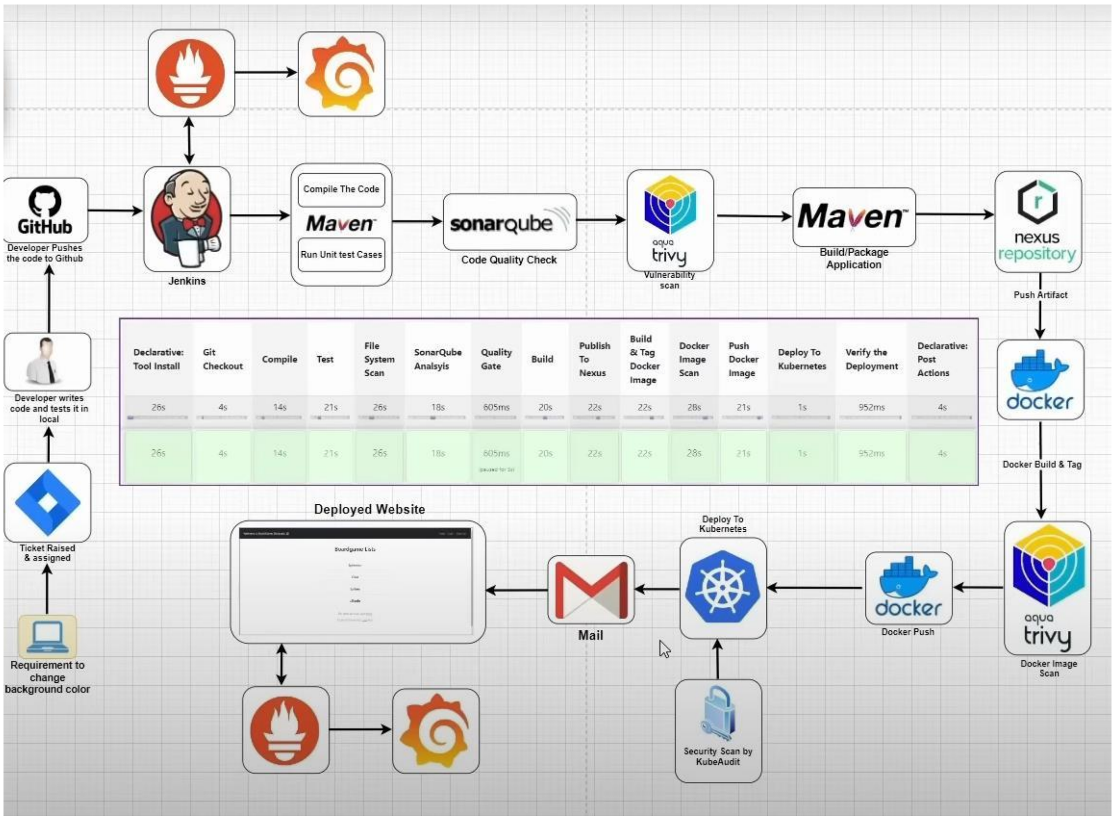
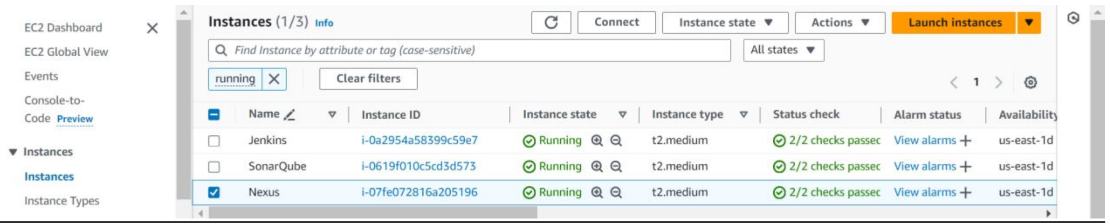
CI/CD-PROJECT



Declarative: Tool Install	Git Checkout	Compile	Test	File System Scan	SonarQube Analsyis	Quality Gate	Build	Publish To Nexus	Build & Tag Docker Image	Docker Image Scan	Push Docker Image	Deploy To Kubernetes	Verify the Deployment	Declarative: Post Actions
26s	4s	14s	21s	26s	18s	605ms	20s	22s	22s	28s	21s	1s	952ms	4s
26s	4s	14s	21s	26s	18s	605ms (paused for 3s)	20s	22s	22s	28s	21s	1s	952ms	4s

Phase-1:

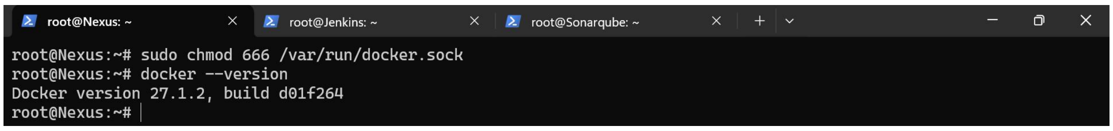
Create 3 EC2 Instances with 30GB RAM and choose t2.medium



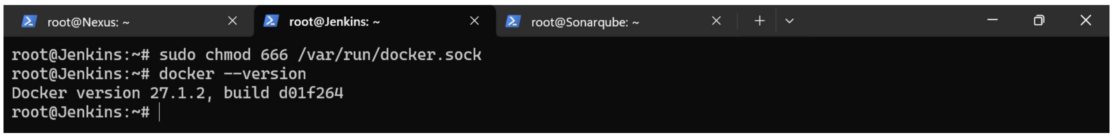
Instances (1/3) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
☐	Jenkins	i-0a2954a58399c59e7	Running	t2.medium	2/2 checks passec	View alarms +	us-east-1d
☐	SonarQube	i-0619f010c5cd3d573	Running	t2.medium	2/2 checks passec	View alarms +	us-east-1d
☑	Nexus	i-07fe072816a205196	Running	t2.medium	2/2 checks passec	View alarms +	us-east-1d

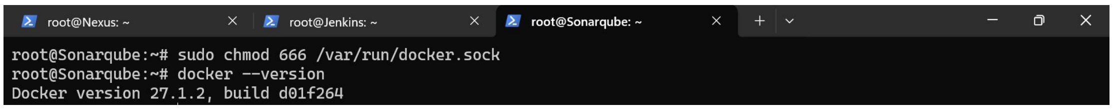
Install Docker on All 3 VMs :



```
root@Nexus:~# sudo chmod 666 /var/run/docker.sock
root@Nexus:~# docker --version
Docker version 27.1.2, build d01f264
root@Nexus:~#
```

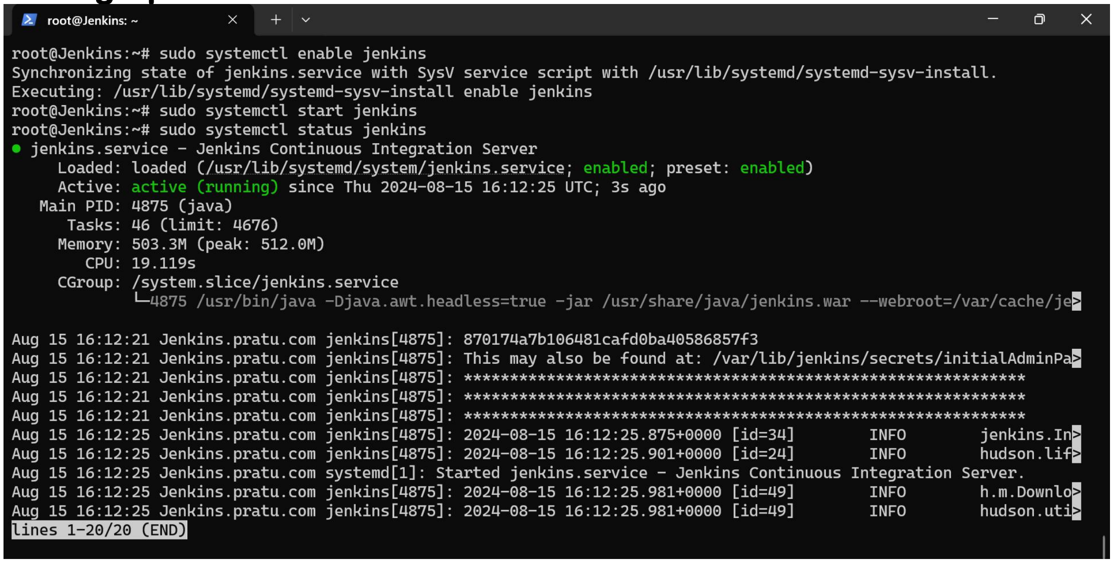


```
root@Jenkins:~# sudo chmod 666 /var/run/docker.sock
root@Jenkins:~# docker --version
Docker version 27.1.2, build d01f264
root@Jenkins:~#
```



```
root@Sonarqube:~# sudo chmod 666 /var/run/docker.sock
root@Sonarqube:~# docker --version
Docker version 27.1.2, build d01f264
```

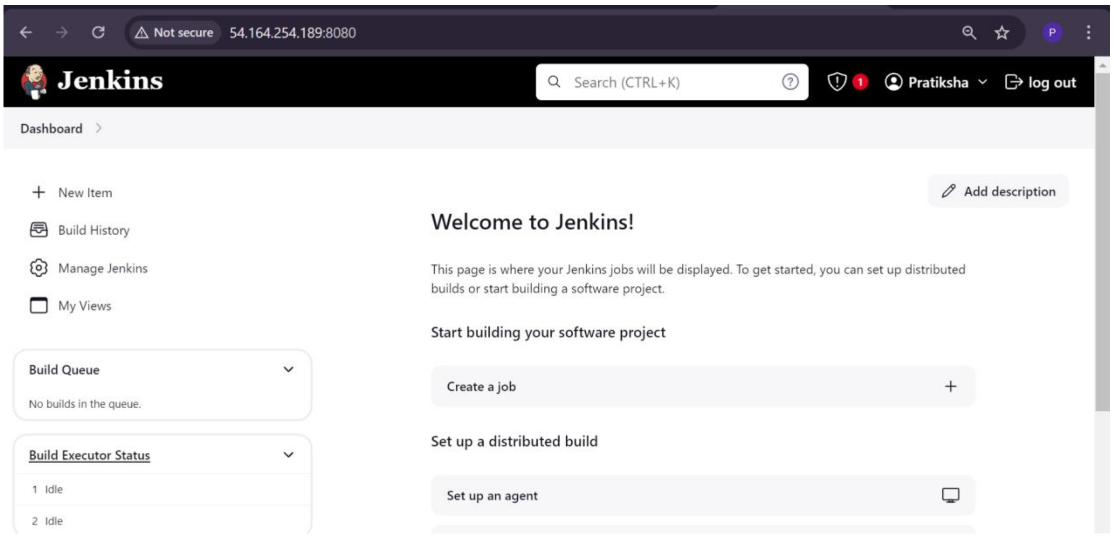
Setting Up Jenkins on Ubuntu :



```
root@Jenkins:~# sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
root@Jenkins:~# sudo systemctl start jenkins
root@Jenkins:~# sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Thu 2024-08-15 16:12:25 UTC; 3s ago
   Main PID: 4875 (java)
      Tasks: 46 (limit: 4676)
     Memory: 503.3M (peak: 512.0M)
        CPU: 19.119s
     CGroup: /system.slice/jenkins.service
             └─4875 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/je>

Aug 15 16:12:21 Jenkins.pratu.com jenkins[4875]: 870174a7b106481cafd0ba40586857f3
Aug 15 16:12:21 Jenkins.pratu.com jenkins[4875]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPa>
Aug 15 16:12:21 Jenkins.pratu.com jenkins[4875]: *************************************************************
Aug 15 16:12:21 Jenkins.pratu.com jenkins[4875]: *************************************************************
Aug 15 16:12:21 Jenkins.pratu.com jenkins[4875]: *************************************************************
Aug 15 16:12:25 Jenkins.pratu.com jenkins[4875]: 2024-08-15 16:12:25.875+0000 [id=34]        INFO        jenkins.In>
Aug 15 16:12:25 Jenkins.pratu.com jenkins[4875]: 2024-08-15 16:12:25.901+0000 [id=24]        INFO        hudson.lif>
Aug 15 16:12:25 Jenkins.pratu.com systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Aug 15 16:12:25 Jenkins.pratu.com jenkins[4875]: 2024-08-15 16:12:25.981+0000 [id=49]        INFO        h.m.Downlo>
Aug 15 16:12:25 Jenkins.pratu.com jenkins[4875]: 2024-08-15 16:12:25.981+0000 [id=49]        INFO        hudson.uti>
lines 1-20/20 (END)
```

Access Jenkins:

- Open a web browser and go to http://your_server_ip_or_domain:8080.
- You will see a page asking for the initial admin password. Retrieve it using:
- sudo cat /var/lib/jenkins/secrets/initialAdminPassword
- Enter the password, install suggested plugins, and create your first admin user.



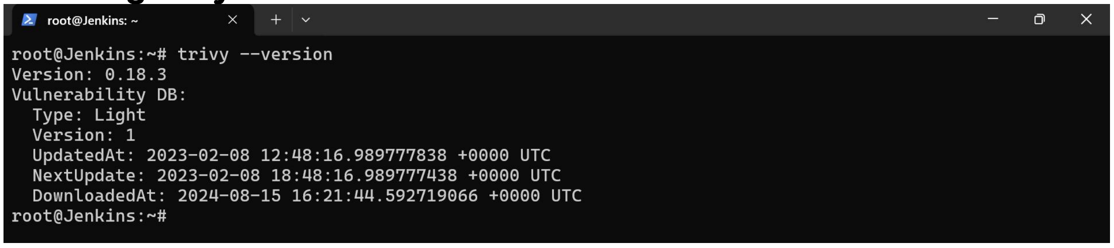
Welcome to Jenkins!

