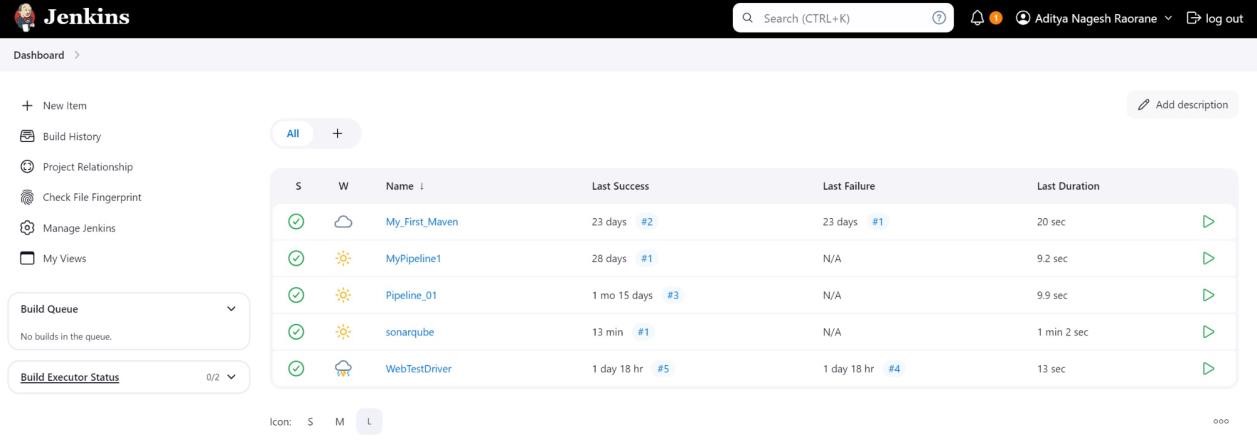
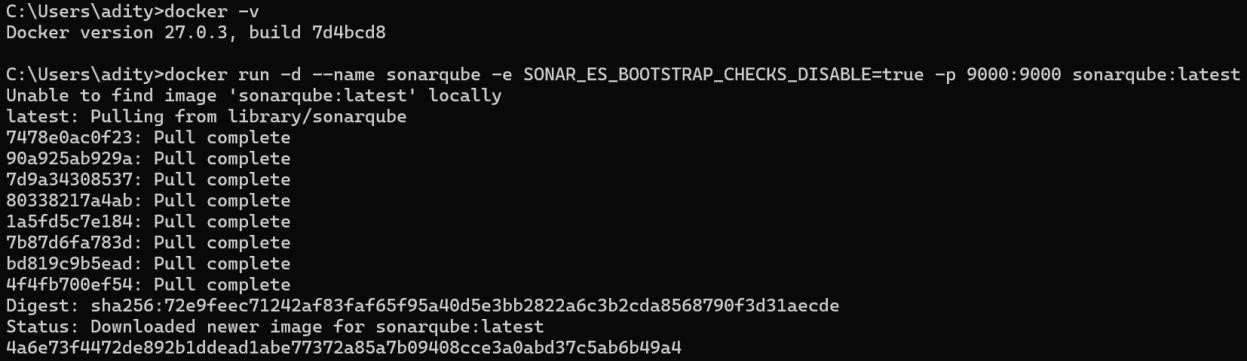
**Aim:** Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

1. Open up Jenkins Dashboard on localhost:8080.

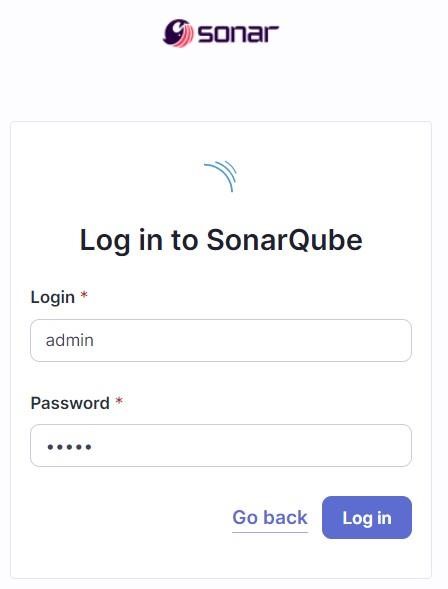


1. Run SonarQube in a Docker container using this command: a] docker -v b] docker pull sonarqube c] docker run -d --name sonarqube -e

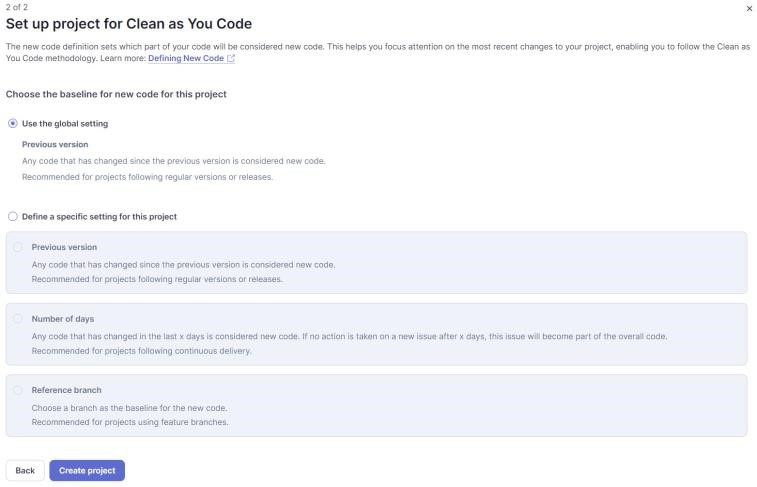
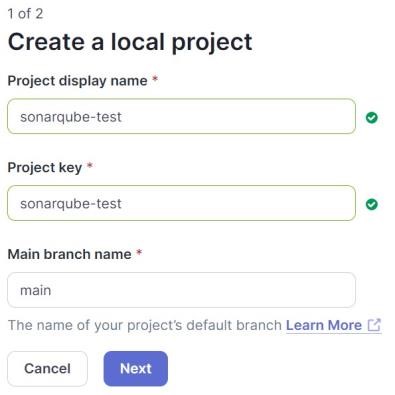
SONAR\_ES\_BOOTSTRAP\_CHECKS\_DISABLE=true -p 9000:9000 sonarqube:latest



