How to install and configure SonarQube on AWS EC2 Ubuntu 22.04 and 20.04 (Full Setup)?

DeshDeepakDhobi (DD) · Follow
9 min read · May 28

Humans do make mistakes and when it comes to mistakes in the codes it is too much to manage and handle and there comes SonarQube to rescue.

SonarQube is a code quality assurance tool used for continuous inspection of code quality to perform automatic reviews with analysis of code to detect bugs and code smells on multiple programming languages and generate an analysis report to ensure code reliability.

Let's start the installation of the latest version (10.0) of SonarQube on Ubuntu 22.04 LTS.

> *Note: The same steps also work on the AWS EC2 Ubuntu 20.04.*
>
> *Server Specification (Please choose as per your requirement):*

OS = Ubuntu 22.04 LTS

CPU: 2 vCPU

RAM: 4 GB

Storage: 20 GB

> *And view the* ***following documentation for the Prerequisites and Hardware recommendations.***

Prerequisites and overview

The prerequisites for installing SonarQube.

docs.sonarqube.org

> *Security Ports to open for the AWS EC2 server:*

i) HTTP = 80

ii) HTTPS = 443

iii) SSH = 22 (open to only required IP)

iv) Custom port = 9000

Type	Protocol	Port range	Source
HTTPS	TCP	443	0.0.0.0/0
Custom TCP	TCP	9000	0.0.0.0/0
HTTP	TCP	80	0.0.0.0/0
SSH	TCP	22	27.34.96.72/32

Let's Start Intsalling and configuring things.

Let's start installing things.

1) SSH into the AWS EC2 Ubuntu Server

```
ssh -i [keyname.pem] [OSname]@[PublicIP_of_EC2Server]
```

For example (for me its):

ssh -i helloworld.pem ubuntu@55.63.86.254

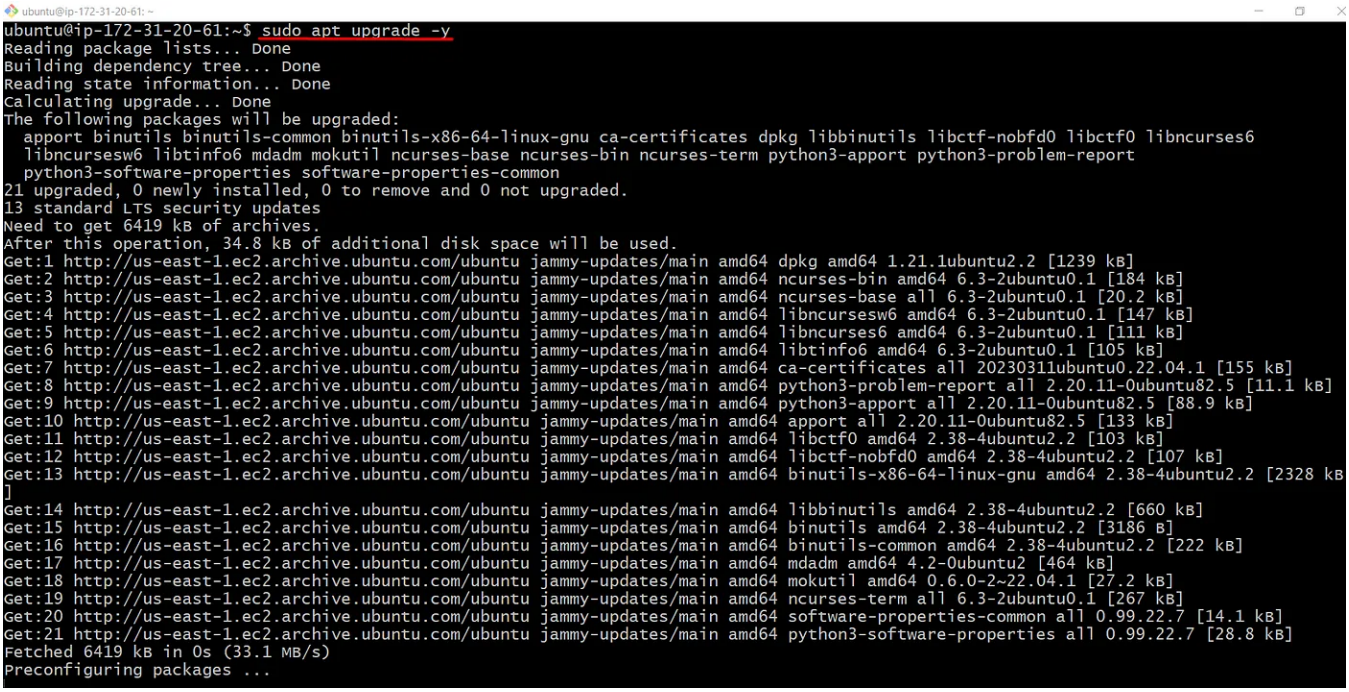
2) Let's rejuvenate our Ubuntu server

```
sudo apt update
```

```
ubuntu@ip-172-31-20-61:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [108 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
0% [5 Packages 14.1 MB/14.1 MB 100%]
```

