# Experiment No 8

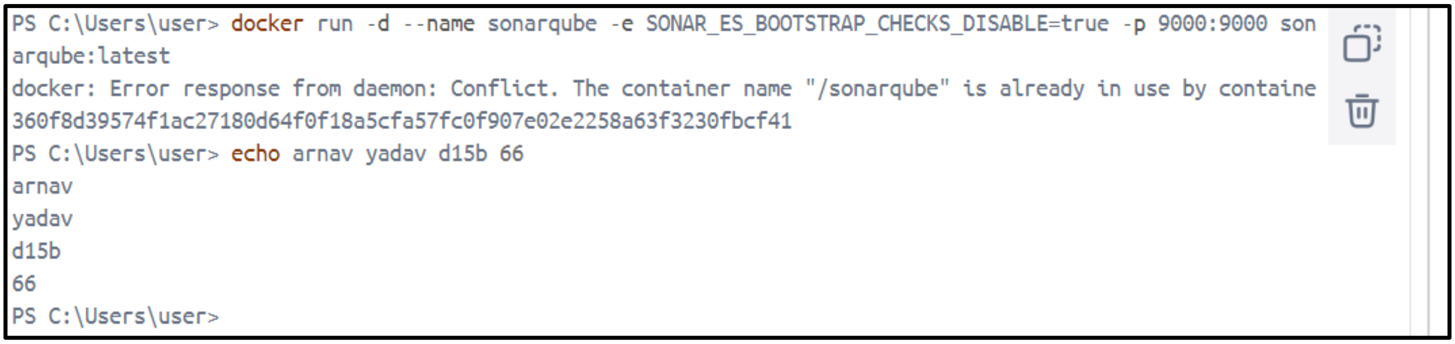
AIM: Integrating Jenkins with SonarQube.

## Theory:

SonarQube is a powerful tool for static code analysis, enabling developers to identify and fix code quality issues, security vulnerabilities, and technical debt. Integrating it with Jenkins allows for automated code analysis during the continuous integration process. This integration enhances code quality by providing immediate feedback to developers, ensuring that code adheres to predefined quality standards.

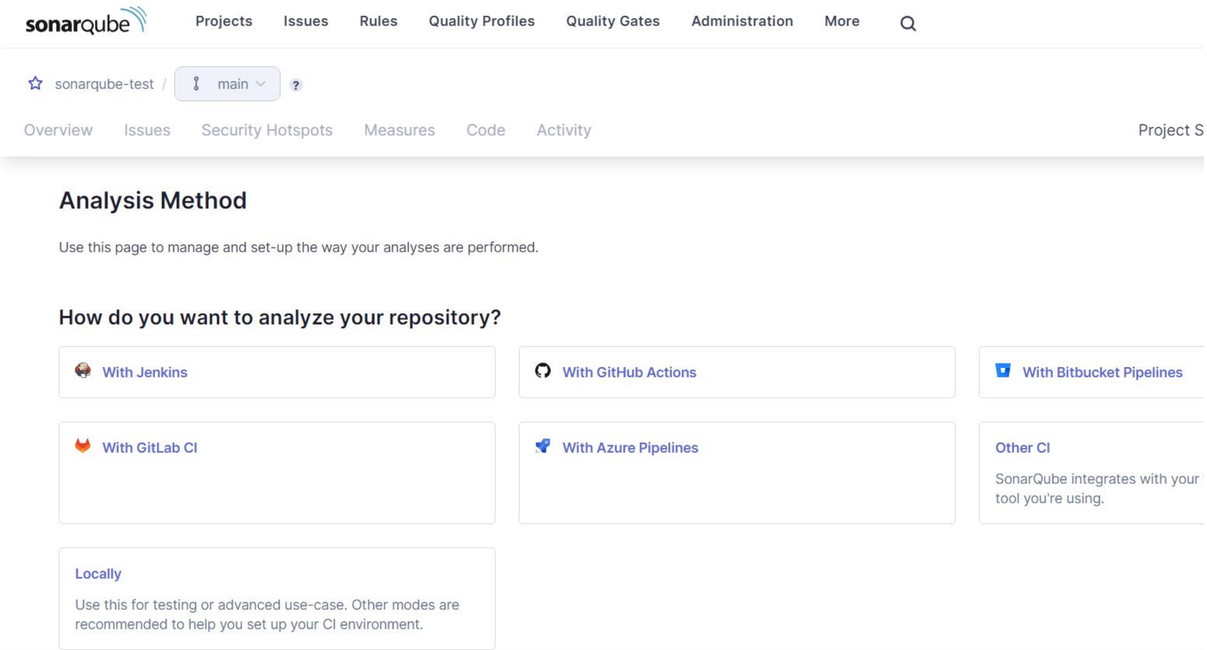
Steps:
Open up Jenkins Dashboard on localhost, port 8080
Run SonarQube in a Docker container using this command -

```
$ docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p
9000:9000
sonarqube:latest
```



```
PS C:\Users\user> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 son
arqube:latest
docker: Error response from daemon: Conflict. The container name "/sonarqube" is already in use by containe
360f8d39574f1ac27180d64f0f18a5cfa57fc0f907e02e2258a63f3230fbcf41
PS C:\Users\user> echo arnav yadav d15b 66
arnav
yadav
d15b
66
PS C:\Users\user>
```

