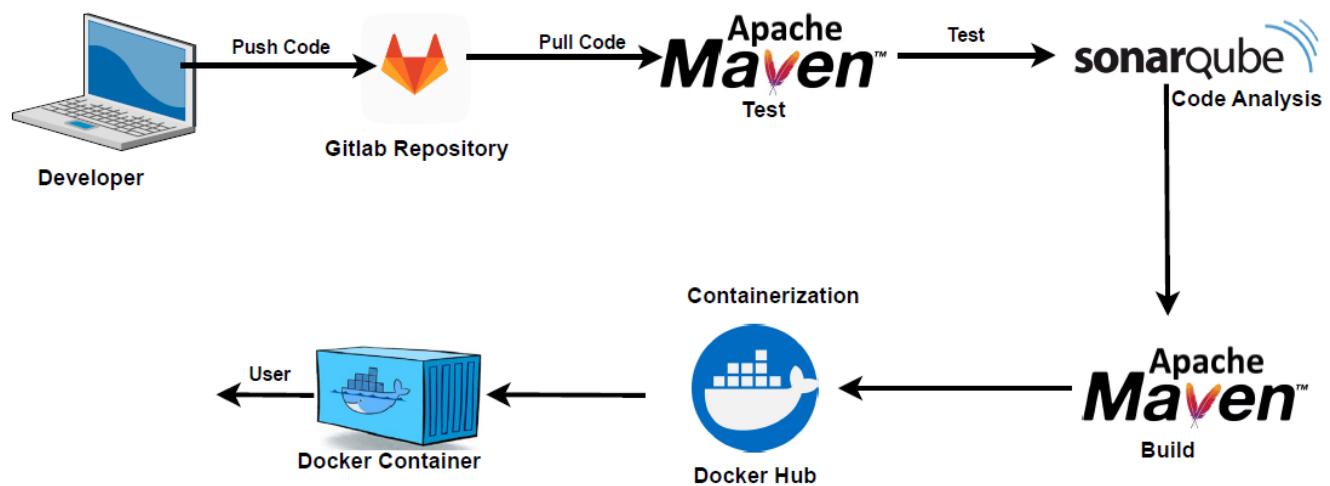


Deploying Java Application using GitLab Pipeline Project with Maven, SonarQube, Docker

In this document, I will demonstrate how to deploy a Java-based application using Gitlab Pipeline. I will use GitLab Runner (Project Runner) on AWS with Docker Executor, SonarQube server on AWS and Docker.

At the end, you will learn how to build a full GitLab-CI pipeline for Java-based applications, with a focus on DevSecOps practices.

Architecture



The developer writes a Java-based code and pushes it to the Gitlab repository that will trigger the Pipeline. The code will be tested with Maven. Next the code analysis is done with SonarQube. Then the analyzed code is build using Maven. Finally, the image is built and containerized using Docker.

IMPLEMENTATION

The pipeline includes four key jobs:

1. Test Job for the Java application using Maven
2. Code Quality Check Job using SonarQube
3. Build and Package the Java Application Job using Maven
4. Containerize the Application and Push Image to Docker Hub Job using Docker

We will also walk through setting up the infrastructure, including:

- **GitLab Runner (Project Runner) on AWS** with Docker Executor
- **SonarQube server on AWS** using Docker for Java code quality analysis

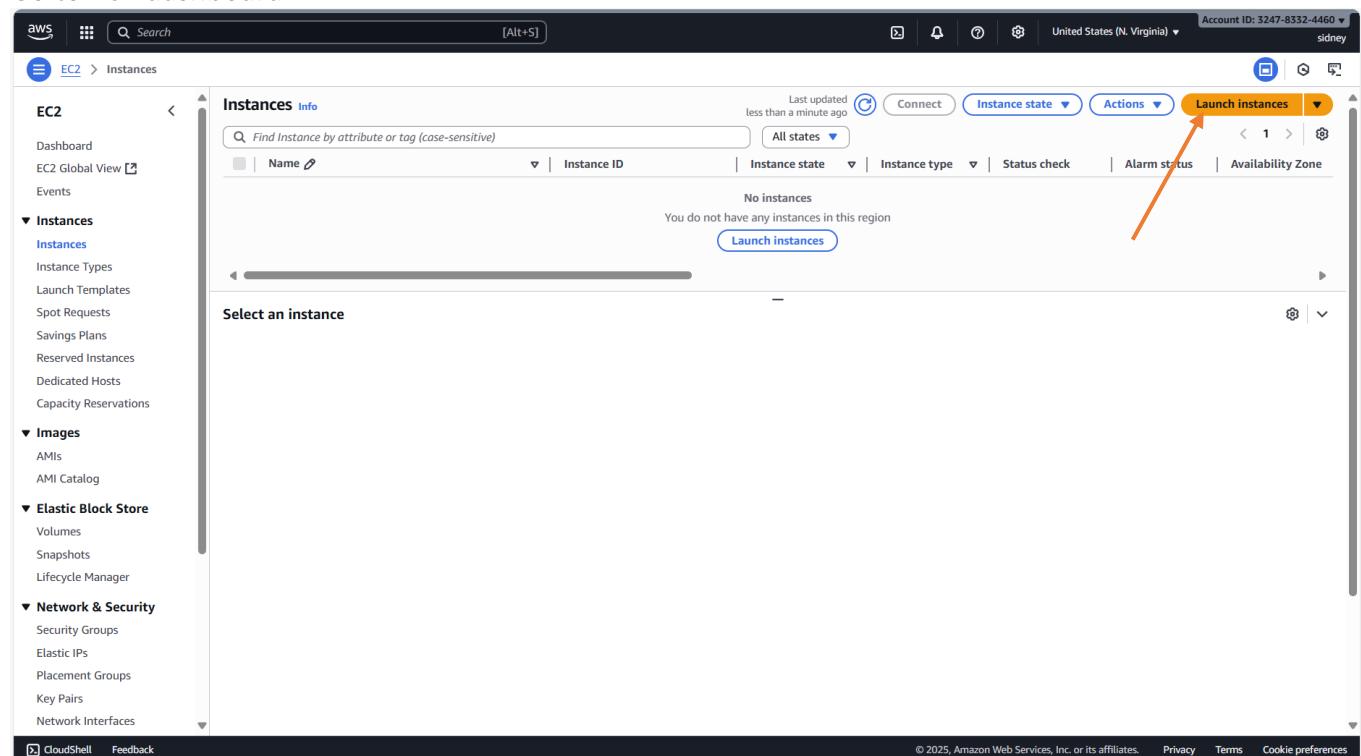
You will learn how to write pipeline scripts for each of these jobs. The Test and Build jobs for the Java app will run on the Project Runner, while the SonarQube code quality check and Containerize job will run on GitLab's Shared Runner.

STEP 1: Create an Ubuntu EC2 instance and Connect to the Instance

We have to create an Ubuntu EC2 instance, namely Gitlab-Server, add ports 22, 80, 43, and 9000 for SSH, HTTP, HTTPS and SonarQube respectively to the instances. Then SSH connect to the Ubuntu EC2 instance.

Part 1: Create an ubuntu EC2 instance

Go to EC2 dashboard



Click on “Launch Instance”

Prepared by Sidney Smith

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 service. In the 'Name and tags' section, the 'Name' field contains 'e.g. My Web Server'. An orange arrow points from this field to the 'Ubuntu' button in the 'Application and OS Images (Amazon Machine Image)' section below. The 'Ubuntu' button is highlighted with a red border.

Give the instance the name “Gitlab-Server”

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 service. In the 'Name and tags' section, the 'Name' field now contains 'Gitlab-Server'. The 'Ubuntu' button in the 'Application and OS Images (Amazon Machine Image)' section is still highlighted with a red border.

On “Application and OS Images (Amazon Machine Image)”, select “Ubuntu”

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Search our full catalog including 1000s of application and OS images

Recents Quick Start

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Debian

ubuntu® Microsoft Red Hat SUSE debian

Search Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type Free tier eligible ▾

ami-0360c520857e3138f (64-bit (x86)) / ami-026fccd88446aa0bf (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Ubuntu Server 24.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

Architecture	AMI ID	Publish Date	Username	Verified provider
64-bit (x86) ▾	ami-0360c520857e3138f	2025-08-21	ubuntu	Verified provider

Scroll down to “Instance Type”, select “t2.medium”

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0499 USD per Hour On-Demand Linux base pricing: 0.0464 USD per Hour
On-Demand RHEL base pricing: 0.0752 USD per Hour On-Demand Windows base pricing: 0.0644 USD per Hour
On-Demand SUSE base pricing: 0.1464 USD per Hour

All generations Compare instance types

Additional costs apply for AMIs with pre-installed software

Then scroll down to “Key Pair”

▼ Instance type [Info](#) | [Get advice](#)

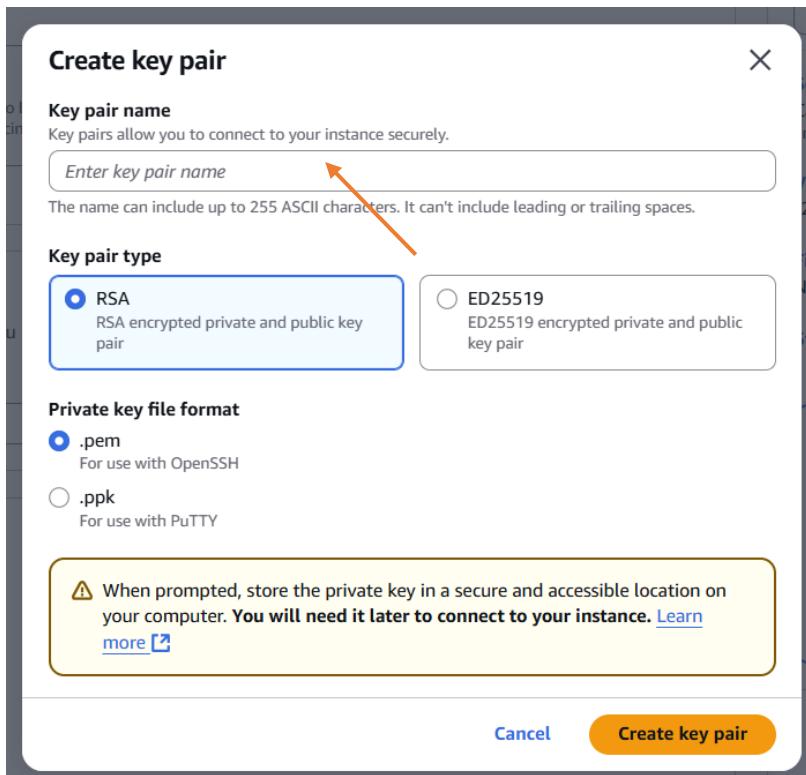
Instance type

t2.micro
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

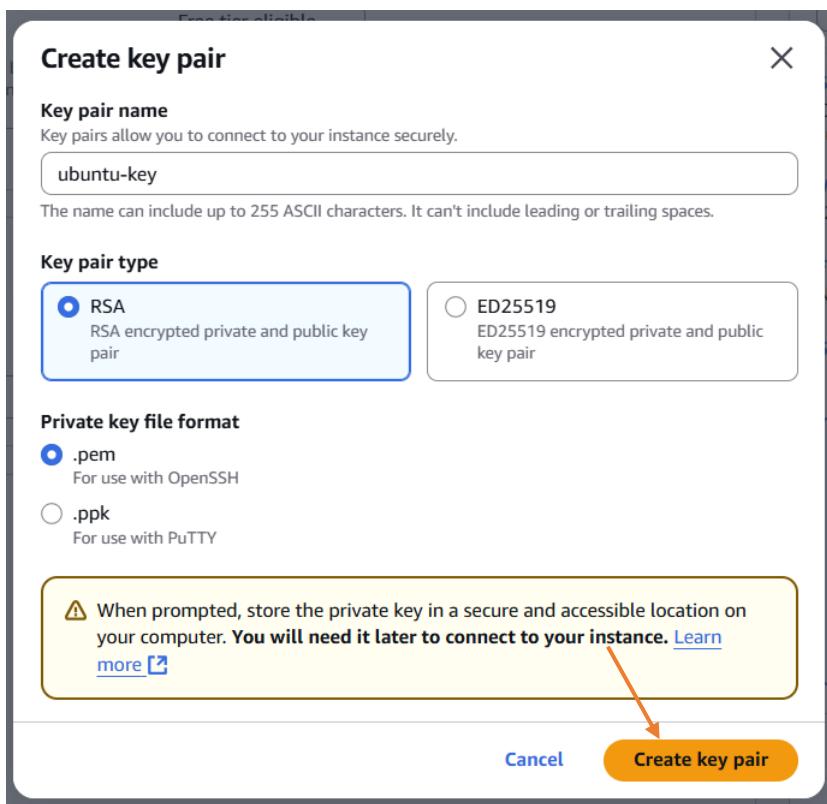
All generations Compare instance types

Additional costs apply for AMIs with pre-installed software

Click on “create new key pair”



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



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

ubuntu-key

[Create new key pair](#)

Scroll down to “Network Settings”

▼ 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-54' with the following rules:

[Allow SSH traffic from](#)

Helps you connect to your instance

Anywhere

▼

[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 “Configuration Storage” and make the value “30GiB”

▼ 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](#)

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 30 GiB

ⓘ **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.

[Cancel](#)



[Launch instance](#)

[Preview code](#)

Click on “Launch Instance”

Prepared by Sidney Smith

The screenshot shows the AWS EC2 'Launch an instance' success page. At the top, there's a green success message: 'Success Successfully initiated launch of instance (i-097749be6331b247f)'. Below it, a 'Launch log' button is visible. A 'Next Steps' section contains a search bar and a numbered list of 6 items: 'Create billing and free tier usage alerts', 'Connect to your instance', 'Connect an RDS database', 'Create EBS snapshot policy', 'Manage detailed monitoring', and 'Create Load Balancer'. Each item has a corresponding blue 'Create' or 'Learn more' button. The bottom of the page includes standard AWS navigation links like CloudShell, Feedback, and a footer with copyright information.

Click on “Instance”

The screenshot shows the AWS EC2 Instances page. On the left, a sidebar lists categories: EC2 (Dashboard, EC2 Global View, Events), Instances (Instances, 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), and Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces). The main area displays 'Instances (1) Info' with a table showing one instance: 'Gitlab-Server' (Instance ID: i-097749be6331b247f, Instance state: Running, Instance type: t2.micro, Status check: Initializing, Alarm status: View alarms, Availability Zone: us-east-1d). Below the table is a 'Select an instance' dropdown menu. The bottom of the page includes standard AWS navigation links like CloudShell, Feedback, and a footer with copyright information.

The instance is initializing, wait for it to pass the “2/2 check”

Prepared by Sidney Smith

The screenshot shows the AWS EC2 Instances page. On the left, a navigation sidebar lists various EC2-related options like Instances, Images, and Network & Security. The main content area displays a table titled 'Instances (1) Info'. A single row is present, representing a running instance named 'Gitlab-Server' with the ID 'i-097749be6331b247f'. The instance is listed as 'Running' with a status check of '2/2 checks passed'. At the top right of the table, there are buttons for 'Connect', 'Instance state', 'Actions', and 'Launch instances'. Below the table, a section titled 'Select an instance' is visible.

The EC2 instance have passed the “2/2 check”

Part 2: Add Port 9000 for SonarQube in the Inbound rule

We have to add Port 9000 to be used to access SonarQube on the browser.

This screenshot is identical to the one above, showing the AWS EC2 Instances page with a single running instance named 'Gitlab-Server'. An orange arrow points to the checkbox next to the instance name in the list table. The rest of the interface and information are the same as the first screenshot.

Select the EC2 instance

Prepared by Sidney Smith

This screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Instances, Images, Elastic Block Store, and Network & Security. The main area displays a table of instances, with one row selected: "Gitlab-Server" (Instance ID: i-0ba7b381f77ce602c). Below the table, there's a detailed view for the selected instance. A red arrow points to the "Security" tab in the top navigation bar of this detailed view.

Click on “Security” tab

This screenshot shows the same AWS EC2 Instances page as the previous one, but with a different focus. A red arrow points to the "Security groups" section under the "Security details" heading. This section lists a single security group: "sg-0608924d3e2da1d98 (launch-wizard-55)".

Click on “Security Group”

Prepared by Sidney Smith

The screenshot shows the AWS EC2 Security Groups page. On the left, there's a sidebar with navigation links for EC2, Instances, Images, Elastic Block Store, and Network & Security. The main content area displays a security group named "sg-0608924d3e2da1d98 - launch-wizard-55". The "Details" section shows the security group ID, owner, description, and VPC ID. Below this, there are tabs for Inbound rules, Outbound rules, Sharing - new, VPC associations - new, and Tags. The Inbound rules section lists three rules: one for SSH (TCP port 22), one for HTTP (TCP port 80), and one for HTTPS (TCP port 443). An orange arrow points from the text "Click on ‘Edit Inbound Rules’" to the "Edit inbound rules" button at the top right of the page.

Click on “Edit Inbound Rules”

The screenshot shows the "Edit inbound rules" page for the same security group. It displays the existing three rules and a table for adding new ones. The "Add rule" button is highlighted with an orange arrow. At the bottom right, there are buttons for "Cancel", "Preview changes", and "Save rules".

Click on “Add Rule”

The screenshot shows the "Edit inbound rules" page with a new row for a custom TCP rule. The "Port range" field is highlighted with an orange arrow. The "Add rule" button is also visible at the bottom left. At the bottom right, there are buttons for "Cancel", "Preview changes", and "Save rules".

Add Port 9000

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-0597052c04e8c6642	SSH	TCP	22	Custom	0.0.0.0/0
sgr-07ecd7e33348d9f29	HTTP	TCP	80	Custom	0.0.0.0/0
sgr-05de55647cd9f70f4	HTTPS	TCP	443	Custom	0.0.0.0/0
-	Custom TCP	TCP	9000	Custom	0.0.0.0/0

Add rule

Cancel Preview changes Save rules

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

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-0597052c04e8c6642	SSH	TCP	22	Custom	0.0.0.0/0
sgr-07ecd7e33348d9f29	HTTP	TCP	80	Custom	0.0.0.0/0
sgr-05de55647cd9f70f4	HTTPS	TCP	443	Custom	0.0.0.0/0
-	Custom TCP	TCP	9000	Anywhere...	sonarqube

Add rule

Cancel Preview changes Save rules

Click on “Save Rules”

Prepared by Sidney Smith

The screenshot shows the AWS EC2 Security Groups console. A green success message at the top states: "Inbound security group rules successfully modified on security group (sg-0608924d3e2da1d98 | launch-wizard-55) Details". The main page title is "sg-0608924d3e2da1d98 - launch-wizard-55". The "Details" section shows the security group name is "launch-wizard-55", the security group ID is "sg-0608924d3e2da1d98", the description is "launch-wizard-55 created 2025-10-08T02:53:58.753Z", and the VPC ID is "vpc-0128e9209eaef1c37". The owner is listed as "324783324460". The inbound rules count is 4 permission entries, and the outbound rules count is 1 permission entry. Below this, there are tabs for "Inbound rules", "Outbound rules", "Sharing - new", "VPC associations - new", and "Tags". The "Inbound rules" tab is selected, displaying a table with four rows of rules. The columns include Security group rule ID, IP version, Type, Protocol, Port range, and Source. The rules are:

Security group rule ID	IP version	Type	Protocol	Port range	Source
sgr-0597052c04e8c6642	IPv4	SSH	TCP	22	0.0.0.0/0
sgr-07ecd7e33348df9f29	IPv4	HTTP	TCP	80	0.0.0.0/0
sgr-02b888954ff3924bd	IPv4	Custom TCP	TCP	9000	0.0.0.0/0
sgr-05de55647cd9f70f4	IPv4	HTTPS	TCP	443	0.0.0.0/0

Part 3: Connect to instance through SSH

The screenshot shows the AWS EC2 Instances console. The main title is "Instances (1) Info". There is a single instance listed: "Gitlab-Server" with Instance ID "i-097749be6331b247f". The instance state is "Running", the instance type is "t2.micro", and it has passed 2/2 checks. The availability zone is "us-east-1d". An orange arrow points from the text "Select the EC2 instance" below to the checkbox next to the instance name "Gitlab-Server". The left sidebar contains navigation links for EC2, Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and CloudShell.

Select the EC2 instance

Prepared by Sidney Smith

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with navigation links like Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and CloudShell. The main area displays a table of instances. One instance, 'Gitlab-Server' (i-097749be6331b247f), is selected and shown in detail. The instance is running, has an 't2.micro' type, and is in the 'us-east-1d' availability zone. A red arrow points to the 'Connect' button in the top right of the instance card.

Click on “Connect”

The screenshot shows the 'Connect to instance' dialog for the selected EC2 instance. It provides instructions for connecting using an SSH client. The 'SSH client' tab is selected. The steps listed are:

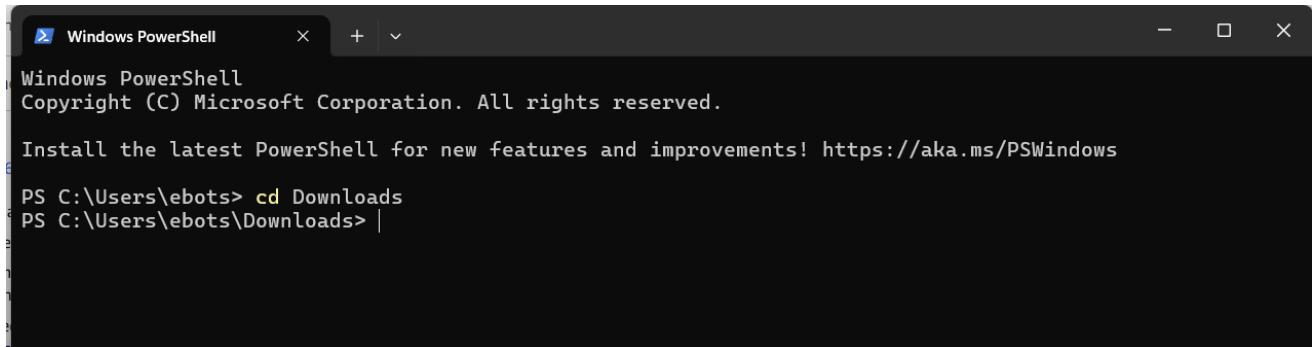
1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is `ubuntu-key.pem`.
3. Run this command, if necessary, to ensure your key is not publicly viewable.
`chmod 400 "ubuntu-key.pem"`
4. Connect to your instance using its Public DNS:
`ssh -i "ubuntu-key.pem" ubuntu@ec2-3-87-90-116.compute-1.amazonaws.com`

An orange arrow points to the command line example at the bottom of the dialog.

