**Hugo Alejandro Hernandez Cardenas**

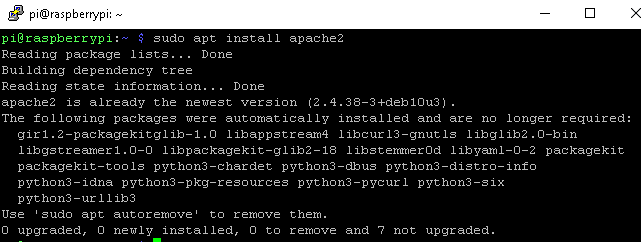
**000190699**

**Documentación Implementación APACHE – PHP – MARIADB**

1. **APACHE – PHP**

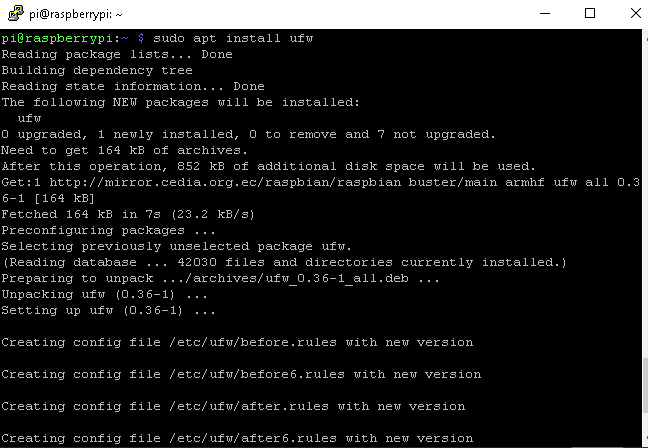
Para poder que el API funcione se debe instalar apache y habilitar las conexiones

*sudo apt install apache2*



El UFW (cortafuegos no complicado) debe instalarse de manera predeterminada en Ubuntu y Debian; de lo contrario, instálelo con el administrador de paquetes APT con el siguiente comando.

*sudo apt install ufw* (Para instalarlo)y luego *sudo allow ‘Apache Full’* (habilitarlo)



Ahora procedemos a instalar el PHP en la Raspberry:

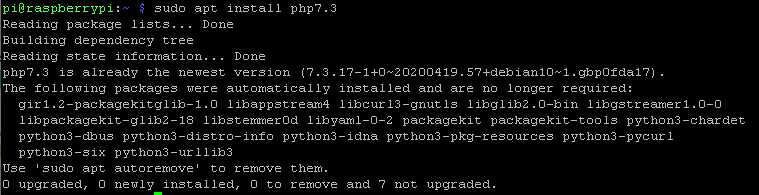
Copiamos el siguiente comando para habilitar los debidos permisos para nuestro servicio PHP

*sudo sh -c 'echo "deb https://packages.sury.org/php/ $(lsb\_release -sc) main" > /etc/apt/sources.list.d/php.list'*



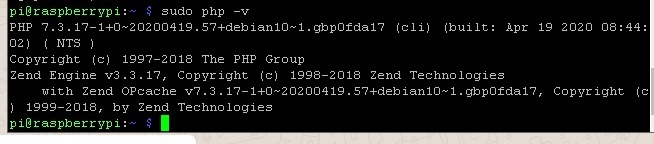
Una vez realizado el paso de permisos, seguimos con la instalación del PHP:

*sudo apt install php7.3*



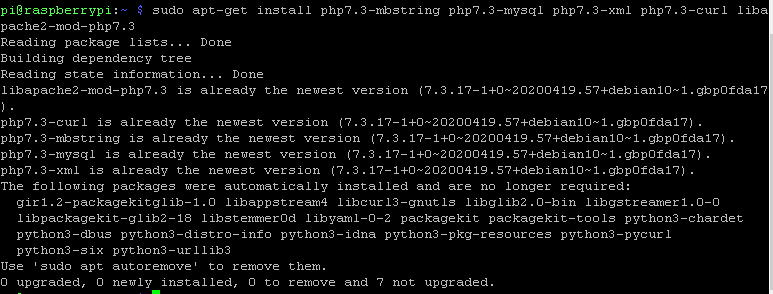
Para verificar la versión de PHP anteriormente instalado ejecutamos el siguiente comando:

