

How to install and configure SonarQube with Database on AWS EC2 Ubuntu

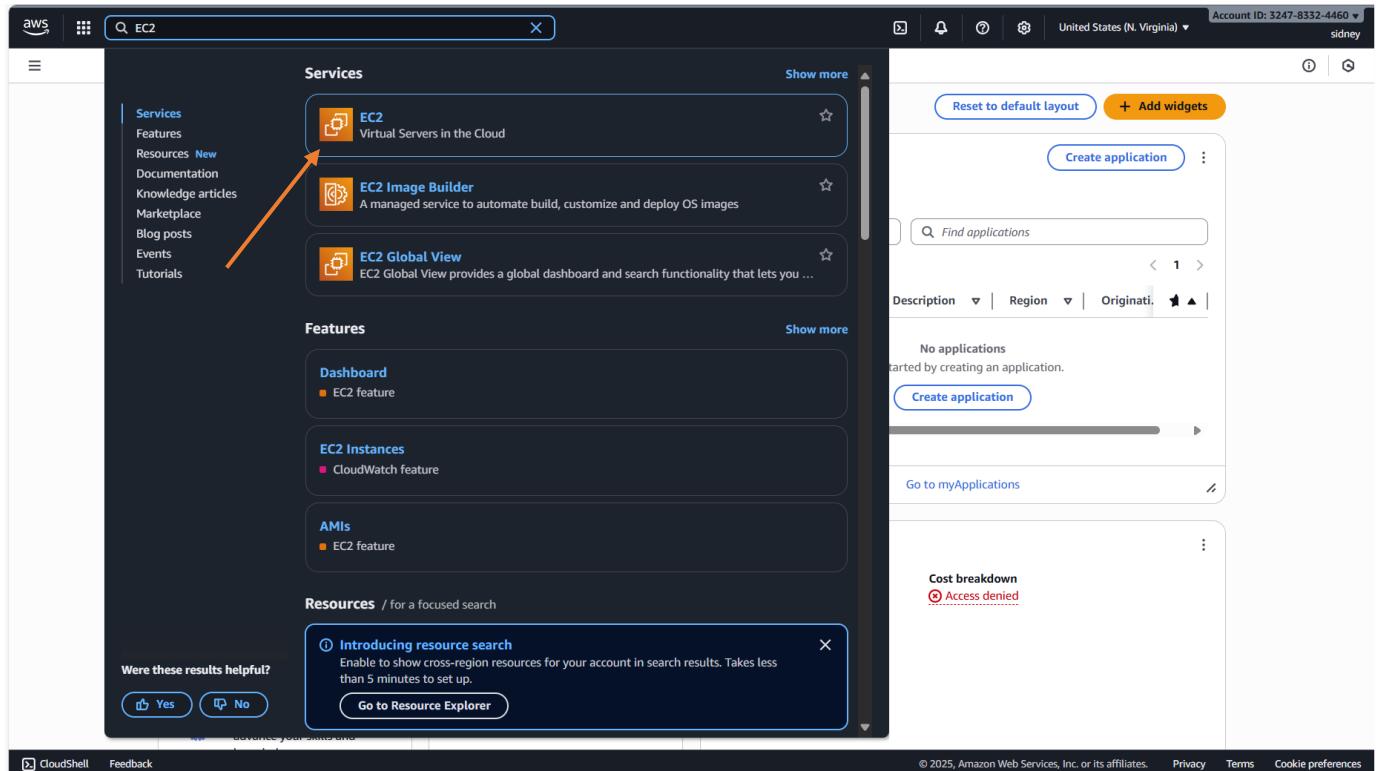
In this document, I will show how to install SonarQube with database on Ubuntu EC2 instance, This could be a step while building a CI/CD pipeline with Jenkins, Gitlab or CircleCI.

STEP 1: Launch an Ubuntu EC2 instance and SSH Connect to it

We will launch an ubuntu EC2 instance and later on SSH connect to it to install our dependencies.

PART 1: Launch an Ubuntu EC2 instance

Go to AWS Management console and search for “EC2” Instance.



Click on “EC2”

The screenshot shows the AWS EC2 landing page. The left sidebar contains a navigation menu with sections like EC2, Instances, Images, Elastic Block Store, Network & Security, and more. The main content area features a hero section for 'Amazon Elastic Compute Cloud (EC2)' with the tagline 'Create, manage, and monitor virtual servers in the cloud.' Below this are sections for 'Benefits and features' (with a sub-section on scalability), 'Use cases', and 'Pricing (US)'. A large orange button labeled 'Launch a virtual server' with 'Launch instance' and 'View dashboard' options is prominently displayed.

Click on “Launch Instance”

The screenshot shows the 'Launch an instance' wizard. Step 1 is 'Application and OS Images (Amazon Machine Image)'. It displays a search bar, a 'Recent' tab, and a 'Quick Start' tab. Under 'Recent', there are icons for Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. A callout box highlights the 'Free tier' information for the selected Amazon Linux 2023 kernel-6.1 AMI. The summary on the right shows 1 instance, the selected AMI, the instance type t2.micro, a new security group, and 1 volume(s) - 8 GiB.

We will call the instance “sonar-server”

The screenshot shows the AWS EC2 "Launch an instance" wizard. In the "Name and tags" step, a single tag "jenkins-server" is added under the "Name" field. There is also a link to "Add additional tags".

Scroll down to “AMI”, select “ubuntu”

The screenshot shows the "Application and OS Images (Amazon Machine Image)" section. Under the "Quick Start" tab, the "Ubuntu" option is selected, highlighted with a black border. Other options like Amazon Linux, macOS, Windows, Red Hat, SUSE Linux, and Debian are also listed. To the right, there is a search bar and a button to "Browse more AMIs". Below the tabs, a specific AMI is detailed: "Ubuntu Server 24.04 LTS (HVM), SSD Volume Type". It includes details like AMI ID, Virtualization type, and ENA support. The "Free tier eligible" status is indicated.

Scroll down to “Instance Type” and select “t2.micro”

The screenshot shows the "Instance type" selection step. The "t2.micro" instance type is selected and highlighted with a black border. It provides basic details: Family: t2, 1 vCPU, 1 GiB Memory, Current generation: true. It also lists base pricing for On-Demand Windows, Ubuntu Pro, SUSE, RHEL, and Linux instances. A note at the bottom states "Additional costs apply for AMIs with pre-installed software". On the right, there are buttons for "All generations" and "Compare instance types".

Scroll down to “Key Pair”

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

Select ▼

Create new key pair

Click on “Create new key pair”

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.
Enter key pair name

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair

Private key file format

.pem For use with OpenSSH

.ppk For use with PuTTY

⚠️ When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel Create key pair

Give the key pair a name, I will call it “sonar-key”

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.
sonar-key

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair

Private key file format

.pem For use with OpenSSH

.ppk For use with PuTTY

⚠️ When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel Create key pair

Click on “Create key pair”

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

sonar-key ▼ Create new key pair

Scroll down to “Network Settings”. Select “Allow HTTP traffic from the internet”, “Allow HTTPS traffic from the internet” and “Allow SSH traffic from”.

▼ Network settings [Info](#) Edit

Network | [Info](#)
vpc-0128e9209eaef1c37

Subnet | [Info](#)
No preference (Default subnet in any availability zone)

Auto-assign public IP | [Info](#)
Enable
Additional charges apply when outside of free tier allowance

Firewall (security groups) | [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

We'll create a new security group called 'launch-wizard-5' with the following rules:

Allow SSH traffic from
Helps you connect to your instance

Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. X

Scroll down to the end

▼ Configure storage [Info](#) Advanced

1x GiB Root volume, 3000 IOPS, Not encrypted

ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage X

Add new volume

The selected AMI contains instance store volumes, however the instance does not allow any instance store volumes. None of the instance store volumes from the AMI will be accessible from the instance

ⓘ Click refresh to view backup information ⟳
The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems Edit

▶ Advanced details [Info](#)

ⓘ Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance. X

Cancel Launch instance Preview code

Click on “Launch Instance”

The screenshot shows the AWS EC2 Instances launch page. At the top, there's a green success banner stating "Successfully initiated launch of instance (i-093e6e5bae1cd3540)". Below it, a "Next Steps" section lists several options: "Create billing and free tier usage alerts", "Connect to your instance", "Connect an RDS database", "Create EBS snapshot policy", "Manage detailed monitoring", "Create Load Balancer", "Create AWS budget", and "Manage CloudWatch alarms". Each option has a corresponding button and a "Learn more" link. An orange arrow points from the text "Click on ‘Instances’" to the "Instances" link in the breadcrumb navigation at the top left.

Click on “Instances”

The screenshot shows the AWS EC2 Instances list page. On the left, a sidebar menu includes "Instances", "Images", "Elastic Block Store", and "Network & Security". The main area displays a table titled "Instances (1) Info" with one row. The instance details are: Instance ID: i-03386f8b488c0086d, Instance state: Running, Status check: Initializing. An orange arrow points from the text "You can see that our just created instance is initializing. Wait for it to pass the “2/2 check”" to the "Status check" column.

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	sonar-server	i-03386f8b488c0086d	Running	t2.micro	Initializing		us-east-1d

You can see that our just created instance is initializing. Wait for it to pass the “2/2 check”

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces. The main area is titled 'Instances (1) Info' and shows a table with one row. The row contains a checkbox, the name 'sonar-server', the Instance ID 'i-03386fb488c0086d', the Instance state 'Running' (with a green icon), the Instance type 't2.micro', and the Status check '2/2 checks passed' (also with a green icon). Below the table is a section titled 'Select an instance'.

The instance has passed the “2/2 check”

PART 2: Add Port 9000 for SonarQube

We will add port 9000 that will be used to access SonarQube through the browser. Now, we have to add port 9000 to enable access SonarQube from our browser. Go to our EC2 instance

This screenshot is identical to the one above it, showing the AWS EC2 Instances page with a single running instance named 'sonar-server'. The status check message has changed to '0/2 checks passed' (with a red icon), indicating that the port configuration has not yet been applied or verified.

Select the EC2 instance

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Instances, Images, and Network & Security. The main area displays a table of instances with one row selected: 'sonar-server' (Instance ID: i-03386f8b488c0086d). At the top of the main area, there are tabs: Details, Status and alarms, Monitoring, Security (which is highlighted with a red arrow), Networking, Storage, and Tags. To the right of the table, there's a detailed view of the selected instance, including its public and private IP addresses, instance type (t2.micro), and security group information.

Click on the “Security” tab

This screenshot is similar to the previous one but focuses on the security details of the selected instance. The 'Security' tab is again highlighted with a red arrow. In the 'Security details' section, it shows the IAM Role (empty) and the Security groups assigned to the instance (sg-0fcfa6cfcc12d64c86 (launch-wizard-11)). Below this, the 'Inbound rules' section is visible, showing three open ports: 443, 22, and 80, all originating from 0.0.0.0/0 and pointing to the launch-wizard-11 security group. The 'Outbound rules' section is also partially visible at the bottom.

Click on the “Security Group” url

EC2 > Security Groups > sg-0fcfa6cf1e12d64c86 - launch-wizard-11

Details

Security group name	sg-0fcfa6cf1e12d64c86	Security group ID	sg-0fcfa6cf1e12d64c86	Description	launch-wizard-11 created 2025-09-06T23:0:56Z	VPC ID	vpc-0128e9209eaef1c37
Owner	324783324460	Inbound rules count	3 Permission entries	Outbound rules count	1 Permission entry		

Inbound rules | Outbound rules | Sharing - new | VPC associations - new | Tags

Inbound rules (3)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-02cebe2cbcba6c711	IPv4	HTTPS	TCP	443	0.0.0.0/0
-	sgr-004e5cad248863dca	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-0435312b1bf37aebo	IPv4	HTTP	TCP	80	0.0.0.0/0

Manage tags | Edit inbound rules

Click on “Edit Inbound rules”

EC2 > Security Groups > sg-0fcfa6cf1e12d64c86 - launch-wizard-11 > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	0.0.0.0/0
sgr-004e5cad248863dca	SSH	TCP	22	Custom	0.0.0.0/0
sgr-0435312b1bf37aebo	HTTP	TCP	80	Custom	0.0.0.0/0

Add rule | Delete

Cancel | Preview changes | Save rules

Click on “Add Rule”

Edit inbound rules Info
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	0.0.0.0/0 <small>X</small>
sgr-004e5cad248863dca	SSH	TCP	22	Custom	0.0.0.0/0 <small>X</small>
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	0.0.0.0/0 <small>X</small>
-	Custom TCP	TCP	0	Custom	0.0.0.0/0 <small>X</small>

Add rule

Cancel Preview changes Save rules

Click on the drop down on “**Port Range**” and enter the value “**9000**”

Edit inbound rules Info
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	0.0.0.0/0 <small>X</small>
sgr-004e5cad248863dca	SSH	TCP	22	Custom	0.0.0.0/0 <small>X</small>
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	0.0.0.0/0 <small>X</small>
-	Custom TCP	TCP	9000	Custom	0.0.0.0/0 <small>X</small>

Add rule

Cancel Preview changes Save rules

Click on the drop down on “**source**” and select “**Anywhere-IPv4**”

Prepared by Sidney Smith Ebot

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	Action
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	Q 0.0.0.0/0 X	Delete
sgr-004e5cad248863dca	SSH	TCP	22	Custom	Q 0.0.0.0/0 X	Delete
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	Q 0.0.0.0/0 X	Delete
-	Custom TCP	TCP	9000	Anywhere	Q 0.0.0.0/0 X	Delete

Add rule

Cancel Preview changes Save rules

Click on “Save rules”

Inbound security group rules successfully modified on security group (sg-0fca6cfee12d64c86 | launch-wizard-11)

Details

Security group name	Security group ID	Description	VPC ID
launch-wizard-11	sg-0fca6cfee12d64c86	launch-wizard-11 created 2025-09-06T23:03:28.836Z	vpc-0128e9209eaef1c37
Owner	324783324460	Inbound rules count	Outbound rules count
		4 Permission entries	1 Permission entry

Inbound rules | Outbound rules | Sharing - new | VPC associations - new | Tags

Inbound rules (4)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
sgr-02cebe2cbcba6c711	IPv4	HTTPS	TCP	443	0.0.0.0/0	
sgr-004e5cad248863dca	IPv4	SSH	TCP	22	0.0.0.0/0	
sgr-0435312b1bf37ae0	IPv4	HTTP	TCP	80	0.0.0.0/0	
sgr-02aff83fa1a9b9a8a	IPv4	Custom TCP	TCP	9000	0.0.0.0/0	

Manage tags | Edit inbound rules

Port 9000 has been added.

PART 4: SSH Connect to the EC2 instance

Now, let us connect to our EC2 instance through SSH

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, etc. The main area shows a table for 'Instances (1) Info' with one entry: Name: sonar-server, Instance ID: i-03386f8b488c0086d, Instance state: Running, Instance type: t2.micro, Status check: 2/2 checks passed, and Availability Zone: us-east-1d. A blue 'Connect' button is visible in the top right of the table header. The URL in the browser is <https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:instanceId=i-03386f8b488c0086d>.

Select the instance

This screenshot shows the same EC2 Instances page as above, but now the 'sonar-server' instance is selected. The 'Connect' button in the top bar is highlighted with an orange arrow. Below it, the instance details page is shown for 'i-03386f8b488c0086d (sonar-server)'. The 'Details' tab is active. The 'Public IPv4 address' is listed as 18.208.247.55. Other details include Instance state: Running, Instance type: t2.micro, and various network and storage configurations. The URL in the browser is <https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:instanceId=i-03386f8b488c0086d>.

Click on “Connect”

The screenshot shows the AWS EC2 Connect interface. At the top, there's a navigation bar with 'EC2' and 'Instances'. Below it, a sub-navigation bar shows 'i-03386f8b488c0086d > Connect to instance'. The main content area is titled 'Connect info' and contains instructions for connecting via an SSH client. It lists four steps: 1. Open an SSH client, 2. Locate your private key file (sonar-key.pem), 3. Run the command 'chmod 400 sonar-key.pem', and 4. Connect to your instance using its Public DNS ('ec2-18-208-247-55.compute-1.amazonaws.com'). An orange arrow points from the text 'Copy this command and open PowerShell' below to the Public DNS URL. A note at the bottom states: 'Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.'

