LAMP PROJECT

Matthieu Pierre Brühwiler

Haaga-Helia University of Applied Sciences
Bachelor of Business Administration
Project Report
Spring 2023

Table of contents

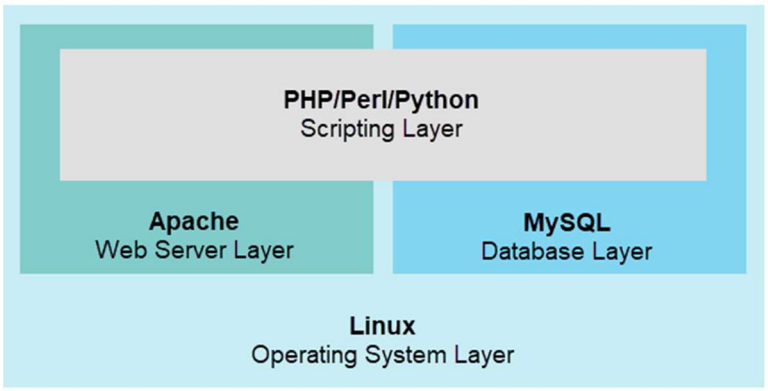
1 What is LAMP?

LAMP is an acronym that is used to denote the software stack: Linux, Apache, MySQL, PHP/Perl/Python. Each letter of the acronym represents a free and open-source building block. Nowadays, LAMP may be one of the most common software stacks and is used in several domains such as e-commerce, e-learning, Software as A Service (SAAS), social media, and many more.

The use of LAMP is to build and deploy dynamic web applications using a MySQL database, and an Apache web server, both running on a Linux computer. To render a dynamic web page, we add the layer of PHP or Perl or Python.

We can represent the LAMP Stack architecture as follows:

Visual representation of the LAMP stack



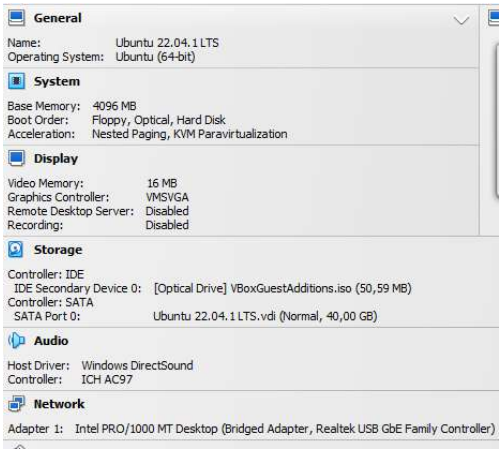
2 Installation and setup

2.1 System environment

As part of the development of this project, I decided to use Ubuntu Desktop 22.04.1 LTS as the Linux distro. Generally, this kind of project is set up in a server environment such as Ubuntu Server. However, I decided to use the desktop version to give the reader a better view and understanding of the project. It is much easier to visualize the results through a web browser than a curl command in the terminal. In terms of the other components, I did not target a specific version of each of them. I only downloaded the latest versions. It is important to note that I decided to use PHP as the scripting part of the project. Indeed, this last choice is because I already knew how to write a dynamic web page with PHP and not with Python or Perl. This meant that I didn't have to concentrate on anything other than the configuration of LAMP.

2.2 OS installation

As stated before, I installed Ubuntu Desktop 22.04.1 LTS as a virtual machine (VM) on Oracle's VirtualBox (VB). To create the VM from VB, I used the tab menu "*Machine*" and then "*New*". I entered a name for the VM, and I selected the ISO file of Ubuntu Desktop that I downloaded from Ubuntu's website. I set a memory of **4096MB** with only one CPU. Regarding the storage, I used the "***Create a Virtual Hard Disk Now***" option with a capacity of **40GB**. I also chose "***VDI (Virtual Disk Image)***" as the hard disk file type variant. At this point, I was finally ready to start my Ubuntu VM. My configuration was looking as below:



About the OS installation, I followed each step, I used the default parameters and entered my personal information such as credentials.

As for all kinds of projects, it is important to start by updating and upgrading the operating system. For that, I used the following command: **sudo apt update && sudo apt upgrade**.

