

# Continuous Integration with Simple Code Analysis

- **Concepts Used:** Jenkins, AWS Cloud9, and SonarQube.
- **Problem Statement:** "Set up a Jenkins pipeline using AWS Cloud9 to perform a simple code analysis on a JavaScript file using SonarQube."
- **Tasks:**
  - Create a Jenkins job using AWS Cloud9.
  - Configure the job to integrate with SonarQube for basic code analysis.
  - Run the Jenkins job with a JavaScript file and review the analysis report.

## 1. Introduction

### Case Study Overview

This case study focuses on implementing a continuous integration (CI) pipeline using Jenkins and SonarQube on an Amazon EC2 instance. Due to access issues with AWS Cloud9, we opted to set up the entire development environment directly on an EC2 instance. The goal is to automate the code analysis process for JavaScript files, enabling developers to identify potential issues early in the development cycle. Continuous integration enhances software quality by ensuring that changes to the codebase are tested and integrated regularly.

### Importance of Continuous Integration

Continuous Integration streamlines the development process by encouraging developers to commit their changes regularly. The advantages of CI include:

- **Early Detection of Errors:** Frequent integration leads to early detection of bugs, reducing the cost and effort needed for fixes.
- **Improved Software Quality:** Automated testing and code analysis contribute to higher code quality and maintainability.
- **Faster Feedback Loop:** Developers receive immediate feedback on their code, enabling quicker iterations and adjustments.
- **Enhanced Collaboration:** CI fosters better communication among team members by ensuring that everyone is working with the latest code.

## 2 . Key Feature and Application

The primary feature of this case study is the integration of Jenkins with SonarQube for automated code analysis. By utilizing this CI pipeline, developers can receive immediate feedback on code quality, security vulnerabilities, and code smells. This practical application helps teams maintain high code standards and facilitates quicker identification and resolution of issues, ultimately improving overall project delivery.

**Automated Quality Checks:** SonarQube analyzes the codebase and reports on issues such as bugs, vulnerabilities, and code smells, providing developers with actionable insights.

**Customizable Quality Gates:** Teams can define quality gates that must be met before code changes can be merged, ensuring adherence to coding standards.

**Visual Reports:** SonarQube offers dashboards that visualize code quality metrics, making it easier for teams to monitor and improve their codebase over time.

## 3 . Third-Year Project Integration

This case study can be linked to third-year project by incorporating automated code analysis into your existing development practices. By adopting CI/CD principles, we can improve the maintainability and reliability of your project, ultimately aligning it with industry standards.

To incorporate this case study into your third-year project's workflow:

1. **Set Up Jenkins and SonarQube:** On an EC2 instance or local environment, set up Jenkins and SonarQube as demonstrated in your case study.
2. **GitHub Integration:** Configure Jenkins to monitor your project's GitHub repository. Each time a new feature or bug fix is pushed, the CI pipeline will automatically trigger.
3. **Code Analysis as a Standard Practice:** Enforce regular code reviews with SonarQube analysis before merging any code changes into the main project branch.
4. **Documentation:** Document this integration in your project report, emphasizing the improvements in code quality, reduced manual testing, and faster feedback cycles.

## 4 . Tools Used

### SAST Overview

**SAST (Static Application Security Testing)** is a method of security testing that analyzes the source code of an application to identify vulnerabilities and security issues without executing the program. It involves examining the code, configurations, and dependencies early in the development process to detect issues like SQL injection, cross-site scripting (XSS), buffer overflows, and insecure coding practices.

- **Early Detection:** SAST identifies vulnerabilities before the code is compiled or executed, allowing developers to fix issues earlier in the development lifecycle.
- **White-box Testing:** It has full access to the application's source code and inspects it for security flaws.
- **Automated Analysis:** Tools scan the codebase, providing detailed reports on potential issues such as security vulnerabilities, coding errors, and adherence to best practices.

SAST is commonly integrated into CI/CD pipelines to ensure continuous security checks and improve code quality.

## Jenkins Overview

Jenkins is an open-source automation server widely used for Continuous Integration and Continuous Delivery (CI/CD). It provides a framework for building, testing, and deploying applications. Some key features of Jenkins include:

- **Extensibility:** Jenkins supports a wide range of plugins, allowing it to integrate with various tools and technologies.
- **Distributed Builds:** Jenkins can distribute build tasks across multiple machines, improving performance and scalability.
- **User-Friendly Interface:** Its web-based interface makes it easy for users to configure jobs, monitor builds, and view results.

## SonarQube Overview

SonarQube is an open-source platform for continuous inspection of code quality. It performs automatic reviews of code to detect bugs, code smells, and security vulnerabilities. Key aspects of SonarQube include:

- **Multi-Language Support:** SonarQube supports multiple programming languages, making it versatile for diverse projects.
- **Quality Gates:** Teams can define quality gates that evaluate the code based on metrics like code coverage, duplications, and maintainability.
- **Integration Capabilities:** SonarQube can be easily integrated with CI tools like Jenkins to automate code analysis during the build process.

## Step-by-step Explanation

### Step 1: Initial Setup and Configuration

#### 1. Set Up EC2 Instances for jenkins and sonarqube:

- Launch a new EC2 instance using Amazon Linux 2.
- Ensure the instance has the necessary security groups to allow inbound traffic on relevant ports (e.g., 8080 for Jenkins, 9000 for SonarQube).

EC2 instance for Sonarqube with configuration :

(Os image : Amazon linux 2, Instance type : T2.small, Allow SSH traffic from : My ip , also select or create key-pair)

EC2 > ... > Launch an instance

### 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

 [Add additional tags](#)

#### ▼ 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

Recents **Quick Start**

Amazon macOS Ubuntu Windows Red Hat SUSE Li

#### ▼ Summary

Number of instances [Info](#)

Software Image (AMI)

Amazon Linux 2023 AMI 2023.6.2...[read more](#)  
ami-06b21ccaeff8cd686

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 includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance

Cancel **Launch instance**

[Preview code](#)

#### ▼ 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

Recents **Quick Start**

Amazon Linux  
aws

macOS  
Mac

Ubuntu  
ubuntu

Windows  
Microsoft

Red Hat  
Red Hat

SUSE Li  
SUSE

[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

#### Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) – Kernel 5.10, SSD Volume Type  
ami-0ddc798b3f1a5117e (64-bit (x86)) / ami-05f16f3539e999b77 (64-bit (Arm))  
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible ▼

#### Description

Amazon Linux 2 comes with five years support. It provides Linux kernel 5.10 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard.

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t2.small  
Family: t2 1 vCPU 2 GiB Memory Current generation: true  
On-Demand Windows base pricing: 0.032 USD per Hour  
On-Demand Linux base pricing: 0.023 USD per Hour  
On-Demand RHEL base pricing: 0.0376 USD per Hour  
On-Demand SUSE base pricing: 0.053 USD per Hour

▼

☐ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

sonar-key ▼

[Create new key pair](#)

Network [Info](#)

vpc-014f21aed2a85fb2e

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of [free tier allowance](#)

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

We'll create a new security group called 'launch-wizard-2' with the following rules:

☒ Allow SSH traffic from  
Helps you connect to your instance

My IP  
152.58.51.21/32 ▼

☐ Allow HTTPS traffic from the internet  
To set up an endpoint, for example when creating a web server

☐ Allow HTTP traffic from the internet  
To set up an endpoint, for example when creating a web server

EC2 > ... > Launch an instance

✓ Success

Successfully initiated launch of instance ([i-06bc7734629d9416a](#))

EC2 instance for Jenkins with configuration :

(os type : Amazon linux 2, Instance type : T2.micro, Allow SSH traffic from : My ip)

[EC2](#) > ... > Launch an instance

## 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

[Add additional tags](#)

### ▼ 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

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

Recents

Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Li

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type  
 ami-0ddc798b3f1a5117e (64-bit (x86)) / ami-05f16f3539e999b77 (64-bit (Arm))  
 Virtualization: hvm    ENA enabled: true    Root device type: ebs

Free tier eligible ▼

**Description**

Amazon Linux 2 comes with five years support. It provides Linux kernel 5.10 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard.

### Create key pair

**Key pair name**  
 Key pairs allow you to connect to your instance securely.

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**

☒ **RSA**  
 RSA encrypted private and public key pair

☐ **ED25519**  
 ED25519 encrypted private and public key pair

**Private key file format**

☒ **.pem**  
 For use with OpenSSH

☐ **.ppk**  
 For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

[Cancel](#)
[Create key pair](#)

▼ Instance type

[Info](#) | [Get advice](#)

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true

On-Demand Windows base pricing: 0.0162 USD per Hour

On-Demand SUSE base pricing: 0.0116 USD per Hour

On-Demand RHEL base pricing: 0.026 USD per Hour

On-Demand Linux base pricing: 0.0116 USD per Hour

Free tier eligible

☒ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

▼ Key pair (login)

[Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

[Create new key pair](#)

▼ Network settings

[Info](#)

Edit

Network

[Info](#)

vpc-014f21aed2a85fb2e

Subnet

[Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP

[Info](#)

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups)

[Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

We'll create a new security group called 'launch-wizard-4' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

My IP

152.58.51.40/32

☐ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

☐ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

EC2

>

...

