

# Installing and Configuring SonarQube: A Step-by-Step Guide



Version 1.0

Prepared by – Ashan Ruwanpathirana

## Table of Contents

<b><i>Installing and Configuring SonarQube: A Step-by-Step Guide .....</i></b>	<b><i>1</i></b>
<b>1. Introduction .....</b>	<b>3</b>
<b>2. Prerequisites.....</b>	<b>3</b>
<b>3. Setting Up PostgreSQL Database .....</b>	<b>3</b>
<b>4. Installing SonarQube Using Docker on LocalHost .....</b>	<b>4</b>
<b>5. Setting up First SonarQube project.....</b>	<b>5</b>
<b>6. Analyze code .....</b>	<b>9</b>

## 1. Introduction

SonarQube is an open-source platform used for continuous inspection of code quality. This guide covers two installation methods: using Docker and using a ZIP file, along with setting up a PostgreSQL database to store SonarQube data.



## 2. Prerequisites

Before starting, ensure you have the following:

- Docker Installed (for Docker method): Install Docker from Docker's official website.
- Java Installed (for ZIP file method): SonarQube requires Java 11 or higher.
- Sufficient System Resources: Ensure your machine has at least 2 GB of RAM allocated to Docker, sufficient CPU, and disk space.

## 3. Setting Up PostgreSQL Database

Set up a PostgreSQL database using Docker, which SonarQube will use to store its data.

1. Pull the latest PostgreSQL Docker image from Docker Hub.

```
docker pull postgres
```

2. Run the PostgreSQL container with a specified database name, user, and password.
  - “-d” - Run the container in detached mode.
  - “--name sonarqube-db” - Name the container as sonarqube-db
  - “-e POSTGRES\_USER=sonar” - Set the PostgreSQL username to sonar.
  - “-e POSTGRES\_PASSWORD=sonar” - Set the PostgreSQL Password to sonar.

- “-e POSTGRES\_DB=sonarqube” - Set the PostgreSQL database name to sonarqube.
- “-p 5432:5432” - Map port 5432 on the host to port 5432 on the container.

## 4. Installing SonarQube Using Docker on LocalHost

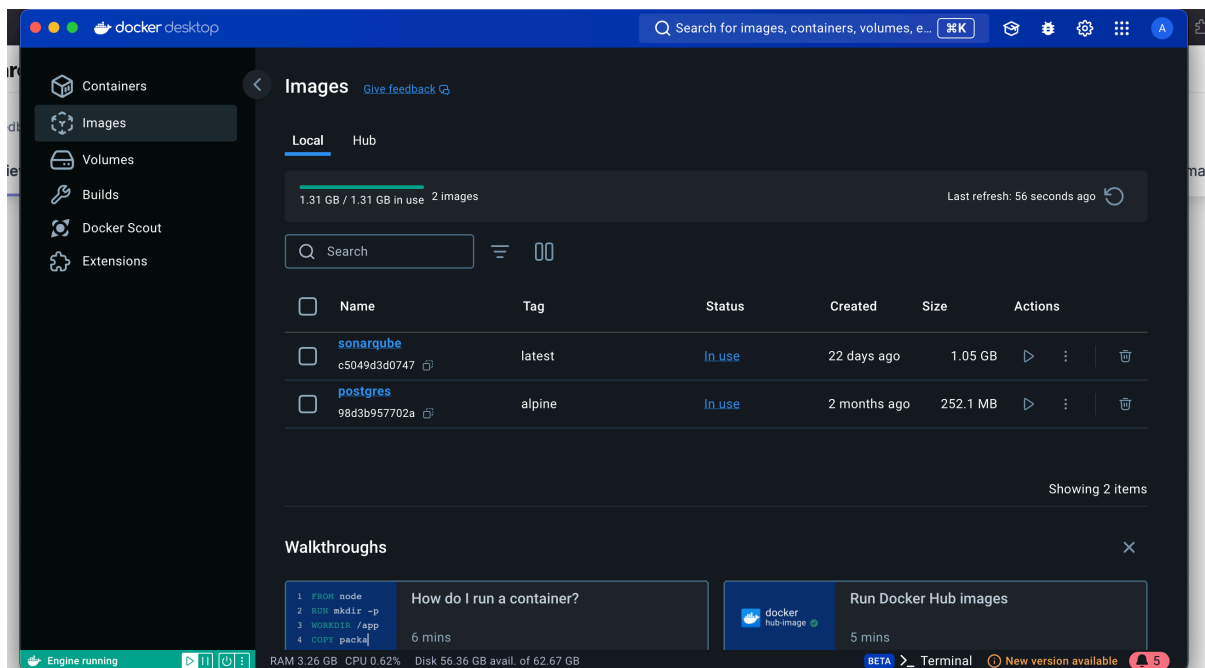
Pull the latest SonarQube image from Docker Hub.

```
docker pull sonarqube
```