Copy this command and open PowerShell

```
ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com
```

A screenshot of a Windows PowerShell window titled 'Windows PowerShell'. The command 'ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com' is visible in the terminal.

Navigate to the “Downloads” folder where our “.pem” file is stored by using the command:

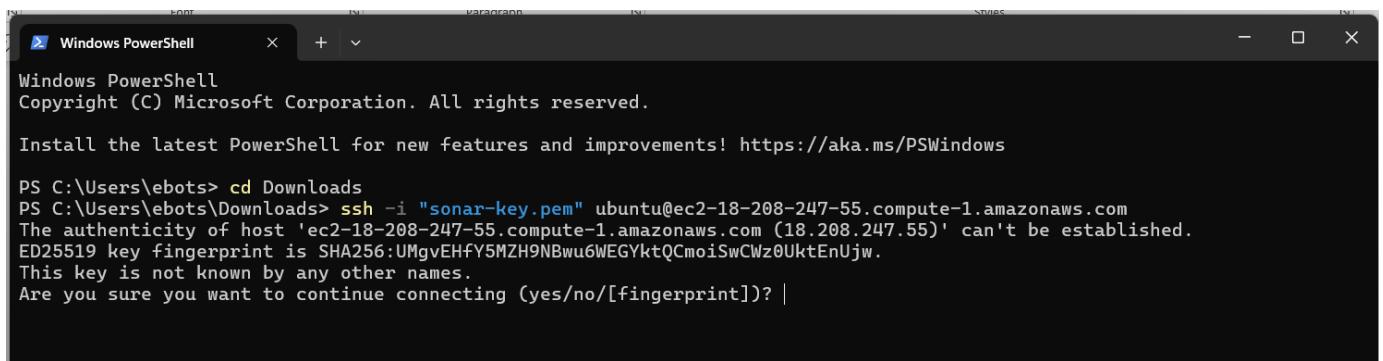
```
cd Downloads
```

A screenshot of a Windows PowerShell window titled 'Windows PowerShell'. The command 'cd Downloads' is visible in the terminal, showing the user navigating to the 'Downloads' folder.

Then run the copied command:

```
ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com
```

Prepared by Sidney Smith Ebot

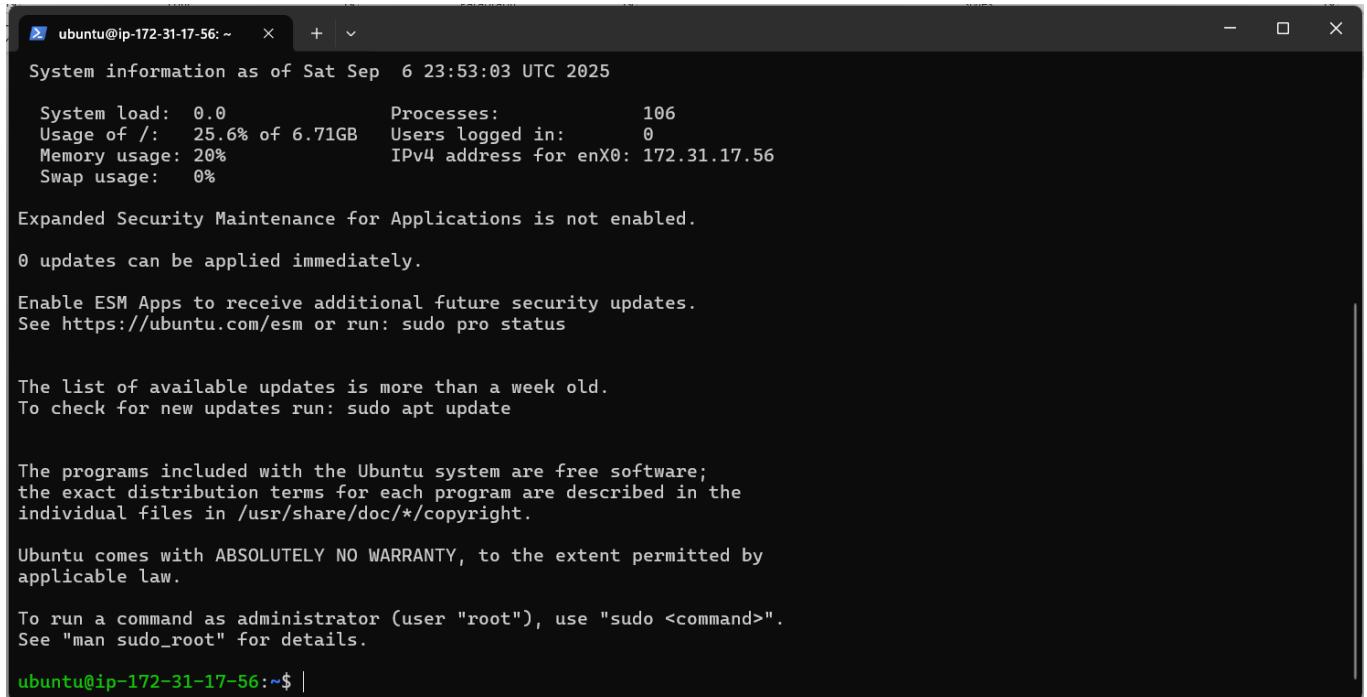


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ebots> cd Downloads
PS C:\Users\ebots\Downloads> ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com
The authenticity of host 'ec2-18-208-247-55.compute-1.amazonaws.com (18.208.247.55)' can't be established.
ED25519 key fingerprint is SHA256:UMgvEHfY5MZH9NBwu6WEGYktQCmoiswCWz0UktEnUjw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? |
```

Type “**yes**” and press ENTER



```
ubuntu@ip-172-31-17-56: ~
System information as of Sat Sep 6 23:53:03 UTC 2025

System load: 0.0 Processes: 106
Usage of /: 25.6% of 6.71GB Users logged in: 0
Memory usage: 20% IPv4 address for enx0: 172.31.17.56
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

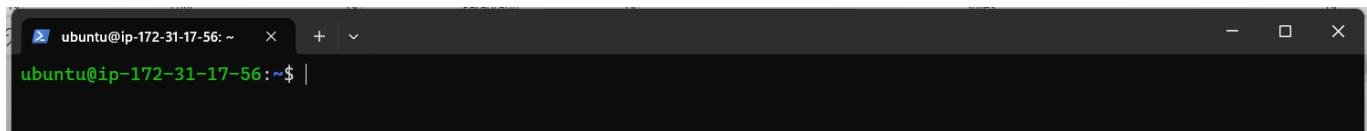
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-17-56:~$ |
```

We have successfully SSH connected to our EC2 instance

Clear the terminal using the command:

```
clear
```



```
ubuntu@ip-172-31-17-56: ~
ubuntu@ip-172-31-17-56:~$ |
```

Grant Root user access using the command:

```
sudo su
```

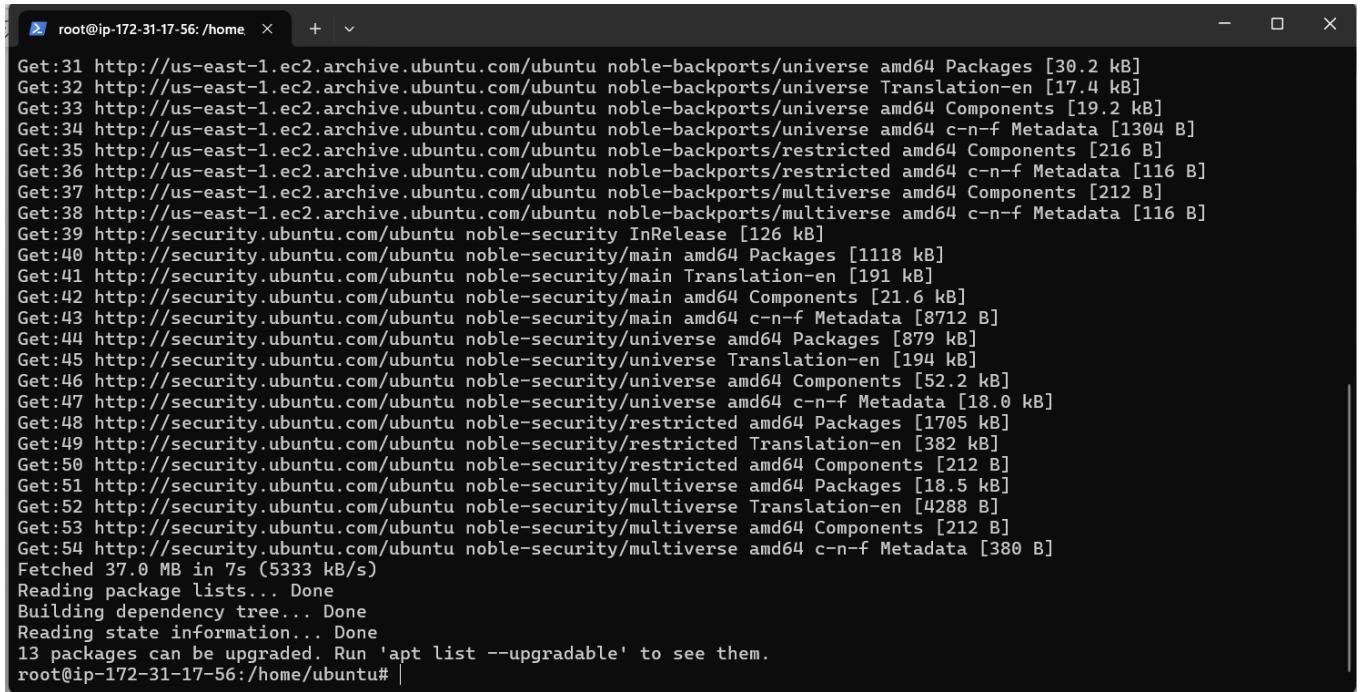


```
root@ip-172-31-17-56:/home ~$ sudo su
root@ip-172-31-17-56:/home/ubuntu# |
```

STEP 2: Install Java JDK 17

Let's rejuvenate our Ubuntu server

`sudo apt update`



```
root@ip-172-31-17-56:/home ~$ sudo apt update
Get:31 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [30.2 kB]
Get:32 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [17.4 kB]
Get:33 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [19.2 kB]
Get:34 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1304 B]
Get:35 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1118 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [191 kB]
Get:42 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [8712 B]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [879 kB]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [194 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.2 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [18.0 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1705 kB]
Get:49 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [382 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [18.5 kB]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [4288 B]
Get:53 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:54 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [380 B]
Fetched 37.0 MB in 7s (5333 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
13 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-17-56:/home/ubuntu# |
```

Update the server

`sudo apt upgrade -y`

```
root@ip-172-31-17-56:/home ~ + | 
Found initrd image: /boot/microcode.cpio /boot/initrd.img-6.14.0-1011-aws
Warning: os-prober will not be executed to detect other bootable partitions.
Systems on them will not be added to the GRUB boot configuration.
Check GRUB_DISABLE_OS_PROBER documentation entry.
Adding boot menu entry for UEFI Firmware Settings ...
done
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# |
```

Install OpenJDK 17

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

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

```
root@ip-172-31-17-56:/home ~ + | 
Setting up openjdk-17-jre:amd64 (17.0.16+8~us1-0ubuntu1~24.04.1) ...
Setting up openjdk-17-jdk:amd64 (17.0.16+8~us1-0ubuntu1~24.04.1) ...
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jconsole to provide /usr/bin/jconsole (jconsole) in auto mode
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.10+dfsg-3ubuntu3.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

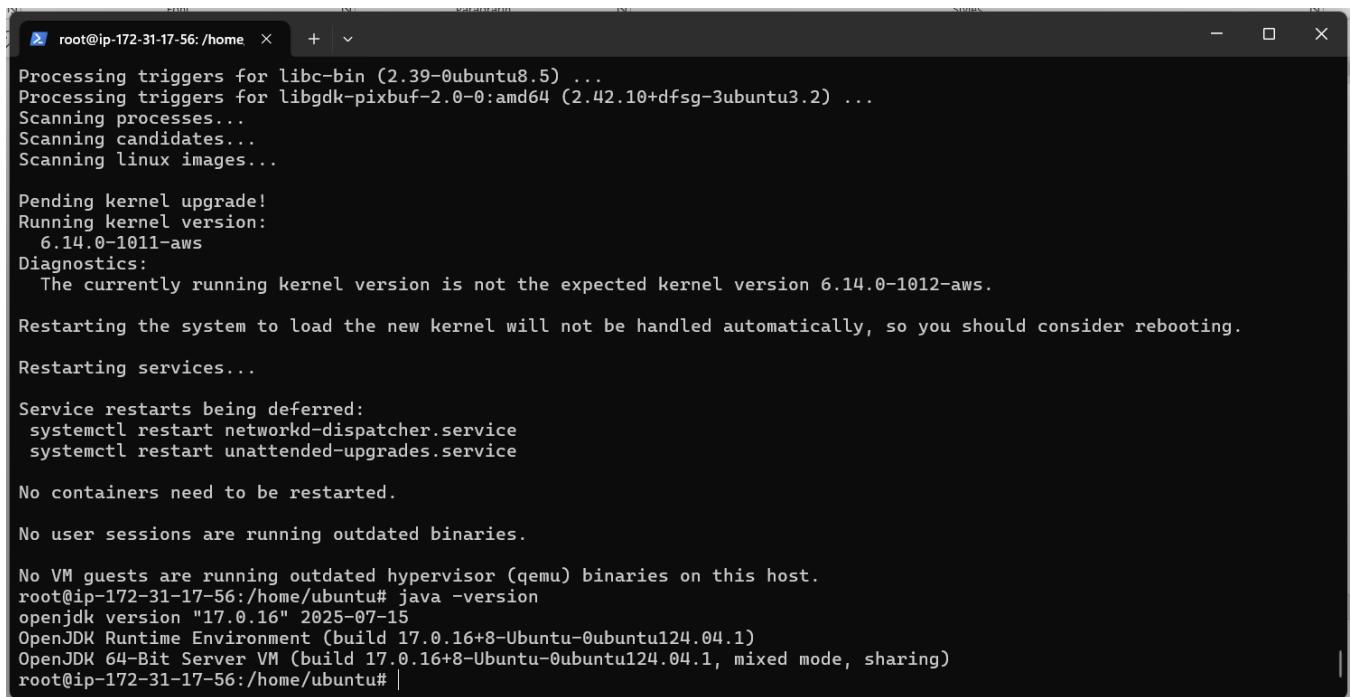
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# |
```

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

```
java -version
```



```
root@ip-172-31-17-56:/home ~ + ^

Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.10+dfsg-3ubuntu3.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# java -version
openjdk version "17.0.16" 2025-07-15
OpenJDK Runtime Environment (build 17.0.16+8-Ubuntu-0ubuntu124.04.1)
OpenJDK 64-Bit Server VM (build 17.0.16+8-Ubuntu-0ubuntu124.04.1, mixed mode, sharing)
root@ip-172-31-17-56:/home/ubuntu# |
```

STEP 3: Install and Configure PostgreSQL

Add the PostgreSQL repository

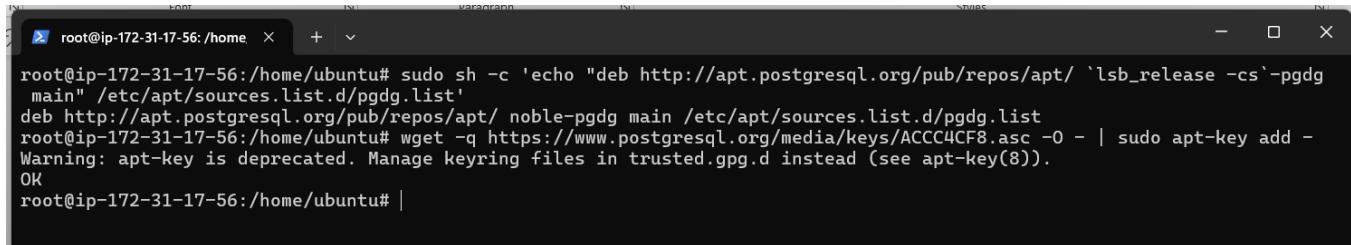
```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main"
/etc/apt/sources.list.d/pgdg.list'
```



```
root@ip-172-31-17-56:/home/ubuntu# sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg
main" /etc/apt/sources.list.d/pgdg.list'
deb http://apt.postgresql.org/pub/repos/apt/ noble-pgdg main /etc/apt/sources.list.d/pgdg.list
root@ip-172-31-17-56:/home/ubuntu# |
```

Add the PostgreSQL signing key

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