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

Start building your software project

Set up a distributed build

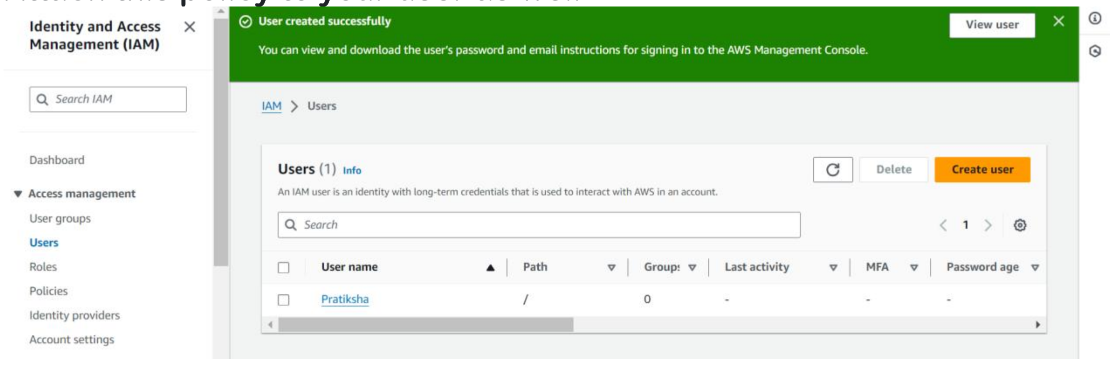
Installing Trivy on Jenkins Server:



```
root@Jenkins:~# trivy --version
Version: 0.18.3
Vulnerability DB:
  Type: Light
  Version: 1
  UpdatedAt: 2023-02-08 12:48:16.989777838 +0000 UTC
  NextUpdate: 2023-02-08 18:48:16.989777438 +0000 UTC
  DownloadedAt: 2024-08-15 16:21:44.592719066 +0000 UTC
root@Jenkins:~#
```

EKS-Setup:
First Create a user in AWS IAM with any name
Attach Policies to the newly created user
below policies
AmazonEC2FullAccess
AmazonEKS_CNI_Policy
AmazonEKSClusterPolicy
AmazonEKSWorkerNodePolicy
AWSCloudFormationFullAccess
IAMFullAccess

One more policy we need to create with content as below
```
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "VisualEditor0",
"Effect": "Allow",
"Action": "eks:*",
"Resource": "*"
}
]
}
```
Attach this policy to your user as well



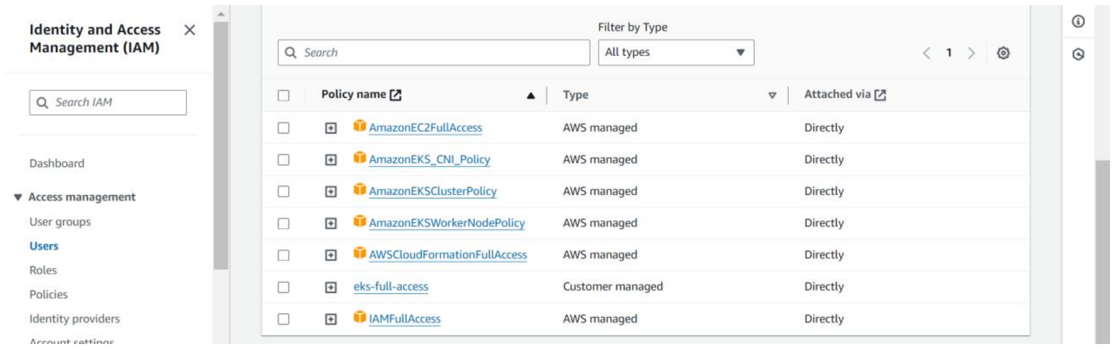
User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Users (1) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age
Pratiksha	/	0	-	-	-



Policy name	Type	Attached via
AmazonEC2FullAccess	AWS managed	Directly
AmazonEKS_CNI_Policy	AWS managed	Directly
AmazonEKSClusterPolicy	AWS managed	Directly
AmazonEKSWorkerNodePolicy	AWS managed	Directly
AWSCloudFormationFullAccess	AWS managed	Directly
eks-full-access	Customer managed	Directly
IAMFullAccess	AWS managed	Directly

AWSCLI :

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
sudo apt install unzip
unzip awscliv2.zip
sudo ./aws/install
aws configure
```

KUBECTL:

```
curl -o kubectl https://amazon-eks.s3.us-west-
2.amazonaws.com/1.19.6/2021-
01-05/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin
kubectl version --short --client
```

EKSCTL:

```
curl --silent --location
"https://github.com/weaveworks/eksctl/releases/latest/download/eksctl
_$(una
me -s)_amd64.tar.gz" | tar xz -C /tmp
sudo mv /tmp/eksctl /usr/local/bin
eksctl version
```

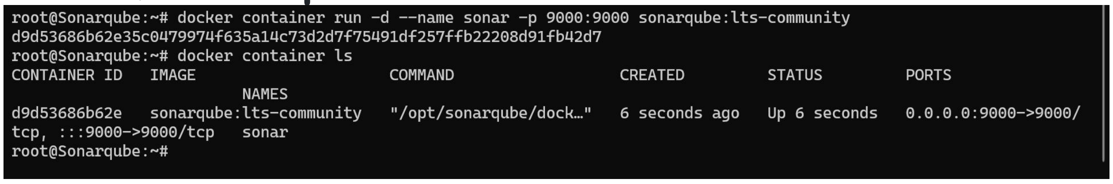
Create EKS CLUSTER:

```
eksctl create cluster --name=my-eks7 \
--region=ap-south-1 \
--zones= ap-south-1a, ap-south-1b \
--version=1.30 \
--without-nodegroupeksctl utils associate-iam-oidc-provider \
--region us-east-1 \
--cluster my-eks2 \
--approve
eksctl create nodegroup --cluster=my-eks7 \
--region= ap-south-1\
--name=node2 \
--node-type=t3.medium \
--nodes=3 \
--nodes-min=2 \
--nodes-max=4 \
--node-volume-size=20 \
--ssh-access \
--ssh-public-key=panduaws \
--managed \
--asg-access \
--external-dns-access \
--full-ecr-access \
--appmesh-access \
--alb-ingress-access
```

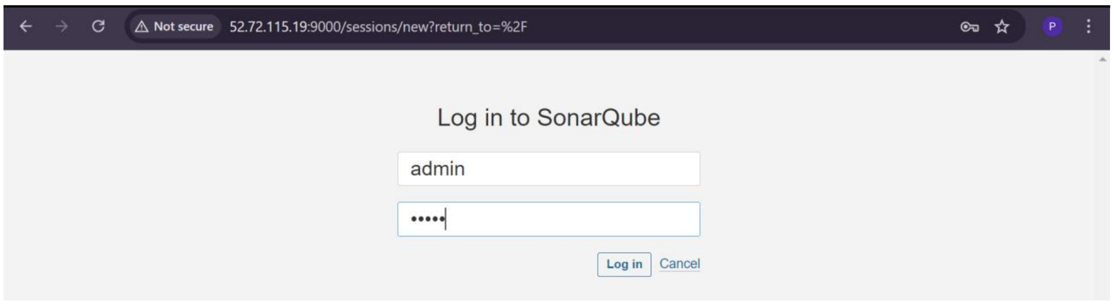
Note: --ssh-public-key=panduaws → Give pem file name in AWS

• **Open INBOUND TRAFFIC IN ADDITIONAL Security Group**

SonarQube Setup:



```
root@Sonarqube:~# docker container run -d --name sonar -p 9000:9000 sonarqube:lts-community
d9d53686b62e35c0479974f635a14c73d2d7f75491df257ffb22208d91fb42d7
root@Sonarqube:~# docker container ls
CONTAINER ID   IMAGE                    COMMAND                  CREATED         STATUS         PORTS
                 NAMES
d9d53686b62e   sonarqube:lts-community   "/opt/sonarqube/dock…"   6 seconds ago   Up 6 seconds   0.0.0.0:9000->9000/
tcp, :::9000->9000/tcp   sonar
root@Sonarqube:~#
```



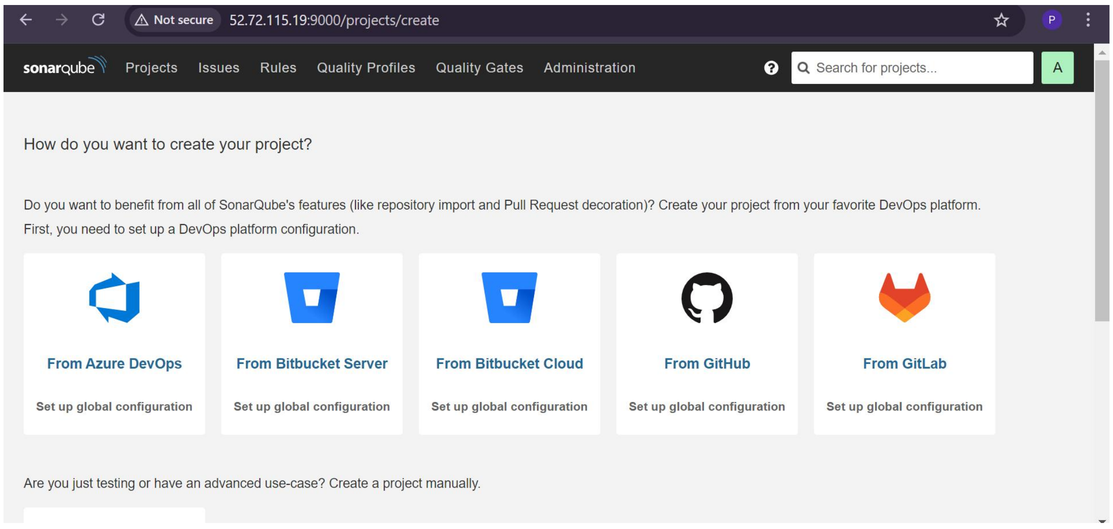
Not secure 52.72.115.19:9000/sessions/new?return_to=%2F

Log in to SonarQube

admin

•••••

Log in Cancel



Not secure 52.72.115.19:9000/projects/create

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration Search for projects...

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform.

First, you need to set up a DevOps platform configuration.

From Azure DevOps — Set up global configuration

From Bitbucket Server — Set up global configuration

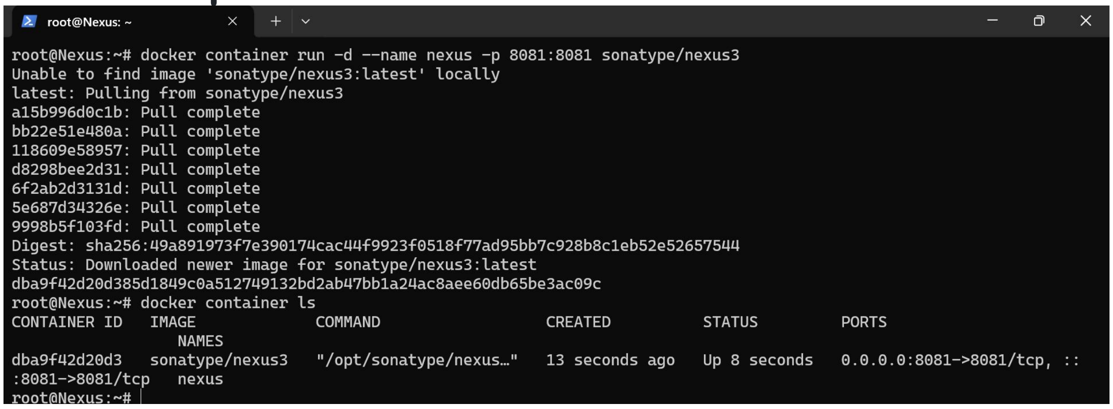
From Bitbucket Cloud — Set up global configuration

From GitHub — Set up global configuration

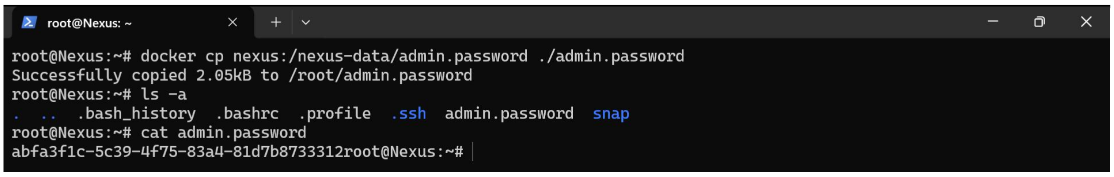
From GitLab — Set up global configuration

Are you just testing or have an advanced use-case? Create a project manually.

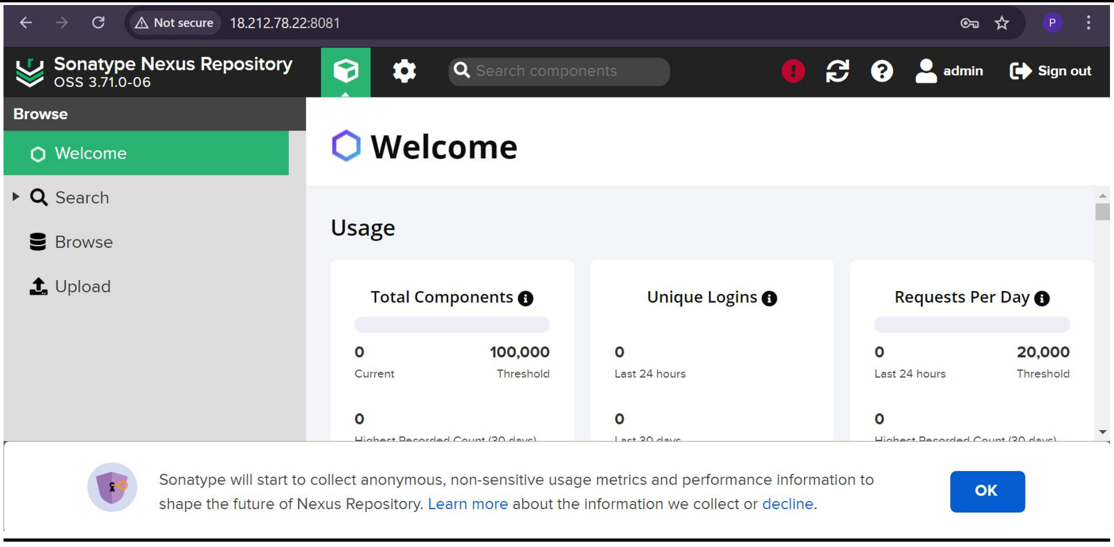
Nexus Setup:



```
root@Nexus:~# docker container run -d --name nexus -p 8081:8081 sonatype/nexus3
Unable to find image 'sonatype/nexus3:latest' locally
latest: Pulling from sonatype/nexus3
a15b996d0c1b: Pull complete
bb22e51e480a: Pull complete
118609e58957: Pull complete
d8298bee2d31: Pull complete
6f2ab2d3131d: Pull complete
5e687d34326e: Pull complete
9998b5f103fd: Pull complete
Digest: sha256:49a891973f7e390174cac44f9923f0518f77ad95bb7c928b8c1eb52e52657544
Status: Downloaded newer image for sonatype/nexus3:latest
dba9f42d20d385d1849c0a512749132bd2ab47bb1a24ac8aee60db65be3ac09c
root@Nexus:~# docker container ls
CONTAINER ID   IMAGE             COMMAND                  CREATED          STATUS         PORTS
                 NAMES
dba9f42d20d3   sonatype/nexus3   "/opt/sonatype/nexus…"   13 seconds ago   Up 8 seconds   0.0.0.0:8081->8081/tcp, ::
:8081->8081/tcp   nexus
root@Nexus:~#
```



