



EL TIEMPO EN CASA

**DESPLIEGUE DE UN SERVIDOR WEB Y UN
SERVIDOR DE BASE DE DATOS PARA
ALMACENAR Y CONSULTAR REGISTROS
METEOROLÓGICOS**

PABLO ANTÓN LAFUENTE

**Grado: Administración de Sistemas
Informáticos en Red**

**Centro de formación: IES Rosa Chacel
Tutora: Marina Villarán Bermejo**

Índice

Índice	1
Introducción	2
Objetivos	2
Requisitos de funcionalidad del proyecto	3
Requisitos para la implementación del proyecto	4
Almacenamiento de los datos	5
Comunicación	6
Servicio web.....	7
Conectividad desde internet.....	8
Servidores y alojamiento.....	9
Sensores	10
Requisitos finales	11
Planificación y secuenciación de actividades	12
Estudio de viabilidad	13
Normas de seguridad y medio ambiente	14
Implementación	15
Despliegue del servidor Proxmox y las dos máquinas virtuales	15
Servicio web Apache	29
Servicio de base de datos MySQL	31
Configuración de la red.....	32
Configuración del DDNS y el router	34
Configuración certificado HTTPS	36
Estructura y comunicación de la base de datos.....	38
Creación de la base de datos y sus usuarios.....	39
Copias de seguridad de la base de datos	40
Conexión a la base de datos con PHP	42
PHP para la recepción de datos e inserción en la BBDD.....	42
Montaje y código de los sensores	43
Estructura del portal web	45
Funciones básicas del portal web.....	46
Funciones avanzadas del portal web.....	48
Conclusión	50
Bibliografía	51

Introducción

El proyecto consistirá esencialmente en una estación meteorológica que recogerá en un panel central en forma de sitio web los datos atmosféricos de diferentes sensores de medición.

Existen muchas estaciones meteorológicas comerciales que cumplen esta función básica, pero como se verá más adelante, tienen unas características muy limitadas.

La estación meteorológica que se va a desarrollar está pensada principalmente para su uso en un entorno doméstico, aunque también ofrece la posibilidad de realizar mediciones desde cualquier otro lugar siempre que se disponga de conexión a internet.

De la misma forma se podrán consultar los datos desde cualquier lugar a través del sitio web y ofrecerá una información mucho más detallada.

Además, tendrá un coste de despliegue inicial bajo y será posible añadir más sensores de igual forma con un coste muy bajo cada uno.

Objetivos

Una vez visto el concepto básico, los objetivos del proyecto son principalmente los siguientes:

-El portal debe cumplir los siguientes requisitos:

- Deben poderse comprobar los datos (temperatura y humedad) en tiempo real de cualquiera de los sensores desde el portal web desde cualquier lugar de internet.

- Deben poderse comprobar otros datos más avanzados que no ofrecen las estaciones comerciales como poder ver el historial de todos los datos de cualquier sensor o las máximas, medias y mínimas en un periodo de tiempo de un sensor o grupo de sensores.

- Deben poderse gestionar los datos de cada sensor y los propios sensores desde el portal web.

-Los sensores deben cumplir las siguientes funciones:

- Conectarse a través de WiFi desde cualquier red de forma segura.

- Deben tener un despliegue sencillo, asistido por la propia web de la estación y el hardware debe de poderse conseguir fácilmente y tener un coste bajo.

- Una vez desplegado el sensor se podrá cambiar su ubicación desde la web sin tener que realizar ningún cambio en el propio sensor.

- El número de sensores debe ser altamente escalable

Requisitos de funcionalidad del proyecto

Una vez definidos los objetivos fundamentales se van a marcar los requisitos que el proyecto deberá cumplir. Estos son los requisitos a nivel de funcionalidad, es decir concretar lo que la estación meteorológica debe ser capaz de hacer.

Mas adelante se estudiará que hardware y software es necesario para hacerlo posible.

En cuanto a los sensores:

- Deben de estar compuestos por partes fáciles de conseguir y que tengan un coste bajo.
- Deben tener conectividad WiFi, leer los datos meteorológicos con precisión y enviarlos regularmente al servidor desde cualquier red de forma segura para ser almacenados.
- Deben de ser sencillos de crear y programar, obteniendo el código ya configurado desde el portal web al añadirlos, así como instrucciones para su montaje.
- Una vez añadidos se podrá cambiar su ubicación desde la web sin necesidad de modificar nada en el propio sensor.

