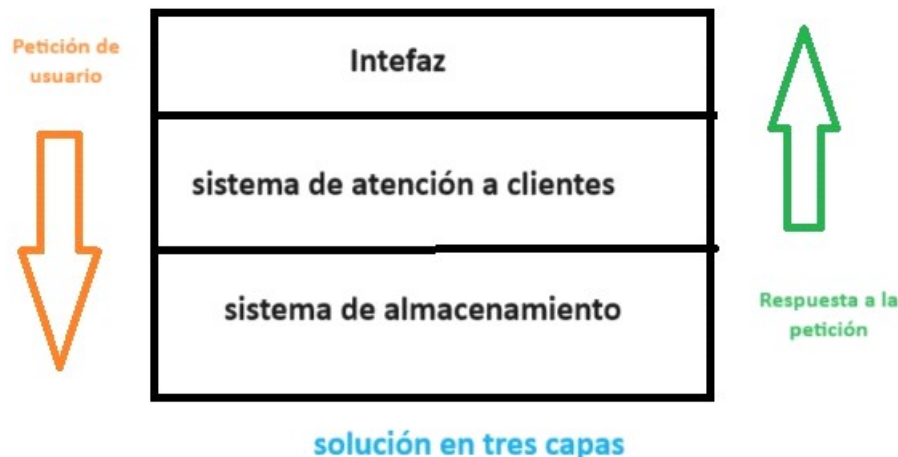


Introducción al diseño en software apilado

Considere el siguiente ejemplo: un usuario revisando su cuenta de banco, puede hacerlo desde su teléfono móvil o desde su computadora, en ambos casos aunque la interfaz que se le presenta es diferente, el resultado de sus acciones es el mismo, el usuario puede consultar el balance de su estado de cuenta, el equipo donde realice la consulta a su vez la transferirá al sistema de acceso a cuentas del Banco, el cual pedirá al sistema de almacenamiento por los datos, ambos sistemas pueden estar ubicados en diferentes localidades, pero a los ojos del usuario la interfaz, el sistema de acceso a cuentas del Banco y el de Base de Datos se presentan como una sola entidad, cuando varios tipos de software están organizados de tal manera que uno soporta al otro e interactúan entre ellos como un todo, se le conoce como **software apilado**, **solución apilada**, **tech stack**, etc.

Continuando con el ejemplo anterior, realizamos una abstracción del problema para modelar una solución, por un lado tenemos al usuario el cual realiza las peticiones, las cuales llegan a un sistema mediante la red, el cual valida la petición de usuario y la direcciona al sistema de almacenamiento, podemos definir entonces en tres secciones el problema:



Seguiente este esquema, necesitamos un software para programar la interfaz de usuario, otro para guardar y administrar la información del usuario y uno para conectar a los otros dos, actualmente existen muchos programas a escoger para completar los requisitos de diseño, usaremos como propuesta los siguientes:

Intefaz: HTML y PHP

Sistema de atención a clientes: Apache Web Server y PHPmyAdmin

Sistema de almacenamiento: MySQL.

Gracias a HTML y PHP, la interfaz puede ejecutarse desde un navegador, lo cual la hará más flexible y el usuario podrá ejecutarla desde varios equipos, el sistema de atención a clientes estará ubicado en un servidor el cual podría compartir con el sistema de almacenamiento, independientemente de si están juntos o separados, ambos programas de software se ejecutaron sobre el sistema operativo Linux, por lo que si tomamos las siglas de todo el software que usaremos, tendremos el siguiente acrónimo:



La implementación de esta solución, requiere manipular al menos cuatro clases de software distintos (cinco si incluimos el uso de HTML), siguiendo nuestro esquema de tres etapas podemos clasificar el software como sigue:

Sistema de almacenamiento

Linux, MySQL, la parte que el usuario no ve llamada también *Back End*

Sistema de atención a clientes

Linux, Apache y PHPMyAdmin, encargado de la conexión entre interfaz de usuario y base de datos conocida como *middleware* o *Aplicaciones (APP)* o *pegamento (GLUE)*.

Interfaz

Navegador Web, HTML y PHP, la parte con la que el usuario interactúa, conocida también como *front end*.

Así un programador que trabaja en Front End de nuestra pila sería llamado FrontEnd Developer y podría ser más específico un Web Developer, uno que administre la parte media sería conocido como MiddleWare Developer y podría ser un APP Developer, por último tendríamos al BackEnd Developer, una persona que se pueda manejar entre toda la estructura de la solución, se le conoce como programador FullStack, estos términos provienen del argot usado en la industria y no reflejan alguna carrera universitaria o área especializada, parecen más bien usados para delinear en una empresa las responsabilidades de un ingeniero de software o programador llamada coloquialmente.

Como nota final, sería resaltar el concepto de software apilado (solución de software o *framework*) como un método de diseño para implementar un sistema de software que resuelve un problema (en este caso una aplicación de banca móvil), no siempre un solo programa o lenguaje de programación podrá cubrir todos los requerimientos para completar una tarea.

Instalación de un servidor LAMP en Raspbian OS

Introducción:

A pesar de que la solución propuesta LAMP es algo "vieja" en el mercado, resalta por su carácter educativo, se explicará como implementar esta solución usando un computador Raspberry Pi y una

equipo externo conectados a una red casera usando el moden provisto por el proveedor de servicios de internet.

Instalacion:

Abrimos una ventana de Terminal (o una sesion shell), para ejecutar los comandos mostrados mas adelante.

