APPASSAMY RANJITH

MACHU ALEXANDRE

PHILIPPE YTHIER AYMERIC

INFO 1 GROUP D 18

IUT ANNECY
UNIVERSITÉ SAVOIE MONT BLANC

NETWORK INSTALLATIONS

I) CLIENT'S DEMAND

Our client “IUT Annecy” would like to develop and deploy communicative web applications. To do so, they need a server.

A server is basically a computer. However contrary to a local computer, the server is connected to a big network, well known as the Internet. That way the server can be accessible from everywhere around the world by their users. The main purpose of the a server are : share and centralize data and resources , host website, web applications (to be developed)

We can find among the famous sever hosts: OVH, Hostinger, Bluehost, Dreamhost and others. But their price are very expensive in long term. Our customer decided to create his own server. That’s why he came to us: to install and make a working sever according to their budget.

The customer budget is very limited. Indeed, they want us to install a server on a Raspberry PI 2 model B. This device come with only 1GB memory. It has 4 USB ports, an ethernet port and an HDMI port to connect to a display device. The storage device can be an external hard drive or USB. In our case the customer chooses a 8GB SD card.

To realise that demand we studied carefully the situation and explained in this report.

II) THE CHOICE OF THE OPERATING SYSTEM

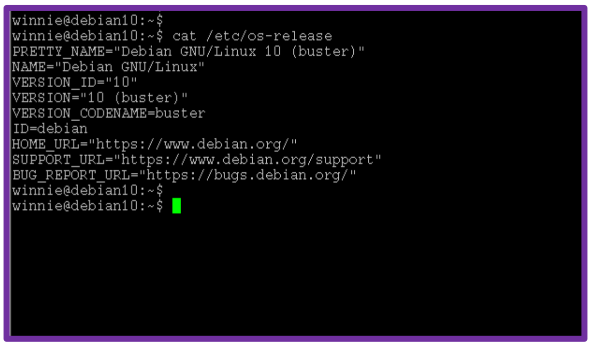
The client gave us a device with 8GB storage which is possible but very limited. So the first step was to choose a well performing OS (Operating System) .

The choice was very fast, we choose Linux based Operating System because it's an Open Source OS available for everyone. Moreover, Linux is UNIX based which was designed to be easy to use, stable, reliable and powerful. We also made this choice because the others server are mainly running on a UNIX based system.

It is also important to precise that Linux system don't need desktop environment (graphical) to be used. It can be used be operated from a screen called terminal which is very efficient and powerful.

However, Linux is not an Operating System it as a lot of variations. We choose the Debian distributions which is known to be the mother of the major Linux distributions. Debian is easy to use and a lightweight, a major fact in this project. Moreover, Debian based distributions offers an easy installation of software, thanks to the apt repository.

Debian (Linux) with a Gnome desktop environment



```
winnie@debian10:~$
winnie@debian10:~$ cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 10 (buster)"
NAME="Debian GNU/Linux"
VERSION_ID="10"
VERSION="10 (buster)"
VERSION_CODENAME=buster
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
winnie@debian10:~$
winnie@debian10:~$
```

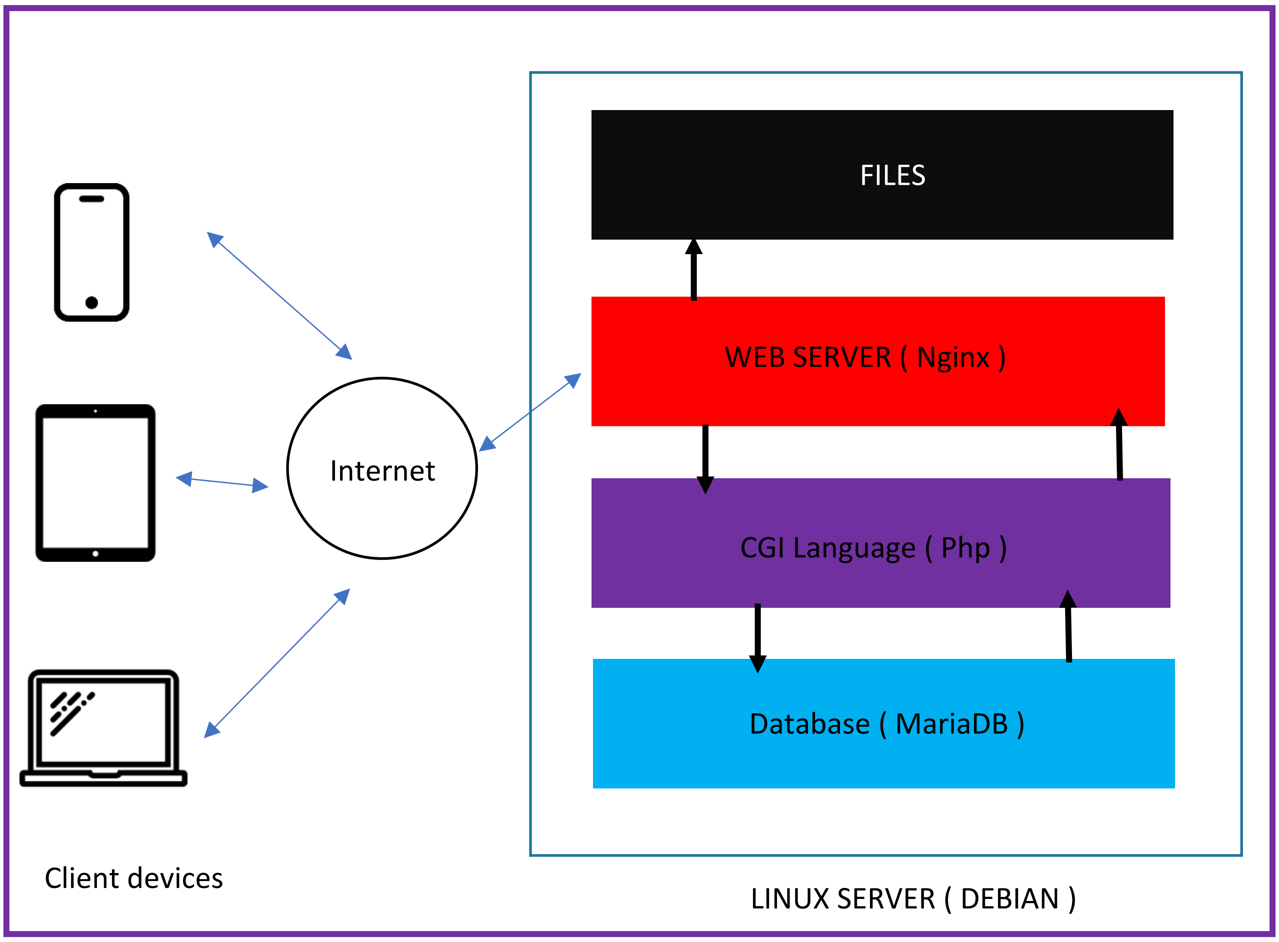
Debian (Linux) server version (without desktop environment)

Debian is for a normal desktop architecture like a Dell, Asus desktop. Our customer wants to run his server on a Raspberry Device which has a different architecture. To do that we had to download a specific OS called "Raspbian" which is designed to run Debian on raspberry devices. So, we installed the software "Raspberry PI Imager" and choose the SD card and wrote on that the image of "Raspberry Pi OS Lite (32bit)".

So, we wrote the image of the OS on the given storage, in our case an 8 GB SD Card and start the installation and the basic configuration of the server, again we choose a graphics free installation according to our customer's limited resources.

III) LEMP SERVER EXPLICATION

This is a representation to explain how works an web server.



LEMP stands for: Linux, Engine-X also known as Nginx, MariaDB, Php. So LEMP it's a combination of multiple software that lets the web development on a server. Each of the software in LEMP have their own usage. Here they are:

- Linux is the server's Operating system that allow to function
- Nginx is the software that tells the server what to do with the client's requests. For example, it tells the location of the webpage's files to the other software
- MariaDB stores information on a database

- Php is a scripting software that fulfils requests with database information. It also builds the webpage an return in to Nginx.

IV) LEMP SERVER INSTALLATION

A) LEMP – NGINX

First, we are going to update and upgrade the apt repository:

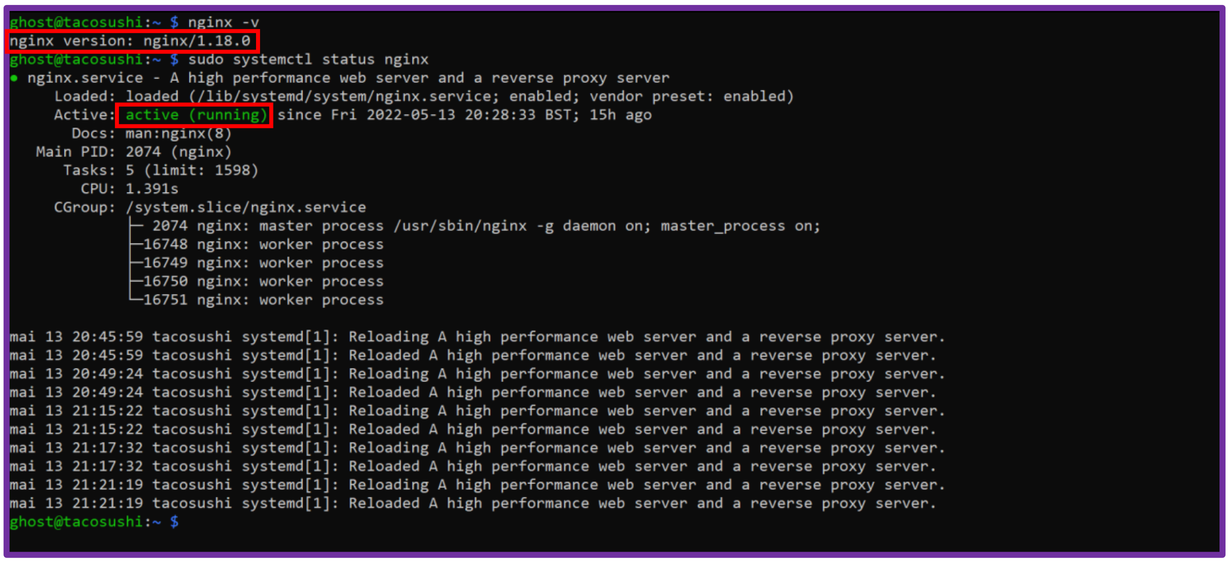
- *sudo apt-get update && sudo apt-get upgrade*

Then you will have to install the latest nginx sever and enable it:

- *sudo apt install ngnix -y*

To check the installation, type the following commands:

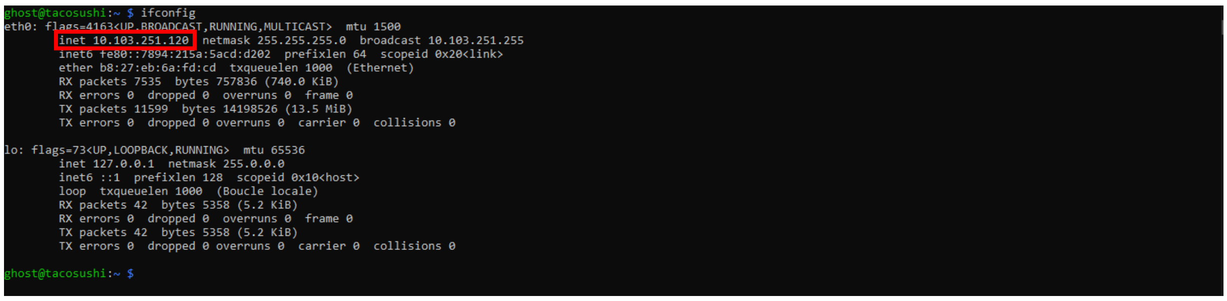
- *nginx -v*
- *sudo systemctl status ngnix*



```
ghost@tacosushi:~ $ nginx -v
nginx version: nginx/1.18.0
ghost@tacosushi:~ $ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2022-05-13 20:28:33 BST; 15h ago
       Docs: man:nginx(8)
   Main PID: 2074 (nginx)
      Tasks: 5 (limit: 1598)
        CPU: 1.391s
     CGroup: /system.slice/nginx.service
             ├─ 2074 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
             ├─16748 nginx: worker process
             ├─16749 nginx: worker process
             ├─16750 nginx: worker process
             └─16751 nginx: worker process

mai 13 20:45:59 tacosushi systemd[1]: Reloading A high performance web server and a reverse proxy server.
mai 13 20:45:59 tacosushi systemd[1]: Reloaded A high performance web server and a reverse proxy server.
mai 13 20:49:24 tacosushi systemd[1]: Reloading A high performance web server and a reverse proxy server.
mai 13 20:49:24 tacosushi systemd[1]: Reloaded A high performance web server and a reverse proxy server.
mai 13 21:15:22 tacosushi systemd[1]: Reloading A high performance web server and a reverse proxy server.
mai 13 21:15:22 tacosushi systemd[1]: Reloaded A high performance web server and a reverse proxy server.
mai 13 21:17:32 tacosushi systemd[1]: Reloading A high performance web server and a reverse proxy server.
mai 13 21:17:32 tacosushi systemd[1]: Reloaded A high performance web server and a reverse proxy server.
mai 13 21:21:19 tacosushi systemd[1]: Reloading A high performance web server and a reverse proxy server.
mai 13 21:21:19 tacosushi systemd[1]: Reloaded A high performance web server and a reverse proxy server.
ghost@tacosushi:~ $
```

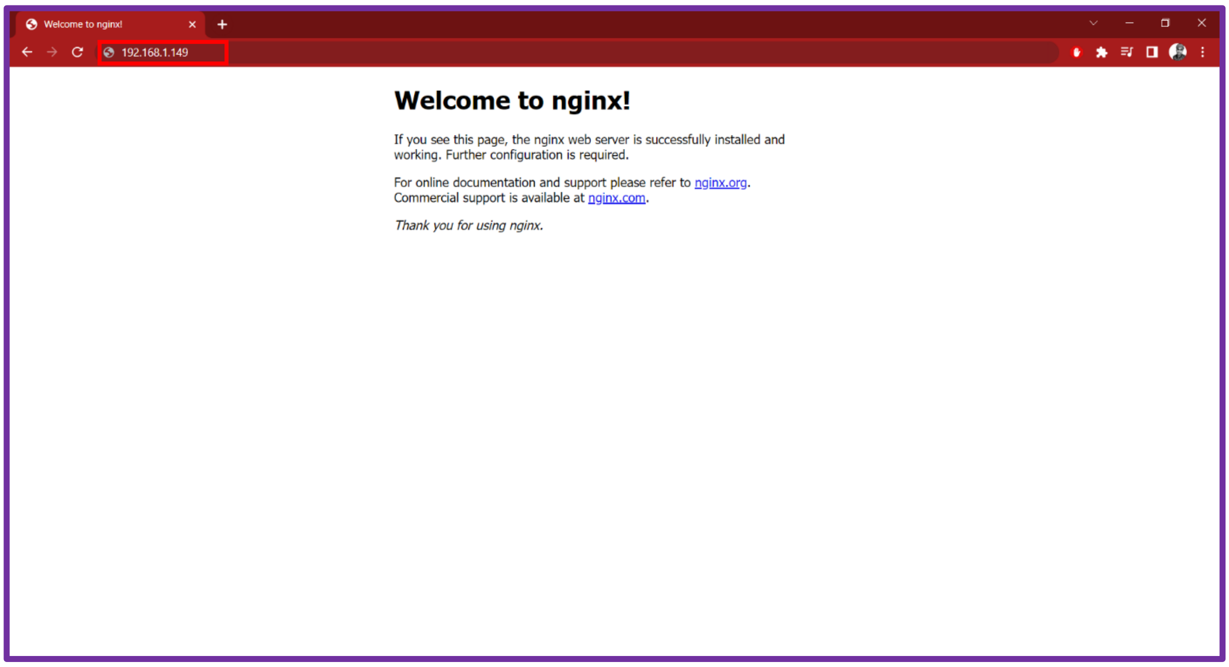
To check the correct installation of nginx we have to test the landing on the default page. To do that you will need another computer connected on the same network. Once done we will install a package to get the server IP address:

- *sudo apt install net-tools*
- *ifconfig*



```
ghost@tacosushi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.103.251.120  netmask 255.255.255.0  broadcast 10.103.251.255
        inet6 fe80::7894:215a:5acd:d202  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:6a:fd:cd  txqueuelen 1000  (Ethernet)
        RX packets 7535  bytes 757836 (740.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11599  bytes 14198526 (13.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Boucle locale)
        RX packets 42  bytes 5358 (5.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 42  bytes 5358 (5.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ghost@tacosushi:~ $
```

We marked our ip address with a red box. In our case it is: 10.103.251.120. Once you have our server IP, switch on the other computer and open a web browser. Then type your ip address in the search bar. You should have a page like this:



Welcome to nginx!

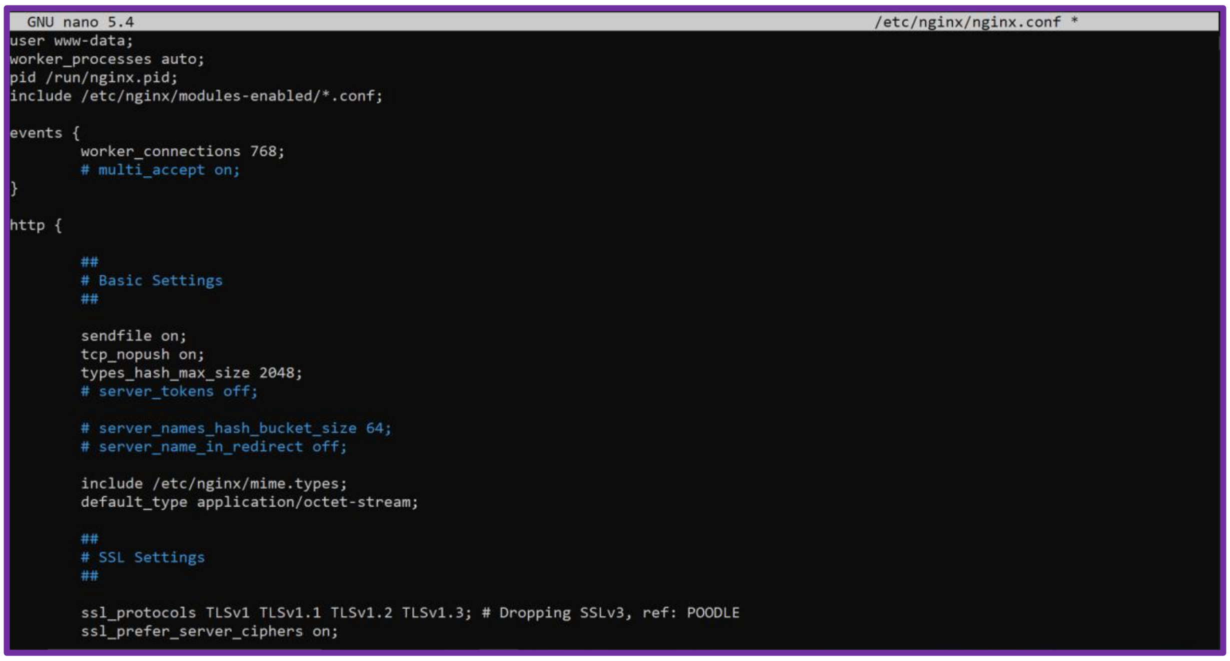
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

It is a must to hide our Nginx server version. Indeed, if someone has access to this kind of information they can probably search for the flaw and so attack the server. This is not what we want. To solve that follow this command:

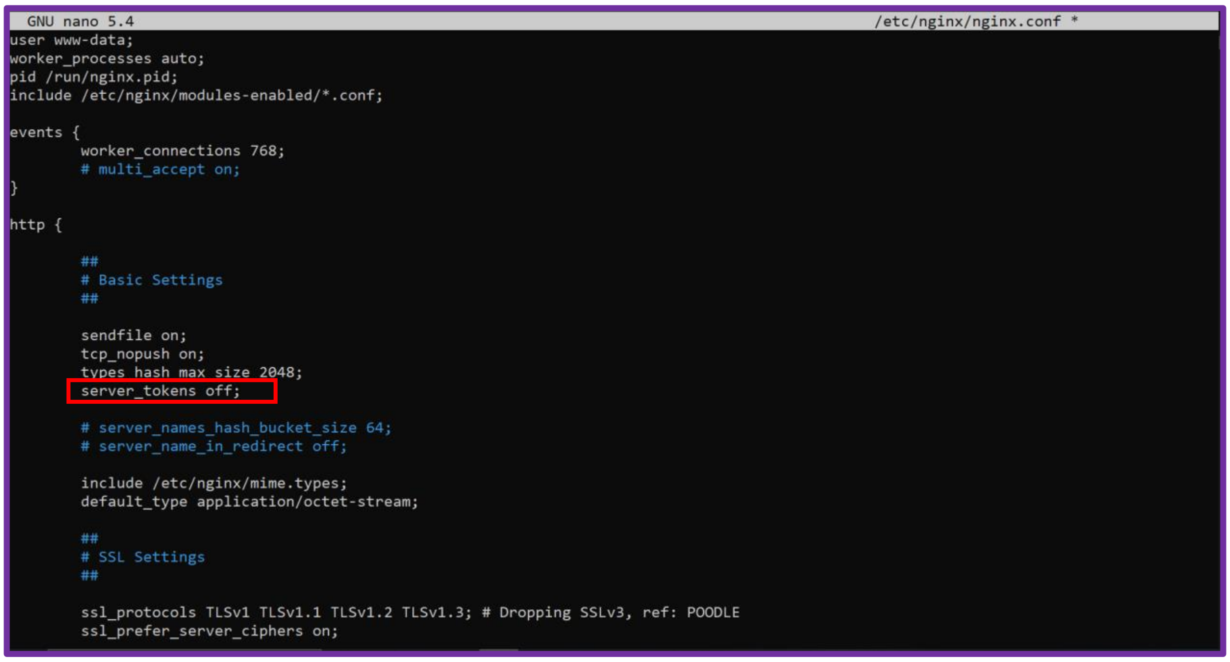
- *sudo nano /etc/nginx/nginx.conf*



```
 GNU nano 5.4                                                                        /etc/nginx/nginx.conf *
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
        worker_connections 768;
        # multi_accept on;
}

http {

        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        types_hash_max_size 2048;
        # server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;

        ##
        # SSL Settings
        ##

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
        ssl_prefer_server_ciphers on;
```

