

CCI - SIO Atelier 2 - SACCHETTO Vladimir le 21/05/2022

Configuration de l'environnement de travail

Rédaction de documentation utilisateur



SOMMAIRE

1. INTRODUCTION.....	3
2. ENVIRONNEMENT DES INSTANCES.....	3
3. TELECHARGER MULTIPASS.....	4
4. INSTALLATION DES INSTANCES.....	4
5. CONFIGURATION DES INSTANCES.....	5
5.1 MISE A JOUR DES LIBRAIRIES.....	6
5.2 INSTALLATION ET CONFIGURATION DE APACHE 2 – PHP – MARIA DB.....	6
5.3 INSTALLATION ET CONFIGURATION DE XDEBUG - PHPMyADMIN.....	8
5.4 CREATION DE L'USER POUR PHPMyADMIN.....	9
5.5 INSTALLATION COMPOSER.....	9
5.6 MONTER LE PROJET GIT ET CONFIGURATION DU VIRTUAL HOST.....	10
5.7 AFFICHER LA PAGE WEB ET LE NOM DE DOMAINE.....	11
6. SCRIPTS POUR PASSAGE DE DEV A TEST.....	13
7. SCRIPTS POUR PASSAGE DE TEST A PROD.....	15

1. 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 :

- Machine de **DEVELOPPEMENT**
- Machine de **TEST**
- 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.

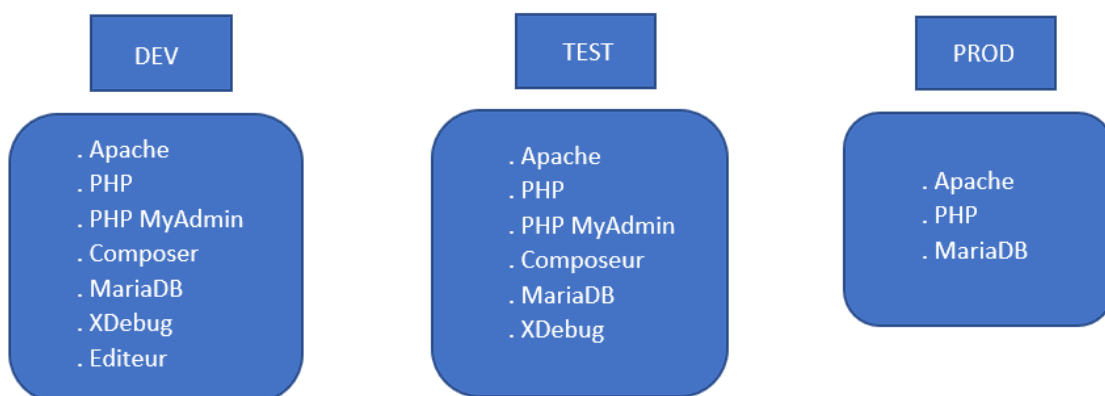
[Créer une instance](#)

[Informations sur Multipass](#)

2. 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 nécessaires et les installer soit à l'aide d'un script **bash** ou avec les commandes saisies l'une après l'autre.

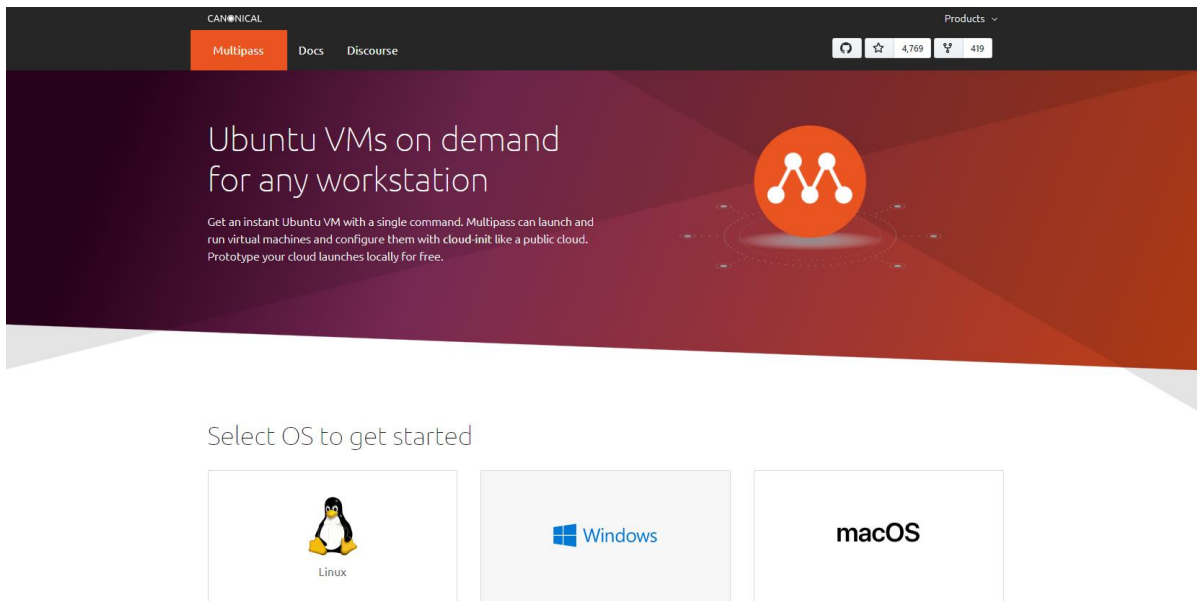
L'environnement pour chaque machine est le suivant :



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.sh** et **Synch.sh**.

