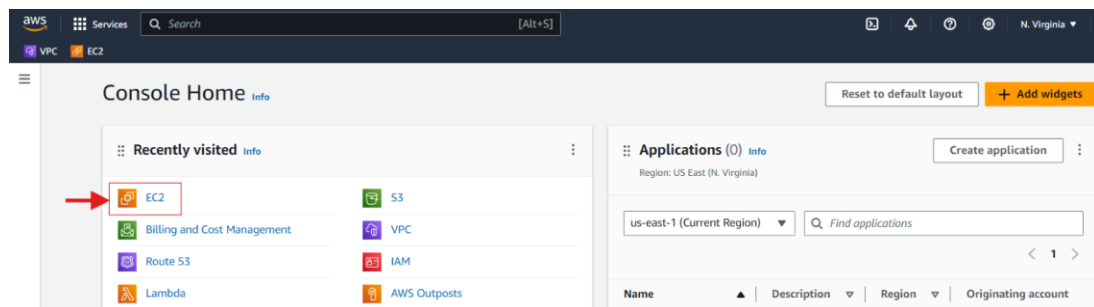


# INSTALLING PRE-REQUISITES

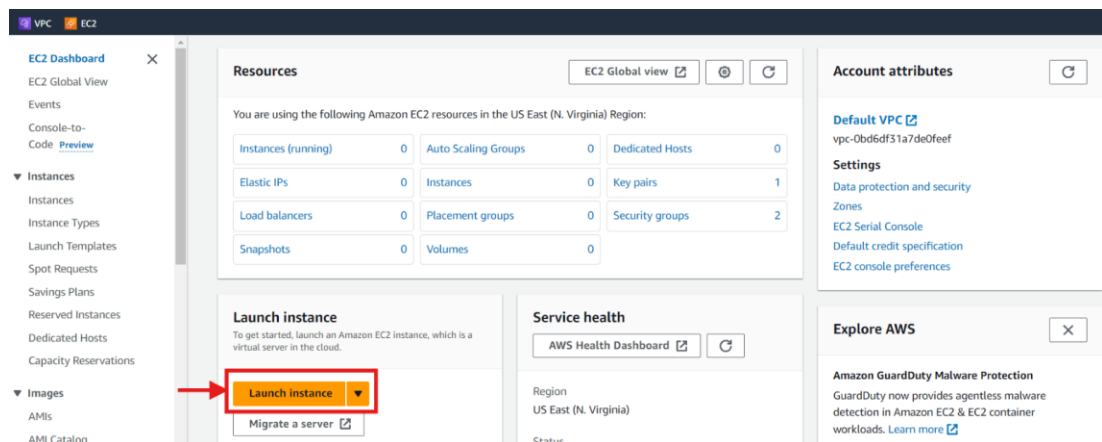
## INSTALL AND SETUP JENKINS

### ✓ Spin Up EC2 instance in AWS

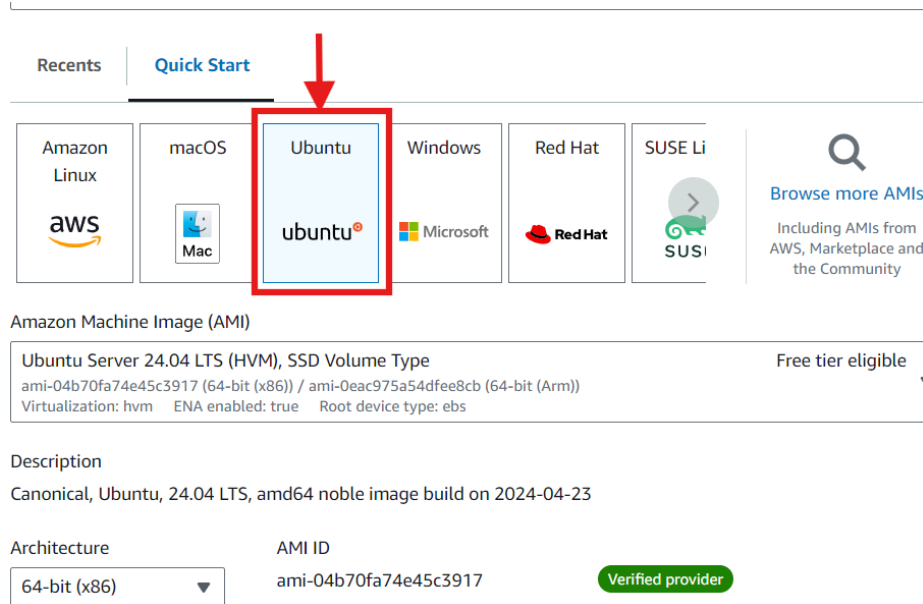
- Login to **AWS console** and **select EC2 service** (If you're using a noon root account make sure you have the necessary permissions/roles)