1. Once the container is up and running, you can check the status of SonarQube at **localhost port 9000**. The login id is “**admin**” and the password is “**aditya**”.

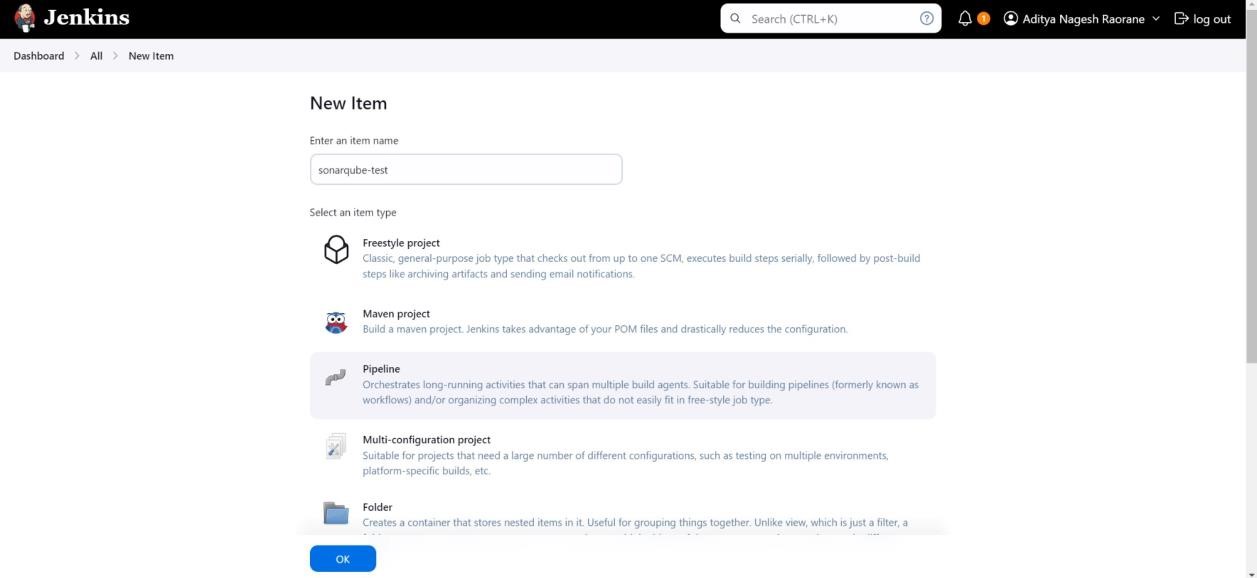


1. **Create a local project in SonarQube** with the name **sonarqube-test**.



Setup the project and come back to Jenkins Dashboard.

1. Create a New Item in Jenkins, choose **Pipeline**.



1. Under **Pipeline Script**, enter the following -

node { stage('Cloning the GitHub

Repo')

{

git '[https://github.com/shazforiot/GOL.git'](https://github.com/shazforiot/GOL.git)

}

stage('SonarQube analysis') { withSonarQubeEnv('sonarqube') { bat

"C:\\Users\\adity\\Downloads\\sonar-scanner-cli-6.1.0.4477-windows-x64\\sonar-s canner-6.1.0.4477-windows-x64\\bin\\sonar-scanner.bat \

-D sonar.login=<YOUR ID> \

-D sonar.password=<YOUR PASSWORD> \

-D sonar.projectKey=<YOUR PROJECT KEY> \

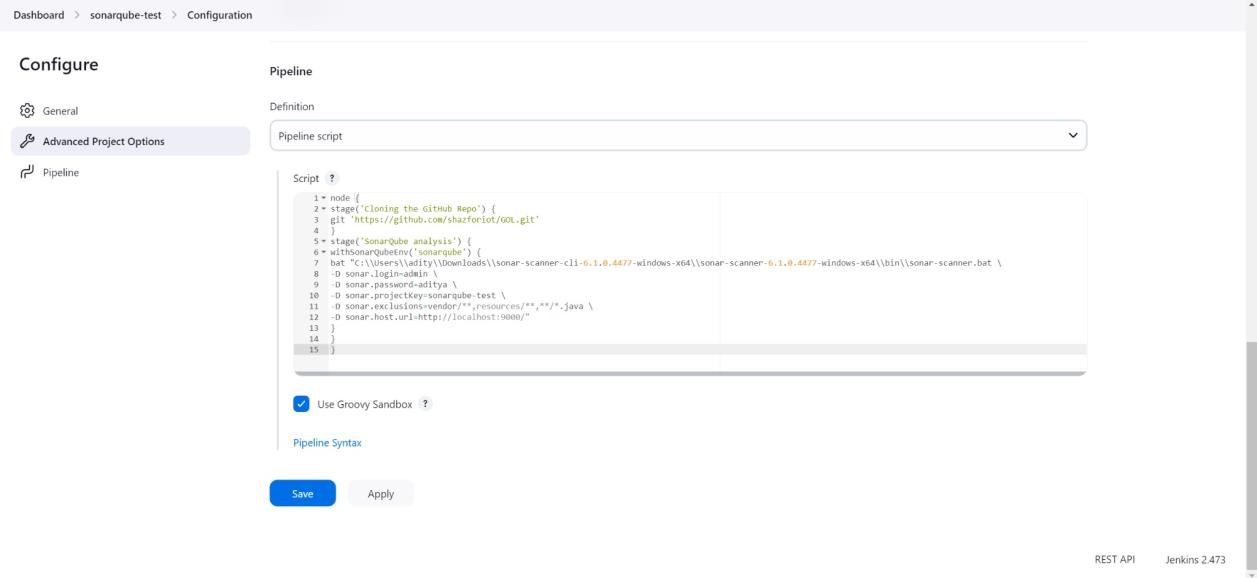
-D sonar.exclusions=vendor/\*\*,resources/\*\*,\*\*/\*.java \

