

Installation Guide: Debian 12 (No GUI), Apache, PostgreSQL, PHP and PHPPgAdmin on Qemu/KVM

By Essediri Iliesse

Table of contents:

1. Introduction	3
Important Formatting Rules:	3
2. Requirements	3
3. Launch the Debian Installer.....	3
Debian Installation Steps.....	4
4. Start the Virtual Machine	6
5. Install Apache	6
6. Install PostgreSQL, set up a database, and connect to it from the host machine	7
7. Install PHP.....	9
8. Install PhpPgAdmin.....	9
9. Additional Checks	10
10. Security Analysis.....	11
11. Conclusion	11

1. Introduction

This guide is intended for future students of the IUT, it explains how to install Debian 12 without a graphical interface, and how to install Apache, PostgreSQL, PHP and PhpPgAdmin on a virtual machine (VM) using Qemu/KVM. This tutorial is step-by-step and includes some screenshot.

Important Formatting Rules:

- Commands executed inside the VM will be written in **bold**
 - Commands executed on the host (Linux) machine will not be bold
 - Commands to be run as root (superuser) will start with #
 - Commands to be run as a regular user will start with \$
-

2. Requirements

Before you start, make sure you have:

- A Linux computer (host machine for the VM)
- Access to the command line
- An internet connection
- The Debian 12 "netinst" ISO (already downloaded in /usr/local/images-ISO/)
- The Qemu/KVM installation and scripts provided (e.g., S2.03-lance-installation)

But If you're doing this on your personal computer:

You will need to install Qemu/KVM manually and download the correct Debian 12 "netinst" ISO image from the [official Debian website](#).

3. Launch the Debian Installer

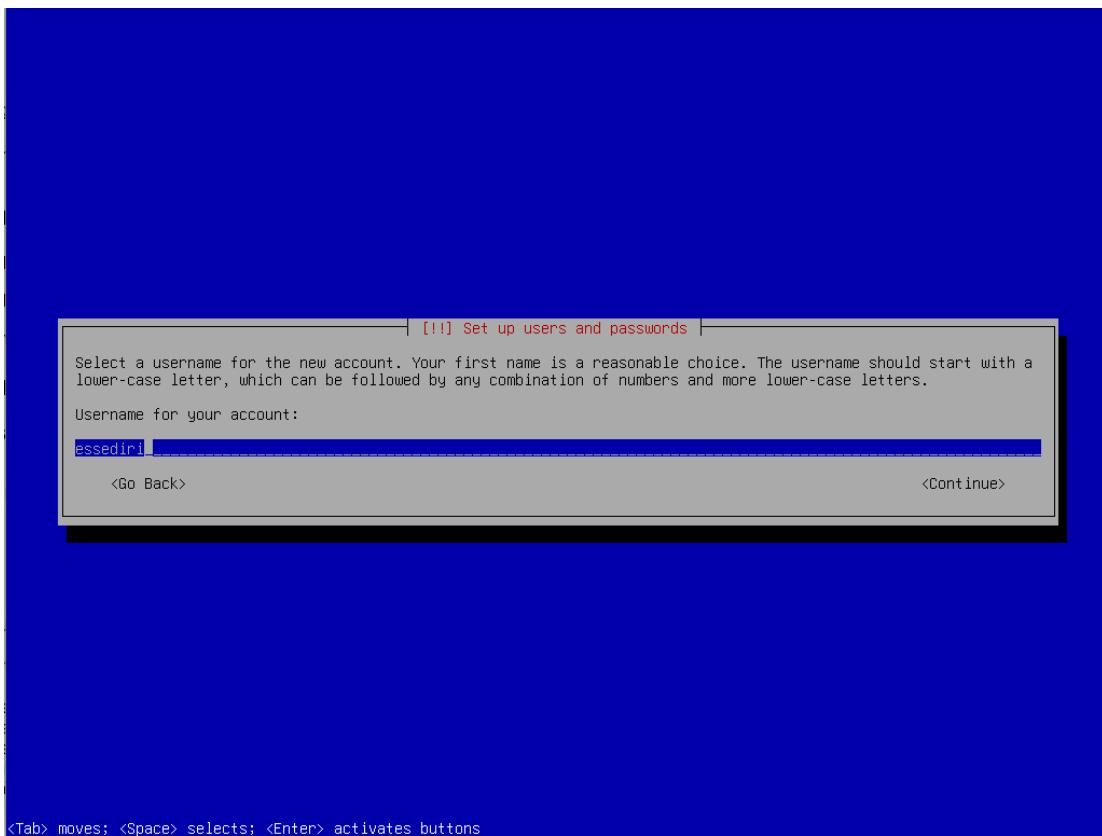
1. Open a terminal on your Linux station.
2. Run the installation script:

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

3. Follow the installer instructions:

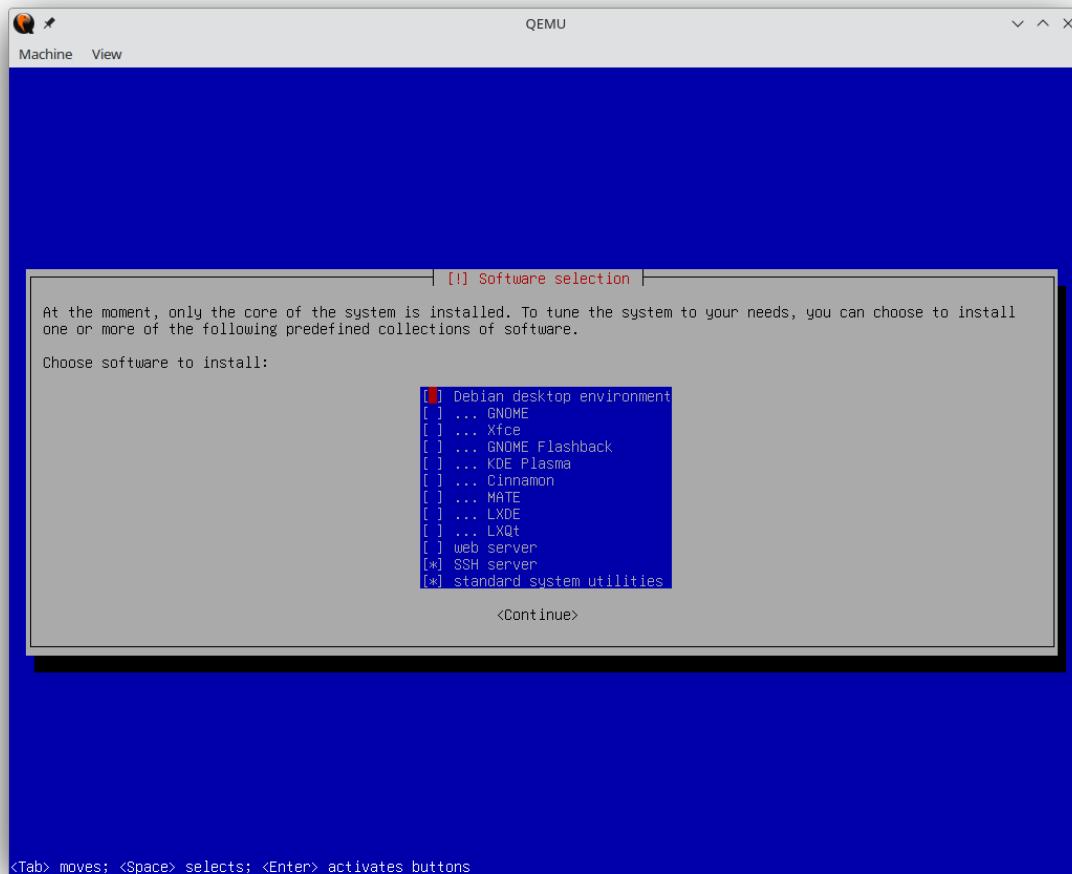
Debian Installation Steps

- Language: English
- Location: Other > Europe > France
- Locale: United States - en_US.UTF-8
- Keyboard: French
- Hostname: type server-your_login_UGA
- Domain name: leave it empty
- Set root password: type root and confirm
- Full name: type your full name
- Username: type your UGA login like in the screenshot



- User password: type etu and confirm
- Partitioning:
 - Choose Guided - use entire disk
 - Then All files in one partition

- Confirm and select Finish partitioning and write changes to disk
- Software selection:
 - Uncheck Debian desktop environment
 - Check SSH server
 - Leave standard system utilities checked
- Install GRUB on /dev/sda like as shown in the screenshot



Once installation is complete, log in as your UGA login with password « etu » and open the konsole.