0.Actualizacion

Hay que asegurar que nuestro sistema esta actualizado, para evitar problemas con la instalacion.

```
rodrigo@raspberrypi:~$ ls
Bookshelf  box86  Desktop  Downloads  goodtft  Music  Nbox86  Public  seekg.cpp  thinclient_drives  wine
box86pi3  chapter12  Documents  error.output.txt  LCD-show  Nbox64  Pictures  rpi-fbcp  Templates  Videos

rodrigo@raspberrypi:~$ sudo apt update
Hit:1 http://deb.debian.org/debian bullseye InRelease
Hit:2 http://deb.debian.org/debian bullseye-updates InRelease
Hit:3 http://security.debian.org/debian-security bullseye-security InRelease
Hit:4 http://archive.raspberrypi.org/debian bullseye InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.

rodrigo@raspberrypi:~$ sudo 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.

rodrigo@raspberrypi:~$
```

Escritos en secuencia los comandos *sudo apt update* y *sudo apt upgrade*, nos permiten descargar los paquetes mas recientes e implementarlos en nuestro sistema respectivamente.

1.Instalacion de Apache Web Server

Ejecutamos los comandos siguientes:

sudo apt install libapache2-mod-php

sudo apt-get install apache2 -y

```
rodrigo@raspberrypi:~$ sudo apt install libapache2-mod-php
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapache2-mod-php7.4 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0 php-common php7.4-cli
  php7.4-common php7.4-json php7.4-opcache php7.4-readline
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom php-pear
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapache2-mod-php libapache2-mod-php7.4 libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0
  php-common php7.4-cli php7.4-common php7.4-json php7.4-opcache php7.4-readline
0 upgraded, 17 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,131 kB of archives.
After this operation, 25.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://deb.debian.org/debian bullseye/main arm64 libapr1 arm64 1.7.0-6+deb11u2 [99.6 kB]
Get:2 http://deb.debian.org/debian bullseye/main arm64 libaprutil1 arm64 1.6.1-5+deb11u1 [89.6 kB]
Get:3 http://deb.debian.org/debian bullseye/main arm64 libaprutil1-dbd-sqlite3 arm64 1.6.1-5+deb11u1 [18.9 kB]
...
Se ha recortado la salida en la terminal para mostrar los puntos importantes

Info: Executing deferred 'a2enmod php7.4' for package libapache2-mod-php7.4
Enabling module php7.4.
^[[BCreated symlink /etc/systemd/system/multi-user.target.wants/apache2.service -> /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service -> /lib/systemd/system/apache-htcacheclean.service.
Setting up libapache2-mod-php (2:7.4+76) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for libc-bin (2.31-13+rpt2+rpi1+deb11u7) ...
Processing triggers for php7.4-cli (7.4.33-1+deb11u4) ...
Processing triggers for libapache2-mod-php7.4 (7.4.33-1+deb11u4) ...

rodrigo@raspberrypi:~$
```

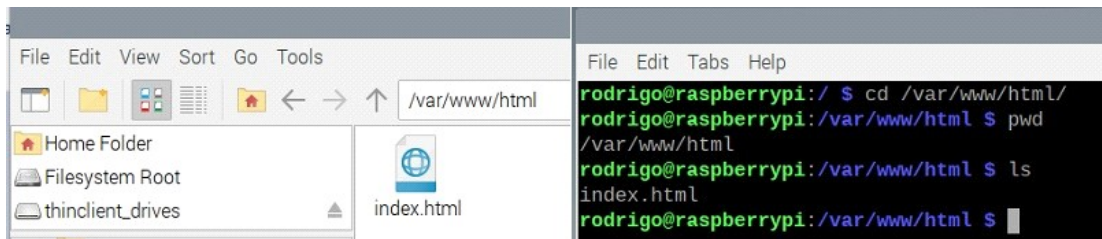
El primer comando instalara la paqueteria que permitira la interaccion con código PHP.

```
rodrigo@raspberrypi:~$ sudo apt-get install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.56-1-debian).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
rodrigo@raspberrypi:~$
```

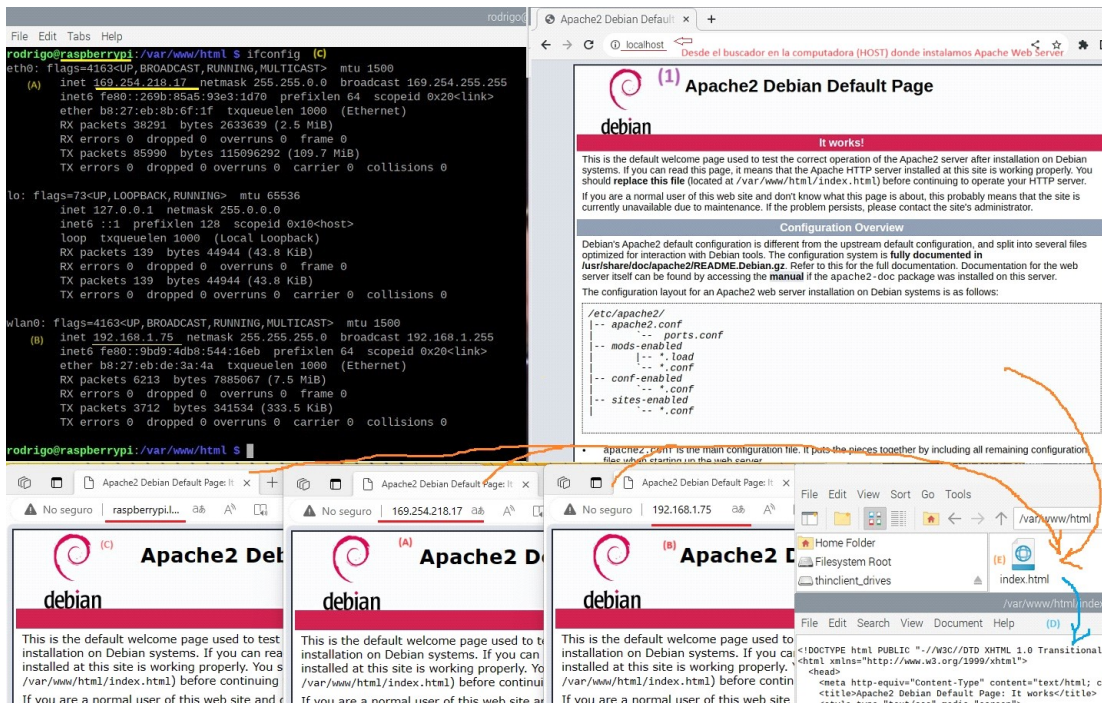
El segundo comando se asegura la instalacion de programa Apache, el cual creara un directorio de trabajo, conectado a la internet, ahi alojaremos nuestras paginas Web y estas seran visibles para otras computadoras através de sus navegadores de Internet.