3. TELECHARGER MULTIPASS

Avant tout nous avons besoin de [télécharger](#) 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**.

4. INSTALLATION DES INSTANCES

Nous allons alors ouvrir le terminal pour l'invite de commande (CMD) ou PowerShell de Windows.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Testez le nouveau système multiplateforme PowerShell https://aka.ms/pscore6

PS C:\Users\vladi>
```

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 **développeur**, **test** ou **production** comme cela a été défini initialement dans notre document.

```
PS C:\Users\vladi> multipass launch --name devslam
```

```
PS C:\Users\vladi> multipass launch --name test
```

```
PS C:\Users\vladi> multipass launch --name prod
```

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

```
PS C:\Users\vladi> multipass list
Name                State      IPv4         Image
developpeur         Stopped   --           Ubuntu 20.04 LTS
devslam             Running   172.28.5.119 Ubuntu 20.04 LTS
prod                Running   172.28.15.61 Ubuntu 20.04 LTS
pure                Stopped   --           Ubuntu 20.04 LTS
test                Running   172.28.7.22  Ubuntu 20.04 LTS
PS C:\Users\vladi>
```

Nous pouvons stopper et redémarrer nos machines à tout moment avec les commandes `multipass stop (nom instance)` et `multipass start (nom instance)`

5. 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`

```
PS C:\Users\vladi> multipass shell test
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-109-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue May  3 13:53:23 CEST 2022

System load:  0.0               Processes:    115
Usage of /:   48.2% of 4.67GB    Users logged in: 0
Memory usage: 51%              IPv4 address for eth0: 172.28.7.22
Swap usage:   0%

16 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

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

ubuntu@test:~$
```

5.1 MISE A JOUR DES LIBRAIRIES

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

```
ubuntu@test:~$ sudo apt update
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Fetched 336 kB in 1s (326 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
25 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

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

```
ubuntu@test:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  alsa-ucm-conf apport bolt cloud-init grub-common grub-pc grub-pc-bin grub2-common landscape-common libnetplan0

Installing new version of config file /etc/grub.d/10_linux ...
update-rc.d: warning: start and stop actions are no longer supported; falling back to defaults
Setting up udev (245.4-4ubuntu3.16) ...
update-initramfs: deferring update (trigger activated)
Setting up ubuntu-release-upgrader-core (1:20.04.38) ...
Setting up open-vm-tools (2:11.3.0-2ubuntu0~ubuntu20.04.2) ...
Installing new version of config file /etc/vmware-tools/tools.conf.example ...
Installing new version of config file /etc/vmware-tools/vgauth.conf ...
Removing obsolete conffile /etc/vmware-tools/vm-support ...
Setting up ubuntu-advantage-tools (27.7~20.04.1) ...
Installing new version of config file /etc/logrotate.d/ubuntu-advantage-tools ...
Setting up bolt (0.9.1-2~ubuntu20.04.1) ...
bolt.service is a disabled or a static unit not running, not starting it.
Setting up grub2-common (2.04-1ubuntu26.15) ...
Setting up grub-pc-bin (2.04-1ubuntu26.15) ...
Setting up apport (2.20.11-0ubuntu27.23) ...
apport-autoreport.service is a disabled or a static unit, not starting it.
Setting up grub-pc (2.04-1ubuntu26.15) ...
Sourcing file '/etc/default/grub'
Sourcing file '/etc/default/grub.d/50-cloudimg-settings.cfg'
Sourcing file '/etc/default/grub.d/init-select.cfg'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-5.4.0-109-generic
Found initrd image: /boot/initrd.img-5.4.0-109-generic
Found linux image: /boot/vmlinuz-5.4.0-107-generic
Found initrd image: /boot/initrd.img-5.4.0-107-generic
done
Setting up systemd (245.4-4ubuntu3.16) ...

Progress: [ 86%] [#####.....]
```

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)

```
ubuntu@test:~$ sudo apt install apache2 php-fpm mariadb-server mariadb-client
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils galera-3 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap
  libbcbi-fast-perl libbcbi-pm-perl libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libencode-locale-perl
  libfcgi-perl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl
  libio-html-perl libjansson4 liblua5.2-0 liblwp-mediatypes-perl libmysqlclient21 libsnappy1v5 libterm-readkey-perl
  libtimedate-perl liburi-perl mariadb-client-10.3 mariadb-client-core-10.3 mariadb-common mariadb-server-10.3
  mariadb-server-core-10.3 mysql-common php-common php7.4-cli php7.4-common php7.4-fpm php7.4-json php7.4-opcache
  php7.4-readline socat ssl-cert
```

