

Simulation of a Banking System to Address the Problem of Race Conditions

Adriana Noemi Pech Cauich

Walter Medina

Installing the LEMP Stack on Ubuntu 20.04

The LEMP (Linux, Nginx, MySQL, PHP) software stack is a group of software that can handle dynamic web pages and PHP-based web apps. It includes a Linux operating system, an Nginx (pronounced “Engine-X”) web server, a MySQL database for storing data, and PHP for processing dynamic content. This tutorial explains how to set up a LEMP stack on a server running Ubuntu 20.04. Ubuntu manages the Linux part of the stack, and we’ll guide you through configuring the rest of the components.

Prerequisites

- Ubuntu 20.04
- A user account with admin or sudo access

Step 1: Installing the Nginx Web Server

We’ll use Nginx, a high-performance web server, to show web pages to our site visitors. We’ll get this software using the apt package manager. As this is the first time using apt in this session, start by updating your server’s package index:

```
$ sudo apt update
```

Execute the below command to install nginx:

```
$ sudo apt install nginx
```

```

adriana@adriana-VirtualBox:~$ 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 Sat 2024-08-03 21:55:20 CST; 14s ago
     Docs: man:nginx(8)
   Process: 82397 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 82398 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Main PID: 82399 (nginx)
    Tasks: 2 (limit: 4542)
   Memory: 1.6M
      CPU: 11ms
   CGroup: /system.slice/nginx.service
           └─82399 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─82401 "nginx: worker process"

```

Figure 1: Caption

```

adriana@adriana-VirtualBox:~$ sudo nano /etc/nginx/sites-available/cacahuate.com
[sudo] contraseña para adriana:
adriana@adriana-VirtualBox:~$ sudo chown www-data:www-data /var/www/cacahuate.com/html
adriana@adriana-VirtualBox:~$ sudo chmod -R 755 /var/www/cacahuate.com/html
adriana@adriana-VirtualBox:~$ http://189.220.43.27

```

Figure 2: Caption

Press Y & ENTER when prompted to confirm the installation of Nginx. After completion, the Nginx web server will run on your Ubuntu 20.04 server. If the UFW firewall is enabled, you must allow Nginx connections. Nginx creates several UFW application profiles during installation. To check the available UFW profiles, use the following command:

```
$ sudo ufw app list
```

It is advisable to activate the most restrictive profile that permits the necessary traffic. As SSL configuration is not covered in this guide, you only need to allow standard HTTP traffic on port 80.

```
$ sudo ufw allow 'Nginx HTTP'
```

Verify the change by checking the status:

```
$ sudo ufw status
```

After adding the new firewall rule, you can verify the server's status by accessing your public IP address or server's domain name in your web browser. If you don't have a domain name pointed to your server and don't know your server's public IP address, you can retrieve it by running one of the following commands:

```
$ ip addr show
$ hostname -I
```

This will display several IP addresses. You can attempt each one consecutively within your internet browser. Enter the IP address you receive in your web browser, and you will be taken to Nginx’s default landing page.

`http://server_domain_or_ip_address`

Step 2: Installing MySQL

Now that your web server is running, the next step is to install the database system to store and manage data on your site. MySQL is a widely utilized database management system in PHP environments, and you can install it using the following command:

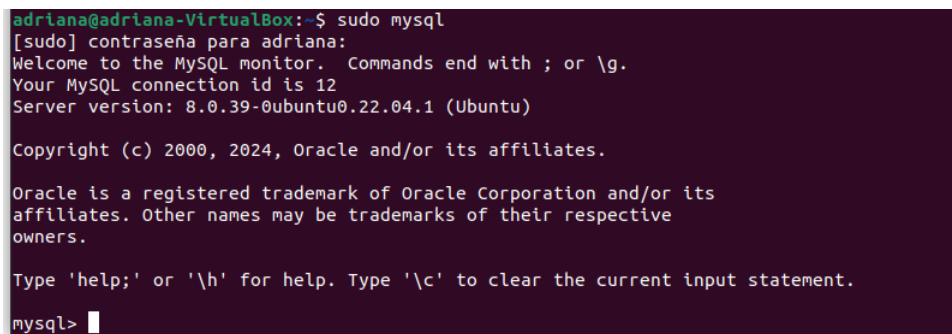
A screenshot of a terminal window with a dark purple background. The prompt is 'adriana@adriana-VirtualBox:~\$'. The user has entered 'sudo mysql'. The terminal shows the MySQL login process: '[sudo] contraseña para adriana:', 'Welcome to the MySQL monitor. Commands end with ; or \g.', 'Your MySQL connection id is 12', and 'Server version: 8.0.39-0ubuntu0.22.04.1 (Ubuntu)'. It also displays copyright information for Oracle and instructions on how to use help and clear the input. The prompt 'mysql>' is visible at the bottom with a cursor.

