

Setting Up and Running SonarQube for Efficient Source Code Project Analysis

Scenario:

As the Lead Software Engineer in a growing tech startup, you are entrusted with the responsibility of implementing a robust code quality management system to ensure the delivery of high-quality software products. With an expanding code base and development team, maintaining a standard code quality across projects has become critical. After evaluating various options, you've decided to integrate SonarQube into the development pipeline to enable continuous code inspection and analysis, thereby identifying and addressing potential technical debt and vulnerabilities early in the development lifecycle.

Description:

In this project, you will lead the implementation of SonarQube for efficient project analysis, enabling the development team **to proactively identify and resolve code quality issues**. Your responsibilities workflow during the SonarQube dashboard, integrating it seamlessly into the existing development workflow, and setting up customized quality gates tailored to the specific requirements of the projects.

Contents

1	Introduction.....	3
2	Documentation.....	6
2.1	SonarQube Documentation.....	6
2.2	Linux Commands and VIM Commands.....	6
3	Pre-requisites.....	7
4	Install and Setup Sonarqube Dashboard.....	8
4.1	Create a sample project in SonarQube Dashboard.....	11
5	Install SonarQube Scanner & Run a SonarQube Scan.....	16
6	Share your Learnings.....	21
6.1	LinkedIn.....	22
6.2	WhatsApp.....	24
7	Clean up Resources.....	25
8	Troubleshooting.....	26
9	Summary.....	27

1 INTRODUCTION

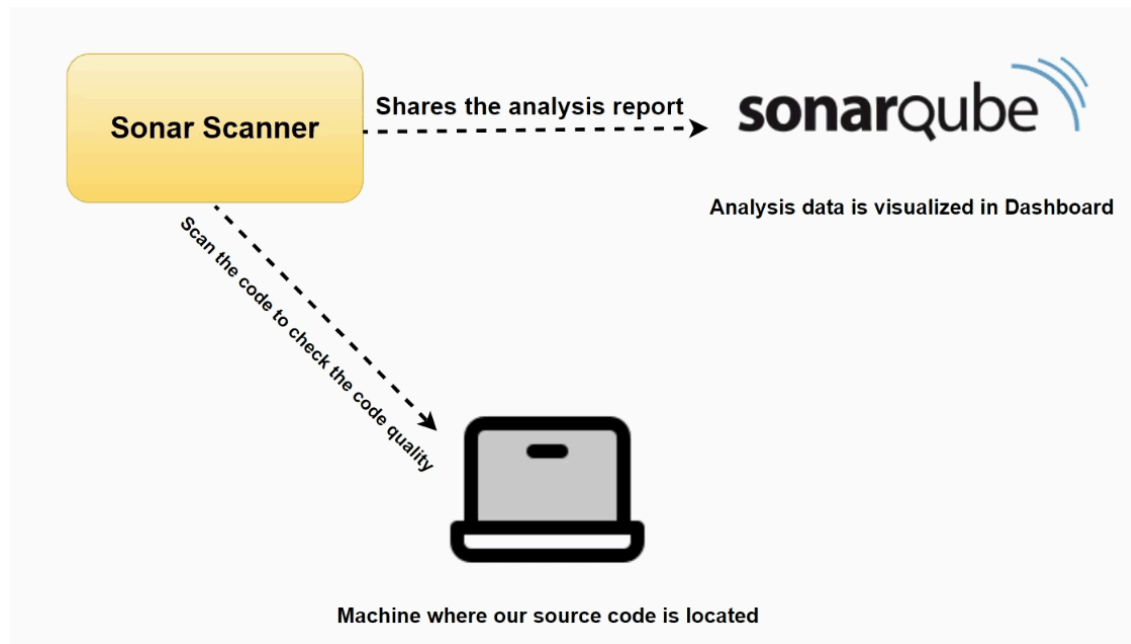
SonarQube, developed by Sonar Source, is an open-source platform designed to provide **continuous inspection of code quality**. It serves as a robust tool for developers to ensure their code adheres to best practices and industry standards, while also **detecting potential bugs, code smells, and security vulnerabilities**. With support for over 20 programming languages, SonarQube plays a crucial role in enhancing the overall quality, reliability, and security of software projects.

sonarqube

Key Features

- **Static Code Analysis:** SonarQube employs static code analysis techniques to analyze source code without executing it. By scanning the codebase, it identifies various issues such as complex code structures, potential bugs, and coding rule violations.
- **Code Smell Detection:** The platform is equipped with a set of predefined rules that identify code smells, which refer to any implementation that could be improved. This feature helps developers enhance the maintainability and readability of their code.
- **Security Vulnerability Detection:** SonarQube's security analysis capabilities enable the identification of potential security vulnerabilities, including common weaknesses and flaws that could be exploited by malicious actors. By highlighting these vulnerabilities, SonarQube aids in the development of more secure and robust software.
- **Customizable Quality Profiles:** Users can customize quality profiles based on their specific project requirements. This feature allows teams to define and enforce coding standards, ensuring consistent code quality across the entire development process.
- **Integration with DevOps Pipelines:** SonarQube seamlessly integrates with DevOps pipelines, enabling developers to incorporate code quality checks within their continuous integration and continuous delivery (CI/CD) processes. This integration facilitates the early detection of issues, reducing the overall time and effort required for quality assurance.

- **Comprehensive Reporting:** The platform provides detailed and comprehensive reports, including dashboards and visualizations, to help teams monitor code quality metrics over time. These reports offer insights into the evolution of code quality, thereby assisting in making informed decisions and prioritizing necessary improvements.



Benefits

- **Improved Code Maintainability:** By identifying and addressing code smells and technical debt, SonarQube aids in improving the overall maintainability of the codebase, making it easier for developers to understand, modify, and extend the software.
- **Enhanced Security:** With its robust security analysis capabilities, SonarQube helps in identifying and addressing potential security vulnerabilities early in the development lifecycle, reducing the risk of security breaches and data leaks.
- **Consistent Coding Standards:** By enforcing predefined coding rules and customizable quality profiles, SonarQube ensures that all code written by the development team adheres to the same set of standards, fostering consistency and uniformity across the project.
- **Efficient Collaboration:** The platform encourages collaboration among team members by providing a shared understanding of the codebase's quality and security status. This shared visibility helps in fostering effective communication and collaboration among developers, testers, and other stakeholders.

