Name: Roma Shirodkar

Div: D15B

Roll No: 58

**Adv DevOps Exp 08**

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:

**SonarQube** is an open-source platform designed for continuous inspection of code quality and security. It provides developers and teams with tools to analyze and improve their codebases by identifying bugs, code smells, vulnerabilities, and other issues that can affect software quality. Here are some key features and components of SonarQube:

### **Key Features of SonarQube:**

1. **Code Quality Analysis**:
   * SonarQube evaluates the quality of code by examining it against various metrics, such as complexity, duplication, coverage, and adherence to coding standards.
2. **Static Code Analysis**:
   * It analyzes code without executing it, allowing for early detection of potential issues in the development process.
3. **Multiple Language Support**:
   * SonarQube supports a wide variety of programming languages, including Java, C#, JavaScript, Python, PHP, and many more, through plugins.
4. **Customizable Quality Profiles**:
   * Users can customize quality profiles to define their own coding rules and standards for specific projects or teams.
5. **Technical Debt Tracking**:
   * SonarQube calculates and tracks technical debt, helping teams understand how much effort is needed to improve code quality over time.
6. **Integration with CI/CD Pipelines**:
   * It can be integrated with continuous integration/continuous deployment (CI/CD) tools like Jenkins, GitLab CI, CircleCI, and others, enabling automated analysis during the build process.

**SonarQube** is an open-source platform designed for continuous inspection of code quality and security. It provides developers and teams with tools to analyze and improve their codebases by identifying bugs, code smells, vulnerabilities, and other issues that can affect software quality. Here are some key features and components of SonarQube:

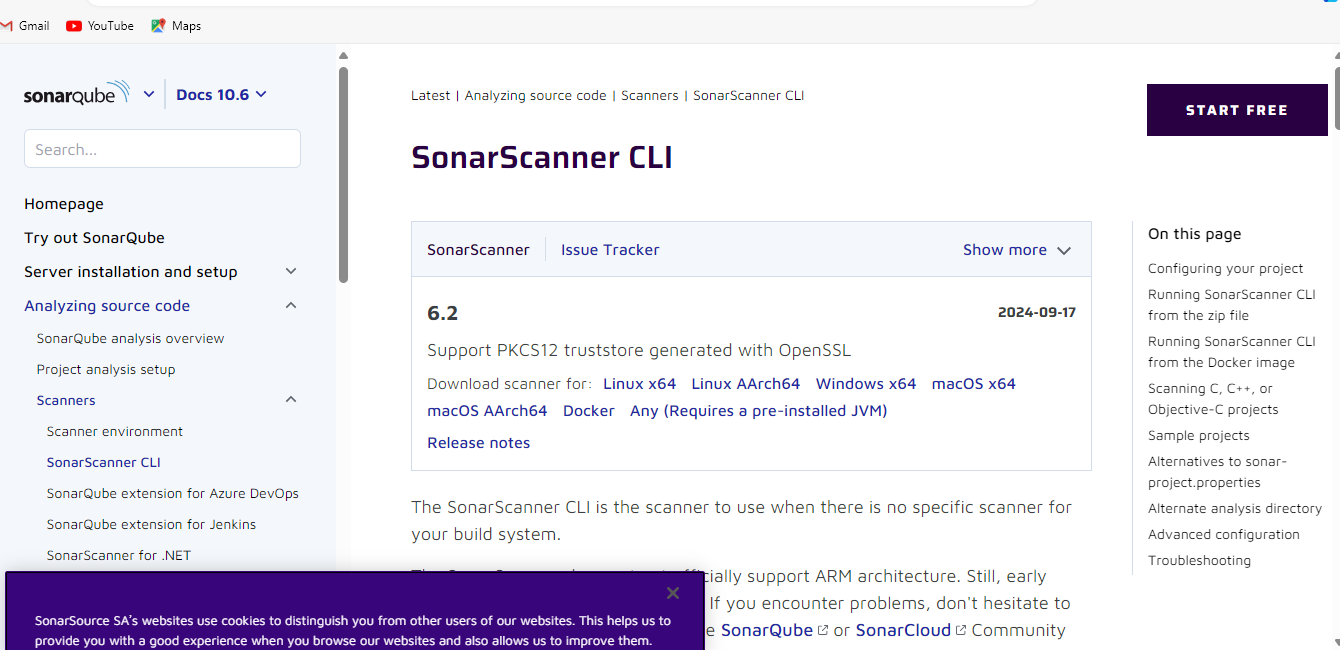
### **Key Features of SonarQube:**

1. **Code Quality Analysis**:
   * SonarQube evaluates the quality of code by examining it against various metrics, such as complexity, duplication, coverage, and adherence to coding standards.
2. **Static Code Analysis**:
   * It analyzes code without executing it, allowing for early detection of potential issues in the development process.
3. **Multiple Language Support**:
   * SonarQube supports a wide variety of programming languages, including Java, C#, JavaScript, Python, PHP, and many more, through plugins.
4. **Customizable Quality Profiles**:
   * Users can customize quality profiles to define their own coding rules and standards for specific projects or teams.
5. **Technical Debt Tracking**:
   * SonarQube calculates and tracks technical debt, helping teams understand how much effort is needed to improve code quality over time.
6. **Integration with CI/CD Pipelines**:
   * It can be integrated with continuous integration/continuous deployment (CI/CD) tools like Jenkins, GitLab CI, CircleCI, and others, enabling automated analysis during the build process.

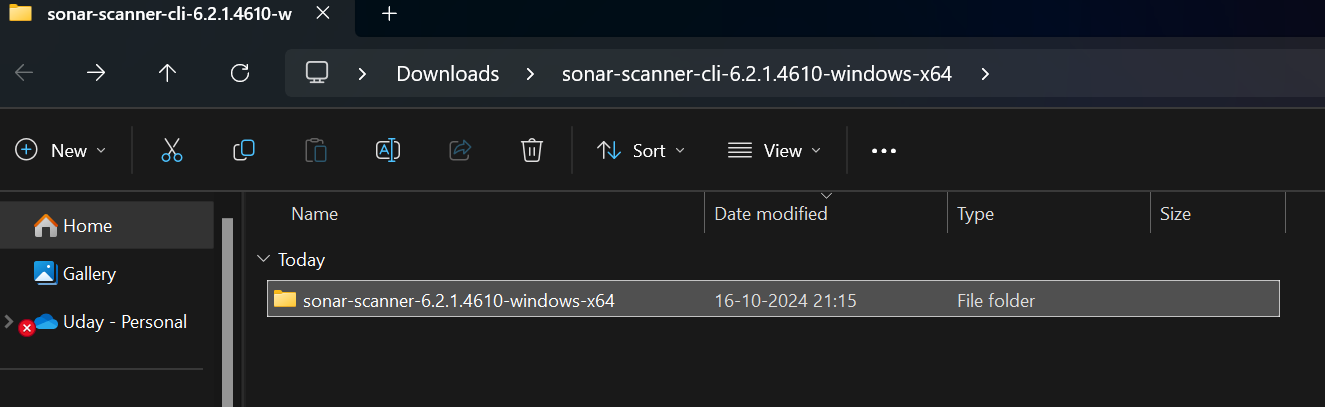
**Step 1:**Download sonar scanner. We will require it in the further steps while building our pipeline

[SonarScanner CLI (sonarsource.com)](https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/)