Après l'installation des environnements un message s'affiche

```
NOTICE: Not enabling PHP 7.4 FPM by default.
NOTICE: To enable PHP 7.4 FPM in Apache2 do:
NOTICE: a2enmod proxy_fcgi setenvif
NOTICE: a2enconf php7.4-fpm
NOTICE: You are seeing this message because you have apache2 package installed.
```

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*)

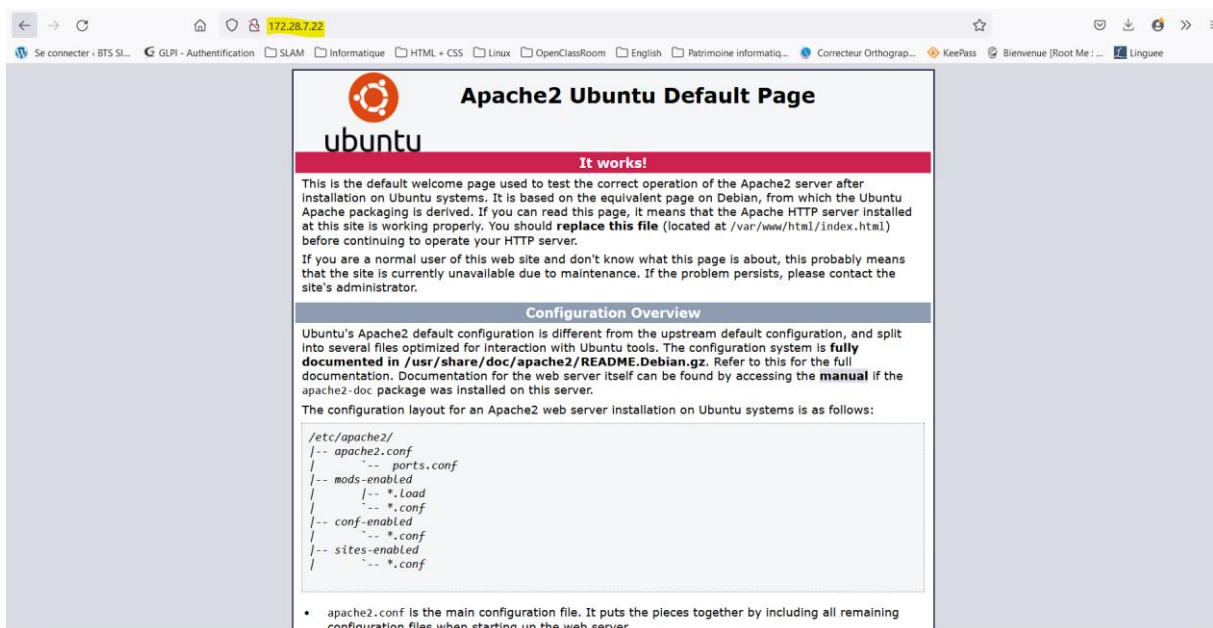
```
ubuntu@test:~$ sudo a2enmod proxy_fcgi setenvif && sudo a2enconf php7.4-fpm
Considering dependency proxy for proxy_fcgi:
Enabling module proxy.
Enabling module proxy_fcgi.
Module setenvif already enabled
To activate the new configuration, you need to run:
    systemctl restart apache2
Enabling conf php7.4-fpm.
To activate the new configuration, you need to run:
    systemctl reload apache2
```

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`

```
ubuntu@test:/etc/apache2$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 52:54:00:87:9a:32 brd ff:ff:ff:ff:ff:ff
    inet 172.28.7.22/20 brd 172.28.15.255 scope global dynamic eth0
        valid_lft 84453sec preferred_lft 84453sec
    inet6 fe80::5054:ff:fe87:9a32/64 scope link
        valid_lft forever preferred_lft forever
```

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`

```
ubuntu@devslam:~$ sudo apt-get install php-xdebug -y phpmyadmin -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  dbconfig-common dbconfig-mysql fontconfig-config fonts-dejavu-core icc-profiles-free javascript-common libfontconfig1 libgd3 libjpeg0 libjpeg-turbo8
  libjpeg8 libjs-jquery libjs-openlayers libjs-sphinxdoc libjs-underscore libonig5 libtiff5 libwebp6 libxpm4 libzip5 php php-bz2 php-curl php-gd
  php-google-recaptcha php-mbstring php-mysql php-phpmyadmin-motranslator php-phpmyadmin-shapefile php-phpmyadmin-sql-parser php-phpeclib php-psr-cache
  php-psr-container php-psr-log php-symfony-cache php-symfony-cache-contracts php-symfony-expression-language php-symfony-service-contracts
  php-symfony-var-exporter php-tcpdf php-twig php-twig-extensions php-xml php-zip php7.4 php7.4-bz2 php7.4-curl php7.4-gd php7.4-mbstring php7.4-mysql
  php7.4-xml php7.4-zip
Suggested packages:
  libgd-tools php-dbase php-libsodium php-mcrypt php-gmp php-symfony-service-implementation php-imagick php-twig-doc php-symfony-translation www-browser
  php-recode php-gd2 php-pragmarx-google2fa php-bacon-qr-code php-samyoul-u2f-php-server
Recommended packages:
  php-mcrypt
```