This guide Covers:

- Install and setup Sonarqube Dashboard
- Create a sample project in SonarQube Dashboard
- Install Sonarqube Scanner & Run a Sonarqube Scan

K21Academy

2 DOCUMENTATION

2.1 SonarQube Documentation

1. Install & Setup Sonarqube

<https://docs.sonarsource.com/sonarqube/latest/try-out-sonarqube/>

2. Sonar Scanner

<https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/>

3. Docker Compose

<https://docs.docker.com/compose/>

2.2 Linux Commands and VIM Commands

Note: If you are new to Linux and want to learn the basics of Linux, there is a basic Linux course available in the "Course" section, under the bonus section. If you are unable to navigate to it, please reach out to our WhatsApp Group or drop us an email at support@k21academy.com.

Bonus 1: Linux For Beginners ✓

- Lesson 1: Register Free Azure Trial Account (18:49 min)
- Lesson 2: Create and Configure Ubuntu Machine on Azure Cloud (19:27 min)
- Lesson 3: Connect Azure Ubuntu Machine From MAC (03:54 min)
- Lesson 4: Linux Distribution, Connectivity, MAN Page (11:05 min)
- Lesson 5: Common Linux Commands (13:45 min)
- Lesson 6: Vi Editor, Permission, Copy File (18:49 min)
- Lesson 7: Create AWS Free Trial Account (06:51 min)

3 PRE-REQUISITES

1. Create a Cloud Account:

⇒ **On Azure:**

- Follow the steps in the Activity Guide titled
Register_For_Azure_Cloud_Account_Accessing_Console_ed**

⇒ **On AWS:**

- Follow the steps in the Activity Guide titled
Register_For_AWS_Free_Trail_Account_ed**

2. Create & Connect to an Ubuntu Virtual Machine:

- Use your Azure or AWS account to create an Ubuntu virtual machine.
- Make sure the virtual machine has size **2 vCPUs and 8 GiB of memory**.
- Make sure you **open all ports** in Ubuntu Virtual Machine.

⇒ **On Azure:**

- Follow the steps in the Activity Guide titled
Create & Connect to Ubuntu Machine on Azure Cloud_ed**

⇒ **On AWS:**

- Follow the steps in the Activity Guide titled
Create & Connect to Ubuntu Ec2 Instance on AWS Cloud_ed**

3. Install Docker:

- Once connected to the Ubuntu virtual machine, install Docker.
- Follow the steps in the Activity Guide titled

Install & Configure Docker on Ubuntu_ed**

```
$ docker --version
$ docker-compose --version
```

```
root@docker:~# docker --version
Docker version 24.0.7, build afdd53b
root@docker:~# docker-compose --version
Docker Compose version v2.23.0
```

4 INSTALL AND SETUP SONARQUBE DASHBOARD

The SonarQube dashboard offers a centralized view of code quality, enabling real-time monitoring and customizable metrics. It aids historical analysis, facilitating data-driven decision-making, and fosters team collaboration for effective quality management.

1. Login to an Ubuntu machine as the root user

```
$ sudo -i
```

```
ubuntu@docker:~$ sudo -i
root@docker:~# pwd
/root
```

2. To begin the activity guide, you'll need to clone the Git repository.

```
$ git clone https://github.com/k21academyuk/sonarqube.git
```

```
root@docker:~# git clone https://github.com/k21academyuk/sonarqube.git
Cloning into 'sonarqube'...
remote: Enumerating objects: 285, done.
remote: Counting objects: 100% (285/285), done.
remote: Compressing objects: 100% (271/271), done.
remote: Total 285 (delta 24), reused 245 (delta 8), pack-reused 0
Receiving objects: 100% (285/285), 12.89 MiB | 61.39 MiB/s, done.
Resolving deltas: 100% (24/24), done.
root@docker:~#
```

3. We are using containerized Docker images to install SonarQube and a PostgreSQL database using Docker Compose.
4. Switch to the **sonarqube** directory and list the files. Among the files, you will find **docker-compose.yml**. Open and view the contents of this file.

```
$ cd sonarqube
```

```
$ ls
```

```
$ cat docker-compose.yml
```



```

root@docker:~# cd sonarqube
root@docker:~/sonarqube# ls
docker-compose.yml  php-project  scripts
root@docker:~/sonarqube# cat docker-compose.yml
version: "3.8"
services:
  sonarqube:
    container_name: sonarqube
    image: sonarqube
    depends_on:
      - sonarqube-database
    environment:
      - SONARQUBE_JDBC_USERNAME=sonarqube
      - SONARQUBE_JDBC_PASSWORD=sonarpass
      - SONARQUBE_JDBC_URL=jdbc:postgresql://sonarqube-database:5432/sonarqube
    volumes:
      - sonarqube_conf:/opt/sonarqube/conf
      - sonarqube_data:/opt/sonarqube/data
      - sonarqube_extensions:/opt/sonarqube/extensions
      - sonarqube_bundled-plugins:/opt/sonarqube/lib/bundled-plugins
    ports:
      - 9000:9000

  sonarqube-database:
    container_name: sonarqube-database
    image: postgres:latest
    environment:
      - POSTGRES_DB=sonarqube
      - POSTGRES_USER=sonarqube
      - POSTGRES_PASSWORD=sonarpass
    volumes:
      - sonarqube_database:/var/lib/postgresql
      - sonarqube_database_data:/var/lib/postgresql/data
    ports:
      - 5432:5432

volumes:
  sonarqube_database_data:
  sonarqube_bundled-plugins:
  sonarqube_conf:
  sonarqube_data:
  sonarqube_database:
  sonarqube_extensions:
root@docker:~/sonarqube# █

```

In this docker-compose.yml file, we are creating two containers: one for SonarQube and another for the SonarQube database. We are using PostgreSQL as the database because SonarQube requires a database to store and manage analysis data and configuration settings.