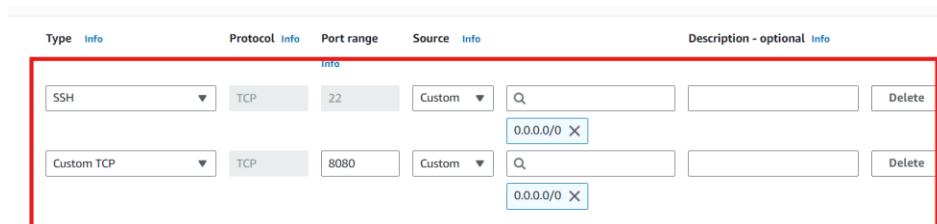
- Select **Launch Instance**, in EC2 service dashboard



- In the launch instance windows, configure the following:-
  - Give the instance a name
  - Select the **Ubuntu Server 20.04 LTS** Quick start Image



- Select any of the Instance types (min **T2.small**)
- **Select a key pair** from available once (\* need for SSH)
- In the network setting leave everything as defaults and **select a Security group with the following Inbound traffic ports opened 22 (SSH) 8080 (Jenkins UI)**



- In a configure Storage section set minimum 10GB.(\* not mandatory)
- Finish and launch the instance.

## ✓ Install Java 17

- Login to server, and run the following commands (\*Ensure user has sudo user privileges)

```
sudo apt update
```

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

## ✓ Setup Jenkins repository in APT

- Run the following commands

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/" | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

## ✓ Install and start Jenkins

- Run the following commands

```
sudo apt-get update && \
sudo apt-get install jenkins -y && \
sudo systemctl enable jenkins && \
sudo systemctl start jenkins && \
sudo systemctl status jenkins
```

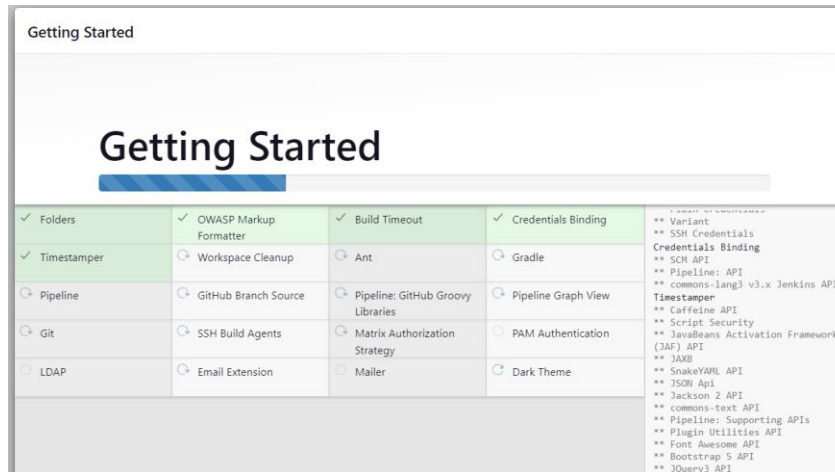
- Make sure that Jenkins service status is running.