```
root@ip-172-31-17-56:/home/ubuntu# sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg
main" /etc/apt/sources.list.d/pgdg.list'
deb http://apt.postgresql.org/pub/repos/apt/ noble-pgdg main /etc/apt/sources.list.d/pgdg.list
root@ip-172-31-17-56:/home/ubuntu# wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
root@ip-172-31-17-56:/home/ubuntu# |
```

Install PostgreSQL

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

```
root@ip-172-31-17-56:/home ~ + | 
performing post-bootstrap initialization ... ok
syncing data to disk ... ok
Setting up postgresql-contrib (16+257build1.1) ...
Setting up postgresql (16+257build1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu#
```

Enable the database server to start automatically on reboot

```
sudo systemctl enable postgresql
```

```
root@ip-172-31-17-56:/home ~ + | 
Setting up postgresql (16+257build1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

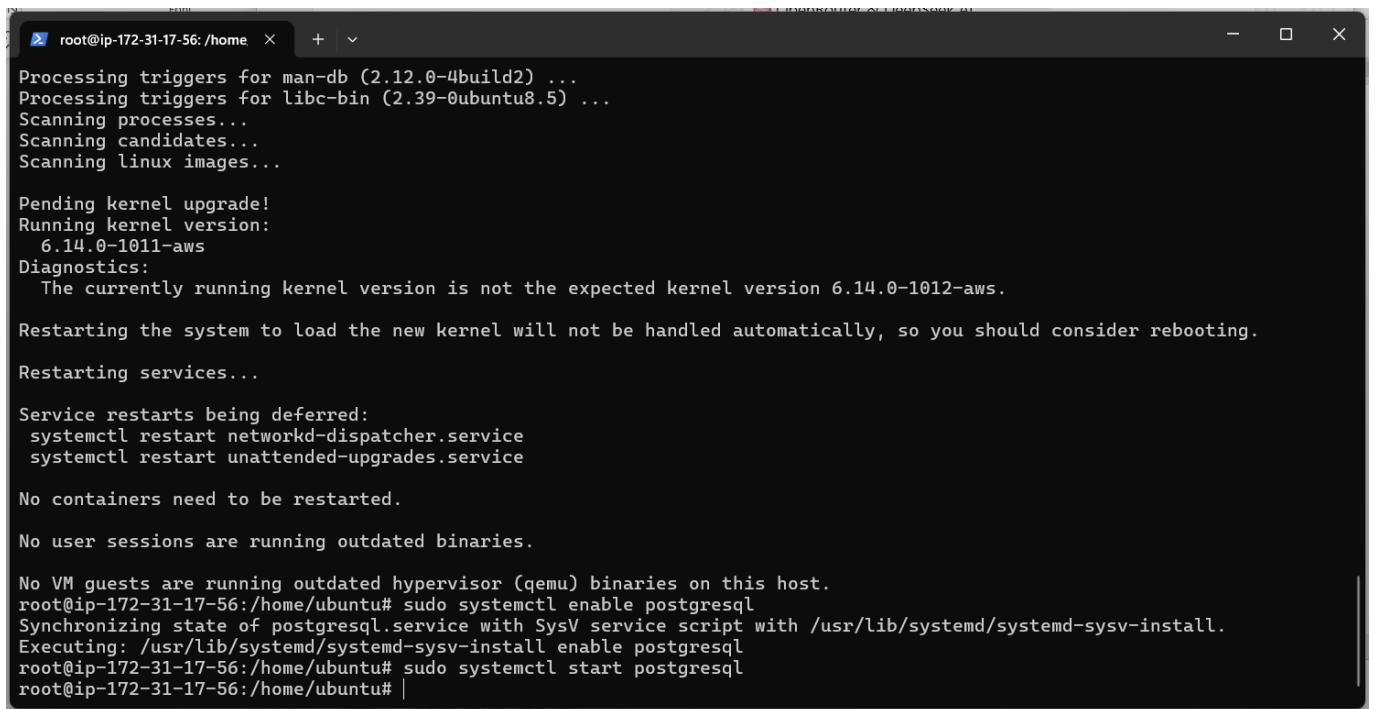
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu#
```

Start the database server

```
sudo systemctl start postgresql
```



root@ip-172-31-17-56:/home ~ + |

```
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

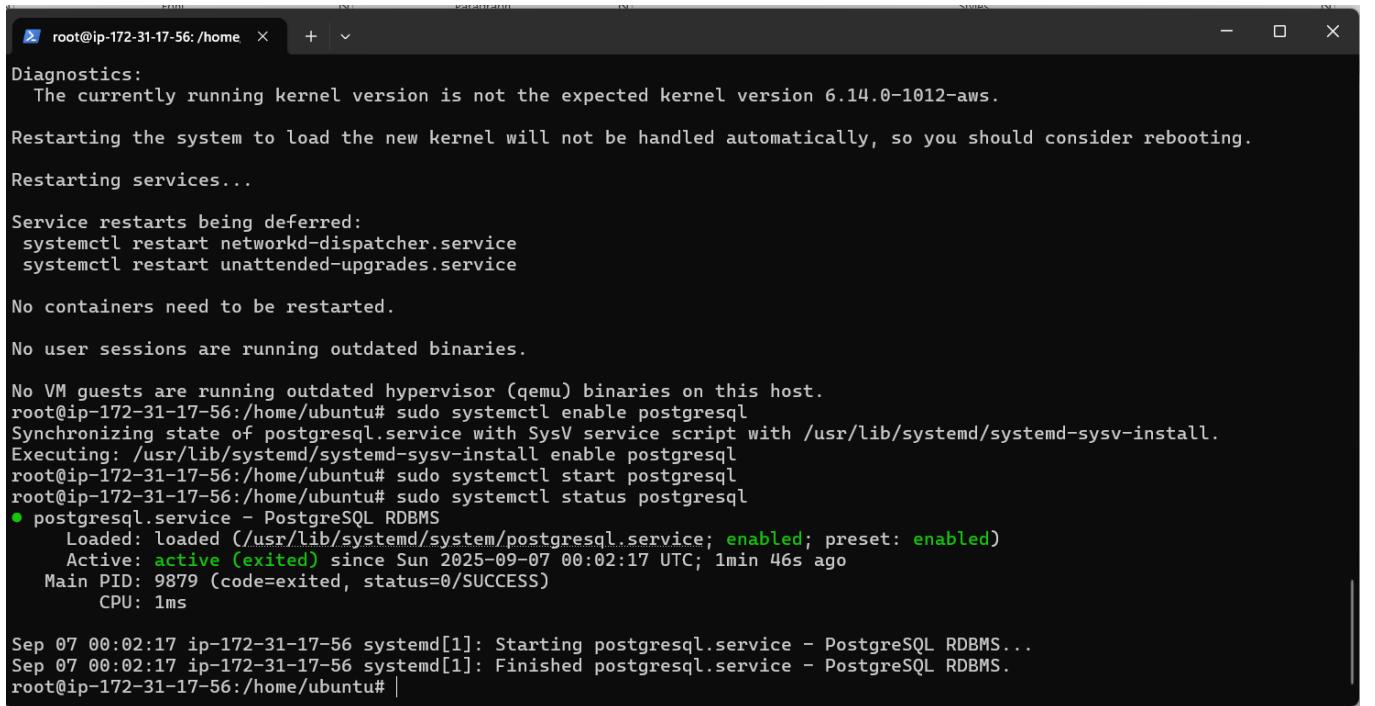
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# |
```

Check the status of the database server

```
sudo systemctl status postgresql
```



root@ip-172-31-17-56:/home ~ + |

```
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
  Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
  Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
    Main PID: 9879 (code=exited, status=0/SUCCESS)
      CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# |
```

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

```
psql --version
```

```
root@ip-172-31-17-56: /home. X + - □ ×

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
      Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# |
```

Switch to the Postgres user

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

```
root@ip-172-31-17-56:~| + | - | X
root@ip-172-31-17-56:/home | + | - | X

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:$ |
```

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

```
root@ip-172-31-17-56:/home ~ + - X
Restarting services...
Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
      Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ |
```

Log in to PostgreSQL

psql

```
root@ip-172-31-17-56:/home ~ + - X
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
      Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:$ createuser ddsonar
postgres@ip-172-31-17-56:$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

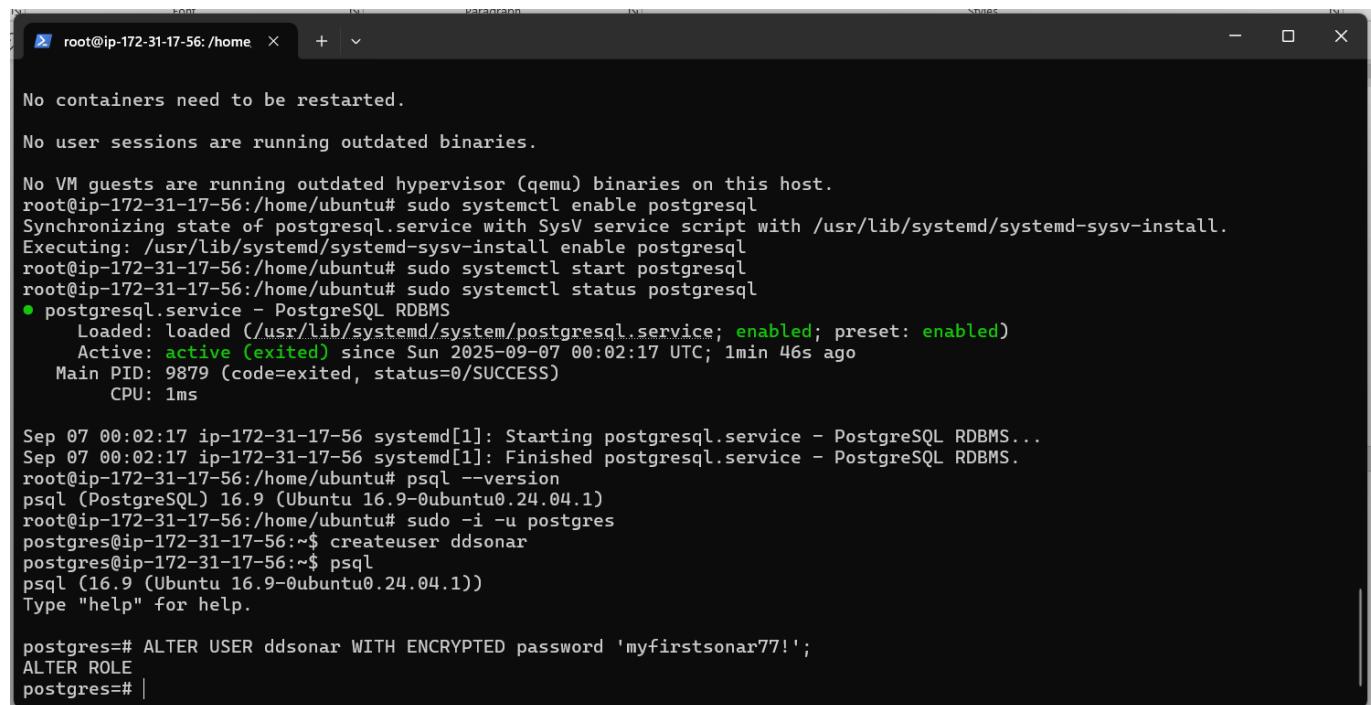
postgres=# |
```

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 'myfirstsonar77!';
```



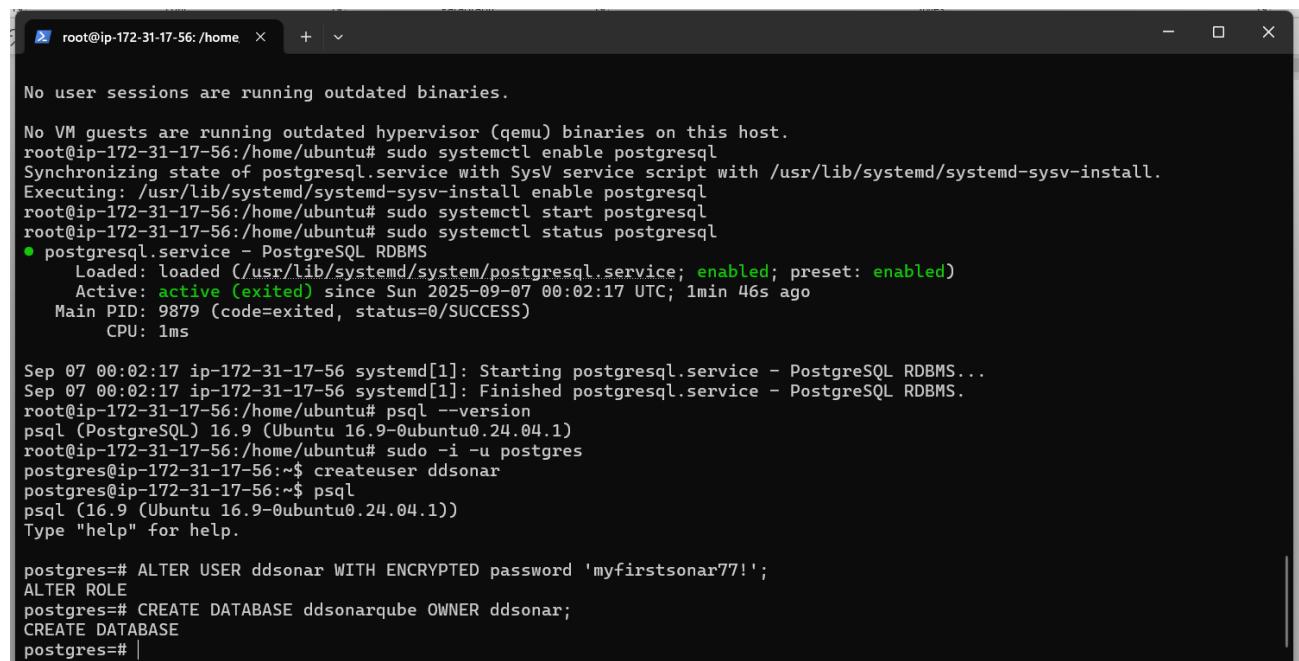
```
No containers need to be restarted.  
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql  
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.  
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql  
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql  
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql  
● postgresql.service - PostgreSQL RDBMS  
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)  
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago  
      Main PID: 9879 (code=exited, status=0/SUCCESS)  
        CPU: 1ms  
  
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...  
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.  
root@ip-172-31-17-56:/home/ubuntu# psql --version  
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)  
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres  
postgres@ip-172-31-17-56:~$ createuser ddsonar  
postgres@ip-172-31-17-56:~$ psql  
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))  
Type "help" for help.  
  
postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';  
ALTER ROLE  
postgres=# |
```

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;
```



```
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql  
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.  
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql  
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql  
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql  
● postgresql.service - PostgreSQL RDBMS  
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)  
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago  
      Main PID: 9879 (code=exited, status=0/SUCCESS)  
        CPU: 1ms  
  
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...  
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.  
root@ip-172-31-17-56:/home/ubuntu# psql --version  
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)  
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres  
postgres@ip-172-31-17-56:~$ createuser ddsonar  
postgres@ip-172-31-17-56:~$ psql  
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))  
Type "help" for help.  
  
postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';  
ALTER ROLE  
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;  
CREATE DATABASE  
postgres=# |
```

Grant all the privileges on the ddsonarqube database to the ddsonar user.

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

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home  +  -
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

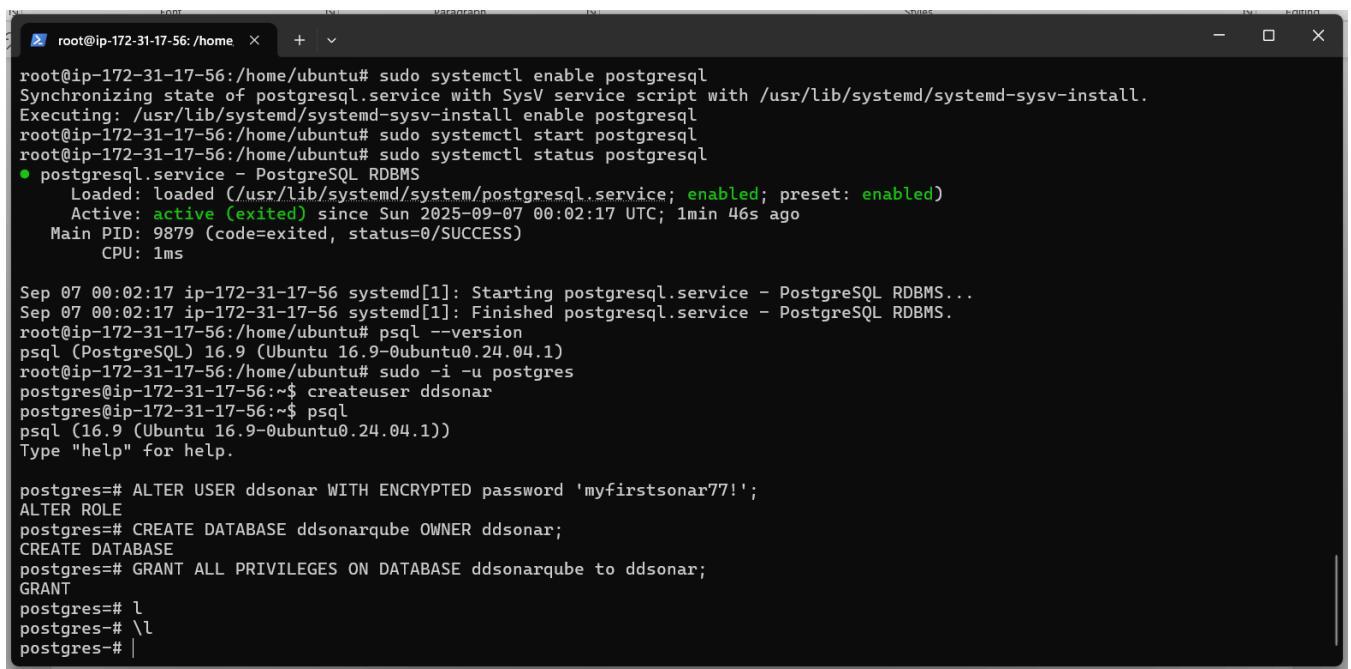
postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# |
```

Let's check the created user and the database.

a) To check the created database using the command:

\1

Press a



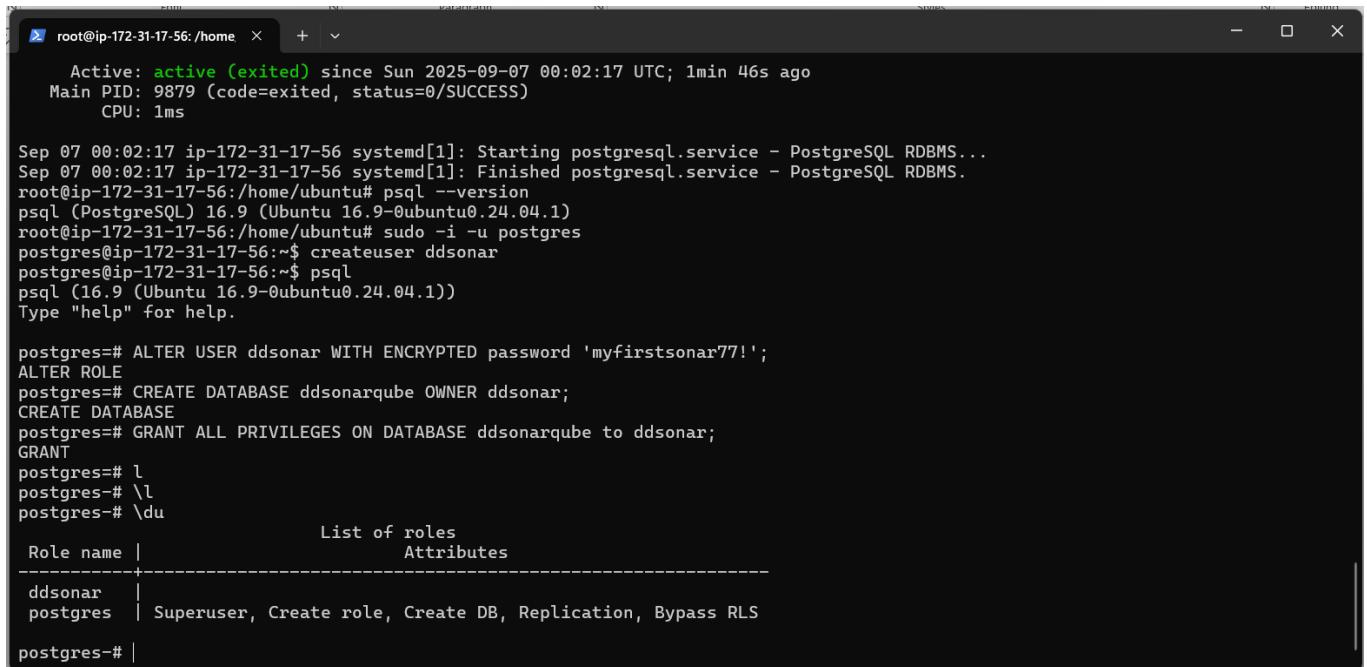
```
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# l
postgres=# \l
postgres=# |
```

b) To check the created database user

\du



```
Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
Main PID: 9879 (code=exited, status=0/SUCCESS)
CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# l
postgres=# \l
postgres=# \du
      List of roles
Role name | Attributes
-----+-----
ddsonar  |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS
postgres# |
```

Exit PostgreSQL.

\q

```
root@ip-172-31-17-56:/home ~ + ~
Main PID: 9879 (code=exited, status=0/SUCCESS)
CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# \l
postgres-# \l
postgres-# \du
          List of roles
 Role name |           Attributes
-----+-----
ddsonar  |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS

postgres-# \q
postgres@ip-172-31-17-56:~$ |
```

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

exit

```
root@ip-172-31-17-56:/home ~ + ~
Main PID: 9879 (code=exited, status=0/SUCCESS)
CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# \l
postgres-# \l
postgres-# \du
          List of roles
 Role name |           Attributes
-----+-----
ddsonar  |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS

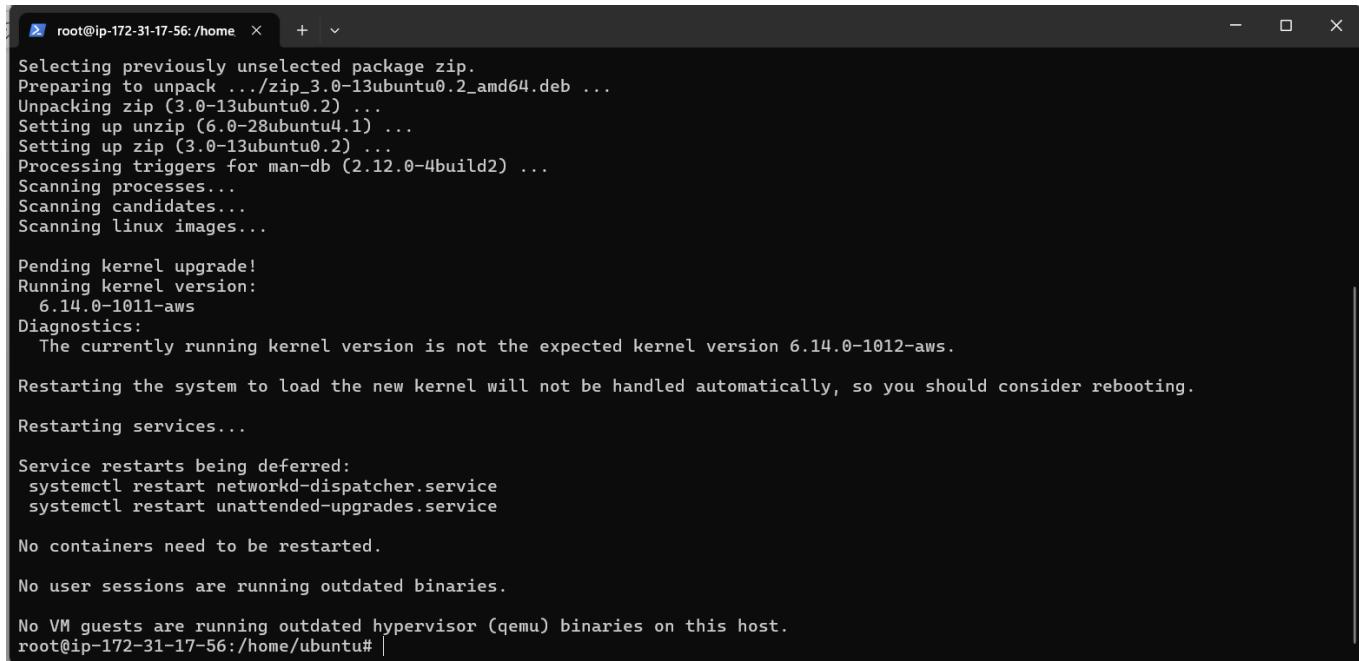
postgres-# \q
postgres@ip-172-31-17-56:~$ exit
logout
root@ip-172-31-17-56:/home/ubuntu# |
```

STEP 4: Install SonarQube

Part 1: Download and Install SonarQube

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

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



```
root@ip-172-31-17-56:/home ~ + ~
Selecting previously unselected package zip.
Preparing to unpack .../zip_3.0-13ubuntu0.2_amd64.deb ...
Unpacking zip (3.0-13ubuntu0.2) ...
Setting up unzip (6.0-28ubuntu4.1) ...
Setting up zip (3.0-13ubuntu0.2) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# |
```

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-10.0.0.68432.zip
```

```
Diagnostics:  
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.  
Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.  
Restarting services...  
Service restarts being deferred:  
systemctl restart networkd-dispatcher.service  
systemctl restart unattended-upgrades.service  
No containers need to be restarted.  
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@ip-172-31-17-56:/home/ubuntu# sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.0.0.68432.zip  
--2025-09-07 00:15:59-- https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.0.0.68432.zip  
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 99.84.188.21, 99.84.188.106, 99.84.188.45, ...  
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)|99.84.188.21|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 352909963 (337M) [binary/octet-stream]  
Saving to: 'sonarqube-10.0.0.68432.zip'  
  
sonarqube-10.0.0.68432.zip      100%[=====] 336.56M   113MB/s    in 3.0s  
2025-09-07 00:16:02 (113 MB/s) - 'sonarqube-10.0.0.68432.zip' saved [352909963/352909963]  
root@ip-172-31-17-56:/home/ubuntu# |
```

Unzip the downloaded file.

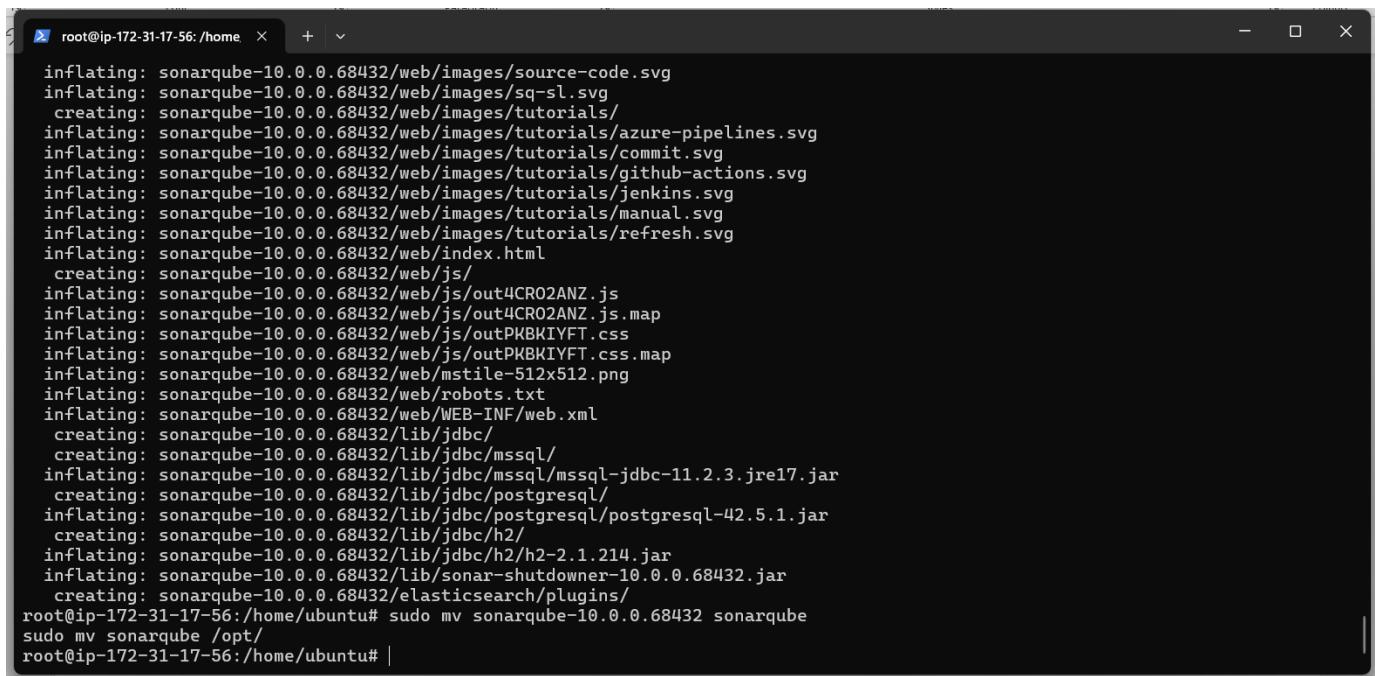
```
sudo unzip sonarqube-10.0.0.68432.zip
```

```
inflating: sonarqube-10.0.0.68432/web/images/SonarLint-connection-ok.png  
inflating: sonarqube-10.0.0.68432/web/images/SonarLint-connection-request.png  
inflating: sonarqube-10.0.0.68432/web/images/source-code.svg  
inflating: sonarqube-10.0.0.68432/web/images/sq-sl.svg  
creating: sonarqube-10.0.0.68432/web/images/tutorials/  
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg  
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg  
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg  
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg  
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg  
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg  
inflating: sonarqube-10.0.0.68432/web/index.html  
creating: sonarqube-10.0.0.68432/web/js/  
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js  
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map  
inflating: sonarqube-10.0.0.68432/web/js/outPKBK1YFT.css  
inflating: sonarqube-10.0.0.68432/web/js/outPKBK1YFT.css.map  
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png  
inflating: sonarqube-10.0.0.68432/web/robots.txt  
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml  
creating: sonarqube-10.0.0.68432/lib/jdbc/  
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/  
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql/mssql-jdbc-11.2.3.jre17.jar  
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/  
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar  
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/  
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar  
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar  
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/  
root@ip-172-31-17-56:/home/ubuntu# |  
sudo mv sonarqube /opt/
```

Move the unzipped files to /opt/sonarqube directory

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

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