>

Launch an instance

✓ Success

Successfully initiated launch of instance (i-08cfc48c1abf69f57)

Instances (1/2)

[Info](#)

Last updated about 1 hour ago

Refresh

Connect

Instance state ▼

Actions ▼

Launch instances ▼

Find Instance by attribute or tag (case-sensitive)

All states ▼

Instance state = running

Clear filters

< 1 >

Settings

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
<input type="checkbox"/>	komaJenkins	i-08cfc48c1abf69f57	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	us-east-1a	ec2-54-
<input checked="" type="checkbox"/>	komaSonarqube	i-06bc7734629d9416a	Running	t2.small	2/2 checks passed	<a href="#">View alarms</a>	us-east-1c	ec2-54-





Switch to root user

```
[ec2-user@ip-172-31-85-150 ~]$ sudo su -
[root@ip-172-31-85-150 ~]#
```

**Java installation : (Install java from amazon-linux-extras as we have Amazon Linux 2)**

```
[root@ip-172-31-85-150 ~]# amazon-linux-extras install java-openjdk11
Topic java-openjdk11 has end-of-support date of 2024-09-30
Installing java-11-openjdk
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-java-openjdk11 amzn2extra-kernel-5.10
17 metadata files removed
6 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core | 3.6 kB 00:00:00
amzn2extra-docker | 2.9 kB 00:00:00
amzn2extra-java-openjdk11 | 3.0 kB 00:00:00
amzn2extra-kernel-5.10 | 3.0 kB 00:00:00
(1/9): amzn2-core/2/x86_64/group_gz | 2.7 kB 00:00:00
(2/9): amzn2-core/2/x86_64/updateinfo | 983 kB 00:00:00
(3/9): amzn2extra-docker/2/x86_64/updateinfo | 20 kB 00:00:00
(4/9): amzn2extra-java-openjdk11/2/x86_64/primary_db | 174 kB 00:00:00
(5/9): amzn2extra-java-openjdk11/2/x86_64/updateinfo | 8.0 kB 00:00:00
(6/9): amzn2extra-docker/2/x86_64/primary_db | 114 kB 00:00:00
(7/9): amzn2extra-kernel-5.10/2/x86_64/updateinfo | 91 kB 00:00:00
(8/9): amzn2extra-kernel-5.10/2/x86_64/primary_db | 31 MB 00:00:00
(9/9): amzn2-core/2/x86_64/primary_db | 71 MB 00:00:01
```

```
root@ip-172-31-85-150:~
```

```
40 mock available [ =stable ]
43 livepatch available [ =stable ]
45 haproxy2 available [ =stable ]
46 collectd available [ =stable ]
47 aws-nitro-enclaves-cli available [ =stable ]
48 R4 available [ =stable ]
_ kernel-5.4 available [ =stable ]
50 selinux-ng available [ =stable ]
52 tomcat9 available [ =stable ]
53 unbound1.13 available [ =stable ]
54 †mariadb10.5 available [ =stable ]
55 kernel-5.10=latest enabled [ =stable ]
56 redis6 available [ =stable ]
58 †postgres12 available [ =stable ]
59 †postgres13 available [ =stable ]
60 mock2 available [ =stable ]
61 dnsmasq2.85 available [ =stable ]
62 kernel-5.15 available [ =stable ]
63 †postgres14 available [ =stable ]
64 firefox available [ =stable ]
65 lustre available [ =stable ]
66 †php8.1 available [ =stable ]
67 awscli1 available [ =stable ]
68 †php8.2 available [ =stable ]
69 dnsmasq available [ =stable ]
70 unbound1.17 available [ =stable ]
72 collectd-python3 available [ =stable ]
* Extra topic has reached end of support.
† Note on end-of-support. Use 'info' subcommand.
[root@ip-172-31-85-150 ~]#
```

```
[root@ip-172-31-85-150 ~]# java --version
openjdk 11.0.23 2024-04-16 LTS
OpenJDK Runtime Environment (Red_Hat-11.0.23.0.9-2.amzn2.0.1) (build 11.0.23+9-LTS)
OpenJDK 64-Bit Server VM (Red_Hat-11.0.23.0.9-2.amzn2.0.1) (build 11.0.23+9-LTS, mixed mode, sharing)
[root@ip-172-31-85-150 ~]#
```

## Sonarqube installation

wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-8.9.2.46101.zip

```
[root@ip-172-31-85-150 opt]# wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-8.9.2.46101.zip
--2024-10-23 13:16:01-- https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-8.9.2.46101.zip
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 99.84.191.71, 99.84.191.87, 99.84.191.75, ...
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)[99.84.191.71]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 276400757 (264M) [application/zip]
Saving to: 'sonarqube-8.9.2.46101.zip'

100%[=====] 276,400,757 26.4MB/s in 9.4s

2024-10-23 13:16:11 (28.0 MB/s) - 'sonarqube-8.9.2.46101.zip' saved [276400757/276400757]

[root@ip-172-31-85-150 opt]#
```

```
[root@ip-172-31-85-150 opt]# ls -l
total 269928
drwxr-xr-x 4 root root      33 Oct 14 20:16 aws
-rw-r--r-- 1 root root      697 Jul 16 12:41 index.html?prefix=Distribution%2Fsonarqube%2Fsonarqube-8.9.2.46101.zip
drwxr-xr-x 2 root root       6 Aug 16 2018 rh
-rw-r--r-- 1 root root 276400757 Feb 16 2022 sonarqube-8.9.2.46101.zip
[root@ip-172-31-85-150 opt]#
```

Unzip the zip file

```
[root@ip-172-31-85-150 opt]# unzip sonarqube-8.9.2.46101.zip
Archive: sonarqube-8.9.2.46101.zip
  creating: sonarqube-8.9.2.46101/
  creating: sonarqube-8.9.2.46101/bin/
  creating: sonarqube-8.9.2.46101/bin/jsw-license/
  inflating: sonarqube-8.9.2.46101/bin/jsw-license/LICENSE.txt
  creating: sonarqube-8.9.2.46101/bin/windows-x86-64/
  inflating: sonarqube-8.9.2.46101/bin/windows-x86-64/StopNTService.bat
  creating: sonarqube-8.9.2.46101/bin/windows-x86-64/lib/
  inflating: sonarqube-8.9.2.46101/bin/windows-x86-64/lib/wrapper.dll
  inflating: sonarqube-8.9.2.46101/bin/windows-x86-64/StartSonar.bat
  inflating: sonarqube-8.9.2.46101/bin/windows-x86-64/wrapper.exe
  inflating: sonarqube-8.9.2.46101/bin/windows-x86-64/StartNTService.bat
  creating: sonarqube-8.9.2.46101/bin/macosx-universal-64/
  creating: sonarqube-8.9.2.46101/bin/macosx-universal-64/lib/
  inflating: sonarqube-8.9.2.46101/bin/macosx-universal-64/lib/libwrapper.jnilib
  inflating: sonarqube-8.9.2.46101/bin/macosx-universal-64/wrapper
  inflating: sonarqube-8.9.2.46101/bin/macosx-universal-64/sonar.sh
  creating: sonarqube-8.9.2.46101/bin/linux-x86-64/
  creating: sonarqube-8.9.2.46101/bin/linux-x86-64/lib/
  inflating: sonarqube-8.9.2.46101/bin/linux-x86-64/lib/libwrapper.so
  inflating: sonarqube-8.9.2.46101/bin/linux-x86-64/wrapper
  inflating: sonarqube-8.9.2.46101/bin/linux-x86-64/sonar.sh
  creating: sonarqube-8.9.2.46101/extensions/
  creating: sonarqube-8.9.2.46101/extensions/jdbc-driver/
  creating: sonarqube-8.9.2.46101/extensions/jdbc-driver/oracle/
  inflating: sonarqube-8.9.2.46101/extensions/jdbc-driver/oracle/README.txt
  creating: sonarqube-8.9.2.46101/extensions/plugins/
  inflating: sonarqube-8.9.2.46101/extensions/plugins/README.txt
  inflating: sonarqube-8.9.2.46101/COPYING
  creating: sonarqube-8.9.2.46101/logs/
  inflating: sonarqube-8.9.2.46101/logs/README.txt
  creating: sonarqube-8.9.2.46101/temp/
  inflating: sonarqube-8.9.2.46101/temp/README.txt
  creating: sonarqube-8.9.2.46101/elasticsearch/
  inflating: sonarqube-8.9.2.46101/elasticsearch/README.asciidoc
  creating: sonarqube-8.9.2.46101/elasticsearch/bin/
```

