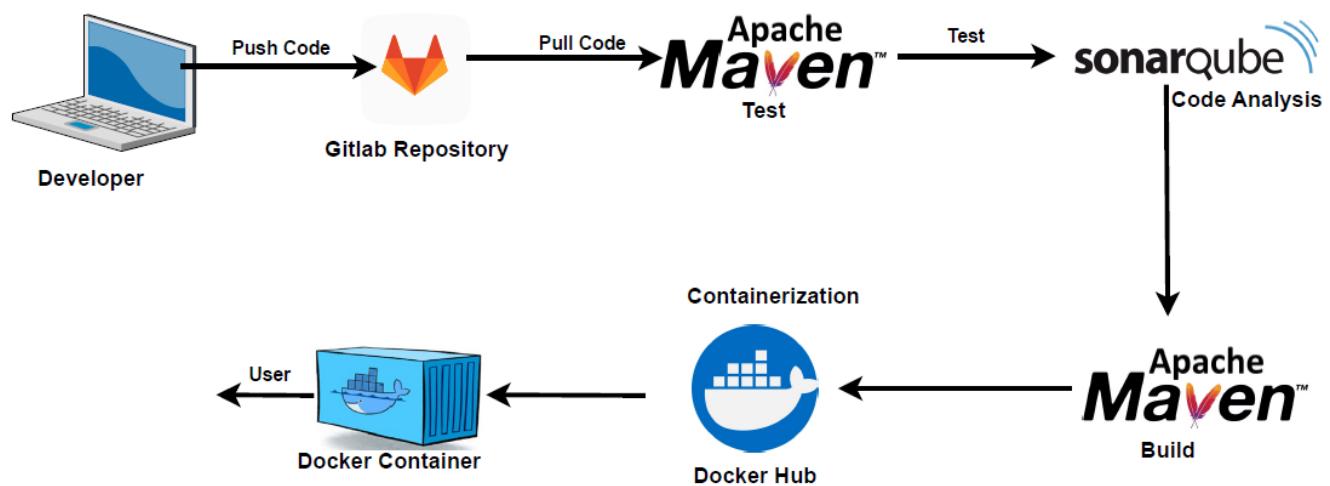


Deploying Java Application using GitLab Pipeline Project with Maven, SonarQube, Docker

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

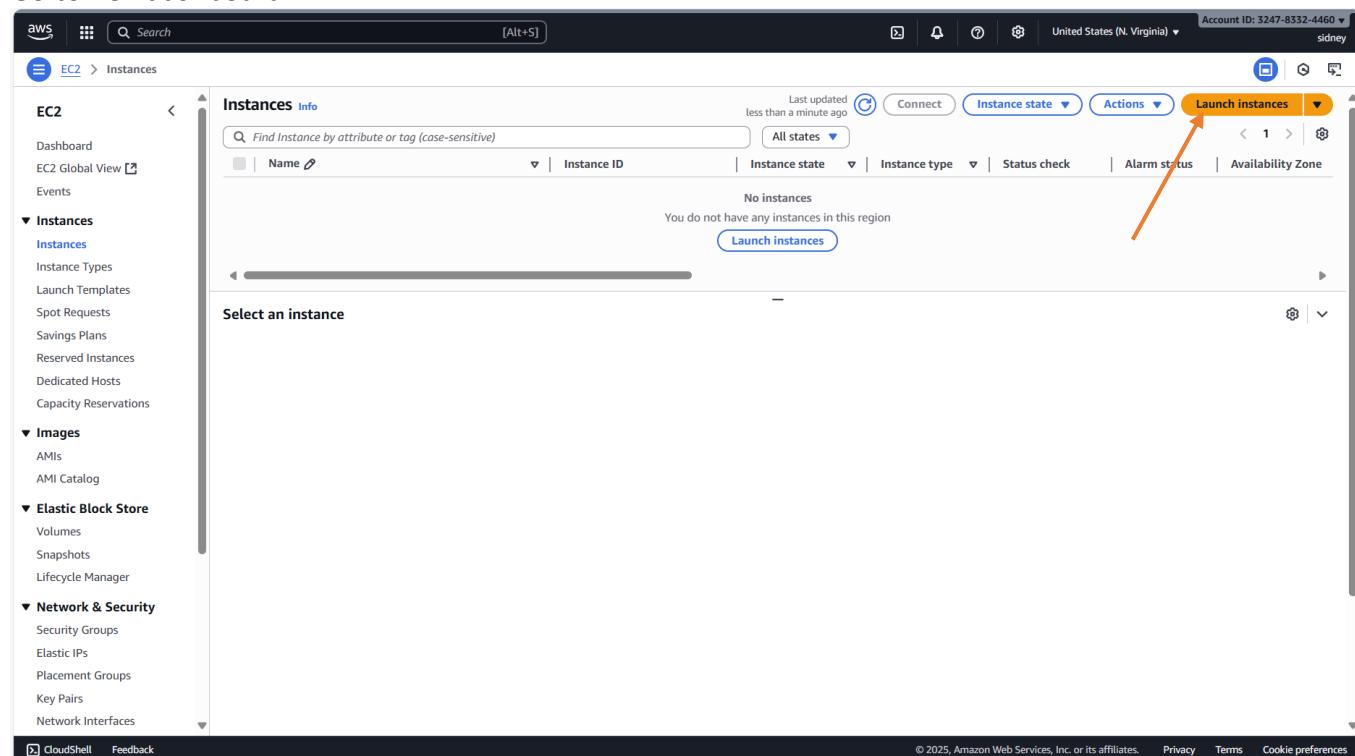
You will learn how to write pipeline scripts for each of these jobs. The Test and Build jobs for the Java app will run on the Project Runner, while the SonarQube code quality check and Containerize job will run on GitLab's Shared Runner.

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' step, a red arrow points from the 'Name' input field, which contains 'e.g. My Web Server', to the 'Add additional tags' button.

Name and tags [Info](#)
Name
e.g. My Web Server [Add additional tags](#)

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](#)

Recent [Quick Start](#)

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Debian

[Browse more AMIs](#) Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)
Amazon Linux 2023 kernel-6.1 AMI
ami-052064a798f08fd3 (64-bit (x86), uefi-preferred) / ami-089f6a79b0e02648a (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

[Cancel](#) [Launch instance](#) [Preview code](#)

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' step, the 'Name' input field now contains 'Gitlab-Server'. A red arrow points from the 'Name' input field to the 'Add additional tags' button.

Name and tags [Info](#)
Name
Gitlab-Server [Add additional tags](#)

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](#)

Recent [Quick Start](#)

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Debian

[Browse more AMIs](#) Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)
Amazon Linux 2023 kernel-6.1 AMI
ami-052064a798f08fd3 (64-bit (x86), uefi-preferred) / ami-089f6a79b0e02648a (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Amazon Linux 2023 (Kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

[Cancel](#) [Launch instance](#) [Preview code](#)

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-026fccc88446aa0bf (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](#)

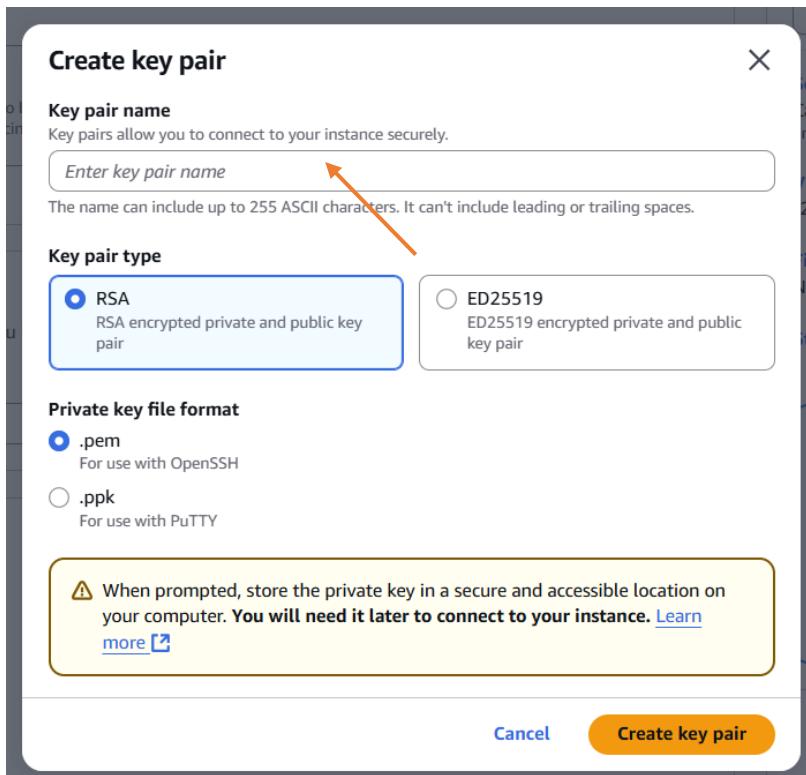
Instance type

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

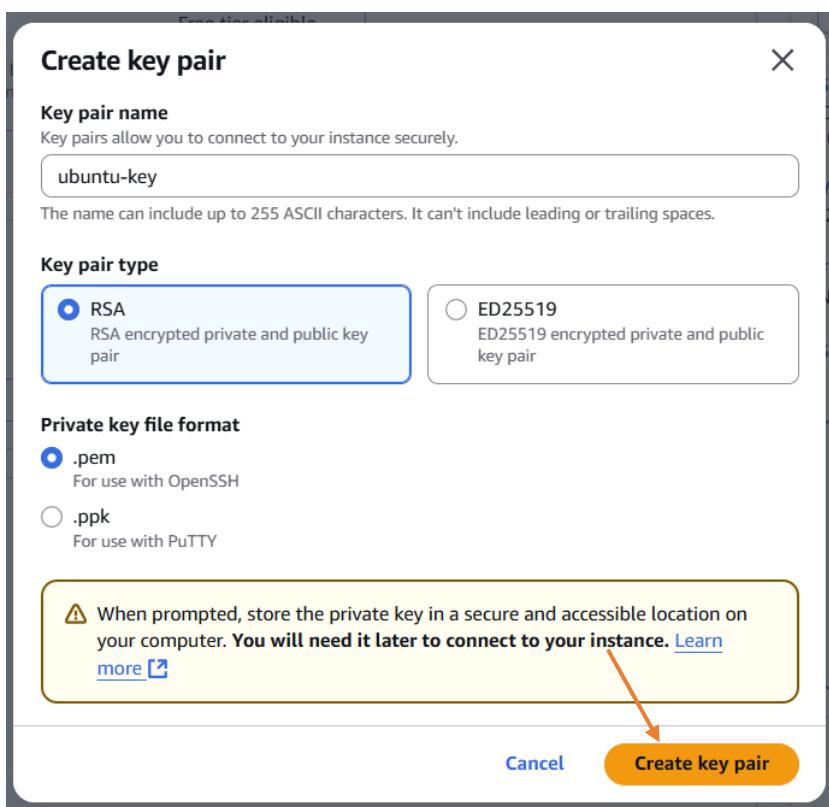
All generations Compare instance types

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

 ▼  [Create new key pair](#)

Scroll down to “Network Settings”

▼ Network settings [Info](#) 

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 Anywhere
Helps you connect to your instance

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

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

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

Scroll down to “Configuration Storage” and make the value “30GiB”

▼ Configure storage [Info](#) [Advanced](#)

1x GiB Root volume, 3000 IOPS, Not encrypted

 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

 C
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](#) 

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.

Click on “Instance”

The screenshot shows the AWS EC2 Instances page. On the left, a navigation sidebar lists 'EC2' (selected), 'Dashboard', 'EC2 Global View', 'Events', 'Instances' (selected), 'Instances 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 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 Dashboard, Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and more. The main content area displays a table titled 'Instances (1) Info'. The table has columns for Name (Gitlab-Server), Instance ID (i-097749be6331b247f), Instance state (Running), Instance type (t2.micro), Status check (2/2 checks passed), Alarm status (View alarms +), and Availability Zone (us-east-1d). A search bar at the top allows filtering by attribute or tag. Action buttons include '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, indicating it is selected. The rest of the interface and data 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 the "Security" tab selected in the detailed view for the "Gitlab-Server" instance. A red arrow points to the "Security groups" section, which lists "sg-0608924d3e2da1d98 (launch-wizard-55)". Below this, there are sections for "Inbound rules" and "Outbound rules", each with a table of security group rule details.

Click on “Security Group”

Prepared by Sidney Smith

This screenshot shows the AWS EC2 Security Groups page for a specific security group named "sg-0608924d3e2da1d98 - launch-wizard-55". The left sidebar is expanded to show categories like Instances, Images, and Network & Security. The main content area displays the security group details and its inbound rules. An orange arrow points from the text "Click on ‘Edit Inbound Rules’" to the "Edit inbound rules" button at the top right of the Inbound rules table.

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-0597052c04e8c6642	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-07ecd7e33348d9f29	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-05de55647cd9f70f4	IPv4	HTTPS	TCP	443	0.0.0.0/0

Click on “Edit Inbound Rules”

This screenshot shows the "Edit inbound rules" configuration page for the same security group. It lists three existing rules and a new "Add rule" button highlighted with an orange arrow. The "Add rule" button is located at the bottom left of the table.

Type	Protocol	Port range	Source	Description - optional
SSH	TCP	22	Custom	0.0.0.0/0
HTTP	TCP	80	Custom	0.0.0.0/0
HTTPS	TCP	443	Custom	0.0.0.0/0

Click on “Add Rule”

This screenshot shows the "Edit inbound rules" configuration page after a new rule has been added. The new rule is listed at the bottom of the table, featuring a custom type ("Custom TCP"), port range ("0"), and source ("Custom"). An orange arrow points to the "0" value in the port range field.

Type	Protocol	Port range	Source	Description - optional
SSH	TCP	22	Custom	0.0.0.0/0
HTTP	TCP	80	Custom	0.0.0.0/0
HTTPS	TCP	443	Custom	0.0.0.0/0
-	Custom TCP	0	Custom	0.0.0.0/0

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 Security Groups console for a security group named "sg-0608924d3e2da1d98 - launch-wizard-55". A green success message at the top states: "Inbound security group rules successfully modified on security group (sg-0608924d3e2da1d98 | launch-wizard-55) Details". The "Details" section shows the security group name, ID, owner, and rule counts. The "Inbound rules" tab is selected, displaying four rules:

Name	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-02b889544ff3924bd	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 Instances console. On the left, the navigation menu is visible under the "EC2" section. In the main area, the "Instances (1) Info" table shows one instance named "Gitlab-Server" with the following details:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Gitlab-Server	i-097749be6331b247f	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1d

An orange arrow points from the text "Select the EC2 instance" to the checkbox next to the "Gitlab-Server" instance name.

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. The 'SSH client' tab is active. It provides instructions for connecting:

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

Below these instructions is an example command:

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

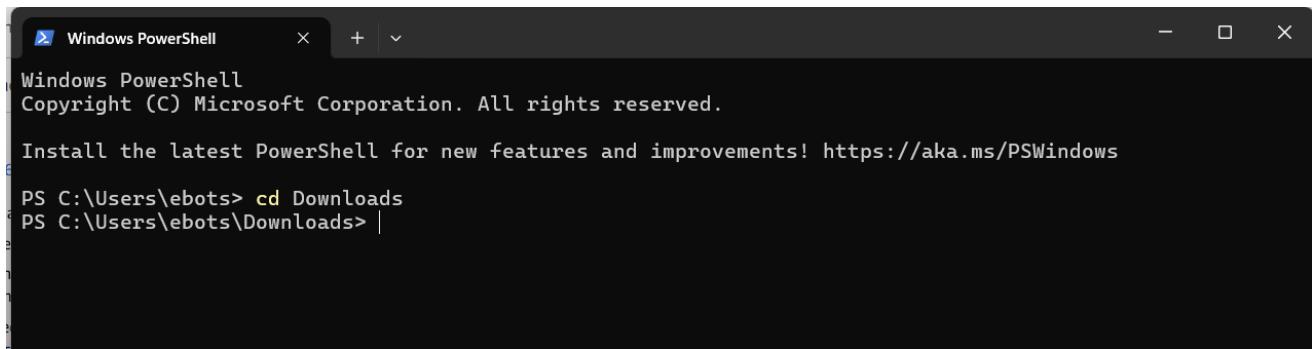
A note at the bottom states: "Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username."

An orange arrow points to the command line input field where the SSH command is being typed.