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

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 `sudo nano 000-default.conf`.

```
ubuntu@test:/etc/apache2/sites-available$ ll
total 20
drwxr-xr-x 2 root root 4096 May  3 14:06 ./
drwxr-xr-x 8 root root 4096 May  3 15:38 ../
-rw-r--r-- 1 root root 1332 Oct  1 2020 000-default.conf
-rw-r--r-- 1 root root 6338 Oct  1 2020 default-ssl.conf
ubuntu@test:/etc/apache2/sites-available$ sudo nano 000-default.conf
```

Ici nous allons indiquer le chemin pour mettre à jour le fichier de Virtual Host et allons écrire à l'avant dernière ligne : `Include /etc/phpmyadmin/apache.conf`

```
GNU nano 4.8                                000-default.conf

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf

Include /etc/phpmyadmin/apache.conf

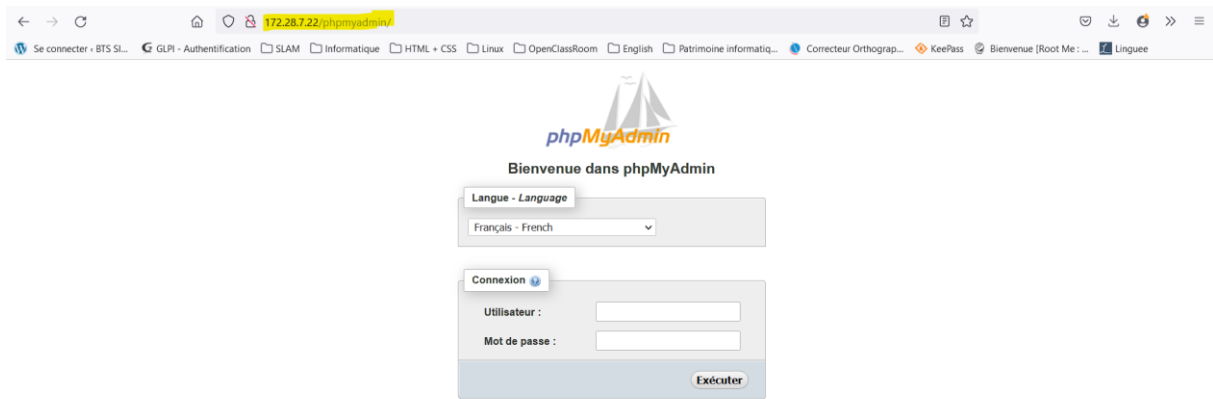
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      M-U Undo        M-A Mark Text
^X Exit          ^R Read File    ^_ Replace      ^U Paste Text   ^T To Spell     _ Go To Line    M-E Redo        M-G Copy Text
```

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 CREATION 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)*

`FLUSH PRIVILEGES;`

Ensuite nous allons relancer mysql `sudo systemctl restart mysql`

```
ubuntu@test:~$ sudo mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 48
Server version: 10.3.34-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE USER "vladimir"@"localhost" IDENTIFIED BY "Vladimir*1";
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO "vladimir"@"localhost" WITH GRANT OPTION;
Query OK, 0 rows affected (0.001 sec)
```

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`

```
ubuntu@test:~$ php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
ubuntu@test:~$ HASH="$(wget -q -O - https://composer.github.io/installer.sig)"
ubuntu@test:~$ php -r "if (hash_file('SHA384', 'composer-setup.php') === '$HASH') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
Installer verified
ubuntu@test:~$ php composer-setup.php
All settings correct for using Composer
Downloading...

Composer (version 2.3.5) successfully installed to: /home/ubuntu/composer.phar
Use it: php composer.phar
```

```
ubuntu@test:~$ sudo mv composer.phar /usr/local/bin/composer
ubuntu@test:~$ composer -v

   _____
  / ____|_ __ \  ___| |__ \
 | |___| |___| | |_) | |___|_|
 |_____|_____/|____/|_____|_|
Composer version 2.3.5 2022-04-13 16:43:00

Usage:
  command [options] [arguments]

Options:
  -h, --help                Display help for the given command. When no command is given display help
for the list command
  -q, --quiet               Do not output any message
  -V, --version             Display this application version
                          Force (or disable --no-ansi) ANSI output
  -n, --no-interaction     Do not ask any interactive question
                          Display timing and memory usage information
                          Whether to disable plugins.
                          Skips the execution of all scripts defined in composer.json file.
  -d, --working-dir=WORKING-DIR If specified, use the given directory as working directory.
                          Prevent use of the cache
  -v|vv|vvv, --verbose    Increase the verbosity of messages: 1 for normal output, 2 for more verbos
e output and 3 for debug
```