```
root@ip-172-31-17-56:/home ~ + | 
inflating: sonarqube-10.0.0.68432/web/images/source-code.svg
creating: sonarqube-10.0.0.68432/web/images/sq-sl.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIVFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIVFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# |
```

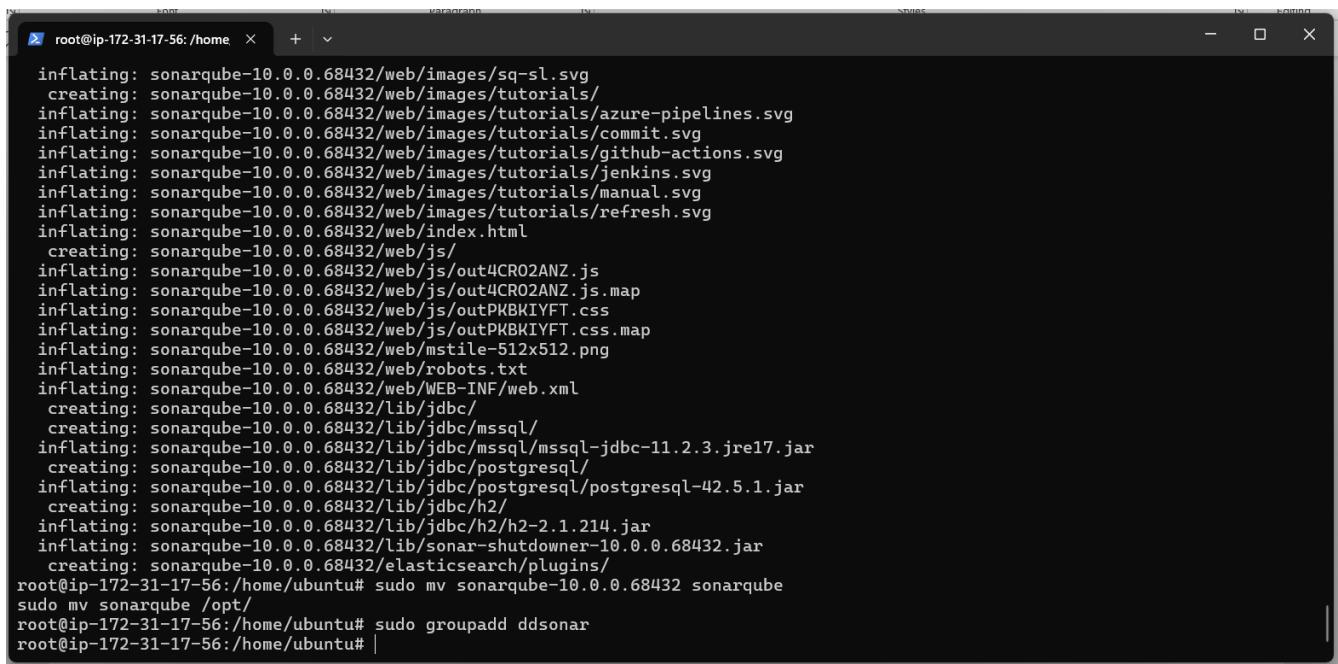
Part 2: Add SonarQube Group and User

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**.

Create a sonar group.

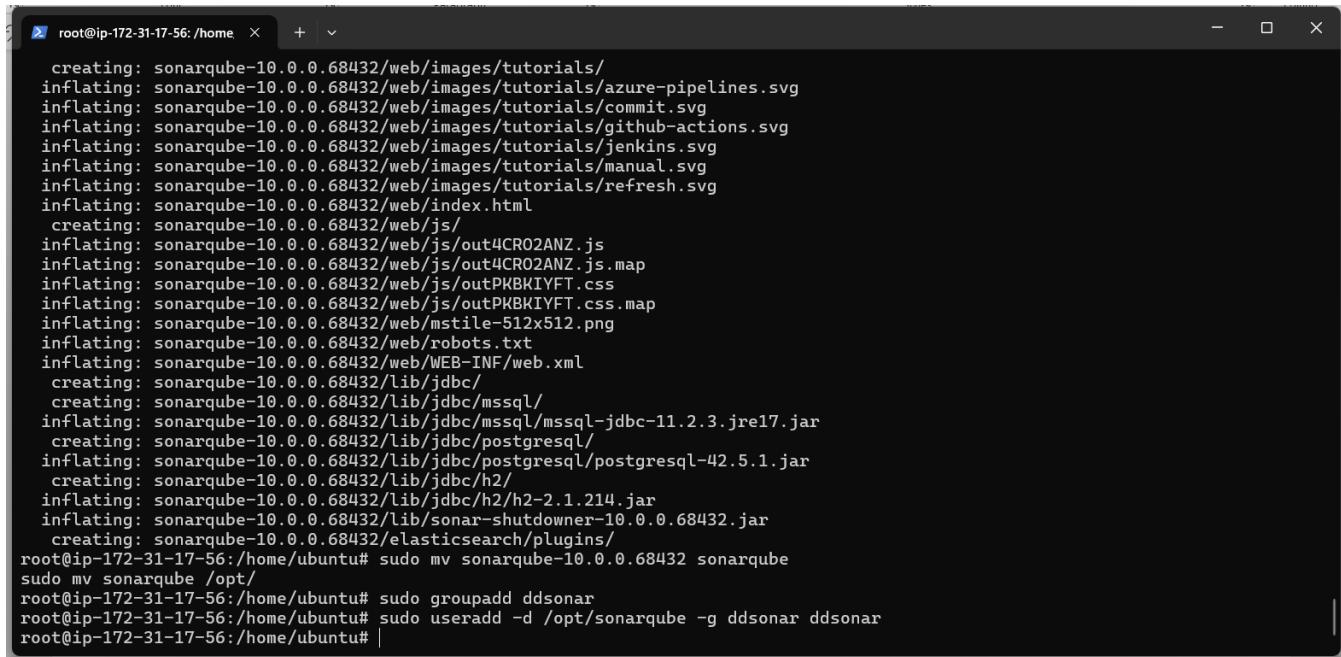
```
sudo groupadd ddsonar
```



```
root@ip-172-31-17-56:/home ~ + | 
inflating: sonarqube-10.0.0.68432/web/images/sq-sl.svg
creating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIVFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIVFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# sudo groupadd ddsonar
root@ip-172-31-17-56:/home/ubuntu# |
```

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

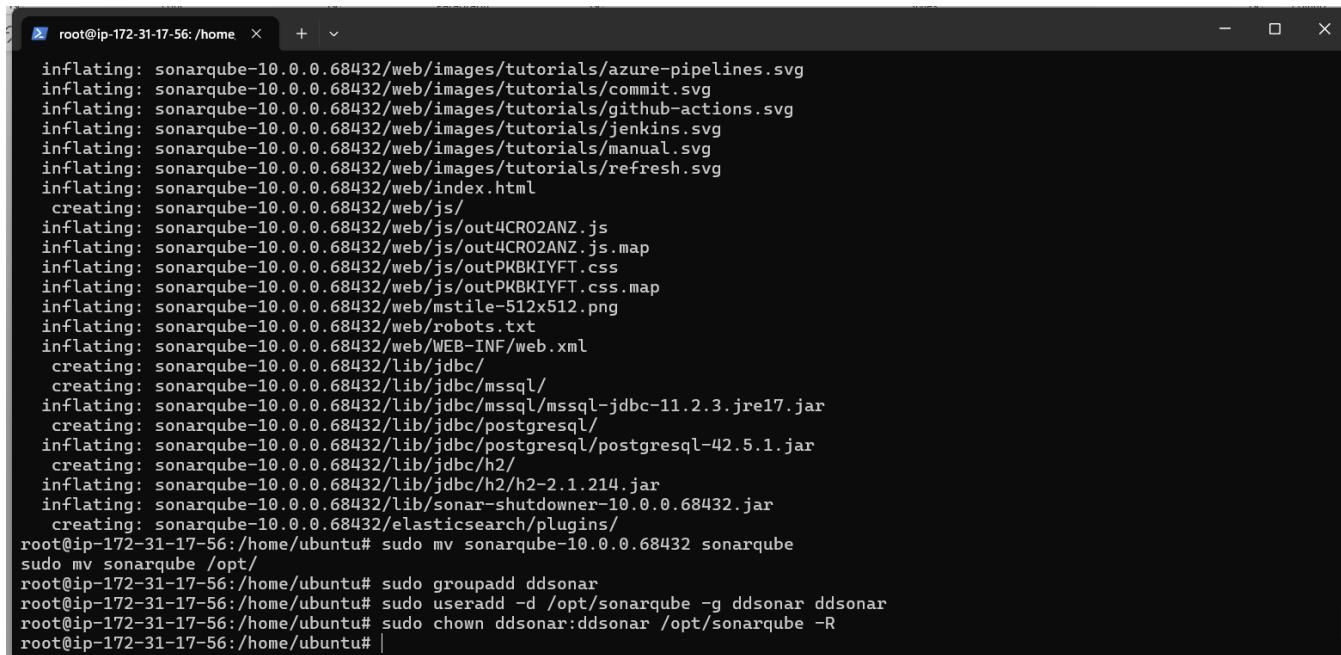
```
sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
```



```
root@ip-172-31-17-56:/home × + ▾
creating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# sudo groupadd ddsonar
root@ip-172-31-17-56:/home/ubuntu# sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
root@ip-172-31-17-56:/home/ubuntu# |
```

Grant the sonar user access to the /opt/sonarqube directory.

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

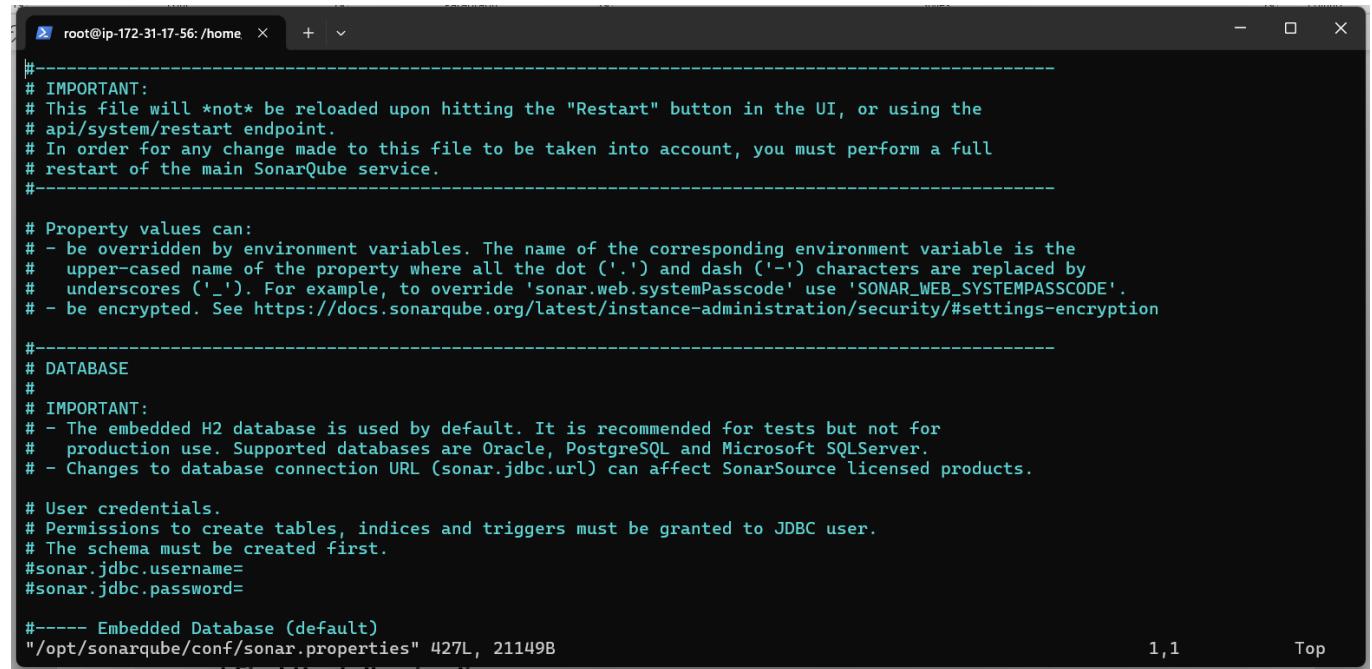


```
root@ip-172-31-17-56:/home × + ▾
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# sudo groupadd ddsonar
root@ip-172-31-17-56:/home/ubuntu# sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
root@ip-172-31-17-56:/home/ubuntu# sudo chown ddsonar:ddsonar /opt/sonarqube -R
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 3: Configure SonarQube

Edit the SonarQube configuration file.

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

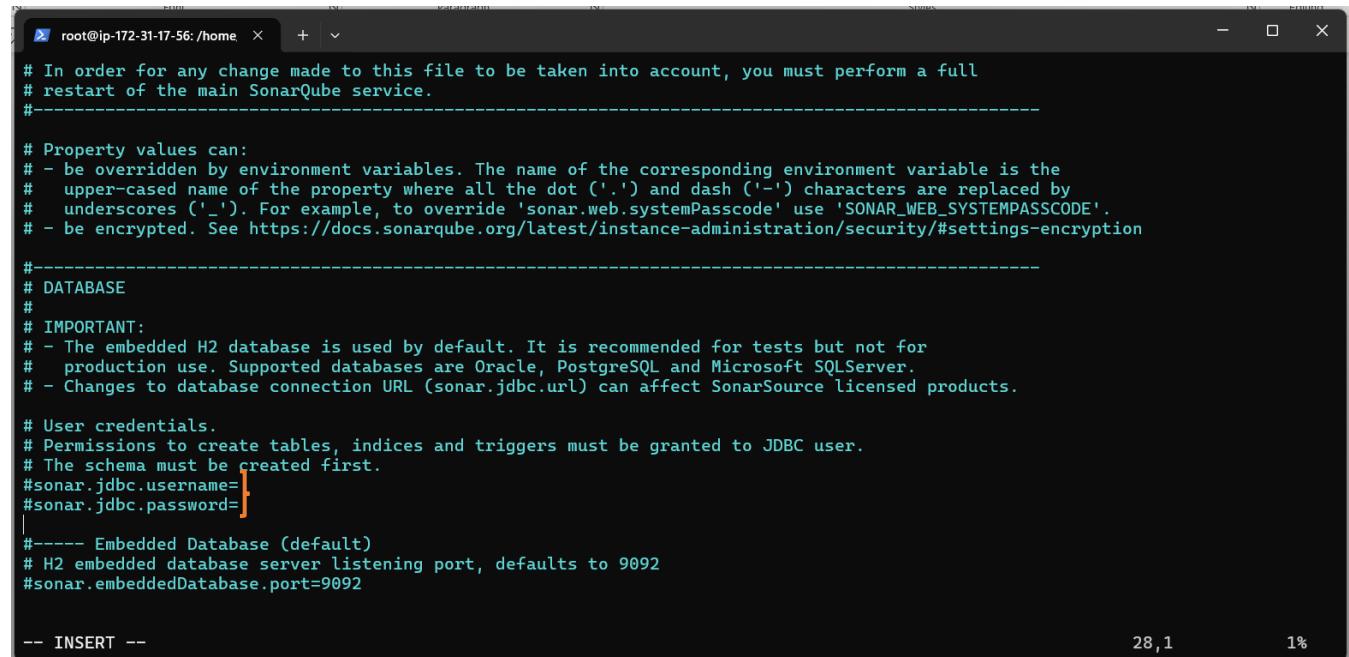


```
#  
# IMPORTANT:  
# This file will *not* be reloaded upon hitting the "Restart" button in the UI, or using the  
# api/system/restart endpoint.  
# In order for any change made to this file to be taken into account, you must perform a full  
# restart of the main SonarQube service.  
  
# Property values can:  
# - be overridden by environment variables. The name of the corresponding environment variable is the  
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by  
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.  
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption  
  
#-----  
# DATABASE  
#  
# IMPORTANT:  
# - The embedded H2 database is used by default. It is recommended for tests but not for  
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.  
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.  
  
# User credentials.  
# Permissions to create tables, indices and triggers must be granted to JDBC user.  
# The schema must be created first.  
#sonar.jdbc.username=  
#sonar.jdbc.password=  
  
#----- Embedded Database (default)  
"/opt/sonarqube/conf/sonar.properties" 427L, 21149B
```

