**✅ 1. Mettre à jour le système**

bash

CopierModifier

sudo apt update && sudo apt upgrade -y

**✅ 2. Installer le serveur OpenSSH**

bash

CopierModifier

sudo apt install openssh-server -y

**✅ 3. Vérifier que le service SSH fonctionne**

bash

CopierModifier

sudo systemctl status ssh

Tu devrais voir :

arduino

CopierModifier

Active: active (running)

Si ce n’est pas le cas, démarre-le :

bash

CopierModifier

sudo systemctl start ssh

**✅ 4. Activer le démarrage automatique au boot**

bash

CopierModifier

sudo systemctl enable ssh

**✅ 5. Vérifier que le port SSH est bien ouvert (optionnel mais conseillé)**

Si tu utilises un pare-feu UFW :

bash

CopierModifier

sudo ufw allow ssh

sudo ufw enable # si ce n’est pas encore fait

sudo ufw status

**✅ 6. Connexion depuis une autre machine**

Depuis un autre PC, tape :

bash

CopierModifier

ssh <utilisateur>@<ip\_du\_serveur>

Exemple :

bash

CopierModifier

ssh maxime@192.168.40.144

**✅ Fichier de configuration (facultatif)**

Le fichier de config se trouve ici :

bash

CopierModifier

sudo nano /etc/ssh/sshd\_config

### ✅ Étapes d’installation de Nginx sur Ubuntu

#### 1. **Mettre à jour le système**

bash

CopierModifier

sudo apt update && sudo apt upgrade -y

#### 2. **Installer Nginx**

bash

CopierModifier

sudo apt install nginx -y

#### 3. **Vérifier que Nginx fonctionne**

bash

CopierModifier

sudo systemctl status nginx

Tu devrais voir un message indiquant que le service est **"active (running)"**.  
Sinon, démarre-le avec :

bash

CopierModifier

sudo systemctl start nginx

#### 4. **Activer Nginx au démarrage**

bash

CopierModifier

sudo systemctl enable nginx

#### 5. **Tester dans un navigateur**

Va sur l'adresse IP de ton serveur ou machine :

cpp

CopierModifier

http://<adresse-ip>

Tu devrais voir la page d’accueil par défaut de Nginx : **"Welcome to nginx!"**

### 🔥 Pare-feu (UFW) – Si activé

Si tu utilises le pare-feu UFW, autorise Nginx :

bash

CopierModifier

sudo ufw allow 'Nginx Full'

Puis vérifie :

bash

CopierModifier

sudo ufw status

#### . 🔌 Vérifie ta configuration réseau (IP, passerelle, DNS)

Tu as mentionné utiliser une IP statique.

Affiche la config réseau :

bash

CopierModifier

ip a

Et la passerelle :

bash

CopierModifier

ip r

Puis vérifie le fichier de config netplan :

bash

CopierModifier

sudo nano /etc/netplan/01-netcfg.yaml

Il doit ressembler à ceci (par exemple) :

yaml

CopierModifier

network:

version: 2

renderer: networkd

ethernets:

ens33:

dhcp4: no

addresses:

- 192.168.40.144/24

gateway4: 192.168.40.254

nameservers:

addresses: [8.8.8.8, 8.8.4.4]

Ensuite applique :

bash

CopierModifier

sudo netplan apply

## 🐳 Étapes officielles pour installer Docker sur Ubuntu (toutes versions)

### ✅ 1. ****Mettre à jour le système****

bash

CopierModifier

sudo apt update && sudo apt upgrade -y

### ✅ 2. ****Installer les dépendances nécessaires****

bash

CopierModifier

sudo apt install ca-certificates curl gnupg lsb-release -y

### ✅ 3. ****Ajouter la clé GPG officielle de Docker****

bash

CopierModifier

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \

sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

### ✅ 4. ****Ajouter le dépôt officiel de Docker****

bash

CopierModifier

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \

https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

### ✅ 5. ****Mettre à jour les paquets avec le nouveau dépôt Docker****

bash

CopierModifier

sudo apt update

### ✅ 6. ****Installer Docker Engine, Docker CLI et containerd****

bash

CopierModifier

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

### ✅ 7. ****Vérifier que Docker fonctionne****

bash

CopierModifier

sudo docker version

sudo systemctl status docker

### ✅ 8. ****(Optionnel) Ajouter ton utilisateur au groupe Docker (pour ne pas utiliser**** sudo****)****

bash

CopierModifier

sudo usermod -aG docker $USER

Ensuite **déconnecte-toi puis reconnecte-toi** ou fais newgrp docker pour que ça prenne effet.

