Advanced DevOps Lab
Experiment 8

Name: Yash Rahate
Class: D15B
Roll No.: 48

Aim:

Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

Theory:

What is a CI/CD Pipeline?
A Continuous Integration/Continuous Delivery (CI/CD) pipeline automates the processes of building, testing, and delivering software. It allows developers to integrate their code changes frequently and deliver new software versions efficiently. The pipeline includes various steps such as coding, building the application, running tests, and deploying the application to users.

What is Jenkins?
Jenkins is an open-source automation server widely used to facilitate CI/CD pipelines. It automates tasks needed to compile code, run tests, and deploy applications. Jenkins integrates with various tools, making it a popular choice for developers looking to streamline their software development processes.

What is SonarQube?
SonarQube is a tool that performs static analysis of code to assess its quality. It checks the source code for bugs, security vulnerabilities, and code smells (issues that may indicate deeper problems). By providing detailed reports, SonarQube helps developers understand the quality of their code and how to improve it.

Integration of Jenkins and SonarQube:
Integrating Jenkins with SonarQube allows the CI/CD pipeline to automatically analyze code quality during the build process. Whenever developers commit changes, Jenkins triggers a SonarQube scan to detect any issues early. This integration ensures that only high-quality code is deployed, reducing the risk of bugs and vulnerabilities.

Importance of Code Quality Analysis:

Using SonarQube in the CI/CD pipeline helps developers identify and fix issues before code is deployed. This proactive approach saves time and resources, improves application quality, and enhances security by addressing vulnerabilities early in development.

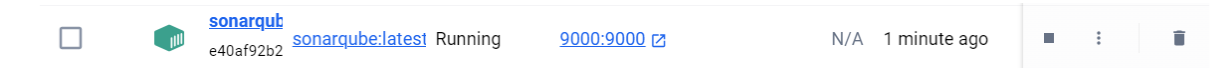
Benefits of SonarQube:

- **Sustainability**: SonarQube helps reduce complexity and vulnerabilities, extending the lifespan of applications.
- **Increased Productivity**: It streamlines development by minimizing the effort required for manual code reviews, lowering maintenance costs.
- **Error Detection**: SonarQube automatically alerts developers to errors, allowing them to fix issues before production.
- **Consistency**: The tool sets standards for code quality, ensuring overall improvement across projects.
- Business Scaling: SonarQube can evaluate multiple projects at once, supporting organizational growth.
- **Enhanced Developer Skills**: Regular feedback helps developers improve their coding practices and fosters continuous learning.

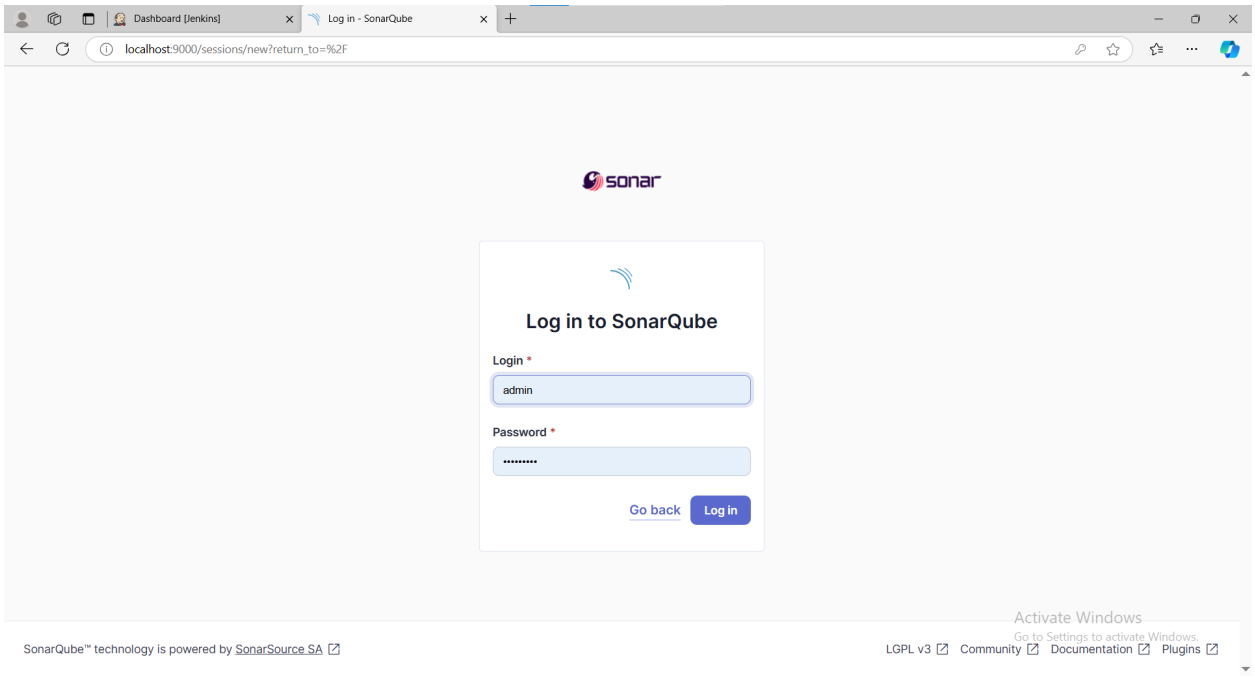
Steps:

Open up Jenkins Dashboard on localhost, port 8080
Run SonarQube in a Docker container using this command -
$ docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000
Sonarqube:latest

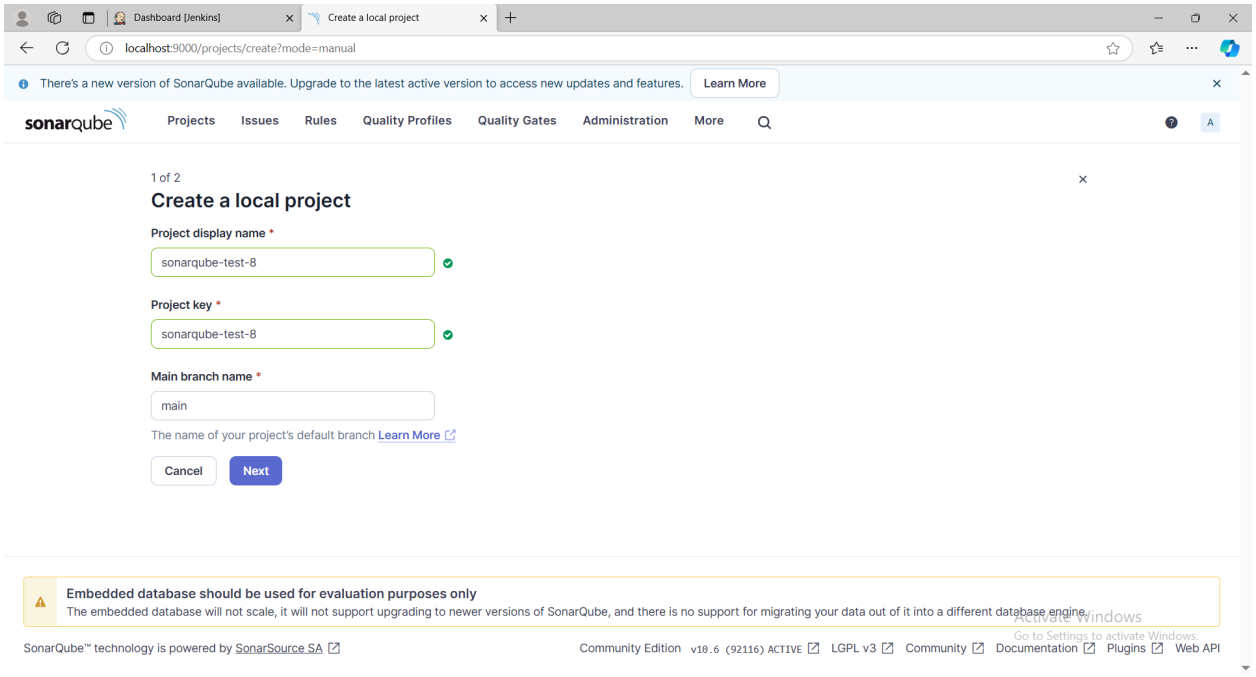
```
C:\Windows\system32>docker start sonarqube
sonarqube
```