La seguridad es importante, por lo que solo esta carpeta y ciertos archivos son visibles desde los navegadores externos, Apache se asegura de esto asi como de las conexiones necesarias.

Verificamos que nuestro servidor Apache este instalado correctamente, revisamos primero el directorio de trabajo:



Comprobamos tambien que pagina index.html se muestre correctamente en el navegador:



Como indica (1) (esquina superior derecha) debemos poder acceder desde el navegaro de computadora donde esta instalado el servidor APACHE, con el nombre de localhost, revisamos las ip asignadas a nuestro Servidor y verificamos que podemos contactarlo desde nuestro navegador externo en (A), (B) y

(C), note que en en (C) hemos usado el nombre de la computadora de nuestro server para poder acceder a el.

Nota: la seguridad es muy importante, este servidor no tiene salida a la Red, solo es "visible" desde la red local povista por nuestro moden inalambrico provisto por la compañía que nos brinda el servicio de internet, mediante la IP de (B), tambien podemos colocar un cable de ethernet para conectar directamente a la computadora mediante el puerto fisico (A), se deseamos "abrir" un canal de acceso de la Red a nuestro Servidor, debemos configurar nuestro modem-conmutador de nuestro proveedor, esto escapa del alcance de este documento.

2. Instalacion de MySQL

Ahora lo siguiente es instalar nuestro gestor de Bases de Datos, en este caso sera MySQL, pero antes necesitaremos configurar en instalar el software para conectar nuestra base de datos a nuestro ambiente de trabajo Web (creado anteriormente por Apache Server), asi como PHP un lenguaje de soporte para HTML que nos permitira interactuar con nuestras bases de datos desde un navegador web.

Instalamos el software de conexión a nuestra base de datos con el comando siguiente:

```
sudo apt-install mariadb-server -y
```

(la opcion -y responde a "priori" la pregunta sobre si deseamos destinar el espacio en disco a la instalación)

```
sudo apt-get install php-mysql -y
```

(para una explicación sobre los parametros del comando *apt* consulte la documentacion de Debian OS)

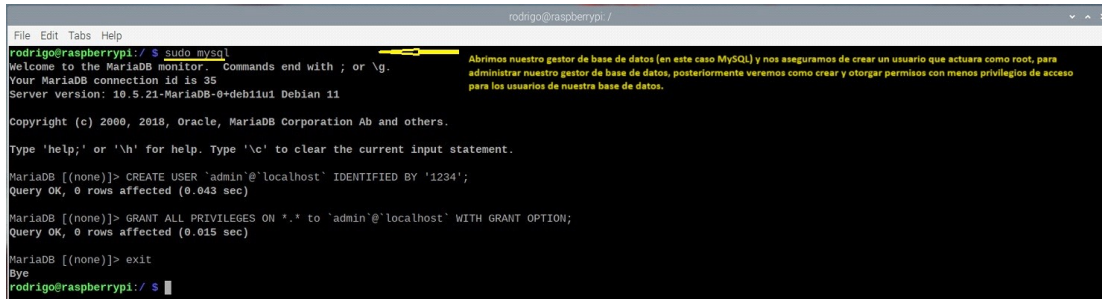
```
rodrigo@raspberrypi:~$ sudo apt install mariadb-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Instalamos el paquete light de mariadb para conectar nuestro servidor al
ambiente de trabajo generado por la aplicación Apache
The following additional packages will be installed:
  galera-4 gawk libfcgi-fast-perl libfcgi-pm-perl libclone-perl libconfig-inifiles-perl libdbd-mariadb-perl libdbi-perl libencode-locale-perl libfcgi-bin libfcgi-perl
  libfcgi10ldbl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libmariadb3
  libsigsegv2 libterm-readkey-perl libtimedate-perl liburi-perl mariadb-client-10.5 mariadb-client-core-10.5 mariadb-common mariadb-server-10.5
  mariadb-server-core-10.5 mysql-common socat
rodrigo@raspberrypi:~$ sudo apt-get install php-mysql -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Instalamos el paquete de software requerido para conectar nuestro gestor de bases de datos con PHP, el cual nos
permitirá acceder a nuestras bases de datos mediante el navegador Web, mas delante se explicara como usar este lenguaje para
conectar con nuestras bases de datos.
The following additional packages will be installed:
  php7.4-mysql
The following NEW packages will be installed:
  php-mysql php7.4-mysql
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 118 kB of archives.
After this operation, 475 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bullseye/main arm64 php7.4-mysql arm64 7.4.33-1+deb11u4 [112 kB]
Get:2 http://deb.debian.org/debian bullseye/main arm64 php7.4-mysql arm64 7.4.33-1+deb11u4
Err:2 http://deb.debian.org/debian bullseye/main arm64 php-mysql all 2:7.4+76
  Cannot initiate the connection to deb.debian.org:80 (2a04:4e42:8a::644). - connect (101: Network is unreachable) Could not connect to deb.debian.org:80 (146.75.106.132)
). - connect (113: No route to host) [IP: 146.75.106.132 80]
Get:1 http://deb.debian.org/debian bullseye/main arm64 php7.4-mysql arm64 7.4.33-1+deb11u4 [112 kB]
Fetched 47.0 kB in 1min 42s (459 B/s)
E: Failed to fetch http://deb.debian.org/debian/pool/main/p/php-defaults/php-mysql_7.4%2b76_all.deb Cannot initiate the connection to deb.debian.org:80 (2a04:4e42:8a::644)
- connect (101: Network is unreachable) Could not connect to deb.debian.org:80 (146.75.106.132). - connect (113: No route to host) [IP: 146.75.106.132 80]
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-missing?
rodrigo@raspberrypi:~$ sudo apt-get update --fix-missing
Hit:1 http://security.debian.org/debian-security bullseye-security InRelease
Hit:2 http://deb.debian.org/debian bullseye InRelease
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Hit:4 http://archive.raspberrypi.org/debian bullseye InRelease
Fetched 44.1 kB in 3s (14.6 kB/s)
Reading package lists... Done
rodrigo@raspberrypi:~$
```

En ocasiones pueden surgir problemas durante la instalacion, en este caso se nos pide actualizar nuevamente nuestro equipo, debido a que el software que estamos instalando requiere nuevas dependencias.