### ✅ 9. ****Tester avec un conteneur de test****

bash

CopierModifier

docker run hello-world

## ✅ Étapes pour installer PostgreSQL sur Ubuntu

### 1. 🆙 Mettre à jour les paquets

bash

CopierModifier

sudo apt update && sudo apt upgrade -y

### 2. 🐘 Installer PostgreSQL et ses utilitaires

bash

CopierModifier

sudo apt install postgresql postgresql-contrib -y

Cela installe :

* le serveur PostgreSQL
* l’outil en ligne de commande psql
* les utilitaires supplémentaires (contrib)

### 3. 🔄 Vérifier que PostgreSQL tourne

bash

CopierModifier

sudo systemctl status postgresql

Tu devrais voir :

arduino

CopierModifier

Active: active (running)

Sinon, démarre-le :

bash

CopierModifier

sudo systemctl start postgresql

Et active-le au démarrage :

bash

CopierModifier

sudo systemctl enable postgresql

### 4. 👤 Se connecter en tant qu'utilisateur PostgreSQL

bash

CopierModifier

sudo -i -u postgres

Puis lancer le shell SQL :

bash

CopierModifier

psql

Tu arrives dans l’interface psql :

makefile

CopierModifier

postgres=#

Quitter :

sql

CopierModifier

\q

### 5. 🔐 Créer une base, un utilisateur et un mot de passe

bash

CopierModifier

sudo -i -u postgres

psql

Dans psql :

sql

CopierModifier

CREATE DATABASE nom\_de\_ma\_base;

CREATE USER mon\_utilisateur WITH PASSWORD 'mon\_mot\_de\_passe';

GRANT ALL PRIVILEGES ON DATABASE nom\_de\_ma\_base TO mon\_utilisateur;

### 6. (Optionnel) 📡 Autoriser l’accès distant

1. Modifier le fichier :

bash

CopierModifier

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

Changer :

c

CopierModifier

# listen\_addresses = 'localhost'

en :

conf

CopierModifier

listen\_addresses = '\*'

1. Modifier les règles d'accès :

bash

CopierModifier

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

Ajouter à la fin :

css

CopierModifier

host all all 0.0.0.0/0 md5

1. Redémarrer PostgreSQL :

bash

CopierModifier

sudo systemctl restart postgresql

### 7. (Optionnel) Tester la connexion avec psql

Depuis une autre machine :

bash

CopierModifier

psql -h <ip\_du\_serveur> -U mon\_utilisateur -d nom\_de\_ma\_base

## ✅ 1. Mise à jour du système

bash

CopierModifier

sudo apt update && sudo apt upgrade -y

## ✅ 2. Installer Node.js et npm via NodeSource (méthode recommandée)

Node.js inclus souvent une version récente de **npm** automatiquement.

### ➤ a. Télécharger et ajouter le dépôt officiel NodeSource

Pour la **dernière version LTS (ex: Node.js 20)** :

bash

CopierModifier

curl -fsSL https://deb.nodesource.com/setup\_lts.x | sudo -E bash -

⚠️ Si tu n’as pas de connexion Internet ou si curl échoue, résous d’abord le problème réseau.

### ➤ b. Installer Node.js (et npm avec)

bash

CopierModifier

sudo apt install nodejs -y

## ✅ 3. Vérifier les versions installées

bash

CopierModifier

node -v

npm -v

Tu devrais voir quelque chose comme :

CopierModifier

v20.x.x

9.x.x

## ✅ 4. (Optionnel) Installer n pour gérer plusieurs versions de Node

bash

CopierModifier

sudo npm install -g n

sudo n lts # pour installer la dernière version LTS

## ✅ 5. (Optionnel) Installer yarn (alternative à npm)

bash

CopierModifier

npm install -g yarn

# Configuration d’un reverse proxy Nginx

## 1. **Installation de Nginx** (si ce n’est pas déjà fait)

bash

CopierModifier

sudo apt update

sudo apt install nginx -y

## 2. **Créer un fichier de configuration pour ton site**

Supposons que tu souhaites faire un reverse proxy vers une application web qui tourne en local sur le port 3000.

### Exemple de fichier config Nginx

Crée un fichier, par exemple /etc/nginx/sites-available/monproxy :

nginx

CopierModifier

server {

listen 80;

server\_name monsite.local; # Remplace par ton nom de domaine ou IP

location / {

proxy\_pass http://localhost:3000; # adresse de ton app backend

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection 'upgrade';

proxy\_set\_header Host $host;

proxy\_cache\_bypass $http\_upgrade;

}

}

## 3. **Activer la configuration**

Crée un lien symbolique dans sites-enabled :

bash

CopierModifier

sudo ln -s /etc/nginx/sites-available/monproxy /etc/nginx/sites-enabled/

## 4. **Tester la configuration et recharger Nginx**

bash

CopierModifier

sudo nginx -t

sudo systemctl reload nginx

## 5. **Ajouter une entrée dans le fichier hosts (si c’est en local)**

Sur ta machine cliente (Windows, Linux, macOS), ajoute dans /etc/hosts ou C:\Windows\System32\drivers\etc\hosts :

lua

CopierModifier

192.168.x.x monsite.local

(Où 192.168.x.x est l’IP de ta machine serveur)

## 6. **Tester dans ton navigateur**

Va sur :

arduino

CopierModifier

http://monsite.local

Tu dois voir ton application qui tourne sur le port 3000 mais via Nginx.

