SAE S2.03

Installation Guide

Debian 12 Server

Lynn Hayot – B1

Table of Contents

Table of Figures

INTRODUCTION

This document is an installation guide for Debian on a virtual machine. After your reading, you should have a Debian 12 server with Apache, PostgreSQL and PHP installed and funcional. All of it should also be consultable from your host machine.

Here, we are going to install Debian 12 on the Qemu/KVM virtual machine, without Graphical User Interface. This means that all interactions with the machine will be performed by the shell, and the mouse pointer will be useless on the virtual machine.

In this tutorial, we will ask you to enter many commands in your shell. To facilitate your comprehension we will use a schema of colors for the different commands.

- The commands to enter in your Linux shell (on your host machine) will be written with a gray background:

```
$ code
```

- The cmmands to enter in your virtual machine shell will be written with a blue background:

```
$ code
```

A last precision, commands to enter as a root user will have a # at the beginning, other commands will have a $.

Before you start installing, make sure you have enough free space to install Debian (about 6Gb), and a stable connection. Then, you are ready to go !

INSTALL DEBIAN

I. Prepare the installation

First of all, you have to install Debian, and for that, you need the **ISO file** corresponding to the latest version. You can easily find this file at the following link: https://cdimage.debian.org/cdimage/release/current/amd64/iso-cd/

Here, we downloaded the net-installation for Debian 12.5.0.

Now you have to **check the integrity of the files**. For that, you have to compare your SHA-512 key with the one of the site (where you downloaded the ISO file), located in the file named “SHA512SUMS”. The two keys must be the same.

To see your SHA-512 file you can use this command in your shell:

$ sha512sum FILE_NAME

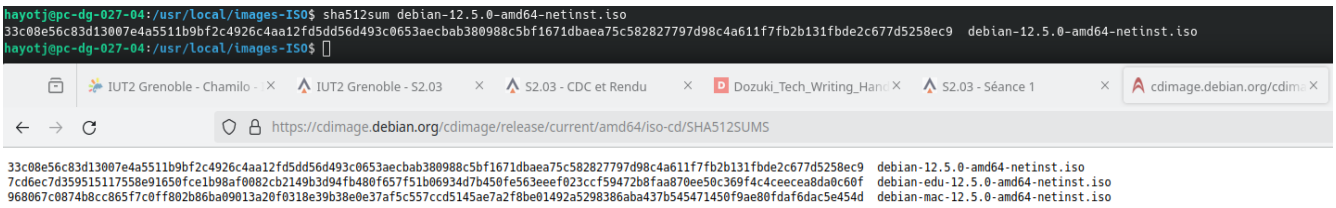


Figure 1: Verification of the two keys

II. Qemu’s parameters

Before installing Debian, you should check **Qemu’s parameters** used to launch the virtual machine. These parameters can be found in the command line of the qemu’s processus. This is an extract of my paramaters used to launch Qemu:

lance_qemu="qemu-system-x86_64 -machine q35 -cpu host -m 4G -enable-kvm -device VGA,xres=1024,yres=768 -display gtk,zoom-to-fit=off -drive $drive -device e1000,netdev=net0 -netdev user,id=net0,hostfwd=tcp::2222-:22,hostfwd=tcp::4443-:443,hostfwd=tcp::8080-:80,hostfwd=tcp::5432-:5432"

- qemu-system is the system of your machine
- 4G is the space used by the system of your virtual machine
- xres and yres are the dimensions of the screen of your virtual machine
- zoom-to-fit means that you can't zoom on your virtual machine
- hostfwd redirect the ports

Web service	Port on the VM	Port on your Linux station
SSH	22	2222
HTTP	80	8080
HTTPS	443	4443
PostgreSQL	5432	5432

III. Install the system

Now, we have a reliable and complete ISO file, we can start the **installation**. Launch your virtual machine with the appropriate command. Exemple of symbolic link:

```
$ S2.03-lance-installation
```

During the installation pass the steps, selecting parameters you want, in our context:

- Language : English
- Location : other/Europe/France
- Locales : United States, en_US.UTF-8
- Keyboard : English
- **Hostname** : *chose a server name*
- **Root Password** : *chose a password, here we will use 'root' . You can use "Show Password" to make sure you have entered the right password.*
- User Account - **Full Name** : *enter your entier name*
- **User Name** : *chose a pseudo/name to connect*

- **User Password** : *chose a password, here we will use ‘etu’ . You can use "Show Password" to make sure you have entered the right password*
- Partition disks : Guided - use entire disk
- Partition disks : All files in one partition
- Partition disks : Yes
- Software Selection : *"Debian desktop" must NOT be checked and "ssh server" must be checked*
- Install GRUB : Yes
- **Device for boot loader : /dev/sda**

  The other options were confirmed with the default selection.