En cuanto al portal web:

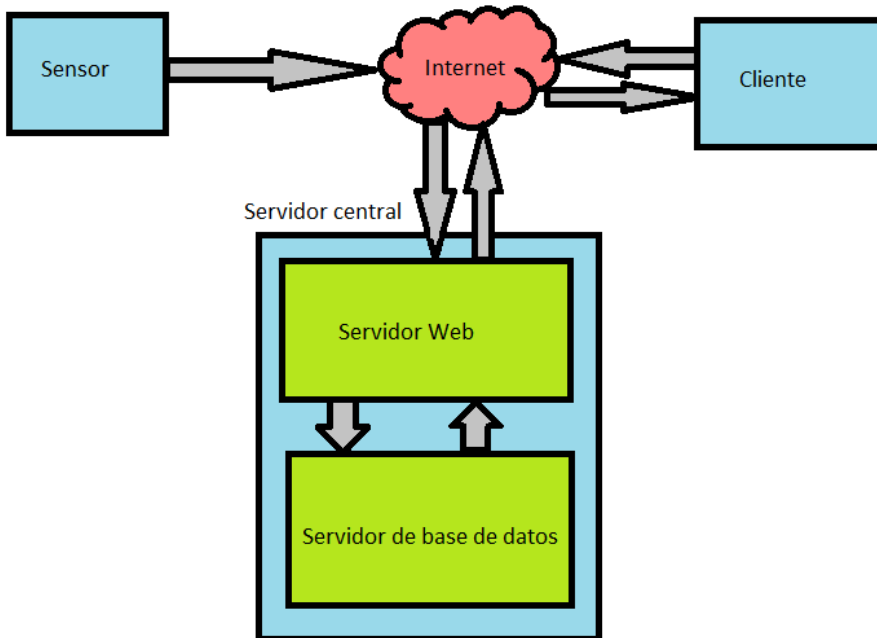
- Debe ser accesible desde cualquier red mediante una conexión segura y en él se podrán ver los datos de todos los sensores independientemente o en grupos clasificados por ubicación. Se podrán ver la temperatura y humedad media, máxima y mínima en ciertos periodos de tiempo.
- Se podrán filtrar y gestionar estos datos, para por ejemplo borrar todos los que ya no sean relevantes.
- Se podrán añadir nuevos sensores, como se ha visto en el apartado de sensores, el portal web proporcionara el código para estos automáticamente según la configuración que elijamos al añadirlos y también proporcionara información sobre cómo crear los sensores.
- Se podrán modificar los sensores existentes, que tendrán un identificador único pero su nombre y ubicación será variable.

Requisitos para la implementación del proyecto

Conociendo los objetivos de funcionalidad del proyecto se van a plantear los requisitos técnicos, tanto de hardware como de software para poder cumplirlos, viendo las diversas opciones disponibles y eligiendo las mejores.

Se comenzará planteando la estructura general de la infraestructura del proyecto para poder investigar las necesidades de cada uno de sus componentes.

La estructura básica del servicio es la siguiente:



Los datos serán recogidos por el sensor y enviados al servidor central, aquí existirán dos servidores, uno que almacenara los datos y otro que después los mostrara al usuario. Mas adelante se verá porque se ha usado esta configuración con dos servidores independientes.

Almacenamiento de los datos

Para almacenar todos los datos necesarios se usará un sistema de base de datos, en este caso será de tipo SQL, ya que es relativamente sencillo y adecuado para este caso.

Las principales opciones disponibles en el mercado en cuanto a bases de datos SQL son las de Oracle, Microsoft y MySQL.

Se elige la base datos MySQL principalmente porque es de código abierto, por lo que no requiere ningún tipo de licencia ni tiene ningún coste, ayudando a mantener el coste del proyecto bajo y también porque es sencilla de instalar y configurar para usos a pequeña escala.

Se crearán copias de seguridad diariamente de forma automática y se almacenarán en un medio independiente y redundante para asegurar los datos.

