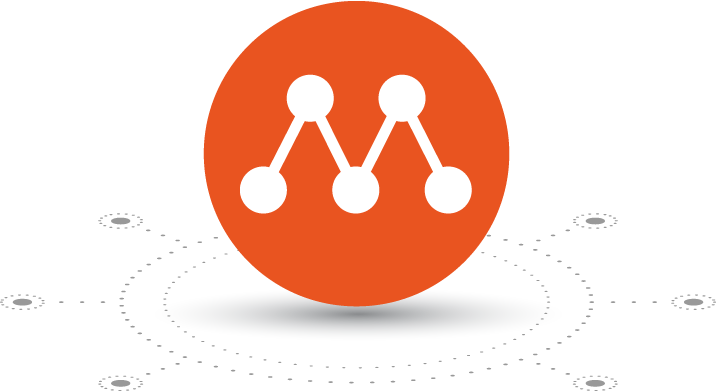


**CCI - SIO Atelier 2 - SACCHETTO Vladimir le 03/05/2022**

Configuration de l’environnement de travail

Rédaction de documentation utilisateur efficace



SOMMAIRE

1. Introduction
2. Environnement des Instances
3. Télécharger Multipass
4. Installation des instances
5. Configuration des instances
   1. Mise à jour des librairies
   2. Installation et Configuration de Apache 2 – PHP – Maria DB
   3. Installation et Configuration de Xdebug - PhpMyAdmin
   4. Création de l’user pour PhpMyAdmin
   5. Installation composer
   6. Cloner le projet git
6. Scripts pour mise en production
7. Introduction

A l’aide de ce document nous allons démontrer comment un jeune développeur Web peut mettre en place une stratégie pour suivre ses projets et tester son code avant de le rendre effectif.

Pour ce faire nous allons devoir créer des instances (machines virtuelles) qui nous donneront la possibilité de suivre nos projets.

Les instances nécessaires sont 3 et ce sont :

* Machine de **TEST**
* Machine de **DEVELOPPEMENT**
* Machine de **PRODUCTION**

Afin de créer ces instances nous avons besoin de télécharger l’outil de création d’instances Ubuntu « Multipass v 1.8.0 ».

Multipass est l’outil qui permet de créer des instances Ubuntu.  Il est conçu pour les développeurs qui souhaitent un nouvel environnement Ubuntu avec une seule commande et fonctionne sous Linux, Windows et macOS.

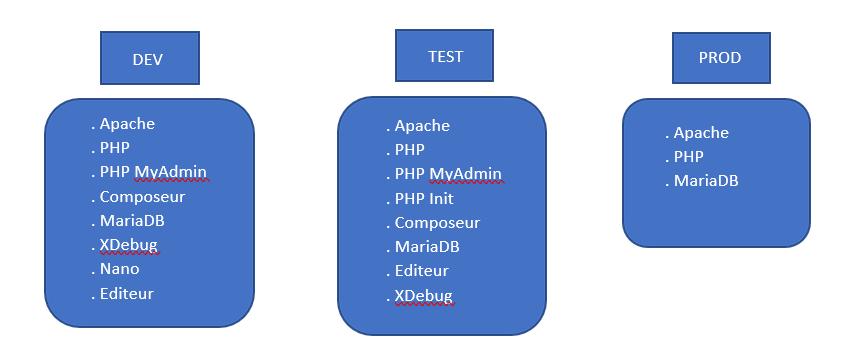
Multipass est une méthode recommandée pour créer des machines virtuelles.

[Créer une instance](https://multipass.run/docs/working-with-instances) [Informations sur Multipass](https://ubuntu.com/server/docs/virtualization-multipass#:~:text=Multipass%20is%20the%20recommended%20method,multipass%20%2D%2Dbeta%20%2D%2Dclassic)

1. Environnement des Instances

Nos machines virtuelles devront respectivement présenter des outils adaptés à l’environnement d’un développeur. Ainsi nous devons nous assurer que chaque instance contienne les outils de travail nécessaire et les installer soit à l’aide d’un script **bash** ou avec les commandes rentrées une par une.