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



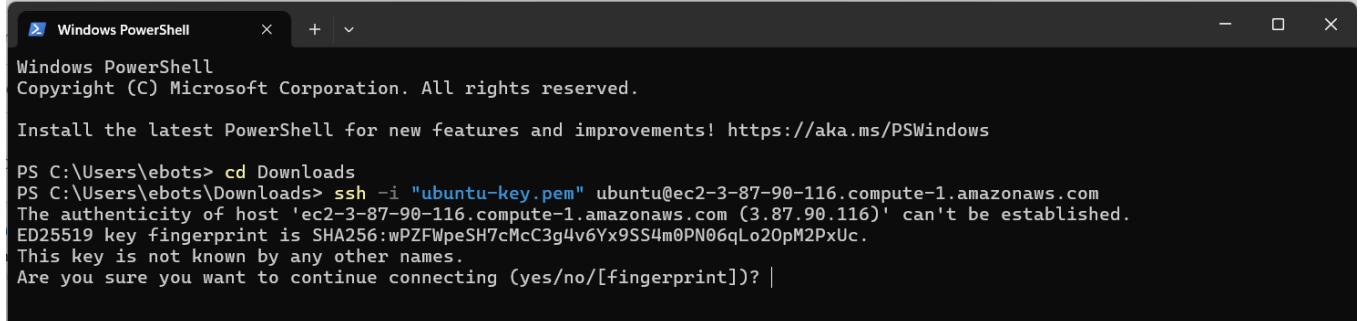
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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
```

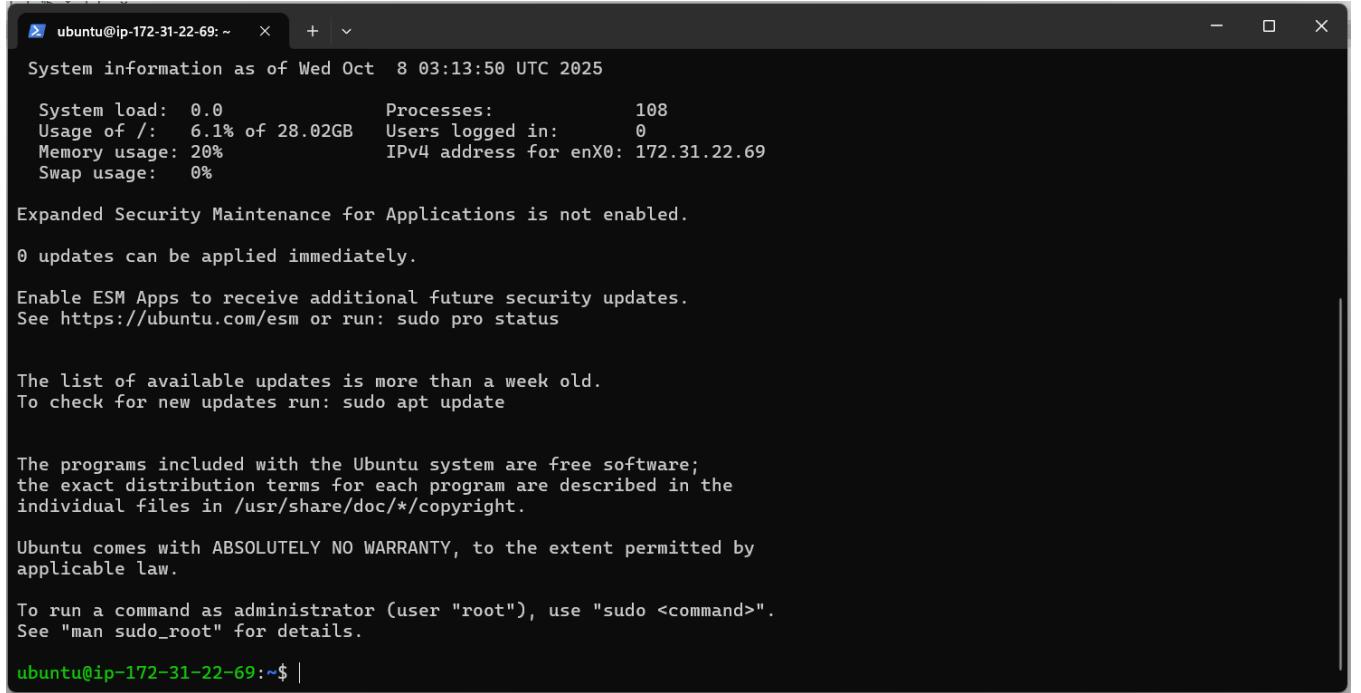


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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:wPZFWeSH7eMcC3g4v6Yx9SS4m0PN06ql020pM2PxUc.
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.

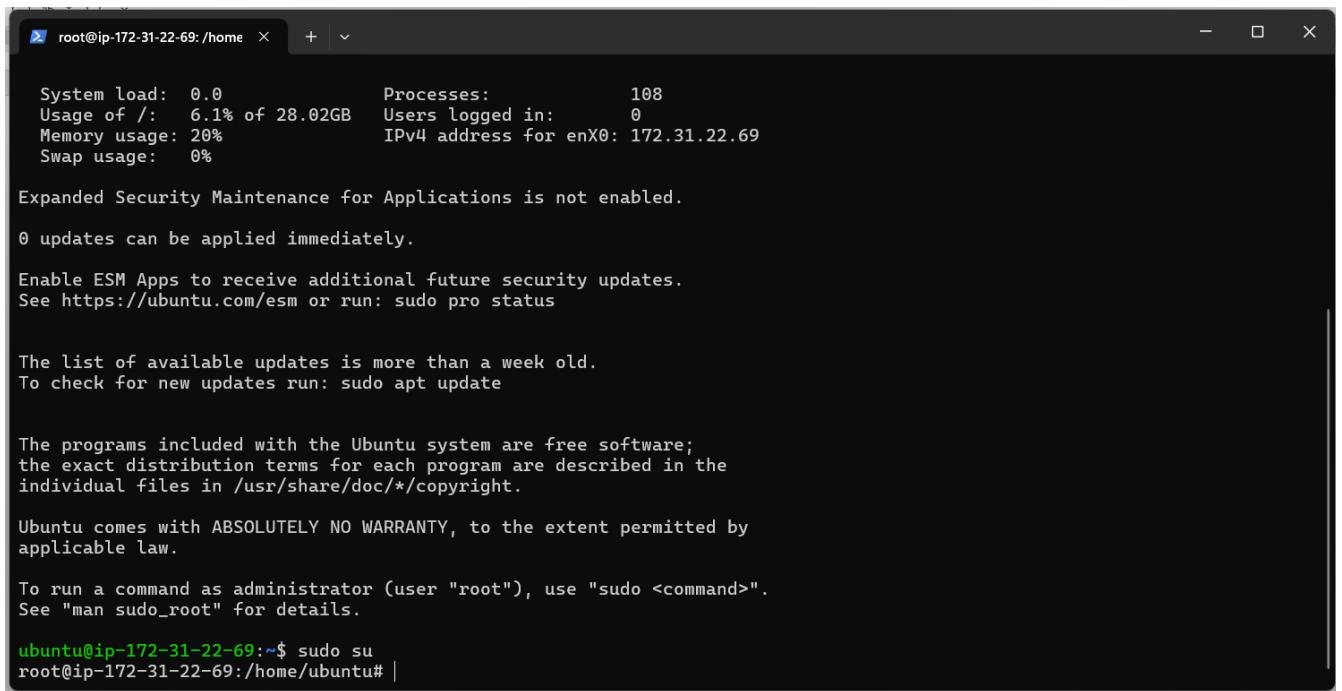
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

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.

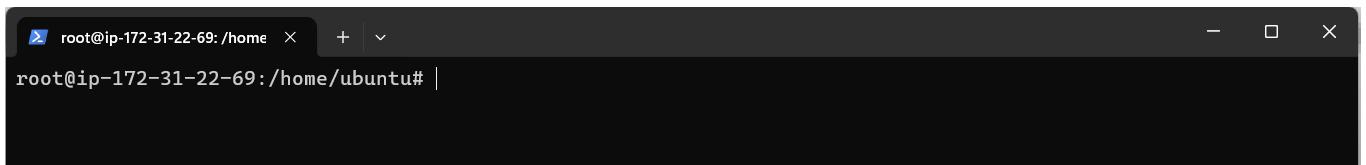
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

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
```



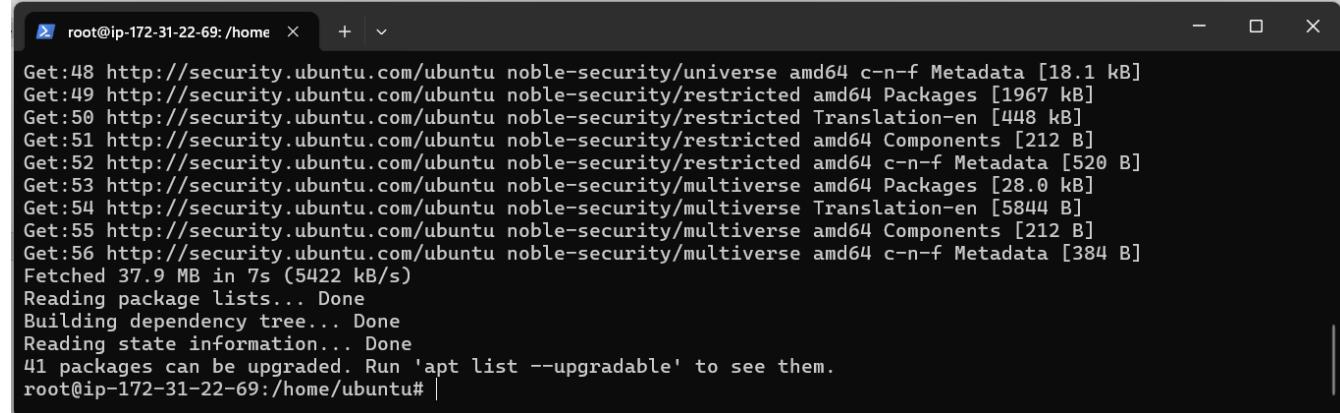
```
root@ip-172-31-22-69:/home  +  ~
root@ip-172-31-22-69:/home/ubuntu# |
```

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:

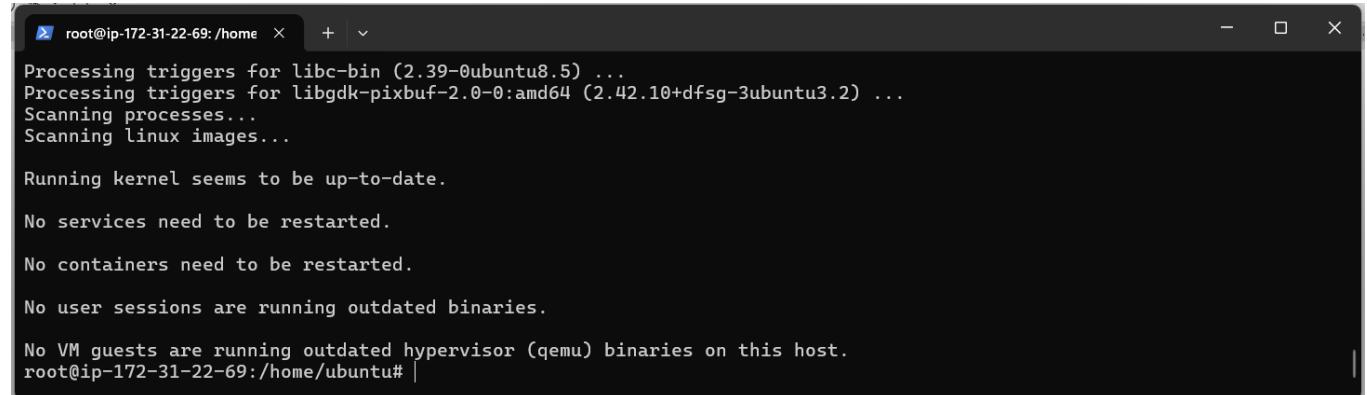
```
sudo apt update
```



```
root@ip-172-31-22-69:/home ~ + | 
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 ~ + | 
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.10+dfsg-3ubuntu3.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

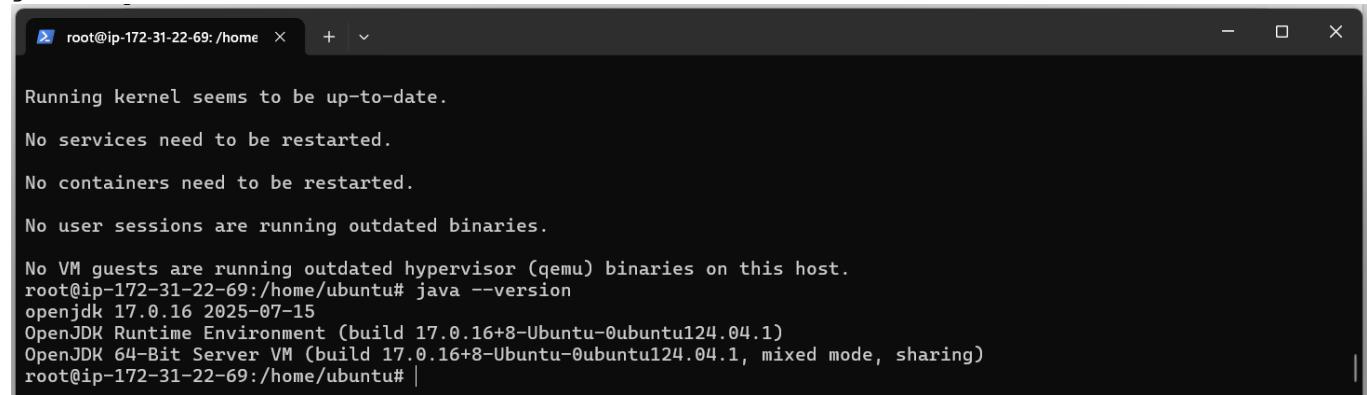
No containers need to be restarted.

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 ~ + | 
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 link, with "Copy link address" being the selected option. The menu also includes "Save link as...", "Inspect", "Checksums", and "Signature". The "Checksums" column lists SHA-512 hashes for each file type. The "Signature" column lists GPG signatures for each file type.

File Type	Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.9.11-bin.tar.gz	apache-maven-3.9.11-bin.tar.gz.sha512	apache-maven-3.9.11-bin.tar.gz.asc
Binary zip archive	apache-maven-3.9.11-bin.zip	apache-maven-3.9.11-bin.zip.sha512	apache-maven-3.9.11-bin.zip.asc
Source tar.gz archive	apache-maven-3.9.11-src.tar.gz	apache-maven-3.9.11-src.tar.gz.sha512	apache-maven-3.9.11-src.tar.gz.asc
Source zip archive	apache-maven-3.9.11-src.zip	apache-maven-3.9.11-src.zip.sha512	apache-maven-3.9.11-src.zip.asc

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

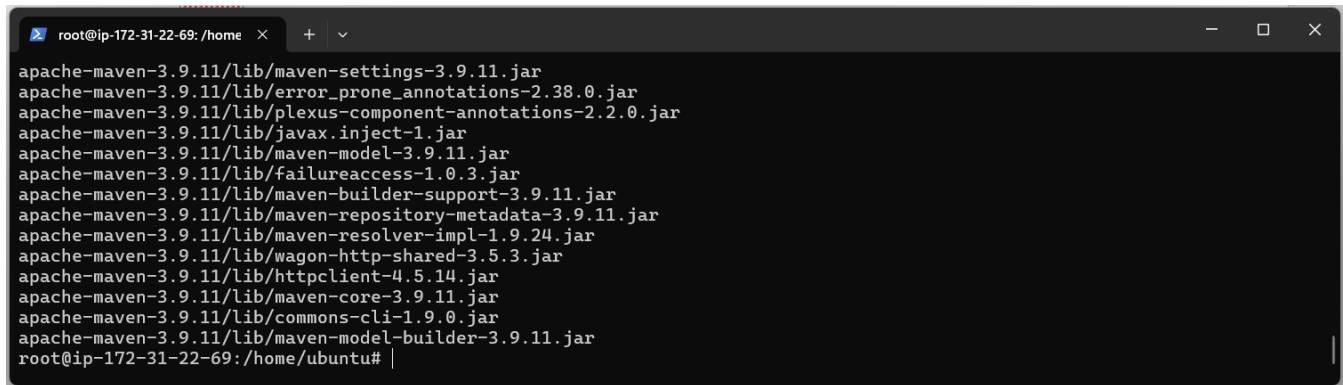
wget [Link]

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

A terminal window on a Linux system (Ubuntu) shows the command `root@ip-172-31-22-69:/home# wget https://dlcdn.apache.org/maven/maven-3/3.9.11/binaries/apache-maven-3.9.11-bin.tar.gz` being entered. The response from the terminal indicates that the file was not found: `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`.

Part 2: Untar the Downloaded file

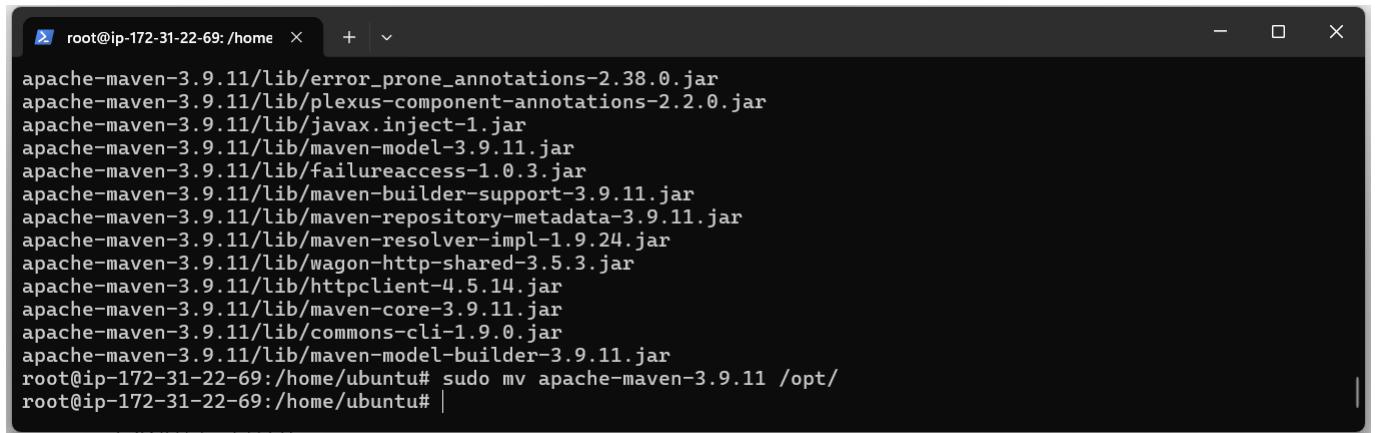
```
tar -xvf apache-maven-3.9.11-bin.tar.gz
```



```
root@ip-172-31-22-69:/home/ubuntu# ls apache-maven-3.9.11/lib/
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
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/
```

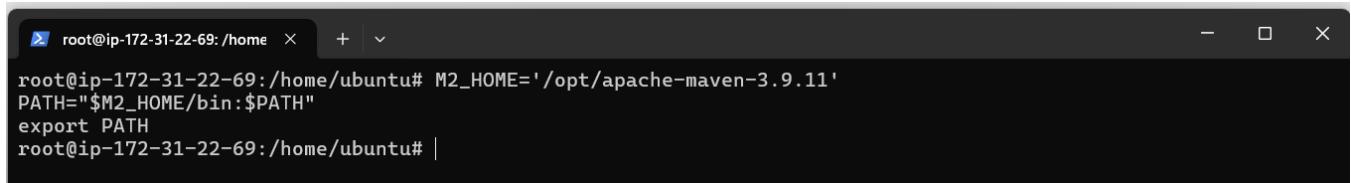


```
root@ip-172-31-22-69:/home/ubuntu# ls apache-maven-3.9.11/lib/
apache-maven-3.9.11/lib/error_prone_annotations-2.38.0.jar
apache-maven-3.9.11/lib/plexus-component-annotations-2.2.0.jar
apache-maven-3.9.11/lib/javax.inject-1.jar
apache-maven-3.9.11/lib/maven-model-3.9.11.jar
apache-maven-3.9.11/lib/failureaccess-1.0.3.jar
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
root@ip-172-31-22-69:/home/ubuntu# sudo mv apache-maven-3.9.11 /opt/
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).