Create project manually



## Analysis Method

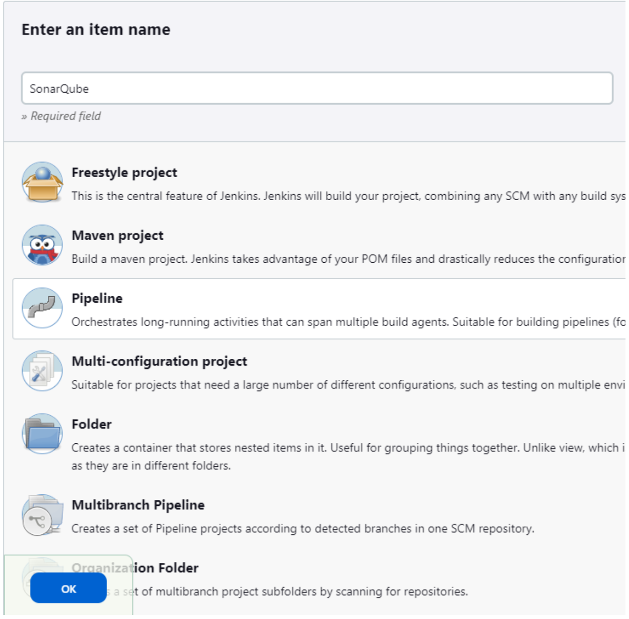
Use this page to manage and set-up the way your analyses are performed.

### How do you want to analyze your repository?

SonarQube integrates with your tool you're using.

Use this for testing or advanced use-case. Other modes are recommended to help you set up your CI environment.

In Jenkins create a pipeline here named “SonarQube”


Enter an item name

SonarQube

» Required field

**Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build sys

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

**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (fo

**Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple envi

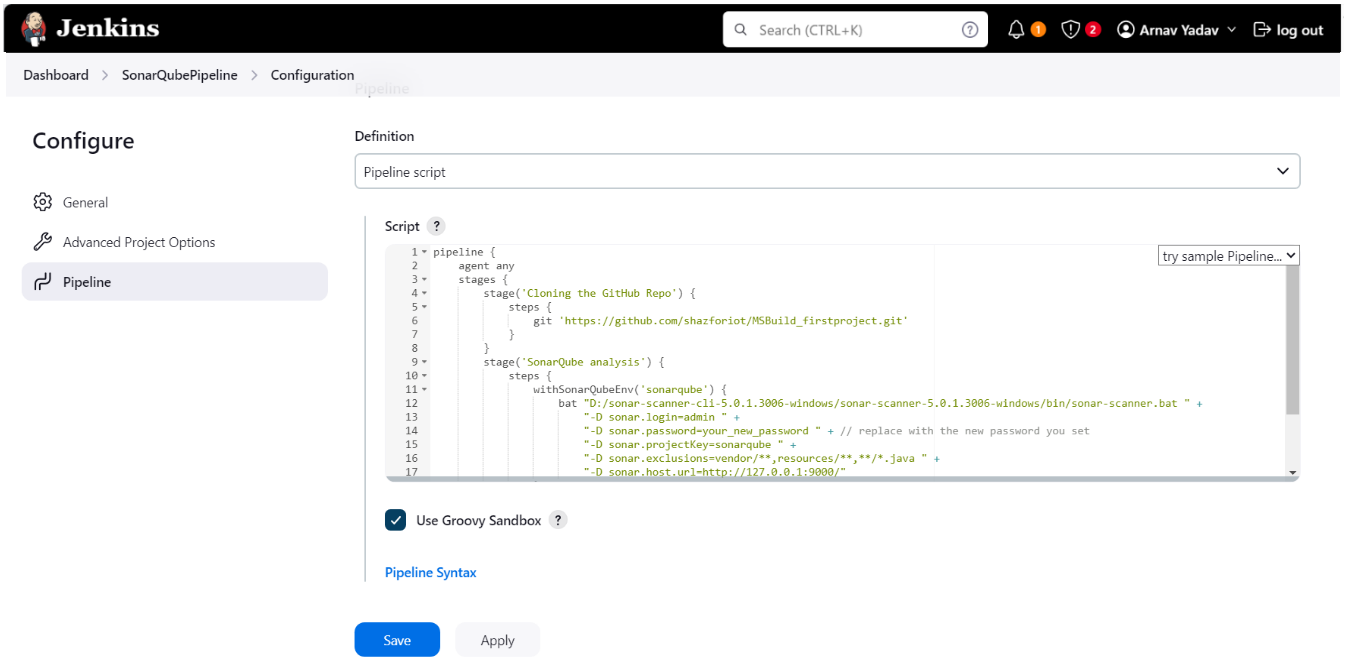
**Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which i as they are in different folders.

**Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**
s a set of multibranch project subfolders by scanning for repositories.

Enter the following in pipeline script:

```
node {
    stage('Cloning the GitHub Repo') {
        git 'https://github.com/PrajaktaUpadhye6/MSBuild_firstproject.git'
    }
    stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube') {
            bat "D:/sonar-scanner-cli-5.0.1.3006-windows/sonar-scanner-5.0.1.3006-
windows/bin/sonar-scanner.bat \
                -D sonar.login=admin \
                -D sonar.password=abc \
                -D sonar.projectKey=AdDevops \
                -D sonar.exclusions=vendor/**,resources/**,**/*.java \
                -D sonar.host.url=http://127.0.0.1:9000/"
        }
    }
}
```



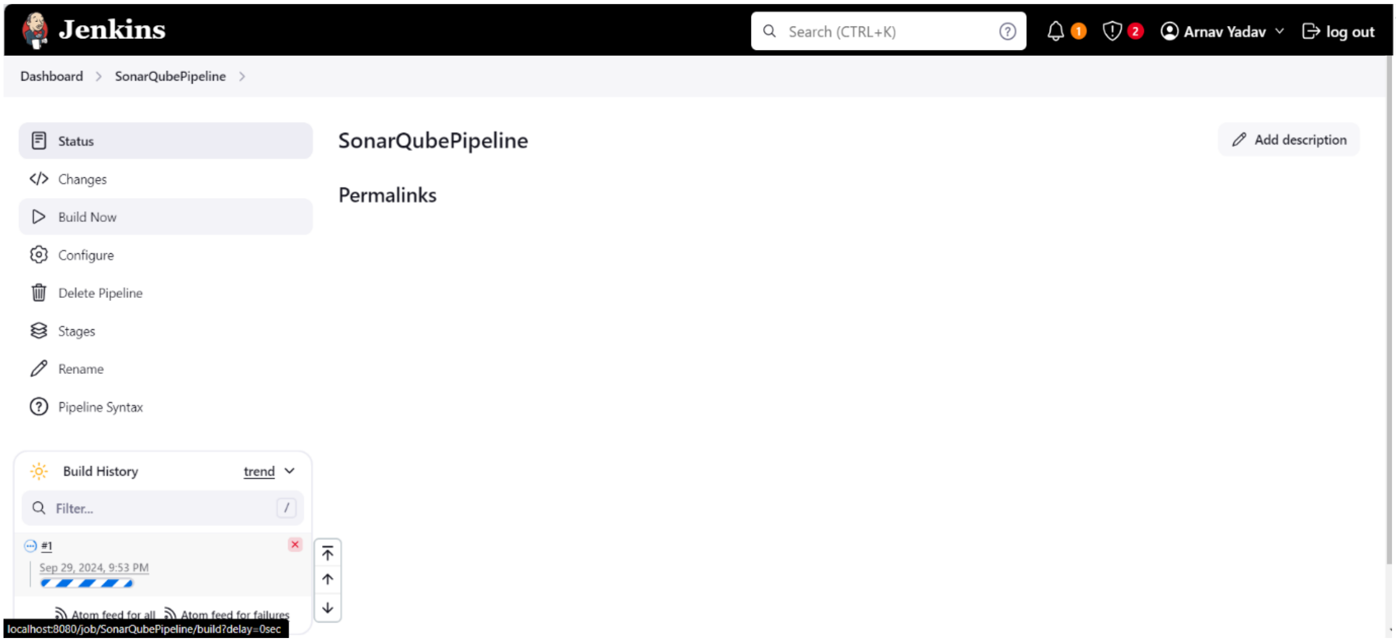
Definition: Pipeline script

Script

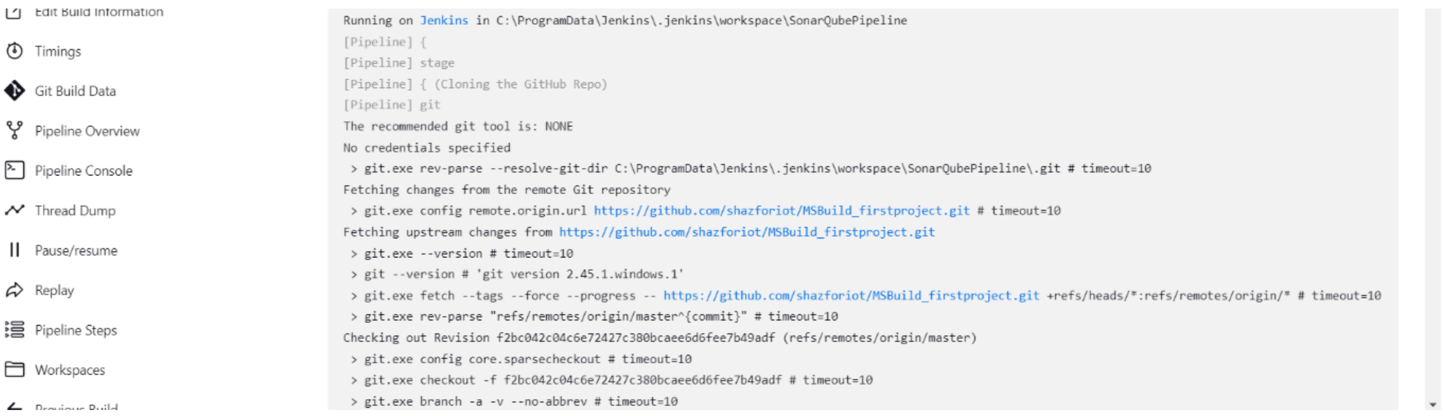
```
pipeline {
    agent any
    stages {