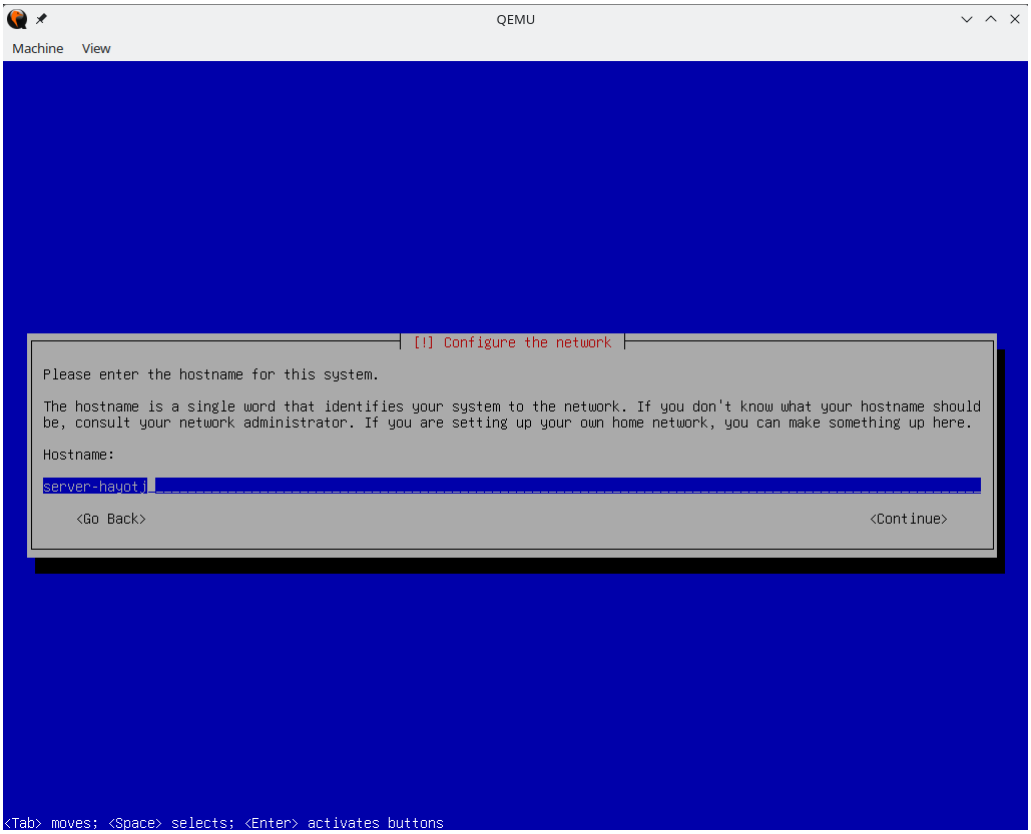


Figure 2: Enter username

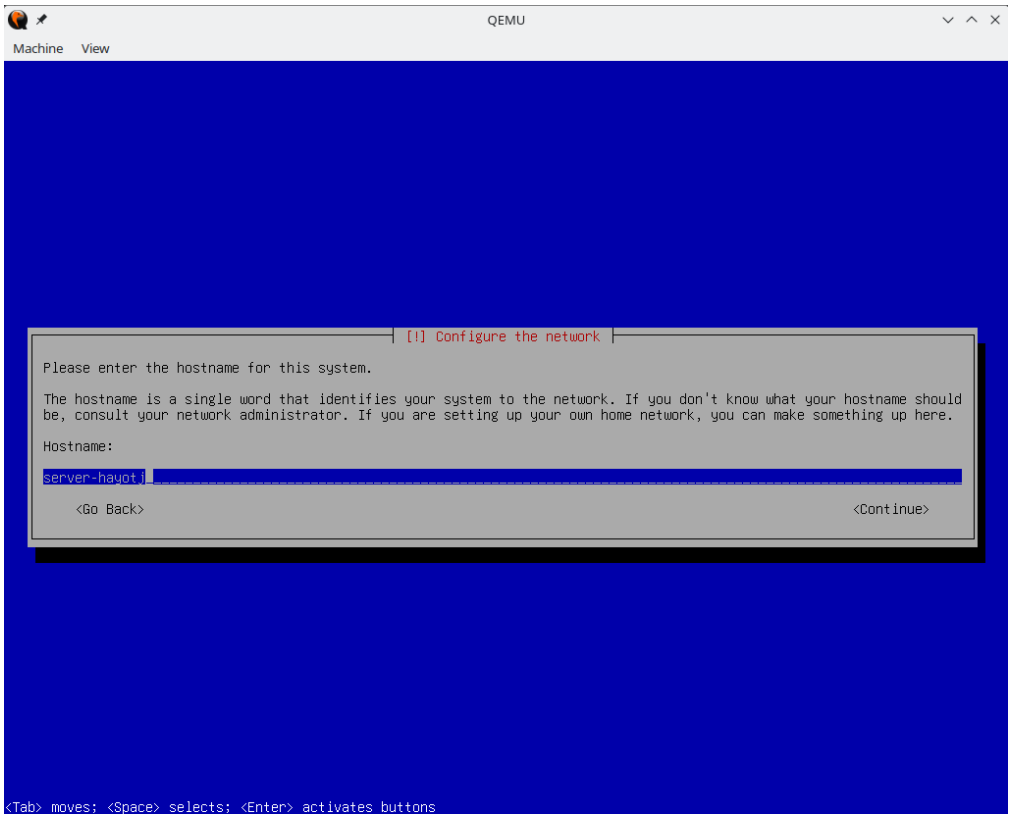


Figure 3: Enter hostname

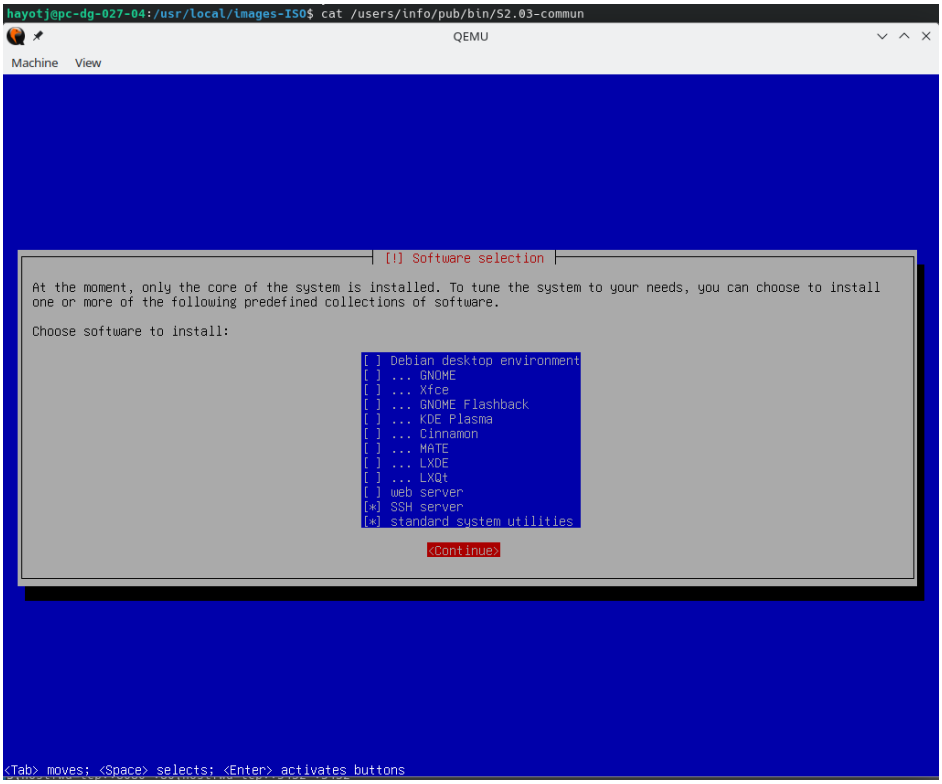


Figure 4: Software Selection

Once the system installed, you have to restart the virtual machine. Connect to a shell root and opwer off the machine.

```
$ su -
```

Enter your password (root)

```
# poweroff
```

IV. Move the disk image (optional)

If you want you can move your disk image to have an **easier acces** to your virtual machine. This step is optional. Now your disk image is placed in the local disk of your Linux machine. In this guide, we want to move it in our server erebus-4. Find the appropriate command on your machine and move it. Exemple of symbolic link:

```
$ S2.03-deplace-image-disque-sur-erebus4
```

V. Verifications

Now, your Debian server should be installed and funcional. Let's check it!