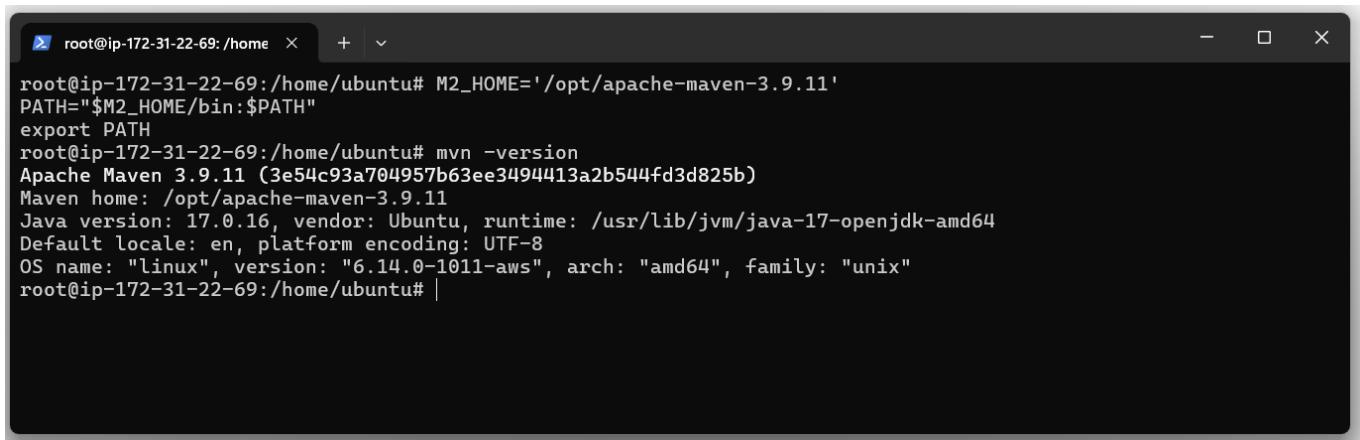
```
M2_HOME='/opt/apache-maven-3.9.11'
PATH="$M2_HOME/bin:$PATH"
export PATH
```



```
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# |
```

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

```
mvn -version
```



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

```
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
```

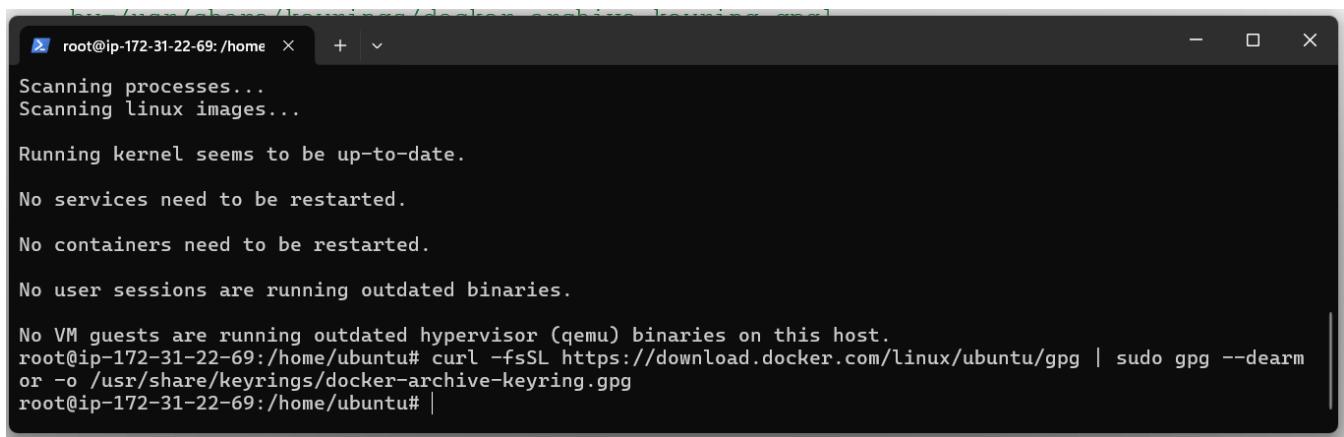


A terminal window titled 'root@ip-172-31-22-69:/home' showing the output of package management commands. It includes status messages like 'Unpacking apt-transport-https', 'Setting up apt-transport-https', and kernel status messages such as 'Running kernel seems to be up-to-date.' and 'No services need to be restarted.'

```
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 a GPG key. It includes status messages like 'Scanning processes...' and 'Running kernel seems to be up-to-date.'

```
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 ~ + ^

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 ~ + ^

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 ~ + ^

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.

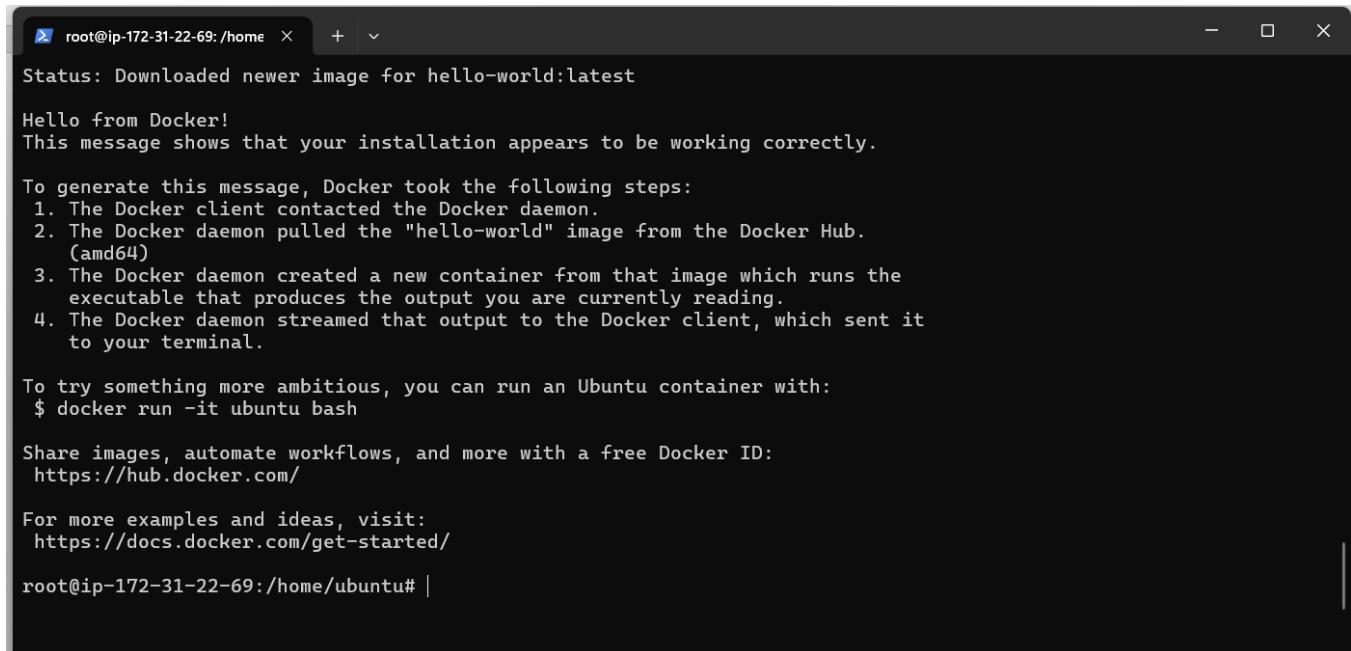
No user sessions are running outdated binaries.

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

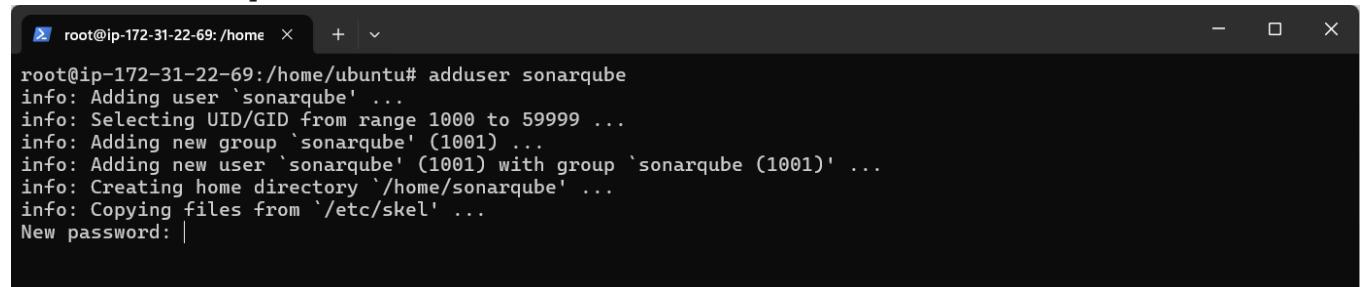
Now, we have to do “**Static Code Analysis**” using SonarQube. Installing SonarQube on an Ubuntu EC2 instance involves several key steps.

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

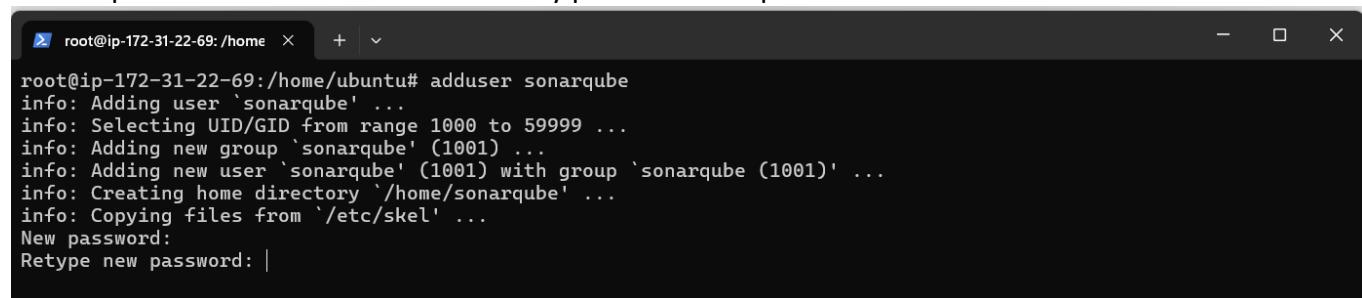
First, we have to create a SonarQube user using the command:

```
adduser sonarqube
```



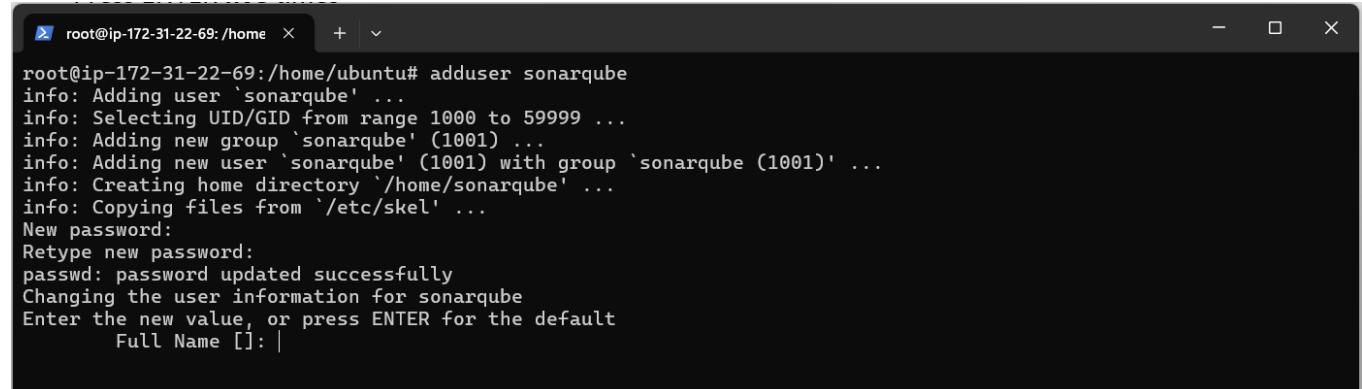
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user 'sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group 'sonarqube' (1001) ...
info: Adding new user 'sonarqube' (1001) with group 'sonarqube (1001)' ...
info: Creating home directory '/home/sonarqube'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

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



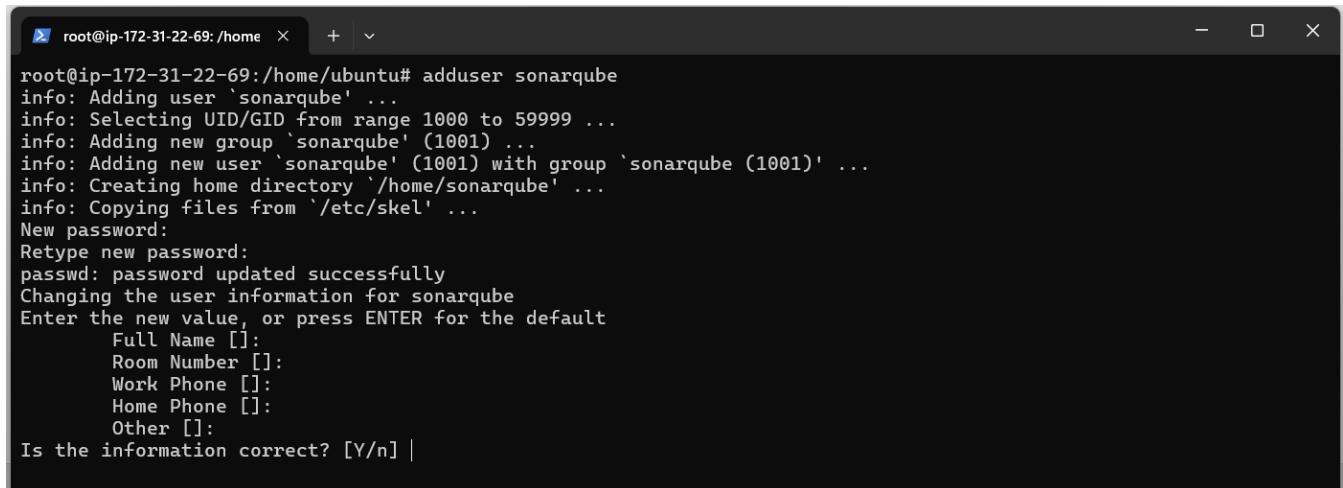
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user 'sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group 'sonarqube' (1001) ...
info: Adding new user 'sonarqube' (1001) with group 'sonarqube (1001)' ...
info: Creating home directory '/home/sonarqube'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

Re-enter the password and press ENTER



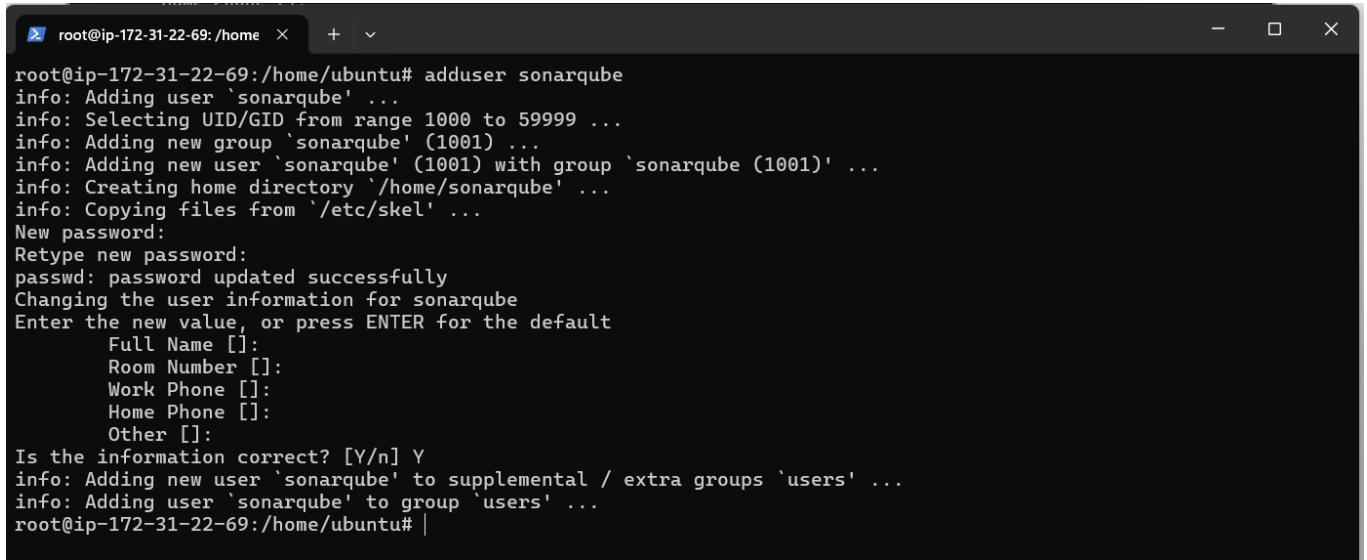
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user 'sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group 'sonarqube' (1001) ...
info: Adding new user 'sonarqube' (1001) with group 'sonarqube (1001)' ...
info: Creating home directory '/home/sonarqube'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



```
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



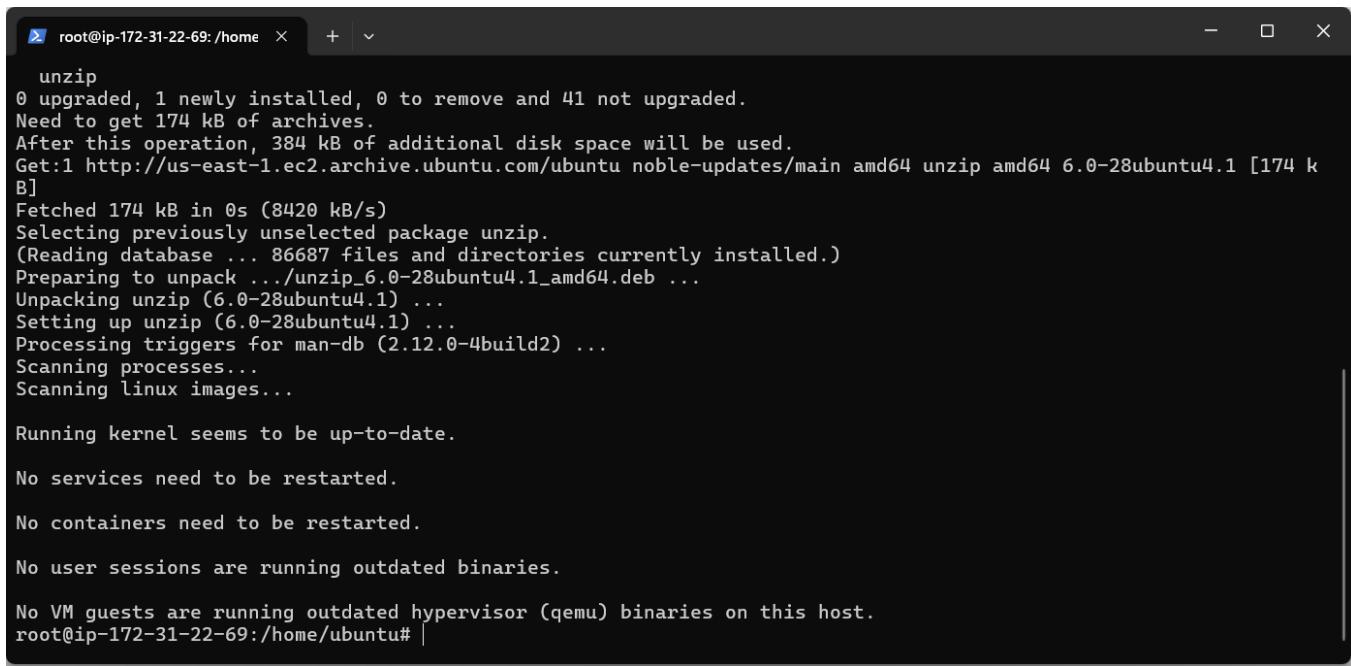
```
root@ip-172-31-22-69:/home/ubuntu# adduser sonarqube
info: Adding user `sonarqube' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `sonarqube' (1001) ...
info: Adding new user `sonarqube' (1001) with group `sonarqube (1001)' ...
info: Creating home directory `/home/sonarqube'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