        stage('Cloning the GitHub Repo') {
            steps {
                git 'https://github.com/shazforiot/MSBuild_firstproject.git'
            }
        }
        stage('SonarQube analysis') {
            steps {
                withSonarQubeEnv('sonarqube') {
                    bat "D:/sonar-scanner-cli-5.0.1.3006-windows/sonar-scanner-5.0.1.3006-windows/bin/sonar-scanner.bat " +
                        "-D sonar.login=admin " +
                        "-D sonar.password=your_new_password " + // replace with the new password you set
                        "-D sonar.projectKey=sonarqube " +
                        "-D sonar.exclusions=vendor/**,resources/**,**/*.java " +
                        "-D sonar.host.url=http://127.0.0.1:9000/"
```

☑ Use Groovy Sandbox

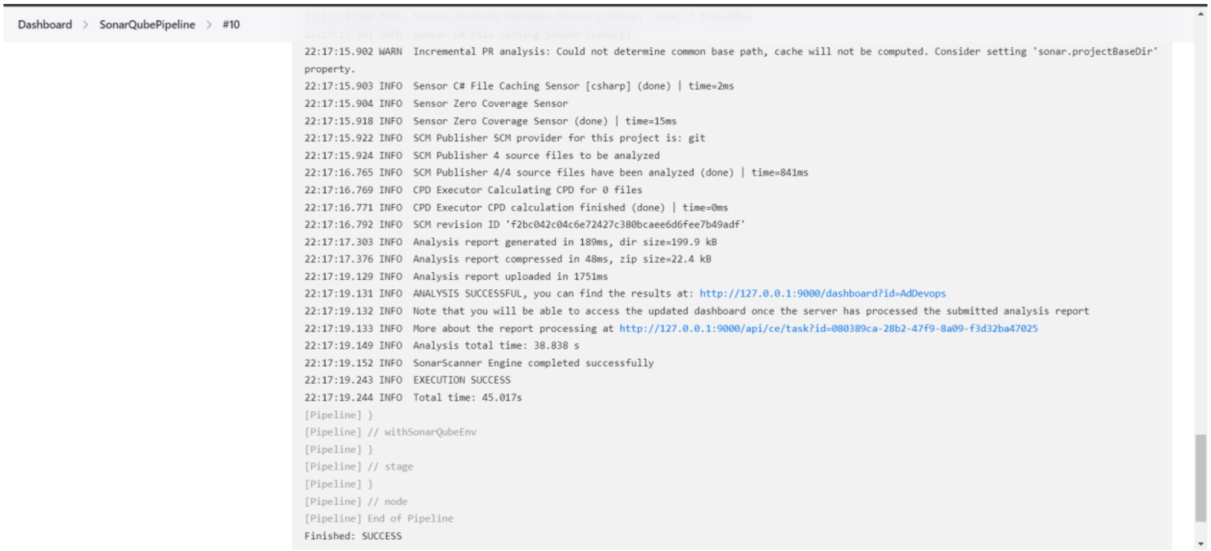
It is a java sample project which has a lot of repetitions and issues that will be detected by
SonarQube.
Build and run:



Console output:

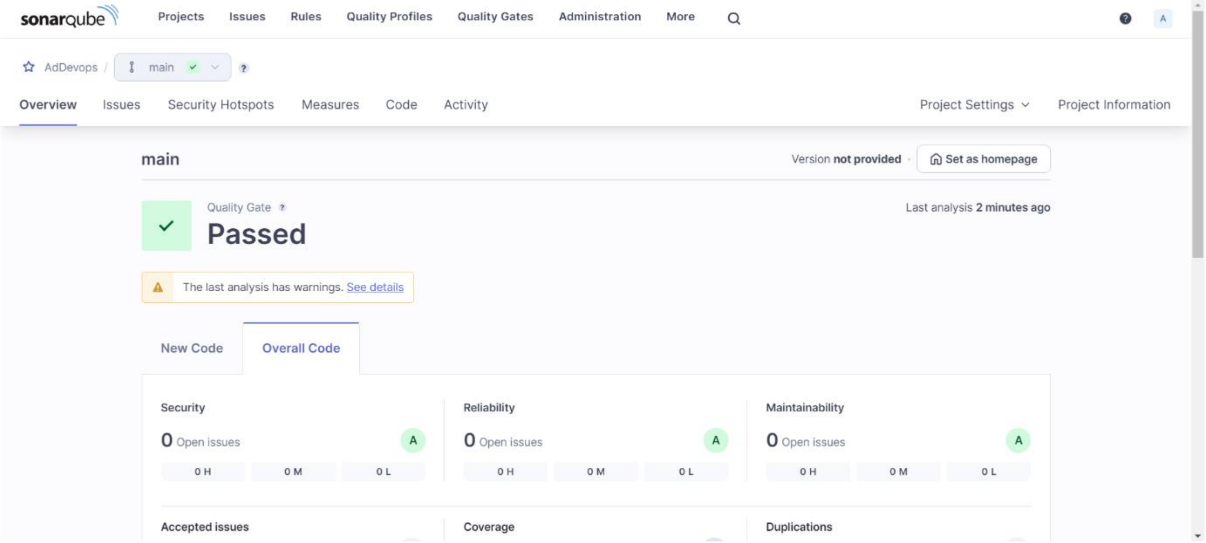


```
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\SonarQubePipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning the GitHub Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SonarQubePipeline\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.45.1.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
 > git.exe branch -a -v --no-abbrev # timeout=10
```



```
22:17:15.900 INFO  Sensor Analysis Warnings import [csharp] (done) | time=10ms
22:17:15.901 INFO  Sensor C# File Caching Sensor [csharp]
22:17:15.902 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir'
property.
22:17:15.903 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=2ms
22:17:15.904 INFO  Sensor Zero Coverage Sensor
22:17:15.918 INFO  Sensor Zero Coverage Sensor (done) | time=15ms
22:17:15.922 INFO  SCM Publisher SCM provider for this project is: git