-D sonar.host.url=[http://localhost:9000/"](http://localhost:9000/)

}

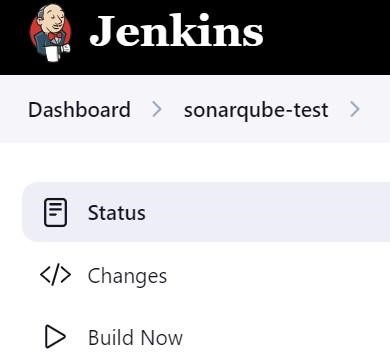
}

}

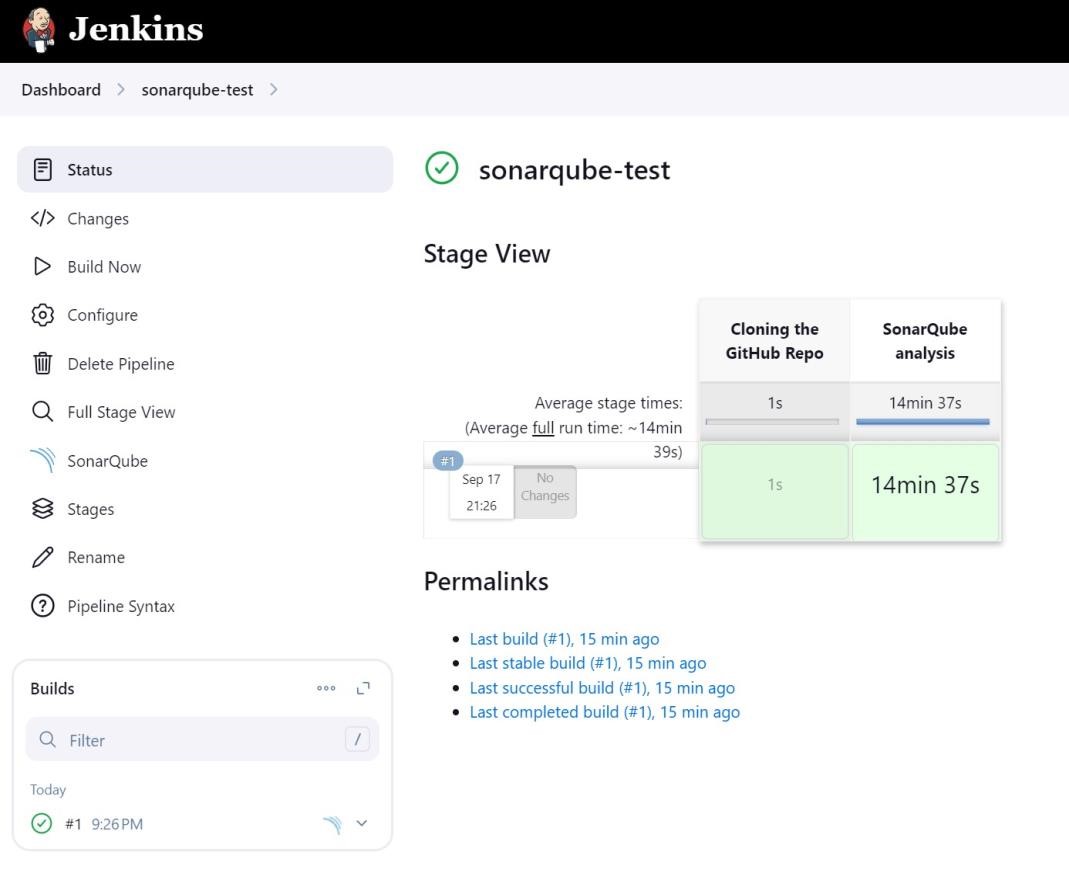


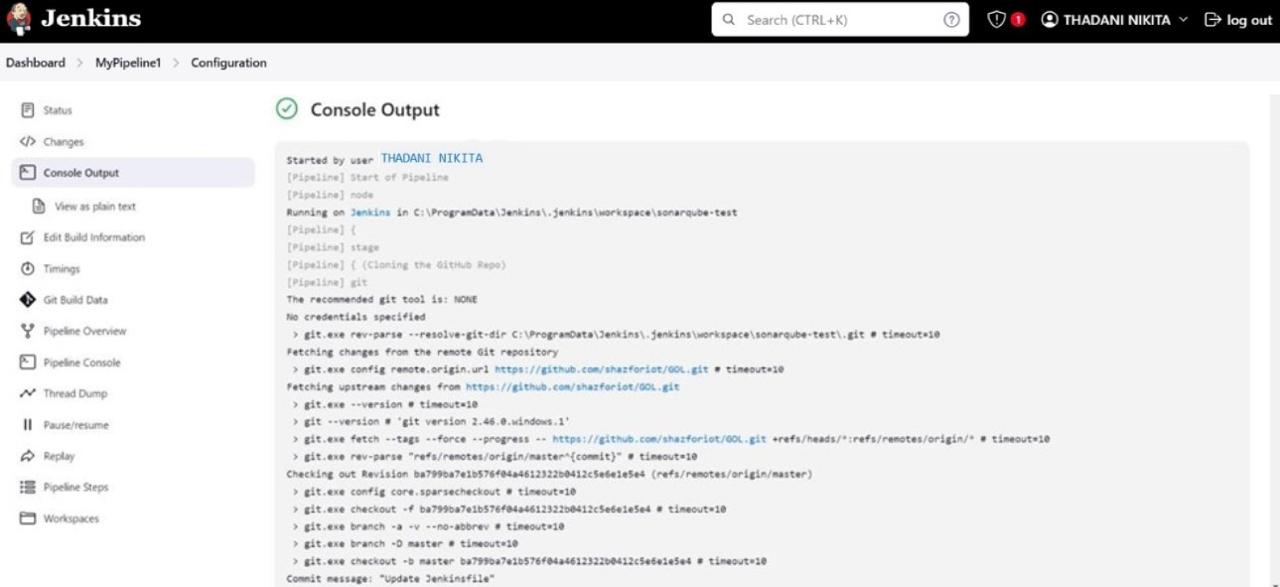