Delete the zip package for no conflict.

```
[root@ip-172-31-85-150 opt]# ls -l
total 269928
drwxr-xr-x 4 root root      33 Oct 14 20:16 aws
-rw-r--r-- 1 root root      697 Jul 16 12:41 index.html?prefix=Distribution%2Fsonarqube%2Fsonarqube-8.9.2.46101.zip
drwxr-xr-x 2 root root       6 Aug 16 2018 rh
drwxr-xr-x 11 root root    141 Jul 27 2021 sonarqube-8.9.2.46101
-rw-r--r-- 1 root root 276400757 Feb 16 2022 sonarqube-8.9.2.46101.zip
[root@ip-172-31-85-150 opt]# rm -f index.html?prefix=Distribution%2Fsonarqube%2Fsonarqube-8.9.2.46101.zip
[root@ip-172-31-85-150 opt]# rm -f sonarqube-8.9.2.46101.zip
[root@ip-172-31-85-150 opt]# ls -l
total 0
drwxr-xr-x 4 root root      33 Oct 14 20:16 aws
drwxr-xr-x 2 root root       6 Aug 16 2018 rh
drwxr-xr-x 11 root root    141 Jul 27 2021 sonarqube-8.9.2.46101
[root@ip-172-31-85-150 opt]#
```

### Creating user for sonarqube

Creating user sonaradmin / Password Sonar@2024qube

```
[root@ip-172-31-85-150 opt]# useradd sonaradmin
[root@ip-172-31-85-150 opt]# passwd sonaradmin
Changing password for user sonaradmin.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@ip-172-31-85-150 opt]# _
```

Adding user to wheelgroup for permissions

```
[root@ip-172-31-85-150 opt]# visudo
[root@ip-172-31-85-150 opt]#
```

```
## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCESSES, LOCATE, DRIVERS

## Allows people in group wheel to run all commands
%wheel ALL=(ALL) ALL
sonaradmin ALL=(ALL) ALL

## Same thing without a password
# %wheel ALL=(ALL) NOPASSWD: ALL
sonaradmin ALL=(ALL) NOPASSWD: ALL

## Allows members of the users group to mount and unmount the
## cdrom as root
# %users ALL=/sbin/mount /mnt/cdrom, /sbin/umount /mnt/cdrom
```

Now start the sonarqube

So to start the sonarqube application we need to switch to sonaradmin user and for sonaradmin to get access of sonarqube repository we need to provide access to sonaradmin user as well as group through the root user.

user:groupname

chown -R sonaradmin:sonaradmin sonarqube-8.9.2

```
[root@ip-172-31-85-150 linux-x86-64]# cd /opt
[root@ip-172-31-85-150 opt]# chown -R sonaradmin:sonaradmin sonarqube-8.9.2.46101
[root@ip-172-31-85-150 opt]#
```

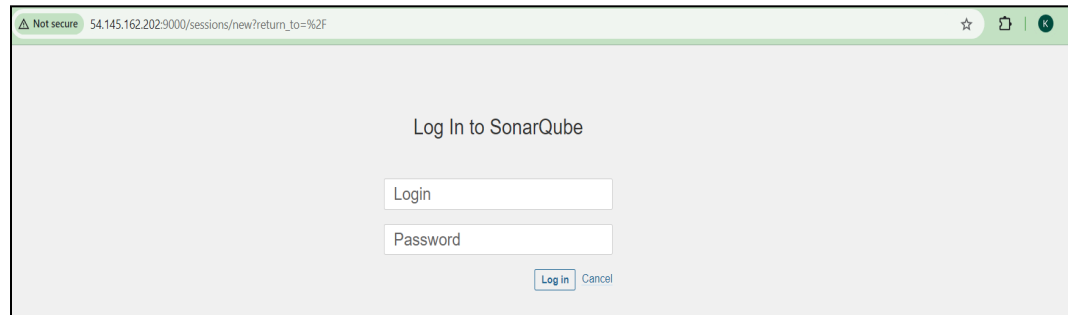
Now switch to sonaradmin user

```
[root@ip-172-31-85-150 opt]# sudo su - sonaradmin
Last login: Wed Oct 23 14:20:39 UTC 2024 on pts/0
[sonaradmin@ip-172-31-85-150 ~]$
```

Start the sonarqube

```
[sonaradmin@ip-172-31-85-150 ~]$ cd /opt
[sonaradmin@ip-172-31-85-150 opt]$ ls -l
total 0
drwxr-xr-x 4 root root 33 Oct 14 20:16 aws
drwxr-xr-x 2 root root 6 Aug 16 2018 rh
drwxr-xr-x 11 sonaradmin sonaradmin 141 Jul 27 2021 sonarqube-8.9.2.46101
[sonaradmin@ip-172-31-85-150 opt]$ cd sonarqube-8.9.2.46101/
[sonaradmin@ip-172-31-85-150 sonarqube-8.9.2.46101]$ ls -l
total 12
drwxr-xr-x 6 sonaradmin sonaradmin 94 Jul 27 2021 bin
drwxr-xr-x 2 sonaradmin sonaradmin 50 Jul 27 2021 conf
-rw-r--r-- 1 sonaradmin sonaradmin 7651 Jul 27 2021 COPYING
drwxr-xr-x 2 sonaradmin sonaradmin 24 Jul 27 2021 data
drwxr-xr-x 7 sonaradmin sonaradmin 132 Jul 27 2021 elasticsearch
drwxr-xr-x 4 sonaradmin sonaradmin 40 Jul 27 2021 extensions
drwxr-xr-x 6 sonaradmin sonaradmin 143 Jul 27 2021 lib
drwxr-xr-x 2 sonaradmin sonaradmin 64 Oct 23 14:15 logs
drwxr-xr-x 3 sonaradmin sonaradmin 38 Oct 23 14:15 temp
drwxr-xr-x 6 sonaradmin sonaradmin 4096 Jul 27 2021 web
[sonaradmin@ip-172-31-85-150 sonarqube-8.9.2.46101]$ cd bin/
[sonaradmin@ip-172-31-85-150 bin]$ ls -l
total 0
drwxr-xr-x 2 sonaradmin sonaradmin 25 Jul 27 2021 jsr-license
drwxr-xr-x 3 sonaradmin sonaradmin 48 Oct 23 14:15 linux-x86-64
drwxr-xr-x 3 sonaradmin sonaradmin 48 Jul 27 2021 macosx-universal-64
drwxr-xr-x 3 sonaradmin sonaradmin 109 Jul 27 2021 windows-x86-64
[sonaradmin@ip-172-31-85-150 bin]$ cd linux-x86-64/
[sonaradmin@ip-172-31-85-150 linux-x86-64]$ ls -l
total 132
drwxr-xr-x 2 sonaradmin sonaradmin 27 Jul 27 2021 lib
-rwxr-xr-x 1 sonaradmin sonaradmin 16393 Jul 27 2021 sonar.sh
-rwxr-xr-x 1 sonaradmin sonaradmin 111027 Jul 27 2021 wrapper
[sonaradmin@ip-172-31-85-150 linux-x86-64]$
```

```
[sonaradmin@ip-172-31-85-150 linux-x86-64]$ ls -l
total 132
drwxr-xr-x 2 sonaradmin sonaradmin 27 Jul 27 2021 lib
-rwxr-xr-x 1 sonaradmin sonaradmin 16393 Jul 27 2021 sonar.sh
-rwxr-xr-x 1 sonaradmin sonaradmin 111027 Jul 27 2021 wrapper
[sonaradmin@ip-172-31-85-150 linux-x86-64]$ ./sonar.sh start
Starting SonarQube...
Started SonarQube.
[sonaradmin@ip-172-31-85-150 linux-x86-64]$ ./sonar.sh status
SonarQube is running (18612).
[sonaradmin@ip-172-31-85-150 linux-x86-64]$
```



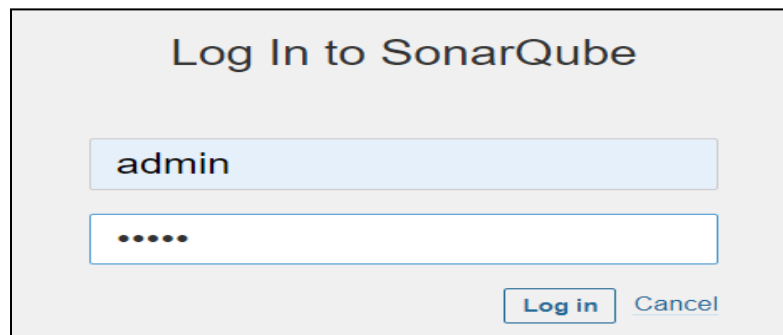
Log In to SonarQube

Login

Password

Log In Cancel

Login with username : admin password : admin



Log In to SonarQube

admin

.....

Log in Cancel

Update password

Update your password

This account should not use the default password.

Enter a new password

All fields marked with \* are required

Old Password \*

New Password \*

Confirm Password \*

Update

sonarqube

ProjectsIssuesRulesQuality ProfilesQuality GatesAdministration

Q Search for projects...

A

My FavoritesAll

Q Search by project name or key

Add Project

0 projects

Perspective: Overall StatusSort by: Name

Once you analyze some projects, they will show up here.