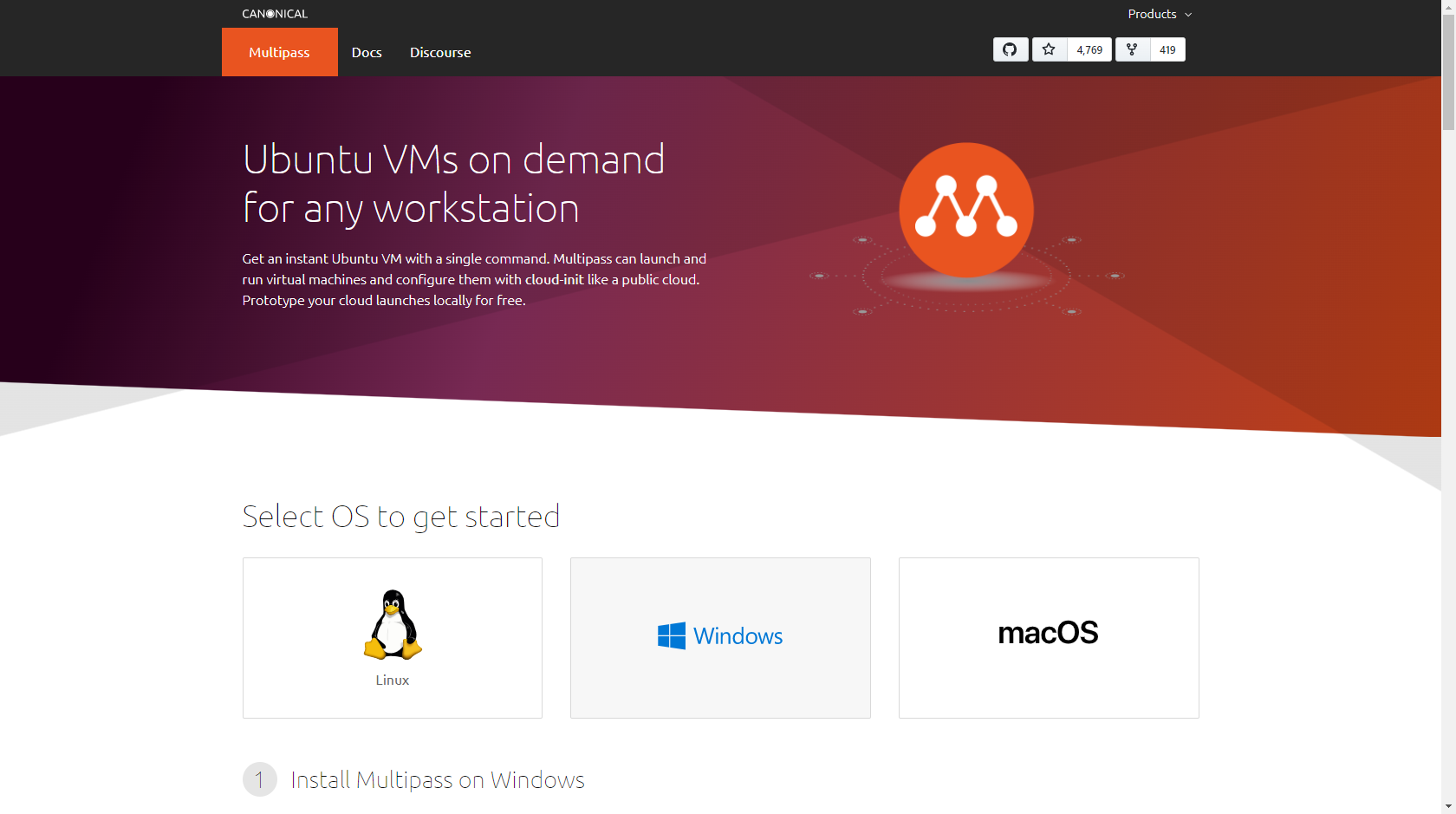
L’environnement pour chaque machine est le suivant :

**

Afin de bien développer et de mettre à jour notre projet au fur et à mesure du temps, nous devons faire communiquer les 3 machines entre elles. Nous devons ainsi utiliser deux scripts que nous allons nommer **Maj Prod** et **Maj Test**. Qui contiennent les indictions

1. Télécharger Multipass

Avant tout nous avons besoin de [télécharger](https://multipass.run/) Multipass.



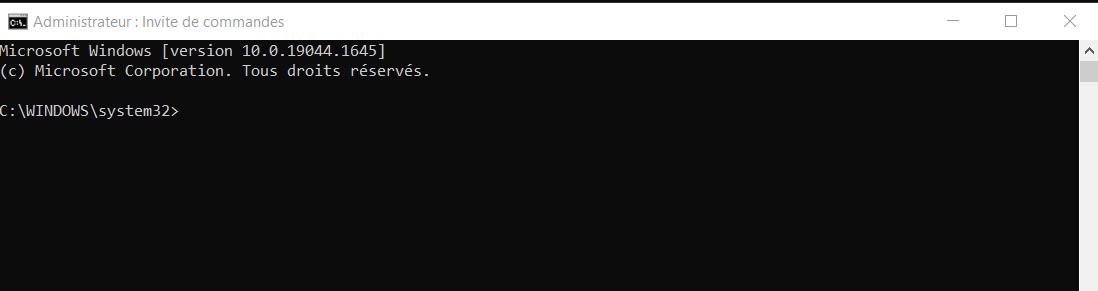
Ceci est la page officielle de Multipass Ubuntu. Ici vous pouvez choisir votre environnement et télécharger la version de Multipass lié à votre OS.

Après avoir téléchargé Multipass vous pouvez procéder à l’installation.

Ce document se base sur un système d’exploitation Windows 10 Professional.

1. Installation des instances

Nous allons alors ouvrir le terminal pour l’invite de commande (CMD) de Windows en tant qu’Administrateur.

**

Afin de créer notre première machine virtuelle nous rentrons la commande multipass launch --name foo.

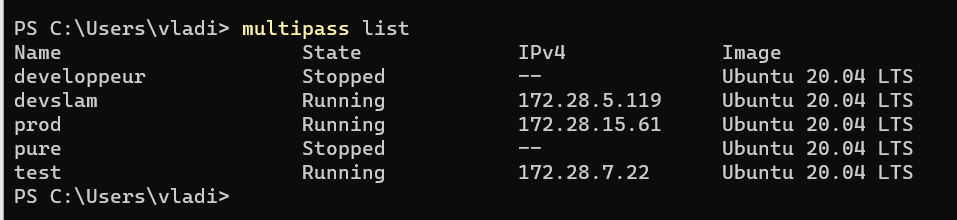
foo est un nom standard de l’instance que nous allons remplacer par **test** ou **production** ou **développeur** comme cela a été défini initialement dans notre document.