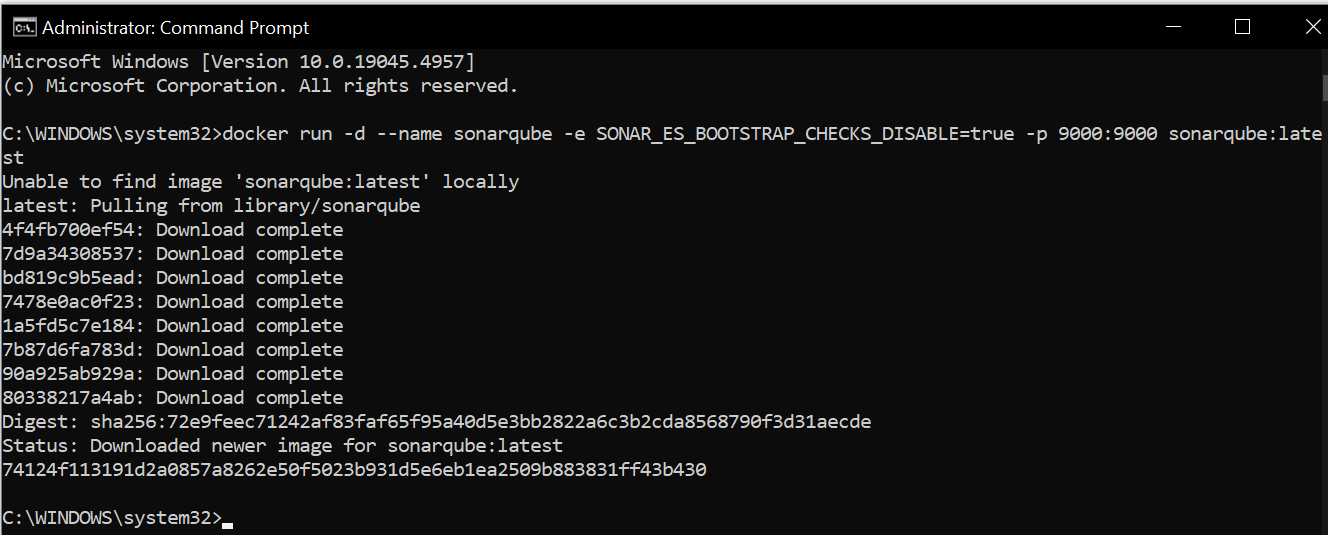
Visit this link and download the sonarqube scanner CLI.

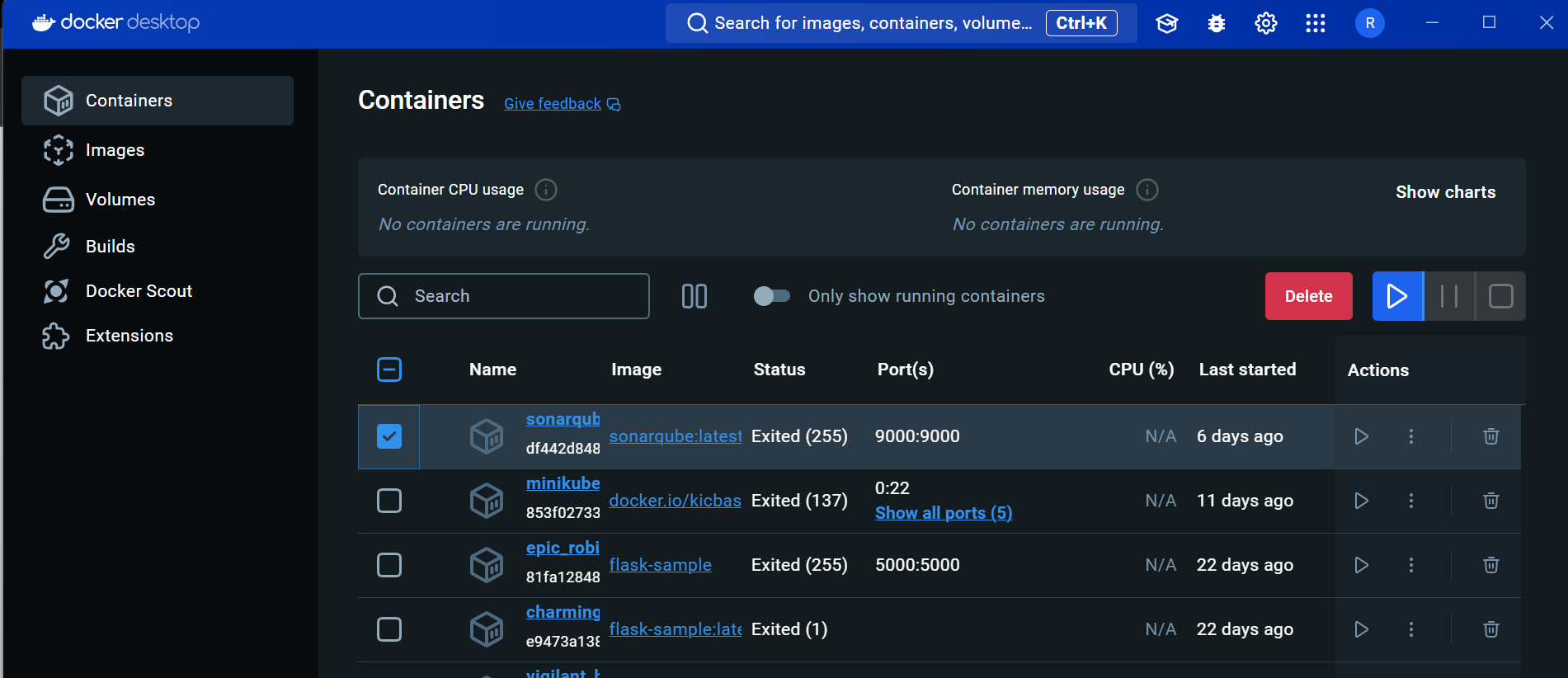
****

Extract the zip files in a folder

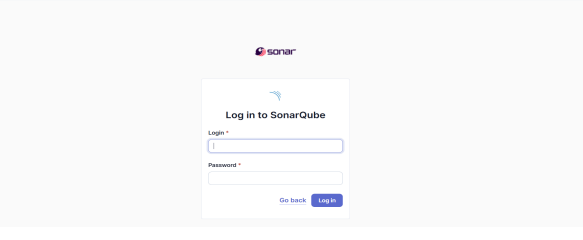


Run this command in an administrative command prompt so as to create a sonarqube image on localhost:9000