5. Start the services defined in the docker-compose.yml file.

```
$ docker-compose up -d
```

```
root@docker:~/sonarqube# docker-compose up -d
Creating network "sonarqube_default" with the default driver
Creating volume "sonarqube_sonarqube_database_data" with default driver
Creating volume "sonarqube_sonarqube_bundled-plugins" with default driver
Creating volume "sonarqube_sonarqube_conf" with default driver
Creating volume "sonarqube_sonarqube_data" with default driver
Creating volume "sonarqube_sonarqube_database" with default driver
Creating volume "sonarqube_sonarqube_extensions" with default driver
Pulling sonarqube-database (postgres:latest)...
latest: Pulling from library/postgres
2cc3ae149d28: Pull complete
d1a63825d58e: Pull complete
ed6f372fe58d: Pull complete
35f975e69306: Pull complete
40c4fe86e99d: Pull complete
4795e1a32ff6: Pull complete
bcb5a54ae87d: Pull complete
d3983228bec6: Pull complete
5378bf7229e9: Pull complete
bba3241011a6: Pull complete
5e1d0413d05a: Pull complete
6a489170d05e: Pull complete
440b39aff272: Pull complete
582c79113570: Pull complete
Digest: sha256:46aa2ee5d664b275f05d1a963b30fff60fb422b4b594d509765c42db46d48881
Status: Downloaded newer image for postgres:latest
Pulling sonarqube (sonarqube:latest)...
latest: Pulling from library/sonarqube
2ec76a50fe7c: Pull complete
fab7f202453a: Pull complete
ee59ca42def8: Pull complete
2ce2282f972f: Pull complete
d2a9e456ba82: Pull complete
243afeaedad6: Pull complete
62234a870633: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:b3c2031c5256c2eba28499b1017eaf8d3c0e1aabfb2045fe0a4e8e5b31d594b
Status: Downloaded newer image for sonarqube:latest
Creating sonarqube-database ... done
Creating sonarqube ... done
```

6. List the running containers.

```
$ docker ps
```

```
root@docker:~/sonarqube# docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS        PORTS
NAMES
31d7a49956c2   sonarqube            "/opt/sonarqube/dock..." About a minute ago Up About a minute 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp
ea5e0ab0f696   postgres:latest      "docker-entrypoint.s..." About a minute ago Up About a minute 0.0.0.0:5432->5432/tcp, :::5432->5432/tcp
sonarqube-database
root@docker:~/sonarqube#
```

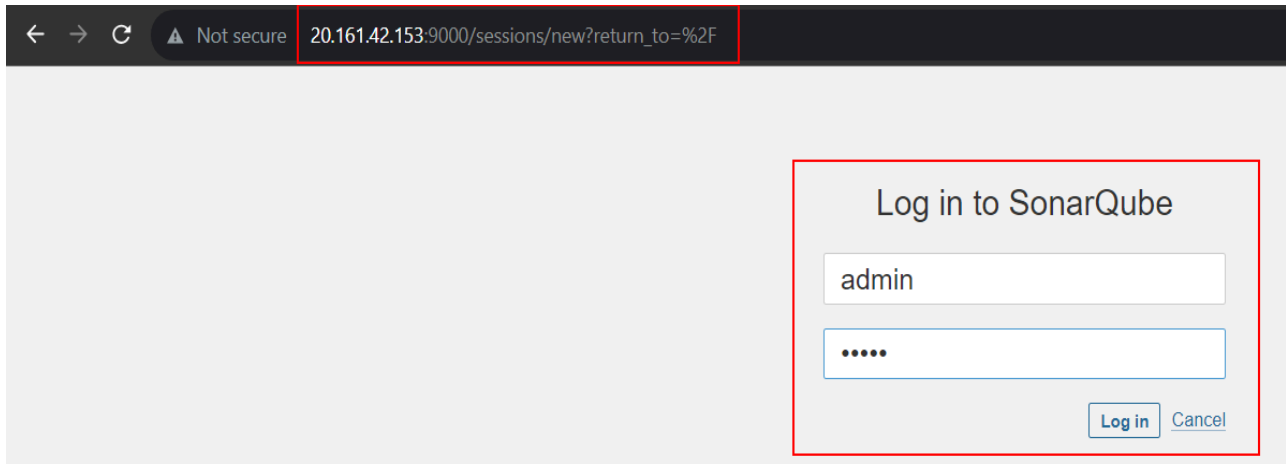
⇒ Two containers are created: sonarqube and sonarqube-database.

7. Copy the public IP address of the machine and paste it into the webpage along with port 9000.

Syntax: <publicip>:9000

Example: 20.161.42.143:9000

8. To log in to SonarQube, use the default username and password, which are both **admin**.



9. Update the password (Old password is admin).

Update your password

This account should not use the default password.

Enter a new password

All fields marked with * are required

Old Password *

New Password *

Confirm Password *

Update





4.1 Create a sample project in SonarQube Dashboard

1. Navigate to Project section and create project manually.

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)?
Create your project from your favorite DevOps platform.

First, you need to set up a DevOps platform configuration.

 Import from Azure DevOps Setup	 Import from Bitbucket Cloud Setup
 Import from GitHub Setup	 Import from GitLab Setup

Are you just testing or have an advanced use-case? Create a project manually.

[Create project manually](#)

2. Enter Project display name and project key then click on Next.

Create a project

Project display name *



Up to 255 characters. Some scanners might override the value you provide.