**



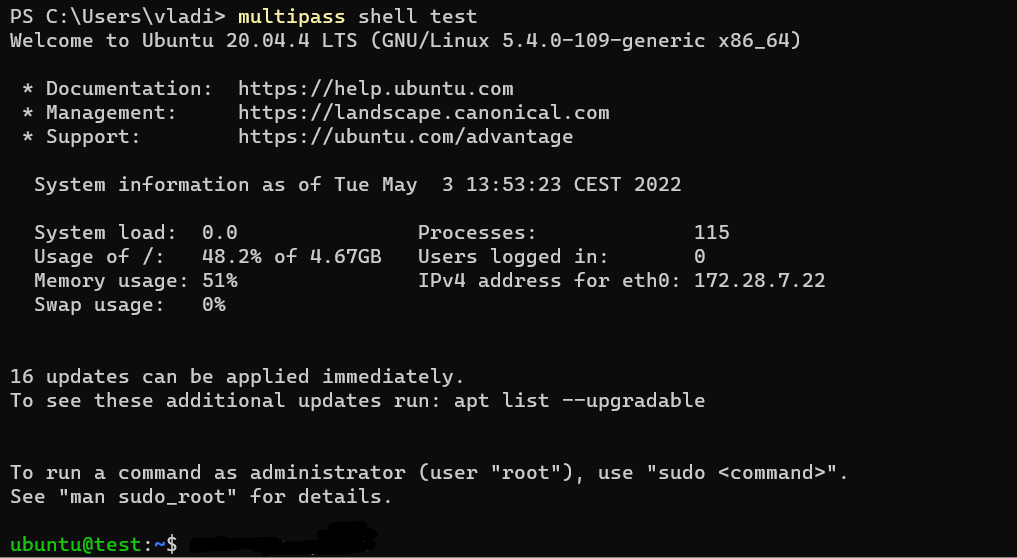


Une fois que l’installation des 3 machines virtuelles a été effectuée, nous pouvons voir l’état des instances avec la commande multipass list

**

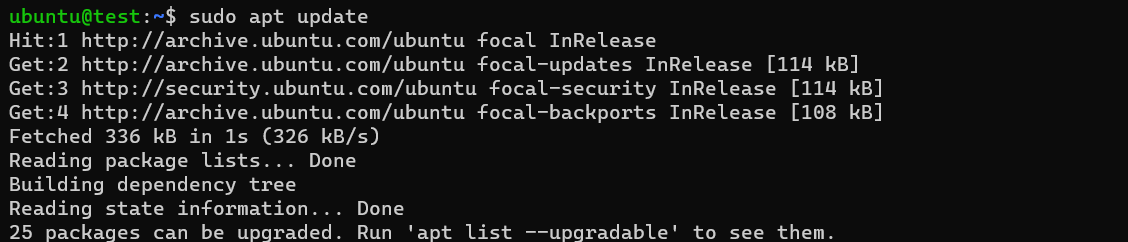
1. Configuration des instances

Afin de pouvoir installer notre environnement de travail dans une des instances, nous allons accéder à notre instance avec la commande multipass shell test

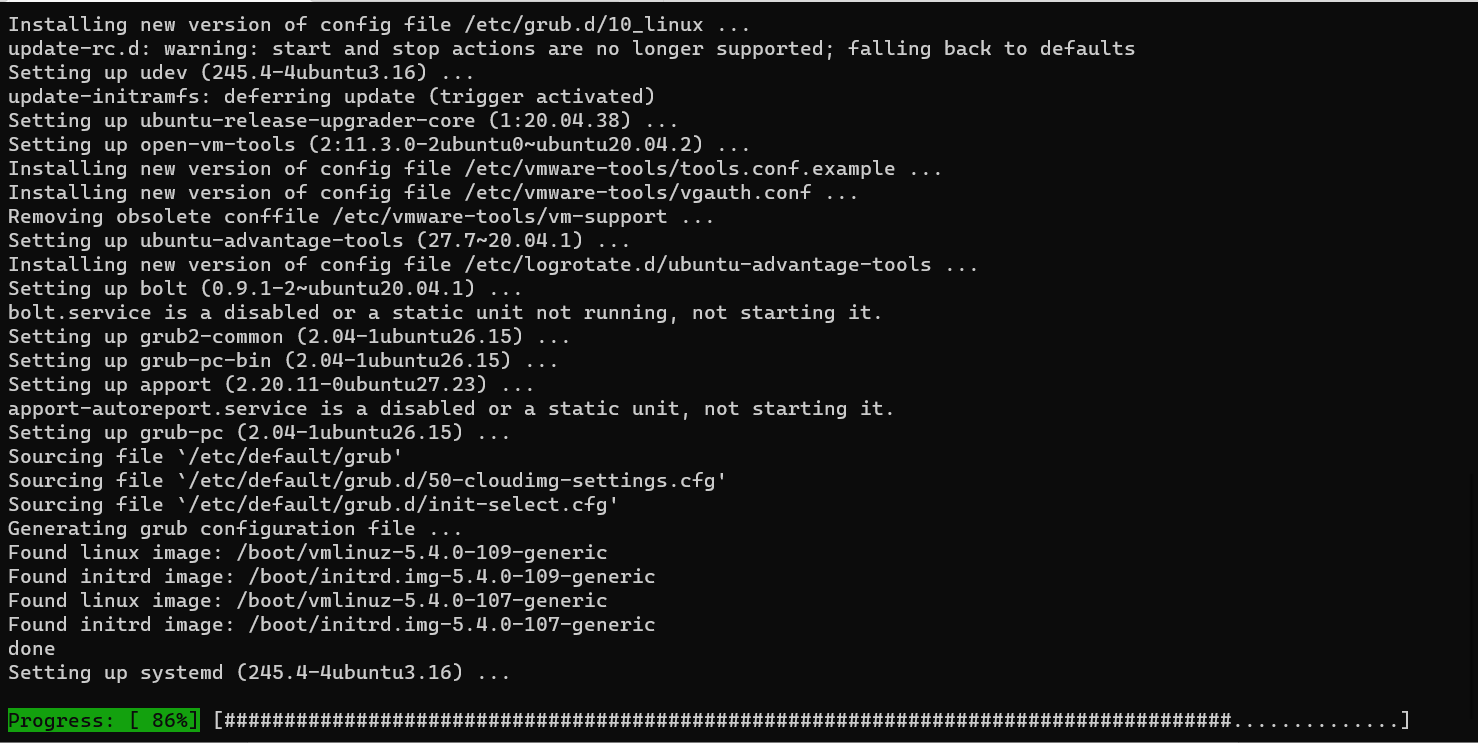
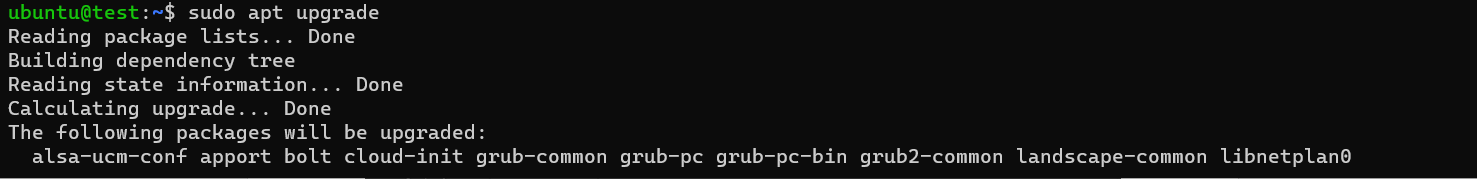