```
root@Nexus:~# docker cp nexus:/nexus-data/admin.password ./admin.password
Successfully copied 2.05kB to /root/admin.password
root@Nexus:~# ls -a
.  ..  .bash_history  .bashrc  .profile  .ssh  admin.password  snap
root@Nexus:~# cat admin.password
abfa3f1c-5c39-4f75-83a4-81d7b8733312root@Nexus:~#
```



Sonatype will start to collect anonymous, non-sensitive usage metrics and performance information to shape the future of Nexus Repository. Learn more about the information we collect or decline.

Phase-2:

Close the repository and create your own repository and push those into your github Repository

1. **clone the repo**:

git clone https://github.com/pratikshaa-01/CI-CD-Project.git

2. change the remote repo

git remote set-url origin https://github.com/pratikshaa-01/CI-CD-Project.git

→ replace with your github repo

git remote add new-origin https://github.com/pratikshaa-01/CI-CD-Project.git

→replace with your github repo

3. Initialize Git Repository

git init

4. Add Files to Git:

Stage all files for the first commit:

git add .

5. Commit Files:

Commit the staged files with a commit message: git commit -m "Initial commit"

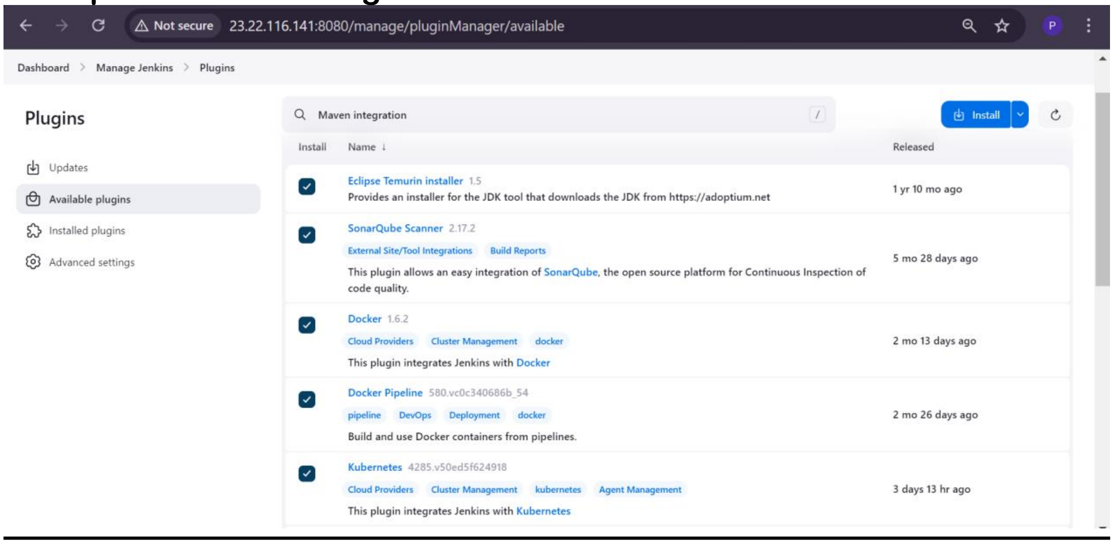
6. Push to GitHub:

Push the local repository to GitHub:

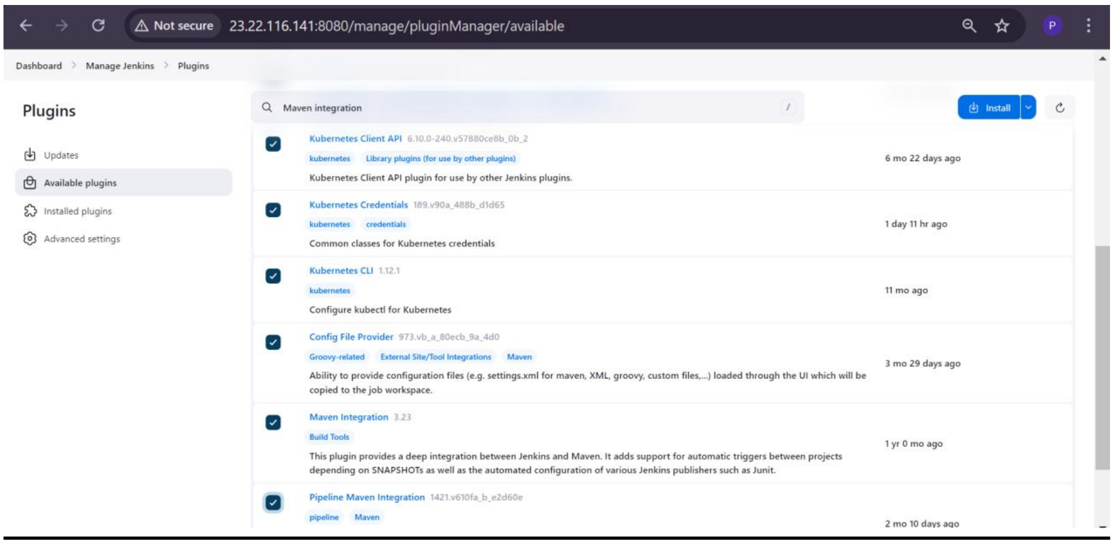
git push -u origin main

Install Plugins in Jenkins :

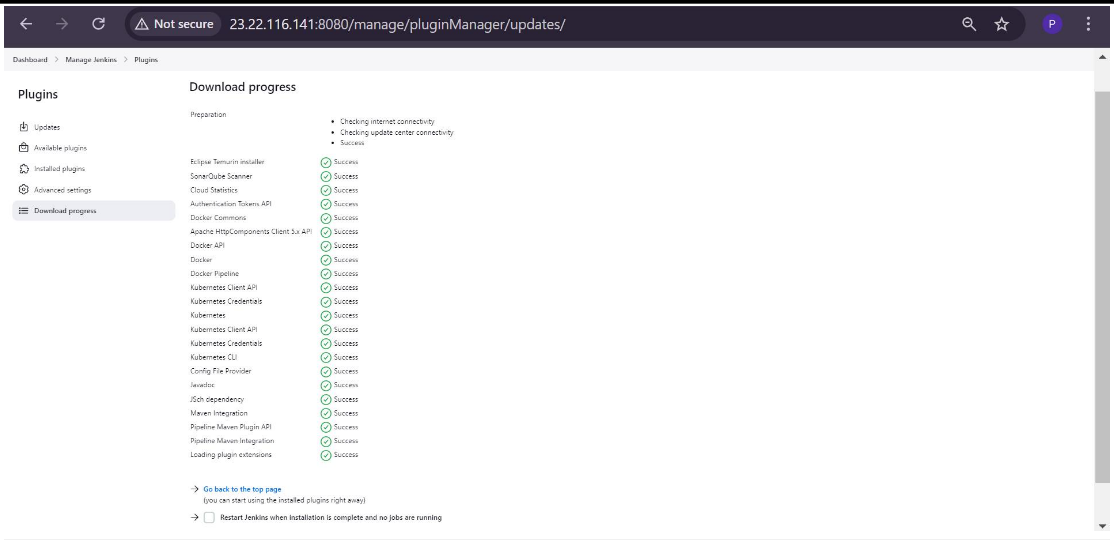
1. Eclipse Temurin installer → for jdk
2. Sonarqube scanner
3. Docker
4. Docker pipeline
5. Kubernetes
6. Kubernetes cli
7. Kubernetes credentials
8. Kubernetes clint api
9. Config file provider → for Nexus
10. Maven integration
11. Pipeline maven integration



Install	Name ↓	Released
✓	Eclipse Temurin installer 1.5 Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net	1 yr 10 mo ago
✓	SonarQube Scanner 2.17.2 External Site/Tool Integrations · Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	5 mo 28 days ago
✓	Docker 1.6.2 Cloud Providers · Cluster Management · docker This plugin integrates Jenkins with Docker	2 mo 13 days ago
✓	Docker Pipeline 580.vc0c340686b_54 pipeline · DevOps · Deployment · docker Build and use Docker containers from pipelines.	2 mo 26 days ago
✓	Kubernetes 4285.v50ed5f624918 Cloud Providers · Cluster Management · kubernetes · Agent Management This plugin integrates Jenkins with Kubernetes	3 days 13 hr ago



✓	Kubernetes Client API 6.10.0-240.v57880ce8b_0b_2 kubernetes · Library plugins (for use by other plugins) Kubernetes Client API plugin for use by other Jenkins plugins.	6 mo 22 days ago
✓	Kubernetes Credentials 189.v90a_488b_d1d65 kubernetes · credentials Common classes for Kubernetes credentials	1 day 11 hr ago
✓	Kubernetes CLI 1.12.1 kubernetes Configure kubectl for Kubernetes	11 mo ago
✓	Config File Provider 973.vb_a_80ecb_9a_4d0 Groovy-related · External Site/Tool Integrations · Maven Ability to provide configuration files (e.g. settings.xml for maven, XML, groovy, custom files,...) loaded through the UI which will be copied to the job workspace.	3 mo 29 days ago
✓	Maven Integration 3.23 Build Tools This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of various Jenkins publishers such as Junit.	1 yr 0 mo ago
✓	Pipeline Maven Integration 1421.v610fa_b_e2d60e pipeline · Maven	2 mo 10 days ago



23.22.116.141:8080/manage/pluginManager/updates/

Dashboard > Manage Jenkins > Plugins

Download progress