*sudo php -v*

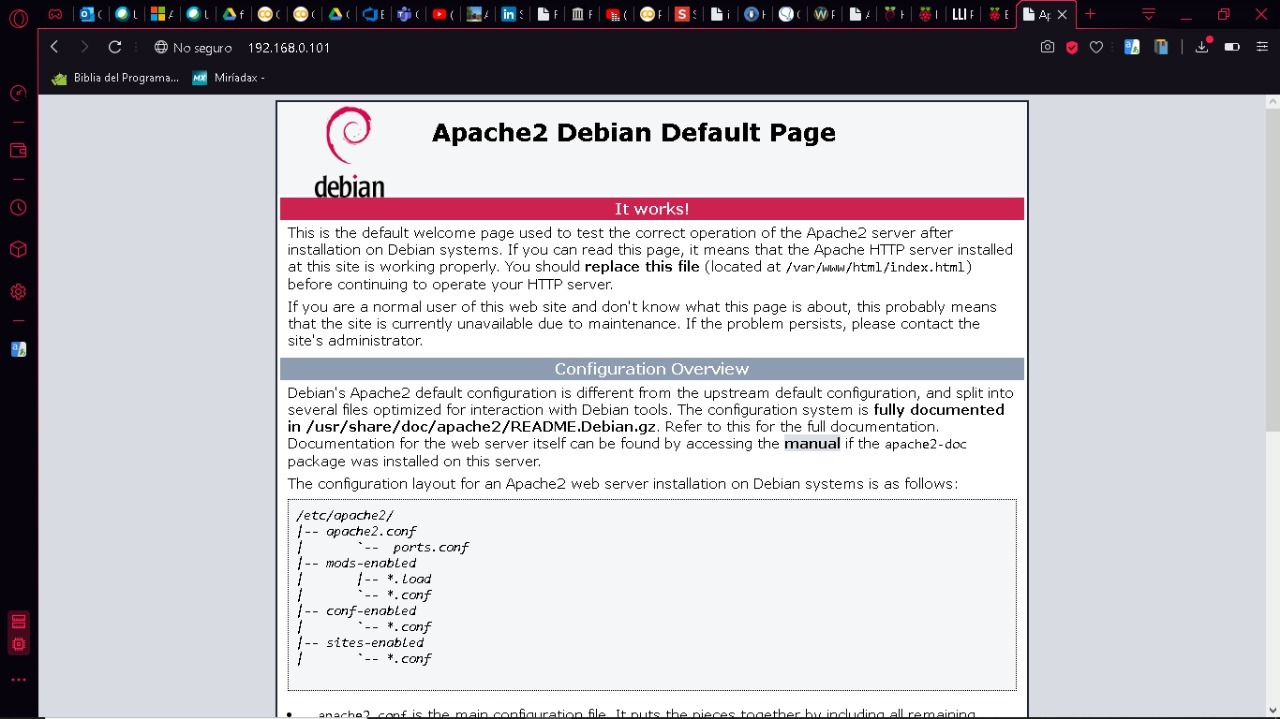


Terminamos de instalar los paquetes del PHP en la Raspberry:

*sudo apt-get install php7.3-mbstring php7.3-mysql php7.3-xml php7.3-curl libapache2-mod-php7.3*



En este instante ya tenemos lista nuestra pagina web que trae por defecto, para verificar ponemos en el buscador de cualquier navegador <https://192.168.0.101> (Seria la dirección IP del Raspberry).



Ahora creamos un archivo de prueba para ver que todos los paquetes estén correctamente instalados y funcionales. Tener en cuenta que estos archivos solo son de prueba por ende luego serán eliminados por motivos de seguridad en el sistema, esto es porque contienen información delicada o confidencial del servidor web.

Para crear un archivo de prueba ejecutamos el siguiente comando:

*sudo nano /var/www/html/info.php*

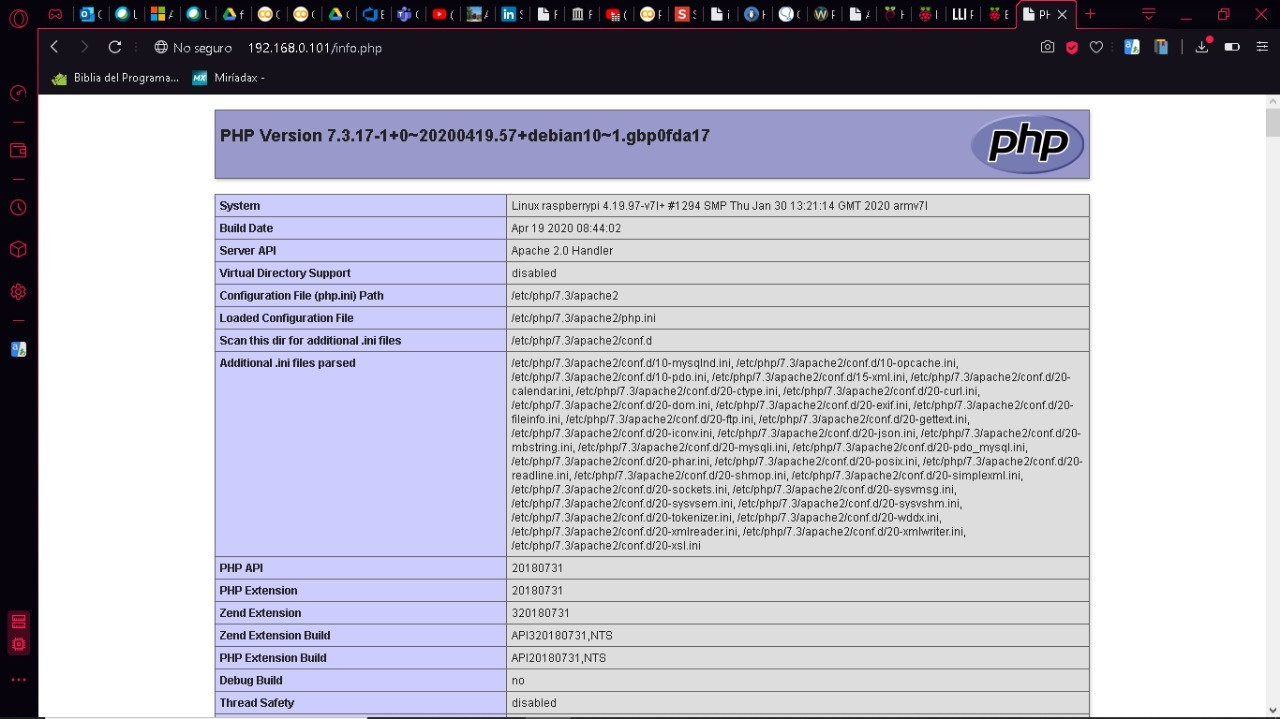
Una vez entremos al archivo de prueba, escribimos el siguiente código y lo guardamos:

*<?php*

*phpinfo();*

*?>*

Luego vamos al navegador <https://192.168.0.101/info.php>, como se puede observar



Ahora eliminamos todos los archivos que tenga por defecto dicha ruta, en este caso index.php y nuestro archivo de prueba info.php

*sudo rm -R /var/www/html/info.php*

*sudo rm -R /var/www/html/index.php*

Ahora con el php y el apache funcional procedemos a descargar el archivo con el APIREST [aqui](http://goo.gl/7kPWWP), luego se pasa el .zip al servidor por medio de FTPS (FilleZilla)

Para mirar que, si se hizo correctamente la trasferencia de archivos, ejecutamos:

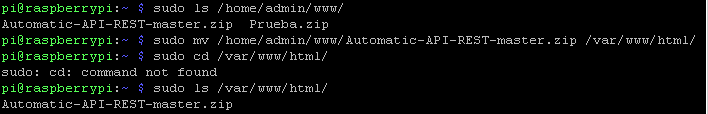
s*udo ls /home/admin/www*

Luego para tenemos que cambiar la ruta del archivo .zip, para lo cual ejecutaremos lo siguiente:

*sudo mv /home/admin/wwwAutomatic-API-REST-master.zip /var/www/html/*

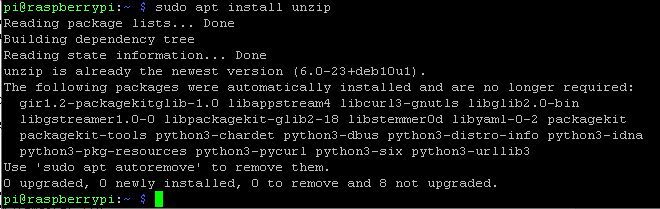
Luego miramos que si se halla movido el archivo:

*sudo ls /var/www/html*



Para descomprimir nuestro archivo tenemos que instalar primero el programa unzpi, por medio del comando:

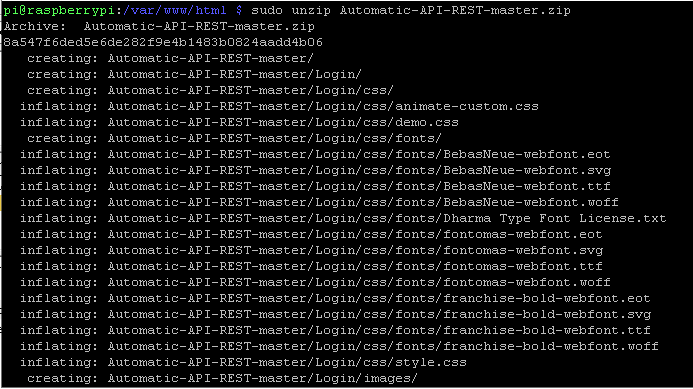
*sudo apt install unzip*



Una vez con el programa unzip instalado procedemos a descomprimir nuestro archivo con el APIREST:

*cd /var/www/html*

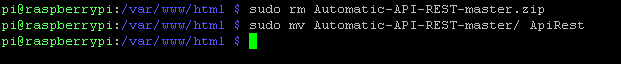
*sudo unzip Automatic-API-REST-master.zip*



Ahora eliminamos el .zip que no nos hace falta y procedemos a cambiar la ubicación del archivo Automatic-API-REST-master a ApiRest

*sudo rm Automatic-API-REST-master.zip*

*sudo mv Automatic-API-REST-master / ApiRest*



Ahora vamos al siguiente archivo para configurar el USER y PASS del login de nuestra ApiRest:

*sudo nano ApiRest/config.php*

Ahora copiamos el siguiente código:

*define("SERVER","localhost");*

*define("USER","admin");*

*define("PASS","1234");*

*define("DB","EduApps");*

*define("USERADMIN","admin");*

*define("PASSADMIN","admin");*



Luego procedemos a mover otro archivo con el siguiente comando:

*sudo mv ApiRest/Server-Configuration/.htaccess .htaccess*



Ahora nos ubicamos en el archivo

*sudo nano .htaccess*

Y copiamos el siguiente código:

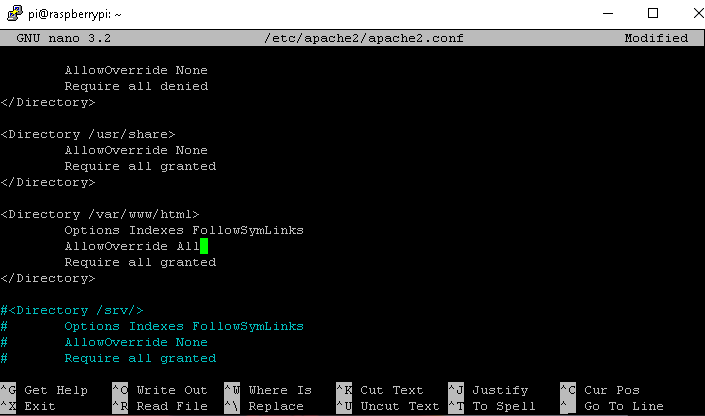
*RewriteEngine on*

*RewriteRule ^api/ Automatic-API-REST/api.php*



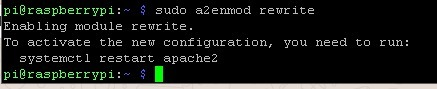
Ahora vamos al archivo de configuraciones de apache y dejamos el archivo con la siguiente configuración:

*sudo nano /etc/apache2/apache2.conf*



Guardamos los cambios realizados anteriormente en el archivo de configuraciones de apache y ejecutamos el siguiente comando para concretar todos los cambios:

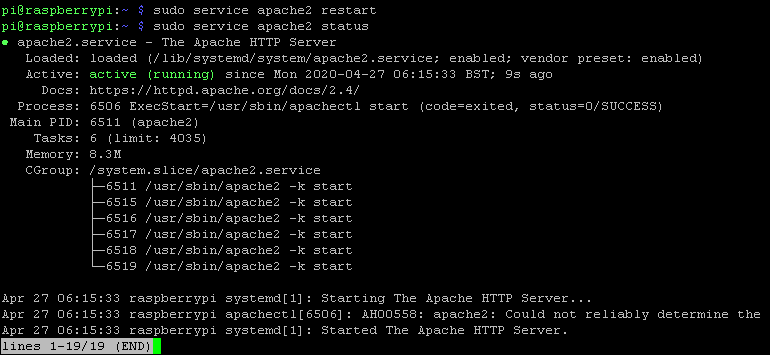
*sudo a2enmod rewrite*



Luego se reinicia el servidor apache y miramos su estado:

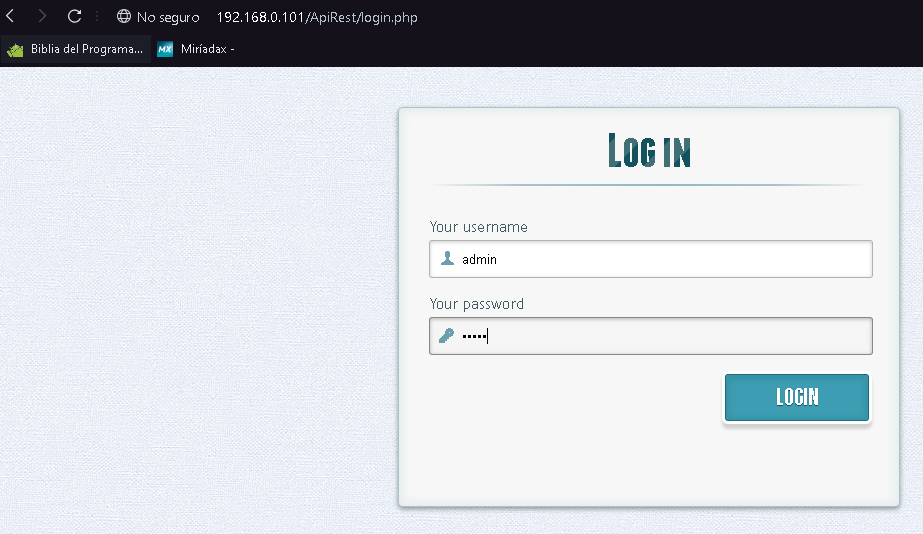
*sudo service apache2 restart*

*sudo service apache2 status*

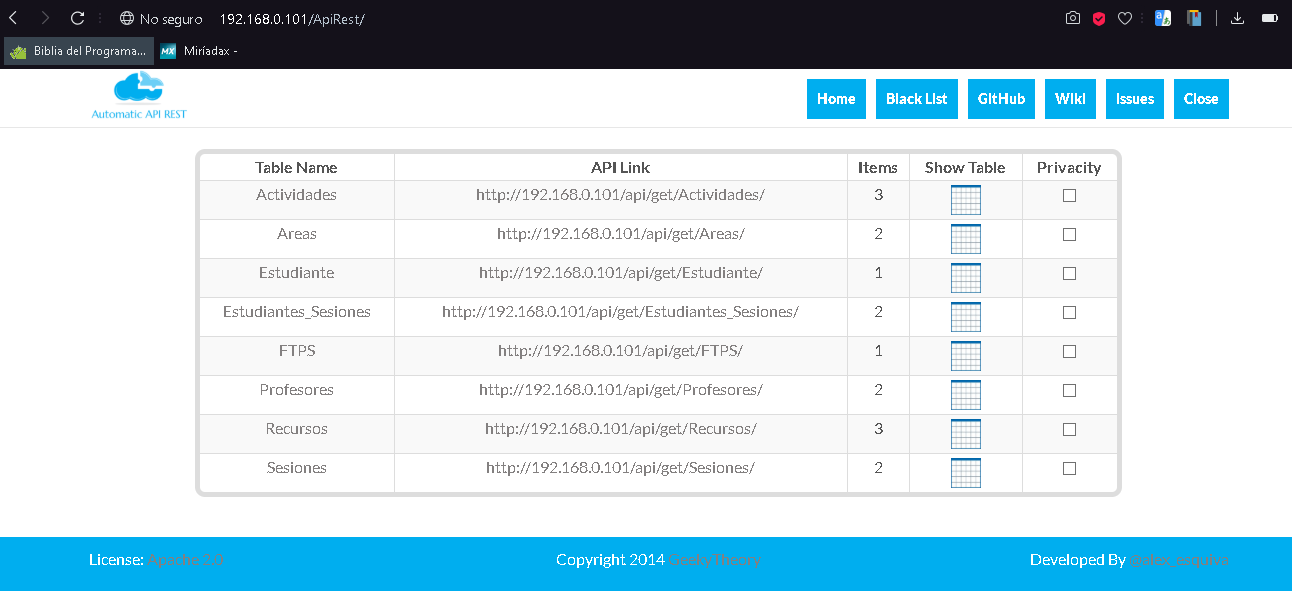


Luego se ingresa al navegador ***http://192.168.0.101/ApiRest*** se procede hacer el LogIn con los datos definidos, si todo ha salido bien se verán las tablas de la base de datos y al dar click en una se verán las columnas