* 1. Mise à jour des librairies

Lorsque nous sommes dans notre machine virtuelle vide, nous allons mettre à jour les sources avec la commande sudo apt update

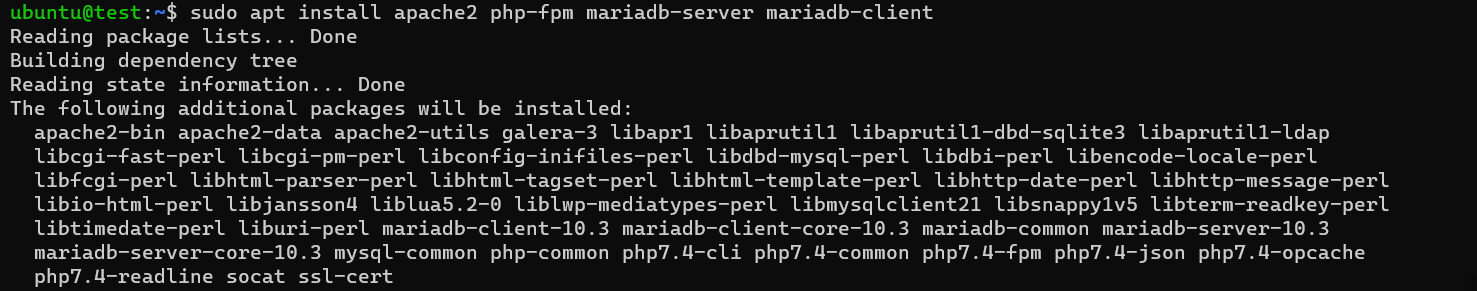


Ensuite nous allons les installer dans la VM avec la commande sudo apt upgrade

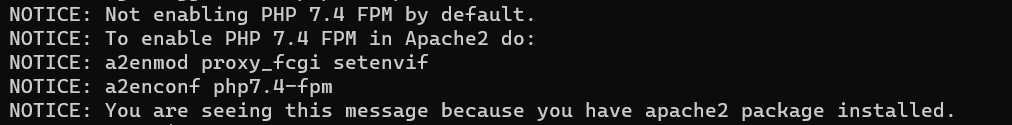


5.2 Installation et Configuration de Apache 2 – PHP – Maria DB

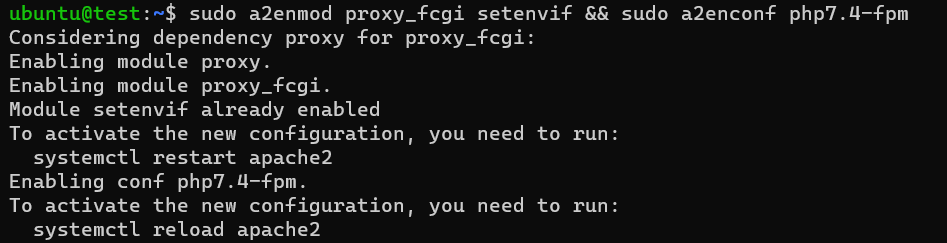
Après avoir effectué la mise à jour des sources, nous allons installer les environnements **Apache 2 – PHP – MariaDB** avec la commande sudo apt install apache2 php-fpm mariadb-server mariadb-client -y *(elle regroupe les trois installations d’un seul coup)*