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

1. Run The Build.



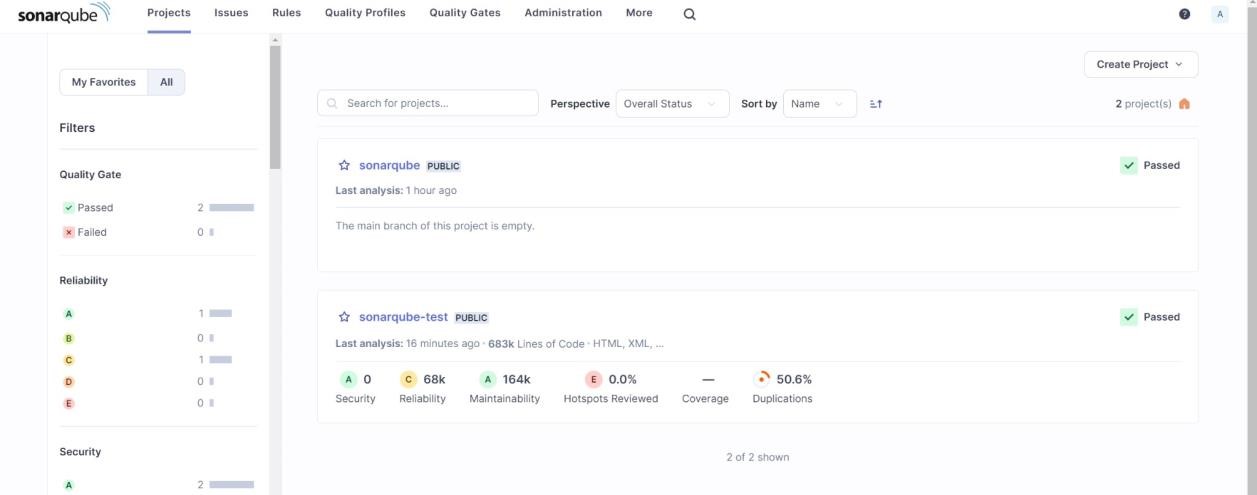
1. Check the console output once the build is complete.