2.3 Apache

One of the first requirements is the Apache web server. To install the latter on a Linux, it requires only one command: **sudo apt install apache2**. After installing it, you can configure several parameters depending on your needs. In my case, I modified apache2.conf file with the following command: "**sudo nano /etc/apache2/apache2.conf**". I checked/changed the following parameters as follow:

```
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 20

#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 5
```

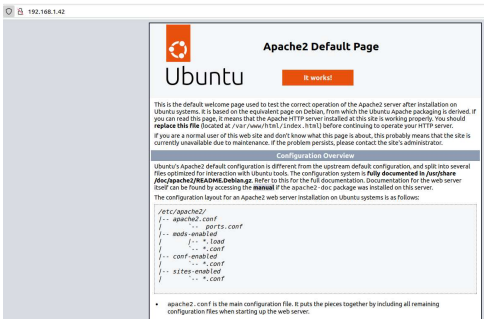
I have set the "*MaxKeepAliveRequests*" parameter to **20** because the server will not have to handle many requests (I don't need a lot of performance for this demonstration). I left the other parameters as default.

To allow access to Apache, it is important to allow access to it from ports 80 (HTTP) and 443 (HTTPS). So that, people from outside will be able to reach your server. To enable the firewall, I used the following commands: **sudo ufw allow in "Apache Full"** and **sudo ufw enable**. However, it is important to note the expression Apache Full allows HTTP and HTTPS requests. So, if you only need HTTPS requests, you can use: **sudo ufw allow in "Apache Secure"**. To ensure that the firewall is correctly set, I used the following command: **sudo ufw status**. It gave me the following result:

```
Status: active

To                         Action      From
--                         ------      ----
Apache Full                ALLOW       Anywhere
Apache Full (v6)           ALLOW       Anywhere (v6)
```

Finally, before testing the server, it is useful to restart it with: **sudo systemctl restart apache2**. After the service restarted correctly, I checked with the commands: **sudo apt install net-tools** and **ifconfig**.

To test the web server, entered my IP address in my browser, and I arrived on the Apache2 Default Page:



2.3.1 Creating a virtual host

I also created a virtual host for my domain on Apache. It is not mandatory in the case of a single website. However, it is a good thing to do in all cases. To create the virtual host, I did the following steps:

Creating a configuration file for the domain named bruhwiler.fi:
sudo nano /etc/apache2/sites-available/bruhwiler.fi.conf

I wrote the following configuration:

```
 GNU nano 6.2                        /etc/apache2/sites-available/bruhwiler.fi.conf
<Directory /var/www/html/bruhwiler.fi/public_html>
        Require all granted
</Directory>
<VirtualHost *:80>
        ServerName bruhwiler.fi
        ServerAlias www.bruhwiler.fi
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html/bruhwiler.fi/public_html

        ErrorLog /var/www/html/bruhwiler.fi/logs/error.log
        CustomLog /var/www/html/bruhwiler.fi/logs/access.log combined
</VirtualHost>
```

Then, I created the directories that will be needed due to the previous configuration file and I also changed the permissions of the theses.
sudo mkdir /var/www/html/bruhwiler.fi
sudo mkdir /var/www/html/bruhwiler.fi/public_html
sudo mkdir /var/www/html/bruhwiler.fi/logs
sudo chown -R www-data:www-data /var/www/html/bruhwiler.fi/public_html
sudo chmod 755 /var/www/html/bruhwiler.fi/public_html

After creating and changing the permissions, I was able to add enable my virtual host with the command: **sudo a2ensite bruhwiler.fi.conf** and I deactivated the default one with the command: **sudo a2dissite 000-default.conf**.

Before restarting Apache, I checked that the configuration was correct using the command: **sudo apache2ctl config test**. At this point, I had an error telling me that the server name was missing in the general configuration file. So, I added the line: "**ServerName bruhwiler.fi**" in the file */etc/apache2/apache2.conf*. After re-checking the configuration of Apache, everything was correct, so I restarted the server with the command: **sudo systemctl restart apache2**. I checked the result from my web browser. The following result is normal because I didn't have an HTML file in the public_html directory.