5.6 MONTER LE PROJET GIT ET CONFIGURATION DU VIRTUAL HOST

Nous allons lancer une commande dans Windows pour monter notre projet présent dans la machine physique vers la VM de **développement**. Nous allons lancer la commande `multipass mount "D:\Cours de la CCI\Atelier 1\portfolio\CCI-SIO21-Portfolio" devslam:/var/www/CCI-SIO21-Portfolio`.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Testez le nouveau système multiplateforme PowerShell https://aka.ms/pscore6

PS C:\Users\vladi> multipass mount "D:\Cours de la CCI\Atelier 1\portfolio\CCI-SIO21-Portfolio" devslam:/var/www/CCI-SIO
21-Portfolio.
```

Ainsi nous pouvons apporter des modifications aux fichiers de notre projet directement sur notre machine Windows et seront automatiquement modifiés dans la VM.

Afin d'afficher notre site nous devons pointer le dossier monté dans le Virtual Host. Nous allons copier le fichier **000-default.conf** présent dans le dossier **/etc/apache2/sites-available** et le renommer en **portfoliov1.conf**.

Cela avec la commande `sudo cp 000-default.conf portfoliov1.conf`

```
ubuntu@test:/etc/apache2/sites-available$ sudo cp 000-default.conf portfoliov1.conf
ubuntu@test:/etc/apache2/sites-available$ ll
total 24
drwxr-xr-x 2 root root 4096 May  3 16:57 ./
drwxr-xr-x 8 root root 4096 May  3 15:38 ../
-rw-r--r-- 1 root root 1372 May  3 15:58 000-default.conf
-rw-r--r-- 1 root root 6338 Oct  1 2020 default-ssl.conf
-rw-r--r-- 1 root root 1372 May  3 16:57 portfoliov1.conf
```

Dans le nouveau fichier de configuration du Virtual Host nous allons indiquer depuis quel répertoire **apache2** doit afficher la page web (**DocumentRoot /var/www/CCI-SIO21-Portfolio/**) et quel fichier prendre en compte (**DirectoryIndex index.html et/ou DirectoryIndex test.php**)

```
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
ServerName dev.vladimir-portfolio.com
ServerAdmin webmaster@localhost
DocumentRoot /var/www/CCI-SIO21-Portfolio/
DirectoryIndex index.html
DirectoryIndex test.php

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
[ Read 35 Lines ]
```

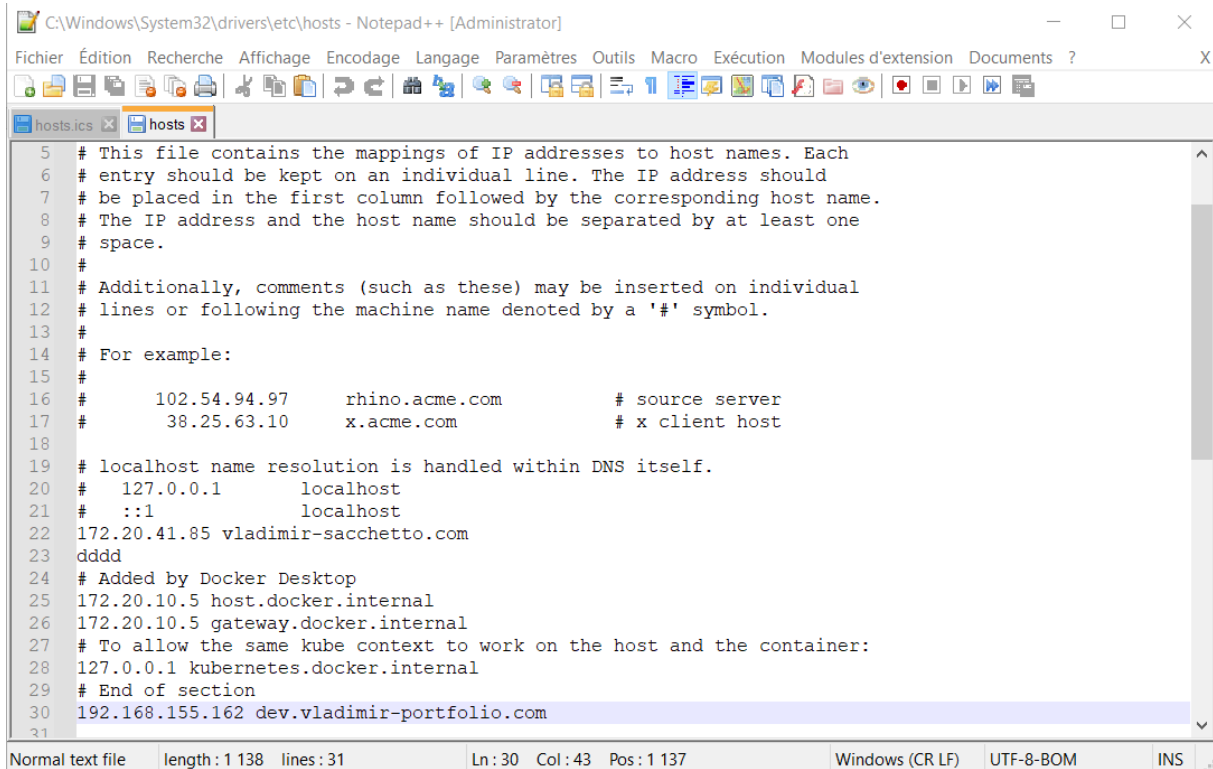
Ensuite nous allons activer le fichier créé avec la commande `sudo a2ensite portfoliov1.conf` et désactiver le fichier standard **000-default.conf** avec la commande `sudo a2dissite 000-default.conf`

5.7 AFFICHER LA PAGE WEB ET LE NOM DE DOMAINE

Nous allons modifier le fichier créé avec la commande `sudo nano portfoliov1.conf`. Dans le fichier de config nous allons indiquer le nom de domaine de notre index.html. Cela avec le **ServerName** (**dev.vladimir-portfolio.com**)

