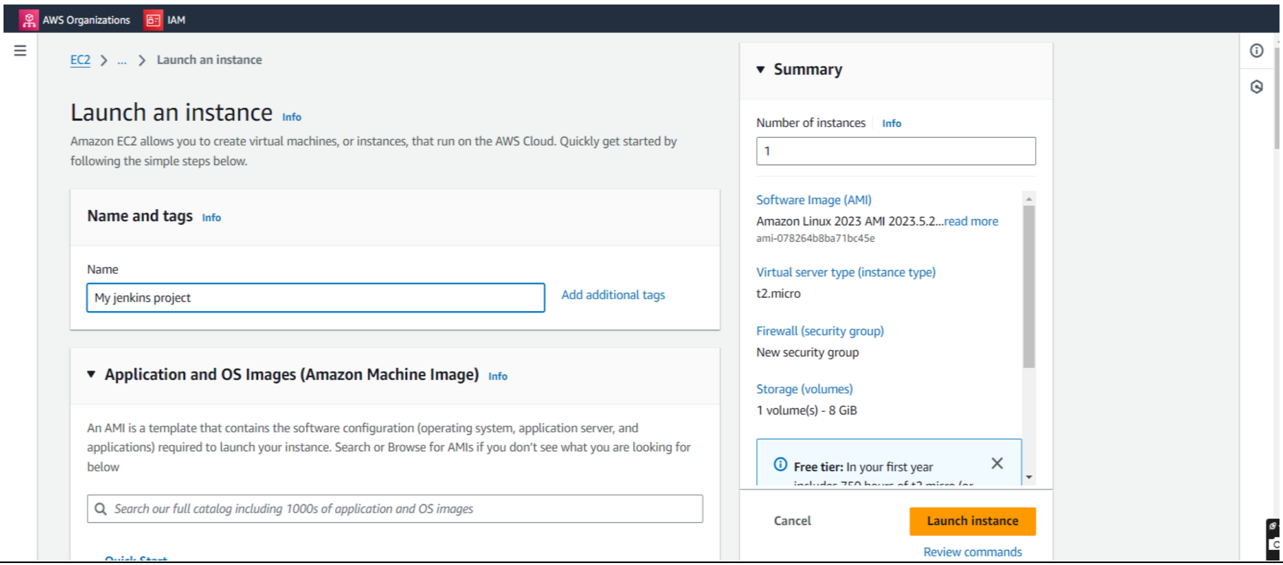
Jenkins CI/CD Project

1 Launch an ec2 instance ,

AMI: ubuntu

instance type: t2.medium

open port: 8080 (for jenkins), allow HTTP traffic and 9000 (for sonar).



Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: My jenkins project

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Summary

Number of instances Info: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.5.2...read more ami-078264b8ba71bc45e

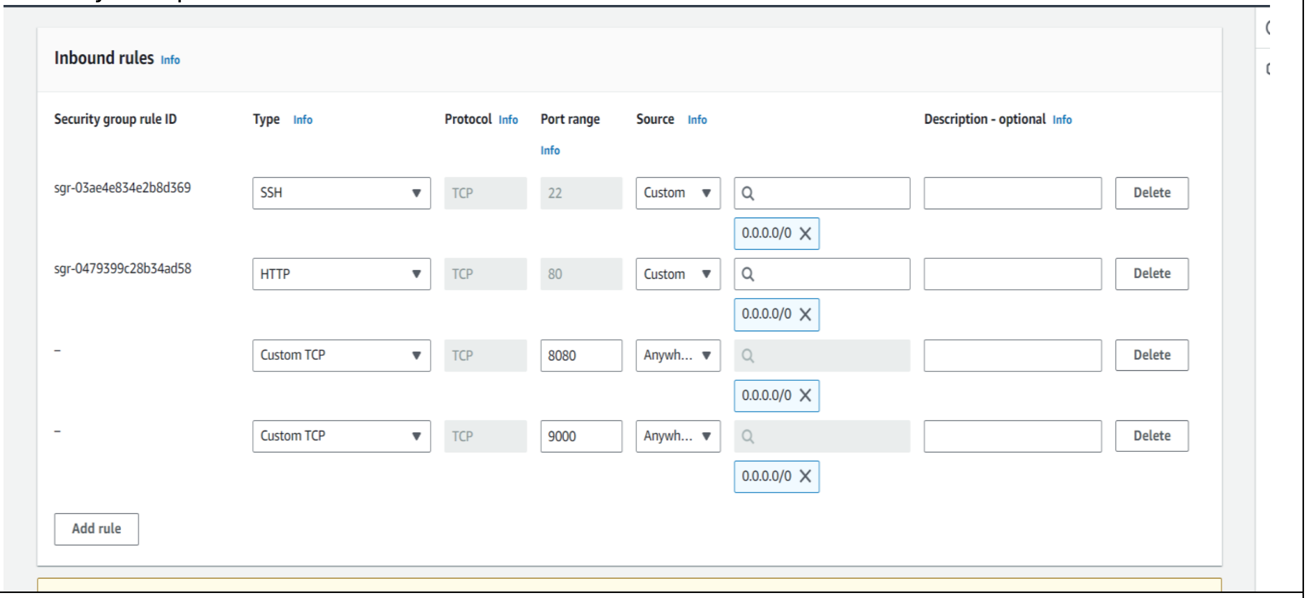
Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year

Security Group:



Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
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

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:

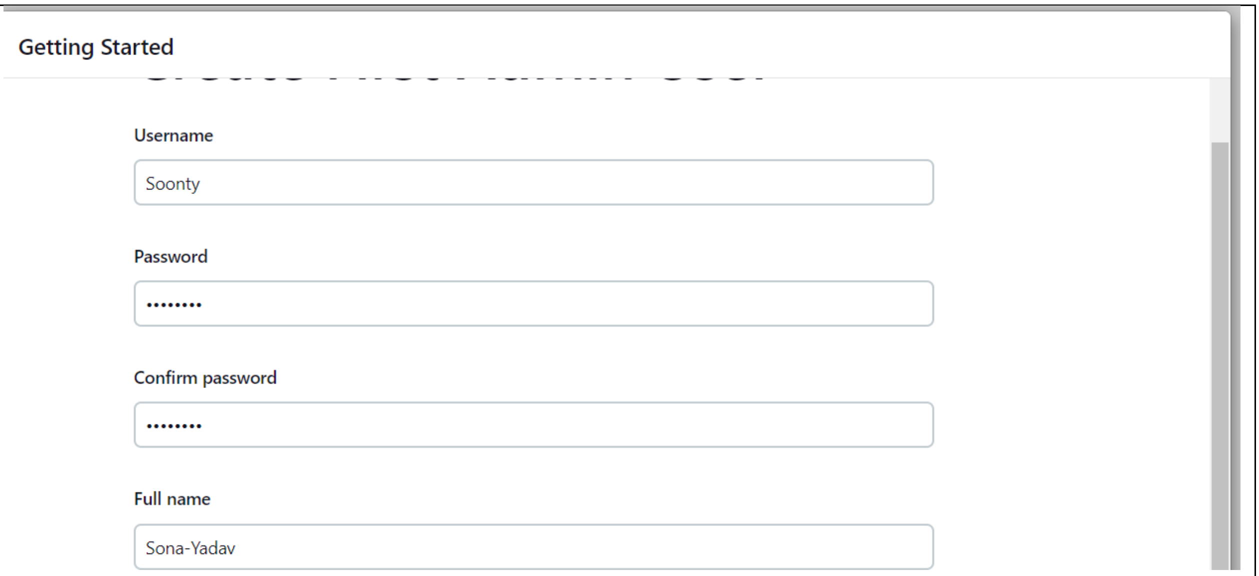
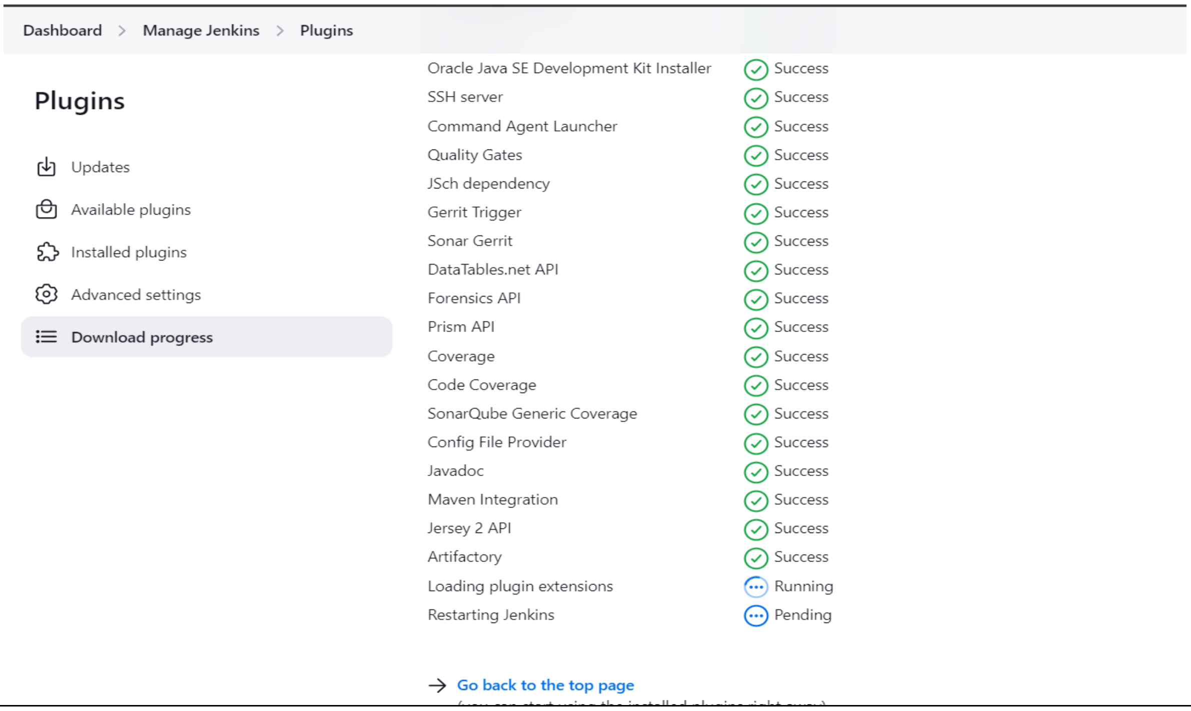
http://<EC2_PUBLIC _IP>:8080/

I. Get the Administrator password by hitting the below command in EC2 server.

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

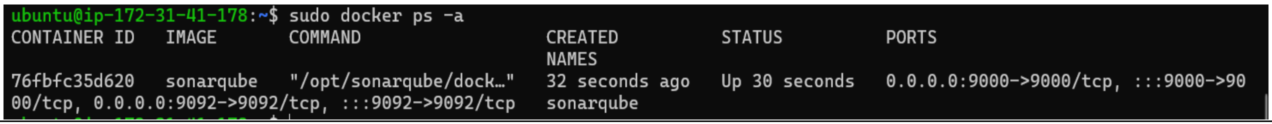
II. Install all suggested plugins

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]

