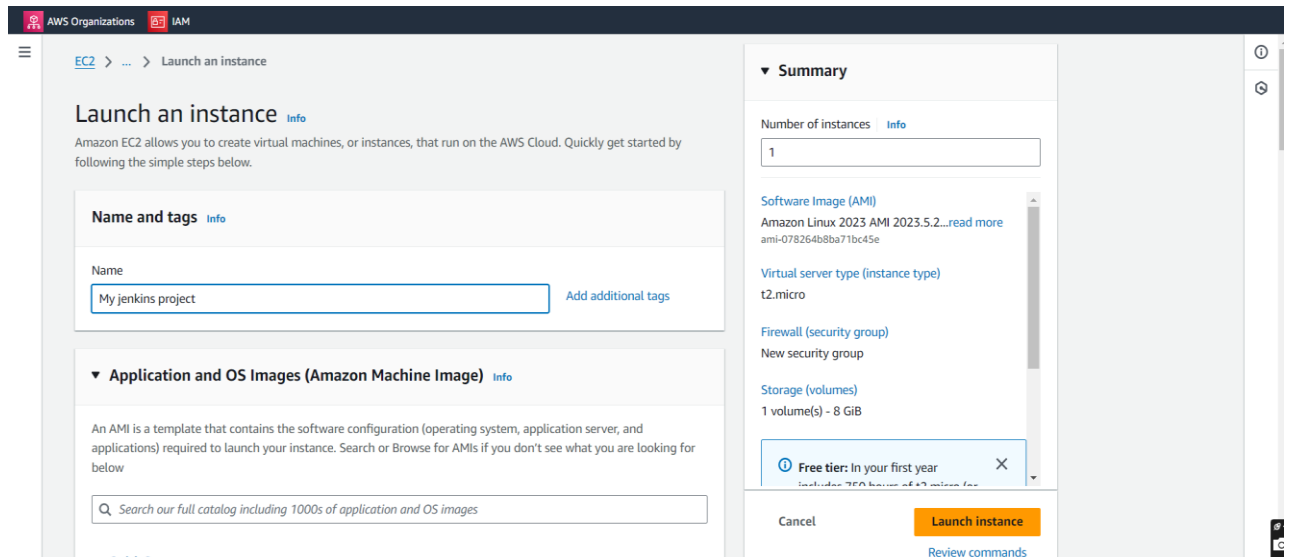
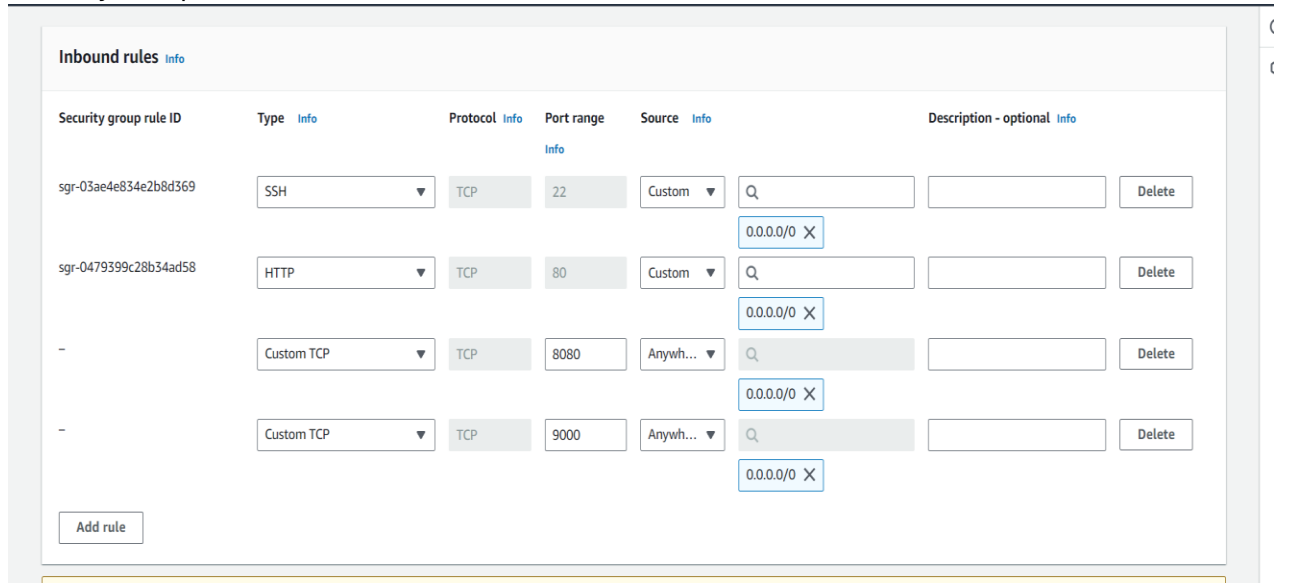


Jenkins Automation Project SOP

1. Launch an ec2 instance ,
AMI: ubuntu
instance type: t2.medium
open port: 8080 (for jenkins), allow HTTP traffic and 9000 (for sonar).