```
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
ServerName dev.vladimir-portfolio.com
ServerAdmin webmaster@localhost
DocumentRoot /var/www/CCI-SIO21-Portfolio/
DirectoryIndex index.html
DirectoryIndex test.php
```

Ensuite nous allons rajouter l'adresse ip de notre VM suivie du nom de domaine dans le fichier **hosts** qui se trouve dans `C:\Windows\System32\drivers\etc`



```

C:\Windows\System32\drivers\etc\hosts - Notepad++ [Administrator]
Fichier  Édition  Recherche  Affichage  Encodage  Langage  Paramètres  Outils  Macro  Exécution  Modules d'extension  Documents  ?
hosts.ics  hosts
5  # This file contains the mappings of IP addresses to host names. Each
6  # entry should be kept on an individual line. The IP address should
7  # be placed in the first column followed by the corresponding host name.
8  # The IP address and the host name should be separated by at least one
9  # space.
10 #
11 # Additionally, comments (such as these) may be inserted on individual
12 # lines or following the machine name denoted by a '#' symbol.
13 #
14 # For example:
15 #
16 #      102.54.94.97      rhino.acme.com      # source server
17 #      38.25.63.10      x.acme.com        # x client host
18 #
19 # localhost name resolution is handled within DNS itself.
20 #   127.0.0.1          localhost
21 #   ::1                localhost
22 172.20.41.85 vladimir-sacchetto.com
23 dddd
24 # Added by Docker Desktop
25 172.20.10.5 host.docker.internal
26 172.20.10.5 gateway.docker.internal
27 # To allow the same kube context to work on the host and the container:
28 127.0.0.1 kubernetes.docker.internal
29 # End of section
30 192.168.155.162 dev.vladimir-portfolio.com
31
Normal text file  length: 1 138  lines: 31  Ln: 30  Col: 43  Pos: 1 137  Windows (CR LF)  UTF-8-BOM  INS

```



6. SCRIPTS POUR PASSAGE DE DEV A TEST

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

Avant tout nous allons faire un clone de notre projet git à l'aide du code HTTPS dans le dossier `/var/www` de la machine de **test** avec la commande `git clone https://github.com/Vladimir9595/CCI-SIO21-Portfolio.git /var/www/CCI-SIO21-Portfolio`.

Ensuite la communication entre les deux VM se fera à l'aide d'un script que nous allons créer à la racine de la VM de **test**.

Ici nous créons le fichier **maj.sh** avec la commande `touch maj.sh`.

```
ubuntu@test:~$ touch maj.sh
```

Ce fichier va contenir des commandes qui tout d'abord vont donner les droits à l'user **ubuntu** de rechercher les dernières modifications du projet sur Github avec un git pull :

- `sudo chown -R ubuntu:ubuntu /var/www/CCI-SIO21-Portfolio`
- `git pull`

Ensuite donne les droits à **apache** de sauvegarder le contenu et l'afficher :

- `sudo chown -R www-data:www-data /var/www/CCI-SIO21-Portfolio`

Puis les droits sur le dossier `.git` à l'user **ubuntu** :

- `sudo chown -R ubuntu:ubuntu /var/www/CCI-SIO21-Portfolio/.git`

Nous allons insérer ces commandes dans le script **maj.sh** avec l'éditeur nano `sudo nano maj.sh`