Procedemos a instalar nuestro gestor de bases de datos, en este caso MySQL (Postgress tambien funcionara igual, solo hay que descargar el software apropiado para configurarlo y conectarlo a nuestro ambiente de trabajo Apache). Entramos a nuestro SGBD:

`sudo mysql`

Tiene la configuracion "Default", sin usuario alguno, configuramos un usuario con autoridad **root**:



```
rodrigo@raspberrypi:/$ sudo mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 35
Server version: 10.5.21-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]> CREATE USER 'admin'@'localhost' IDENTIFIED BY '1234';
Query OK, 0 rows affected (0.043 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'admin'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.015 sec)

MariaDB [(none)]> exit
Bye
rodrigo@raspberrypi:/$
```

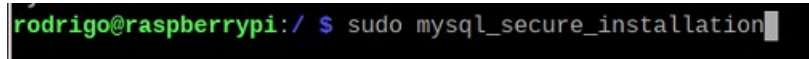
Creamos un usuario y lo dotamos de privilegios con los comandos siguientes de SQL:

`CREATE USER `admin`@`localhost` IDENTIFIED BY '1234';`

`GRANT ALL PRIVILEGES ON *.* to `admin`@`localhost` WITH GRANT OPTION;`

`exit`

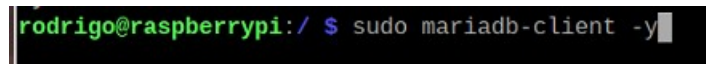
salimos de nuestro SGBD, y abremos terminado de configurar el acceso a nuestras bases de datos al nuestro usuario **root**.



```
rodrigo@raspberrypi:/$ sudo mysql_secure_installation
```

Es posible lanzar la configuracion de manera interactiva con el comando anterior, pero nos haran mas preguntas adicionales para la configuracion de nuestro SGBD, las cuales escapan del alcance de este Documento, el lector puede revisar la documentacion de MySQL y tomar este camino si lo desea.

Adicionalmente tambien puede instalar el paquete siguiente:



```
rodrigo@raspberrypi:/$ sudo mariadb-client -y
```

Para obtener mayor funcionalidad en la coneccion entre su SGBD y el area de trabajo Apache Server.

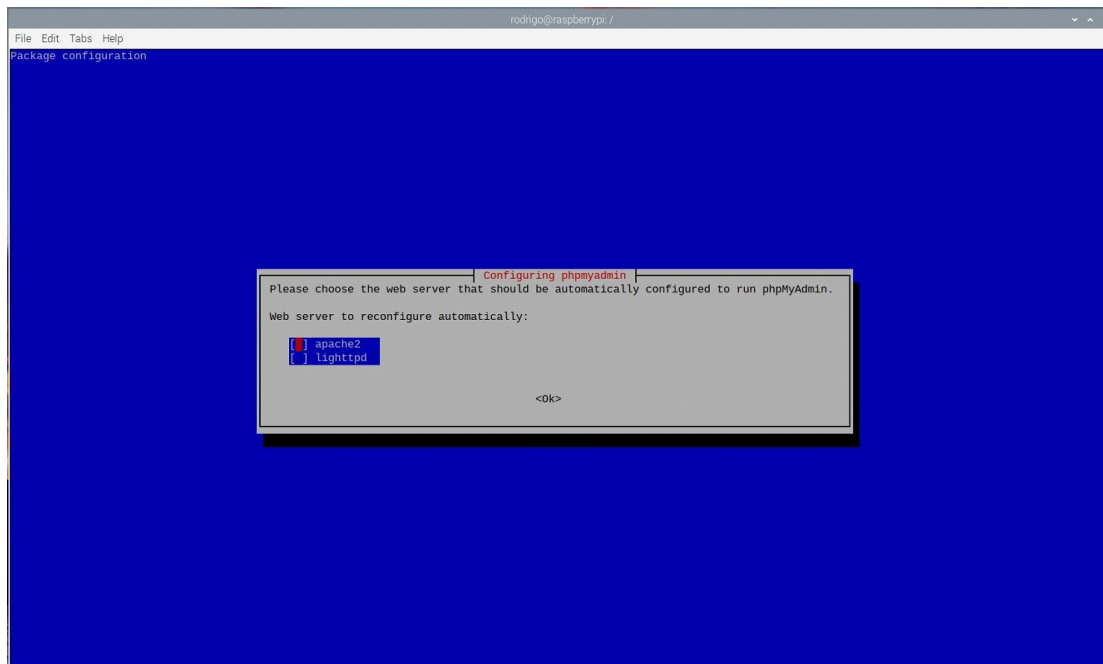
2. Instalacion de PHP admin

En vez de instalar solo el lenguaje PHP que nos permitira conectar a nuestro SGBD mediante nuestro navegador, aprovecharemos e instalaremos una herramienta que nos permitira dar un mantenimiento basico a nuestras bases de datos, sin tener que acceder directamente a nuestro SGBD, ofreciendo hasta cierto punto una capa mas de protecci3n.