Project key *



The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

Main branch name *

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

[Next](#)

3. Select use the global setting option and click on Create Project.

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 follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

Choose the baseline for new code for this project

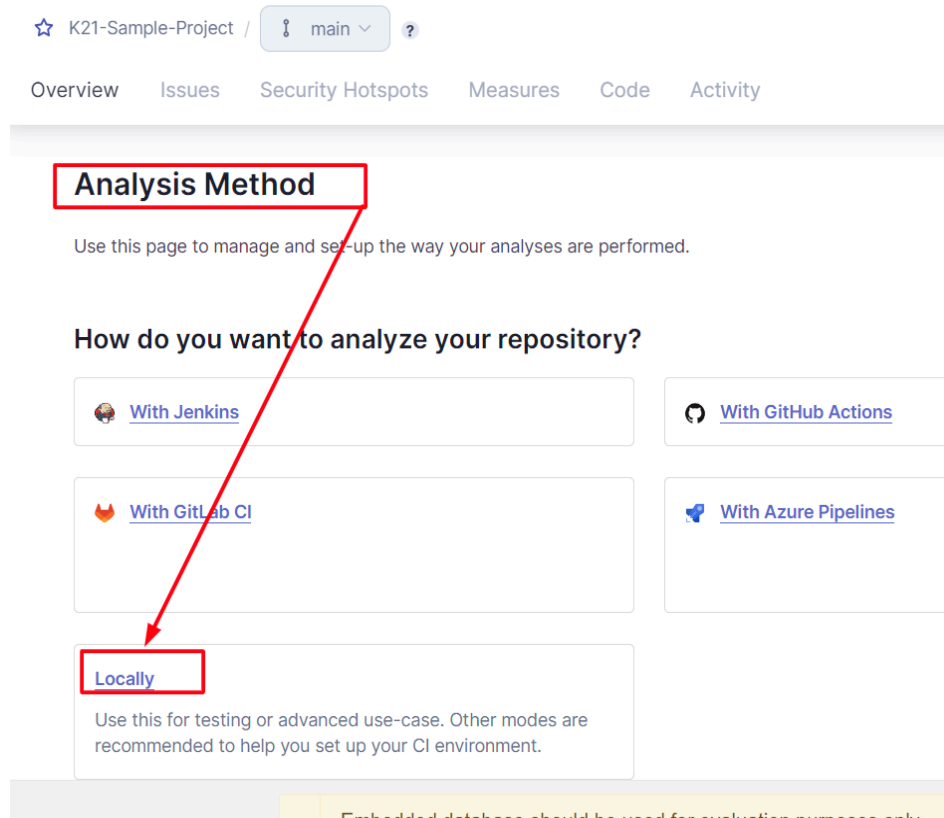
☒ Use the global setting

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

4. Now the project is created, the next step is to analyse the project source code.
5. In the analysis method section click on Locally option



6. Generate a project token with a unique name.
7. SonarQube **tokens are like special keys that allow other programs or tools to securely connect and use SonarQube's features.** Creating these tokens is like giving permission in a controlled way, ensuring that only the right programs or tools are allowed to interact with SonarQube. It's a way to keep everything safe and only let trusted things communicate with SonarQube.

☆ K21-Sample-Project / main ?

Overview Issues Security Hotspots Measures Code Activity

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

☒ Generate a project token

Token name ? Expires in

Analyze "K21-Sample-Project" 30 days **Generate**

! 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.

☐ Use existing token


The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point in time in your [user account](#).

8. Click on continue.

Analyze your project

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

Analyze "K21-Sample-Project": `sqp_32d065a14065aa8cd46334a1e285c0cf0ab50b8a` 

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point in time in your [user account](#).

Continue

9. Here, our project is based on PHP and using the Linux operating system, so please select the appropriate option.

1 Provide a token

Analyze "K21-Sample-Project": sqp_32d065a14065aa8cd46334a1e285c0cf0ab50b8a

2 Run analysis on your project

What option best describes your build?

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=K21-Sample-Project \
-Dsonar.sources=. \
-Dsonar.host.url=http://20.161.42.153:9000 \
-Dsonar.token=sqp_32d065a14065aa8cd46334a1e285c0cf0ab50b8a
```

Copy

In the image above, we got the Sonar-scanner command to scan our source code. To use it, we must install the SonarQube Scanner in the command line interface (CLI) since it's not currently installed on our system, causing an error when we try to run the command in the directory with our source code.

5 INSTALL SONARQUBE SCANNER & RUN A SONARQUBE SCAN

Installing SonarQube Scanner is crucial for performing static code analysis. After configuring the scanner, running a sonar scan helps identify bugs, code smells, and security issues. Analysing the results from the SonarQube dashboard allows developers to enhance code quality and ensure a robust development process.

1. Switch to scripts directory to install sonar-scanner.

```
$ ls  
$ cd scripts  
$ ls
```

```
root@docker:~/sonarqube# ls  
docker-compose.yml  php-project  scripts  
root@docker:~/sonarqube# cd scripts  
root@docker:~/sonarqube/scripts# ls  
install_sonar_scanner.sh
```

2. Please check whether the script is executable. If it is not, make it an executable file.

```
$ ls -l  
$ chmod 755 install_sonar_scanner.sh  
$ ls -l
```

```
root@docker:~/sonarqube/scripts# ls -l  
total 4  
-rw-r--r-- 1 root root 668 Nov  9 05:23 install_sonar_scanner.sh  
root@docker:~/sonarqube/scripts# chmod 755 install_sonar_scanner.sh  
root@docker:~/sonarqube/scripts# ls -l  
total 4  
-rwxr-xr-x 1 root root 668 Nov  9 05:23 install_sonar_scanner.sh
```

3. View the script.

