.34.0.linux-amd64/prometheus
prometheus-2.34.0.linux-amd64/promtool
root@ip-10-0-1-230:/home/ubuntu#

```


Step41:- Now install grafana in the Grafana server and after successful installation of Grafana, now go to browser and give grafana server ip-address:3000 (3000 is default port number for grafana) and **use admin and admin as username and password** as they are default and login to the grafana home page.



Step42:- Now setup the docker and Prometheus in another using by telling docker that Prometheus would track docker on port 9323

i.e., vi /etc/docker/daemon.json press

I to insert

```
{  
    "metrics-addr" : "0.0.0.0:9323",  
    "experimental" : true
```

} then save and exit and restart the docker

```
root@ip-10-0-1-242:/home/ubuntu/prometheus-2.34.0.linux-amd64# cat /etc/docker/daemon.json
{
  "metrics-addr" : "0.0.0.0:9323",
  "experimental" : true
}

root@ip-10-0-1-242:/home/ubuntu/prometheus-2.34.0.linux-amd64#
```

Step43:- Now go to any browser and give **docker ip-address:9323/metrics** and in the below image you will see that the docker stats have been started successfully

```
← → ↺ Not Secure http://98.130.31.85:9323/metrics
Import bookmarks... Finish

engine_daemon_container_actions_seconds_bucket{action="start",le="1"} 24
engine_daemon_container_actions_seconds_bucket{action="start",le="2.5"} 24
engine_daemon_container_actions_seconds_bucket{action="start",le="5"} 24
engine_daemon_container_actions_seconds_bucket{action="start",le="10"} 24
engine_daemon_container_actions_seconds_bucket{action="start",le="+Inf"} 24
engine_daemon_container_actions_seconds_sum{action="start"} 4.6220752229999995
engine_daemon_container_actions_seconds_count{action="start"} 24
# HELP engine_daemon_container_states_containers The count of containers in various states
# TYPE engine_daemon_container_states_containers gauge
engine_daemon_container_states_containers{state="paused"} 0
engine_daemon_container_states_containers{state="running"} 20
engine_daemon_container_states_containers{state="stopped"} 0
# HELP engine_daemon_engine_cpus_cpus The number of cpus that the host system of the engine has
# TYPE engine_daemon_engine_cpus_cpus gauge
engine_daemon_engine_cpus_cpus 2
# HELP engine_daemon_engine_info The information related to the engine and the OS it is running on
# TYPE engine_daemon_engine_info gauge
engine_daemon_engine_info{architecture="x86_64",commit="01f442b",daemon_id="0a77bb76-d7ca-4bff-9330-75ed782f507e",graphdriver="overlay2",kernel="6.8.0-1024-aws",os="Ubuntu 24.04.2 LTS",os_type="linux",os_version="24.04",version="28.1.1"} 1
# HELP engine_daemon_engine_memory_bytes The number of bytes of memory that the host system of the engine has
# TYPE engine_daemon_engine_memory_bytes gauge
engine_daemon_engine_memory_bytes 8.200802304e+09
# HELP engine_daemon_events_subscribers_total The number of current subscribers to events
# TYPE engine_daemon_events_subscribers_total gauge
engine_daemon_events_subscribers_total 0
# HELP engine_daemon_events_total The number of events logged
# TYPE engine_daemon_events_total counter
engine_daemon_events_total 287
# HELP engine_daemon_health_check_start_duration_seconds The number of seconds it takes to prepare to run health checks
# TYPE engine_daemon_health_check_start_duration_seconds histogram
engine_daemon_health_check_start_duration_seconds_bucket{le="0.005"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="0.01"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="0.025"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="0.05"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="0.1"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="0.25"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="0.5"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="1"} 0
engine_daemon_health_check_start_duration_seconds_bucket{le="2.5"} 0
```

Step44:- Now add docker job in the Prometheus.yml file to give this stats to Prometheus vi prometheus.yml

- job_name: "docker"

metrics_path defaults to '/metrics' # scheme defaults to 'http'.
static_configs:

- targets: ["localhost:9323"]

Save the file and exit and start the Prometheus using `./prometheus`

```
# The job name is added as a label job=<job_name> to any timeseries
- job_name: "prometheus"