22:17:15.924 INFO  SCM Publisher 4 source files to be analyzed
22:17:16.765 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=841ms
22:17:16.769 INFO  CPD Executor Calculating CPD for 0 files
22:17:16.771 INFO  CPD Executor CPD calculation finished (done) | time=0ms
22:17:16.792 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
22:17:17.303 INFO  Analysis report generated in 189ms, dir size=199.9 kB
22:17:17.376 INFO  Analysis report compressed in 48ms, zip size=22.4 kB
22:17:19.129 INFO  Analysis report uploaded in 1751ms
22:17:19.131 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=AdDevops
22:17:19.132 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
22:17:19.133 INFO  More about the report processing at http://127.0.0.1:9000/api/ce/task?id=080389ca-28b2-47f9-8a09-f3d32ba47025
22:17:19.149 INFO  Analysis total time: 38.838 s
22:17:19.152 INFO  SonarScanner Engine completed successfully
22:17:19.243 INFO  EXECUTION SUCCESS
22:17:19.244 INFO  Total time: 45.017s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

sonarqube:



main

Version **not provided**

Quality Gate

**Passed**

Last analysis **2 minutes ago**

The last analysis has warnings. See details

Security
0 Open issues A
0 H 0 M 0 L

Reliability
0 Open issues A
0 H 0 M 0 L

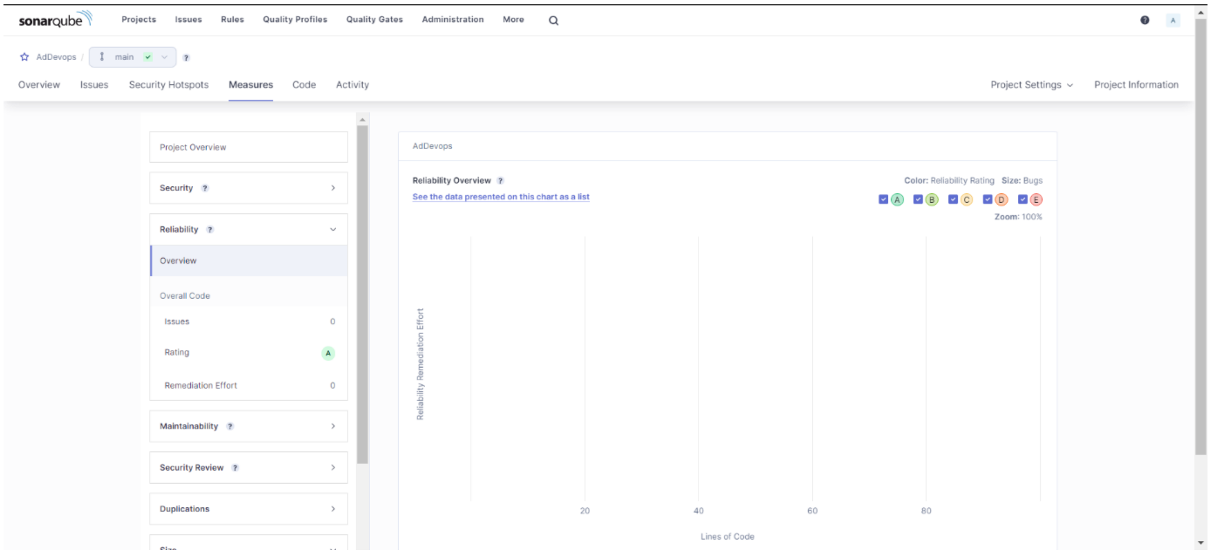
Maintainability
0 Open issues A
0 H 0 M 0 L

Accepted issues

Coverage

Duplications

Reliability:



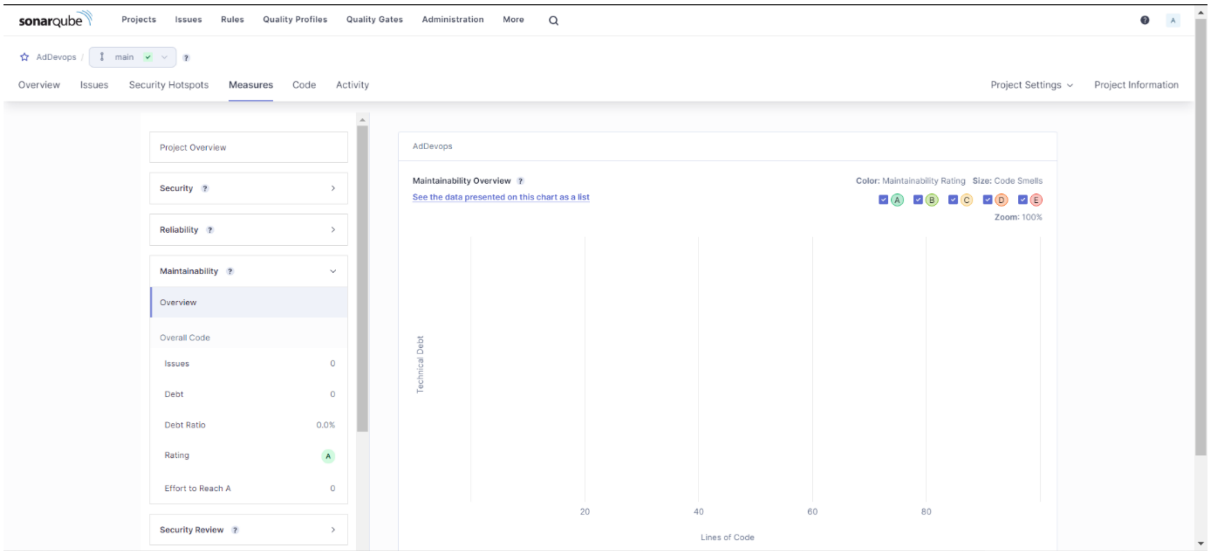
Overall Code

| | |
|---|---|
| Issues | 0 |
| Rating | A |
| Remediation Effort | 0 |

Color: Reliability Rating Size: Bugs

Zoom: 100%

## Maintanaibility:



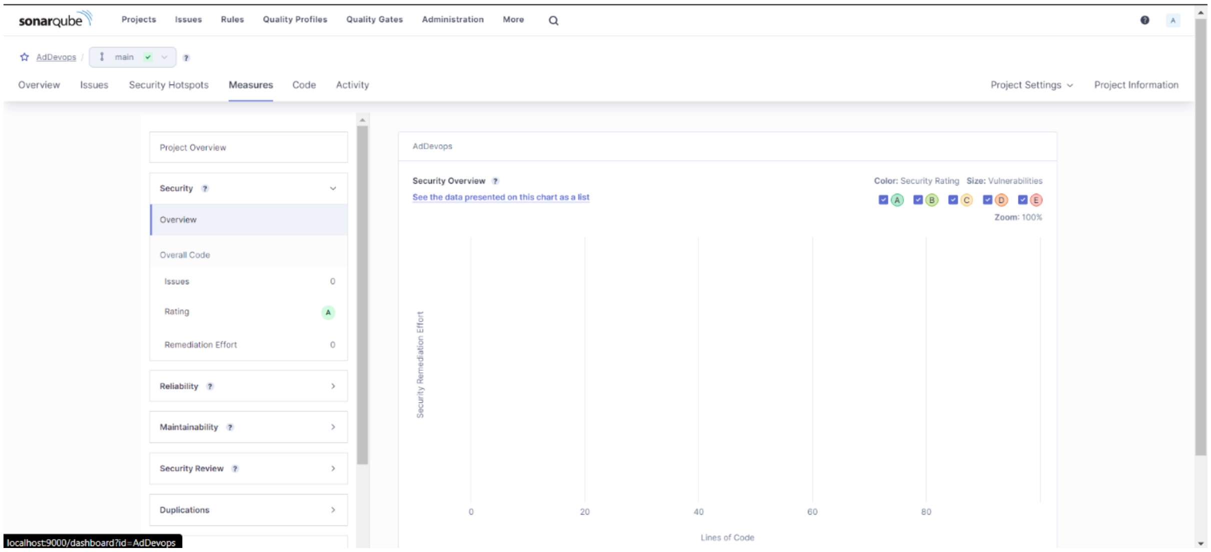
Overall Code

| | |
|---|---|
| Issues | 0 |
| Debt | 0 |
| Debt Ratio | 0.0% |
| Rating | A |
| Effort to Reach A | 0 |

Color: Maintainability Rating Size: Code Smells

Zoom: 100%

## Security:



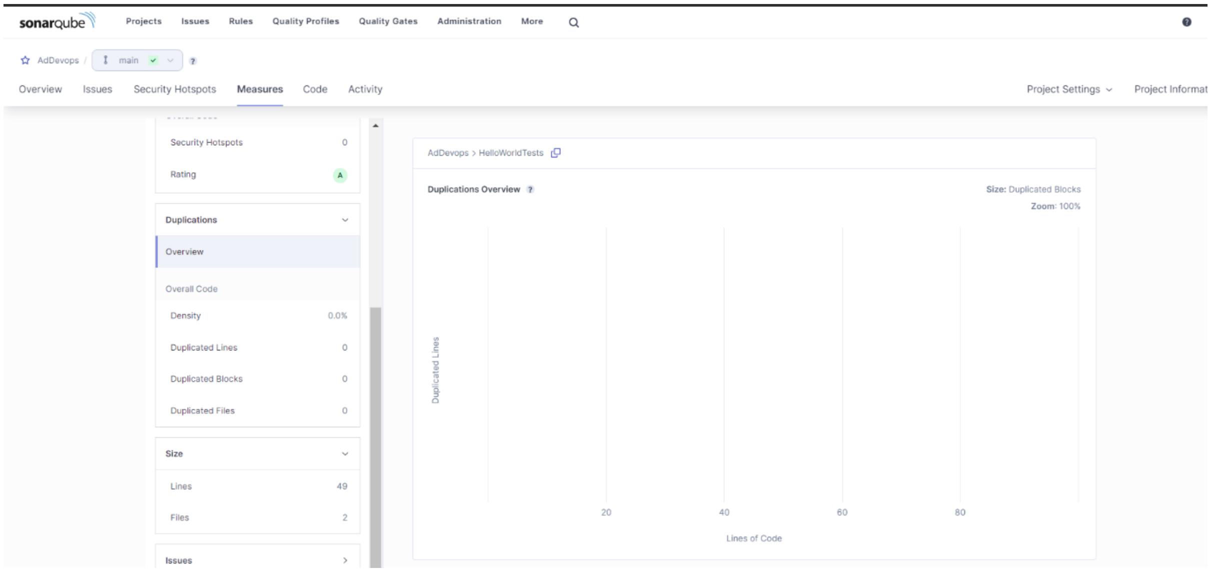
Overall Code