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Tue 2024-07-02 08:09:21 UTC; 4s ago
     Main PID: 4659 (java)
       Tasks: 40 (limit: 2338)
      Memory: 343.2M (peak: 393.8M)
         CPU: 48.472s
    CGroup: /system.slice/jenkins.service
            └─4659 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080
```

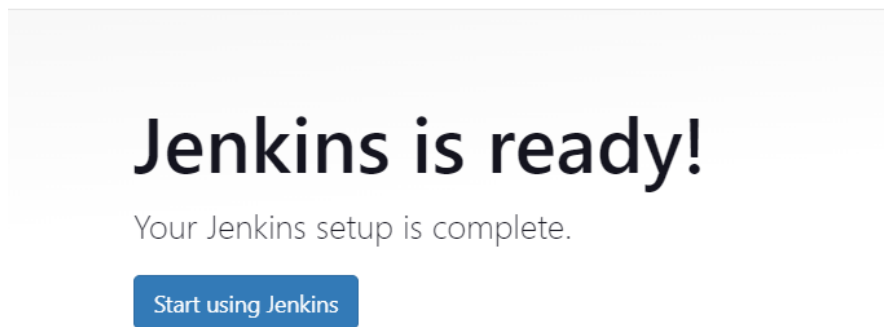
- Access the https URL of Jenkins (<https://<server-public-ip>:8080>) and setup the server.
- Extract the Default Admin password, run the following command

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

- Start the setup and install suggested plugins



- Finish the setup



## INSTALL AND SETUP SONARQUBE SERVER

### ✓ Spin Up EC2 instance in AWS

- Login to **AWS console** and **select EC2 service** (If you're using a new root account make sure you have the necessary permissions/roles.)
- Select **Launch Instance**, in EC2 service dashboard
- In the launch instance windows, configure the following:-
  - Give the instance a name
  - Select the **Ubuntu Server 20.04 LTS** Quick start Image
  - Select any of the Instance types (min **T2.medium**)
  - **Select a key pair** from available once (\* need for SSH)
  - In the network setting leave everything as defaults and **select a Security group with the following Inbound traffic ports opened 22 (SSH) 9000 (Sonarqube UI)**

Type	Info	Protocol	Info	Port range	Source	Info	Description - optional	Info
SSH		TCP		22	Custom			Delete
Custom TCP		TCP		9000	Custom			Delete

- In a configure Storage section set minimum 10GB.(\* not mandatory)
- Finish and launch the instance.

## ✓ Install Docker

- Login to server, and run the following commands (\*Ensure user has sudo user privileges)
- Add Docker's official GPG key:

```
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

- Add the repository to Apt sources:

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

- Install docker and its components

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

## ✓ Configure non-root executable permission to Docker

- Run the following commands

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

## ✓ Start SonarQube Container

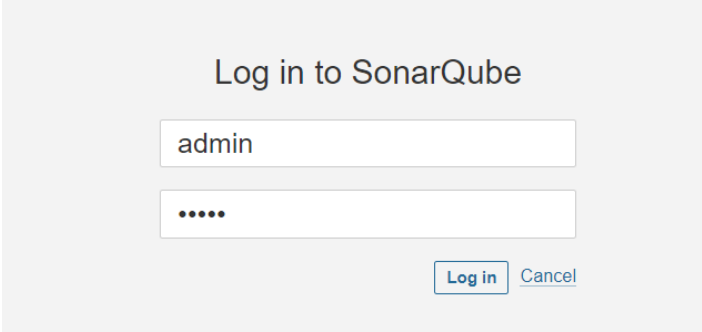
- Run the following commands (\*here we're using the lts community version since it doesn't require any license)

```
docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
ubuntu@ip-172-31-90-0:~$ docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
2ec76a50fe7c: Pull complete
fab7f202453a: Pull complete
ee59ca42def8: Pull complete
2ce2282f972f: Pull complete
d2a9e456ba82: Pull complete
3c4f423f73fe: Pull complete
0110391fdd36: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:0d36ee97e5f5458351913e69aa5ae874d832d886704573bbdce43779c0294733
Status: Downloaded newer image for sonarqube:lts-community
41a99420487275eb30d3051338658f66268308a095116cd125ea3e6b3fec7549
```

- Check if the container is up and running by running following commands

```
docker ps
ubuntu@ip-172-31-90-0:~$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS        PORTS
41a994204872   sonarqube:lts-community "/opt/sonarqube/dock..." About a minute ago Up About a minute 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp
sonar
```

- Login to Sonarqube UI (<http://<server-public-ip>:9000>) using the default password (**Username: admin, Password: admin**).



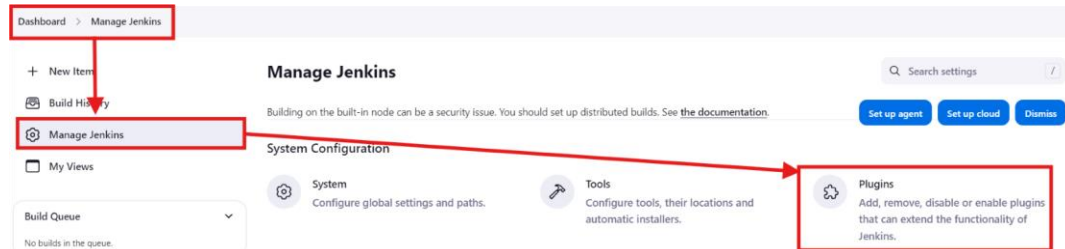
The image shows a web browser window with the title "Log in to SonarQube". It contains a login form with two input fields: one for the username, which has "admin" entered, and one for the password, which has four dots "...." entered. Below the password field are two buttons: "Log in" and "Cancel".

- Update the password and finish the setup.

