

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

## 1. Introduction

### Case Study Overview:

This case study revolves around a DevOps-centric approach to software development and deployment. We will utilize Jenkins and SonarQube to implement a Continuous Integration (CI) pipeline with static code analysis. The focus is on setting up a Jenkins pipeline on AWS Cloud9 to automate the static code analysis of Python and Java applications via SonarQube. The objective is to ensure the reliability and maintainability of code through automatic checks for potential errors and security vulnerabilities during each integration.

### Key Feature and Application:

The core of this study lies in integrating Jenkins for CI and SonarQube for static code analysis. This integration automates the process of analyzing code quality, helping developers detect bugs and vulnerabilities early in the development process. With this automation, coding standards can be enforced, and potential risks like code smells or security weaknesses can be identified before deployment.

### Practical Application:

- **Jenkins:** Automates the CI pipeline, triggering builds and static code analysis whenever code changes occur.
- **SonarQube:** Performs static analysis to provide feedback on the maintainability, reliability, and security of the code.
- **AWS Cloud9:** Serves as the development environment, offering a cloud-based IDE that integrates smoothly with AWS services.

### Third-Year Project Integration (EventEase Project):

The EventEase project, which I developed during my third year, can benefit from the integration of this CI/CD pipeline approach to optimize the development process. EventEase is a comprehensive event management platform where users can organize events, manage attendees, and schedule activities seamlessly. The platform also provides features such as real-time notifications, ticketing, and user history tracking for past events.

By incorporating Jenkins and SonarQube into EventEase, the following improvements can be achieved:

- **Automated Testing:** Every new feature, bug fix, or update will trigger automated tests, ensuring that the platform remains stable and bug-free after each code commit.
- **Code Quality:** SonarQube will continuously monitor the quality of the codebase, making sure the project grows with clean, maintainable code, which in turn reduces technical debt.

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

- **Security Scanning:** SonarQube's security checks will help detect vulnerabilities early, ensuring that sensitive data like event details and user information is protected before the code is deployed to production.

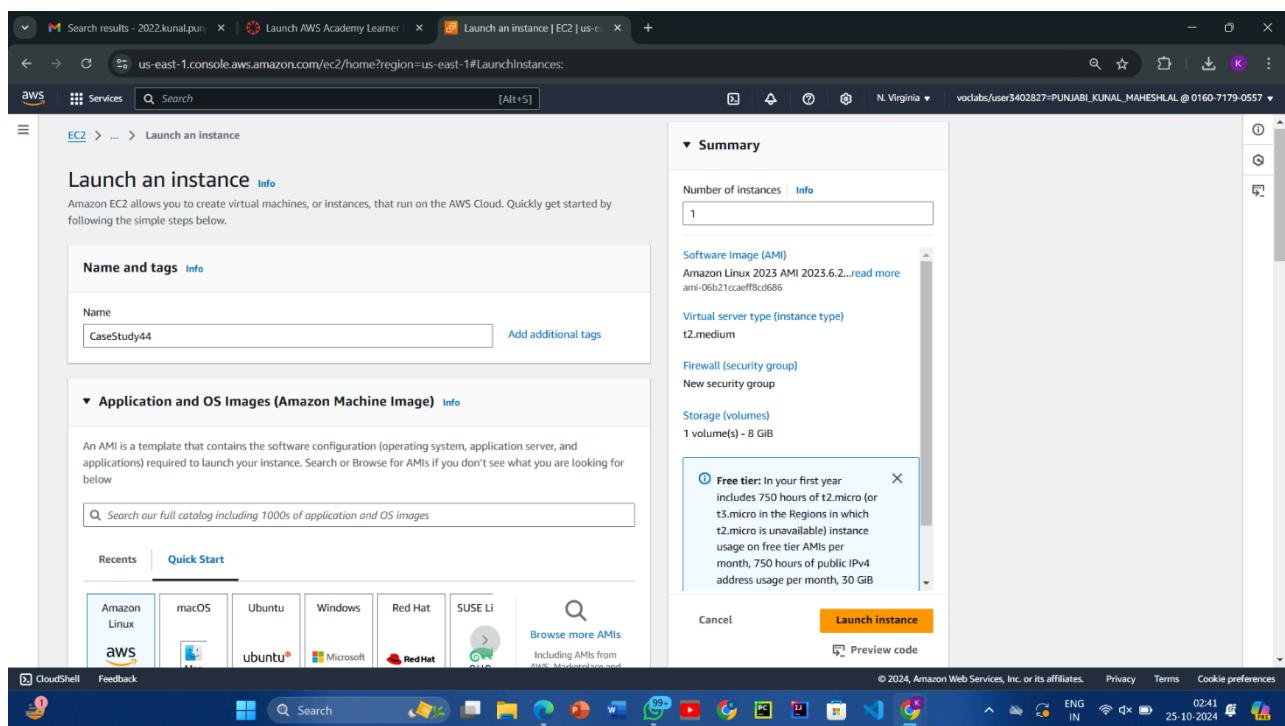
## 2. Demonstration

### Problem Statement

1] Jenkins on an EC2

instance:Resource:<https://www.jenkins.io/doc/tutorials/tutorial-for-installing-jenkins-on-AWS/>

1) Launch a AWS EC2 instance with a Linux OS .



2) instance type as t2.medium.

Name : Kunal Mahesh Punjabi

D15A

ROLL NO – 44

The screenshot shows the AWS EC2 Launch Instances wizard. The instance type is set to t2.medium. A tooltip for the Free tier is displayed, stating: "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 usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GB". The "Launch instance" button is highlighted.

### 3) Create a key pair for our instance

The screenshot shows the AWS EC2 Launch Instances wizard. A new key pair named "iamkunal44" is being created. The Network settings section shows the selection of a security group and network interface. A tooltip for the Free tier is displayed, stating: "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 usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GB". The "Launch instance" button is highlighted.

### 4) Allow the TCP, HTTP and HTTPS network access for all connections over the network.

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

The screenshot shows the 'Edit inbound rules' section of the AWS EC2 Security Groups interface. It lists five rules:

| Security group rule ID | Type       | Protocol | Port range | Source        | Description - optional |
|------------------------|------------|----------|------------|---------------|------------------------|
| sgr-0666a74e77e1ec5cB  | SSH        | TCP      | 22         | Anywhere-Int. | 0.0.0.0/0              |
| -                      | Custom TCP | TCP      | 0          | Anywhere-Int. | 0.0.0.0/0              |
| -                      | All TCP    | TCP      | 0 - 65535  | Anywhere-Int. | 0.0.0.0/0              |
| -                      | HTTP       | TCP      | 80         | Anywhere-Int. | 0.0.0.0/0              |
| -                      | HTTPS      | TCP      | 443        | Anywhere-Int. | 0.0.0.0/0              |

A warning message at the bottom states: "⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only."

## 5) SSH your connection

```
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\kunup>cd C:\Users\kunup\Downloads

C:\Users\kunup\Downloads>ssh -i "iamkunal44.pem" ec2-user@34.228.168.225
The authenticity of host '34.228.168.225 (34.228.168.225)' can't be established.
ED25519 key fingerprint is SHA256:/7NMwChv3jd0bxlisarVjE9Ybnrmq2NfSG7CxES7Jq4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '34.228.168.225' (ED25519) to the list of known hosts.

#_ _###_ Amazon Linux 2023
\_###_
\###| https://aws.amazon.com/linux/amazon-linux-2023
\#/ __>
\_\_/_ /_
\_\_/_ /_
\_m/_ /_
[ec2-user@ip-172-31-18-225 ~]$
```