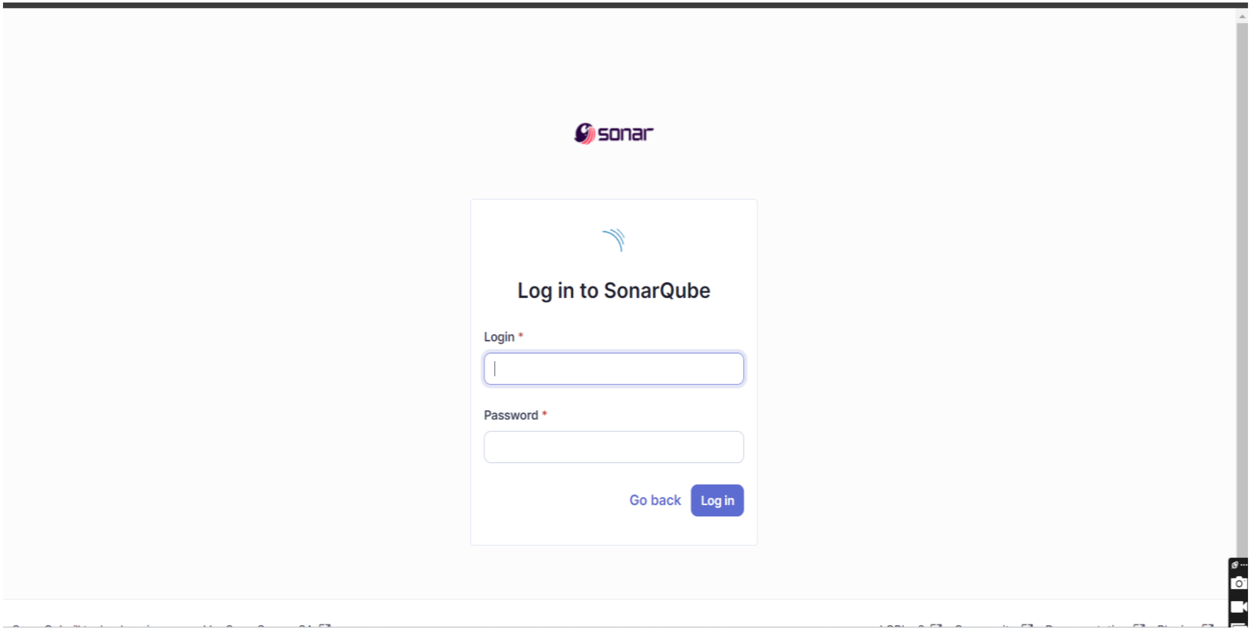


```
ubuntu@ip-172-31-41-178:~$ sudo docker ps -a
CONTAINER ID   IMAGE       COMMAND                  CREATED          STATUS          PORTS
                                                    NAMES
76fbfc35d620   sonarqube   "/opt/sonarqube/dock…"   32 seconds ago   Up 30 seconds   0.0.0.0:9000->9000/tcp, :::9000->90
00/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



Log in to SonarQube

Login *

Password *

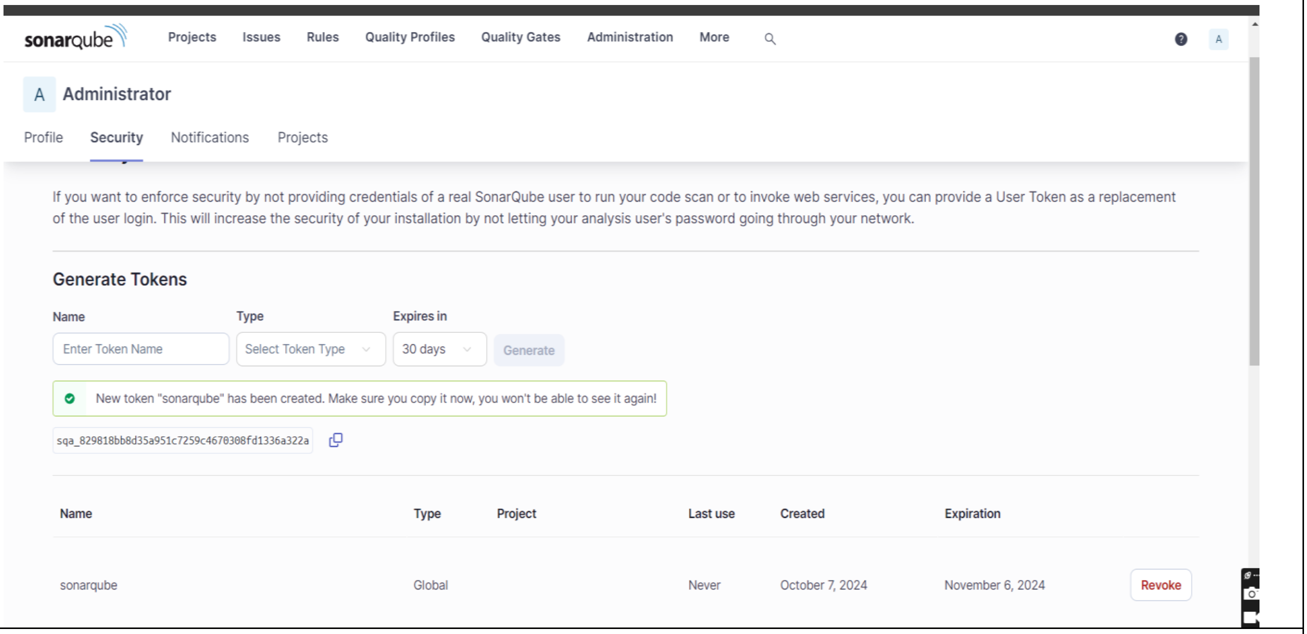
For the first time, use the following credentials for login:

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



A Administrator

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

Generate Tokens

Name: Enter Token Name

