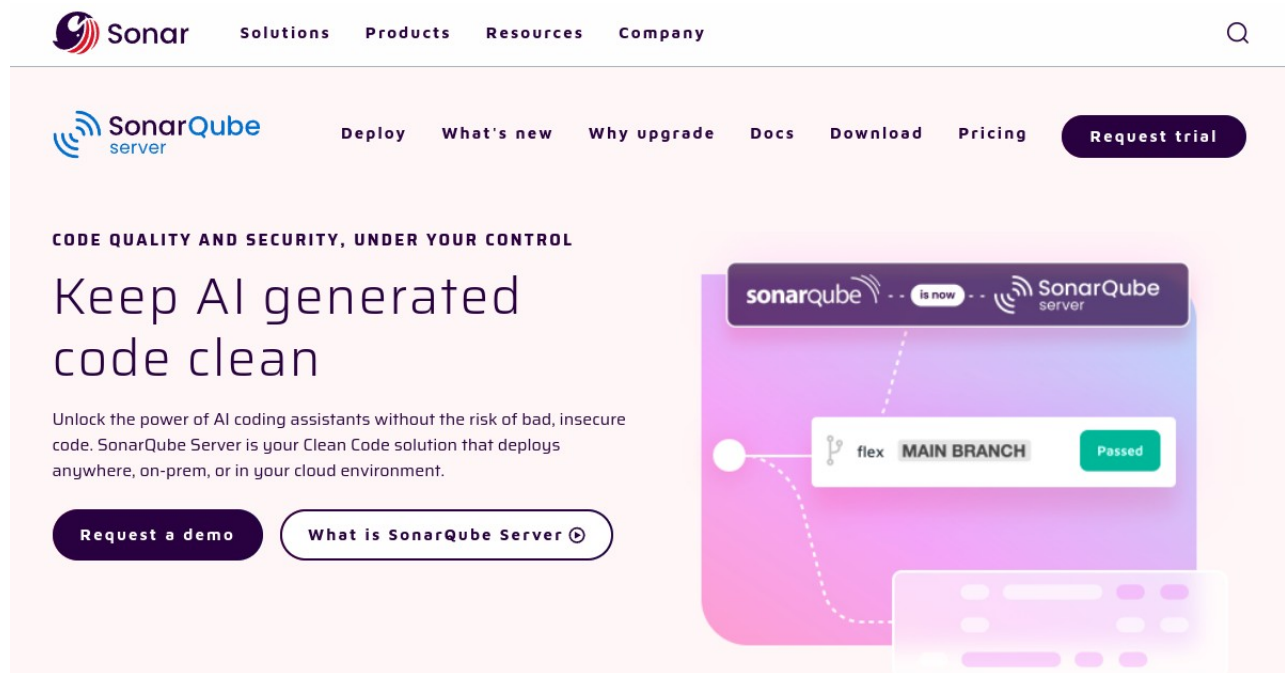


# Static analysis using SonarQube

## 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 PostgreSQL database.

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

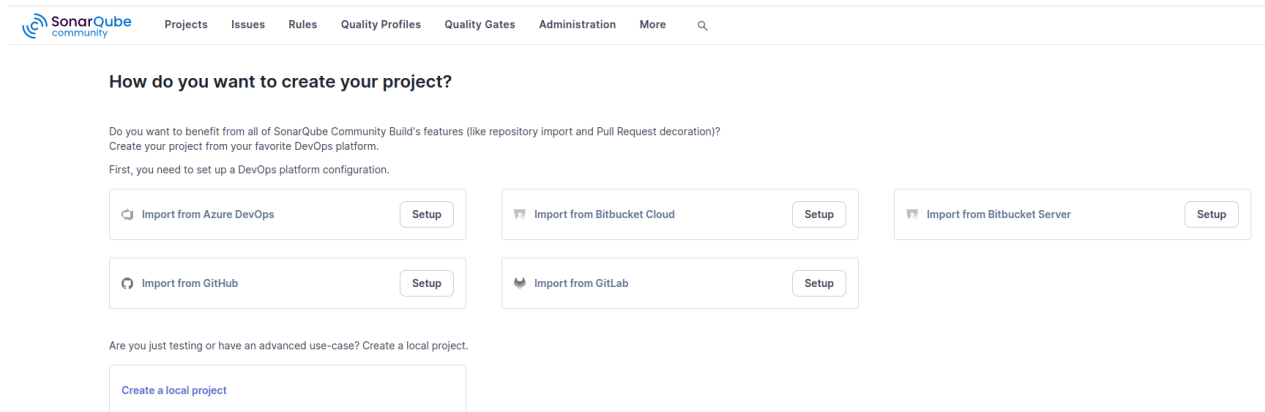
SonarQube UI will be available at

<http://localhost:9000>

Default login: admin / admin

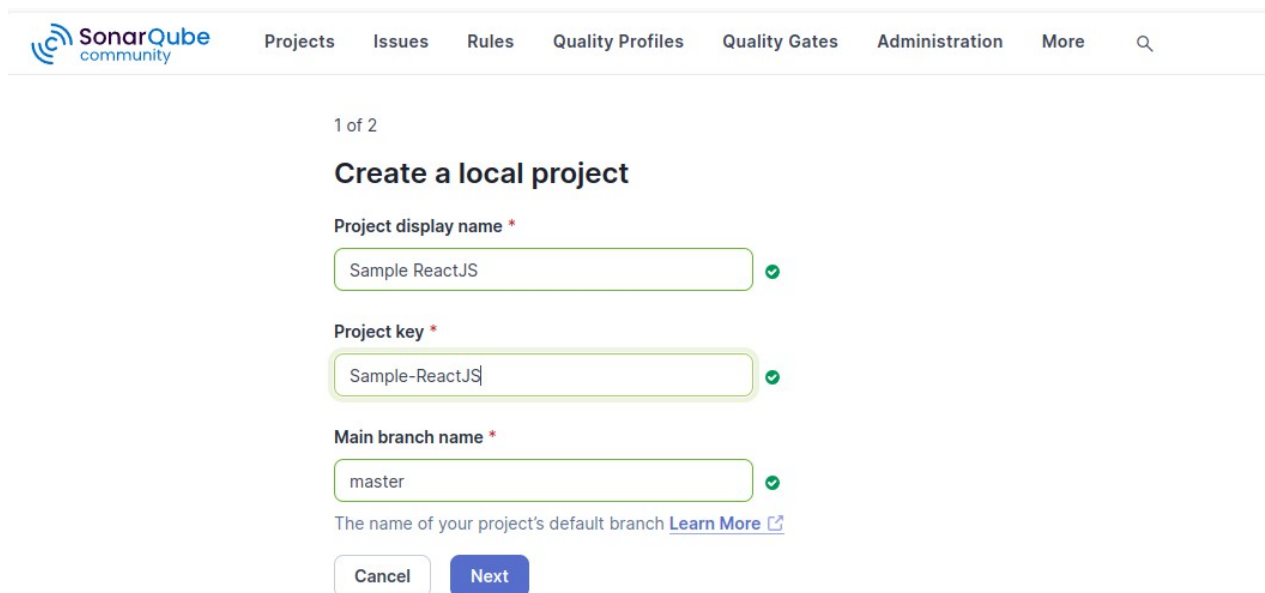
## 4. Create SonarQube Projects

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



The image shows the SonarQube Community homepage. At the top is a navigation bar with links: Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. Below the navigation bar is a section titled "How do you want to create your project?". It contains a paragraph asking if the user wants to benefit from all features of SonarQube Community Build, followed by a link to "Create your project from your favorite DevOps platform." and a note that a DevOps platform configuration is needed first. Below this are five buttons: "Import from Azure DevOps", "Import from Bitbucket Cloud", "Import from Bitbucket Server", "Import from GitHub", and "Import from GitLab". Each button has a "Setup" link next to it. At the bottom, there is a link "Create a local project" under the heading "Are you just testing or have an advanced use-case? Create a local project."