Figure 3: Caption

```
$ sudo apt install mysql-server
```

After installing MySQL, it is a good idea to run a security script included with it. This script helps remove any insecure default settings and ensures secure access to your database system. To start the interactive script, enter the following command:

```
$ sudo mysql_secure_installation
```

You’ll be prompted to configure the VALIDATE PASSWORD PLUGIN—type Y for yes or any other key to continue without enabling it. If you select “yes,” you’ll be prompted to choose a level of password validation. Opting for level 2 for strong validation may lead to errors if passwords lack numbers, upper and lowercase letters, or special characters. When finished, check if you can log in to the MySQL console:

```
$ sudo mysql
```

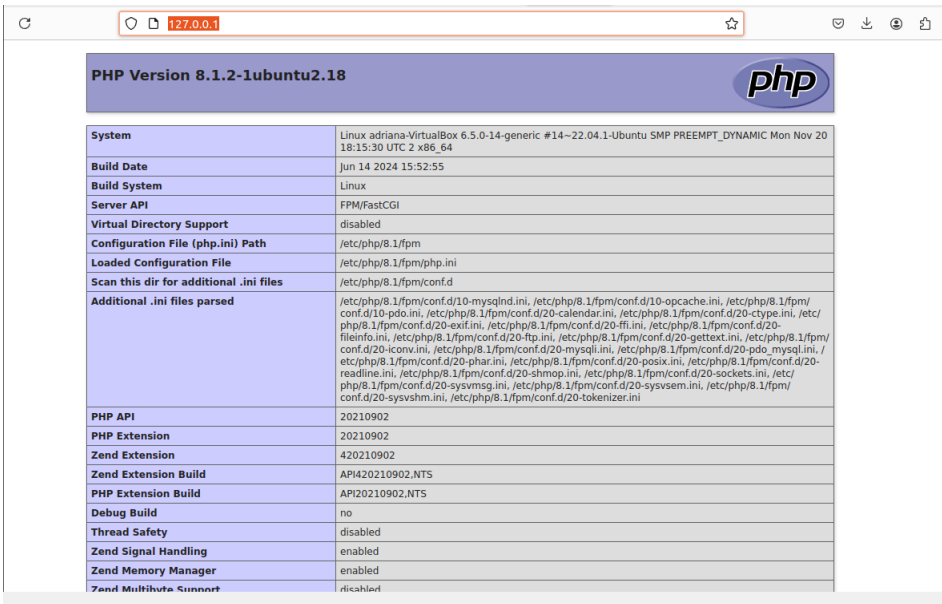
This command connects to the MySQL server using the administrative database user root, indicated by sudo. The expected output is as follows: To exit the MySQL console, use the following command:

```
mysql> exit
```

Your MySQL server is now installed & secured.

Step 3: Installing PHP

After installing Nginx to serve your content and MySQL to store and manage your data, the next step is to install PHP to generate dynamic content. Apache embeds the PHP interpreter within every request, while Nginx needs an external program to handle PHP processing. This external program mediates between the PHP interpreter and the web server. While this setup improves performance for many PHP-based websites, it requires extra configuration steps. You'll need to install php8.1-fpm, known as "PHP FastCGI Process Manager." This program uses the current PHP version (at the time of writing) to direct Nginx to forward PHP requests to the software for processing. Additionally, you'll require php-mysql, a PHP module that facilitates communication between PHP and MySQL-based databases. Core PHP packages will be installed automatically as dependencies.