- Launch your virtual machine.
- **IP and Ethernet caracteristics**:

  You may be able to reach an external machine. Here is a few lines of commands to check quickly :

  ```
  $ ip addr
  ```

  ```
  $ ip neigh
  ```

  ```
  $ traceroute MACHINE_NAME
  ```

- **Xorg server's absence** :

  ```
  $ dpkg -l | grep xorg
  ```

```
hayotj@server-hayotj:~$ dpkg -l | grep xorg
hayotj@server-hayotj:~$ _
```

Figure 5: Xorg absence

- You can look at some specifications:

$ cat /etc/fstab

```
hayotj@server-hayotj:~$ cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# systemd generates mount units based on this file, see systemd.mount(5).
# Please run 'systemctl daemon-reload' after making changes here.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=193a95d0-e2d8-4c3b-8bee-1a14c085d029 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=af227380-a8ec-4556-a43e-74e1f35a2aa2 none            swap    sw              0       0
/dev/sr0        /media/cdrom0   udf,iso9660 user,noauto     0       0
hayotj@server-hayotj:~$
```

Figure 6: etc/fstab

- **Access via an external machine** :

$ su -

(you can use su – USER_NAME to return at user shell)

systemctl status ssh

```
root@server-hayotj:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-05-28 08:42:15 CEST; 13min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 485 (sshd)
      Tasks: 1 (limit: 4645)
     Memory: 6.7M
        CPU: 26ms
     CGroup: /system.slice/ssh.service
             └─485 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

May 28 08:42:15 server-hayotj systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
May 28 08:42:15 server-hayotj sshd[485]: Server listening on 0.0.0.0 port 22.
May 28 08:42:15 server-hayotj sshd[485]: Server listening on :: port 22.
May 28 08:42:15 server-hayotj systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
```

Figure 7: status ssh

Exemples of ssh and http utilisation:

$ ssh toto@localhost -p 2222

http://localhost:8080/ (in your web browser)

Congrates! You have a funcional virtual machine accessible by your Linux station.

INSTALL SOFTWARES

In this part of the guide, we will install the softwares Apache, Postgresql and PHP. We will use the APT method (Advanced Package Tool) to install the different packages to make things easier. *But, you can use an other method if you want and know how to do it.*

I. Install Apache

If you want to install Apache on your virtual machine, you can look at the Apache documentation. But if you want a faster or easier way, you just need to **follow the following instructions:**

- Connect to root user

```
$ su -
```

- Install and activate the software

```
# apt install apache2
```

```
# service apache2 start
```

- Verify that your Apache is started ("**Active**")

```
# service apache2 start
```

```
# systemctl status apache2
```

```
root@server-hayotj:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-05-28 08:35:13 CEST; 2min 1s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 977 (apache2)
      Tasks: 55 (limit: 4645)
     Memory: 9.4M
        CPU: 43ms
     CGroup: /system.slice/apache2.service
             ├─977 /usr/sbin/apache2 -k start
             ├─979 /usr/sbin/apache2 -k start
             └─980 /usr/sbin/apache2 -k start

May 28 08:35:13 server-hayotj systemd[1]: Starting apache2.service - The Apache HTTP Server...
May 28 08:35:13 server-hayotj apachectl[976]: AH00558: apache2: Could not reliably determine the server's fully qualified domai
May 28 08:35:13 server-hayotj systemd[1]: Started apache2.service - The Apache HTTP Server.
lines 1-16/16 (END)
```

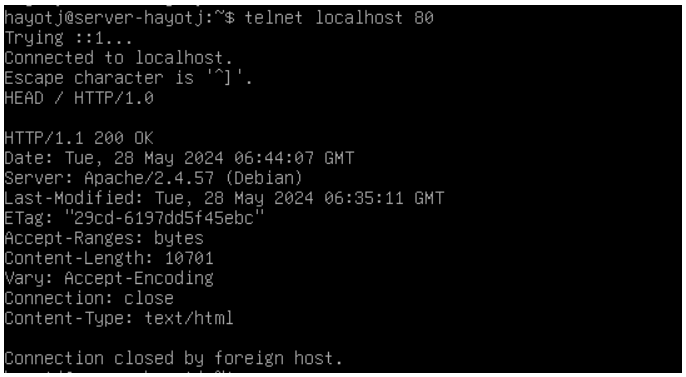
Figure 8: status apache

- Verify Telnet is installed and started ("**Active**")

(# apt install telnet) if telnet is not found

systemctl status telnet

- Connect to Apache server with telnet



```
hayotj@server-hayotj:~$ telnet localhost 80
Trying ::1...
Connected to localhost.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Tue, 28 May 2024 06:44:07 GMT
Server: Apache/2.4.57 (Debian)
Last-Modified: Tue, 28 May 2024 06:35:11 GMT
ETag: "29cd-6197dd5f45ebc"
Accept-Ranges: bytes
Content-Length: 10701
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

Connection closed by foreign host.
```