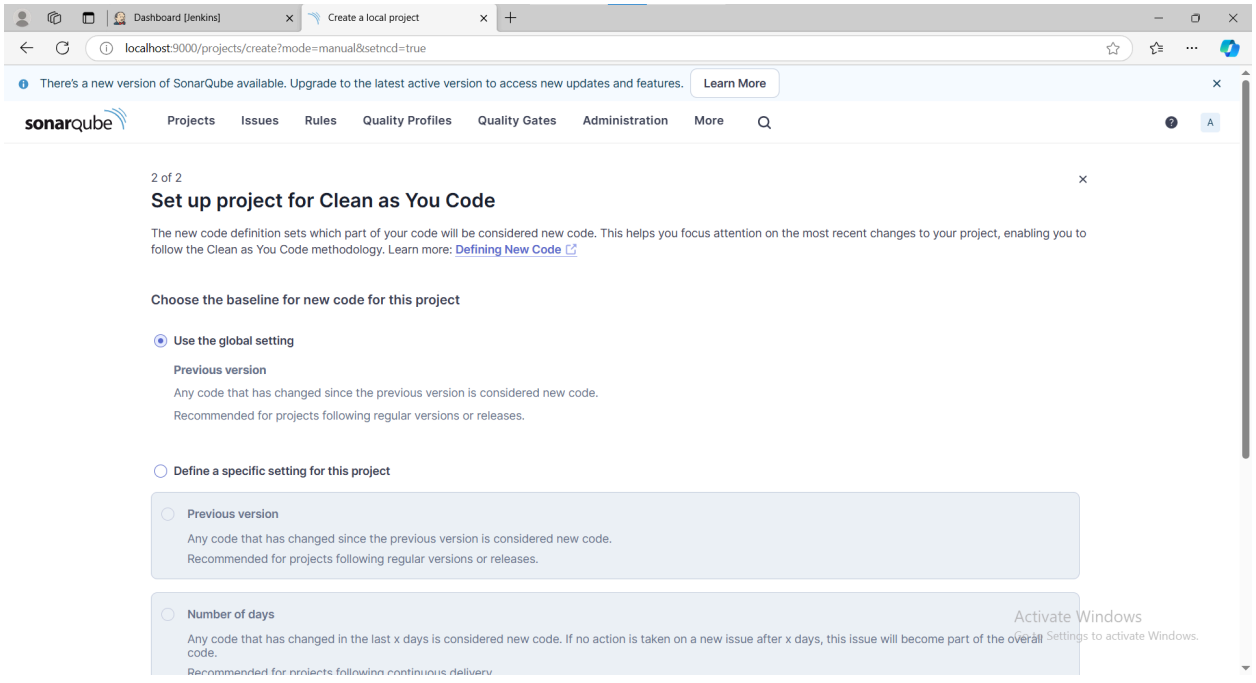


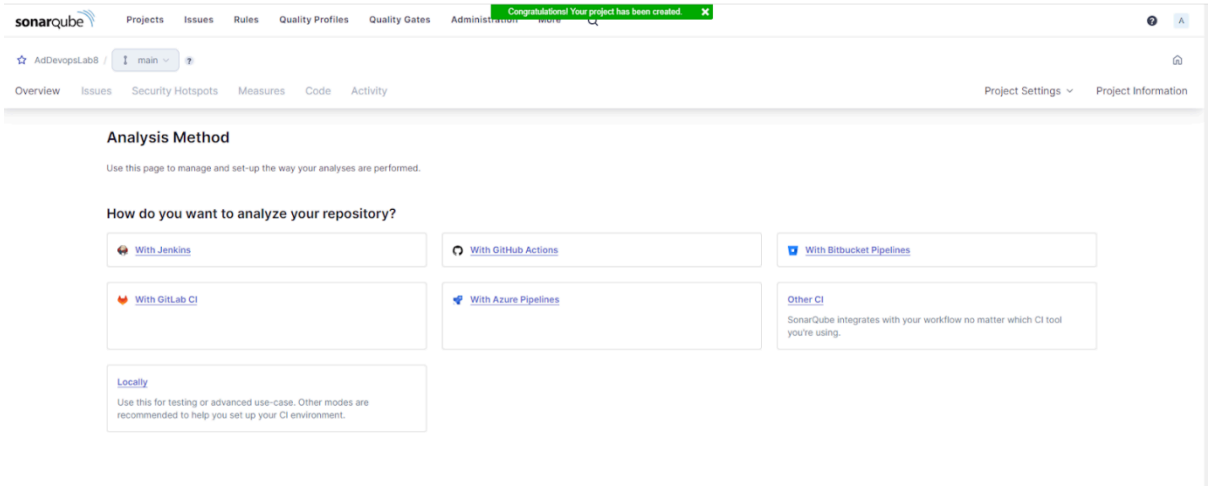
sonarqube



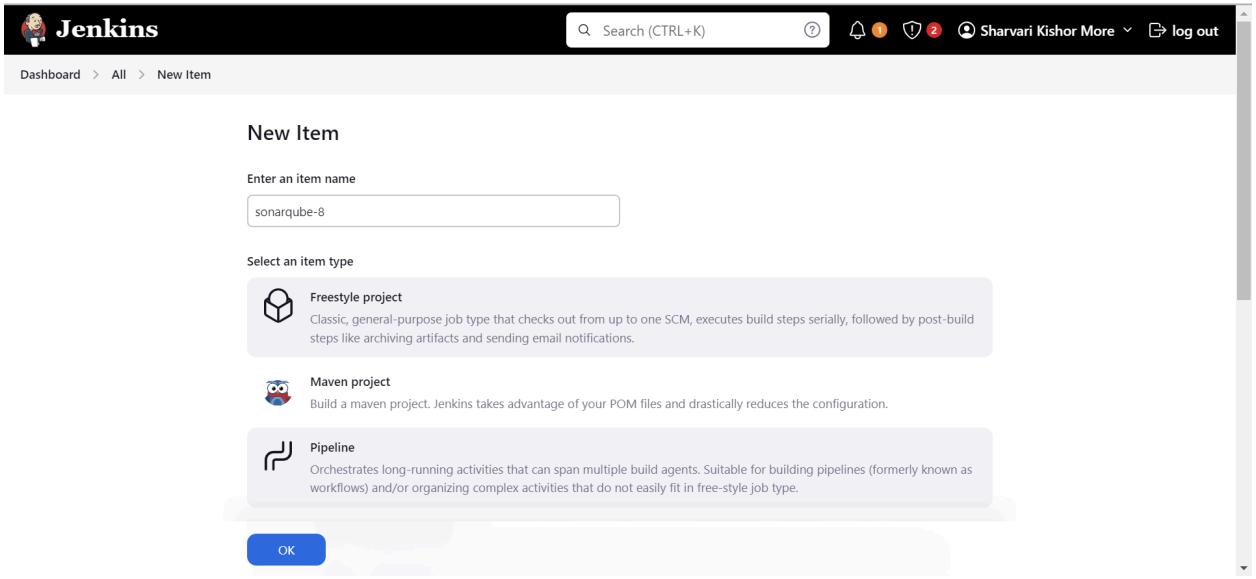
Create project manually


Dashboard [Jenkins]
Create a local project
localhost:9000/projects/create?mode=manual
There's a new version of SonarQube available. Upgrade to the latest active version to access new updates and features.
Learn More
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
1 of 2
Create a local project
Project display name *
sonarqube-test-8
Project key *
sonarqube-test-8
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.
SonarQube™ technology is powered by SonarSource SA
Community Edition v10.6 (92116) ACTIVE
LGPL v3
Community
Documentation
Plugins
Web API