Après l’installation des environnements un message s’affiche

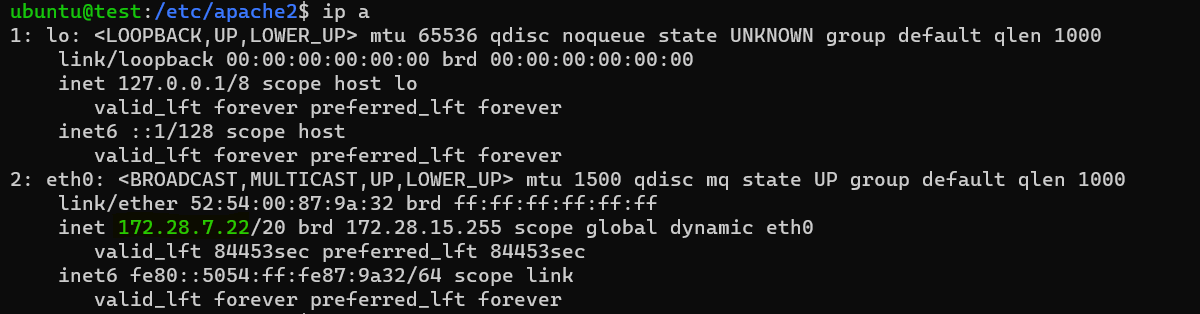


Ce message nous sert à configurer PHP dans le serveur Apache avec les commandes sudo a2enmod proxy\_fcgi setenvif && sudo a2enconf php7.4-fpm *(a2 = apache2, en = enable)*

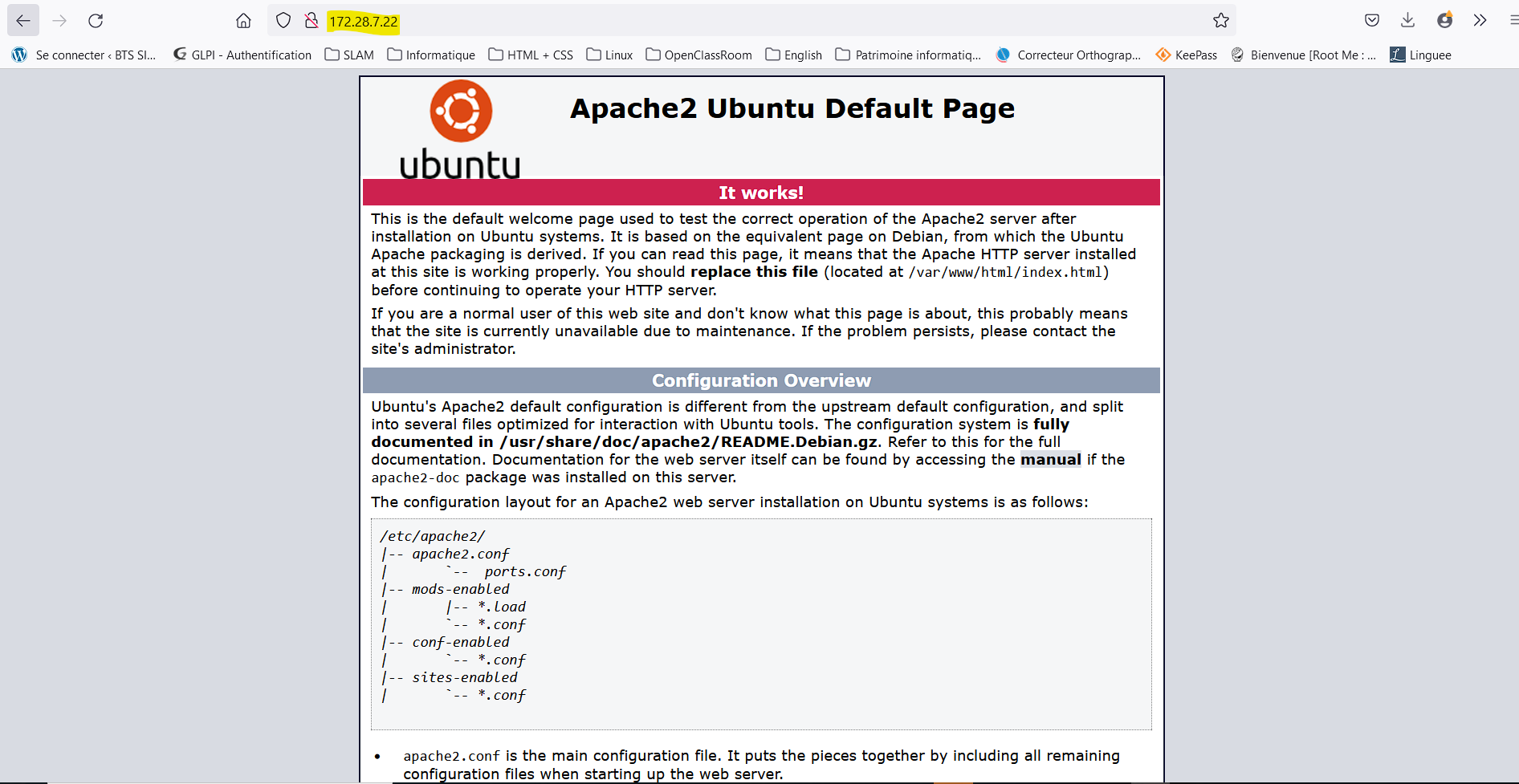


Nous allons ensuite relancer apache2 sudo systemctl restart apache2

Nous pouvons voir le statut d’Apache en allant dans l’adresse ip du serveur Apache. Nous allons donc retrouver l’adresse ip avec la commande ip a