Figure 9: status telnet

- Access to Apache default page from your Linux Station

$ firefox http://localhost:8080

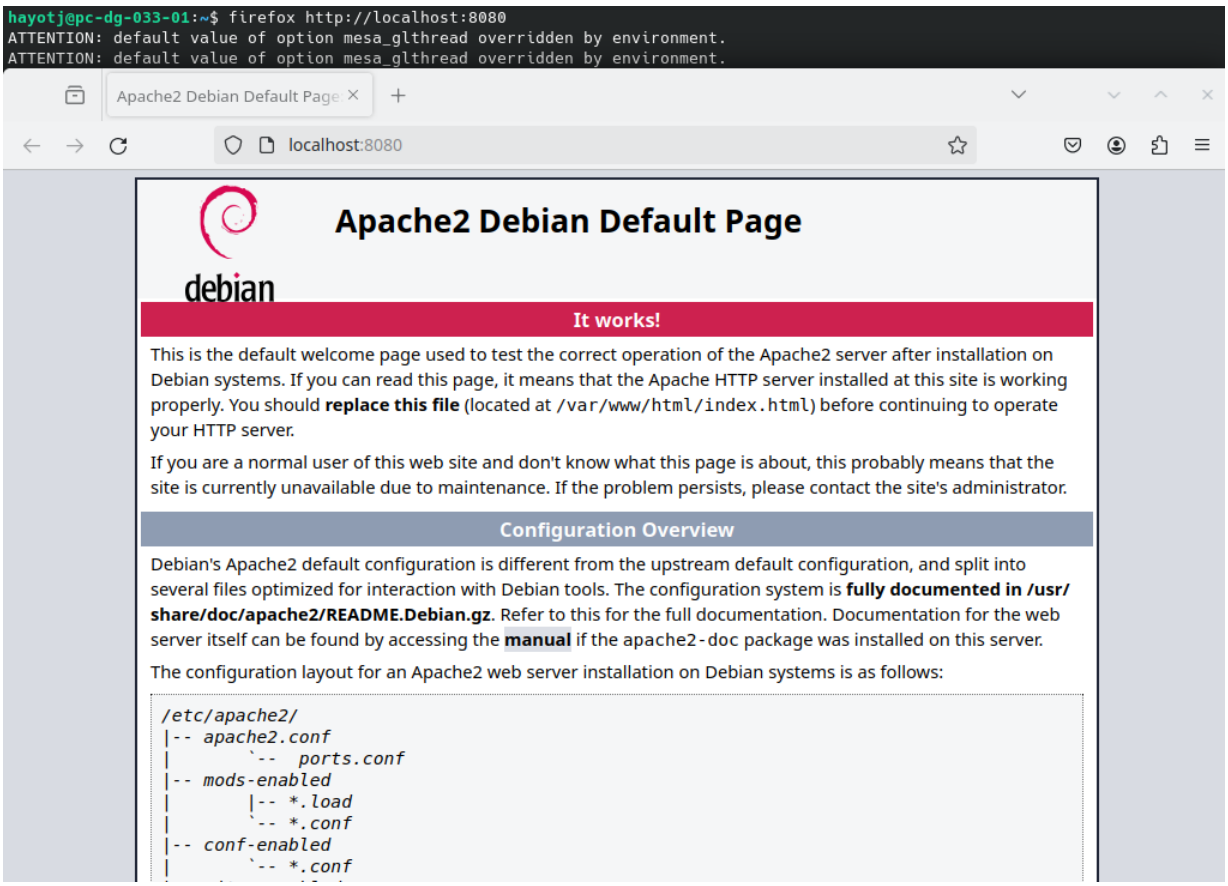


Figure 10: default-page apache

If all is **active** and you **success access to Apache default page** without problem, you can continue your installation with the **next part of the guide**. Else, you should retry the previous instructions and make sure you did not forget one. If it doesn't work, the better solution is to ask for help someone who is more experimented.

II. Install PostgreSQL

To **install PostgreSQL** you can follow the next instructions. In case that you need more help in the future, I give you the official site of PostgreSQL.

1) Install PostgreSQL on your virtual machine

- You need to install the package with apt with the root user (su -):

```
# apt install postgresql
```

- You can verify your installation with a connection to the postgres user

```
# su – postgresql
```

- Make sure your software is active:

```
# systemctl status postgresql
```

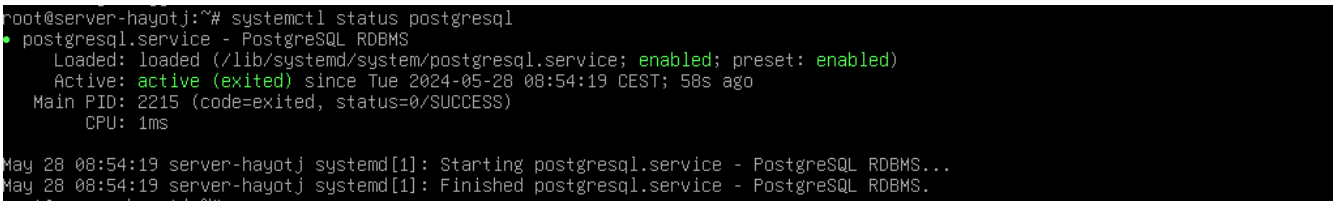


Figure 11: status postgresql

2) Create your user profile on postgres and your own database

- Check that your database does not exist yet, you can see all the databases created by default with the command:

```
$  psql -l
```

or

\l (connected to postgres user)

```
postgres@server-hayotj:~$ psql -l
                                                List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    | ICU Locale | Locale Provider |   Access privileges
-----------+----------+----------+-------------+-------------+------------+-----------------+-----------------------
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            | =c/postgres          +
           |          |          |             |             |            |                 | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            | =c/postgres          +
           |          |          |             |             |            |                 | postgres=CTc/postgres
(3 rows)
```

Figure 12: databases

- Connect to PostgreSQL from your user shell with the command:

  $ psql postgres

  The postgres user have all the permissions on the databases and users, use it when you want to create new user or database, or give them new permissions.

- Now you are connected as postgres user, you need to **create a new user profile** for you. Here, we are going to define a password for our user, but there are many other options to create a user, if you want to know them, you can look at the PostgreSQL Documentation. The most important thing is that **you must use your user name for your new PostgreSQL user**. It's possible to use an other name for your postgreSQL user but it will be easier to use your user name.

  CREATE USER hayotj WITH password 'etu';

- Let's going to **create your first database** ! You already connect to postgres user so you just need to creat your database with the name you chosed, and the user you created as owner. We will name our database 'base':

  CREATE DATABASE base WITH OWNER=hayotj;

3) Try PostgreSQL

Now you can use PostgreSQL from your virtual machine. Once you created your user and your database, you can disconnect with the command:

\q

To make sure the previous steps were efficient, you should try to make some requests in your database. Then, return to your user shell and try to connect to your database :

su – hayotj

$ psql base

If you don't encounter any problem, you should be able to create tables, and manage your data. I give you here some exemples of what you can do.