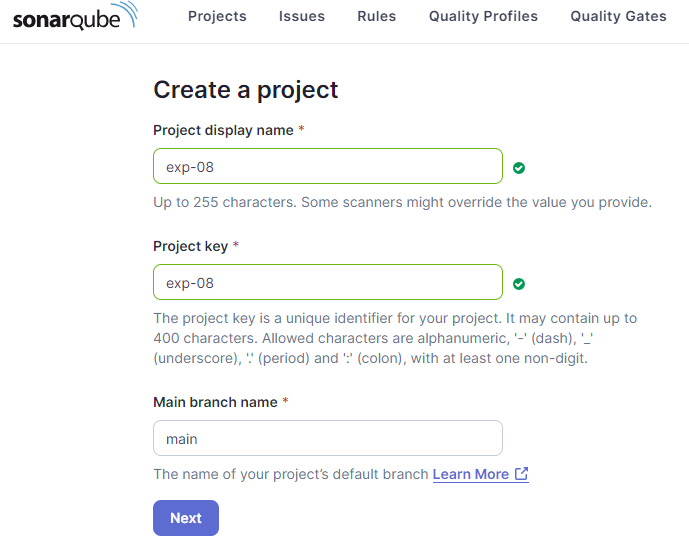




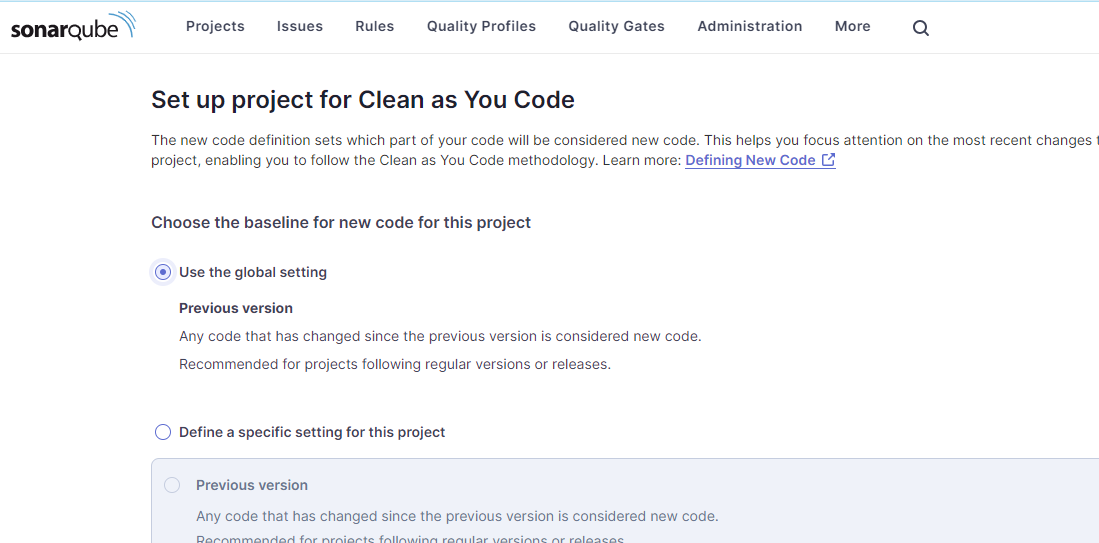
Visit localhost:9000 on your browser and login to your account( the default credentials are admin and admin for username and password. It allows to change it and set our own password for our account)



Next, we are supposed to Create a project. Give some suitable name to your project. A project will be generated for it automatically. We can change it or keep it the same

****

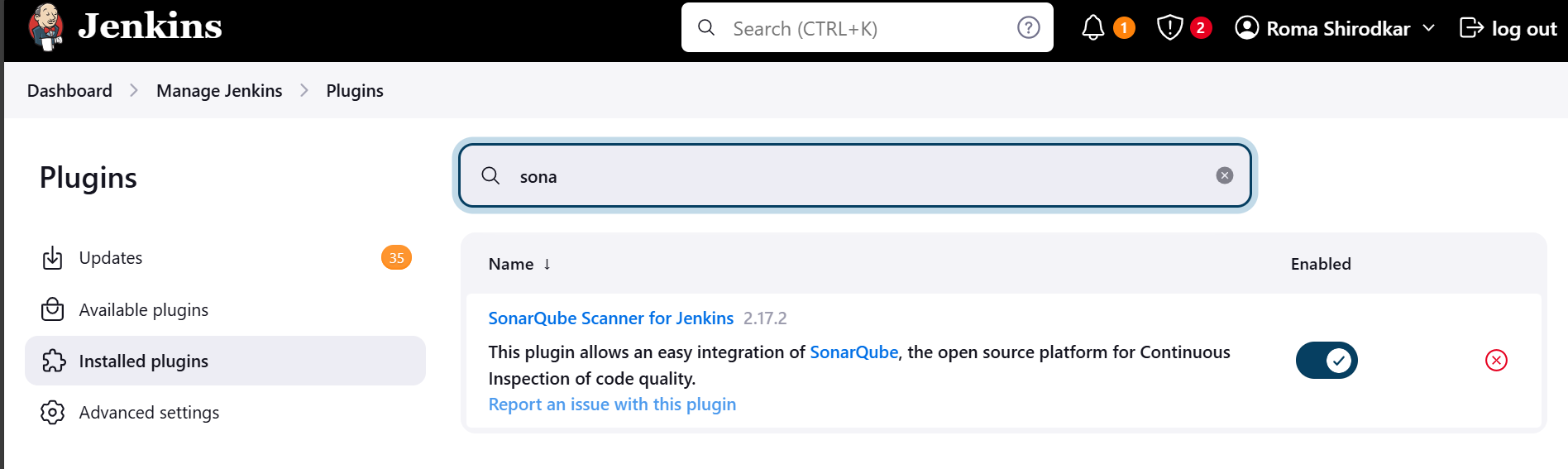
Choose global settings option for setting up your project

****

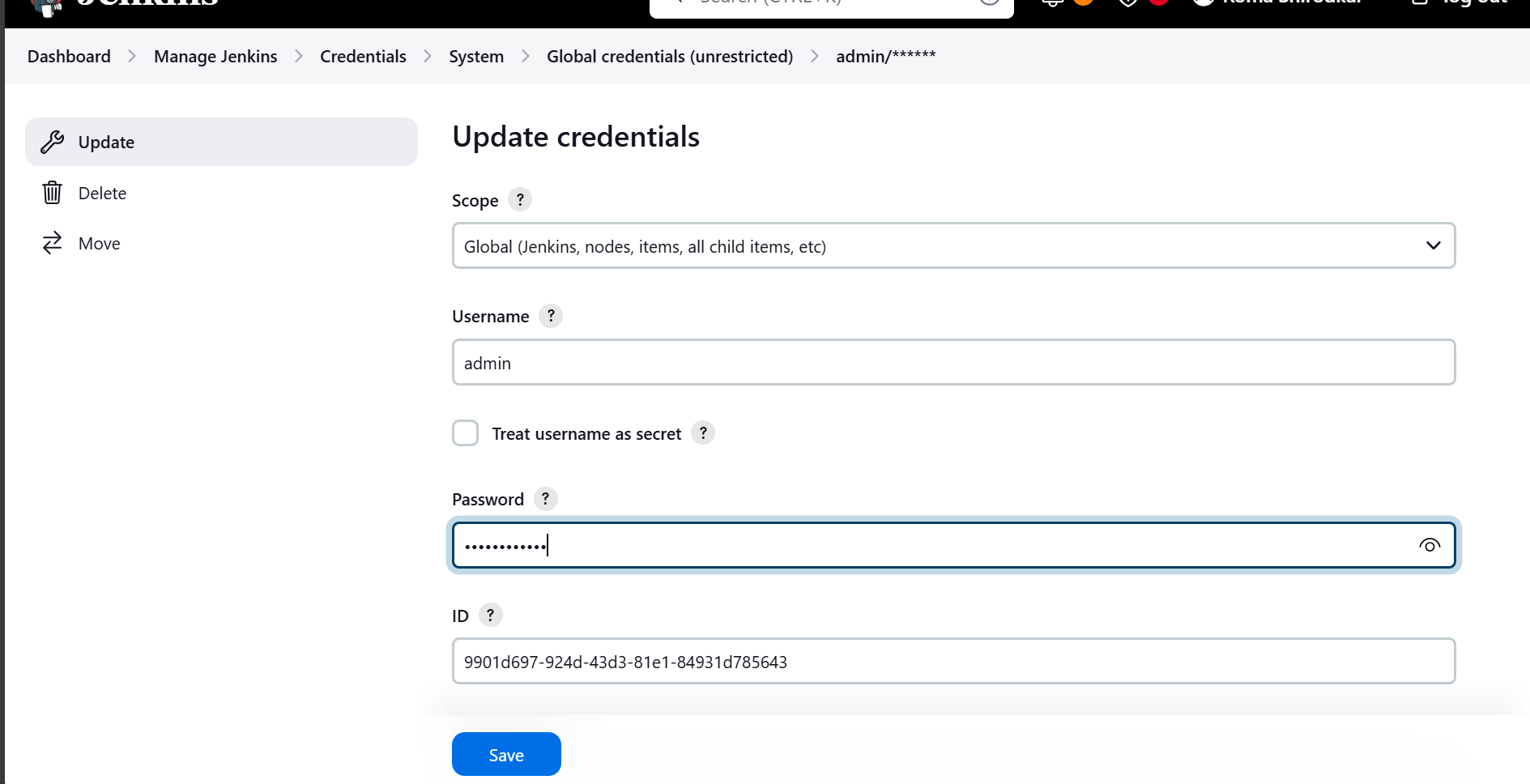
Now, on jenkins ..

For a few contents inside the script to be built using our pipeline, we need to configure those changes inside the global configuration settings inside Manage Jenkins.

First, Go to plugins inside Manage Jenkins and install the Sonarqube Scanner plugin which is necessary for the build to be executed.