Security Group:



Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sgr-03ae4e834e2b8d369	SSH	TCP	22	Custom	0.0.0.0/0	Delete
sgr-0479399c28b34ad58	HTTP	TCP	80	Custom	0.0.0.0/0	Delete
-	Custom TCP	TCP	8080	Anywh...	0.0.0.0/0	Delete
-	Custom TCP	TCP	9000	Anywh...	0.0.0.0/0	Delete

Add rule

2. Connect to ec2 instance using ssh command:
ssh -i "path of the pem.file" username@public ip address
3. Install Jenkins on Ubuntu
<https://github.com/Sona-Yadav/jenkin-Project/blob/main/jenkins.sh>
4. Sign Into Jenkins console:
http://<EC2_PUBLIC_IP>:8080/
 - I. Get the Administrator password by hitting the below command in EC2 server.
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
 - II. Install all suggested plugins
 - III. Create first user

Getting Started

Username

Soonty

Password

.....

Confirm password

.....

Full name

Sona-Yadav

IV. Add the Plugins

Dashboard -> Manage Jenkins -> Plugins -> Available plugins

Plugins for Sonar/Jfrog -

Sonar Gerrit

SonarQube Scanner

SonarQube Generic Coverage

Sonar Quality Gates

Quality Gates

Artifactory

Pipeline: Stage View

Jfrog (not used in this project)

Dashboard > Manage Jenkins > Plugins

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Download progress

Oracle Java SE Development Kit Installer	✓ Success
SSH server	✓ Success
Command Agent Launcher	✓ Success
Quality Gates	✓ Success
JSch dependency	✓ Success
Gerrit Trigger	✓ Success
Sonar Gerrit	✓ Success
DataTables.net API	✓ Success
Forensics API	✓ Success
Prism API	✓ Success
Coverage	✓ Success
Code Coverage	✓ Success
SonarQube Generic Coverage	✓ Success
Config File Provider	✓ Success
Javadoc	✓ Success
Maven Integration	✓ Success
Jersey 2 API	✓ Success
Artifactory	✓ Success
Loading plugin extensions	⋮ Running
Restarting Jenkins	⋮ Pending

→ [Go back to the top page](#)

5. Setup Docker

<https://github.com/Sona-Yadav/jenkin-Project/blob/main/docker.sh>

docker -v

6. Install SonarQube

<https://github.com/Sona-Yadav/jenkin-Project/blob/main/sonarqube.sh>

Step 6.1 Start docker container if it's not up:

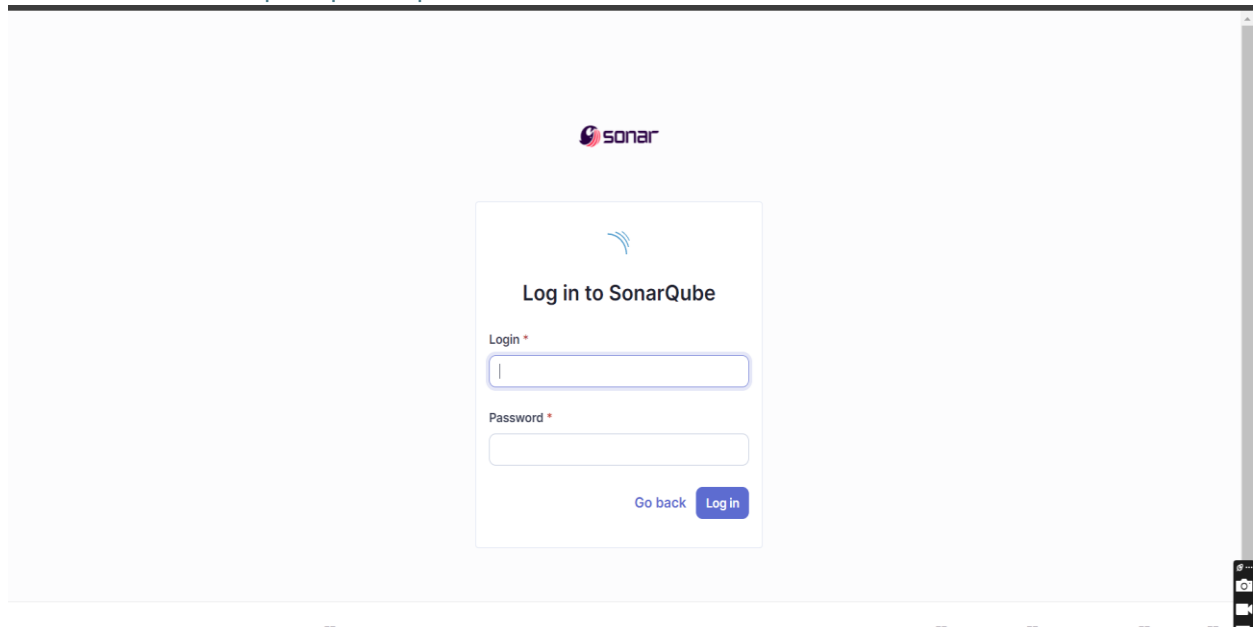
sudo docker ps -a [Get the container ID]

```
ubuntu@ip-172-31-41-178:~$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
76fbfc35d620   sonarqube     "/opt/sonarqube/dock... 32 seconds ago Up 30 seconds 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp, 0.0.0.0:9092->9092/tcp, :::9092->9092/tcp
sonarqube
```