- *reach the line with "# server_tokens off;"*

- Remove the "#" at the beginning of the line



```
 GNU nano 5.4                                                                        /etc/nginx/nginx.conf *
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
        worker_connections 768;
        # multi_accept on;
}

http {

        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        types_hash_max_size 2048;
        server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;

        ##
        # SSL Settings
        ##

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
        ssl_prefer_server_ciphers on;
```

- Press: *Ctrl + X*
- Press: *Y*
- *sudo systemctl restart nginx*

B) LAMP – PHP

Now we are going to install php:

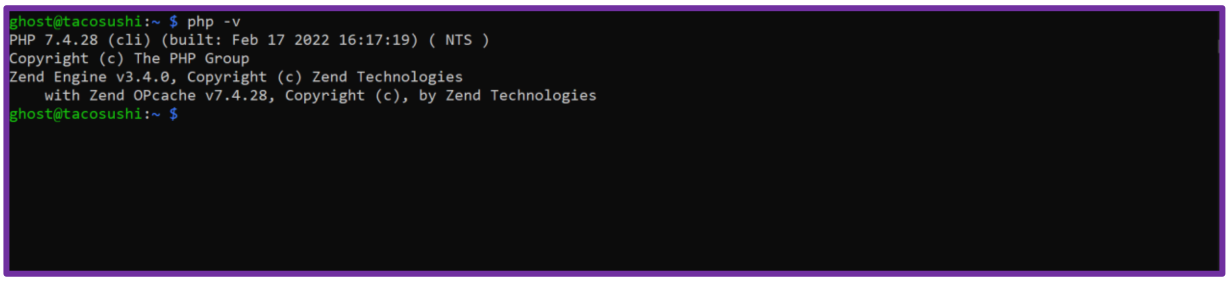
- *sudo apt install php*
- sudo apt install php7.4 php7.4-fpm

We are also going to install some additional packages to complete the installations. This will help to let the communication between Php and MariaDB. To do so, run this small command line:

- *sudo apt-get install -y php-mysql php-zip php-gd php-mbstring php-curl php-xml php-pear php-bcmath*

In order to check if the installation was successful, type:

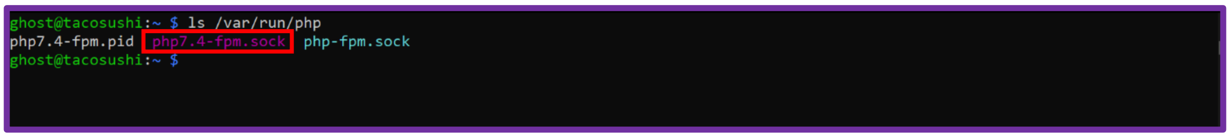
- *php -v*



```
ghost@tacosushi:~ $ php -v
PHP 7.4.28 (cli) (built: Feb 17 2022 16:17:19) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
    with Zend OPcache v7.4.28, Copyright (c), by Zend Technologies
ghost@tacosushi:~ $
```

Now php is installed on our server in local. But we are going to do some commands to have php on the LEMP Server.

ls /var/run/php



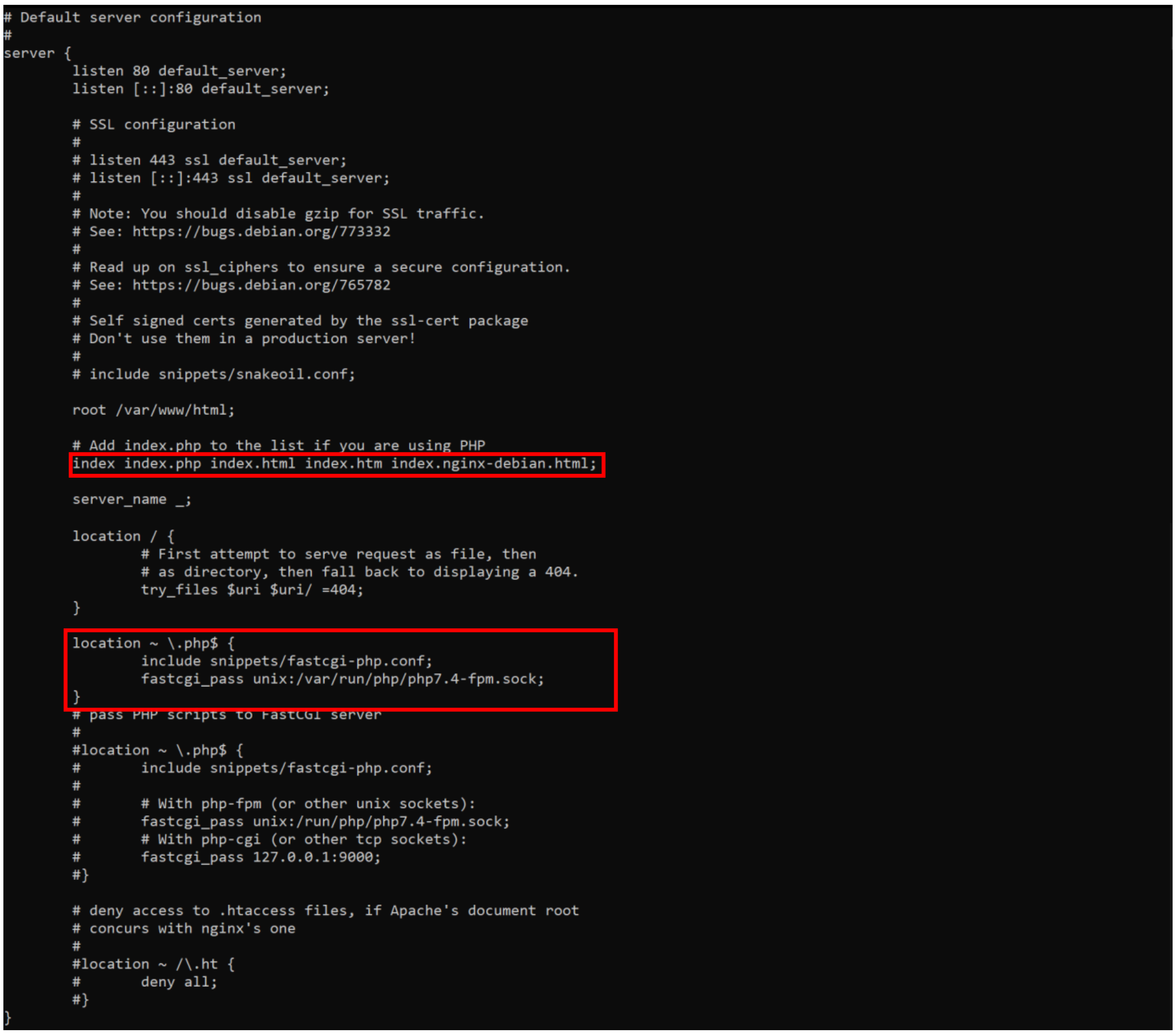
```
ghost@tacosushi:~ $ ls /var/run/php
php7.4-fpm.pid  php7.4-fpm.sock  php-fpm.sock
ghost@tacosushi:~ $
```

The purple sock is what we need. In our case is “php7.4-fpm.sock”

We will add a code block in the sites files to let them have php. Make sure your file looks like the screenshot below. The modifications we added on this steps are again inside the red box. The lines starting with a “#” are just comments so you can ignore them. The lines where we edited or we added are inside the block server { } .

Type :

- *sudo nano /etc/nginx/sites-available/default*



```
# Default server configuration
#
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
        # Note: You should disable gzip for SSL traffic.
        # See: https://bugs.debian.org/773332
        #
        # Read up on ssl_ciphers to ensure a secure configuration.
        # See: https://bugs.debian.org/765782
        #
        # Self signed certs generated by the ssl-cert package
        # Don't use them in a production server!
        #
        # include snippets/snakeoil.conf;

        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.php index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }

        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        }
        # pass PHP scripts to FastCGI server
        #
        #location ~ \.php$ {
        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/run/php/php7.4-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;
        #}

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #
        #location ~ /\.ht {
        #       deny all;
        #}
}
```

Notice that in the second red box, in the last line we added a path: “/var/run/php/php7.4-fpm.sock”. This the purple sock we previously found. Make sure you adapt it according to your values. And it is important to do not forgot the semi-colons at the end of lines. Other wise you will get errors and your server wont work.

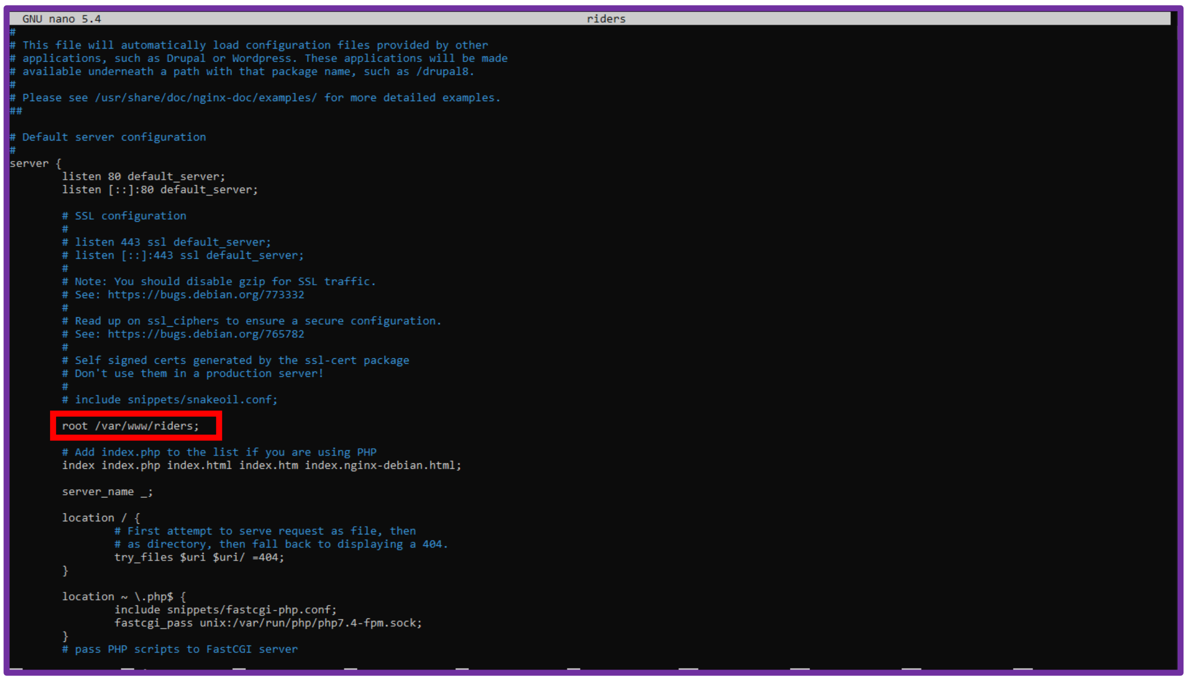
Then type this command. Otherwise you will have to repeat every single modification in the directory: /var/run/sites-enabled. The following command will automatically edit in both directory:

- *ln -s /etc/nginx.sites-available /etc/nginx/sites-enabled*
- *sudo systemctl reload nginx*

Then we will create some files and repository, in order to host our future website. We use the value riders in the example. You have to change it according to your preferences.