Find the following lines:

```
#sonar.jdbc.username=
```

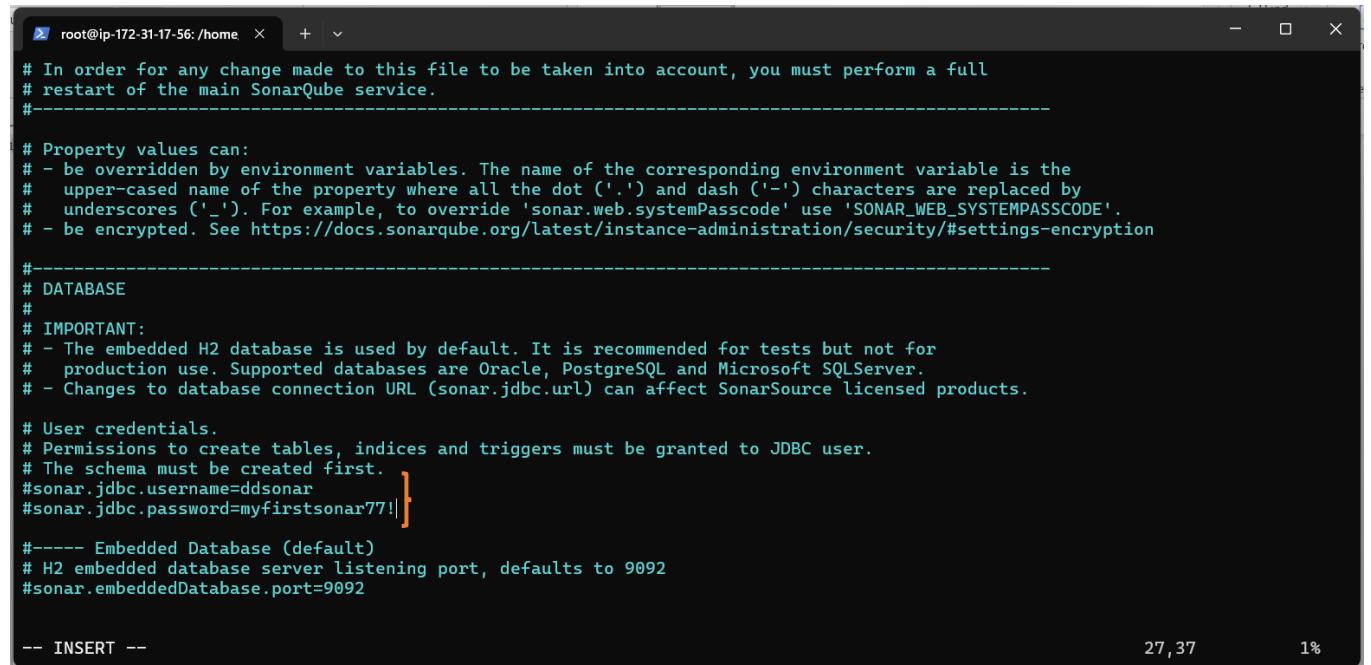
```
#sonar.jdbc.password=
```



```
# In order for any change made to this file to be taken into account, you must perform a full  
# restart of the main SonarQube service.  
  
# Property values can:  
# - be overridden by environment variables. The name of the corresponding environment variable is the  
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by  
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.  
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption  
  
#-----  
# DATABASE  
#  
# IMPORTANT:  
# - The embedded H2 database is used by default. It is recommended for tests but not for  
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.  
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.  
  
# User credentials.  
# Permissions to create tables, indices and triggers must be granted to JDBC user.  
# The schema must be created first.  
#sonar.jdbc.username=  
#sonar.jdbc.password=  
  
#----- Embedded Database (default)  
# H2 embedded database server listening port, defaults to 9092  
#sonar.embeddedDatabase.port=9092  
  
-- INSERT --
```

Uncomment the lines, and add the database user and Database password you created in Step 3. For me, it's:

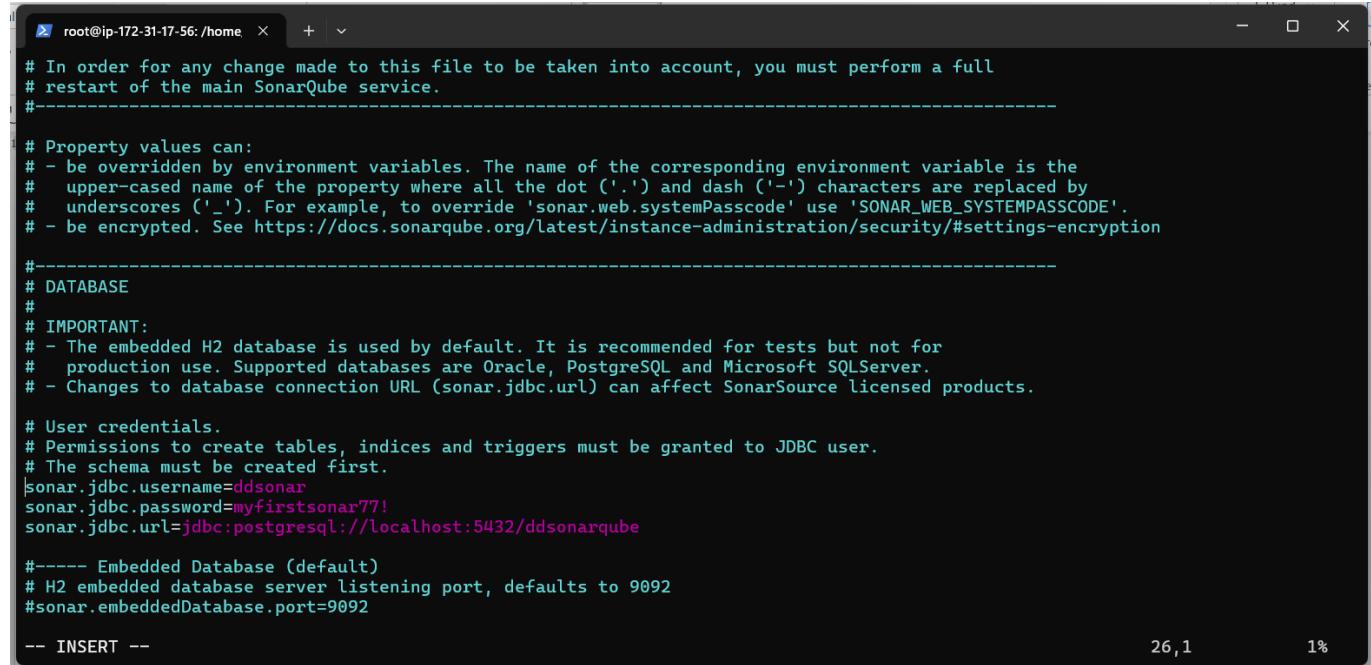
```
sonar.jdbc.username=ddsonar  
sonar.jdbc.password=myfirstsonar77!
```



```
# In order for any change made to this file to be taken into account, you must perform a full  
# restart of the main SonarQube service.  
#-----  
  
# Property values can:  
# - be overridden by environment variables. The name of the corresponding environment variable is the  
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by  
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.  
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption  
#-----  
# DATABASE  
#  
# IMPORTANT:  
# - The embedded H2 database is used by default. It is recommended for tests but not for  
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.  
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.  
  
# User credentials.  
# Permissions to create tables, indices and triggers must be granted to JDBC user.  
# The schema must be created first.  
#sonar.jdbc.username=ddsonar  
#sonar.jdbc.password=myfirstsonar77!]  
#----- Embedded Database (default)  
# H2 embedded database server listening port, defaults to 9092  
#sonar.embeddedDatabase.port=9092  
  
-- INSERT --
```

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

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

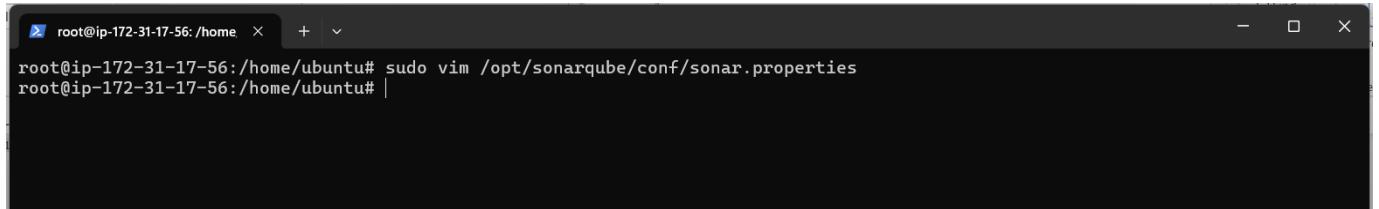


```
# In order for any change made to this file to be taken into account, you must perform a full  
# restart of the main SonarQube service.  
#-----  
  
# Property values can:  
# - be overridden by environment variables. The name of the corresponding environment variable is the  
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by  
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.  
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption  
#-----  
# DATABASE  
#  
# IMPORTANT:  
# - The embedded H2 database is used by default. It is recommended for tests but not for  
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.  
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.  
  
# User credentials.  
# Permissions to create tables, indices and triggers must be granted to JDBC user.  
# The schema must be created first.  
#sonar.jdbc.username=ddsonar  
sonar.jdbc.password=myfirstsonar77!  
sonar.jdbc.url=jdbc:postgresql://localhost:5432/ddsonarqube  
#----- Embedded Database (default)  
# H2 embedded database server listening port, defaults to 9092  
#sonar.embeddedDatabase.port=9092  
  
-- INSERT --
```

Here, ddsonarqube is the database name created.

Save and exit the file.

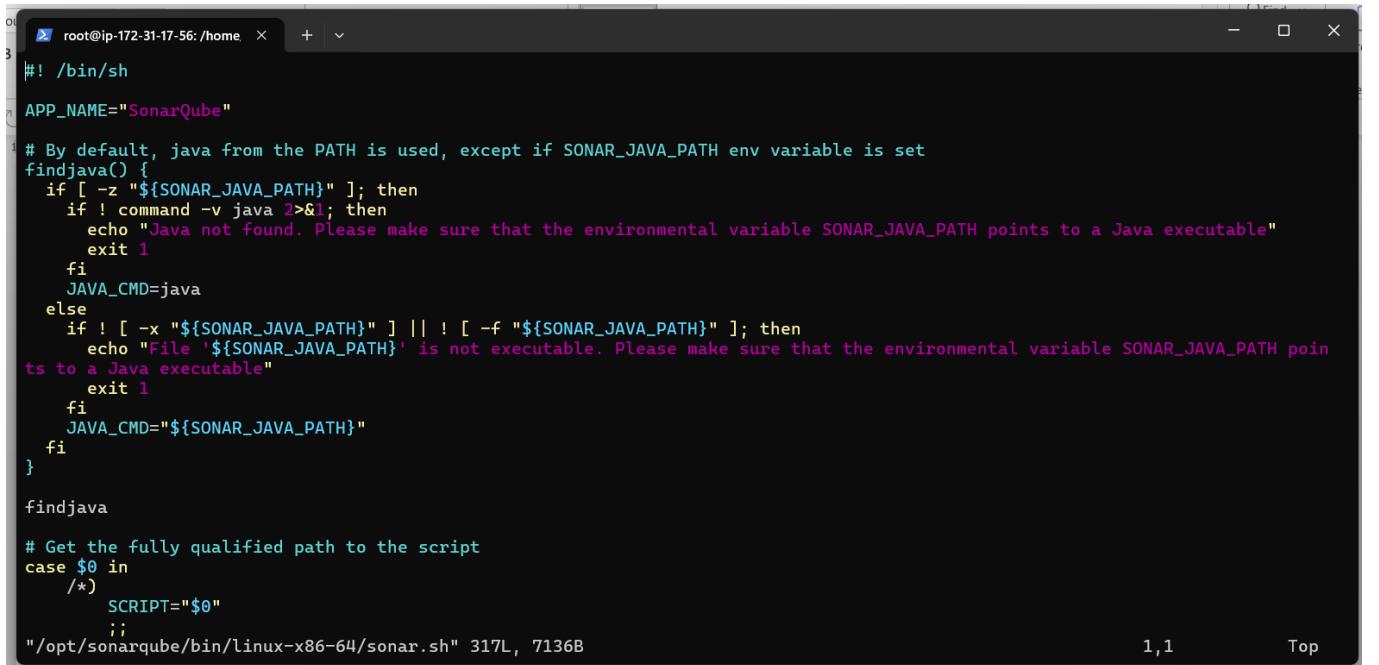
Press ESC, followed by :wq and press ENTER



```
root@ip-172-31-17-56:/home ~ + | ~
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# |
```

Edit the sonar script file.

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



```
o 1 root@ip-172-31-17-56:/home ~ + | ~
3 #! /bin/sh
APP_NAME="SonarQube"
# By default, java from the PATH is used, except if SONAR_JAVA_PATH env variable is set
findjava() {
    if [ -z "${SONAR_JAVA_PATH}" ]; then
        if ! command -v java 2>&1; then
            echo "Java not found. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD=java
    else
        if ! [ -x "${SONAR_JAVA_PATH}" ] || ! [ -f "${SONAR_JAVA_PATH}" ]; then
            echo "File '${SONAR_JAVA_PATH}' is not executable. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD="${SONAR_JAVA_PATH}"
    fi
}
findjava
# Get the fully qualified path to the script
case $0 in
    /*)
        SCRIPT="$0"
        ;;
    "/opt/sonarqube/bin/linux-x86-64/sonar.sh" 317L, 7136B
1,1          Top
```

Add the following line

`RUN_AS_USER=ddsonar`

The screenshot shows a terminal window with the following content:

```
#!/bin/sh
RUN_AS_USER=ddsonar
APP_NAME="SonarQube"

# By default, java from the PATH is used, except if SONAR_JAVA_PATH env variable is set
findjava() {
    if [ -z "${SONAR_JAVA_PATH}" ]; then
        if ! command -v java 2>&1; then
            echo "Java not found. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD=java
    else
        if ! [ -x "${SONAR_JAVA_PATH}" ] || ! [ -f "${SONAR_JAVA_PATH}" ]; then
            echo "'${SONAR_JAVA_PATH}' is not executable. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD="${SONAR_JAVA_PATH}"
    fi
}

findjava

# Get the fully qualified path to the script
case $0 in
    /*)
        SCRIPT="$0"
    -- INSERT --
```

The terminal window has a red arrow pointing to the line `RUN_AS_USER=ddsonar`. The status bar at the bottom right shows `3,20 Top`.

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

Save and exit the file.

Press ESC, followed by :wq and press ENTER

The screenshot shows a terminal window with the following content:

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu#
```

Part 4: Setup Systemd service

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

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



A screenshot of a terminal window titled "root@ip-172-31-17-56: /home". The window shows a single line of text: "/etc/systemd/system/sonar.service" [New]. The terminal has a dark background with light-colored text. The status bar at the bottom right shows "0,0-1" and "All".

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
```

```
[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
```

-- INSERT --

15,1 All

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
```

Save and exit the file.

Press ESC, followed by :wq then press ENTER

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# |
```

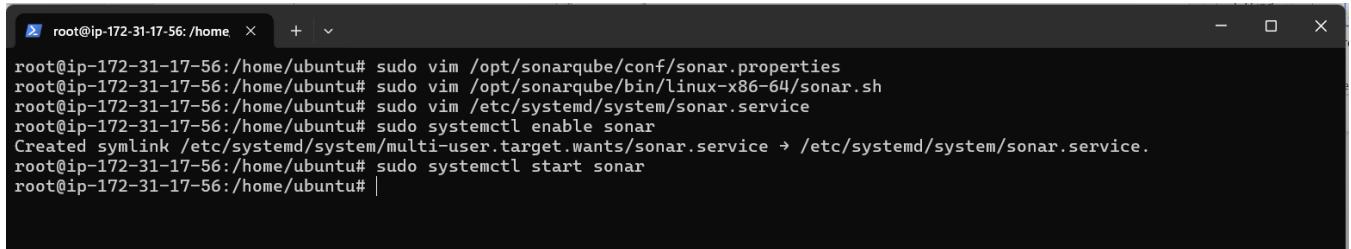
Enable the SonarQube service to run at system startup.

```
sudo systemctl enable sonar
```