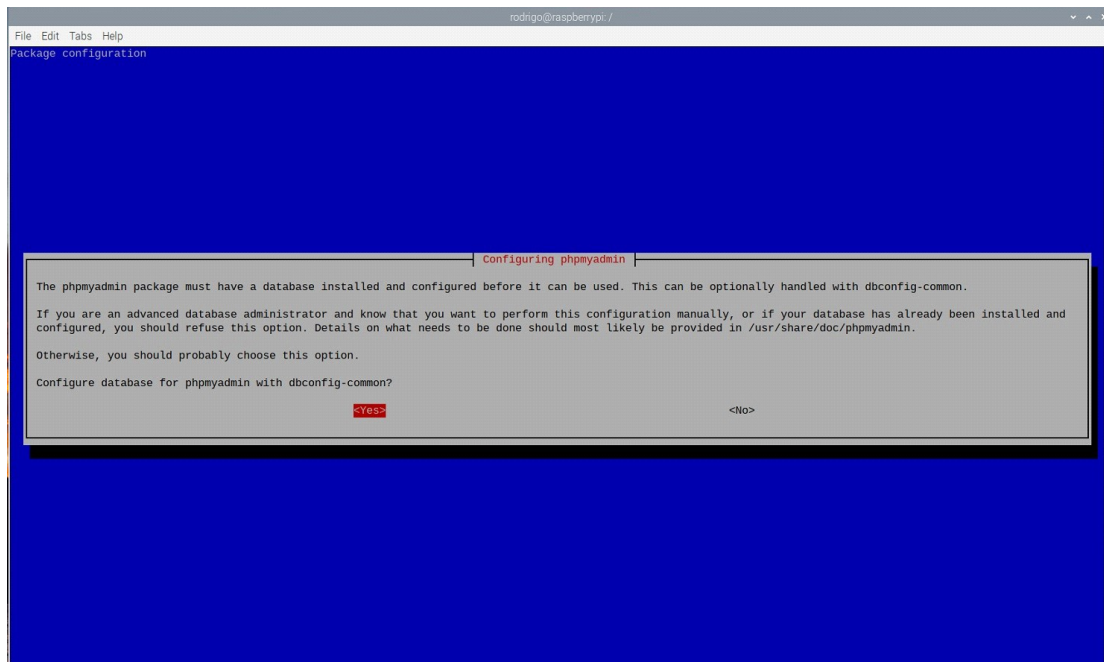
```
rodrigo@raspberrypi: ~  
File Edit Tabs Help  
rodrigo@raspberrypi:~ $ sudo apt-get install phpmyadmin -y
```

El comando `sudo apt-get install phpmyadmin` instalara la aplicacion.

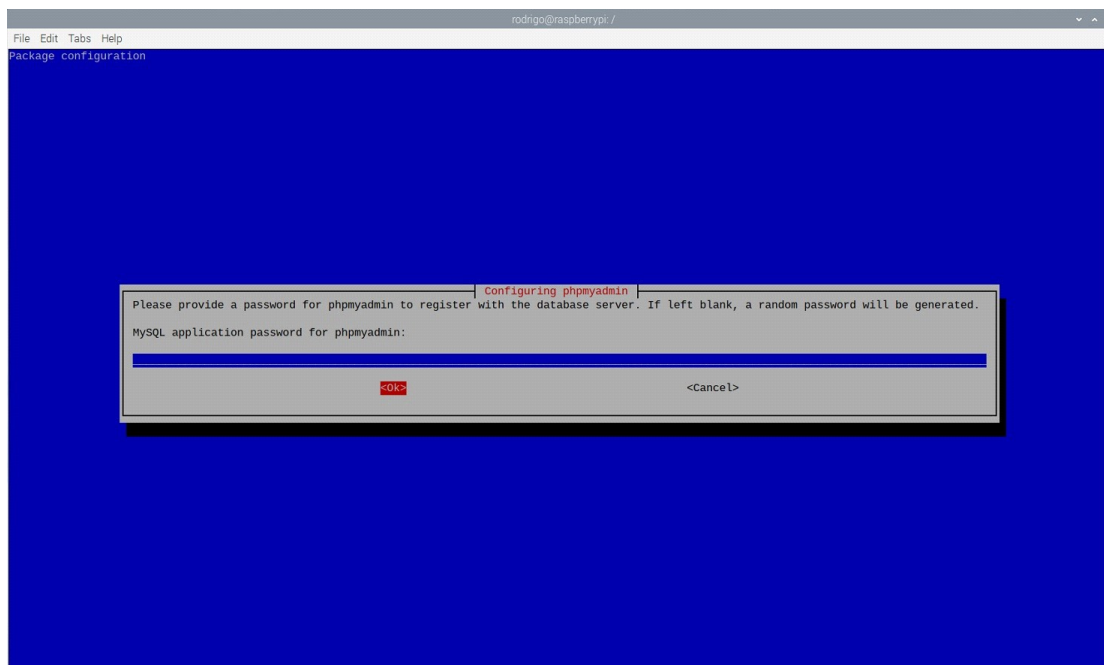
Durante la instalacion se nos pedira configurar alguno ajustes a nuestro software le primero consiste en indicar que tipo de aplicacion nos da "salida" hacia el navegador, en nuestro caso elegimos la opcion Apache2, presionamos la tecla barra espaciadora para seleccionar la casilla, seguida de la tecla tabulador o las flechas del teclado para seleccionar OK, prsionamos Enter para completar:



La siguiente seccion nos pregunta si deseamos configurar los ajustes finos de PHPmyAdmin o una instalacion comun, elegiremos la opcion YES (si el lector sabe lo que esta haciendo puede configurar manualmente eligiendo la opcion no).



Se nos pedira crear un usuario maestro de acceso, dejamos este espacio en blanco, debido a que anteriormente ya configuramos un usuario Root en MySQL, el cual usaremos para acceder.