Update the server

```
sudo apt upgrade -y
```

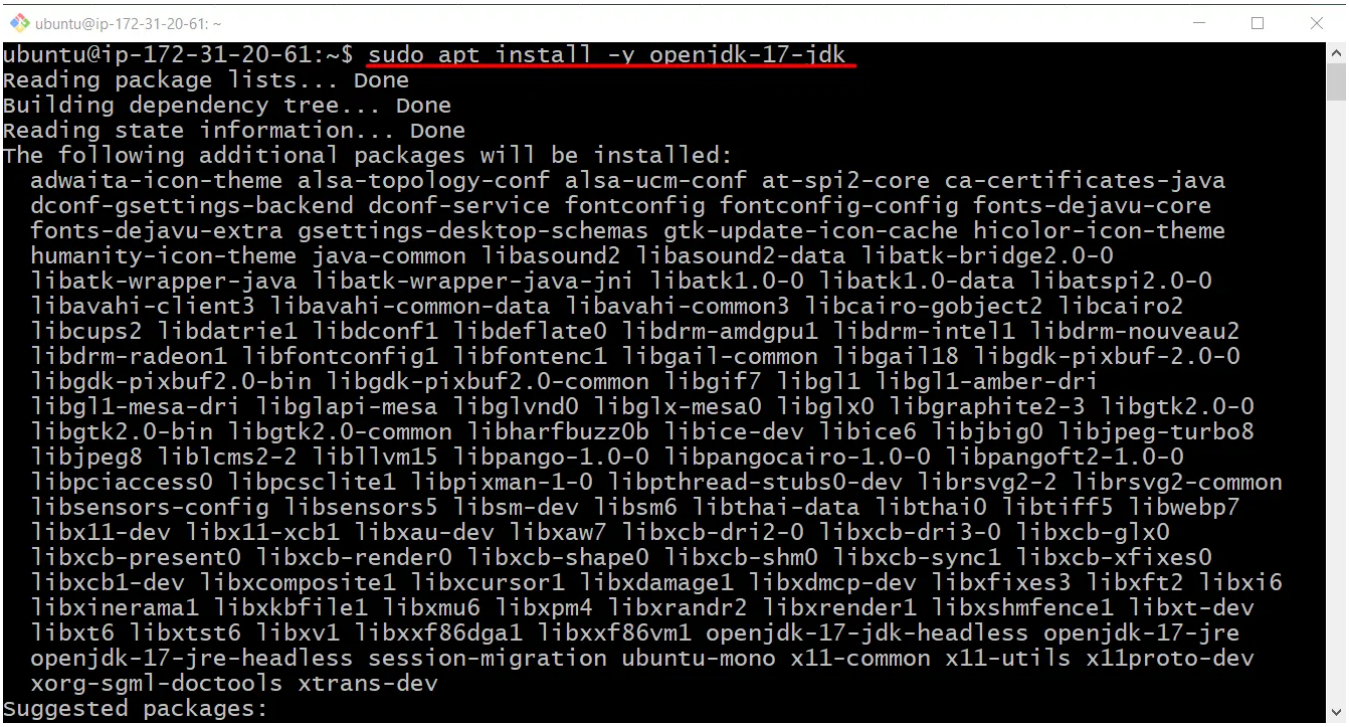


Upgrade the server

3) Install OpenJDK 17

i) Install OpenJDK 17 (needed for the latest version of SonarQube (version 10.0).

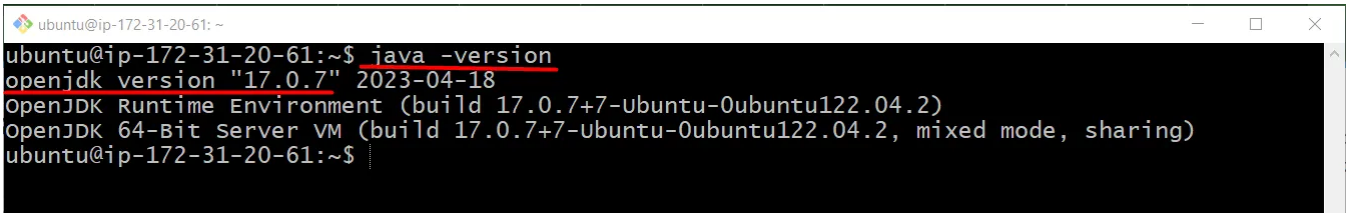
```
sudo apt install -y openjdk-17-jdk
```



Installing JDK 17

ii) Let's check the installed version of Java. **VALIDATION IS IMPORTANT.**

```
java -version
```

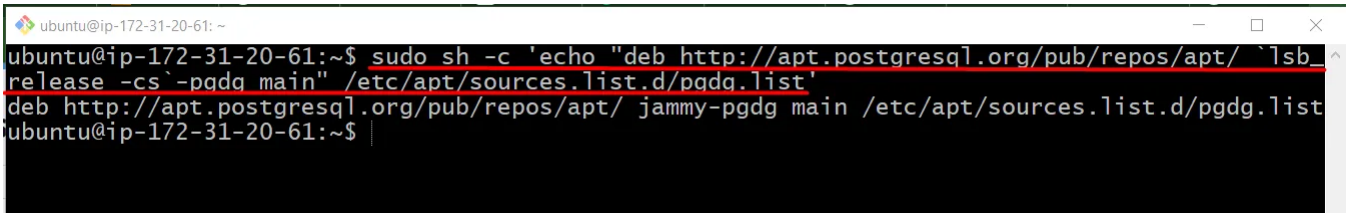


Checking the Java version

4) Install and Configure PostgreSQL

i) Add the PostgreSQL repository.

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/
`lsb_release -cs`-pgdg main" /etc/apt/sources.list.d/pgdg.list'
```

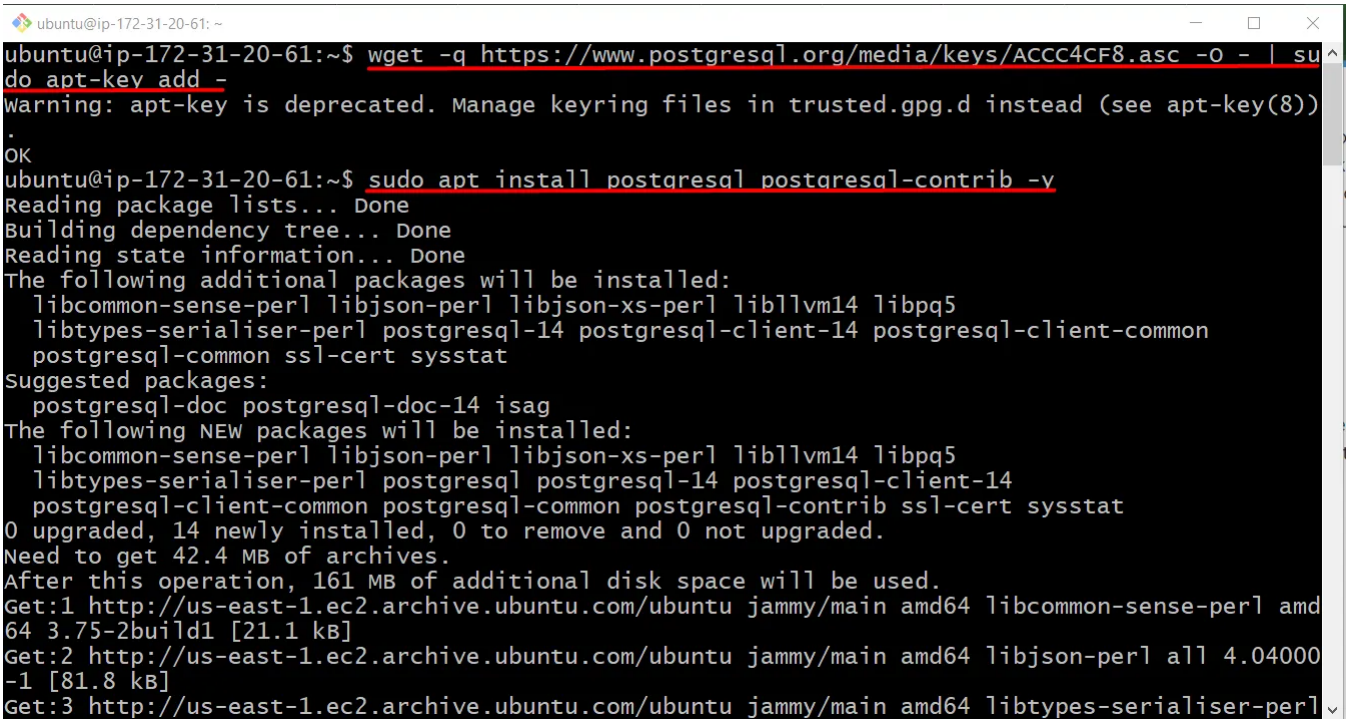


ii) Add the PostgreSQL signing key.