## 1. Installer JDK 21 (Java 21)

### Option A — Installer OpenJDK 21 via PPA (si disponible)

Ubuntu ne fournit pas toujours la dernière version dans ses dépôts officiels.  
Tu peux installer OpenJDK 21 via un PPA tiers ou directement télécharger depuis Adoptium.

### Option B — Installer OpenJDK 21 via Adoptium (recommandé)

1. **Télécharger et installer OpenJDK 21**

bash

CopierModifier

wget https://github.com/adoptium/temurin21-binaries/releases/latest/download/OpenJDK21U-jdk\_x64\_linux\_hotspot.tar.gz

sudo mkdir -p /usr/lib/jvm

sudo tar -xzf OpenJDK21U-jdk\_x64\_linux\_hotspot.tar.gz -C /usr/lib/jvm

1. **Configurer la variable JAVA\_HOME et alternatives**

Remplace jdk-21.0.x+xx par le dossier extrait exact (vérifie dans /usr/lib/jvm)

bash

CopierModifier

sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk-21.0.x+xx/bin/java 1

sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/jdk-21.0.x+xx/bin/javac 1

sudo update-alternatives --config java

sudo update-alternatives --config javac

1. **Vérifier l’installation**

bash

CopierModifier

java -version

javac -version

## 2. Installer Maven

### Via APT (version stable mais parfois pas la dernière)

bash

CopierModifier

sudo apt update

sudo apt install maven -y

### Vérifier la version de Maven

bash

CopierModifier

mvn -version

### Option avancée : installer la dernière version Maven manuellement

1. Télécharger la dernière version (exemple 3.9.4)

bash

CopierModifier

wget https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.tar.gz

sudo tar -xzf apache-maven-3.9.4-bin.tar.gz -C /opt/

sudo ln -s /opt/apache-maven-3.9.4 /opt/maven

1. Configurer les variables d’environnement

Édite /etc/profile.d/maven.sh :

bash

CopierModifier

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

Ajoute :

bash

CopierModifier

export M2\_HOME=/opt/maven

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

Enregistre, puis rends exécutable :

bash

CopierModifier

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

source /etc/profile.d/maven.sh

1. Vérifie la version

bash

CopierModifier

mvn -version

Pour installer la version **LTS (Long Term Support)** de **Python** sur Ubuntu (généralement Python 3.x), voici la marche à suivre simple et efficace.

## 1. Mettre à jour la liste des paquets

bash

CopierModifier

sudo apt update

sudo apt upgrade -y

## 2. Installer Python (version recommandée 3.x)

Sur Ubuntu 22.04 et plus récent, Python 3 est installé par défaut, mais si besoin :

bash

CopierModifier

sudo apt install python3 python3-pip -y

Cela installe Python 3 et pip pour gérer les paquets Python.

## 3. Vérifier la version installée

bash

CopierModifier

python3 --version

pip3 --version

## 4. (Optionnel) Installer python-is-python3 pour simplifier

Cela permet d’utiliser la commande python au lieu de python3 :

bash

CopierModifier

sudo apt install python-is-python3 -y

## 5. (Optionnel) Installer une version spécifique de Python via deadsnakes PPA

Si tu souhaites une version spécifique (par exemple Python 3.11) :

bash

CopierModifier

sudo add-apt-repository ppa:deadsnakes/ppa

sudo apt update

sudo apt install python3.11 python3.11-venv python3.11-dev -y

Puis tu peux lancer Python 3.11 avec :

bash

CopierModifier

python3.11

Si tu veux, je peux t’aider à :

* créer un environnement virtuel Python (venv),
* installer des paquets avec pip,
* gérer plusieurs versions Python avec pyenv.

### 1. Télécharger le paquet .deb de Chrome

Exécute cette commande pour télécharger la dernière version stable :

bash

CopierModifier

wget https://dl.google.com/linux/direct/google-chrome-stable\_current\_amd64.deb

### 2. Installer le paquet avec dpkg

bash

CopierModifier

sudo dpkg -i google-chrome-stable\_current\_amd64.deb

### 3. Résoudre les dépendances si besoin

Si tu as des erreurs de dépendances, lance :

bash

CopierModifier

sudo apt-get install -f

### 4. Lancer Google Chrome

Tu peux lancer Chrome en tapant :

bash

CopierModifier

google-chrome

ou

bash

CopierModifier

google-chrome-stable