Como se ha indicado antes, esta base de datos se alojará en un servidor independiente del que será el servidor web.

El motivo principal es mejorar la seguridad, ya que uno de los requisitos es que tanto los sensores como clientes se puedan conectar desde cualquier lugar de internet, por lo que el servidor web tendrá que estar expuesto al exterior y es mejor mantenerlo aislado de la base de datos. Además, esto facilitará su configuración y mantenimiento y se podrán realizar las copias de seguridad necesarias de forma más sencilla.

Comunicación

Ahora vamos a ver como los datos llegaran del sensor al servidor de base de datos, existen dos opciones, que los sensores se conecten directamente a la base de datos para que envíen la información o que se conecten a un portal web, que recogerá la información para luego enviarla a la base de datos.

Se ha elegido esta última opción por dos motivos principales que responden a los requisitos del proyecto:

- La seguridad, uno de los requisitos es que tanto los sensores como clientes se puedan conectar desde cualquier lugar a través de internet, más tarde se estudiara como esto va a ser posible, pero de esta forma solo el portal web tendrá que estar expuesto a internet, la base de datos se podrá mantener aislada sin salida al exterior ya que se comunicara mediante una red interna únicamente con el servidor web. Esto significa menos puertos y servicios expuestos al exterior y por lo tanto menor riesgo de ataques.

- Que se puedan procesar los datos antes de introducirlos a la base de datos, esto es importante ya que así podremos cumplir otro de los objetivos del proyecto que es que los sensores sean seguros, para esto el sensor enviará un código de seguridad cada vez que envíe datos, este será comprobado por el servidor y solo si el código es correcto y por lo tanto el sensor está autorizado enviara los datos a la base de datos.

Por lo tanto, los sensores se conectarán a un portal web de forma segura y enviarán la información, en el apartado del servicio web se verán más detalles.

Servicio web

Existen varias opciones para alojar los portales web que serán necesarios, las principales disponibles en el mercado son Apache, NGINX y Microsoft ISS.

En este caso se elige Apache, ya que es de código abierto, por lo que no requiere ningún tipo de licencia ni tiene ningún coste, ayudando a mantener el coste del proyecto bajo y porque es sencillo de configurar y desplegar.

Todos los portales web usaran el protocolo HTTPs para que la conexión sea segura.

En el servidor se usará PHP para procesar todos los datos y hacer las operaciones necesarias, ya que es un lenguaje relativamente sencillo que funciona muy bien con bases de datos.

Los sensores se conectarán a un portal web y enviarán un formulario POST que el servidor podrá procesar con PHP para verificar e introducir los datos a la base de datos.

Para mostrar los datos al usuario final, se accederá a otro portal web que se conectará a la base de datos, donde obtendrá los datos para calcular toda la información necesaria mediante PHP, como las máximas y mínimas o la media y las presentará de forma útil y legible.

Conectividad desde internet

Ahora vamos a ver otro de los requisitos, que es que tanto los sensores que envían la información como el cliente que la consulta puedan hacerlo desde internet.

Como se ha visto antes, el servicio se ha diseñado para que esto sea posible solo con que el servidor web esté disponible al exterior.

Para que un equipo esté disponible al exterior es necesario primero abrir los puertos necesarios en el router y después conocer la dirección IP pública. En el proyecto se va a usar una red doméstica, como la que cualquier persona puede tener en casa, por lo que es muy probable que la IP cambie cada cierto tiempo, además es mucho más cómodo acceder a una dirección URL que a una dirección IP, por lo que se va a estudiar cómo resolver estos problemas.

Normalmente habría que comprar un dominio y registrarlo en un servicio DNS para que apuntara a nuestra IP, pero esto implica un coste, y como se ha visto la dirección IP va a cambiar.

Pero existe otra opción, una tecnología conocida como DDNS, o Dynamic DNS, que consiste en que un proveedor concede una URL que se podrá actualizar siempre que sea necesario (se hace normalmente de forma automática) para que apunte a una IP. Estos servicios también implican un coste en la mayoría de casos, pero algunos proveedores ofrecen de forma gratuita un número reducido de URLs.