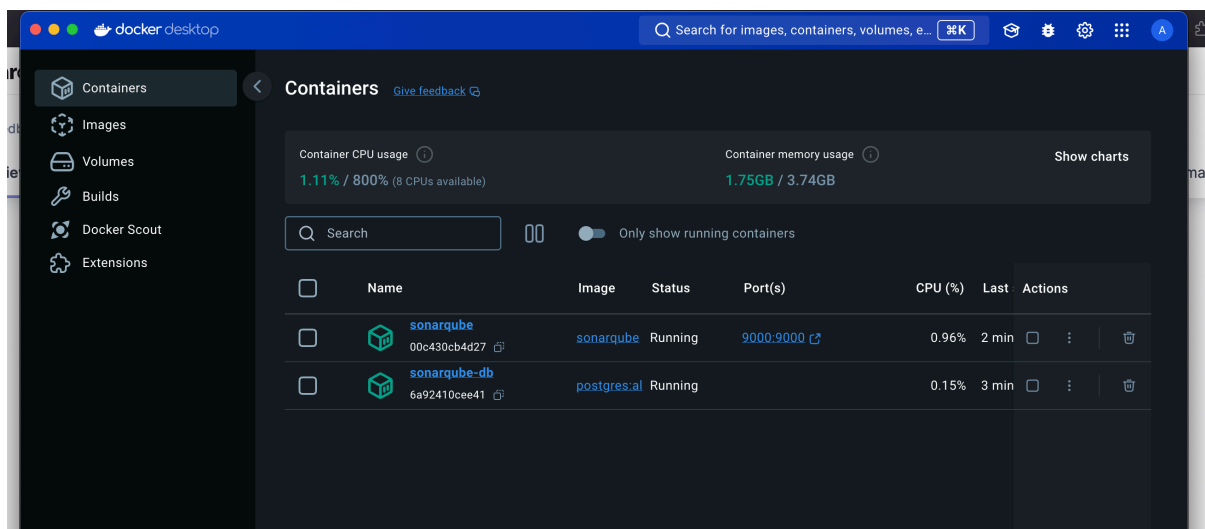
Run the SonarQube container and link it to the PostgreSQL database.

```
docker run -d --name sonarqube -p 9000:9000 --link sonarqube-db:db -e SONARQUBE_JDBC_URL=jdbc:postgresql://db:5432/sonarqube -e SONARQUBE_JDBC_USERNAME=sonar -e SONARQUBE_JDBC_PASSWORD=sonar sonarqube
```

- “docker run” - This command is used to create and start a new container from a specified image.
- “-d” - This flag runs the container in detached mode, meaning it runs in the background.
- “--name sonarqube” - This option assigns a name (sonarqube) to the container, making it easier to reference later.
- “-p 9000:9000” - This option maps port 9000 of your host machine to port 9000 of the container. It allows you to access the SonarQube web interface via <http://localhost:9000>.
- “--link sonarqube-db:db” - This creates a link to another container named sonarqube-db, allowing the SonarQube container to communicate with it. The alias db is used inside the SonarQube container to refer to this linked container.
- “-e SONARQUBE\_JDBC\_URL=jdbc:postgresql://db:5432/sonarqube” - This sets the JDBC URL for connecting to the PostgreSQL database. It specifies that the database server is reachable at the alias db on port 5432, and it’s looking for a database named sonarqube.
- “-e SONARQUBE\_JDBC\_USERNAME=sonar” - This sets the username to connect to the PostgreSQL database.
- “-e SONARQUBE\_JDBC\_PASSWORD=sonar” - This sets the password for the specified username.
- “Sonarqube” - This is the name of the Docker image that will be used to create the container. In this case, it's the official SonarQube image from Docker Hub.