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# |
```

Start the SonarQube service.

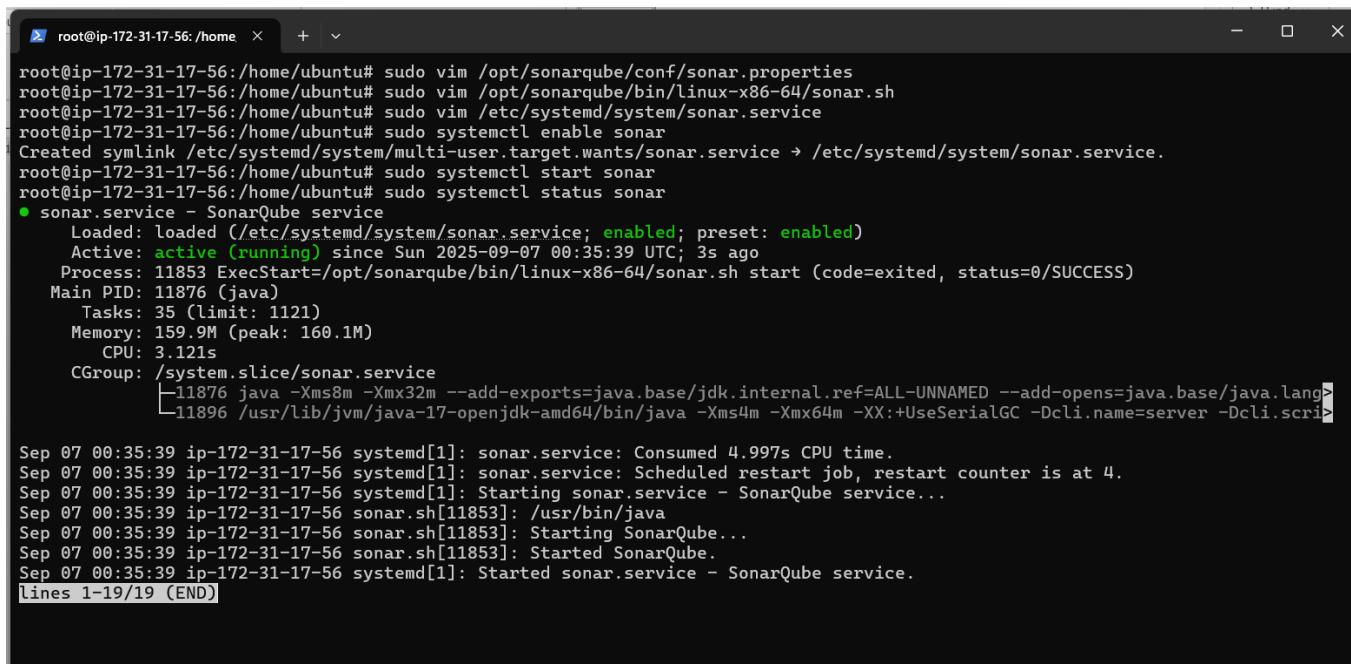
```
sudo systemctl start sonar
```



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start sonar
root@ip-172-31-17-56:/home/ubuntu#
```

Check the service status.

```
sudo systemctl status sonar
```



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start sonar
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status sonar
● sonar.service - SonarQube service
  Loaded: loaded (/etc/systemd/system/sonar.service; enabled; preset: enabled)
  Active: active (running) since Sun 2025-09-07 00:35:39 UTC; 3s ago
    Process: 11853 ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start (code=exited, status=0/SUCCESS)
   Main PID: 11876 (java)
      Tasks: 35 (limit: 1121)
     Memory: 159.9M (peak: 160.1M)
        CPU: 3.121s
       CGroup: /system.slice/sonar.service
               └─11876 java -Xms8m -Xmx32m --add-exports=java.base/jdk.internal.ref=ALL-UNNAMED --add-opens=java.base/java.lang>
                  ├─11896 /usr/lib/jvm/java-17-openjdk-amd64/bin/java -Xms4m -Xmx64m -XX:+UseSerialGC -Dcli.name=server -Dcli.script=>
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: sonar.service: Consumed 4.997s CPU time.
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: sonar.service: Scheduled restart job, restart counter is at 4.
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: Starting sonar.service - SonarQube service...
Sep 07 00:35:39 ip-172-31-17-56 sonar.sh[11853]: /usr/bin/java
Sep 07 00:35:39 ip-172-31-17-56 sonar.sh[11853]: Starting SonarQube...
Sep 07 00:35:39 ip-172-31-17-56 sonar.sh[11853]: Started SonarQube.
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: Started sonar.service - SonarQube service.
Lines 1-19/19 (END)
```

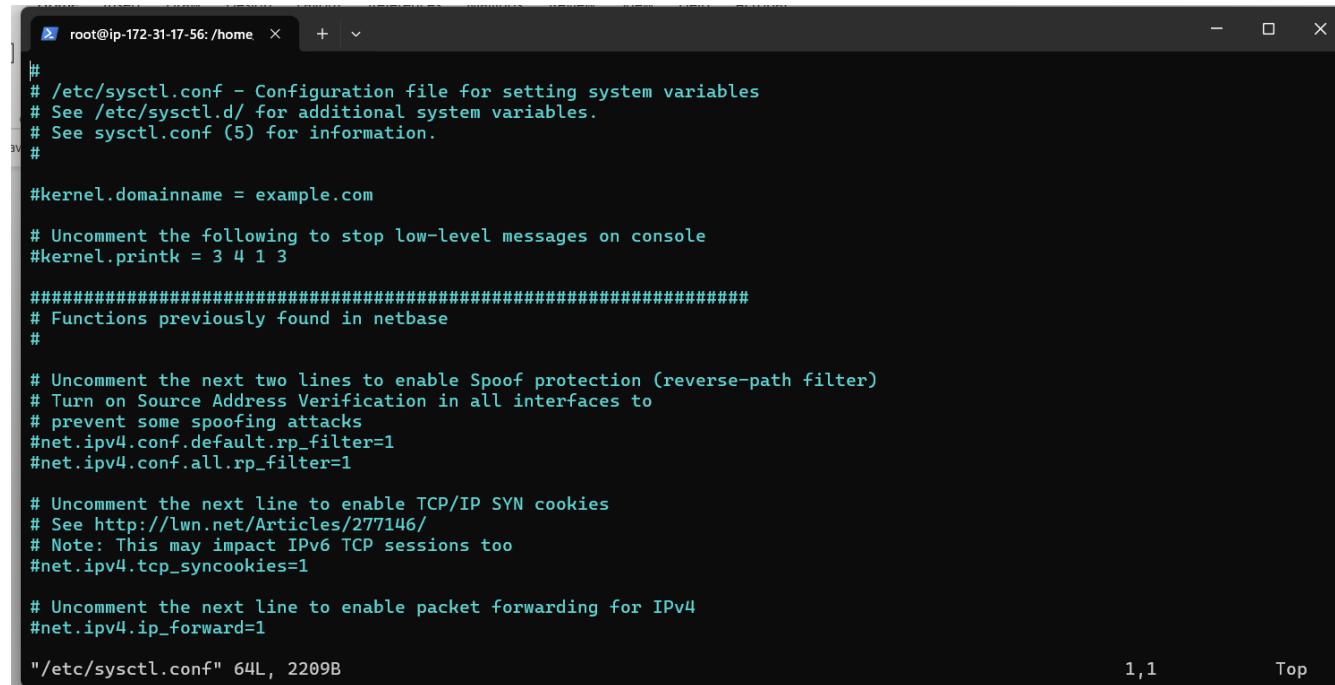
It is up and running. Press “q” to exit this mode

Part 5: Modify Kernel System Limits

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

Edit the sysctl configuration file.

```
sudo vim /etc/sysctl.conf
```



```
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.
#
#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3
#####
# Functions previously found in netbase
#
# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

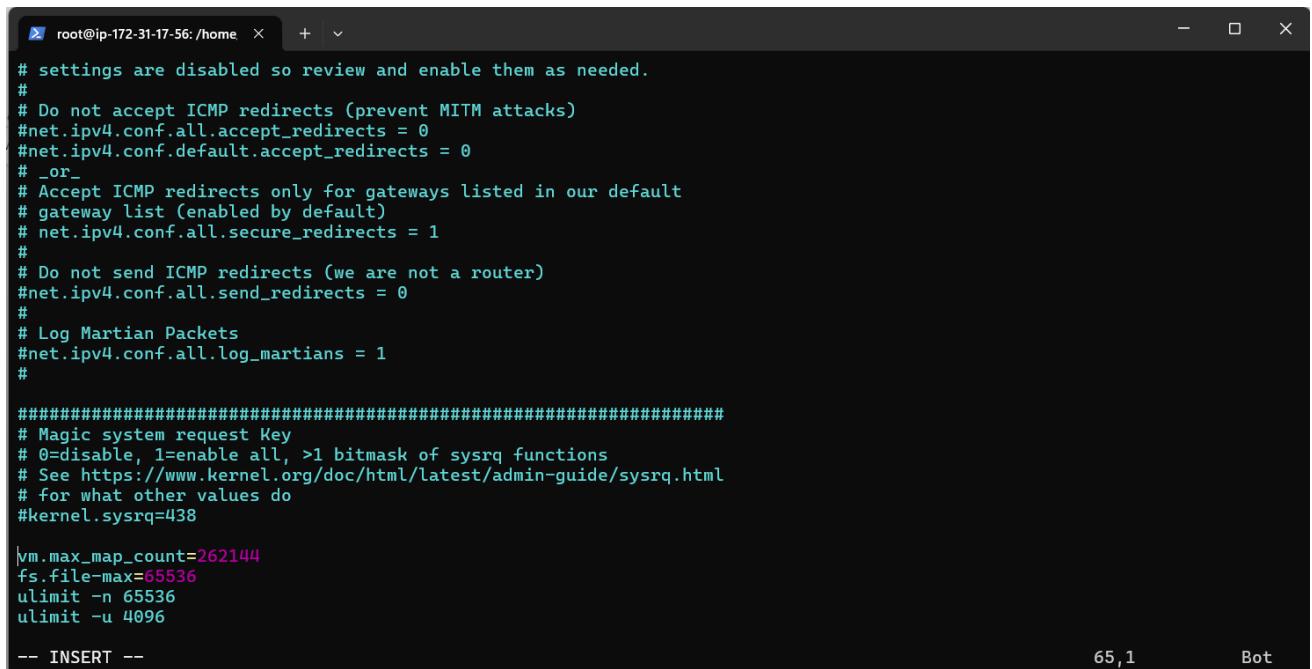
# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1

"/etc/sysctl.conf" 64L, 2209B
```

Add the following lines.

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



```
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv4.conf.default.accept_redirects = 0
#_or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
#net.ipv4.conf.all.send_redirects = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
#
#####
# Magic system request key
# 0=disable, 1=enable all, >1 bitmask of sysrq functions
# See https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html
# for what other values do
#kernel.sysrq=438

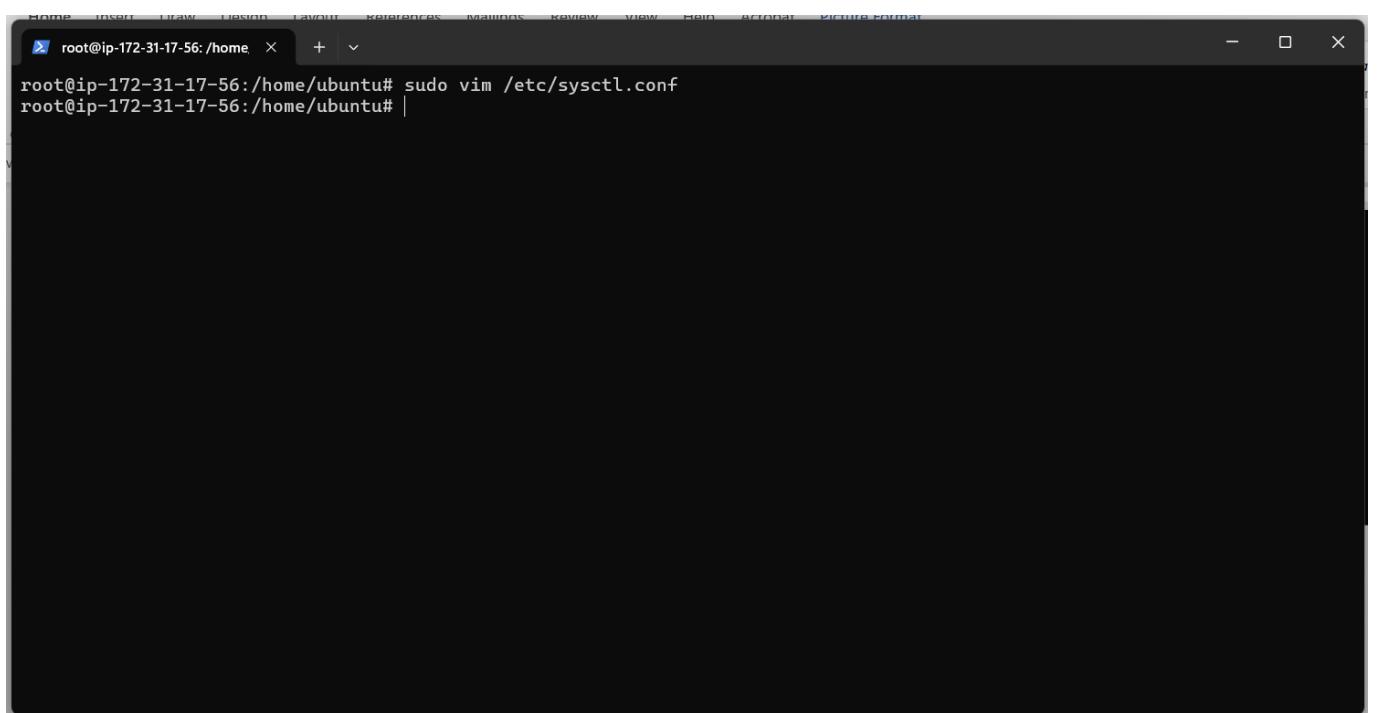
vm.max_map_count=262144
fs.file-max=65536
ulimit -n 65536
ulimit -u 4096

-- INSERT --
```

65,1 Bot

Save and exit the file

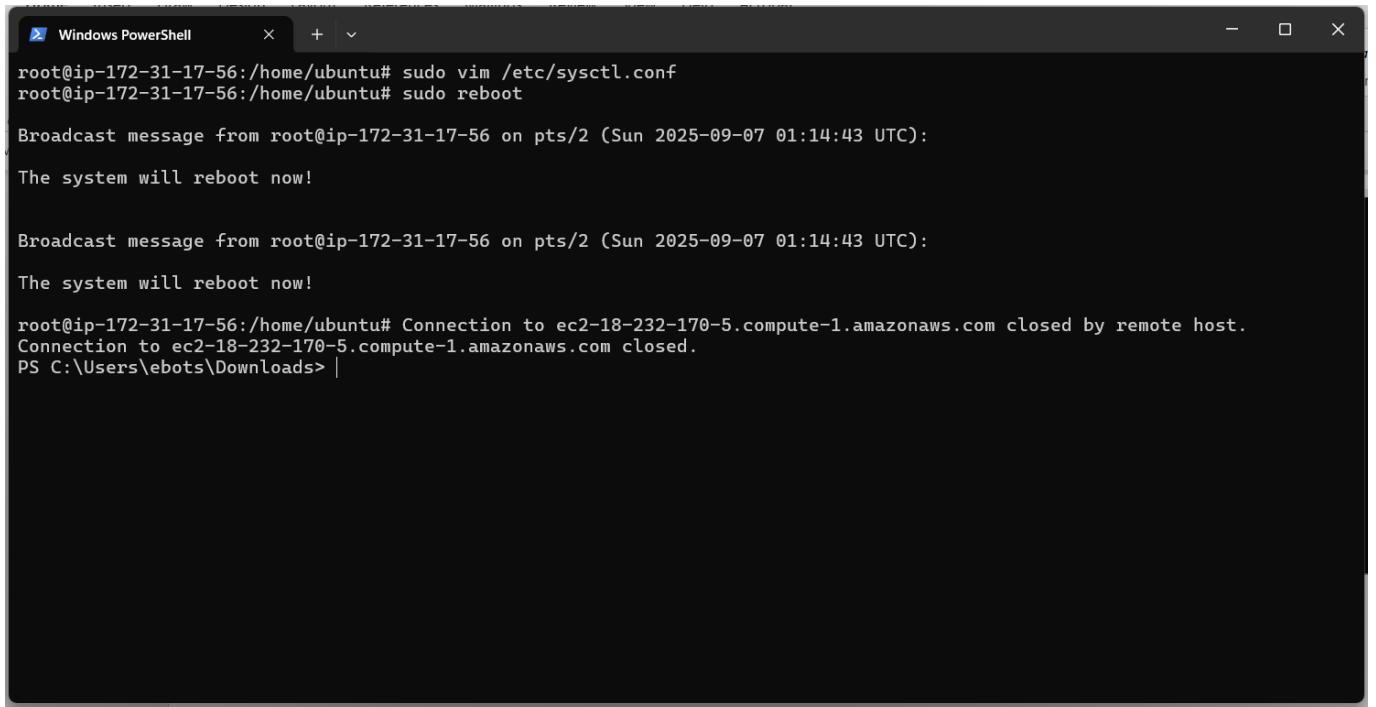
Press ESC, followed by :wq and press ENTER



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/sysctl.conf
root@ip-172-31-17-56:/home/ubuntu# |
```

Reboot the system to apply the changes.

```
sudo reboot
```



A screenshot of a Windows PowerShell window titled "Windows PowerShell". The window shows a root session on an Ubuntu system. The terminal output is as follows:

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/sysctl.conf
root@ip-172-31-17-56:/home/ubuntu# sudo reboot

Broadcast message from root@ip-172-31-17-56 on pts/2 (Sun 2025-09-07 01:14:43 UTC):
The system will reboot now!