Type: Select Token Type

Expires in: 30 days

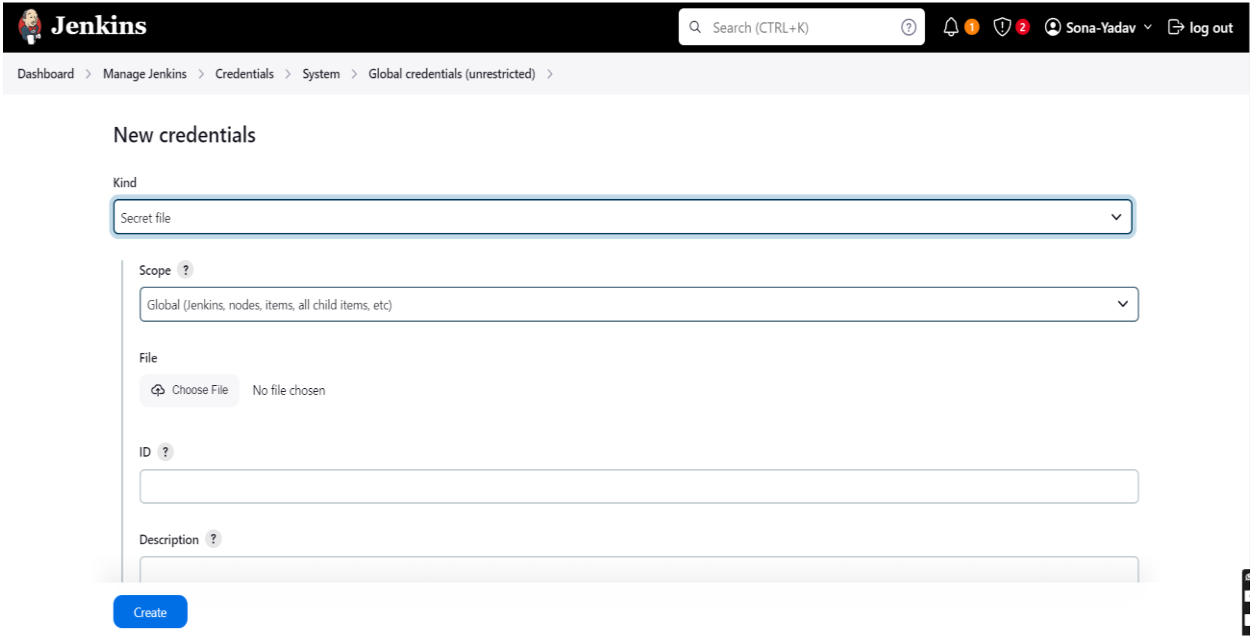
New token "sonarqube" has been created. Make sure you copy it now, you won't be able to see it again!

sqa_829818bb8d35a951c7259c4670308fd1336a322a

Name	Type	Project	Last use	Created	Expiration	
sonarqube	Global		Never	October 7, 2024	November 6, 2024	Revoke

Step 6.4 Add credentials of Sonar to Jenkins

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



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

New credentials

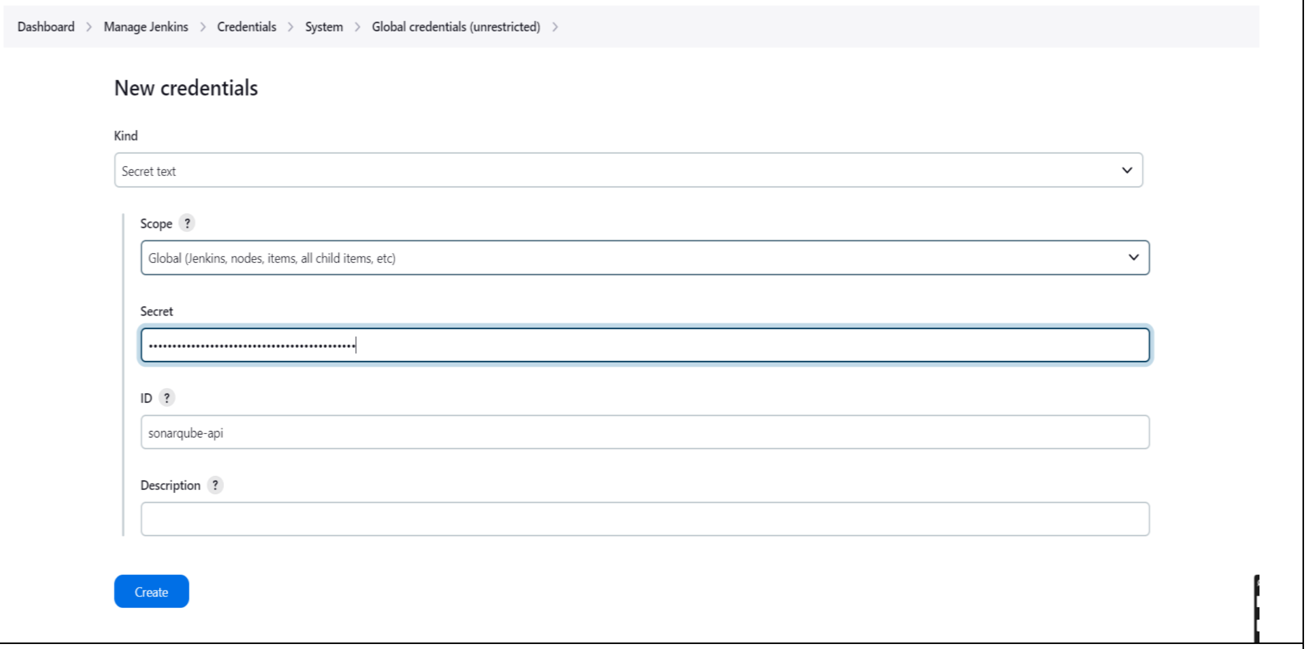
Kind: Secret file

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

File: Choose File No file chosen

ID:

Description:



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