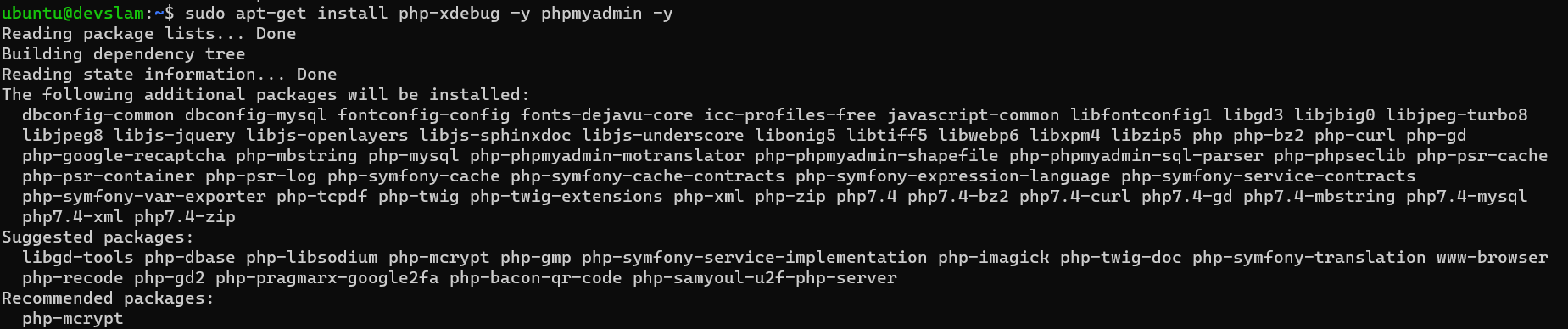


Puis nous prenons l’adresse ip et la mettant dans le navigateur



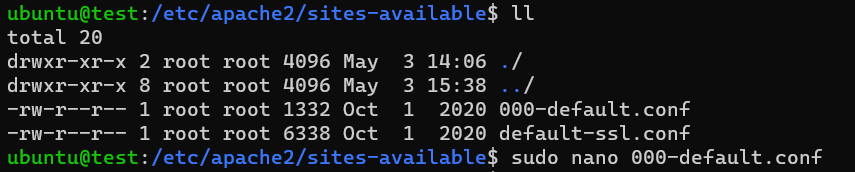
5.3 Installation et Configuration de Xdebug - PhpMyAdmin

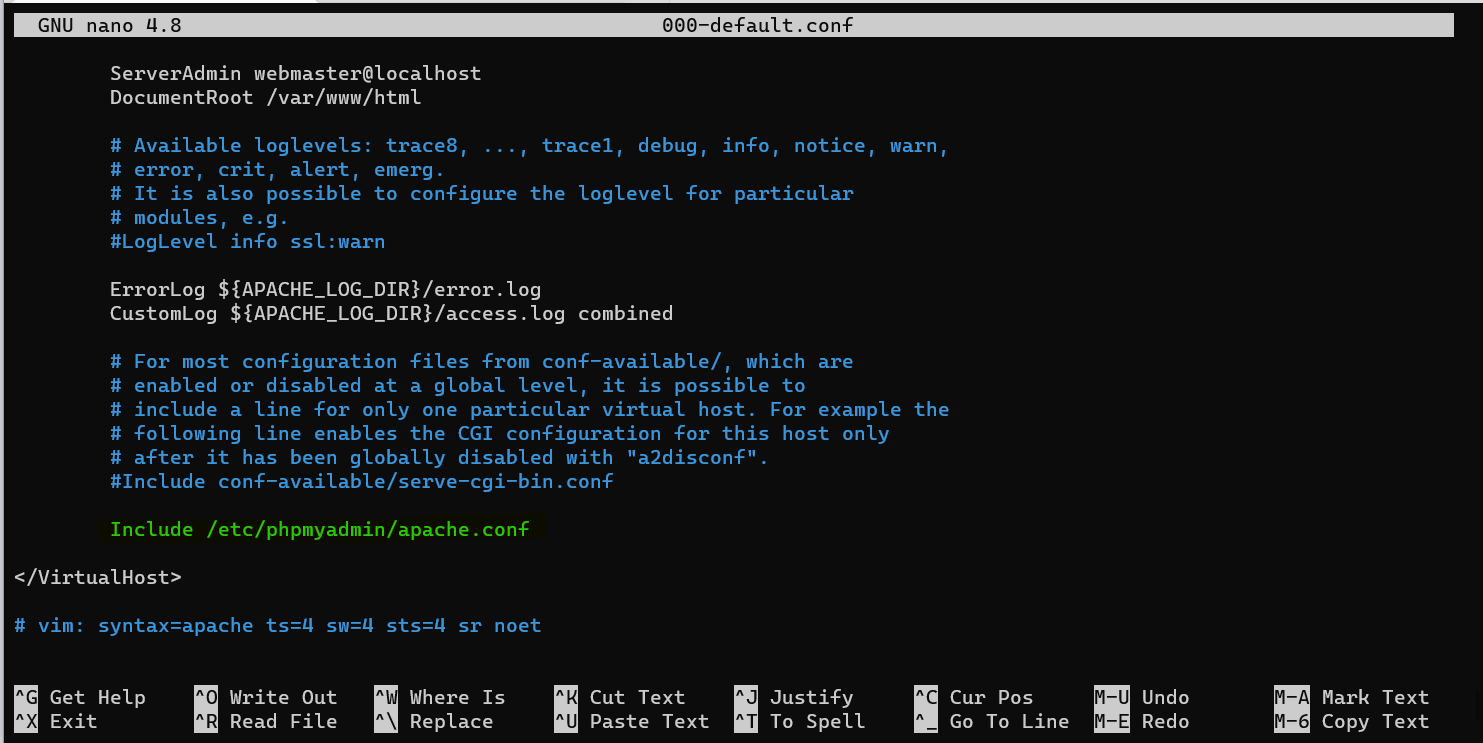
Nous allons ensuite installer **Xdebug** et **PHPMyAdmin** avec la commande sudo apt-get install php-xdebug -y phpmyadmin -y



Après l’installation des deux outils nous allons modifier la configuration Apache et rajouter le fichier **apache.conf** contenu dans **phpmyadmin**.