# metrics_path defaults to '/metrics'
# scheme defaults to 'http'.

static_configs:
  - targets: ["localhost:9090"]

- job_name: "docker"

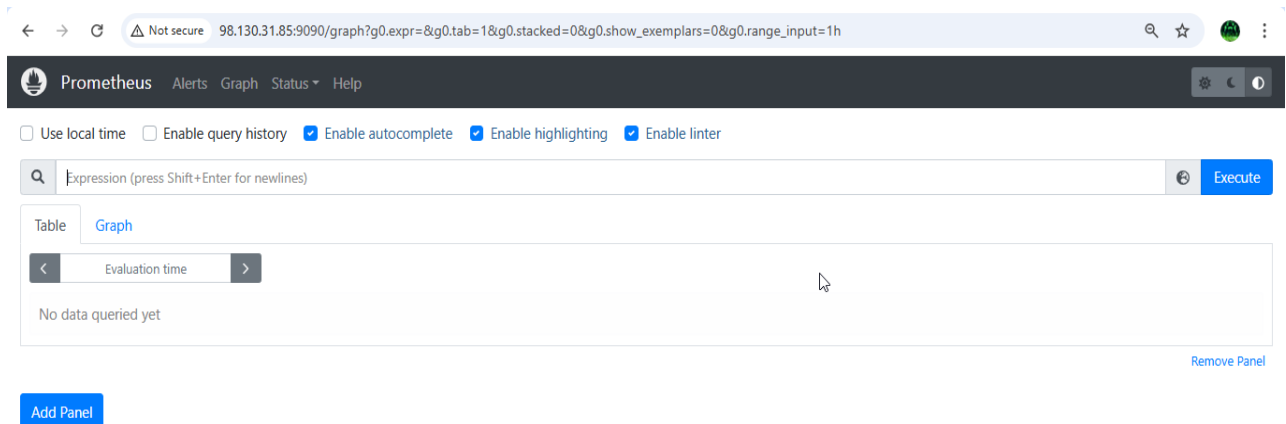
# metrics_path defaults to '/metrics'
# scheme defaults to 'http'.

static_configs:
  - targets: ["localhost:9323"]

