Installation Laravel sous Debian 13

Qu’est ce que Laravel ?

Laravel est un framework PHP utilisé pour développer des sites et des applications web.
Un framework est une aide, une sorte de boîte à outils, pour développer plus facilement et proprement.

Il fournit une structure et des outils prêts à l’emploi pour faciliter le développement : base de données, authentification, templates, etc.

Son objectif est de simplifier et d’organiser le code, afin de développer des applications web plus rapidement et de manière plus propre et organisée.

Indirectement cela permet d’avoir un site plus performant sur le long terme.

Début

Par simplicité, j'étais en utilisateur root pour faire toute la documentation.

Cela fait que je n'ai pas eu besoin de rentrer le mot *sudo* avant chaque commande.

vous pouvez aussi être connecté en root, étant le 1er service à être installé, mais quand le service sera en place, il ne sera pas recommandé de se connecter avec ce compte ci

Avant de commencer, rentrez la commande :

```
apt update
```

C'est pour synchroniser la table de liste des paquets entre celui de votre machine et du serveur qui les contient.

Petite astuce : appuyer sur la touche Tabulation (Tab ; celui en haut a gauche du clavier avec deux flèches) pour compléter des parties de commandes ou de chemins de fichiers.

Installer Laravel sur la machine

Avant d'installer Laravel, il faut installer d'autres logiciels / paquets sur la machine, pour que Laravel puisse fonctionner, et par la suite le site.

On commence par faire :

```
sudo apt install curl
```

```
:~# apt install curl
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  linux-image-6.12.57+deb13-amd64
Veuillez utiliser « apt autoremove » pour le supprimer.

Installation de :
  curl

Installation de dépendances :
  libcurl4t64     libldap2      librtmp1    libsasl2-modules     libssh2-1t64
  libldap-common  libnghttp3-9  libsasl2-2  libsasl2-modules-db

Paquets suggérés :
  libsasl2-modules-gssapi-mit        libsasl2-modules-ldap  libsasl2-modules-sql
  | libsasl2-modules-gssapi-heimdal  libsasl2-modules-otp

Sommaire :
  Mise à niveau de : 0. Installation de : 10Supprimé : 0. Non mis à jour : 0
Taille du téléchargement : 1 404 kB
  Espace nécessaire : 3 554 kB / 17,3 GB disponible

Continuer ? [O/n]
```

Puis on appuie sur la lettre O et entrée

C’est un logiciel utilisé pour simplifier l’installation de certains services

Et ensuite on rentre la commande :

```
/bin/bash -c "$(curl -fsSL https://php.new/install/linux/8.4)"
```

Il va afficher un écran noir, puis cet écran :

C’est un bon début !

On rentre ensuite la commande :

```
reboot
```

Cela va redémarrer la machine

En se reconnectant, écrivez la commande :

```
laravel
```

Cela va permettre de voir si ça marche
Si vous avez cet écran :

```
            :~# laravel
Laravel Installer 5.24.7

Usage:
  command [options] [arguments]

Options:
  -h, --help            Display help for the given command. When no command is given display help for the list command
      --silent          Do not output any message
  -q, --quiet           Only errors are displayed. All other output is suppressed
  -V, --version         Display this application version
      --ansi|--no-ansi  Force (or disable --no-ansi) ANSI output
  -n, --no-interaction  Do not ask any interactive question
  -v|vv|vvv, --verbose  Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for deb
ug

Available commands:
  completion  Dump the shell completion script
  help        Display help for a command
  list        List commands
  new         Create a new Laravel application
```

C’est que c’est bon !

Si il y a des erreurs en rouge, soit vous n’êtes pas root, soit il faut rentrer cette commande :

```
composer global require laravel/installer
```

Installer les dépendances externes

PHP

C’est lui qui va permettre de lire certains fichiers du site.
Il y a PHP à installer mais aussi des dépendances, pour que tout le monde puisse communiquer avec lui, et que lui puisse répondre à tout les besoins des autres services :

```
apt install php8.4-mbstring php8.4-xml php8.4-mysql php8.4-curl
php8.4-zip php8.4-bcmath php8.4-gd libapache2-mod-php8.4
```

```
root@debian13:/var/www/siteWEB# apt install php8.4-mbstring php8.4-xml php8.4-mysql php8.4-curl php8.4
-zip php8.4-bcmath php8.4-gd
Installation de :
  php8.4-bcmath  php8.4-curl  php8.4-gd  php8.4-mbstring  php8.4-mysql  php8.4-xml  php8.4-zip

Installation de dépendances :
  fontconfig-config  libdeflate0        libheif-plugin-aomenc    liblerc4       libxpm4
  fonts-dejavu-core  libfontconfig1     libheif-plugin-dav1d     libonig5       libxslt1.1
  fonts-dejavu-mono  libgav1-1          libheif-plugin-libde265  librav1e0.7    libyuv0
  libabsl20240722    libgcrypt20        libheif-plugin-x265      libsharpyuv0   libzip5
  libaom3            libgd3             libheif1                 libsvtav1enc2  php-common
  libavif16          libgomp1           libimagequant0           libtiff6       php8.4-common
  libdav1d7          libgpg-error-l10n  libjbig0                 libwebp7
  libde265-0         libgpg-error0      libjpeg62-turbo          libx265-215

Paquets suggérés :
  rng-tools                libheif-plugin-jpegdec  libheif-plugin-j2kenc   libheif-plugin-svtenc
  libgd-tools              libheif-plugin-jpegenc  libheif-plugin-kvazaar
  libheif-plugin-ffmpegdec libheif-plugin-j2kdec   libheif-plugin-rav1e

Sommaire :
  Mise à niveau de : 0. Installation de : 45Supprimé : 0. Non mis à jour : 0
Taille du téléchargement : 16,2 MB
  Espace nécessaire : 71,5 MB / 16,8 GB disponible

Continuer ? [O/n]
```

On écrit la lettre O, puis entrée.

```
Préparation du dépaquetage de .../41-php8.4-mbstring_8.4.16-1~deb13u1_amd64.deb ...
Dépaquetage de php8.4-mbstring (8.4.16-1~deb13u1) ...
Sélection du paquet php8.4-mysql précédemment désélectionné.
Préparation du dépaquetage de .../42-php8.4-mysql_8.4.16-1~deb13u1_amd64.deb ...
Dépaquetage de php8.4-mysql (8.4.16-1~deb13u1) ...
Sélection du paquet php8.4-xml précédemment désélectionné.
Préparation du dépaquetage de .../43-php8.4-xml_8.4.16-1~deb13u1_amd64.deb ...
Dépaquetage de php8.4-xml (8.4.16-1~deb13u1) ...
Sélection du paquet php8.4-zip précédemment désélectionné.
Préparation du dépaquetage de .../44-php8.4-zip_8.4.16-1~deb13u1_amd64.deb ...
Dépaquetage de php8.4-zip (8.4.16-1~deb13u1) ...
Paramétrage de libsharpyuv0:amd64 (1.5.0-0.1) ...
Paramétrage de php-common (2:96) ...
Created symlink '/etc/systemd/system/timers.target.wants/phpsessionclean.timer' → '/usr/lib/systemd/
stem/phpsessionclean.timer'.
Paramétrage de libaom3:amd64 (3.12.1-1) ...
Paramétrage de liblerc4:amd64 (4.0.0+ds-5) ...
Paramétrage de libxpm4:amd64 (1:3.5.17-1+b3) ...
Paramétrage de libgpg-error0:amd64 (1.51-4) ...
Paramétrage de libzip5:amd64 (1.11.3-2) ...
Paramétrage de php8.4-common (8.4.16-1~deb13u1) ...
Creating config file /etc/php/8.4/mods-available/calendar.ini with new version

Progression : [ 58%] [                                                              ]
```

Il va écrire beaucoup de lignes, puis PHP sera installé.