Dashboard [Jenkins]
Create a local project
localhost:9000/projects/create?mode=manual&setncd=true
There's a new version of SonarQube available. Upgrade to the latest active version to access new updates and features.
Learn More
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
2 of 2
Set up project for Clean as You Code
The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, enabling you to follow the Clean as You Code methodology. Learn more: Defining New Code
Choose the baseline for new code for this project
Use the global setting
Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.
Define a specific setting for this project
Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.
Number of days
Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.
Recommended for projects following continuous delivery



In Jenkins create a pipeline here named “SonarQube”



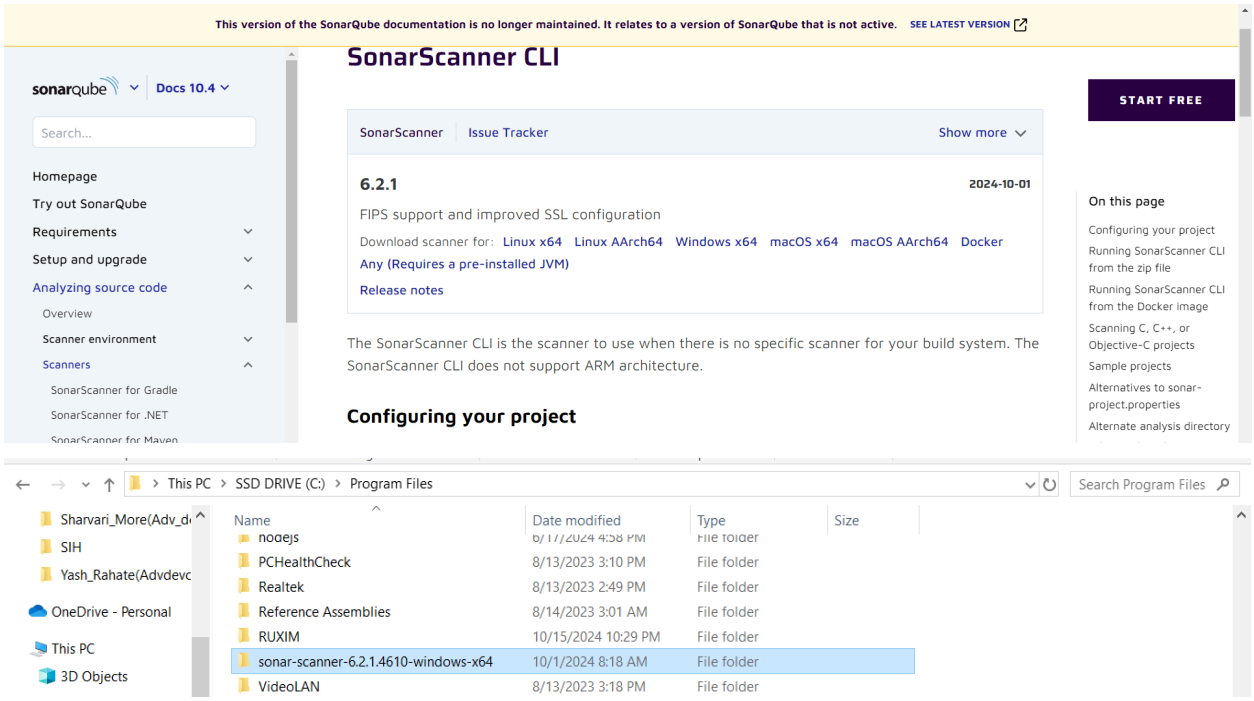
Enter the following in pipeline script:

```
pipeline {
    agent any
    stages {
        stage('SonarQube Analysis') {
            steps {
                script {
                    bat "\"C:\\Program
Files\\sonar-scanner-6.2.1.4610-windows-x64\\bin\\sonar-scanner.bat\" " +
                    "-D sonar.login=admin " +
                    "-D sonar.password=yash@5565 " +
                    "-D sonar.projectKey=sonarqube-test-8 " +
                    "-D sonar.exclusions=vendor/**,resources/**,**/*.java " +
                    "-D sonar.host.url=http://127.0.0.1:9000/"
```

```
            }
        }
    }
  }
}
```

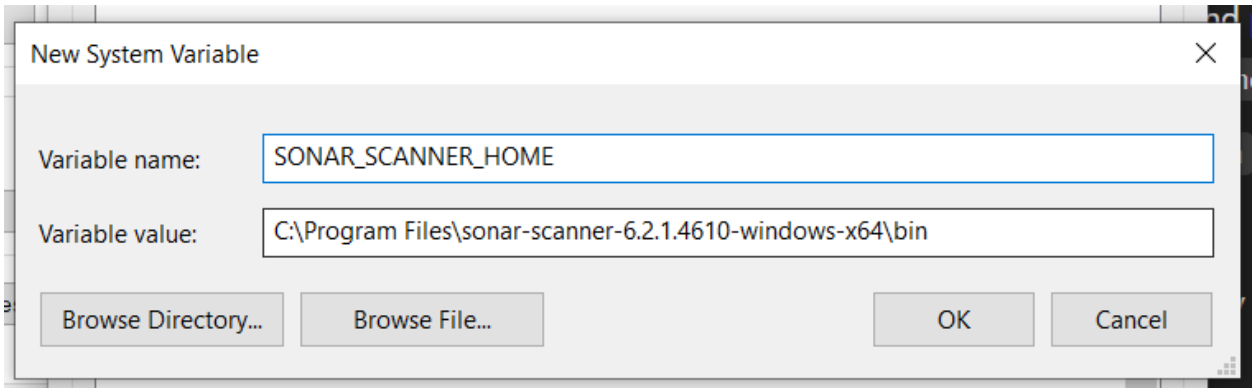
Download SonarQube Scanner:**Download SonarQube Scanner**:

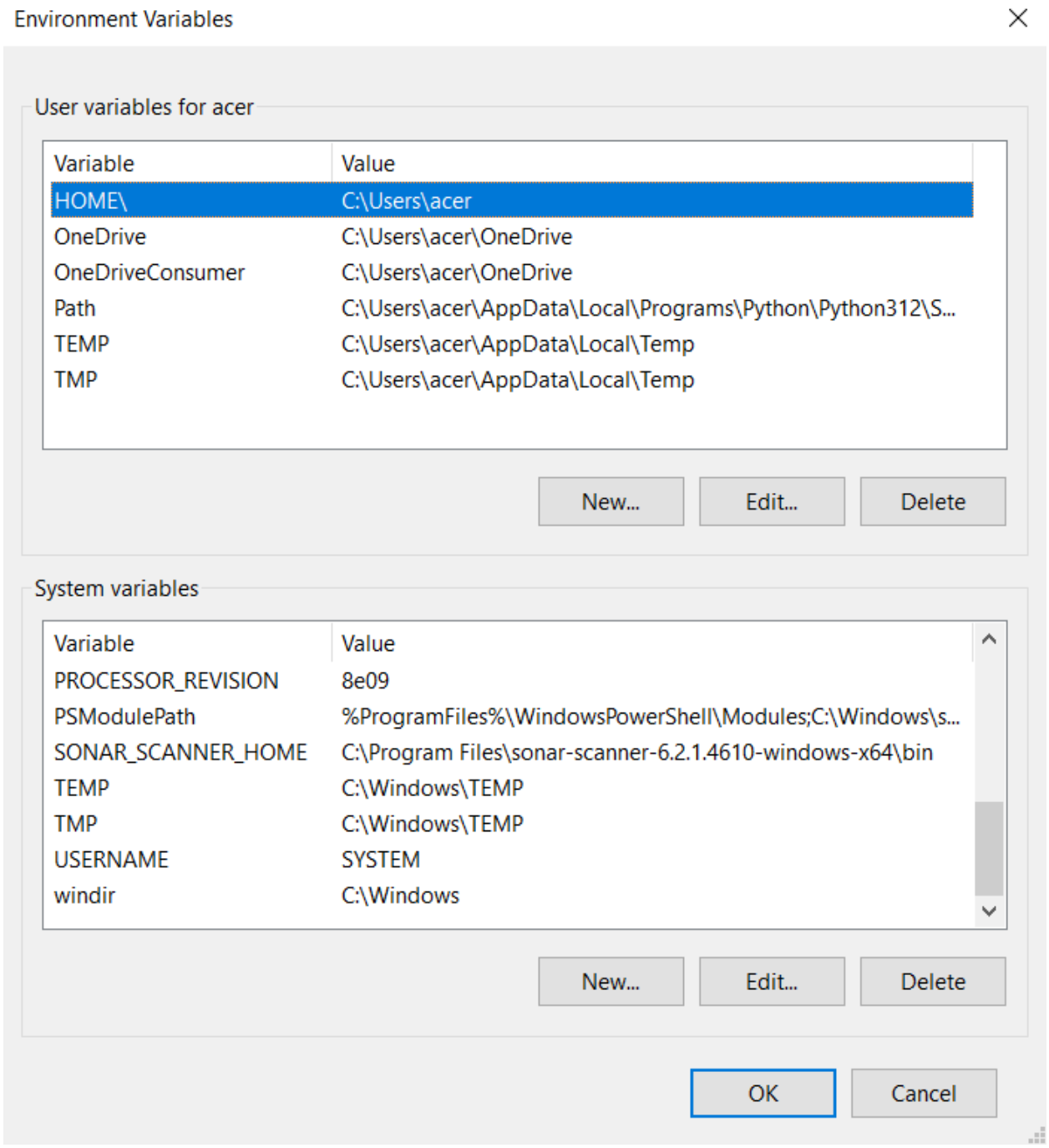
- Visit the official SonarQube website at SonarQube Scanner Downloads.
- Choose the version for your operating system (e.g., Windows, Linux).
- Extract the downloaded archive to a directory of your choice, e.g., C:\Program Files\sonar-scanner.

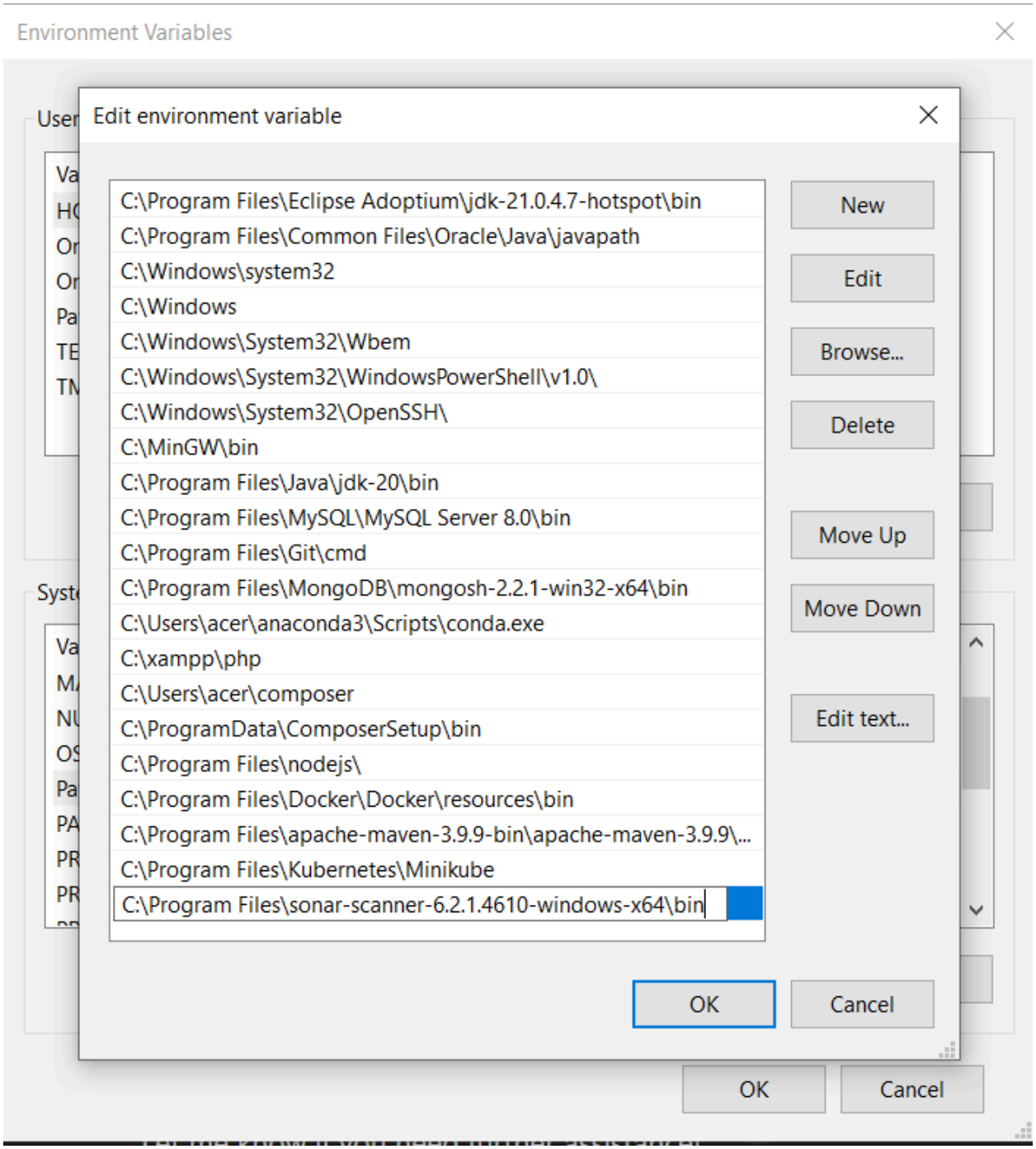


Set Environment Variables:

- Open your system's environment variables settings.
- Add a new environment variable for SONAR_SCANNER_HOME, and point it to the folder where you extracted the scanner (e.g., C:\Program Files\sonar-scanner).
- Edit the PATH variable and append %SONAR_SCANNER_HOME%\bin.


New System Variable
Variable name:
SONAR_SCANNER_HOME
Variable value:
C:\Program Files\sonar-scanner-6.2.1.4610-windows-x64\bin
Browse Directory...
Browse File...
OK
Cancel


Environment Variables
User variables for acer
Variable
Value
HOME\
C:\Users\acer
OneDrive
C:\Users\acer\OneDrive
OneDriveConsumer
C:\Users\acer\OneDrive
Path
C:\Users\acer\AppData\Local\Programs\Python\Python312\S...
TEMP
C:\Users\acer\AppData\Local\Temp
TMP
C:\Users\acer\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
PROCESSOR_REVISION
8e09
PSModulePath
%ProgramFiles%\WindowsPowerShell\Modules;C:\Windows\s...
SONAR_SCANNER_HOME
C:\Program Files\sonar-scanner-6.2.1.4610-windows-x64\bin
TEMP
C:\Windows\TEMP
TMP
C:\Windows\TEMP
USERNAME
SYSTEM
windir
C:\Windows
New...
Edit...
Delete
OK
Cancel



Test Installation:

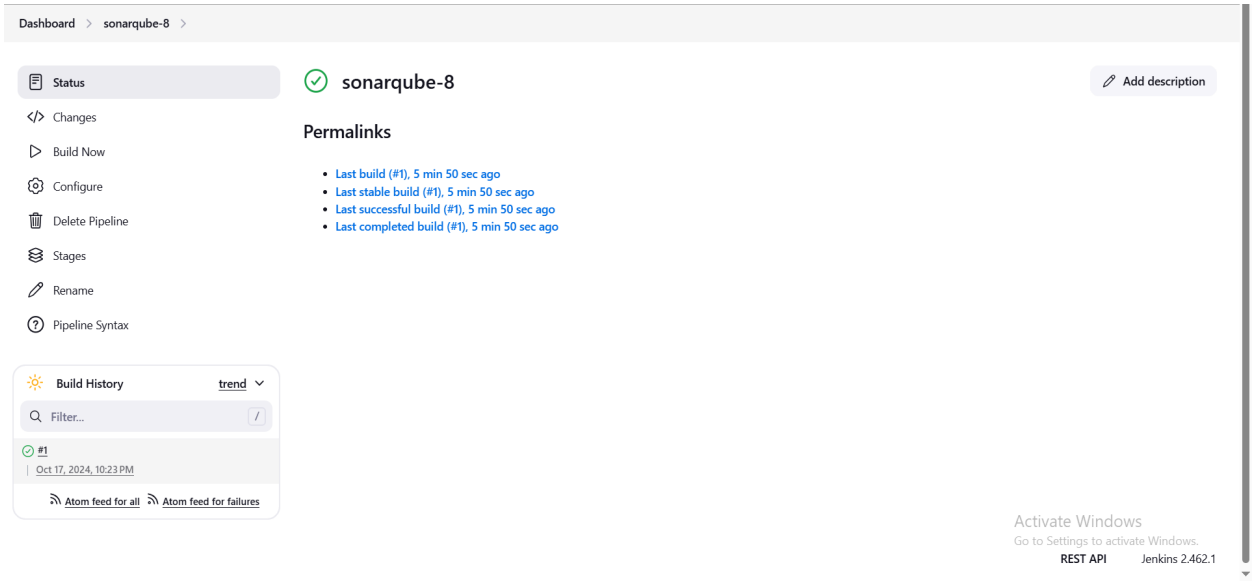
- Open a command prompt and run sonar-scanner -v to verify that the scanner is correctly installed.

```
C:\Users\acer>sonar-scanner -v
22:20:11.692 INFO  Scanner configuration file: C:\Program Files\sonar-scanner-6.2.1.4610-windows-x64\bin\..\conf\sonar-s
canner.properties
22:20:11.706 INFO  Project root configuration file: NONE
22:20:11.742 INFO  SonarScanner CLI 6.2.1.4610
22:20:11.747 INFO  Java 17.0.12 Eclipse Adoptium (64-bit)
22:20:11.749 INFO  Windows 10 10.0 amd64
```

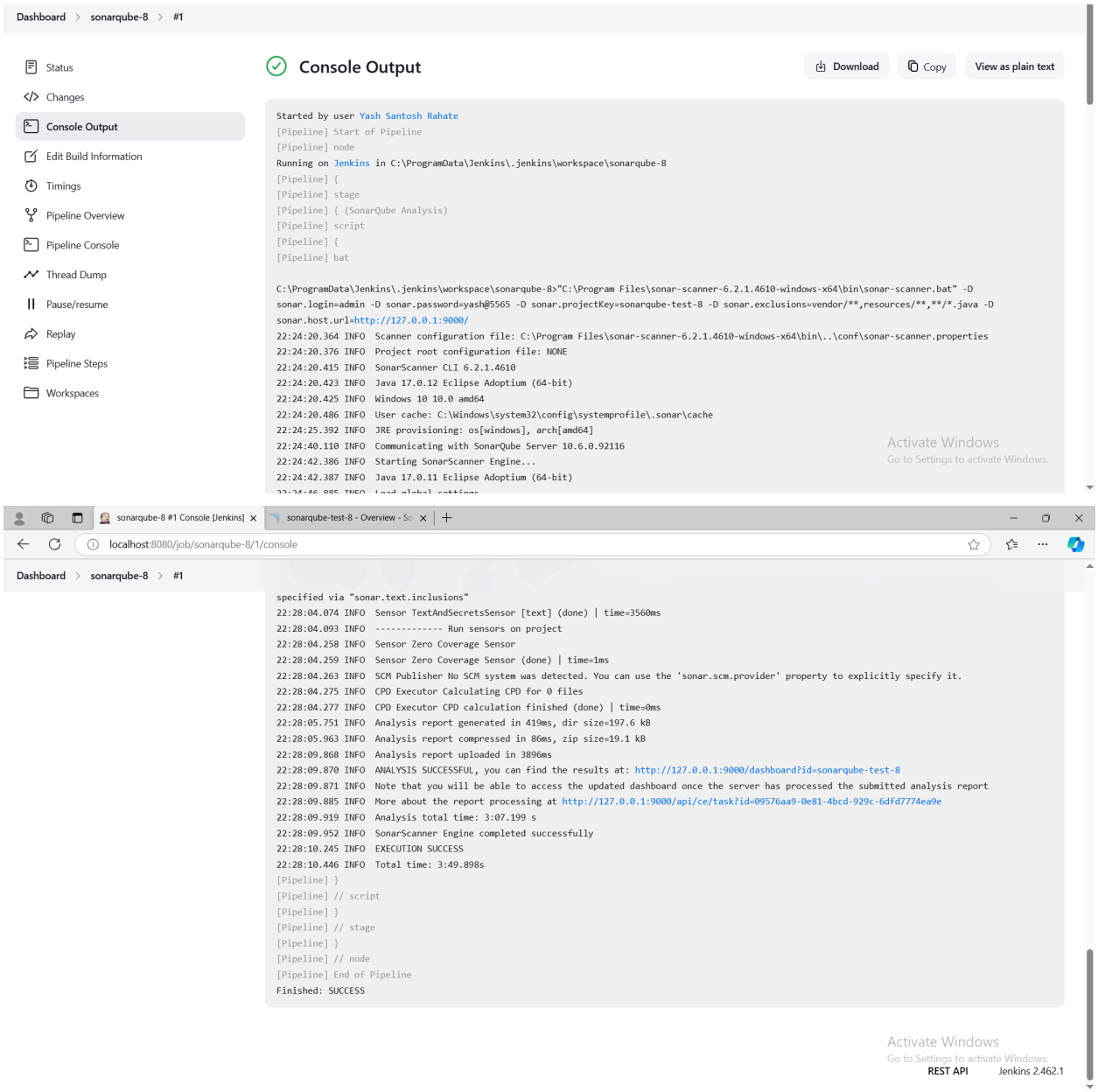
```
pipeline {
    agent any
    stages {
        stage('SonarQube Analysis') {
            steps {
                script {
                    bat "\"C:\\Program
Files\\sonar-scanner-6.2.1.4610-windows-x64\\bin\\sonar-scanner.bat\" " +
                    "-D sonar.login=admin " +
                    "-D sonar.password=yash@5565 " +
                    "-D sonar.projectKey=sonarqube-test-8 " +
                    "-D sonar.exclusions=vendor/**,resources/**,**/*.java " +
                    "-D sonar.host.url=http://127.0.0.1:9000/"
                }
            }
        }
    }
}
```

It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

Build and run:

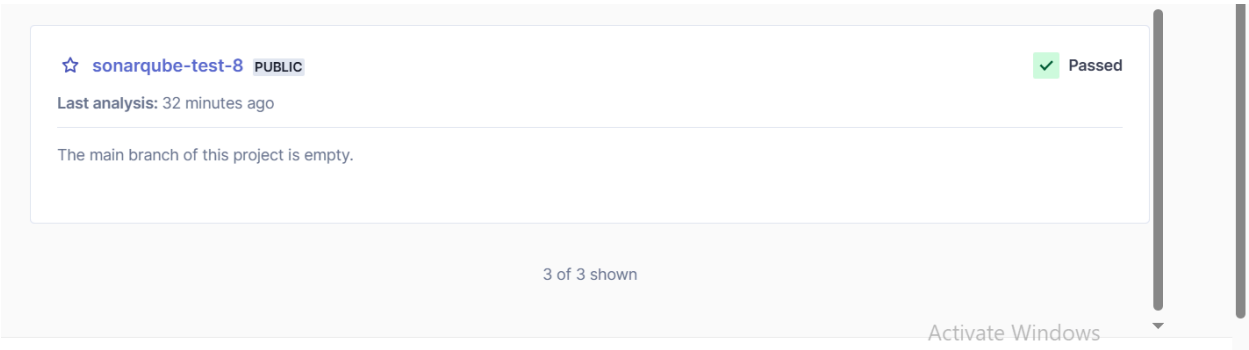


Console output:


Dashboard > sonarqube-8 > #1
Status
Changes
Console Output
Edit Build Information
Timings
Pipeline Overview
Pipeline Console
Thread Dump
Pause/resume
Replay
Pipeline Steps
Workspaces
Console Output
Download
Copy
View as plain text
Started by user Yash Santosh Rahate
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-8
[Pipeline] {
[Pipeline] stage
[Pipeline] { (SonarQube Analysis)
[Pipeline] script
[Pipeline] {
[Pipeline] bat
C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-8>"C:\Program Files\sonar-scanner-6.2.1.4610-windows-x64\bin\sonar-scanner.bat" -D
sonar.login=admin -D sonar.password=yash@5565 -D sonar.projectKey=sonarqube-test-8 -D sonar.exclusions=vendor/**,resources/**,**/*.java -D
sonar.host.url=http://127.0.0.1:9000/
22:24:20.364 INFO Scanner configuration file: C:\Program Files\sonar-scanner-6.2.1.4610-windows-x64\bin\..\conf\sonar-scanner.properties
22:24:20.376 INFO Project root configuration file: NONE
22:24:20.415 INFO SonarScanner CLI 6.2.1.4610
22:24:20.423 INFO Java 17.0.12 Eclipse Adoptium (64-bit)
22:24:20.425 INFO Windows 10 10.0 amd64
22:24:20.486 INFO User cache: C:\Windows\system32\config\systemprofile\.sonar\cache
22:24:25.392 INFO JRE provisioning: os[windows], arch[amd64]
22:24:40.110 INFO Communicating with SonarQube Server 10.6.0.92116
22:24:42.386 INFO Starting SonarScanner Engine...
22:24:42.387 INFO Java 17.0.11 Eclipse Adoptium (64-bit)
Activate Windows
Go to Settings to activate Windows.

sonarqube-8 #1 Console [Jenkins]
sonarqube-test-8 - Overview
localhost:8080/job/sonarqube-8/1/console
Dashboard > sonarqube-8 > #1
specified via "sonar.text.inclusions"
22:28:04.074 INFO Sensor TextAndSecretsSensor [text] (done) | time=3560ms
22:28:04.093 INFO ------------ Run sensors on project
22:28:04.258 INFO Sensor Zero Coverage Sensor
22:28:04.259 INFO Sensor Zero Coverage Sensor (done) | time=1ms
22:28:04.263 INFO SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
22:28:04.275 INFO CPD Executor Calculating CPD for 0 files
22:28:04.277 INFO CPD Executor CPD calculation finished (done) | time=0ms
22:28:05.751 INFO Analysis report generated in 419ms, dir size=197.6 kB
22:28:05.963 INFO Analysis report compressed in 86ms, zip size=19.1 kB
22:28:09.868 INFO Analysis report uploaded in 3896ms
22:28:09.870 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube-test-8
22:28:09.871 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
22:28:09.885 INFO More about the report processing at http://127.0.0.1:9000/api/ce/task?id=09576aa9-0e81-4bcd-929c-6dfd7774ea9e
22:28:09.919 INFO Analysis total time: 3:07.199 s
22:28:09.952 INFO SonarScanner Engine completed successfully
22:28:10.245 INFO EXECUTION SUCCESS
22:28:10.446 INFO Total time: 3:49.898s
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
Activate Windows
Go to Settings to activate Windows.
REST API
Jenkins 2.462.1

sonarqube:



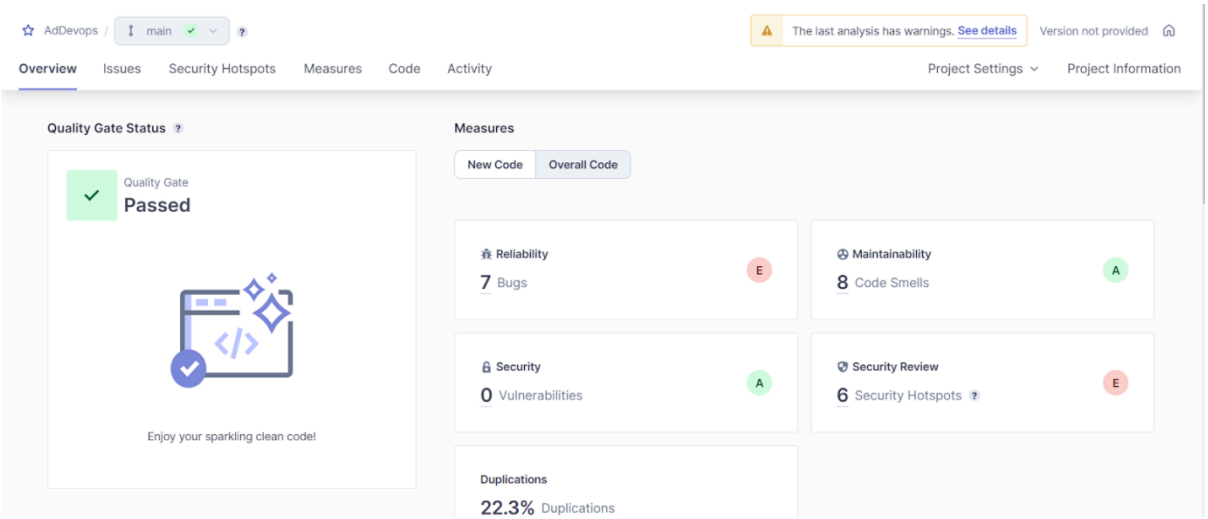
Definition

Pipeline script

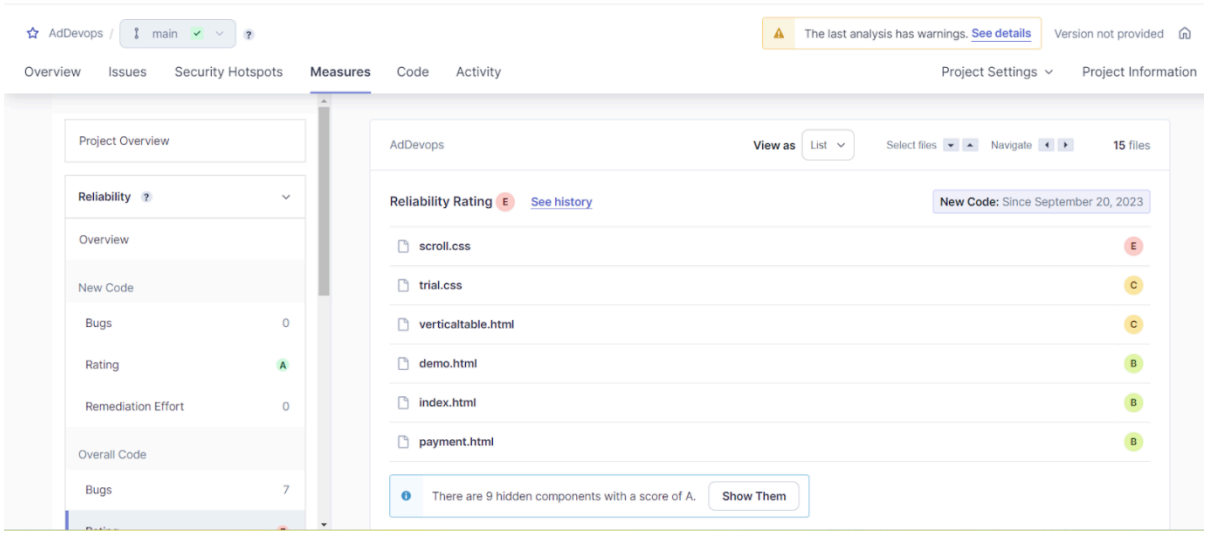
Script ?

try sample Pipeline...

