**Install Docker on Ubuntu**

**1. Update packages**

bash

CopyEdit

sudo apt update

sudo apt upgrade -y

**2. Install dependencies**

bash

CopyEdit

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

**3. Add Docker’s GPG key and repository**

bash

CopyEdit

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

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

**4. Install Docker**

bash

CopyEdit

sudo apt update

sudo apt install docker-ce docker-ce-cli containerd.io -y

**5. Enable and start Docker**

bash

CopyEdit

sudo systemctl enable docker

sudo systemctl start docker

**6. Add your user to the Docker group (optional, for running docker without sudo)**

bash

CopyEdit

sudo usermod -aG docker $USER

newgrp docker

**Part 2: Install Jenkins on Ubuntu**

**1. Install Java (Jenkins requires Java)**

bash

CopyEdit

sudo apt install openjdk-17-jdk -y

**2. Add Jenkins repo and GPG key**

bash

CopyEdit

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee \

/usr/share/keyrings/jenkins-keyring.asc > /dev/null

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

**3. Install Jenkins**

bash

CopyEdit

sudo apt update

sudo apt install jenkins -y

**4. Start and enable Jenkins**

bash

CopyEdit

sudo systemctl enable jenkins

sudo systemctl start jenkins

**Part 3: Access Jenkins Web UI**

**1. Get initial admin password**

bash

CopyEdit

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

**2. Access Jenkins UI in browser**

Navigate to:  
http://<your-server-ip>:8080  
Enter the password from above and complete setup.

**Part 4: Let Jenkins Use Docker**

By default, Jenkins runs as its own user (jenkins), which won’t have Docker access unless you allow it.

**1. Add jenkins user to the docker group:**

bash

CopyEdit

sudo usermod -aG docker jenkins

sudo systemctl restart jenkins

## Optional: Install Docker inside Jenkins (via Plugin)

1. In Jenkins UI: **Manage Jenkins > Plugins > Available**
2. Install **Docker** and **Docker Pipeline** plugins
3. Configure Docker hosts and credentials as needed

**Install postgresql for Ubuntu**

**1 . Download the PostgreSQL on Linux server.**

Ubuntu includes PostgreSQL by default. To install PostgreSQL on Ubuntu, use the apt (or other apt-driving) command:

code : sudo apt install postgresql

**2 . Now we need to install the pgadmin4 for the web to run the postgresql**

so we will download the pgadmin4 for the Linux world

To use this repository, run the following commands:

# Setup the repository

# Install the public key for the repository (if not done previously):

curl -fsS https://www.pgadmin.org/static/packages\_pgadmin\_org.pub | sudo gpg --dearmor -o /usr/share/keyrings/packages-pgadmin-org.gpg

# Create the repository configuration file:

sudo sh -c 'echo "deb [signed-by=/usr/share/keyrings/packages-pgadmin-org.gpg] https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb\_release -cs) pgadmin4 main" > /etc/apt/sources.list.d/pgadmin4.list && apt update'

# Install pgAdmin

# Install for both desktop and web modes:

sudo apt install pgadmin4

# Install for desktop mode only:

sudo apt install pgadmin4-desktop

# Install for web mode only:

sudo apt install pgadmin4-web

# Configure the webserver, if you installed pgadmin4-web:

sudo /usr/pgadmin4/bin/setup-web.sh

run these command on the server

The Apache web server is running and must be restarted for the pgAdmin4 installation to complete. Continue (y/n)? y

Apache successfully restarted. You can now start using pgAdmin4 in web mode at http://127.0.0.1/pgadmin4

**Full Process: Connect PostgreSQL to a Remote Server**

**STEP 1: Install PostgreSQL (if not already installed)**

sudo apt update

sudo apt install postgresql

sudo systemctl status PostgreSQL

**STEP 2: Create a PostgreSQL User and Database**

**1. Switch to the postgres user**

sudo -u postgres

psql

**2. Create a new user and database**

CREATE USER remoteuser WITH PASSWORD 'StrongP@ssw0rd';

CREATE DATABASE remotedb;

GRANT ALL PRIVILEGES ON DATABASE remotedb TO remoteuser;

\q

**STEP 3: Configure PostgreSQL to Accept Remote Connections**

**1. Edit postgresql.conf to listen on all IPs**

sudo nano /etc/postgresql/\*/main/postgresql.conf

find there

#listen\_addresses = 'localhost'

Change it to:

listen\_addresses = '\*'

then save and exit

**2. Edit pg\_hba.conf to Allow Remote Connections**

sudo nano /etc/postgresql/\*/main/pg\_hba.conf

At the bottom, add:

# Allow all IPs (use cautiously, see note below)

host all all 0.0.0.0/0 md5

Important: For security, replace 0.0.0.0/0 with a specific IP or CIDR block (e.g., 192.168.1.100/32) if possible.

**3. Restart PostgreSQL**

sudo systemctl restart postgresql

**Step 1: install springboot and maven download for the Ubuntu server if you are develop the project in server and if you are just run the docker file in the server then no need to download it.**

**Step 1: Install Java (JDK 17 or later)**

Spring Boot 3+ requires Java 17 or higher.

bash

CopyEdit

sudo apt update

sudo apt install openjdk-17-jdk -y

**Verify:**

bash

CopyEdit

java -version

**Step 2: Install Maven**

Ubuntu repositories often have outdated Maven versions. Use the latest from Apache:

**Download and Install Latest Maven:**

bash

CopyEdit

# Set version

MAVEN\_VERSION=3.9.6

# Download and extract

wget https://downloads.apache.org/maven/maven-3/${MAVEN\_VERSION}/binaries/apache-maven-${MAVEN\_VERSION}-bin.tar.gz

sudo mkdir -p /opt/maven

sudo tar -xvzf apache-maven-${MAVEN\_VERSION}-bin.tar.gz -C /opt/maven

sudo ln -s /opt/maven/apache-maven-${MAVEN\_VERSION} /opt/maven/latest

**Add Maven to PATH:**

Create a file /etc/profile.d/maven.sh:

bash

CopyEdit

sudo nano /etc/profile.d/maven.sh

Paste:

bash

CopyEdit

export M2\_HOME=/opt/maven/latest

export MAVEN\_HOME=/opt/maven/latest

export PATH=${M2\_HOME}/bin:${PATH}

Make it executable:

bash

CopyEdit

sudo chmod +x /etc/profile.d/maven.sh

source /etc/profile.d/maven.sh

**Verify:**

bash

CopyEdit

mvn -v

**Step 3: Create a Spring Boot Project**

**Option 1: Use** [**Spring Initializr**](https://start.spring.io/)

Generate a ZIP with your config:

* Project: Maven
* Language: Java
* Spring Boot: 3.2.x (or latest)
* Dependencies: Web, JPA, MySQL, etc.

Then:

bash

CopyEdit

unzip your-project.zip

cd your-project

./mvnw spring-boot:run

**Option 2: Use Maven CLI to scaffold manually (minimal):**

bash

CopyEdit

mvn archetype:generate \

-DgroupId=com.example.demo \

-DartifactId=demo \

-DarchetypeArtifactId=maven-archetype-quickstart \

-DinteractiveMode=false

Note: This will not include Spring Boot — it's a basic Java app.

**Step 4: Build and Run the App**

Once you have a Spring Boot app:

bash

CopyEdit

cd your-project

mvn clean package

java -jar target/\*.jar

Or run with:

bash

CopyEdit

mvn spring-boot:run

**THANK YOU**