Copy the command:

```
ssh -i "ubuntu-key.pem" ubuntu@ec2-3-87-90-116.compute-1.amazonaws.com
```

Open PowerShell and navigate to the folder where the “key pair” .pem file is saved. It is saved in my “Downloads” folder



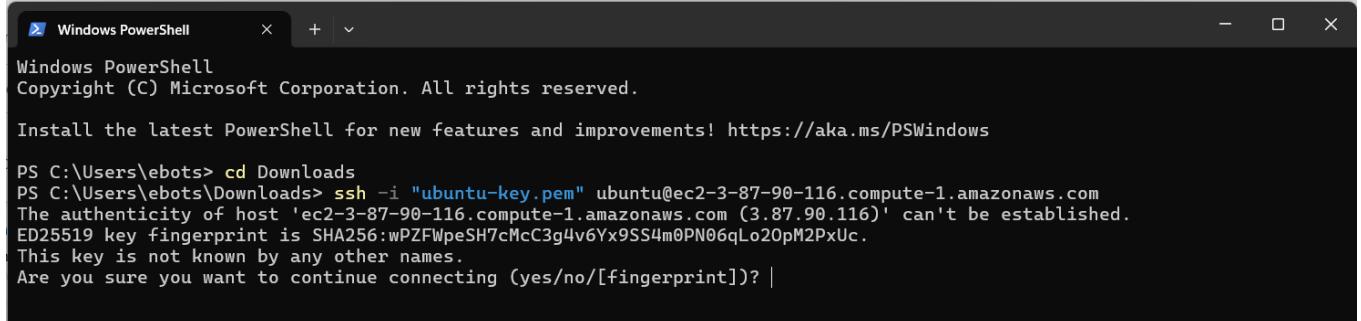
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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

PS C:\Users\ebots> cd Downloads
PS C:\Users\ebots\Downloads> |
```

Then run the command:

```
ssh -i "ubuntu-key.pem" ubuntu@ec2-3-87-90-116.compute-1.amazonaws.com
```

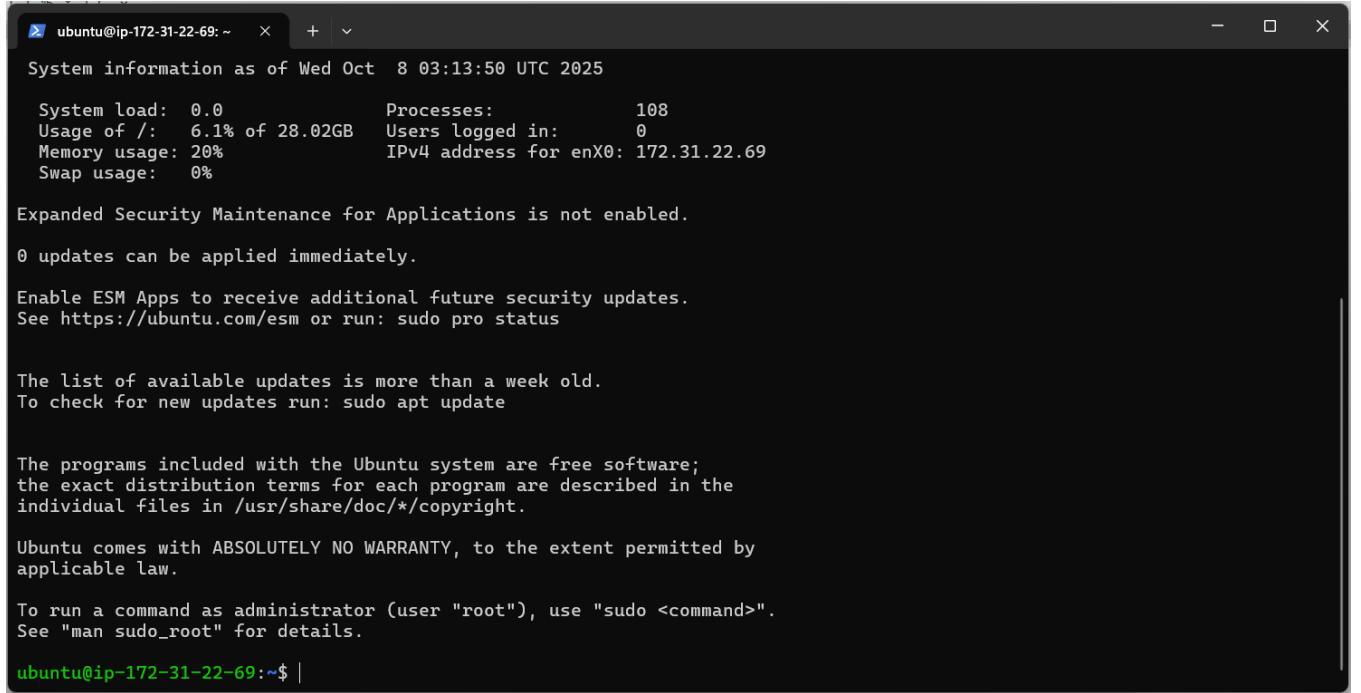


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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 "ubuntu-key.pem" ubuntu@ec2-3-87-90-116.compute-1.amazonaws.com
The authenticity of host 'ec2-3-87-90-116.compute-1.amazonaws.com (3.87.90.116)' can't be established.
ED25519 key fingerprint is SHA256:wPZFwpeSH7cMcC3g4v6Yx9SS4m0PN06ql020pM2PxUc.
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-172-31-22-69:~> System information as of Wed Oct 8 03:13:50 UTC 2025
System load: 0.0      Processes:          108
Usage of /: 6.1% of 28.02GB   Users logged in:    0
Memory usage: 20%           IPv4 address for enX0: 172.31.22.69
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.

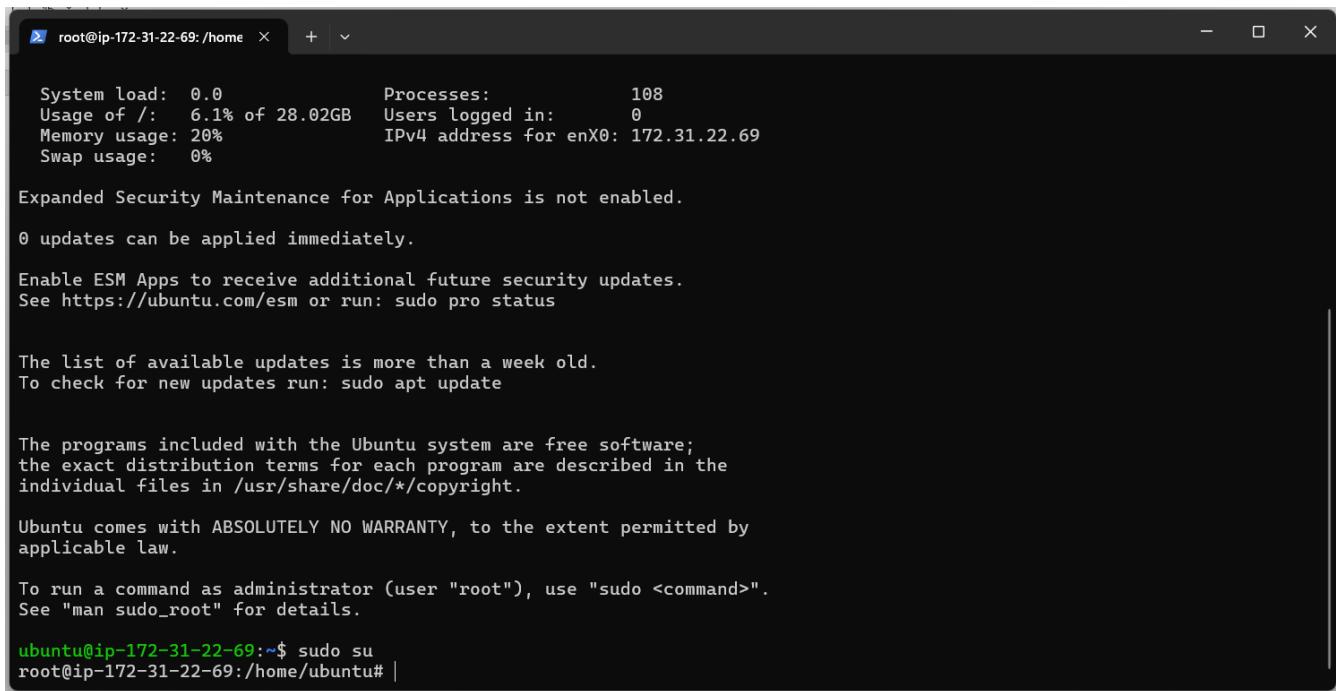
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-22-69:~$ |
```

Grant root user privilege using the command:

```
sudo su
```

Prepared by Sidney Smith



```
root@ip-172-31-22-69:/home  +  ▾

System load: 0.0      Processes: 108
Usage of /: 6.1% of 28.02GB  Users logged in: 0
Memory usage: 20%          IPv4 address for enX0: 172.31.22.69
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-172-31-22-69:~$ sudo su
root@ip-172-31-22-69:/home/ubuntu# |
```

Clear the terminal using the command:

```
clear
```



```
root@ip-172-31-22-69:/home  +  ▾

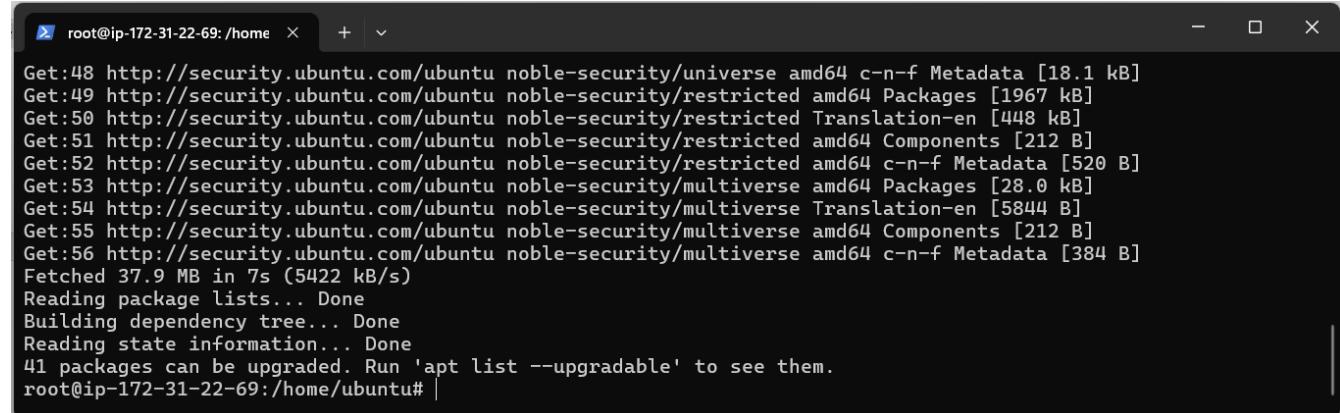
root@ip-172-31-22-69:/home/ubuntu# |
```

STEP 2: Install Java JDK 17 on EC2 server

We will now install Java JDK 17 on our server.

First, we update the packages using the command:

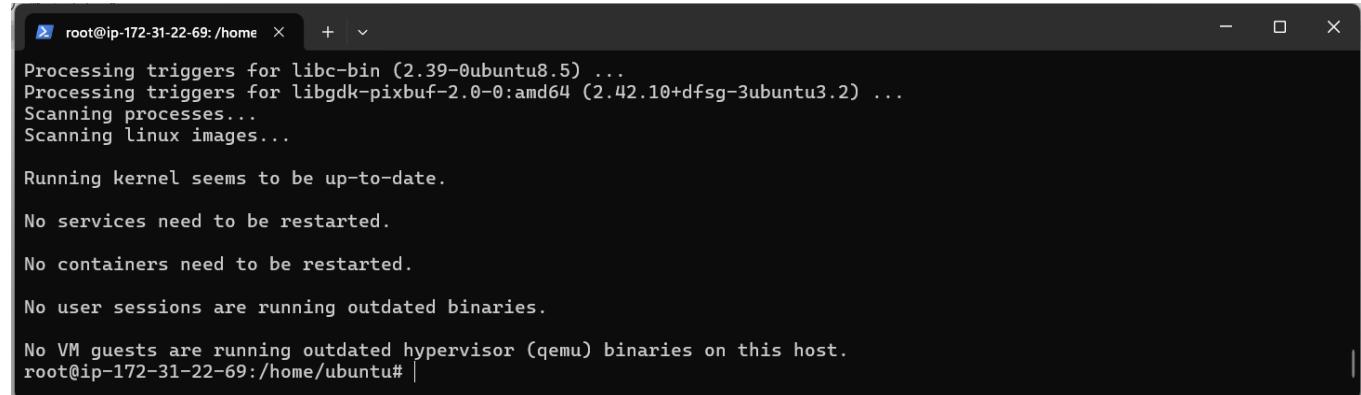
```
sudo apt update
```



```
root@ip-172-31-22-69:/home x + v
Get:48 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [18.1 kB]
Get:49 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1967 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [448 kB]
Get:51 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [520 B]
Get:53 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [28.0 kB]
Get:54 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [5844 B]
Get:55 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:56 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [384 B]
Fetched 37.9 MB in 7s (5422 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
41 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-22-69:/home/ubuntu# |
```

Install Java JDK 17 using the command:

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



```
root@ip-172-31-22-69:/home x + v
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Processing triggers for libgdk-pixbuf2-0:amd64 (2.42.10+dfsg-3ubuntu3.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

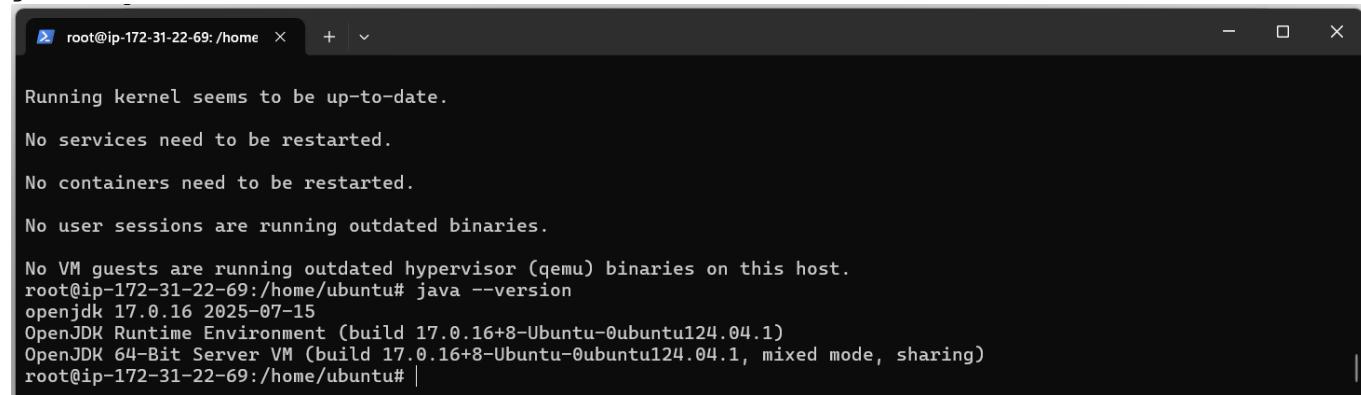
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-22-69:/home/ubuntu# |
```

Verify Java is Installed by running the command:

```
java --version
```



```
root@ip-172-31-22-69:/home x + v
Running kernel seems to be up-to-date.

No services need to be restarted.

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-172-31-22-69:/home/ubuntu# java --version
openjdk 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-172-31-22-69:/home/ubuntu# |
```

STEP 3: Install and Configure Maven with Jenkins for Code Compile

The next step is Code compile using Maven. Let us install Maven. Go to this link:

Go to the URL: <https://maven.apache.org/download.cgi> the link for the “Binary tar.gz archive” file.

The screenshot shows the Apache Maven Project download page at <https://maven.apache.org/download.cgi>. The main content is titled "Downloading Apache Maven 3.9.11". A sidebar on the left has a "Downloads" section selected. In the "Files" section, there are three options: "Binary tar.gz archive", "Binary zip archive", and "Source tar.gz archive". The "Binary tar.gz archive" option is highlighted with a red arrow pointing to it. A context menu is open over this option, with "Copy link address" being the selected item. The menu also includes "Save link as...", "Inspect", "Checksums", and "Signature". The "Checksums" and "Signature" sections show SHA-512 checksums and PGP signatures respectively for each file type.

Select “Copy Link address”

<https://dlcdn.apache.org/maven/maven-3/3.9.11/binaries/apache-maven-3.9.11-bin.tar.gz>

Part 1: Download the Maven Binaries

Then run the following commands to download Maven

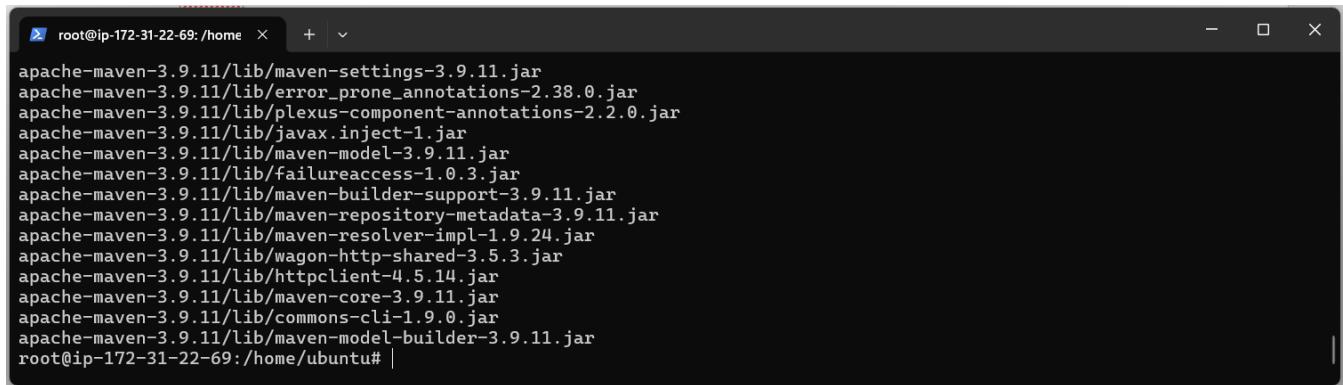
wget [Link]

```
 wget https://dlcdn.apache.org/maven/maven-3/3.9.11/binaries/apache-maven-3.9.11-bin.tar.gz
```

```
root@ip-172-31-22-69:/home/+ ~ % root@ip-172-31-22-69:/home/ubuntu# https://dlcdn.apache.org/maven/maven-3/3.9.11/binaries/apache-maven-3.9.11-bin.tar.gz
bash: https://dlcdn.apache.org/maven/maven-3/3.9.11/binaries/apache-maven-3.9.11-bin.tar.gz: No such file or directory
root@ip-172-31-22-69:/home/ubuntu# |
```

Part 2: Untar the Downloaded file

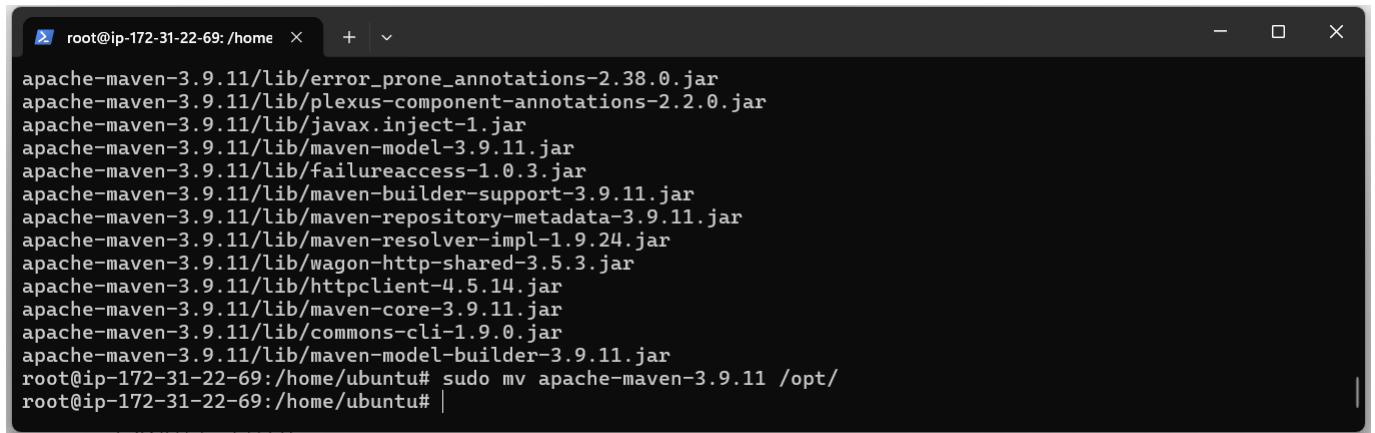
```
tar -xvf apache-maven-3.9.11-bin.tar.gz
```



A terminal window titled 'root@ip-172-31-22-69: /home' showing a list of Maven jar files in the '/lib' directory. The files listed are: apache-maven-3.9.11/lib/maven-settings-3.9.11.jar, apache-maven-3.9.11/lib/error_prone_annotations-2.38.0.jar, apache-maven-3.9.11/lib/plexus-component-annotations-2.2.0.jar, apache-maven-3.9.11/lib/javax.inject-1.jar, apache-maven-3.9.11/lib/maven-model-3.9.11.jar, apache-maven-3.9.11/lib/failureaccess-1.0.3.jar, apache-maven-3.9.11/lib/maven-builder-support-3.9.11.jar, apache-maven-3.9.11/lib/maven-repository-metadata-3.9.11.jar, apache-maven-3.9.11/lib/maven-resolver-impl-1.9.24.jar, apache-maven-3.9.11/lib/wagon-http-shared-3.5.3.jar, apache-maven-3.9.11/lib/httpclient-4.5.14.jar, apache-maven-3.9.11/lib/maven-core-3.9.11.jar, apache-maven-3.9.11/lib/commons-cli-1.9.0.jar, apache-maven-3.9.11/lib/maven-model-builder-3.9.11.jar. The prompt at the bottom is 'root@ip-172-31-22-69:/home/ubuntu# |'

Part 3: Untar the mvn package to the /opt folder

```
sudo mv apache-maven-3.9.11 /opt/
```

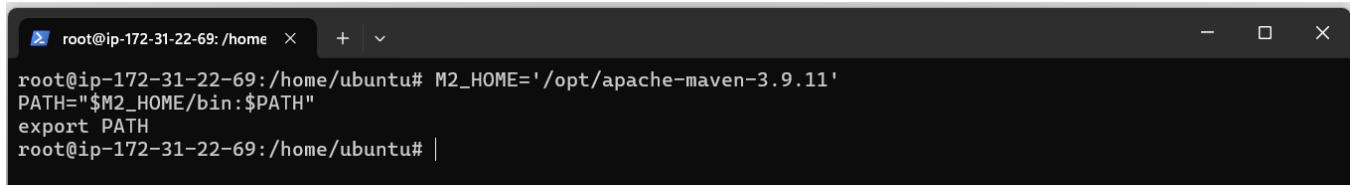


A terminal window titled 'root@ip-172-31-22-69: /home' showing the command 'sudo mv apache-maven-3.9.11 /opt/' being run. The output shows the same list of Maven jar files as the previous window, followed by the command itself and the prompt 'root@ip-172-31-22-69:/home/ubuntu# |'

Part 4: Setting M2_HOME and Path Variables

Add the following lines to the user profile file (.profile).