| | |
|---|---|
| Issues | 0 |
| Rating | A |
| Remediation Effort | 0 |

Color: Security Rating Size: Vulnerabilities

Zoom: 100%

Duplications:



Security Hotspots 0
Rating A

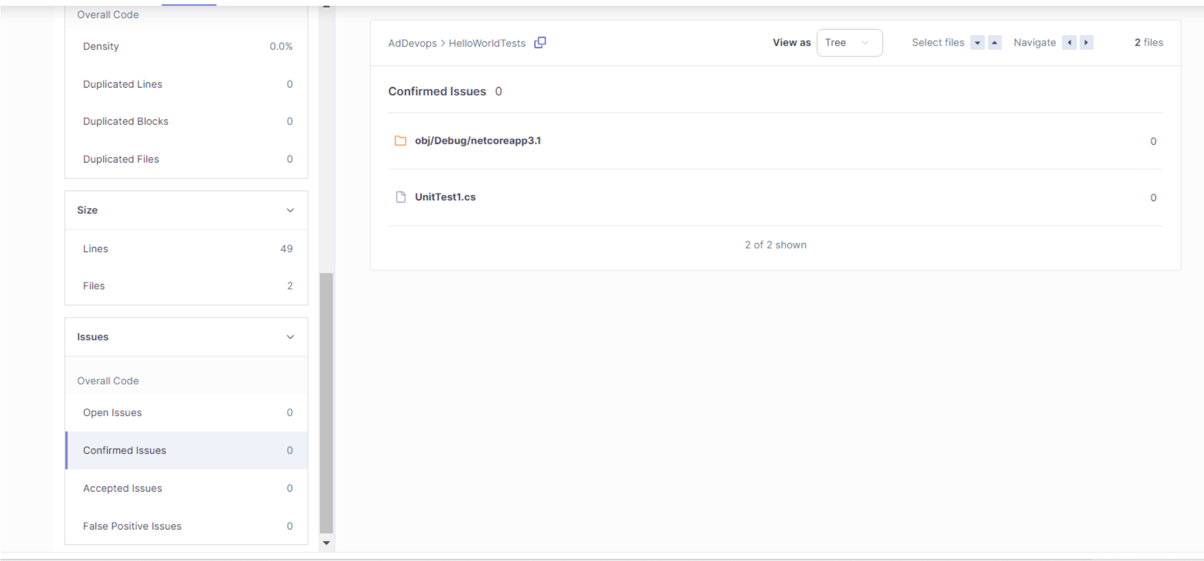
Overall Code
Density 0.0%
Duplicated Lines 0
Duplicated Blocks 0
Duplicated Files 0

Lines 49
Files 2

AdDevops > HelloWorldTests

Duplications Overview
Size: Duplicated Blocks
Zoom: 100%

Issues:



Overall Code
Density 0.0%
Duplicated Lines 0
Duplicated Blocks 0
Duplicated Files 0

Lines 49
Files 2

Overall Code
Open Issues 0
Confirmed Issues 0
Accepted Issues 0
False Positive Issues 0

AdDevops > HelloWorldTests
2 files

Confirmed Issues 0

obj/Debug/netcoreapp3.1 0
UnitTest1.cs 0

2 of 2 shown

Conclusion: Thus, we have successfully integrated Jenkins with SonarQube. This integration enhances code quality by providing immediate feedback to developers, ensuring that code adheres to predefined quality standards.