prometheus.yml" 38L, 1098B
```

As you can see that the Prometheus have been started from the above image

Step45:- Now go browser and give docker ip:9090 and enter , then you will be successfully enter into the Prometheus homepage



Step46:- Now click on status ⑦ targets then you will see the status of the of the docker and prometheus.

← → ↻ Not secure 98.130.31.85:9090/targets

Prometheus Alerts Graph Status ▾ Help

Targets

All Unhealthy Collapse All 🔍 Filter by endpoint or labels

docker (1/1 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9323/metrics	UP	instance="localhost:9323" job="docker"	1.664s ago	4.081ms	

prometheus (1/1 up) [show less](#)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	7.817s ago	4.181ms	

Step47:- Now go to grafana homepage ⑦ configurations ⑦ Data sources

← → ↻ Not secure 40.192.98.246:3000/?orgId=1 ☆

General / Home

Welcome to Grafana


Need help? [Documentation](#) [Tutorials](#) [Community](#) [Public Slack](#)


Basic
The steps below will guide you to quickly get up and running with your Grafana installation.

Configuration

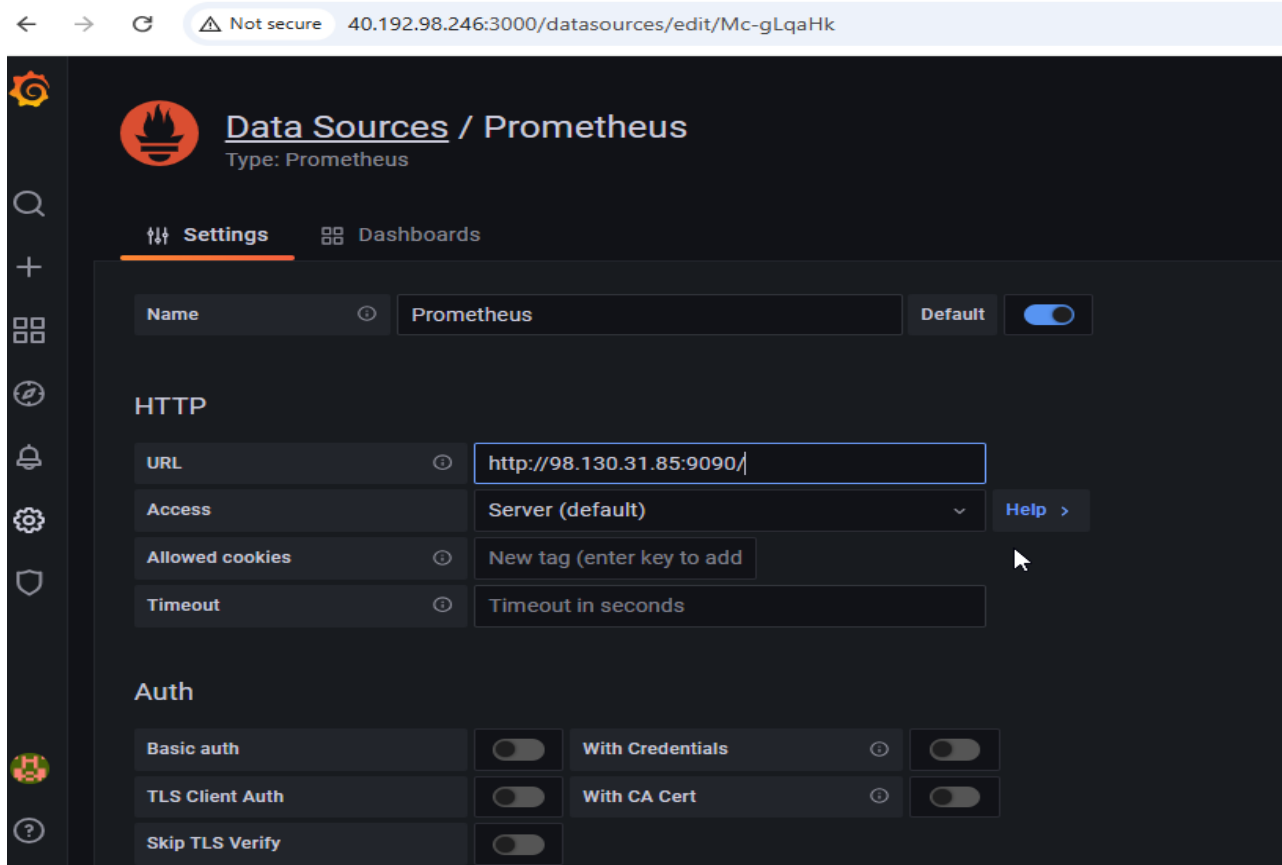
- Data sources
- Users
- Teams
- Plugins
- Preferences

TUTORIAL
DATA SOURCE AND DASHBOARDS
Grafana fundamentals
Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.

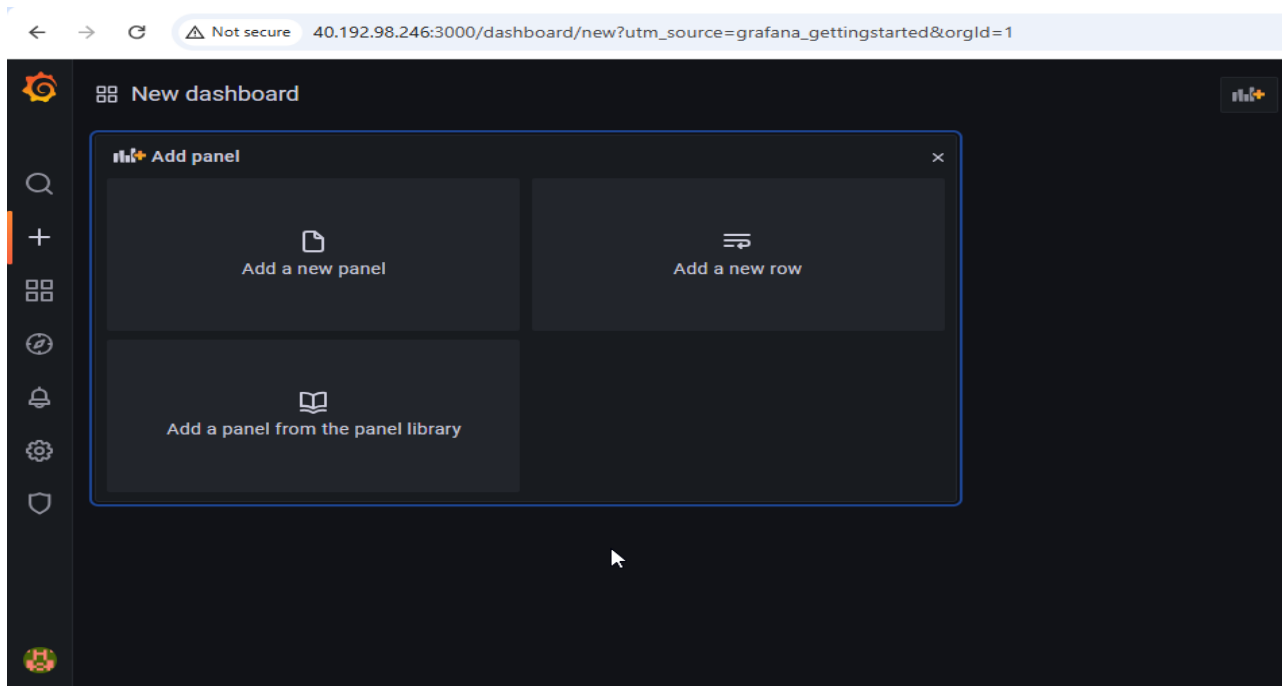
DATA SOURCES
Add your first data source

[Learn how in the docs](#)

DASHBOARDS
