

Guide to installing the Debian 11 system on a virtual machine

Authors :

Jarod Tivollier C1 BUT1

Contents

Step 1: INSTALL ISO IMAGE.....	3
Step 2: CREATE THE VIRTUAL MACHINE.....	4
Step 3: INSTALLING THE DEBIAN 11 SYSTEM.....	6
Step 4: APACHE INSTALLATION.....	8
Optional Step 1: APACHE TEST.....	9
Step 5: POSTGRESQL INSTALLATION.....	11
Optional Step 2: POSTGRESQL TEST.....	12
Part I: CREATING A ROLE AND DATABASE.....	12
Part II: CREATING A TABLE AND ADDING ATTRIBUTES.....	13
Part III: ACCESSING DATABASES ON THE HOST MACHINE.....	14
Step 6: PHP INSTALLATION.....	16
Optional Step 3: PHP TEST.....	16
Step 7: SECURITY CHECK.....	17
Optional Step 4: STORAGE.....	17
Optional Step 5: ALL SERVER INFORMATION.....	18

Step 1: INSTALL ISO IMAGE

An ISO image is a computer file that contains an exact copy of a storage medium, such as a CD, DVD, or Blu-ray disc. The ISO image contains all the data and structure of the original media, including files, folders, and file systems.

Here the ISO image of the latest Debian version is Debian 11 nicknamed "bullseye"

you can install the ISO image at this URL:

<https://www.debian.org/download>

Step 2: CREATE THE VIRTUAL MACHINE

The first thing to do is to install "Qemu" Qemu is a free virtual machine software, which can emulate a processor and, more generally, a different architecture if necessary. It will be the software that will support your virtual machine. You can install it at this address.

<https://www.qemu.org/download/>

Secondly you have to create the disk image on Qemu, launch a terminal and execute this command:

qemu-img create « \$image » nG

\$image = name of your image file (e.g. disk)

n = allocated disk space (e.g. 5)

Modify these 2 parameters as you want finally to launch the virtual machine just run this command and modify the parameters as you wish.

Finally third, to be able to launch the virtual machine run this long command:

```
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"
```

Here's what each parameter in this command is all about:

- **qemu-system-x86_64:** This is the main QEMU binary used to emulate a x86_64 system.
- **-machine q35:** Specifies the Q35 virtual machine template. Q35 is a modern virtual machine architecture with advanced features.
- **-cpu host:** Uses the CPU host for emulation, which means that QEMU will use the features and performance of the host machine's processor.

- **-m 4G**: Allocates 4 GB of memory to the virtual machine. This specifies the amount of RAM available to the virtual system.
- **-enable-kvm**: Enables hardware virtualization (KVM) support. This allows QEMU to use CPU virtualization extensions to improve performance.
- **-device VGA,xres=1024,yres=768**: Adds a VGA display device to the virtual machine with a resolution of 1024x768 pixels. This will display a graphical interface in the QEMU emulator window.
- **-display gtk,zoom-to-fit=off**: Uses the GTK GUI to display the graphical output of the virtual machine. The **zoom-to-fit=off** option disables automatic zoom in the GUI, allowing the specified resolution to be displayed without automatic adjustment.
- **-drive \$drive**: Specifies the path to the disk image used by the virtual machine. The **\$drive** variable must be replaced with the actual path of the disk image.
- **-device e1000,netdev=net0**: Adds an e1000 virtual network adapter to the virtual machine and attaches it to the **net0** network appliance.
- **-netdev user,id=net0,hostfwd=tcp::2222-:22,hostfwd=tcp::4443-:443,hostfwd=tcp::8080-:80,hostfwd=tcp::5432-:5432**: Creates a virtual network appliance of type "user" with the identifier **net0**. This device allows the virtual machine to access the Internet through the host's network connection. The "hostfwd" options define the redirection of TCP ports from the host to the virtual machine. In this example, ports 2222, 4443, 8080, and 5432 on the host are redirected to ports 22, 443, 80, and 5432 on the virtual machine, respectively.

Step 3: INSTALLING THE DEBIAN 11 SYSTEM

The system will ask you a lot of different settings to install your DEBIAN 11 system. Adjust the settings as you see fit according to what suits you best, for me here are the settings I modified.