2.4 MySQL

As stated before, MySQL is a relational open-source database. To install the database, I used the command: **sudo apt install MySQL-server**. The configuration of the database is relatively simple. Firstly, I had to set a password for the user root. For this, I used the command: **sudo mysql** to get connected to MySQL from the terminal. Then, I was able to change the password for the user root: **ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'MY_PASSWD';**. As you may have guessed, I replaced MY_PASSWORD with a strong password. I also created a table for my future web page using the SQL commands:

CREATE DATABASE webdata; → create a database named *webdata*.
CREATE USER 'webuser' IDENTIFIED BY 'password'; → create a new user named *webuser*.
GRANT ALL ON webdata.* TO 'webuser'; → grant all rights to the webuser on webdata database

I also created a table in the webdata database for future needs using the SQL script:

```
mysql> CREATE TABLE webdata.shopping_list(
    -> name VARCHAR(50),
    -> quantity INT);
Query OK, 0 rows affected (0.00 sec)
```

Finally, when I finished the MySQL configuration, I was able to leave the SQL shell. However, I still had to increase the security of my database, so I used the command:
sudo mysql_secure_installation.

At this point, MySQL asked me the following questions, and I answered:
Validate password component → **y** (yes).
Password validation policy → **1** (Medium).
Change the password for root → **n** (no).
Remove anonymous users → **y** (yes).
Disallow root login remotely → **y** (yes).
Remove the test database and access it → **y** (yes).
Reload privilege table row → **y** (yes).

2.5 PHP

At this point, I only had to install PHP and create an HTML and PHP page to render information on the webpage. To install PHP, I used: **sudo apt install php libapache2-mod-php php-mysql**. The installation of PHP did not require a specific configuration, however, I had to write a basic HTML with a PHP script page and store it under the following path: */var/www/html/bruhwiler.fi/public_html*. My PHP script:

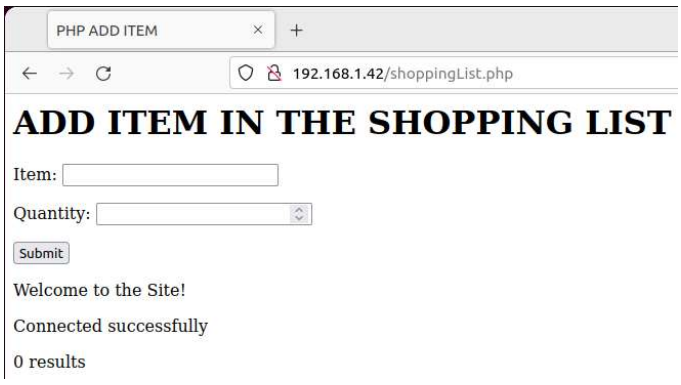
```
<html>
<head>
    <title>PHP ADD ITEM</title>
</head>
    <body>
        <h1>ADD ITEM IN THE SHOPPING LIST</h1>
        <form action="shoppingList.php" method="post">
            <p>Item: <input type="text" name="name" /></p>
            <p>Quantity: <input type="number" name="quantity" /></p>
            <p><input type="submit" name="submit" value="Submit" /></p>
        </form>
    <?php echo '<p>Welcome to the Site!</p>';

    // Create variables for the MySQL connection
    $servername = "localhost";
    $username = "webuser";
    $password = "Testing123$";

    // Create MySQL connection
    $conn = mysqli_connect($servername, $username, $password);

    // If the conn is empty ==> connection has failed.
    if (!$conn) {
        die('<p>Connection failed: </p>' . mysqli_connect_error());
    }
    echo '<p>Connected successfully</p>';


    // Save the item and quantity from the form into shopping list the database
    if (isset($_POST['submit'])) {
        $name = $_POST['name'];
        $quantity = $_POST['quantity'];
        $sql = "INSERT INTO webdata.shopping_list (name, quantity) VALUES ('$name', '$quantity')";
        if (mysqli_query($conn, $sql)) {
            echo '<p>New record created successfully</p>';
        } else {
            echo '<p>Error: </p>' . $sql . '<p> </p>' . mysqli_error($conn);
        }
    }

    // Display the shopping list
    $sql = "SELECT * FROM webdata.shopping_list";
    $result = mysqli_query($conn, $sql);
    if (mysqli_num_rows($result) > 0) {
        echo '<table border="1">';
        echo '<tr><th>Item name</th><th>Quantity</th></tr>';
        while ($row = mysqli_fetch_assoc($result)) {
            echo '<tr><td>' . $row['name'] . '</td><td>' . $row['quantity'] . '</td></tr>';
        }
        echo '</table>';
    } else {
        echo '<p>0 results</p>';
    }

    ?>
</body>
</html>
```

It is important to note that I restarted the server after adding the file in the directory *pubcli_html*.