Apache2

Laravel permet de faire tourner le site. Mais pas de le diffuser sur internet.

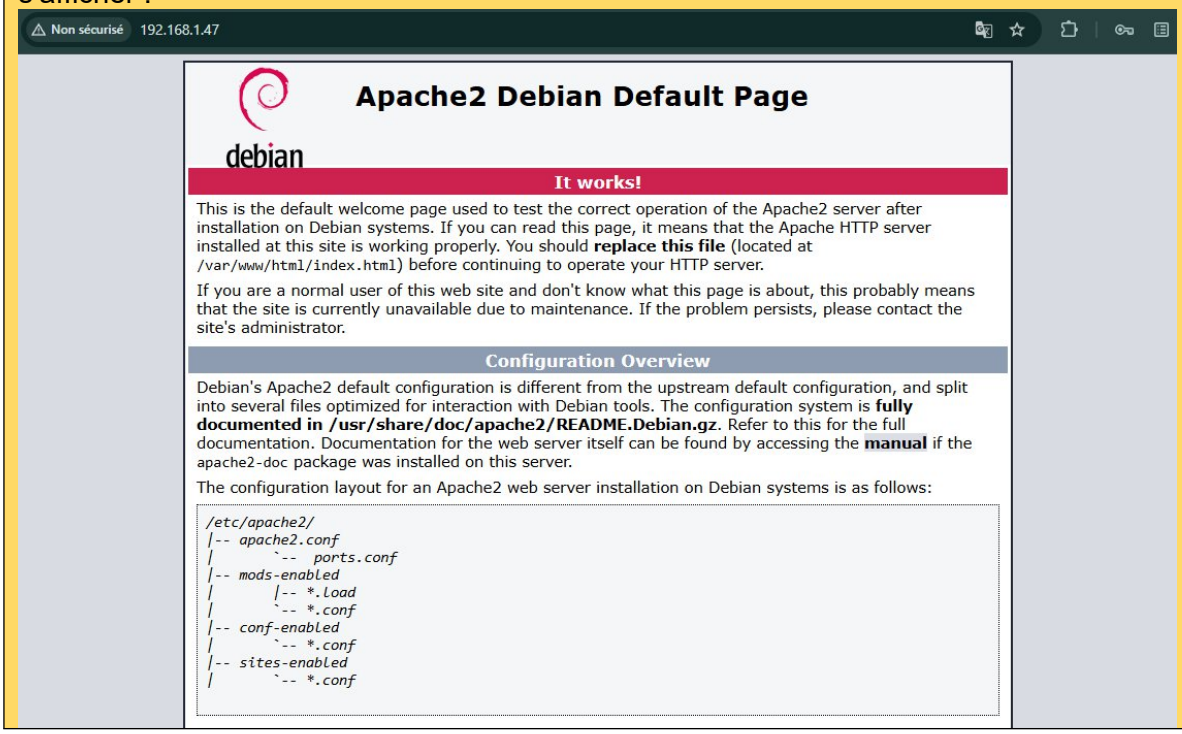
Pour cela il faut installer Apache2, avec la commande :

```
apt install apache2
```

```
            :~# apt install apache2
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  linux-image-6.12.57+deb13-amd64
Veuillez utiliser « apt autoremove » pour le supprimer.

Installation de :
  apache2

Installation de dépendances :
  apache2-bin   apache2-utils  libaprutil1-dbd-sqlite3  libaprutil1t64  ssl-cert
  apache2-data  libapr1t64     libaprutil1-ldap         liblua5.4-0

Paquets suggérés :
  apache2-doc  apache2-suexec-pristine | apache2-suexec-custom  ufw  www-browser

Sommaire :
  Mise à niveau de : 0. Installation de : 10Supprimé : 0. Non mis à jour : 0
Taille du téléchargement : 2 394 kB
  Espace nécessaire : 8 566 kB / 17,2 GB disponible
```

On appuie sur la lettre O , puis entrée

Si vous êtes un peu curieux,et que vous avez un navigateur internet (Chrome, mozilla...), vous pouvez voir la réussite de l'installation de apache2.
Pour cela, il faut rentrer l'ip de la machine sur notre navigateur, et vous aurez cette page qui va s'afficher :



C’est la page WEB par défaut de Apache2, donc ça marche !
Il faut que l’ordi où vous avez le navigateur soit sur le même réseau que la machine, sinon ça ne marchera pas.

et on rentre les commandes :

```
a2enmod php8.4
a2enmod rewrite
```

Pour que Apache2 soit capable de gérer le PHP, on active le module qui va permettre de faire la liaison entre Apache2 et PHP.

```
root@debian13:/var/www/siteWEB# a2enmod php8.4
Considering dependency mpm_prefork for php8.4:
Considering conflict mpm_event for mpm_prefork:
Considering conflict mpm_worker for mpm_prefork:
Module mpm_prefork already enabled
Considering conflict php5 for php8.4:
Module php8.4 already enabled
```

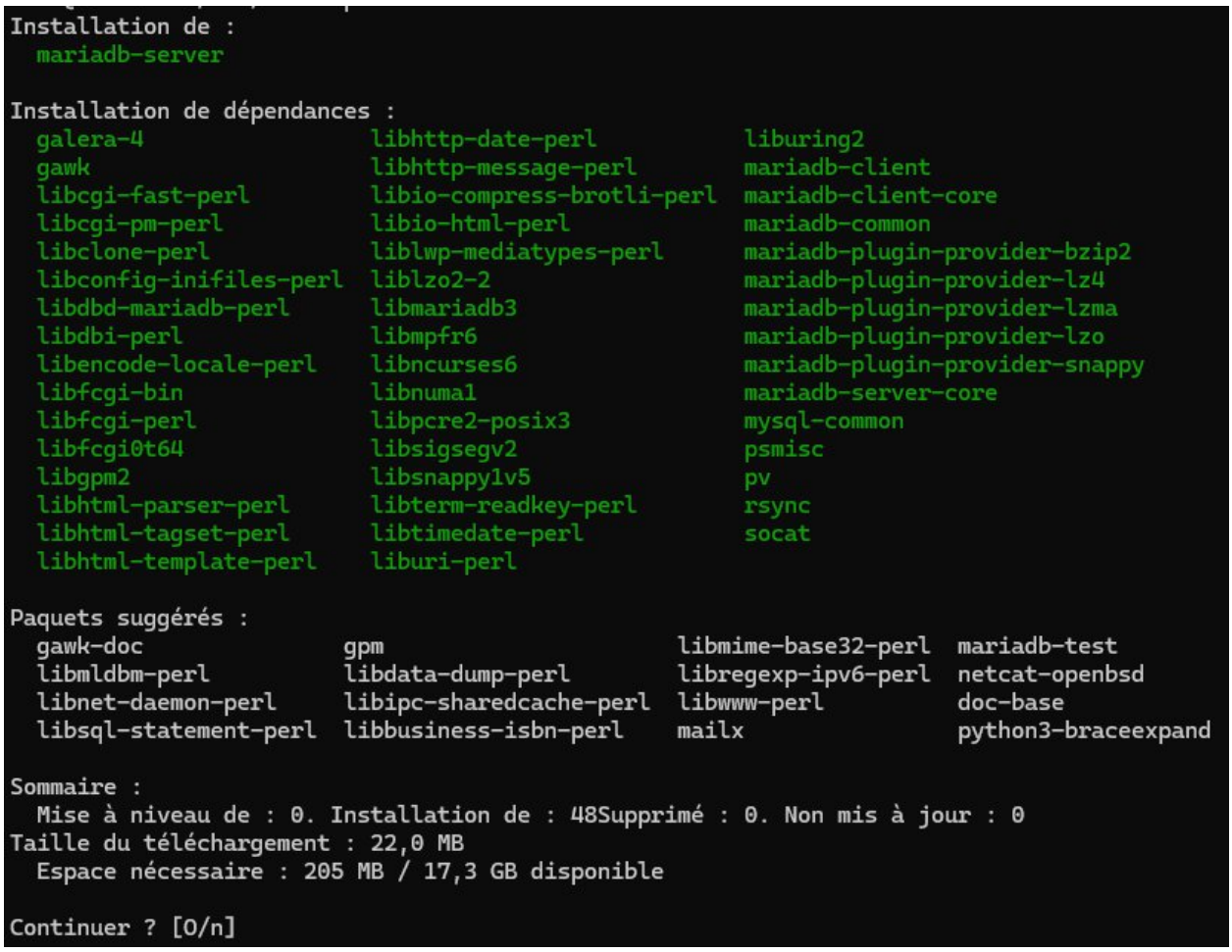
Il nous dit « Module php8.4 already enabled », c’est que le module est bien activé.