- *cd /etc/nginx/sites-available*
- *sudo mkdir /var/www/riders*
- *sudo cp -r default riders*
- *sudo nano riders*

Make sure you have the right values. The edited section is inside the red box.



```
  GNU nano 5.4                                                                                     riders
#
# This file will automatically load configuration files provided by other
# applications, such as Drupal or Wordpress. These applications will be made
# available underneath a path with that package name, such as /drupal8.
#
# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.
##

# Default server configuration
#
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
        # Note: You should disable gzip for SSL traffic.
        # See: https://bugs.debian.org/773332
        #
        # Read up on ssl_ciphers to ensure a secure configuration.
        # See: https://bugs.debian.org/765782
        #
        # Self signed certs generated by the ssl-cert package
        # Don't use them in a production server!
        #
        # include snippets/snakeoil.conf;

        root /var/www/riders;

        # Add index.php to the list if you are using PHP
        index index.php index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }

        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        }
        # pass PHP scripts to FastCGI server
```

C) LAMP – MARIADB

Actually, the letter M in the word LEMP, stands for MySQL. However, we choose Maria DB for a specific reason. Maria DB is actually based on MySQL Community version. In opposite of MySQL, Maria DB is a full open source software. That said it is important to notice that we won't have any conflicts with MySQL policy in case of future changes.

To install Maria DB, type:

- *sudo apt-get install -y mariadb-server*

In order to run a secure installation and configuration, type:

- *sudo mariadb-secure-installation*

Notice that you will now enter the mariadb console! The console request is represented in blue. The action you have to do are in purple. We precise that this information will be used later. So make sure to write them to do not forgot them!

Enter current password for root (enter for none) :

Press Enter

Switch to unix_socket authentication [Y/n] :

Type "n" and Enter

Change the password to root? [Y/n] :

Type "y" and Enter

Define your new passwords (Don't forgot this password!)

Remove anonymous users? [Y/n]

Type "y" and Enter

Disallow root login remotely? [Y/n] :

Type "y" and Enter

Remove test database and access to it? [Y/n] :

Type “y” and Enter

The configuration of Maria DB is ready. To check the correct installation, we have to check thanks to these commands:

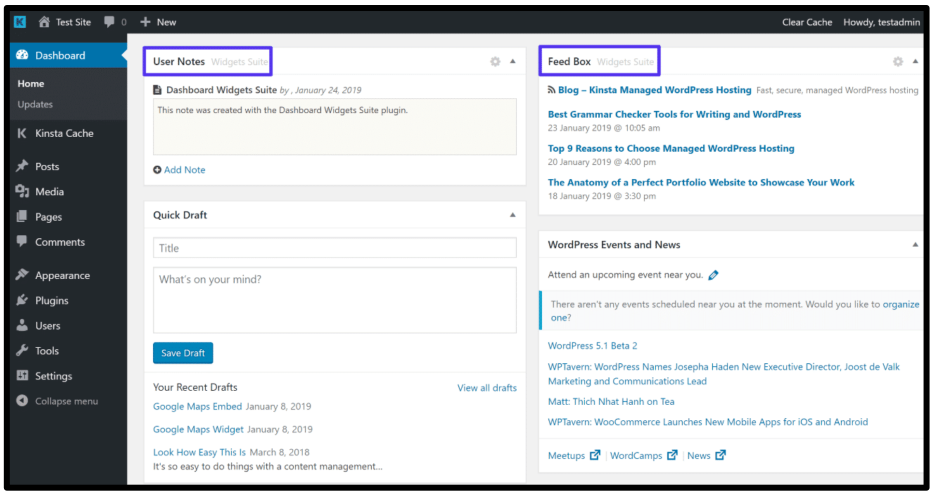
- *sudo mysql -u root -p*
- *Enter your password*
- *show databases;*
  Don’t ignore the semi-colon ! It is very important !
  You should get a grid of the databases
- *exit*

V) WORDPRESS – EXPLAINED

WordPress is a CMS, which stands for Content Management System. WordPress was created in 2003 but it is today the most known CMS around the world.

A CMS is a platform that let you create, edit, and publish websites. To that you don't need any particular skills like HTML, CSS or JavaScript. WordPress is a very user-friendly CMS that doesn't need any experience, that beginners can easily create users, page and articles.

The creations can be easily customable thanks to the large choice of the plugins. It is important to precise that some plugins are not free. As there are many, you will probably need some time at the beginning to find the correct ones. You can also create your own plugins to your taste, however in that case you will skills for the Php language.



Dashboard Widgets Suite by , January 24, 2019

This note was created with the Dashboard Widgets Suite plugin.

Google Maps Embed January 8, 2019

Google Maps Widget January 8, 2019

Look How Easy This Is March 8, 2018

It's so easy to do things with a content management...

Blog – Kinsta Managed WordPress Hosting Fast, secure, managed WordPress hosting

Best Grammar Checker Tools for Writing and WordPress
23 January 2019 @ 10:05 am

Top 9 Reasons to Choose Managed WordPress Hosting
20 January 2019 @ 4:00 pm

The Anatomy of a Perfect Portfolio Website to Showcase Your Work
18 January 2019 @ 3:30 pm

Attend an upcoming event near you.

There aren't any events scheduled near you at the moment. Would you like to organize one?

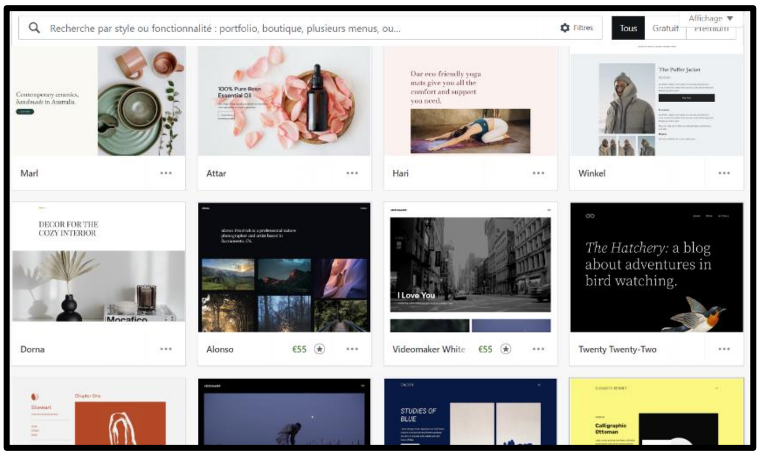
WordPress 5.1 Beta 2

WPTavern: WordPress Names Josepha Haden New Executive Director, Joost de Valk Marketing and Communications Lead

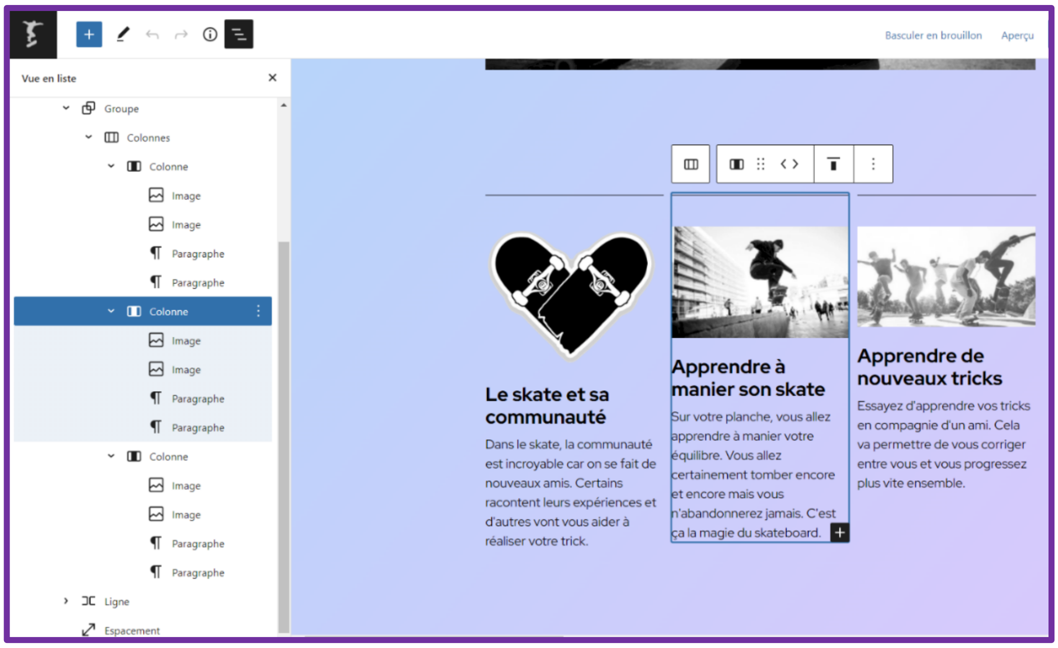
Matt: Thich Nhat Hanh on Tea

WPTavern: WooCommerce Launches New Mobile Apps for iOS and Android

Overview of the Wordpress dashboard


Recherche par style ou fonctionnalité : portfolio, boutique, plusieurs menus, ou...
Filtres
Tous
Gratuit
Premium
Affichage
Marl
Attar
Hari
Winkel
Dorna
Alonso
€55
Videomaker White
€55
Twenty Twenty-Two
The Hatchery: a blog about adventures in bird watching.
I Love You
DECOR FOR THE COZY INTERIOR

To make a WordPress website, you just have to choose a template among what the site proposes (as on the following image) or to look for one that you like. It is also possible to make your website entirely from scratch.



Le skate et sa communauté

Dans le skate, la communauté est incroyable car on se fait de nouveaux amis. Certains racontent leurs expériences et d'autres vont vous aider à réaliser votre trick.

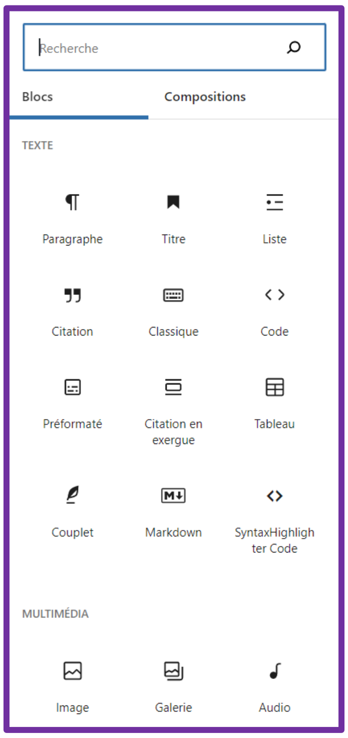
Apprendre à manier son skate

Sur votre planche, vous allez apprendre à manier votre équilibre. Vous allez certainement tomber encore et encore mais vous n'abandonnerez jamais. C'est ça la magie du skateboard.

