Sonar Setup

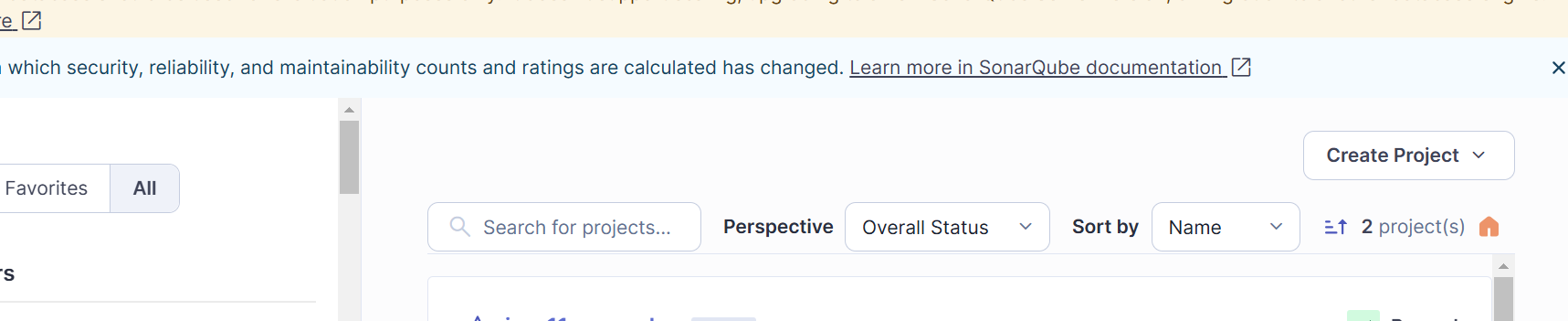
By default, the server running within the container will listen on port 9000. You can expose the container port 9000 to the host port 9000 with the -p 9000:9000 argument to docker run, like the command below:

docker run --name sonarqube-custom -p 9000:9000 sonarqube:community

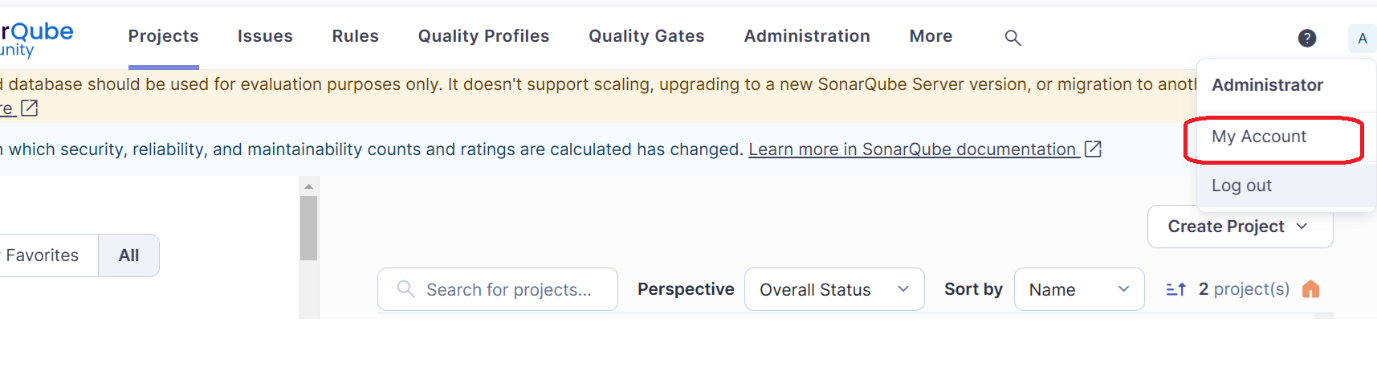
You can then browse to http://localhost:9000 or http://host-ip:9000 in your web browser to access the web interface.

Login in the sonar portal

Create project



Create Token



Based on the Jenkins pipeline from your document, here is the **equivalent GitLab CI/CD pipeline** in .gitlab-ci.yml format that performs the same stages and integrates with SonarQube for code quality analysis:

.**gitlab-ci.yml**

stages:

- checkout

- test

- report

- package

- sonar

variables:

SONARQUBE\_URL: "http://192.168.111.130:9000" # Adjust this if needed

SONARQUBE\_PROJECT\_KEY: "myproject"

GIT\_DEPTH: "0" # Required for SonarQube analysis

MAVEN\_OPTS: "-Dmaven.repo.local=.m2/repository"

cache:

paths:

- .m2/repository

before\_script:

- echo "Using Java version:"

- java -version

- echo "Using Maven version:"

- mvn -version

checkout:

stage: checkout

script:

- echo "Checkout is handled automatically by GitLab CI"

test:

stage: test

script:

- mvn clean

- mvn compile

- mvn test

generate-cucumber-report:

stage: report

script:

- mvn verify

create-package:

stage: package

script:

- mvn package

generate-report:

stage: report

script:

- mvn verify

sonarqube-analysis:

stage: sonar

script:

- mvn sonar:sonar \

-Dsonar.projectKey="$SONARQUBE\_PROJECT\_KEY" \

-Dsonar.host.url="$SONARQUBE\_URL" \

-Dsonar.login="$SONAR\_TOKEN"

only:

- master

**GitLab CI/CD Secret Setup**

1. Go to **GitLab > Project > Settings > CI/CD > Variables**.
2. Add a variable:
   * SONAR\_TOKEN → set it to your actual SonarQube token.