MariaDB

C’est un logiciel pour gérer les bases de données.
Très pratique car plus rapide que de mettre dans un simple fichier, et plus facile à automatiser, pour qu’il soit utilisable par d’autres logiciels et même par plusieurs utilisateurs en même temps.
(ce qui est difficile avec un simple fichier)

Pour cela on fait un :

```
Apt install mariadb-server
```



On rentre la lettre O, puis entrée.

Ensuite on va créer une database, et un compte pour la database.
Cela va permettre de donner les identifiants à une application *(notre laravel par exemple)* pour qu'elle se débrouille toute seule, avec son propre compte.

Pour cela, nous allons rentrer dans **MariaDB**.
Il suffit d'écrire :

```
mariadb
```

Et nous voilà dans le **prompt MariaDB**, où seul le langage SQL est compris.
Nous allons donc rentrer quelques commandes, puis ensuite nous pourrons fuir de ce monde, et retourner sur le terminal que nous connaissons...

```
CREATE DATABASE database;
```

database est le nom que je donne à ma base de données.
Vous pouvez l'appeler comme vous voulez

***Retenez** le nom on en aura besoin plus tard*.

```
CREATE USER 'userWEB'@'localhost' IDENTIFIED BY 'motdepasse';
```

Les apostrophes sont obligatoires.

- userWEB : mon nom d'utilisateur, mais vous pouvez mettre ce que vous voulez
Retenez le, on en aura besoin plus tard

- localhost : C'est une sécurité. Cela veut dire que le compte ne pourra se connecter seulement qu'en localhost. Cela veut dire seulement sur la machine, et pas ailleurs.
On peut mettre une IP, car il arrive que le service de base de données et le site ne soit pas sur le même machine et donc pas la même adresse IP.

- motdepasse : c'est le mot de passe de l'utilisateur userWEB.
Retenez le mot de passe, on en aura besoin plus tard.

```
GRANT ALL PRIVILEGES ON database.* TO 'userWEB'@'localhost';
```

On donne toutes les permissions de la base de données *database* ; créer précédemment ; à *userWEB*. *On doit préciser le @'localhost'*
*l'étoile * après le database. signifie qu'on donne un accès à tout le contenu de cette database et non pas qu'à une partie.*

```
FLUSH PRIVILEGES;
```

Met à jour les autorisations d'utilisateurs. Au lieu de redémarrer toute la machine, on rentre cette commande.

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at https://github
.com/MariaDB/Server

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 DATABASE database;
Query OK, 1 row affected (0,001 sec)

MariaDB [(none)]> CREATE USER 'userWEB'@'localhost' IDENTIFIED BY 'motdepasse';
Query OK, 0 rows affected (0,005 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON database.* TO 'userWEB'@'localhost';
Query OK, 0 rows affected (0,005 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]>
```

Et nous pouvons partir de MariaDB avec la commande :

```
exit
```

Si il vous dit : « *Bye* » c'est bon, vous êtes de retour dans le prompt.

Importer une base de donnée

Cette partie n'est utile que si vous avez un fichier .sql qui contient une base de données. Sinon vous pouvez aller à l'étape suivante

Pour cela, nous allons rentrer une commande :

```
mysql -u userWEB -p database < fichier.sql
```

- -u userWEB : –u permet de préciser un utilisateur *(qui peut être donc autre que celui qu'on utilise actuellement)*
  userWEB pour l'utilisateur qu'on a créé pour la base de données un peu plus haut

- -p : c'est pour qu'il nous demande le mot de passe de l'utilisateur dès qu'on appuiera sur entrée

- Database : le nom de la base de données qui va recevoir les données du fichier.

- < fichier.sql : le nom de mon fichier qui contient la base de données à importer.
  Je n'ai pas mis le chemin complet du fichier puisqu'il est à l'endroit où je me situe
  Quand je rentre la commande ls, il apparaît directement et n'est pas dans un dossier ailleurs, s*inon vous devez mettre son chemin complet, par exemple : /var/www/siteWEB/fichier.sql*

```
root@debian13:~# mysql -u userWEB -p database < fichier.sql
Enter password:
root@debian13:~#
```

Il vous demande d'écrire le mot de passe. *(celui que nous avons choisi lorsque nous étions dans le prompt mariaDB)*

Si il ne dit rien, c'est qu'il a réussi l'opération

Si il dit : **ERROR 1045 (28000): Access denied for user 'userWEB'@'localhost' (using password: YES)**
C'est que vous vous êtes soit :
- Trompé dans le mot de passe
- Trompé sur le nom de la base de données
- Trompé sur le nom d'utilisateur
- Oublié une des commandes de privilèges à la création de la base de données, plus haut

Importer le site sur la machine

Si l'envie vous prend de coder du html, vous pouvez juste modifier le fichier */var/www/html/index.html* à votre envie pour modifier la page par défaut en ce que vous voulez.
Mais vous n'allez pas avoir un très bon résultat, n'étant que du html, sans css ou javascript.

La plupart du temps, les fichiers pour un site sont sur GitHub, une plateforme de partage de code. Plus simple quand plusieurs développeurs travaillent sur un même projet.
Vous pouvez aussi avoir stocké ces fichiers de site ailleurs.

Dans tout les cas, nous allons partir du principe que tout les fichiers du site sont sous un **.zip** *(pas un autre format)*.
On peut le faire soit par internet, soit par support de stockage externe (clé USB, ou disque dur externe).

Dans les deux cas nous allons installer un autre paquet :

```
apt install unzip
```

Nativement Linux ne sait pas ouvrir les .zip. Ce paquet, va l'aider.

Mettre les fichiers du site par internet

Imaginons que le zip de votre site soit sur ce lien URL :
https://codeload.github.com/lien/versun/zip
c'est un faux lien, pour contextualiser. Vous mettrez votre lien à la place de celui ci.

Nous allons d'abord nous déplacer dans l'arborescence de notre serveur, vers le dossier où le site va être placé. Pour cela on rentre la commande :

```
cd /var/www/
```

Ensuite nous rentrons la commande :

```
wget https://codeload.github.com/lien/versun/zip
```

```
--2026-03-12 09:50:23--  https://codeload.github.com/
Résolution de codeload.github.com (codeload.github.com)… 64:ff9b::8c52:7909, 140.82.121.10
Connexion à codeload.github.com (codeload.github.com)|64:ff9b::8c52:7909|:443… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : non indiqué [application/zip]
Sauvegarde en : « main »

main                        [     <=>                                ]   1,53M  1,39MB/s    ds 1,1s