Apprendre de nouveaux tricks

Essayez d'apprendre vos tricks en compagnie d'un ami. Cela va permettre de vous corriger entre vous et vous progressez plus vite ensemble.

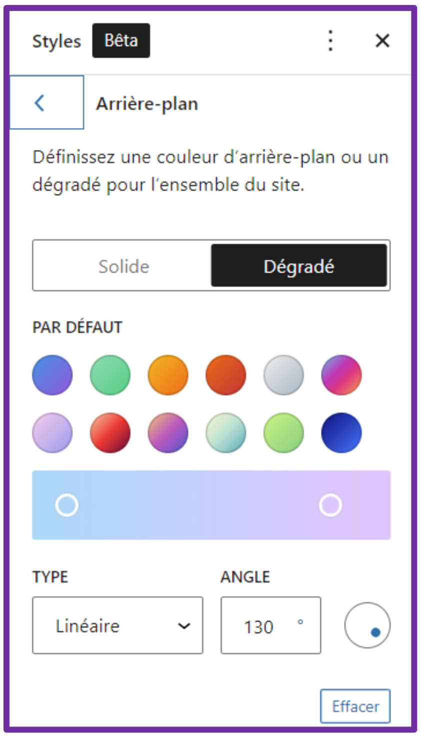
The way to make your website is very simple, it works in layers. As you can see on the left side of the screenshot, there are different layers that make up this website.



Tools will be available for you to add the elements you want.

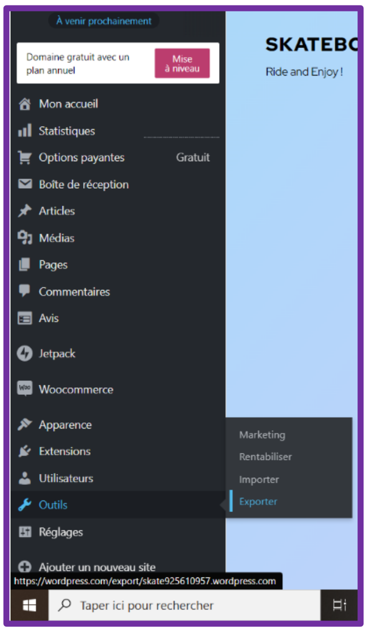
This right menu shows the frequently used components to make a website. If your search a particular component you can search it thanks to the search bar located at the top of the menu.

You can also customise your website by adding filters to your images, or by making a colour gradient on your page. In short you are free to express your creativity in a simple way

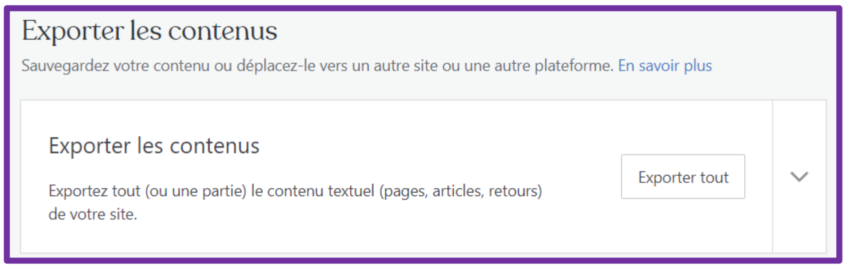


Définissez une couleur d'arrière-plan ou un dégradé pour l'ensemble du site.

Once done you can publish the website or you can export it. In order to export it go to the main dashboard. Then click on “Tools” and then click on “Export”.



Click on “Export all”.



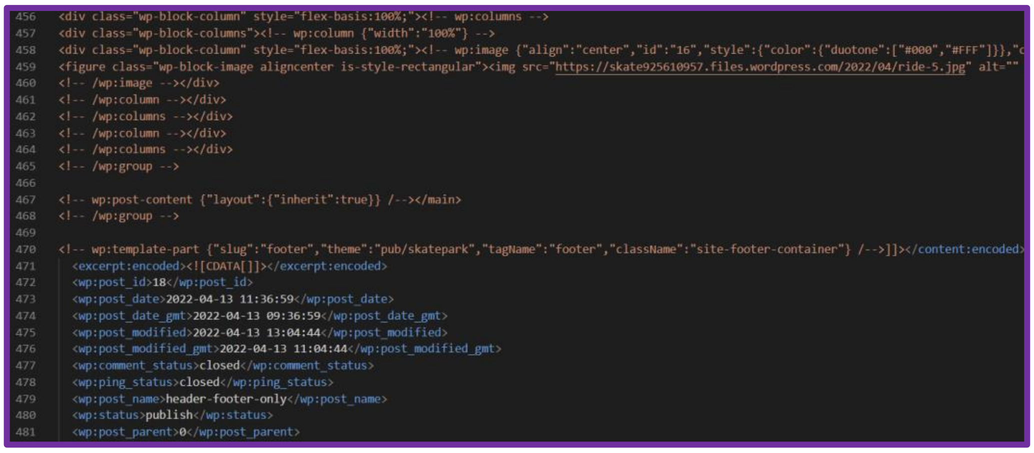
Exporter les contenus

Sauvegardez votre contenu ou déplacez-le vers un autre site ou une autre plateforme. En savoir plus

Exporter les contenus

Exportez tout (ou une partie) le contenu textuel (pages, articles, retours) de votre site.

Exporter tout



```
<div class="wp-block-column" style="flex-basis:100%;"><!-- wp:columns -->
<div class="wp-block-columns"><!-- wp:column {"width":"100%"} -->
<div class="wp-block-column" style="flex-basis:100%;"><!-- wp:image {"align":"center","id":"16","style":{"color":{"duotone":["#000","#FFF"]}},"c
<figure class="wp-block-image aligncenter is-style-rectangular"><img src="https://skate925610957.files.wordpress.com/2022/04/ride-5.jpg" alt=""
<!-- /wp:image --></div>
<!-- /wp:column --></div>
<!-- /wp:columns --></div>
<!-- /wp:column --></div>
<!-- /wp:columns --></div>
<!-- /wp:group -->

<!-- wp:post-content {"layout":{"inherit":true}} /--></main>
<!-- /wp:group -->

<!-- wp:template-part {"slug":"footer","theme":"pub/skatepark","tagName":"footer","className":"site-footer-container"} /-->]]></content:encoded>
  <excerpt:encoded><![CDATA[]]></excerpt:encoded>
  <wp:post_id>18</wp:post_id>
  <wp:post_date>2022-04-13 11:36:59</wp:post_date>
  <wp:post_date_gmt>2022-04-13 09:36:59</wp:post_date_gmt>
  <wp:post_modified>2022-04-13 13:04:44</wp:post_modified>
  <wp:post_modified_gmt>2022-04-13 11:04:44</wp:post_modified_gmt>
  <wp:comment_status>closed</wp:comment_status>
  <wp:ping_status>closed</wp:ping_status>
  <wp:post_name>header-footer-only</wp:post_name>
  <wp:status>publish</wp:status>
  <wp:post_parent>0</wp:post_parent>
```

This will send you a link to your email address. Click on that in order to do download your WordPress in a Zip file. All you have to do is extract it and write the XML code for your site. And Voila ! So as you can see it’s very easy. So now that we presented what is WordPress, it’s time to install it.

VI) WORDPRESS – INSTALLATION

Type the following commands in order to install WordPress.

First we will change the working directory. We will go in the temporary files directory where we are going to download the WordPress installation files archive which we will use later:

- *cd /tmp*
- *sudo wget https://wordpress.org/latest.zip*

WordPress stores data on database. So that's important to create a dedicate database for WordPress in the software MariaDB that we previously installed. So, we will enter in the databases as an administrator with a password authentication:

- *sudo mariadb -u root -p*
- Enter your password of MariaDB that you defined previsouly

Now your in the MariaDB console, type the following commands and adapt it to your situation. The word in blue are the word to be adapted for your situation. We insist again with the importance of the semi-colon, they are important. So make sure to do not ignore them.

Create a Database with the name you want, we choose "wordpress_db" for the following examples. But make sure you adapt it. Also, you will need the them in other steps of the installation. Make sure you don't forget them. We suggest you to write those informations in a piece of paper or in other way. Once created we will check if it's in the list of the databases. The database "wordpress_db" should appear in the list :

- *CREATE DATABASE wordpress_db;*
- *SHOW DATABASES;*

Once the database is successfully created, we need to create an administrator with a password. He is the one who will have privileges to handle data and their access. In our case, to illustrate the command we choose "admin_user_id" as the administrator user's id and "your_admin_super_password" for the administrator's password.

- *CREATE USER 'admin_user_id'@'localhost' IDENTIFIED BY 'your_admin_super_password';*
- *GRANT ALL PRIVILEGES ON wordpress_db.* TO admin_user_id@localhost;*
- *FLUSH PRIVILEGES;*
- *exit*

Once finished with database we have to unzip the archive that we recently download. To do so, we will install a package :

- *sudo apt-get update && sudo apt-get upgrade*
- *sudo apt install zip*

We will remove the apache default landing web page when we type “localhost” on the web browser. Instead we will install wordpress there and later manage the pages we want to land on.

- *sudo rm /var/www/html.index.html*
- *sudo unzip latest.zip -p /var/www/html*

This operation can take few times depending on your server speed. Once done we will go to the folder where we unzipped the archive. Then we will move the all the content of “wordpress” directory to the root directory of your website. Once done, the “wordpress” directory is empty so will delete it.

- *cd /var/www/html*
- *sudo mv wordpress/* /var/www/html/*
- *sudo rm wordpress/ -Rf*

After that, we have to change the access to these files in term of the website security.

- *sudo chown -R www-data:www-data /var/www/html/*
- *sudo chown -R www-data:www-data /var/www/riders*

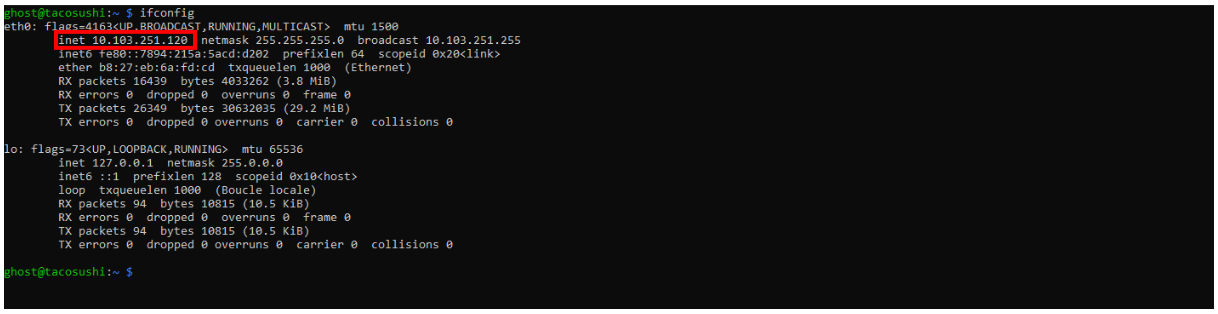
Once done we are going to configure wordpress. Before that, make sure you have a device with a desktop environment. It can be anything, MacBook, Windows or a Linux with desktop environment. You can even use your phone, but in term of visibility this is not recommended.