# POST INSTALLATION CONFIGURATION

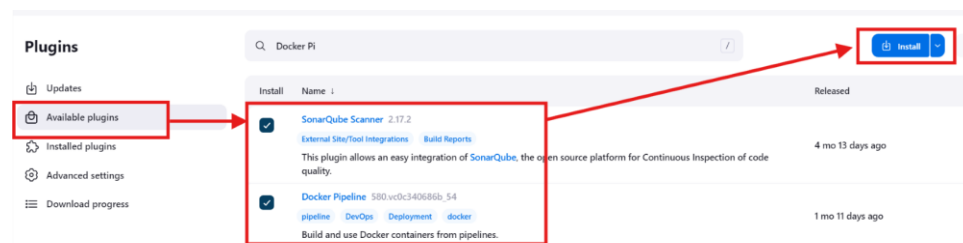
## ✓ Install the necessary Plugins ( SonarScanner, Docker Pipeline)

- In the **Jenkins Dashboard** select **Manage Jenkins**.



- Select **Plugins > Available Plugins > Search for**

- **SonarQube Scanner (Configure Sonar server and Scanner)**
- **Docker Pipeline (for using docker node to run pipelines)**



- Install the Plugin's

→ [Go back to the top page](#)

(you can start using the installed plugins right away)



→ ☐ Restart Jenkins when installation is complete and no jobs are running

## ✓ Configure Sonar Server and Scanner in Jenkins

- In the **Jenkins Dashboard** select **Manage Jenkins**.
- Select the **System** option and scroll down to **Sonar-Server** Section

Dashboard > Manage Jenkins > System >

☐ Disable deferred wipeout on this node ?  
☐ Disk Space Monitoring Thresholds  
☐ Environment variables  
☐ Tool Locations

---

**SonarQube servers**

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

☐ Environment variables

SonarQube installations

List of SonarQube installations

[Add SonarQube](#)

- Setup the Sonar-server instance.
- Give a preferred name to the instance (\*to be used in pipeline stage import Sonar sever configuration as environment variables)
- Input the server URL
- Create a Token in Sonar server by heading over to **User > Security > Global Token Create Token**. Make sure to copy the token.

Tokens

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

**Generate Tokens**

Name: 
 Type: 
 Expires in: 
[Generate](#)

Name	Type	Project	Last use	Created	Expiration
No tokens					

- Select add token, and create a new Secret file from the drop-down menu

Jenkins Credentials Provider: Jenkins

**Add Credentials**

Domain:

Kind:

Scope ?

Secret:

ID ?



## ➤ Save the Configuration

SonarQube installations  
List of SonarQube installations

Name  
Sonarqube

Server URL  
Default is http://localhost:9000  
http://184.72.80.50:9000/

Server authentication token  
SonarQube authentication token. Mandatory when anonymous access is disabled.  
Sonarqube-Token  
+ Add

Advanced

Save Apply

## ➤ Ensure to tick the option Environment variable (\*to ingest configuration as environment variable to builds)

### SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build

☒ Environment variables

## ✓ Configure SonarScanner

- In the **tools section** headover to **SonarScanner**
- Configure the scanner **Version**(\* not mandatory if latest version is suitable), **and click Install Automatically**
- Save the changes

SonarQube Scanner installations

Add SonarQube Scanner

SonarQube Scanner

Name  
SonarScanner

☒ Install automatically ?

Install from Maven Central

Version  
SonarQube Scanner 6.1.0.4477

Add Installer

Save Apply

## ✓ Configure Docker

- In the **tools section headover to Docker**
- Give it a name
- Input the installation root directory if already installed docker on host machine, else tick install automatically
- Save the changes

The screenshot shows the Jenkins configuration page for Docker. The breadcrumb navigation at the top is "Dashboard > Manage Jenkins > Tools", which is highlighted with a red box. Below this, the page title is "Docker installations". A red arrow points from the "Tools" part of the breadcrumb to the "Add Docker" button. The "Add Docker" button is a light gray button with the text "Add Docker". Below this button is a form titled "Docker" with a hamburger menu icon on the left. The form contains three fields: "Name" with the value "docker", "Installation root" with a question mark icon and an empty text box, and a checkbox labeled "Install automatically" with a question mark icon. A red box highlights the entire form area. Below the form is another "Add Docker" button. A red arrow points from this button to the "Save" button. The "Save" button is a blue button with the text "Save", and it is highlighted with a red box. To the right of the "Save" button is an "Apply" button, which is a light gray button with the text "Apply".

Dashboard > Manage Jenkins > Tools

Docker installations

Add Docker

**Docker**

Name

docker

Installation root ?

☐ Install automatically ?

Add Docker

**Save** Apply