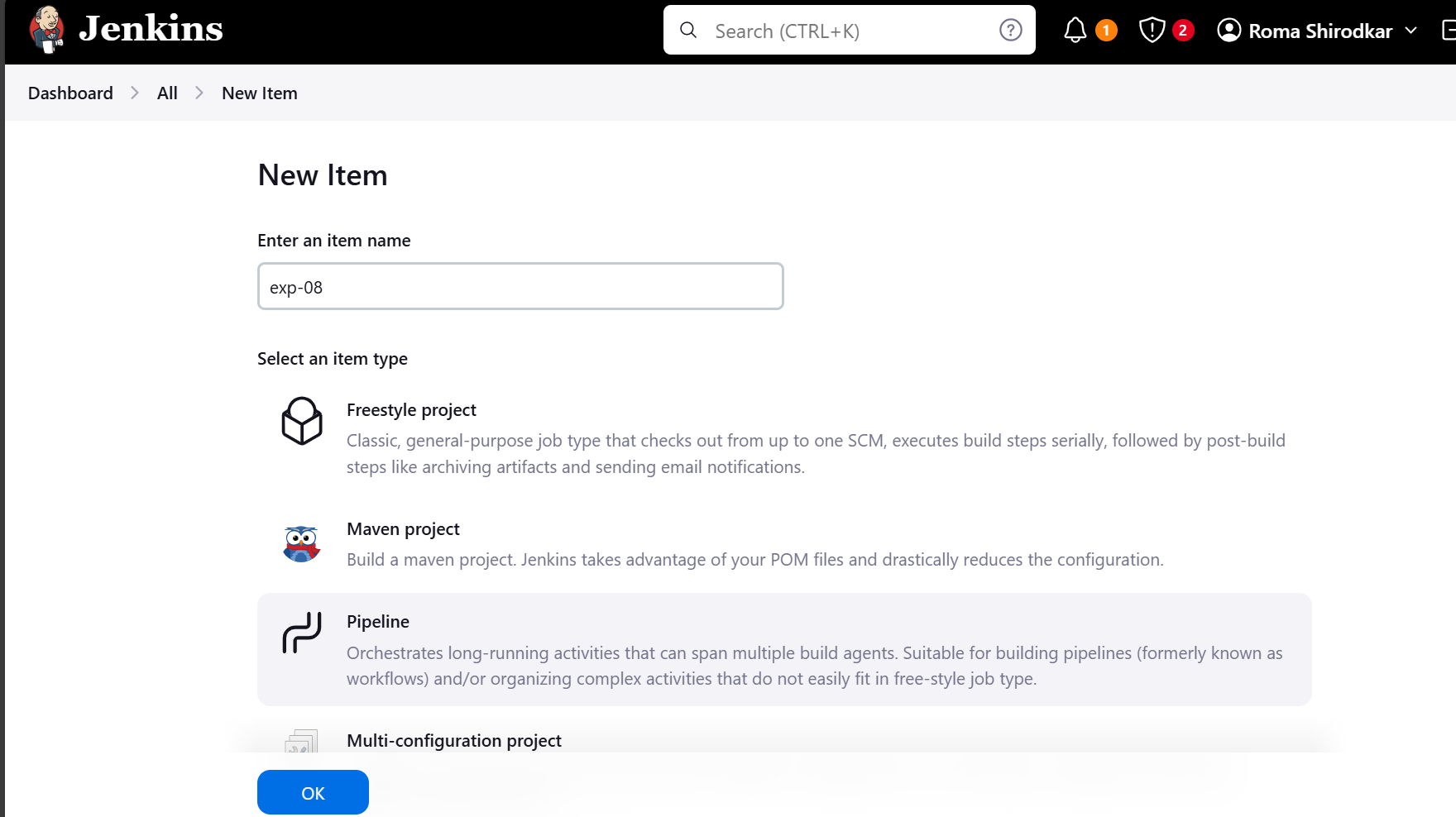
Next, we will have to add our Sonarqube project specific credentials in the Credentials section of manage jenkins



Here, i added the username i.e admin and my password … for which jenkins generated a automated ID for the credential block.

Now, come to the dashboard

Create new item and select pipeline project. Give it a suitable name

****

**Put this script in the Pipeline Definition section and select the mode as Pipeline script**

node {

stage('Cloning the GitHub Repo') {

git 'https://github.com/shazforiot/GOL.git'

}

stage('SonarQube analysis') {

withSonarQubeEnv('sonarqube') {

withCredentials([usernamePassword(credentialsId: '9901d697-924d-43d3-81e1-84931d785643', usernameVariable: 'SONAR\_USER', passwordVariable: 'SONAR\_PASSWORD')]) {

bat """ C:/Users/udays/Downloads/sonar-scanner-cli-6.2.1.4610-windows-x64/sonar-scanner-6.2.1.4610-windows-x64/bin/sonar-scanner.bat ^

-D sonar.login=%SONAR\_USER% ^

-D sonar.password=%SONAR\_PASSWORD% ^

-D sonar.projectKey=exp-08 ^

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

-D sonar.host.url=http://127.0.0.1:9000/

"""

