Install and Configure PHP

PHP is the scripting language that enables dynamic content generation. It acts as the logic layer of the LAMP stack, processing user input, querying the database, and generating HTML to be served by Apache.

1. Install PHP

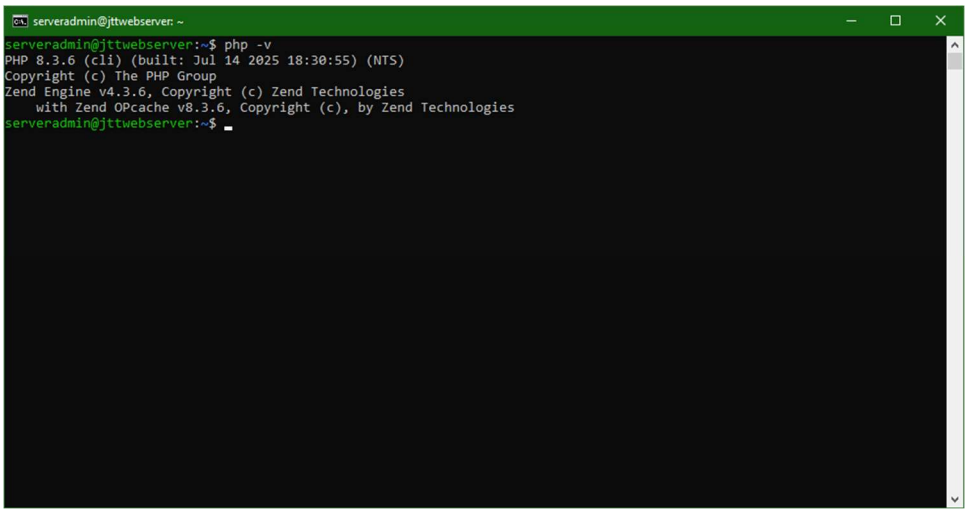
    a. For PHP, we need to install three packages: the core PHP interpreter, a common library module that enables Apache to handle PHP files, and a module that allows PHP to communicate with MySQL. To install the packages, type the following command and press enter:

    ```
    sudo apt install php libapache2-mod-php php-mysql -y
    ```

    b. Verify the installation by typing the following command and pressing enter:

    ```
    php -v
    ```

    You should see an output that displays the version and copyright information for PHP.



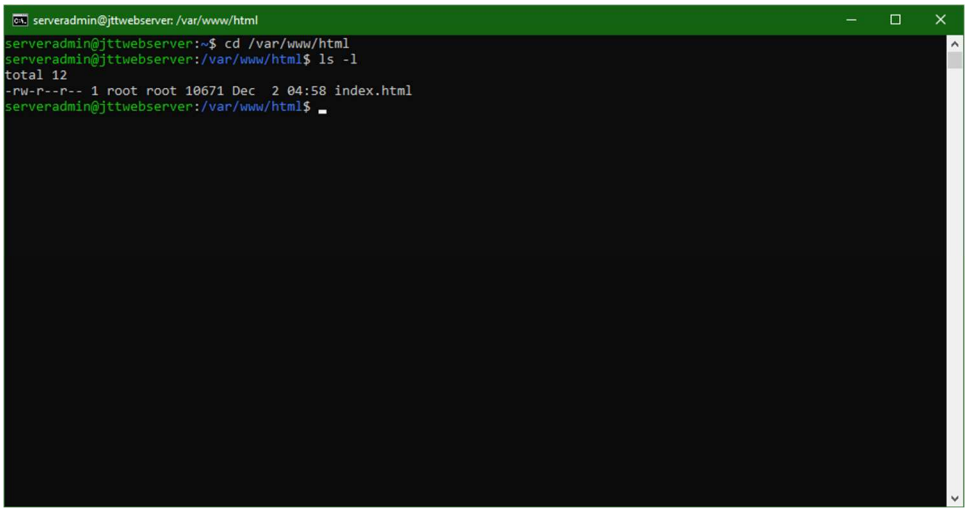
2. Test PHP with Apache

    a. We will create a test file in Apache's root directory to ensure PHP can communicate with the web server. First, let's navigate to Apache's root directory

and view its contents by typing the following commands and pressing enter after each line:

```
cd /var/www/html
ls -l
```