En este caso solo hará falta una, por lo que es posible tener una URL con la que enviar datos desde los sensores y consultar la información desde un cliente desde cualquier lugar de internet de forma gratuita.

El proveedor elegido que ofrece unas pocas direcciones dinámicas de forma gratuita es DuckDNS. Se instalará un servicio en el servidor web que actualizará la URL, por ejemplo `eltiempo.duckdns.org`, para que siempre este apuntando a la IP del servidor.

Servidores y alojamiento

Se debe tener en cuenta que esta es la parte más flexible del proyecto.

Aunque se ha diseñado para que el servicio use dos servidores independientes, uno para el servicio web y otro para la base de datos, y como se verá a continuación se va a usar un sistema bastante complejo para el alojamiento y las copias de seguridad, esto sería la configuración optima, pero no es realmente necesaria.

Disponiendo de la página web y la estructura de la base de datos el proyecto se podría alojar en un solo servidor, y prácticamente cualquier sistema, serviría con simplemente ejecutar XAMPP en un equipo domestico con Windows o dedicar un ordenador de bajos recursos.

Dado este apunte, a continuación, se vera la implantación optima del servicio.

Una vez conocido el software que se va a emplear tanto para el servidor web como para el servidor de base de datos, es necesaria una estructura y sistema base en el que estos servicios puedan funcionar.

Como base para ejecutar los dos servidores necesarios se ha va a usar un hipervisor, que en este caso será de tipo 1, es decir, está instalado en un ordenador dedicado y los servidores corren sobre este sistema con un rendimiento casi nativo.

Existen varias opciones en cuanto a este tipo de hipervisores, algunas de las más conocidas a nivel comercial son Microsoft Hyper-V y VMware EXCi, el problema es que este software es de uso profesional y las licencias tienen un coste muy alto, sin embargo, existe otra opción, el hipervisor Proxmox. Es también una solución comercial, pero se basa en software libre, y solo es necesario una licencia si queremos tener acceso a soporte directo, por lo que esta es claramente la mejor opción.

Una vez decidida la base para el alojamiento, es necesario elegir los sistemas que se virtualizaran para alojar los servicios en sí.

Las dos opciones principales en cuanto a sistemas operativos son Windows o Linux, en este caso se utilizarán sistemas Linux como base, debido a los siguientes factores:

- Linux es un sistema de código abierto, por lo que no supondrá ningún coste adicional.
- Tiene un consumo de recursos muy bajo comparado con un sistema Windows, lo que también ayudara a reducir el coste de alojamiento o del hardware.
- Tiene una mejor compatibilidad con los servicios web y de base de datos elegidos, y su despliegue es más sencillo.

Una vez elegido Linux, existen multitud de opciones en cuanto a distribuciones, aunque en este caso se optara por Ubuntu Server.

Ubuntu es una de las distribuciones de Linux más usada a nivel doméstico debido a que es sencillo y eficiente de usar y al estar extendida tiene un soporte y documentación muy amplios.

La versión Server elimina la interfaz gráfica, que no será necesaria para las configuraciones que se van a realizar. Esto reduce notablemente el consumo de recursos y por lo tanto ayudara también reducir el coste de alojamiento o del hardware.

Sensores

Una vez conocido el protocolo por el cual los datos se van a enviar al servidor, se va a estudiar cual sería el tipo de hardware más óptimo para los sensores para cumplir con los requisitos de bajo coste, despliegue sencillo, etc.

Como se ha visto, el sensor deberá leer los datos de temperatura y humedad y conectarse a un portal web para enviar los datos mediante un formulario “POST”.

Por lo tanto, es necesario un dispositivo con conexión wifi, capaz de acceder a la web y realizar envíos de formularios y que pueda leer la temperatura y humedad externa.

Una opción sería usar algún tipo de ordenador con un sensor externo, pero esto es bastante complejo, y además no es viable ya que el coste por sensor sería muy alto, así como el consumo de energía, y realmente no se va a aprovechar a penas el potencial de un ordenador común realizando una tarea tan sencilla como la del sensor.