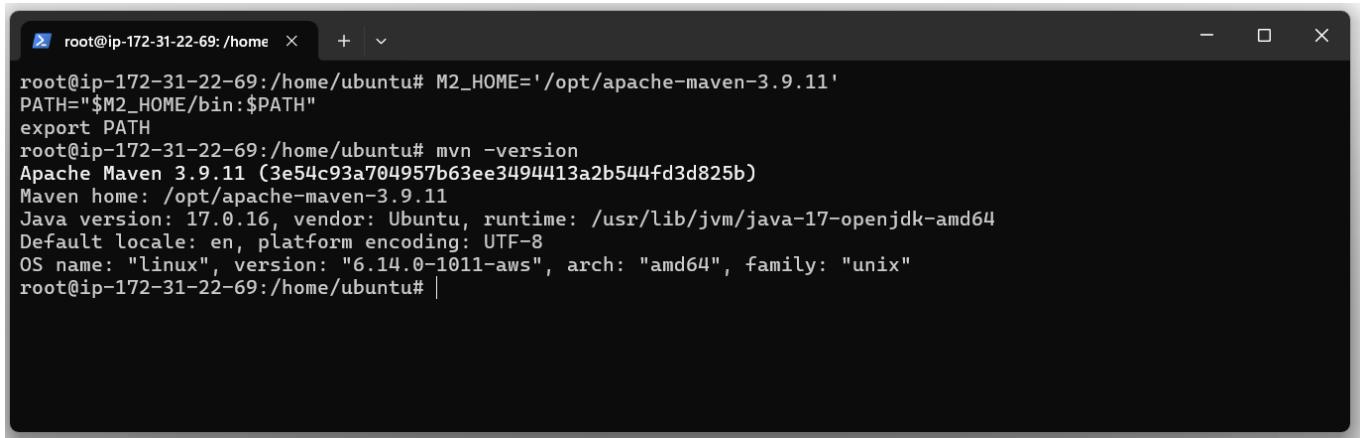
```
M2_HOME='/opt/apache-maven-3.9.11'  
PATH="$M2_HOME/bin:$PATH"  
export PATH
```



A terminal window titled 'root@ip-172-31-22-69: /home' showing the addition of environment variables. The commands entered are 'M2_HOME='/opt/apache-maven-3.9.11'', 'PATH="\$M2_HOME/bin:\$PATH"', and 'export PATH'. The prompt at the bottom is 'root@ip-172-31-22-69:/home/ubuntu# |'

Part 5: Verify the Maven installation by using the command:

```
mvn -version
```



```
root@ip-172-31-22-69:/home/ubuntu# M2_HOME='/opt/apache-maven-3.9.11'  
PATH="$M2_HOME/bin:$PATH"  
export PATH  
root@ip-172-31-22-69:/home/ubuntu# mvn -version  
Apache Maven 3.9.11 (3e54c93a704957b63ee3494413a2b544fd3d825b)  
Maven home: /opt/apache-maven-3.9.11  
Java version: 17.0.16, vendor: Ubuntu, runtime: /usr/lib/jvm/java-17-openjdk-amd64  
Default locale: en, platform encoding: UTF-8  
OS name: "linux", version: "6.14.0-1011-aws", arch: "amd64", family: "unix"  
root@ip-172-31-22-69:/home/ubuntu# |
```

Maven version 3.9.11 has been installed.

STEP 4: Install Docker

Installing Docker on Ubuntu can be done by setting up the official Docker repository and then installing the necessary packages. This method ensures you receive the latest stable versions and updates.

Part 1: Update Package Index and Install Prerequisites

```
sudo apt update  
sudo apt install -y apt-transport-https ca-certificates curl software-properties-common
```

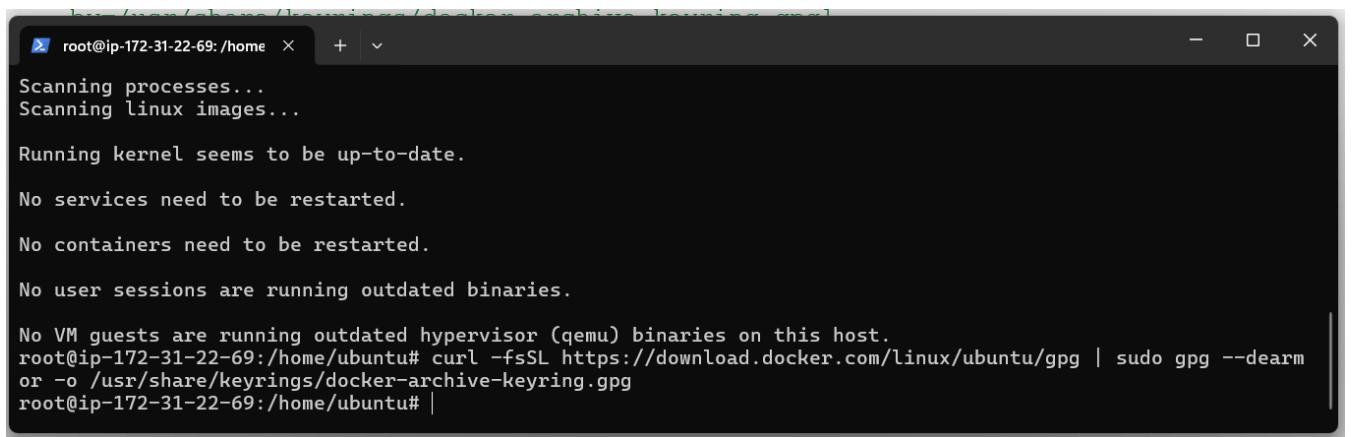


A terminal window titled 'root@ip-172-31-22-69:/home' showing the output of package management commands. It includes status messages like 'Unpacking apt-transport-https', 'Setting up apt-transport-https', and various scanning and kernel status messages.

```
Unpacking apt-transport-https (2.8.3) ...  
Setting up apt-transport-https (2.8.3) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
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-172-31-22-69:/home/ubuntu# |
```

Part 2: Add Docker's GPG Key

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
```



A terminal window titled 'root@ip-172-31-22-69:/home' showing the output of a curl command to download Docker's GPG key. It includes status messages like 'Scanning processes...', 'Running kernel seems to be up-to-date.', and a final command line showing the key was added to the keyring.

```
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
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-172-31-22-69:/home/ubuntu# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg  
root@ip-172-31-22-69:/home/ubuntu# |
```

Part 3: Add the Docker Repository

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```



```
root@ip-172-31-22-69:/home ~ + - X
Running kernel seems to be up-to-date.
No services need to be restarted.
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-172-31-22-69:/home/ubuntu# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearm
or -o /usr/share/keyrings/docker-archive-keyring.gpg
root@ip-172-31-22-69:/home/ubuntu# echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings
/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /
etc/apt/sources.list.d/docker.list > /dev/null
root@ip-172-31-22-69:/home/ubuntu# |
```

Part 4: Install Docker Engine

```
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```



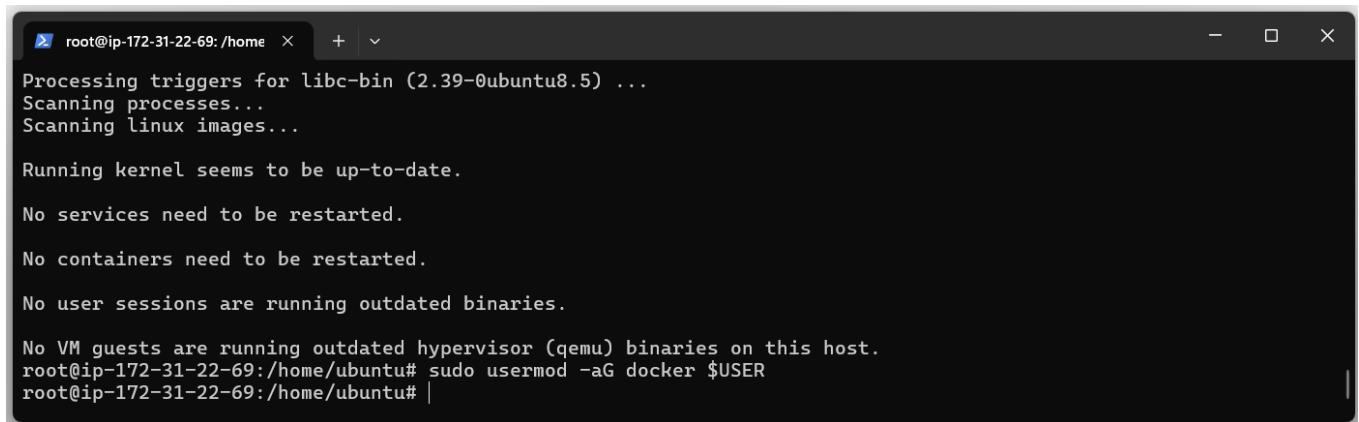
```
root@ip-172-31-22-69:/home ~ + - X
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.
No services need to be restarted.
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-172-31-22-69:/home/ubuntu# |
```

Part 5: Add Your User to the Docker Group (to run Docker commands without sudo):

```
sudo usermod -aG docker $USER
```



```
root@ip-172-31-22-69:/home ~ + - X
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning linux images...

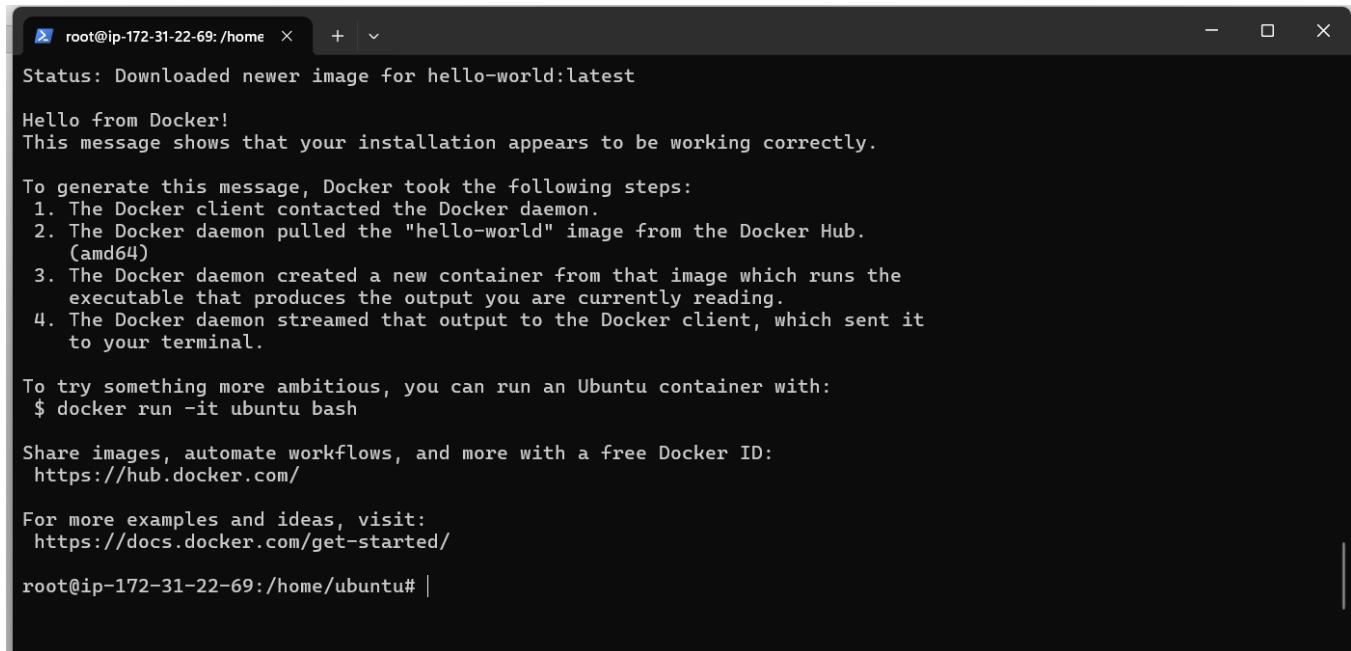
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-22-69:/home/ubuntu# sudo usermod -aG docker $USER
root@ip-172-31-22-69:/home/ubuntu# |
```

You will need to log out and log back in (or restart your system) for this change to take effect.

Part 6: Verify the Installation

```
docker run hello-world
```



The screenshot shows a terminal window titled "root@ip-172-31-22-69:/home". The terminal displays the following text:

```
Status: Downloaded newer image for hello-world:latest
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
root@ip-172-31-22-69:/home/ubuntu# |
```

STEP 5: Install SonarQube on EC2 Server

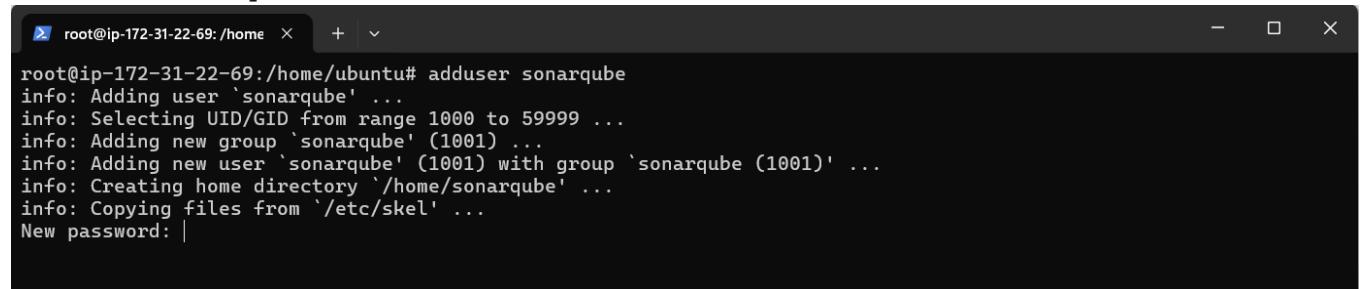
Now, we have to do “**Static Code Analysis**” using SonarQube. Installing SonarQube on an Ubuntu EC2 instance involves several key steps.

we are setting up our sonar on the same server as Jenkins and this requires opening an additional port for SonarQube to run. SonarQube on default runs on port 9000

Part 1: Create a SonarQube user

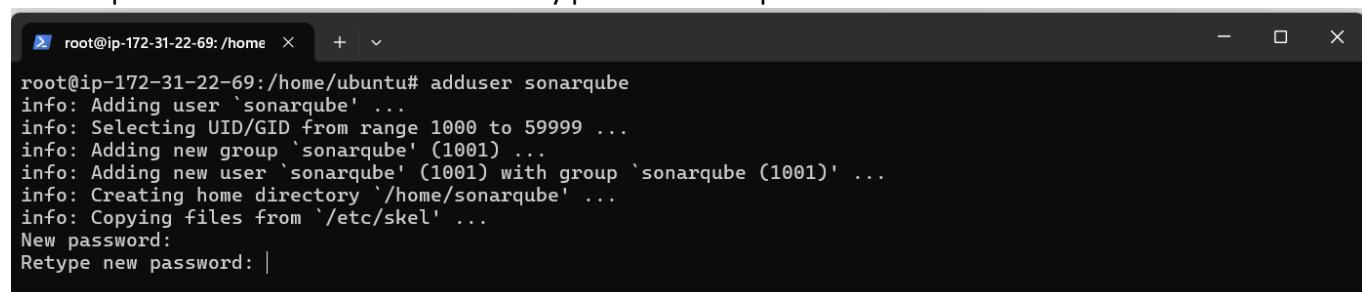
First, we have to create a SonarQube user using the command:

```
adduser sonarqube
```



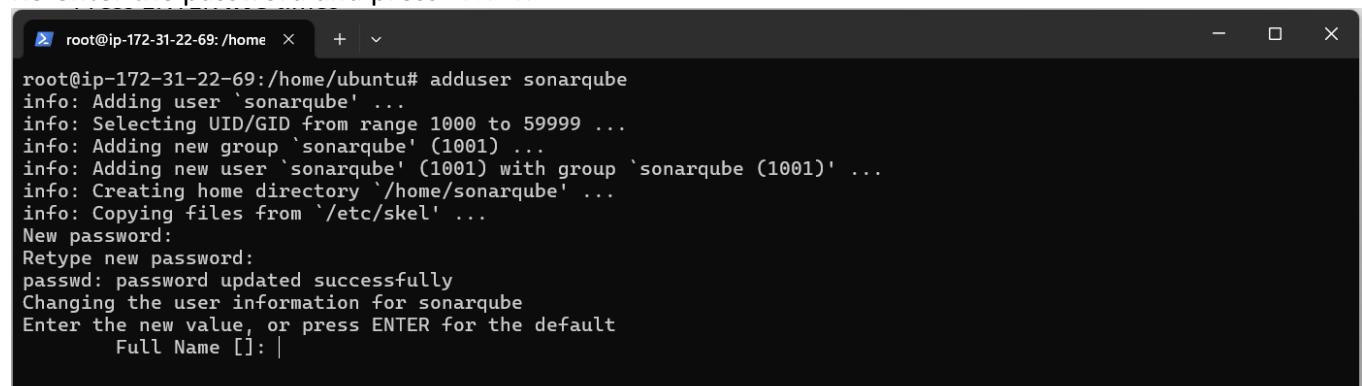
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user `sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `sonarqube' (1001) ...
info: Adding new user `sonarqube' (1001) with group `sonarqube (1001)' ...
info: Creating home directory `/home/sonarqube' ...
info: Copying files from `/etc/skel' ...
New password: |
```

Enter a password. I will use “**admin**” as my password and press Enter



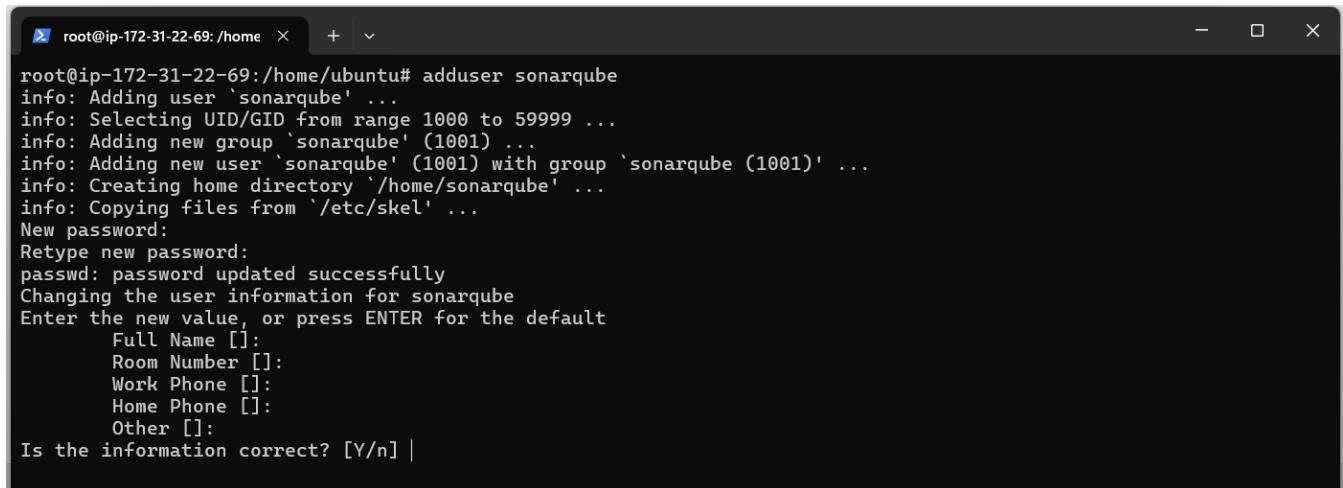
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user `sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `sonarqube' (1001) ...
info: Adding new user `sonarqube' (1001) with group `sonarqube (1001)' ...
info: Creating home directory `/home/sonarqube' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password: |
```

Re-enter the password and press ENTER



```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user `sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `sonarqube' (1001) ...
info: Adding new user `sonarqube' (1001) with group `sonarqube (1001)' ...
info: Creating home directory `/home/sonarqube' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for sonarqube
Enter the new value, or press ENTER for the default
Full Name []: |
```

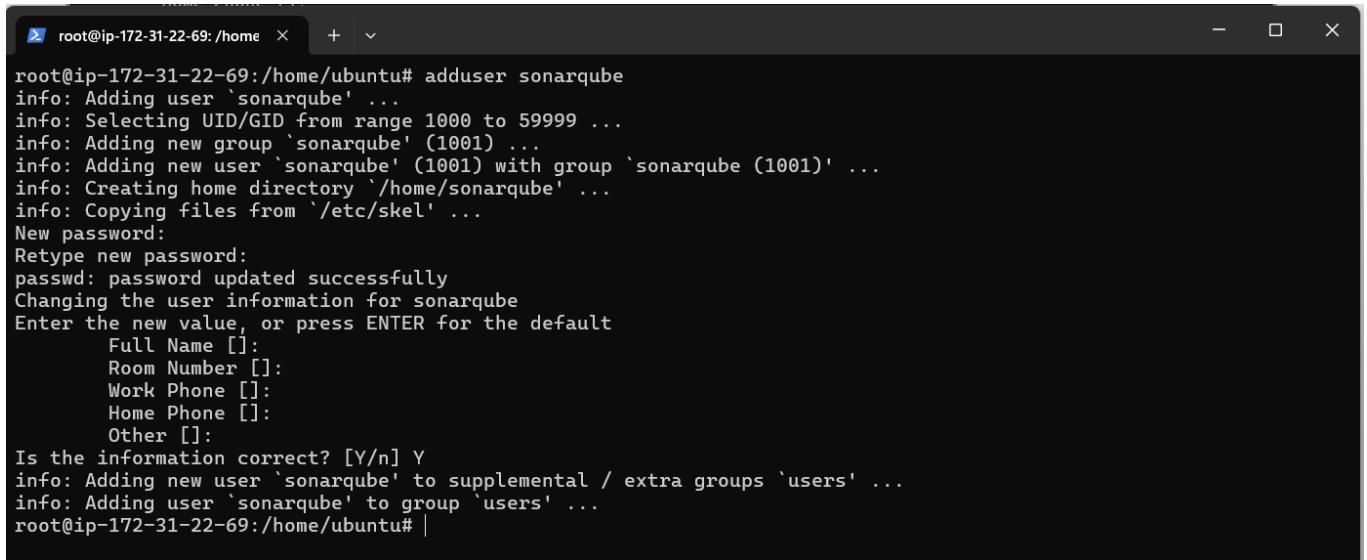
Press ENTER **five** times



A terminal window titled "root@ip-172-31-22-69:/home" showing the process of adding a new user named "sonarqube". The terminal displays the following command and its execution:

```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user `sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `sonarqube' (1001) ...
info: Adding new user `sonarqube' (1001) with group `sonarqube (1001)' ...
info: Creating home directory `/home/sonarqube' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for sonarqube
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] |
```

Type "Y" and press ENTER



A terminal window titled "root@ip-172-31-22-69:/home" showing the completion of the user addition process. The terminal displays the following command and its execution:

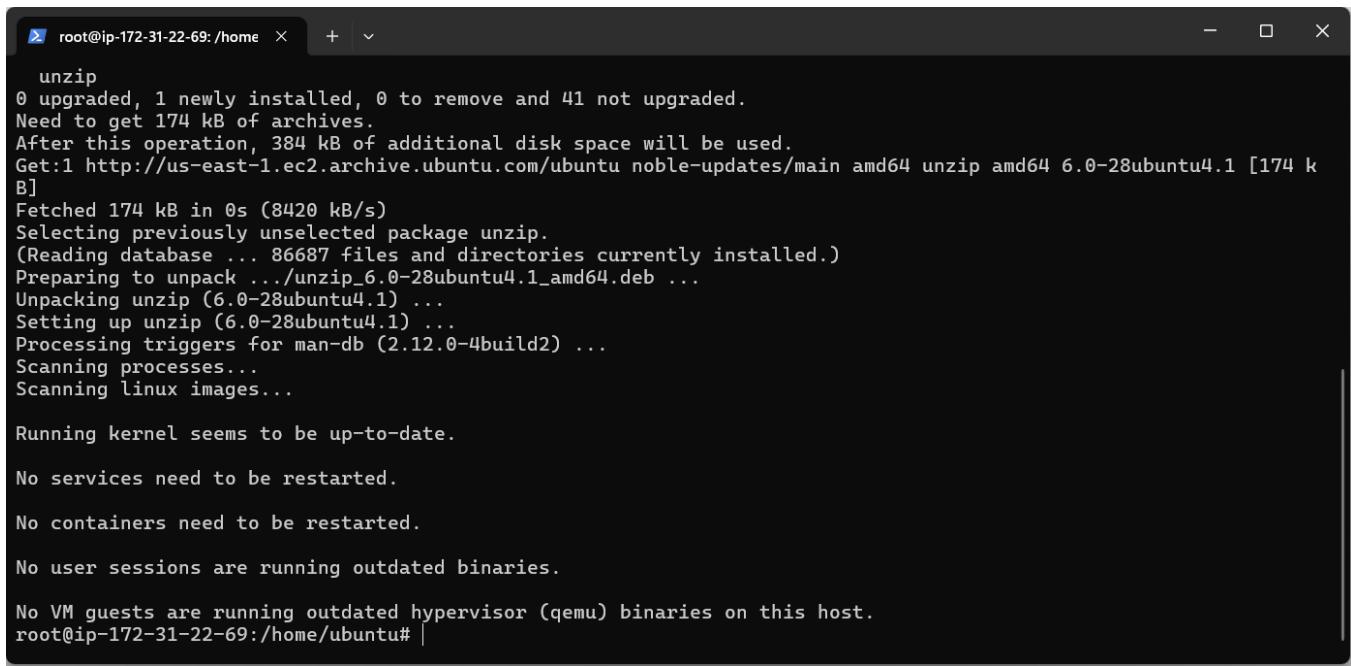
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user `sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `sonarqube' (1001) ...
info: Adding new user `sonarqube' (1001) with group `sonarqube (1001)' ...
info: Creating home directory `/home/sonarqube' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for sonarqube
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] Y
info: Adding new user `sonarqube' to supplemental / extra groups `users' ...
info: Adding user `sonarqube' to group `users' ...
root@ip-172-31-22-69:/home/ubuntu# |
```

The user has been added

Part 2: Install unzip

We have to install unzip. This will be used to unzip the SonarQube zip file we are going to download.

apt install unzip