To access a root console, use the command **\$ sudo -i**. If sudo is not available on your system, you can instead type **\$ su** - and enter the root user's password to log in as root.

You can now shut-down the VM

```
# systemctl poweroff
```

4. Start the Virtual Machine

1. On your host machine, run:

```
# S2.03-lance-machine-virtuelle
```

2. Log in with your user account.

3. Check the system: **\$ cat /etc/fstab** just like the screenshot :

```
essediri@server-essediri:~$ 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=82eaf423-fb58-4cf8-bc1a-c8abaa31608a /          ext4    errors=remount-ro 0      1
# swap was on /dev/sda5 during installation
UUID=fda268d2-0630-4b86-ac84-5b45d0774e35 none     swap    sw            0      0
/dev/sr0   /media/cdrom0 udf,iso9660 user,noauto 0      0
essediri@server-essediri:~$
```

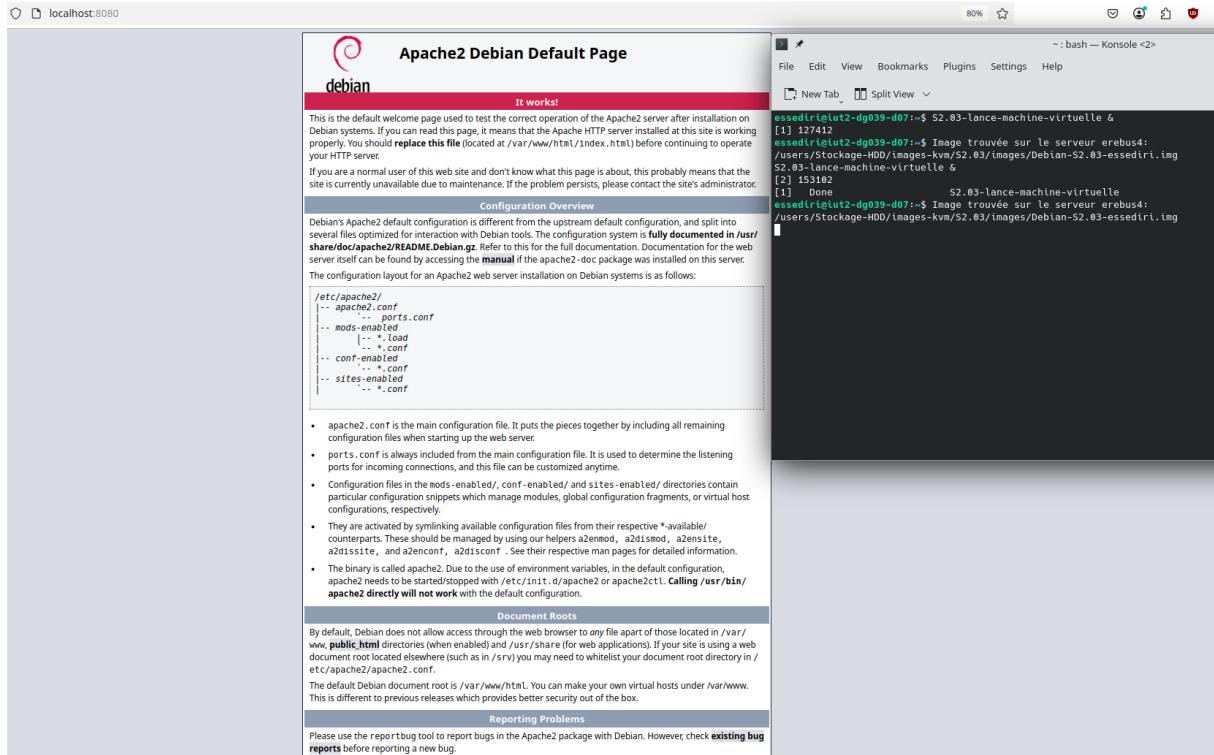
5. Install Apache

1. Update packages: **# apt update**
2. Install Apache: **# apt install apache2**
3. Start the service: **# systemctl start apache2**
4. Check Apache: **# systemctl status apache2**

```
root@server-essediri:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-04-08 10:22:19 CEST; 43min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 485 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 535 (apache2)
    Tasks: 11 (limit: 4642)
   Memory: 53.4M
      CPU: 1.268s
     CGroup: /system.slice/apache2.service
             ├─535 /usr/sbin/apache2 -k start
             ├─538 /usr/sbin/apache2 -k start
             ├─539 /usr/sbin/apache2 -k start
             ├─656 /usr/sbin/apache2 -k start
             ├─657 /usr/sbin/apache2 -k start
             ├─687 /usr/sbin/apache2 -k start
             ├─688 /usr/sbin/apache2 -k start
             ├─689 /usr/sbin/apache2 -k start
             ├─690 /usr/sbin/apache2 -k start
             ├─696 /usr/sbin/apache2 -k start
             └─702 /usr/sbin/apache2 -k start

Apr 08 10:22:17 server-essediri systemd[1]: Starting apache2.service - The Apache HTTP Server...
Apr 08 10:22:19 server-essediri apachectl[511]: AH00558: apache2: Could not reliably determine the server's fully qualified do
Apr 08 10:22:19 server-essediri systemd[1]: Started apache2.service - The Apache HTTP Server.
lines 1-25/25 (END)
```

5. On your host machine, open Firefox: Go to: <http://localhost:8080> You should see the Apache default page. If you can't see the page like in the screenshot, you should restart from the step 5.



6. Install PostgreSQL, set up a database, and connect to it from the host machine

