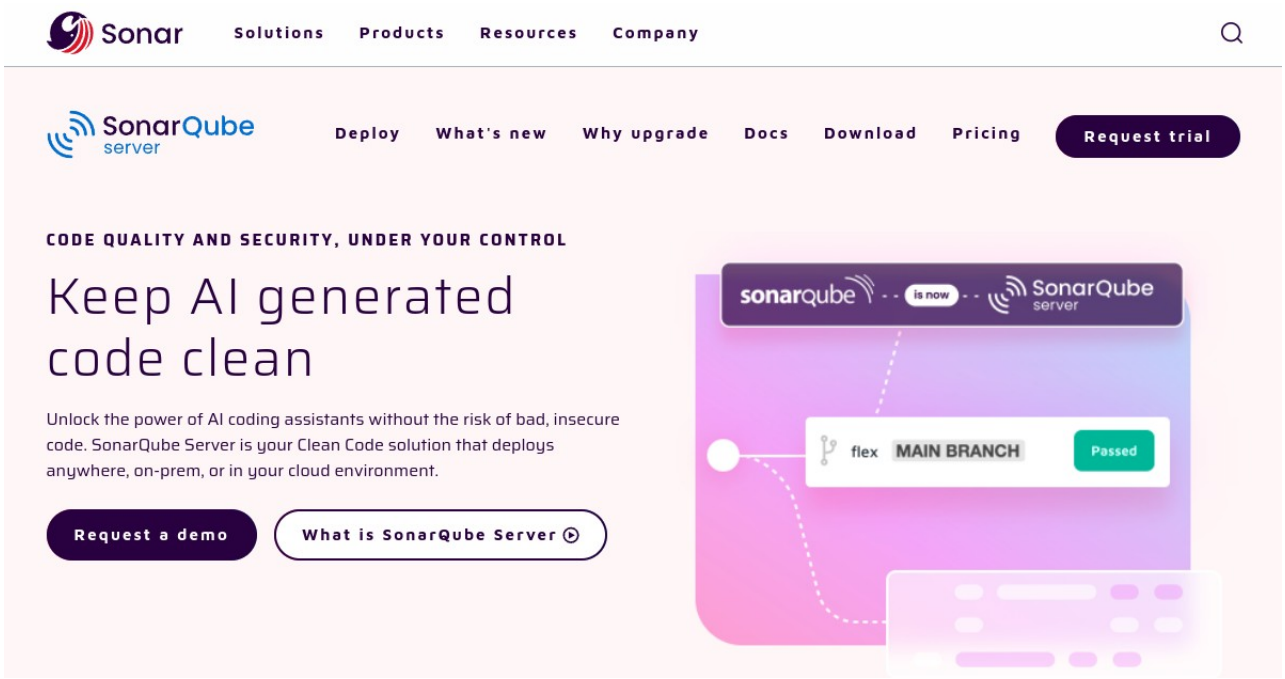
# Static analysis using SonarQube

## 1. Overview




In this lab, you are going to learn how to use SonarQube to generate static code analysis report.

SonarQube includes support for the programming languages Java (including Android), C#, PHP, JavaScript, TypeScript, C/C++, Ruby, Kotlin, Go, COBOL, PL/SQL, PL/I, ABAP, VB.NET, VB6, Python, RPG, Flex, Objective-C, Swift, CSS, HTML, and XML.

# 2. Outcomes

Upon completion of this session, you should be able to

- Use SonarQube to analysis source code
- Incorporating SonarQube into your team project
- Analysis the findings generated by SonarQube

# 3. Docker Compose for SonarQube

Use the provided sonarqube-compose.yml file to set up a SonarQube instance with a PostgresSQL databse.