2026-03-12 09:50:25 (1,39 MB/s) - « main » sauvegardé [1606871]
```

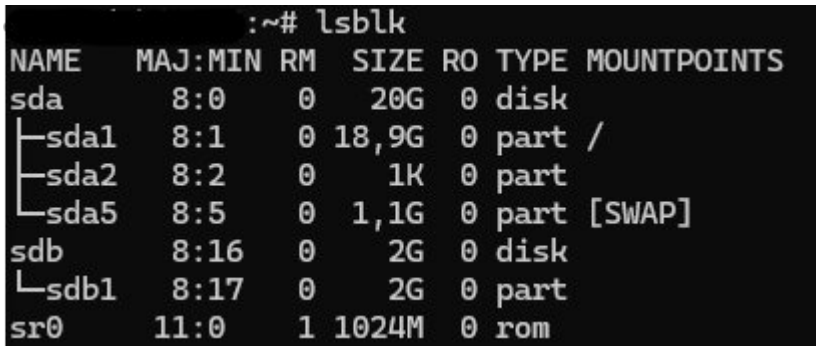
Et le zip de votre site est téléchargé !

Mettre les fichiers du site par stockage externe

Si les fichiers de votre site sont sur un stockage externe comme une clé USB, un disque dur externe, la méthode est la suivante :

Branchez votre stockage externe à la machine, puis faites un :

```
lsblk
```



Dans mon cas ma clé USB se nomme sdb1.
Il peut avoir des variations de noms en fonction du type de stockage ou si celui que vous branchez a plusieurs partitions.
En tout cas, si vous avez une machine avec un disque dur interne, et que vous branchez une clé USB, il y a toute les chances que ça soit comme sur l'image.

Vous pouvez le faire machine allumé. Mais si elle n'apparaît pas, faites un :
reboot
(avec le stockage externe qui reste branché)
Cela va redémarrer la machine, et le stockage va apparaître. Sinon c'est que le stockage n'est pas compatible.

Pour accéder aux données dans ce stockage externe il faut le monter dans un **dossier**. Pour cela nous allons créer un **dossier**.

```
mkdir /mnt/usb
```

Il faut le faire dans le dossier **/mnt** de préférence
On peut donner le nom qu'on veut, autre que usb.

Ensuite on peut monter le stockage dans ce dossier :

```
mount /dev/sdb1 /mnt/usb
```

- /dev/sdb1 : /dev pour « device » est le dossier qui contient le fichier d'un périphérique. Obligé de le mettre dans la commande
  sdb1 : C'est le nom qu'on voit quand on fait un **lsblk**. Dans mon cas c'est donc sdb1, mais ça peut être différent pour vous comme sdc1 ou sdd2 par exemple.

- /mnt/usb : c'est le chemin du dossier créé précédemment

Ensuite pour voir le contenu du stockage externe, il faut faire un

```
ls /mnt/usb
```

```
~# ls /mnt/usb/
'$RECYCLE.BIN'   image.bmp   main   'System Volume Information'   Tépé4
```

usb étant le nom que j'ai donné à mon dossier

Dans ma clé USB , le fichier zip qui contient le site se nomme **main**.
Mais il peut s'appeler différemment chez vous, et avoir l'extension .zip, ça ne pose pas de souci

Ensuite pour mettre le fichier main au bonne endroit, on le copie

On fait donc un :

```
cp /mnt/usb/main /var/www/main
```

- Cp : c'est pour faire la copie
- /mnt/usb/main : il faut donner le chemin complet du fichier
- /var/www/main : la destination du fichier. Il ne faut pas oublier de préciser son nom. *Dans mon exemple je met le même nom (main) mais on peut, pour la destination, changer son nom pour autre chose.*

C'est avec une commande équivalente qu'on change le nom des fichiers sur Linux
La commande **mv**
par exemple :
mv /home/user/jeSuisUnCanard.txt /home/user/PlusMaintenant.txt

Mettre les fichiers du site

De mon coté, mon fichier s'appelle toujours main. Chez vous c'est peut-être un autre nom, ce n'est pas grave

Nous allons rentrer la commande suivante pour nous placer là où sera le site :

```
cd /var/www
```

```
:~# cd /var/www/
:/var/www# ls
html  main
```

Si vous faites un :

```
ls
```

Vous verrez *html* qui est le site par défaut de Apache2 et votre site en .zip
(dans mon cas, l'extension .zip n'apparaît pas mais le résultat sera le même).

Maintenant on dézippe le zip, pour le convertir en dossier.
Pour cela, on va rentrer la commande :

```
Unzip main
```

Main étant le nom du fichier à dézipper

```
  inflating: SegPonsBois-main/resources/views/produit/boutique.blade.php
   creating: SegPonsBois-main/routes/
  inflating: SegPonsBois-main/routes/console.php
  inflating: SegPonsBois-main/routes/web.php
   creating: SegPonsBois-main/storage/
   creating: SegPonsBois-main/storage/app/
  inflating: SegPonsBois-main/storage/app/.gitignore
   creating: SegPonsBois-main/storage/app/private/
extracting: SegPonsBois-main/storage/app/private/.gitignore
   creating: SegPonsBois-main/storage/app/public/
extracting: SegPonsBois-main/storage/app/public/.gitignore
   creating: SegPonsBois-main/storage/framework/
   creating: SegPonsBois-main/storage/framework/cache/
extracting: SegPonsBois-main/storage/framework/cache/.gitignore
   creating: SegPonsBois-main/storage/framework/cache/data/
extracting: SegPonsBois-main/storage/framework/cache/data/.gitignore
   creating: SegPonsBois-main/storage/framework/sessions/
extracting: SegPonsBois-main/storage/framework/sessions/.gitignore
   creating: SegPonsBois-main/storage/framework/testing/
extracting: SegPonsBois-main/storage/framework/testing/.gitignore
   creating: SegPonsBois-main/storage/framework/views/
extracting: SegPonsBois-main/storage/framework/views/.gitignore
   creating: SegPonsBois-main/storage/logs/
extracting: SegPonsBois-main/storage/logs/.gitignore
  inflating: SegPonsBois-main/tailwind.config.js
  inflating: SegPonsBois-main/test.html
   creating: SegPonsBois-main/tests/
   creating: SegPonsBois-main/tests/Feature/
  inflating: SegPonsBois-main/tests/Feature/ExampleTest.php
  inflating: SegPonsBois-main/tests/TestCase.php
   creating: SegPonsBois-main/tests/Unit/
  inflating: SegPonsBois-main/tests/Unit/ExampleTest.php
  inflating: SegPonsBois-main/vite.config.js
```

Il va faire apparaître chaque fichier qu'il dézippe, vous montrer qu'il travaille.

Si on refait un :

```
ls
```

Nous voyons un nouveau dossier

```
debian13:/var/www# ls
html  main  siteWEB
```

Le mien s'appelle « siteWEB ». Il peut se nommer autrement chez vous, ce n'est pas inquiétant.

Pour faire de la place, on peut supprimer le zip
Pour cela on fait un :
Rm main

Main étant le nom de mon fichier zip

Ensuite on initialise le site pour Laravel avec ces commandes :

```
Cd /var/www/siteWEB
composer install
```

Si vous avez fait le petit fifrelin comme moi, en étant connecté en root, il va vous écrire cette ligne en jaune :

```
root@debian13:/var/www/siteWEB# composer install
Do not run Composer as root/super user! See https://getcomposer.org/root for details
Continue as root/super user [yes]?
```

Vous pouvez écrire : yes

Il vous prévient c'est que cette commande va télécharger et exécuter des fichiers pour mettre en place des dépendances pour certaines parties du code du site.
Le risque est qu'il pourrait avoir une possibilité qu'une dépendance soit vérolée, mais le risque est extrêmement faible.
Si vraiment, vous avez peur, déconnectez vous et faites le en utilisateur classique.