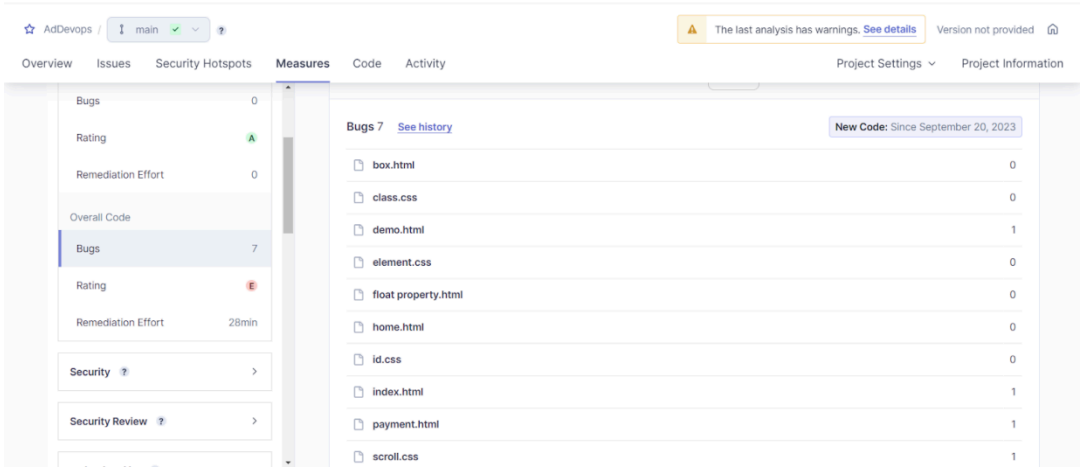
```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                git url: 'https://github.com/YashRahate/Netflix_frontend_clone.git', branch: 'main'
            }
        }

        stage('SonarQube Analysis') {
            steps {
                script {
                    bat "\"C:\\Program Files\\sonar-scanner-6.2.1.4610-windows-x64\\bin\\sonar-scanner.bat\" " +
                    "-D sonar.login=admin " +
                    "-D sonar.password=yash@5565 " +
                    "-D sonar.projectKey=sonarqube-test-8 " +
                    "-D sonar.exclusions=vendor/**,resources/**,**/*.java " +
                    "-D sonar.host.url=http://127.0.0.1:9000/"
                }
            }
        }
    }
}
```



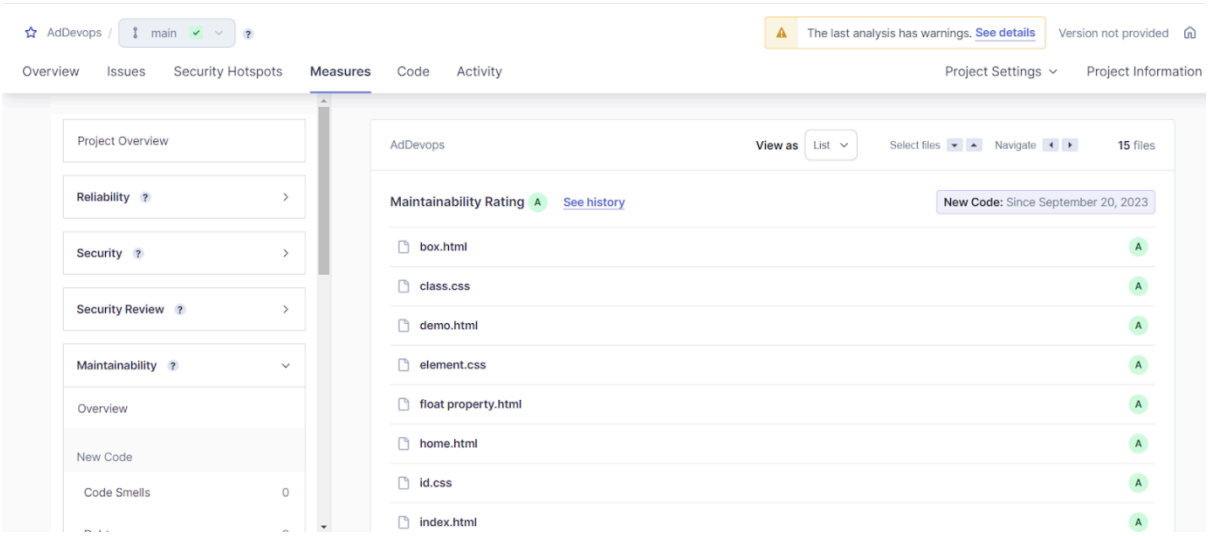
Reliability:



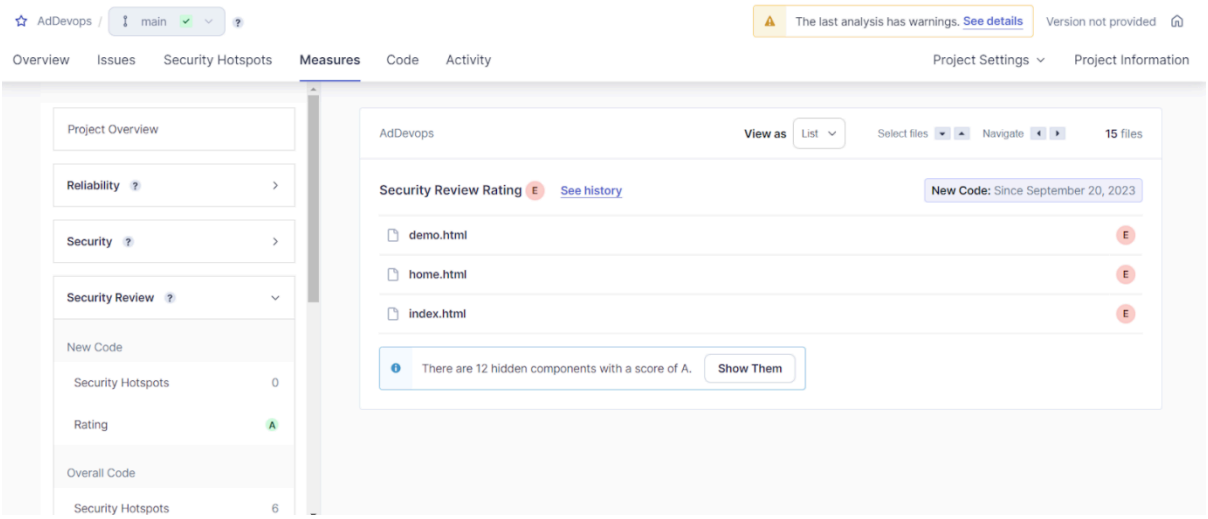
Bugs:



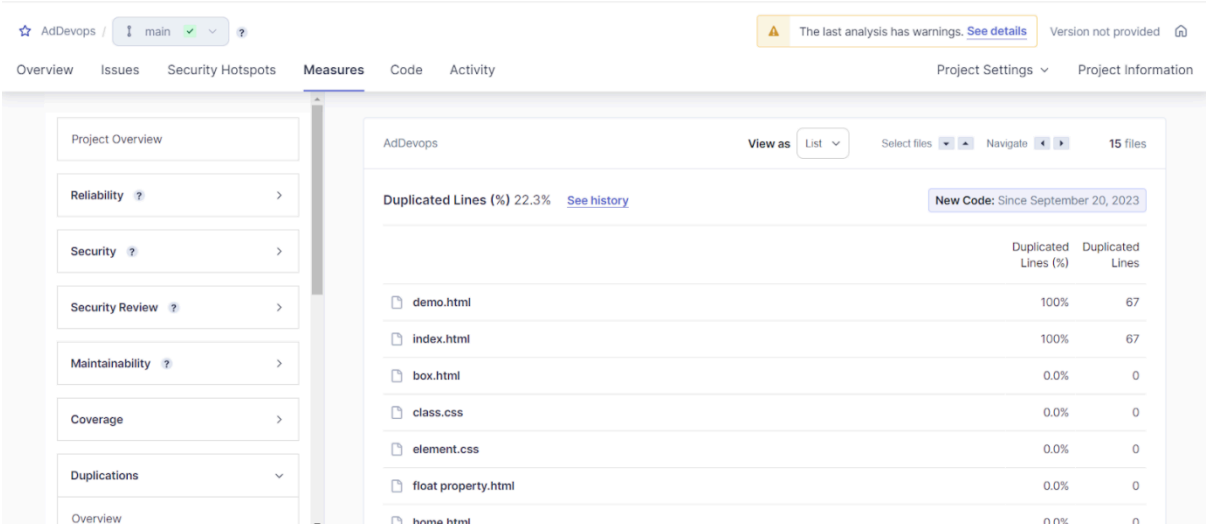
Maintanaibility:



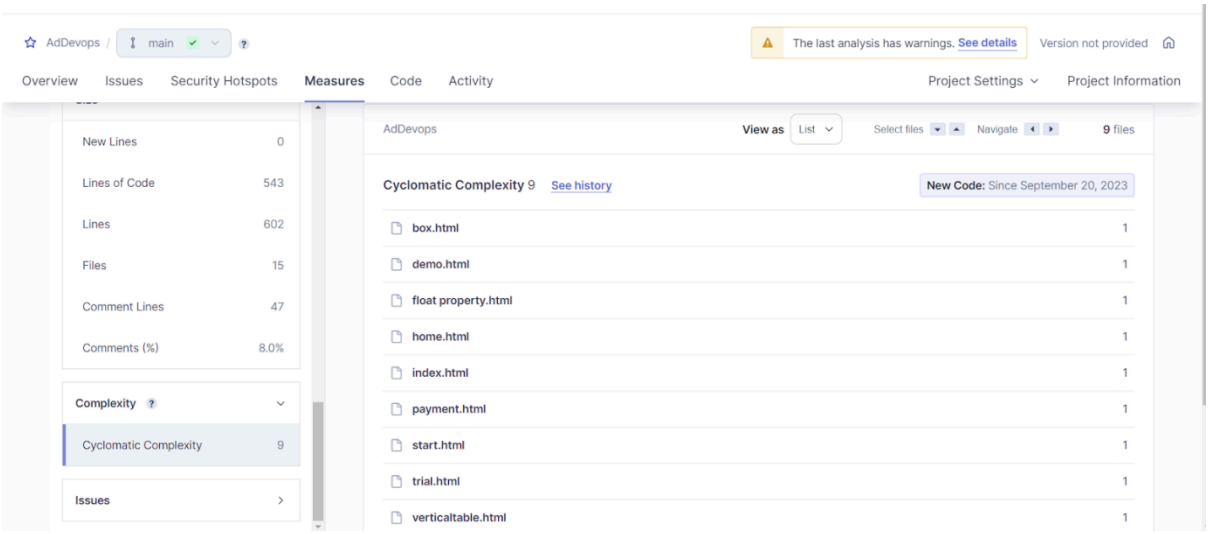
Security:



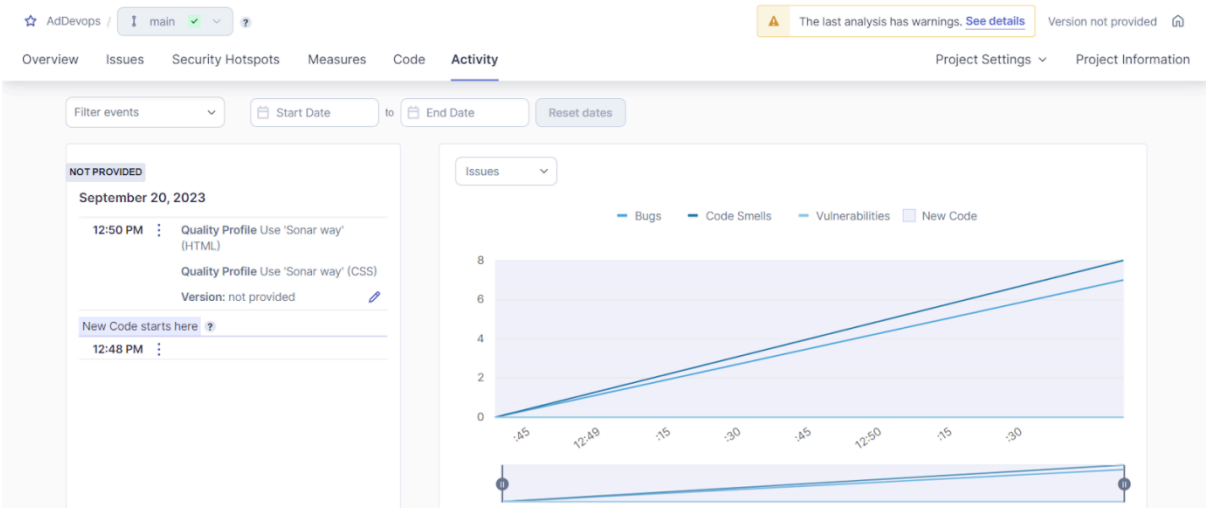
Duplications:

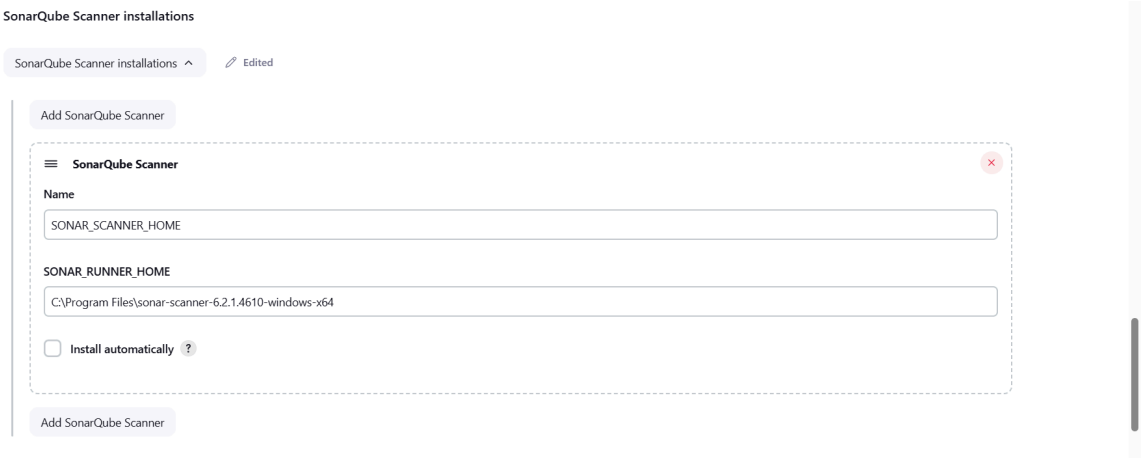


Cyclomatic complexity:



Activity:





Conclusion:

Integrating Jenkins with SonarQube in a CI/CD pipeline allows developers to automatically analyze code for bugs and security vulnerabilities during the development process. This helps ensure that only high-quality code is delivered, making applications more secure and reliable.