```
Preparation
                                  • Checking internet connectivity
                                  • Checking update center connectivity
                                  • Success
Eclipse Temurin installer         ✓ Success
SonarQube Scanner                 ✓ Success
Cloud Statistics                  ✓ Success
Authentication Tokens API         ✓ Success
Docker Commons                    ✓ Success
Apache HttpComponents Client 5.x API ✓ Success
Docker API                        ✓ Success
Docker                            ✓ Success
Docker Pipeline                   ✓ Success
Kubernetes Client API             ✓ Success
Kubernetes Credentials            ✓ Success
Kubernetes                        ✓ Success
Kubernetes Client API             ✓ Success
Kubernetes Credentials            ✓ Success
Kubernetes CLI                    ✓ Success
Config File Provider              ✓ Success
Javadoc                           ✓ Success
JSch dependency                   ✓ Success
Maven Integration                 ✓ Success
Pipeline Maven Plugin API         ✓ Success
Pipeline Maven Integration        ✓ Success
Loading plugin extensions         ✓ Success
```

→ Go back to the top page
(you can start using the installed plugins right away)

→ ☐ Restart Jenkins when installation is complete and no jobs are running

Now we installed the tools and Now we need to configure them

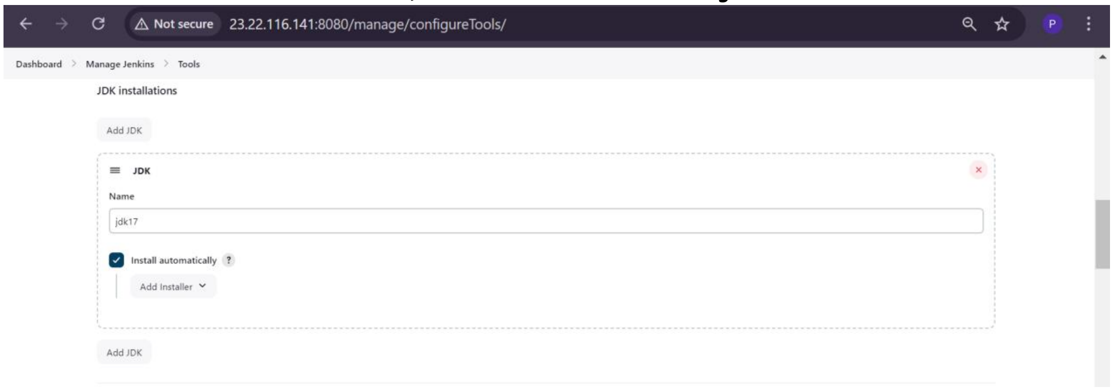
Go to → manage Jenkins→Tools→

1. Jdk→ name= jdk17 , install automatically from adoptium.net, version= jdk17 latest

2. Sonarqube scanner → name=sonar-scanner, Install automatically

3. Maven → name= maven3, version= 3.6.3

4. Docker→ name=docker, install automatically from docker.com



23.22.116.141:8080/manage/configureTools/

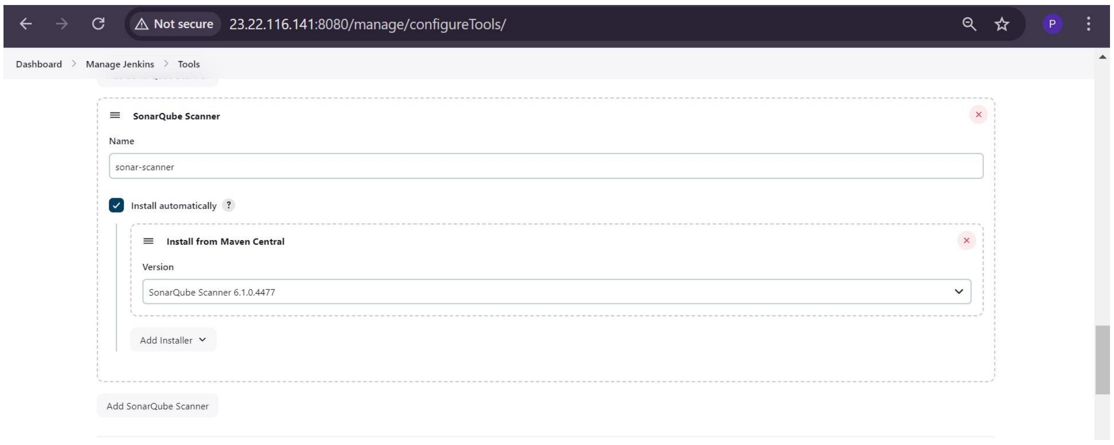
Dashboard > Manage Jenkins > Tools

JDK installations

JDK

Name: jdk17

☑ Install automatically



23.22.116.141:8080/manage/configureTools/

Dashboard > Manage Jenkins > Tools

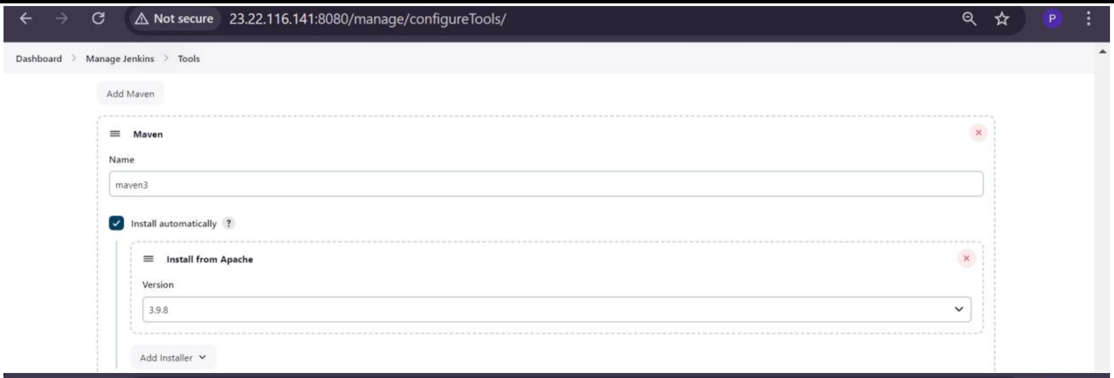
SonarQube Scanner

Name: sonar-scanner

☑ Install automatically

Install from Maven Central

Version: SonarQube Scanner 6.1.0.4477


23.22.116.141:8080/manage/configureTools/

Dashboard > Manage Jenkins > Tools

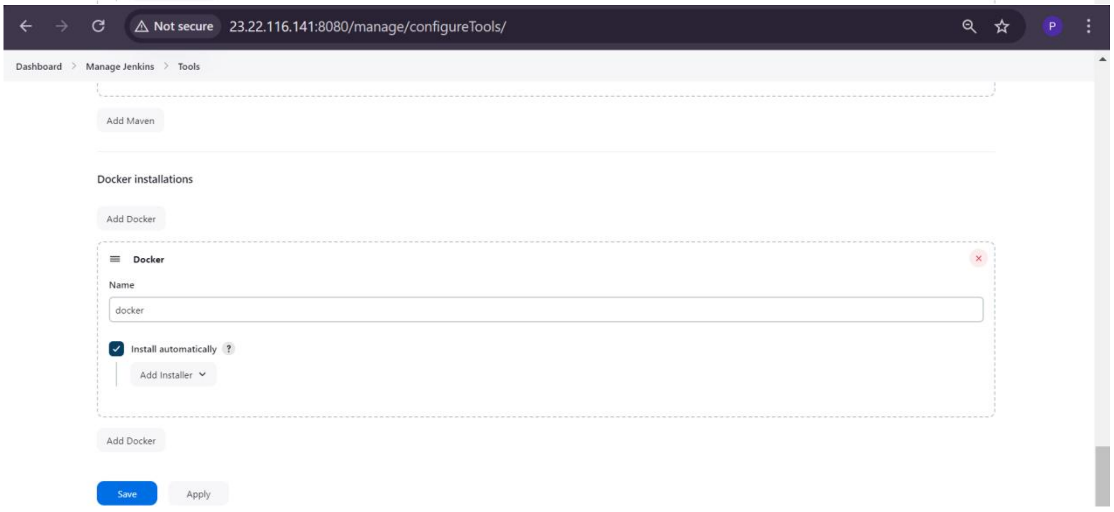
Maven

Name: maven3

Install automatically ✓

Install from Apache

Version: 3.9.8


23.22.116.141:8080/manage/configureTools/

Docker installations

Docker

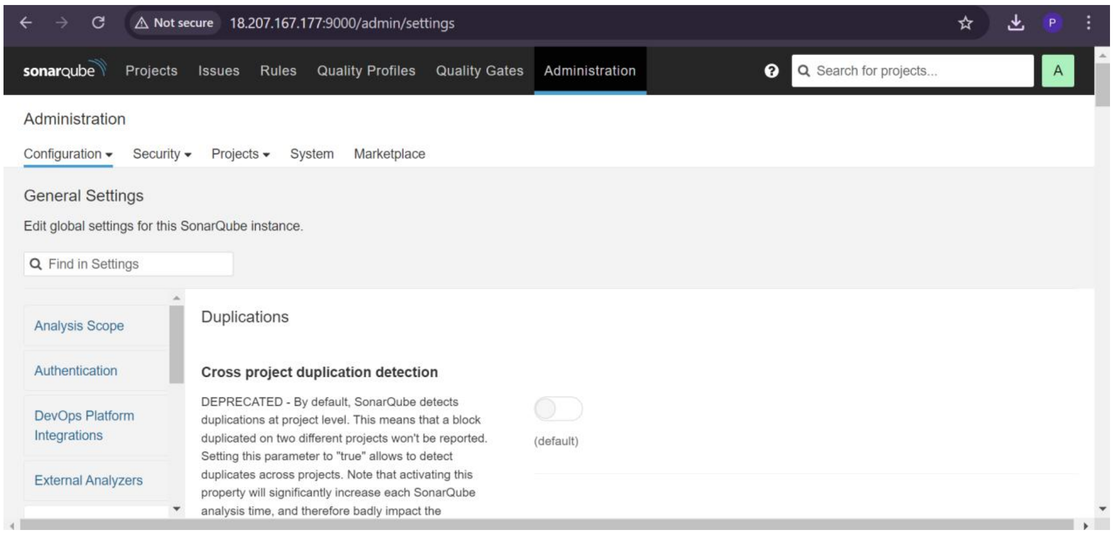
Name: docker

Install automatically ✓

Now configure the sonarqube server in Jenkins

Firstly generate the token in sonarqube

Goto → Administaration→ security→ users→update token→ name= sonartoken and Generate


18.207.167.177:9000/admin/settings

Administration

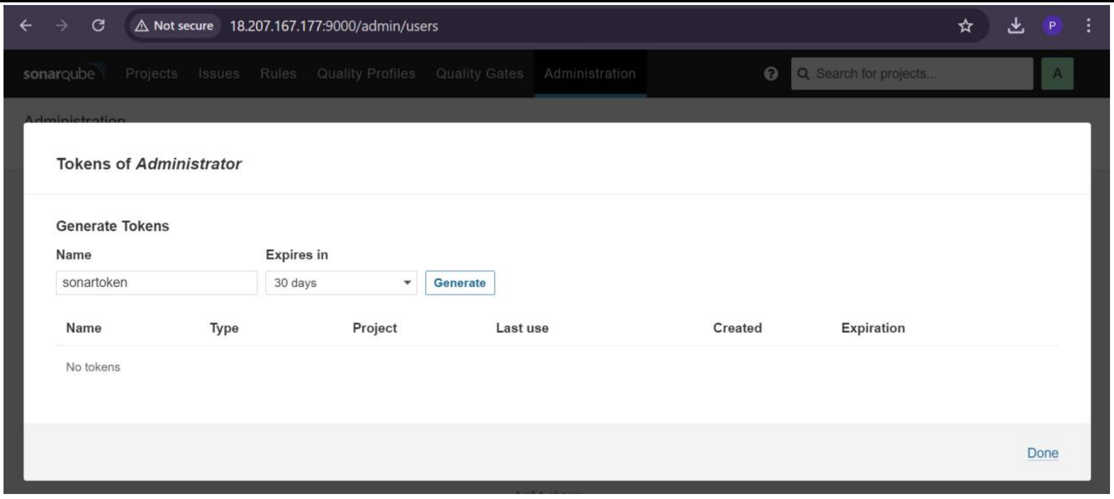
General Settings

Edit global settings for this SonarQube instance.

Duplications

Cross project duplication detection

DEPRECATED - By default, SonarQube detects duplications at project level. This means that a block duplicated on two different projects won't be reported. Setting this parameter to "true" allows to detect duplicates across projects. Note that activating this property will significantly increase each SonarQube analysis time, and therefore badly impact the


18.207.167.177:9000/admin/users

Tokens of *Administrator*

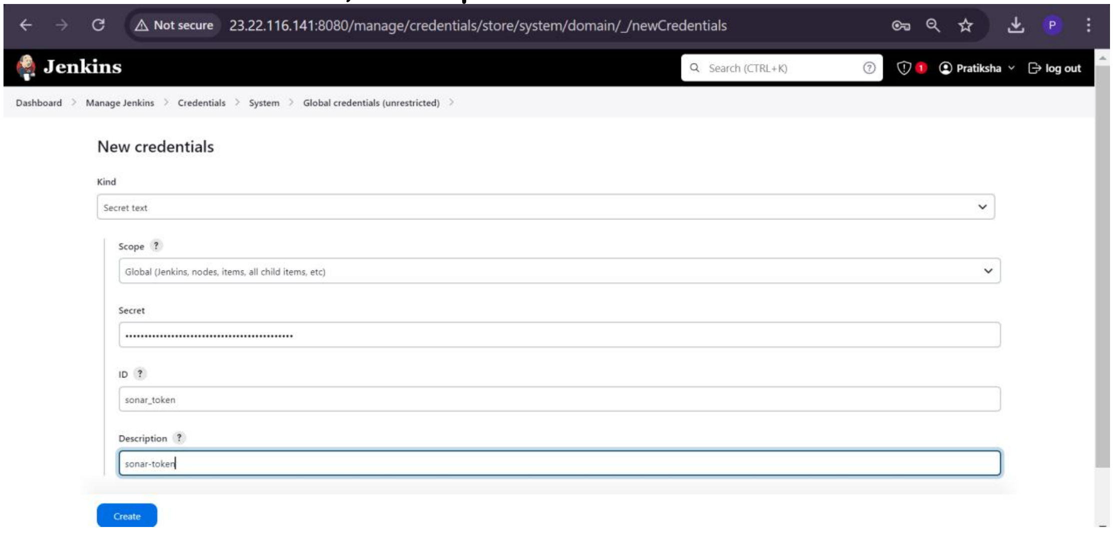
Generate Tokens

Name	Expires in	
sonartoken	30 days	Generate

Name	Type	Project	Last use	Created	Expiration
No tokens					

add the token in jenkins :

goto→ manage jenkins→credentials→ global→ kind= secret text→secret=<your token>id=sonar-token, description=sonar=token


23.22.116.141:8080/manage/credentials/store/system/domain/_/newCredentials

Jenkins
Search (CTRL+K)
Pratiksha
log out
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

New credentials

Kind: Secret text

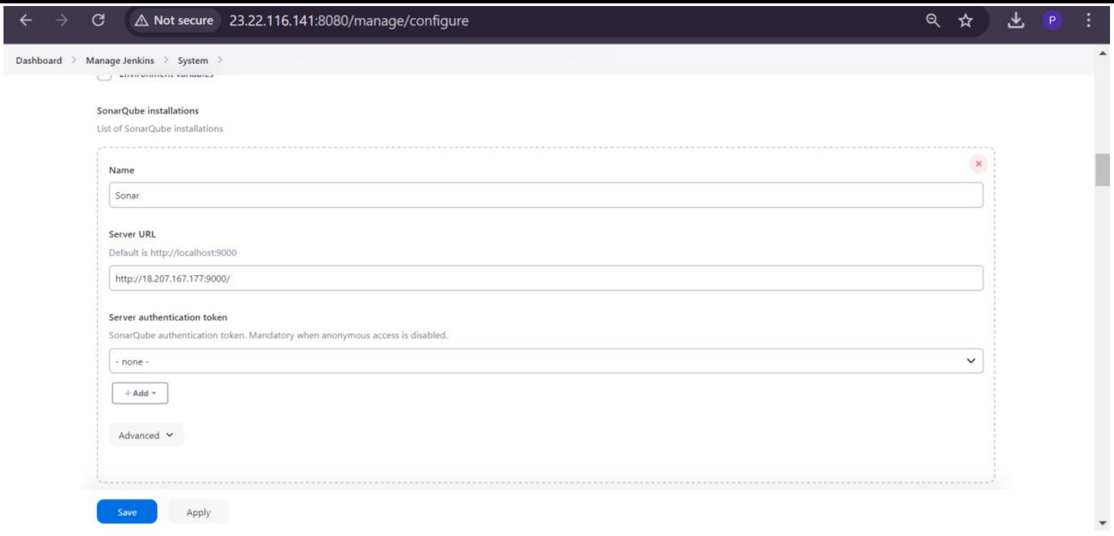
Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret: ••

ID: sonar_token

Description: sonar-token

Go to → manage Jenkins→ system→sonarqube server→name=sonar, url=http://publicip:9000, token=sonar-token



Not secure 23.22.116.141:8080/manage/configure

Dashboard > Manage Jenkins > System

SonarQube installations

List of SonarQube installations

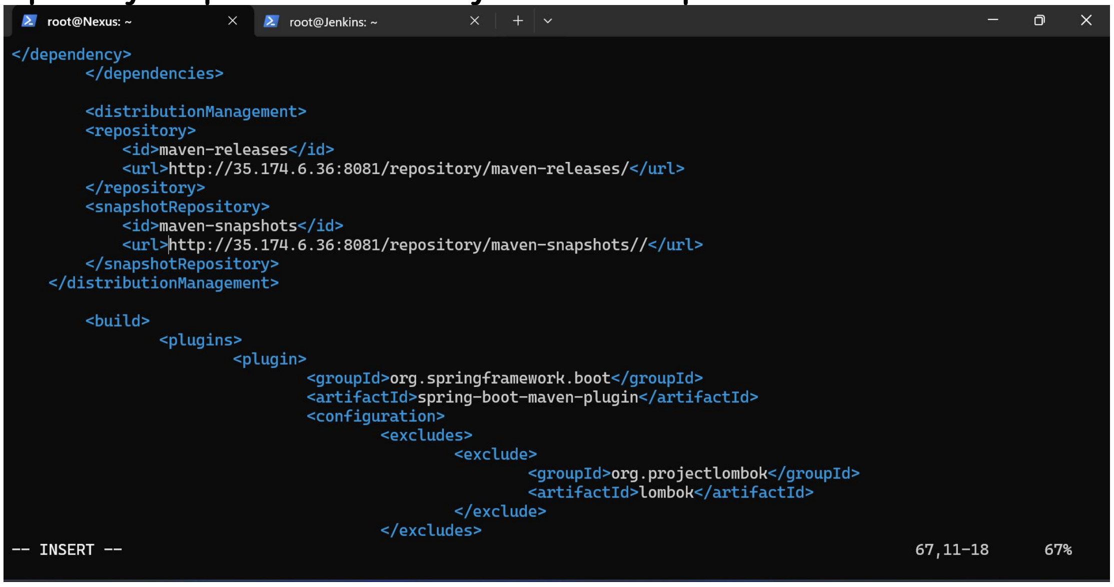
Name: Sonar

Server URL: http://18.207.167.177:9000/ (Default is http://localhost:9000)

Server authentication token: - none - (SonarQube authentication token. Mandatory when anonymous access is disabled.)

Sonarqube scanner→ This is the tool that actually scans your code and sends the results to the SonarQube server.
Sonarqube server→ Displays analysis results .

Nexus Configuration:
Update your pom.xml file with your nexus repositories

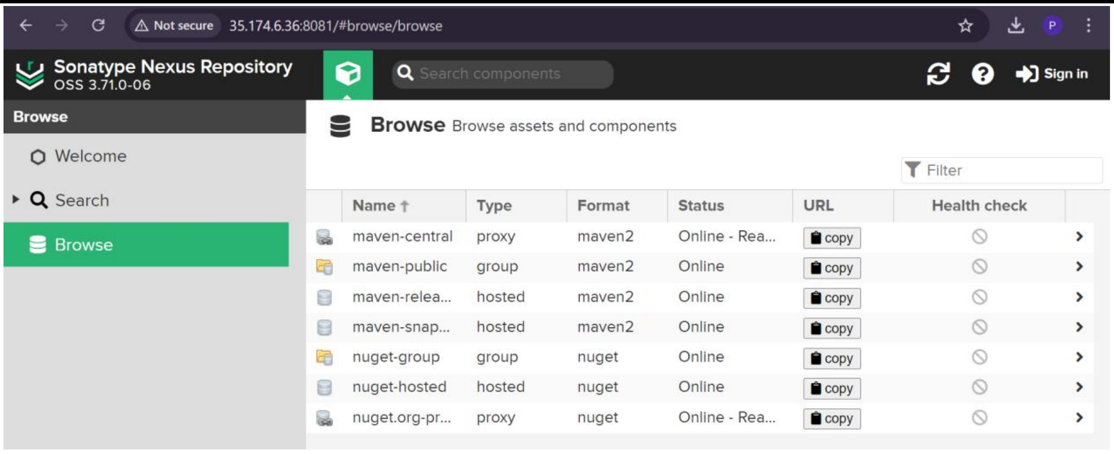


```
</dependency>
        </dependencies>

        <distributionManagement>
        <repository>
            <id>maven-releases</id>
            <url>http://35.174.6.36:8081/repository/maven-releases/</url>
        </repository>
        <snapshotRepository>
            <id>maven-snapshots</id>
            <url>http://35.174.6.36:8081/repository/maven-snapshots//</url>
        </snapshotRepository>
    </distributionManagement>