```
  - Installing phar-io/version (3.2.1): Extracting archive
  - Installing phar-io/manifest (2.0.4): Extracting archive
  - Installing myclabs/deep-copy (1.12.1): Extracting archive
  - Installing phpunit/phpunit (11.4.3): Extracting archive
  - Installing tecnickcom/tcpdf (6.8.0): Extracting archive
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

Deprecated: voku\helper\ASCII::to_ascii(): Implicitly marking parameter $replace_single_chars_only as
nullable is deprecated, the explicit nullable type must be used instead in /var/www/siteWEB/vendor/vok
u/portable-ascii/src/voku/helper/ASCII.php on line 795

  INFO  Discovering packages.

  laravel/pail ..... DONE
  laravel/sail ..... DONE
  laravel/tinker ..... DONE
  laravel/ui ..... DONE
  nesbot/carbon ..... DONE
  nunomaduro/collision ..... DONE
  nunomaduro/termwind ..... DONE

79 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

ça a fonctionné !
et si on fait un :

```
ls
```

On pourra même voir que le dossier *vendor* est apparu, ce qui veut dire que c'est bon.

Nous allons ensuite rentrer une commande pour donner les permissions à Apache2 de pouvoir utiliser les fichiers de notre site.
Pour cela on rentre les commandes suivantes :

```
chown -R www-data:www-data /var/www/siteWEB
```

- Chown –R : change le propriétaire des fichiers et des dossiers, et le paramètre –R est pour récursif, soit même dans les sous dossiers
- www-data:www-data : c'est le nom d'utilisateur et le groupe de Apache2
- /var/www/siteWEB : c'est le chemin complet vers votre site

```
chmod -R 755 /var/www/siteWEB
```

On donne pour chaque fichier dans ce dossier les permissions 755, ce qui veut dire que le propriétaire du fichier peut tout faire sans souci.

```
chmod -R 775 /var/www/siteWEB/storage
chmod -R 775 /var/www/siteWEB/bootstrap/cache
```

Ce sont des dossiers sur tout les sites Laravel, et on autorise avec 775, que le groupe puisse modifier les fichiers.

Les permissions sur Linux sont faites de la façon suivante :

Pour chaque fichier, il y a 3 « types » :
- Propriétaire → celui qui a créé le fichier (par défaut, *on peut ensuite le changer, comme on a fait*)
- Groupe → une équipe (celle du propriétaire)
- Autres → les autres groupes/utilisateurs

Chaque « groupe » peut avoir le droit de:
- Lecture (r) → voir le fichier
- Écriture (w) → modifier le fichier
- Exécution (x) → lancer le fichier

Les chiffres :
On utilise des nombres pour représenter les droits
(on peut aussi utiliser les lettres, mais c'est plus pour juste rajouter un droit)
- 4 = lecture
- 2 = écriture
- 1 = exécution

Et donc on additionne pour le chiffre de chaque droit pour chaque « groupe »

Par exemple : 755 = rwx r-x r-x
donc : - Propriétaire → peut tout faire
- Groupe → peut lire et exécuter
- Autres → peuvent lire et exécuter

Créer le fichier de configuration entre Apache2 et le site

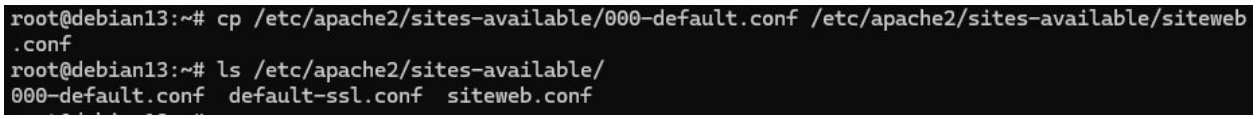
Pour que Apache2 comprenne que les fichiers de votre site sont... un site, il faut un fichier de configuration.

Ayant beaucoup de choses à écrire dans ce fichier, nous allons copier le fichier *000-defaut.conf* et modifier le contenu.
Cela nous permettra de ne pas avoir à écrire beaucoup de lignes.

Donc on écrit la commande :

```
cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/siteweb.conf
```

Siteweb.conf : c'est le nom que je donne à mon fichier copié. Vous pouvez l'appeler comme vous voulez.



Et en faisant un

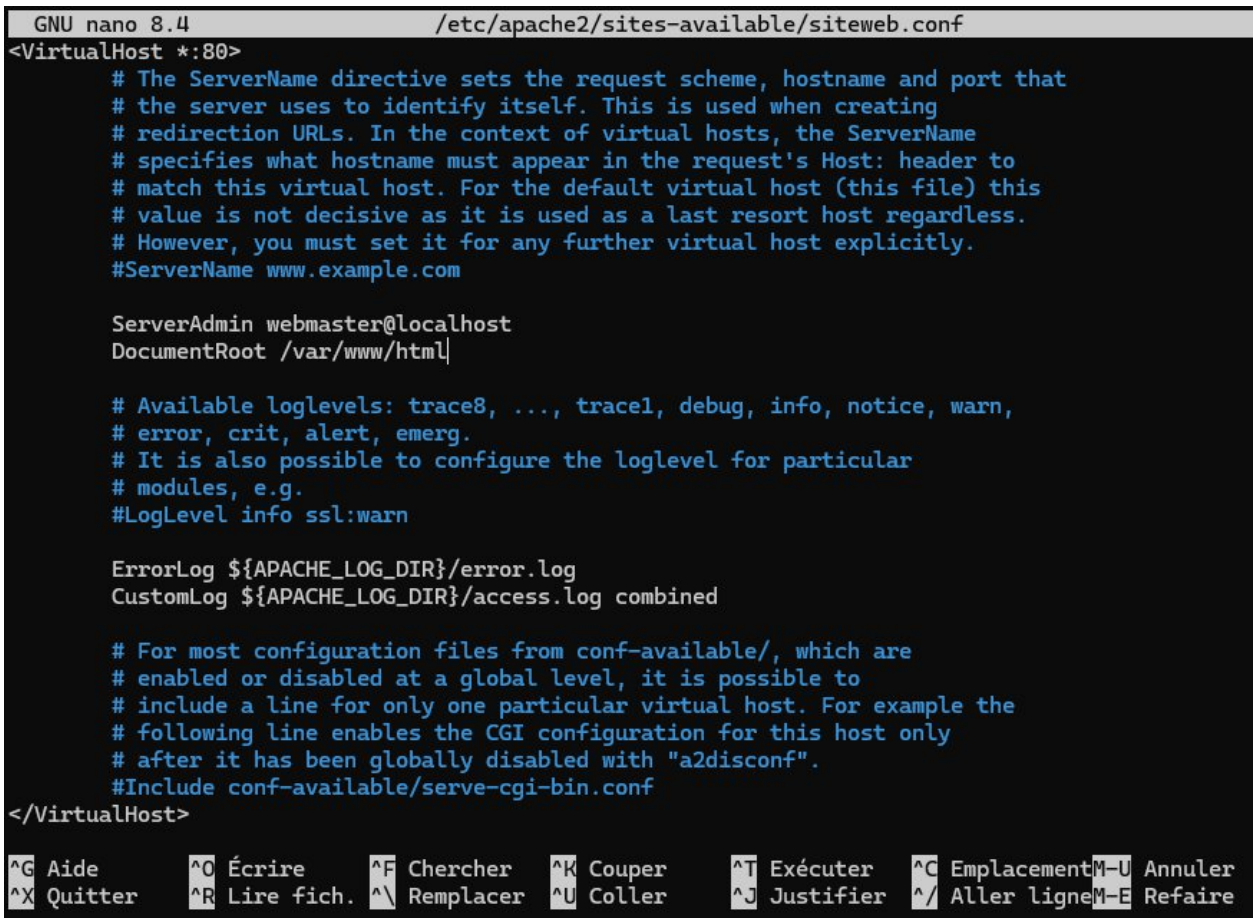
```
ls
```

on voit bien notre fichier créé.