## 6) Execute the following commands :-

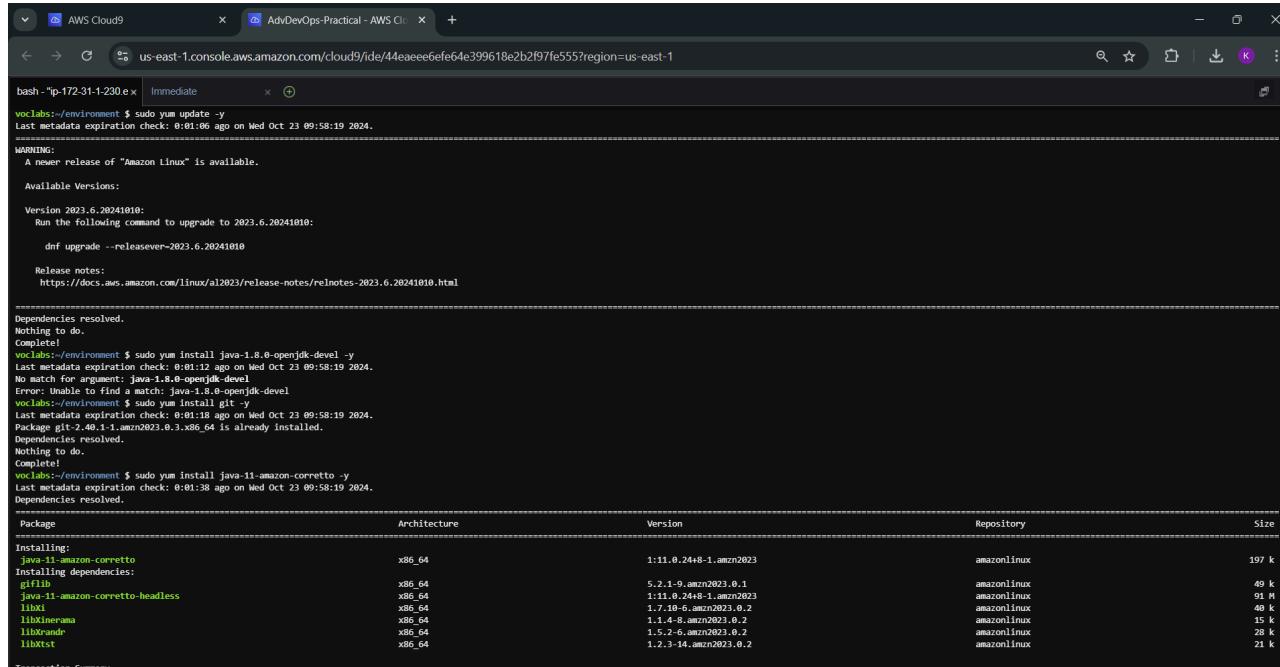
- **sudo yum update -y**: Updates all installed packages on the system to the latest available versions.
- **sudo wget -O /etc/yum.repos.d/jenkins.repo**  
<https://pkg.jenkins.io/redhat-stable/jenkins.repo>: Downloads the Jenkins repository file and saves it in the system's repository folder.
- **sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key**: Imports the Jenkins GPG key to authenticate the Jenkins packages.

Name : Kunal Mahesh Punjabi

D15A

ROLL NO – 44

- **sudo yum upgrade:** Upgrades the system packages to their latest versions based on available repositories.
- **sudo yum install jenkins -y:** Installs Jenkins from the configured repository.
- **sudo systemctl enable jenkins:** Enables Jenkins to automatically start at system boot.
- **sudo systemctl start jenkins:** Starts the Jenkins service immediately.
- **sudo systemctl status jenkins:** Displays the current status of the Jenkins service.



The screenshot shows two terminal sessions in AWS Cloud9. The top session is titled 'bash - "ip-172-31-1-230.e...' and the bottom session is titled 'Install 7 Packages'. Both sessions show the command 'sudo yum update' followed by a list of packages being updated. The package list includes Java, Git, and various Jenkins components like java-11-amazon-corretto-headless and libx11. The bottom session then shows the installation of Jenkins components, with progress bars indicating the download and installation of packages like libx11, libxtst, and java-11-amazon-corretto-headless.

```
bash - "ip-172-31-1-230.e x" Immediate + us-east-1.console.aws.amazon.com/cloud9/ide/44ae006e6e64e399618e2b2f97fe555?region=us-east-1
Last metadata expiration check: 0:01:06 ago on Wed Oct 23 09:58:19 2024.
...
WARNING: A newer release of "Amazon Linux" is available.
Available Versions:
Version 2023.6.20241010:
Run the following command to upgrade to 2023.6.20241010:
dnf upgrade --releasever=2023.6.20241010
Release notes:
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.6.20241010.html

Dependencies resolved.

Nothing to do.
Complete!
vclabs:~/environment $ sudo yum install java-1.8.0-openjdk-devel -y
Last metadata expiration check: 0:01:12 ago on Wed Oct 23 09:58:19 2024.
No Match for argument: java-1.8.0-openjdk-devel
Error: Unable to find a match: java-1.8.0-openjdk-devel
vclabs:~/environment $ sudo yum install git -y
Last metadata expiration check: 0:01:18 ago on Wed Oct 23 09:58:19 2024.
Package git-2.40.1-1.amzn2023.0.3.x86_64 is already installed.
Dependencies resolved.

Nothing to do.
Complete!
vclabs:~/environment $ sudo yum install java-11-amazon-corretto -y
Last metadata expiration check: 0:01:38 ago on Wed Oct 23 09:58:19 2024.
Dependencies resolved.

Transaction Summary
=====
Installing:
java-11-amazon-corretto           x86_64          1:11.0.24+8-1.amzn2023
Installing dependencies:
glibc                           x86_64          5.2.1-9.amzn2023.0.1
java-11-amazon-corretto-headless   x86_64          1:11.0.24+8-1.amzn2023
libX11                          x86_64          1.7.10-6.amzn2023.0.2
libXinerama                     x86_64          1.1.4-8.amzn2023.0.2
libXrandr                        x86_64          1.5.2-6.amzn2023.0.3
libXtst                          x86_64          1.2.3-14.amzn2023.0.2
Transaction Summary
=====
Total download size: 91 M
Installed size: 227 M
Downloading Packages:
(1/7): glibc-5.2.1-9.amzn2023.0.1.x86_64.rpm          932 kB/s | 49 kB  00:00
(2/7): java-11-amazon-corretto-11.0.24+8-1.amzn2023.x86_64.rpm 3.0 MB/s | 197 kB  00:00
(3/7): libX11-1.7.10-6.amzn2023.0.2.x86_64.rpm        1.2 MB/s | 48 kB  00:00
(4/7): libXinerama-1.1.4-8.amzn2023.0.2.x86_64.rpm    649 kB/s | 15 kB  00:00
(5/7): libXrandr-1.5.2-6.amzn2023.0.2.x86_64.rpm      746 kB/s | 1 kB  00:00
(6/7): libXtst-1.2.3-14.amzn2023.0.2.x86_64.rpm       596 kB/s | 21 kB  00:00
(7/7): java-11-amazon-corretto-headless-11.0.24+8-1.amzn2023.x86_64.rpm 12 MB/s | 88 MB  00:00 ETA
Broadcast message from root@ip-172-31-18-94.us-west-2.compute.internal (Wed 2024-10-23 10:00:01 UTC):
System shutdown has been cancelled
Total 17 MB/s | 91 MB  00:05
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction commit
Preparing :
 1/7: libX11-1.7.10-6.amzn2023.0.2.x86_64
 2/7: libXtst-1.2.3-14.amzn2023.0.2.x86_64
 3/7: libXrandr-1.5.2-6.amzn2023.0.2.x86_64
 4/7: libXinerama-1.1.4-8.amzn2023.0.2.x86_64
 5/7: libXrandr-1.5.2-6.amzn2023.0.2.x86_64
 6/7: libXtst-1.2.3-14.amzn2023.0.2.x86_64
 7/7: java-11-amazon-corretto-11.0.24+8-1.amzn2023.x86_64
Running scriptlet: java-11-amazon-corretto-11.0.24+8-1.amzn2023.x86_64
Verifying  : glibc-5.2.1-9.amzn2023.0.1.x86_64
Verifying  : java-11-amazon-corretto-11.0.24+8-1.amzn2023.x86_64
Verifying  : java-11-amazon-corretto-headless-11.0.24+8-1.amzn2023.x86_64
Verifying  : libX11-1.7.10-6.amzn2023.0.2.x86_64
Verifying  : libXinerama-1.1.4-8.amzn2023.0.2.x86_64
Verifying  : libXrandr-1.5.2-6.amzn2023.0.2.x86_64
Verifying  : libXtst-1.2.3-14.amzn2023.0.2.x86_64
WARNING: A newer release of "Amazon Linux" is available.
Available Versions:
Version 2023.6.20241010:
```

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

```
bash ->ip-172-31-1-230.ec2.us-east-1.compute.amazonaws.com [Immediate] x +
```

Complete!

```
vocabs:~/environment $ java -version
openjdk version "17.0.12" 2024-07-16 LTS
OpenJDK Runtime Environment Corretto-17.0.12-7.1 (build 17.0.12+7-LTS)
OpenJDK Java Virtual Machine Compat Mode (build 17.0.12+7-LTS, sharing)
vocabs:~/environment $ curl -s https://pkg.jenkins.io/redhat-stable/jenkins.repo
[vocabs:~/environment $ curl -s https://pkg.jenkins.io/redhat-stable/jenkins.repo
2024-10-23 10:00:52 - https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.38.133, 2a04:4e42:77::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'

/etc/yum.repos.d/jenkins.repo 100%[=====] 85 --.-KB/s in 0s

2024-10-23 10:00:52 (6.43 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

vocabs:~/environment $ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
vocabs:~/environment $ sudo yum install jenkins -y
Jenkins-stable
Dependencies resolved.
-----
```

| Package                       | Architecture | Version     | Repository |
|-------------------------------|--------------|-------------|------------|
| Installing:<br><b>jenkins</b> | noarch       | 2.462.3-1.1 | jenkins    |

Transaction Summary

```
Install 1 Package
```

Total download size: 89 M  
Installed size: 89 M  
Downloading Packages:  
jenkins-2.462.3-1.1.noarch.rpm

Total  
Running transaction check  
Transaction check succeeded.  
Running transaction test  
Transaction test succeeded.  
Running the actual transaction  
Preparing:  
Running scriptlet: jenkins-2.462.3-1.1.noarch  
Installing : jenkins-2.462.3-1.1.noarch  
Running scriptlet: jenkins-2.462.3-1.1.noarch

us-east-1.console.aws.amazon.com [Cloud9 IDE] 44eaeec6fe64e399618e2b2f97fe555?region=us-east-1

```
bash ->ip-172-31-1-230.ec2.us-east-1.compute.amazonaws.com [Immediate] x +
```

Installing : jenkins-2.462.3-1.1.noarch  
Running scriptlet: jenkins-2.462.3-1.1.noarch  
Verifying : jenkins-2.462.3-1.1.noarch

WARNING:  
A newer release of "Amazon Linux" is available.