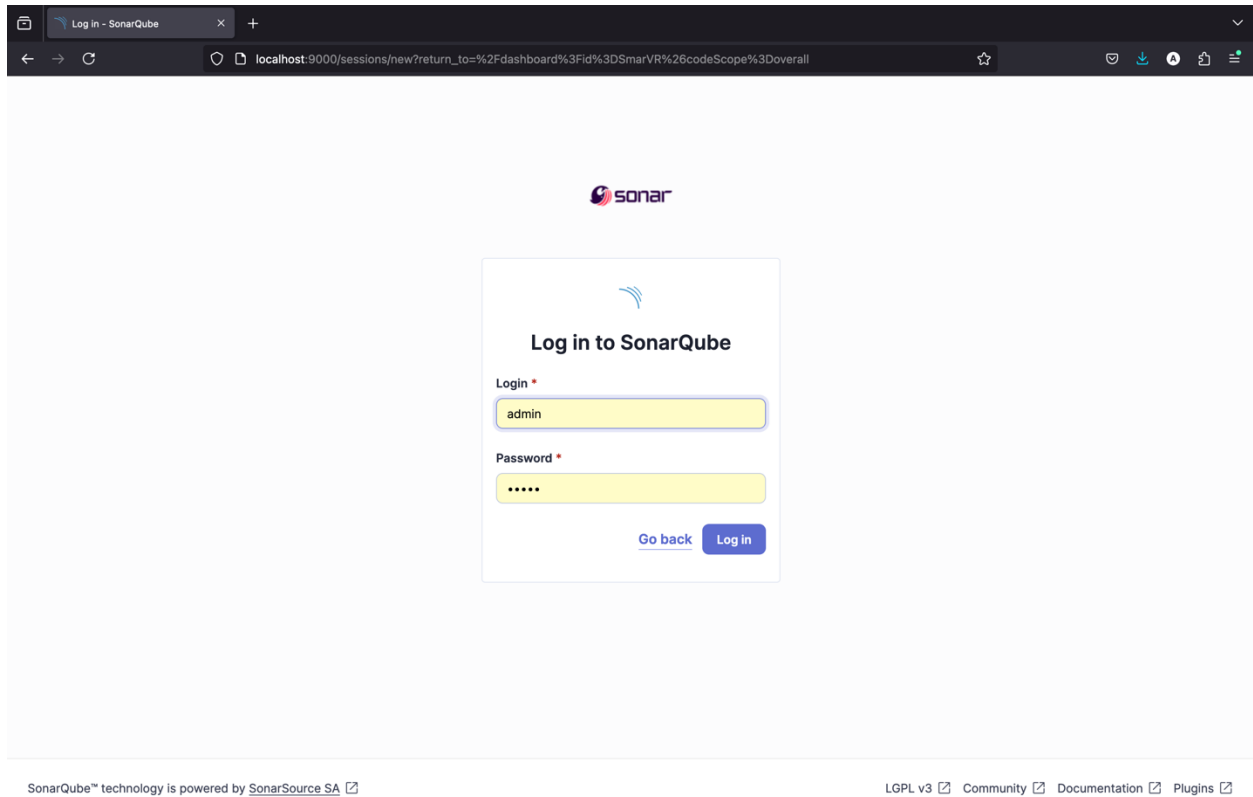


Postgres and SonarQube Images



Running as Containers

## 5. Setting up First SonarQube project



Type username=admin and password=admin

Create a local project

localhost:9000/projects/create?mode=manual

sonarqube

ProjectsIssuesRulesQuality ProfilesQuality GatesAdministrationMore

1 of 2

Create a local project

Project display name \*

Project key \*

Main branch name \*

main

The name of your project's default branch [Learn More](#)

CancelNext

SonarQube™ technology is powered by [SonarSource SA](#)

Community Edition v10.6 (92116) ACTIVE LGPL v3 Community Documentation Plugins Web API