        <build>
                <plugins>
                        <plugin>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-plugin</artifactId>
                                <configuration>
                                        <excludes>
                                                <exclude>
                                                        <groupId>org.projectlombok</groupId>
                                                        <artifactId>lombok</artifactId>
                                                </exclude>
                                        </excludes>
-- INSERT --                                                                                  67,11-18      67%
```

Copy the maven-releases URL , maven-snapshots URL and update in the pom.xml file

<url>http://35.174.6.36:8081/repository/maven-releases/>

<url>http://35.174.6.36:8081/repository/maven-snapshots/>


Not secure 35.174.6.36:8081/#browse/browse

Sonatype Nexus Repository OSS 3.71.0-06

Browse Browse assets and components

Name	Type	Format	Status	URL	Health check
maven-central	proxy	maven2	Online - Rea...	copy	
maven-public	group	maven2	Online	copy	
maven-relea...	hosted	maven2	Online	copy	
maven-snap...	hosted	maven2	Online	copy	
nuget-group	group	nuget	Online	copy	
nuget-hosted	hosted	nuget	Online	copy	
nuget.org-pr...	proxy	nuget	Online - Rea...	copy	

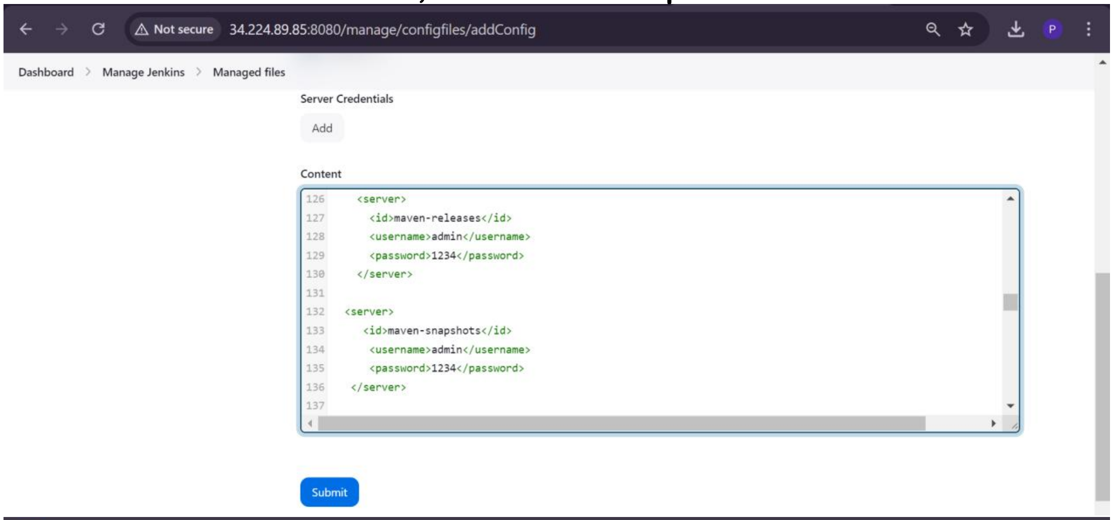
Nexus authentication with Jenkins:

Go to→ manage Jenkins→manage files→add new config→ select global mavensettings.xml,

id=maven-setting → click on next

Go to content

Add the servers with name, username and password

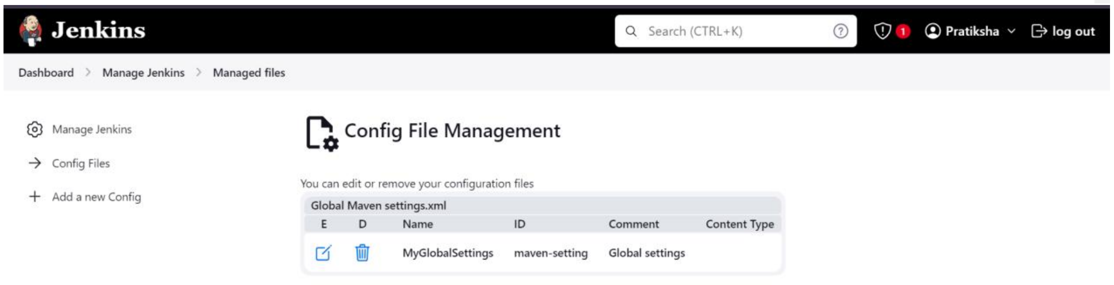

Not secure 34.224.89.85:8080/manage/configfiles/addConfig

Dashboard > Manage Jenkins > Managed files

Server Credentials

Content

```
    <server>
      <id>maven-releases</id>
      <username>admin</username>
      <password>1234</password>
    </server>

  <server>
     <id>maven-snapshots</id>
      <username>admin</username>
      <password>1234</password>
   </server>
```


Dashboard > Manage Jenkins > Managed files

Config File Management

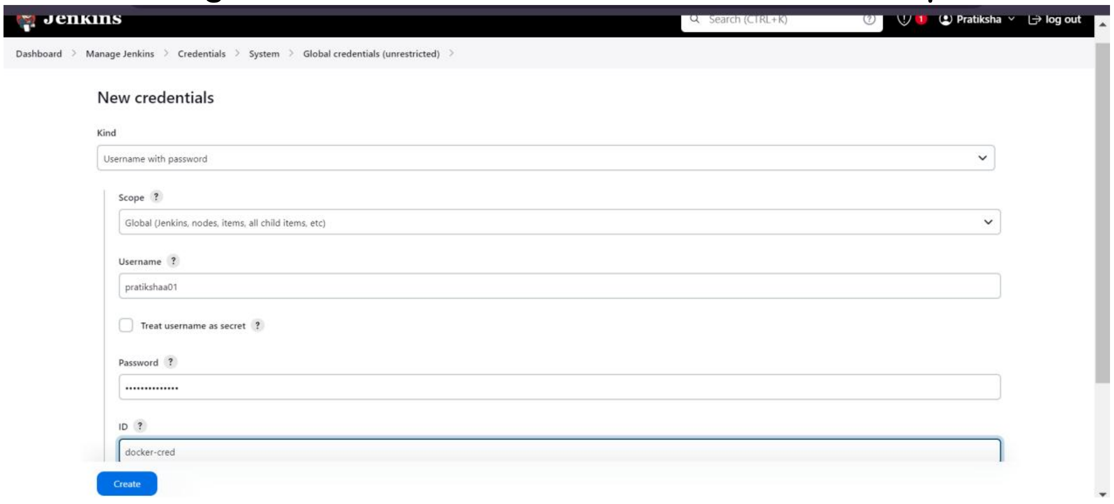
You can edit or remove your configuration files

Global Maven settings.xml

E	D	Name	ID	Comment	Content Type
		MyGlobalSettings	maven-setting	Global settings	

Add DockerHub credentials in Jenkins:

Goto→ manage Jenkins→credentials→kind=username and password


Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

New credentials

Kind: Username with password

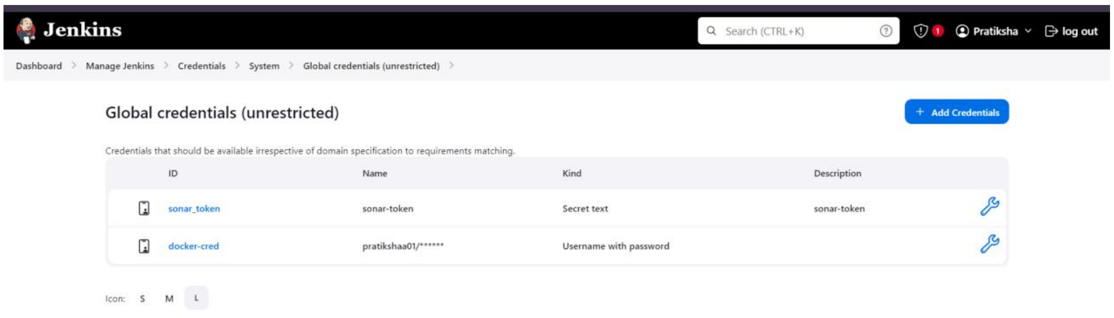
Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: pratikshaa01

Treat username as secret: ☐

Password: ••••••••••••••

ID: docker-cred


Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

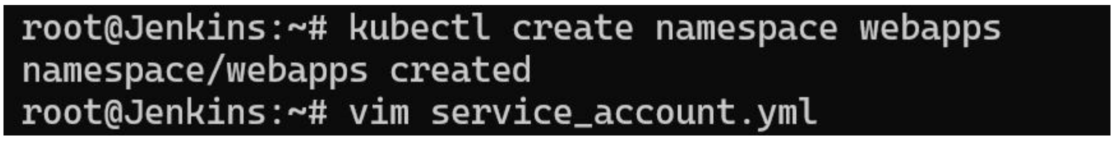
ID	Name	Kind	Description
sonar_token	sonar-token	Secret text	sonar-token
docker-cred	pratikshaa01/******	Username with password	

Create Service Account, Role & Assign that role, And create a secret for Service Account and generate a Token in Jenkins server

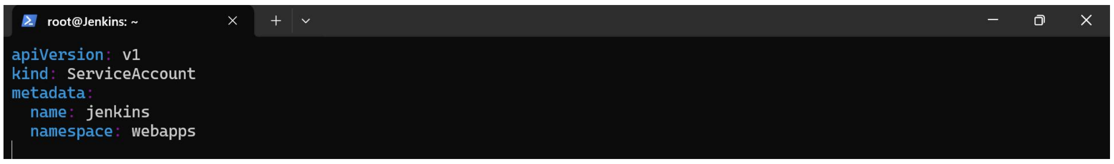
Creating Service Account

First create the namespace using

Kubectl create namespace webapps

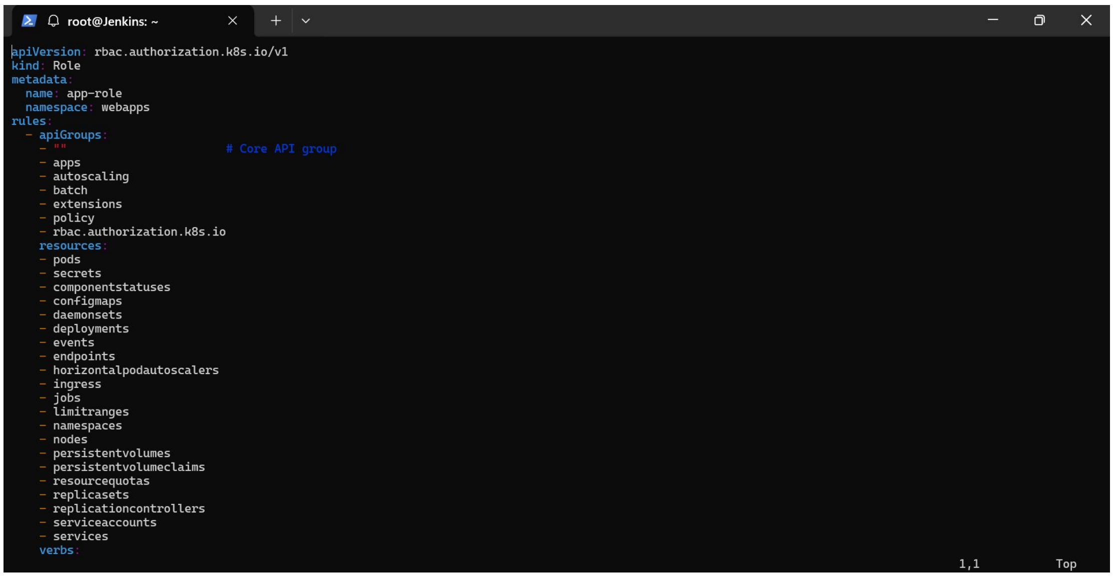


```
root@Jenkins:~# kubectl create namespace webapps
namespace/webapps created
root@Jenkins:~# vim service_account.yml
```

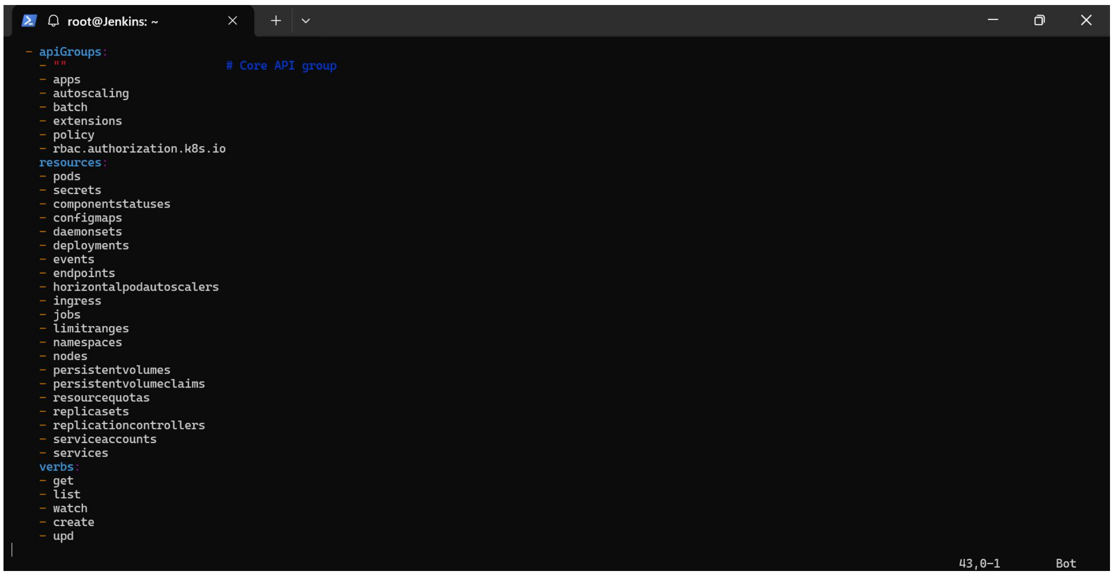


```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: jenkins
  namespace: webapps
```

Create Role :

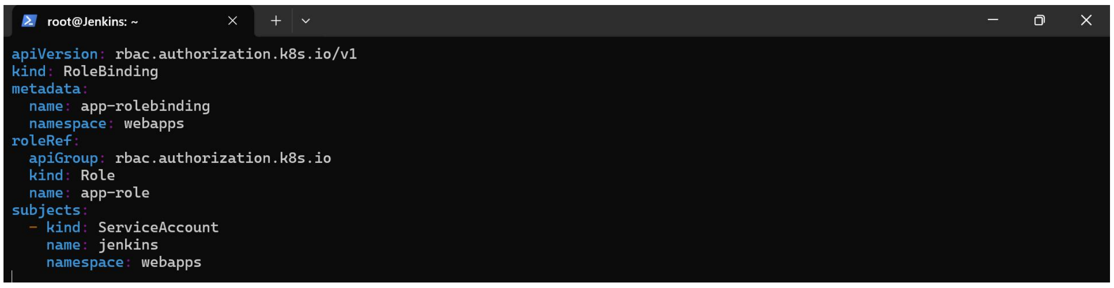


```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: app-role
  namespace: webapps
rules:
  - apiGroups:
    - ""                       # Core API group
    - apps
    - autoscaling
    - batch
    - extensions
    - policy
    - rbac.authorization.k8s.io
    resources:
    - pods
    - secrets
    - componentstatuses
    - configmaps
    - daemonsets
    - deployments
    - events
    - endpoints
    - horizontalpodautoscalers
    - ingress
    - jobs
    - limitranges
    - namespaces
    - nodes
    - persistentvolumes
    - persistentvolumeclaims
    - resourcequotas
    - replicasets
    - replicationcontrollers
    - serviceaccounts
    - services
    verbs:
                                                                  1,1           Top
```

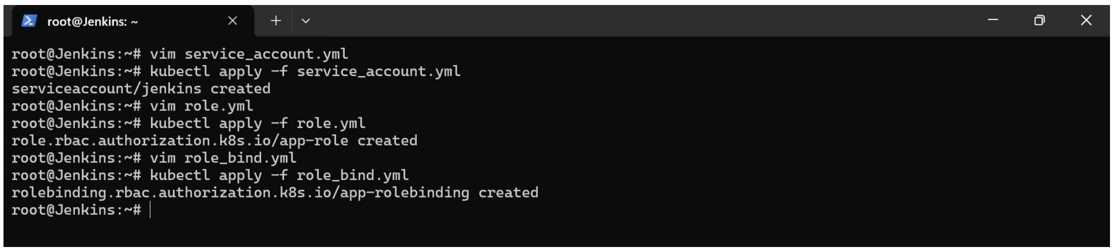