You will notice an index.html file, this is the file that displayed the Apache2 Default Page when we tested our Apache connection. You should also notice a -rw-r--r-- and root root in the output. The first – tells us this is a file, the rw- tells us the file owner (which is the first root) has read and write permissions, the first r-- tells that anyone else in the owners group (which is the second root) has read-only permissions, and the second r-- tells that all others also only have read-only permissions. This means only the server's root user can edit this file, which goes for any files in the root directory.



b. Now, let's create our test file by typing the following command and pressing enter:

```
sudo nano info.php
```

In the nano editor, type the following PHP script:

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

Exit the nano editor by pressing CTRL+X, typing y, and pressing enter.

c. To check if PHP is communicating with Apache, open a web browser on your host machine and type http://<your-ip-address>/info.php replacing <your-ip-address> with the static IP address for your Ubuntu Server.

You should see the PHP info page, confirming successful communication with Apache.

3. Test PHP with MySQL
   a. Now let's create a test file in the same directory to ensure PHP can communicate with MySQL by typing the following command and pressing enter:

```
sudo nano test.php
```

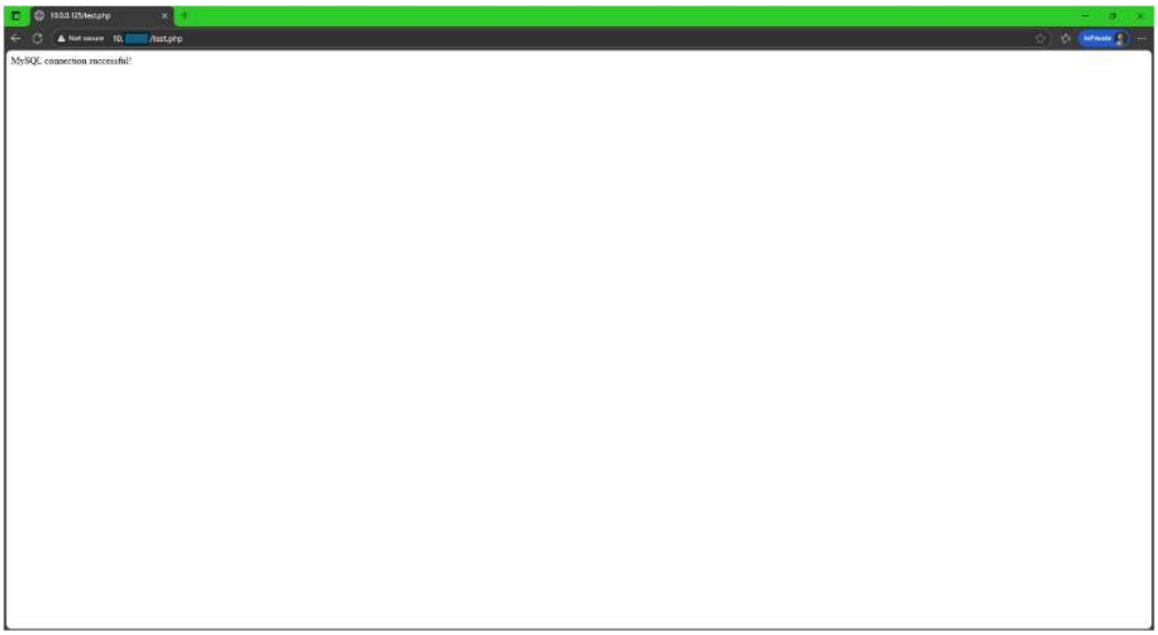
In the nano editor type the following script replacing <your-mysql-password> with the MySQL password created earlier (which will be Password1, unless you decided to change it during the mysql_secure_installation):

```
<?php
$servername = "localhost";
$username = "root";
$password = "<your-mysql-password>";

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "MySQL connection successful!";
?>
```

Press CTRL+X, type y, and press enter to exit the nano editor. Now open your web browser again and type http://<your-ip-address>/test.php replacing <your-ip-address> with the static IP address for your Ubuntu Server. You should see the “MySQL connection successful!” message displayed in your web browser.



Congratulations! You have successfully installed and configured a dynamic web server within a VirtualBox environment using Ubuntu Server and the LAMP stack. By completing each step–installing and securing the operating system, deploying Apache, MySQL, and PHP, and integrating them into a cohesive environment–you have built a fully functional platform capable of serving dynamic content. By following this tutorial, you now have a working foundation for experimentation, further customization, and integration with additional tools or services.