## Project Initialization dashboard

SonarQube

localhost:9000/tutorials?id=Redback-Orion-test

sonarqube

ProjectsIssuesRulesQuality ProfilesQuality GatesAdministrationMore

Redback-Orion test / main

OverviewIssuesSecurity HotspotsMeasuresCodeActivityMore

Project SettingsProject 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

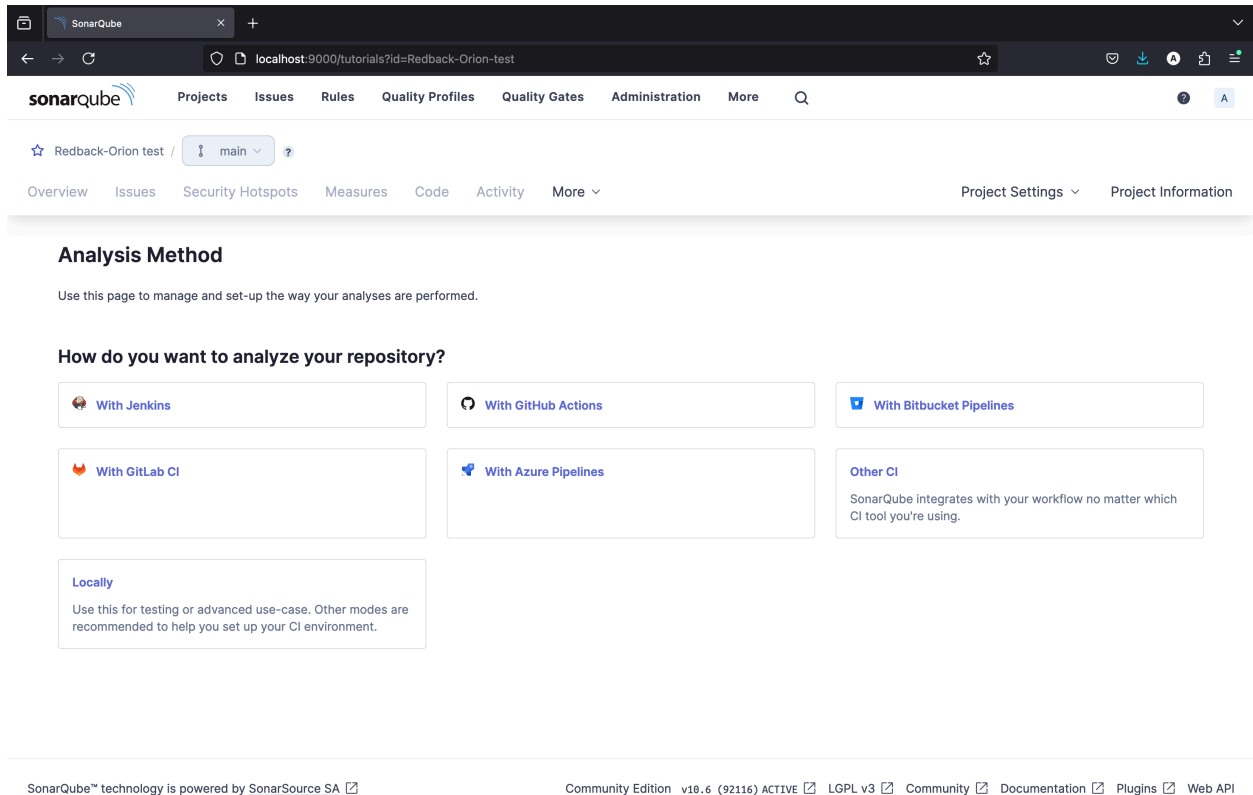
Locally

SonarQube integrates with your workflow no matter which CI tool you're using.

SonarQube™ technology is powered by [SonarSource SA](#)

Community Edition v10.6 (92116) ACTIVE LGPL v3 Community Documentation Plugins Web API

## Project Analysis – Choose appropriate method where code based



## 1. Build Your Analysis

- SonarQube will provide you instructions on how to evaluate your project after you've created it. Generally, you will have options for various build tools (e.g., Gradle, Ant, Maven, etc.).
- Depending on the build system for your project, select the appropriate option.

## 2. Create a Token

- Tokens are used by SonarQube to authenticate users during analysis. For your project, click the "Generate" button to create a token. This token will be utilized in your local analysis or CI/CD workflow, so keep it safe.



## 6. Analyze code

### 1. Install the SonarScanner

You need to install the SonarScanner, which is used to perform the analysis. You can download it from the [SonarScanner download page](#).

### 2. Configure the SonarScanner

Create a configuration file named `sonar-project.properties` in your project directory. Here's a basic template

```
sonar.projectKey=my-first-project
```

```
sonar.projectName=My First Project
```

```
sonar.projectVersion=1.0
```

```
sonar.sources=.
```

```
sonar.language=java # Change this based on your project's language
```

```
sonar.host.url=http://localhost:9000
```

```
sonar.login="YOUR_GENERATED_TOKEN"
```

### 3. Run code analysis

- Open a terminal and navigate to your project directory.
- Run the following command

```
sonar-scanner
```

- This command will start the analysis process, sending the results to SonarQube server.