```
wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo
apt-key add -
```

iii) Install PostgreSQL.

```
sudo apt install postgresql postgresql-contrib -y
```



Installing PostgreSQL and its dependencies

iv) Enable the database server to start automatically on reboot.

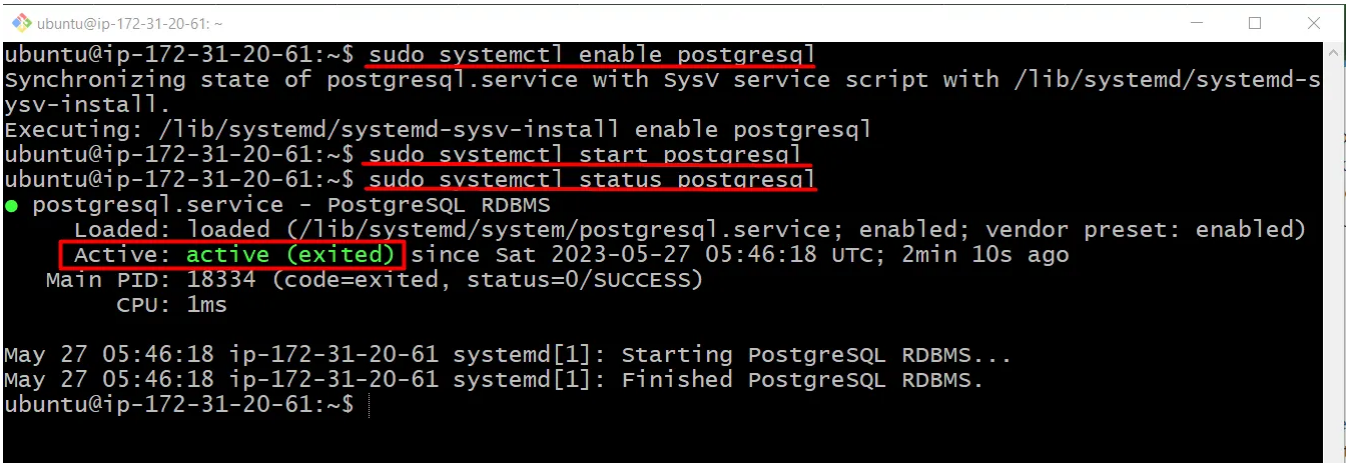
```
sudo systemctl enable postgresql
```

v) Start the database server.

```
sudo systemctl start postgresql
```

vi) Check the status of the database server

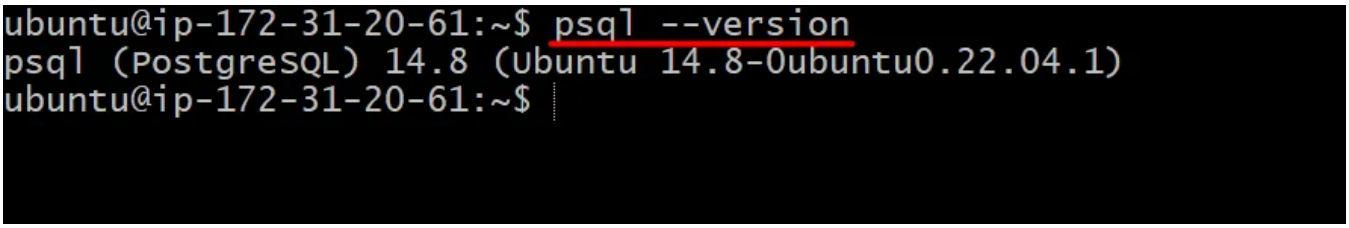
```
sudo systemctl status postgresql
```



Postgresql status check

vii) Let's check the installed version of the install Postgres DB. **VALIDATION IS IMPORTANT.**

```
psql --version
```



Postgresql version check

viii) Switch to the Postgres user.

```
sudo -i -u postgres
```

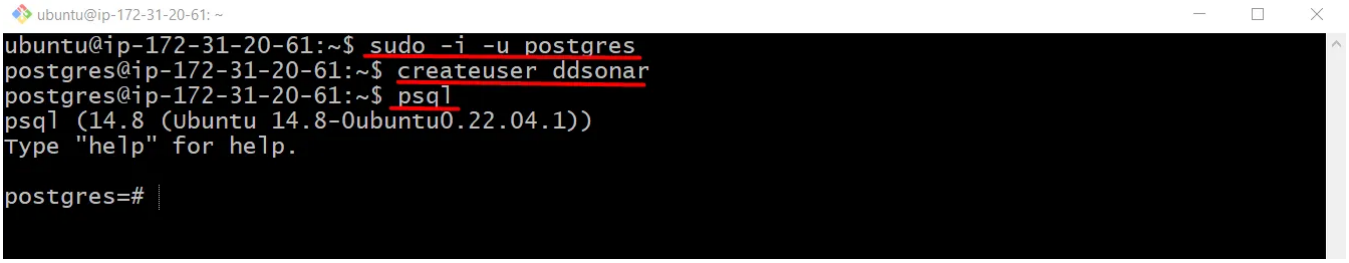
ix) Create a database user named **ddsonar.**

> ***Note:*** *You can provide the name of your own but make a note of it, as we will be needing this name in further steps.*

```
createuser ddsonar
```

x) Log in to PostgreSQL.