Here is how you can analyze new projects

Add a project

0 of 0 shown

Filters

Quality Gate

Passed0Failed0

Reliability ( Bugs )

A0B0C0D0E0

Security ( Vulnerabilities )

A0B0C0D0E0

Security Review ( Security Hotspots )

A ≥ 80%B 70% - 80%C 50% - 70%D 30% - 50%E < 30%

Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA

Community Edition - Version 8.9.2 (build 46101) - LGPL v3 - Community - Documentation - Plugins - Web API - About

Create authentication key for jenkins in sonarqube  
administration>security>users>tokens>add a token

sonarqube

ProjectsIssuesRulesQuality ProfilesQuality GatesAdministration

Q Search for projects...

A

Administration

ConfigurationSecurityProjectsSystemMarketplace

Users

Create and administer individual users

Create User

Q Search by login or name...

	SCM Accounts	Last connection	Groups	Tokens
<div>A Administrator admin</div>		< 1 hour ago	sonar-administrators sonar-users	1

1 of 1 shown

Tokens of Administrator

Generate Tokens

Enter Token NameGenerate

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

Copy81cb29eced4c23e5301c2e44597c82d9781101c5

Name	Last use	Created	
jenkins_token	Never	October 23, 2024	Revoke

81cb29eced4c23e5301c2e44597c82d9781101c5

## Jenkins installation on another ec2 instance

Connect the ec2 instance

```
C:\Users\Komal>ssh -i Downloads/jen-key.pem ec2-user@54.145.106.192
```

```
C:\Users\Komal>ssh -i Downloads/jen-key.pem ec2-user@54.145.106.192
The authenticity of host '54.145.106.192 (54.145.106.192)' can't be established.
ED25519 key fingerprint is SHA256:L1NyV0Smf8yyiI4QzR1HBLCTKdaQbkdIdeL2MN8dydU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.145.106.192' (ED25519) to the list of known hosts.

      _#_
     _###_
    _####_
   _#####_
  _#####_
 _#####_
#####_

Amazon Linux 2

AL2 End of Life is 2025-06-30.

A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-172-31-44-105 ~]$
```

```
[ec2-user@ip-172-31-44-105 ~]$ sudo su -
[root@ip-172-31-44-105 ~]#
```

## Java installation

Openjdk11 package available in amazon-linux-extras repository

```
[root@ip-172-31-44-105 ~]# amazon-linux-extras install java-openjdk11
Topic java-openjdk11 has end-of-support date of 2024-09-30
Installing java-11-openjdk
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-java-openjdk11 amzn2extra-kernel-5.10
17 metadata files removed
6 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
amzn2extra-docker
amzn2extra-java-openjdk11
amzn2extra-kernel-5.10
(1/9): amzn2-core/2/x86_64/group_gz
(2/9): amzn2-core/2/x86_64/updateinfo
(3/9): amzn2extra-java-openjdk11/2/x86_64/primary_db
(4/9): amzn2extra-kernel-5.10/2/x86_64/updateinfo
(5/9): amzn2extra-docker/2/x86_64/updateinfo
(6/9): amzn2extra-java-openjdk11/2/x86_64/updateinfo
(7/9): amzn2extra-docker/2/x86_64/primary_db
(8/9): amzn2extra-kernel-5.10/2/x86_64/primary_db
(9/9): amzn2-core/2/x86_64/primary_db
```

```
[root@ip-172-31-44-105 ~]# java --version
openjdk 11.0.23 2024-04-16 LTS
OpenJDK Runtime Environment (Red_Hat-11.0.23.0.9-2.amzn2.0.1) (build 11.0.23+9-LTS)
OpenJDK 64-Bit Server VM (Red_Hat-11.0.23.0.9-2.amzn2.0.1) (build 11.0.23+9-LTS, mixed mode, sharing)
[root@ip-172-31-44-105 ~]#
```

## Jenkins Installation

wget -O /etc/yum.repos.d/jenkins.repo <https://pkg.jenkins.io/redhat-stable/jenkins.repo>

Downloading jenkins repo

```
[root@ip-172-31-44-105 ~]# wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2024-10-23 15:12:55-- https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.38.133, 2a04:4e42:78::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.38.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

100%[=====]
2024-10-23 15:12:55 (9.41 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[root@ip-172-31-44-105 ~]#
```

Download key for authentication

rpm --import <https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key>

```
[root@ip-172-31-44-105 ~]# rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
```

## Now install jenkins

```
[root@ip-172-31-44-105 ~]# yum install jenkins -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                | 3.6 kB  00:00:00
jenkins                                   | 2.9 kB  00:00:00
jenkins/primary_db                        | 51 kB  00:00:00
Resolving Dependencies
--> Running transaction check
--> Package jenkins.noarch 0:2.462.3-1.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====================================================================================================================================
 Package               Arch              Version              Repository            Size
=====================================================================================================================================
Installing:
jenkins                noarch            2.462.3-1.1         jenkins               89 M
Transaction Summary
-----
Install 1 Package

Total download size: 89 M
Installed size: 89 M
Downloading packages:
jenkins-2.462.3-1.1.noarch.rpm           | 89 MB  00:00:06
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.462.3-1.1.noarch
    Verifying : jenkins-2.462.3-1.1.noarch
Installed:
jenkins.noarch 0:2.462.3-1.1
Complete!
[root@ip-172-31-44-105 ~]#
```

## Start jenkins

```
[root@ip-172-31-44-105 ~]# service jenkins start
Redirecting to /bin/systemctl start jenkins.service
[root@ip-172-31-44-105 ~]#
```

## Setup jenkins to automatically start at boot | Chkconfig jenkins on

```
[root@ip-172-31-44-105 ~]# chkconfig jenkins on
Note: Forwarding request to 'systemctl enable jenkins.service'.
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[root@ip-172-31-44-105 ~]#
```

```
ec2-user@ip-172-31-44-105 ~]$ service jenkins start
Redirecting to /bin/systemctl start jenkins.service
Failed to start jenkins.service: The name org.freedesktop.PolicyKit1 was not provided by any .service files
See system logs and 'systemctl status jenkins.service' for details.
ec2-user@ip-172-31-44-105 ~]$ sudo yum install polkit -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                | 3.6 kB  00:00:00
Resolving Dependencies
--> Running transaction check
--> Package polkit.x86_64 0:0.112-26.amzn2.1 will be installed
--> Processing Dependency: libmozjs-17.0.so(mozjs_17.0)(64bit) for package: polkit-0.112-26.amzn2.1.x86_64
--> Processing Dependency: polkit-pkla-compat for package: polkit-0.112-26.amzn2.1.x86_64
--> Processing Dependency: libmozjs-17.0.so()(64bit) for package: polkit-0.112-26.amzn2.1.x86_64
--> Running transaction check
--> Package mozjs17.x86_64 0:17.0.0-20.amzn2.0.1 will be installed
--> Package polkit-pkla-compat.x86_64 0:0.1-4.amzn2.0.2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====================================================================================================================================
 Package               Arch              Version              Repository            Size
=====================================================================================================================================
Installing:
polkit                 x86_64            0.112-26.amzn2.1     amzn2-core
Installing for dependencies:
mozjs17                x86_64            17.0.0-20.amzn2.0.1  amzn2-core
polkit-pkla-compat     x86_64            0.1-4.amzn2.0.2      amzn2-core
Transaction Summary
-----
Install 1 Package (+2 Dependent packages)

Total download size: 1.6 M
```

## Start the jenkins

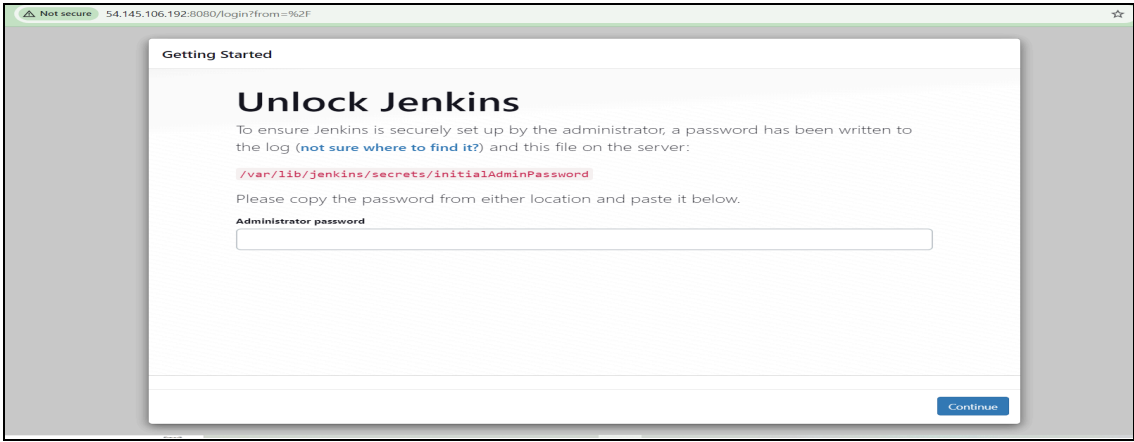
```
[root@ip-172-31-44-105 ~]# service jenkins start
Redirecting to /bin/systemctl start jenkins.service
[root@ip-172-31-44-105 ~]#
```