Broadcast message from root@ip-172-31-17-56 on pts/2 (Sun 2025-09-07 01:14:43 UTC):
The system will reboot now!

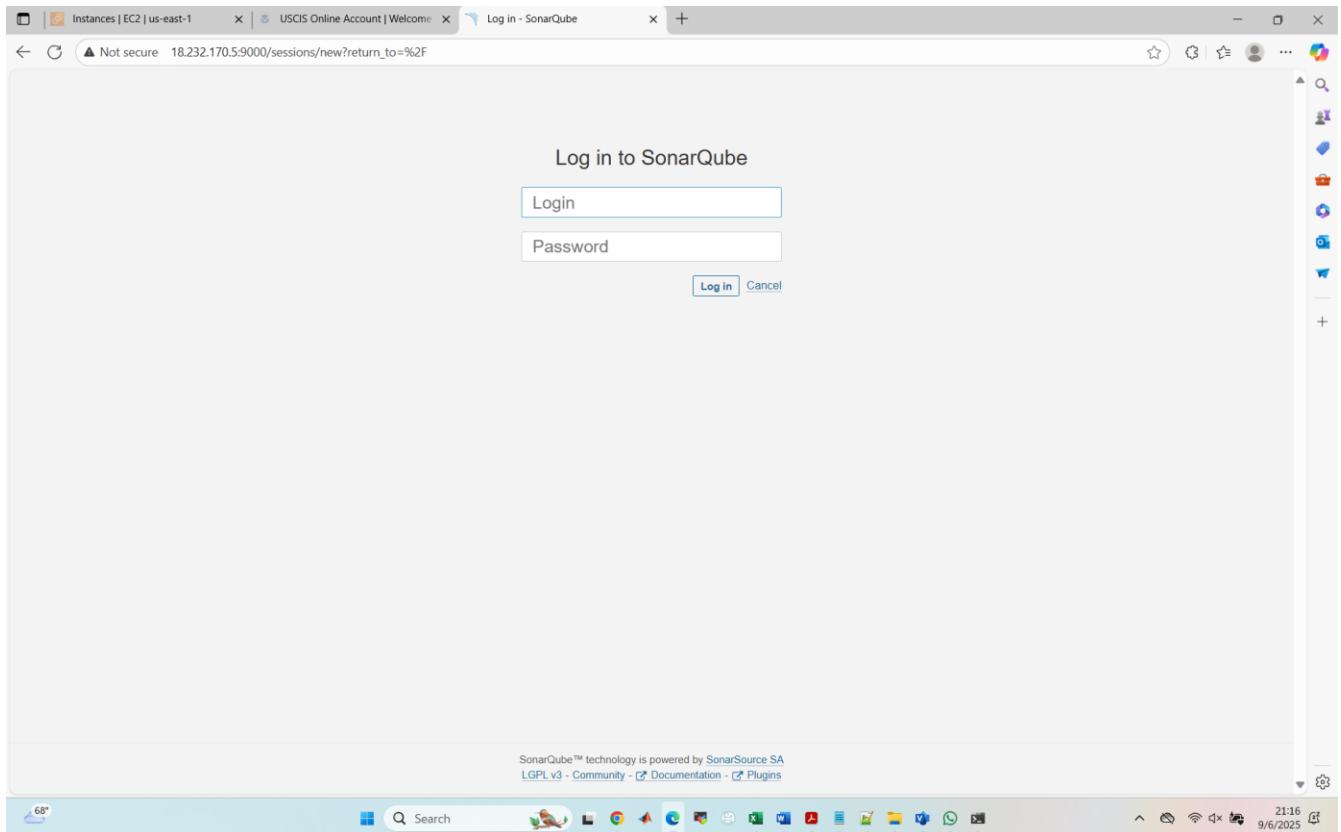
root@ip-172-31-17-56:/home/ubuntu# Connection to ec2-18-232-170-5.compute-1.amazonaws.com closed by remote host.
Connection to ec2-18-232-170-5.compute-1.amazonaws.com closed.
PS C:\Users\ebots\Downloads> |
```

STEP 5: Access SonarQube Web Interface

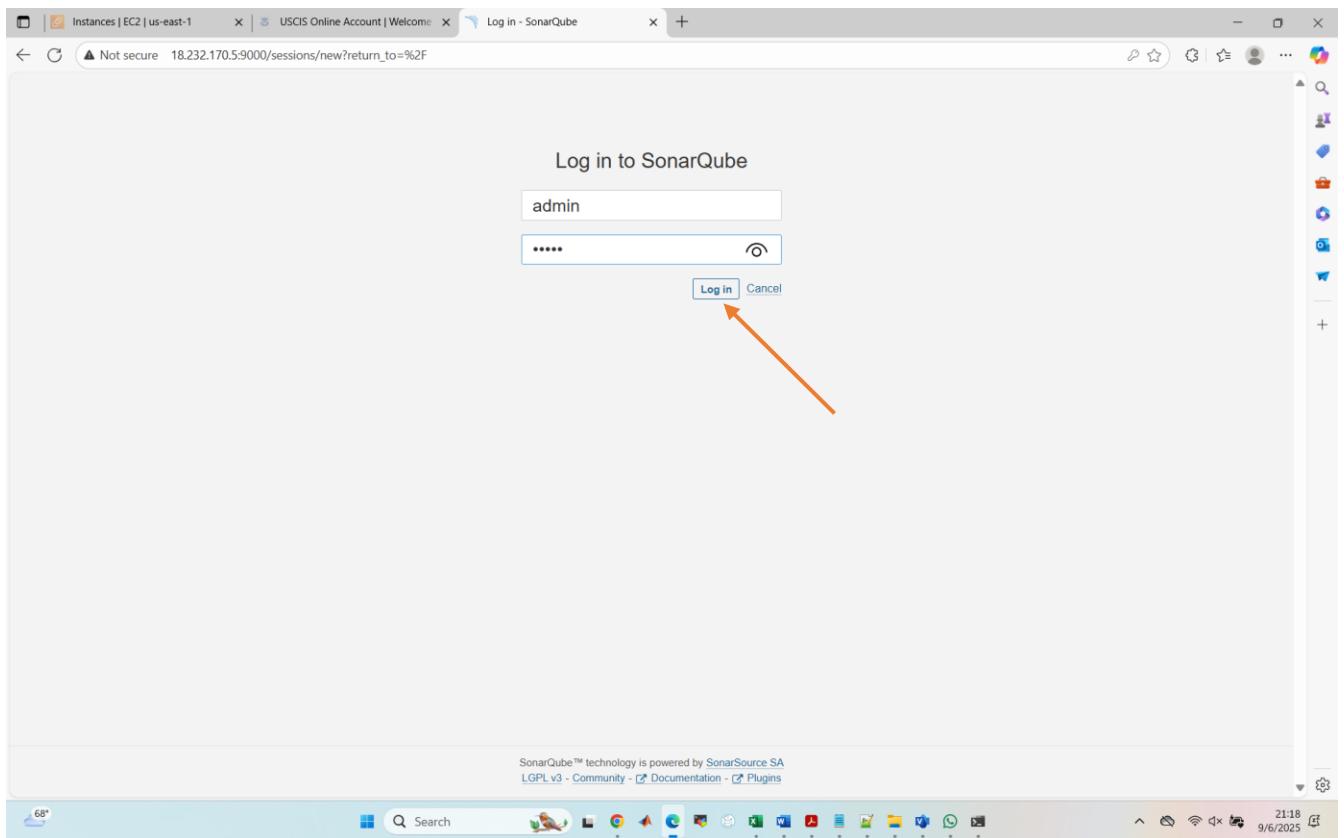
Access SonarQube in a web browser at your server's IP address on port 9000.

For example, <http://Public IPv4 Address:9000>

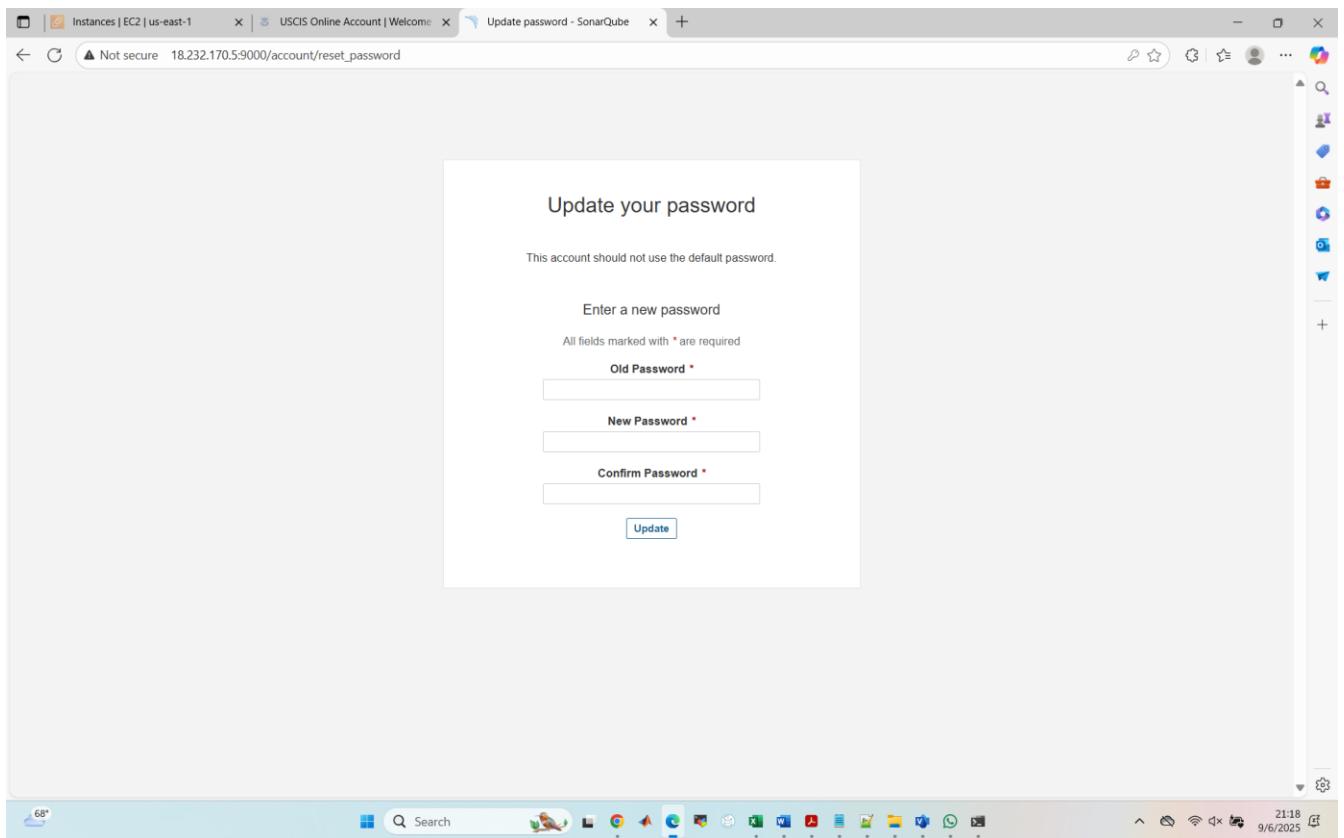
That is **http://18.232.170.5:9000**



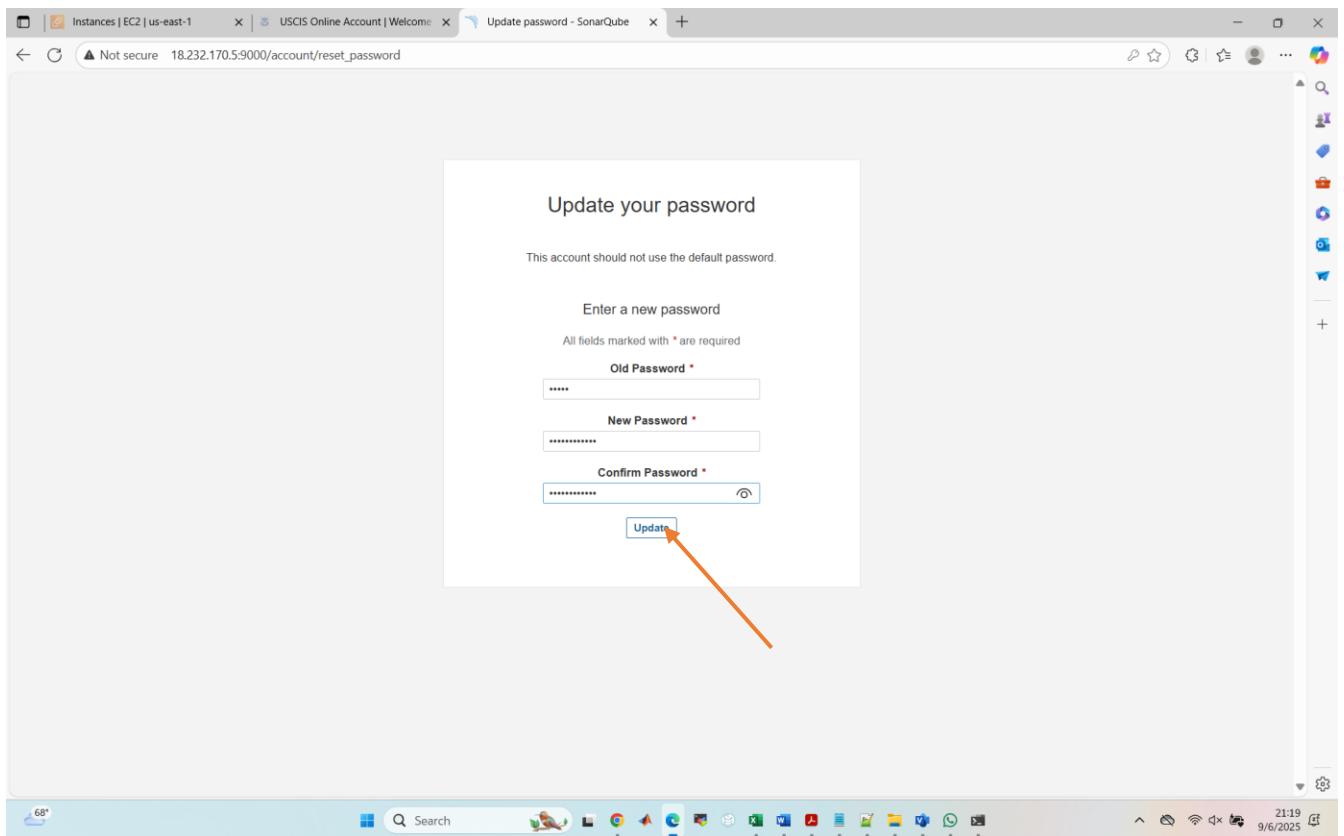
Note: the default username and password are **admin** and **admin** respectively.



Click on “Login”. In the next step, SonarQube will prompt you to change your password. CHANGE THE PASSWORD.



Change the Old password with a New one.



Click on “Update”

The screenshot shows the SonarQube interface for creating a new project. At the top, there are tabs for 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and a search bar. Below the tabs, a message asks, "How do you want to create your project?". It explains that users can benefit from SonarQube's features like repository import and Pull Request decoration by creating a project from their favorite DevOps platform. It then lists five options for importing projects:

- From Azure DevOps**: Represented by a blue icon of a cloud with a gear.
- From Bitbucket Server**: Represented by a blue icon of a cloud with a gear.
- From Bitbucket Cloud**: Represented by a blue icon of a cloud with a gear.
- From GitHub**: Represented by the GitHub logo (a white octocat inside a dark oval).
- From GitLab**: Represented by the GitLab logo (a red and orange stylized animal head).

Below these options, there is a link for "Set up global configuration" for each. Further down, there is a section for manually creating a project, indicated by a "Manually" button next to a double-angle bracket icon (<>).

At the bottom of the page, there is footer text: "SonarQube™ technology is powered by SonarSource SA Community Edition - Version 10.0 (build 68432) - LGPL v3 - Community - Documentation - Plugins - Web API". The browser status bar at the bottom right shows the date as 9/6/2025 and the time as 21:22.

We have successfully installed the SonarQube Community version 10.0 on AWS EC2 Ubuntu.