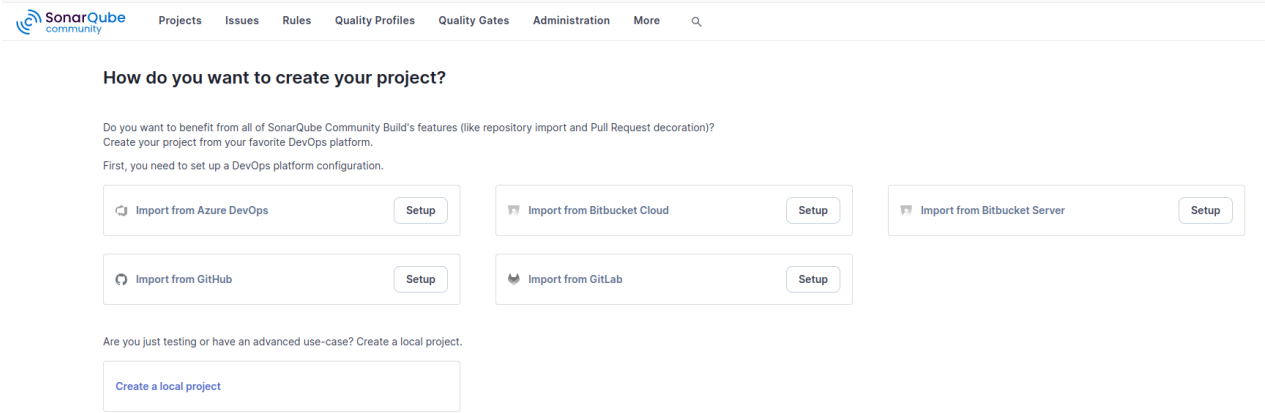
```
docker-compose -f sonarqube-compose.yml up
```

SonarQube UI will be available at

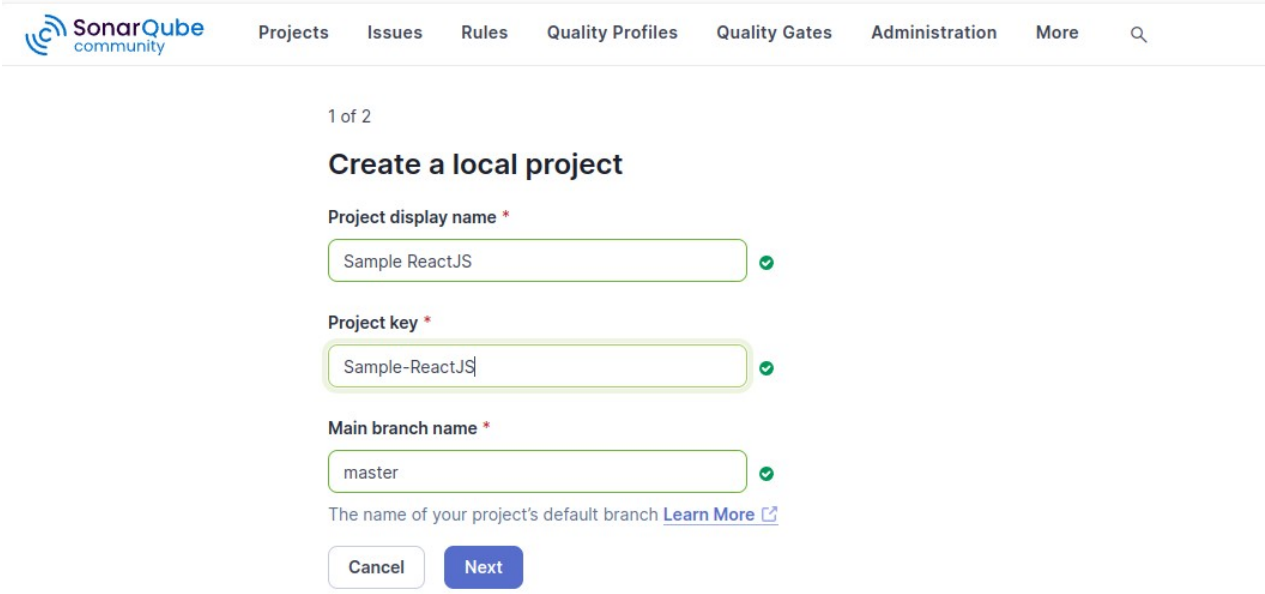
```
http://localhost:9000
Default login: admin / admin
```