Available Versions:

Version 2023.6.20241010:  
Run the following command to upgrade to 2023.6.20241010:  
dnf upgrade --releasever=2023.6.20241010

Release notes:  
<https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.6.20241010.html>

Installed:  
jenkins-2.462.3-1.1.noarch

Complete!

```
vocabs:~/environment $ sudo systemctl start jenkins
sudo systemctl enable jenkins
vocabs:~/environment $ sudo systemctl status jenkins
● Jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; disabled; preset: disabled)
   Active: active (running) since Wed 2024-10-23 10:01:48 UTC; 6s ago
     Main PID: 45 (java)
        Tasks: 45 (limit: 1112)
       Memory: 324.6M
          CPU: 15.0708s
         Group: 0
        CGroup: /system.slice/jenkins.service
           └─ 45 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Oct 23 10:01:41 ip-172-31-1-230.ec2.internal jenkins[3264]: This may also be found at: /var/lib/jenkins/secrets/initAdminPassword
Oct 23 10:01:41 ip-172-31-1-230.ec2.internal jenkins[3264]: ****
Oct 23 10:01:48 ip-172-31-1-230.ec2.internal jenkins[3264]: 2024-10-23 10:01:48.395+0000 [id:32] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
Oct 23 10:01:48 ip-172-31-1-230.ec2.internal jenkins[3264]: 2024-10-23 10:01:48.448+0000 [id:24] INFO hudson.lifecycle.Lifecycle$OnReady: Jenkins is fully up and running
Oct 23 10:01:48 ip-172-31-1-230.ec2.internal jenkins[3264]: 2024-10-23 10:01:48.450+0000 [id:24] INFO hudson.model.DownloadService$DownloadableJob: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Oct 23 10:01:48 ip-172-31-1-230.ec2.internal jenkins[3264]: 2024-10-23 10:01:48.452+0000 [id:47] INFO hudson.util.Retries$Start: Performed the action check update server successfully at the attempt #1
Oct 23 10:01:53 ip-172-31-1-230.ec2.internal jenkins[3264]: 2024-10-23 10:01:53.571+0000 [id:61] WARNING h.n.DiskSpaceMonitorDescriptor$MarkNodeOfflineOrOnline: Making Built-In Node offline temporarily due to the lack of disk space
vocabs:~/environment $ curl http://169.254.169.254/latest/meta-data/public_ip
```

| Inbound rules (3)   |      |                        |            |            |          |            |            |  |  |
|---|------|------------------------|------------|------------|----------|------------|------------|--|--|
| <input type="button" value="Create inbound rule"/> Manage tags Edit inbound rules |      |                        |            |            |          |            |            |  |  |
| <input type="text"/> Search   |      |                        |            |            |          |            |            |  |  |
|   | Name | Security group rule... | IP version | Type       | Protocol | Port range | Source     |  |  |
| <input type="checkbox"/>  | -    | sgr-0f575606d307f731d  | IPv4       | Custom TCP | TCP      | 8080       | 0.0.0.0/0  |  |  |
| <input type="checkbox"/>  | -    | sgr-0df8ee3261be4c812  | IPv4       | SSH        | TCP      | 22         | 35.172.155 |  |  |
| <input type="checkbox"/>  | -    | sgr-08139f8886610bb... | IPv4       | SSH        | TCP      | 22         | 35.172.155 |  |  |

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Getting Started

# Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

**Administrator password**

.....

**Continue**

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Getting Started

## Create First Admin User

Username

Password

Confirm password

Full name

E-mail address

Jenkins 2.462.3

Skip and continue as admin

Save and Continue

Name : Kunal Mahesh Punjabi

D15A

ROLL NO – 44

Getting Started

# Jenkins is ready!

Your Jenkins setup is complete.

[Start using Jenkins](#)

Jenkins 2.462.3

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

The screenshot shows a browser window for the Jenkins plugin manager at the URL `localhost:8082/pluginManager/`. The title bar says "Updates - Plugins [Jenkins]". The main content area is titled "Plugins" and contains a search bar with "Search plugin updates". On the left, there is a sidebar with links: "Updates" (selected), "Available plugins", "Installed plugins", "Advanced settings", and "Download progress". The main panel displays a message: "No updates available" with a small lock icon. At the bottom, a note states: "Disabled rows are already upgraded, awaiting restart. Shaded but selectable rows are in progress or failed." In the bottom right corner, there are links for "REST API" and "Jenkins 2.462.3".

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Thus, Jenkins is successfully configured on the EC2 Linux instance.

**Task:** SonarQube analysis of a Java/Python Project on Jenkins  
Pipeline:-

A] Sonarqube project:-

Python Project : <https://github.com/piomin/sample-java-sonar>

1) Create a sonarqube project named **casestudy\_24**.

localhost:9000/projects/create?mode=manual

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

1 of 2

Create a local project

Project display name \*

 ✓

Project key \*

 ✓

Main branch name \*

The name of your project's default branch [Learn More](#)

Cancel Next

⚠ 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](#) v10.7 (96327) ACTIVE [LGPL v3](#) [Community](#) [Documentation](#) [Plugins](#)

2) Sonarqube configurations in jenkins:-

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

The screenshot shows the Jenkins configuration interface for managing SonarQube servers. The top navigation bar includes links for Dashboard, Manage Jenkins, System, and SonarQube servers. The main content area is titled "SonarQube servers". A note states: "If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build." A checkbox labeled "Environment variables" is checked. Below this, there's a section for "SonarQube installations" with a link to "List of SonarQube installations". A new server configuration is being created, with fields for "Name" (set to "sonarqube") and "Server URL" (set to "http://localhost:9000"). Under "Server authentication token", it says "SonarQube authentication token. Mandatory when anonymous access is disabled." and shows a dropdown menu with "- none -" selected. There is also a "+ Add" button and an "Advanced" dropdown. At the bottom are "Save" and "Apply" buttons.

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Dashboard > Manage Jenkins > Tools