## Check status

```
[root@ip-172-31-44-105 ~]# service jenkins status
Redirecting to /bin/systemctl status jenkins.service
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2024-10-23 15:17:54 UTC; 20min ago
     Main PID: 690 (java)
    CGroup: /system.slice/jenkins.service
            └─690 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins

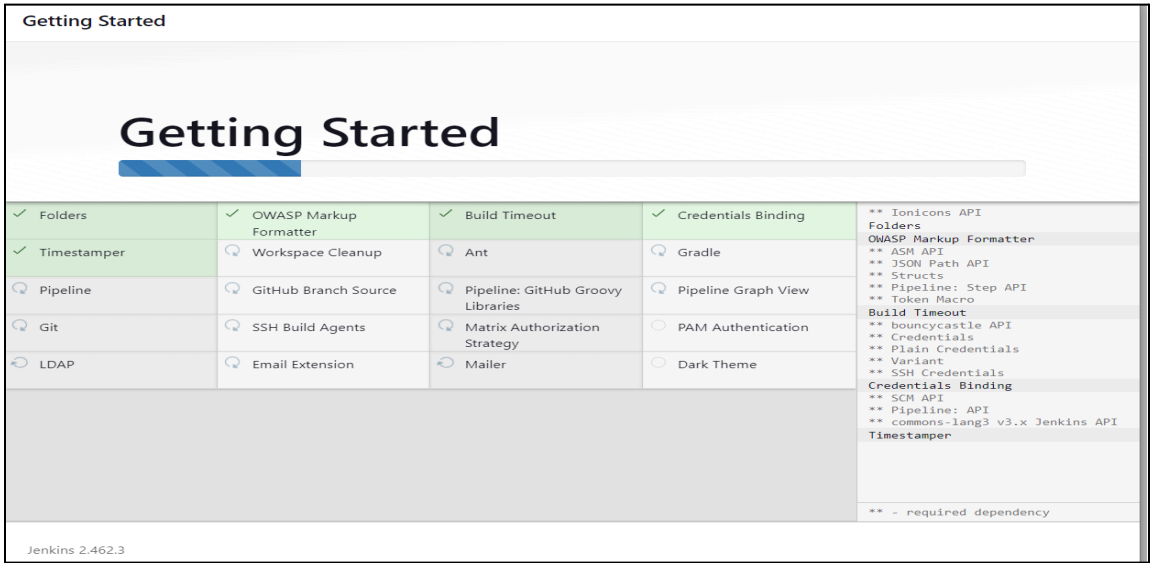
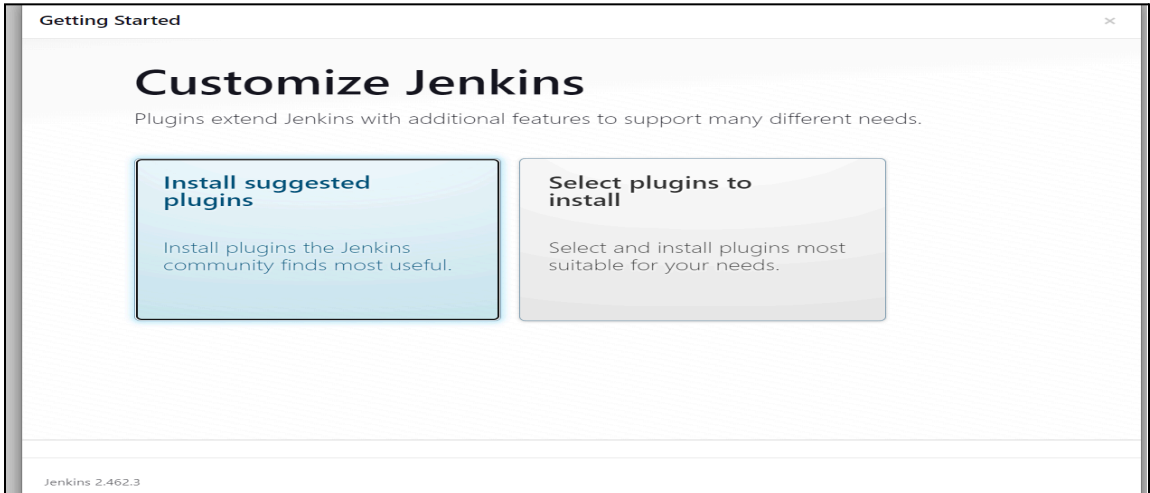
Oct 23 15:17:46 ip-172-31-44-105.ec2.internal jenkins[690]: 50afecb1fb894b3d874061a4edb633c7
Oct 23 15:17:46 ip-172-31-44-105.ec2.internal jenkins[690]: This may also be found at: /var/lib/jenkins/secre
Oct 23 15:17:46 ip-172-31-44-105.ec2.internal jenkins[690]: *****
Oct 23 15:17:46 ip-172-31-44-105.ec2.internal jenkins[690]: *****
Oct 23 15:17:46 ip-172-31-44-105.ec2.internal jenkins[690]: *****
Oct 23 15:17:54 ip-172-31-44-105.ec2.internal jenkins[690]: 2024-10-23 15:17:54.043+0000 [id=31] INFO
```





Get the password

```
[root@ip-172-31-44-105 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
50afecb1fb894b3d874061a4edb633c7
[root@ip-172-31-44-105 ~]#
```





## Update default profile

Getting Started

## Create First Admin User

Username  
komaljenkins

Password  
.....

Confirm password  
.....

Full name  
Komal Milind Deolekar

E-mail address  
d2022.komal.deolekar@ves.ac.in

Jenkins 2.462.3 Skip and continue as admin Save and Continue

Skip this because we are working on ec2 so ip address can change.

Getting Started

## Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.462.3 Not now Save and Finish

Getting Started

## Jenkins is ready!

You have skipped the configuration of the Jenkins URL.  
To configure the Jenkins URL, go to "Manage Jenkins" page.

Your Jenkins setup is complete.

Start using Jenkins

← → ↻ 🔒 Not secure 54.145.106.192:8080

**Jenkins** 🔍 Search (CTRL+K) 🔔 🔒 🔑 Komal Milind Deolekar 🚪 log out

Dashboard >

- + New Item
- 📄 Build History
- ⚙️ Manage Jenkins
- 📁 My Views

**Build Queue** ▾

No builds in the queue.

**Build Executor Status** ▾

- 1 Idle
- 2 Idle

### 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

Create a job +

Set up a distributed build

- Set up an agent 🖨
- Configure a cloud ☁
- Learn more about distributed builds ⓘ

REST API Jenkins 2.462.3



```
[ec2-user@ip-172-31-44-105 ~]$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 13226  100 13226    0     0    532k    0 --:--:-- --:--:-- --:--:--  538k
=> nvm is already installed in /home/ec2-user/.nvm, trying to update using git
=> => Compressing and cleaning up git repository

=> nvm source string already in /home/ec2-user/.bashrc
=> bash_completion source string already in /home/ec2-user/.bashrc
=> Close and reopen your terminal to start using nvm or run the following to use it now:

export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
[ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion" # This loads nvm bash_completion
[ec2-user@ip-172-31-44-105 ~]$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 13226  100 13226    0     0   601k    0 --:--:-- --:--:-- --:--:--  615k
=> nvm is already installed in /home/ec2-user/.nvm, trying to update using git
=> => Compressing and cleaning up git repository

=> nvm source string already in /home/ec2-user/.bashrc
=> bash_completion source string already in /home/ec2-user/.bashrc
=> Close and reopen your terminal to start using nvm or run the following to use it now:

export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
[ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion" # This loads nvm bash_completion
[ec2-user@ip-172-31-44-105 ~]$ . ~/.nvm/nvm.sh
[ec2-user@ip-172-31-44-105 ~]$ nvm install node
Downloading and installing node v23.0.0...
Downloading https://nodejs.org/dist/v23.0.0/node-v23.0.0-linux-x64.tar.xz...
#####
Computing checksum with sha256sum
Checksums matched!
node: /lib64/libm.so.6: version `GLIBC_2.27' not found (required by node)
node: /lib64/libc.so.6: version `GLIBC_2.27' not found (required by node)
node: /lib64/libc.so.6: version `GLIBC_2.28' not found (required by node)
nvm is not compatible with the npm config "prefix" option: currently set to ""
Run `nvm use --delete-prefix v23.0.0` to unset it.
[ec2-user@ip-172-31-44-105 ~]$
```