New credentials

Kind: Secret text

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

Secret: ••

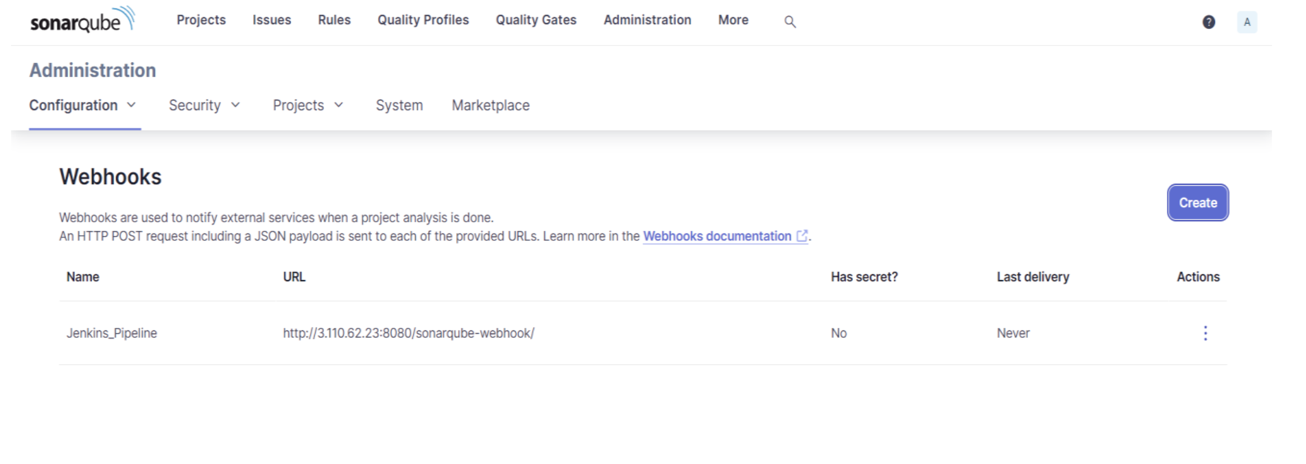
ID: sonarqube-api

Description:

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/



Administration

Configuration ⌄ Security ⌄ Projects ⌄ System Marketplace

Webhooks

Create

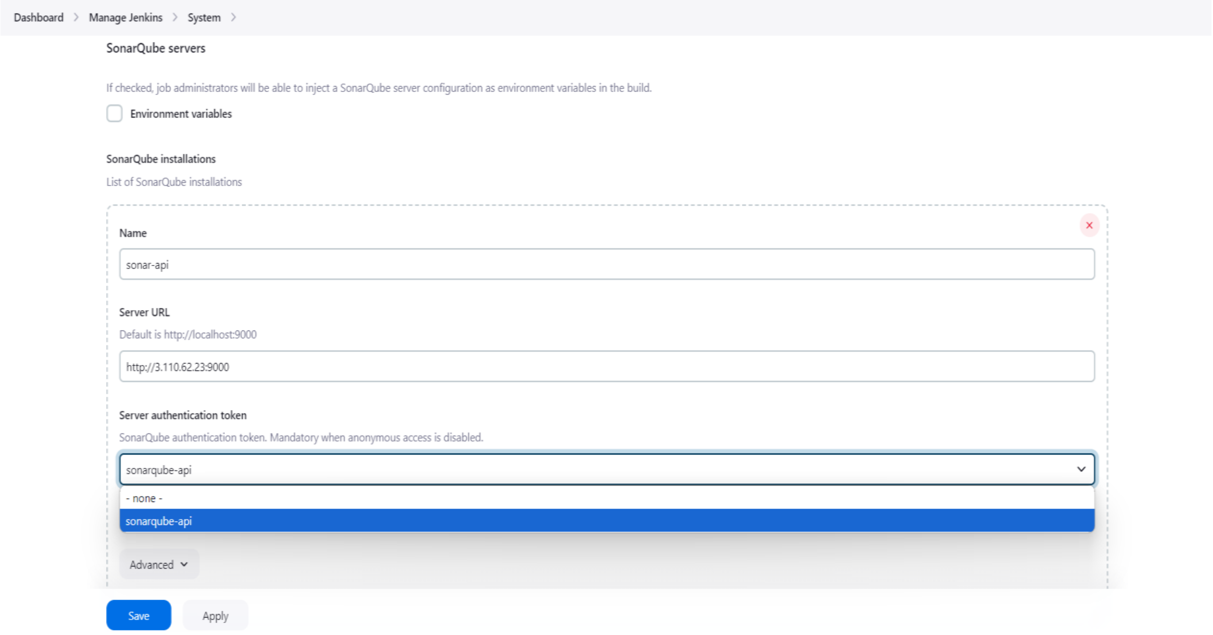
Webhooks are used to notify external services when a project analysis is done.
An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the Webhooks documentation.

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



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

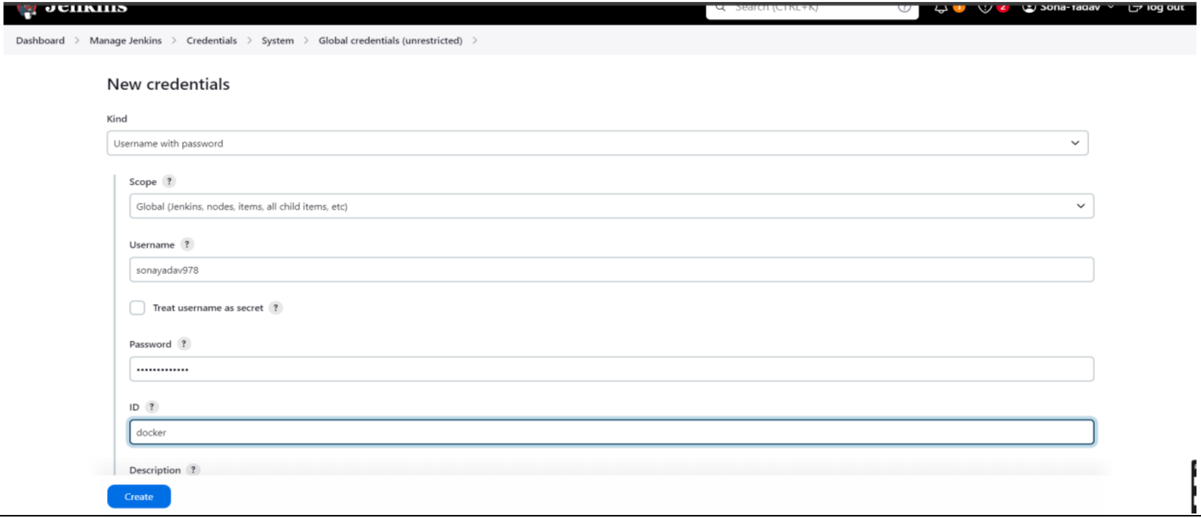
Name
sonar-api