La mejor opción es un microcontrolador. Es en esencia un ordenador, ya que dispone de almacenamiento, RAM, etc. pero tiene unas prestaciones muy reducidas y solo ejecutara el código con el que sea programado. Uno de los más conocidos es Arduino, con un microcontrolador como este se eliminan todos los problemas anteriores ya que tienen un coste y un consumo muy bajos, está diseñado para comunicarse con dispositivos como sensores y una vez programado con el código correspondiente funcionara si ningún problema.

Queda el problema de que conectar a internet una placa Arduino mediante wifi para que acceda a un portal web no es sencillo, sin embargo, existe una placa similar pero que esta más orientada a su uso con internet, el microcontrolador ESP32.

Funciona igual que Arduino, pero tiene integrada conexión wifi y dispone de librerías especiales para hacer conexiones web y enviar o recibir datos, por lo que es el dispositivo perfecto para el proyecto.

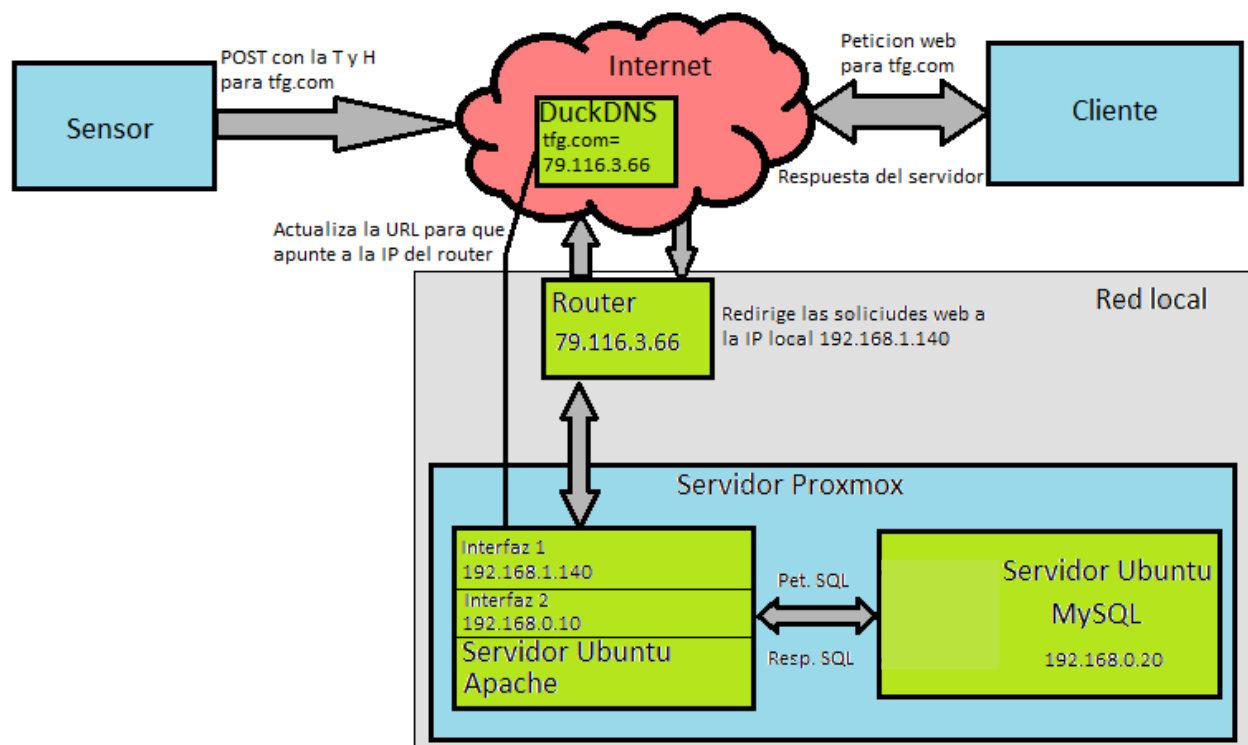
En cuanto al sensor de temperatura y humedad, hay varias opciones. Los más comunes para este tipo de tarea (lectura de temperatura y humedad) son el DHT11 y el AHT21. Después de realizar pruebas con ambos, se ha comprobado que el AHT21 ofrece una precisión mayor y un mayor rango de medición, además la conexión es más fiable ya que usa el protocolo I2C, por lo que finalmente se usara este.