PHP Version 8.1.2-1ubuntu2.18	
System	Linux adriana-VirtualBox 6.5.0-14-generic #14~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Mon Nov 20 18:15:30 UTC 2 x86_64
Build Date	Jun 14 2024 15:52:55
Build System	Linux
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/fpm
Loaded Configuration File	/etc/php/8.1/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/fpm/conf.d
Additional .ini files parsed	/etc/php/8.1/fpm/conf.d/10-mysqlnd.ini, /etc/php/8.1/fpm/conf.d/10-opcache.ini, /etc/php/8.1/fpm/conf.d/10-pdo.ini, /etc/php/8.1/fpm/conf.d/20-calendar.ini, /etc/php/8.1/fpm/conf.d/20-ctype.ini, /etc/php/8.1/fpm/conf.d/20-exif.ini, /etc/php/8.1/fpm/conf.d/20-fileinfo.ini, /etc/php/8.1/fpm/conf.d/20-ftp.ini, /etc/php/8.1/fpm/conf.d/20-gettext.ini, /etc/php/8.1/fpm/conf.d/20-iconv.ini, /etc/php/8.1/fpm/conf.d/20-mysqli.ini, /etc/php/8.1/fpm/conf.d/20-pdo_mysql.ini, /etc/php/8.1/fpm/conf.d/20-phar.ini, /etc/php/8.1/fpm/conf.d/20-posix.ini, /etc/php/8.1/fpm/conf.d/20-readline.ini, /etc/php/8.1/fpm/conf.d/20-shmop.ini, /etc/php/8.1/fpm/conf.d/20-sockets.ini, /etc/php/8.1/fpm/conf.d/20-sysvmsg.ini, /etc/php/8.1/fpm/conf.d/20-sysvsem.ini, /etc/php/8.1/fpm/conf.d/20-sysvshm.ini, /etc/php/8.1/fpm/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902.NTS
PHP Extension Build	API20210902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multithread Support	disabled

Figure 4: Caption

```
$ sudo apt install php8.1-fpm php-mysql
```

When prompted, press Y/yes and ENTER to confirm the installation. Now that your PHP components have been installed, the next step is configuring Nginx to use them.

```
adriana@adriana-VirtualBox:/var/www/127.0.0.1$ cat info.php
<?php
phpinfo();
?>
```

Figure 5: Caption

Step 4: Configure Nginx to Use the PHP Processor

In the Nginx web server context, server blocks can encapsulate configuration details and host multiple domains on a single server. We will use “your_domain” as an example domain name. Please note: We have used accuwebtraining.com in images in place of your_domain throughout the blog. You need to use “your domain name” there. By default, Nginx on Ubuntu 20.04 has a single server block enabled, serving documents from the directory /var/www/html. This setup is suitable for a single site, but for multiple sites, creating a structured directory within /var/www for each website is better. This way, /var/www/html remains the default directory served if a client request does not match any specific sites. To set up the root web directory for “your_domain,” follow these steps:

```
adriana@adriana-VirtualBox:~$ mysql -u AdrianaP -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 8.0.39-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| performance_schema |
| test_DB      |
+-----+
3 rows in set (0.04 sec)
```

Figure 6: Caption

```

adriana@adriana-VirtualBox:/var/www/127.0.0.1$ sudo nano index.php
adriana@adriana-VirtualBox:/var/www/127.0.0.1$ cat index.php
<?php
$servername = "127.0.0.1";
$username = "AdrianaP";
$password = "C@nel@23";
$dbname = "test_DB";

// Crear conexión
$conn = new mysqli($servername, $username, $password, $dbname);

// Verificar la conexión
if ($conn->connect_error) {
    die("Conexión fallida: " . $conn->connect_error);
}

// Consultar la base de datos
$sql = "SELECT id, name FROM technology_list";
$result = $conn->query($sql);

```

Figure 7: Caption

```
$ sudo mkdir /var/www/your_domain
```

Assign ownership of the directory using the \$USER environment variable, representing your ‘current system user’:

```
$ sudo chown -R $USER:$USER /var/www/your_domain
```

Create a new configuration file using your preferred command-line editor in Nginx’s sites-available directory. Here, we are using vi editor:

```
$ sudo vi /etc/nginx/sites-available/your_domain
```

Insert the following basic configuration into the newly created file:

```

server {
    listen 80;
    server_name accuwebtraining.com www.your_domain;
    root /var/www/your_domain;
    index index.html index.htm index.php;
    location / {
        try_files $uri $uri/ =404;
    }
    location ~ /\.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
    }
    location ~ /\.ht {
        deny all;
    }
}

```

Here is what each of these directives and location blocks do:

- **Listen** – By default, Nginx listens on port 80 for HTTP requests. This port can be changed if needed.
- **Root** – Specifies the document root, denoting the directory where the files hosted by this website are stored.
- **Index** – Configures the order in which Nginx prioritizes index files for your website. Prioritize index.html over index.php for quick maintenance landing page setup in PHP apps.