}

}

}

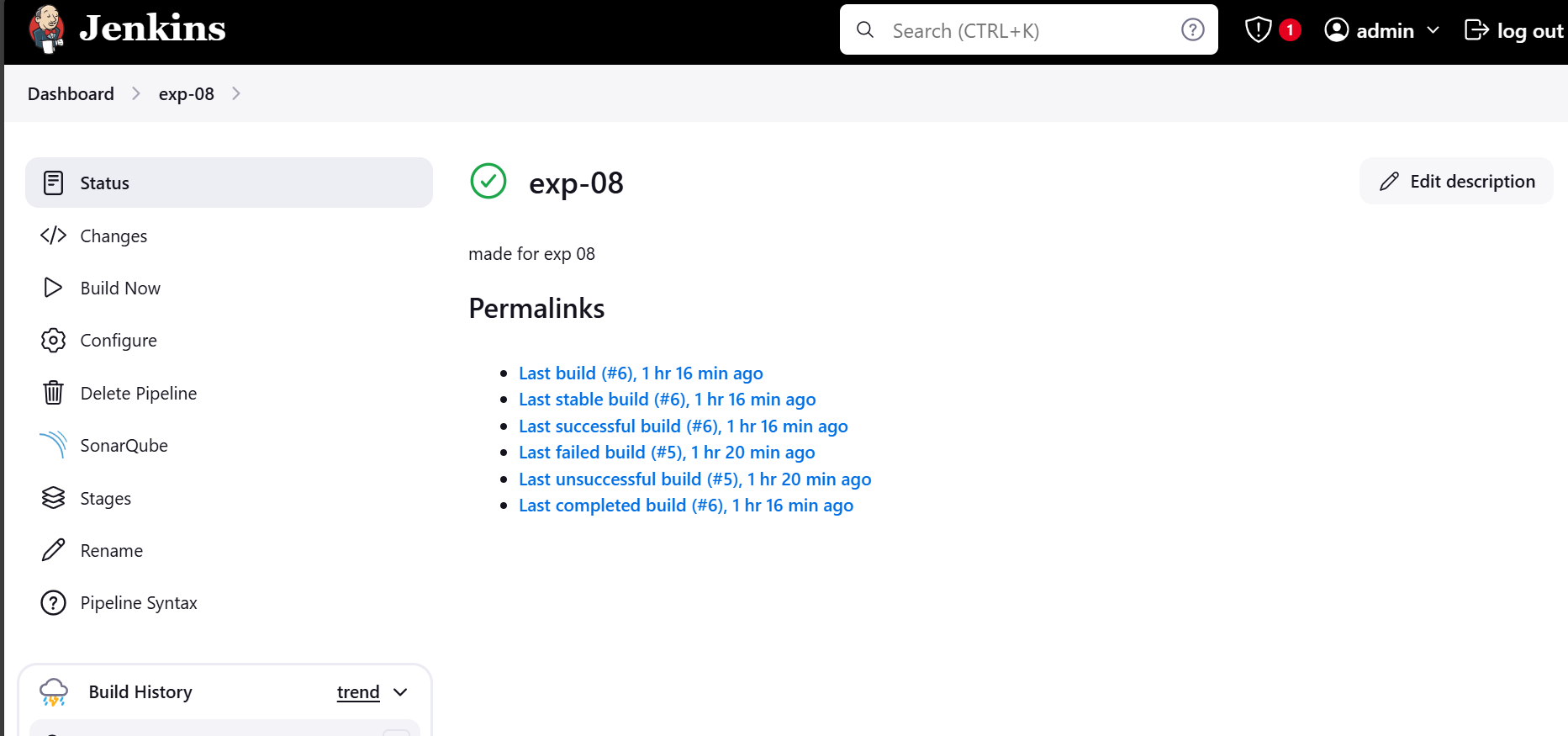
}



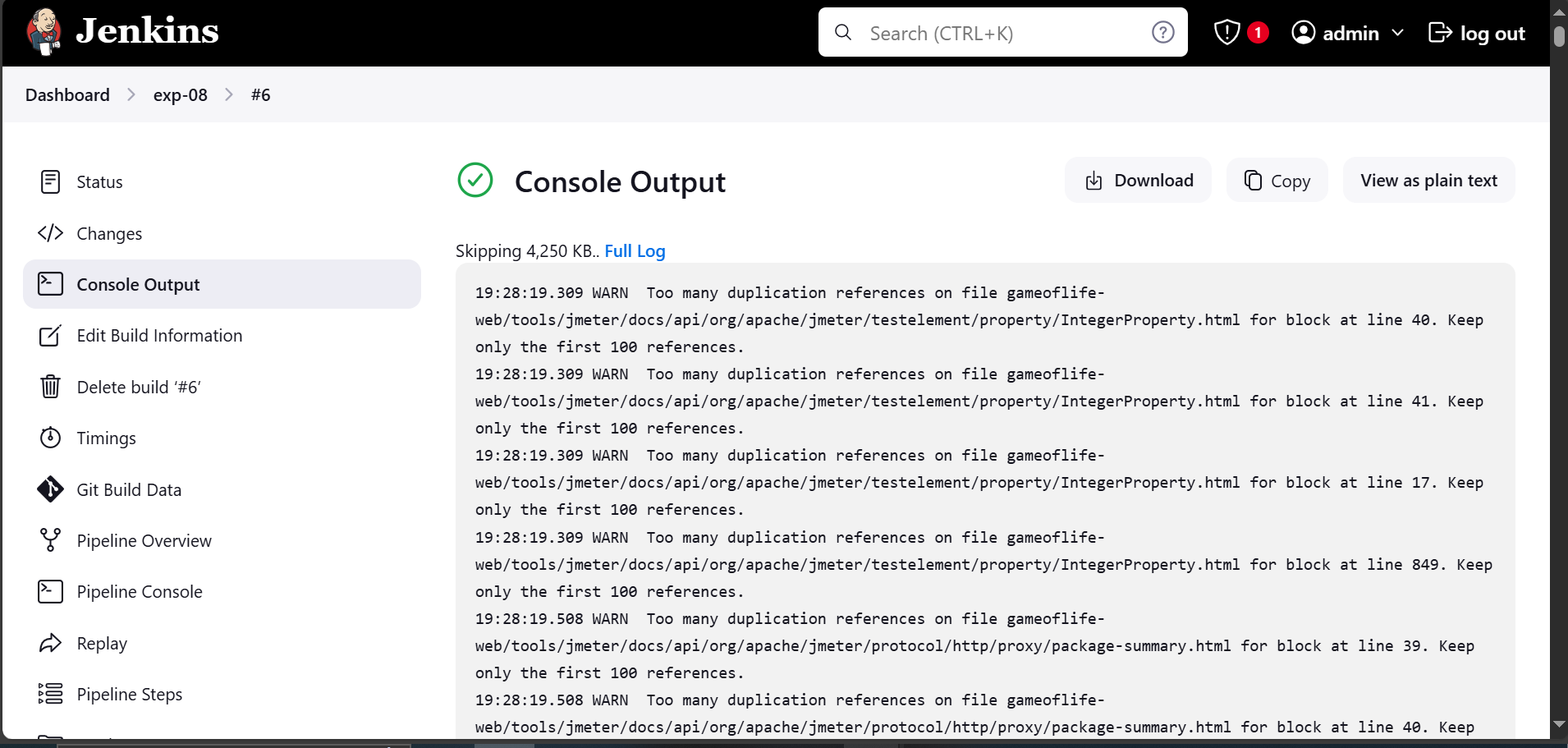
If you would notice carefully, i have used the withCredentials function within the pipeline script which makes the build avail the facility of using the credentials that i had set earlier

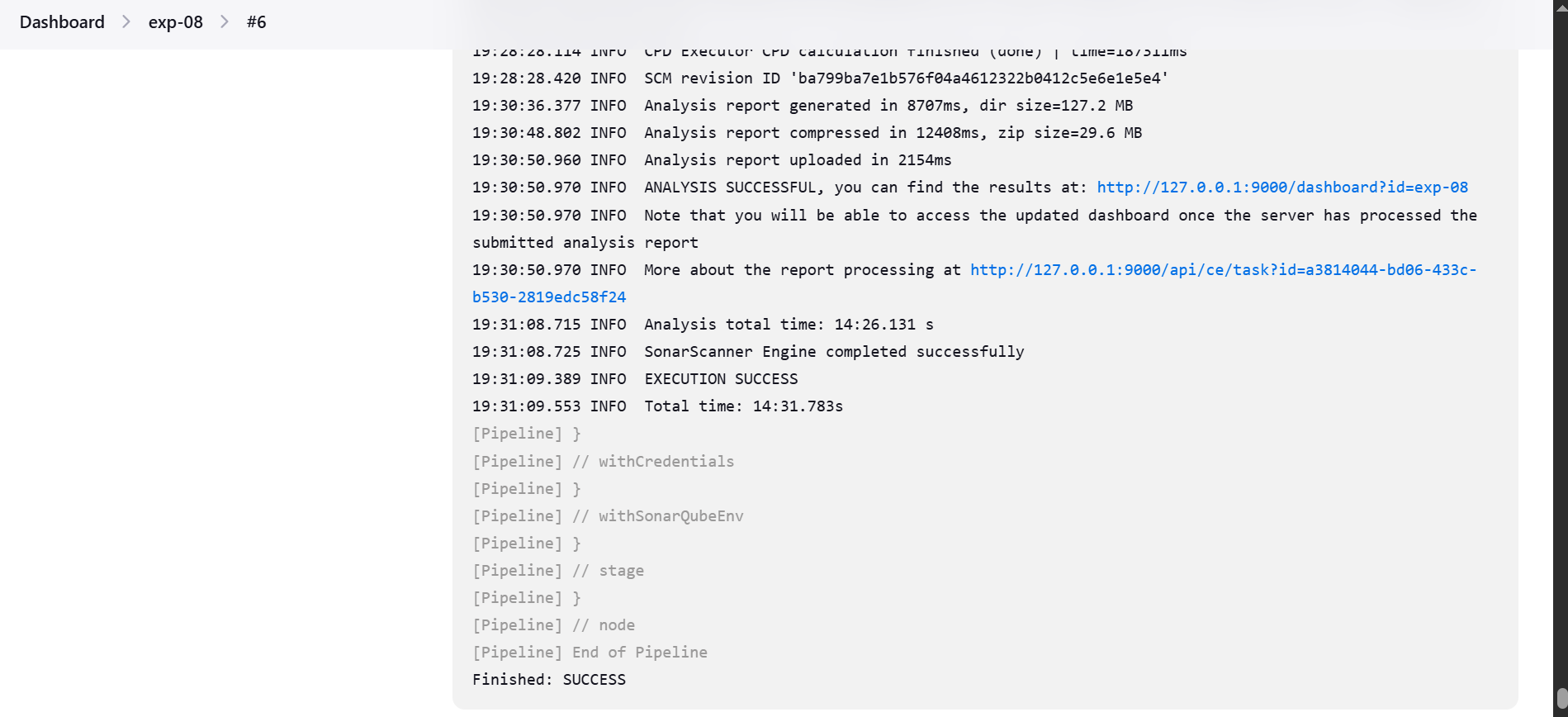
I have also added a line where i have mentioned the path of the sonar-scanner.bat file, since there are problems with system configurations with the installation of sonarqube