```
ubuntu@test:~$ sudo nano maj.sh
```

```
GNU nano 4.8                                maj.sh
echo '-----'
echo 'Mise à jour du projet en cours'
echo '-----'

cd /var/www/CCI-SIO21-Portfolio;

echo 'Recherche des dernières modification sur Github';

sudo chown -R ubuntu:ubuntu /var/www/CCI-SIO21-Portfolio;
git pull;

echo 'Attribution des droits à Apache';

sudo chown -R www-data:www-data /var/www/CCI-SIO21-Portfolio;

sudo chown -R ubuntu:ubuntu /var/www/CCI-SIO21-Portfolio/.git;
```

Nous pouvons lancer à tout moment notre script en saisissant le nom du fichier **maj**

```

ubuntu@test:~$ maj
-----
Mise à jour du projet en cours
-----
Recherche des dernières modification sur Github
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 1), reused 4 (delta 1), pack-reused 0
Unpacking objects: 100% (4/4), 1.03 KiB | 353.00 KiB/s, done.
From https://github.com/Vladimir9595/CCI-SIO21-Portfolio
   3d2b5c5..eed687d  main      -> origin/main
Updating 3d2b5c5..eed687d
Fast-forward
Attribution des droits à Apache

```

Avant de terminer nous devons pouvoir envoyer certains éléments vers la production en excluant d'autres. Pour ce faire, dans la machine de **test**, nous pouvons copier tout ce qui est présent dans le dossier cloné (*/var/www/CCI-SIO21-Portfolio*) vers un dossier que nous allons nommer **Preprod** (*/var/www/Preprod*). Puis nous allons supprimer ce que nous ne voulons pas envoyer en production.

Nous allons insérer les commandes dans notre script *maj.sh*

```

echo 'Préparation des fichiers à transférer en production';

sudo rm -rf /var/www/Preprod;
sudo mkdir /var/www/Preprod;

sudo cp -R /var/www/CCI-SIO21-Portfolio/* /var/www/Preprod/;

sudo chown -R ubuntu:ubuntu /var/www/Preprod;

cd /var/www/Preprod;

sudo rm 'Charte graphique.md';

```

Puis nous allons vérifier la présence des éléments dans le dossier *Preprod*

```

ubuntu@test:~$ ll /var/www/Preprod/
total 44
drwxr-xr-x 5 ubuntu ubuntu 4096 May 31 09:33 ./
drwxr-xr-x 5 root root 4096 May 31 09:33 ../
drwxr-xr-x 2 ubuntu ubuntu 4096 May 31 09:33 Font/
drwxr-xr-x 2 ubuntu ubuntu 4096 May 31 09:33 Images/
-rw-r--r-- 1 ubuntu ubuntu 6344 May 31 09:33 'Mentions légales.html'
drwxr-xr-x 2 ubuntu ubuntu 4096 May 31 09:33 Style/
-rw-r--r-- 1 ubuntu ubuntu 10120 May 31 09:33 index.html
-rw-r--r-- 1 ubuntu ubuntu 30 May 31 09:33 test.php

```

7. SCRIPTS POUR PASSAGE DE TEST A PROD

Nous allons ensuite créer un script **synch.sh** qui va pouvoir synchroniser les éléments présents dans le dossier de **Preprod** de la machine de **test** (*/var/www/Preprod*) vers un dossier qui est créé dans la machine de **production** */var/www/CCI-SIO21-Portfolio*.

```
ubuntu@test:~$ touch synch.sh
```

Cette action s'effectue avec la commande **rsync**. Cette commande permet de transférer des fichiers d'une machine à l'autre ainsi faire une copie et mettre à jour les dernières modifications.

Ce script est lancé dans la machine de **test**. Nous allons donner les droits à l'utilisateur sur le dossier qui contient le clone de notre projet git :

- `sudo chown -R ubuntu:ubuntu /var/www/CCI-SIO21-Portfolio;`

Ensuite nous allons nous connecter en SSH à notre serveur de production et allons créer un répertoire dans le */var/www/* puis donnons les droits à l'utilisateur sur le dossier créé :

- `ssh (adresse ip du serveur) -l ubuntu -p 1128 'sudo mkdir -p /var/www/CCI-SIO21-Portfolio';`
- `ssh (adresse ip du serveur) -l ubuntu -p 1128 'sudo chown -R ubuntu:www-data /var/www/CCI-SIO21-Portfolio';`

Après avoir effectué ces actions nous allons utiliser notre méthode **rsync** pour transférer nos fichiers présents dans le dossier de Preprod de la machine de **test** au serveur de **production** :

- `rsync -azP -e 'ssh -p 1128' /var/www/Preprod/ ubuntu@(adresse ip du serveur):/var/www/CCI-SIO21-Portfolio;`

Nous allons enfin donner les droits à Apache sur le dossier CCI-SIO21-Portfolio :

- `ssh (adresse ip du serveur) -l ubuntu -p 1128 'sudo chown -R www-data:www-data /var/www/CCI-SIO21-Portfolio';`