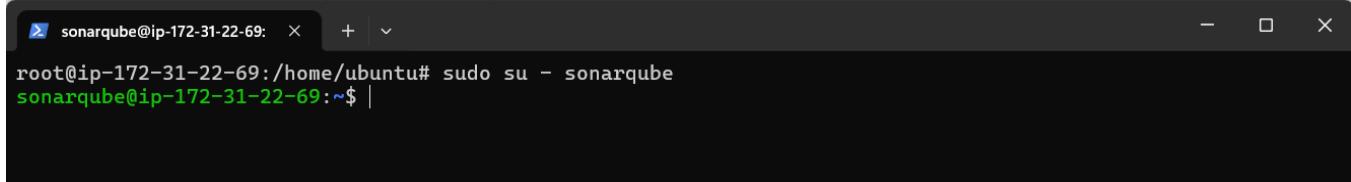
No user sessions are running outdated binaries.

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
```

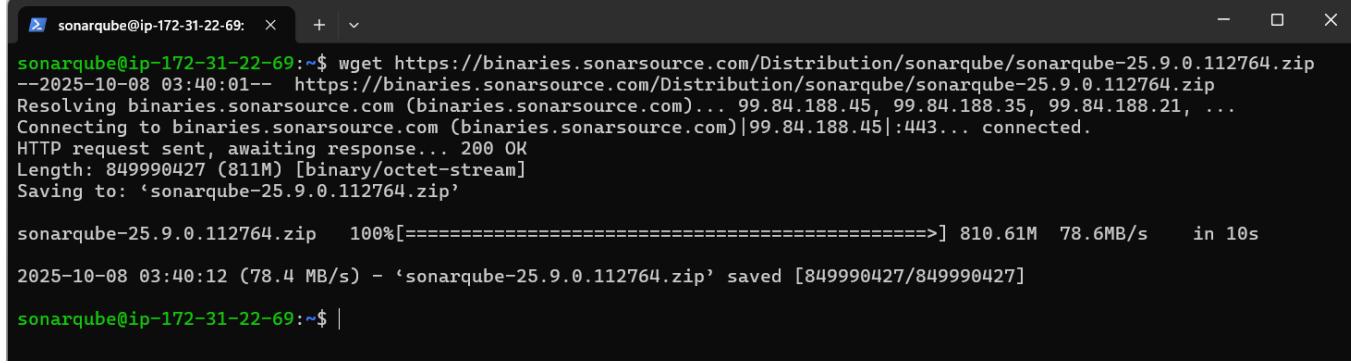


```
sonarqube@ip-172-31-22-69: ~ x + v
root@ip-172-31-22-69:/home/ubuntu# sudo su - sonarqube
sonarqube@ip-172-31-22-69:~$ |
```

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
sonarqube-25.9.0.112764 lib/jdbc/mssql/lang-tag-1.7.jar
sonarqube-25.9.0.112764 lib/jdbc/mssql/nimbus-jose-jwt-10.0.2.jar
sonarqube-25.9.0.112764 lib/jdbc/mssql/accessors-smart-2.5.2.jar
sonarqube-25.9.0.112764 lib/jdbc/mssql/asm-9.7.1.jar
sonarqube-25.9.0.112764 lib/jdbc/postgresql/
sonarqube-25.9.0.112764 lib/jdbc/postgresql/postgresql-42.7.7.jar
sonarqube-25.9.0.112764 lib/jdbc/h2/
sonarqube-25.9.0.112764 lib/jdbc/h2/h2-2.3.232.jar
sonarqube-25.9.0.112764 lib/sonar-shutdowner-25.9.0.112764.jar
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$ |
```

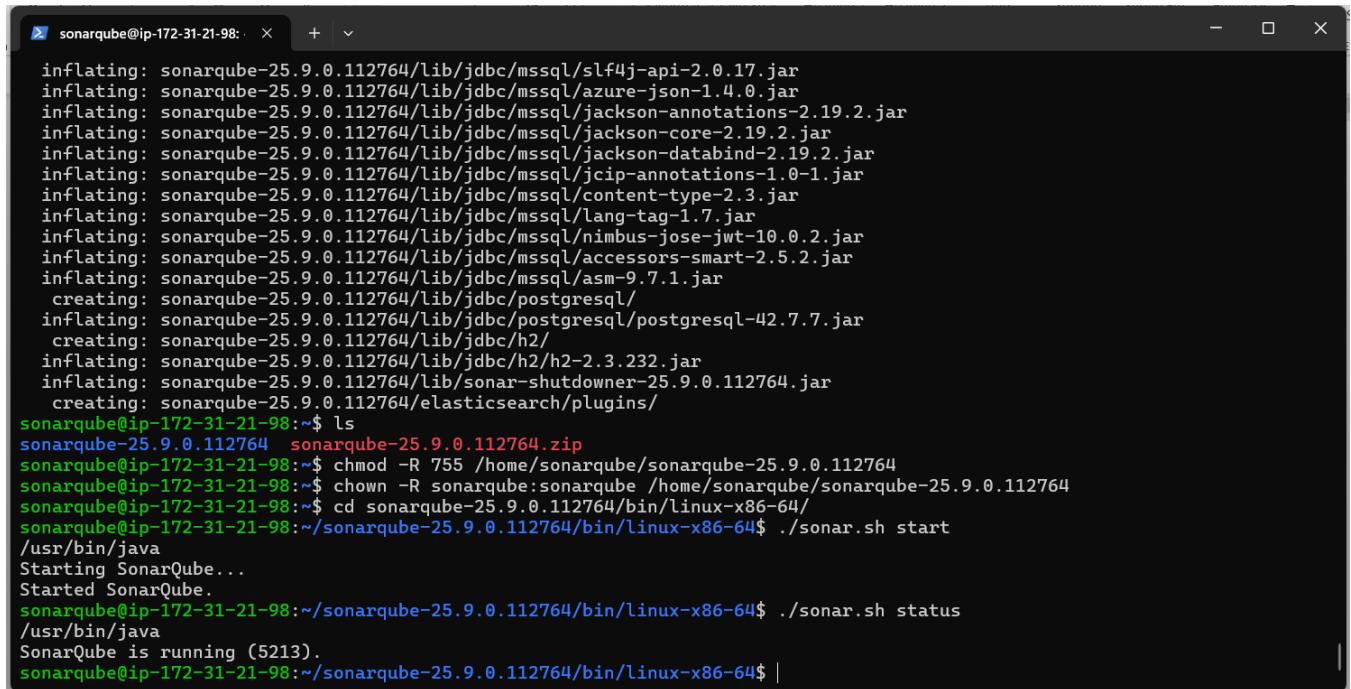
We will start SonarQube by running the command:

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

```
sonarqube@ip-172-31-22-69: ~$ ls
sonarqube-25.9.0.112764 lib/jdbc/postgresql/
sonarqube-25.9.0.112764 lib/jdbc/postgresql/postgresql-42.7.7.jar
sonarqube-25.9.0.112764 lib/jdbc/h2/
sonarqube-25.9.0.112764 lib/jdbc/h2/h2-2.3.232.jar
sonarqube-25.9.0.112764 lib/sonar-shutdowner-25.9.0.112764.jar
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.

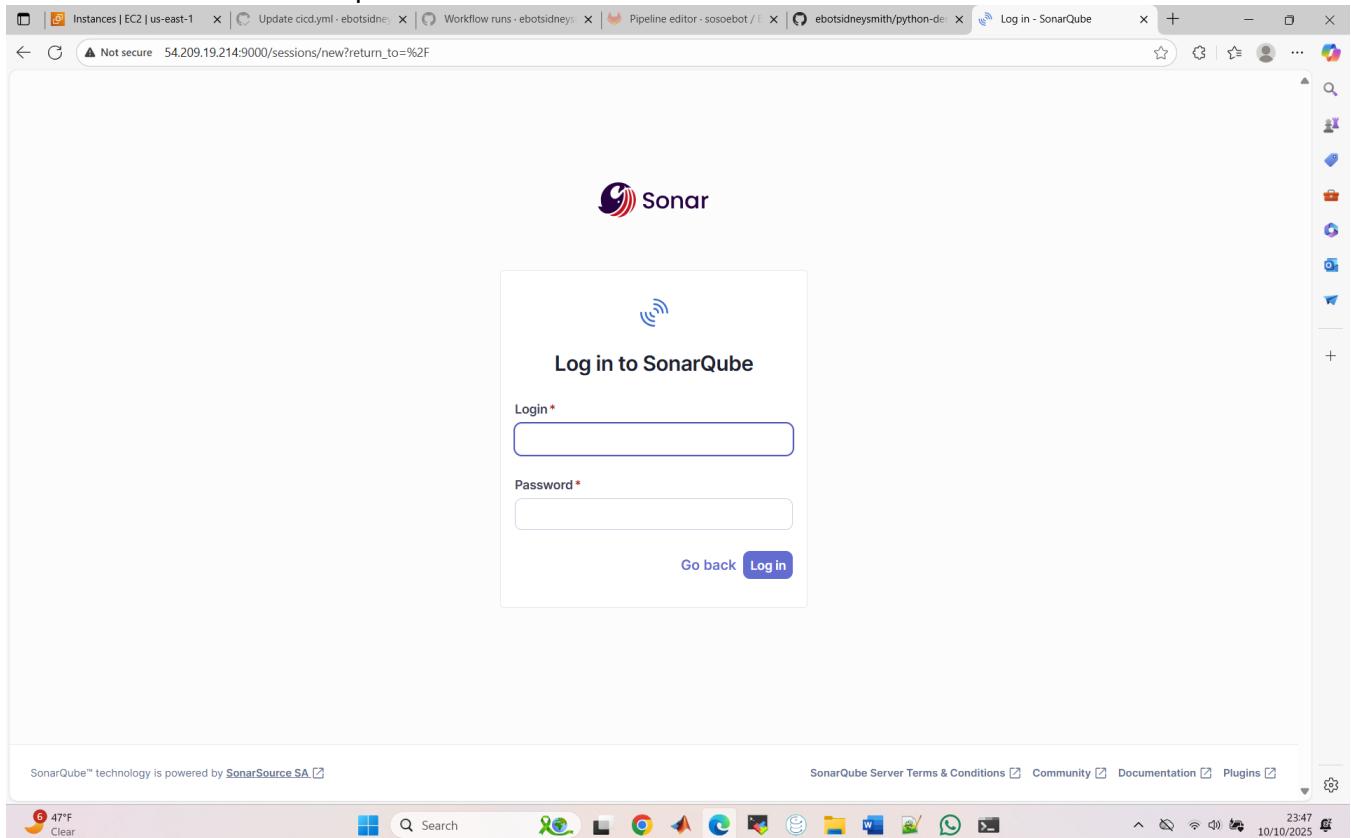
STEP 6: Access SonarQube on Browser

We will now access SonarQube on the browser using

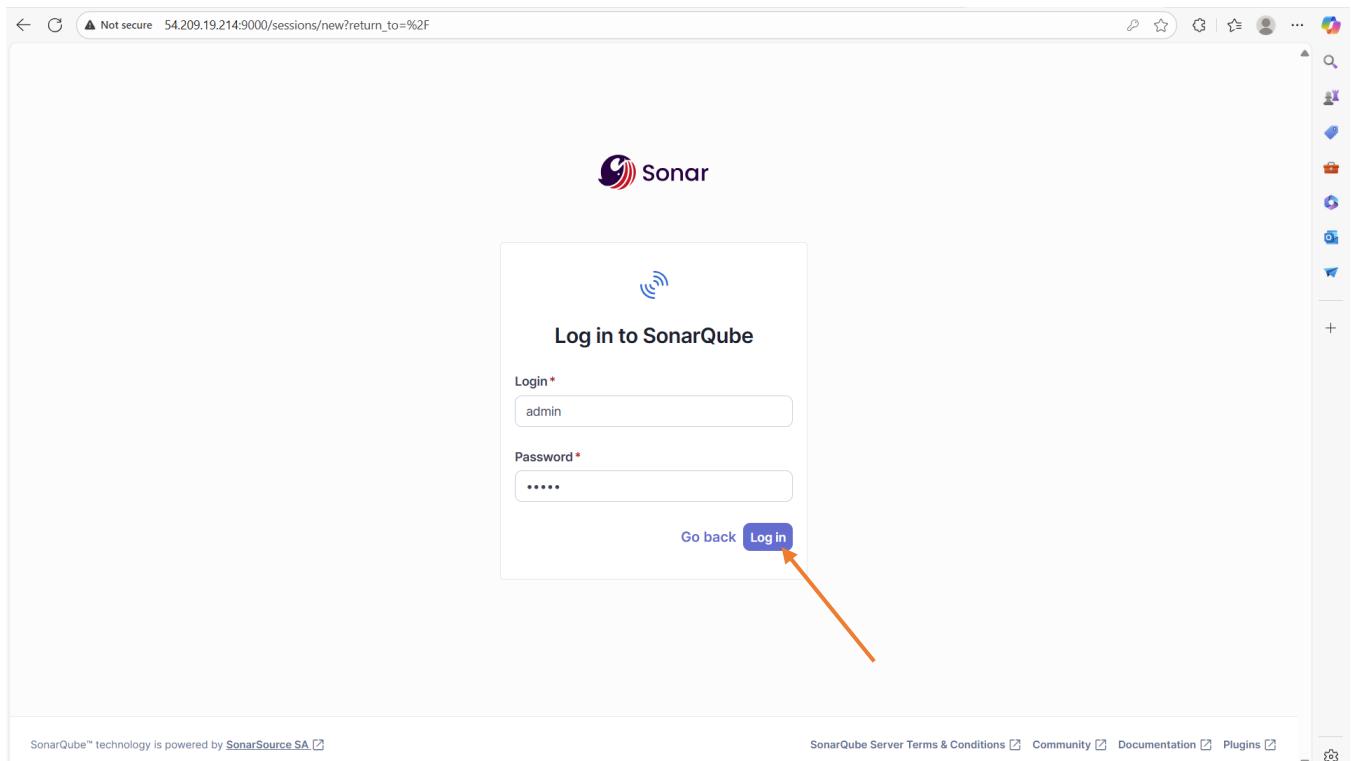
`http://<Public IPv4 address>:9000`

That is `http://54.209.19.214:9000`

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

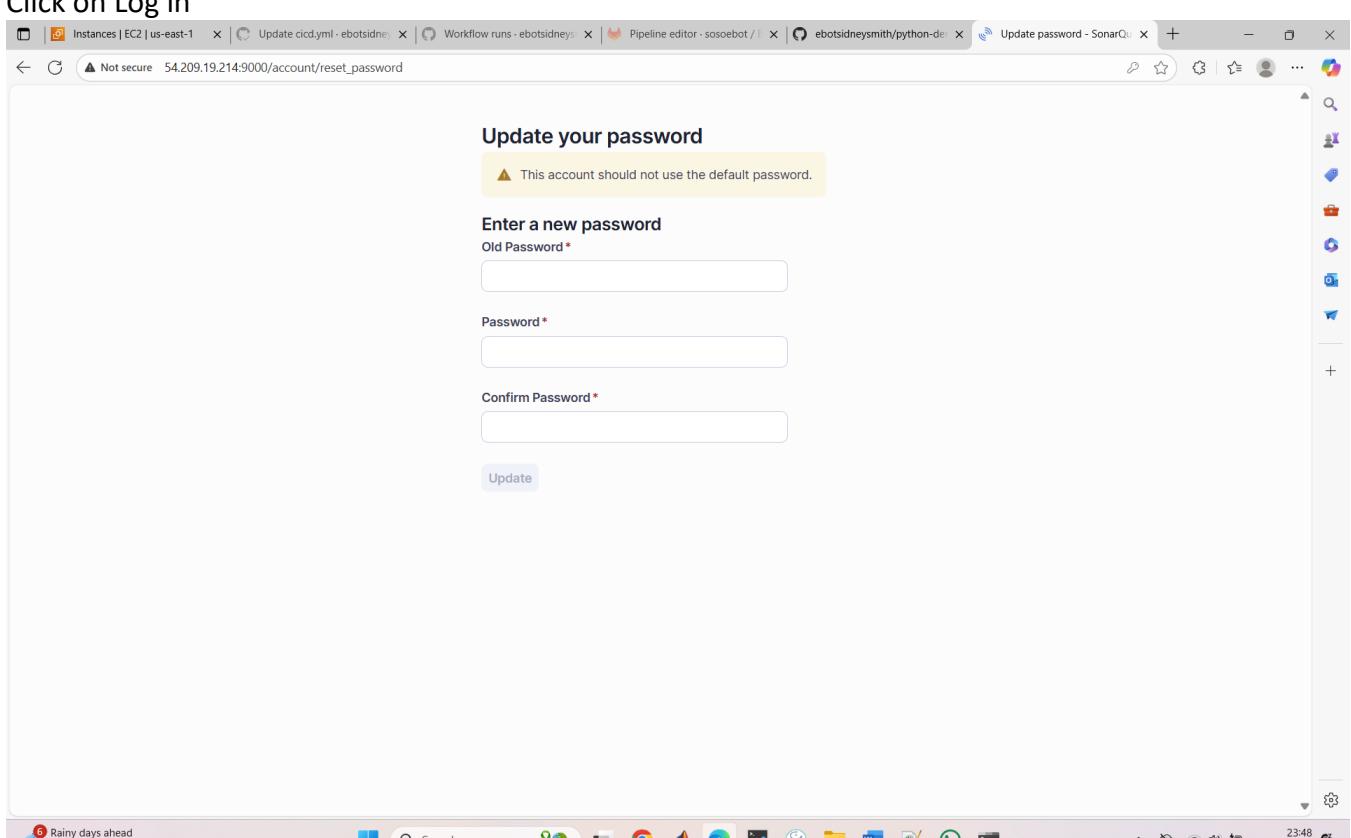


Enter the default username and password “**admin**”



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

Click on Log in



Update your password

⚠ This account should not use the default password.

Enter a new password

Old Password *

Password *

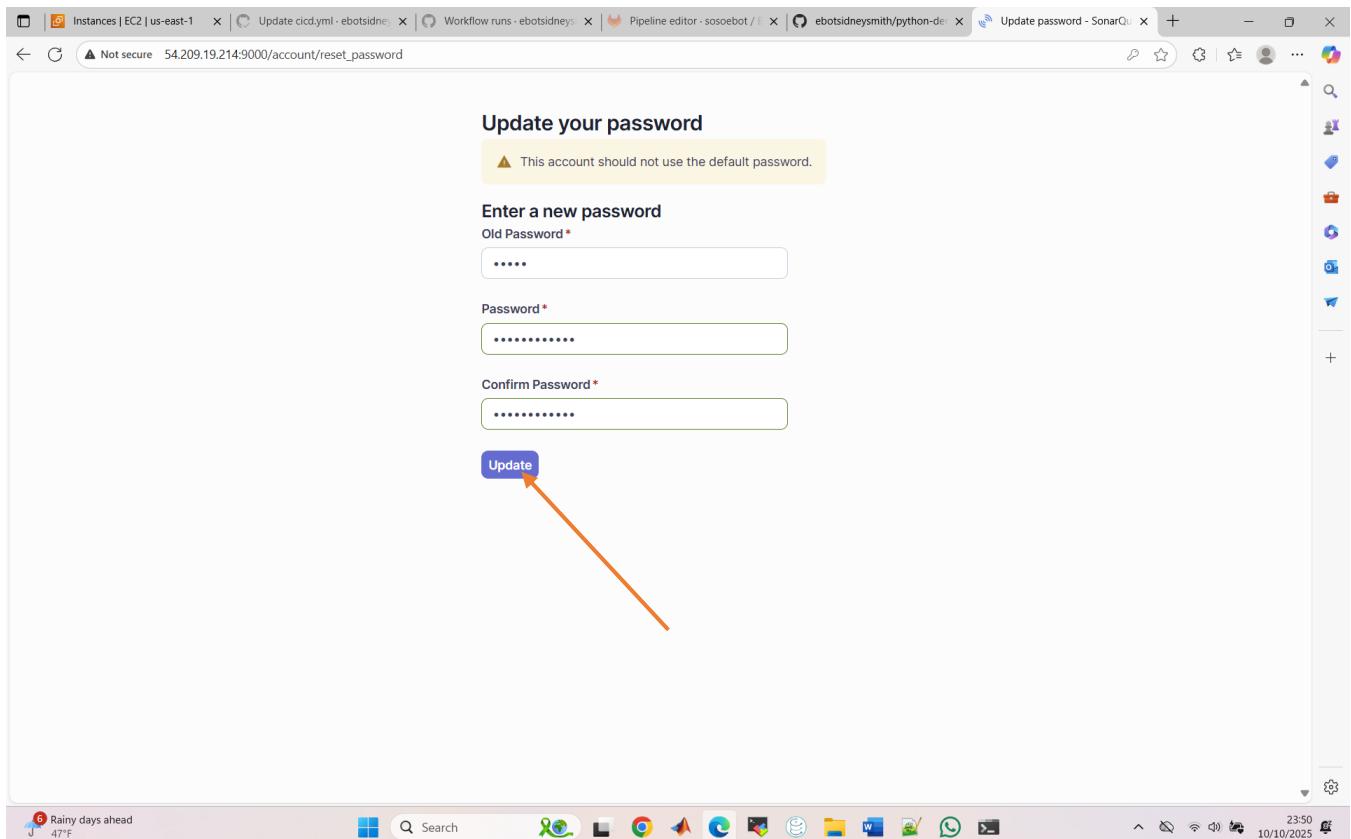
Confirm Password *

Update

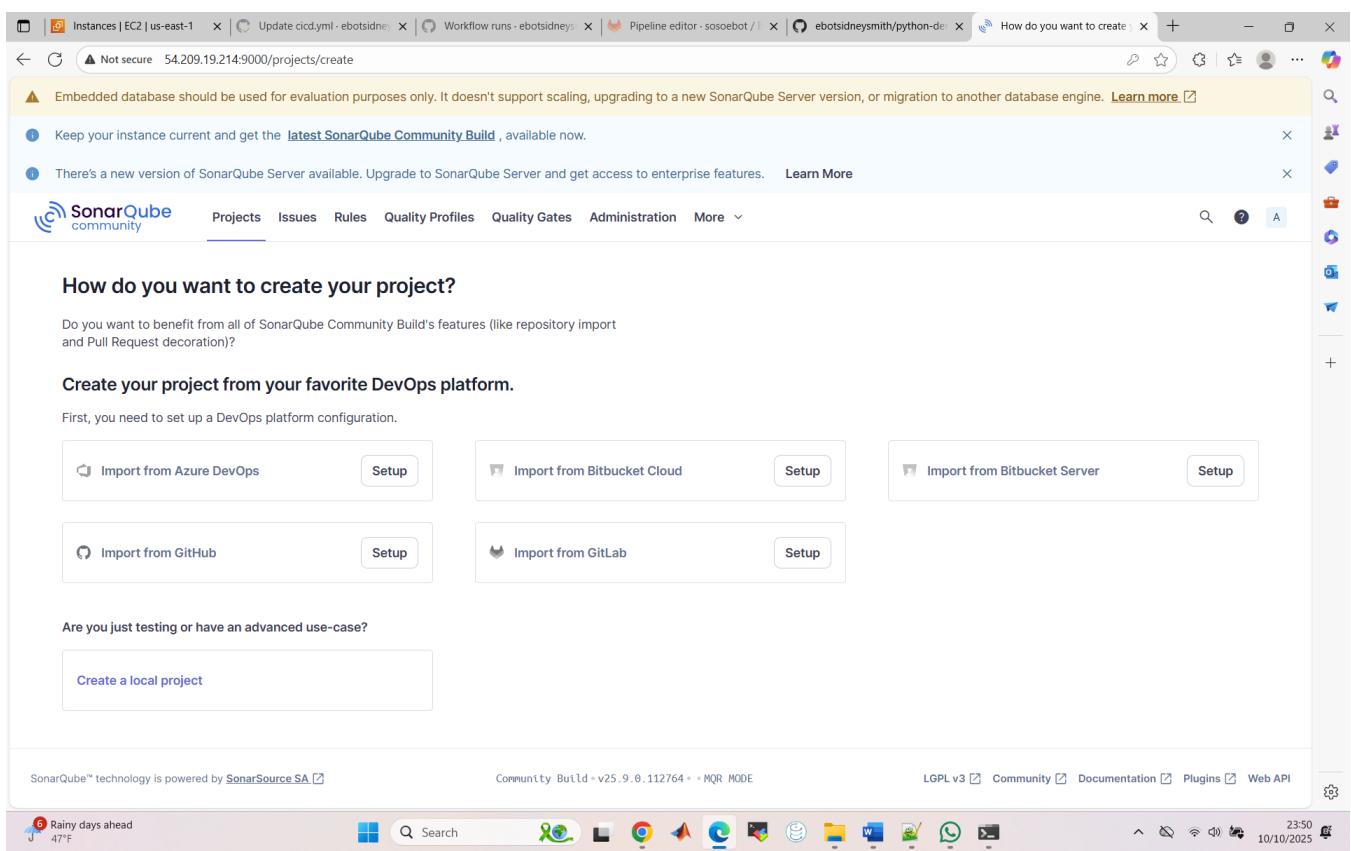
Rainy days ahead 47°F 23:48 10/10/2025

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

The screenshot shows the SonarQube Community interface at <http://54.209.19.214:9000/projects/create>. At the top, there are two yellow warning messages: one about using an embedded database for evaluation purposes and another about upgrading to a new SonarQube Server version. Below the messages, the navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. A sidebar on the right contains icons for user management, repository imports, and other system functions.

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.

Import options:

- Import from Azure DevOps (Setup)
- Import from Bitbucket Cloud (Setup)
- Import from Bitbucket Server (Setup)
- Import from GitHub (Setup)
- Import from GitLab (Setup)

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

Create a local project

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

Click on “Create a local Project”

The screenshot shows the 'Create a local project' form at <http://54.145.61.212:9000/projects/create?mode=manual>. The top banner and sidebar are identical to the previous screenshot. The main form has three fields highlighted with orange arrows: 'Project display name' (a text input field), 'Project key' (a text input field), and 'Main branch name' (a text input field containing 'main'). Below these fields is a note: 'The name of your project's default branch [Learn More](#)'. At the bottom of the form are 'Cancel' and 'Next' buttons. The browser status bar shows the date as 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 interface for setting up a new project. The top navigation bar includes links for index.html, sosoebot.org - details, Instances | EC2 | us-east-1, Key pairs | EC2 | us-east-1, ebotsidneysmith/jenkins, jenkins-pipeline - Jenkins, Create a local project, and other tabs.

A warning message at the top 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 content area is titled "Set up project for Clean as You Code". It explains that the new code definition sets which part of your code will be considered new code. It highlights the "Clean as You Code" methodology and provides a link to "Defining New Code".

The configuration section is titled "Choose the baseline for new code for this project". It offers three options:

- Use the global setting (selected):
 - 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.

At the bottom of the configuration section, there are "Back" and "Create project" buttons. An orange arrow points to the "Create project" button.

The status bar at the bottom right shows the date and time: 06:31 9/12/2025.

Click on "create project"

Prepared by Sidney Smith

The screenshot shows the SonarQube interface for setting up analysis methods. The 'With Jenkins' option is highlighted with a blue border. An orange arrow points from the text 'Our project has been created.' below to the 'With GitLab CI' option. Other options shown include 'With GitHub Actions', 'With Bitbucket Pipelines', 'With Azure Pipelines', and 'Other CI'. A note about 'Locally' is also present.

Analysis Method

Use this page to manage and set-up the way your analyses are performed.

How do you want to analyze your repository?

- With Jenkins
- With GitHub Actions
- With Bitbucket Pipelines
- With GitLab CI
- With Azure Pipelines
- Other CI
- Locally

SonarQube™ technology is powered by [SonarSource SA](#). Community Build v25.9.0.112764 - MQR MODE. LGPL v3. Documentation. Plugins. Web API.

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 'Bind project' page for a project named 'mywebsite'. The 'Analysis Method' dropdown is set to 'Maven'. A red arrow points to the 'Maven' button in the dropdown menu.

1 What option best describes your project?

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

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, ...)

SonarQube™ technology is powered by [SonarSource SA](#) Community Build v25.9.0.112764 • MQR MODE LGPL v3 • Community • Documentation • Plugins • Web API

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 'Bind project' page for a project named 'mywebsite'. The 'Analysis Method' dropdown is set to 'GitLab CI'. A red arrow points to the 'GitLab CI' button in the dropdown menu.

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: [Generate a token](#)
- Uncheck the **Protect Variable** checkbox.
- Check the **Mask Variable** checkbox.

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.

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

General

Integrations

Webhooks

Access tokens

Repository

Merge requests

CI/CD

Packages and registries

Monitor

Usage quotas

SSH HTTPS

Runner token: ebotsmith
Runner description: Sidney Smith Ebot

Project information

Invite your team

Add members to this project and start collaborating with your team.

Code

Code v

Created on

October 07, 2025

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

General pipelines

Customize your pipeline configuration.

Auto DevOps

Automate building, testing, and deploying your applications based on your continuous integration and delivery configuration. [How do I get started?](#)

Runners

Runners are processes that pick up and execute CI/CD jobs for GitLab. [What is GitLab Runner?](#)

Artifacts

A job artifact is an archive of files and directories saved by a job when it finishes.

Variables

Variables store information that you can use in job scripts. Each project can define a maximum of 8000 variables. [Learn more.](#)

Pipeline trigger tokens

Trigger a pipeline for a branch or tag by generating a trigger token and using it with an API call. The token impersonates a user's project access and permissions. [Learn more.](#)

Deploy freezes

Add a freeze period to prevent unintended releases during a period of time for a given environment. You must update the deployment jobs in `.gitlab-ci.yml` according to the deploy freezes added here. [Learn more.](#) Specify deploy freezes using cron syntax.

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 displays "Project runners" (0), "Other available runners" (#50055565 (AKqlnI7D) with a "project-runner" tag), "Group runners" (0), and "Instance runners" (112). An orange arrow points to the "Create project runner" button at the top right 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 has an orange arrow pointing to it. Below it is a checkbox for "Run untagged jobs". The "Configuration (optional)" section includes a "Runner description" input field and several checkboxes for "Paused", "Protected", and "Lock to current projects". At the bottom, there's a "Maximum job timeout" input field with placeholder text "Enter the job timeout in seconds. Must be a minimum of 600 seconds."

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”

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 Linux:

```
# Download the binary for your system
sudo curl -L --output /usr/local/bin/gitlab-runner
```

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
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 runner authentication token `glrt-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco` is highlighted in the UI.

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-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco
```

Step 2
Choose an executor when prompted by the command line. Executors run builds in different environments. Not sure which?

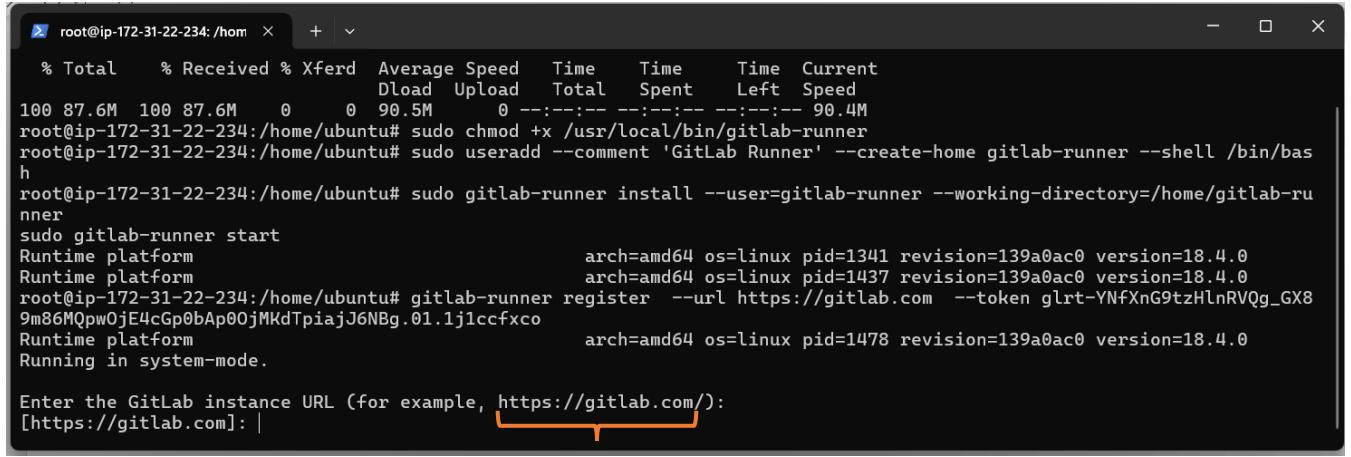
Step 3 (optional)
Manually verify that the runner is available to pick up jobs.

```
$ gitlab-runner run
```

Then follow the steps

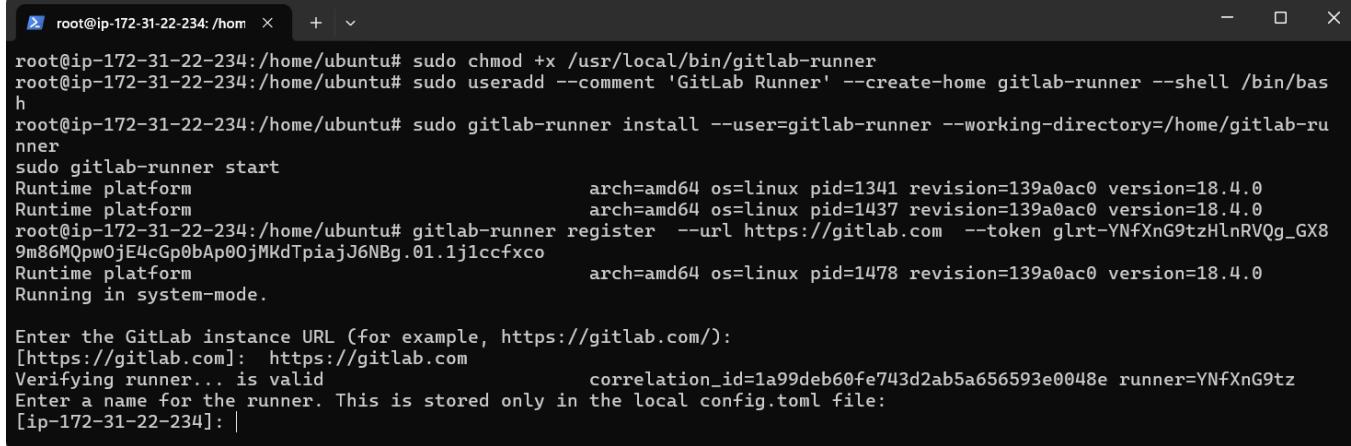
Copy and paste the following command into your command line to register the runner.