~/.nvm/nvm.sh

## Nvm install 16

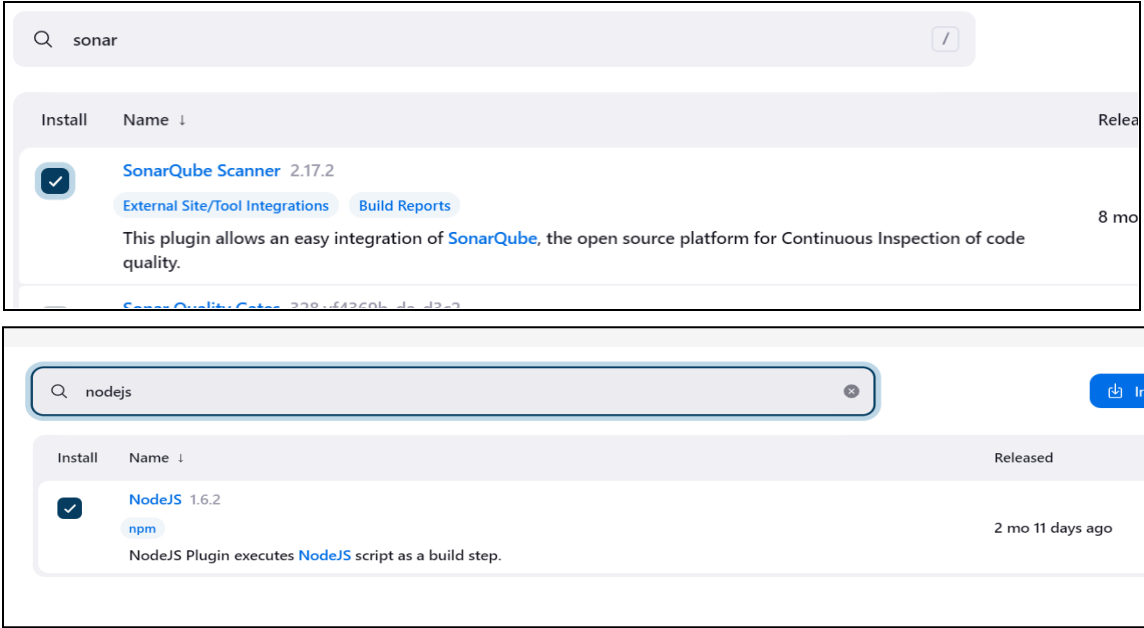
```
[ec2-user@ip-172-31-44-105 ~]$ nvm install 16
Downloading and installing node v16.20.2...
Downloading https://nodejs.org/dist/v16.20.2/node-v16.20.2-linux-x64.tar.xz...
#####
Computing checksum with sha256sum
Checksums matched!
Now using node v16.20.2 (npm v8.19.4)
Creating default alias: default -> 16 (-> v16.20.2)
[ec2-user@ip-172-31-44-105 ~]$
```

```
[ec2-user@ip-172-31-44-105 ~]$ nvm use 16
Now using node v16.20.2 (npm v8.19.4)
[ec2-user@ip-172-31-44-105 ~]$
```

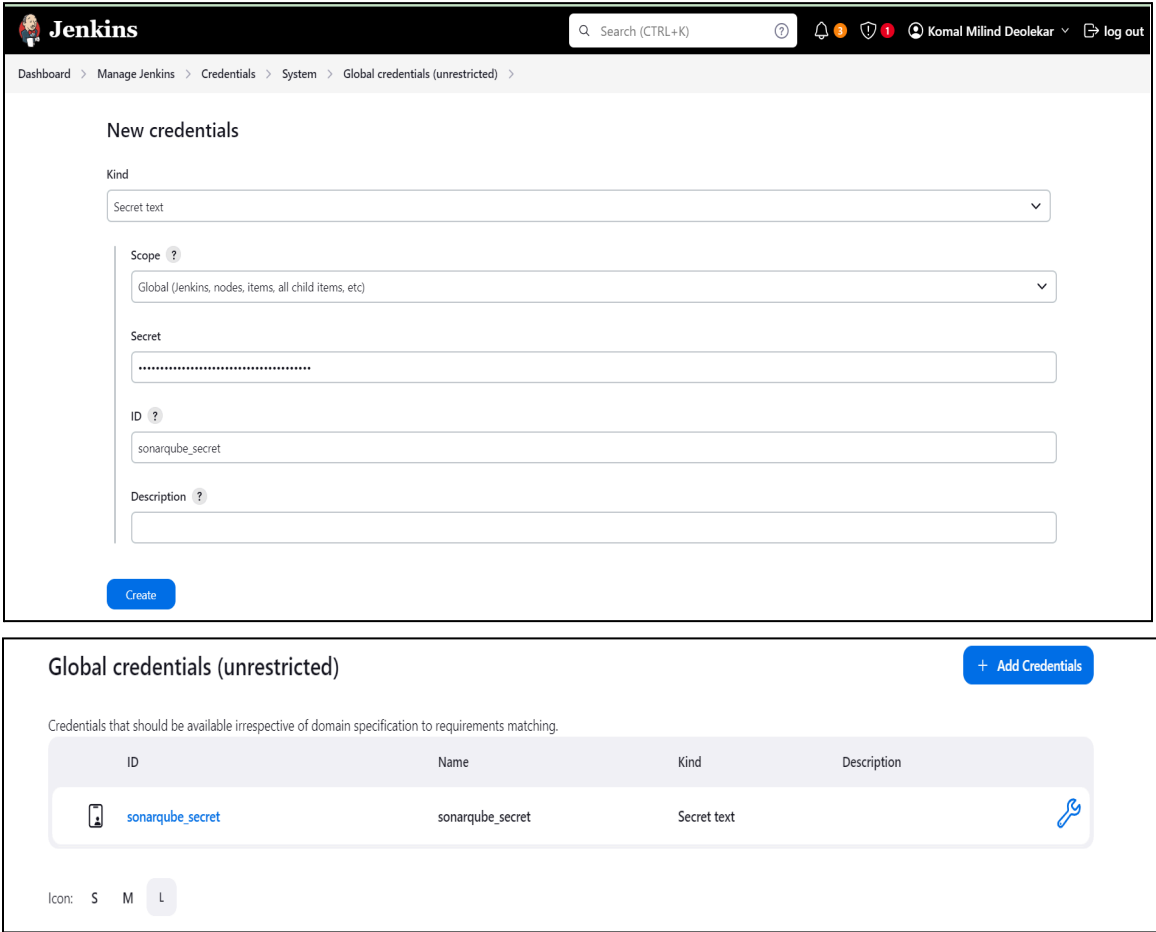
```
[ec2-user@ip-172-31-44-105 ~]$ node -v
v16.20.2
[ec2-user@ip-172-31-44-105 ~]$ npm -v
8.19.4
```

## Install plugins on jenkins

<input checked="" type="checkbox"/>	<b>GitHub Integration</b> 0.7.0 <a href="#">emailx</a> <a href="#">Build Triggers</a> GitHub Integration Plugin for Jenkins	8 mo 6 days
<input type="checkbox"/>	<b>GitHub Authentication</b> 597.ve0c3480fcb_d0 <a href="#">github</a> <a href="#">Security</a> <a href="#">Authentication and User Management</a> Authentication plugin using GitHub OAuth to provide authentication and authorization capabilities for GitHub and GitHub Enterprise.	11 mo ago
<input type="checkbox"/>	<b>GitHub Pull Request Builder</b> 1.42.2 <a href="#">github</a> <a href="#">Build Triggers</a> <div>             Warning: This plugin version may not be safe to use. Please review the following security notices:             <ul style="list-style-type: none"> <li>CSRF vulnerability and missing permission checks</li> <li>Missing permission check allows enumerating credentials IDs</li> </ul> </div>	3 yr 8 mo a
<input checked="" type="checkbox"/>	<b>Pipeline: GitHub</b> 2.8-159.09e4403bc62f <a href="#">pipeline</a> <a href="#">github</a> Allows programmatic access to GitHub via new global variables in pipeline builds.	9 mo 24 da
	<b>GitHub Organization Folder</b> 1.6	



Adding sonarqube token in jenkins



## Setting sonarqube in jenkins ( manage jenkins &gt; 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**

**Server URL**  
Default is http://localhost:9000

**Server authentication token**  
SonarQube authentication token. Mandatory when anonymous access is disabled.

[+ Add](#)

[Save](#) [Apply](#)

## Setting up nodejs in jenkins

**NodeJS installations**

[Add NodeJS](#)

**NodeJS**

**Name**

**Required**

☒ Install automatically ?

**Install from nodejs.org**

**Version**

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

## Setting sonarqube scanner (manage jenkins &gt; tools)

Dashboard > Manage Jenkins > Tools

**SonarQube Scanner installations**

[Add SonarQube Scanner](#)

**SonarQube Scanner**

**Name**

☒ Install automatically ?