On va ensuite le modifier. Pour cela on ouvre le fichier avec nano :

```
nano /etc/apache2/site-available/siteweb.conf
```

Et on se retrouve sur ça :



Les couleurs peuvent être différentes

Nous allons modifier la ligne :

```
DocumentRoot /var/www/html
```

Par le chemin de notre site, vers le dossier public.

```
DocumentRoot /var/www/siteWEB/public
```

siteWEB par rapport à moi, mais si votre dossier se nomme autrement, mettais le nom de votre dossier

ServerAdmin est pour mettre une adresse mail, si le site a un souci, il fait apparaître cette adresse mail pour qu'on puisse vous contacter.
La plupart du temps, si le site à un souci, il ne pourra pas afficher cette adresse mail, donc on laisse bien souvent celle par défaut

En dessous de DocumentRoot, on rajoute ces lignes :

```
<Directory /var/www/siteWEB/public>
    AllowOverride All
    Require all granted
</Directory>
```

Étant un site Laravel fonctionnant avec Apache2, il lui faut ce petit paragraphe pour fonctionner correctement.

```
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/siteWEB/public

<Directory /var/www/siteWEB/public>
    AllowOverride All
    Require all granted
</Directory>

        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
```

voilà ce que donne l'ajout de ces lignes dans le fichier

Les lignes en bleu sur la capture d'écran, précédées d'un Hashtag #, sont des commentaires. Elles ne sont pas prises en compte par le service utilisant ce fichier et servent d'indications pour nous les humains.

Les deux lignes qui parlent de logs sont pour indiquer à Apache2 ou mettre les logs et les erreurs. Ce sont ces lignes que nous voyons plus tard, pour trouver les soucis quand il y en a.

Nous pouvons ensuite faire la combinaison de touches :

```
Ctrl + s
```

Pour sauvegarder, et

```
Ctrl + x
```

Pour sortir du fichier

Maintenant il faut activer cette configuration à Apache
Mais d'abord on désactive celle par défaut.
pour cela on rentre la commande :

```
a2dissite 000-default.conf
```

```
root@debian13:~# a2dissite 000-default.conf
Site 000-default disabled.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Il nous dit que c'est bien désactivé, et que pour ce soit pris en compte, on recharge Apache2.
On va le faire plus tard

Maintenant on active la config que l'on vient de créer :

```
A2ensite siteweb.conf
```

```
root@debian13:~# a2ensite siteweb.conf
Enabling site siteweb.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Et nous n'allons pas encore reload puisqu'il reste encore quelques trucs à faire avant de lancer le site.

Le fichier .env

Le fichier le plus important pour un site sous laravel. C'est ce fichier qui indique toute les informations utiles à son fonctionnement. Sans lui, le site ne fonctionne pas ou assez mal.

Pour commencer, nous allons nous déplacer dans le dossier du site avec la commande :

```
Cd /var/www/siteWEB
```

siteWEB étant le dossier qui contient mon site, mais peut être différent si vous aviez mis autre chose dans les étapes précédentes.

Puis nous allons rentrer dans le fichier .env

```
Nano .env
```

Il y a plusieurs possibilités au moment ou le fichier est ouvert :

- Soit il sera petit avec quelques options

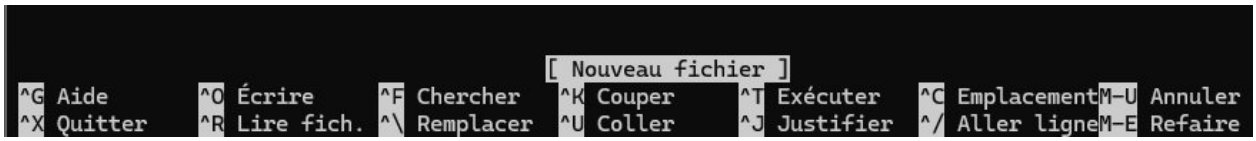
```
APP_NAME=Laravel
APP_ENV=production
APP_KEY=base64:
APP_DEBUG=false
APP_URL=https://monsite.fr

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=larabois
DB_USERNAME=laravel
DB_PASSWORD=motdepasse

SESSION_DRIVER=file
CACHE_STORE=file
QUEUE_CONNECTION=sync
```

- Soit il est très long :

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:ZV1vHpCIYGD3Caxxl6hjFNQq1QT8F3aqcwSd6JtcfyM=
APP_DEBUG=true
APP_URL=http://localhost

APP_LOCALE=en
APP_FALLBACK_LOCALE=en
APP_FAKER_LOCALE=en_US

APP_MAINTENANCE_DRIVER=file
# APP_MAINTENANCE_STORE=database

# PHP_CLI_SERVER_WORKERS=4

BCRYPT_ROUNDS=12

LOG_CHANNEL=stack
LOG_STACK=single
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=larabois
DB_USERNAME=root
DB_PASSWORD=root

SESSION_DRIVER=database
SESSION_LIFETIME=120
SESSION_ENCRYPT=false
SESSION_PATH=/
SESSION_DOMAIN=null

BROADCAST_CONNECTION=log
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database

CACHE_STORE=database
# CACHE_PREFIX=

MEMCACHED_HOST=127.0.0.1

REDIS_CLIENT=phpredis
REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=log
MAIL_SCHEME=null
MAIL_HOST=127.0.0.1
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_FROM_ADDRESS="hello@example.com"
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false

VITE_APP_NAME="${APP_NAME}"
```

Long, il a toute les options, même celles pas utiles

– Soit il va dire :



Dans les trois cas, se sera la **même méthode** :
Si la ligne utile n'apparaît pas, écrivez la, sinon remplacer ce qui est écrit après le '**=**' par ce qu'il faut.

Voici tout ce qu'il faut pour que ça marche sans souci :
Pour différentier les indications que je dis des lignes à mettre dans le .env, je commencerai mes phrases par un **#** *(et en jaune, pour faire la distinction avec ce qu'il faut mettre)* :

```
APP_NAME=Laravel
#vous pouvez mettre ce que vous voulez comme nom
APP_ENV=production
#pour préciser à Laravel qu'il est en fonctionnement de production,
#et non en dev
APP_KEY=base64:
#on ne met rien, on lui met la prochaine fois
APP_DEBUG=false
APP_URL=http://192.168.1.47
#Celui là, pour test que tout aille bien on mettra l'ip de la machine
#( avec http ).
#Sinon on pourra mettre en https:// et le nom de domaine

LOG_CHANNEL=stack
LOG_STACK=single
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=warning
#C'est comment vont se mettre les logs, ce sont les paramètres simple
#sans prise de tête


#Très important : c'est avec ces infos que Laravel va se connecter à
#notre base de données
DB_CONNECTION=mysql
#mariaDB se base sur mysql, donc pas besoin d'y toucher
DB_HOST=127.0.0.1
#c'est l'adresse ou se trouve le serveur mariaDB. Vu qu'il est sur la
#même machine, on met 127.0.0.1
DB_PORT=3306
#C'est le port d'écoute de la base de donnée. Par défaut il met 3306,
#si on ne l'a pas modifier sur le service MariaDB, on laisse cette
#valeur
DB_DATABASE=database
#Le nom de la base de données que l'on a créer avec la DB
#( pas d'espace entre le nom et le égal )
DB_USERNAME=userWEB
#Le nom d'utilisateur que l'on a créer avec la DB
#( pas d'espace entre le nom et le égal )
DB_PASSWORD=motdepasse
#le mot de passe qu'on a spécifié à la création du compte
#(pas d'espace entre le nom et le égal )

SESSION_DRIVER=database
SESSION_LIFETIME=120
SESSION_ENCRYPT=false
SESSION_PATH=/
SESSION_DOMAIN=null
#c'est quand un client se connecte avec son compte sur le site qu'il
```

```
#reste connecté.
#SESSION_LIFETIME est mis à 120, donc 2H c'est la norme sur les
#sites.