1. On your VM: # **apt install postgresql**
2. Start PostgreSQL: # **systemctl start postgresql**
3. Check the state : # **systemctl status postgresql** (this command is also available for other thing like SSH)

```
root@server-essediri:~# systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Tue 2025-04-08 10:22:22 CEST; 44min ago
     Process: 596 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
    Main PID: 596 (code=exited, status=0/SUCCESS)
      CPU: 1ms

Apr 08 10:22:22 server-essediri systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Apr 08 10:22:22 server-essediri systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@server-essediri:~#
```

```
root@server-essediri:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
  Active: active (running) since Tue 2025-04-08 10:22:18 CEST; 42min ago
    Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 489 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 514 (sshd)
   Tasks: 1 (limit: 4642)
  Memory: 6.6M
    CPU: 98ms
   CGroup: /system.slice/ssh.service
           └─514 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Apr 08 10:22:17 server-essediri systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Apr 08 10:22:18 server-essediri sshd[514]: Server listening on 0.0.0.0 port 22.
Apr 08 10:22:18 server-essediri sshd[514]: Server listening on :: port 22.
Apr 08 10:22:18 server-essediri systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
Apr 08 10:48:03 server-essediri sshd[899]: Accepted password for essediri from 10.0.2.2 port 49654 ssh2
Apr 08 10:48:03 server-essediri sshd[899]: pam_unix(sshd:session): session opened for user essediri(uid=1000) by (uid=0)
Apr 08 10:48:03 server-essediri sshd[899]: pam_env(sshd:session): deprecated reading of user environment enabled
Apr 08 10:48:03 server-essediri sshd[899]: pam_unix(sshd:session): session closed for user essediri
root@server-essediri:~# _
```

3. Switch to the postgres user: # su - postgres

4. List databases: \l (if you didn't have the same its normal)

```
postgres=# \l
              List of databases
   Name    | Owner | Encoding | Collate | Ctype | ICU Locale | Locale Provider | Access privileges
----+-----+-----+-----+-----+-----+-----+-----+
esson_database | essediri | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc |
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc |
template0 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc |
template1 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc |
(4 rows)
postgres=#