SonarQube Scanner installations

SonarQube Scanner installations

Add SonarQube Scanner

**SonarQube Scanner**

Name: sonarqube

Install automatically

**Install from Maven Central**

Version: SonarQube Scanner 6.2.0.4084

Add Installer

Add SonarQube Scanner



Dashboard > Manage Jenkins > Tools

Maven installations

Maven installations

Add Maven

**Maven**

Name: Maven

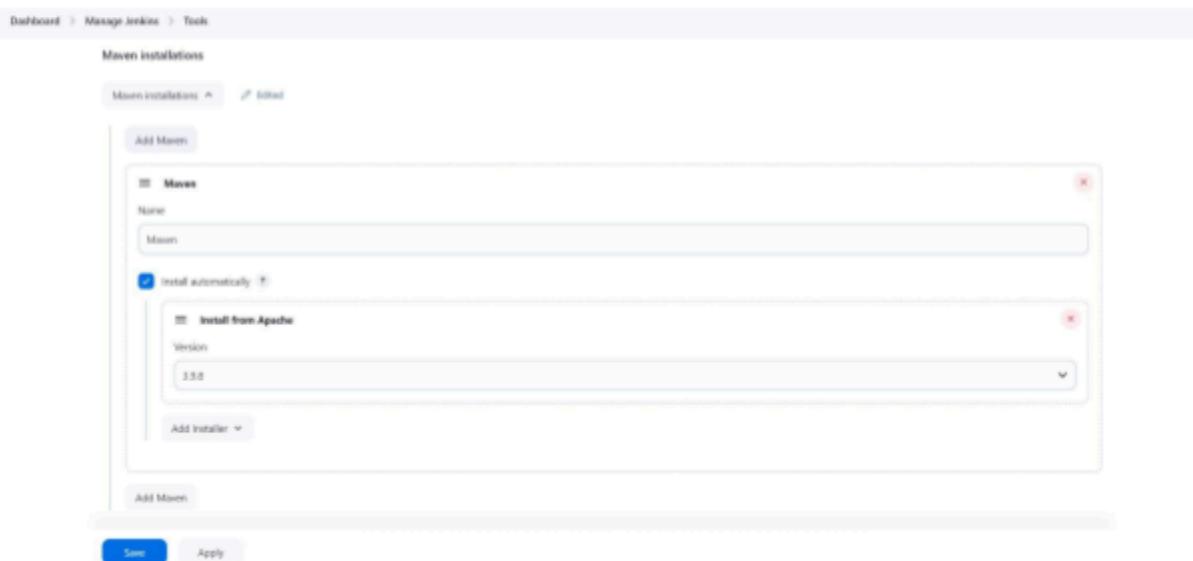
Install automatically

**Install from Apache**

Version: 3.3.0

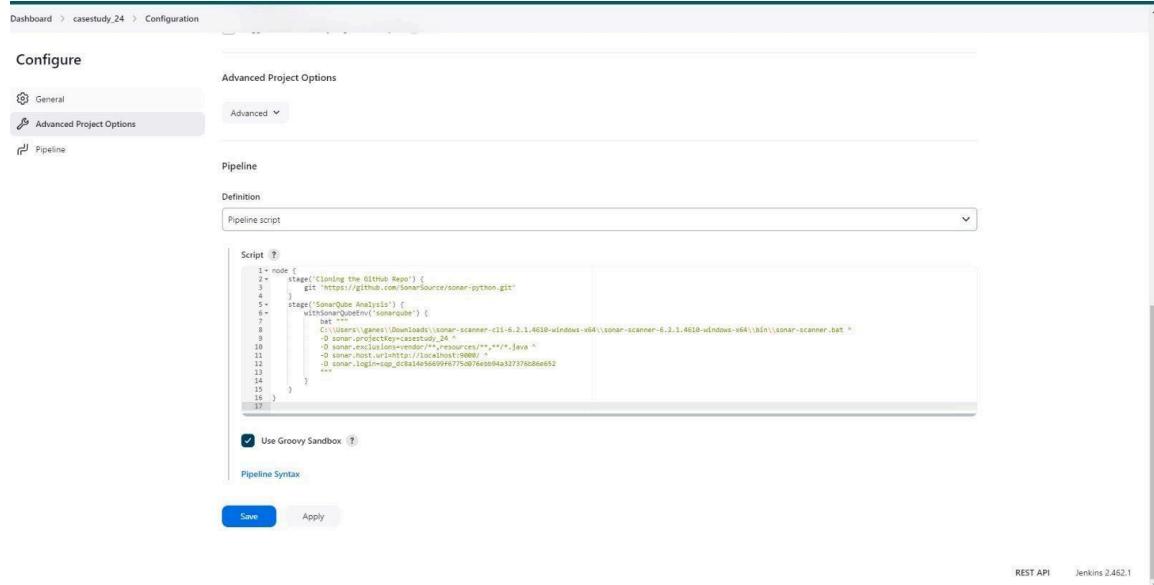
Add Installer

Add Maven



Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

### 3) Jenkins Pipeline:-



The screenshot shows the Jenkins Pipeline configuration page for a project named 'casestudy\_24'. The 'Pipeline' tab is selected under 'Definition'. The 'Script' field contains the following Groovy code:

```
1+ node {
2+     stage('Cloning the GitHub Repo') {
3+         git 'https://github.com/SonarSource/sonar-python.git'
4+     }
5+     stage('SonarQube Analysis') {
6+         withSonarQubeEnv('sonarqube') {
7+             bat """
8+                 C:\Users\nish\Downloads\sonar-scanner-cli-6.1.0.4477-windows-x64\sonar-scanner-6.1.0.4
9+                 477-windows-x64\bin\sonar-scanner.bat " +
10+                 -D sonar.login=admin " +
11+                 -D sonar.password=HareKrishna#108 " +
12+                 -D sonar.projectKey=casestudy_24 " +
13+                 -D sonar.exclusions=vendor/**,resources/**,*/*.java " +
14+                 -D sonar.host.url=http://localhost:9000/"
15+             }
16+         }
17+     }
18+ }
```

Below the script, there is a checkbox labeled 'Use Groovy Sandbox' which is checked. At the bottom of the page are 'Save' and 'Apply' buttons.

Pipeline  
Script:  
node {  
 stage('Cloning the GitHub Repo') { git  
 'https://github.com/SonarSource/sonar-python.git'  
 }  
 stage('SonarQube Analysis') {  
 withSonarQubeEnv('sonarqube') {  
 bat  
 "C:\\\\Users\\\\nish\\\\Downloads\\\\sonar-scanner-cli-6.1.0.4477-windows-x64\\\\sonar-scanner-6.1.0.4  
477-windows-x64\\\\bin\\\\sonar-scanner.bat " +  
 "-D sonar.login=admin " +  
 "-D sonar.password=HareKrishna#108 " +  
 "-D sonar.projectKey=casestudy\_24 " +  
 "-D sonar.exclusions=vendor/\*\*,resources/\*\*,\*/\*.java " +  
 "-D sonar.host.url=http://localhost:9000/"  
 }  
 }  
 }  
}

4) Jenkins Output :

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

 Console Output

 Download

 Copy

[View as plain text](#)

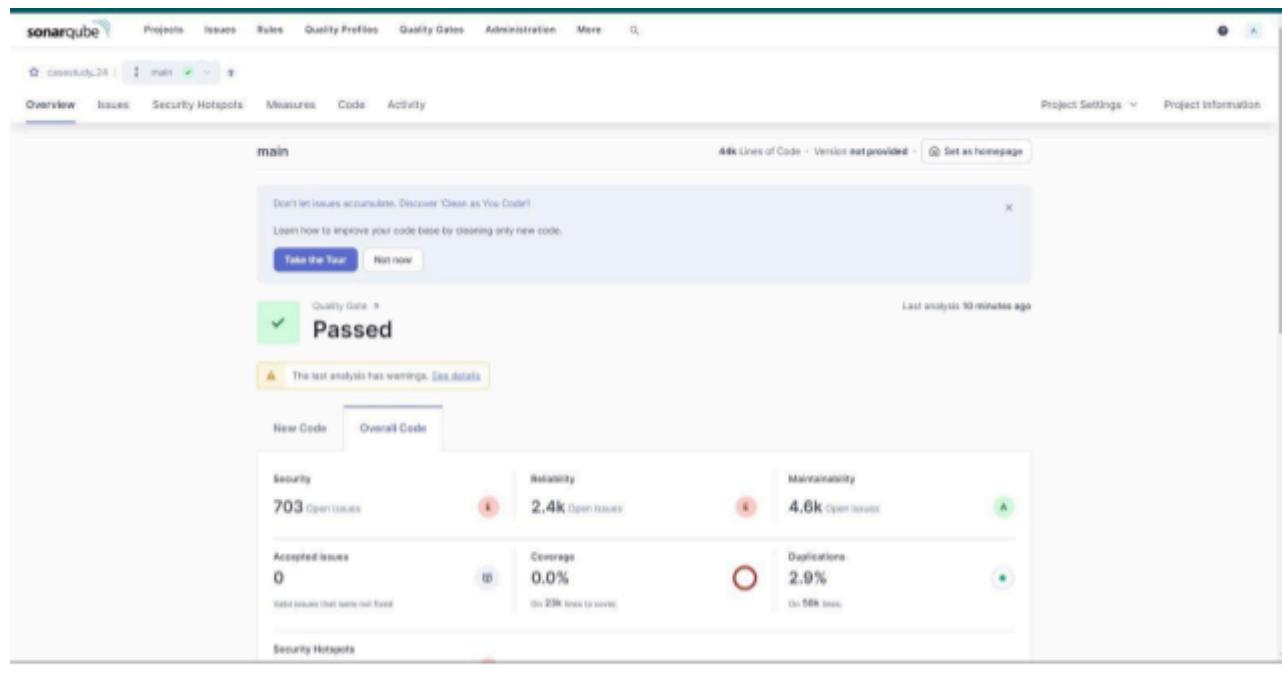
```
[Pipeline] start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\jenkins\workspace\casestudy_24
[Pipeline] {
[Pipeline] stage
[Pipeline] | {Cloning the GitHub Repo}
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\casestudy_24\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/SonarSource/sonar-python.git # timeout=10
Fetching upstream changes from https://github.com/SonarSource/sonar-python.git
> git.exe --version # timeout=10
> git --version # git version 2.43.0.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/SonarSource/sonar-python.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision 45fd33ef84936b07fb0f9790b4d35918ec4ef900 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 45fd33ef84936b07fb0f9790b4d35918ec4ef900 # timeout=10
> git.exe branch -a -v --no-abbrev # timeout=10
> git.exe branch -D master # timeout=10
> git.exe checkout -b master 45fd33ef84936b07fb0f9790b4d35918ec4ef900 # timeout=10
Commit message: "Update all non-major dependencies (#2007)"
> git.exe rev-list --no-walk 45fd33ef84936b07fb0f9790b4d35918ec4ef900 # timeout=10
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] | {SonarQube Analysis}
[Pipeline] withSonarQubeEnv
Injecting SonarQube environment variables using the configuration: sonarqube
[Pipeline] {
[Pipeline] bat

* The filename starts with "test"
* The filename contains "test," or "tests."
* Any directory in the file path is named: "doc", "docs", "test" or "tests"
* Any directory in the file path has a name ending in "test" or "tests"

11:17:38.212 2870 using git CLI to retrieve untracked files
11:17:38.274 2870 Analyzing language associated files and files included via "sonar.text.inclusions" that are tracked by git
11:17:37.643 2870 1732 source files to be analyzed
11:17:37.652 2870 1730/1732 files analyzed, current file: python-checks/src/main/resources/org/sonar/python/checks/hardcoded_credentials_call_check_meta.json
11:17:38.876 2870 1731/1732 source files have been analyzed
11:17:38.998 2870 Sonar TestAndSecretsSensor [test] (done) | time=1567ms
11:17:39.016 2870 ..... Run sensors on project
11:17:39.518 2870 Sonar Zero Coverage Sensor
11:17:39.544 2870 Sonar Zero Coverage Sensor (done) | time=323ms
11:17:39.728 2870 CPD Executor 129 files had no CPD blocks
11:17:39.728 2870 CPD Executor Calculating CPD for 723 files
11:17:39.943 2870 CPD Executor CPD calculation finished (done) | time=840ms
11:17:39.959 2870 SCM revision ID 'infidele-futurum@0f7798844233718ec4ef900'
11:18:00.836 2870 Analysis report generated in 2679ms, dir size=6.2 MB
11:18:00.777 2870 analysis report compressed in archive, zip size=3.8 MB
11:18:03.059 2870 Analysis report uploaded in 13ms
11:18:03.591 2870 ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=casestudy\_24
11:18:03.591 2870 Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
11:18:03.591 2870 More about the report processing at http://localhost:9000/api/ca/task?id=81276f441-cf89-4010-9eab-211074420965
11:18:04.843 2870 Analysis total time: 7:28.294 s
11:18:04.958 2870 SonarScanner Engine completed successfully
11:18:05.258 2870 EXECUTION SUCCESS
11:18:05.258 2870 Total time: 7:28.279s
[Pipeline]
[Pipeline] // withSonarQubeEnv
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] end of Pipeline
Finished: SUCCESS
```