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



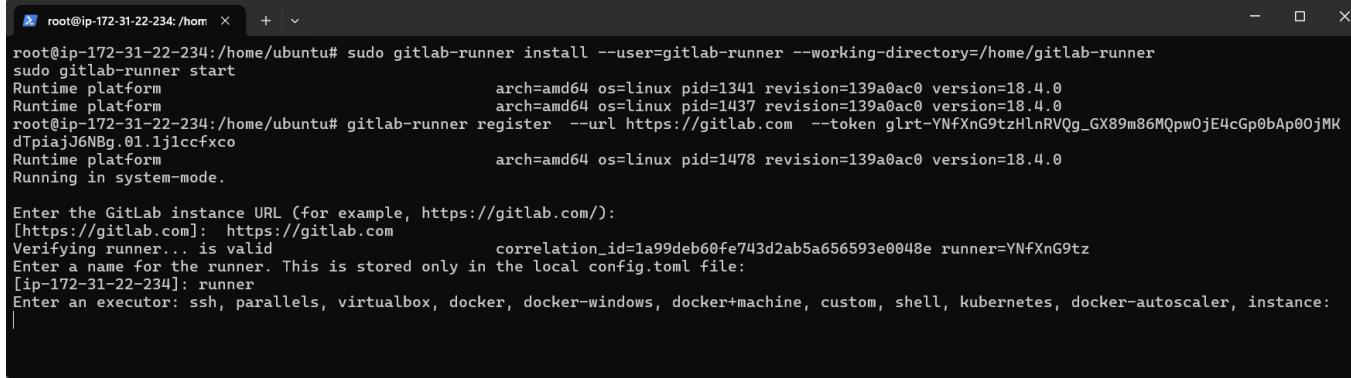
```
% Total % Received % Xferd Average Speed Time Time Time Current  
100 87.6M 100 87.6M 0 0 90.5M 0 --:-- --:-- --:-- 90.4M  
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner  
root@ip-172-31-22-234:/home/ubuntu# sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --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



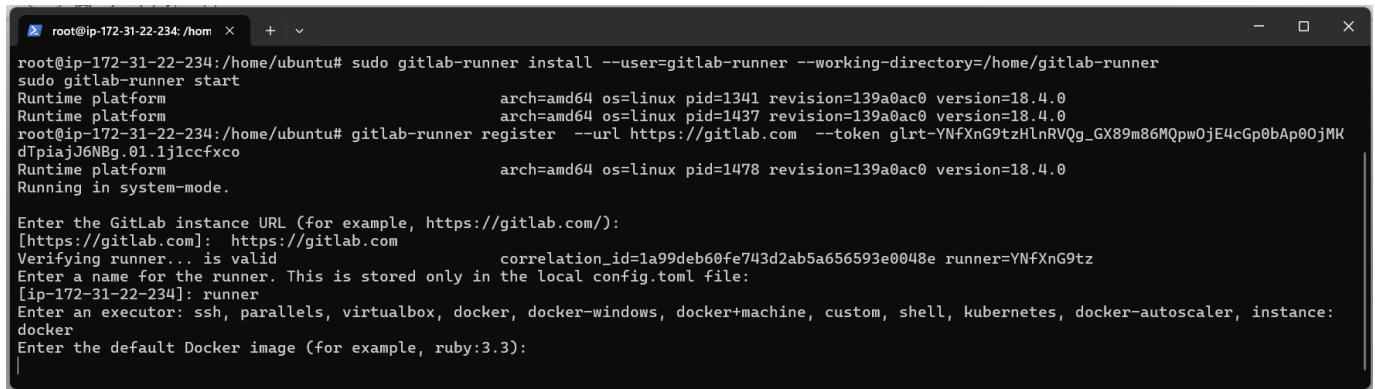
```
root@ip-172-31-22-234:/home/ubuntu# sudo chmod +x /usr/local/bin/gitlab-runner  
root@ip-172-31-22-234:/home/ubuntu# sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --shell /bin/bash  
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner  
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”

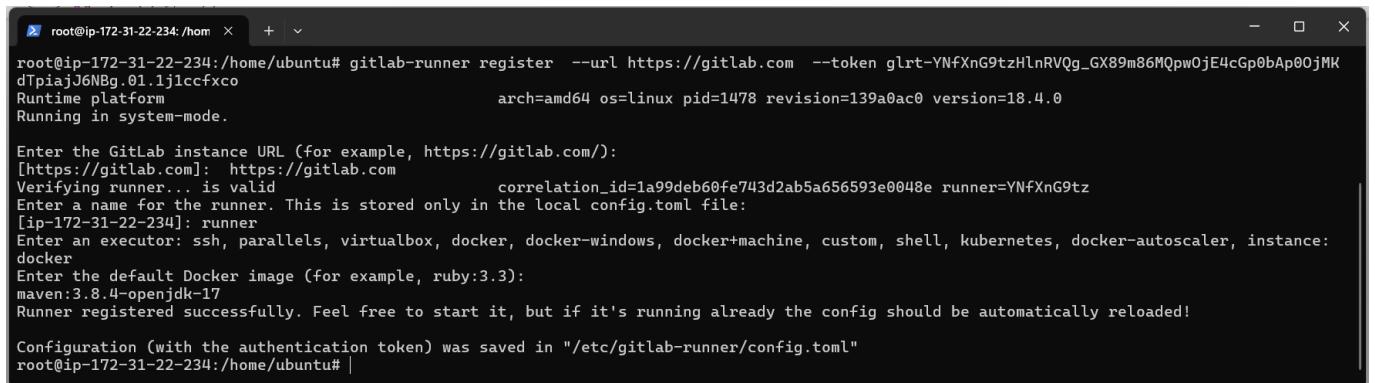


```
root@ip-172-31-22-234:/home/ubuntu# sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner
sudo gitlab-runner start
Runtime platform arch=amd64 os=linux pid=1341 revision=139a0ac0 version=18.4.0
Runtime platform arch=amd64 os=linux pid=1437 revision=139a0ac0 version=18.4.0
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfxnG9tzHlnRVQg_GX89m86MOpw0jE4cGp0bAp00jMK
dTpiaj6NBg.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



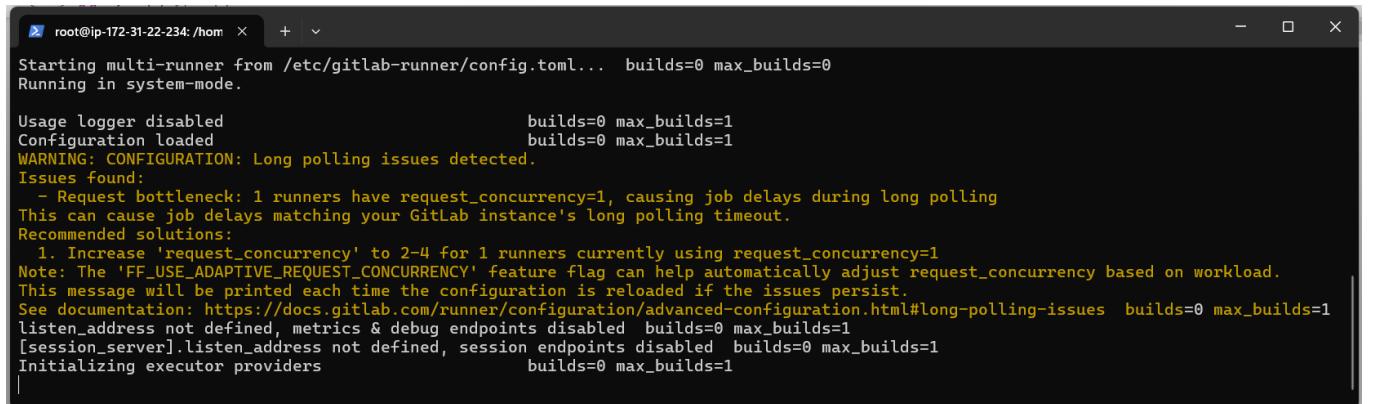
```
root@ip-172-31-22-234:/home/ubuntu# gitlab-runner register --url https://gitlab.com --token glrt-YNfxnG9tzHlnRVQg_GX89m86MOpw0jE4cGp0bAp00jMK
dTpiaj6NBg.01.1j1ccfxco
Runtime platform arch=amd64 os=linux pid=1478 revision=139a0ac0 version=18.4.0
Running in system-mode.

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# gitlab-runner run
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 page has three main sections: Step 1, Step 2, and Step 3 (optional). Step 1 shows a command to register the runner with a specific authentication token. Step 2 allows selecting an executor. Step 3 (optional) shows a success message: 'You've registered a new runner!' with a green signal icon. A red arrow points from the text 'You can see that the runner has been registered. Click on "View Runners"' to the 'View runners' button at the bottom of the page.

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-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco
```

ⓘ The runner authentication token `glrt-YNfXnG9tzHlnRVQg_GX89m86MQpw0jE4cGp0bAp00jMKdTpiajJ6NBg.01.1j1ccfxco` the runner, this token is stored in the `config.toml` and cannot be accessed again from the UI.

Step 2
Choose an executor when prompted by the command line. Executors run builds in different environments. Not sure which one to use? See our executors documentation.

Step 3 (optional)
Manually verify that the runner is available to pick up jobs.