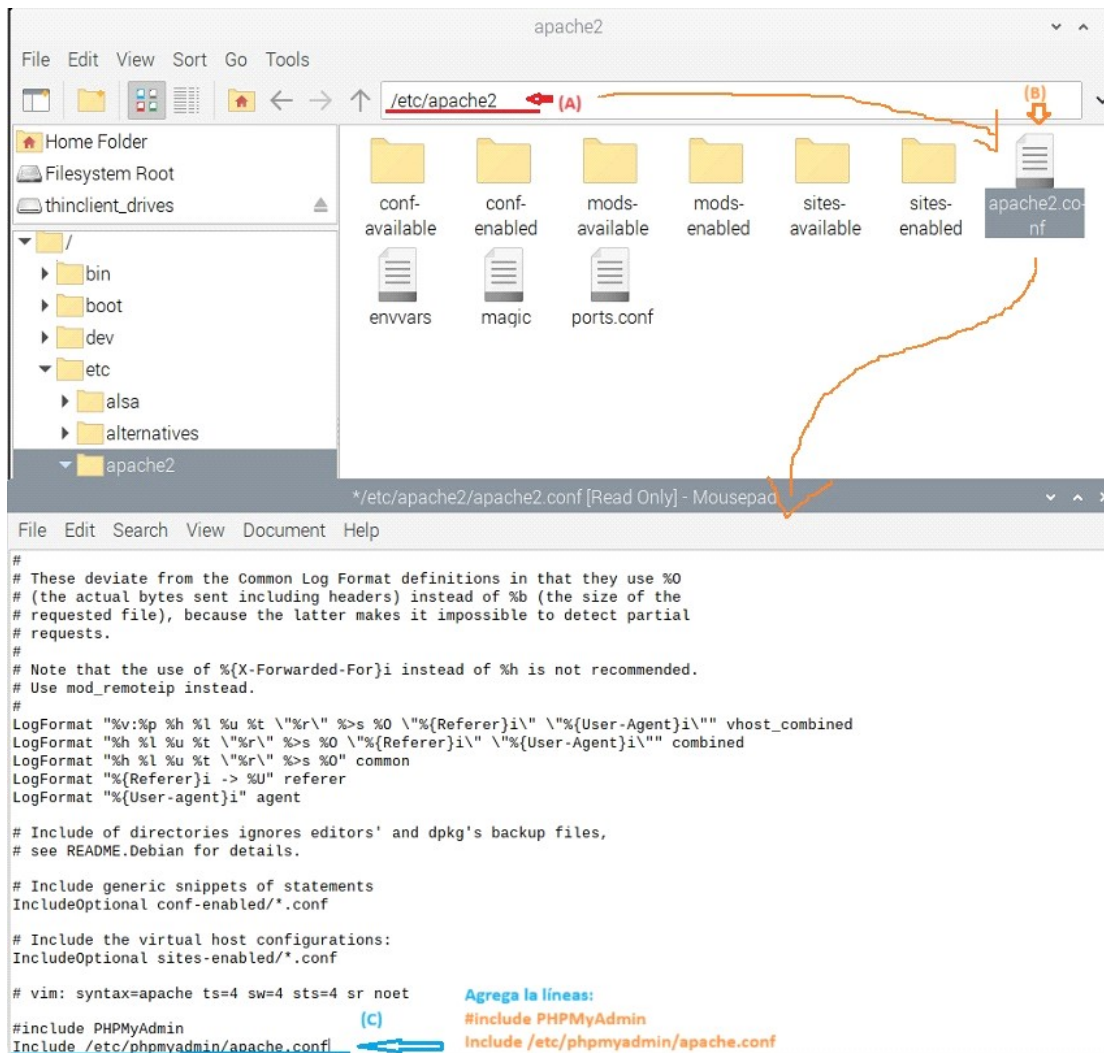


Una vez terminada la configuracion debemos agregar el software PHPMyAdmin al archivo de configuracion de Apache2, para evitar tener problemas de acceso remoto por el navegador. no movemos a la carpeta:

`/etc/apache2` (A) buscamos el archivo `apache2.conf` (B) lo abrimos con el editor de texto y agregamos al final (C) las lineas siguientes:

```
#include PHPMyAdmin
```


Include /etc/phpmyadmin/apache.conf



También podemos realizar la acción desde la Terminal, en caso de que no se nos permita escribir en el archivo o simplemente para agilizar el proceso:

sudo vi /etc/apache2/apache2.conf

y al final del archivo agregamos las dos líneas anteriores:

#include PHPMyAdmin

Include /etc/phpmyadmin/apache.conf

```
rodrigo@raspberrypi:~$ sudo vi /etc/apache2/apache2.conf
rodrigo@raspberrypi:~$

# directive.
#
AccessFileName .htaccess

#
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<FilesMatch "\.ht$">
    Require all denied
</FilesMatch>

#
# The following directives define some format nicknames for use with
# a CustomLog directive.
#
# These deviate from the Common Log Format definitions in that they use %O
# (the actual bytes sent including headers) instead of %b (the size of the
# requested file), because the latter makes it impossible to detect partial
# requests.
#
# Note that the use of %{X-Forwarded-For}i instead of %h is not recommended.
# Use mod_remoteip instead.
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf

# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

#Include PHPMyAdmin
include /etc/phpmyadmin/apache.conf
rodrigo@raspberrypi:~$ sudo reboot
```

En este caso usamos VI como editor, pero puede ser el que el lector elija

Cuando el editor se abra ir al final del archivo para agregar unas líneas adicionales

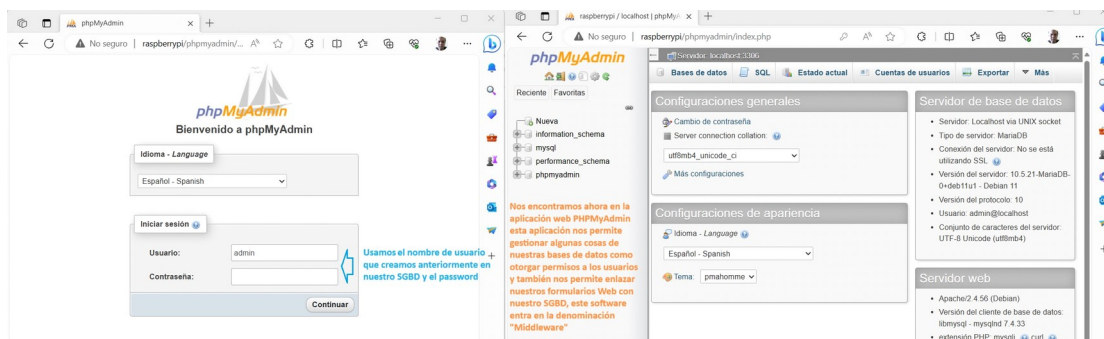
Agregamos estas dos líneas:
La primera es un comentario sobre el significado de la segunda línea
En la segunda línea pedimos que Apache agregue el archivo de configuración apache.conf para el software PHPMyAdmin