```
psql
```



SSH into the Postgresql

xi) Set a password for the **ddsonar** user. Use a strong password in place of *my_strong_password*.

```
ALTER USER [Created_user_name] WITH ENCRYPTED password 'my_strong_password';
```

For example (In my case it will be):

```
ALTER USER ddsonar WITH ENCRYPTED password 'mwd#2%#!!#%rgs';
```

xii) Create a SonarQube database and set the owner to ddsonar.

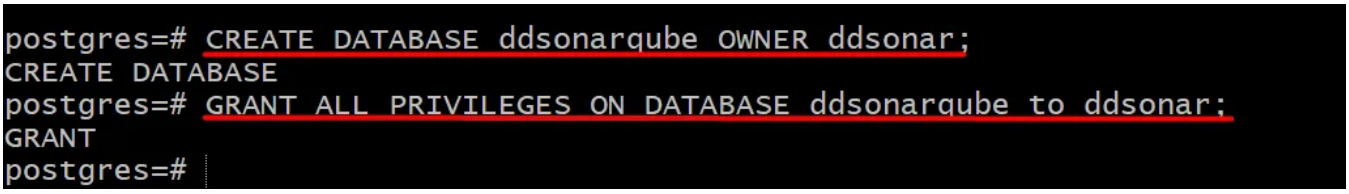
```
CREATE DATABASE [database_name] OWNER [Created_user_name];
```

For example (In our case it will be) — feel free to give an awesome database name as per your requirement:

```
CREATE DATABASE ddsonarqube OWNER ddsonar;
```

xiii) Grant all the privileges on the **ddsonarqube** database to the **ddsonar** user.

```
GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
```

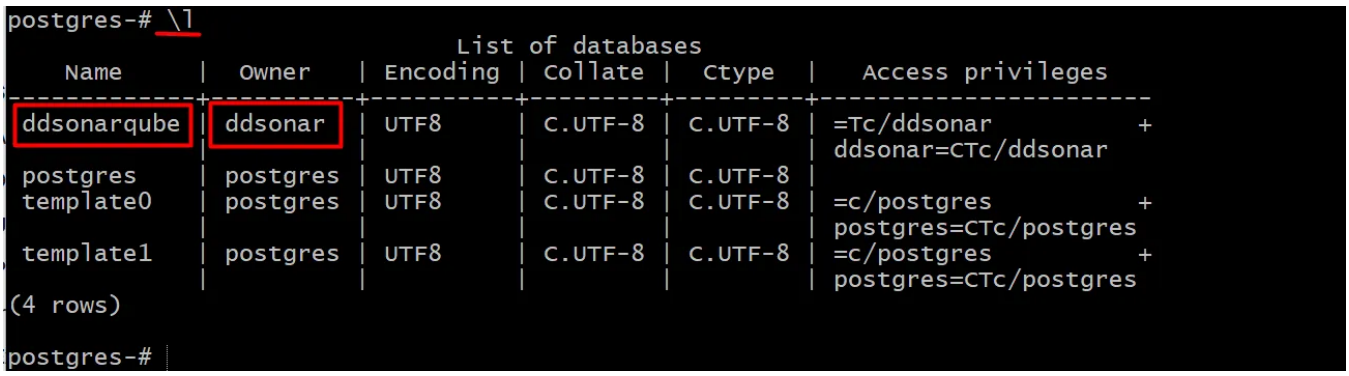


Create a database and grant it ALL PRIVILEGES.

xiv) Let's check the created user and the database.

a) To check the created database

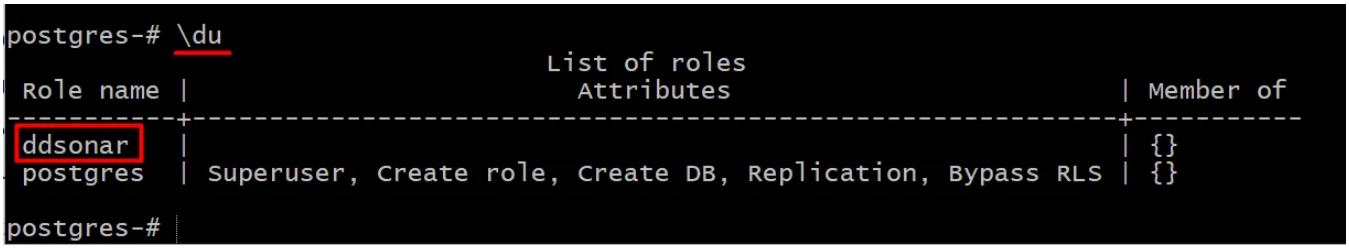
```
\l
```



Checking the created Database name

b) To check the created database user

```
\du
```



checking the created database user

xv) Exit PostgreSQL.

```
\q
```

xvi) Return to your non-root sudo user account.