Create your first dashboard

[Learn how in the docs](#)

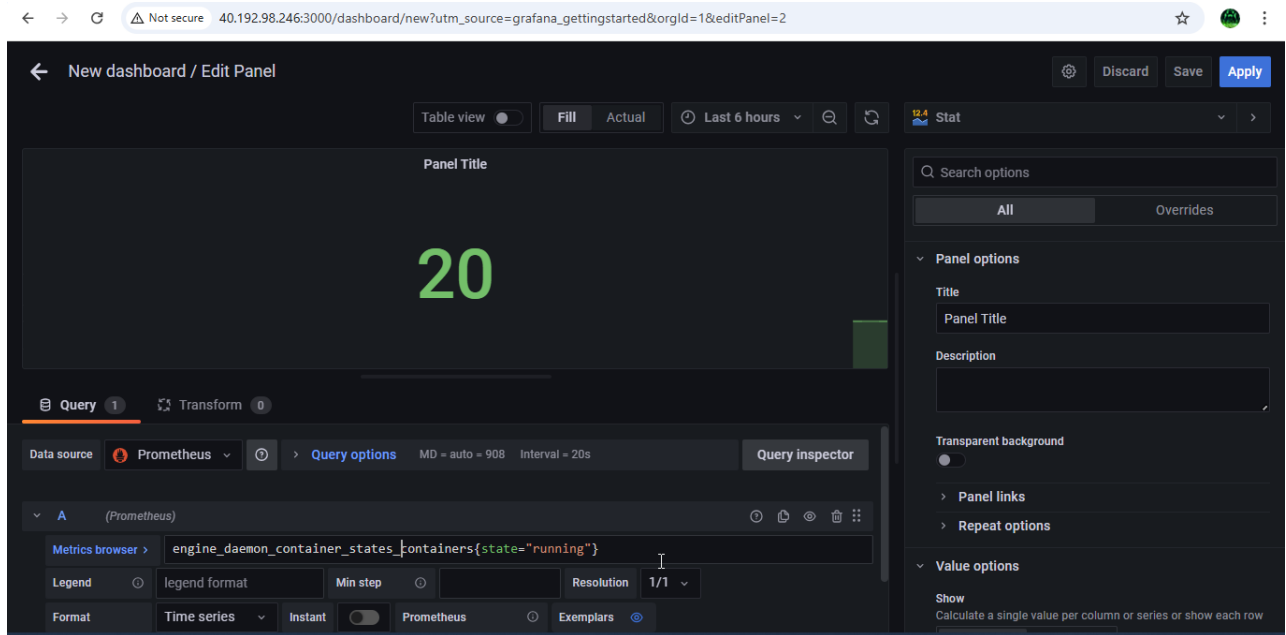
Step48:- Now click on add Data sources ⑦ Prometheus and give ipaddress:9090 and click on save and test



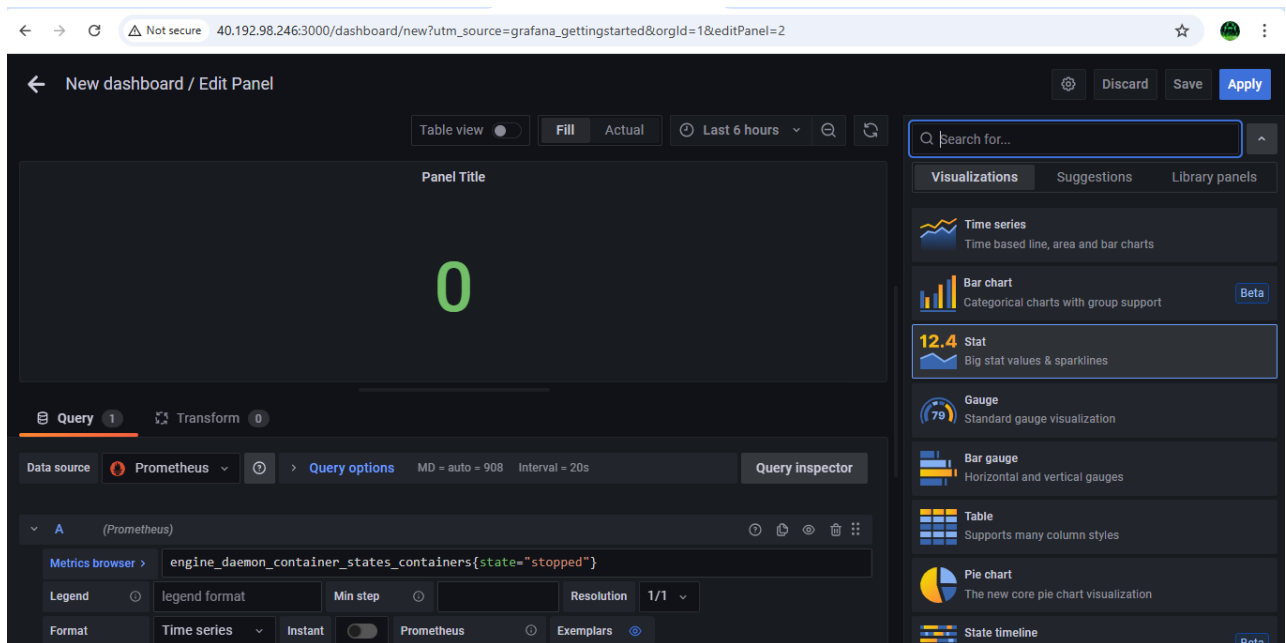
Step49:- Now click on Dash board ⑦ add new panel



Step50:- Now in the metrics browser give engine daemon container states containers{state="running"} and you will see the result that same as in the metrics from the browser



engine_daemon_container_states_containers{state="stopped"}



Step51:- The values shown in the panel must be equal to the that of shown in the docker stats, here the container which we created is in exited state so it is showing as stopped state in stats