Now, save and build the pipeline

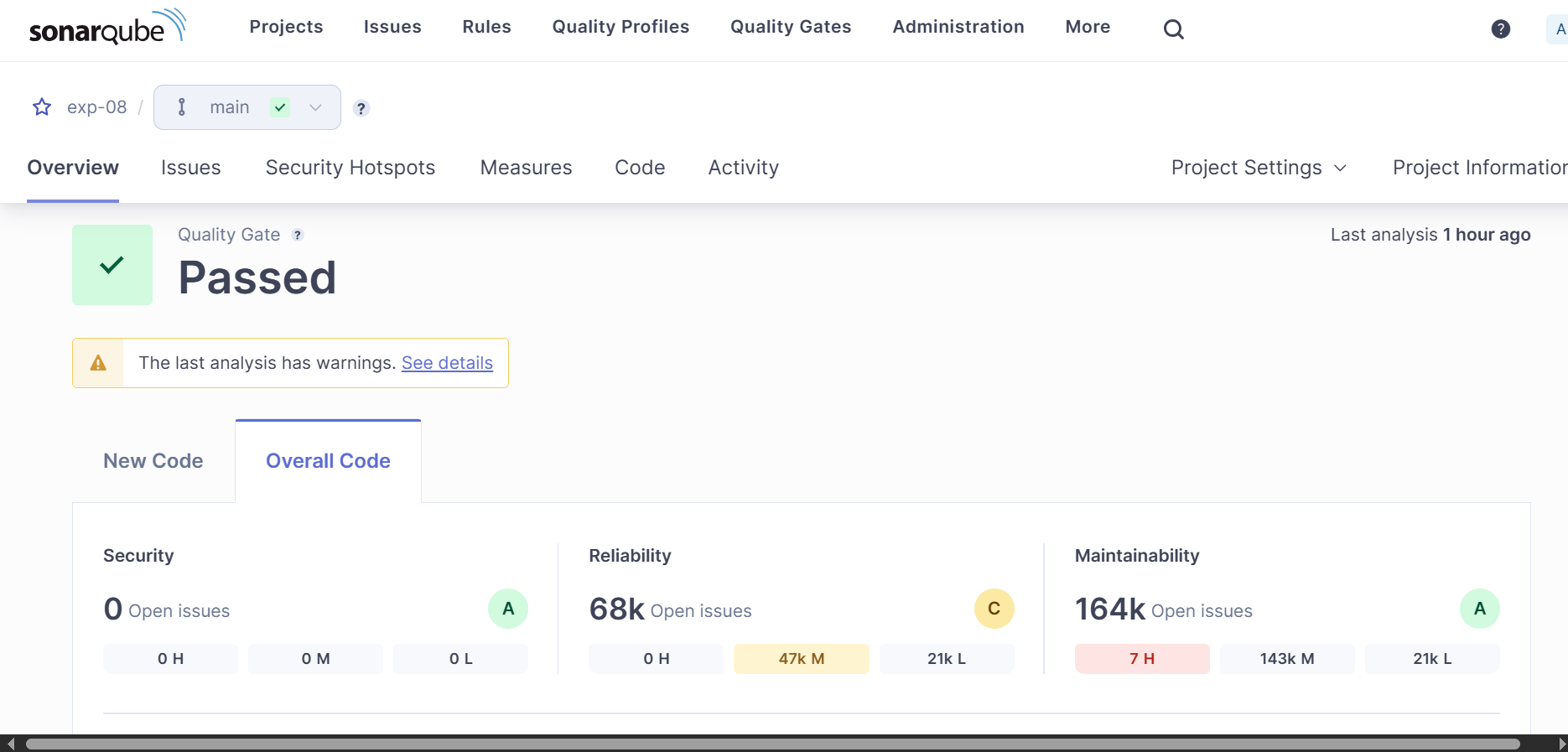


It generally takes 10-15 minutes for building this pipeline.



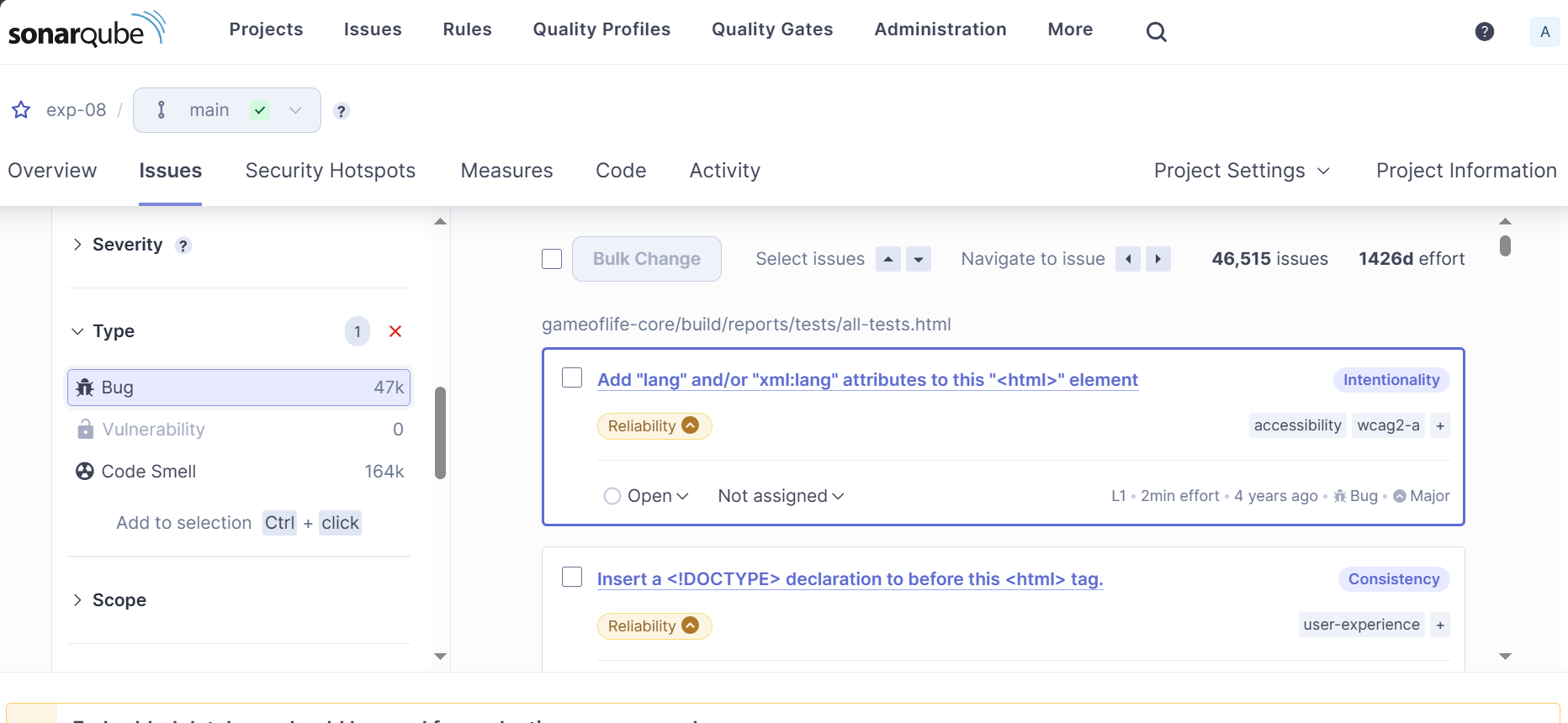


After a successful build, we are supposed to go back to our Sonarqube profile. We will see this type of a message on the screen, where it offers us multiple options to analyze our code