- Create my table

  CREATE TABLE personnages (nom varchar primary key, element varchar, categorie varchar, etoiles int);

- Verify the presence of my table

  \d

- Insert some lines of database

  INSERT INTO Personnages VALUES('Kazuha', 'Anemo', 'Support', 5);

  INSERT INTO Personnages VALUES('Hu Tao', 'Pyro', 'DPS', 5);

  INSERT INTO Personnages VALUES('Chongyun', 'Cryo', 'SubDPS', 4);

```
base=> \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 hayotj    |                                                            | {}
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

base=> CREATE TABLE Personnages(nom varchar primary key, element varchar, categorie varchar, etoiles int);
CREATE TABLE
base=> \d
             List of relations
 Schema |    Name     | Type  | Owner
--------+-------------+-------+--------
 public | personnages | table | hayotj
(1 row)

base=> INSERT INTO Personnages VALUES ('Kazuha', 'Anemo', 'support', 5);
INSERT 0 1
base=> INSERT INTO Personnages VALUES ('Hu Tao', 'Pyro', 'DPS', 5);
INSERT 0 1
base=> INSERT INTO Personnages VALUES ('Chongyun', 'Cryo', 'SubDPS', 4);
INSERT 0 1
base=> SELECT nom, etoiles FROM Personnages;
   nom    | etoiles
----------+---------
 Kazuha   |       5
 Hu Tao   |       5
 Chongyun |       4
(3 rows)

base=> SELECT nom FROM Personnages WHERE Categorie = 'DPS';
  nom
--------
 Hu Tao
(1 row)

base=> SELECT nom, element FROM Personnages WHERE etoiles = 5;
  nom   | element
--------+---------
 Kazuha | Anemo
 Hu Tao | Pyro
(2 rows)

base=> INSERT INTO Personnages VALUES ('Kazuha', 'Pyro', 'support', 5);
ERROR:  duplicate key value violates unique constraint "personnages_pkey"
DETAIL:  Key (nom)=(Kazuha) already exists.
```

Figure 13: sql from VM

4) Enable access to PostgreSQL from your Linux machine

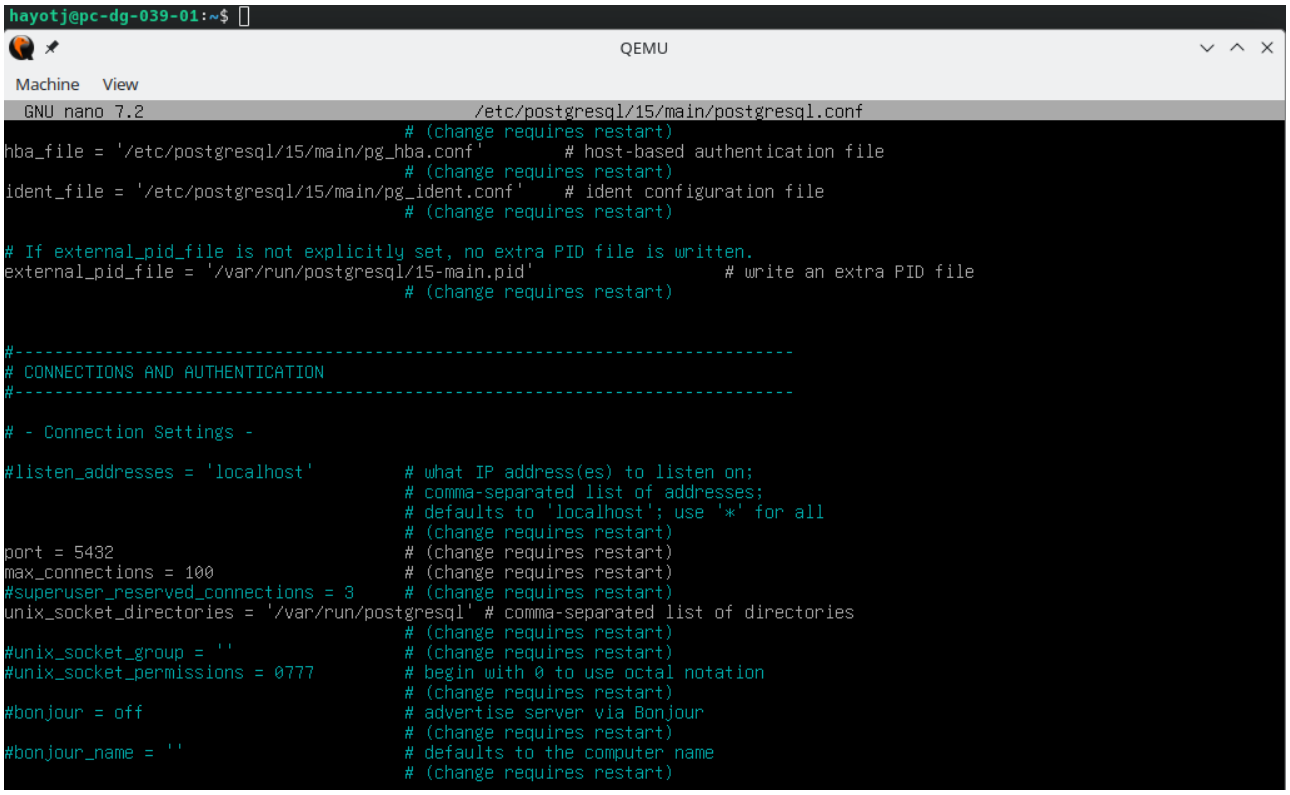
Once you have a funcional PostgreSQL, you can **make your databases accessible from you Linux machine**. For that, you will must modify **two configuration files** and restart your postgre server.