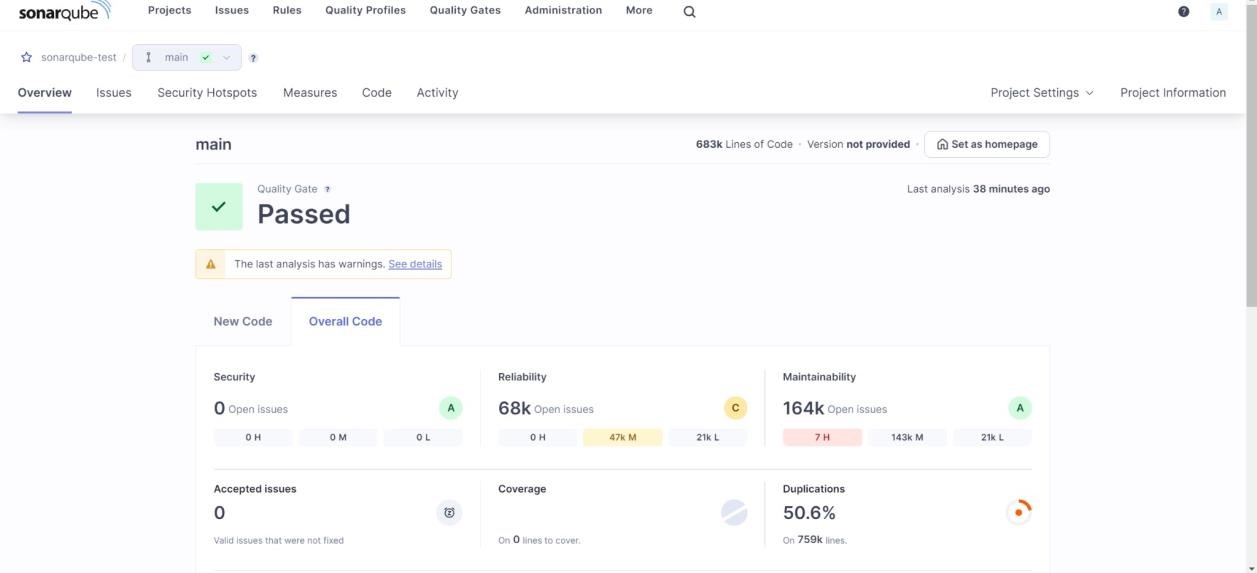






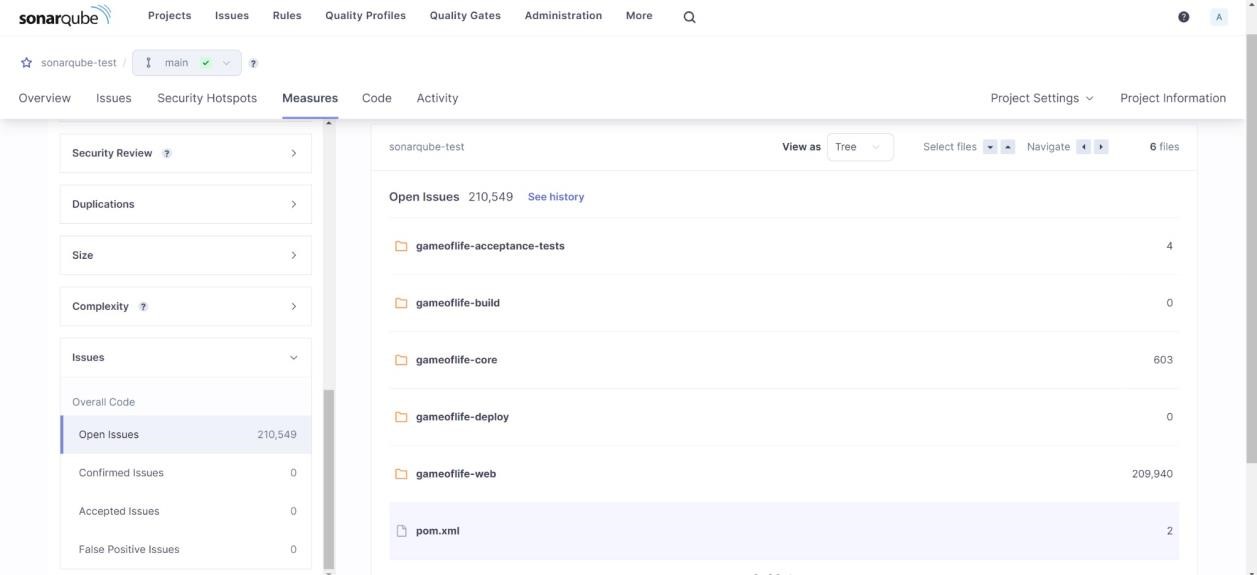
1. After that, check the project in SonarQube.



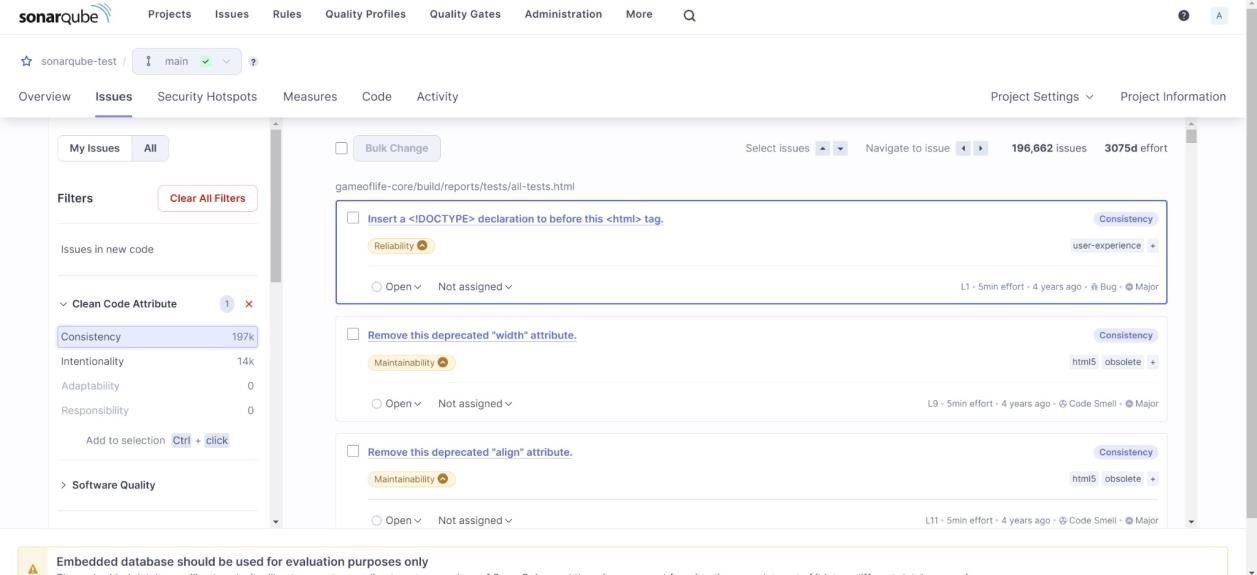


Under different tabs, check all different issues with the code.