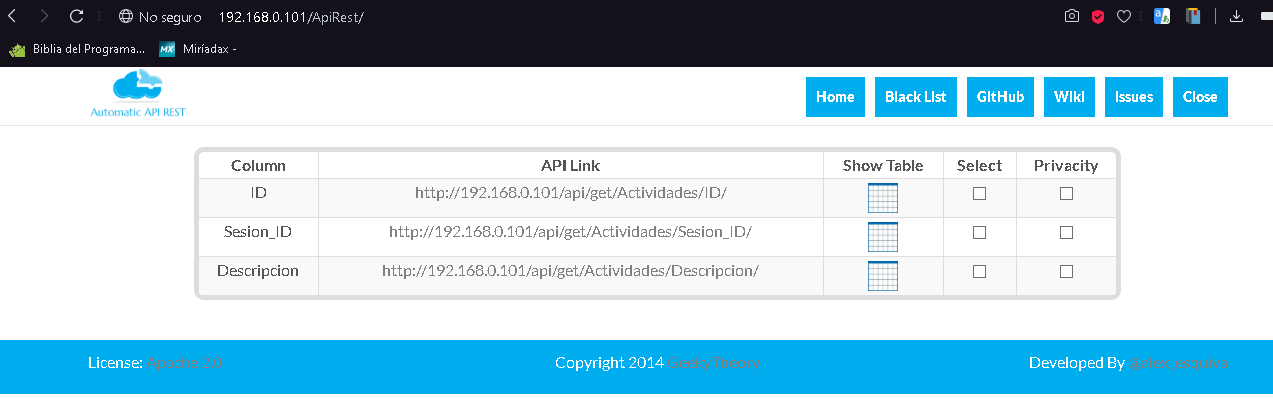
https://192.168.0.101/ApiRest



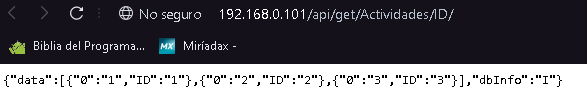
USER admin PASS admin



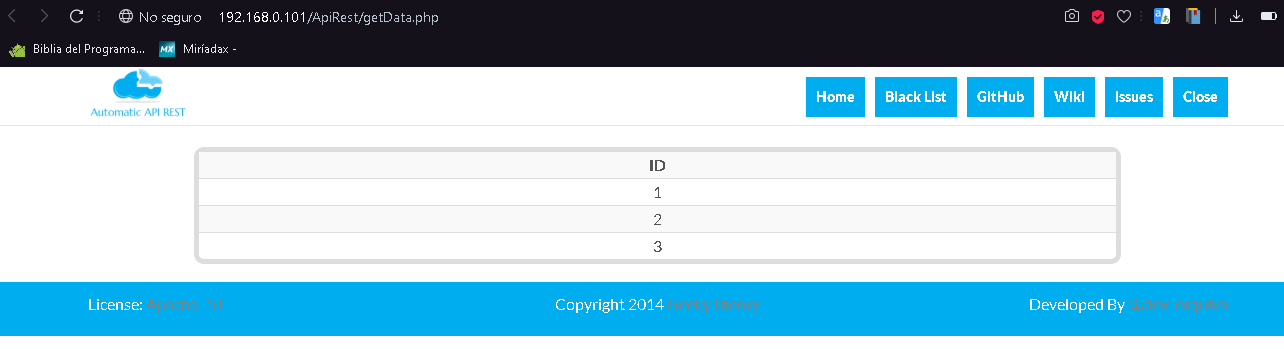
Vamos a la tabla Actividades



Ahora vamos al API Link



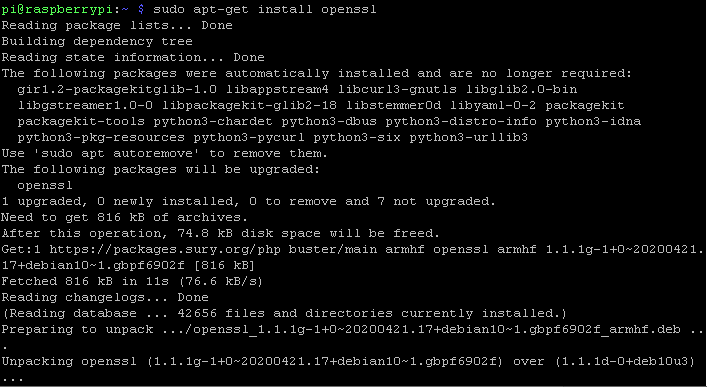
Cerramos dicha ventana y le damos en Show Table



1. **SERVIDOR FTPS**

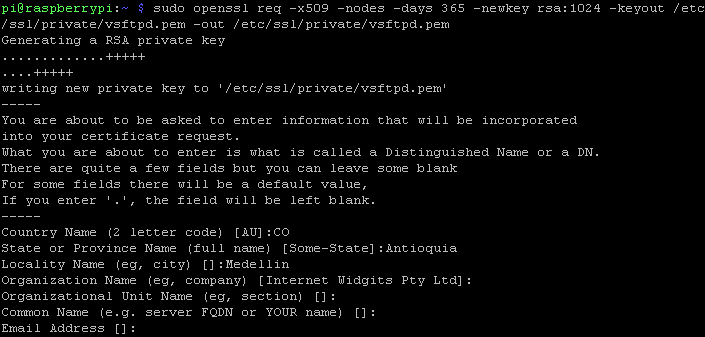
Una vez se tenga el servidor FTP funcionando, para hacer que soporte autentificación por TLS/SLL para cifrar la conexiones, primero se debe tener instala OpenSSL (para obtener certificados)

*sudo apt-get install openssl*



Se crea el certificado:

*sudo openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout /etc/ssl/private/vsftpd.pem -out /etc/ssl/private/vsftpd.pem*



Preguntará por algunos datos para incluir en el certificado (se ponen cualquiera) y luego se procede a editar el archivo vsftpd.conf descomentariando o agregando las siguientes lineas

*sudo nano /etc/vsftpd.conf*

Una vez en el archivo ingresamos o descomentamos lo siguiente:

*rsa\_cert\_file=/etc/ssl/private/vsftpd.pem*

*rsa\_private\_key\_file=/etc/ssl/privete/vsftpd.pem*

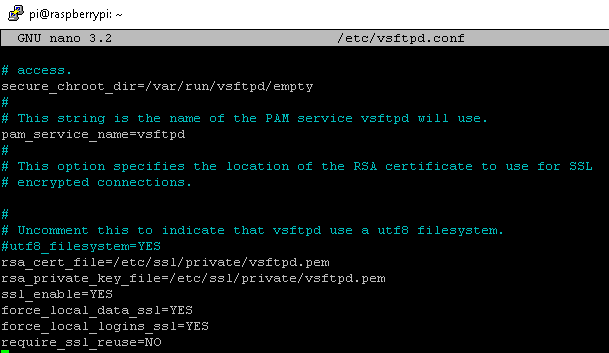
*ssl\_enable=YES*

*require\_ssl\_reuse=NO*

*force\_local\_data\_ssl=YES*

*force\_local\_logins\_ssl=YES*

*require\_ssl\_reuse=NO*



Guardamos los cambios y reiniciamos el servidor.

*sudo service vsftp restart*

Ahora vamos a crear un usuario para el servidor FTPS

* Crear usuario admincon ftpaccess grupo y /usr/bin/nologin shell.

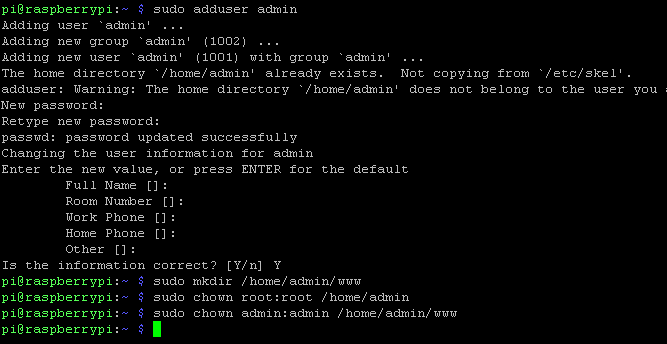
*sudo adduser admin*

* Crear una carpeta dentro de home para escribir y cambiar la propiedad de esa carpeta.