```
$ gitlab-runner run
```

This may not be needed if you manage your runner as a system or user service.

You've registered a new runner!

Your runner is online and ready to run jobs.

To view the runner, go to Project > CI/CD Settings > Runners.

View runners

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, Group runners, and Instance runners. The 'Project runners' section shows one runner registered: '#50078084 (YNfXnG9t)'. A red arrow points from the text 'You can see the green signal which means it is available to run jobs. The Gitlab runner is set up.' to the green signal icon next to the runner's name. The 'Group runners' and 'Instance runners' sections are currently empty.

Runners
Runners are processes that pick up and execute CI/CD jobs for GitLab. What is GitLab Runner?

Project runners 1
These runners are assigned to this project.

Assigned project runners

- #50078084 (YNfXnG9t)
project-runner

Other available runners

- #50055565 (AKqlni7D)
project-runner

Group runners 0
These runners are shared across projects in this group. Group runners can be managed with the Runner API.

This group does not have any group runners yet. To register them, go to the group's Runners page.

Instance runners 112
These runners are available to all groups and projects.
Each CI/CD job runs on a separate, isolated virtual machine.

Turn on instance runners for this project

- #11573930 (KzYhZxBv)
1-blue.shared-gitlab-org.runners-manager.gitlab.com
gitlab-org
- #11573990 (NL4gfoBe)
2-blue.shared-gitlab-org.runners-manager.gitlab.com
gitlab-org

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', each listing available runners and their details.

Click on “Build”

The screenshot shows the same GitLab CI/CD Settings page as before, but the 'Pipeline editor' menu item in the left sidebar is now highlighted with an orange arrow. The main content area remains the same, displaying the CI/CD settings interface.

Select “Pipeline Editor”

The screenshot shows the GitLab Pipeline editor interface. On the left, there is a sidebar with various project management options like Issues, Merge requests, Manage, Plan, Code, Build, Pipelines, Jobs, Pipeline editor (which is selected and highlighted in blue), Pipeline schedules, Artifacts, Secure, Deploy, Operate, Monitor, Analyze, Settings, What's new, and Help. In the center, there is a large purple circular icon with a green checkmark and a white 'D'. Below the icon, the text reads "Configure a pipeline to automate your builds, tests, and deployments". A sub-instruction below says "Create a .gitlab-ci.yml file in your repository to configure and run your first pipeline." At the bottom right of the central area is a blue "Configure pipeline" button.

Click on “Configure Pipeline”

The screenshot shows the GitLab Pipeline editor configuration page. The sidebar on the left is identical to the previous screenshot. The main area has a warning message at the top: "⚠ This GitLab CI configuration is invalid: Reference not found. Learn more". Below this, there are tabs for Edit, Visualize, Validate, and Full configuration. The Full configuration tab is selected and shows a code editor with a sample .gitlab-ci.yml file. The file contains comments explaining the basic structure of a CI/CD pipeline. The code editor has syntax highlighting for YAML. At the bottom of the configuration page, there is a "Commit message" field and a "Update .gitlab-ci.yml file" button.

Delete the existing code

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. In the main area, there's a single stage labeled '1'. At the bottom, a commit message box contains the text 'Update .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 content area. The menu items include 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 pipeline configuration with a failing job named 'test-job'.

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 management sections like Build, Secure, Deploy, Operate, Monitor, Analyze, and Settings. Under the Settings section, 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". An orange arrow points from the text "Click on the Project runner" to the first runner. Below the project runners, there is a section for "Group runners" which is currently empty. At the bottom, there is a section for "Instance runners" with two entries: "#11573930 (KzYhZxBv)" and "#11573990 (NL4gfoBe)". A "Turn on instance runners for this project" toggle switch is present.

Click on the Project runner

The screenshot shows the detailed view of a project runner named "#50078084 (YNfXnG9t)". The top navigation bar shows the URL as https://gitlab.com/sosoebot/boardgame/-/runners/50078084. The left sidebar is identical to the one in the previous screenshot. The main content area is titled "Runner #50078084 (YNfXnG9t)". It shows the runner is "Online" and was created by "Sidney Smith Ebot" 45 minutes ago. The runner has no description, last contact 4 minutes ago, and no configuration. It has a maximum job timeout of "None", token expiry "Never expires", and a tag "project-runner" highlighted with an orange arrow. Below this, there is a "Assigned Projects" section showing "sosoebot / Boardgame" as the assigned project. There is also a "Runners" section showing "1" runner assigned to this one. At the bottom, there is a "Jobs" table with one entry: a failed job for commit "00ad00ae" from 4 minutes ago, which was queued at 00:00:00 and took 00:00:11. The status of the job is "Failed".

You can see the tag = "project-runner"

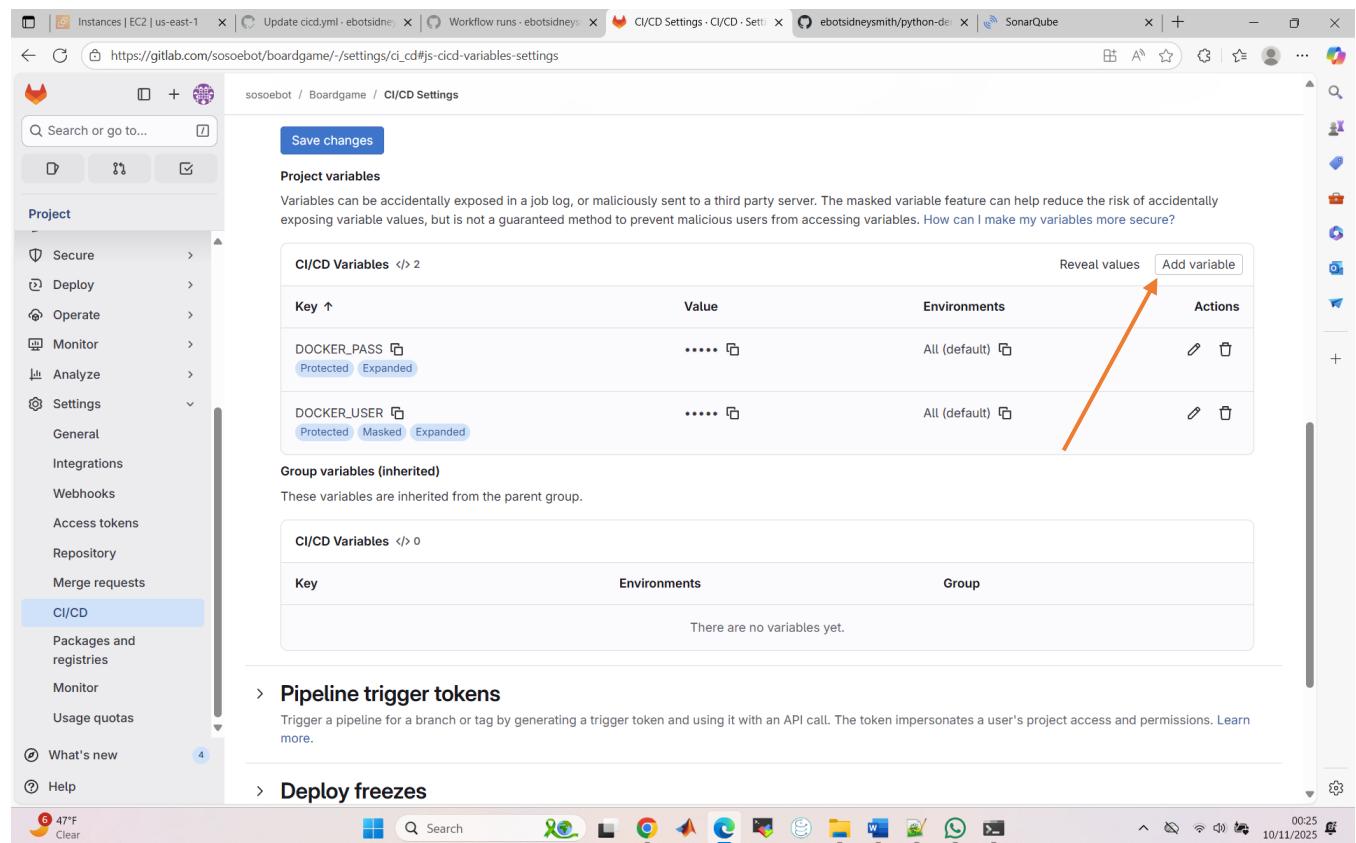
Part 2: Adding “sonarqube-check”

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 various project settings like Secure, Deploy, Operate, Monitor, Analyze, and Settings. Under Settings, the CI/CD tab is selected. The main area is titled "CI/CD Variables" and shows two variables defined: `DOCKER_PASS` (Protected, Expanded) and `DOCKER_USER` (Protected, Masked, Expanded). 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. At the bottom, there are sections for "Pipeline trigger tokens" and "Deploy freezes".

Click on “Add Variable”

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Variables 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 masked (indicated by four dots). The "Visibility" section on the right shows the "Masked" option selected. An orange arrow points from the "Masked" label to the "Value" field for "DOCKER_PASS".

The screenshot shows the same GitLab CI/CD Variables Settings page, but the "Protect variable" checkbox has been unchecked. The "Visibility" section now shows the "Visible" option selected. An orange arrow points from the "Visible" label to the "Value" field for "DOCKER_PASS".

For "Key", enter "SONAR_TOKEN"

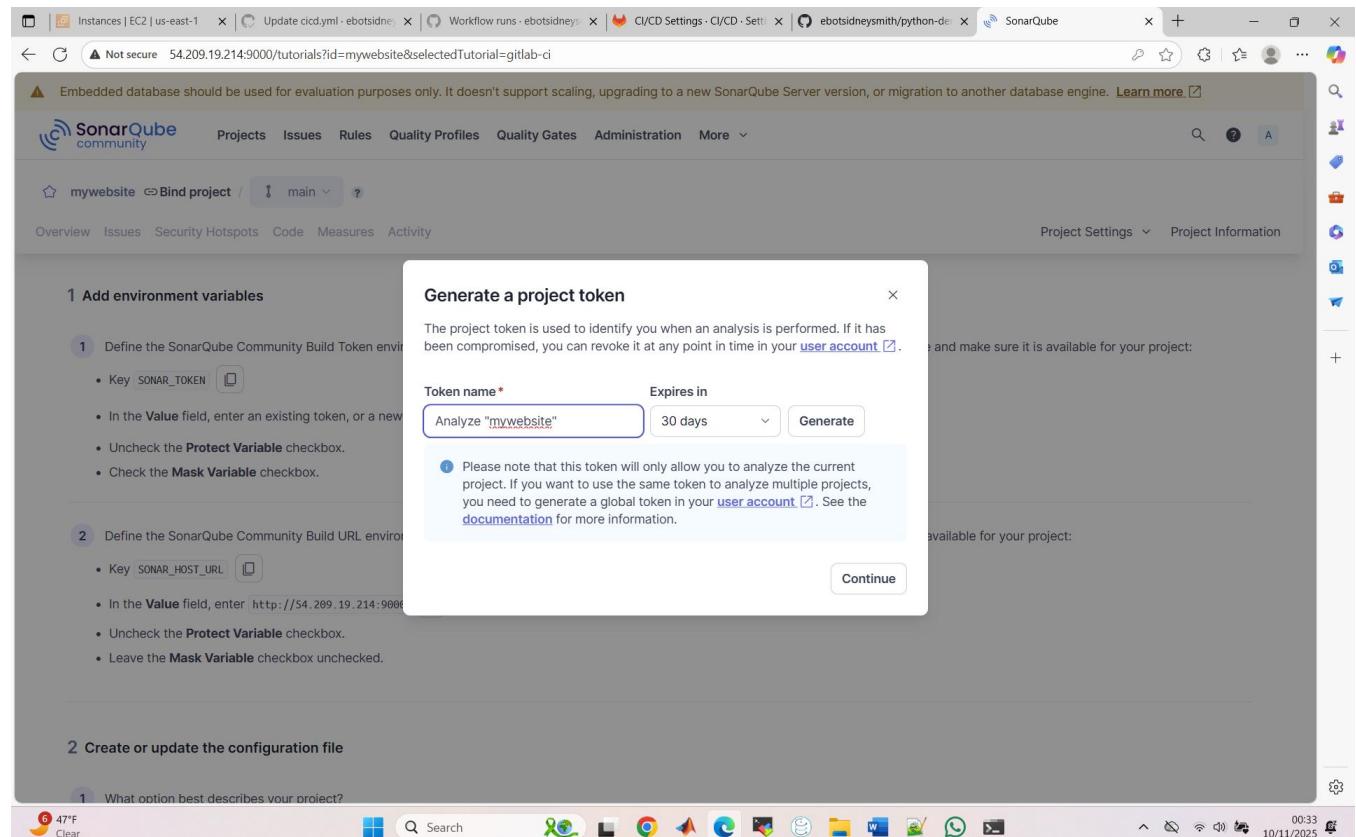
For the “Value”, go to the steps in the Sonarqube browser

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 GitLab and SonarQube. The SonarQube page displays the 'Generate a project token' dialog. The dialog has fields for 'Token name*' (set to 'Analyze "mywebsite"') and 'Expires in' (set to '30 days'). A 'Generate' button is visible. Below the form, there 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.' A 'Continue' button is at the bottom right of the dialog. The background shows the SonarQube project settings page with sections for environment variables and configuration files.

Click on “Generate”

Prepared by Sidney Smith

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.

Generate a project token

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

Analyze "mywebsite":

sqp_c766aa7cf13d5bd082c55a709a209faf68bbca6b

A New token "sqp_c766aa7cf13d5bd082c55a709a209faf68bbca6b" has been created. Make sure you copy it now, you won't be able to see it again!

Continue

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

1 What option best describes your project?

1 Allow merge request pipelines to access protected variables and runners

Save changes

Display manually-defined pipeline variables

Display all manually-defined variables in the pipeline details page after running a pipeline manually. Learn more.

Display pipeline variables

All manually-defined CI/CD variables and their values are visible to maintainers, which is a security risk if including this feature if variables could contain sensitive data. Developers can only view manually-defined variables in the pipeline details page.

Save changes

Project variables

Variables can be accidentally exposed in a job log, or maliciously sent to a third party server. The masked variable flag allows you to hide variable values from job logs without exposing variable values, but is not a guaranteed method to prevent malicious users from accessing variables. How to use CI/CD variables

CI/CD Variables </> 2

Key ↑	Value	Envir
DOCKER_PASS <input type="button" value="Protected"/> <input type="button" value="Expanded"/> <input type="button" value="Copy"/>	All (d)
DOCKER_USER <input type="button" value="Protected"/> <input type="button" value="Masked"/> <input type="button" value="Expanded"/> <input type="button" value="Copy"/>	All (d)

Group variables (inherited)

These variables are inherited from the parent group.

CI/CD Variables </> 0

Key	Environments
SONAR_TOKEN	sqp_c766aa7cf13d5bd082c55a709a209faf68bbca6b

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, 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' with three entries:

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

Each variable has a 'Protected' and 'Expanded' button. To the right of the table, there are sections for 'Visibility' (Visible, Masked, Masked and hidden), 'Flags' (Protect variable checked, Expand variable reference checked), 'Description (optional)', 'Key' (input field), and 'Value' (input field). Buttons for 'Add variable' and 'Cancel' are at the bottom right.

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 settings like Secure, Deploy, Operate, Monitor, Analyze, Settings, and CI/CD. Under CI/CD, 'Variables' is selected. The main area shows three variables:

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

Below the table, it says "Group variables (inherited)". To the right, there's a modal for adding a new variable:

- Key:** (highlighted with an orange arrow)
- Description (optional):**
- Flags:**
 - Protect variable (unchecked)
 - Expand variable reference (\$ will be treated as the start of a reference to another variable.)
- Value:**
- Buttons:** Add variable, Cancel

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 (highlighted with an orange arrow)
 - In the **Value** field, enter (highlighted with an orange arrow)
 - 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 interface for managing CI/CD variables. On the left, there's a sidebar with project settings like Secure, Deploy, Operate, Monitor, Analyze, and Settings. Under Settings, the 'CI/CD' section is selected. The main content area shows a table of CI/CD 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 modal window is open for adding a new variable:

- Key:** SONAR_HOST_URL
- Description (optional):** The description of the variable's value or usage.
- Flags:** Protect variable (unchecked), Expand variable reference (\$ will be treated as the start of a reference to another variable) (checked).
- Value:** (Empty field)

An orange arrow points from the "Value" field in the SonarQube modal to the "Value" field in the GitLab modal.

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

Prepared by Sidney Smith

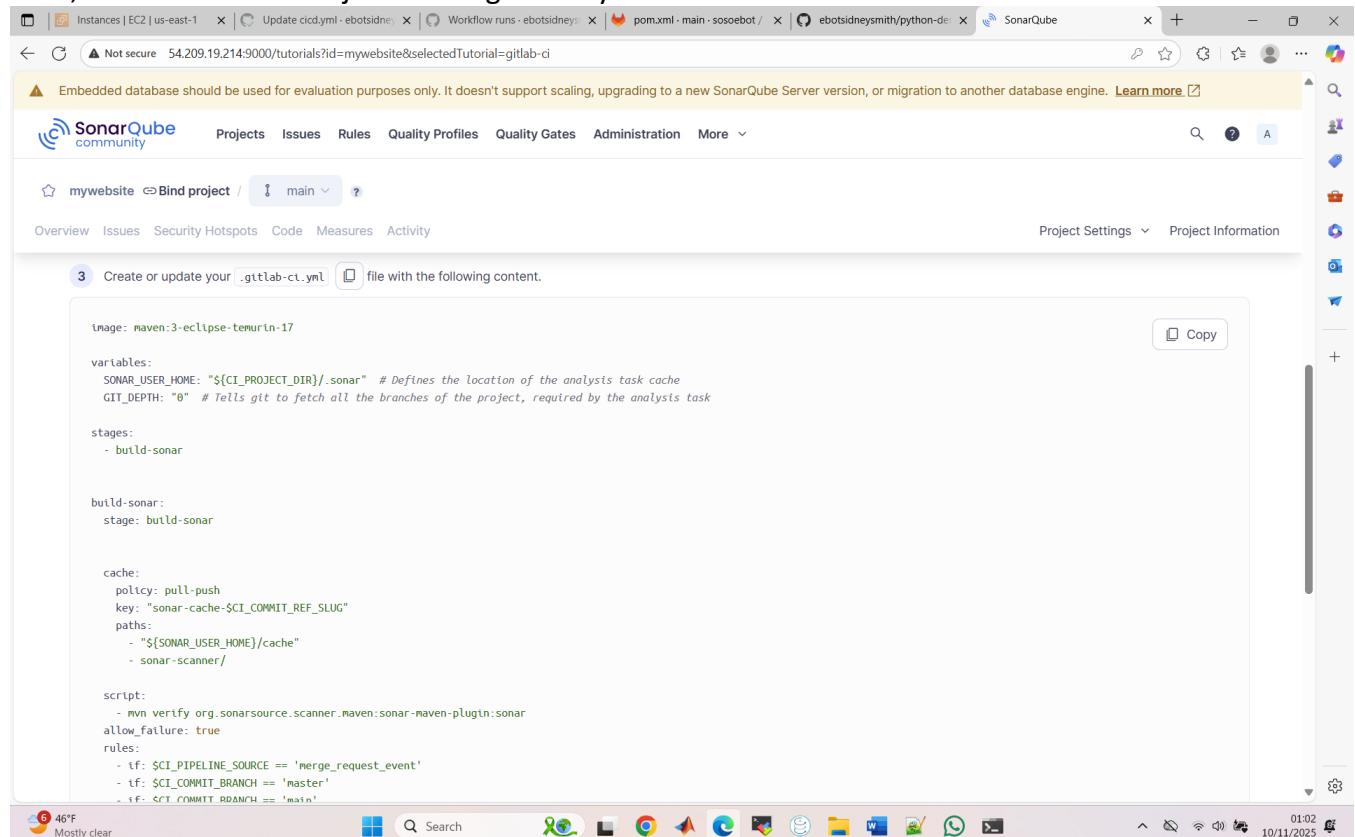
The screenshot shows the GitLab CI/CD Variables Settings page for a project named 'sosoebot / Boardgame'. The 'CI/CD' tab is selected in the sidebar. A modal dialog box is open, prompting for a new variable. The 'Key' field contains 'SONAR_HOST_URL', and the 'Value' field contains 'http://54.209.19.214:9000'. The 'Visibility' section is set to 'Masked'. Other variables listed include DOCKER_PASS, DOCKER_USER, and SONAR_TOKEN.

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 'SONAR_HOST_URL' entry. The 'Value' column for this variable is masked with '.....'. The 'Actions' column for each variable includes edit and delete icons.

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

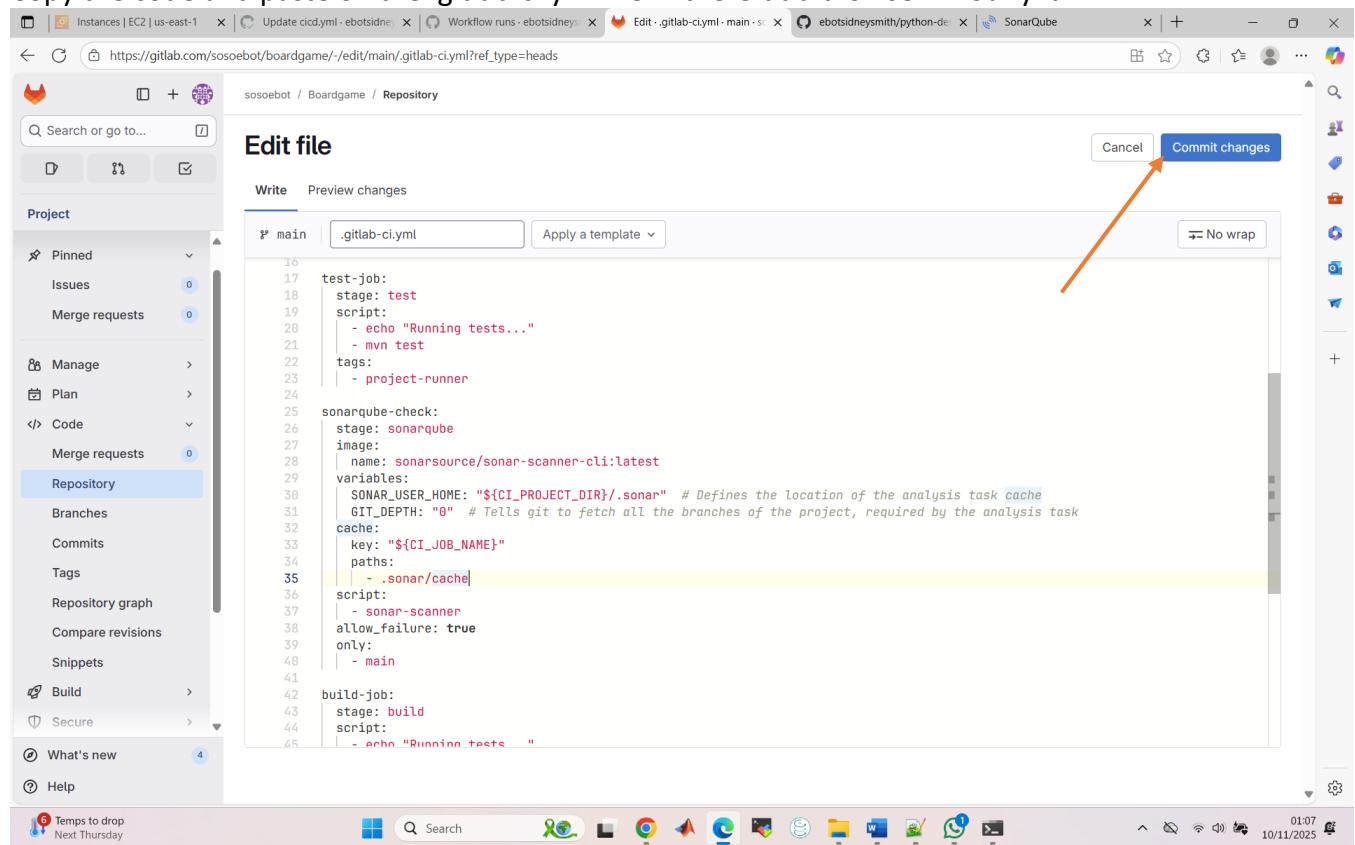
build-sonar:
stage: build-sonar

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 'Commit changes' dialog box overlaid on a GitLab repository page. The dialog has two tabs: 'Commit message' (which is currently active, showing 'Edit .gitlab-ci.yml') and 'Branch' (with options 'Commit to the current main branch' and 'Commit to a new branch'). The background shows the repository structure and code editor for the .gitlab-ci.yml file.

Click on “Commit changes” again

The screenshot shows a confirmation message: 'Your changes have been committed successfully.' Below this, the .gitlab-ci.yml file is displayed with its contents:

```
main > boardgame / .gitlab-ci.yml

.gitlab-ci.yml
Sidney Smith Ebot authored just now
54d04048 Find file Blame History

This GitLab CI configuration is valid. Learn more

.gitlab-ci.yml 1.21 KiB
1 # Use Maven image with OpenJDK 17
2 image: maven:3.8.4-openjdk-17
3
4
5 stages:
6 - test
7 - sonarqube
8 - build
9 - containerize
10
11 variables:
12 IMAGE_NAME: ebotsidneysmith/demoproj
13 IMAGE_TAG: boardgame-0.0.1
14
15
16 test-job:
17 stage: test
18 script:
19   - echo "Running tests..."
20   - mvn test
21 tags:
22   - 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 displays four 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)" and "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. Buttons for 'Save changes', 'Delete variable', and 'Cancel' are at the bottom right.

Adding the docker password

Prepared by Sidney Smith

The screenshot shows the GitLab CI/CD Variables Settings page for a project named '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.

Flags

- 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.

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 GitLab 'CI/CD Settings' page for the 'Boardgame' project. The left sidebar has 'Build' selected under 'Pipelines'. The main content area is titled 'Variables' and contains sections for 'Minimum role to use pipeline variables' (set to 'No one allowed'), 'Access protected resources in merge request pipelines' (with a checked checkbox for 'Allow merge request pipelines to access protected variables and runners'), and 'Project variables' (with a note about masking variable values). A blue arrow points from the 'Build' link in the sidebar to the 'Pipelines' tab in the main content area.

Select “Pipeline”

The screenshot shows the GitLab 'Pipelines' page for the 'Boardgame' project. The left sidebar has 'Pipelines' selected under 'Build'. The main content area lists a single pipeline: 'Edit sonar-project.properties' (Status: Passed, Pipeline ID: #18, Duration: 00:04:07, Created: 12 minutes ago). The pipeline has four stages, all of which are green and marked with a checkmark. A large orange arrow points from the 'Pipelines' link in the sidebar to the stages of the listed pipeline.

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.