```
# HELP engine_daemon_container_states_containers The count of containers in various states
# TYPE engine_daemon_container_states_containers gauge
engine_daemon_container_states_containers{state="paused"} 0
engine_daemon_container_states_containers{state="running"} 20
engine_daemon_container_states_containers{state="stopped"} 0
# HELP engine_daemon_engine_cpus_cpus The number of cpus that the host system has if the engine has
```

Step52:- Now go and check the containers running and in stopped state again and check the details again in the stats

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
91f602e0f4e0	75392e3500e3	"start_runit"	20 minutes ago	Up 20 minutes		k8s_calico-node_calico-node-c
qnbkf_kube-system	42036f07-3a4e-40ec-a1f7-66ef84ccee96_2	"/etc..."	20 minutes ago	Up 20 minutes		k8s_coredns_coredns-76f75df57
ede9e73e8764	cbb01a7bd410	"/coredns -conf /etc..."	20 minutes ago	Up 20 minutes		k8s_coredns_coredns-76f75df57
4-4j5rwr_kube-system	97b8dd3f-d081-421f-b8d9-befe66d0287a_2	"/coredns -conf /etc..."	20 minutes ago	Up 20 minutes		k8s_POD_coredns-76f75df574-ms
c38609c7b1b2	cbb01a7bd410	"/pause"	20 minutes ago	Up 20 minutes		k8s_POD_coredns-76f75df574-4j
4-ms7lj_kube-system	51f5b119-3bdd-4934-a757-2523fceedd47_2	"/pause"	20 minutes ago	Up 20 minutes		k8s_calico-kube-controllers_c
7c24457a16c5	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		k8s_POD_calico-kube-controller
7lj_kube-system	51f5b119-3bdd-4934-a757-2523fceedd47_12	"/pause"	20 minutes ago	Up 20 minutes		k8s_kube-scheduler_kube-sched
0ead39a58da8	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		k8s_POD_kube-scheduler-ip-10-
5rwr_kube-system	97b8dd3f-d081-421f-b8d9-befe66d0287a_10	"/usr/bin/kube-contr..."	20 minutes ago	Up 20 minutes		k8s_kube-apiserver_kube-apise
49c64d9c04e8	f9c3c1813269	"/bin/node_exporter ..."	20 minutes ago	Up 20 minutes		k8s_node-exporter_prometheus-
24abb70777e5	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		k8s_kube-proxy_kube-proxy-rzk
rs-9d57d8f49-2bdnj_kube-system	693f5004-d190-4242-97b0-8562f49b7787_9	"kube-scheduler --au..."	20 minutes ago	Up 20 minutes		k8s_kube-controller-manager_k
aa4590fbd454	9ea0bd82ed4f	"/pause"	20 minutes ago	Up 20 minutes		k8s_etcd_etcd-ip-10-0-1-242_k
uler-ip-10-0-1-242_kube-system	d04a4ee1c17a254de8e4627f63052e36_2	"/pause"	20 minutes ago	Up 20 minutes		k8s_POD_calico-node-cqnb_kub
af4deaf0a653	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		k8s_POD_kube-proxy-rzkwd_kube
0-1-242_kube-system	d04a4ee1c17a254de8e4627f63052e36_2	"/bin/node_exporter ..."	20 minutes ago	Up 20 minutes		k8s_POD_kube-apiserver-ip-10-
5f2b8c2c419f	f44c6889a2d2	"kube-apiserver --ad..."	20 minutes ago	Up 20 minutes		k8s_POD_kube-controller-manag
79d41470649d	255ec253085f	"/bin/node_exporter ..."	20 minutes ago	Up 20 minutes		
prometheus-node-exporter-stf79_default	9b40ea04-45bd-48fd-9189-88363a2ec373_2	"/usr/local/bin/kube..."	20 minutes ago	Up 20 minutes		
8ff932a49856	f71614796eb7	"/pause"	20 minutes ago	Up 20 minutes		
wd_kube-system	9a75c0a7-73d8-4733-a724-5296bb5dae86_2	"kube-controller-man..."	20 minutes ago	Up 20 minutes		
74a2a8abe308	b0cdcf76ac8e	"/pause"	20 minutes ago	Up 20 minutes		
ube-controller-manager-ip-10-0-1-242_kube-system	e746a0c1e0e6fc16ddf7524c7c76f522_2	"/etc..."	20 minutes ago	Up 20 minutes		
f308a697c493	a9e7e6b294ba	"etcd --advertise-cl..."	20 minutes ago	Up 20 minutes		
ube-system	63c7572858f6cbb1e69e585a771c2b8_2	"/pause"	20 minutes ago	Up 20 minutes		
ab3d7d5df1ec	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		
e-system	42036f07-3a4e-40ec-a1f7-66ef84ccee96_2	"/pause"	20 minutes ago	Up 20 minutes		
1a59792f79a1	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		
-system	9a75c0a7-73d8-4733-a724-5296bb5dae86_2	"/pause"	20 minutes ago	Up 20 minutes		
3fe530e885d0	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		
0-1-242_kube-system	56d40a089d2264f5e98d93627e0b4d60_2	"/pause"	20 minutes ago	Up 20 minutes		
55f165720145	registry.k8s.io/pause:3.6	"/pause"	20 minutes ago	Up 20 minutes		

Step53:- This is how we monitor the health of a container automatically and visualizing the report using Prometheus and Grafana.