The first step is to find your server IP address. For that you will again have to type few commands on your server:

- *sudo apt install net-tools*

- *ifcongif*

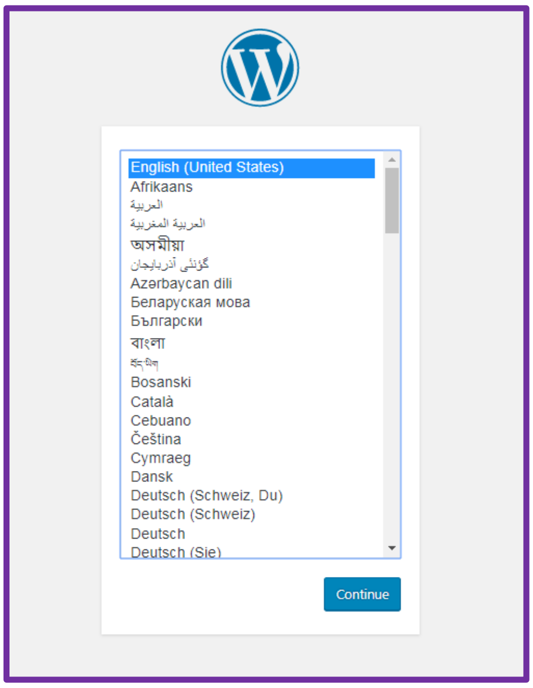
Note this address in a piece of paper or somewhere else, because it’s essential for the next step.



```
ghost@tacosushi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.103.251.120  netmask 255.255.255.0  broadcast 10.103.251.255
        inet6 fe80::7894:215a:5acd:d202  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:6a:fd:cd  txqueuelen 1000  (Ethernet)
        RX packets 16439  bytes 4033262 (3.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 26349  bytes 30632035 (29.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Boucle locale)
        RX packets 94  bytes 10815 (10.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 94  bytes 10815 (10.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ghost@tacosushi:~ $
```

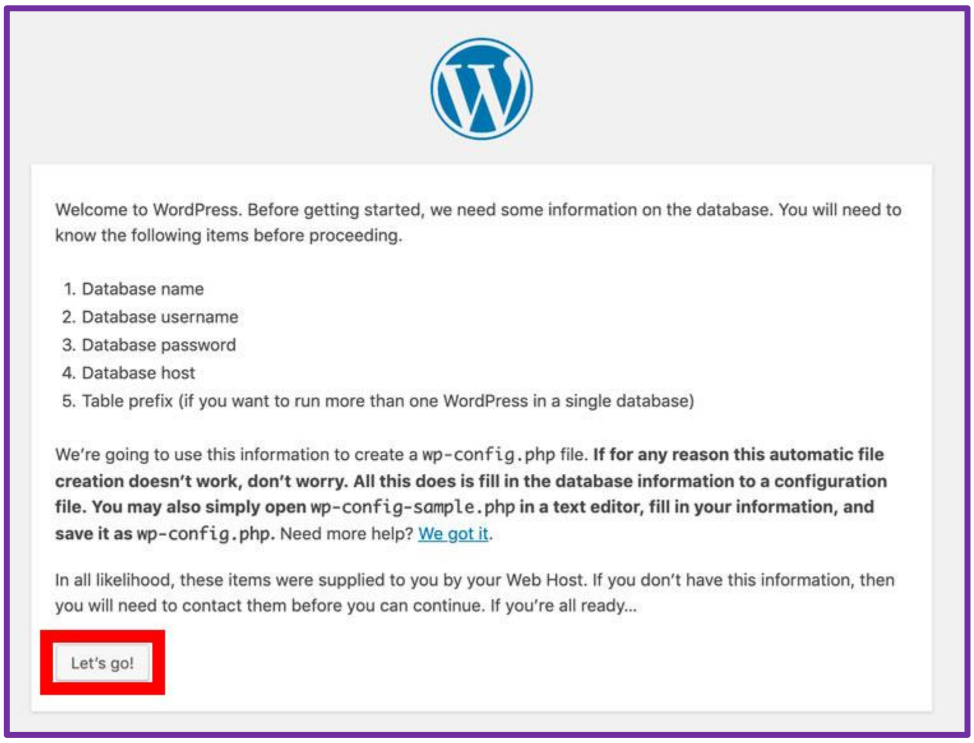
Once done, switch to you’re the computer which has a graphical interface.

Open your web browser and type the IP address you previously wrote.



You will reach a wordpress page which asks to select a language. Choose one, we recommend English as it’s the international language and used a lot in computer science.

Then you will reach this page. Press the “Let’s go !” button.



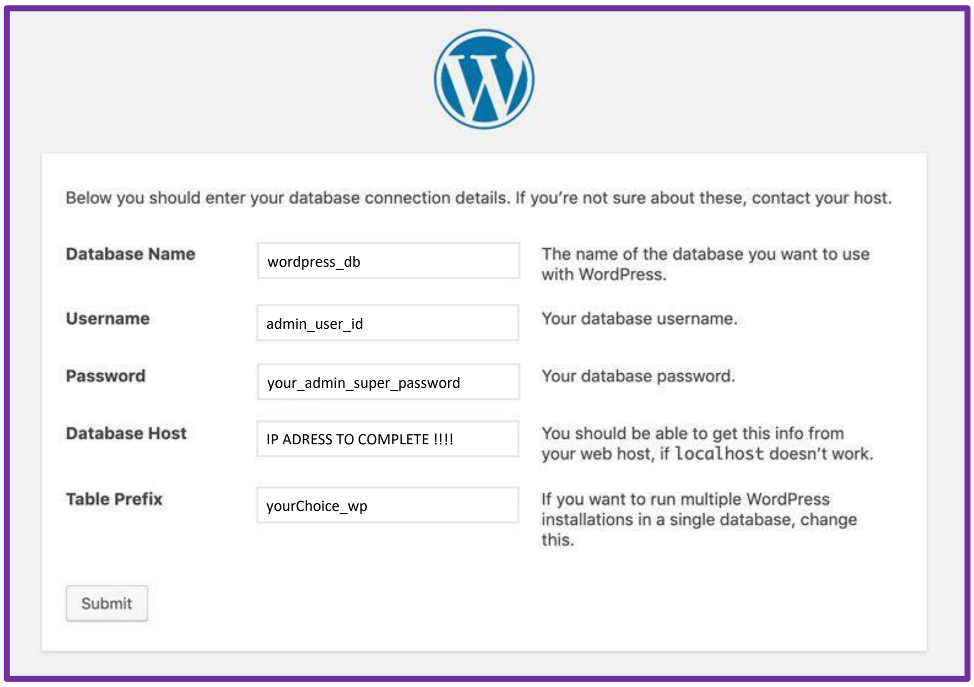
Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

We're going to use this information to create a `wp-config.php` file. **If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open `wp-config-sample.php` in a text editor, fill in your information, and save it as `wp-config.php`.** Need more help? We got it.

In all likelihood, these items were supplied to you by your Web Host. If you don't have this information, then you will need to contact them before you can continue. If you're all ready…

Once finished you will land on a page which asks for details on the database and the administrator. Be careful, some of the informations are those you used in the previous steps. So this is the time to use them. For the guide we are also going to us the same information we used in the previous steps.

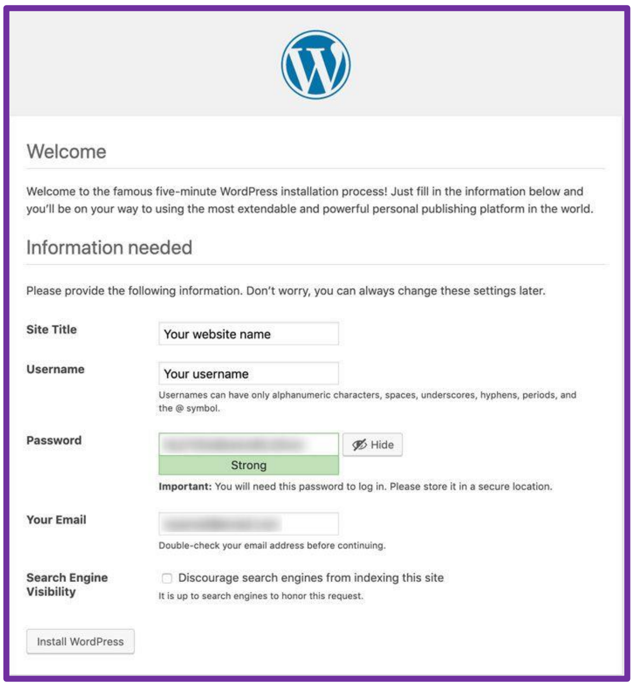


Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	wordpress_db	The name of the database you want to use with WordPress.
Username	admin_user_id	Your database username.
Password	your_admin_super_password	Your database password.
Database Host	IP ADRESS TO COMPLETE !!!!	You should be able to get this info from your web host, if `localhost` doesn't work.
Table Prefix	yourChoice_wp	If you want to run multiple WordPress installations in a single database, change this.

As you can see, the first three information are what u defined while configuring the database. The IP address is the one of your linux server which you also wrote. As it concerns the table prefix it is your choice. The default one is “wp_”. However this can be source of potential risks because it is known by everyone. So we kindly request to adapt it according to your situation.

Then you will have to complete another page with about the site informations. These informations are independent from the last ones. You can complete them as you want. Just make sure that you give a valid and reachable e-mail.



Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.

Site Title: Your website name

Username: Your username

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password: Hide — Strong

Important: You will need this password to log in. Please store it in a secure location.

Your Email:

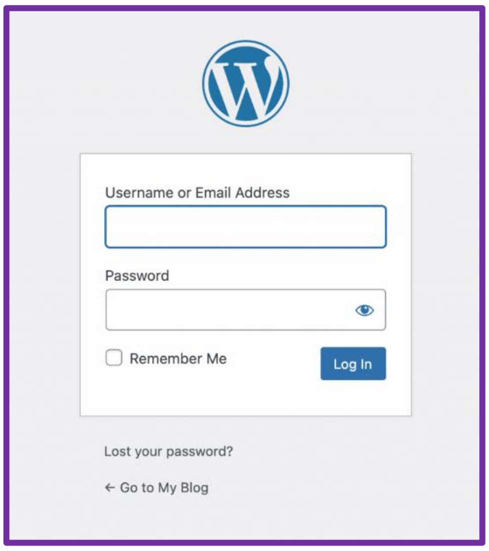
Double-check your email address before continuing.

Search Engine Visibility: ☐ Discourage search engines from indexing this site

It is up to search engines to honor this request.

Install WordPress

Congratulations ! You finished the installation of wordpress. All you have to do now is to login and create your pages. This what the login page of wordpress looks like.