```
exit
```

Returning to the non-root user

5) Download and Install SonarQube

i) Install the zip utility, which is needed to unzip the SonarQube files.

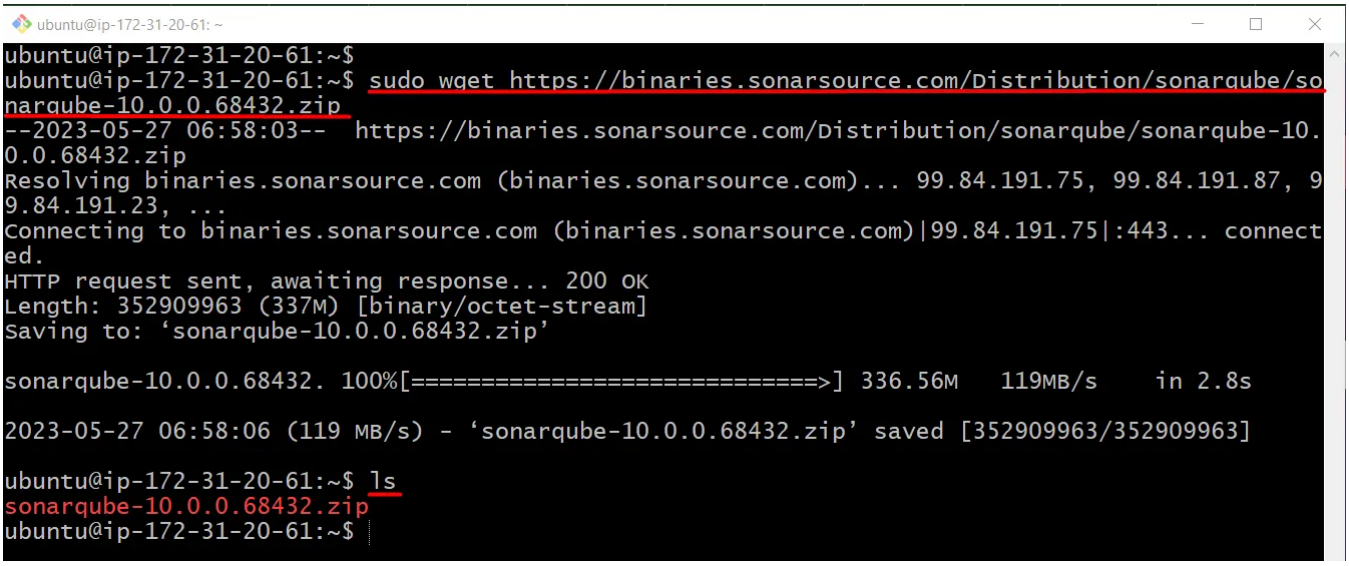
```
sudo apt install zip -y
```

ii) Locate the latest download URL from the SonarQube official download page.

> *Download the SonarQube distribution files. (you can download the latest SonarQube distribution using the following link)*
>
> *https://www.sonarsource.com/products/sonarqube/downloads/*

Here we are installing the latest version of SonarQube 10.0 community edition (free one)

```
sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.
```



Downloading the latest version (10.0) of SonarQube distribution files.

iii) Unzip the downloaded file.

```
sudo unzip sonarqube-10.0.0.68432.zip
```

iv) Move the unzipped files to **/opt/sonarqube** directory

```
sudo mv sonarqube-10.0.0.68432 sonarqube
```

```
sudo mv sonarqube /opt/
```

6) Add SonarQube Group and User

Create a dedicated user and group for SonarQube, which can not run as the root user.

> ***Note:*** *You can give any name for the sonar user and group. I have here given the user and group name to be the same i.e* ***ddsonar****.*

i) Create a sonar group.

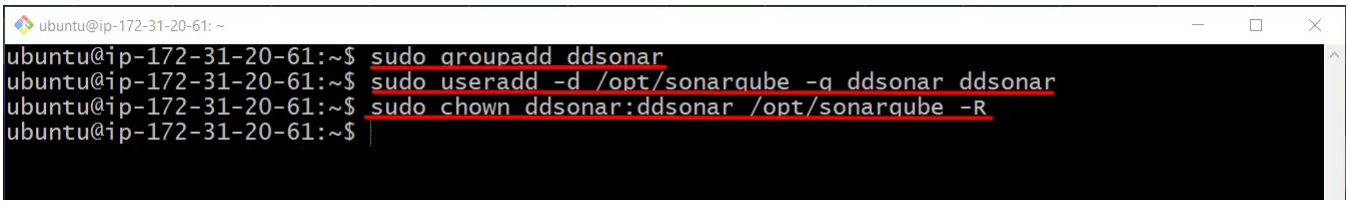
```
sudo groupadd ddsonar
```

ii) Create a sonar user and set */opt/sonarqube* as the home directory.

```
sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
```

iii) Grant the sonar user access to the */opt/sonarqube* directory.

```
sudo chown ddsonar:ddsonar /opt/sonarqube -R
```



Creating sonar user and group

7) Configure SonarQube

i) Edit the SonarQube configuration file.