**Install from Maven Central**

**Version**

[Add Installer](#)

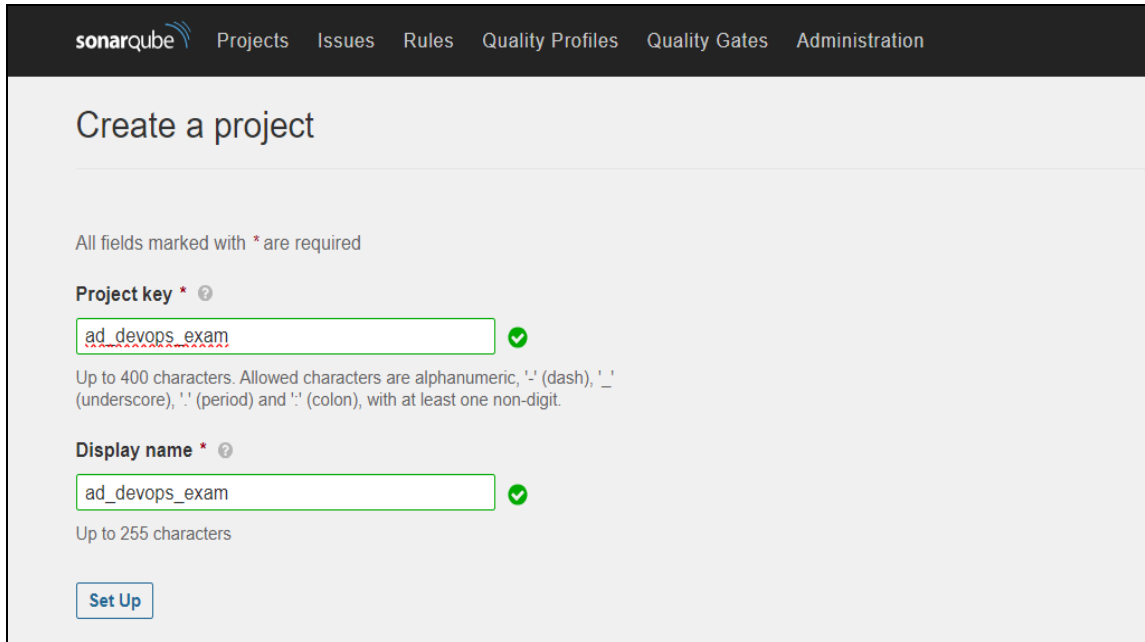
[Add SonarQube Scanner](#)

[Save](#) [Apply](#)

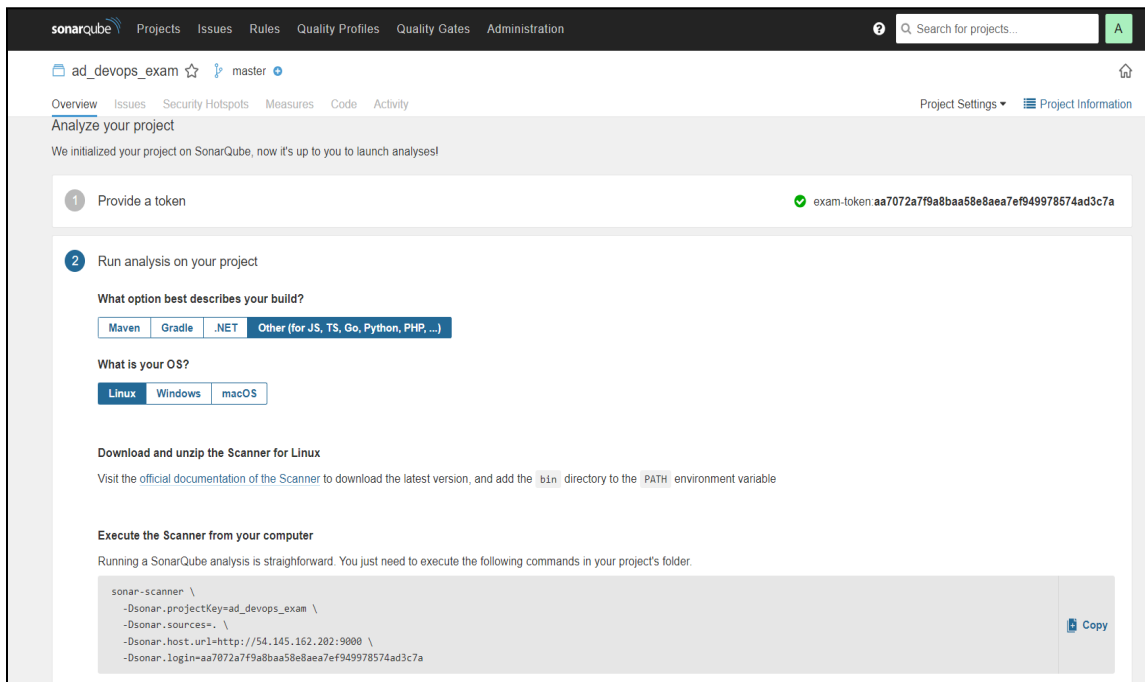
## Step 2: Create project in SonarQube

### 1. Create a New Project:

- Provide name.
- Generate token then continue and then select fields according to you project configuration it will provide you the script.



The screenshot shows the 'Create a project' page in SonarQube. The page has a dark header with the SonarQube logo and navigation links: Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. The main content area is titled 'Create a project'. Below the title, a note states 'All fields marked with \* are required'. There are two input fields: 'Project key \*' and 'Display name \*', both containing the text 'ad\_devops\_exam'. Each field has a green checkmark icon to its right. Below the 'Project key' field, a tooltip explains the format: 'Up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '\_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.' Below the 'Display name' field, a tooltip states 'Up to 255 characters'. At the bottom left of the form is a 'Set Up' button.



The screenshot shows the project configuration page for 'ad\_devops\_exam' in SonarQube. The page has a dark header with the SonarQube logo and navigation links. A search bar is visible on the right. The page title is 'ad\_devops\_exam' with a star icon and a 'master' branch indicator. Below the title are tabs for 'Overview', 'Issues', 'Security Hotspots', 'Measures', 'Code', and 'Activity'. The 'Overview' tab is selected. The main content area is titled 'Analyze your project' and contains a message: 'We initialized your project on SonarQube, now it's up to you to launch analyses!'. There are two main steps: 1. 'Provide a token' with a green checkmark and a token value 'exam-token: aa7072a7f9a8baa58e8aea7ef949978574ad3c7a'. 2. 'Run analysis on your project'. Under step 2, there are two sections: 'What option best describes your build?' with buttons for 'Maven', 'Gradle', '.NET', and 'Other (for JS, TS, Go, Python, PHP, ...)' (selected); and 'What is your OS?' with buttons for 'Linux', 'Windows', and 'macOS' (selected). Below these is a section 'Download and unzip the Scanner for Linux' with a link to the official documentation. At the bottom is a section 'Execute the Scanner from your computer' with a code block containing the following commands:

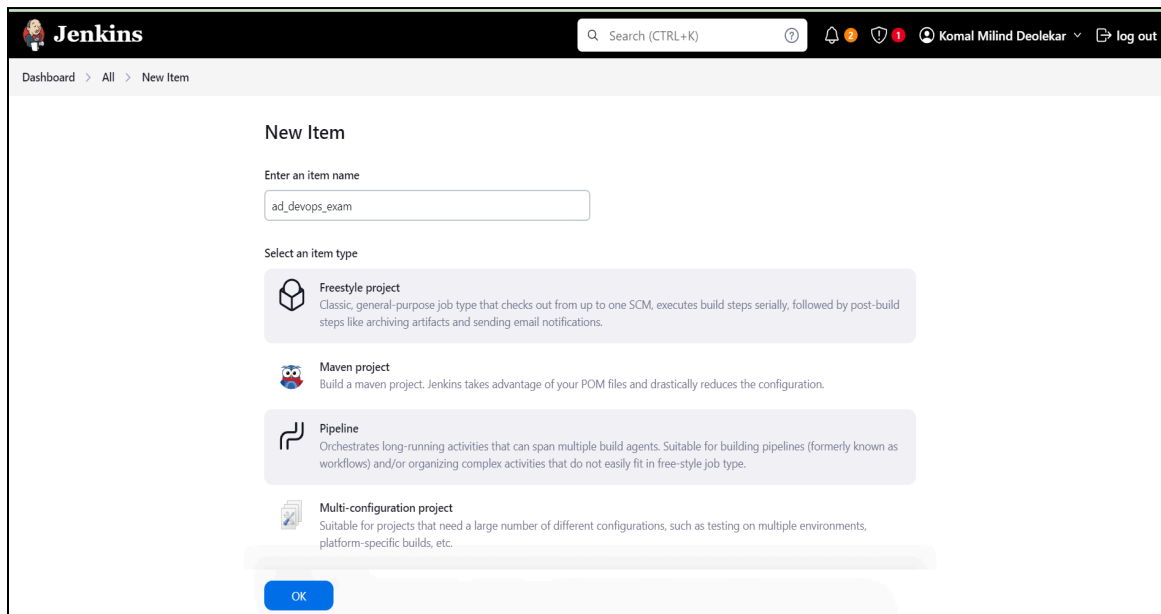
```
sonar-scanner \
-Dsonar.projectKey=ad_devops_exam \
-Dsonar.sources=. \
-Dsonar.host.url=http://54.145.162.202:9000 \
-Dsonar.login=aa7072a7f9a8baa58e8aea7ef949978574ad3c7a
```

A 'Copy' button is located to the right of the code block.

## Step 3: Configure Jenkins Job

1. **Create a New Jenkins Job:**
  - a. Click on "New Item" in Jenkins.
  - b. Choose "Pipeline" and name it (e.g., "JavaScript Code Analysis").
2. **Configure Source Code Management:**
  - a. In the job configuration, set up the source code management to pull from your Git repository.
3. **Add Build Steps:**
  - a. Add a build step to execute a shell command to run the SonarQube scanner. Make sure to configure the SonarQube parameters.
4. **Configure Post-Build Actions:**
  - a. Under post-build actions, add "SonarQube" and specify the project key.