```
root@ip-172-31-22-69:/home x + v
unzip
0 upgraded, 1 newly installed, 0 to remove and 41 not upgraded.
Need to get 174 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 unzip amd64 6.0-28ubuntu4.1 [174 kB]
Fetched 174 kB in 0s (8420 kB/s)
Selecting previously unselected package unzip.
(Reading database ... 86687 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4.1_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4.1) ...
Setting up unzip (6.0-28ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

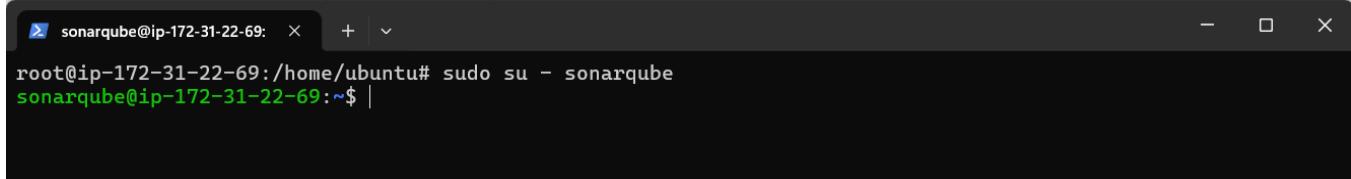
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-22-69:/home/ubuntu#
```

Part 3: Switch to the user

Switch to the SonarQube user using the command:

```
sudo su - sonarqube
```

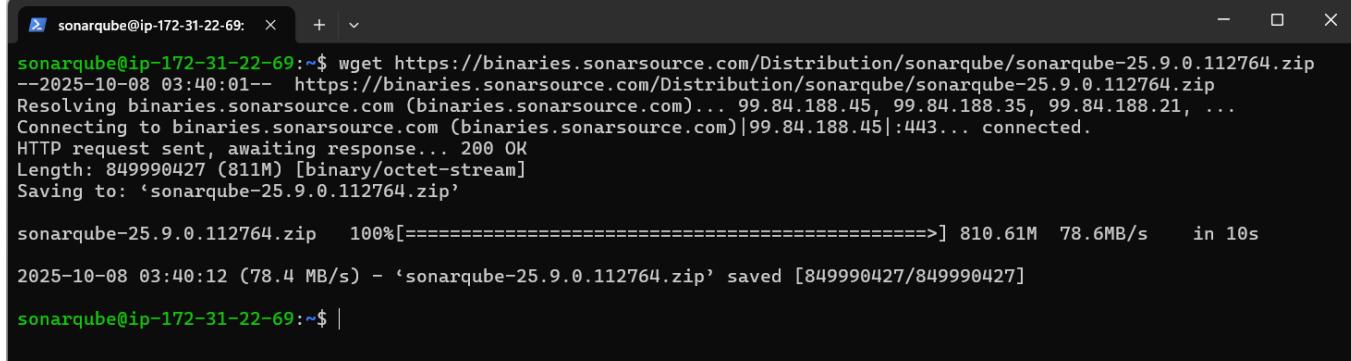


```
sonarqube@ip-172-31-22-69: ~ x + v
root@ip-172-31-22-69:/home/ubuntu# sudo su - sonarqube
sonarqube@ip-172-31-22-69:~$ |
```

Part 4: Download SonarQube:

Download SonarQube using the command

```
wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-25.9.0.112764.zip
```



```
sonarqube@ip-172-31-22-69:~$ wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-25.9.0.112764.zip
--2025-10-08 03:40:01-- https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-25.9.0.112764.zip
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 99.84.188.45, 99.84.188.35, 99.84.188.21, ...
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)|99.84.188.45|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 849990427 (811M) [binary/octet-stream]
Saving to: 'sonarqube-25.9.0.112764.zip'

sonarqube-25.9.0.112764.zip 100%[=====] 810.61M 78.6MB/s in 10s
2025-10-08 03:40:12 (78.4 MB/s) - 'sonarqube-25.9.0.112764.zip' saved [849990427/849990427]
sonarqube@ip-172-31-22-69:~$ |
```

Part 5: Extract Downloaded SonarQube file

```
unzip *
```

```
sonarqube@ip-172-31-22-69: ~$ 
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jackson-annotations-2.19.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jackson-core-2.19.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jackson-databind-2.19.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jcip-annotations-1.0-1.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/content-type-2.3.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/lang-tag-1.7.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/nimbus-jose-jwt-10.0.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/accessors-smart-2.5.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/asm-9.7.1.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/
inflating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/postgresql-42.7.7.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/h2/
inflating: sonarqube-25.9.0.112764/lib/jdbc/h2/h2-2.3.232.jar
inflating: sonarqube-25.9.0.112764/lib/sonar-shutdowner-25.9.0.112764.jar
creating: sonarqube-25.9.0.112764/elasticsearch/plugins/
sonarqube@ip-172-31-22-69:~$ |
```

The file has been unzipped. Check the content of the folder using the command:

```
ls
```

```
sonarqube@ip-172-31-22-69: ~$ 
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jackson-databind-2.19.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jcip-annotations-1.0-1.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/content-type-2.3.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/lang-tag-1.7.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/nimbus-jose-jwt-10.0.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/accessors-smart-2.5.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/asm-9.7.1.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/
inflating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/postgresql-42.7.7.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/h2/
inflating: sonarqube-25.9.0.112764/lib/jdbc/h2/h2-2.3.232.jar
inflating: sonarqube-25.9.0.112764/lib/sonar-shutdowner-25.9.0.112764.jar
creating: sonarqube-25.9.0.112764/elasticsearch/plugins/
sonarqube@ip-172-31-22-69:~$ ls
sonarqube-25.9.0.112764  sonarqube-25.9.0.112764.zip
sonarqube@ip-172-31-22-69:~$ |
```

Part 6: Set Permissions:

```
chmod -R 755 /home/sonarqube/sonarqube-25.9.0.112764
```

```
sonarqube@ip-172-31-22-69: ~$ 
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/jcip-annotations-1.0-1.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/content-type-2.3.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/lang-tag-1.7.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/nimbus-jose-jwt-10.0.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/accessors-smart-2.5.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/asm-9.7.1.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/
inflating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/postgresql-42.7.7.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/h2/
inflating: sonarqube-25.9.0.112764/lib/jdbc/h2/h2-2.3.232.jar
inflating: sonarqube-25.9.0.112764/lib/sonar-shutdowner-25.9.0.112764.jar
creating: sonarqube-25.9.0.112764/elasticsearch/plugins/
sonarqube@ip-172-31-22-69:~$ ls
sonarqube-25.9.0.112764  sonarqube-25.9.0.112764.zip
sonarqube@ip-172-31-22-69:~$ chmod -R 755 /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ |
```

Then run the command:

```
chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-25.9.0.112764
```

```
sonarqube@ip-172-31-22-69: ~$ ls
sonarqube-25.9.0.112764 sonarqube-25.9.0.112764.zip
sonarqube@ip-172-31-22-69:~$ chmod -R 755 /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ |
```

Part 7: Start SonarQube:

Navigate to the directory linux-x86-64 using the command:

```
cd sonarqube-25.9.0.112764/bin/linux-x86-64/
```

```
sonarqube@ip-172-31-22-69: ~$ ls
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/lang-tag-1.7.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/nimbus-jose-jwt-10.0.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/accessors-smart-2.5.2.jar
inflating: sonarqube-25.9.0.112764/lib/jdbc/mssql/asm-9.7.1.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/
inflating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/postgresql-42.7.7.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/h2/
inflating: sonarqube-25.9.0.112764/lib/jdbc/h2/h2-2.3.232.jar
inflating: sonarqube-25.9.0.112764/lib/sonar-shutdowner-25.9.0.112764.jar
creating: sonarqube-25.9.0.112764/elasticsearch/plugins/
sonarqube@ip-172-31-22-69:~$ ls
sonarqube-25.9.0.112764 sonarqube-25.9.0.112764.zip
sonarqube@ip-172-31-22-69:~$ chmod -R 755 /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ cd sonarqube-25.9.0.112764/bin/linux-x86-64/
sonarqube@ip-172-31-22-69:~/sonarqube-25.9.0.112764/bin/linux-x86-64$ |
```

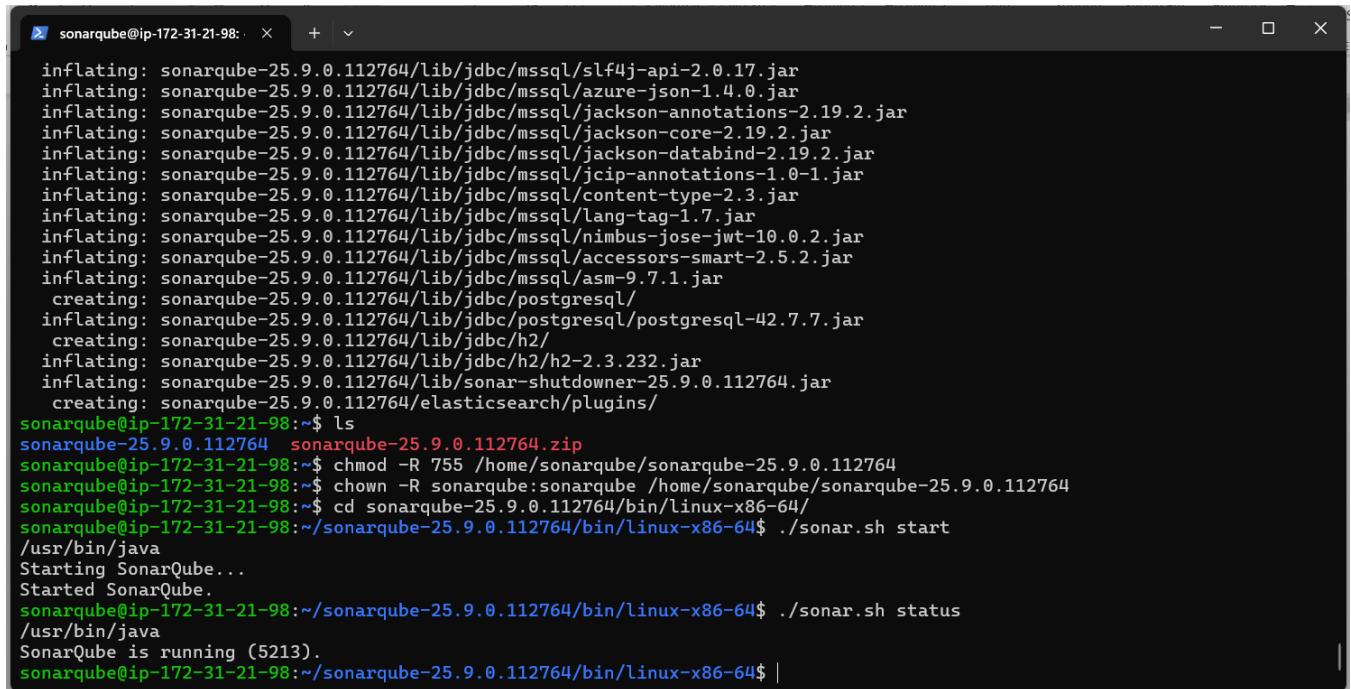
We will start SonarQube by running the command:

```
./sonar.sh start
```

```
sonarqube@ip-172-31-22-69: ~$ ls
creating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/
inflating: sonarqube-25.9.0.112764/lib/jdbc/postgresql/postgresql-42.7.7.jar
creating: sonarqube-25.9.0.112764/lib/jdbc/h2/
inflating: sonarqube-25.9.0.112764/lib/jdbc/h2/h2-2.3.232.jar
inflating: sonarqube-25.9.0.112764/lib/sonar-shutdowner-25.9.0.112764.jar
creating: sonarqube-25.9.0.112764/elasticsearch/plugins/
sonarqube@ip-172-31-22-69:~$ ls
sonarqube-25.9.0.112764 sonarqube-25.9.0.112764.zip
sonarqube@ip-172-31-22-69:~$ chmod -R 755 /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-22-69:~$ cd sonarqube-25.9.0.112764/bin/linux-x86-64/
sonarqube@ip-172-31-22-69:~/sonarqube-25.9.0.112764/bin/linux-x86-64$ ./sonar.sh start
/usr/bin/java
Starting SonarQube...
Started SonarQube.
sonarqube@ip-172-31-22-69:~/sonarqube-25.9.0.112764/bin/linux-x86-64$ |
```

Check Sonarqube status using the command:

```
./sonar.sh status
```



```
sonarqube@ip-172-31-21-98: ~ $ ls
sonarqube-25.9.0.112764 sonarqube-25.9.0.112764.zip
sonarqube@ip-172-31-21-98: ~ $ chmod -R 755 /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-21-98: ~ $ chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-25.9.0.112764
sonarqube@ip-172-31-21-98: ~ $ cd sonarqube-25.9.0.112764/bin/linux-x86-64/
sonarqube@ip-172-31-21-98:~/sonarqube-25.9.0.112764/bin/linux-x86-64$ ./sonar.sh start
/usr/bin/java
Starting SonarQube...
Started SonarQube.
sonarqube@ip-172-31-21-98:~/sonarqube-25.9.0.112764/bin/linux-x86-64$ ./sonar.sh status
/usr/bin/java
SonarQube is running (5213).
sonarqube@ip-172-31-21-98:~/sonarqube-25.9.0.112764/bin/linux-x86-64$ |
```

You can access it by navigating to the IP address of your server followed by port 9000 in a web browser.

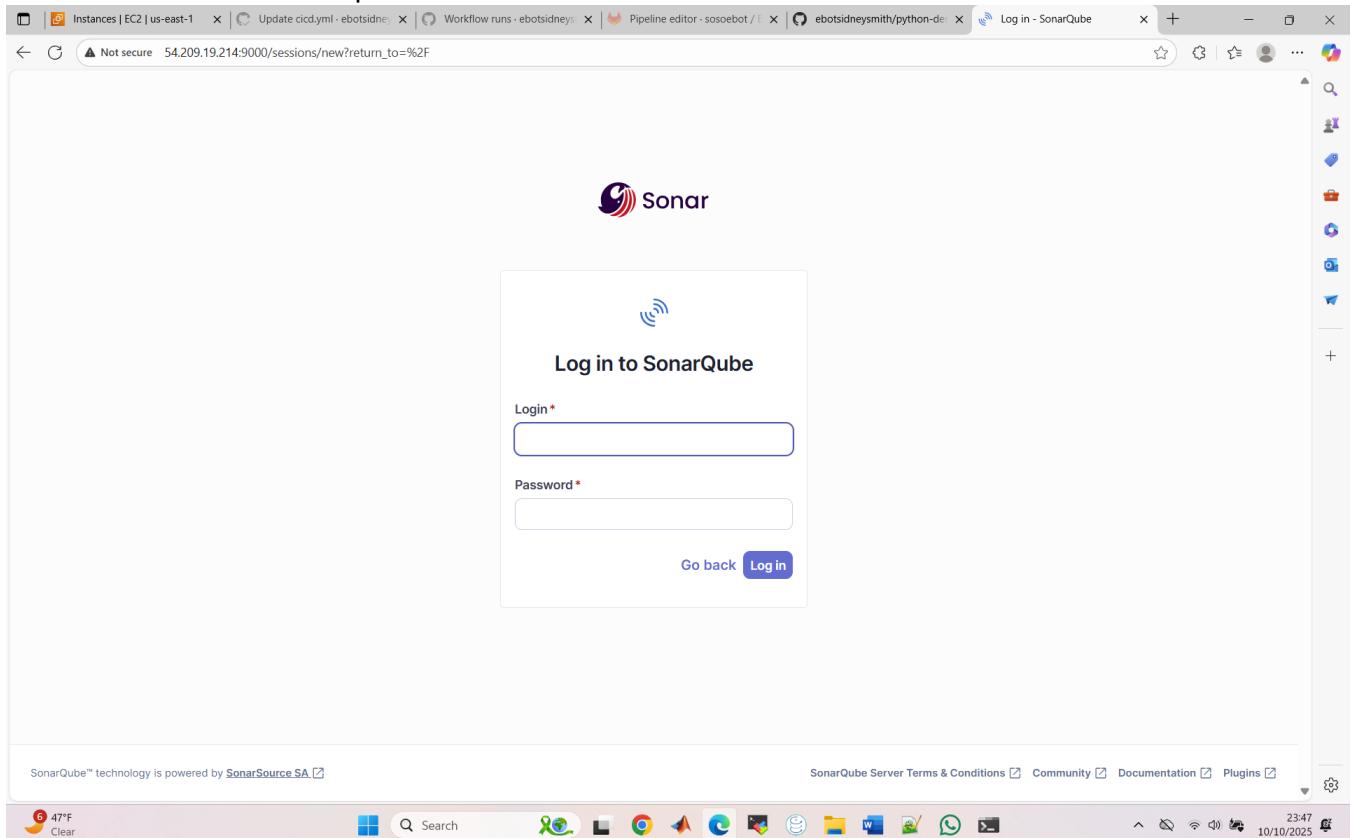
STEP 6: Access SonarQube on Browser

We will now access SonarQube on the browser using

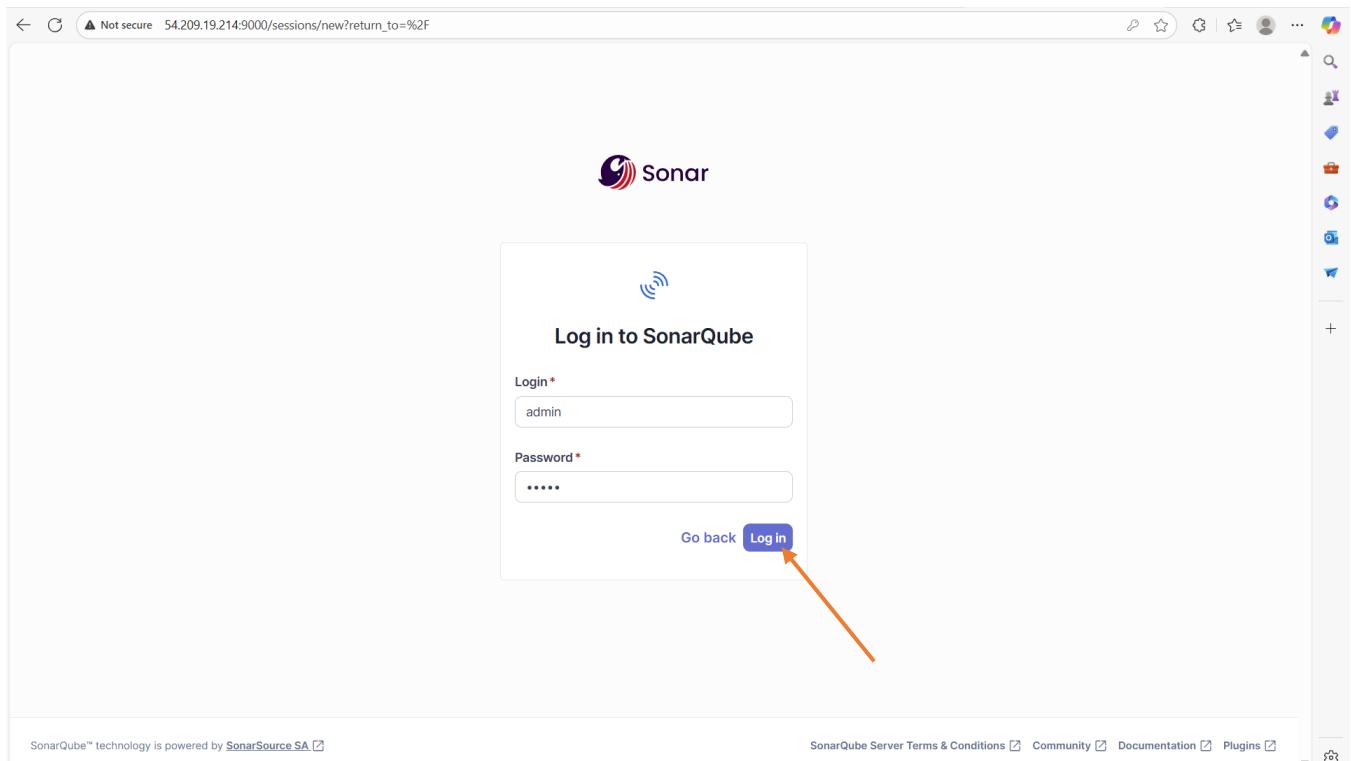
`http://<Public IPv4 address>:9000`

That is **http://54.209.19.214:9000**

The default username and password for SonarQube are both "**admin**".

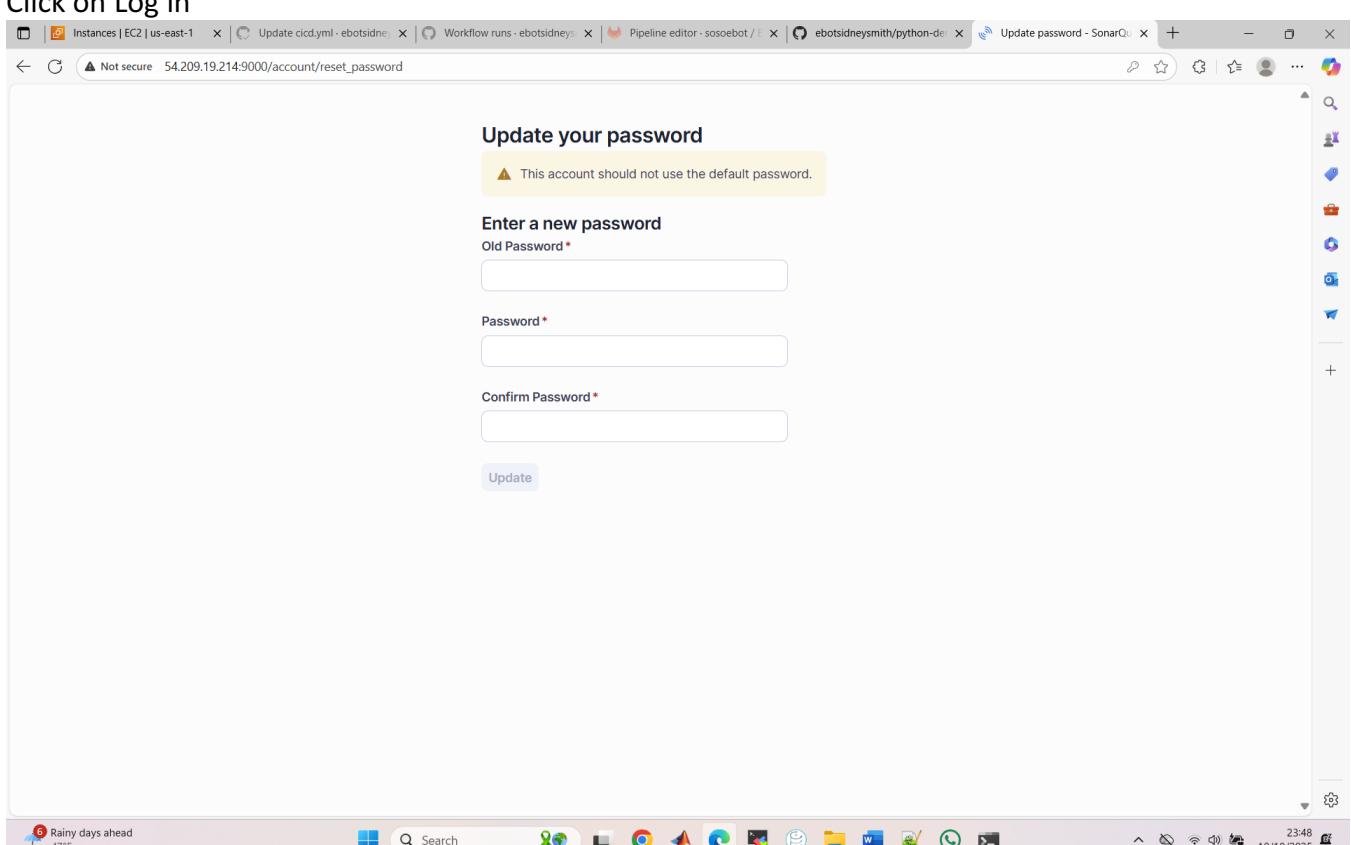


Enter the default username and password “**admin**”



SonarQube™ technology is powered by [SonarSource SA](#). SonarQube Server Terms & Conditions, Community, Documentation, Plugins.

Click on Log in



This account should not use the default password.

Enter a new password

Old Password*

Password*

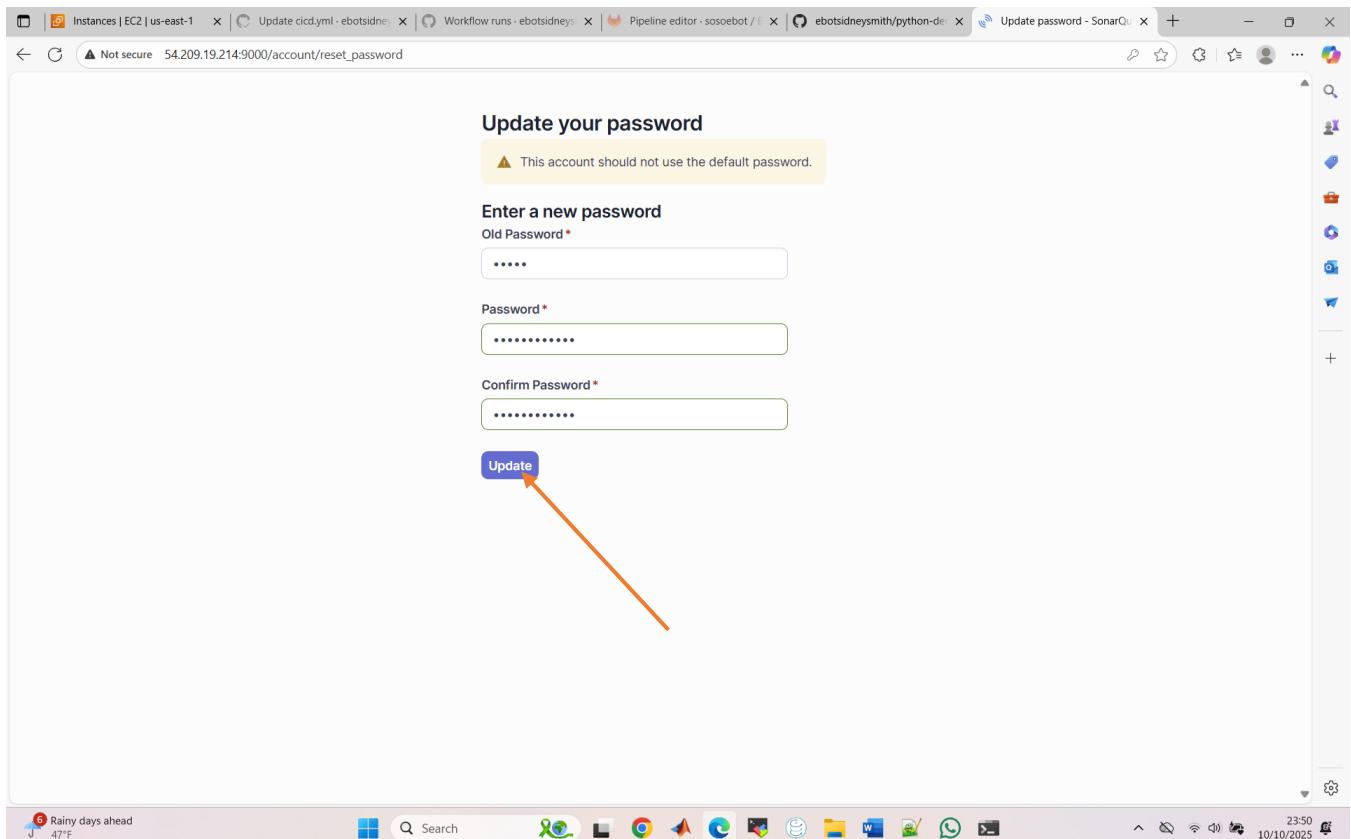
Confirm Password*

Update

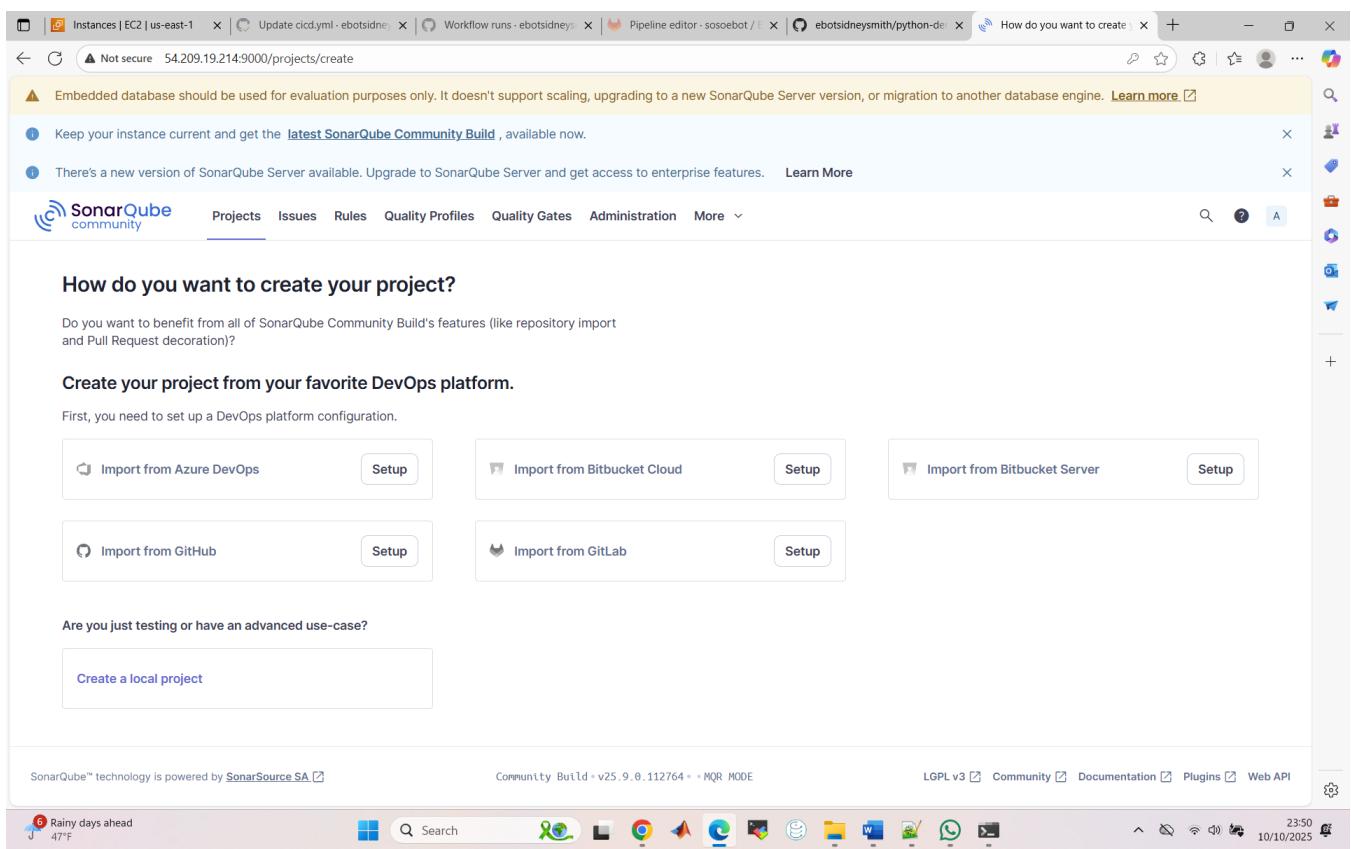
Rainy days ahead 47°F 23:48 10/10/2025

Then modify the password by entering the old password and your new password. Then click on “Update”

Prepared by Sidney Smith



Click on "Update"



Part 1: Create a Local Project

How do you want to create your project?

Do you want to benefit from all of SonarQube Community Build's features (like repository import and Pull Request decoration)?

Create your project from your favorite DevOps platform.

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

Are you just testing or have an advanced use-case?

SonarQube™ technology is powered by [SonarSource SA](#). Community Build v25.9.0.112764 • MQR MODE. LGPL v3. Community Documentation Plugins Web API. Rainy days ahead 47°F 10/10/2025

Click on “Create a local Project”

1 of 2

Create a local project

Project display name *

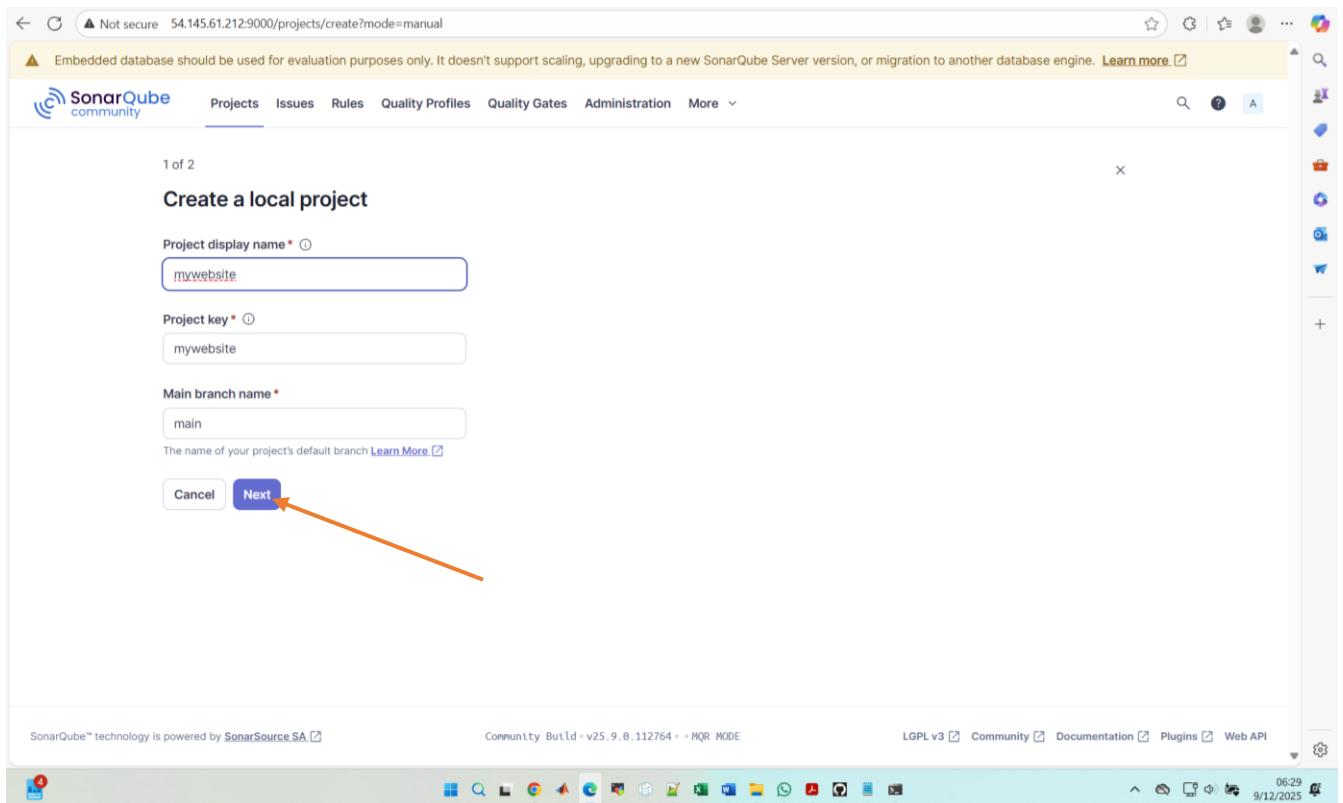
Project key *

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

SonarQube™ technology is powered by [SonarSource SA](#). Community Build v25.9.0.112764 • MQR MODE. LGPL v3. Community Documentation Plugins Web API. 06:28 9/12/2025

Give the project a name, I will call it “mywebsite”

Prepared by Sidney Smith



1 of 2

Create a local project

Project display name * ⓘ
mywebsite

Project key * ⓘ
mywebsite

Main branch name *
main

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

Cancel **Next**

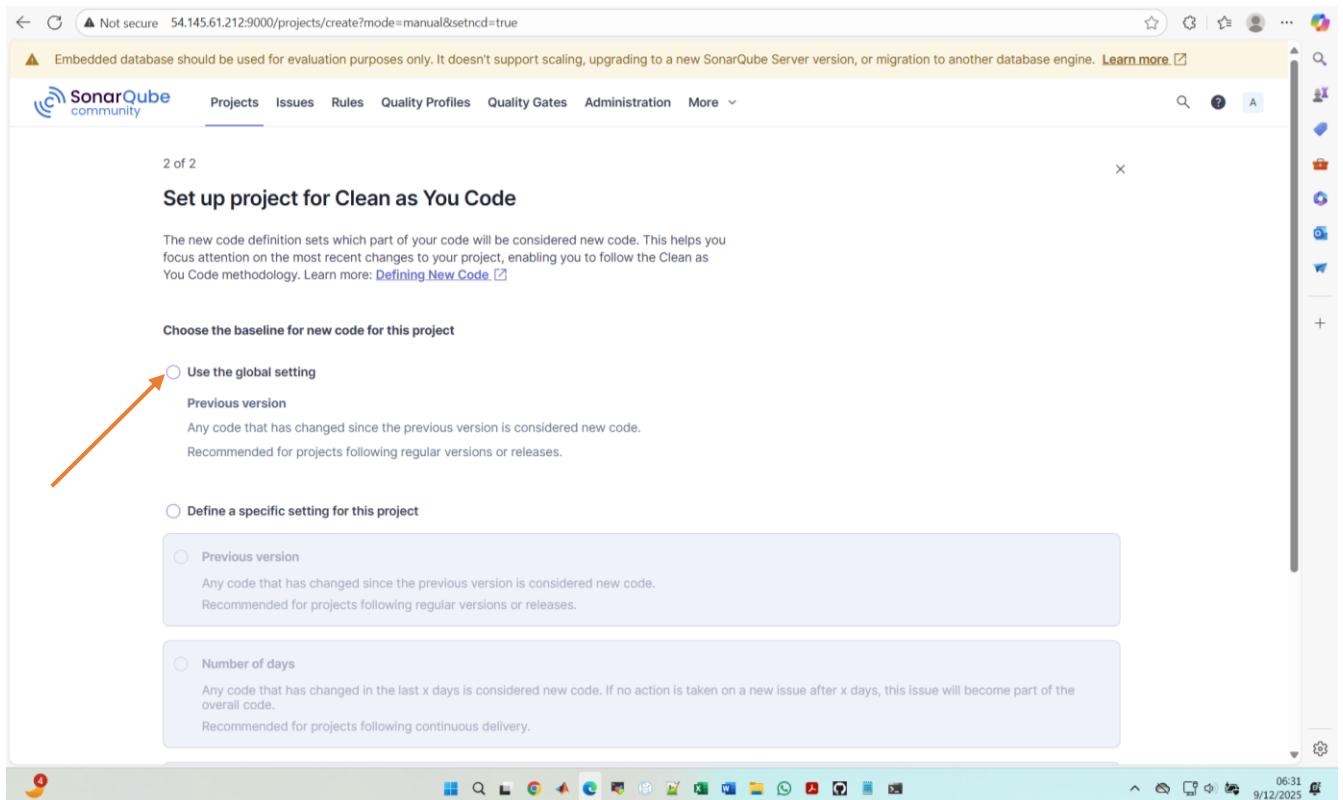
SonarQube™ technology is powered by [SonarSource SA](#) ⓘ

Community Build v25.9.0.112764 • MQR MODE

LGPL v3 ⓘ Community ⓘ Documentation ⓘ Plugins ⓘ Web API

06:29 9/12/2025

Click on “Next”



2 of 2

Set up project for Clean as You Code

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

Choose the baseline for new code for this project

Use the global setting

Previous version

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

Define a specific setting for this project

Previous version

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

Number of days

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

06:31 9/12/2025

Select “Use the global setting”

Prepared by Sidney Smith

The screenshot shows the SonarQube 'Create a local project' interface. The user is on the 'Set up project for Clean as You Code' step. The 'Use the global setting' option is selected. Below it, the 'Previous version' option is described as 'Any code that has changed since the previous version is considered new code. Recommended for projects following regular versions or releases.' The 'Define a specific setting for this project' section is expanded, showing three options: 'Previous version', 'Number of days', and 'Reference branch'. The 'Create project' button is highlighted with a red arrow.

2 of 2

Set up project for Clean as You Code

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

Choose the baseline for new code for this project

Use the global setting

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

Define a specific setting for this project

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

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

Reference branch
Choose a branch as the baseline for the new code.
Recommended for projects using feature branches.

[Back](#) [Create project](#)

Click on “create project”

Prepared by Sidney Smith

The screenshot shows the SonarQube community interface. At the top, there are several tabs: Instances | EC2 | us-east-1, Update cicd.yml - ebotsidney..., Workflow runs - ebotsidney..., Pipeline editor - sosoebot / ebotsidneysmith/python-de..., and SonarQube. Below the tabs, a banner states: "Embedded database should be used for evaluation purposes only. It doesn't support scaling, upgrading to a new SonarQube Server version, or migration to another database engine. [Learn more](#)". The main navigation bar includes Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search icon. The current project is "mywebsite" under "Bind project". The main content area is titled "Analysis Method" and contains the following text: "Use this page to manage and set-up the way your analyses are performed." Below this, there are several options for setting up analysis: "With Jenkins", "With GitHub Actions", "With Bitbucket Pipelines", "With GitLab CI" (which has a red arrow pointing to it), "With Azure Pipelines", and "Other CI". There is also a "Locally" option with a note: "Use this for testing or advanced use-case. Other modes are recommended to help you set up your CI environment." At the bottom of the page, there is footer information: "SonarQube™ technology is powered by [SonarSource SA](#)", "Community Build v25.9.0.112764 - NQR MODE", "LGPL v3", "Community", "Documentation", "Plugins", "Web API", and a system tray with icons for weather, battery, and date.

Our project has been created.

Part 3: Setup the Repository

Now, let us setup the repository that SonarQube can analyze. We will use Jenkins, so click on “With Gitlab”

Prepared by Sidney Smith

The screenshot shows the SonarQube Community interface for a project named 'mywebsite'. The 'Bind project' section is active. In the 'Analysis Method' dropdown, the 'Maven' option is selected, indicated by a red arrow. Other options like Gradle, JS/TS & Web, .NET, Python, and 'Other (for Go, PHP, ...)' are also listed.

Here you can run analysis of your project, so specify the code that you are using on your project. Since we are using a Java-based code, select “**Maven**”

The screenshot shows the SonarQube Community interface for a project named 'mywebsite'. The 'Bind project' section is active. In the 'Analysis Method' dropdown, the 'GitLab CI' option is selected, indicated by a red arrow. Other options like 'SonarQube' and 'GitHub Actions' are also listed.

Scroll down

Prepared by Sidney Smith

2 Create or update the configuration file

1 What option best describes your project?

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

2 Add the following to your pom.xml file:

```
<properties>
  <sonar.projectKey>mywebsite</sonar.projectKey>
  <sonar.projectName>mywebsite</sonar.projectName>
  <sonar.qualitygate.wait>true</sonar.qualitygate.wait>
</properties>
```

3 Create or update your .gitlab-ci.yml file with the following content.

```
image: maven:3-eclipse-temurin-17

variables:
  SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar" # Defines the location of the analysis task cache
  GIT_DEPTH: "0" # Tells git to fetch all the branches of the project, required by the analysis task

stages:
- build-sonar

build-sonar:
  stage: build-sonar

  cache:
    policy: pull-push
    key: "sonar-cache-$CI_COMMIT_REF_SLUG"
    paths:
      - "${SONAR_USER_HOME}/cache"
      - sonar-scanner

  script:
    - mvn verify org.sonarsource.scanner.maven:sonar-maven-plugin:sonar
  allow_failure: true
  rules:
    - if: $CI_PIPELINE_SOURCE == 'merge_request_event'
      - if: $CI_COMMIT_BRANCH == 'Master'
      - if: $CI_COMMIT_BRANCH == 'main'
      - if: $CI_COMMIT_BRANCH == 'develop'
```

⚠ GitLab vulnerability report is only available with GitLab Ultimate. You may safely remove the sonarqube-vulnerability-report stage if you have not subscribed to this service.

Note that this is a minimal base configuration to run a SonarQube Community Build analysis on your main branch and merge requests, and fetch the vulnerability report (if applicable). If you already have a pipeline configured and running, you might want to add the example above to your existing yml file.



And you are done!

We will use the above information when writing the job for SonarQube.

STEP 7: Installing Gitlab Runner on Ubuntu EC2 Instance

Part 1: Create Gitlab runner

Go to the Gitlab Project

The repository for this project is empty
To get started, clone the repository or upload some files.

Instructions

Add files

SSH HTTPS

Project information

Invite your team

Created on
October 07, 2025

Click on “Settings” and select “CI/CD”

General pipelines

Auto DevOps

Runners

Artifacts

Variables

Pipeline trigger tokens

Deploy freezes

Job token permissions

Click on “Runners”

The screenshot shows the GitLab interface for a project named "Boardgame". The left sidebar is open, showing various project settings like Build, Secure, Deploy, Operate, Monitor, Analyze, and Settings. The "CI/CD" tab is selected. In the main content area, under "Auto DevOps", the "Runners" section is expanded. It contains three categories: "Project runners" (0), "Other available runners" (#50055565 (AKqlnI7D) labeled "project-runner"), and "Group runners" (0). A red arrow points to the "Create project runner" button in the top right corner of the "Project runners" section.

Click on “Create project runner”

The screenshot shows the "Create project runner" form for the "Java-Maven" project. The left sidebar shows pinned issues and merge requests. The main form has a "Tags" section with a text input field for comma-separated tags, which is highlighted with a red arrow. Below it is a "Configuration (optional)" section with fields for "Runner description", "Paused" (unchecked), "Protected" (unchecked), "Lock to current projects" (unchecked), and "Maximum job timeout".

Give it a name. I will call it “**project-runner**”

Prepared by Sidney Smith

Tags

Add tags to specify jobs that the runner can run. Learn more.

project-runner

Separate multiple tags with a comma. For example, `macos, shared`.

Run untagged jobs
Use the runner for jobs without tags in addition to tagged jobs.

Configuration (optional)

Runner description

Paused
Stop the runner from accepting new jobs.

Protected
Use the runner on pipelines for protected branches only.

Lock to current projects
Use the runner for the currently assigned projects only. Only administrators can change the assigned projects.

Maximum job timeout

Maximum amount of time the runner can run before it terminates. If a project has a shorter job timeout period, the job timeout period of the instance runner is used instead.

Enter the job timeout in seconds. Must be a minimum of 600 seconds.

Create runner

Click on “Create Runner”

https://gitlab.com/sosoebot/java-maven/-/runners/50055565/register

Runner created.

Register runner

Platform

Operating systems

Linux macOS Windows

Cloud

Google Cloud GKE

Containers

Docker Kubernetes

GitLab Runner must be installed before you can register a runner. How do I install GitLab Runner?

Step 1

Copy and paste the following command into your command line to register the runner.