sudo docker start <containerID>

Step 6.2 Login into sonar dashboard:

<http://<publicip address>:9000>



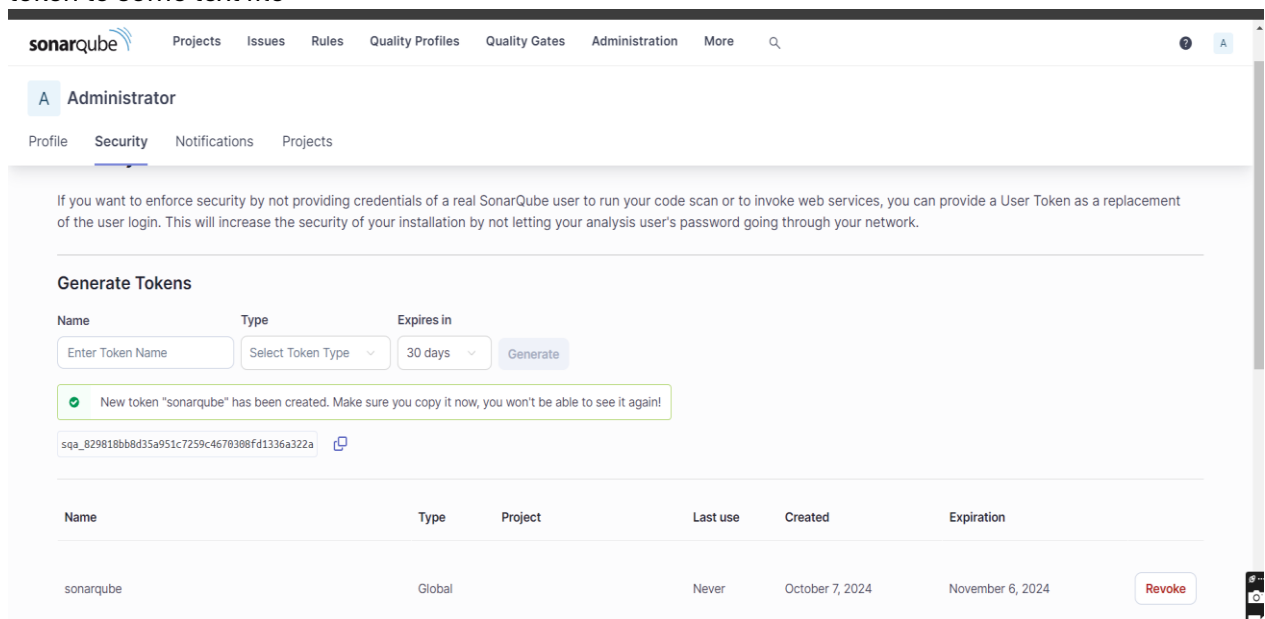
For the first time, use the following credentials for login:

Username – admin

Password – admin

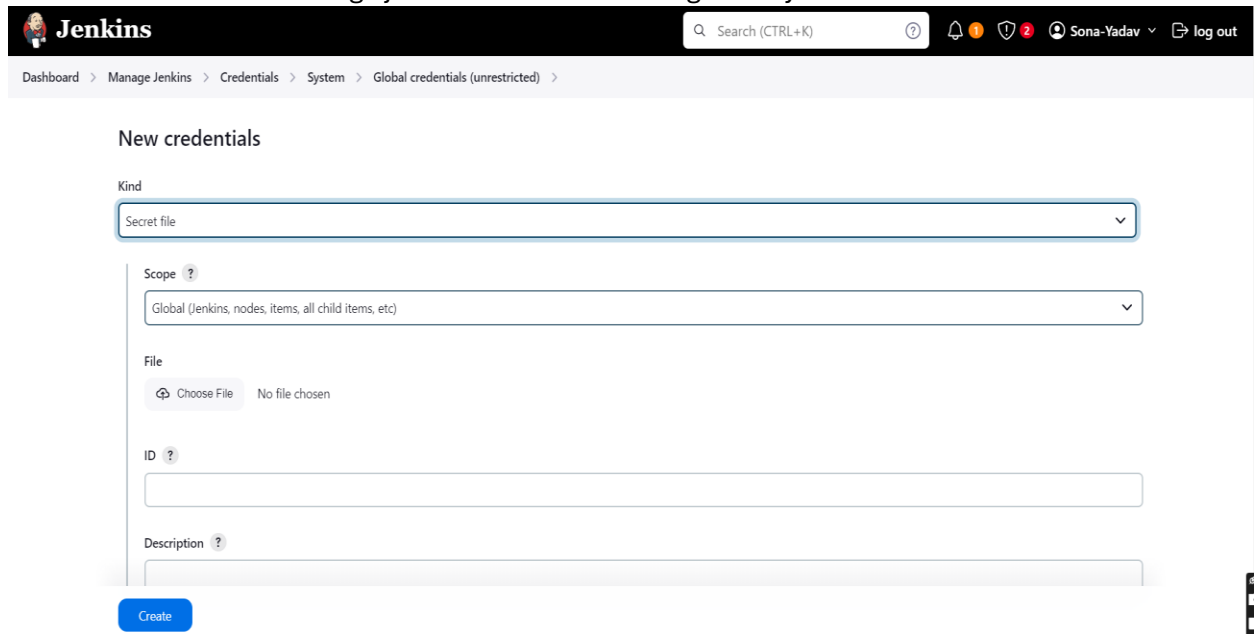
Step 6.3 Create Sonar token for Jenkins

Sonar Dashboard -> Administration (My profile) -> My Account -> Security -> Create token -> Save the token to some text file