Language : English

Locales : United States, en_US.UTF-8

Location : other/Europe/France

Keyboard : French

Hostname : server-tivollij

Root Password : « root »

User Account - Full Name : Jarod Tivollier

User Name : tivollij

User Password : « etu »

Partition disks : Guided - use entire disk

Partition disks : All files in one partition

Partition disks : Yes

Software Selection : check that "Debian desktop" is not checked and that "ssh server" is checked and "Standard System Utilities"

Install GRUB : Yes

Device for boot loader : /dev/sda

For Software Selection:



For Hostname of your machine:



You can check your "Partition disks" settings in more detail with this command:

cat /etc/fstab

You will normally arrive on this file:

```
tivollij@server-tivollij:~$ 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=8b607967-3a69-4d38-a988-13ee2c5fe01a / ext4 errors=remount-ro 0 1
# swap was on /dev/sda5 during installation
UUID=7b020c2a-4f89-4f3b-af6f-18952b5de218 none swap sw 0 0
/dev/sr0 /media/cdrom0 udf,iso9660 user,noauto 0 0
tivollij@server-tivollij:~$ _
```

Step 4: APACHE INSTALLATION

Apache allows you to create a localhost hosting server to allow you to host your HTML site that works only for Unix operating systems (which is DEBIAN 11).

Log in to your Root account with the command:

sudo -i

And enter your root password mine being "root", then, run this command:

apt-get update && apt-get upgrade

This command brings your packages up to date. It is very important otherwise the different installation in this tutorial will not work. To install Apache run this command:

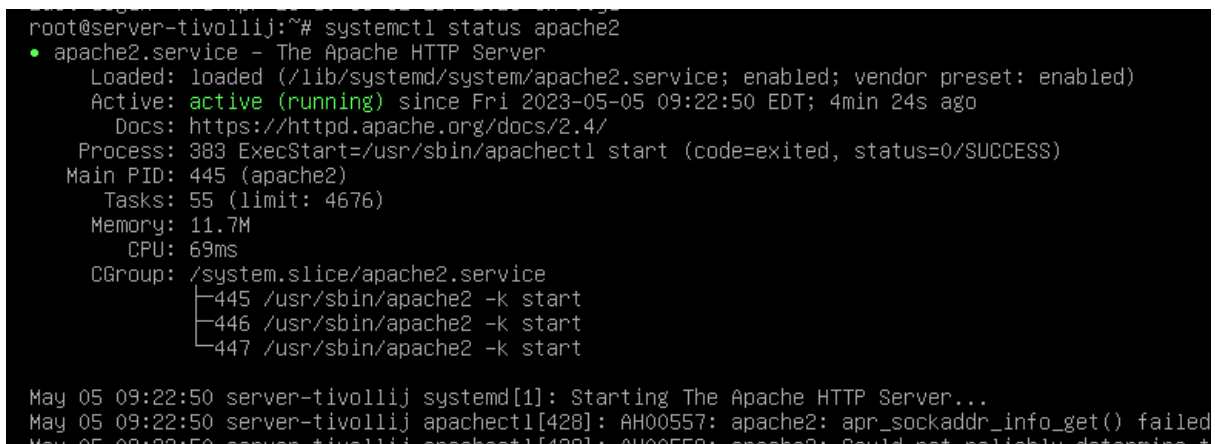
apt install apache2

Then this command:

service apache2 start

After that your Apache2 software will then be operational to verify it run this command:

systemctl status apache2 (by being on the Root account)



```
root@server-tivollij:~# systemctl status apache2
• apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-05-05 09:22:50 EDT; 4min 24s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 383 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 445 (apache2)
    Tasks: 55 (limit: 4676)
   Memory: 11.7M
      CPU: 69ms
   CGroup: /system.slice/apache2.service
           └─445 /usr/sbin/apache2 -k start
             └─446 /usr/sbin/apache2 -k start
               └─447 /usr/sbin/apache2 -k start

May 05 09:22:50 server-tivollij systemd[1]: Starting The Apache HTTP Server...
May 05 09:22:50 server-tivollij apachectl[428]: AH00557: apache2: apr_sockaddr_info_get() failed
May 05 09:22:50 server-tivollij apachectl[428]: AH00558: apache2: Could not reliably determine the
```

If everything works then the console shows you this (image above)

The line that will allow you to check this is the line with the green word "active (running)"
If it doesn't work use this command:

systemctl restart apache

This restarts the Apache service.