# 4. Create SonarQube Projects

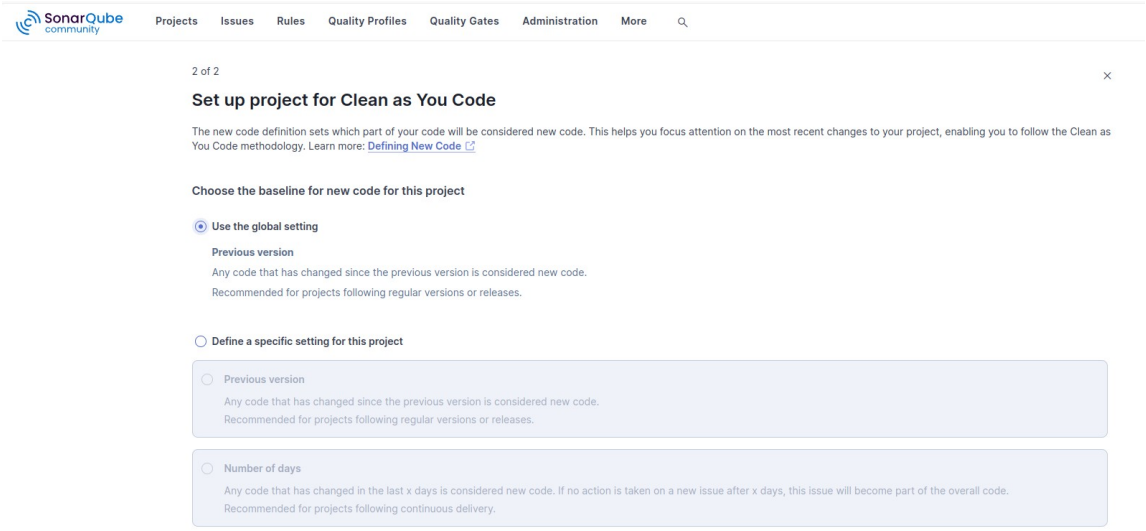
You can use SonarQube for code scanning while developing your project.