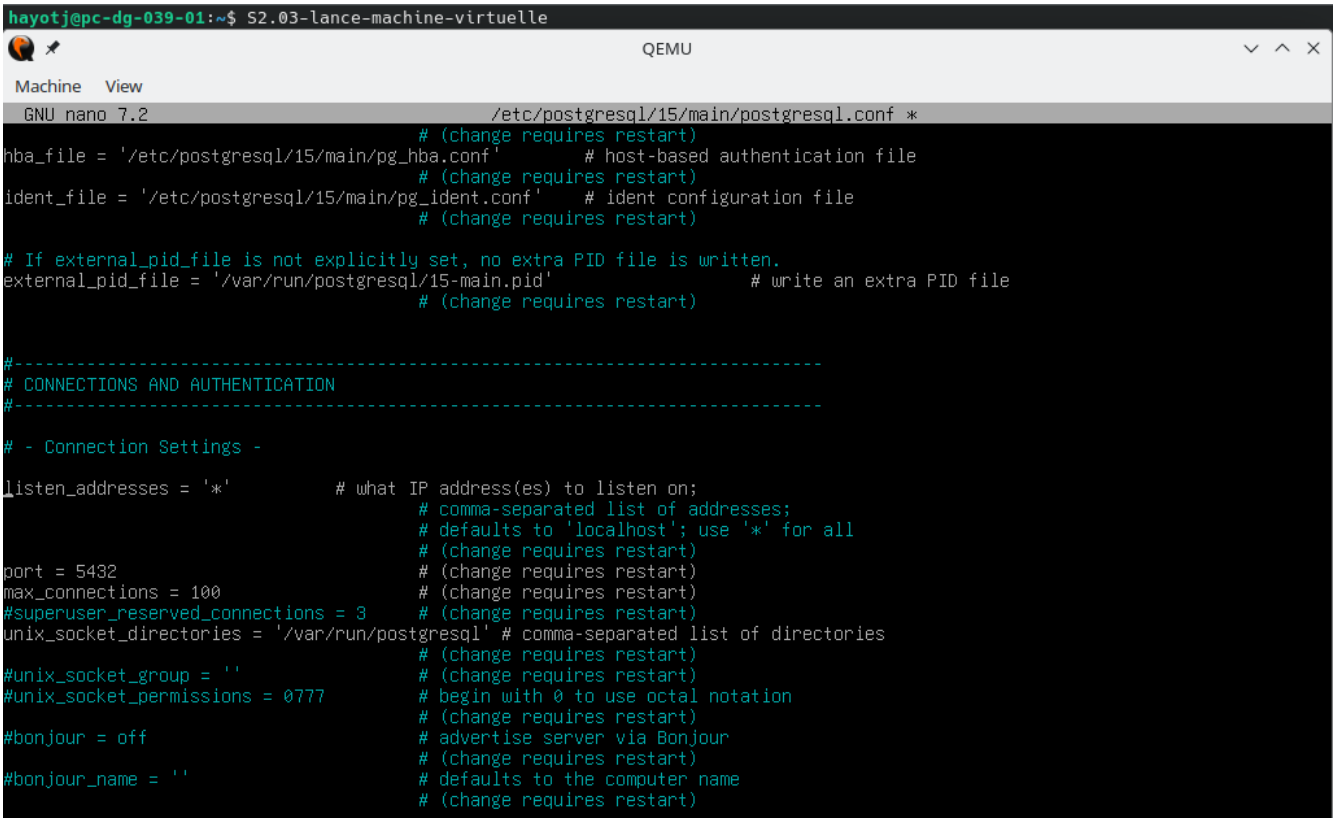
- First, you have to **modify the file named "postgresql.conf"**, you can use the text-editor nano.

```
# nano /etc/postgresql/15/main/postgresql.conf
```

In this file, you have only one line to modify, you can find it with grep:

grep listen_adress





You have to discomment out the line and replace all the text inside the quote by *. Then you can save with ctrl s and exit with ctrl x.

- Now, the server listen to IP adresses non-local, but you need to define an authentification rule, so you must **modify a second file**:

```
# nano /etc/postgresql/15/main/pg_hba.conf
```

Here you just need to add these lines:

#IPv4 remote connections:

host all all 0.0.0.0/0 scram-sha-256

Save and exit

- **Confirm these changes** with the command:

service postgresql restart

Now, you should be able to connect to your database from your Linux station with the same user profile, try it:

```
$ psql -h localhost base
```

```
hayotj@pc-dg-025-15:~$ psql -h localhost base
Password for user hayotj:
psql (15.7 (Debian 15.7-0+deb12u1), server 15.6 (Debian 15.6-0+deb12u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

base=> \d
           List of relations
 Schema |    Name     | Type  | Owner
--------+-------------+-------+--------
 public | personnages | table | hayotj
(1 row)

base=> SELECT nom FROM Personnages;
    nom
-----------
 Kazuha
 Hu Tao
 Chongyun
 Alhaitham
(4 rows)

base=> INSERT INTO Personnages VALUES ('Chlorinde', 'Electro', 'DPS', 5);
INSERT 0 1
base=> SELECT * FROM Personnages;
    nom    | element | categorie | etoiles
-----------+---------+-----------+---------
 Kazuha    | Anemo   | support   |       5
 Hu Tao    | Pyro    | DPS       |       5
 Chongyun  | Cryo    | SubDPS    |       4
 Alhaitham | Dendro  | DPS       |       5
 Chlorinde | Electro | DPS       |       5
(5 rows)
```

Figure 14: sql from Linux

You can also **verify if your password is correctly hidden** by the sha- 256 method. For that you must connect to the default database postgres and select the contenu of the table pg_shadow.

```
# psql postgres
```

SELECT * FROM pg_shadow;

```
postgres@server-hayotj:~$ psql postgres
psql (15.6 (Debian 15.6-0+deb12u1))
Type "help" for help.

postgres=# SELECT * FROM pg_shadow;
 usename  | usesysid | usecreatedb | usesuper | userepl | usebypassrls |
        passwd                                                        | valuntil | useconfig
----------+----------+-------------+----------+---------+--------------+-------------------------------------------------------
-----------------------------------------------------------------------+----------+-----------
 postgres |       10 | t           | t        | t       | t            |
                                                                       |          |
 hayotj   |    16388 | f           | f        | f       | f            | SCRAM-SHA-256$4096:pBv2CCnF4wLryO40wEnKIQ==$FpYTekYNH8P
oUoXn4vzmOXV/Me4GmwqwI2DdNqBrS3U=:FqDJv53ABKhdOErY2Mknjn6x7In7qnvNkFz7rM2gsaQ= |          |
(2 rows)
```

Figure 15: pg_shadow

III. Install PHP

It's time to install PHP, you ofcourse can consult Installation Guide for PHP, but you just need to install the package and test it Let's go !