```
$ cat install_sonar_scanner.sh
```

```
root@docker:~/sonarqube/scripts# cat install_sonar_scanner.sh
#!/bin/bash

cd /tmp || exit

echo "Downloading sonar-scanner....."
if [ -f "/tmp/sonar-scanner-cli-6.0.0.4432-linux.zip" ]; then
    sudo rm /tmp/sonar-scanner-cli-6.0.0.4432-linux.zip
fi

wget -q https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-6.0.0.4432-linux.zip
echo "Download completed."

echo "Unzipping downloaded file..."
unzip -qo sonar-scanner-cli-6.0.0.4432-linux.zip
echo "Unzip completed."
rm sonar-scanner-cli-6.0.0.4432-linux.zip

echo "Installing to /opt..."
if [ -d "/var/opt/sonar-scanner-6.0.0.4432-linux" ]; then
    sudo rm -rf /var/opt/sonar-scanner-6.0.0.4432-linux
fi
sudo mv sonar-scanner-6.0.0.4432-linux /var/opt

echo "alias sonar-scanner=/var/opt/sonar-scanner-6.0.0.4432-linux/bin/sonar-scanner" >> ~/.bashrc
exec bash
```

This Bash script automates the installation of the SonarQube Scanner on a Linux system. It downloads the SonarQube Scanner archive, extracts it, moves it to the /var/opt directory, and creates an alias in the user's .bashrc file for convenient use.

4. Here, we are using unzip to download the binaries. If unzip is not installed, please install it.

```
$ apt install unzip
```

```
root@docker:~/sonarqube/scripts# apt install unzip
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 168 kB of archives.
After this operation, 593 kB of additional disk space will be used.
Get:1 http://azure.archive.ubuntu.com/ubuntu focal-updates/main amd64 unzip amd64 6.
```

5. Run the script to install sonarqube scanner.

```
$ bash install_sonar_scanner.sh
```

```
root@docker:~/sonarqube/scripts# bash install_sonar_scanner.sh
Downloading sonar-scanner.....
Download completed.
Unzipping downloaded file...
Unzip completed.
Installing to /opt...
root@docker:/tmp#
```

6. Now, we need to run a scan inside our project folder.
7. Switch to the php-project folder and execute the sonar-scanner command that we obtained from the SonarQube dashboard.

```
$ cd ~/sonarqube/php-project
```

```
$ ls
```

```
sonar-scanner \
```

```
-Dsonar.projectKey=K21-Sample-Project \
```

```
-Dsonar.sources=. \
```

```
-Dsonar.host.url=http://20.161.42.153:9000 \
```

```
-Dsonar.token=sqp_32d065a14065aa8cd46334a1e285c0cf0ab50b8a
```

In my case, the sonar-scanner command is mentioned above. Please replace it with the command that you obtained in Section 4.1.

```
root@docker:/tmp# cd ~/sonarqube/php-project
root@docker:~/sonarqube/php-project# ls
LICENSE  backend  cssjs    db        images   index.php  register.php  scss
about.php  css      database  fonts     includes js          screenshots
root@docker:~/sonarqube/php-project#
```

The screenshot shows the SonarQube dashboard for a project named 'K21-Sample-Project'. The 'Run analysis on your project' section is active, showing options for build tools (Maven, Gradle, .NET, Other) and operating systems (Linux, Windows, macOS). A red box highlights the 'Execute the Scanner' section, which contains the following command:

```
sonar-scanner \
-Dsonar.projectKey=K21-Sample-Project \
-Dsonar.sources=. \
-Dsonar.host.url=http://20.84.125.170:9000 \
-Dsonar.token=sqp_bc21d4a81fdee1f9674b098a65bebe1fd8a6710
```

A red arrow points from this command to a terminal window. The terminal window shows the execution of the command, with the following output:

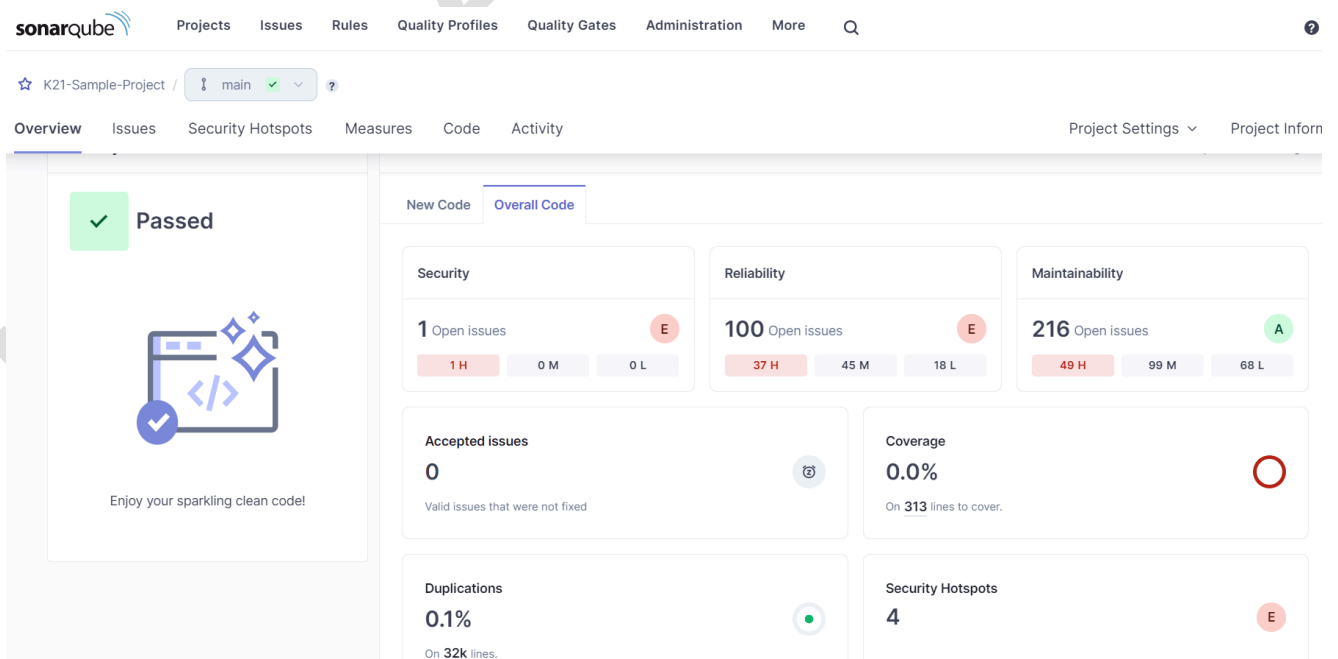
```
root@docker:/tmp# cd ~/sonarqube/php-project
root@docker:~/sonarqube/php-project# ls
LICENSE  backend  cssjs    db        images   index.php  register.php  scss
about.php  css      database  fonts     includes js          screenshots
root@docker:~/sonarqube/php-project#
root@docker:~/sonarqube/php-project# cd /tmp
root@docker:/tmp#
root@docker:/tmp# cd ~/sonarqube/php-project
root@docker:~/sonarqube/php-project# ls
LICENSE  backend  cssjs    db        images   index.php  register.php  scss
about.php  css      database  fonts     includes js          screenshots
root@docker:~/sonarqube/php-project#
root@docker:~/sonarqube/php-project#
root@docker:~/sonarqube/php-project#
root@docker:~/sonarqube/php-project# sonar-scanner \
-Dsonar.projectKey=K21-Sample-Project \
-Dsonar.sources=. \
-Dsonar.host.url=http://20.84.125.170:9000 \
-Dsonar.token=sqp_bc21d4a81fdee1f9674b098a65bebe1fd8a6710
```