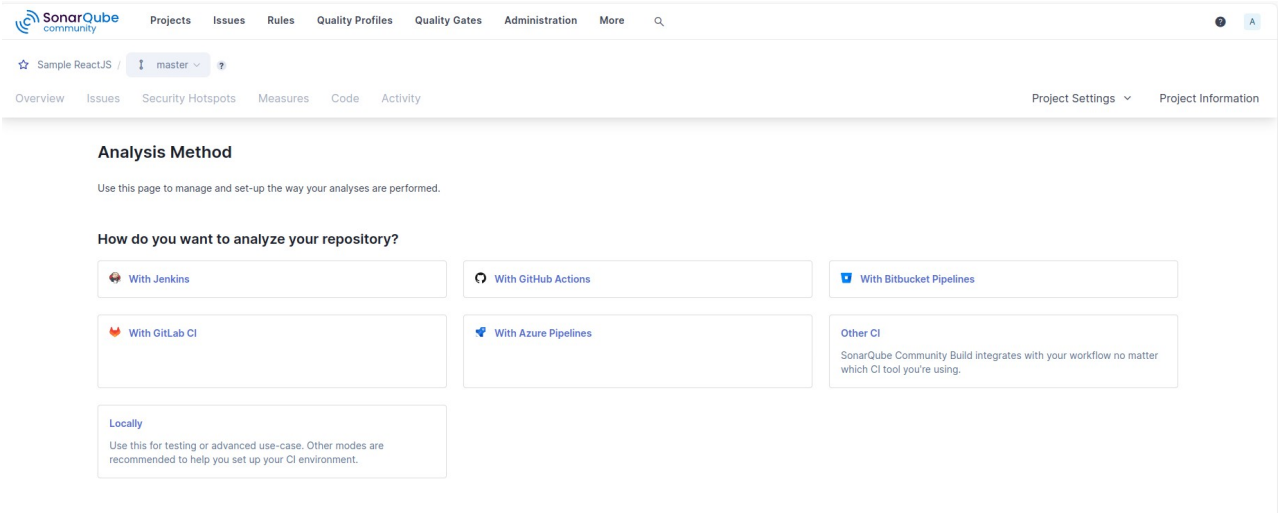

1. Click the “Create a local project” link, located at the bottom left of the page.



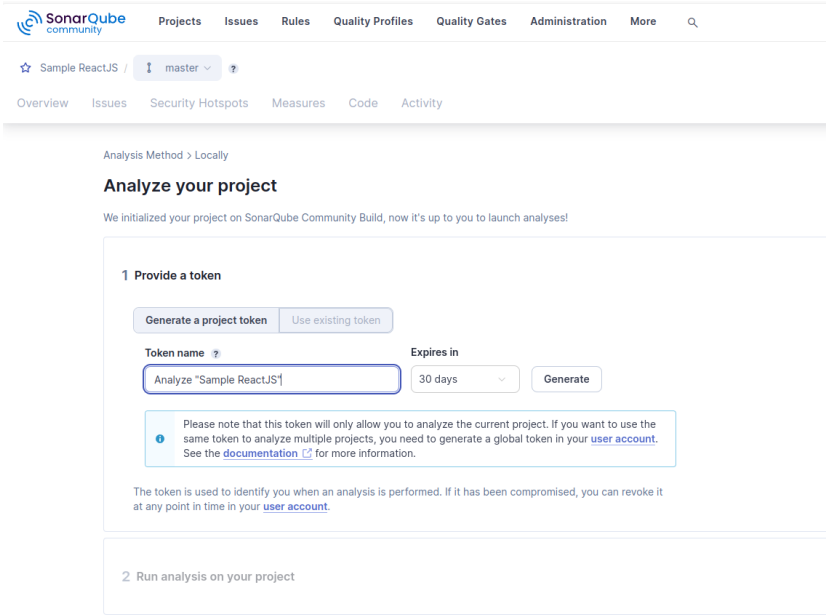

2. Input the project key and project name.



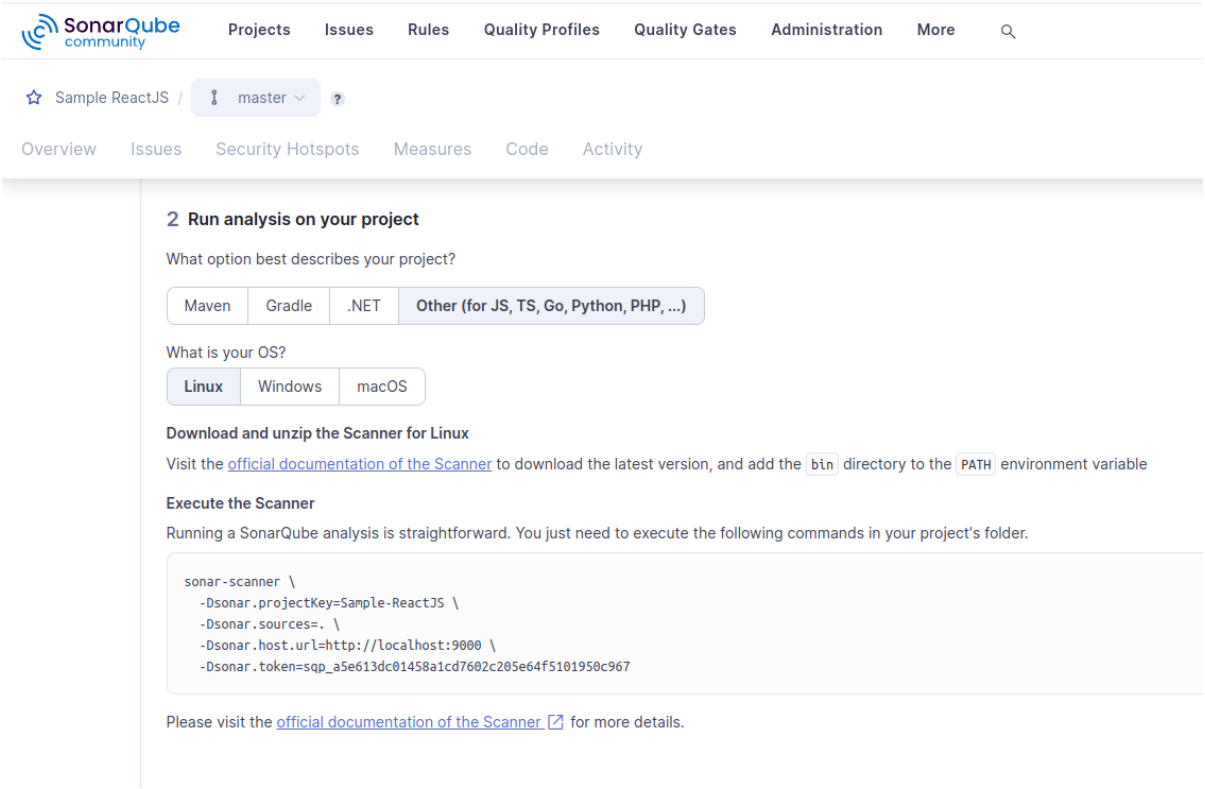

3. Press “Use global settings”.




4. Press “Locally”.




5. Generate a token for your project, copy the token and you will be using the token in the Github Actions.

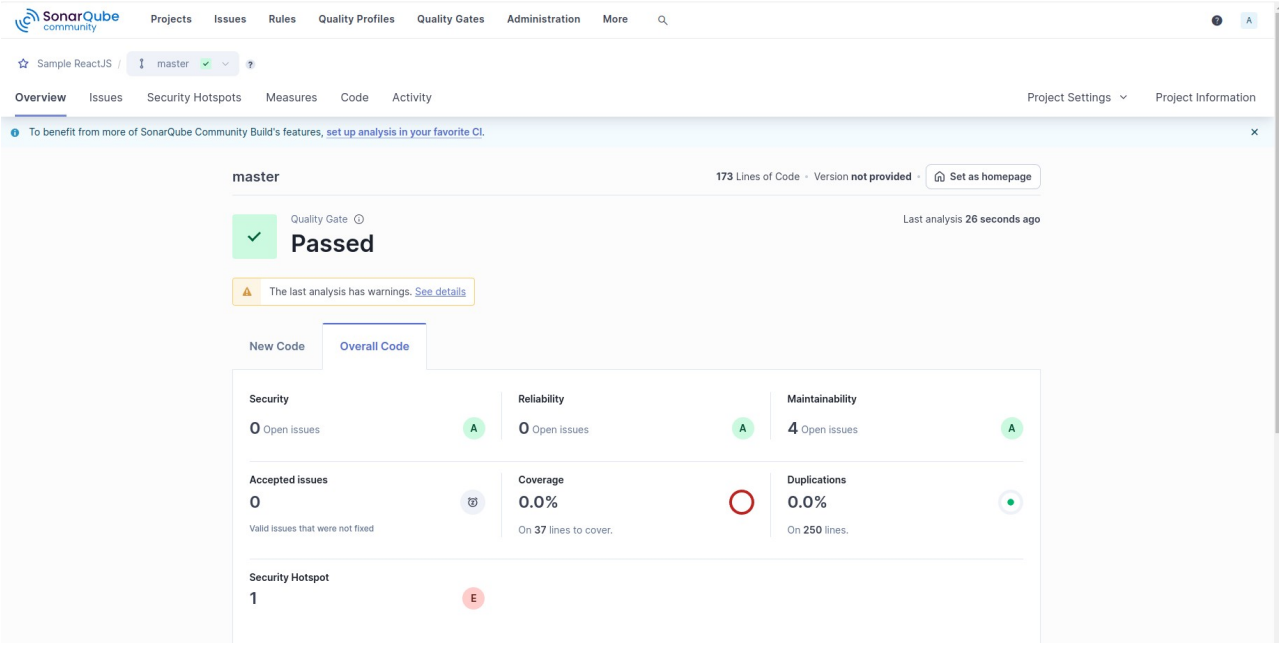



6. Select the language and OS you are using, copy the token information.

7. You can download and install sonar scanner on your laptop. Or you can use docker to run sonar scanner.

```
docker run --rm --network host -v "$(pwd):/usr/src" sonarsource/sonar-scanner-cli \
  -Dsonar.projectKey=<Sample-ReactJS> \
  -Dsonar.sources=/usr/src \
  -Dsonar.host.url=http://localhost:9000 \
  -Dsonar.login=<sonar token>
```

8. View your Sonar Scanner result in SonarQube.



# 5. GitHub Action for SonarQube Analysis

**Network Configuration**

You have to configure your network so that Github Action can reach your SonarQube instance through your domain name.

**Generate a SonarQube Token**

- Go to your sonarqube console (e.g. http://localhost:9000)
- Navigate to User > My Account > Security