- First, **install the package** like all the others before:

```
# apt install php
```

To test, we need to create a new file and add code in it.

- Create the file

```
$ touch /var/www/html/info.php
```

- Edit it

```
$ nano /var/www/html/info.php
```

- Add the code

```
<?php
phpinfo();
phpinfo(INFO_MODULES);
?>
```

- Try your installation **following this link** from your linux browser:

http://localhost:8080/info.php

IV. Install PHPPgAdmin

First, congratulation on reaching this point. Here is **the last step** of our tutorial. We are going to install PHPPgAdmin.

- Start by installing the package:

```
# apt install phppgadmin
```

- Edit the file named "Connection.php"

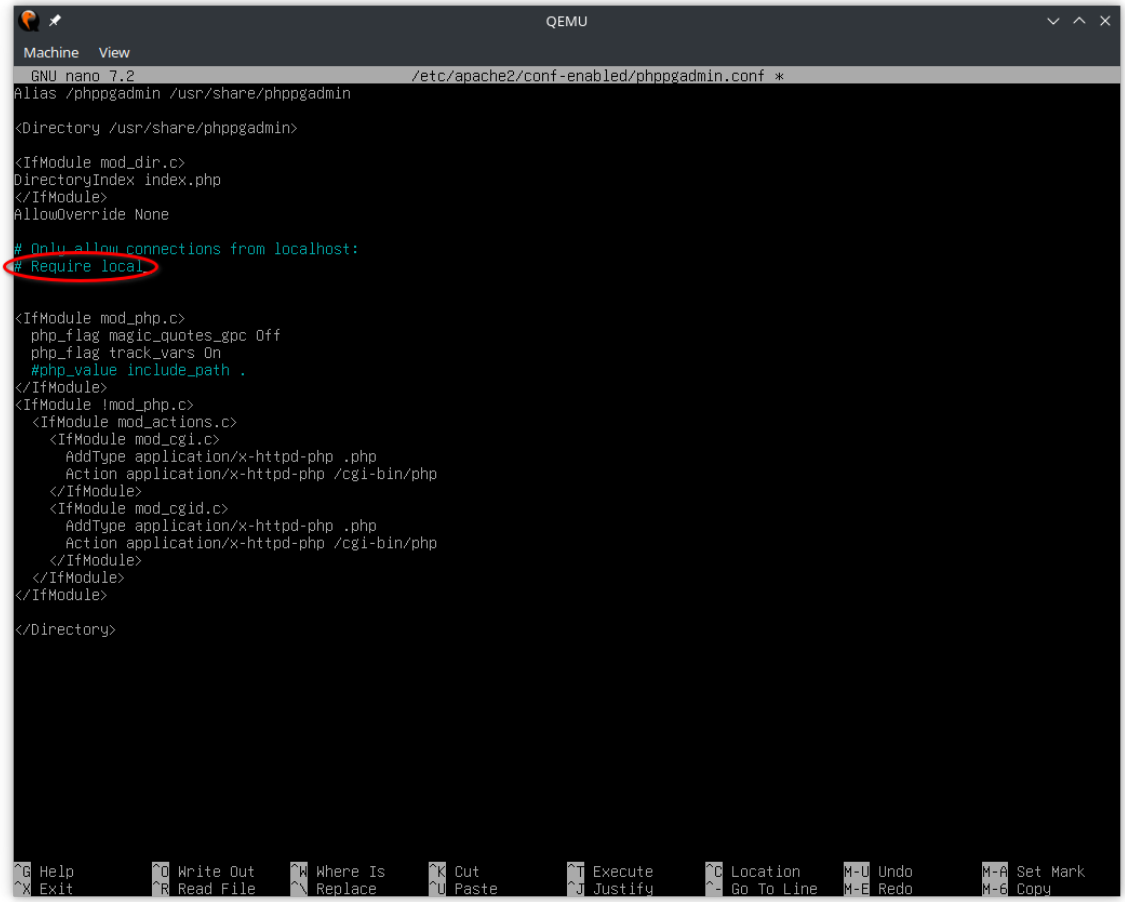
```
# nano /usr/share/phppgadmin/classes/database/Connection.php
```

You have to replace the line "case '14': return 'Postgres';break;" by "case '15': return 'Postgres';break;"

- Edit an other configuration file

```
# nano /etc/apache2/conf-available/phppgadmin.conf
```

The line “Require local” has to be replaced by “Allow from all”



- Restart Apache:

```
# service apache2 restart
```

- Finally, try to access the following page from your Linux station:

  http://localhost:8080/phppgadmin

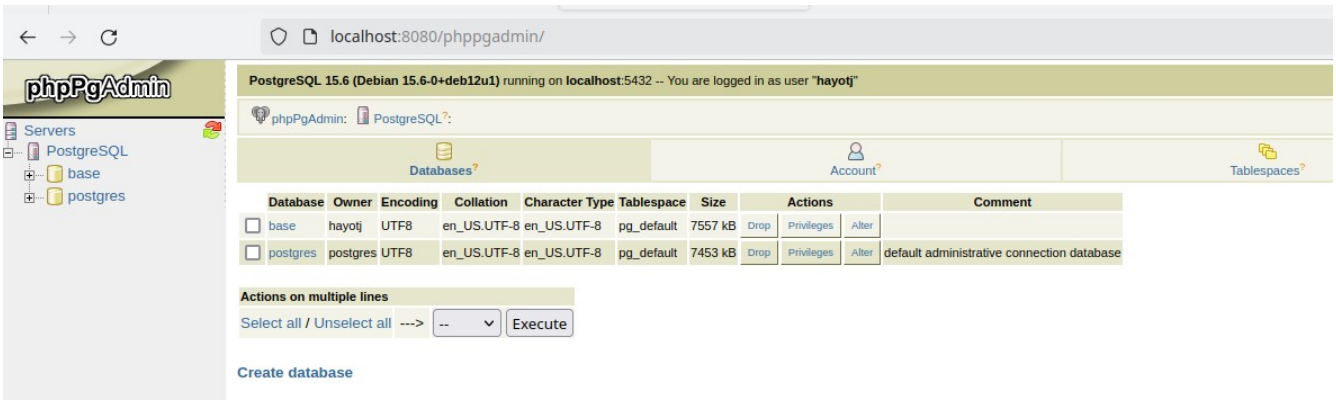


Figure 16: sql from pgadmin

If all is set, you finished your installation. Congratulations! Now you have a virtual machine with a funcional Debian server, useful softwares, a PostgreSQL user profile and a direct access to your database from your linux station.