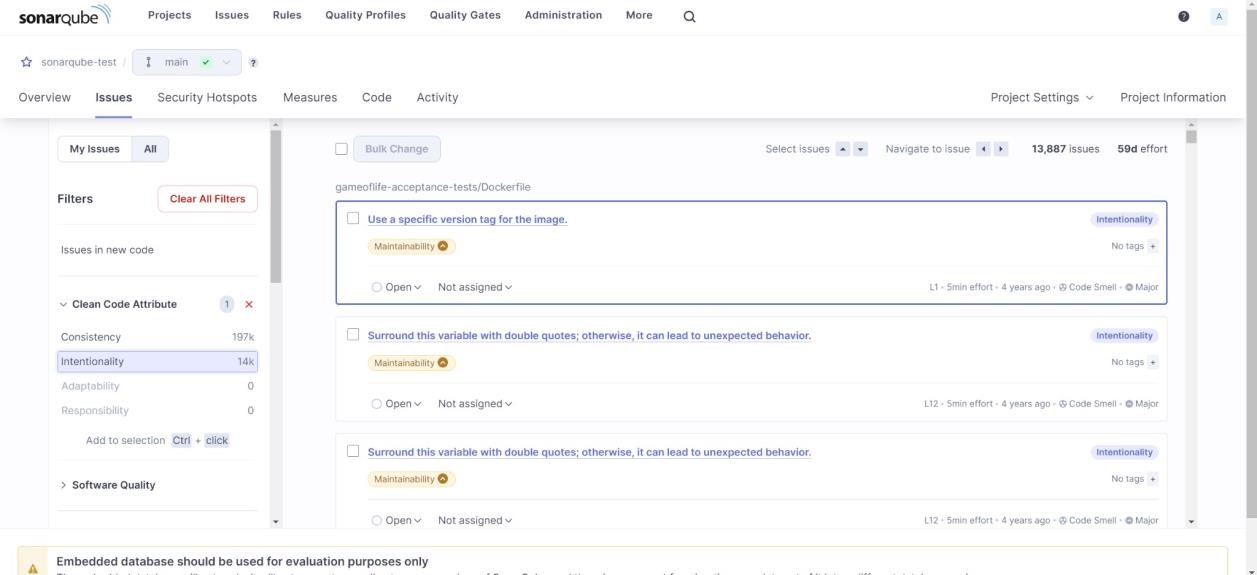
# Code Problems Open Issues



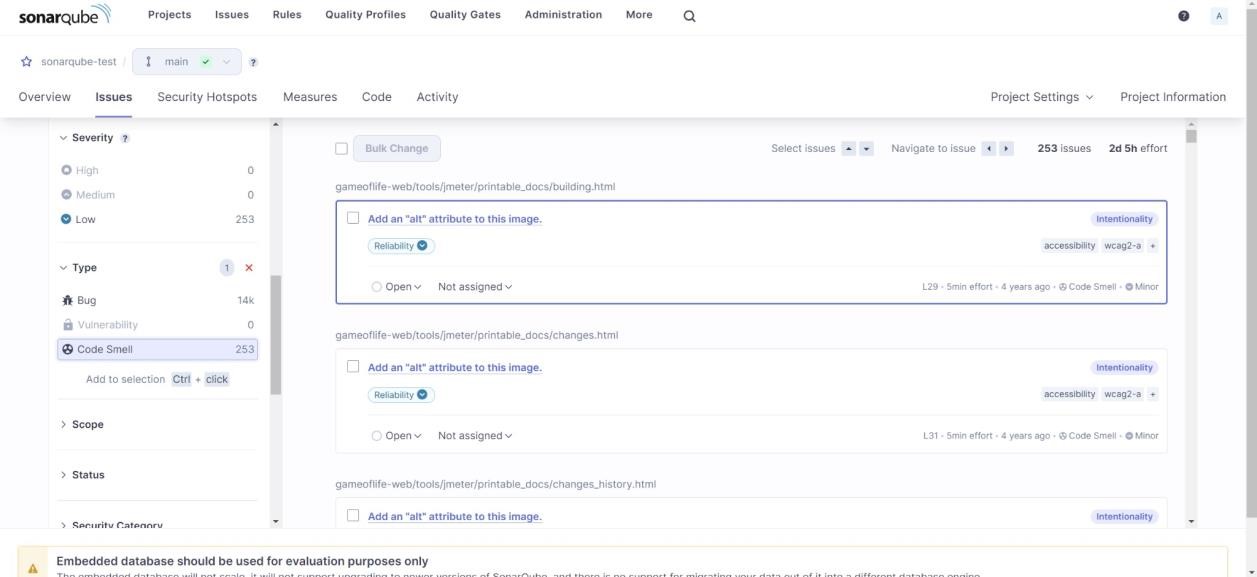
# Consistency



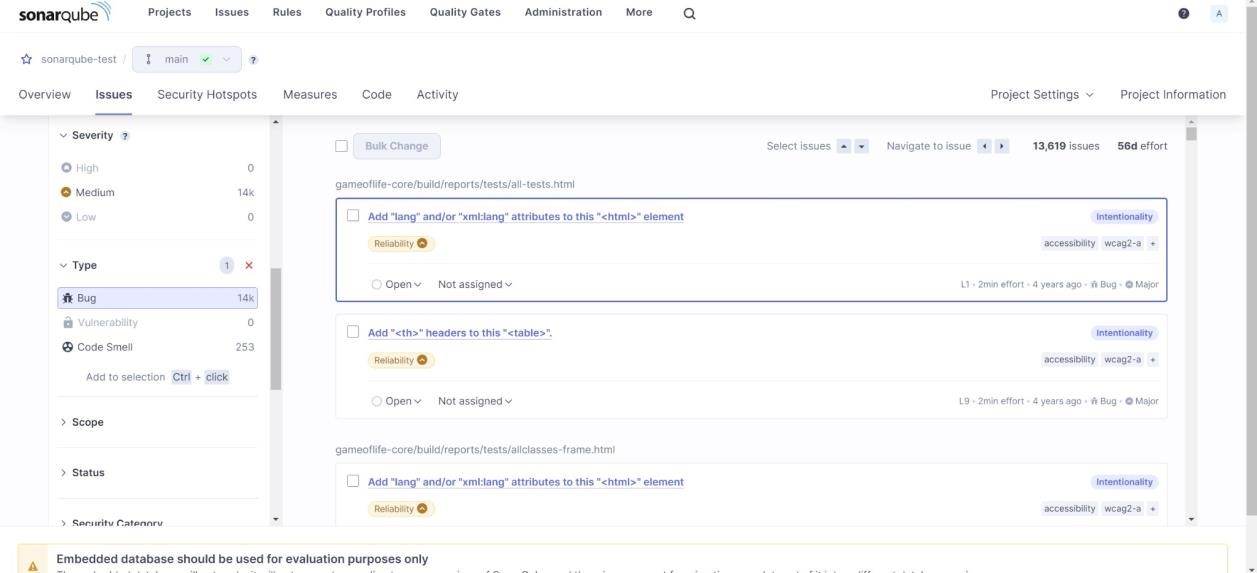
# Intentionality



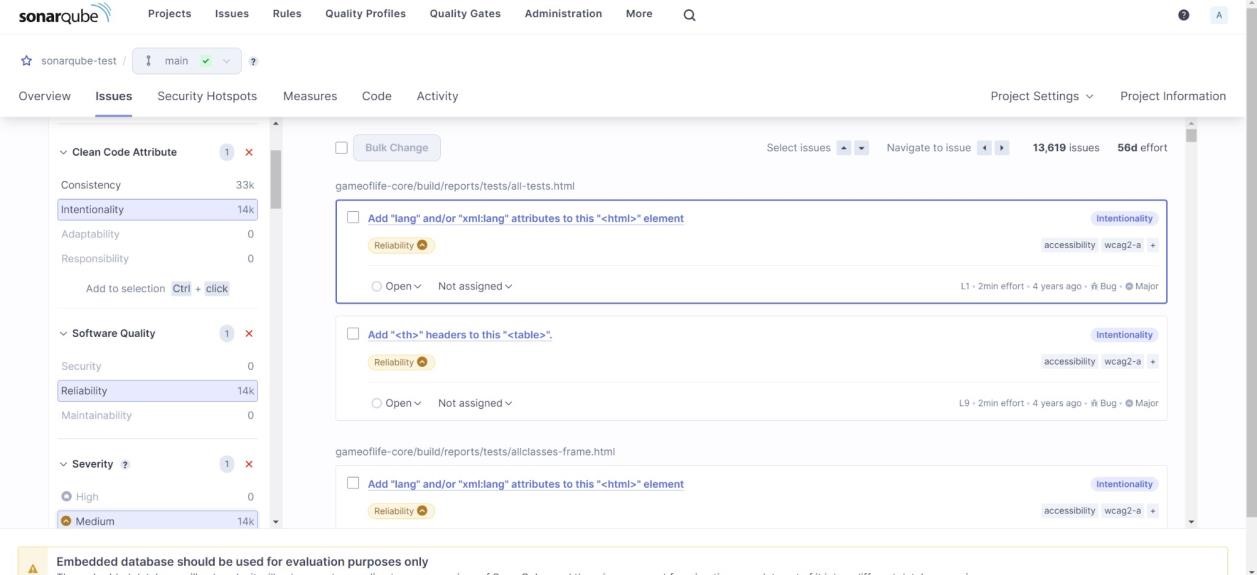