*sudo mkdir /home/admin/www*

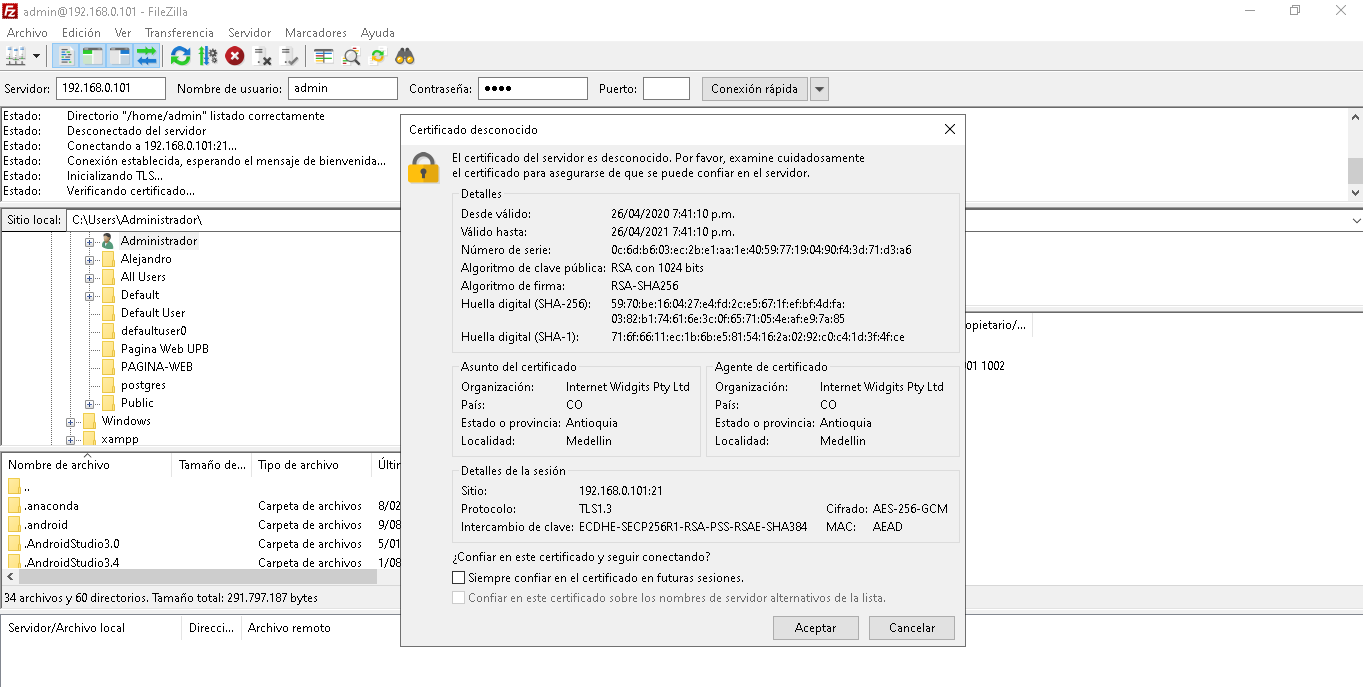
*sudo chwn root:root /home/admin*

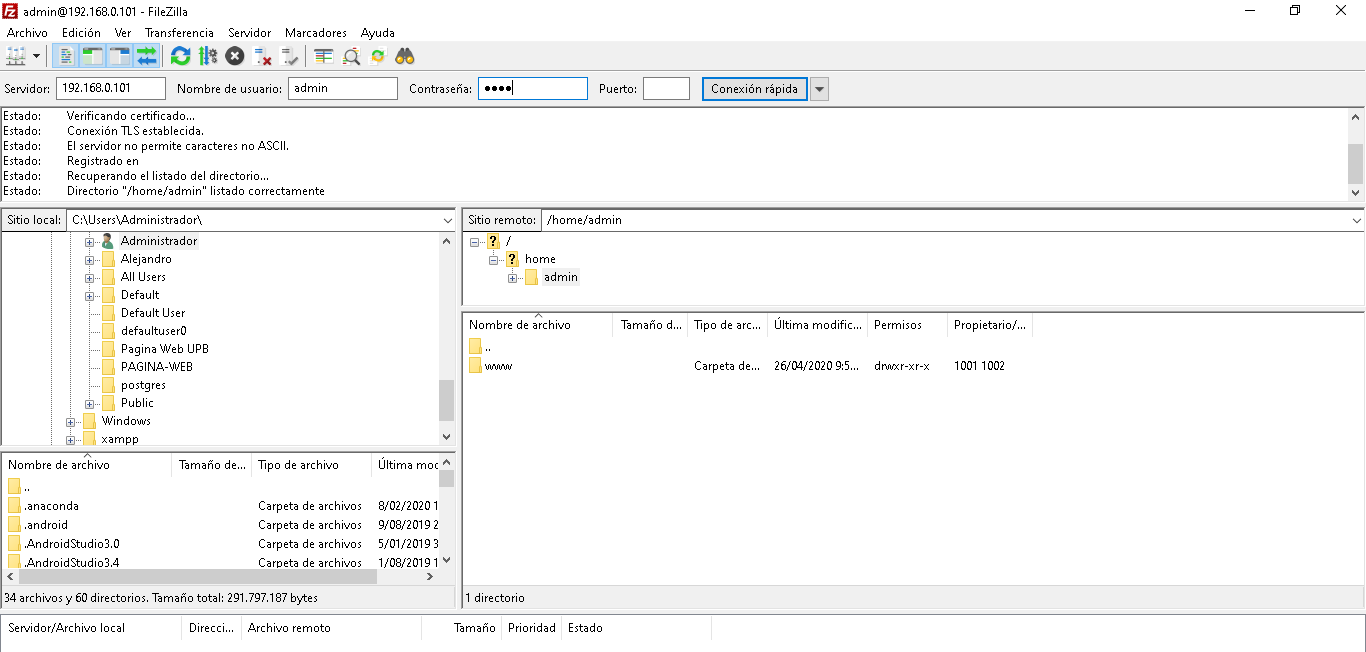
*sudo chown admin:admin /home/admin/www*