```
GNU nano 4.8 synch.sh
echo "echo '-----'"
echo "echo 'Connexion au serveur et création du répertoire'"
echo "echo '-----'"

ssh ip236 -l ubuntu -p 1128 'sudo rm -rf /var/www/CCI-SIO21-Portfolio';
ssh ip236 -l ubuntu -p 1128 'sudo mkdir -p /var/www/CCI-SIO21-Portfolio';
ssh ip236 -l ubuntu -p 1128 'sudo chown -R ubuntu:www-data /var/www/CCI-SIO21-Portfolio';
echo 'Droit sur /var/www distant attribués à ubuntu';
rsync -azP -e 'ssh -p 1128' /var/www/Preprod/ ubuntu@ip236 :/var/www/CCI-SIO21-Portfolio;
echo 'Fichiers/Dossiers du projet copiés et envoyés';
ssh ip236 -l ubuntu -p 1128 'sudo chown -R www-data:www-data /var/www/CCI-SIO21-Portfolio';
echo 'Droit sur /var/www distant attribués à www-data';

echo "echo '-----'"
echo "echo ' Envoi du code en test vers le serveur de production'"
echo "echo '-----'"

echo "echo '-----'"
echo "echo 'Projet envoyé en production.'"

```


Nous allons enregistrer nos modifications puis on rend le script exécutable avec la commande `sudo chmod +x synch.sh`

```
ubuntu@test:~$ sudo chmod +x synch.sh
ubuntu@test:~$ ll
total 52
drwxr-xr-x 6 ubuntu ubuntu 4096 May 31 11:00 ./
drwxr-xr-x 3 root root 4096 May 16 09:24 ../
-rw----- 1 ubuntu ubuntu 3221 May 30 21:29 .bash_history
-rw-r--r-- 1 ubuntu ubuntu 220 Feb 25 2020 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3805 May 16 10:54 .bashrc
drwx----- 3 ubuntu ubuntu 4096 May 16 10:03 .cache/
drwxrwxr-x 3 ubuntu ubuntu 4096 May 16 10:03 .config/
-rw-rw-r-- 1 ubuntu ubuntu 49 May 30 21:44 .gitconfig
drwxrwxr-x 3 ubuntu ubuntu 4096 May 16 10:03 .local/
-rw-r--r-- 1 ubuntu ubuntu 807 Feb 25 2020 .profile
drwx----- 2 ubuntu ubuntu 4096 May 30 09:29 .ssh/
-rw-r--r-- 1 ubuntu ubuntu 0 May 16 09:29 .sudo_as_admin_successful
-rw-rw-r-- 1 ubuntu ubuntu 1116 May 31 09:33 maj.sh
-rwxrwxr-x 1 ubuntu ubuntu 1164 May 31 09:42 synch.sh*
```

Après avoir exécuté notre script (`sudo ./synch.sh`) nous allons nous connecter à notre machine de production et vérifions si les informations ont bien été prises en compte.

```
ubuntu@VM-SLAM21-SV:~$ ll /var/www/CCI-SIO21-Portfolio/
total 44
drwxr-xr-x 5 ubuntu ubuntu 4096 May 31 07:33 ./
drwxr-xr-x 4 root root 4096 May 31 08:30 ../
drwxr-xr-x 2 ubuntu ubuntu 4096 May 31 07:33 Font/
drwxr-xr-x 2 ubuntu ubuntu 4096 May 31 07:33 Images/
-rw-r--r-- 1 ubuntu ubuntu 6344 May 31 07:33 'Mentions l'$'\303\251'gales.html'
drwxr-xr-x 2 ubuntu ubuntu 4096 May 31 07:33 Style/
-rw-r--r-- 1 ubuntu ubuntu 10120 May 31 07:33 index.html
-rw-r--r-- 1 ubuntu ubuntu 30 May 31 07:33 test.php
```

Nous allons modifier le fichier de Virtual Host (comme indiqué aux pages 10 et 11) et allons activer le fichier .conf qui pointe notre projet.

```
GNU nano 4.8 /etc/apache2/sites-available/portfoliov1.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/CCI-SIO21-Portfolio/
    DirectoryIndex index.html
    DirectoryIndex test.php

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```



```

ubuntu@VM-SLAM21-SV:~$ sudo a2ensite portfoliov1.conf
[sudo] password for ubuntu:
Enabling site portfoliov1.
To activate the new configuration, you need to run:
    systemctl reload apache2
ubuntu@VM-SLAM21-SV:~$ sudo a2dissite 000-default.conf
Site 000-default disabled.
To activate the new configuration, you need to run:
    systemctl reload apache2
ubuntu@VM-SLAM21-SV:~$ sudo systemctl restart apache2

```

```

ubuntu@VM-SLAM21-SV:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0@if297: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 02:00:00:3a:4e:7f brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 178.33.64.236/29 brd 178.33.64.239 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::ff:fe3a:4e7f/64 scope link
        valid_lft forever preferred_lft forever

```