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

8. Add the docker HUB credentials ID

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



Jenkins
Search (CTRL+K)
sona-yadav
log out
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

9. Install Maven

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

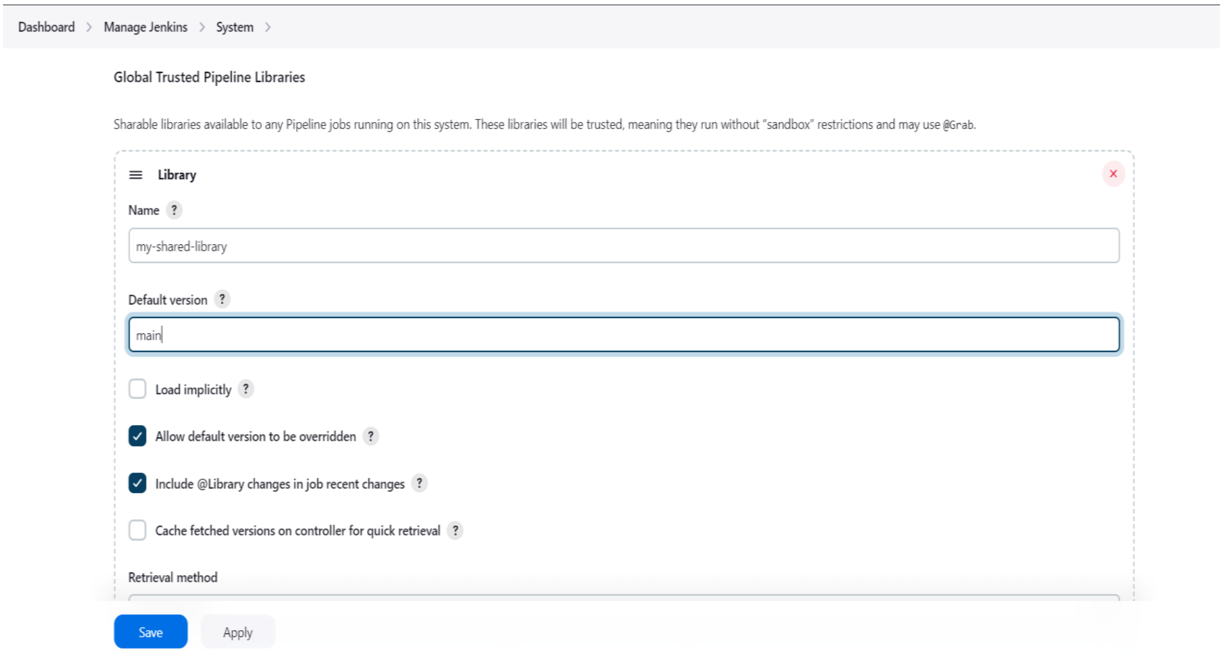
10. Install TRIVY for docker image scan

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

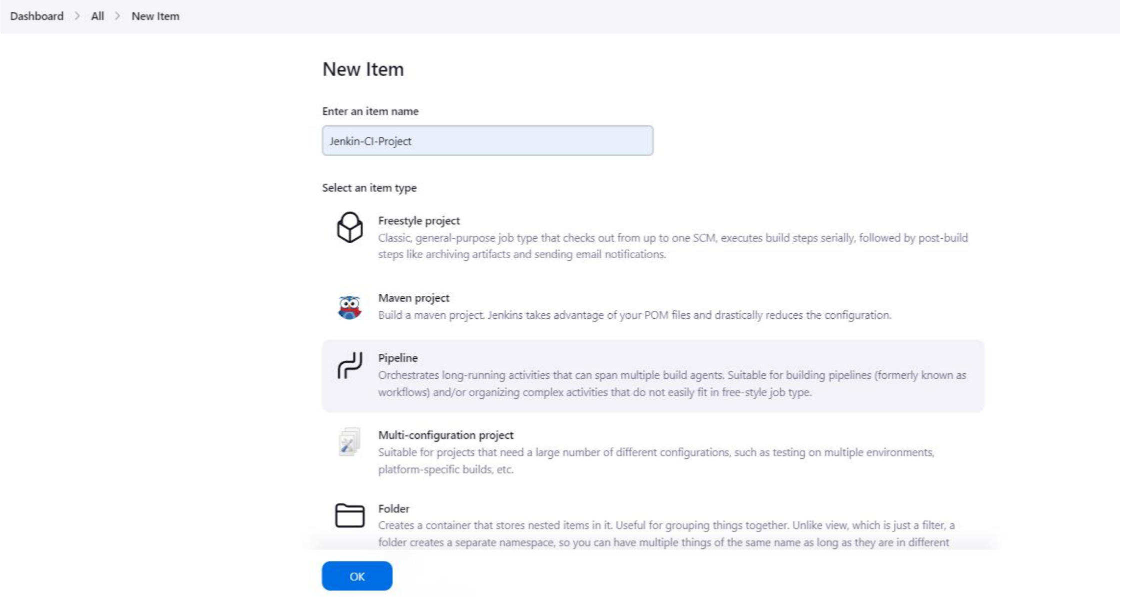
☑ Include @Library changes in job recent changes

□ Cache fetched versions on controller for quick retrieval

Retrieval method

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.

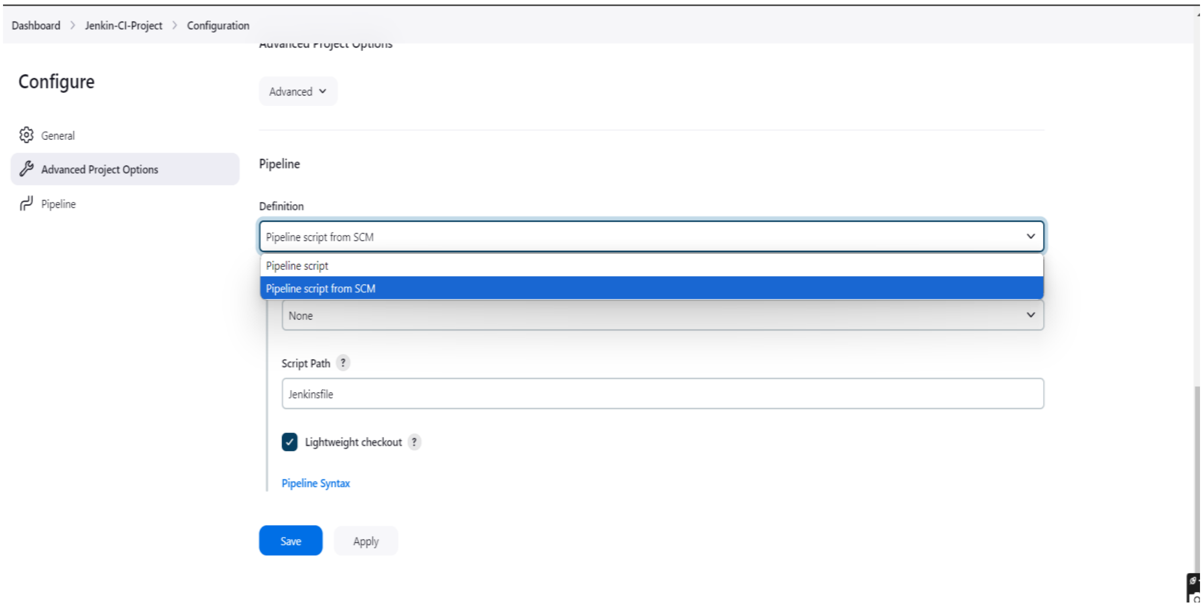