```
sudo nano /opt/sonarqube/conf/sonar.properties
```

a) Find the following lines:

#sonar.jdbc.username=

#sonar.jdbc.password=

b) Uncomment the lines, and add the database user and Database password you created in *Step 4 (xi and xii)*. For me, it's:

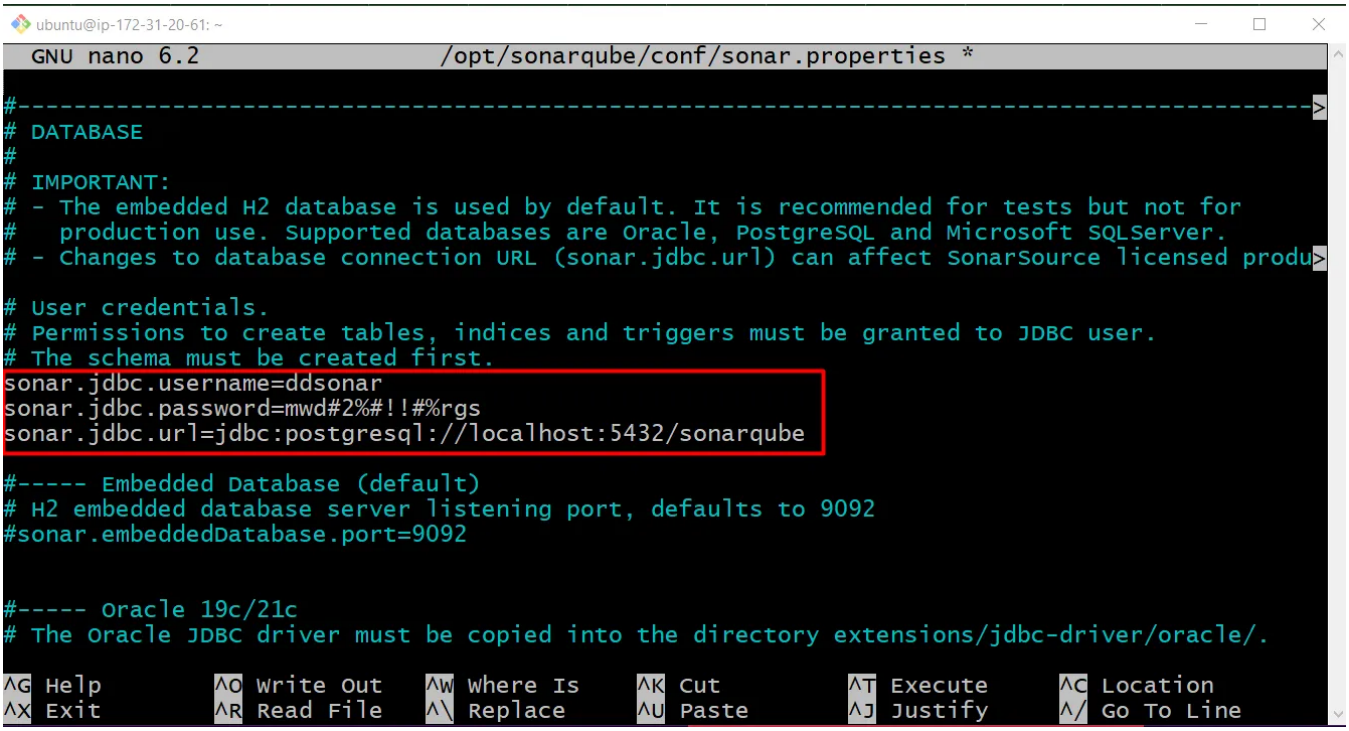
sonar.jdbc.username=**ddsonar**

sonar.jdbc.password=**mwd#2%#!!#%rgs**

c) Below these two lines, add the following line of code.

sonar.jdbc.url=jdbc:postgresql://localhost:5432/ddsonarqube

> *Here, **ddsonarqube** is the database name created.*



Adding the required line of details in the file sonar.properties.

d) Save and exit the file.

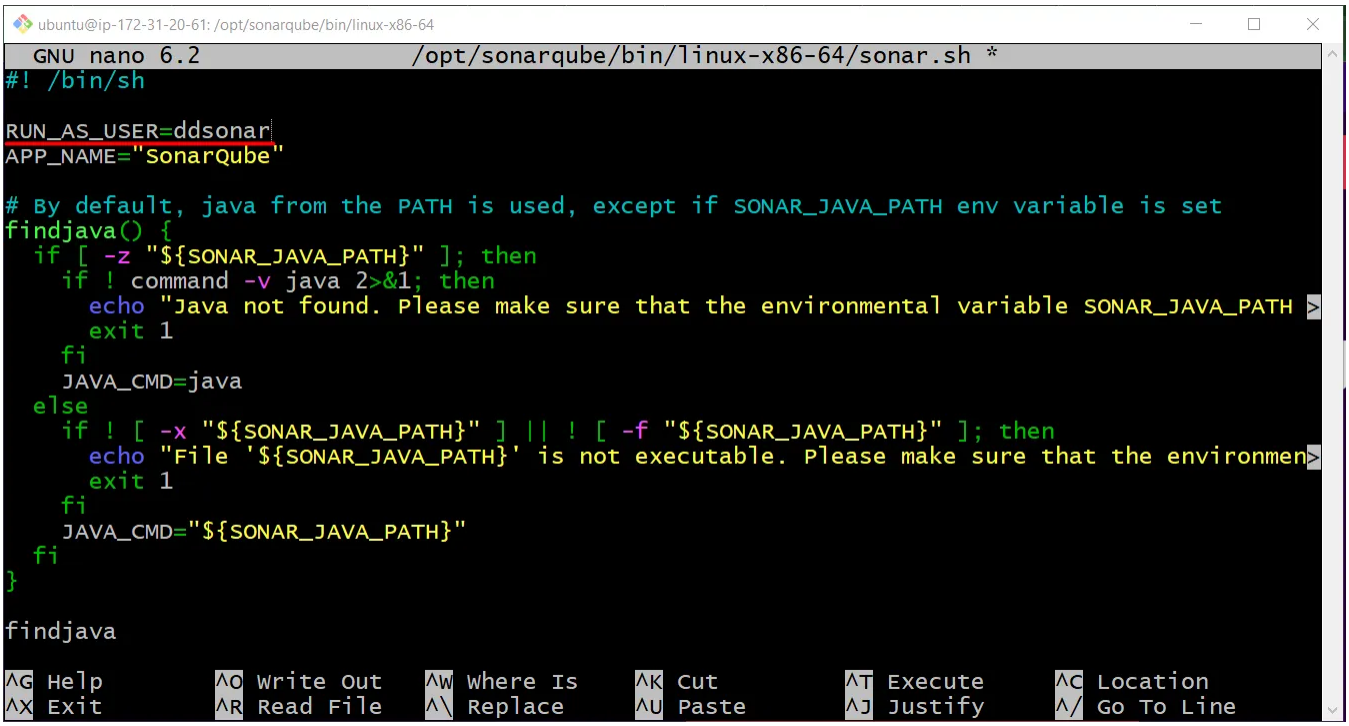
ii) Edit the sonar script file.

```
sudo nano /opt/sonarqube/bin/linux-x86-64/sonar.sh
```

a) Add the following line

RUN_AS_USER=ddsonar

> *Here, **ddsonar** is the name of the user that we have created in step number 6 (ii).*



Adding the RUN_AS_USER=ddsonar in the file sonar.sh

b) Save and exit the file.

8) Setup Systemd service

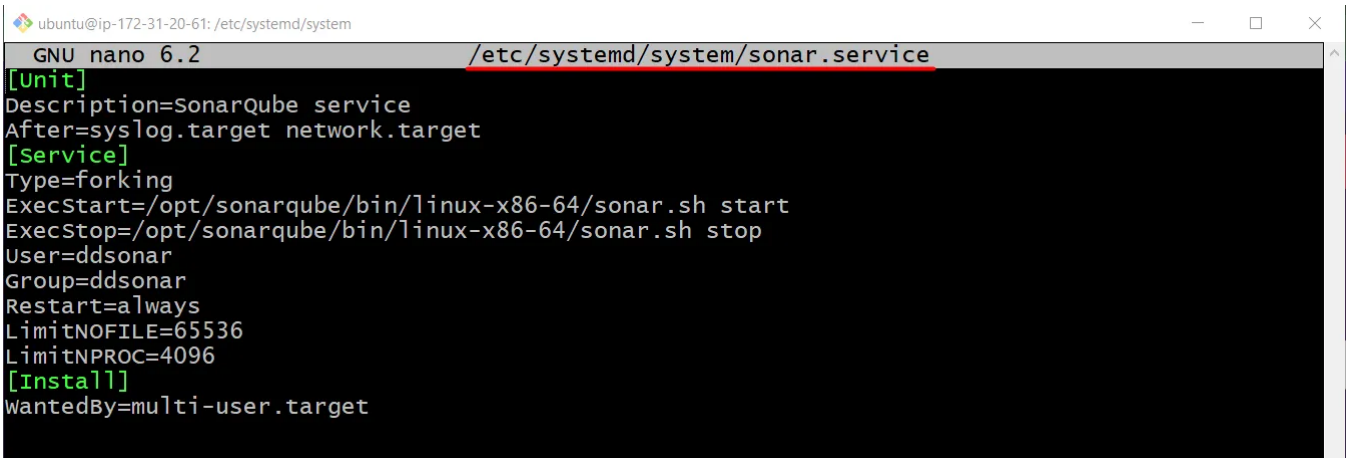
i) Create a **systemd** service file to start SonarQube at system boot.