```

09:26:25.982 INFO ----- Run sensors on project
09:26:26.021 INFO Sensor Zero Coverage Sensor
09:26:26.036 INFO Sensor Zero Coverage Sensor (done) | time=15ms
09:26:26.038 INFO SCM Publisher SCM provider for this project is: gi
t
09:26:26.041 INFO SCM Publisher 113 source files to be analyzed
09:26:27.747 INFO SCM Publisher 113/113 source files have been analy
zed (done) | time=1706ms
09:26:27.765 INFO CPD Executor 8 files had no CPD blocks
09:26:27.765 INFO CPD Executor Calculating CPD for 9 files
09:26:27.790 INFO CPD Executor CPD calculation finished (done) | tim
e=24ms
09:26:27.797 INFO SCM revision ID '195cd380ca200904ef060f9f4b91ec6dd
adf0cb5'
09:26:27.982 INFO Analysis report generated in 165ms, dir size=1.9 M
B
09:26:28.222 INFO Analysis report compressed in 239ms, zip size=548.
3 kB
09:26:28.321 INFO Analysis report uploaded in 99ms
09:26:28.322 INFO ANALYSIS SUCCESSFUL, you can find the results at:
http://20.84.125.170:9000/dashboard?id=K21-Sample-Project
09:26:28.322 INFO Note that you will be able to access the updated d
ashboard once the server has processed the submitted analysis report
09:26:28.322 INFO More about the report processing at http://20.84.1
25.170:9000/api/ce/task?id=ed001cfb-bbel-4404-98c7-e935f8e3f98e
09:26:29.031 INFO Analysis total time: 37.028 s
09:26:29.033 INFO EXECUTION SUCCESS
09:26:29.034 INFO Total time: 40.056s
root@docker:~/sonarqube/php-project#

```

8. Navigate to SonarQube Dashboard you will get the analysis of your code.




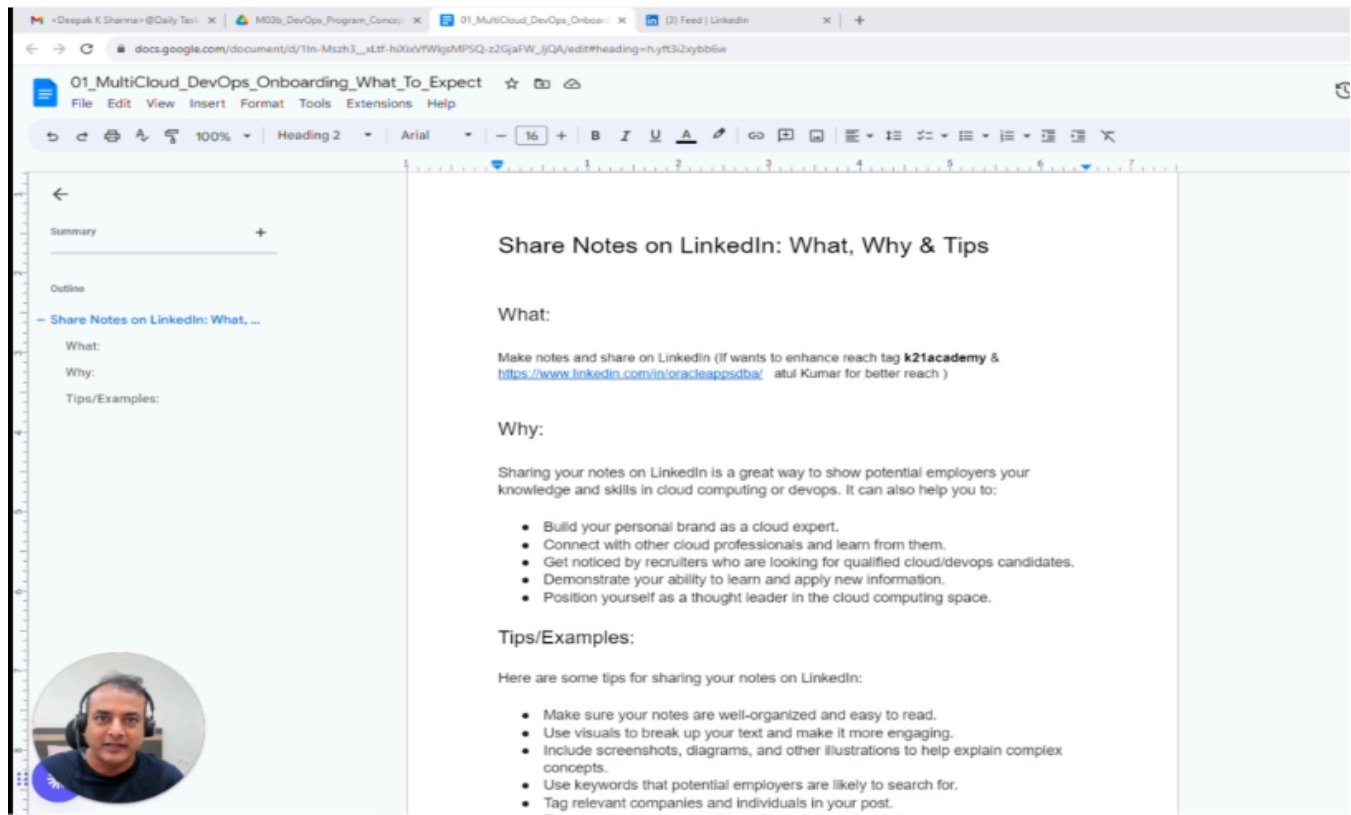
NOTE: The page has metrics such as Security, Reliability, Maintainability, Accepted Issues, Coverage, Duplications and Security Hotspots. The following table briefly explains each of these terms.


Security	Potential vulnerabilities in the code that could be exploited by attackers.
Reliability	Bugs in the code that could cause it to behave incorrectly.
Maintainability	Code smells or structural problems that make the code harder to maintain and extend.
Accepted Issues	Issues identified but accepted as they are not being fixed immediately.
Coverage	The percentage of code covered by automated tests.
Duplications	The percentage of code that is duplicated, making maintenance more difficult and error-prone.
Security Hotspots	Areas in the code that require manual review to ensure they are not security vulnerabilities.

6 SHARE YOUR LEARNINGS


In this section, you are going to share whatever you learned during this lab on LinkedIn and WhatsApp.

 <https://www.loom.com/share/d91f9abbec794f4f85e7d99bb82c2ff6>



 As you know, building a strong **professional profile** is essential in today's job market. One way to showcase your skills and knowledge is by sharing your labs and projects on LinkedIn.

Your **LinkedIn profile** is a powerful tool in your job search. Sharing your labs and projects is just one way to demonstrate your expertise and stand out to potential employers. If you don't have LinkedIn we strongly recommend you create one for yourself.

 By doing so, you can demonstrate to potential employers or connections that you have **hands-on experience** in your field and are actively engaged in learning and growing your skills.

6.1 LinkedIn

Take a screenshot and share it on your LinkedIn. So, this will attract recruiters and employers to your profile and increase your reach. Do remember to tag