```
$ gitlab-runner register  
--url https://gitlab.com  
--token glrt-AKqIn17DimLM2kE9eC-4Sm86MOpw0jE4cGpkawp00jMKdTpiajJ6NBg.01.1j1vhqdy1
```

The runner authentication token `glrt-AKqIn17DimLM2kE9eC-4Sm86MOpw0jE4cGpkawp00jMKdTpiajJ6NBg.01.1j1vhqdy1` displays here for a short time only. After you register

My EC2 instance is ubuntu, so I will select “Linux”.

Part 2: Install Gitlab Runner

Click on “How do I install Gitlab Runner”

The screenshot shows the GitLab 'Register runner' page. On the left, there's a sidebar with project navigation. In the center, under 'Platform', 'Operating systems' has 'Linux' selected. On the right, a terminal window displays the command to download the GitLab Runner binary for amd64 architecture. An orange arrow points from the 'Architecture' dropdown in the top right to the terminal command.

```
# Download the binary for your system
sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64
```

Copy the first line of code and paste on the terminal to Download the binary for your system

```
sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64
```

```
root@ip-172-31-22-234:/home/ubuntu# sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left Speed
100 87.6M 100 87.6M    0     0  90.5M      0 --:--:-- --:--:-- 90.4M
root@ip-172-31-22-234:/home/ubuntu# |
```

Give it permission to execute

```
sudo chmod +x /usr/local/bin/gitlab-runner
```

```
root@ip-172-31-22-234:/home/ubuntu# sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left Speed
100 87.6M 100 87.6M    0     0  90.5M      0 --:--:-- --:--:-- 90.4M
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner
root@ip-172-31-22-234:/home/ubuntu# |
```

Create a GitLab Runner user

```
sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --shell /bin/bash
```

```
root@ip-172-31-22-234:/home/ubuntu# sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload Upload Total Spent   Left Speed
100 87.6M  100 87.6M    0     0  90.5M    0 --:--:-- --:--:-- --:--:-- 90.4M
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner
root@ip-172-31-22-234:/home/ubuntu# sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner
--shell /bin/bash
root@ip-172-31-22-234:/home/ubuntu# |
```

Install and run as a service

```
sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner
sudo gitlab-runner start
```

```
root@ip-172-31-22-234:/home/ubuntu# sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload Upload Total Spent   Left Speed
100 87.6M  100 87.6M    0     0  90.5M    0 --:--:-- --:--:-- --:--:-- 90.4M
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner
root@ip-172-31-22-234:/home/ubuntu# sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --shell /bin/bash
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner
sudo gitlab-runner start
Runtime platform
Runtime platform
arch=amd64 os=linux pid=1341 revision=139a0ac0 version=18.4.0
arch=amd64 os=linux pid=1437 revision=139a0ac0 version=18.4.0
root@ip-172-31-22-234:/home/ubuntu# |
```

Head back to our Gitlab repository

The screenshot shows the GitLab interface for registering a runner. On the left, the sidebar lists projects like 'Boardgame', pinned issues, merge requests, and various management sections. The main area shows a message that a runner has been created and lists Docker and Kubernetes as available executors. A large 'Step 1' section contains a command-line snippet for registering the runner with a URL and token, followed by a note about the authentication token being stored in config.toml. A 'Step 2' section asks to choose an executor, and a 'Step 3 (optional)' section shows a command to verify the runner's availability. To the right, a 'Install GitLab Runner' dialog box provides detailed instructions for installing the runner binary, giving it execute permissions, creating a user, and starting the service. It also includes a link to see more installation methods and architectures.

Then follow the steps

Copy and paste the following command into your command line to register the runner.

```
gitlab-runner register --url https://gitlab.com --token glrt-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco
```

```
root@ip-172-31-22-234:/home% Total % Received % Xferd Average Speed Time Time Time Current  
100 87.6M 100 87.6M 0 0 90.5M 0 --:--:--:--:--:-- 90.4M  
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner  
root@ip-172-31-22-234:/home/ubuntu# sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --shell /bin/bash  
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner  
sudo gitlab-runner start  
Runtime platform arch=amd64 os=linux pid=1341 revision=139a0ac0 version=18.4.0  
Runtime platform arch=amd64 os=linux pid=1437 revision=139a0ac0 version=18.4.0  
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco  
Runtime platform arch=amd64 os=linux pid=1478 revision=139a0ac0 version=18.4.0  
Running in system-mode.  
  
Enter the GitLab instance URL (for example, https://gitlab.com/):  
[https://gitlab.com]: |
```

Copy this and paste on the terminal, then press Enter

```
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner  
root@ip-172-31-22-234:/home/ubuntu# sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --shell /bin/bash  
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner  
sudo gitlab-runner start  
Runtime platform arch=amd64 os=linux pid=1341 revision=139a0ac0 version=18.4.0  
Runtime platform arch=amd64 os=linux pid=1437 revision=139a0ac0 version=18.4.0  
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco  
Runtime platform arch=amd64 os=linux pid=1478 revision=139a0ac0 version=18.4.0  
Running in system-mode.  
  
Enter the GitLab instance URL (for example, https://gitlab.com/):  
[https://gitlab.com]: https://gitlab.com  
Verifying runner... is valid correlation_id=1a99deb60fe743d2ab5a656593e0048e runner=YNfXnG9tz  
Enter a name for the runner. This is stored only in the local config.toml file:  
[ip-172-31-22-234]: |
```

Let us use the name “runner”, so enter “runner”

```
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner  
sudo gitlab-runner start  
Runtime platform arch=amd64 os=linux pid=1341 revision=139a0ac0 version=18.4.0  
Runtime platform arch=amd64 os=linux pid=1437 revision=139a0ac0 version=18.4.0  
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco  
Runtime platform arch=amd64 os=linux pid=1478 revision=139a0ac0 version=18.4.0  
Running in system-mode.  
  
Enter the GitLab instance URL (for example, https://gitlab.com/):  
[https://gitlab.com]: https://gitlab.com  
Verifying runner... is valid correlation_id=1a99deb60fe743d2ab5a656593e0048e runner=YNfXnG9tz  
Enter a name for the runner. This is stored only in the local config.toml file:  
[ip-172-31-22-234]: runner  
Enter an executor: ssh, parallels, virtualbox, docker, docker-windows, docker+machine, custom, shell, kubernetes, docker-autoscaler, instance:  
|
```

Our executor is “docker”. So, enter “docker”

Prepared by Sidney Smith

```
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner
sudo gitlab-runner start
Runtime platform arch=amd64 os=linux pid=1341 revision=139a0ac0 version=18.4.0
Runtime platform arch=amd64 os=linux pid=1437 revision=139a0ac0 version=18.4.0
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfxnG9tzHlnRVQg_GX89m86MOpw0jE4cGp0bAp00jMKdTpiaj6NBg_01.1j1ccfxco
Runtime platform arch=amd64 os=linux pid=1478 revision=139a0ac0 version=18.4.0
Running in system-mode.

Enter the GitLab instance URL (for example, https://gitlab.com/):
[https://gitlab.com]: https://gitlab.com
Verifying runner... is valid correlation_id=1a99deb60fe743d2ab5a656593e0048e runner=YNfxnG9tz
Enter a name for the runner. This is stored only in the local config.toml file:
[ip-172-31-22-234]: runner
Enter an executor: ssh, parallels, virtualbox, docker, docker-windows, docker+machine, custom, shell, kubernetes, docker-autoscaler, instance: docker
Enter the default Docker image (for example, ruby:3.3):
|
```

Our project is a Java-based application. So, enter the command:

```
maven:3.8.4-openjdk-17
```

```
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfxnG9tzHlnRVQg_GX89m86MOpw0jE4cGp0bAp00jMKdTpiaj6NBg_01.1j1ccfxco
Runtime platform arch=amd64 os=linux pid=1478 revision=139a0ac0 version=18.4.0
Running in system-mode.

Enter the GitLab instance URL (for example, https://gitlab.com/):
[https://gitlab.com]: https://gitlab.com
Verifying runner... is valid correlation_id=1a99deb60fe743d2ab5a656593e0048e runner=YNfxnG9tz
Enter a name for the runner. This is stored only in the local config.toml file:
[ip-172-31-22-234]: runner
Enter an executor: ssh, parallels, virtualbox, docker, docker-windows, docker+machine, custom, shell, kubernetes, docker-autoscaler, instance: docker
Enter the default Docker image (for example, ruby:3.3):
maven:3.8.4-openjdk-17
Runner registered successfully. Feel free to start it, but if it's running already the config should be automatically reloaded!

Configuration (with the authentication token) was saved in "/etc/gitlab-runner/config.toml"
root@ip-172-31-22-234:/home/ubuntu# |
```

We have successfully registered the runner with Gitlab instance. Manually verify that the runner is available to pick up jobs using the command:

```
gitlab-runner run
```

```
root@ip-172-31-22-234:/home# Starting multi-runner from /etc/gitlab-runner/config.toml... builds=0 max_builds=0
Running in system-mode.

Usage logger disabled builds=0 max_builds=1
Configuration loaded builds=0 max_builds=1
WARNING: CONFIGURATION: Long polling issues detected.
Issues found:
  - Request bottleneck: 1 runners have request_concurrency=1, causing job delays during long polling
This can cause job delays matching your GitLab instance's long polling timeout.
Recommended solutions:
  1. Increase 'request_concurrency' to 2-4 for 1 runners currently using request_concurrency=1
Note: The 'FF_USE_ADAPTIVE_REQUEST_CONCURRENCY' feature flag can help automatically adjust request_concurrency based on workload.
This message will be printed each time the configuration is reloaded if the issues persist.
See documentation: https://docs.gitlab.com/runner/configuration/advanced-configuration.html#long-polling-issues builds=0 max_builds=1
listen_address not defined, metrics & debug endpoints disabled builds=0 max_builds=1
[session_server].listen_address not defined, session endpoints disabled builds=0 max_builds=1
Initializing executor providers builds=0 max_builds=1
|
```

Head back to the Gitlab project

The screenshot shows the 'Register runner' page for a project named 'Boardgame'. The main content area displays three steps: Step 1 (Runner created), Step 2 (Choose an executor), and Step 3 (optional). Step 3 has a note about manually verifying the runner's availability. A success message at the bottom states: 'You've registered a new runner!' with a green checkmark icon. To the right, a sidebar titled 'Install GitLab Runner' provides installation instructions for 'Architecture amd64'.

You can see that the runner has been registered. Click on “View Runners”

The screenshot shows the 'Runners' settings page for the 'Boardgame' project. It lists three categories of runners: 'Project runners' (1 assigned), 'Group runners' (0 available), and 'Instance runners' (112 available). The 'Project runners' section shows a single entry: '#50078084 (YNfXnG9t)' with a green signal icon, labeled 'project-runner'. An orange arrow points to this green signal icon. The 'Group runners' and 'Instance runners' sections are currently empty.

You can see the green signal which means it is available to run jobs. The Gitlab runner is set up.

STEP 8: Adding Jobs

Let us proceed now with the adding of jobs for testing, building and analysis of the code. Followed by building and pushing the Docker image to Docker hub by Docker. We will name the jobs: test-job, sonarqube-check, build-job and build_image_push-job.

The screenshot shows the GitLab CI/CD Settings page for a project named 'Boardgame'. The left sidebar has a 'Build' menu item highlighted with an orange arrow. The main content area displays sections for 'Project runners', 'Group runners', and 'Instance runners'. A 'Create project runner' button is located at the top right of the 'Project runners' section. The 'Instance runners' section includes a toggle switch for 'Turn on instance runners for this project'.

Click on “Build”

The screenshot shows the same GitLab CI/CD Settings page as before, but the 'Pipeline editor' tab is now selected under the 'Build' menu in the sidebar. This highlights the 'Jobs', 'Pipeline schedules', and 'Artifacts' sections in the main content area. The 'Create project runner' button remains at the top right of the 'Project runners' section.

Select “Pipeline Editor”

https://gitlab.com/sosoebot/boardgame/-/ci/editor?branch_name=main

sosoebot / Boardgame / Pipeline editor

main

Project

- Issues 0
- Merge requests 0
- Manage >
- Plan >
- Code >
- Build >
 - Pipelines
 - Jobs
 - Pipeline editor
 - Pipeline schedules
 - Artifacts
 - Secure >
 - Deploy >
 - Operate >
 - Monitor >
 - Analyze >
 - Settings >
- What's new 4
- Help

Configure a pipeline to automate your builds, tests, and deployments

Create a `.gitlab-ci.yml` file in your repository to configure and run your first pipeline.

Configure pipeline

Click on “Configure Pipeline”

https://gitlab.com/sosoebot/boardgame/-/ci/editor?branch_name=main

sosoebot / Boardgame / Pipeline editor

main

Project

- Issues 0
- Merge requests 0
- Manage >
- Plan >
- Code >
- Build >
 - Pipelines
 - Jobs
 - Pipeline editor
 - Pipeline schedules
 - Artifacts
 - Secure >
 - Deploy >
 - Operate >
 - Monitor >
 - Analyze >
 - Settings >
- What's new 4
- Help

This GitLab CI configuration is invalid: Reference not found. Learn more

Edit Visualize Validate Full configuration

CI/CD Catalog Help Editor accessibility guide

```
1 # This file is a template, and might need editing before it works on your project.
2 # This is a sample GitLab CI/CD configuration file that should run without any modifications.
3 # It demonstrates a basic 3 stage CI/CD pipeline. Instead of real tests or scripts,
4 # it uses echo commands to simulate the pipeline execution.
5 #
6 # A pipeline is composed of independent jobs that run scripts, grouped into stages.
7 # Stages run in sequential order, but jobs within stages run in parallel.
8 #
9 # For more information, see: https://docs.gitlab.com/ee/ci/yaml/#stages
10 #
11 # You can copy and paste this template into a new '.gitlab-ci.yml' file.
12 # You should not add this template to an existing '.gitlab-ci.yml' file by using the 'include:' keyword.
13 #
14 # To contribute improvements to CI/CD templates, please follow the Development guide at:
15 # https://docs.gitlab.com/development/cicd/templates/
16 # This specific template is located at:
17 # https://gitlab.com/gitlab-org/gitlab/-/blob/master/lib/gitlab/ci/templates/Getting-Started.gitlab-ci.yml
18
19 stages:      # List of stages for jobs, and their order of execution
20   - build
21   - test
22   - deploy
23
24 build-job:    # This job runs in the build stage, which runs first.
25   stage: build
26   script:
```

Commit message

Update .gitlab-ci.yml file

Delete the existing code

The screenshot shows the GitLab Pipeline editor interface. On the left, there's a sidebar with various project management links like Issues, Merge requests, and Pipelines. The 'Pipeline editor' link is currently selected. The main area displays a pipeline configuration with one stage labeled '1'. A commit message box at the bottom indicates an update to the .gitlab-ci.yml file.

Now, we can start writing code for our jobs.

Part 1: Adding “test-job”

Our first job is the test with Maven

Adding the job

```
test-job:  
stage: test  
script:  
  - echo "Running tests..."  
  - mvn test  
tags:  
  - project-runner
```

The “tags” is obtained as follows

Prepared by Sidney Smith

The screenshot shows the GitLab Pipeline editor interface. On the left, there's a sidebar with various project management options like Issues, Merge requests, and Pipelines. The 'Pipeline editor' option is selected. A dropdown menu is open over the 'CI/CD' tab in the main navigation bar, listing options such as General, Integrations, Webhooks, Access tokens, Repository, Merge requests, CI/CD (which is currently selected), Packages and registries, Monitor, and Usage quotas. The main content area displays a failed pipeline named '#2086452497' with the message 'Pipeline failed for 00ad00ae 37 seconds ago'. Below this, a code snippet for a 'test-job' is shown:

```
1 test-job:
2   stage: test
3   script:
4     - echo "Running tests..."
5     - mvn test
```

Click on CI/CD

The screenshot shows the 'CI/CD Settings' page. The sidebar has the 'CI/CD' option selected. The main content area lists several sections: General pipelines, Auto DevOps, Runners, Artifacts, Variables, Pipeline trigger tokens, Deploy freezes, and Job token permissions. An orange arrow points from the text 'Click on “Runners”' to the 'Runners' section heading.

Click on “Runners”

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Settings page for a project named "Boardgame". The left sidebar is open, showing various project settings like Build, Secure, Deploy, Operate, Monitor, Analyze, and Settings. Under Settings, the "CI/CD" tab is selected. The main content area is titled "Auto DevOps" and "Runners". It displays two project runners assigned to the project: "#50078084 (YNfXnG9t)" and "#50055565 (AKqIn7D)". Both runners are labeled "project-runner". Below this, there is a section for "Group runners" which is currently empty. At the bottom, there is a section for "Instance runners" with several entries, including "#11573930 (KzYhZxBv)" and "#11573990 (NL4gfoBe)". A red arrow points from the text "Click on the Project runner" to the "#50078084" entry.

Click on the Project runner

The screenshot shows the details page for the project runner "#50078084 (YNfXnG9t)". The top navigation bar shows the URL as https://gitlab.com/sosoebot/boardgame/-/runners/50078084. The page title is "Runner #50078084 (YNfXnG9t)". The left sidebar is identical to the one in the previous screenshot. The main content area shows the runner's status as "Online", its creation by "Sidney Smith Ebot" 45 minutes ago, and its configuration as "None". It also shows maximum job timeout as "None", token expiry as "Never expires", and tags as "project-runner". A red arrow points from the text "You can see the tag = “project-runner”" to the "project-runner" tag in the "Tags" section. Below this, there are sections for "Assigned Projects" (showing "sosoebot / Boardgame" as the owner) and "Jobs" (listing one failed job). The "Jobs" table has columns: Status, Job, Project, Commit, Finished at, Duration, Queued, and Tags. The first job row shows "Failed" under Status, "#11637492365" under Job, "Boardgame" under Project, "00ad00ae" under Commit, "4 minutes ago" under Finished at, "00:00:11" under Duration, "00:00:00" under Queued, and "project-runner" under Tags.

You can see the tag = “project-runner”

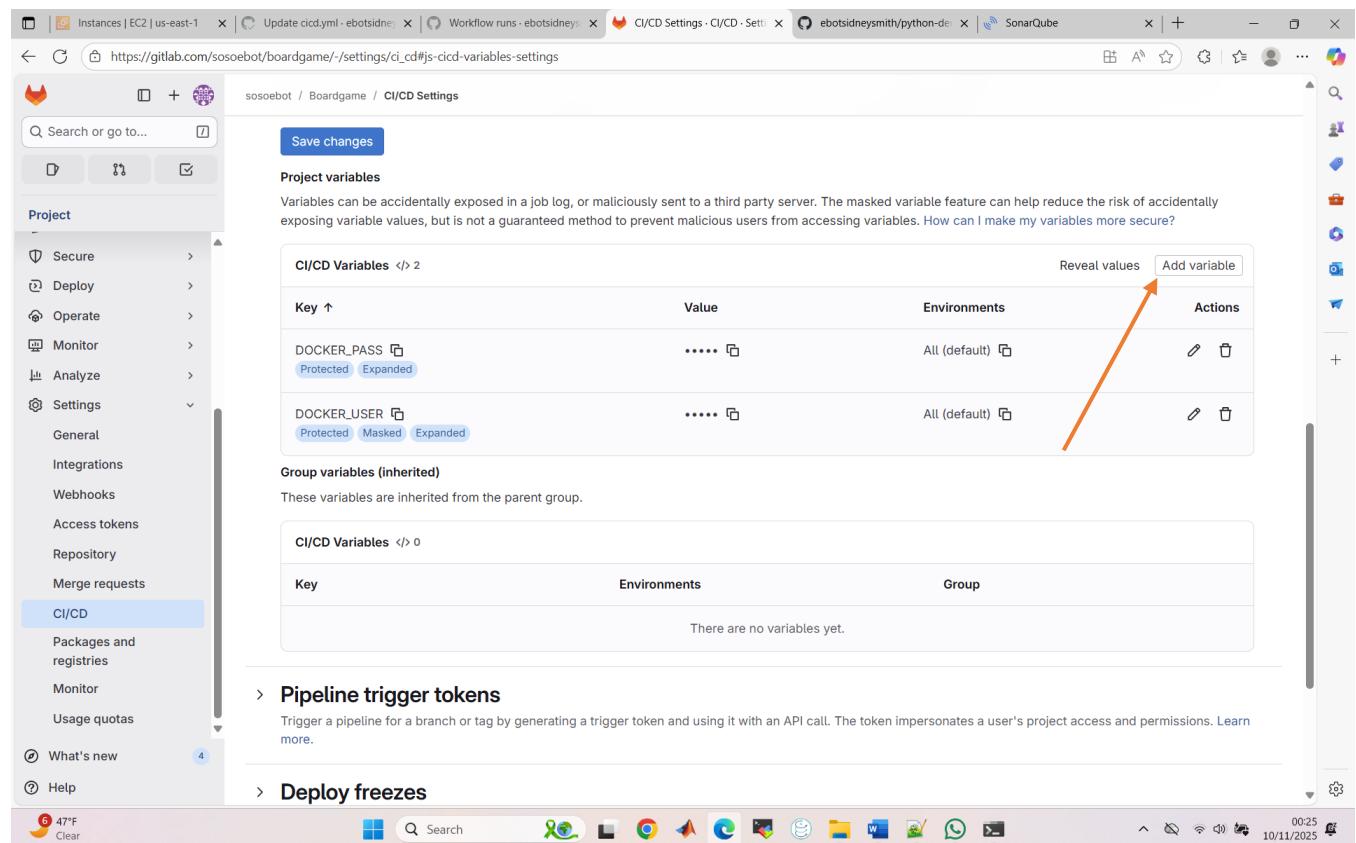
Part 2: Adding “sonarqube-check”

Now we will add the job to do the code analysis. We will follow these steps:

Adding Environment Variables

1 Add environment variables

- 1 Define the SonarQube Community Build Token environment variable. In GitLab, go to **Settings > CI/CD > Variables** to add the following variable and make sure it is available for your project:
 - Key `SONAR_TOKEN` 
 - In the **Value** field, enter an existing token, or a newly generated one: 
 - Uncheck the **Protect Variable** checkbox.
 - Check the **Mask Variable** checkbox.



The screenshot shows the GitLab interface for managing CI/CD variables. On the left, there's a sidebar with project navigation. The main area is titled "CI/CD Variables" and lists two variables: "DOCKER_PASS" and "DOCKER_USER". Both variables are marked as "Protected" and "Expanded". The "DOCKER_USER" variable is also marked as "Masked". A red arrow points to the "Add variable" button at the top right of the table. Below the table, there's a section for "Group variables (inherited)" which is currently empty.

Click on “Add Variable”

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Settings page for a project named "sosoebot / Boardgame". The "CI/CD Variables" section lists two variables: "DOCKER_PASS" and "DOCKER_USER", both of which are currently set to "Protected" and "Expanded". To the right of the variables, there is a "Visibility" section with three options: "Visible", "Masked" (which is selected), and "Masked and hidden". An orange arrow points from the "Masked" option to the "Value" field of the "DOCKER_PASS" variable. Below the visibility section is a "Flags" section containing "Protect variable" and "Expand variable reference" checkboxes, both of which are checked. A "Description (optional)" field is present, and a "Key" field is empty. A "Value" field is also present. At the bottom of the page, there are "Add variable" and "Cancel" buttons.

This screenshot is identical to the one above, but the "Protect variable" checkbox in the "Flags" section is now unchecked. An orange arrow points from the "Key" field to the "Value" field of the "DOCKER_PASS" variable, indicating that the variable's value is now masked.

For "Key", enter "SONAR_TOKEN"

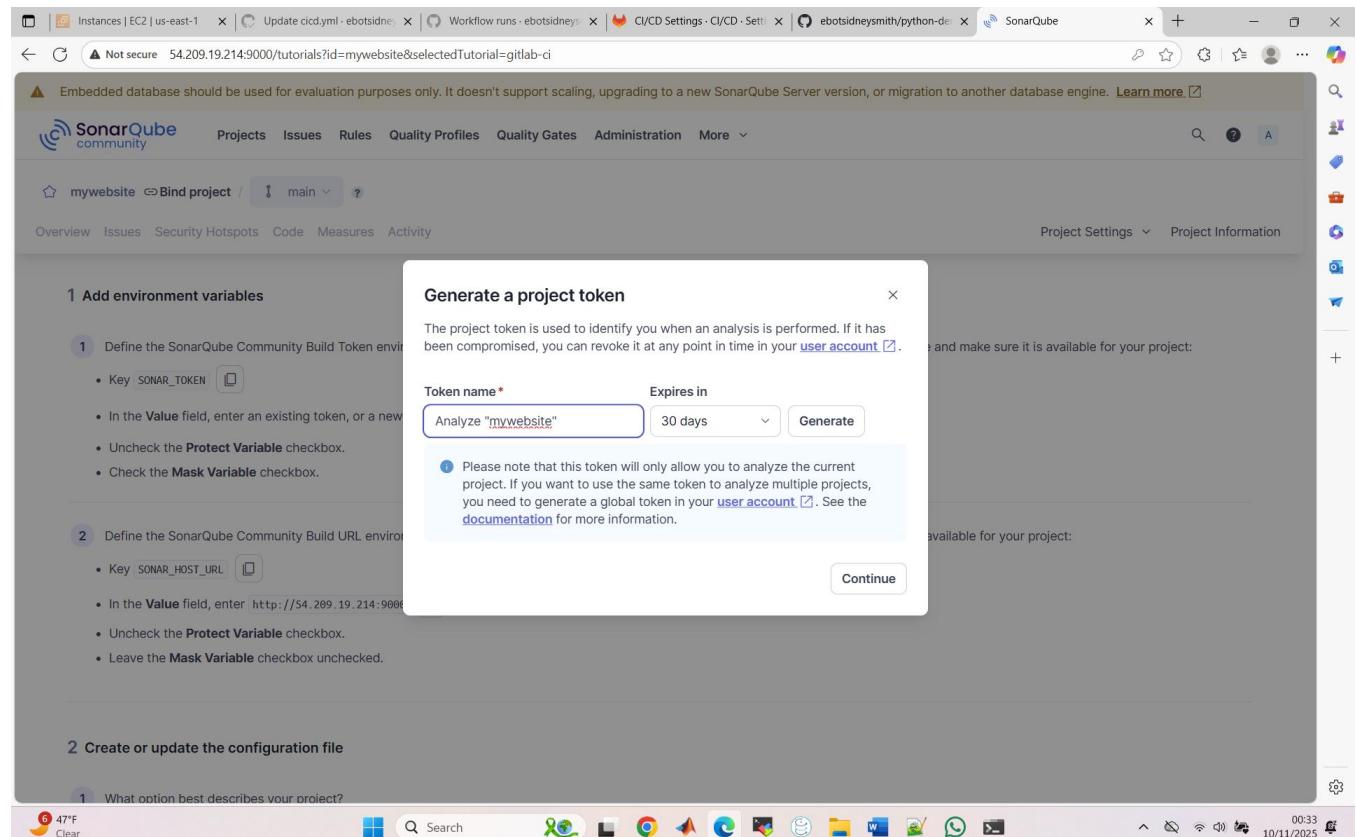
For the “Value”, go to the steps in the Sonarqube browser

1 Add environment variables

- 1 Define the SonarQube Community Build Token environment variable. In GitLab, go to **Settings > CI/CD > Variables** to add the following variable and make sure it is available for your project:

- Key `SONAR_TOKEN` 
- In the **Value** field, enter an existing token, or a newly generated one:  
- Uncheck the **Protect Variable** checkbox.
- Check the **Mask Variable** checkbox.

Click on “Generate a Token”



The screenshot shows a browser window with multiple tabs open, including ones for AWS Instances, GitLab CI/CD, and SonarQube. The main content is a SonarQube project configuration page for 'mywebsite'. A modal dialog box is open, titled 'Generate a project token'. It contains fields for 'Token name*' (set to 'Analyze "mywebsite"') and 'Expires in' (set to '30 days'). A 'Generate' button is present. Below these fields is a note: 'Please note that this token will only allow you to analyze the current project. If you want to use the same token to analyze multiple projects, you need to generate a global token in your user account. See the documentation for more information.' At the bottom of the dialog is a 'Continue' button. The background of the browser shows the SonarQube interface with various navigation links and a sidebar.

Click on “Generate”

Prepared by Sidney Smith

The screenshot shows the SonarQube interface for a project named "mywebsite". A modal window titled "Generate a project token" is open, displaying a success message: "New token \"sqp_c766aa7cf13d5bd082c55a709a209faf68bbca6b\" has been created. Make sure you copy it now, you won't be able to see it again!" An orange arrow points from this message to the "Copy" button next to the token value.

1 Add environment variables

- 1 Define the SonarQube Community Build Token environment variable:
 - Key: SONAR_TOKEN
 - In the Value field, enter an existing token, or a new token.
 - Uncheck the Protect Variable checkbox.
 - Check the Mask Variable checkbox.
- 2 Define the SonarQube Community Build URL environment variable:
 - Key: SONAR_HOST_URL
 - In the Value field, enter <http://54.209.19.214:9000>
 - Uncheck the Protect Variable checkbox.
 - Leave the Mask Variable checkbox unchecked.

2 Create or update the configuration file

- 1 What option best describes your project?

Copy the token: sqp_c766aa7cf13d5bd082c55a709a209faf68bbca6b
Enter this on the “Value” field on the Gitlab browser

The screenshot shows the GitLab CI/CD Variables settings page for a project named "sosoebot". The "CI/CD Variables" section lists two variables: "DOCKER_PASS" and "DOCKER_USER", both of which are masked. A modal window titled "Add variable" is open, showing the "Value" field populated with the copied token "sqp_c766aa7cf13d5bd082c55a709a209faf68bbca6b". An orange arrow points from this value field to the "Add variable" button.

CI/CD Variables </> 2

Key ↑	Value	Envir
DOCKER_PASS	All (d)
DOCKER_USER	All (d)

Group variables (inherited)
These variables are inherited from the parent group.

CI/CD Variables </> 0

Key	Environments
	G

Add variable Cancel

Click on “Add Variable”

Now, we have to the URL

The screenshot shows the GitLab interface for managing CI/CD variables. The left sidebar is open, showing various project settings like Secure, Deploy, Operate, Monitor, Analyze, Settings, General, Integrations, Webhooks, Access tokens, Repository, Merge requests, and CI/CD. The CI/CD section is currently selected. The main content area displays 'Manually-defined pipeline variables' and a table of variables:

Key ↑	Value	Envir
DOCKER_PASS	All (d)
DOCKER_USER	All (d)
SONAR_TOKEN	All (d)

Below the table, it says 'Group variables (inherited)'. A note states: 'These variables are inherited from the parent group.' On the right side, there are sections for 'Visibility' (Visible, Masked, Masked and hidden), 'Flags' (Protect variable, Expand variable reference), and 'Description (optional)' with a text input field and a note about variable precedence.

Uncheck the “Protect Variable” and leave the “Mask Variable” checkbox unchecked

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Variables settings page. On the left, there's a sidebar with project navigation. The main area displays three environment variables:

- DOCKER_PASS**: Value is masked (.....), Environment is All (d), status is Protected and Expanded.
- DOCKER_USER**: Value is masked (.....), Environment is All (d), status is Protected, Masked, and Expanded.
- SONAR_TOKEN**: Value is masked (.....), Environment is All (d), status is Masked and Expanded.

To the right, there are sections for **Visibility**, **Flags**, **Description (optional)**, and a **Value** input field. A red arrow points from the "Key" input field back to the "Key" field on the GitLab browser.

For “Key”, head back to SonarQube browser

- 2 Define the SonarQube Community Build URL environment variable. Still in **Settings > CI/CD > Variables** add a new variable and make sure it is available for your project:
 - Key **SONAR_HOST_URL**
 - In the **Value** field, enter **http://54.209.19.214:9000**
 - Uncheck the **Protect Variable** checkbox.
 - Leave the **Mask Variable** checkbox unchecked.

Copy the “Key” and paste it on the “Key” field on the Gitlab browser

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Variables Settings page. On the left, there's a sidebar with project settings like Secure, Deploy, Operate, Monitor, Analyze, Settings, General, Integrations, Webhooks, Access tokens, Repository, Merge requests, and CI/CD. The CI/CD section is currently selected. The main area displays three environment variables:

Key	Value	Environment
DOCKER_PASS	All (d)
DOCKER_USER	All (d)
SONAR_TOKEN	All (d)

Below the table, it says "Group variables (inherited)". A tooltip for "inherited" says: "These variables are inherited from the parent group." To the right, there's a "Description (optional)" field and a "Key" field containing "SONAR_HOST_URL". Below that is a "Value" field with a placeholder "Enter value here". At the bottom right of the modal, there are "Add variable" and "Cancel" buttons. An orange arrow points from the "Value" field in the GitLab modal to the "Value" field in the SonarQube browser window.

For the “Value”, head back to SonarQube browser

- 2 Define the SonarQube Community Build URL environment variable. Still in **Settings > CI/CD > Variables** add a new variable and make sure it is available for your project:
 - Key
 - In the **Value** field, enter `http://54.209.19.214:9000`
 - Uncheck the **Protect Variable** checkbox.
 - Leave the **Mask Variable** checkbox unchecked.

Copy the “Value” and paste it on the “Value” field on the Gitlab browser

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Variables Settings page for a project named 'sosoebot / Boardgame'. The 'CI/CD' tab is selected in the sidebar. On the right, there's a configuration panel for visibility and flags. A modal window is open, prompting for a new variable key 'SONAR_HOST_URL' and a value 'http://54.209.19.214:9000'. An orange arrow points from the 'Add variable' button in the modal to the 'Add variable' button in the main table.

CI/CD Variables

Key ↑	Value	Environments	Actions
DOCKER_PASS	All (default)	Edit Delete
DOCKER_USER	All (default)	Edit Delete
SONAR_TOKEN	All (default)	Edit Delete

Add variable

Then click on “Add Variable”

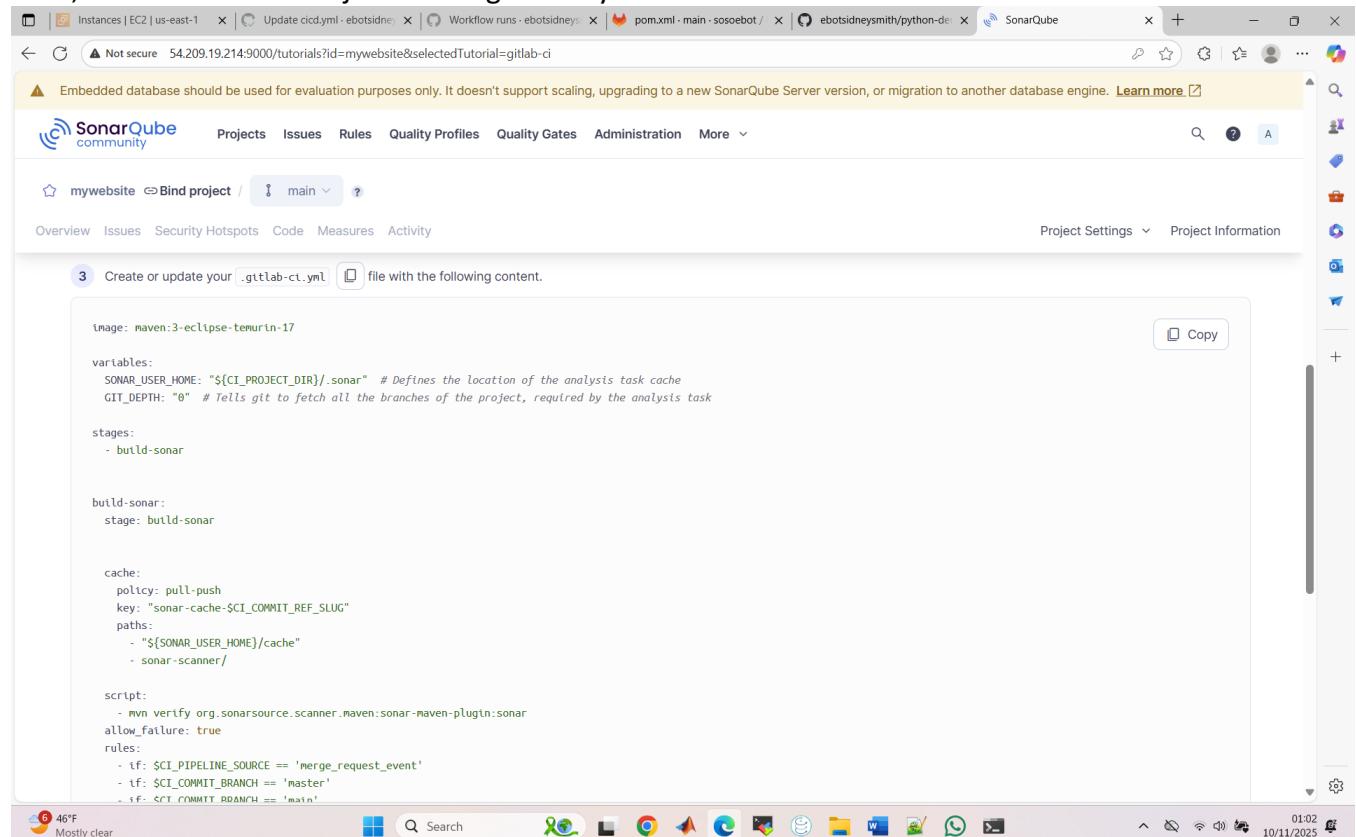
The screenshot shows the same GitLab CI/CD Variables Settings page after the variable has been added. The 'CI/CD Variables' table now includes the new entry for 'SONAR_HOST_URL'. The 'Add variable' button is no longer visible in the modal or the table header.

CI/CD Variables

Key ↑	Value	Environments	Actions
DOCKER_PASS	All (default)	Edit Delete
DOCKER_USER	All (default)	Edit Delete
SONAR_HOST_URL	All (default)	Edit Delete
SONAR_TOKEN	All (default)	Edit Delete

The variables have been added

Now, we have to add the job to the `.gitlab-ci.yml` file



```

image: maven:3-eclipse-temurin-17

variables:
  SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar" # Defines the location of the analysis task cache
  GIT_DEPTH: "0" # Tells git to fetch all the branches of the project, required by the analysis task

stages:
- build-sonar

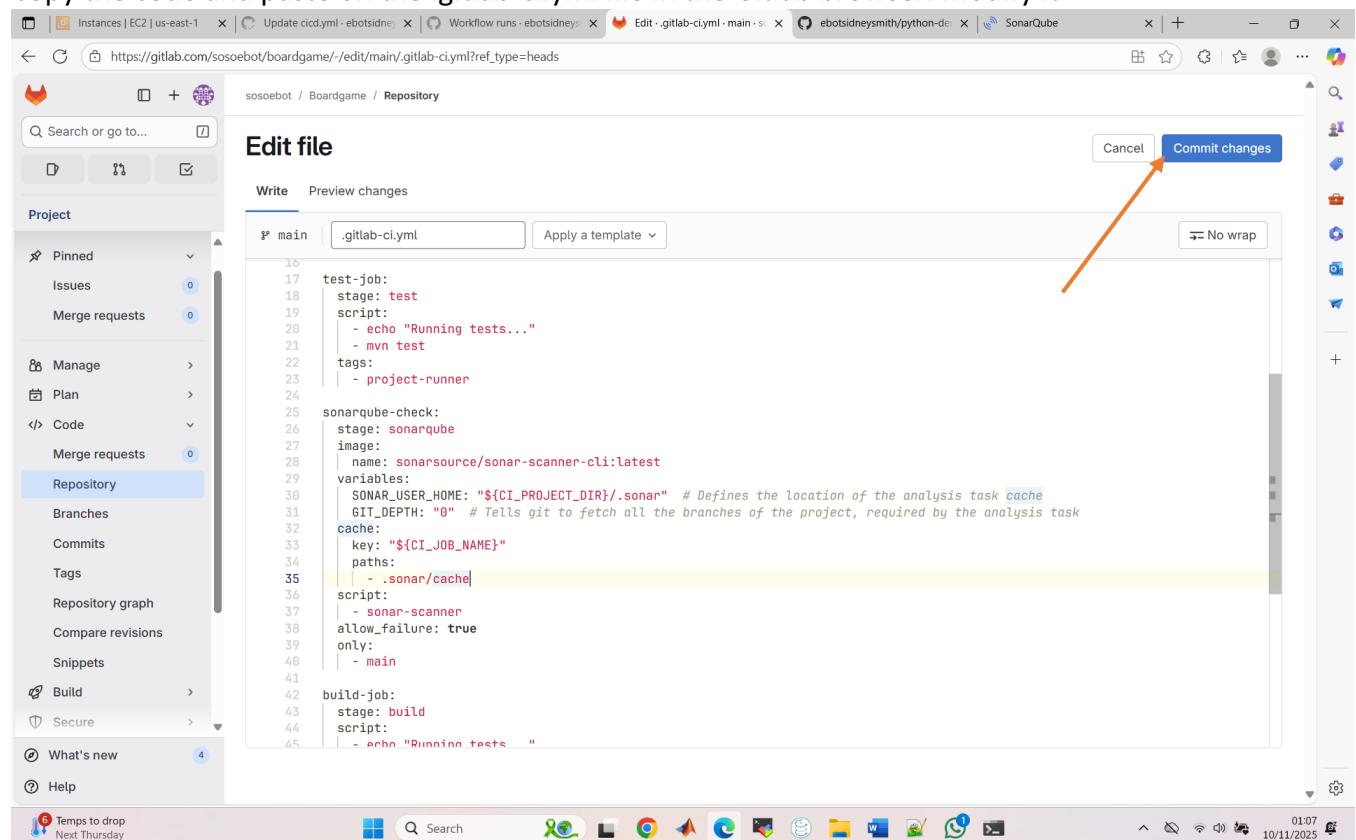
build-sonar:
stage: build-sonar

cache:
  policy: pull-push
  key: "sonar-cache-$CI_COMMIT_REF_SLUG"
  paths:
    - ${SONAR_USER_HOME}/cache
    - sonar-scanner

script:
  - mvn verify org.sonarsource.scanner.maven:sonar-maven-plugin:sonar
allow_failure: true
rules:
  - if: $CI_PIPELINE_SOURCE == 'merge_request_event'
  - if: $CI_COMMIT_BRANCH == 'master'
  - if: $CI_COMMIT_BRANCH == 'main'

```

Copy the code and paste on the `.gitlab-ci.yml` file in the Gitlab browser. Modify it



Edit file

Write Preview changes

main .gitlab-ci.yml Apply a template No wrap

```

16 test-job:
17   stage: test
18   script:
19     - echo "Running tests..."
20     - mvn test
21   tags:
22     - project-runner
23
24 sonarqube-check:
25   stage: sonarqube
26   image:
27     name: sonarsource/sonar-scanner-cli:latest
28   variables:
29     SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar" # Defines the location of the analysis task cache
30     GIT_DEPTH: "0" # Tells git to fetch all the branches of the project, required by the analysis task
31   cache:
32     key: "${CI_JOB_NAME}"
33     paths:
34       - .sonar/cache
35   script:
36     - sonar-scanner
37   allow_failure: true
38   only:
39     - main
40
41 build-job:
42   stage: build
43   script:
44     - echo "Running tests"

```

Then click on “Commit Changes”

Prepared by Sidney Smith

The screenshot shows a GitLab repository interface. On the left, a sidebar lists project management options like Issues, Merge requests, and Repository. The main area displays the contents of the `.gitlab-ci.yml` file. A modal window titled "Commit changes" is overlaid, containing fields for "Commit message" (with placeholder "Edit .gitlab-ci.yml") and "Branch" (with radio buttons for "Commit to the current main branch" and "Commit to a new branch").

```
test-job:
  Commit message
  Branch
  Commit to the current main branch
  Commit to a new branch
  .sonar/cache
  script:
    - sonar-scanner
  allow_failure: true
  only:
    - main

build-job:
  stage: build
  script:
    - echo "Running tests..."
```

Click on “Commit changes” again

The screenshot shows the same GitLab repository after the changes have been committed. A message at the top of the page says "Your changes have been committed successfully." Below this, the `.gitlab-ci.yml` file is shown again, with a note "This GitLab CI configuration is valid. Learn more".

```
# Use Maven image with OpenJDK 17
image: maven:3.8.4-openjdk-17

stages:
- test
- sonarqube
- build
- containerize

variables:
IMAGE_NAME: ebotsidneysmith/demoapp
IMAGE_TAG: boardgame-0.0.1

test-job:
stage: test
script:
- echo "Running tests..."
- mvn test
tags:
- project-runner
```

Adding the “sonarqube-check” job

```
sonarqube-check:
  stage: sonarqube
  image:
    name: sonarsource/sonar-scanner-cli:latest
  variables:
    SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar" # Defines the location of the analysis task cache
    GIT_DEPTH: "0" # Tells git to fetch all the branches of the project, required by the analysis task
  cache:
    policy: pull-push
    key: "sonar-cache-$CI_COMMIT_REF_SLUG"
    paths:
      - "${SONAR_USER_HOME}/cache"
      - sonar-scanner/
  script:
    - sonar-scanner
  allow_failure: true
  only:
    - main
```

Finally, make sure you change the code on the “sonar-project.properties” file to match the project name you used on SonarQube.