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



The image shows the "Create a local project" form in SonarQube. It is titled "1 of 2" and "Create a local project". The form has three input fields: "Project display name \*" with the value "Sample ReactJS", "Project key \*" with the value "Sample-ReactJS", and "Main branch name \*" with the value "master". Each field has a green checkmark icon to its right. Below the input fields is a link "The name of your project's default branch [Learn More](#)". At the bottom are two buttons: "Cancel" and "Next".

2. Input the project key and project name.

2 of 2

### Set up project for Clean as You Code

The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, enabling you to follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

**Choose the baseline for new code for this project**

☒ **Use the global setting**

**Previous version**  
Any code that has changed since the previous version is considered new code.  
Recommended for projects following regular versions or releases.

☐ **Define a specific setting for this project**

☐ **Previous version**  
Any code that has changed since the previous version is considered new code.  
Recommended for projects following regular versions or releases.

☐ **Number of days**  
Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.  
Recommended for projects following continuous delivery.

### 3. Press “Use global settings”.

Sample ReactJS / master

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

### Analysis Method

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

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

With Jenkins

With GitHub Actions

With Bitbucket Pipelines

With GitLab CI

With Azure Pipelines

**Other CI**  
SonarQube Community Build integrates with your workflow no matter which CI tool you're using.

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

### 4. Press “Locally”.

The screenshot shows the SonarQube Community web interface. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. Below this, a breadcrumb trail shows 'Sample ReactJS / master'. The main content area is titled 'Analyze your project' and includes a sub-header 'Analysis Method > Locally'. A message states: 'We initialized your project on SonarQube Community Build, now it's up to you to launch analyses!'. The first step, '1 Provide a token', contains two buttons: 'Generate a project token' and 'Use existing token'. Below these, there is a 'Token name' field with a help icon, containing the text 'Analyze "Sample ReactJS"', and an 'Expires in' dropdown menu set to '30 days'. A 'Generate' button is to the right. A blue information box below the form states: 'Please note that this token will only allow you to analyze the current project. If you want to use the same token to analyze multiple projects, you need to generate a global token in your [user account](#). See the [documentation](#) for more information.' A note at the bottom explains the token's purpose and how to revoke it. The second step, '2 Run analysis on your project', is partially visible at the bottom.

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

**2 Run analysis on your project**

What option best describes your project?

Maven Gradle .NET Other (for JS, TS, Go, Python, PHP, ...)

What is your OS?

Linux Windows macOS

**Download and unzip the Scanner for Linux**

Visit the [official documentation of the Scanner](#) to download the latest version, and add the `bin` directory to the `PATH` environment variable

**Execute the Scanner**

Running a SonarQube analysis is straightforward. You just need to execute the following commands in your project's folder.

```
sonar-scanner \
-Dsonar.projectKey=Sample-ReactJS \
-Dsonar.sources=. \
-Dsonar.host.url=http://localhost:9000 \
-Dsonar.token=sqp_a5e613dc01458a1cd7602c205e64f5101950c967
```

Please visit the [official documentation of the Scanner](#) for more details.

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.

The screenshot displays the SonarQube web interface for a project named 'Sample ReactJS'. The 'master' branch is selected, showing a 'Passed' status for the Quality Gate. The analysis was completed 26 seconds ago. The interface provides a detailed breakdown of the code quality metrics:

Metric	Value	Quality
Security	0 Open issues	A
Reliability	0 Open issues	A
Maintainability	4 Open issues	A
Accepted issues	0	B
Coverage	0.0% (On 37 lines to cover)	F
Duplications	0.0% (On 250 lines)	B
Security Hotspot	1	E

## 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**