Es recomendable reiniciar para asegurar que se apliquen los cambios de configuración

Tras terminar de configurar nuestra aplicación, recomendamos reiniciar con:

sudo reboot

Para asegurar que los cambios fueron efectuados, debemos revisar si nuestro Servidor es accesible, el servidor solo sera "visible" dentro de nuestra red local, la cual es provista por nuestro Modem-Commutador de nuestro proveedor de Internet. Abrimos el navegador de cualquier equipo conectado a la red (de preferencia otro que no sea el servidor), y usamos cualquiera de las rutas que utilizamos anteriormente para verificar la conexión con Apache, en la figura nos referimos al servidor por su nombre (en este caso *raspberrypi*):



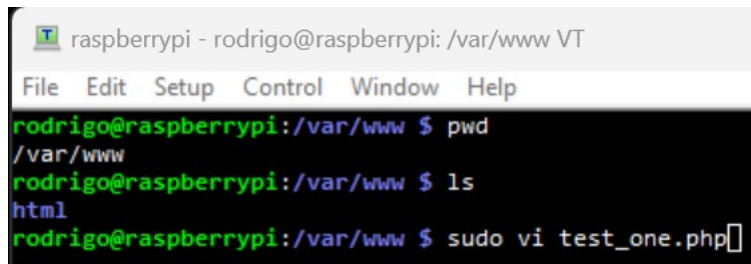
Agregamos a la dirección el sufijo *phpmyadmin* como en la imagen anterior:

raspberrypi/phpmyadmin/index.php

Se nos pedira ingresar usuario y password, usamos los que definimos en MySQL, ahora en la pagina siguiente veremos la pagina completa de index.php, este centro de control nos permite gestionar cosas basicas de nuestras Bases de Datos, agregar usuarios y hacer respaldos de seguridad, por mencionar algunas tareas. este software entra dentro de la categoria MiddleWare en un esquema FullStack, MySQL sería lo que se conoce como BackEnd, nos centraremos en esa sección en la parte siguiente de este tutorial cuando creamos nuestra base de datos de muestra. Posteriormente una vez creada nos centraremos en la parte que se mostrara a los Usuarios de Nuestra Base de Datos, mediante un formulario que permita las operaciones basicas (y a la vez resguarde la informacion), usando el argot trabajaremos en el FrontEnd.

Verificacion de funcionamiento de PHP en nuestro Sevidor

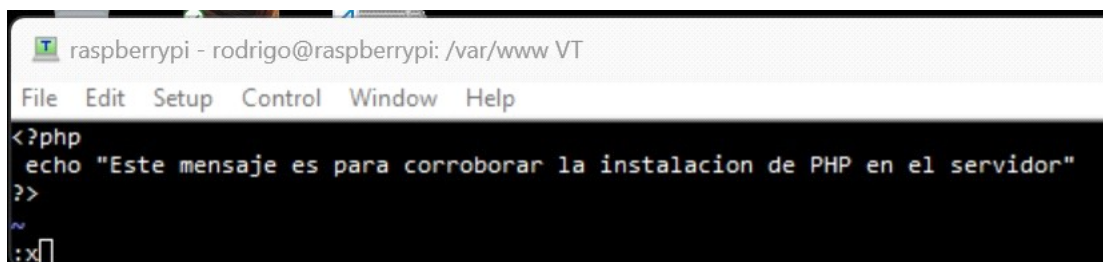
Crearemos un archivo de prueba usando el lenguaje PHP, para comenzar a entender el funcionamiento del diseño de paginas Web, nos movemos a la carpeta `/var/www` :



```
raspberrypi - rodrigo@raspberrypi: /var/www VT
File Edit Setup Control Window Help
rodrigo@raspberrypi:/var/www $ pwd
/var/www
rodrigo@raspberrypi:/var/www $ ls
html
rodrigo@raspberrypi:/var/www $ sudo vi test_one.php
```

En la ubicación creamos un archivo de prueba llamado "`test_one.php`" usamos **sudo** para poder escribir en la carpeta, **nota:** la carpeta WWW es accesible desde un explorador, por lo que debe contar con los permisos correctos de acceso.

Escribimos el siguiente fragmento de código usando el editor Vi en este caso:



```
raspberrypi - rodrigo@raspberrypi: /var/www VT
File Edit Setup Control Window Help
<?php
echo "Este mensaje es para corroborar la instalacion de PHP en el servidor"
?>
~
:x
```

Grabamos el archivo, y nos vamos a nuestro explorador web preferido desde otro computadora que no sea la Raspberry Pi y que tenga acceso a la red donde esta nuestro Web Server.