```
sudo nano /etc/systemd/system/sonar.service
```

ii) Paste the following lines to the file.

```
[Unit]
Description=SonarQube service
After=syslog.target network.target
[Service]
Type=forking
ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop
User=ddsonar
Group=ddsonar
Restart=always
LimitNOFILE=65536
LimitNPROC=4096
[Install]
WantedBy=multi-user.target
```

> ***Note:*** *Here in the above script, make sure to change the* ***User*** *and* ***Group*** *section with the value that you have created.* ***For me its:***
>
> *User=**ddsonar***
>
> *Group=**ddsonar***



Adding the lines of script to the file sonar.service

iii) Save and exit the file.

iv) Enable the SonarQube service to run at system startup.

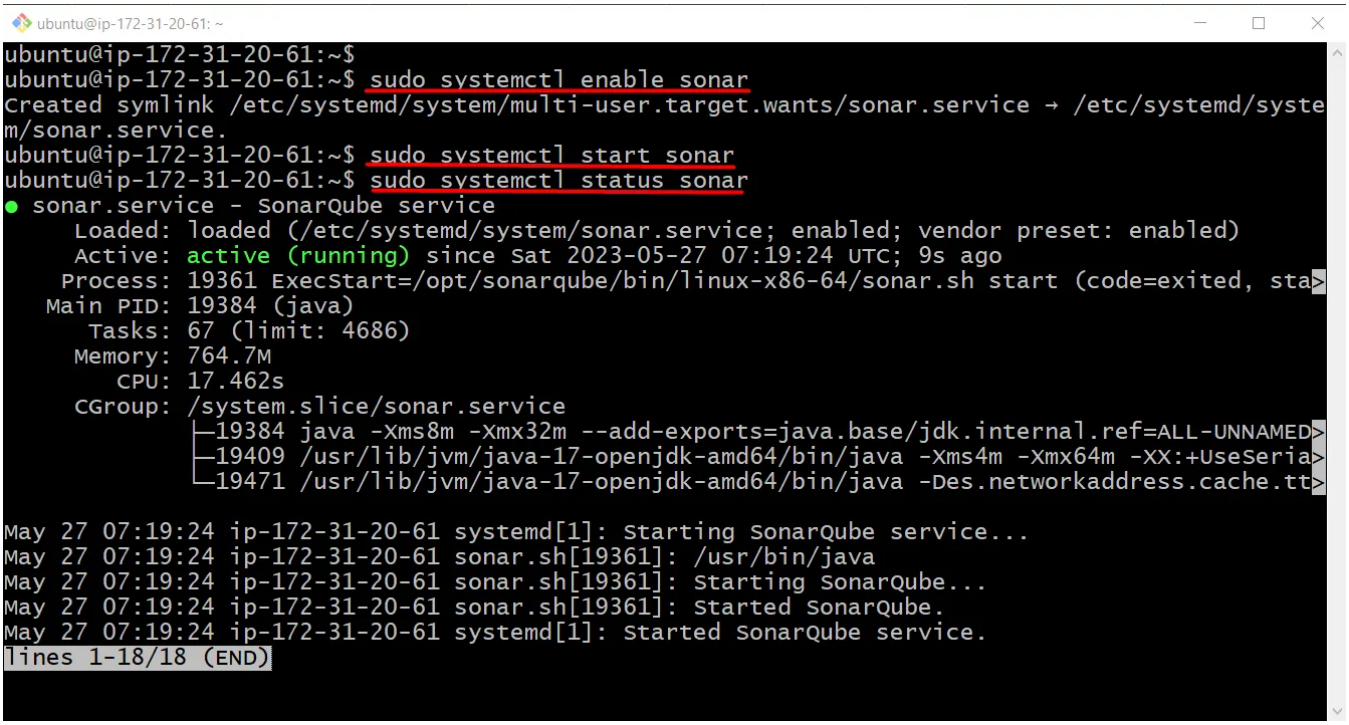
```
sudo systemctl enable sonar
```

v) Start the SonarQube service.

```
sudo systemctl start sonar
```

vi) Check the service status.

```
sudo systemctl status sonar
```



Starting and checking the status of the SonarQube

vii) Hurray, It's up and running.

9) Modify Kernel System Limits

SonarQube uses Elasticsearch to store its indices in an MMap FS directory. It requires some changes to the system defaults.

i) Edit the **sysctl** configuration file.

```
sudo nano /etc/sysctl.conf
```

ii) Add the following lines.

```
vm.max_map_count=262144
fs.file-max=65536
ulimit -n 65536
ulimit -u 4096
```

Adding the lines of codes at the file /etc/sysctl.conf

iii) Save and exit the file.

iv) Reboot the system to apply the changes.