# Code Smells



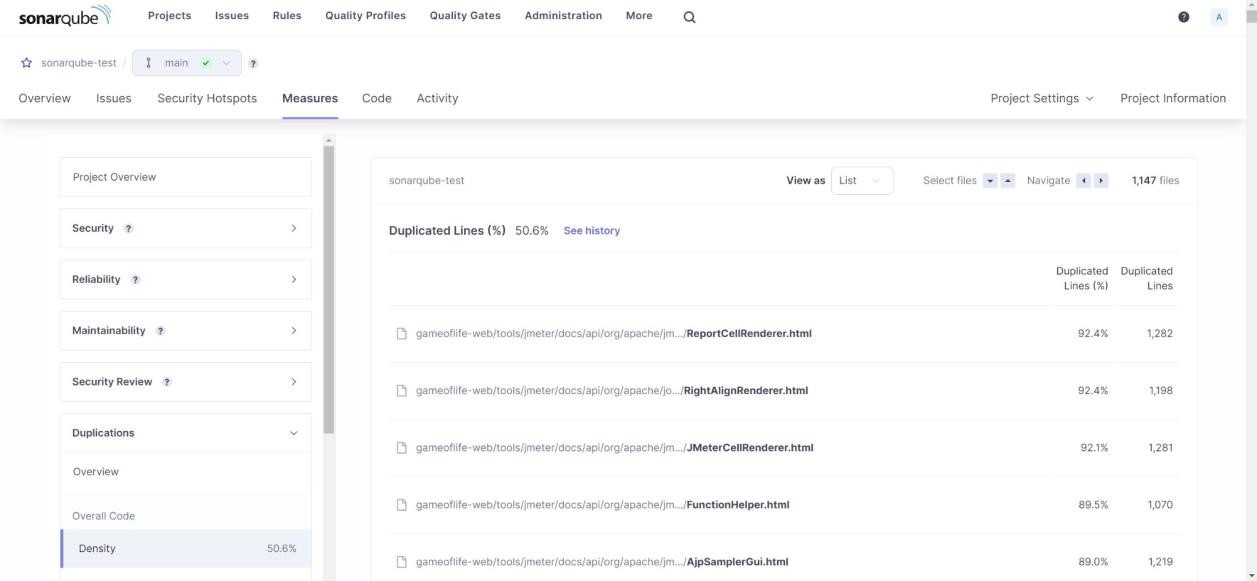
# Bugs



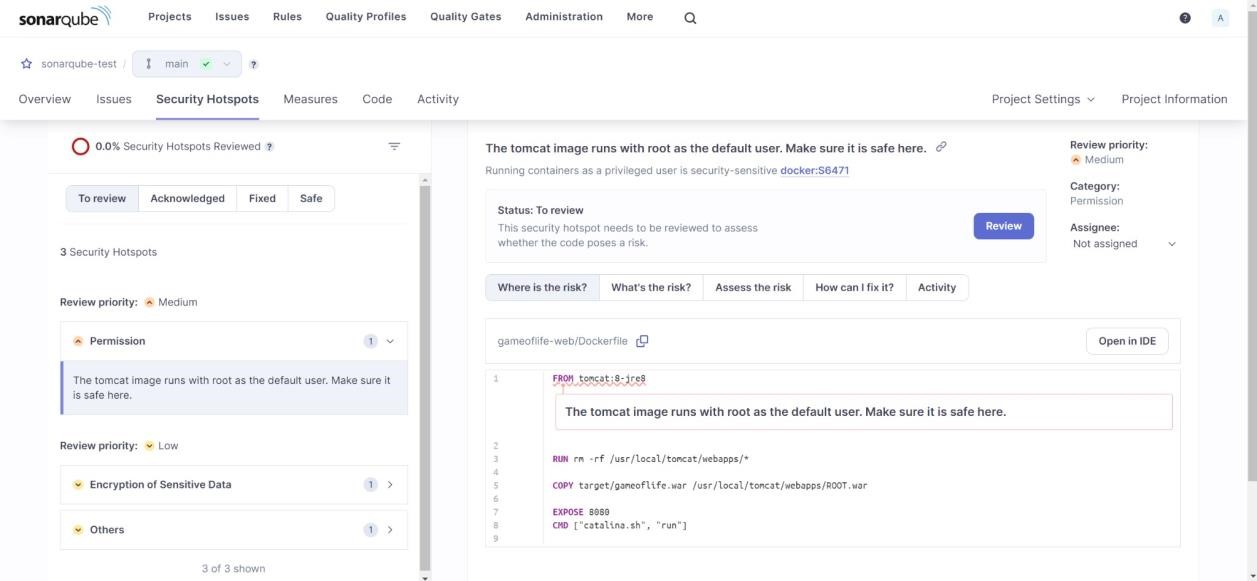
# Reliability



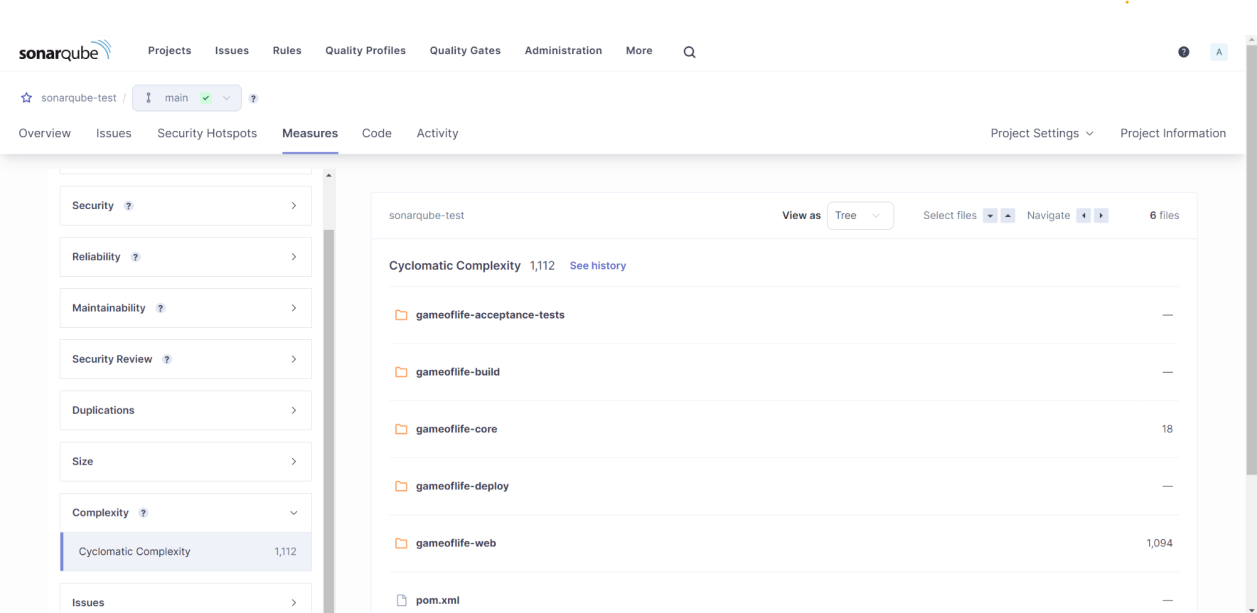
# Duplicates



# Security Hotspot



# Cyclomatic Complexity



In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

**Conclusion:**

In this experiment, we performed a static analysis of the code to detect bugs, code smells, and security vulnerabilities on our sample Java application.