Sonarqube Analysis:-

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44



### Build casestudy\_24



Thus, the Python project was successfully analyzed with SonarQube.

B ]For Maven Project:-

- 1) Make a Jenkins Pipeline.

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

New Item

Enter an item name

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.
-  **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 folders.
-  **Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.
-  **Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Java code to be analyzed :- <https://github.com/SonarSource/sonar-scanner-maven.git>

**2) Create a sonarqube project named casestudy24maven.**

1 of 2

Create a local project

Project display name \*

Project key \*

Main branch name \*

The name of your project's default branch [Learn More](#)

Pipeline  
Script:node {  
 stage('Cloning the GitHub Repo') {  
 git 'https://github.com/SonarSource/sonar-scanner-maven.git'  
 }  
}

Name : Kunal Mahesh Punjabi

D15A

ROLL NO – 44

```
stage('SonarQube Analysis') {
    withSonarQubeEnv('sonarqube') {
        bat
        "C:\\\\Users\\\\nish\\\\Downloads\\\\sonar-scanner-cli-6.1.0.4477-windows-x64\\\\sonar-scanner-6.1.0.4
477-windows-x64\\\\bin\\\\sonar-scanner.bat " +
        "-D sonar.login=admin " +
        "-D sonar.password=HareKrishna#108 " +
        "-D sonar.projectKey=casestudy24maven " +
        "-D sonar.exclusions=vendor/**,resources/**,**/*.java " +
        "-D sonar.host.url=http://localhost:9000/"
    }
}
```

The screenshot shows the Jenkins Pipeline configuration page for a job named 'casestudy.maven\_24'. The 'Pipeline' tab is selected, and the 'Pipeline script' section contains the following Groovy code:

```
node {
    stage('Cloning the GitHub Repo') {
        git 'https://github.com/SonarSource/sinor-scanner-maven.git'
    }
    stage('SonarQube Analysis') {
        withSonarQubeEnv('sonarqube') {
            C:\\\\Users\\\\nish\\\\Downloads\\\\sonar-scanner-cli-6.2.1.4018\\\\sonar-scanner-6.2.1.4018-windows-x64\\\\bin\\\\sonar-scanner.bat
            -D sonar.login=nish17aa592d5a6dcb45538031b15a9765c04
            -D sonar.password=HareKrishna#108
            -D sonar.projectKey=casestudy24maven
            -D sonar.exclusions=vendor/**,resources/**,**/*.java
            ...
        }
    }
}
```

Below the script, there is a checkbox labeled 'Use Groovy Sandbox' which is checked. At the bottom of the pipeline configuration are 'Save' and 'Apply' buttons.

- 3) Build and check its console output:-

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Console Output

Download Copy View as plain text

```
[Pipeline] start on pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\casestudy_24_java
[Pipeline]
[Pipeline] stage
[Pipeline] { ((Cloning the GitHub Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\casestudy_24_java\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/mohsiqba/sonar-maven.git # timeout=10
Fetching upstream changes from https://github.com/mohsiqba/sonar-maven.git
> git.exe -version # timeout=10
> git --version # git version 2.43.0.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/mohsiqba/sonar-maven.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master{commit}" # timeout=10
Checking out Revision 7e0b6392e64cdcf2498beed78c79309d000dacb5 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 7e0b6392e64cdcf2498beed78c79309d000dacb5 # timeout=10
> git.exe branch -a -v --no-abbrev # timeout=10
> git.exe branch -D master # timeout=10
> git.exe checkout -b master 7e0b6392e64cdcf2498beed78c79309d000dacb5 # timeout=10
Commit message: "sonarscanner with maven - maven multimodule project"
> git.exe rev-list --no-walk 7e0b6392e64cdcf2498beed78c79309d000dacb5 # timeout=10
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { ((SonarQube Analysis)
[Pipeline] withSonarQubeEnv
Injecting SonarQube environment variables using the configuration: sonarqube
[Pipeline] {
[Pipeline] bat
[Pipeline] bat

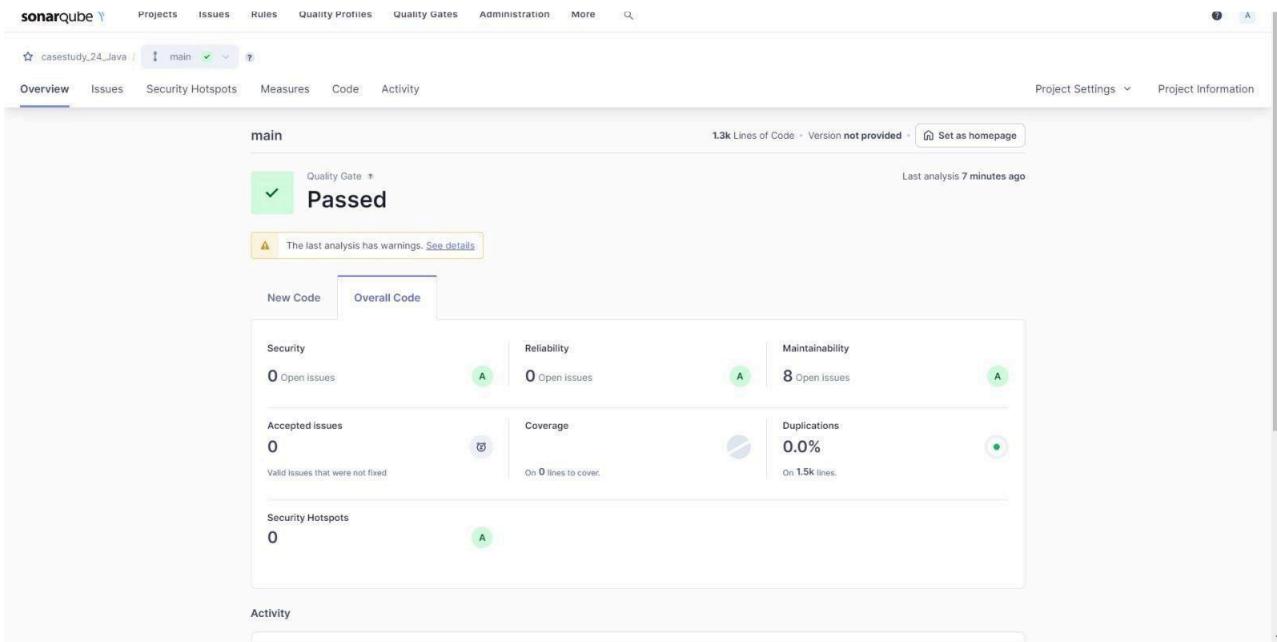
* The filename contains "test." or "tests."
* Any directory in the file path is named: "doc", "docs", "test" or "tests"
* Any directory in the file path has a name ending in "test" or "tests"

12:32:42.052 INFO Using git CLI to retrieve untracked files
12:32:42.100 INFO Analyzing language associated files and files included via "sonar.text.inclusions" that are tracked by git
12:32:42.159 INFO 23 source files to be analyzed
12:32:42.351 INFO 23/23 source files have been analyzed
12:32:42.354 INFO Sensor TextAndSecretsSensor [text] (done) | time=1272ms
12:32:42.369 INFO ----- Run sensors on project
12:32:42.452 INFO Sensor Zero Coverage Sensor
12:32:42.456 INFO Sensor Zero Coverage Sensor (done) | time=4ms
12:32:42.459 INFO SCM Publisher 18 source files to be analyzed
12:32:42.461 INFO SCM Publisher 18/18 source files have been analyzed (done) | time=549ms
12:32:42.471 INFO SCM Publisher 18/18 source files to be analyzed
12:32:43.211 INFO SCM Publisher 18/18 source files have been analyzed (done) | time=549ms
12:32:43.214 INFO CPD Executor Calculating CPD for 0 files
12:32:43.215 INFO CPD Executor CPD calculation finished (done) | time=0ms
12:32:43.234 INFO SCM revision ID: "7e0b6392e64cdcf2498beed78c79309d000dacb5"
12:32:43.583 INFO Analysis report generated in 153ms, dir size=330.3 kB
12:32:43.732 INFO Analysis report compressed in 131ms, zip size=98.9 kB
12:32:43.901 INFO Analysis report uploaded in 160ms
12:32:43.902 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=casestudy_24_java
12:32:43.903 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
12:32:43.904 INFO More about the report processing at: http://localhost:9000/api/ce/task?id=1637051a-00d5-4f73-ba51-8005c1fd436b
12:32:43.921 INFO Analysis total time: 15.476 s
12:32:43.922 INFO Sonarscanner Engine completed successfully
12:32:44.008 INFO EXECUTION SUCCESS
12:32:44.012 INFO Total time: 17.091s
[Pipeline]
[Pipeline] // withSonarQubeEnv
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] end of Pipeline
Finished: SUCCESS
```