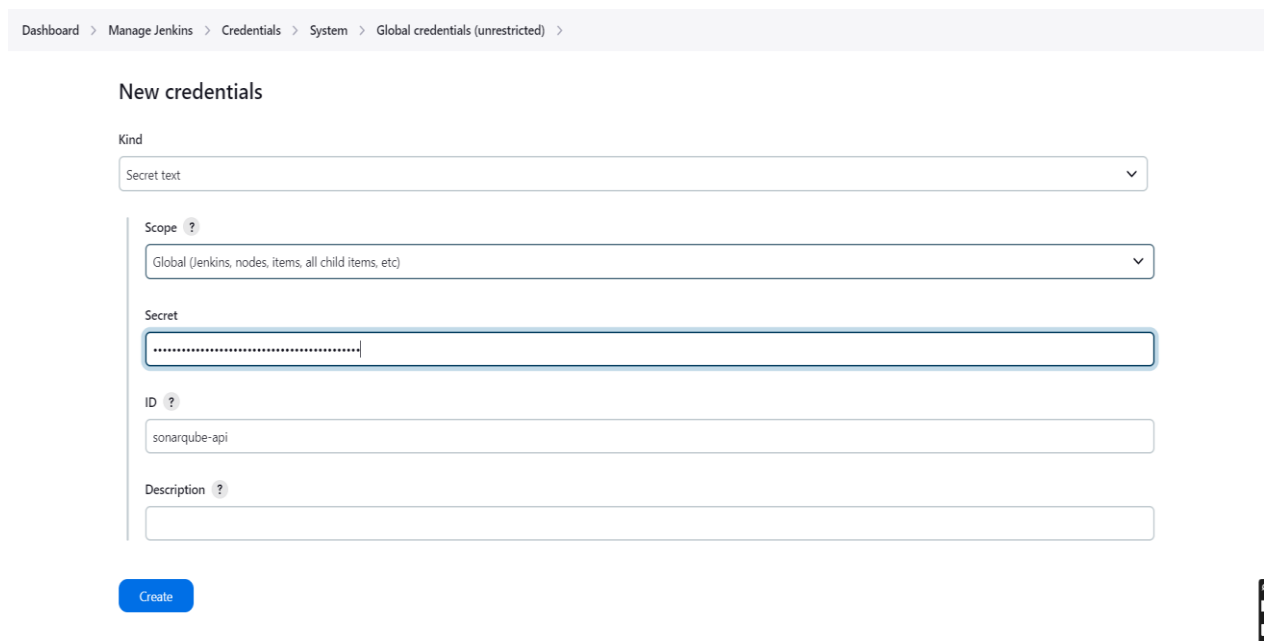


Step 6.4 Add credentials of Sonar to Jenkins

Jenkins Dashboard-> Manage jenkins-> credentials -> global system-> Add



The screenshot shows the Jenkins 'New credentials' form. The breadcrumb trail at the top is 'Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >'. The form has the following fields: 'Kind' is a dropdown menu with 'Secret file' selected; 'Scope' is a dropdown menu with 'Global (Jenkins, nodes, items, all child items, etc)' selected; 'File' is a section with a 'Choose File' button and the text 'No file chosen'; 'ID' is a text input field; 'Description' is a text input field; and a blue 'Create' button at the bottom.



The screenshot shows the Jenkins 'New credentials' form with the following fields: 'Kind' is a dropdown menu with 'Secret text' selected; 'Scope' is a dropdown menu with 'Global (Jenkins, nodes, items, all child items, etc)' selected; 'Secret' is a text input field filled with dots; 'ID' is a text input field with the value 'sonarqube-api'; 'Description' is a text input field; and a blue 'Create' button at the bottom.

Step 6.5 Integrate Sonar to Jenkins

Sonar Dashboard -> Administration -> Configuration -> webhooks -> Create-> Add the below name and url and save

http://<EC2_IP>:8080/sonarqube-webhook/

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

Administration

Configuration Security Projects System Marketplace

Webhooks

Webhooks are used to notify external services when a project analysis is done.
An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the [Webhooks documentation](#).

Create

Name	URL	Has secret?	Last delivery	Actions
Jenkins_Pipeline	http://3.110.62.23:8080/sonarqube-webhook/	No	Never	⋮

7. Jenkins Dashboard -> Manage Jenkins -> configure system

Step 7.1 Inside System-> scroll down-> ADD SONARQUBE

Click on sonarqube servers -> add URL and name -> Click on add token -> Select Secret text -> Add the sonar token from step10.3 -> Give name of token as sonarqube-api

Dashboard > Manage Jenkins > System >

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

☐ Environment variables

SonarQube installations

List of SonarQube installations

Name

sonar-api

Server URL

Default is http://localhost:9000

http://3.110.62.23:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonarqube-api

- none -

sonarqube-api

Advanced

Save Apply

8. Add the docker HUB credentials ID

Jenkins dashboard -> Manage Jenkins -> Credentials -> System -> click on global credentials

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

New credentials

Kind

Username with password

Scope

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

Username

sonayadav978

☐ Treat username as secret

Password

ID

docker

Description

Create

9. Install Maven

<https://github.com/Sona-Yadav/jenkin-Project-/blob/main/maven.sh>

10. Install TRIVY for docker image scan

<https://github.com/Sona-Yadav/jenkin-Project-/blob/main/trivy.sh>

11. Add the Jenkins Shared library

Go to Manage Jenkins -> Configure system -> Global pipeline library -> Add below data

Name - my-shared-library

Default version – main

Git -https://github.com/praveen1994dec/jenkins_shared_lib.git

Dashboard > Manage Jenkins > System >

Global Trusted Pipeline Libraries

Sharable libraries available to any Pipeline jobs running on this system. These libraries will be trusted, meaning they run without "sandbox" restrictions and may use @Grab.

Library

Name ?

my-shared-library

Default version ?

main

☐ Load implicitly ?

☒ Allow default version to be overridden ?

☒ Include @Library changes in job recent changes ?

☐ Cache fetched versions on controller for quick retrieval ?

Retrieval method

Save

Apply

12. Create a pipeline Job

Jenkins dashboard-> new item-> name-> select pipeline->Ok


Dashboard > All > New Item


New Item


Enter an item name


Jenkin-CI-Project


Select an item type

 **Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

 **Maven project**
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

 **Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different

OK

Add pipeline script as SCM

Dashboard > Jenkin-CI-Project > Configuration

Configure

General

Advanced Project Options

Pipeline

Pipeline

Definition

Pipeline script from SCM

Pipeline script

Pipeline script from SCM

None

Script Path ?

Jenkinsfile

☒ Lightweight checkout ?

Pipeline Syntax

Save Apply

Dashboard > Jenkin-CI-Project > Configuration

Configure

General

Advanced Project Options

Pipeline

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/Sona-Yadav/java_app_3.0.git

Credentials ?

- none -

+ Add

Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

Save Apply

13. Once pipeline is Run Check

- The Jenkins logs

• Sonarqube dashboard for report