Create a job and select pipeline and in pipeline script write the script



**Script :**

```
pipeline {
  agent any
  tools {
    nodejs 'node' // Ensure Node.js is installed in Jenkins
  }
  stages {
    stage('Checkout') {
      steps {
        cleanWs()
        git branch: 'main', url: 'https://github.com/KomalDeolekar0607/ad_dev_test.git'
      }
    }
  }
}
```

```

    }
  }
  stage('Install Dependencies') {
    steps {
      // Install nvm
      sh '''
      curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh | bash
      source ~/.nvm/nvm.sh
      nvm install 16 # Install the desired Node.js version
      nvm use 16
      npm install
      '''
    }
  }
  stage('SonarQube Analysis') {
    steps {
      // SonarQube Scanner command
      withSonarQubeEnv('sonarqube') {
        sh '''
        /opt/sonar-scanner/bin/sonar-scanner \
        -Dsonar.projectKey=new_test_project \
        -Dsonar.sources=. \
        -Dsonar.host.url=http://54.145.162.202:9000 \
        -Dsonar.login=0ba67fecb520ea316d4b1625ef11d71a04d198e4
        '''
      }
    }
  }
}
post {
  always {
    // Check SonarQube Quality Gate result
    script {
      def qg = waitForQualityGate()
      if (qg.status != 'OK') {
        error "Pipeline failed due to quality gate failure: ${qg.status}"
      }
    }
  }
}
}

```



```

1 pipeline {
2   agent any
3   tools {
4     nodejs 'node' // Ensure Node.js is installed in Jenkins
5   }
6   stages {
7     stage('Checkout') {
8       steps {
9         cleanWs()
10        git branch: 'main', url: 'https://github.com/KomalDeolekar0607/ad_dev_test.git'
11      }
12    }
13    stage('Install Dependencies') {
14      steps {
15        // Install nvm
16        sh '''
17          curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh | bash
18          source ~/.nvm/nvm.sh
19          nvm install 16 # Install the desired Node.js version
20          nvm use 16
21          npm install
22          ...
23        '''
24      }
25    }
26    stage('SonarQube Analysis') {
27      steps {
28        // SonarQube Scanner command
29        withSonarQubeEnv('sonarqube') {
30          sh '''
31            /opt/sonar-scanner/bin/sonar-scanner \
32              -Dsonar.projectKey=new_test_projcr \
33              -Dsonar.sources=. \
34              -Dsonar.host.url=http://54.145.162.202:9000 \
35              -Dsonar.login=0ba67fecb520ea316d4b1625ef11d71a04d198e4
36            ...
37          '''
38        }
39      }
40    }
41    post {
42      always {
43        script {
44          def qg = waitForQualityGate()
45          if (qg.status != 'OK') {
46            error "Pipeline failed due to quality gate failure: ${qg.status}"
47          }
48        }
49      }
50    }
51  }
52}

```

## Now build the job.

Jenkins

Search (CTRL+K)

?

🔔

🛡️

👤 Komal Milind Deolekar

🚪 log out

Dashboard

>

ad\_devops\_exam

>

Status

✖

ad\_devops\_exam

✎ Add description

</> Changes

> Build Now

⚙️ Configure

🗑️ Delete Pipeline

🔍 Full Stage View

📋 Stages

✎ Rename

📖 Pipeline Syntax

🌩️ Build History

trend

▼

🔍 Filter...

/

🕒 #10

Oct 23, 2024, 7:25 PM

🕒 #9

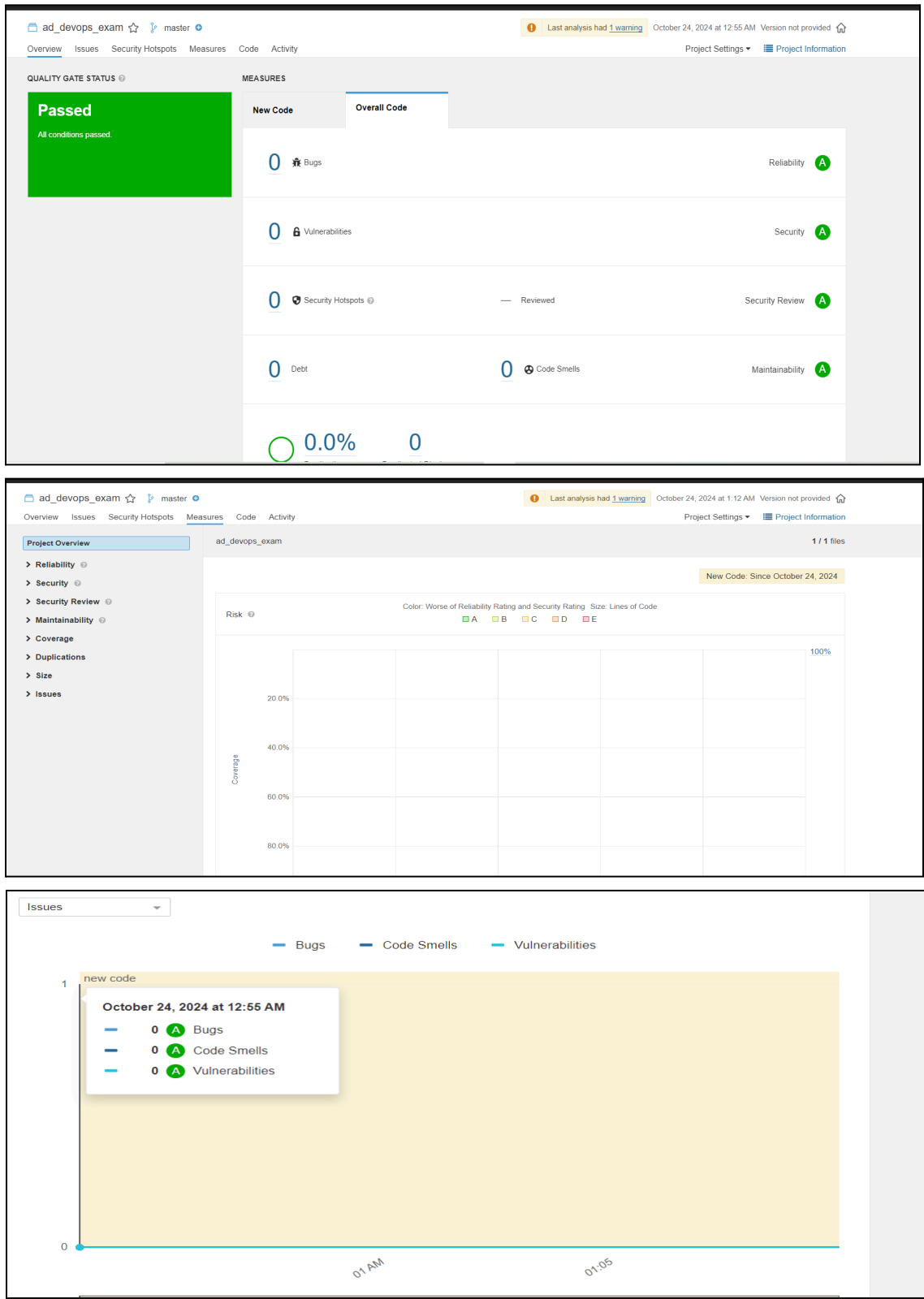
Oct 23, 2024, 7:20 PM

Stage View

	Declarative: Tool Install	Checkout	Install Dependencies	SonarQube Analysis	Declarative: Post Actions
Average stage times:	2s	1s	2s	2s	159ms
<div>#10</div> <div>Oct 24 00:55</div> <div>No Changes</div>	91ms	435ms	2s	18s	173ms <small>(paused for 9min 48s)</small>
<div>#9</div> <div>Oct 24 00:50</div> <div>No Changes</div>	97ms	520ms	3s	540ms <small>failed</small>	103ms
<div>#8</div> <div>Oct 24 00:37</div> <div>2 commits</div>	113ms	530ms	3s	858ms <small>failed</small>	112ms
<div>#7</div> <div>Oct 23 23:03</div> <div>No Changes</div>	133ms	586ms	6s <small>failed</small>	36ms <small>failed</small>	100ms

Sonarqube dashboard

Now you can analyze the project in sonarqube



## 5. Conclusion

The successful integration of Jenkins and SonarQube within an EC2 instance demonstrates the effectiveness of automating code quality checks through Continuous Integration. By following the outlined steps, I set up a Jenkins pipeline that automatically performed static code analysis on a JavaScript file, using SonarQube to evaluate the code's quality.

In the final analysis, the SonarQube report indicated 0 bugs, and all metrics—such as security vulnerabilities, code smells, and maintainability—were rated as optimal (green). This result validates the correctness of the code in the provided repository and highlights the importance of maintaining high-quality standards through continuous code inspection.

The green indicators signify that the project adheres to good coding practices, has minimal technical debt, and is free from critical issues, which is crucial for long-term project sustainability. This case study not only illustrates the practical use of CI/CD tools but also emphasizes the value of automated tools like SonarQube in ensuring the reliability and maintainability of software projects over time.

Going forward, this CI pipeline can be extended and adapted for larger projects, incorporating additional tests and checks to further enhance the development process.