MAIL_MAILER=log
MAIL_SCHEME=null
MAIL_HOST=127.0.0.1
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_FROM_ADDRESS="hello@example.com"
MAIL_FROM_NAME="${APP_NAME}"
#Sera utile si vous voulez que le site web envoie automatiquement des
#mails,
#ou envoyer des mails depuis le site, il faudra configurer un serveur
#de mail et rentrer les informations ici

CACHE_STORE=file
QUEUE_CONNECTION=sync
#c'est pour un meilleur fonctionnement du site
```

Dès que les modifications sont effectuées, vous pouvez faire un

```
ctrl + s
```

Pour sauvegarder, puis

```
Ctrl + x
```

Pour quitter le fichier

Si vous étiez dans la situation que *nano .env* créer un nouveau fichier
faites les commandes :

```
chown www-data:www-data /var/www/siteWEB/.env
chmod 755 /var/www/siteWEB/.env
```

Les dernières commandes

Maintenant ça va être un ensemble de commandes pour initialiser Laravel, démarrer les services, et que tout fonctionne

Avant toute chose, assuez-vous d'être dans le dossier du site

```
root@debian13:/var/www/siteWEB#
```

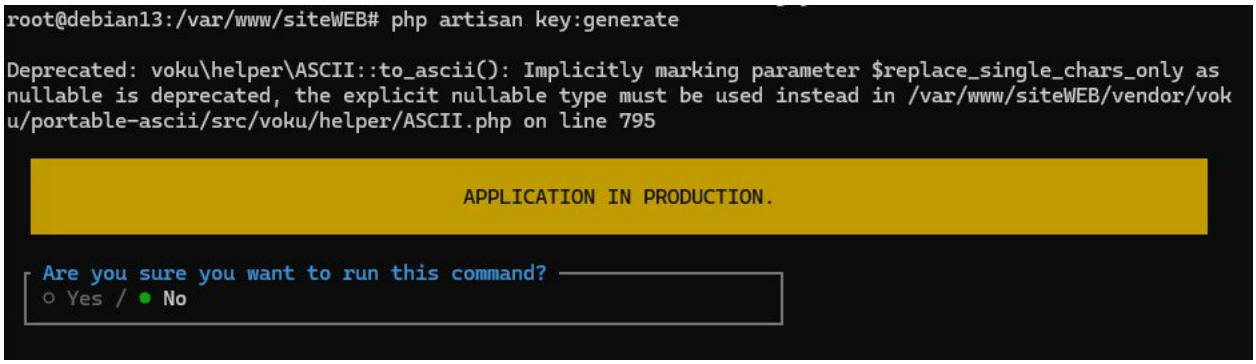
Sinon faite :

```
cd /var/www/siteWEB
```

SiteWEB étant le nom de mon dossier, mettez le votre si ce n'est pas le même

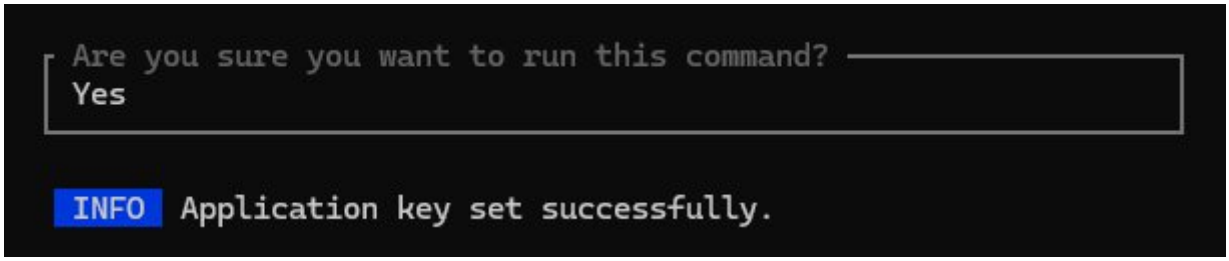
On commence par les commandes pour Laravel :

```
Php artisan key:generate
```



Avec les flèches, on se met sur « **yes** » puis entrée

L'erreur Deprecated, est juste un avertissement sur une variable utilisée à l'ancienne, ce n'est pas grave, rien ne va casser à cause de ça.
Dans mon exemple j'ai cette erreur, mais vous pouvez ne pas en avoir ou une équivalente



Si vous avez ça, c'est parfait

Ensuite on passe un coup de balai pour supprimer le cache avec les commandes :

```
php artisan config:clear
php artisan cache:clear
```

```
root@debian13:/var/www/siteWEB# php artisan config:clear
php artisan cache:clear

Deprecated: voku\helper\ASCII::to_ascii(): Implicitly marking parameter $replace_single_chars_only as
nullable is deprecated, the explicit nullable type must be used instead in /var/www/siteWEB/vendor/vok
u/portable-ascii/src/voku/helper/ASCII.php on line 795

  INFO  Configuration cache cleared successfully.


Deprecated: voku\helper\ASCII::to_ascii(): Implicitly marking parameter $replace_single_chars_only as
nullable is deprecated, the explicit nullable type must be used instead in /var/www/siteWEB/vendor/vok
u/portable-ascii/src/voku/helper/ASCII.php on line 795

  INFO  Application cache cleared successfully.
```

petite folie, j'ai exécuté les deux commandes en même temps
Vous pouvez les faire une par une, le résultat sera le même.

Laravel nous dit que le cache a bien été nettoyé.
Et nous indique le même souci. Mais vous pouvez très bien n'avoir aucun message d'avertissement

Ensuite :

```
php artisan migrate
```

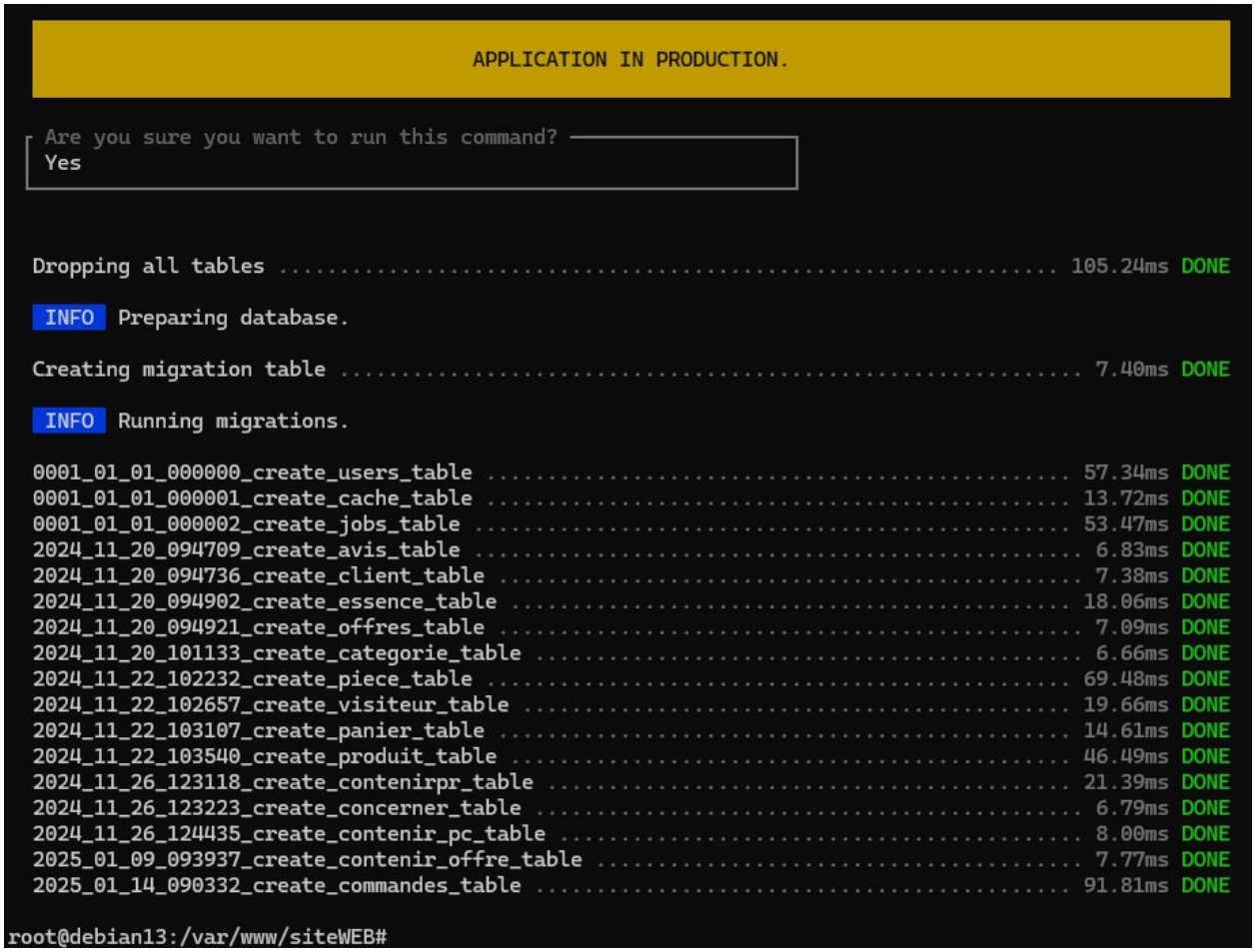
C'est pour que Laravel initialise la base de données.

Donc si vous avez mal écrit les infos dans le fichier .env ou mal mis les privilèges au moment de la création de la base de données, vous allez le savoir avec cette commande.

Il va nous demander une confirmation, on fait « **yes** »