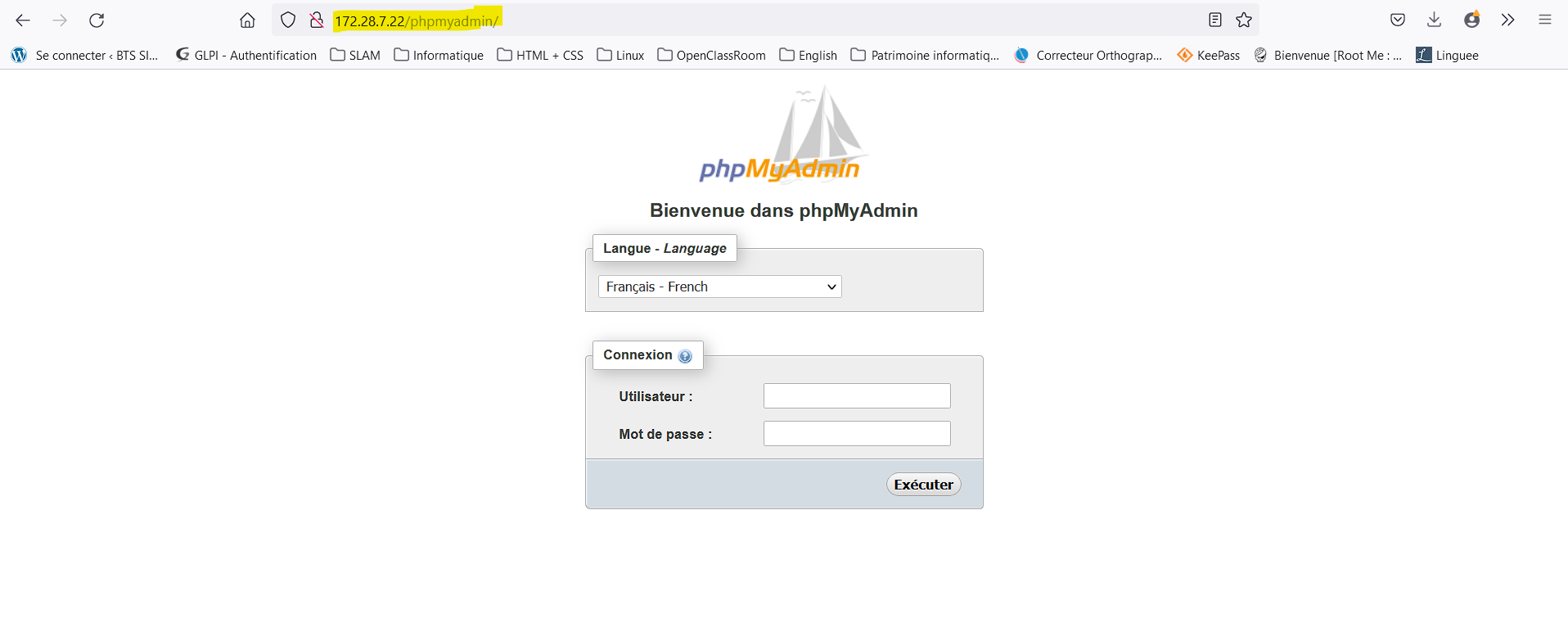
Pour ce faire nous allons nous rendre dans le dossier */etc/apache2/sites-available*. Ici nous allons modifier le fichier **000-default.conf** avec la commande Include /etc/phpmyadmin/apache.conf





Nous allons ensuite relancer Apache2 afin que la modification soit prise en compte

Pour afficher la page PHPMyAdmin dans le navigateur nous allons prendre l’adresse ip de Apache2 et mettre un */phpmyadmin* : *http://172.28.7.22/phpmyadmin/* afin d’afficher la page



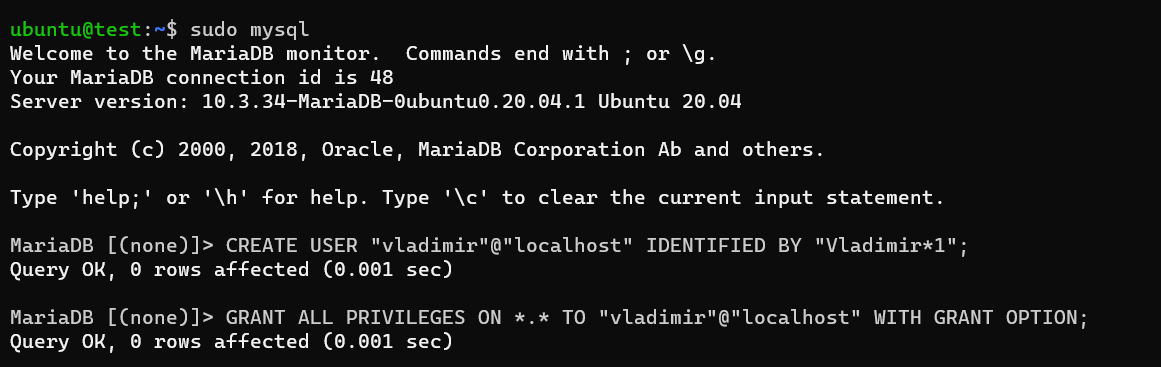
5.4 Création de l’user pour PhpMyAdmin

Afin de créer les Users nous allons modifier le SQL avec la commande sudo mysql

Dans le SQL nous allons créer l’User et lui attribuer un mot de passe ainsi que lui donner tous les droits.