You can't push or pull repositories using SSH until you add an SSH key to your profile.
[Add SSH key](#) [Don't show again](#)

Boardgame

main boardgame

Project information

- o 18 Commits
- 2 Branches
- 0 Tags
- 43 MiB Project Storage
- README
- CI/CD configuration
- + Add LICENSE
- + Add CHANGELOG
- + Add CONTRIBUTING
- + Add Kubernetes cluster
- + Add Wiki
- + Configure Integrations

Created on
October 07, 2025

File List:

Name	Last commit	Last update
.github/workflows	This is my first commit	4 days ago
.mvn(wrapper	This is my first commit	4 days ago
src	This is my first commit	4 days ago
.gitignore	This is my first commit	4 days ago
.gitlab-ci.yml	Update .gitlab-ci.yml file	27 minutes ago
Dockerfile	This is my first commit	4 days ago
Jenkinsfile	This is my first commit	4 days ago
Jenkinsfile123	This is my first commit	4 days ago
README.md	This is my first commit	4 days ago
deployment-service.yaml	This is my first commit	4 days ago

We will add a text file. Click on “+”

You can't push or pull repositories using SSH until you add an SSH key to your profile.
[Add SSH key](#) [Don't show again](#)

Boardgame

main boardgame

Project information

- o 18 Commits
- 2 Branches
- 0 Tags
- 43 MiB Project Storage
- README
- CI/CD configuration
- + Add LICENSE
- + Add CHANGELOG
- + Add CONTRIBUTING
- + Add Kubernetes cluster
- + Add Wiki
- + Configure Integrations

Created on
October 07, 2025

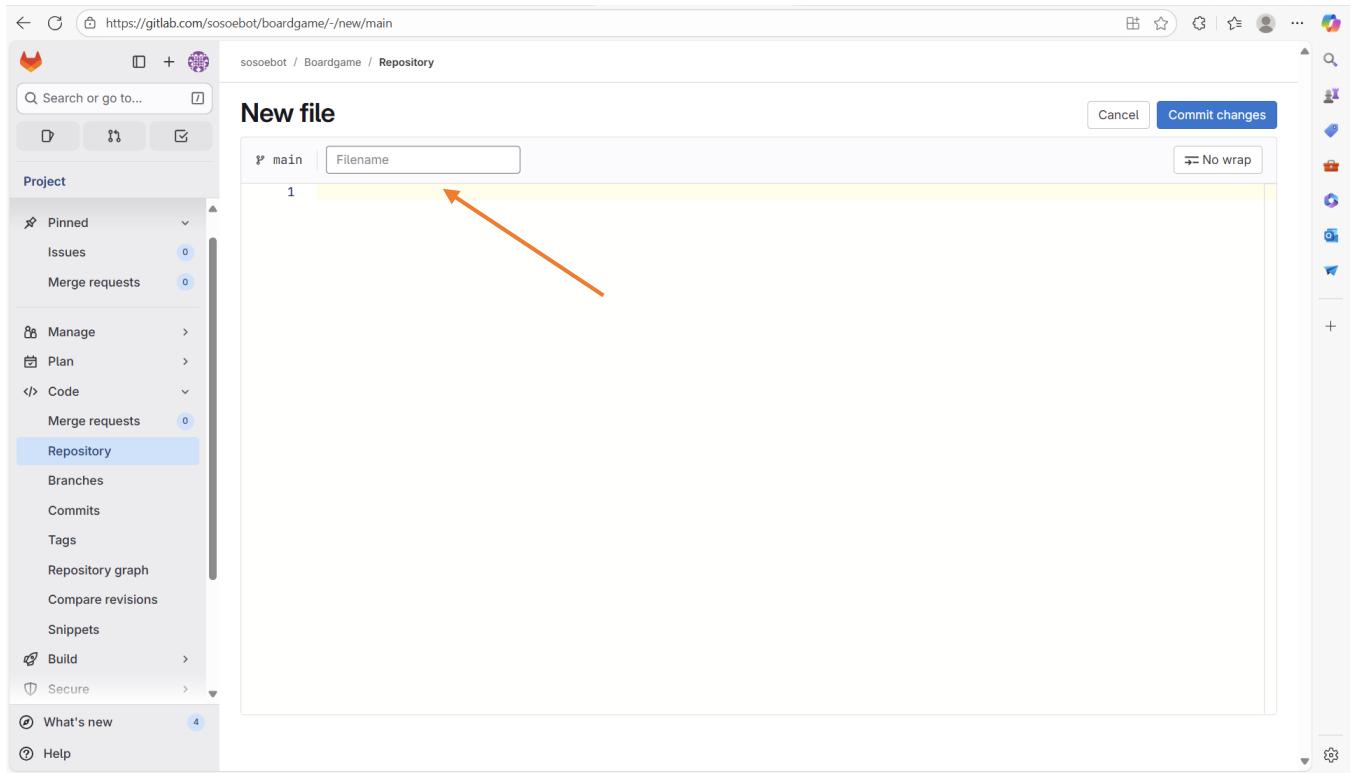
File List:

Name	Last commit	Last update
.github/workflows	This is my first commit	4 days ago
.mvn(wrapper	This is my first commit	4 days ago
src	This is my first commit	4 days ago
.gitignore	This is my first commit	4 days ago
.gitlab-ci.yml	Update .gitlab-ci.yml file	27 minutes ago
Dockerfile	This is my first commit	4 days ago
Jenkinsfile	This is my first commit	4 days ago
Jenkinsfile123	This is my first commit	4 days ago
README.md	This is my first commit	4 days ago
deployment-service.yaml	This is my first commit	4 days ago

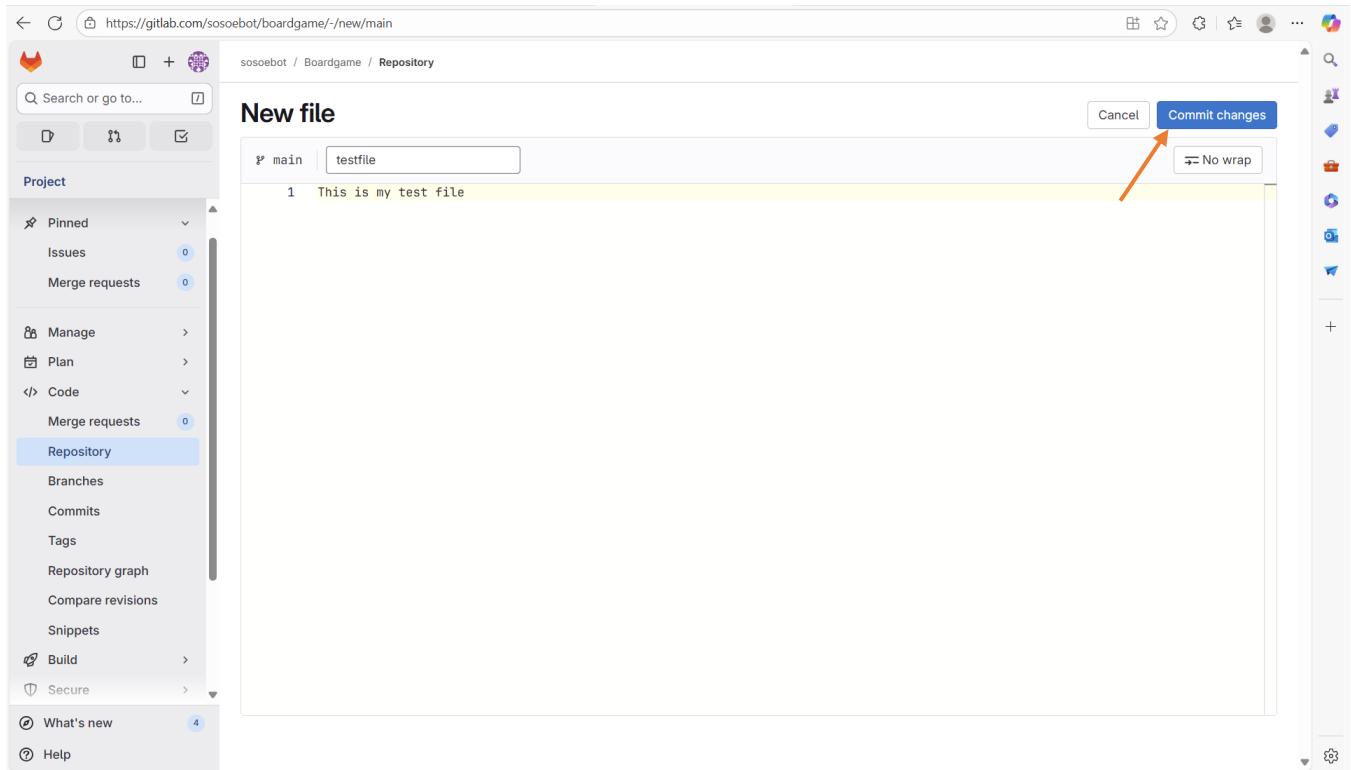
New File Context Menu:

- This directory
- New file
- Upload file
- New directory
- This repository
- New branch
- New tag

Select “New File”

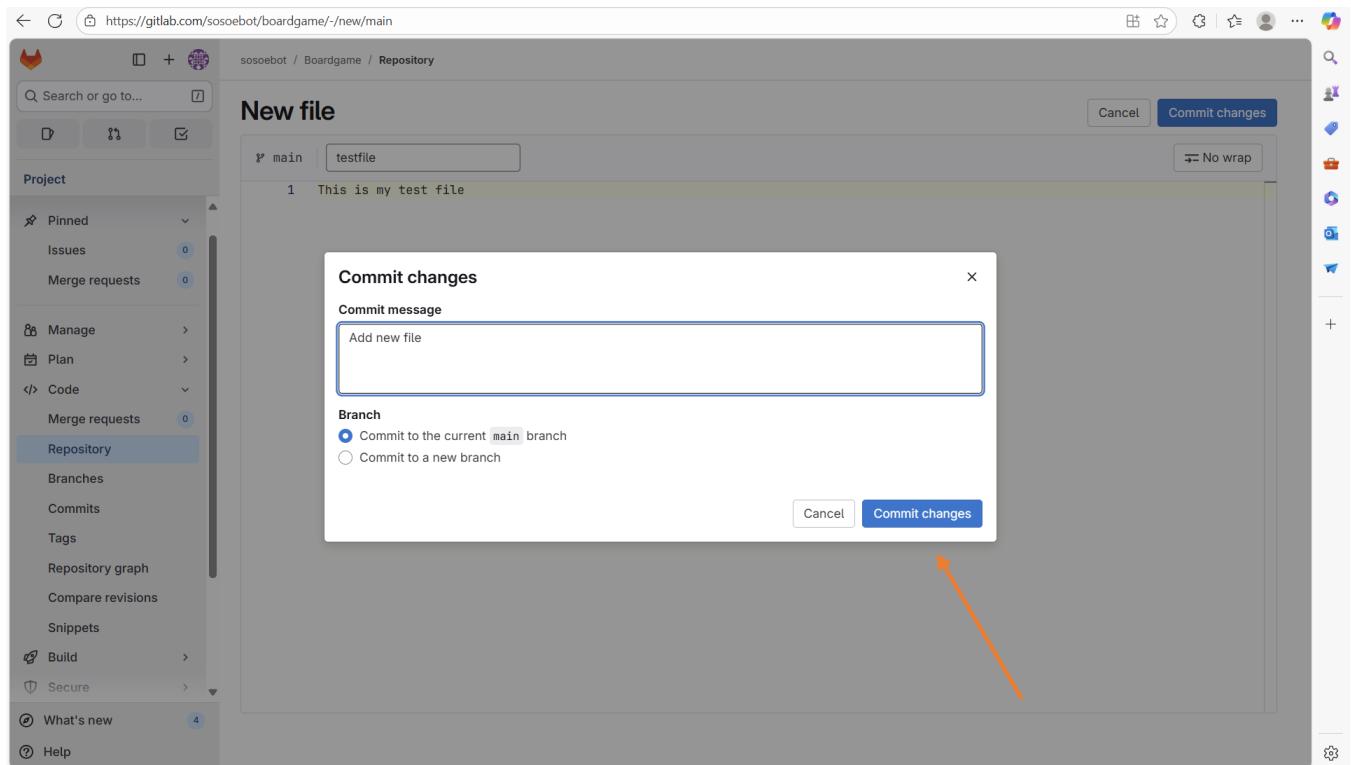


Give the file a name and add some text on the file

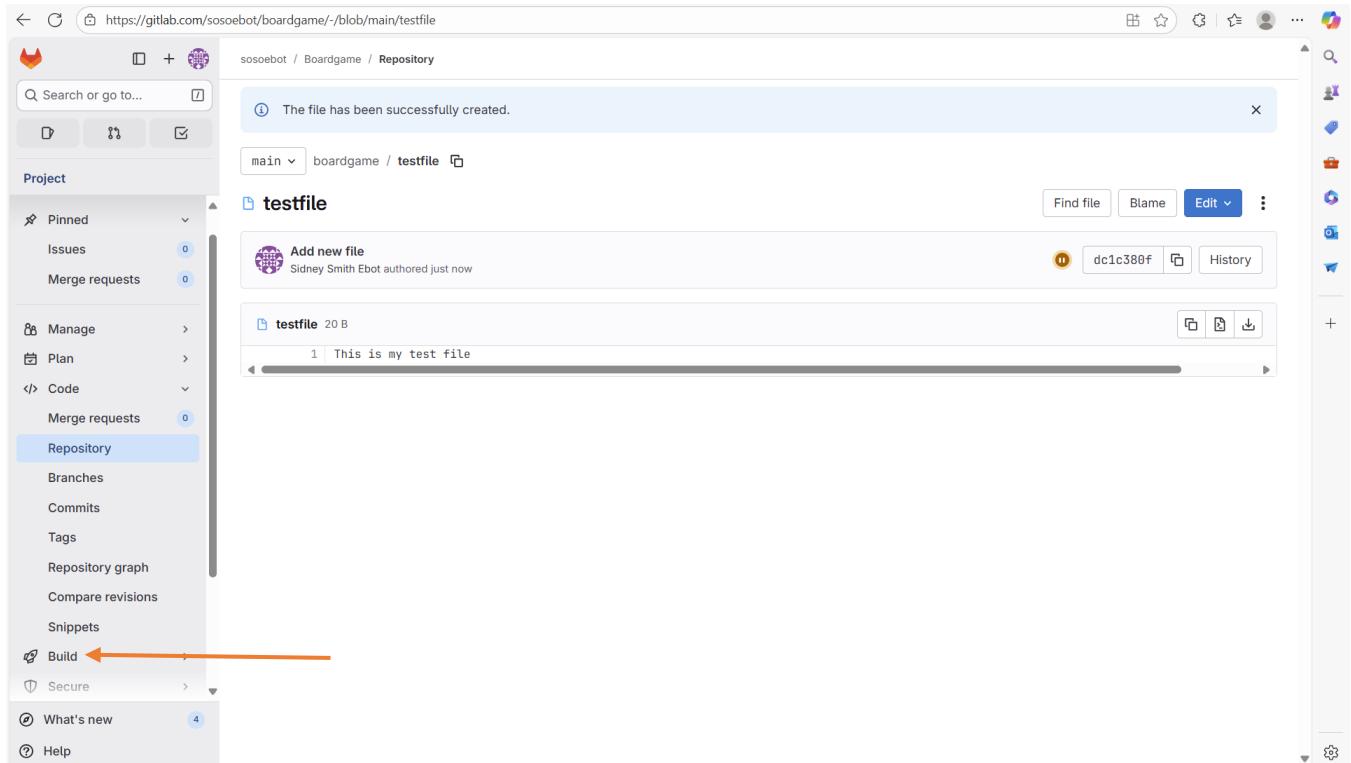


Then commit the changes by clicking on “Commit changes”

Prepared by Sidney Smith



Then click on “Commit changes” again



Then click on “Build”

Prepared by Sidney Smith

The screenshot shows a GitLab repository page for 'sosobot / Boardgame'. A message at the top says 'The file has been successfully created.' Below it, a file named 'testfile' is shown with a size of 20 B. The file content is '1 | This is my test file'. On the left, a sidebar menu is open under 'Repository', showing options like 'Branches', 'Commits', and 'Tags'. A dropdown menu is overlaid on the sidebar, with 'Pipelines' highlighted. An orange arrow points from the text 'And select "Pipeline"' to the 'Pipelines' option in the dropdown.

And select “Pipeline”

The screenshot shows the 'Pipelines' page for the same repository. It lists two pipelines: one 'Running' and one 'Passed'. The 'Running' pipeline was triggered by a commit and is currently executing stages. The 'Passed' pipeline was triggered 17 minutes ago and completed successfully. The sidebar on the left is identical to the previous screenshot, with 'Pipelines' selected.

Status	Pipeline	Created by	Stages	Actions
Running	Add new file #19 ↗ main ↘ dc1c380f [latest] [branch]	[User Icon]	[Green checkmark] [Blue circle] [Yellow circle] [Red circle]	[Delete] [More]
Passed	Edit sonar-project.properties #18 ↗ main ↘ 3b3b4687 [branch]	[User Icon]	[Green checkmark] [Green checkmark] [Green checkmark] [Green checkmark]	[More]

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. The pipeline is named 'main' and consists of four 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 various project management and pipeline-related options.

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

The screenshot shows a SonarQube dashboard for a project named 'mywebsite'. The main header includes navigation links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. Below the header, there's a search bar and a dropdown menu set to 'main'. The main content area displays the 'main' branch analysis. It shows 1.3k Lines of Code and a Version of 'not provided'. A prominent green 'Passed' status is displayed with a checkmark icon. A message at the top encourages users to 'Don't let issues accumulate. Discover 'Clean as You Code!''. Below this, a note states 'The last analysis has warnings. See details'. The dashboard also features tabs for New Code and Overall Code, and sections for Security (2 Open issues), Reliability (18 Open issues), and Maintainability (41 Open issues).

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
not provided Quality Gate: Passed

New analysis: +0 Issues

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

[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

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

The screenshot shows the Docker Hub interface for the repository `ebotsidneysmith/demoapp`. The left sidebar is collapsed, showing options like 'Repositories', 'Hardened Images', 'Collaborations', 'Settings', 'Default privacy', 'Notifications', 'Billing', 'Usage', 'Pulls', and 'Storage'. The main content area displays the repository details: 'ebotsidneysmith/demoapp' was last pushed 2 minutes ago, with a size of 281.1 MB. There are tabs for 'General', 'Tags', 'Image Management (BETA)', 'Collaborators', 'Webhooks', and 'Settings'. The 'Tags' tab is selected, showing one tag: `boardgame-0.0.1`. A red arrow points from the text 'This is the built image.' to this tag. To the right of the tags table, there's a 'Docker commands' section with the command `docker push ebotsidneysmith/demoapp:tagname`. A sidebar for 'buildcloud' is visible on the right.

ebotsidneysmith / demoapp / General

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](#)

Using 0 of 1 private repositories.

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](#)

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](#)

This is the built image.