```
sudo reboot
```

10) Access SonarQube Web Interface

i) Access SonarQube in a web browser at your server's IP address on port 9000.

For example, http://IP:9000

> ***Note:*** *the default username and password are* ***admin*** *and* ***admin*** *respectively.*

Fresh installed SonarQube GUI

ii) Change the Old password with a New one.

Log in with username **admin** and password **admin.** In the next step, SonarQube will prompt you to change your password. **CHANGE THE PASSWORD.**

Change the old password.

iii) And yes, we have successfully installed the SonarQube Community version 10.0 on AWS EC2 Ubuntu 22.

We are inside the SonarQube. Let's get started.

Congratulations, Our SonarQube has been installed successfully.

Remaining Extra Work.

Configuring the Reverse Proxy using Nginx

Now we want a domain name to take over the IP of the server so that instead of hitting the IP we can use the domain name that is TLS enabled. Let's see how to do so.

> *I will be using the sub-domain name* ***sonarqube.onecloudhelper.com.*** *This is mine. Please use yours.*

1) Install Nginx

```
sudo apt install nginx -y
```

Installing Nginx

2) Adding the DNS records.

My domain is hosted on the AWS Route53, thus I will be adding the records in there.

Adding the IP of the server on the DNS zone.

And now our domain **sonarqube.onecloudhelper.com** is ready to be used for the nginx configuration setup.

3) Create a new Nginx configuration file for the site.

```
sudo nano /etc/nginx/sites-enabled/ddsonarqube
```

4) Add this configuration so that Nginx will route incoming traffic to SonarQube.

```
server {
listen 80;
server_name sonarqube.onecloudhelper.com;
location / {
proxy_pass http://127.0.0.1:9000;
}
}
```

> *Here, replace the part **server_name sonarqube.onecloudhelper.com**; with your domain or sub-domain name.*

5) Next, make sure your configuration file has no syntax errors.

```
sudo nginx -t
```

Checking the Nginx configuration file for any errors.

6) Enable the Nginx server.

```
sudo systemctl enable nginx
```

7) Restart the nginx server

```
sudo systemctl restart nginx
```

8) Check the status of the nginx server

```
sudo systemctl status nginx
```

Starting and Enabling the Nginx server

9) Use the configured domain.

You can now visit **http://sonarqube.onecloudhelper.com** in your web browser. You'll be greeted with the SonarQube web interface.

The configured domain sonarqube.onecloudhelper.com is working perfectly.

> *Wait, seems there is a problem ahead, let's solve it.*

There seems to be a problem ahead

> *Problem:*
>
> *But the domain is working on HTTP protocol and we need to make it secure by making it work with the HTTPS protocol with SSL certificate.*

Installing SSL certificate for maximum security

We will use the free SSL certificate provided by Certbot for our sub-domain sonarqube.onecloudhelper.com. Let's see how we will do this.

i) Installing certbot

```
sudo snap install --classic certbot
```

Installing Certbot dependencies

ii) Generate SSL certificate

```
sudo certbot --nginx
```

Generating SSL certificate for the required domain. Please read the asked question carefully and answer them accordingly.

Success message that the SSL certificate has been deployed on the domain sonarqube.onecloudhelper.com

iii) Certificate deployed successfully.

Now if we hit the domain sonarqube.onecloudhelper.com, it automatically redirects to HTTPS.

SSL certificate implemented successfully

AAHHH....

> *It was a lot of work.*

Was it?

> *Let's recap what we have done till now.*

i) Installed and configured JDK 17 and PostgreSQL.

ii) Installed and configured SonarQube.

iii) Installed and configured the Nginx to run the SonarQube on a domain.

iv) Installed the SSL certificate on the Subdomain.

v) And finally made it work.

Congratulations, You did it and everything looks perfect, you are ready to go ahead using the awesome SonarQube and start making your code cleaner and smarter.

Thank you and We will meet soon....

Cheers, Cheers, Cheers, Cheers.....

Sonarqube Aws Ec2 Nginx Lets Encrypt Cloud Computing

Follow

Written by DeshDeepakDhobi (DD)

57 Followers

AWS Community Builder | 2x - AWS Certified | DevOps Engineer | L2 Support Engineer | Problem solver, enthusiast learner and helper.

More from DeshDeepakDhobi (DD)

DeshDeepakDhobi (DD)

How to happily SSH/Login into the AWS-managed ECS Fargate Containers?

AWS ECS Fargate is a serverless approach to deploying containers that helps the customers focus on the major development works rather than...

7 min read · Jun 28, 2022

12

DeshDeepakDhobi (DD) in AWS Tip

How to solve the issue of “Failed” status for AWS ACM certificates?

Problem:

3 min read · Feb 6

1

DeshDeepakDhobi (DD) in AWS Tip

How to Install, Deploy, and Test Kubernetes (k8s) Cluster on Ubuntu 20.04?

Kubernetes is an open-source container orchestration system for automating software deployment, scaling, and management.

9 min read · Feb 12

2

DeshDeepakDhobi (DD) in AWS Tip

How RDP into AWS Windows Instance GUI using AWS Sessions Manager and a few fun commands?

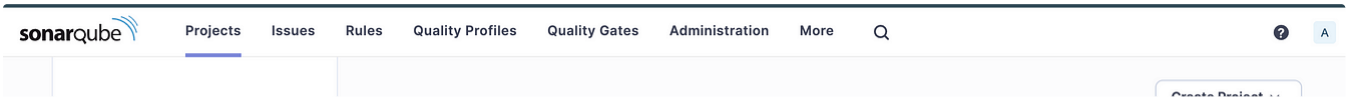
AWS EC2 instances provide us with the SSHing capability using the concept of Public and Private keys.

5 min read · Feb 6

11

See all from DeshDeepakDhobi (DD)

Recommended from Medium



Humza Arshad Khan

SonarQube Installation on ubuntu 20.04

How to Install #sonarqube in Ubuntu Linux—Download and Setup SonarQube

3 min read · Oct 19

Hopefully Surprising Blog in Stackademic

SonarQube Community Edition: Comprehensive guide for a free personal setup

Supercharge your code with SonarQube!

8 min read · Oct 3

1

Thomas Puntillo

Creating a Jenkins container with persistent data using Docker Compose

Hello there! I'm so glad you decided to read this article because today we'll be discussing a tool that will change your life (assuming...

10 min read · Jul 9

85 1

Theara Seng

Building a Full-Stack Product App with React, Spring Boot, and Docker Compose

In this blog, I am going to share with you a very simple product app that uses React, Spring Boot, and MySQL as the frontend, backend, and...

5 min read · Oct 28

15

Mohsin Rubel

CI/CD Pipeline Setup with Jenkins, SonarQube, Docker, and Nginx

Project Objective: Set up a Jenkins Freestyle Project for a portfolio hosted on GitHub. Environment: Three VMs — 1st for Jenkins, 2nd for...

3 min read · Sep 12

2