Maven project — Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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

Add pipeline script as SCM



Dashboard > Jenkin-CI-Project > Configuration

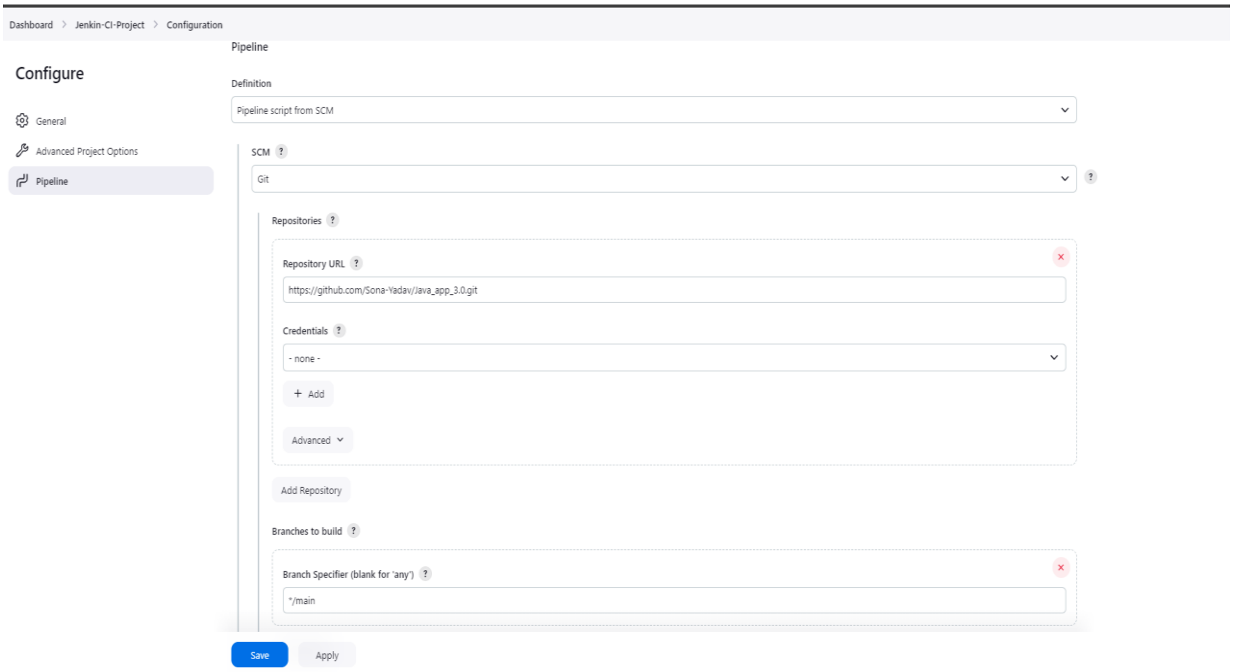
Configure

Pipeline

Definition: Pipeline script from SCM

Script Path: Jenkinsfile

☑ Lightweight checkout



Dashboard > Jenkin-CI-Project > Configuration

Configure

Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/Sona-Yadav/Java_app_3.0.git

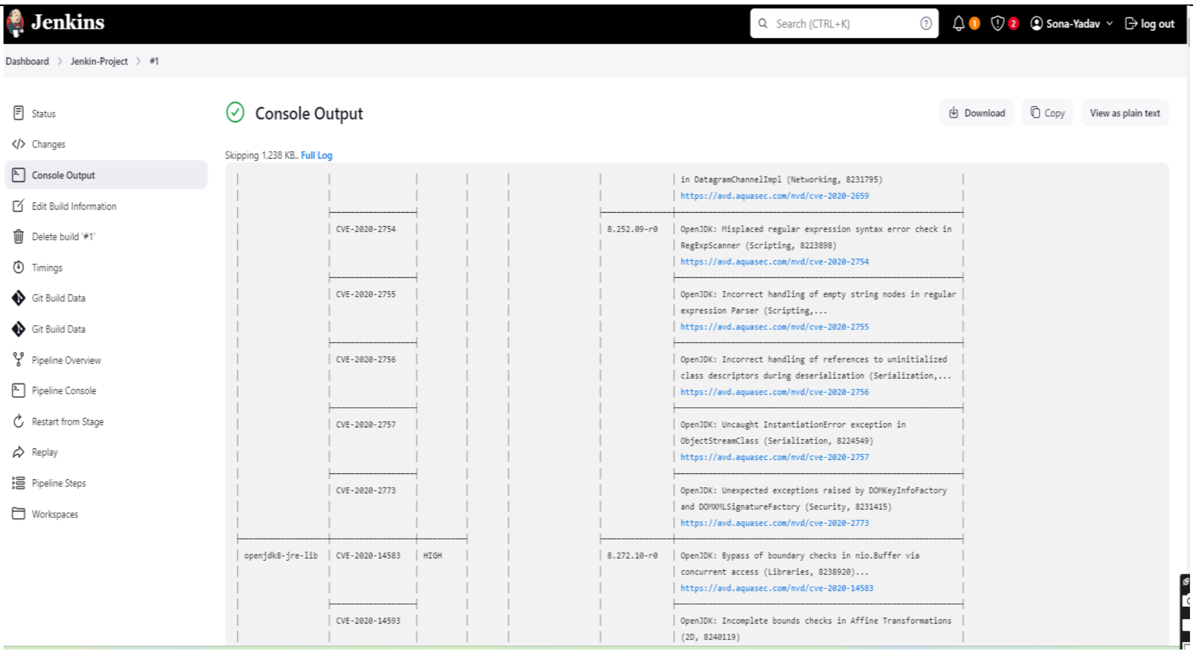
Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */main

13. Once pipeline is Run Check

- The Jenkins logs

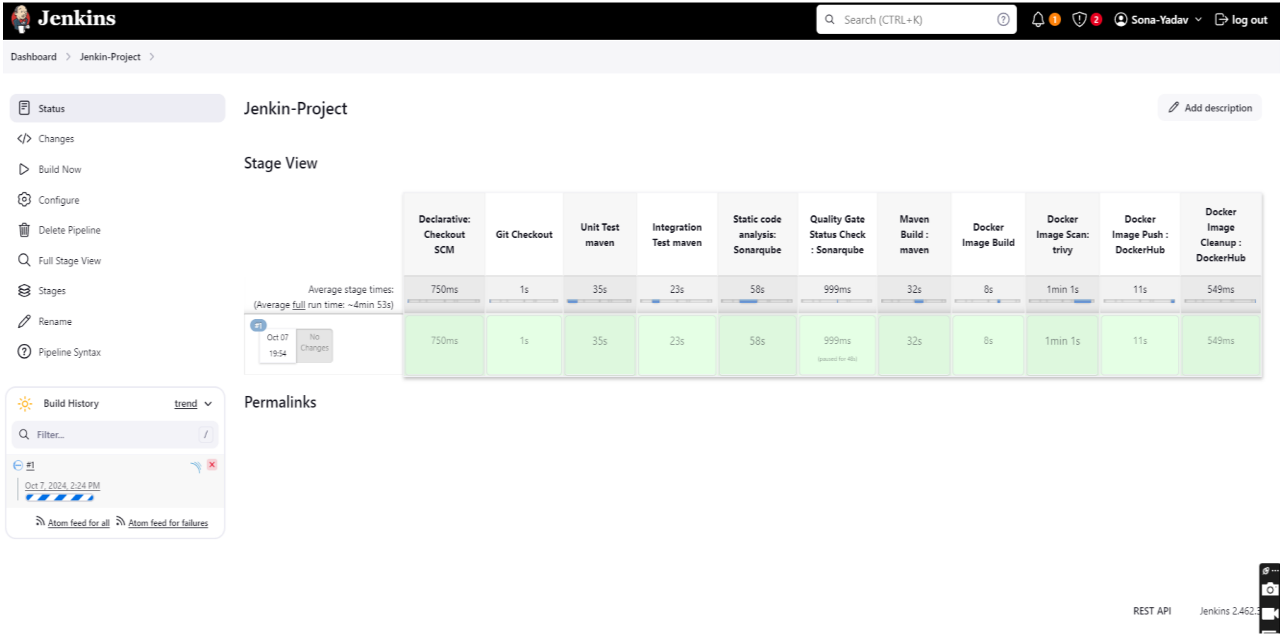


Console Output

Skipping 1,238 KB.. Full Log