Optional Step 1: APACHE TEST

To test the proper functioning of Apache you can simply go to the default page of Apache to do this to run on your virtual machine console the command

telnet localhost 80 (on your normal account not root)

This will show you this:


```
$ telnet localhost 80
Trying ::1...
Connected to localhost.
Escape character is '^]'.
```

After that type this command:

HEAD / HTTP/1.0

After that the console will answer you by HTTP/1.1 200 OK and much more. But the most interesting line and this one allows us to know the link between the server and the page. Finally, you can access your page on your non-virtual machine from the URL "http://localhost:8080".

Séance 3 x S2.03 - Séance 2 x PhpPgAdmin — V x GitHub - phppgac x PHP: Debian GNU x S2.03 - CDC et Re x PostgreSQL: D



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.


Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2`. Due to the use of environment variables, in the default configuration, `apache2` needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.



Then you arrive on this page if everything works.

Step 5: POSTGRESQL INSTALLATION

PostgreSQL is a database management and creation software that is very useful if you want to store for example login and password of your website.

To install PostgreSQL run this command (being on root account):

apt install postgresql

To verify that PostgreSQL is installed, either you run the command that is the same for Apache:

systemctl status postgresql

Where you will get this:

```
tivollij@server-tivollij:~$ systemctl status postgresql
• postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
   Active: active (exited) since Fri 2023-05-05 10:37:25 EDT; 22min ago
   Process: 9408 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 9408 (code=exited, status=0/SUCCESS)
   CPU: 795us
```

Same as for Apache if it is not "active" to execute the command:

systemctl restart postgresql

Or to check if it is indeed install run this command:

su - postgres

The result of this command is to log in to the postgres account and this only works if PostgreSQL is installed. Run the exit command if you want to go back to your account.

Optional Step 2: POSTGRESQL TEST

Part I: CREATING A ROLE AND DATABASE

To test POSTGRESQL first log in to the PostgreSQL admin account to do so type this command:

su - postgres

This will connect you to the postgres account and you can use this command that will allow you to have the list of default databases on PostgreSQL

psql -l

And it will show you this:

```
postgres@server-tivollij:~$ psql -l
```

List of databases					
Name	Owner	Encoding	Collate	Ctype	Access privileges
jarod_tivollier	tivollij	UTF8	en_US.UTF-8	en_US.UTF-8	
postgres	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	
template0	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=c/postgres + postgres=CTc/postgres
template1	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=c/postgres + postgres=CTc/postgres

(4 rows)

Normally you will not have the database called "jarod_tivollier" ; it will be created later in this tutorial.

Start by connecting to the postgres database to do this Run this command

psql -U -d postgres

Then create a role with the command CREATE ROLE with your username here "tivollij"

CREATE ROLE username WITH LOGIN PASSWORD"your password";

Then add the rights that will allow you to create modify databases

**ALTER ROLE username SUPERUSER;
ALTER ROLE username CREATEDB;**

You can check the creation of the role and the addition of rights with this command:

/du

```
postgres=# \du
```

Role name	List of roles Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
tivollij	Superuser, Create role, Create DB	{}

"Tivollij" here will be replaced by your username.

We will then your database for this disconnect from postgres with the exit command and then connect to your role that you have just created with the command

su - username

Then to create your database type command taking into account that it does not need space or add "_" to separate 2 words as I did on the first image of this part and not to capitalize:

CREATE DATABASE name_of_your_DB;

Following this you have more than to connect to your database with the command quote a little higher but changing 1 parameter

psql -U -d name_of_your_DB

Part II: CREATING A TABLE AND ADDING ATTRIBUTES

To create a table make sure you are in your database or you want to create tables to create a table you must use the CREATE TABLE command and add as a parameter the attribute and its type here is the command:

**CREATE TABLE name_of_your_table(name_of_attribute
type_of_attribute);**

Example for using this command: "CREATE TABLE number (int digit)" int here is the type to define an integer so you can only add integers in this attribute.

13

To add elements in your attributes that you have created run this command:

```
INSERT INTO name_of_your_table VALUES  
(the_number_of_what_you_want);
```

Then to see this modification/addition type this command:

```
SELECT * FROM name_of_your_table;
```

And you will get a result similar to this:

```
jarod_tivollier=# SELECT * FROM simple;  
un  
----  
 2  
(1 row)
```

Here my table is called "simple" and my attribute "un".