- K21Academy (<https://www.linkedin.com/company/k21academy>) &
- Atul Kumar (<https://www.linkedin.com/in/atulk21academy/>) as we will circulate in our network too to increase your reach.

Here is a sample which includes steps inside a post followed by a screenshot.

K21Academy



Adelakun Joshua • 3rd+

+ Follow ...

1d • Edited •

AWS CLOUDWATCH BILLING ALARM

Problem:

Today, one of my clients reached out to me. They have AWS billing challenges. They just got an email alert from AWS stating that, they have to pay certain amount of bills. When they evaluated their end of the month service bills, they discovered that they are billed for a lot of services that they are actually not using on a day-to-day basis. But unfortunately, they didn't properly monitor and manage their AWS resources, which incur charges.

My recommendation:

I recommended AWS Cloudwatch alarm to my client. And I configured the cloudwatch alarm for them.

AWS Cloudwatch:

Amazon CloudWatch is a monitoring and management service that provides data and actionable insights for AWS, on-premises, hybrid, and other cloud applications and infrastructure resources.

How I Set the Cloudwatch Billing Alarm Up:

1. I Opened the CloudWatch console at <https://lnkd.in/gugv5tEE> ✓.
 2. In the navigation pane, I choose Alarms, and then choose All alarms. I Chooosed Create alarm.
 3. Then, I Selected 'metric'. In Browse, I choose 'Billing', and then 'Total Estimated Charge'.
 4. I Selected the box for the 'EstimatedCharges metric', and then I choose 'Select metric'.
 5. For 'Statistic', I choose Maximum.
 6. For 'Period', I choose 6 hours.
 7. For 'Threshold type', I choose Static.
 8. For 'Whenever EstimatedCharges is' . . . , I selected Greater.
- For 'than' . . . , I defined the value that I want to cause the alarm to trigger. For example, \$10 USD.

SNS topic included the email address by which my client will receive email when the billing amount crosses the billing threshold specified.

Note:

You can select an existing Amazon SNS topic, create a 'new Amazon SNS topic', or use a topic ARN to notify other account. If you want your alarm to send multiple notifications for the same alarm state or for different alarm states, choose 'Add notification'.

11. I choosed Next.

12. Under Name and description, I entered a name for the alarm. (Optional) Enter a description of the alarm.

13. Then I Choosed 'Next'.

14. Under 'Preview and create', after confirming that the configuration is correct, and then i choosed 'Create alarm'.

Conclusion: Organization that properly set up a cloudwatch billing alarm, will be able to properly monitor their aws resources usage. The metrics provided by the cloudwatch alarm through the dashboard will also help the organization to effectively control their spending.

#aws

#awscloud

#JTechconsult

#K21Academy: Learn Cloud From Experts

#AtulKumar

#SumtiMehta

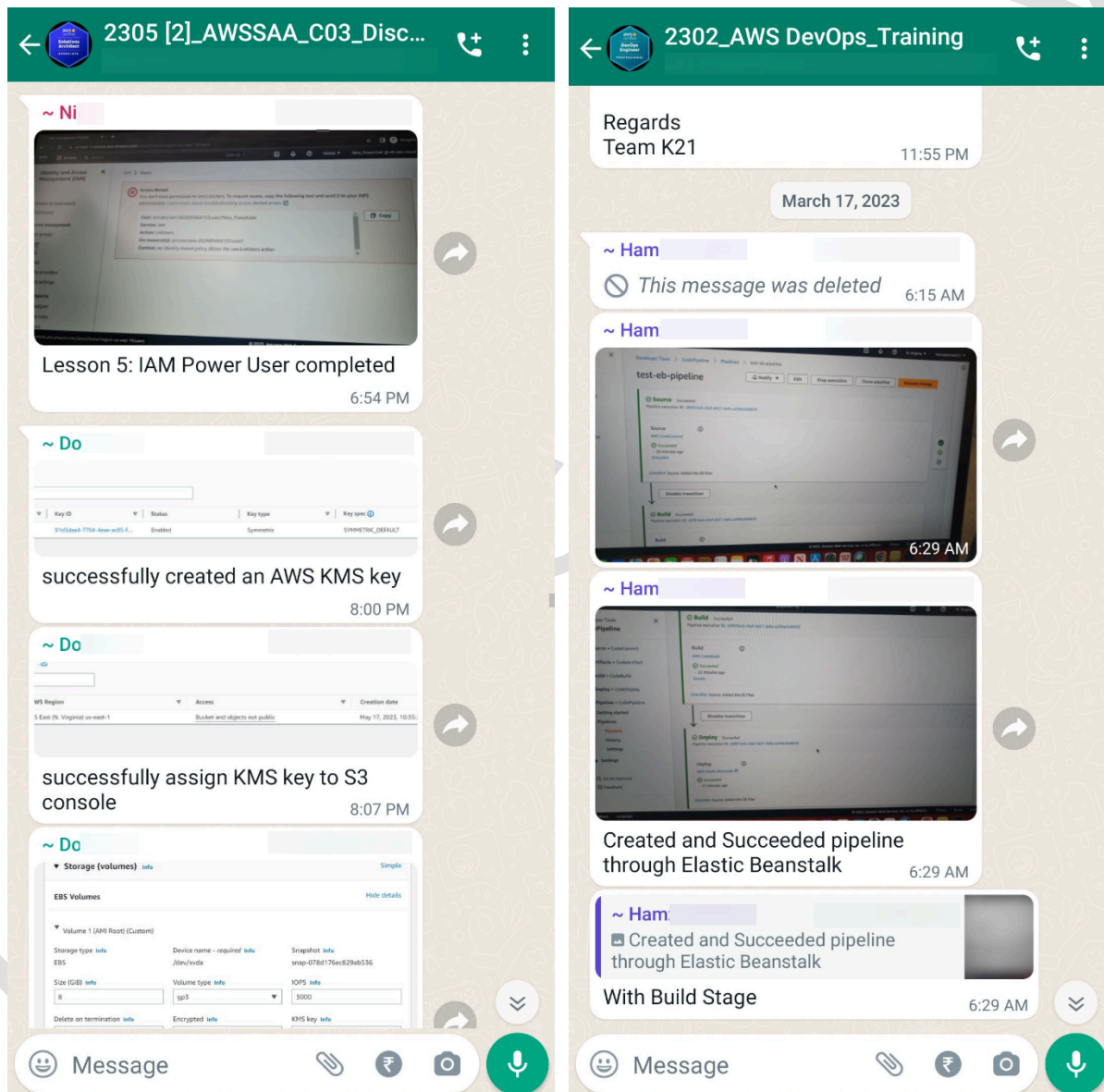
Cloudwatch billing alarm architectural diagram:



6.2 WhatsApp

Share screenshot in WhatsApp group, this will tell us that you are making progress, make yourself accountable, and inspire others (including yourself) to work on other remaining hands-on labs.

Now go and share your screenshot in WhatsApp group please. **Three lucky winner every quarter will get prizes worth 100 / 50 / 25 USD each**



7 CLEAN UP RESOURCES

1. Remove Sonarqube & SonarQube database Containers.