REST API Jenkins 2.462.1

## SonarQube analysis

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44



#### Attributes:-

**Consistency:** Ensures the code follows uniform patterns and styles throughout, making it easier to read and understand.

**Intentionality:** Reflects that the code is written with clear purpose and reasoning, avoiding unnecessary complexity.

**Adaptability:** Indicates how flexible the code is to accommodate changes or new features without requiring major rewrites.

**Responsibility:** Reflects that the code is well-structured with clear ownership of functions or classes, adhering to the Single Responsibility Principle.

**Security:** Measures how well the code protects against vulnerabilities and malicious attacks. **Reliability:** Assesses the code's ability to function correctly under different conditions and its resistance to failures.

**Maintainability:** Evaluates how easily the code can be modified, debugged, or enhanced for long-term use.

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

The screenshot shows the SonarQube interface for the 'Issues' tab. On the left, there's a sidebar with filters for 'My Issues' and 'All'. Below that are sections for 'Clean Code Attribute' (Consistency, Intentionality, Adaptability, Responsibility) and 'Software Quality' (Security). The main panel lists two code files with issues:

- Person.js**: An 'Unexpected var, use let or const instead.' issue (Intentionality, es2015, bad-practice) with status 'Open' and 'Not assigned'.
- badfortune.py**: Two 'Remove the unused local variable' issues (Intentionality, unused) with status 'Open' and 'Not assigned'.

A note at the bottom states: "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."

### {} Code smells:-

A code smell is an indicator of potential design issues in code that, while not causing immediate errors, can lead to problems over time, such as maintainability and scalability issues. Common examples include large classes, long methods, and duplicated code, which increase complexity and technical debt. Identifying and addressing code smells through refactoring improves code quality and readability, ensuring long-term reliability and maintainability

### ii} Security Hotspots:-

The screenshot shows the SonarQube interface for the 'Issues' tab. On the left, there's a sidebar with filters for 'Severity' (High, Medium, Low), 'Type' (Bug, Vulnerability, Code Smell), and 'Scope'. The 'Code Smell' option is selected. The main panel lists two code files with issues:

- Person.js**: An 'Unexpected var, use let or const instead.' issue (Intentionality, es2015, bad-practice) with status 'Open' and 'Not assigned'.
- badfortune.py**: Two 'Remove the unused local variable' issues (Intentionality, unused) with status 'Open' and 'Not assigned'.

A note at the bottom states: "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."

At the bottom, it says: "SonarQube™ technology is powered by SonarSource SA".

A **security hotspot** is a piece of security-sensitive code that requires review to determine if it poses a threat. Unlike vulnerabilities, which need immediate fixes, hotspots must be assessed by the developer to decide if action is required. Fixing hotspots enhances an application's resilience against attacks. SonarQube assigns review priority based on standards like OWASP Top 10 and CWE Top 25, with hotspots categorized as high, medium, or low priority. Developers review the code, assess risks, and apply necessary fixes or mark it as safe. Addressing security hotspots strengthens code security while allowing for informed, context-based decisions.

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Successfully analyzed the Java Maven project using SonarQube.

### 3] Java Project:-

Project Link: <https://github.com/mohsiqba/sonar-maven.git>

#### 1)SonarQube Project:

1 of 2  
Create a local project  
Project display name \*  
casetudy4java  
Project key \*  
casetudy4java  
Main branch name \*  
main  
The name of your project's default branch [Learn More](#)  
Cancel Next

⚠️ 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.

#### 2)Jenkins Pipeline:

Dashboard > All > New Item

### New Item

Enter an item name  
casetudy4java

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

OK

#### Pipeline Script:-

```
node {  
    stage('Cloning the GitHub Repo') { git  
        'https://github.com/mohsiqba/sonar-maven.git'  
    }  
    stage('SonarQube Analysis') {  
        withSonarQubeEnv('sonarqube') {  
            bat
```

Name : Kunal Mahesh Punjabi

D15A

ROLL NO – 44

```
"C:\\\\Users\\\\nish\\\\Downloads\\\\sonar-scanner-cli-6.1.0.4477-windows-x64\\\\sonar-scanner-6.1.0.4  
477-windows-x64\\\\bin\\\\sonar-scanner.bat " +  
    "-D sonar.login=admin " +  
    "-D sonar.password=HareKrishna#108 " +  
    "-D sonar.projectKey=casestudy24java " +  
    "-D sonar.exclusions=vendor/**,resources/**,**/*.java " +  
    "-D sonar.host.url=http://localhost:9000/"  
}  
}  
}
```

Pipeline

Definition

Pipeline script

```
Script ?  
1 node {  
2     stage('Cloning the GitHub Repo') {  
3         // Clone the GitHub repository  
4         git 'https://github.com/mohsiqba/sonar-maven.git'  
5     }  
6     stage('SonarQube Analysis') {  
7         withSonarQubeEnv('sonarqube') {  
8             bat """  
9                 C:\\\\Users\\\\nisht\\\\Downloads\\\\sonar-scanner-cli-6.2.1.4610-windows-x64\\\\sonar-scanner-6.2.1.4610-windows-x64\\\\bin\\\\sonar-scanner.bat ^  
10                -D sonar.login=admin ^  
11                -D sonar.password=HareKrishna#108 ^  
12                -D sonar.projectKey=casestudy_24_Java ^  
13                -D sonar.exclusions=vendor/**,resources/**,**/*.java ^  
14                -D sonar.host.url=http://localhost:9000/  
15            """  
16        }  
17    }  
18 }
```

Use Groovy Sandbox ?