- Generate a new token.
- Store this token as a GitHub Secret named SONARQUBE_TOKEN
    - Go to Settings > Secrets and variables > Actions > New repository secret.
    - Add SONARQUBE_TOKEN (your SonarQube authentication token).
    - Add SONAR_HOST_URL (your SonarQube instance URL, reachable by Github)

Modify the provided sonarqube workflow template according to your project needs.

Then copy the workflow file to .github/workflows/sonarqube.yml

When you push code, GitHub Actions will:

1. Run SonarQube Scanner to analyze the project.
2. Send results to SonarQube Dashboard.

# 6. Viewing Reports

After the workflow runs, view the analysis in SonarQube UI:

`http://localhost:9000/dashboard?id=my-javascript-project`

# 7. Docker memory issue

You may face the out-of-memory issue on the SonarQube docker. You may use the following command to increase the memory from 2GB to 3GB:

```
docker run --memory=3g the-remaining-command
```

# 8. Reference

https://docs.sonarqube.org/latest/

https://docs.sonarsource.com/sonarqube-server/9.9/analyzing-source-code/scanners/sonarscanner/

https://docs.sonarsource.com/sonarqube-server/10.6/devops-platform-integration/github-integration/adding-analysis-to-github-actions-workflow/

**END OF DOCUMENT**