```

5. Create a new user: CREATE USER (login UGA) WITH password 'etu' ;

6. Create a new database: CREATE DATABASE (login UGA) ;

7. Connect to the database: \$ psql your_login

8. Create a table and insert data like in the screenshot :

```
postgres=# CREATE TABLE utilisateur(nom VARCHAR(50),grade VARCHAR(50));
CREATE TABLE
postgres=# \d utilisateur
          Table "public.utilisateur"
 Column | Type | Collation | Nullable | Default
----+-----+-----+-----+-----+
 nom | character varying(50) | | |
 grade | character varying(50) | | |
postgres=# INSERT INTO utilisateur (nom) VALUES essediri;
ERROR: syntax error at or near "essediri"
LINE 1: INSERT INTO utilisateur (nom) VALUES essediri;
postgres=# INSERT INTO utilisateur (nom) VALUES ('essediri');
INSERT 0 1
postgres=# SELECT * FROM utilisateur;
   nom | grade
----+-----+
 essediri | 
(1 row)
postgres=#

```

9. View pg_shadow table:

```
SELECT * FROM pg_shadow;
```

username	usesysid	usecreatedb	usesuper	userreplic	usebypassrls	passwd	valuntil	useconfig
postgres	10	t	t	t	t			
essediri	16388	t	f	f	f	SCRAM-SHA-256\$4096:PxAQ8WGuMuCzrZI/Mk0rig==:jVBjH04ot1JIneopDERHKsLWXMN10QeTECDhxVg6iqU=:bczL250AopbtzmGpLzbck3zgmLmjIUeMIPjX6++FgkI=		

(2 rows)

10. Exit PostgreSQL:

```
\q
```

11. From your Linux station:

```
psql -h localhost postgres
```

```
essediri@iut2-dg039-d07:~$ psql -h localhost postgres
Password for user essediri:
psql (15.12 (Debian 15.12-0+deb12u2))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

postgres=> 
```

7. Install PHP

1. On your VM: # apt install php-common libapache2-mod-php php-cli
 2. Create a test PHP file: # echo "" | tee /var/www/html/info.php
 3. From the browser on your host: Go to: http://localhost:8080/info.php You should see PHP info.
-

9. Install PhpPgAdmin

1. Add Backports to sources.list : # nano /etc/apt/sources.list
2. Add at the end of the files, write this sentence :

http://deb.debian.org/debian bookworm-backports main
3. Make an update : # apt update
4. Install PhpPgAdmin: # apt install phppgadmin/bookworm-backport

5. Edit the file Connection.php and change:

```
case '14': return 'Postgres'; break;
```

to:

```
case '15': return 'Postgres'; break;
```

6. Restart Apache : # **systemctl restart apache2**

7. Go to: <http://localhost:8080/phpPgAdmin>

The screenshot shows the phpPgAdmin interface. At the top, it says "localhost:8080/phpPgAdmin/" in the address bar. The main window title is "phpPgAdmin". On the left, there's a sidebar with "Serveurs" and "PostgreSQL" selected. A warning message "Erreurs lors du chargement" is visible. The main area has a title "Résultats de la requête" and contains the SQL query "SELECT count(*) FROM utilisateur;". Below the query, the result is shown: "count" with value "0" and "1 ligne(s)". At the bottom, there are links for "Éditer SQL", "Étendre", "Télécharger", and "Rafraîchir".

Congratulations, you can now use this to administrate your database !

9. Additional Checks

1. Copy file page_sae_S2.03.php and check the page on your host machine :

```
# cp /users/info/www/intranet/enseignements/S2.03/page_sae_S2.03.php  
/var/www/html/
```

2. Open http://localhost:8080/page_sae_S2.03.php

3. Check disk space: `$ df -h`

```
root@server-essedir:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G    0  1.9G   0% /dev
tmpfs           392M  480K  392M   1% /run
/dev/sda1        3.0G  1.6G  1.2G  58% /
tmpfs           2.0G  1.1M  2.0G   1% /dev/shm
tmpfs           5.0M    0  5.0M   0% /run/lock
tmpfs           392M    0  392M   0% /run/user/1000
root@server-essedir:~#
```

10. Security Analysis

- Keep your system up to date: `# apt update && apt upgrade`
 - PostgreSQL: edit pg_hba.conf to restrict access

Use strong passwords and avoid running unnecessary services.

11. Conclusion

You have successfully installed Debian 12, Apache, PostgreSQL, and PHP on a virtual machine without a graphical interface. You can now serve web pages and databases securely from your VM.