```
  - apiGroups:
    - ""                       # Core API group
    - apps
    - autoscaling
    - batch
    - extensions
    - policy
    - rbac.authorization.k8s.io
    resources:
    - pods
    - secrets
    - componentstatuses
    - configmaps
    - daemonsets
    - deployments
    - events
    - endpoints
    - horizontalpodautoscalers
    - ingress
    - jobs
    - limitranges
    - namespaces
    - nodes
    - persistentvolumes
    - persistentvolumeclaims
    - resourcequotas
    - replicasets
    - replicationcontrollers
    - serviceaccounts
    - services
    verbs:
    - get
    - list
    - watch
    - create
    - upd

                                                                  43,0-1        Bot
```

Bind the role to service account:

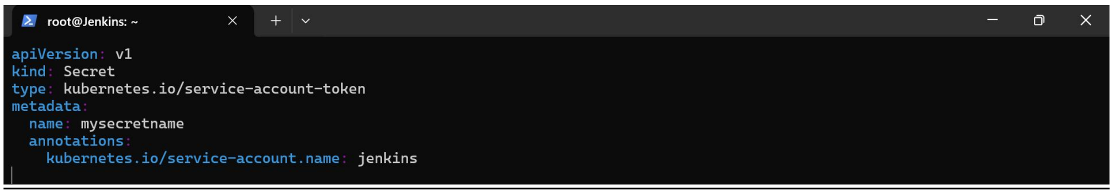


```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-rolebinding
  namespace: webapps
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: app-role
subjects:
  - kind: ServiceAccount
    name: jenkins
    namespace: webapps
```

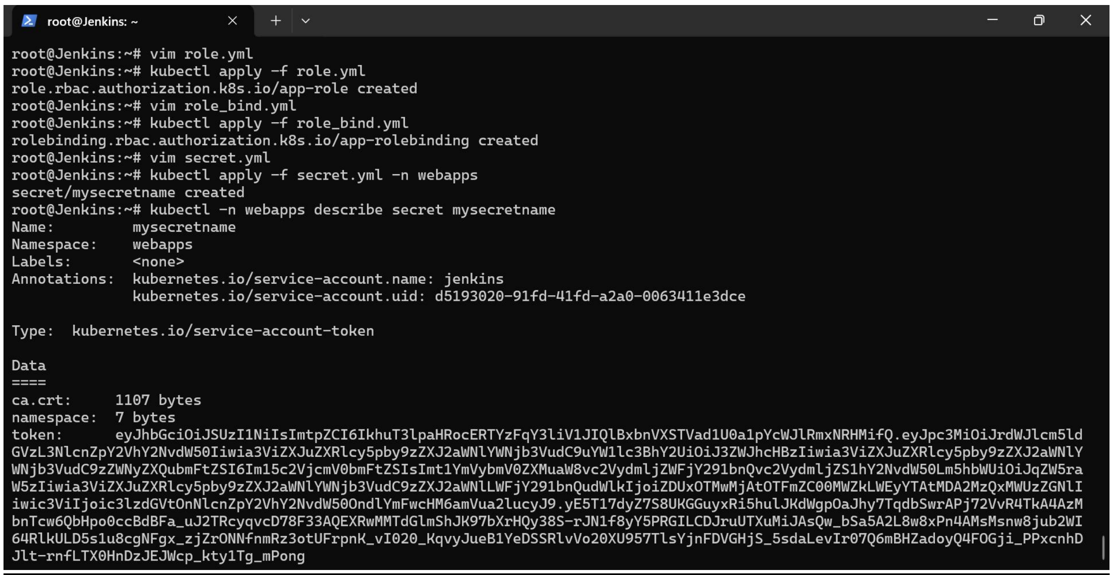


```
root@Jenkins:~# vim service_account.yml
root@Jenkins:~# kubectl apply -f service_account.yml
serviceaccount/jenkins created
root@Jenkins:~# vim role.yml
root@Jenkins:~# kubectl apply -f role.yml
role.rbac.authorization.k8s.io/app-role created
root@Jenkins:~# vim role_bind.yml
root@Jenkins:~# kubectl apply -f role_bind.yml
rolebinding.rbac.authorization.k8s.io/app-rolebinding created
root@Jenkins:~#
```

Generate token using service account in the namespace:
apiVersion: v1
kind: Secret
type: kubernetes.io/service-account-token
metadata:
name: mysecretname
annotations:
kubernetes.io/service-account.name: myserviceaccount
kubectl -n webapps describe secret mysecretname

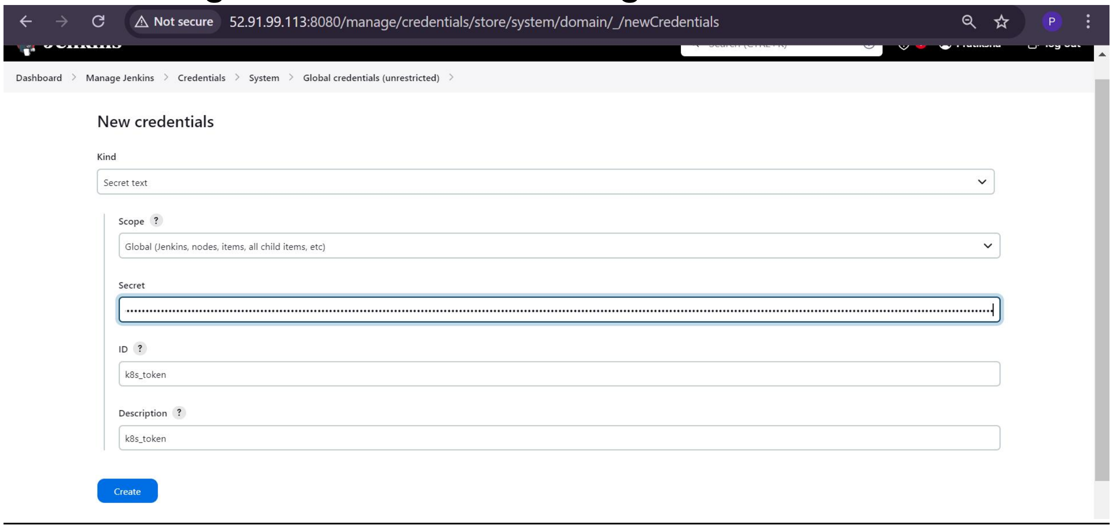

root@Jenkins: ~

```
apiVersion: v1
kind: Secret
type: kubernetes.io/service-account-token
metadata:
  name: mysecretname
  annotations:
    kubernetes.io/service-account.name: jenkins
```


root@Jenkins: ~

```
root@Jenkins:~# vim role.yml
root@Jenkins:~# kubectl apply -f role.yml
role.rbac.authorization.k8s.io/app-role created
root@Jenkins:~# vim role_bind.yml
root@Jenkins:~# kubectl apply -f role_bind.yml
rolebinding.rbac.authorization.k8s.io/app-rolebinding created
root@Jenkins:~# vim secret.yml
root@Jenkins:~# kubectl apply -f secret.yml -n webapps
secret/mysecretname created
root@Jenkins:~# kubectl -n webapps describe secret mysecretname
Name:         mysecretname
Namespace:    webapps
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: jenkins
              kubernetes.io/service-account.uid: d5193020-91fd-41fd-a2a0-0063411e3dce