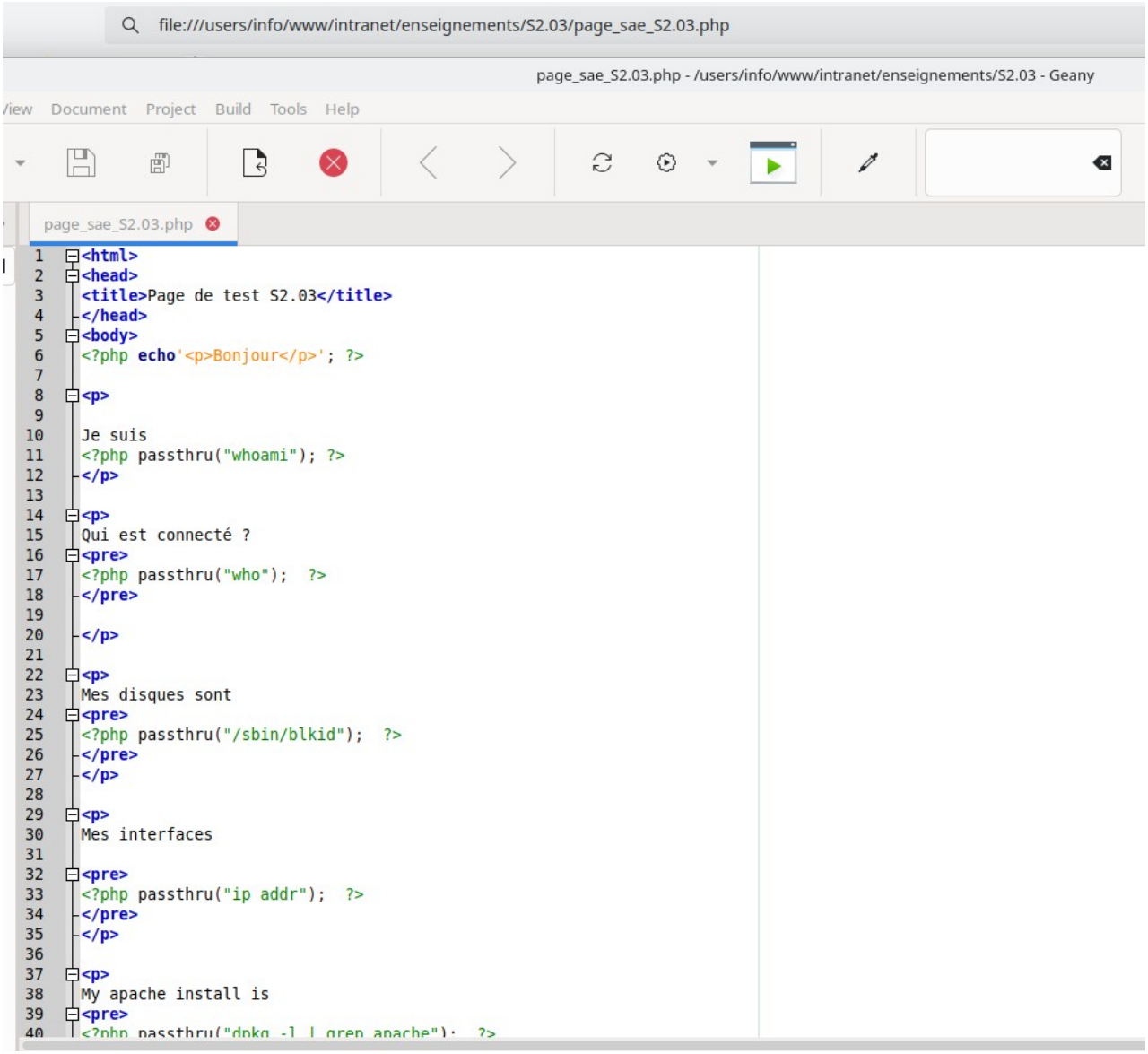
CONCLUSION

Just a little test for the end, you should be able to enter the command...

```
# sbin/sblkid
```

...and see the following page from your host machine:

/users/info/www/intranet/enseignements/S2.03/page_sae_S2.03.php



Let’s see how these installation took our space:

```
$ du -h
```

```
$ df -h
```

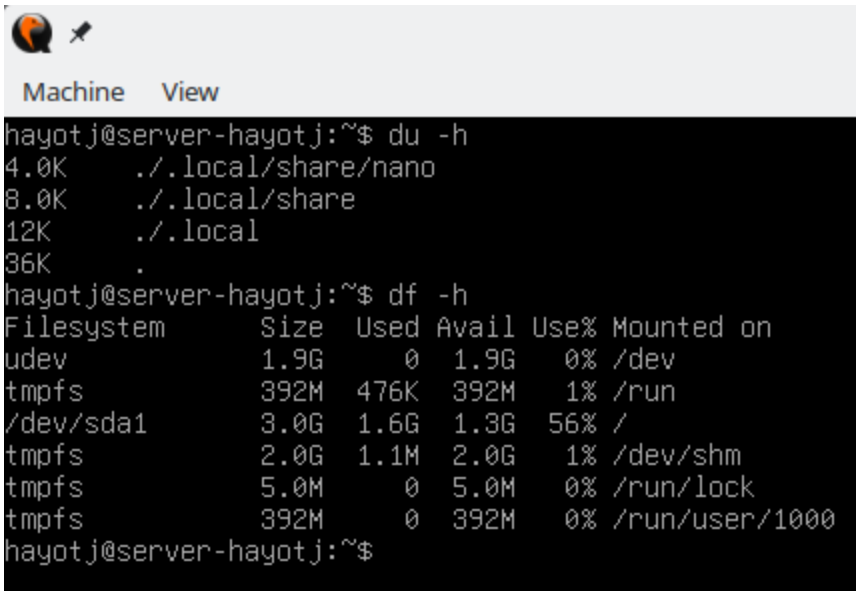


Figure 17: space

Thank you for reading!