Username or Email Address

Password

☐ Remember Me

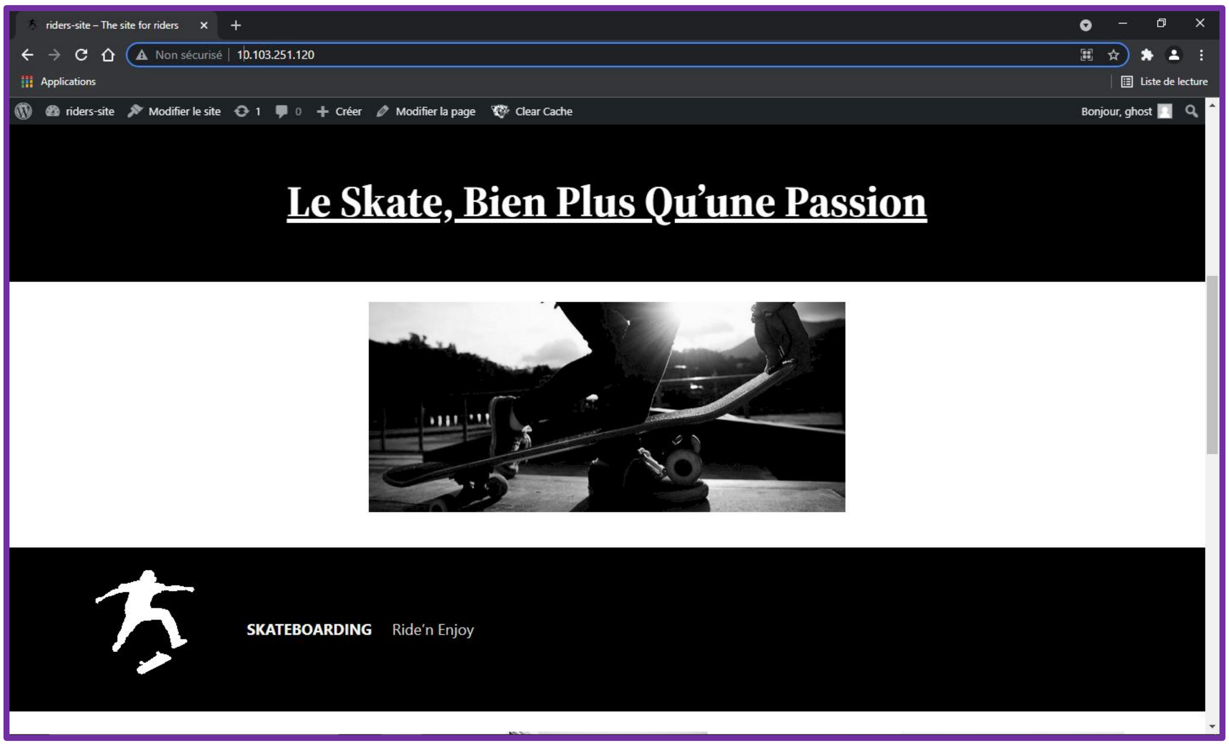
Log In

Lost your password?

← Go to My Blog

Once you have created your web page, you have option to publish it.

Then type your ip address in the browser search bar and you will reach your page.



VII) PROBLEMS IN THE FUTURE

A) FIREWALL

The safety is not sure. Indeed, your server will always be under danger situations. We illustrate the most common problems you can face after the installation and how to face them.

Well the first thing to think about is the fact that: if you want to access your website located on your server, you will have to connect your server to the internet. To be more specific you will have to connect to a box and then open a port. Once you did that your server is can be accessed by everyone if they know your IP address of your internet box. In today's world there are many hackers who can easily attack your server.

To avoid that we are going to configure a firewall. The process is simple, it plays the role of wall between the external network aka Internet and your local network like your home office etc. When configuring it, we can tell which IP address can access which devices on which port. To make it simple, the server is like an appartement and the ports are the several doors. You can change the default ports value If you judge the need.

There are some important default ports to know:

- Port 22 for the SSH
- Port 80 for HTTP
- Port 443 for HTTPS

We are just going to tell the firewall to allow external device to access our server from these ports only.

Follow these commands to install and configure it:

- *sudo apt-get install ufw*
- *sudo ufw enable*
- *sudo ufw allow 22*
- *sudo ufw allow 80*
- *sudo ufw allow 443*

B) HTTP AND HTTPS

The original link in the server is in the format http://. However, we are going to configure our server in order to have the format https://.

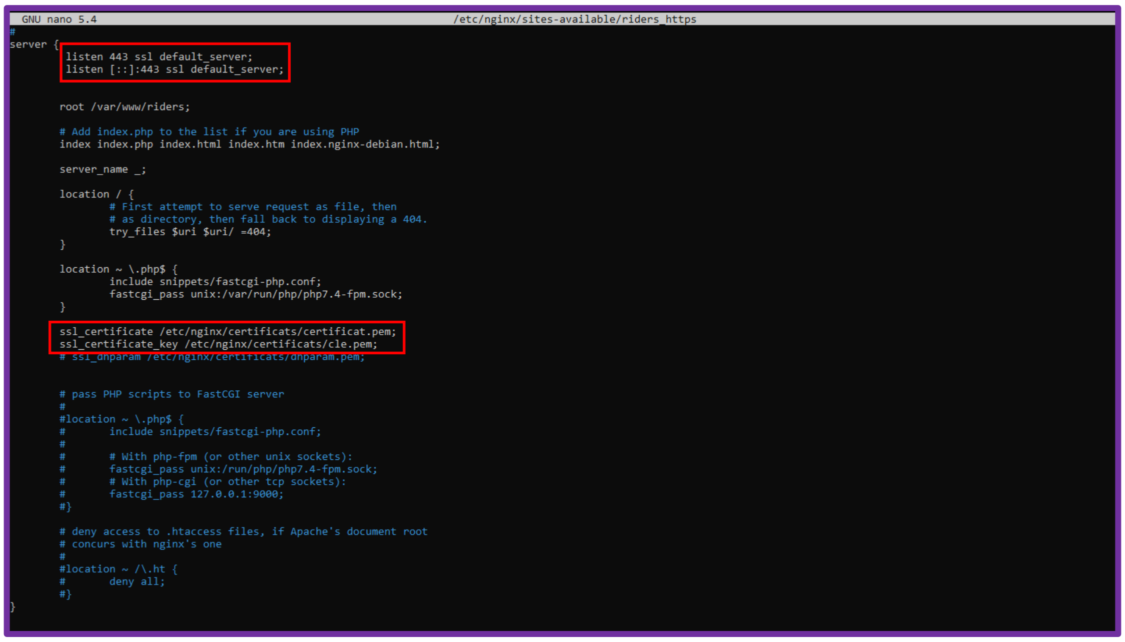
You might ask what are differences:

- Use the Transport layer instead of Application layer
- Data are encrypted
- Use port 443
- Require certificates

To do execute the following commands:

- *cd /etc/nginx/*
- *mkdir certificats*
- *cd certificats*

- *sudo openssl req -newkey rsa:4096 -sha512 -nodes -keyout cle.pem -x509 -days 365 -out certificat.pem*
- *sudo openssl x509 -text -noout -in certificat.pem*
- *sudo openssl pkcs12 -inkey cle.pem -in certificat.pem -export -out certificat.p12*
- *sudo openssl pkcs12 -in certificat.p12 -noout -info*
- *sudo cp -r riders riders_https*
- *sudo nano riders_https*



```
  GNU nano 5.4                                        /etc/nginx/sites-available/riders_https
#
server {
        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;


        root /var/www/riders;

        # Add index.php to the list if you are using PHP
        index index.php index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }

        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        }

        ssl_certificate /etc/nginx/certificats/certificat.pem;
        ssl_certificate_key /etc/nginx/certificats/cle.pem;
        # ssl_dhparam /etc/nginx/certificats/dhparam.pem;


        # pass PHP scripts to FastCGI server
        #
        #location ~ \.php$ {
        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/run/php/php7.4-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;
        #}

        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #
        #location ~ /\.ht {
        #       deny all;
        #}
}
```

C) BACKUP

We never know what can happen. Just imagine your server crash and that erase all your data from your storage device. You maybe just lost your storage device while using it somewhere else than your familiar place. In the worst case you might spilled liquid on your server and it doesn't turn on anymore.

These are only few examples. In all these cases you will just change the devices or the storage devices. However, if you don't have a backup you will have to restart all of the previous steps. Moreover, these steps are only for installation and configuration. Imagine you had plenty of websites on your server, this can be a serious problem.

Don't panic, here is the advice we recommend. In our case the client's storage device is a 8GB SD card so we are going to convert it in .iso or .dd format file. That way you can just write the image on your new SD card and have all the data. To do that you find many software but we will show the method we used: Linux command lines. To do that we used another computer running on a Linux distribution with a desktop environment. The location we want to save our image is: /home/my_username/backup and the SD card is the device on called sda.

To write create an image and compress it:

- *cd /home/my_username/backup*
- *sudo dd if=/dev/sda bs=8M status progress | gzip -9 –stdout > rpi.img.gz*

To write the image in the SD Card:

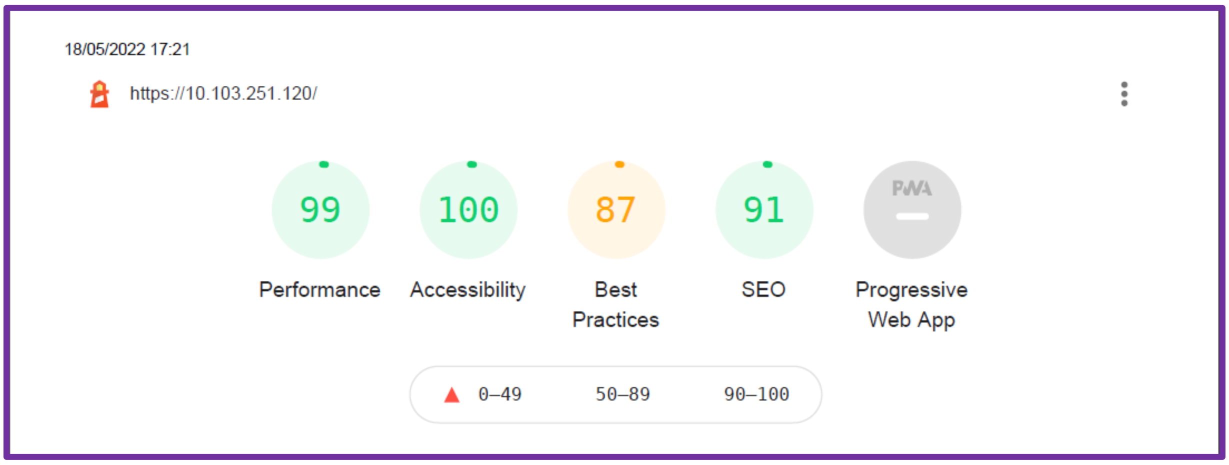
- *cd /home/my_username/backup*
- *sudo zcat rpi.img.gz | dd of=/dev/sda status progress*

VIII) DEPLOYMENT TEST

It's important to realise some test to make sure that the users will have a good experience on our website. It's also important to reduce the loading times, to make sure that the site works correctly. The more performant the site is, the more users will be able to enjoy it simultaneously.

We used LigthHouse, a Chrome tool to check the performance and quality of the website.