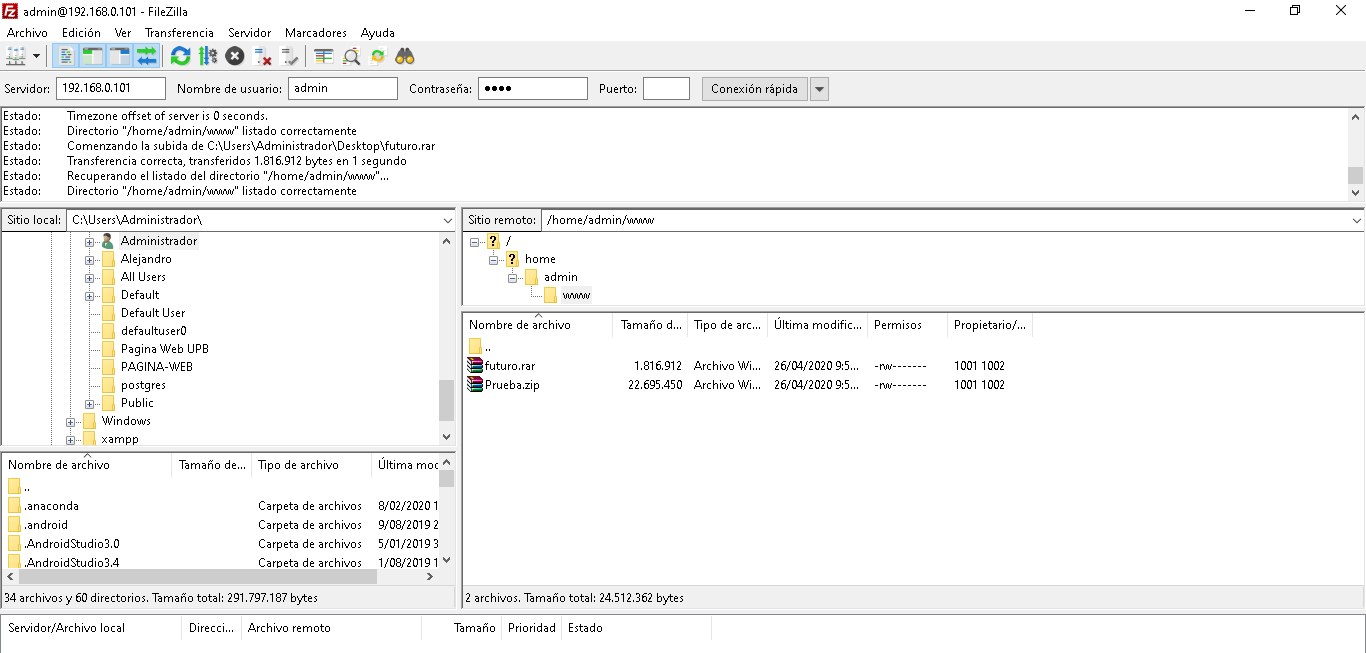
Cuando se vaya a establecer una conexión por medio de un cliente FileZille nos mostrará una ventada que nos dice que el servidor utiliza un certificado autofirmado.

Acá se puede observar la conexión por medio del servidor ftps://192.168.0.101, user = Admin y pass = 1234. Una vez realizada la conexión nos muestra el certificado.





En esta parte se realiza una transferencia desde la computadora al Raspberry por medio de FTPS de manera exitosa.



Como se puede observar en la ruta en la cual se hace la transferencia tenemos los archivos que pasamos por medio del FTPS



1. **CAMBIOS BASE DE DATOS MARIADB**

Se debe adicionar a la base de datos actual la siguiente tabla

USE EduApps;

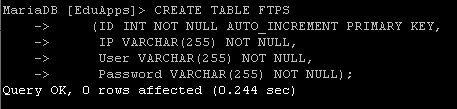
CREATE TABLE FTPS

    (ID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

     IP VARCHAR(255) NOT NULL,

     User VARCHAR(255) NOT NULL,

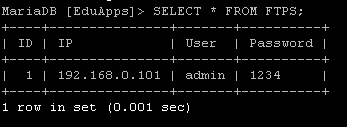
     Password VARCHAR(255) NOT NULL);



Se insertan los siguientes datos en la tabla anteriormente creada:

INSERT INTO FTPS (ID,IP,User,Password) VALUES (NULL, '192.168.0.101', 'admin', '1234');





Además se realiza la inserción de datos a la tabla recursos por los distintos cambias en las rutas de transferencia

INSERT INTO Recursos VALUES (NULL,1, '/home/admin/www/oscar/1/1/archivo.pdf');

INSERT INTO Recursos VALUES (NULL,2, '/home/admin/www/oscar/1/2/archivo.apk');

INSERT INTO Recursos VALUES(NULL,2, '/home/admin/www/oscar/1/2/archivo.png');