Requisitos finales

Después de ver todos los requisitos de cada componente y su comunicación, los requisitos finales que es necesario implementar en cuanto a software y hardware serian:

- Software:
 - Servidor Proxmox VE
 - Dos sistemas Ubuntu Server
 - Servicio Web apache HTTPs con sistema de actualización de DDNS
 - Servicio de base de datos MySQL con sistema de copias de seguridad
 - Web para recibir los datos de los sensores y añadirlos a la base de datos
 - Código Arduino para acceder a la web y hacer POST de los datos
 - Web para mostrar al cliente los datos de la base de datos
- Hardware:
 - Placa ESP32
 - Sensor AHT21
 - Alojamiento base

El esquema final detallado del servicio sería el siguiente



Planificación y secuenciación de actividades

Las diferentes etapas que se van llevan a cabo para la realización del proyecto son las siguientes:

- Investigación y planificación para conocer todo lo que será necesario crear y desplegar-1h
- Instalación y configura el servidor Proxmox -1h
- Instalación y configuración básica de los sistemas Ubuntu Server-1h
- Instalación y configuración del servicio web-30 min
- Instalación y configuración del servicio de base de datos-15min
- Configuración del entorno de red definitivo de los servidores-30min
- Configuración del DDNS, el servicio de actualización y los puertos del router-30min
- Configuración del certificado y acceso HTTPs-15 min
- Planteamiento de la estructura general de la base de datos y su comunicacion-15min
- Creación la base datos SQL en el servidor, y de los usuarios necesarios-15min
- Configuración del sistema de copias de seguridad de la base de datos-15min
- Creación del código que conecta la base de datos con PHP-15min
- Creación del código PHP para recibir e insertar los datos de los sensores-30min
- Montaje y se desarrolló del código para los sensores-1h
- Planteamiento de la estructura del portal web para los clientes-30min
- Desarrollo de la estructura del sitio y las funciones básicas de visualización de datos-3h
- Desarrollo de las funciones más avanzadas, como filtros o añadir y editar sensores-5h
- Optimización del codigo-1h
- Pruebas de todas las funciones para descubrir y arreglar cualquier error, conclusion-1h

En total se calcula que la implementación y desarrollo tomaría unas 18 horas de trabajo, aunque se consideraran 20 horas en caso de cualquier retraso o problema.

Estudio de viabilidad

A continuación, se van a estudiar los costes aproximados del proyecto para comprobar su viabilidad económica.