Pipeline Syntax

Save

Apply

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

Console Output

Download Copy View as plain text

```
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\jenkins\workspace\casestudy_24_java
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning the Github Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\casestudy_24_java\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/mohsiqba/sonar-maven.git # timeout=10
Fetching upstream changes from https://github.com/mohsiqba/sonar-maven.git
> git.exe --version # timeout=10
> git --version # "git version 2.43.0.windows.1"
> git.exe fetch --tags --force -progress -- https://github.com/mohsiqba/sonar-maven.git +refs/heads/*:refs/remotes/origin/*
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision 7e0b6392e64cdcf2498beed78c79309d000dacb5 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 7e0b6392e64cdcf2498beed78c79309d000dacb5 # timeout=10
> git.exe branch -a -v --no-abbrev # timeout=10
> git.exe branch -D master # timeout=10
> git.exe branch -b master # timeout=10
> git.exe checkout -b master 7e0b6392e64cdcf2498beed78c79309d000dacb5 # timeout=10
Commit message: "sonarscanner with maven - maven multimodule project"
> git.exe rev-list --no-walk 7e0b6392e64cdcf2498beed78c79309d000dacb5 # timeout=10
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (SonarQube Analysis)
[Pipeline] withSonarQubeEnv
Injecting SonarQube environment variables using the configuration: sonarqube
[Pipeline] {
[Pipeline] bat
```

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

```
Dashboard > cassandry_24.java > #2

    * The filename contains "test." or "tests."
    * Any directory in the file path is named: "doc", "docs", "test" or "tests"
    * Any directory in the file path has a name ending in "test" or "tests"

12:32:42.892 INFO Using git GLI to retrieve untracked files
12:32:42.338 INFO Analyzing language associated files and files included via "runner-test.includes" that are tracked by git
12:32:42.355 INFO 19 source files to be analyzed
12:32:42.355 INFO 18/19 source files have been analyzed
12:32:42.354 INFO Sensor TestableDeploymentSensor [test] (None) | timer=127ms
12:32:42.349 INFO ..... Run sensors on project
12:32:42.052 INFO Sensor Zero Coverage Sensor
12:32:42.050 INFO Sensor Zero Coverage Sensor (None) | timer=0ms
12:32:42.050 INFO ADN Publisher ADN provider for this project has: git
12:32:42.050 INFO ADN Publisher 18/18 source files to be analyzed
12:32:42.311 INFO ADN Publisher 18/18 source files have been analyzed (None) | timer=0ms
12:32:42.314 INFO CPU Counter Calculating CPU for 8 files
12:32:45.215 INFO CPU Counter CPU calculation finished (None) | timer=0ms
12:32:45.234 INFO ADN Analysis ID: 39eac2d4e4cfc24980ed9bc78389000000035
12:32:45.983 INFO Analysis report generated in 438ms, dir: c:\temp\8.1.48
12:32:45.732 INFO analysis report compressed in 11ms, zip size=68.9 KB
12:32:45.881 INFO Analysis report uploaded to S3
12:32:45.881 INFO ANALYSIS SUBMISSION, you can find the results at: http://localhost:8080/dashboard#id=cassandry_24.java
12:32:45.881 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
12:32:45.889 INFO More about the report processing at: http://localhost:8080/api/v2/cash?Id=1d71d091a-080b-4f7a-bab1-0000c3f40000
12:32:45.912 INFO Analysis total time: 10.478 s
12:32:45.912 INFO SmartScanner Engine completed successfully
12:32:46.010 INFO ANALYSIS SUCCESS
12:32:46.012 INFO Total time: 27.999s
[Pipeline]
[Pipeline] // withUser@datam6
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

The screenshot shows the SonarQube main dashboard for the 'main' branch of the 'comunity\_24.java' project. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a user icon. Below the navigation is a search bar with the query 'comunity\_24.java'. The main content area has tabs for Overview, Issues, Security Hotspots, Measures, Code, and Activity. The Overview tab is selected. The dashboard displays a green 'Passed' status with a checkmark icon. It shows 13k Lines of Code and a Version not provided. The last analysis was 7 minutes ago. A warning message indicates that the test analysis has warnings, with a 'See details' link. Below this, there are two tabs: 'New Code' (selected) and 'Overall Code'. The 'New Code' section contains metrics for Security (0 Open Issues), Reliability (0 Open Issues), Maintainability (B Open Issues), Accepted Issues (0), Coverage (0/0 lines to cover), Duplications (0.0%), and Security Hotspots (0). The 'Overall Code' section is currently empty. At the bottom, there's an 'Activity' section which is also empty.

Successfully analyzed the Java project using SonarQube. 4] Java Project:-  
Project Link: <https://github.com/nerdschoolbergen/code-smells.git>

## 1) SonarQube Project:

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More Q X

1 of 2

## Create a local project

Project display name \*

 ✓

Project key \*

 ✓

Main branch name \*

The name of your project's default branch [Learn More](#)

[Cancel](#) [Next](#)

**⚠️ 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.

## 2) Jenkins Pipeline:

Dashboard > All > New Item

## New Item

Enter an item name

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

**OK**

## Pipeline Script:-

```
node {
    stage('Cloning the GitHub Repo') { git
        'https://github.com/nerdschoolbergen/code-smells.git'
    }
    stage('SonarQube Analysis') {
        withSonarQubeEnv('sonarqube') {
            bat
                "C:\\\\Users\\\\nish\\\\Downloads\\\\sonar-scanner-cli-6.1.0.4477-windows-x64\\\\sonar-scanner-6.1.0.4477-windows-x64\\\\bin\\\\sonar-scanner.bat" +
                    "-D sonar.login=admin" + // Ensure there is a space before the next option
                    "-D sonar.password=HareKrishna#108" +
                    "-D sonar.host.url=https://sonarqube.nerdschoolbergen.com"
        }
    }
}
```

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44

```
-D sonar.projectKey=javacodesmell " +  
-D sonar.exclusions=vendor/**,resources/**,**/*.java " +  
-D sonar.host.url=http://localhost:9000/"  
}  
}  
}
```

The screenshot shows the Jenkins Pipeline configuration page. The pipeline script is defined as follows:

```
node {  
    stage('Cloning the GitHub Repo') {  
        git 'https://github.com/nerdschoolbergen/code-smells.git'  
    }  
    stage('SonarQube Analysis') {  
        withSonarQubeEnv('sonarque') {  
            bat C:\Users\ganesh\Downloads\sonar-scanner-cli-6.2.1.4610-windows-x64\sonar-scanner-6.2.1.4610-windows-x64\bin\sonar-scanner.bat ^  
                -D sonar.log.level=INFO ^  
                -D sonar.sources=src\main\java ^  
                -D sonar.projectKey=javacodesmell ^  
                -D sonar.exclusions=vendor/**,resources/**,**/*.java ^  
                -D sonar.host.url=http://localhost:9000/  
        }  
    }  
}
```

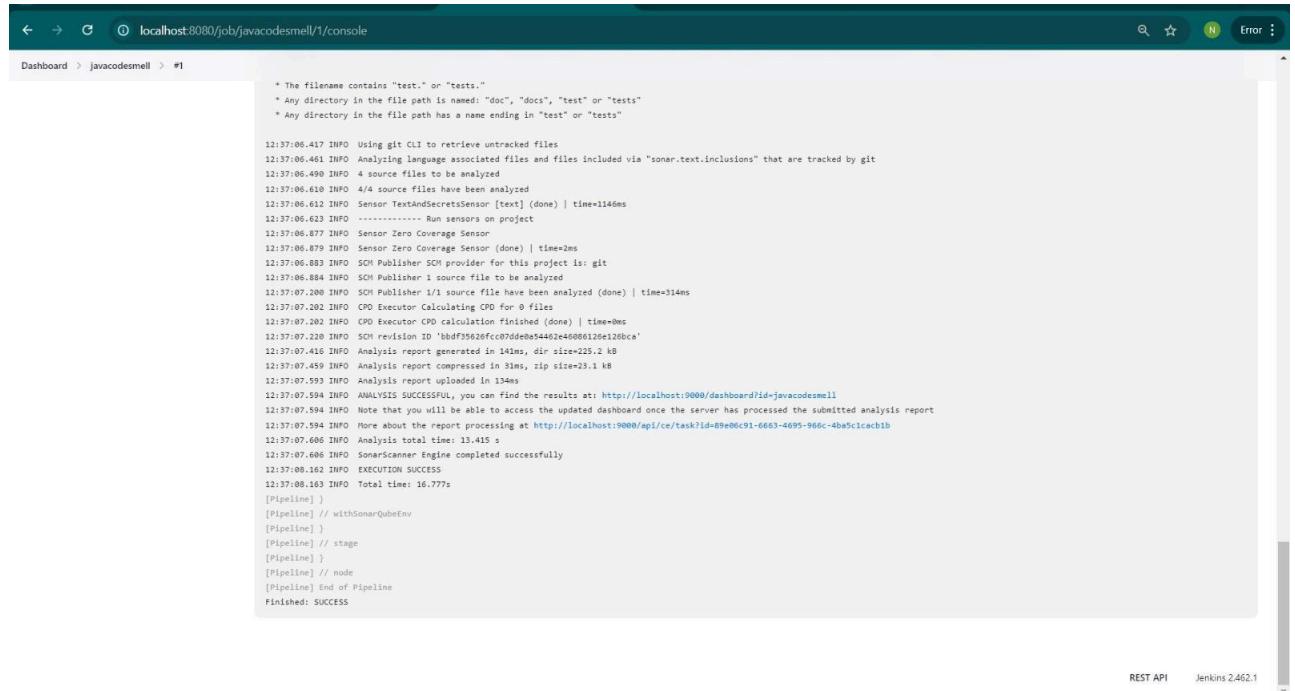
The 'Use Groovy Sandbox' checkbox is checked. The pipeline syntax is valid. At the bottom, there are 'Save' and 'Apply' buttons.

### 3) Console Output:

The screenshot shows the Jenkins console output for a pipeline run. The steps are:

- Started by user Nishant Sachin Khetel
- [Pipeline] Start of Pipeline (hide)
- [Pipeline] node
- Running on Jenkins in C:\ProgramData\Jenkins\jenkins\workspace\javacodesmell
- [Pipeline] {
- [Pipeline] stage
- [Pipeline] { (Cloning the GitHub Repo)
- [Pipeline] git
- The recommended git tool is: NONE
- No credentials specified
- Cloning the remote Git repository
- Cloning repository https://github.com/nerdschoolbergen/code-smells.git
- > git.exe init C:\ProgramData\Jenkins\jenkins\workspace\javacodesmell # timeout=10
- > git.exe fetch --tags --force --progress -- https://github.com/nerdschoolbergen/code-smells.git +refs/heads/\*:refs/remotes/origin/\* # timeout=10
- > git.exe config remote.origin.url https://github.com/nerdschoolbergen/code-smells.git # timeout=10
- > git.exe config --add remote.origin.fetch +refs/heads/\*:refs/remotes/origin/\* # timeout=10
- Avoid second fetch
- > git.exe rev-parse "refs/remotes/origin/master~(commit)" # timeout=10
- Checking out Revision bbdff35626fcc07dde0a54462e46086126e126bca (refs/remotes/origin/master)
- > git.exe config core.sparsecheckout # timeout=10
- > git.exe checkout -f bbdff35626fcc07dde0a54462e46086126e126bca # timeout=10
- > git.exe branch -a -v --no-abrev # timeout=10
- > git.exe checkout -b master bbdff35626fcc07dde0a54462e46086126e126bca # timeout=10
- Commit message: "remove whitespace"
- First time build. Skipping changelog.
- [Pipeline] }
- [Pipeline] // stage
- [Pipeline] { (SonarQube Analysis)
- [Pipeline] withSonarQubeEnv
- Injecting SonarQube environment variables using the configuration: sonarque

Name : Kunal Mahesh Punjabi  
D15A  
ROLL NO – 44



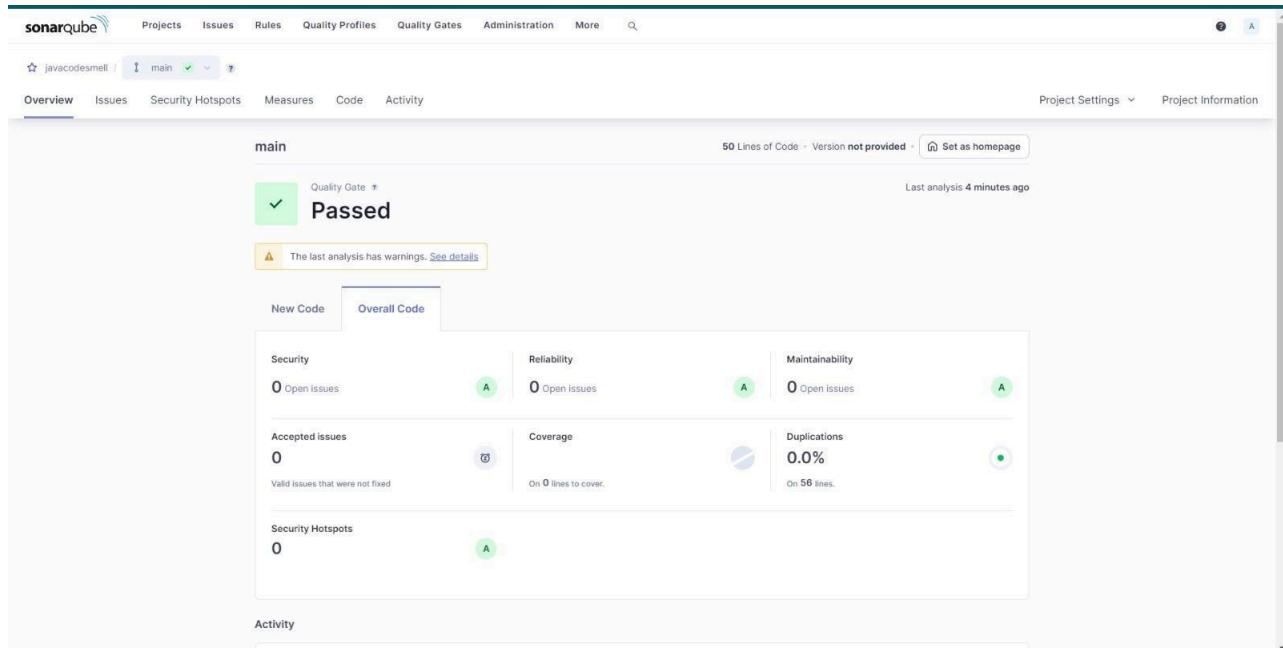
The screenshot shows the Jenkins job log for 'javacodesmell' with the build number '#1'. The log output is as follows:

```
* The filename contains "test." or "tests."
* Any directory in the file path is named: "doc", "docs", "test" or "tests"
* Any directory in the file path has a name ending in "test" or "tests"

12:37:06.417 INFO Using git CLI to retrieve untracked files
12:37:06.461 INFO Analyzing language associated files and files included via "sonar.text.inclusions" that are tracked by git
12:37:06.498 INFO 4 source files to be analyzed
12:37:06.610 INFO 4/4 source files have been analyzed
12:37:06.612 INFO Sensor TextAndSecretsSensor [text] (done) | time=1146ms
12:37:06.623 INFO ----- Run sensors on project
12:37:06.877 INFO Sensor Zero Coverage Sensor
12:37:06.879 INFO Sensor Zero Coverage Sensor (done) | time=2ms
12:37:06.883 INFO SCM Publisher SCM provider for this project is: git
12:37:06.884 INFO SCM Publisher 1 source file to be analyzed
12:37:07.208 INFO SCM Publisher 1/1 source file have been analyzed (done) | time=314ms
12:37:07.202 INFO CPD Executor Calculating CPD for 0 files
12:37:07.202 INFO CPD Executor CPD calculation finished (done) | time=0ms
12:37:07.220 INFO SCM revision ID 'bbdf35e26fccf07d0deba54462e4a08612e12b0ca'
12:37:07.416 INFO Analysis report generated in 140ms, dir size=25.2 kB
12:37:07.459 INFO Analysis report compressed in 31ms, zip size=23.1 kB
12:37:07.593 INFO Analysis report uploaded in 134ms
12:37:07.594 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=javacodesmell
12:37:07.594 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
12:37:07.594 INFO more about the report processing at http://localhost:9000/api/ce/task?id=89e06c91-6603-4695-986c-4ba5c1ac8b
12:37:07.600 INFO Analysis total time: 13.415 s
12:37:07.600 INFO SonarScanner Engine completed successfully
12:37:08.162 INFO EXECUTION SUCCESS
12:37:08.163 INFO Total time: 16.777s
[Pipeline]
[Pipeline] // withSonarQubeEnv
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

REST API Jenkins 2.46.1

## SonarQube Analysis:



The screenshot shows the SonarQube analysis results for the 'main' project. The dashboard includes the following key metrics:

- Quality Gate:** Passed (Green)
- Last analysis:** 4 minutes ago
- Code Metrics:** 50 Lines of Code - Version not provided
- New Code:** 0 Open issues
- Overall Code:** 0 Open issues
- Reliability:** 0 Open issues
- Maintainability:** 0 Open issues
- Coverage:** 0 lines to cover
- Duplications:** 0.0%
- Security Hotspots:** 0

Successfully analyzed the Java project using SonarQube.

## Conclusion :

Through this case study, Jenkins and SonarQube were successfully integrated to automate the continuous integration and static code analysis of Python and Java projects. The CI pipeline

Name : Kunal Mahesh Punjabi

D15A

ROLL NO – 44

helps maintain high code quality and ensures that code adheres to security standards before being pushed to production.