CREATE USER “xxx”@”xxx” IDENTIFIED BY ”yyy”; *(creation User + MDP)*

GRANT ALL PRIVILEGES ON \*.\* TO "xxx"@"xxx" WITH GRANT OPTION; *(tous les droits à l’User créé)*



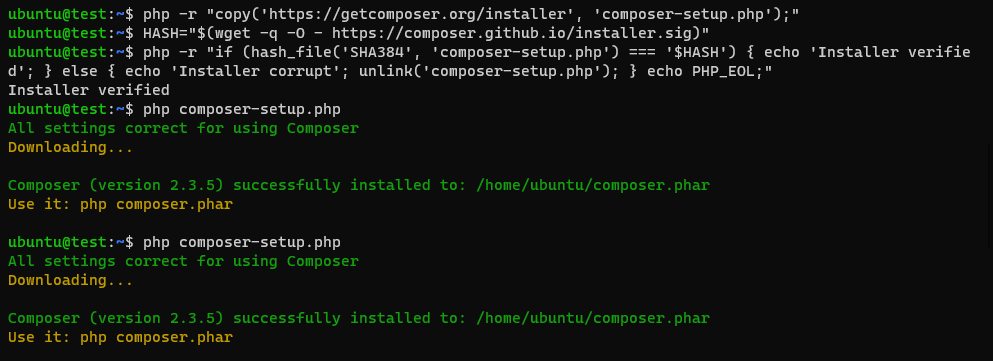
5.5 Installation composer

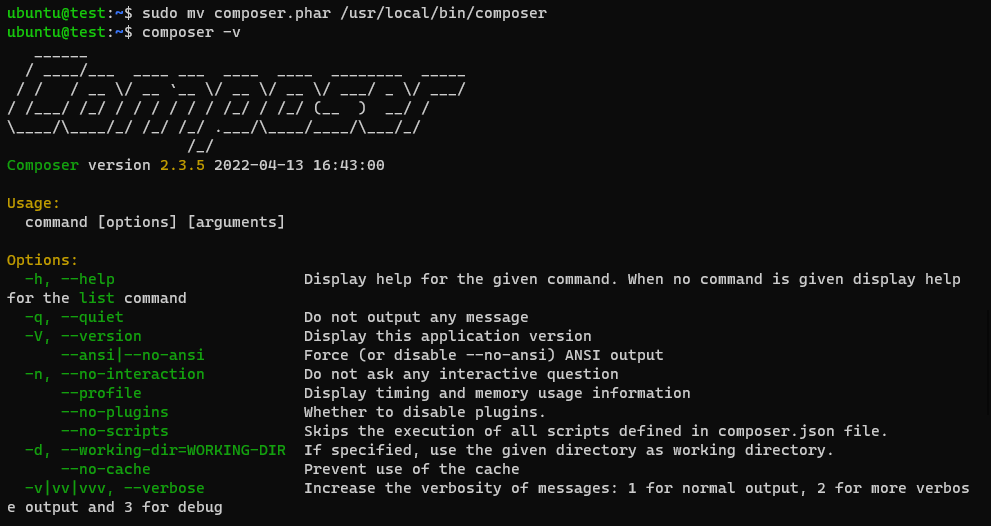
Nous allons installer composer avec les commandes :

* php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
* HASH="$(wget -q -O - https://composer.github.io/installer.sig)"
* php -r "if (hash\_file('SHA384', 'composer-setup.php') === '$HASH') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP\_EOL;"
* php composer-setup.php
* php -r "unlink('composer-setup.php');"

Ensuite nous le mettons dans le dossier */usr/local/bin/composer*

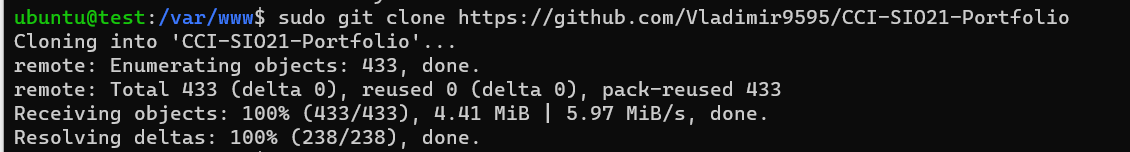
sudo mv composer.phar /usr/local/bin/composer





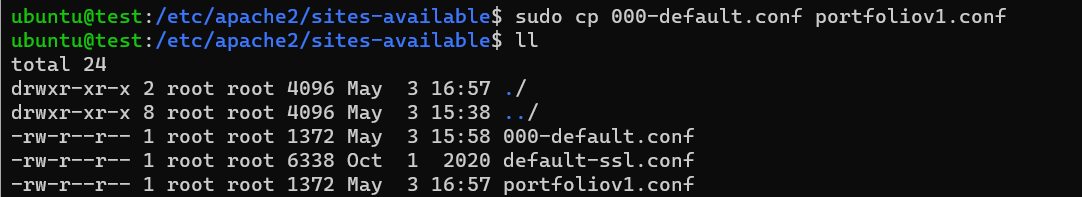
5.6 Cloner le projet git

Nous allons ensuite cloner notre projet git Portfolio dans le dossier */var/www* avec la commande sudo git clone et l’url du git



Nous allons ensuite copier le fichier **000-default.conf** présent dans le dossier **/etc/apache2/sites-available** et le renommeren **portfoliov1.conf.**

Cela avec la commandesudo cp 000-default.conf portfoliov1.conf



1. Scripts pour mise en production

Nous allons maintenant voir comment faire communiquer les instances afin de mettre à jour nos modifications au fur et à mesure du temps.

Cela sera fait à l’aide d’un script.

Je créé un fichier .sh que j’appelle up et un prod. Ici je mets les liens au projet git

<https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04-fr#etape-1-installation-de-mysql>