You can modify these settings based on the specific needs of your application.

- **Server_name** – Specifies the domain names or IP addresses to which this server block should respond. Direct this directive to your server's domain name or public IP address.
- **Location /** – The initial location block contains a `try_files` directive, which is responsible for verifying the existence of files or directories corresponding to a URL request. If Nginx cannot locate the required resource, it will generate a 404 error.
- **Location php** – This location block executes PHP, directing Nginx to utilize the `fastcgi-php.conf` configuration file and the `php8.1-fpm.sock` file. These declarations specify the socket associated with php8.1-fpm for proper PHP processing.
- **Location /ht** – The final location block addresses .

1 Step 5: Configuring the PHP Script for Withdrawal Operation

To configure the PHP script to interact with the MySQL database and perform a withdrawal operation, follow these steps:

1.1 Create the Database and Tables

First, ensure that the database and necessary tables exist. Use the MySQL command-line interface to create them if they don't already exist.

```
[language=sql, caption=Creating Database and Tables] CREATE DATABASE
bank; USE bank;

CREATE TABLE accounts ( id INT PRIMARY KEY, balance DEC-
IMAL(10, 2) );

CREATE TABLE logs ( id INT AUTO_INCREMENT PRIMARY KEY, amount DECIM
INSERT INTO accounts (id, balance) VALUES (1, 1000.00); – Initial-
ize account with ID 1
```

1.2 Create the PHP Script

This PHP script connects to a MySQL database to perform a withdrawal operation. It first checks the balance of an account and verifies if sufficient funds are available for the withdrawal. If funds are sufficient, the script updates the account balance and records the transaction in a logs table. If any errors occur during these operations, they are reported to the user. The script assumes a hardcoded account ID and withdrawal amount, so in a production environment, it would need modifications to handle user inputs securely and dynamically.

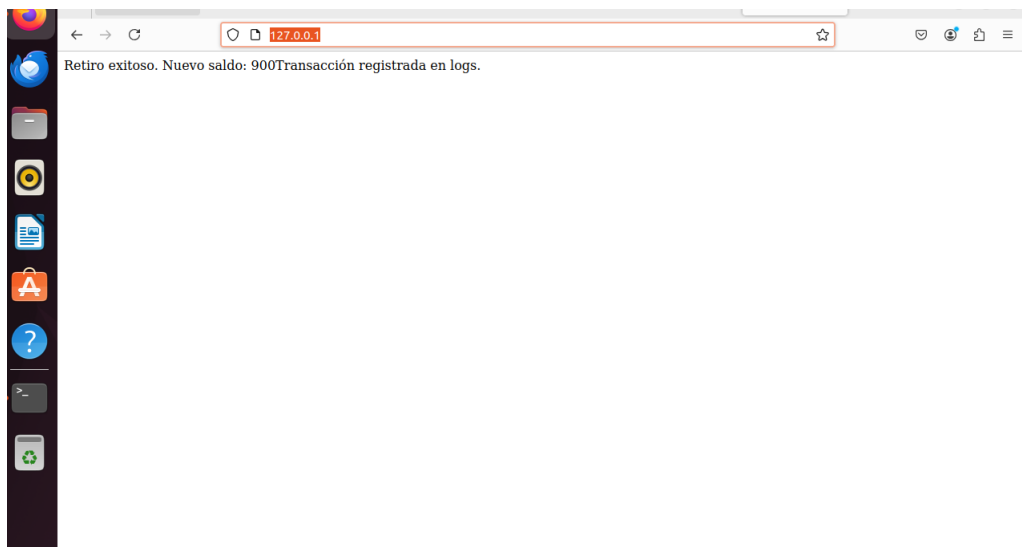


Figure 8: Caption