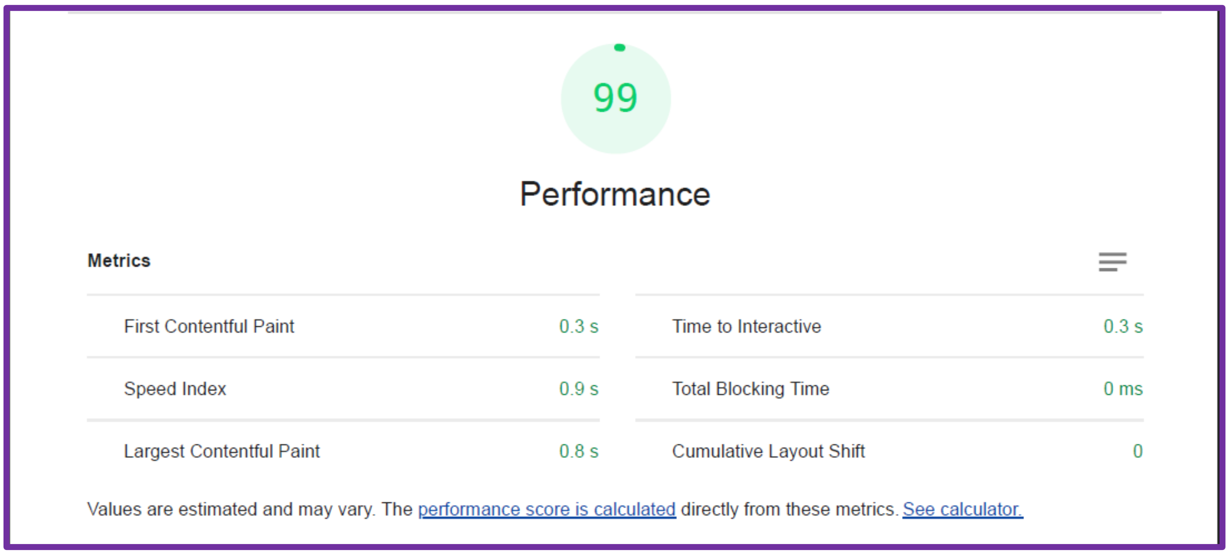
Here are the results:


18/05/2022 17:21
https://10.103.251.120/

Performance	Accessibility	Best Practices	SEO	Progressive Web App
99	100	87	91	PWA

▲ 0–49 ● 50–89 ● 90–100

All these results are further explained such as the performance:


99

Performance

Metrics

First Contentful Paint	0.3 s	Time to Interactive	0.3 s
Speed Index	0.9 s	Total Blocking Time	0 ms
Largest Contentful Paint	0.8 s	Cumulative Layout Shift	0

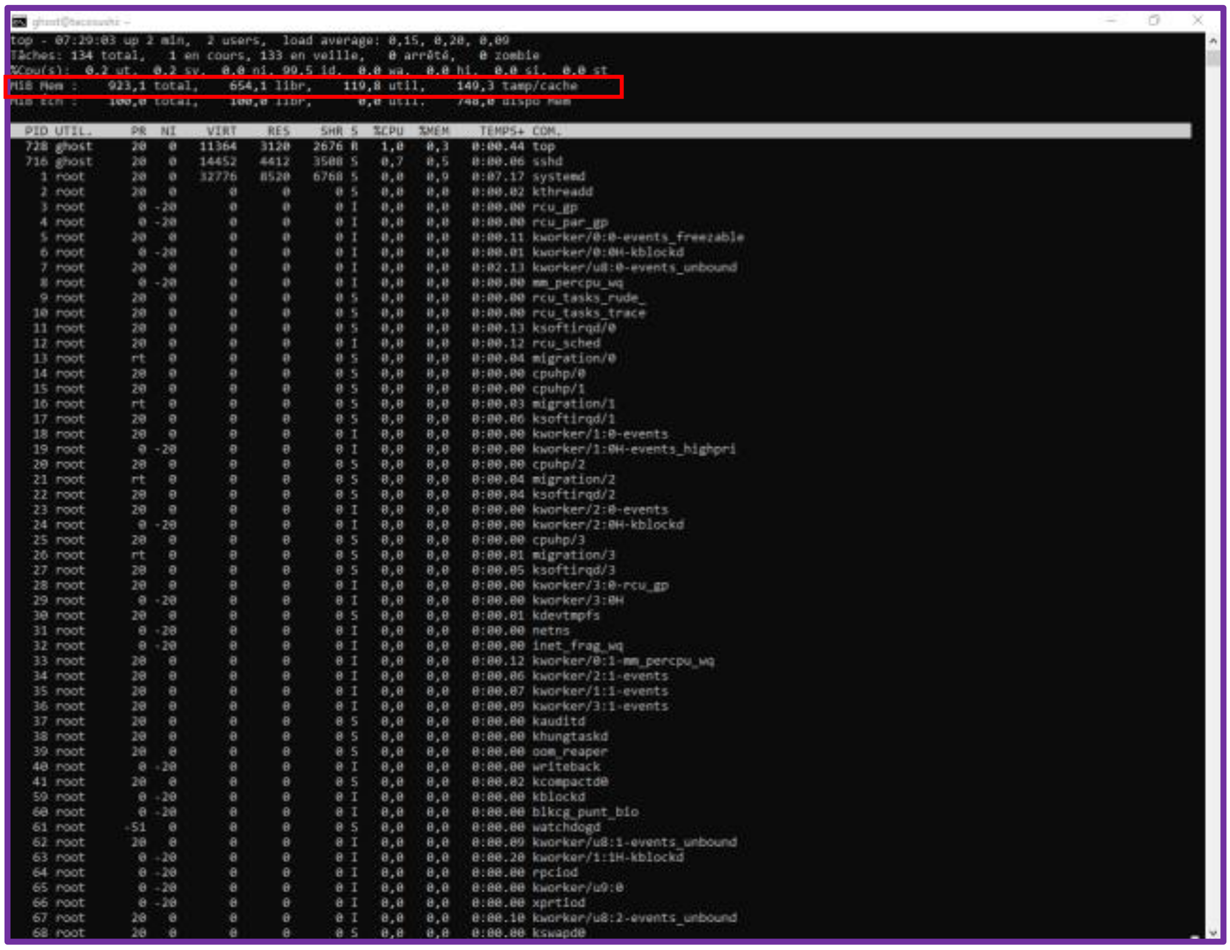
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

One of the main problems is that the website is not recognised as HTTPS. It’s actually not a problem because the request is encrypted as a normal HTTPS, however as we are using an SSL certificate, it’s not identified as a real certificate.

Thanks to the command

- *top*

We can obtain this screen. As you can see the server uses 119.8 MiB out of 923.1 MiB which is approximatively 13 % of the RAM



```
top - 07:29:03 up 2 min,  2 users,  load average: 0,15, 0,20, 0,09
Tâches: 134 total,   1 en cours, 133 en veille,   0 arrêté,   0 zombie
%Cpu(s):  0,2 ut,  0,2 sy,  0,0 ni, 99,5 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem :    923,1 total,    654,1 libr,    119,8 util,    149,3 tamp/cache
MiB Ech :    100,0 total,    100,0 libr,      0,0 util.    740,0 dispo Mem

  PID UTIL.     PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TEMPS+ COM.
  728 ghost     20   0   11364   3120   2676 R   1,0   0,3   0:00.44 top
  716 ghost     20   0   14452   4412   3508 S   0,7   0,5   0:00.06 sshd
    1 root      20   0   32776   8520   6768 S   0,0   0,9   0:07.17 systemd
    2 root      20   0       0      0      0 S   0,0   0,0   0:00.02 kthreadd
    3 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_par_gp
    5 root      20   0       0      0      0 I   0,0   0,0   0:00.11 kworker/0:0-events_freezable
    6 root       0 -20       0      0      0 I   0,0   0,0   0:00.01 kworker/0:0H-kblockd
    7 root      20   0       0      0      0 I   0,0   0,0   0:02.13 kworker/u8:0-events_unbound
    8 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 mm_percpu_wq
    9 root      20   0       0      0      0 S   0,0   0,0   0:00.00 rcu_tasks_rude_
   10 root      20   0       0      0      0 S   0,0   0,0   0:00.00 rcu_tasks_trace
   11 root      20   0       0      0      0 S   0,0   0,0   0:00.13 ksoftirqd/0
   12 root      20   0       0      0      0 I   0,0   0,0   0:00.12 rcu_sched
   13 root      rt   0       0      0      0 S   0,0   0,0   0:00.04 migration/0
   14 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/1
   16 root      rt   0       0      0      0 S   0,0   0,0   0:00.03 migration/1
   17 root      20   0       0      0      0 S   0,0   0,0   0:00.06 ksoftirqd/1
   18 root      20   0       0      0      0 I   0,0   0,0   0:00.00 kworker/1:0-events
   19 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/1:0H-events_highpri
   20 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/2
   21 root      rt   0       0      0      0 S   0,0   0,0   0:00.04 migration/2
   22 root      20   0       0      0      0 S   0,0   0,0   0:00.04 ksoftirqd/2
   23 root      20   0       0      0      0 I   0,0   0,0   0:00.00 kworker/2:0-events
   24 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/2:0H-kblockd
   25 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/3
   26 root      rt   0       0      0      0 S   0,0   0,0   0:00.01 migration/3
   27 root      20   0       0      0      0 S   0,0   0,0   0:00.05 ksoftirqd/3
   28 root      20   0       0      0      0 I   0,0   0,0   0:00.00 kworker/3:0-rcu_gp
   29 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/3:0H
   30 root      20   0       0      0      0 S   0,0   0,0   0:00.01 kdevtmpfs
   31 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 netns
   32 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 inet_frag_wq
   33 root      20   0       0      0      0 I   0,0   0,0   0:00.12 kworker/0:1-mm_percpu_wq
   34 root      20   0       0      0      0 I   0,0   0,0   0:00.06 kworker/2:1-events
   35 root      20   0       0      0      0 I   0,0   0,0   0:00.07 kworker/1:1-events
   36 root      20   0       0      0      0 I   0,0   0,0   0:00.09 kworker/3:1-events
   37 root      20   0       0      0      0 S   0,0   0,0   0:00.00 kauditd
   38 root      20   0       0      0      0 S   0,0   0,0   0:00.00 khungtaskd
   39 root      20   0       0      0      0 S   0,0   0,0   0:00.00 oom_reaper
   40 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 writeback
   41 root      20   0       0      0      0 S   0,0   0,0   0:00.02 kcompactd0
   59 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kblockd
   60 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 blkcg_punt_bio
   61 root     -51   0       0      0      0 S   0,0   0,0   0:00.00 watchdogd
   62 root      20   0       0      0      0 I   0,0   0,0   0:00.09 kworker/u8:1-events_unbound
   63 root       0 -20       0      0      0 I   0,0   0,0   0:00.20 kworker/1:1H-kblockd
   64 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rpciod
   65 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/u9:0
   66 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 xprtiod
   67 root      20   0       0      0      0 I   0,0   0,0   0:00.10 kworker/u8:2-events_unbound
   68 root      20   0       0      0      0 S   0,0   0,0   0:00.00 kswapd0
```

To have more informations about the performances, please consider the following pages which contains the full LightHouse report.