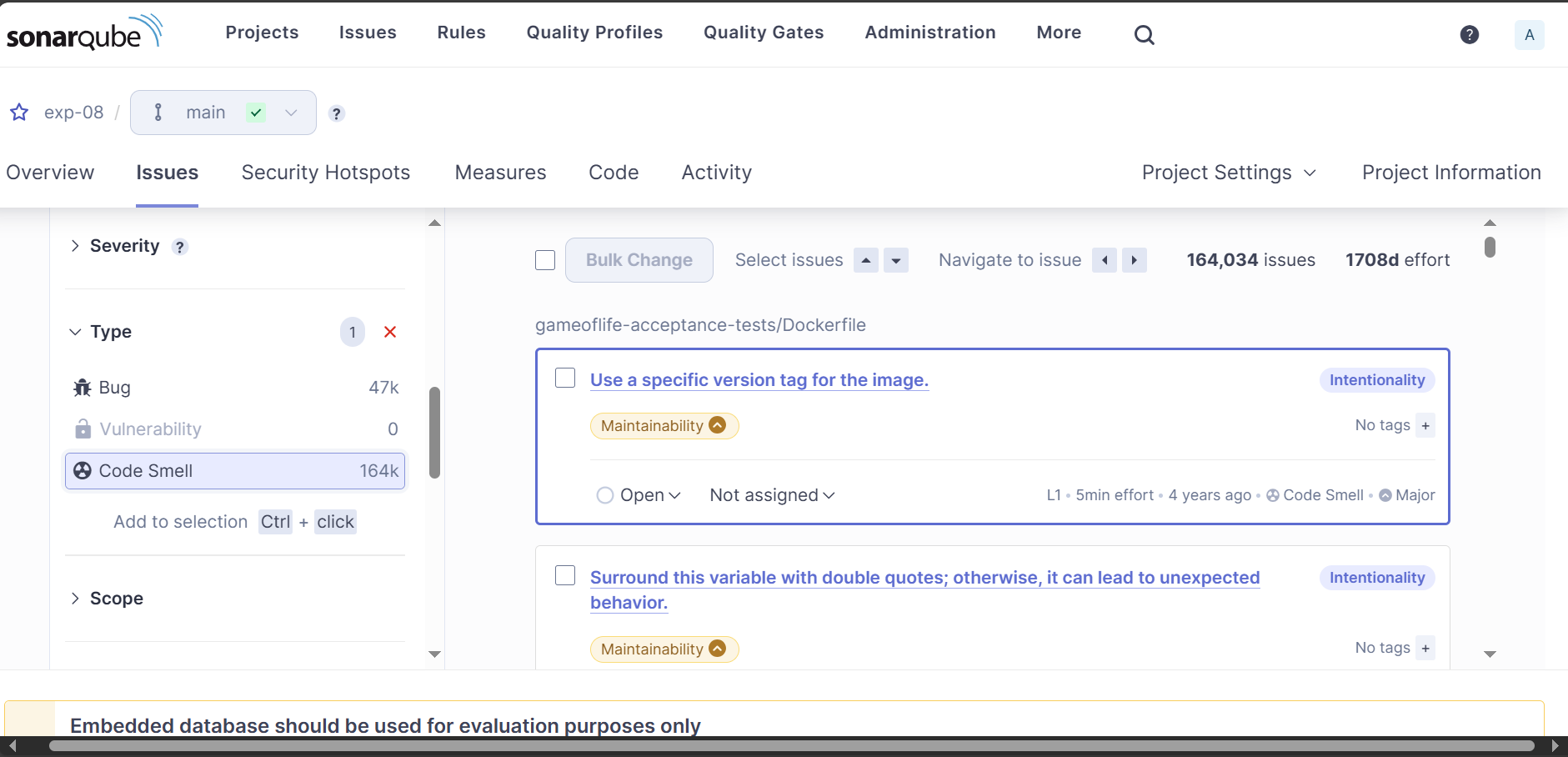


Within the issues section, let us explore different tabs

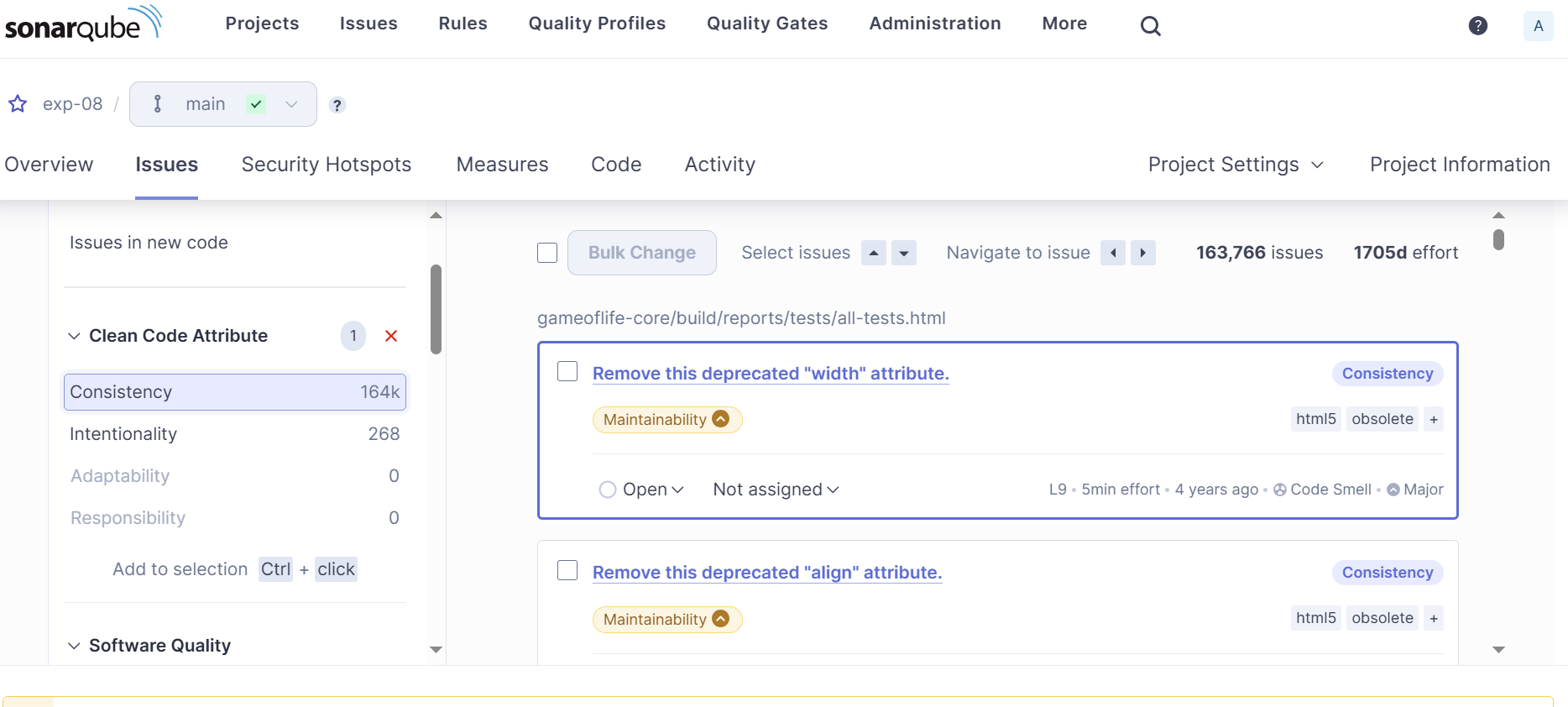
1. Bugs inside Type section



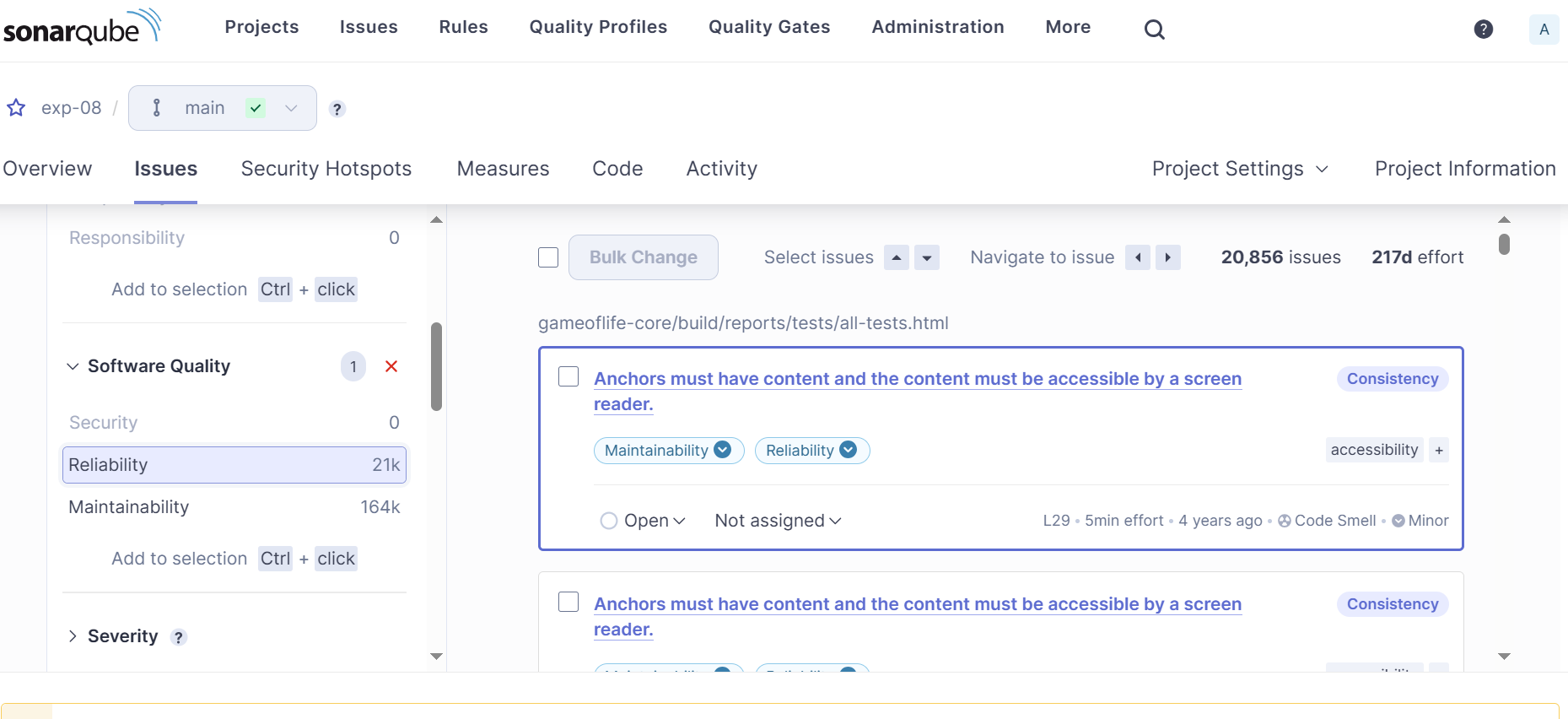
1. Code Smell inside the Type section



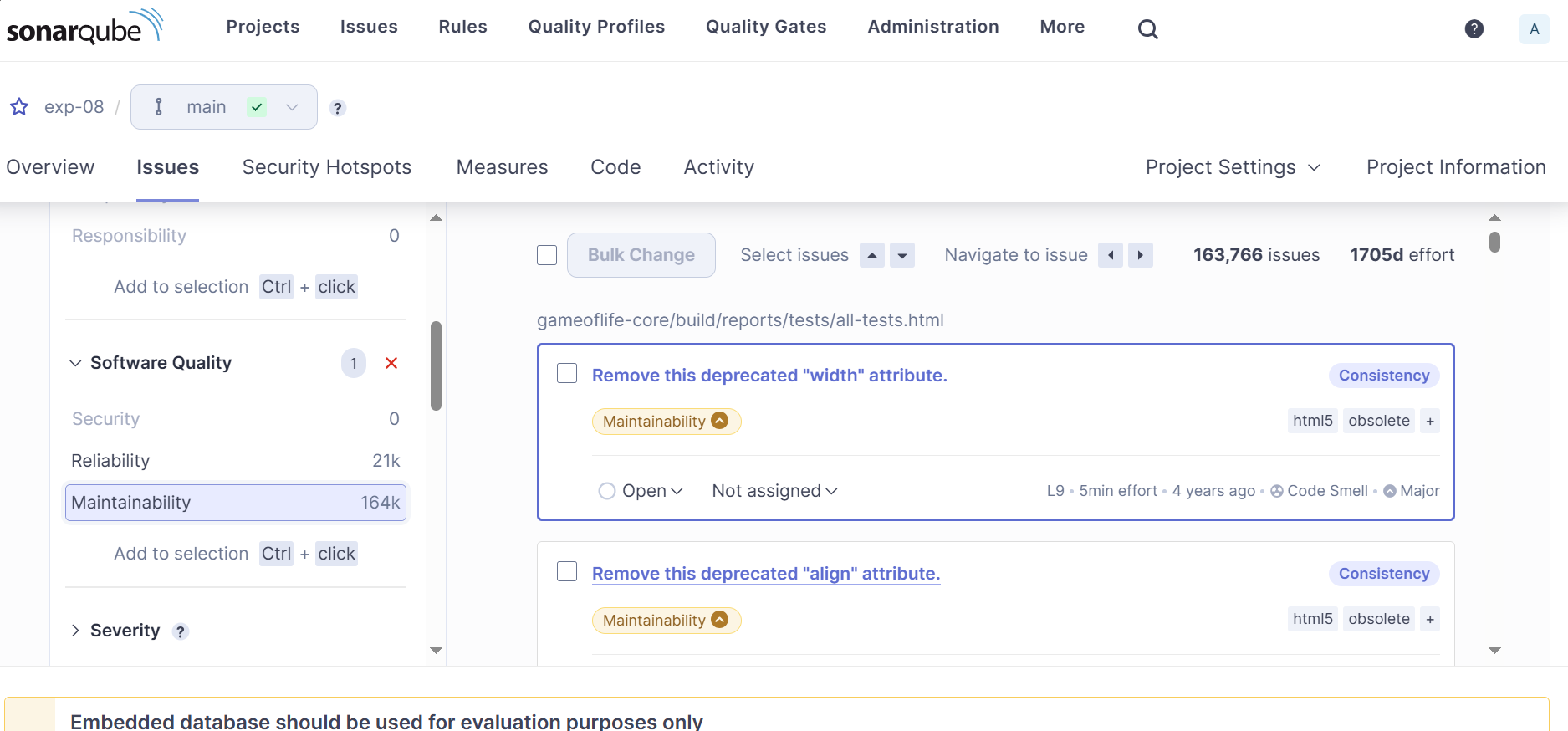
1. Consistency inside Clean Code Attribute



1. Reliability inside Software Quality



1. Maintainability



**Conclusion:**

In this experiment, we set up a Jenkins CI/CD pipeline integrated with SonarQube to automate static analysis on a sample application. Jenkins was configured to trigger builds and run SonarQube’s analysis with every code change, detecting bugs, code smells, and security vulnerabilities. This pipeline provided continuous monitoring and ensured early detection of issues, improving code quality and security. The experiment showcased how integrating CI/CD pipelines with SonarQube enhances development efficiency and ensures better, more reliable software