Part III: ACCESSING DATABASES ON THE HOST MACHINE

We come to the complicated part of the PostgreSQL test, access your database stored on your virtual machine, on your host machine for this you will have to modify 2 files of the PostgreSQL software which are the file "postgresql.conf" and "pg_hba.conf" start with the changes on the file "postgresql.conf". Use this command to open the file on your virtual machine:

```
nano /etc/postgresql/13/main/postgresql.conf
```

Modify the line where it is "listen_adresses" delete the "#" then replace "localhost" by "*" then find the line where it is "password_encryption" delete the "#" then modify if necessary "MD5" by "SCRAM-SHA-256".

Then we will modify the file "pg_hba.conf" using this command to access it:

```
nano /etc/postgresql/13/main/pg_hab.conf
```

Find the line where this is located "IPv4 remote connections" then add this "host all 0.0.0.0/0 scram-sha-256" then all places where there is "md5" replace with "scram-sha-256". "Scram-SHA-256" is a type of encryption for passwords.

Finally on your host machine connect to your database with this command:

psql -h localhost name_of_your_DB

I will ask you for your password and then you can check with /d that it is your database here is what it should look like:

```
tivollij@pc-dg-035-07:~$ psql -h localhost jarod_tivollier
Password for user tivollij:
psql (13.11 (Debian 13.11-0+deb11u1), server 13.9 (Debian 13.9-0+deb11u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

tivollij@jarod_tivollier=> \d
      List of relations
 Schema | Name  | Type  | Owner
-----+-----+-----+-----
 public | simple | table | tivollij
(1 row)

tivollij@jarod_tivollier=> SELECT * FROM simple;
 un
----
  2
  5
(2 rows)
```

And if you want you can see how your password is encrypted by displaying a hidden table of PostgreSQL by typing this command:

SELECT * FROM pg_shadow;

And you will get this:

```
jarod_tivollier=# SELECT * FROM pg_shadow;
 username | usesysid | usecreatedb | usesuper | use repl | usebypassrls |          | valuntil | useconfig
-----+-----+-----+-----+-----+-----+-----+-----+-----
 postgres |      10 | t           | t        | t        | t            |          |          |
 tivollij |   16389 | t           | t        | f        | f            | SCRAM-SHA-256$4096:3P72PX9L0kj31m2VboH2Dw==$xpdfP9C3PzJ
PWBvRC5bfTKfBydPJdf6IxxRM7hufFMs=: /SVJGFxHLy0j93dJHBpHek54NVPHDE2SJWq+L i2CkcY= |          |
(2 rows)
```

Step 6: PHP INSTALLATION

PHP is an "extension" for HTML that allows you to write scripts to execute specific things on a website.

To install PHP run this command:

apt-get install php (by being on the root account)

Always to know if it is well installed run this command:

systemctl status php

Optional Step 3: PHP TEST

To know if PHP is installed we will create a file in .php for that, type this command:

cat > info.php (create your "info.php name file")

Then:

nano info.php (allows you to edit this file)

And finally write these lines in your file:

```
<?php
phpinfo();19
phpinfo(INFO_MODULES);
?>
```

Then move this file to the folder where all your pages stored on your Apache server localhost /var/www/html/ are located using this command:

mv info.php /var/www/html/

Which will now allow access through this URL on your non-virtual machine
<http://localhost:8080/info.php>
it will open the default PHP page.

Step 7: SECURITY CHECK

To be able to do security checks it is enough to execute 2 commands which are:

apt upgrade
apt update

"apt upgrade" checks if updates are needed and "apt update" updates are found by "apt upgrade".

```
root@server-tivollij:~# apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@server-tivollij:~# apt update
Get:1 http://deb.debian.org/debian bullseye InRelease [116 kB]
Get:2 http://security.debian.org/debian-security bullseye-security InRelease [48.4 kB]
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Get:4 http://security.debian.org/debian-security bullseye-security/main Sources [201 kB]
Get:5 http://deb.debian.org/debian bullseye/main Sources [8,637 kB]
Get:6 http://security.debian.org/debian-security bullseye-security/main amd64 Packages [245 kB]
Get:7 http://security.debian.org/debian-security bullseye-security/main Translation-en [161 kB]
Get:8 http://deb.debian.org/debian bullseye-updates/main Sources.diff/Index [18.5 kB]
Get:9 http://deb.debian.org/debian bullseye-updates/main amd64 Packages.diff/Index [18.5 kB]
Get:10 http://deb.debian.org/debian bullseye-updates/main Translation-en.diff/Index [7,239 B]
Get:11 http://deb.debian.org/debian bullseye-updates/main Sources T-2023-05-24-2006.01-F-2023-05-24-2006.01.pdiff [547 B]
Get:12 http://deb.debian.org/debian bullseye-updates/main amd64 Packages T-2023-05-24-2006.01-F-2023-05-24-2006.01.pdiff [362 B]
Get:13 http://deb.debian.org/debian bullseye-updates/main Translation-en T-2023-05-24-2006.01-F-2023-05-24-2006.01.pdiff [355 B]
Get:14 http://deb.debian.org/debian bullseye/main amd64 Packages [8,183 kB]
Get:15 http://deb.debian.org/debian bullseye/main Translation-en [6,240 kB]
Fetched 23.9 MB in 4s (6,257 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
35 packages can be upgraded. Run 'apt list --upgradable' to see them.
N: Repository 'http://deb.debian.org/debian bullseye InRelease' changed its 'Version' value from '11.6' to '11.7'
```

Optional Step 4: STORAGE

To see how many spaces are left for your virtual machine you can run this command:

df -k

And you will get this:

```
tivollij@server-tivollij:~$ df -k
Filesystem      1K-blocks    Used Available Use% Mounted on
udev             1995196         0   1995196   0% /dev
tmpfs            402612         488    402124   1% /run
/dev/sda1       3067812  1726328   1165460  60% /
tmpfs           2013040         16   2013024   1% /dev/shm
tmpfs             5120          0        5120   0% /run/lock
tmpfs           402608          0    402608   0% /run/user/1000
```

With the remaining space of all your disks partitions.

Optional Step 5: ALL SERVER INFORMATION

If you want to access all your server information whether php, PostgreSQL, SSH and well you can thanks to php for this you will have to create a PHP file and copy it to `/var/www/html/`

For my part I had a file that did this so here's what it's supposed to show you:

```
P2.01 — Publier son p... IUT2 Grenoble - S2.03 x S2.03 - Séance 1 x S2.03 - Séance 2 x S2.03 - CDC et Rendu x New Tab
localhost:8080/page_sae_S2.03.php

Bonjour

Je suis www-data

Qui est connecté ?

tivollij tty1 Jun 9 08:30

Mes disques sont

Mes interfaces

1: lo: mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s2: mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:12:34:56 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s2
        valid_lft 85971sec preferred_lft 85971sec
    inet6 fec0::5054:ff:fe12:3456/64 scope site dynamic mngtmpaddr
        valid_lft 85975sec preferred_lft 13975sec
    inet6 fe80::5054:ff:fe12:3456/64 scope link
        valid_lft forever preferred_lft forever

My apache install is

ii apache2 2.4.56-1-deb11u1 amd64 Apache HTTP Server
ii apache2-bin 2.4.56-1-deb11u1 amd64 Apache HTTP Server (modules and other binary files)
ii apache2-data 2.4.56-1-deb11u1 all Apache HTTP Server (common files)
ii apache2-utils 2.4.56-1-deb11u1 amd64 Apache HTTP Server (utility programs for web servers)
ii libapache2-mod-php 2:7.4+76 all server-side, HTML-embedded scripting language (Apache 2 module) (default)
ii libapache2-mod-php7.4 7.4.33-1+deb11u3 amd64 server-side, HTML-embedded scripting language (Apache 2 module)

My apache status is

* apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-06-09 08:23:06 EDT; 7min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 368 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 456 (apache2)
    Tasks: 9 (limit: 4676)
   Memory: 25.9M
      CPU: 275ms
   CGroup: /system.slice/apache2.service
           |-456 /usr/sbin/apache2 -k start
           |-477 /usr/sbin/apache2 -k start
           |-478 /usr/sbin/apache2 -k start
           |-479 /usr/sbin/apache2 -k start
           |-480 /usr/sbin/apache2 -k start
           |-481 /usr/sbin/apache2 -k start
           |-862 /usr/sbin/apache2 -k start
           |-938 sh -c systemctl status apache2
           |-939 systemctl status apache2

My postgresql install is
```



THE END