```
$ docker rm -f sonarqube sonarqube-database
```

```
$ docker rmi -f sonarqube postgres:latest
```

```
root@docker:~/sonarqube/php-project# docker rm -f sonarqube sonarqube-database
sonarqube
sonarqube-database
root@docker:~/sonarqube/php-project# docker rmi -f sonarqube postgres:12
Untagged: sonarqube:latest
Untagged: sonarqube@sha256:8ef1cabefb065c6f72c0191eb5a8dc6ca4dc95723244ed532cfdbb57a1d2f8f7
Deleted: sha256:0d9346034781f773e66518d525beb98ed4e5e339315a689ba03fc4b24ee345b8
Deleted: sha256:c4d5db241ca70ab1b1a9f8080ccfb997aa65429a5bee34ea26a80791a9e2a005
Deleted: sha256:05de3d9055b73e8f01fd1205e3aa6e64ccd0f6706b2b2b68cf681a4ddc856ab9
Deleted: sha256:882c3a869d77fa43b30a20a3fe830962b5ea816ca8321127caf5ce14034e7836
Deleted: sha256:a0da28140ab9b8eeaf354a1dba85b9e81fa4014384c94d2866f71386b3d3cbc2
Deleted: sha256:2facda8a051bf00337223fb53544f229af20e23d1ea9f88ceee8c097bcb8f0fb
Deleted: sha256:6ddb97c04586452e94e051121cab7578969a3f550663eaae8abfaa556cdb78c
Deleted: sha256:256d88da41857db513b95b50ba9a9b28491b58c954e25477d5dad8abb465430b
Untagged: postgres:12
Untagged: postgres@sha256:974bda3757f2968440efdb77b621621d4782a5b608cddca472284117d6ded13a
Deleted: sha256:d480c196f9b02e6b5fe3a9cfd6aa0ed8a17545c8eb6c90235409ea3945d9a2b3
Deleted: sha256:6c324f971f05d18d52706f7da94b6178337b4d167e19b1bb1c9269b7e58c6405
Deleted: sha256:1e4972f5b50d10b8bfa685d474748a67c7036d68fc308c53d1a8420c7b01afd4
```

8 TROUBLESHOOTING

<All the errors along with their fixes are to be added here>

K21Academy

9 SUMMARY

In this guide we Covered:

- Install and setup Sonarqube Dashboard
- Create a sample project in SonarQube Dashboard
- Install Sonarqube Scanner & Run a Sonarqube Scan

K21Academy