```
   APPLICATION IN PRODUCTION.

 Are you sure you want to run this command?
  ○ Yes / ● No
```

Si ça se passe bien, il vous dira :



Si vous avez cette erreur ou équivalent :

Vous ne pourrez rien y faire, c'est un problème de programmation, les fichiers pour initialiser la base de données on un problème de programmation.

Si il vous dit problèmes de permissions, c'est soit que vous avez mal écrit les infos dans le fichier **.env**, ou alors, mal mis les privilège au moment de la création de la base de données.

Ensuite :

```
php artisan session:table
```

Pour les sessions clientes, qu'elles puissent être stockées quelque part

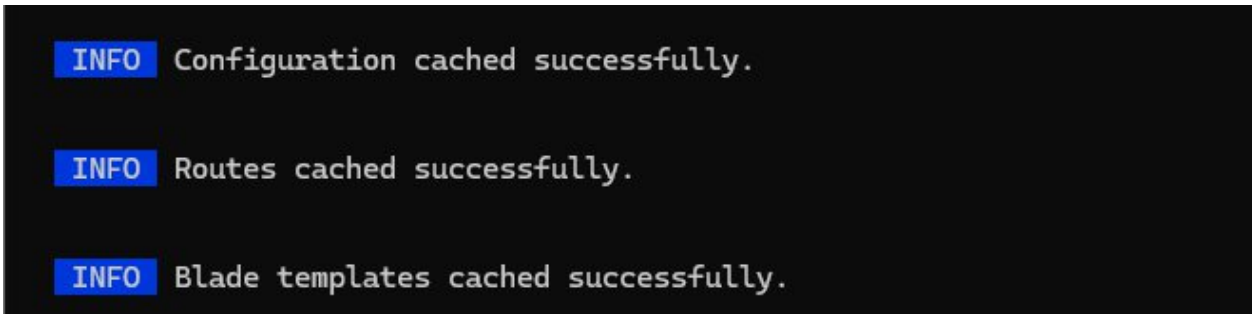
Il peut dire qu'elle est déjà créée, et donc c'est bon aussi :


```
php artisan storage:link
```

Pour qu'il trouve le dossier « storage » pour stocker des fichiers

Et ensuite un dernier coup de balai pour faire joli :

```
php artisan config:cache
php artisan route:cache
php artisan view:cache
```



Et on peut reload Apache2

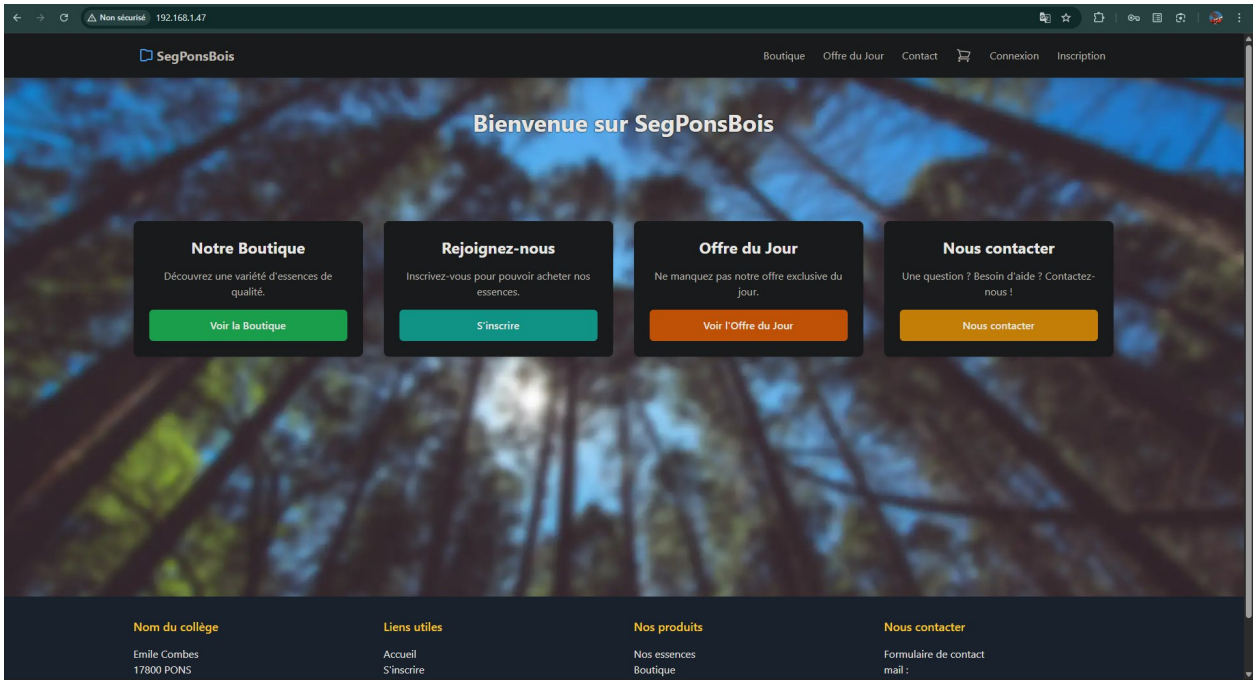
```
Systemctl reload Apache2
```

La différence entre reload et restart :

Reload : il va recharger la config. Il reste donc allumé. L'avantage c'est que si la nouvelle config à un souci, il va rester sur la config qu'il était, et donc permet de ne pas avoir son site qui pourrait ne plus être accessible

Restart : il redémarre avec la config dans les fichiers. Si le service est lent, consomme trop, parfois, juste le restart suffit pour bien repartir
C'est aussi pour avoir la sureté que la config est bonne et peut donc repartir en toute situation.

Et si vous allez, sur un navigateur d'un pc sur le même réseau que votre serveur, et rentrer l'ip de votre serveur, vous avez votre site !



Bravo, votre site sous Laravel fonctionne.