```
sonar.projectKey=mywebsite
sonar.qualitygate.wait=true
sonar.projectName=mywebsite
sonar.java.binaries=.
```

Part 3: Adding “build-job”

Now we will add the job to do the code analysis.

Adding the “build-job”

```
build-job:
  stage: build
  script:
    - echo "Running tests..."
    - mvn package
  artifacts:
    paths:
      - target/*.jar
  tags:
    - project-runner
```

Part 4: Adding “build_image_push-job”

Now we will add the job to do the code analysis.

Adding the “build_image_push-job”

```
build_image_push-job:  
  image: docker:27.1.1  
  services:  
    - docker:27.1.1-dind  
  variables:  
    DOCKER_TLS_CERTDIR: ""  
  stage: containerize  
  before_script:  
    - docker login -u $DOCKER_USER -p $DOCKER_PASS  
  script:  
    - docker build -t $IMAGE_NAME:$IMAGE_TAG .  
    - docker push $IMAGE_NAME:$IMAGE_TAG
```

Adding Variables

Here we have some variables, namely DOCKER_USER and DOCKER_PASS

Adding Docker user

The screenshot shows the GitLab CI/CD Variables Settings page. On the left, there's a sidebar with project navigation. The main area has a 'Save changes' button at the top. It shows four variables under 'CI/CD Variables':

Key	Value	Envir
DOCKER_PASS	All (d)
DOCKER_USER	All (d)
SONAR_HOST_URL	All (d)
SONAR_TOKEN	All (d)

Below the table, it says 'Group variables (inherited)' with a note: 'These variables are inherited from the parent group.' To the right, there are sections for 'VISIBILITY' (Visible, Masked, Masked and hidden), 'Flags' (Protect variable, Expand variable reference), and a 'Description (optional)' field with placeholder text 'The description of the variable's value or usage.' At the bottom are 'Save changes', 'Delete variable', and 'Cancel' buttons.

Adding the docker password

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Variables Settings page for the project 'sosobot / Boardgame'. The left sidebar is collapsed, and the main content area displays the following:

Project variables

Variables can be accidentally exposed in a job log, or maliciously sent to a third party server. The masked variable feature allows you to expose variable values, but is not a guaranteed method to prevent malicious users from accessing variables. How

CI/CD Variables </> 4

Key ↑	Value	Envir
DOCKER_PASS	All (d)
DOCKER_USER	All (d)
SONAR_HOST_URL	All (d)
SONAR_TOKEN	All (d)

Group variables (inherited)

These variables are inherited from the parent group.

CI/CD Variables </> 0

Key	Environments
	There are no variables yet.

VISIBILITY

- Visible**
Can be seen in job logs.
- Masked**
Masked in job logs but value can be revealed in CI/CD settings. Requires values to meet regular expressions requirements.
- Masked and hidden**
Masked in job logs, and can never be revealed in the CI/CD settings after the variable is saved.

Flags

- Protect variable**
Export variable to pipelines running on protected branches and tags only. [Edit](#)
- Expand variable reference**
\$ will be treated as the start of a reference to another variable.

Description (optional)

The description of the variable's value or usage.

Key

DOCKER_PASS

You can use CI/CD variables with the same name in different places, but the variables might overwrite each other. What is the order of precedence for variables?

Value

(Redacted)

Buttons

Save changes | Delete variable | Cancel

STEP 9: Enabling Jobs to run sequentially

We have to make the code sequentially since some jobs/stages depends on each other.

To do this we have to add stages as follows:

```
stages:  
  - test  
  - sonarqube  
  - build  
  - containerize
```

Final Code

```
# Use Maven image with OpenJDK 17  
image: maven:3.8.4-openjdk-17  
  
stages:  
  - test  
  - sonarqube  
  - build  
  - containerize }  
  
variables:  
  IMAGE_NAME: ebotsidneymsmith/demoapp  
  IMAGE_TAG: boardgame-0.0.1  
  
test-job:  
  stage: test ←  
  script:  
    - echo "Running tests..."  
    - mvn test  
  tags:  
    - project-runner  
  
sonarqube-check:  
  stage: sonarqube ←  
  image:  
    name: sonarsource/sonar-scanner-cli:latest  
  variables:  
    SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar" # Defines the location of the analysis  
task cache  
  GIT_DEPTH: "0" # Tells git to fetch all the branches of the project, required by the  
analysis task  
  cache:  
    policy: pull-push  
    key: "sonar-cache-$CI_COMMIT_REF_SLUG"  
    paths:
```

```
- "${SONAR_USER_HOME}/cache"
- sonar-scanner/
script:
- sonar-scanner
allow_failure: true
only:
- main

build-job:
stage: build ←
script:
- echo "Running tests..."
- mvn package
artifacts:
paths:
- target/*.jar
tags:
- project-runner

build_image_push-job:
image: docker:27.1.1
services:
- docker:27.1.1-dind
variables:
DOCKER_TLS_CERTDIR: ""
stage: containerize ←
before_script:
- docker login -u $DOCKER_USER -p $DOCKER_PASS
script:
- docker build -t $IMAGE_NAME:$IMAGE_TAG .
- docker push $IMAGE_NAME:$IMAGE_TAG
```

STEP 10: Result

Click on “Build”

The screenshot shows the 'Variables' section of the GitLab CI/CD Settings. On the left, there's a sidebar with project navigation. The 'Build' section is expanded, showing 'Pipelines' which is highlighted with an orange arrow. Below it are 'Jobs', 'Pipeline editor', 'Pipeline schedules', and 'Artifacts'. The main content area has a heading 'Defined pipeline variables' with a sub-section 'Manually-defined variables in the pipeline details page after running a pipeline manually. Learn more.' A checkbox 'Allow merge request pipelines to access protected variables and runners' is checked. At the bottom are 'Save changes' and 'Project variables' sections.

Select “Pipeline”

The screenshot shows the 'Pipelines' page. The sidebar has the 'Build' section expanded, with 'Pipelines' selected and highlighted with an orange arrow. The main table lists one pipeline: 'Edit sonar-project.properties' (Status: Passed, Pipeline ID: #18, Duration: 00:04:07, Run time: 12 minutes ago). The pipeline has four stages, all of which are green with checkmarks. An orange arrow points from the 'Passed' status in the table to the green checkmark in the stages column. The table columns are 'Status', 'Pipeline', 'Created by', 'Stages', and 'Actions'.

You can see that the build is successful.

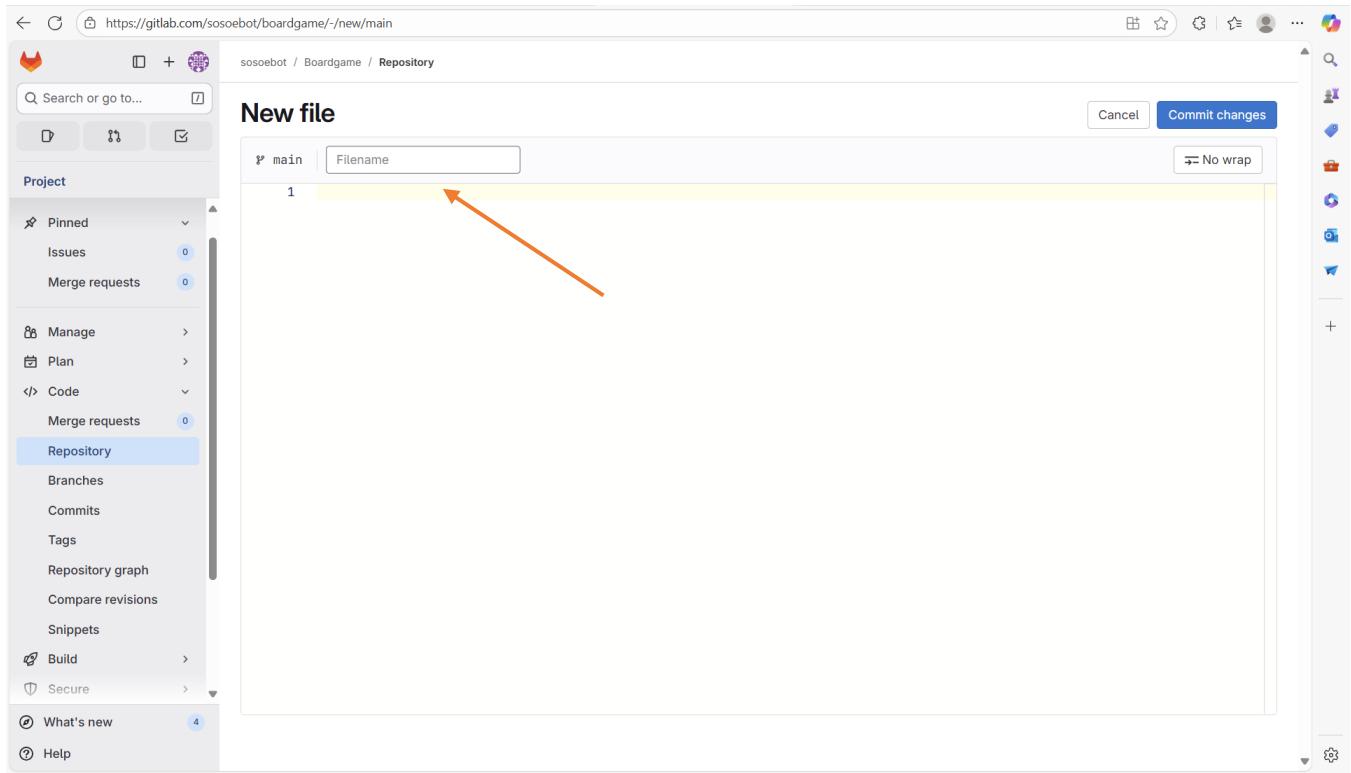
Now, let us try to make some changes in the Gitlab repo and commit the changes. Then see if the Pipeline will start automatically.

The screenshot shows the GitLab interface for the 'Boardgame' repository. The left sidebar contains project management options like Plan, Code, Build, Secure, Deploy, Operate, Monitor, Analyze, and Settings. The main area displays a list of files in the 'main' branch. A prominent orange arrow points from the text "We will add a text file. Click on "+"" to the blue '+' button located at the top right of the file list. The '+' button has a dropdown menu open, showing options such as 'New file', 'Upload file', 'New directory', 'This repository', 'New branch', and 'New tag'. The status bar at the bottom indicates it's 02:01 on 10/11/2025.

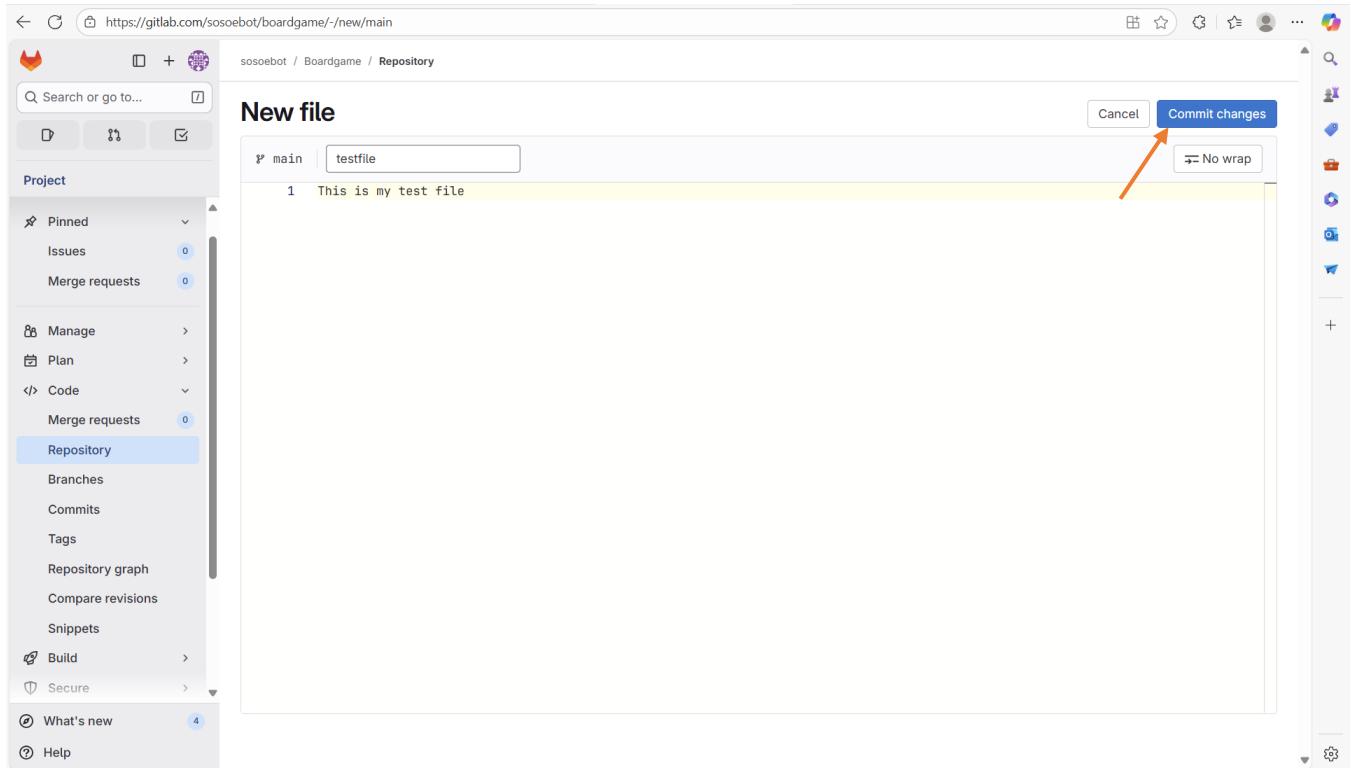
We will add a text file. Click on “+”

This screenshot is identical to the previous one, showing the 'Boardgame' repository in GitLab. The '+' button's dropdown menu is open, and the 'New file' option is highlighted with an orange arrow. The rest of the interface, including the sidebar and file list, remains the same.

Select “New File”

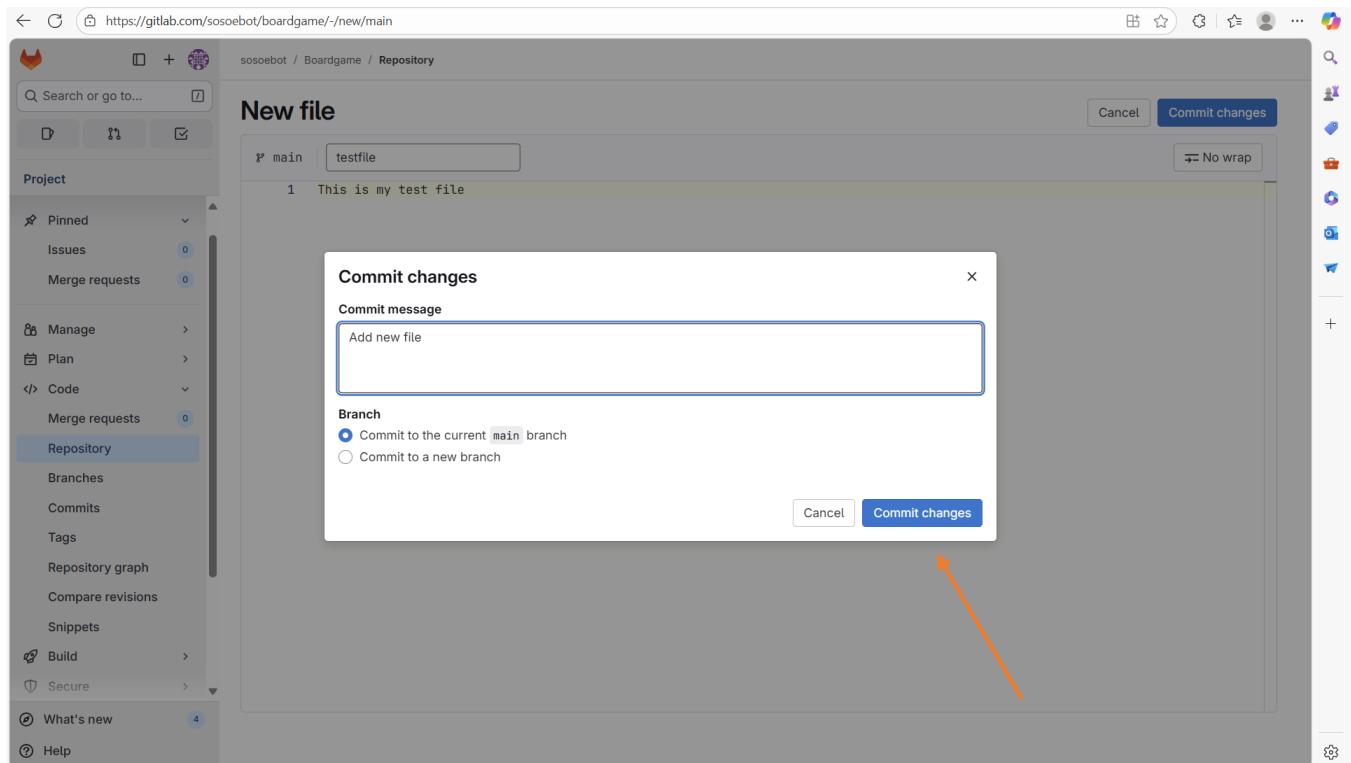


Give the file a name and add some text on the file

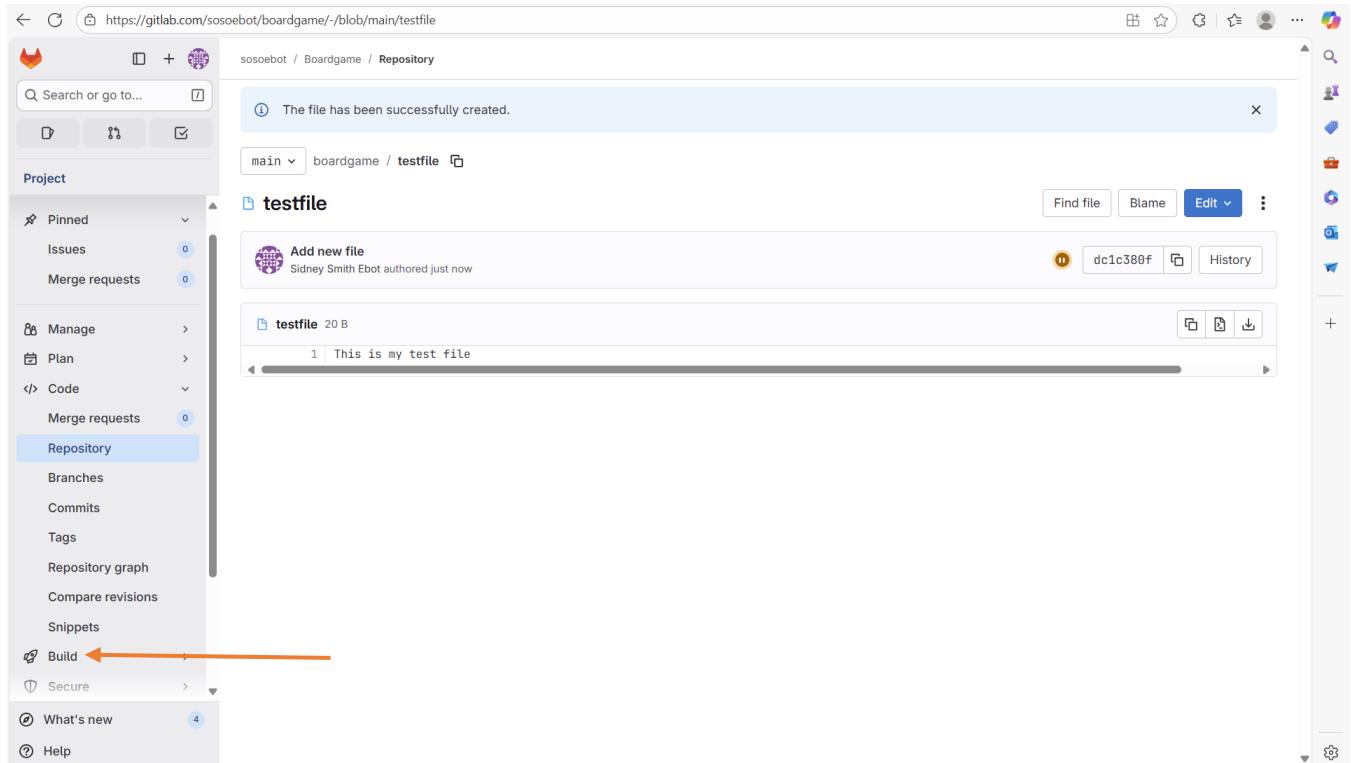


Then commit the changes by clicking on “Commit changes”

Prepared by Sidney Smith



Then click on “Commit changes” again



Then click on “Build”

Prepared by Sidney Smith

The screenshot shows a GitLab repository named 'sosobot / Boardgame'. A message at the top says 'The file has been successfully created.' Below it, a file named 'testfile' is shown with the content '1 | This is my test file'. In the sidebar, under the 'Repository' section, there is a dropdown menu with several options: Pipelines, Jobs, Pipeline editor, Pipeline schedules, and Artifacts. The 'Pipelines' option is highlighted with a blue background and has an orange arrow pointing to it from the bottom right.

And select “Pipeline”

The screenshot shows the 'Pipelines' page for the 'Boardgame' project. It lists two pipelines:

Status	Pipeline	Created by	Stages	Actions
Running	Add new file #19 ↗ main ↘ dc1c380f	Sidney Smith Ebot	Passed, Pending, Pending, Pending	Stop, Discard
Passed	Edit sonar-project.properties #18 ↗ main ↘ 3b3b4687	Sidney Smith Ebot	Passed, Passed, Passed, Passed	Discard

The sidebar on the left is identical to the previous screenshot, showing the 'Pipelines' option selected in the dropdown menu.

You can see that the pipeline has started running automatically

Prepared by Sidney Smith

The screenshot shows a GitLab pipeline interface. At the top, a message indicates a successful pipeline creation by 'Sidney Smith Ebot' for commit 'dc1c380f' 4 minutes ago. Below this, the pipeline structure is displayed as a sequence of stages: test, sonarqube, build, and containerize. Each stage contains a single job: 'test-job', 'sonarqube-check', 'build-job', and 'build_image_push-job' respectively. All jobs are marked as passed (green checkmarks). The pipeline editor sidebar on the left is visible, showing the current selection is 'Pipelines'.

You can see that the Pipeline is successful. Then check the SonarQube results

The screenshot shows a SonarQube dashboard for the project 'mywebsite'. The main header indicates 'Not secure' and the URL '54.209.19.214:9000/dashboard?id=mywebsite&codeScope=overall'. The dashboard displays a summary of the analysis: '1.3k Lines of Code' and 'Version not provided'. A prominent green 'Passed' status is shown under the 'Quality Gate' section. Below this, a callout box encourages users to 'Discover 'Clean as You Code!'' and provides options to 'Take the Tour' or 'Not now'. The 'Overall Code' tab is selected, showing metrics for 'New Code' and 'Overall Code'. Under 'New Code', there are 2 open issues in 'Security'. Under 'Overall Code', there are 18 open issues in 'Reliability' and 41 open issues in 'Maintainability'. The right side of the dashboard includes 'Project Settings' and 'Project Information' dropdowns, along with various navigation links like 'Overview', 'Issues', 'Security Hotspots', 'Code', 'Measures', and 'Activity'.

Accepted issues
0 Valid issues that were not fixed

Coverage
0.0% On 254 lines to cover.

Duplications
0.0% On 1.6K lines.

Security Hotspot
1 E

Activity

Graph type Issues

Issues New Code

60
40
20
0

01:45 01:50 01:55 02 AM

October 11, 2025 at 2:05 AM Quality Gate: Passed
not provided

New analysis: +0 Issues

October 11, 2025 at 1:44 AM Quality Gate: Passed
First analysis: 56 Issues • 0.0% Coverage • 0.0% Duplications

See full history of analyses

You can see that the code passed the Quality Gate analysis.

Now, let me check my Docker hub if there is a new folder there containing the image

https://hub.docker.com/repositories/ebotsidneysmith

hub Explore My Hub Search Docker Hub CtrlK

ebotsidneysmith Docker Personal

Repositories All repositories within the ebotsidneysmith namespace.

Create a repository

Search by repository name All content

Name	Last Pushed	Contains	Visibility	Scout
ebotsidneysmith/demoapp	1 minute ago	IMAGE	Public	Inactive
ebotsidneysmith/demorepo	14 days ago	IMAGE	Public	Inactive

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

Cookies Settings Reject All Accept All Cookies

You can see that the image has been built and push to my docker hub

ebotsidneysmith/demoapp

Last pushed 2 minutes ago • Repository size: 281.1 MB

Add a description *(1)*

Add a category *(1)*

General Tags Image Management BETA Collaborators Webhooks Settings

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
boardgame-0.0.1		Image	less than 1 day	2 minutes

See all

Docker commands

To push a new tag to this repository:

```
docker push ebotsidneysmith/demoapp:tagname
```

buildcloud

Build with Docker Build Cloud

Accelerate image build times with access to cloud-based builders and shared cache.

Docker Build Cloud executes builds on optimally-dimensioned cloud infrastructure with dedicated per-organization isolation.

Get faster builds through shared caching across your team, native multi-platform support, and encrypted data transfer - all without managing infrastructure.

Go to Docker Build Cloud →

This is the built image.