```
|                 |                  |          |     |               |                 | in DatagramChannelImpl (Networking, 8231795)                 |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-2659                    |
|                 ├──────────────────┤          |     |               ├─────────────────┼──────────────────────────────────────────────────────────────┤
|                 | CVE-2020-2754    |          |     |               | 8.252.09-r0     | OpenJDK: Misplaced regular expression syntax error check in  |
|                 |                  |          |     |               |                 | RegExpScanner (Scripting, 8223898)                           |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-2754                    |
|                 ├──────────────────┤          |     |               |                 ├──────────────────────────────────────────────────────────────┤
|                 | CVE-2020-2755    |          |     |               |                 | OpenJDK: Incorrect handling of empty string nodes in regular |
|                 |                  |          |     |               |                 | expression Parser (Scripting,...                             |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-2755                    |
|                 ├──────────────────┤          |     |               |                 ├──────────────────────────────────────────────────────────────┤
|                 | CVE-2020-2756    |          |     |               |                 | OpenJDK: Incorrect handling of references to uninitialized   |
|                 |                  |          |     |               |                 | class descriptors during deserialization (Serialization,...  |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-2756                    |
|                 ├──────────────────┤          |     |               |                 ├──────────────────────────────────────────────────────────────┤
|                 | CVE-2020-2757    |          |     |               |                 | OpenJDK: Uncaught InstantiationError exception in            |
|                 |                  |          |     |               |                 | ObjectStreamClass (Serialization, 8224549)                   |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-2757                    |
|                 ├──────────────────┤          |     |               |                 ├──────────────────────────────────────────────────────────────┤
|                 | CVE-2020-2773    |          |     |               |                 | OpenJDK: Unexpected exceptions raised by DOMKeyInfoFactory   |
|                 |                  |          |     |               |                 | and DOMXMLSignatureFactory (Security, 8231415)               |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-2773                    |
├─────────────────┼──────────────────┼──────────┤     |               ├─────────────────┼──────────────────────────────────────────────────────────────┤
| openjdk8-jre-lib| CVE-2020-14583   | HIGH     |     |               | 8.272.10-r0     | OpenJDK: Bypass of boundary checks in nio.Buffer via         |
|                 |                  |          |     |               |                 | concurrent access (Libraries, 8238920)...                    |
|                 |                  |          |     |               |                 | https://avd.aquasec.com/nvd/cve-2020-14583                   |
|                 ├──────────────────┤          |     |               |                 ├──────────────────────────────────────────────────────────────┤
|                 | CVE-2020-14593   |          |     |               |                 | OpenJDK: Incomplete bounds checks in Affine Transformations  |
|                 |                  |          |     |               |                 | (2D, 8240119)                                                |
```

- Jenkins pipeline overview



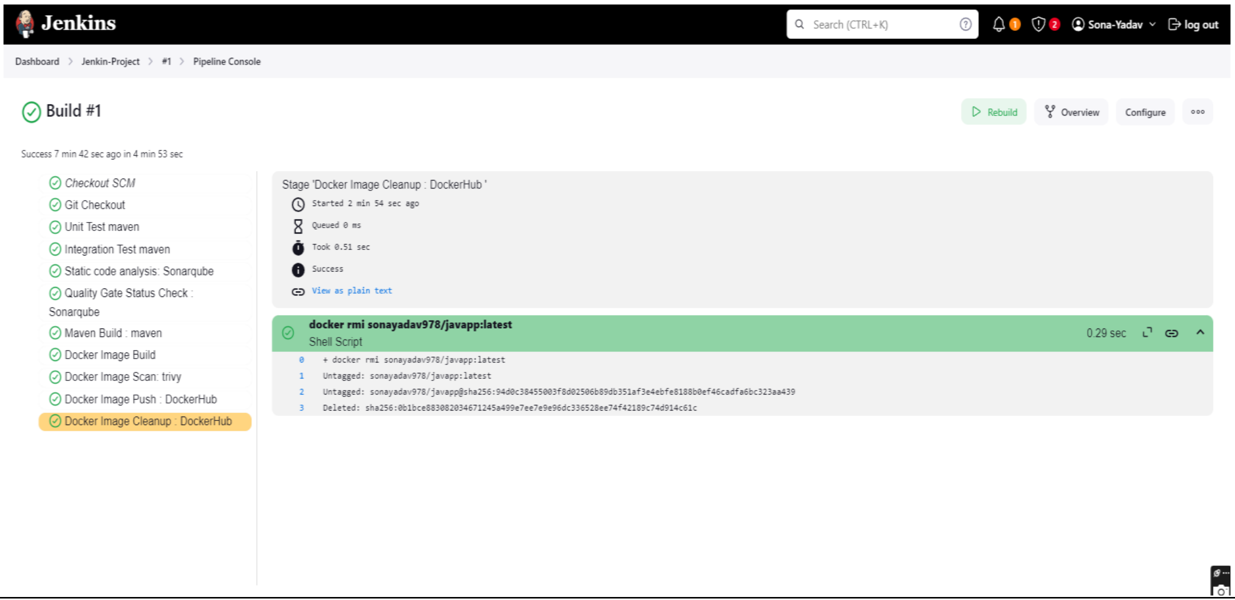
Jenkin-Project

Stage View

	Declarative: Checkout SCM	Git Checkout	Unit Test maven	Integration Test maven	Static code analysis: Sonarqube	Quality Gate Status Check : Sonarqube	Maven Build : maven	Docker Image Build	Docker Image Scan: trivy	Docker Image Push : DockerHub	Docker Image Cleanup : DockerHub
Average stage times: (Average full run time: ~4min 53s)	750ms	1s	35s	23s	58s	999ms	32s	8s	1min 1s	11s	549ms
#1 Oct 07 19:54 No Changes	750ms	1s	35s	23s	58s	999ms (paused for 48s)	32s	8s	1min 1s	11s	549ms

Permalinks

- Jenkins console



Build #1

Success 7 min 42 sec ago in 4 min 53 sec

- Checkout SCM
- Git Checkout
- Unit Test maven
- Integration Test maven
- Static code analysis : Sonarqube
- Quality Gate Status Check : Sonarqube
- Maven Build : maven
- Docker Image Build
- Docker Image Scan : trivy
- Docker Image Push : DockerHub
- Docker Image Cleanup : DockerHub

Stage 'Docker Image Cleanup : DockerHub '

Started 2 min 54 sec ago

Queued 0 ms

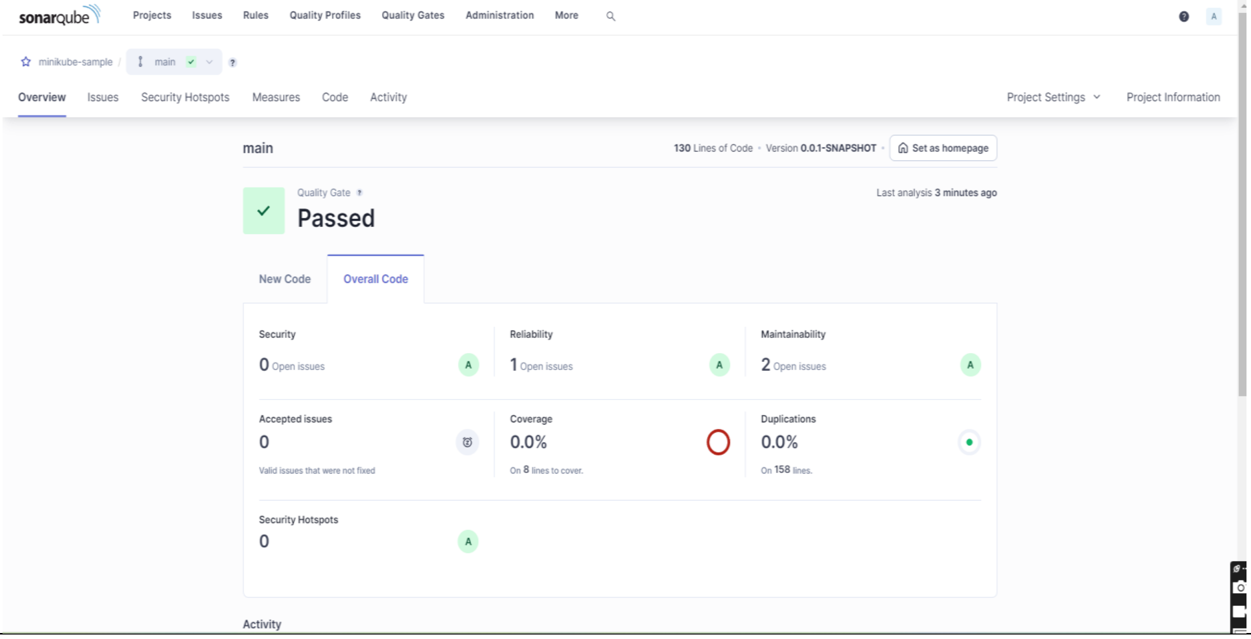
Took 0.51 sec

Success

View as plain text

docker rmi sonayadav978/javapp:latest
Shell Script
0.29 sec

```
+ docker rmi sonayadav978/javapp:latest
Untagged: sonayadav978/javapp:latest
Untagged: sonayadav978/javapp@sha256:94d0c38455003f8d02506b89db351af3e4ebfe8188b0ef46cadfa6bc323aa439
Deleted: sha256:0b1bce883082034671245a499e7ee7e9e96dc336528ee74f42189c74d914c61c
```

	• Sonarqube dashboard for report 

main

130 Lines of Code · Version 0.0.1-SNAPSHOT

Quality Gate

Passed

Last analysis 3 minutes ago

Security	Reliability	Maintainability
0 Open issues A	1 Open issues A	2 Open issues A

Accepted issues	Coverage	Duplications
0	0.0%	0.0%
Valid issues that were not fixed	On 8 lines to cover.	On 158 lines.

Security Hotspots

0 A

Activity