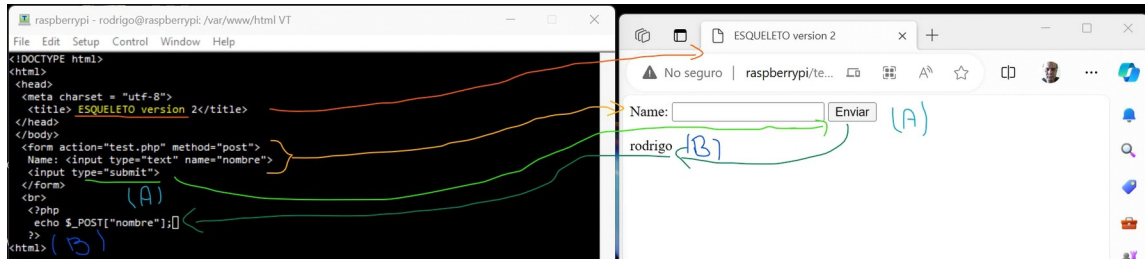
En la imagen anterior se muestra la direccion `raspberrypi/test_one.php` la cual corresponde a nuestro servidor y especificamos el archivo creado anteriormente, podemos ver que muestra el mensaje de prueba, hemos corroborado que nuestro Servidor esta funcionando correctamente.

NOTA: Cada que se realice un cambio en el archivo, se debe guardar y la pagina del explorador debe "refrescarse" mediante el icono para esta accion o la tecla F5, de otra manera no se observaran los

cambios, recuerde que no estamos compilando código alguno.

La relacion HTML y PHP

PHP fue creado para dar mas funcionalidad al lenguaje HTML por lo que envolvera nuestro código PHP para conectar a la base de datos, antes de proceder a la implementacion, debemos crear nuestro Cascaron básico de HTML:



izquierda terminal de nuestra raspberry pi, derecha explorar Web

El proceso de creacion sera el mismo, enviaremos todos nuestros archivos (.php) a la carpeta `var/www/html/` y usaremos el Explorador Web para visualizar el efecto en la pagina `raspberrypi/nombreArchivo.php`, en la imagen anterior podemos ver que dentro de las marcas:

(A)

`<title> ESTE ES EL ESQUELETO</title>` esta el nombre que se muestra en la pestaña del explorador Web.

(B)

`<input type = "submit">` Código que crea un boton para que la forma realice una acción, en este caso toma el contenido del cuadro de texto y lo envia mediante el método POST hacia el código PHP, que se encarga de mostrar el contenido de vuelta:

`<?php echo $_POST["nombre"]; ?>`

Uniando nuestra pagina Web con nuestra base de datos

Creamos una base de datos sencilla para mostrar la conexion con PHP:

```
raspberrypi - rodrigo@raspberrypi: /var/www/html VT
File Edit Setup Control Window Help
MariaDB [(none)]> CREATE USER 'usuariodb'@'localhost' IDENTIFIED BY '1234';
Query OK, 0 rows affected (0.029 sec)

MariaDB [(none)]> CREATE DATABASE phptest;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> GRANT ALL ON phptest.* TO 'usuariodb'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.047 sec)

MariaDB [(none)]> quit
Bye
rodrigo@raspberrypi:/var/www/html $ mysql -u usuariodb -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 32
Server version: 10.5.21-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input state ment.
```

- (A) -Creamos el usuario: *usuariodb* con password 1234
- (B) -Creamos una base de datos de prueba *phptest*
- (C) -Garantizamos todos los permisos al usuario creado solo para la base de datos de prueba.
- (D) -Salimos para que los cambios se efectuen (tambien puede usar FLUSH)
- (G) -Ingresamos con el usuario de prueba y su clave, para entrar a la base de datos de prueba.


```

MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| phptest |
+-----+
2 rows in set (0.002 sec)

MariaDB [(none)]> USE phptest; (A)
Database changed
MariaDB [phptest]> CREATE TABLE nombres(nombre VARCHAR(50)); (B)
Query OK, 0 rows affected (0.043 sec)

MariaDB [phptest]> DESCRIBE nombres; (C)
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nombre | varchar(50) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.006 sec)

MariaDB [phptest]> 

```

(A)- Ingresamos a la base de datos de prueba

(B)- Creamos una tabla sencilla con la cual interactuar

(C)- Revisamos la creacion de nuestra Tabla

```

MariaDB [phptest]> INSERT INTO nombres (nombre) VALUES ('Rodrigo');
Query OK, 1 row affected (0.007 sec) (A)

MariaDB [phptest]> SELECT * FROM nombres; (B)
+-----+
| nombre |
+-----+
| Rodrigo |
+-----+
1 row in set (0.001 sec)

MariaDB [phptest]> quit (C)

```

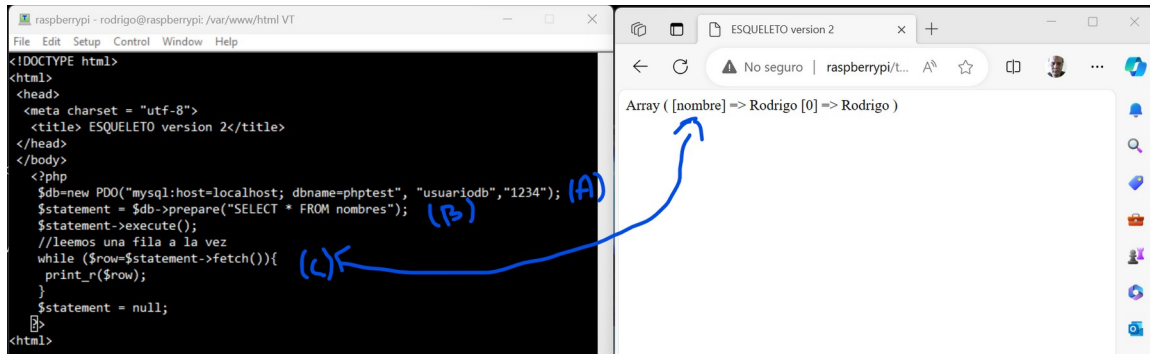
(A)- Insertamos un valor en la tabla

(B)- Verificamos la insercion del dato

(C)- Salimos del SGBD

En la terminal nos movemos hasta la carpeta de edicion de paginas Web (var/www/html), creamos un archivo de prueba *test.php*.

Comenzamos a ingresar el siguiente código:



(A)-Creamos la coneccion con la base de datos.

(B)-Enviamos la Query al SGBD

(C)-Recuperamos la respuesta del SGBD y lo mostramos en nuestra pagina Web.