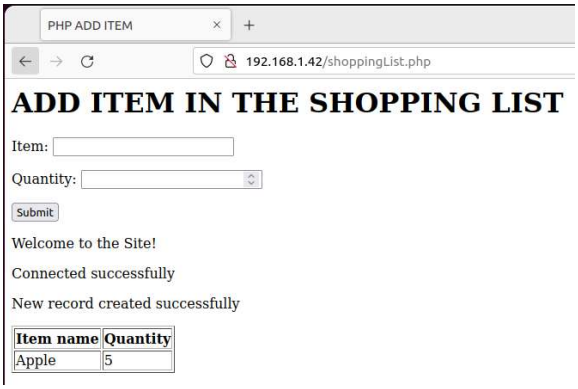
3 Results and testing

Finally, as stated before, I tested that the Apache server was working properly using my IP address on my web browser. I also did it at the end of the project, with the PHP script. I give me the following result:

Before adding item

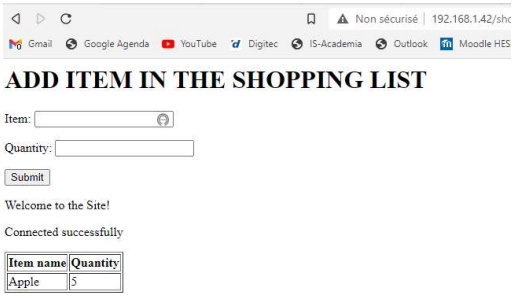


After adding an item

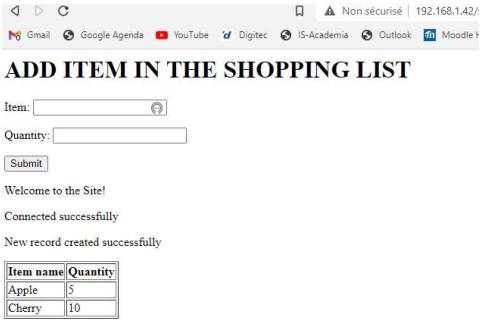


I also tested from my real computer (not my VM) to see if it was also working properly, and I had the same result.

Before adding item



After adding an item



4 Issues

During the realisation of the project, I have been lucky, because I didn't have a lot of issues. One of my issues was due to MySQL's May 2022 update. Indeed, when I tried to enter the command for securing MySQL (sudo mysql_secure_installation), I had the following error message: "Failed! Error: SET PASSWORD has no significance for user 'root'@'localhost' as the authentication method used doesn't store authentication data in the MySQL server. Please consider using ALTER USER instead if you want to change authentication parameters." The latter is due to a change in the new version of MySQL that does not allow to use of the root account without a password. To solve this problem, I followed the steps on the following website: https://devanswers.co/how-to-fix-failed-error-set-password-has-no-significance-for-user-rootlocalhost/.

As stated before, I also had an issue with the virtual host in Apache. The web server did not want to restart because I didn't specify the server name in the general configuration file of Apache. I found the solution to the problem gracefully to the command: **sudo apache2ctl configtest**. The latter showed me an error message that explained to me what was missing and how to easily solve it. To make sure I didn't do anything wrong, I also checked on the following website: https://unix.stackexchange.com/questions/155150/where-in-apache-2-do-you-set-the-servername-directive-globally

References

1. Phoenix – What is LAMP Stack? – https://phoenixnap.com/kb/what-is-a-lamp-stack
2. Wikipedia – LAMP (software bundle) – https://en.wikipedia.org/wiki/LAMP_(software_bundle)
3. Linode – Install a LAMP Stack on Ubuntu 22.04 – https://www.linode.com/docs/guides/how-to-install-a-lamp-stack-on-ubuntu-22-04/
4. DigitalOcean – How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 18.04 – https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-ubuntu-18-04
5. DevAnswers – How to Fix: “Failed! Error: SET PASSWORD has no significance for user ‘root’@’localhost’” – https://devanswers.co/how-to-fix-failed-error-set-password-has-no-significance-for-user-rootlocalhost/
6. Linux Handbook – Installing LAMP Server on Localhost on Ubuntu – https://linuxhandbook.com/lamp-stack-ubuntu/