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1107 bytes
namespace:  7 bytes
token:      eyJhbGciOiJSUzI1NiIsImtpZCI6IkhuT3lpaHRocERTYzFqY3liV1JIQlBxbnVXSTVad1U0a1pYcWJlRmxNRHMifQ.eyJpc3MiOiJrdWJlcm5ld
GVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJ3ZWJhcHBzIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlY
WNjb3VudC9zZWNyZXQubmFtZSI6Im15c2VjcmV0bmFtZSIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50Lm5hbWUiOiJqZW5ra
W5zIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiZDUxOTMwMjAtOTFmZC00MWZkLWEyYTAtMDA2MzQxMWUzZGNlI
iwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW50OndlYmFwcHM6amVua2lucyJ9.yE5T17dyZ7S8UKGGuyxRi5hulJKdWgpOaJhy7TqdbSwrAPj72VvR4TkA4AzM
bnTcw6QbHpo0ccBdBFa_uJ2TRcyqvcD78F33AQEXRwMMTdGlmShJK97bXrHQy38S-rJN1f8yY5PRGILCDJruUTXuMiJAsQw_bSa5A2L8w8xPn4AMsMsnw8jub2WI
64RlkULD5s1u8cgNFgx_zjZrONNfnmRz3otUFrpnK_vI020_KqvyJueB1YeDSSRlvVo20XU957TlsYjnFDVGHjS_5sdaLevIr07Q6mBHZadoyQ4FOGji_PPxcnhD
Jlt-rnfLTX0HnDzJEJWcp_kty1Tg_mPong
```

Add this token in Jenkins server
Goto→ manage Jenkins→ credentials→global→kind= secret text


Not secure 52.91.99.113:8080/manage/credentials/store/system/domain/_/newCredentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

New credentials

Kind: Secret text

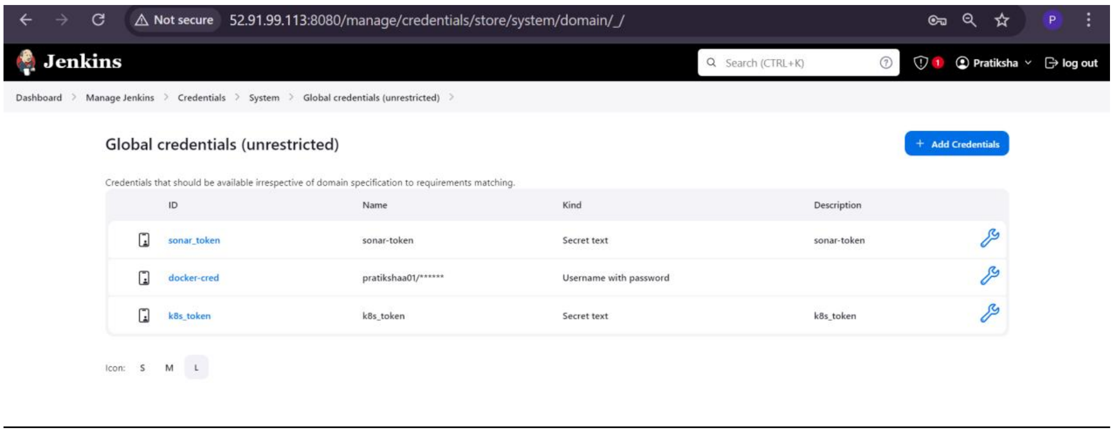
Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret: ••

ID: k8s_token

Description: k8s_token

Create



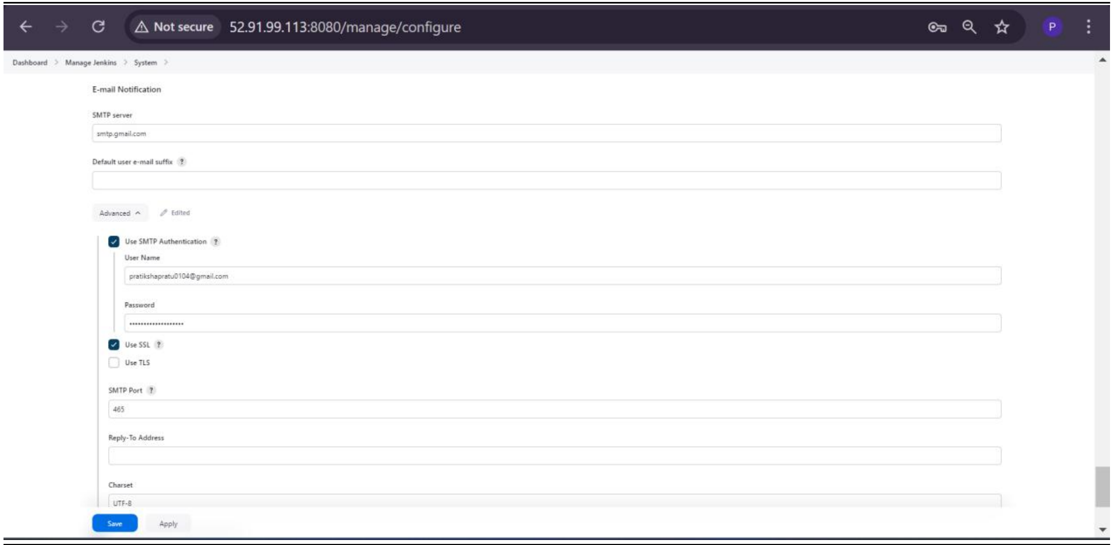
52.91.99.113:8080/manage/credentials/store/system/domain/_/

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
sonar_token	sonar-token	Secret text	sonar-token
docker-cred	pratikshaa01/******	Username with password	
k8s_token	k8s_token	Secret text	k8s_token



52.91.99.113:8080/manage/configure

Dashboard > Manage Jenkins > System >

E-mail Notification

SMTP server: smtp.gmail.com

Default user e-mail suffix:

Advanced — Edited

[✓] Use SMTP Authentication

User Name: pratikshapratu0104@gmail.com

Password: ••••••••••••••••••••

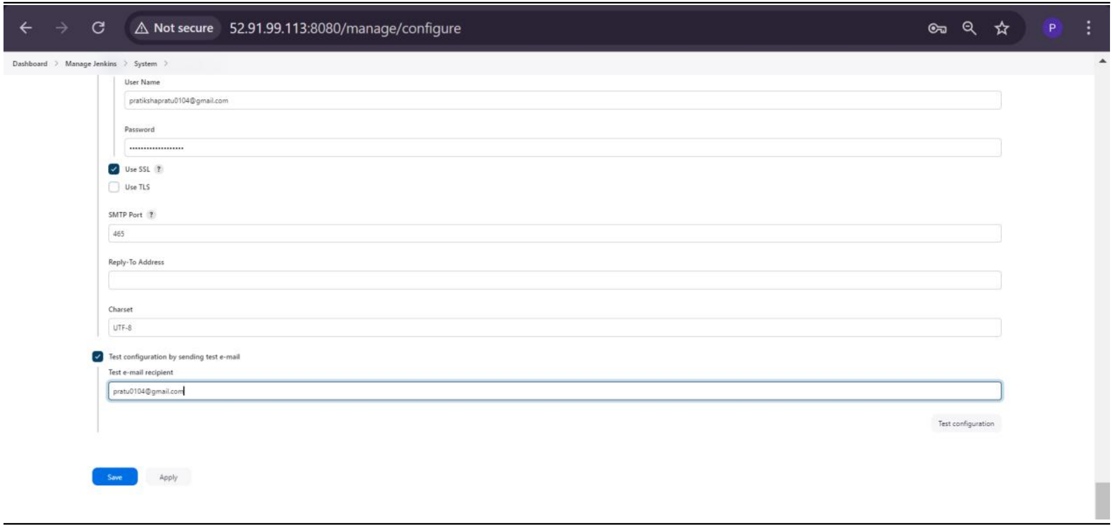
[✓] Use SSL

[ ] Use TLS

SMTP Port: 465

Reply-To Address:

Charset: UTF-8



52.91.99.113:8080/manage/configure

Dashboard > Manage Jenkins > System >

User Name: pratikshapratu0104@gmail.com

Password: ••••••••••••••••••••

[✓] Use SSL

[ ] Use TLS

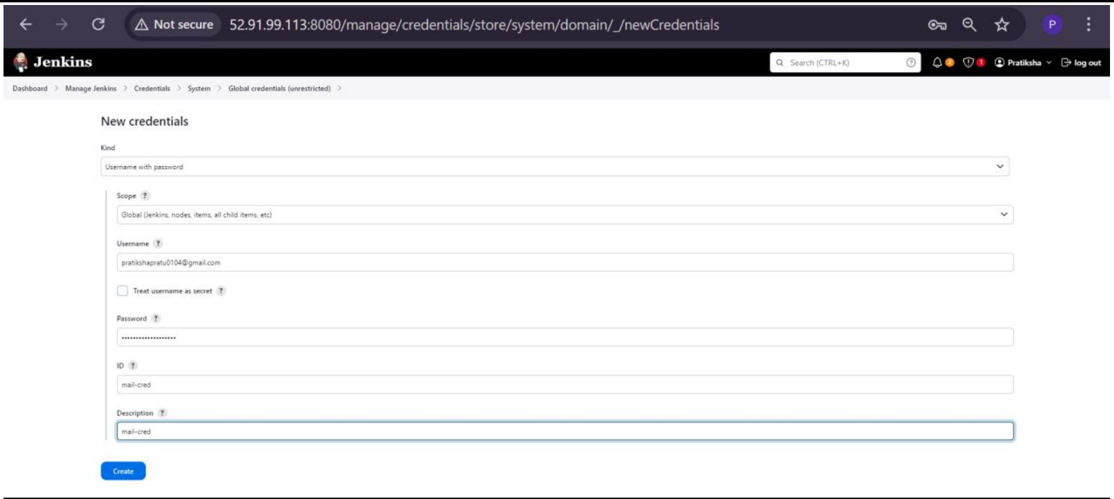
SMTP Port: 465

Reply-To Address:

Charset: UTF-8

[✓] Test configuration by sending test e-mail

Test e-mail recipient: pratu0104@gmail.com


Not secure 52.91.99.113:8080/manage/credentials/store/system/domain/_/newCredentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

New credentials

Kind: Username with password

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

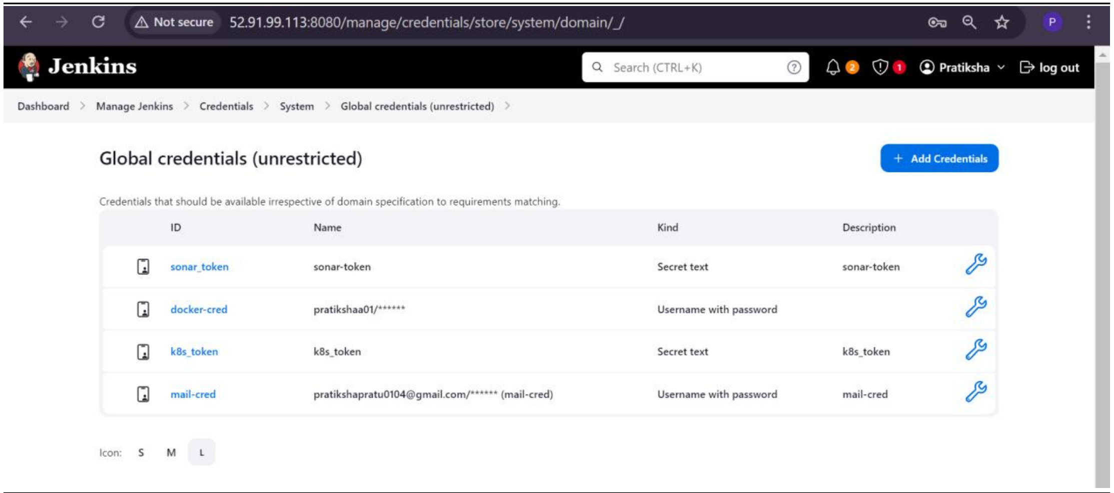
Username: pratikshapratu0104@gmail.com

Treat username as secret

Password: ••••••••••••••••••••

ID: mail-cred

Description: mail-cred


Not secure 52.91.99.113:8080/manage/credentials/store/system/domain/_/

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
sonar_token	sonar-token	Secret text	sonar-token
docker-cred	pratikshaa01/******	Username with password	
k8s_token	k8s_token	Secret text	k8s_token
mail-cred	pratikshapratu0104@gmail.com/****** (mail-cred)	Username with password	mail-cred

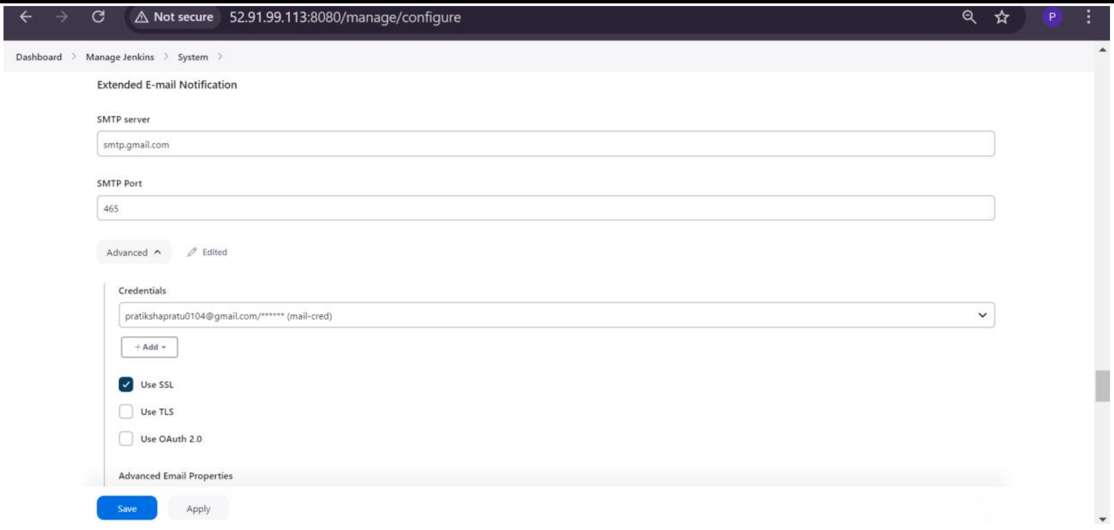
Email Notification Configurations:

Goto this URL https://myaccount.google.com/apppasswords

generate apppassword and copy that password jijv akam wedd ujip next go to Jenkins→manage Jenkins→system→E-mail Notification→smtp server=
smtp.gmail.com, Advanced→ Use smtp Authentication→username="<yourgamilname>", password="<apppassword>", port= 465 and Test configuration by sending test e-mail.

Goto manage Jenkins→ credentials→ add the gmail and password

Now goto→ manage Jenkins→ system→ Extended E-mail Notification→ smtp server=smtp.gmail.com, select the mail-cred


52.91.99.113:8080/manage/configure

Dashboard > Manage Jenkins > System >

Extended E-mail Notification

SMTP server: smtp.gmail.com

SMTP Port: 465

Credentials: pratikshapratu0104@gmail.com/****** (mail-cred)

[✓] Use SSL
[ ] Use TLS
[ ] Use OAuth 2.0

Advanced Email Properties

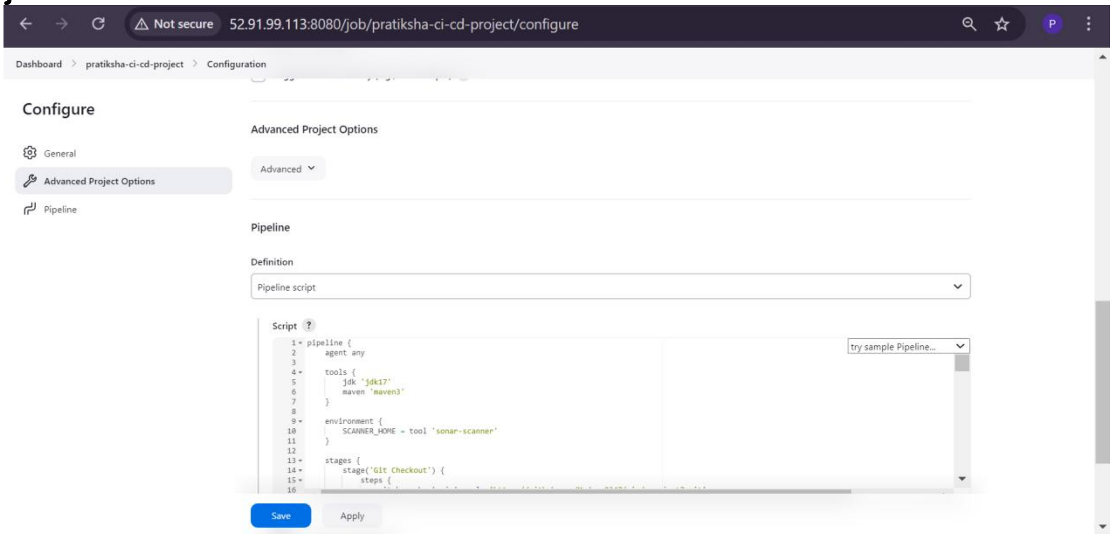
<u>Now write the Jenkinsfile:</u>

```
pipeline {
   agent any

   tools {
      jdk 'jdk17'
      maven 'maven3'
   }

   environment {
      SCANNER_HOME = tool 'sonar-scanner'
   }

   stages {
      stage('Git Checkout') {
         steps {
            git branch: 'main', url: 'https://github.com/pratikshaa-01/CI-CD-Project.git'
         }
      }

      stage('Compile') {
         steps {
            sh "mvn compile"
         }
      }

      stage('Test') {
         steps {
            sh "mvn package -DskipTests=true"
         }
      }

      stage('Trivy Scan File System') {
         steps {
            sh "trivy fs --format table -o trivy-fs-report.html ."
```

```
        }
    }

    stage('SonarQube Analysis') {
        steps {
            withSonarQubeEnv('sonar') {
                sh '''$SCANNER_HOME/bin/sonar-scanner \
                -Dsonar.projectKey=Mission \
                -Dsonar.projectName=Mission \
                -Dsonar.java.binaries=.'''
            }
        }
    }

    stage('Build') {
        steps {
            sh "mvn package -DskipTests=true"
        }
    }

    stage('Deploy Artifacts To Nexus') {
        steps {
            withMaven(globalMavenSettingsConfig: 'maven-setting', jdk:
'jdk17', maven: 'maven3') {
                sh "mvn deploy -DskipTests=true"
            }
        }
    }

    stage('Build & Tag Docker Image') {
        steps {
            script {
                withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                    sh "docker build -t pratikshaa01/cicd-project:latest ."
                }
            }
        }
    }

    stage('Trivy Scan Image') {
        steps {
            sh "trivy image --format table -o trivy-image-report.html
pratikshaa01/cicd-project:latest"
        }
    }

    stage('Publish Docker Image') {
        steps {
            script {
```

```
            withDockerRegistry(credentialsId: 'docker-cred', toolName: 
'docker') {
                sh "docker push pratikshaa01/cicd-project:latest"
            }
          }
        }
      }

      stage('Deploy to EKS') {
        steps {
          withKubeConfig(credentialsId: 'k8s-token', namespace: 
'webapps', serverUrl: 
'https://FE0E7FFC80B64E124F6F3EA8EDA2FE7E.sk1.ap-south-
1.eks.amazonaws.com') {
            sh "kubectl apply -f ds.yml -n webapps"
            sleep 60
          }
        }
      }

      stage('Verify deployment') {
        steps {
          withKubeConfig(credentialsId: 'k8s-token', namespace: 
'webapps', serverUrl: 
'https://FE0E7FFC80B64E124F6F3EA8EDA2FE7E.sk1.ap-south-
1.eks.amazonaws.com') {
            sh "kubectl get pods -n webapps"
            sh "kubectl get svc -n webapps"
          }
        }
      }
    }

    post {
      always {
        script {
          def jobName = env.JOB_NAME
          def buildNumber = env.BUILD_NUMBER
          def pipelineStatus = currentBuild.result ?: 'UNKNOWN'
          def bannerColor = pipelineStatus.toUpperCase() == 
'SUCCESS' ? 'green' : 'red'
          def body = """
          <html>
          <body>
          <div style="border: 4px solid ${bannerColor}; padding: 10px;">
          <h2>${jobName} - Build ${buildNumber}</h2>
          <div style="background-color: ${bannerColor}; padding: 
10px;">
          <h3 style="color: white;">Pipeline Status: 
${pipelineStatus.toUpperCase()}</h3>
```

```
            </div>
            <p>Check the <a href="${BUILD_URL}">console
output</a>.</p>
            </div>
            </body>
            </html>
            """
            emailext(
               subject: "${jobName} - Build ${buildNumber} -
${pipelineStatus.toUpperCase()}",
               body: body,
               to: 'pratikshapratu0104@gmail.com',
               from: 'jenkins@example.com',
               replyTo: 'jenkins@example.com',
               mimeType: 'text/html',
               attachmentsPattern: 'trivy-image-report.html'
            )
         }
      }
   }
}
```


Not secure
52.91.99.113:8080/job/pratiksha-ci-cd-project/configure
Dashboard > pratiksha-ci-cd-project > Configuration
Configure
General
Advanced Project Options
Pipeline
Advanced
try sample Pipeline...
Save
Apply

Advanced Project Options

Pipeline

Definition: Pipeline script

Script

```
pipeline {
    agent any

    tools {
        jdk 'jdk17'
        maven 'maven3'
    }

    environment {
        SCANNER_HOME = tool 'sonar-scanner'
    }

    stages {
        stage('Git Checkout') {
            steps {
```

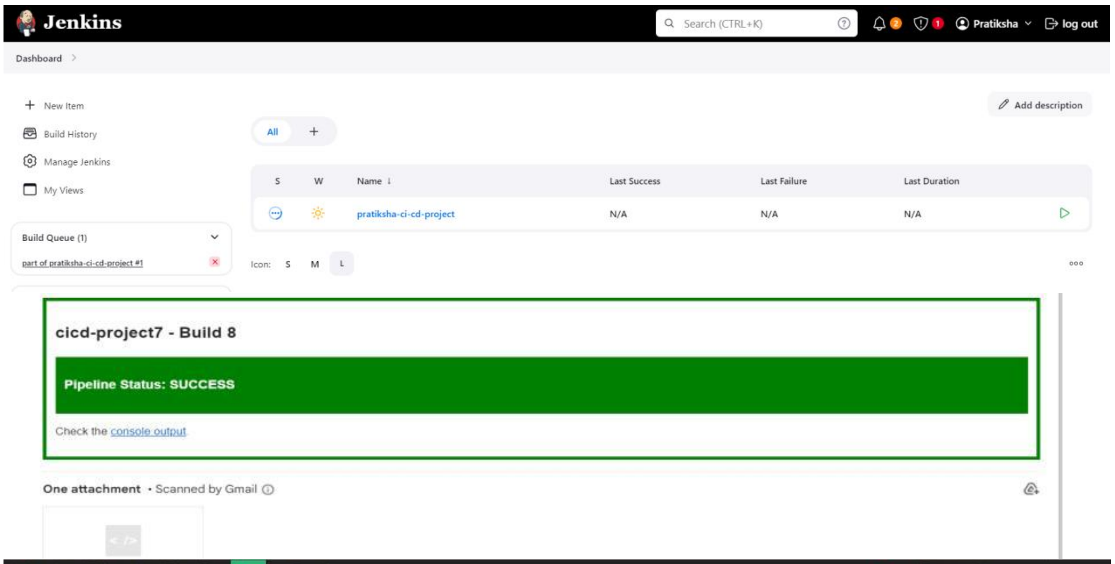
<u>**SSH on jenkins server:**</u>

Kubectl get pods -n webapps

Kubectl get svc -n webapps

Kubectl configuration cleaned up

Sending mail to: pratu0104@gmail.com



Build Queue (1)

part of pratiksha-ci-cd-project #1

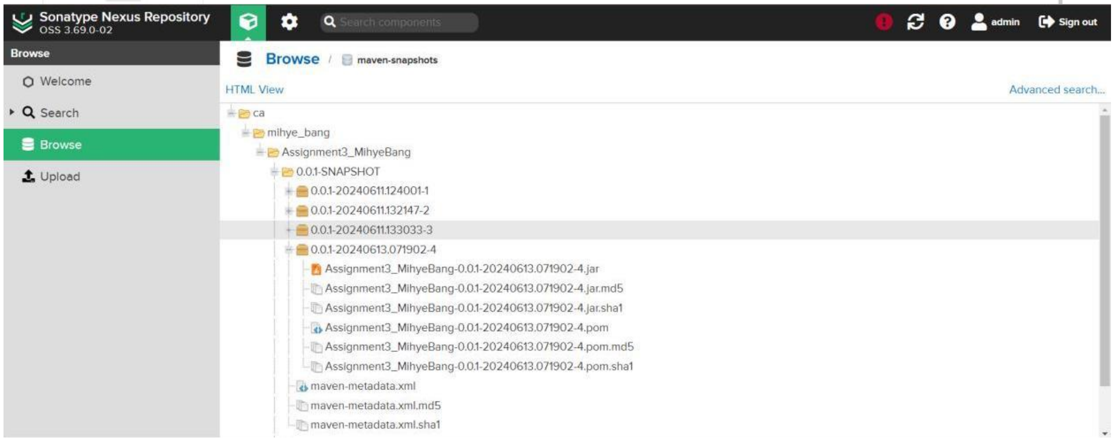
S	W	Name ↓	Last Success	Last Failure	Last Duration
		pratiksha-ci-cd-project	N/A	N/A	N/A

cicd-project7 - Build 8

Pipeline Status: SUCCESS

Check the console output

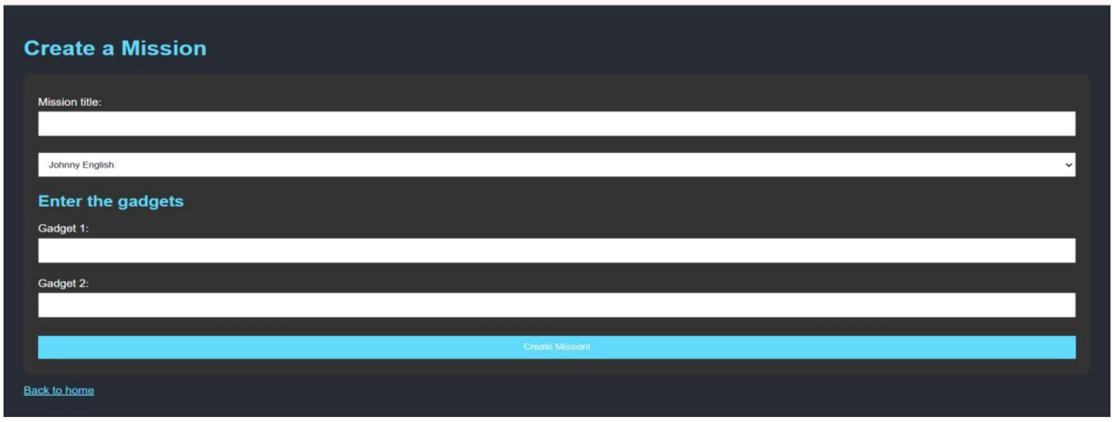
One attachment • Scanned by Gmail



```
ca
  mihye_bang
    Assignment3_MihyeBang
      0.0.1-SNAPSHOT
        0.0.1-20240611.124001-1
        0.0.1-20240611.132147-2
        0.0.1-20240611.133033-3
        0.0.1-20240613.071902-4
          Assignment3_MihyeBang-0.0.1-20240613.071902-4.jar
          Assignment3_MihyeBang-0.0.1-20240613.071902-4.jar.md5
          Assignment3_MihyeBang-0.0.1-20240613.071902-4.jar.sha1
          Assignment3_MihyeBang-0.0.1-20240613.071902-4.pom
          Assignment3_MihyeBang-0.0.1-20240613.071902-4.pom.md5
          Assignment3_MihyeBang-0.0.1-20240613.071902-4.pom.sha1
        maven-metadata.xml
        maven-metadata.xml.md5
        maven-metadata.xml.sha1
```

Kubectl get all -n webapps

Access the Application using the External-ip http://ac7b1a92512c243848be2a7df6fcee96-328134965.ap-south-1. elb.amazonaws.com:8080/addMission



Create a Mission

Mission title:

Johnny English

Enter the gadgets

Gadget 1:

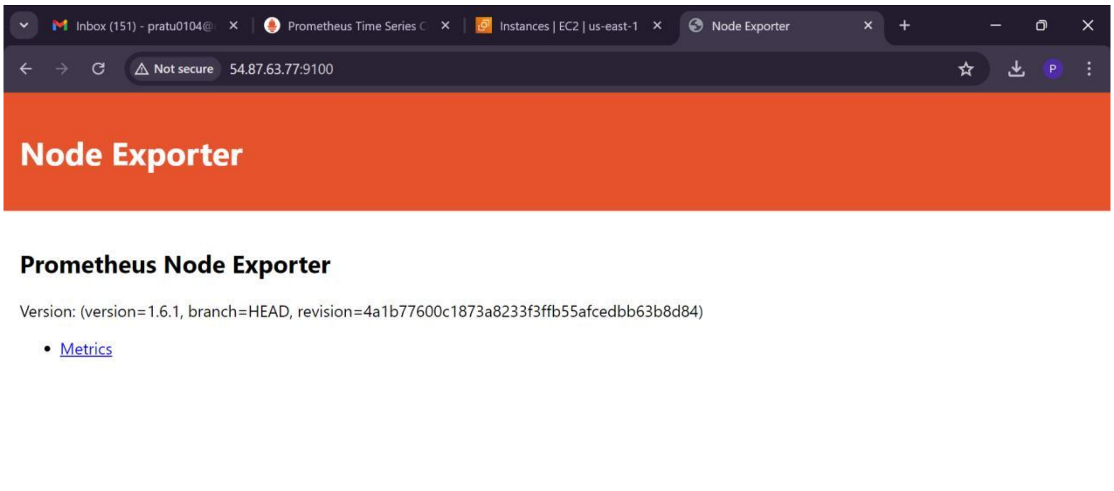
Gadget 2:

Create Mission!

Back to home

Setup Prometheus,Grafana,node-exporter,blackbox exporter

1. Install Node Exporter in Jenkins server



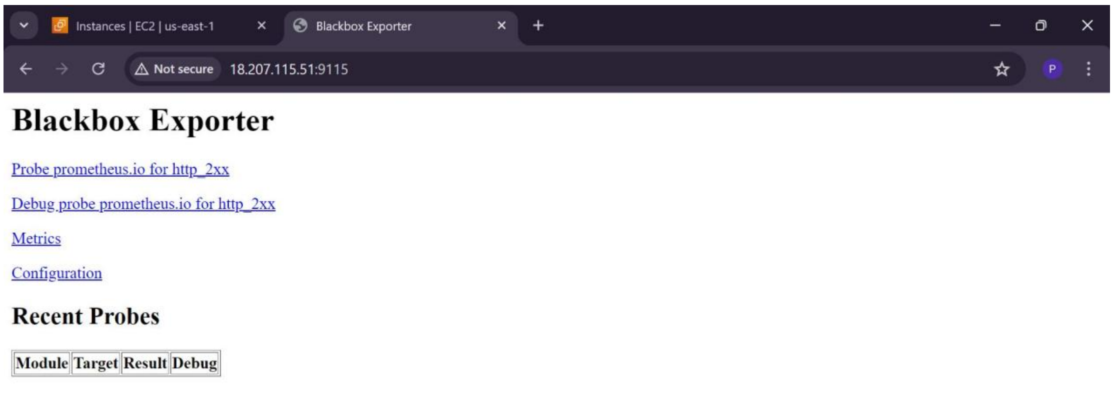
Node Exporter

Prometheus Node Exporter

Version: (version=1.6.1, branch=HEAD, revision=4a1b77600c1873a8233f3ffb55afcedbb63b8d84)

- Metrics

2. Install Blackbox Exporter in Jenkins server1. Download Blackbox Exporter



Blackbox Exporter

Probe prometheus.io for http_2xx

Debug probe prometheus.io for http_2xx

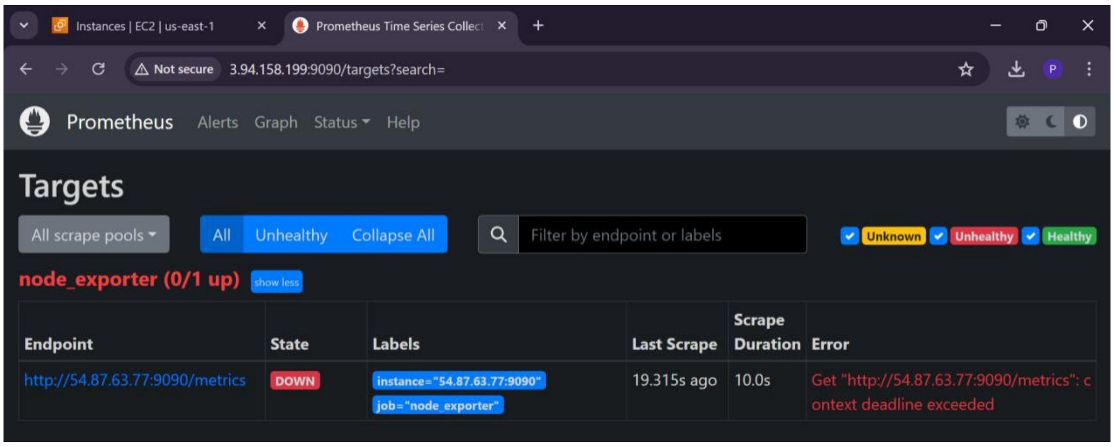
Metrics

Configuration

Recent Probes

Module	Target	Result	Debug

3. Install Prometheus in Jenkins server

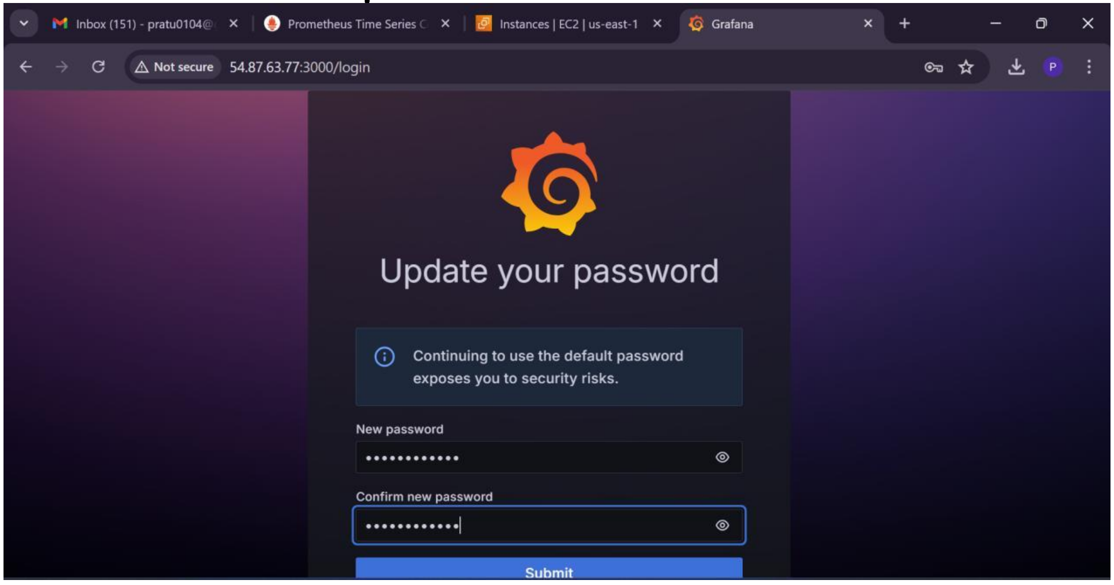

Instances | EC2 | us-east-1
Prometheus Time Series Collect
Not secure
3.94.158.199:9090/targets?search=
Prometheus
Alerts
Graph
Status
Help
All scrape pools
All
Unhealthy
Collapse All
Filter by endpoint or labels
Unknown
Unhealthy
Healthy
show less

Targets

node_exporter (0/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://54.87.63.77:9090/metrics	DOWN	instance="54.87.63.77:9090" job="node_exporter"	19.315s ago	10.0s	Get "http://54.87.63.77:9090/metrics": context deadline exceeded

4. Installation and Setup of Grafana



54.87.63.77:3000/login

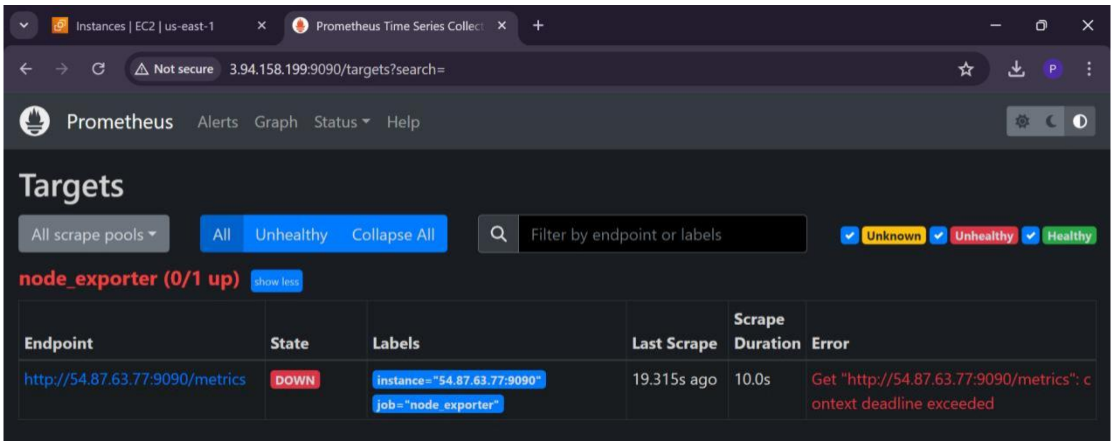
Update your password

Continuing to use the default password exposes you to security risks.

New password

Confirm new password

Submit



3.94.158.199:9090/targets?search=

Targets

node_exporter (0/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://54.87.63.77:9090/metrics	DOWN	instance="54.87.63.77:9090" job="node_exporter"	19.315s ago	10.0s	Get "http://54.87.63.77:9090/metrics": context deadline exceeded

~Done By: Pratiksha