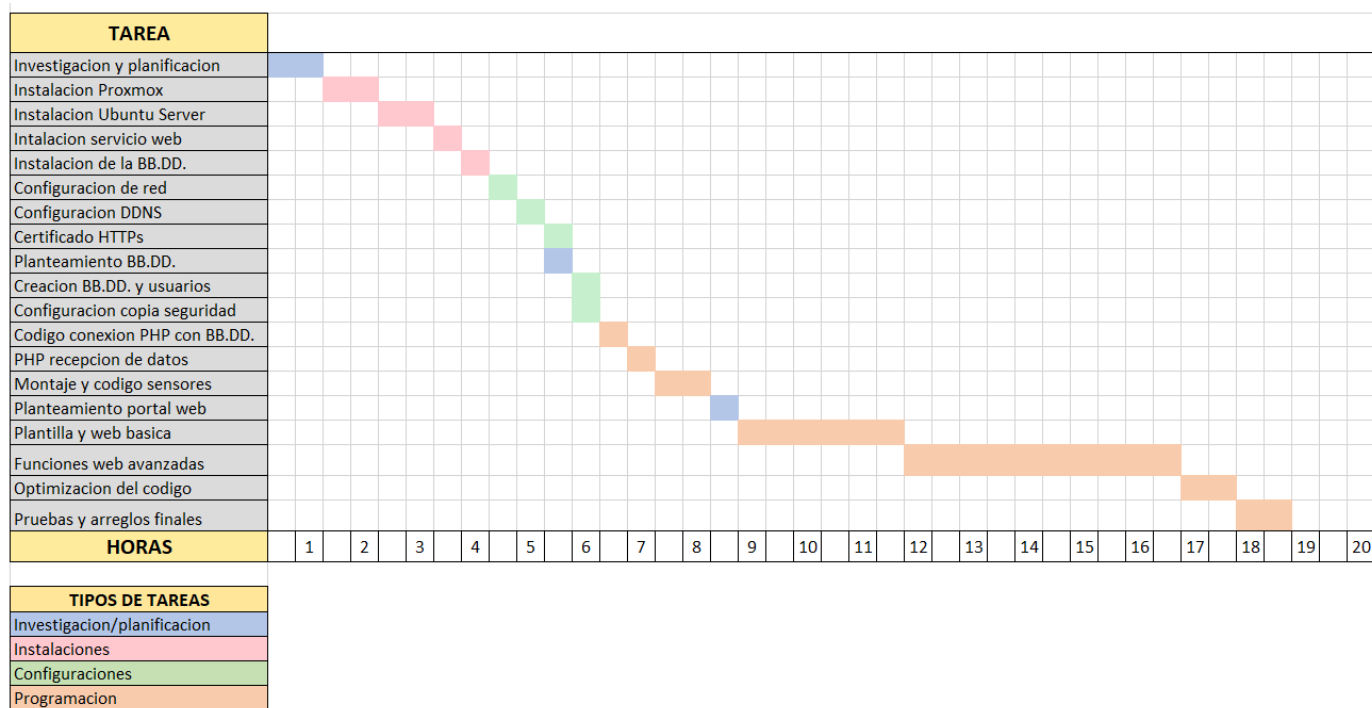
Para empezar, los costes de desarrollo (que tomaría unas 20 horas según las tareas vistas anteriormente) con un salario promedio de un programador web y de base de datos de 20€/h ascenderían a unos 400€. Cabe destacar que estos costes solo serian para el desarrollo inicial del proyecto.

Después se deben tener en cuenta los costes de alojamiento, para alojar el servicio a nivel doméstico bastaría con un equipo de unos 2GB de RAM y 1 núcleo o una RaspberryPi, por ejemplo, que conlleva un coste de unos 80€ actualmente, y un coste de electricidad de no más de 20€ al año.

Para alojar el proyecto en condiciones idóneas sería necesario un equipo algo más potente (4GB de RAM/2 núcleos). Un equipo de estas características puede tener un coste de unos 100€, y un consumo anual de unos 60€. Cabe destacar que una vez se dispone de cualquiera de estos equipos, este puede cumplir muchas más funciones que este servicio, como almacenamiento, distribución de multimedia etc. que lo harían mucho más rentable.

Por último, el coste de los sensores. A día de hoy el sensor AHT21 tiene un coste de 1.2€ y la placa ESP32 tiene un coste de 3.5€. Esto sitúa el coste total de cada sensor en menos de 5€.

Por lo tanto, los costes iniciales serian de unos 400€ para el desarrollo, desde 0 a unos 100€ para el alojamiento dependiendo de que plataforma se elija y de unos 20 a 60€ al año de consumo eléctrico.



Normas de seguridad y medio ambiente

En cuanto a la seguridad en el proyecto, se han implantado diferentes medidas, ya que al disponer de un servidor expuesto a internet existen ciertos riesgos. Entre ellas:

- El router solo tiene abierto el puerto necesario para el servicio web, y dirige todos los paquetes de este puerto al servidor web.
- Todos los portales web usan el protocolo HTTPs, por lo que toda la comunicación esta encriptada y es segura.
- El servidor de base de datos está aislado del exterior y solo puede comunicarse con el servidor web a través de una red interna.
- Solo el sensor designado puede enviar datos al servidor ya que su identidad se verifica mediante un código hashado, evitando posibles ataques de spam o DDoS.
- Se realizan copias de seguridad periódicas de la base de datos para evitar cualquier posible pérdida de datos.

En cuanto a las consideraciones medioambientales, se recomienda usar antiguo hardware para la implementación, dándole una nueva vida y evitando que se desheche, y como precauciones se deben proteger los sensores instalados en el exterior ya que podrían estropearse o fallar al exponerse a los elementos climatológicos y generar residuos innecesarios o contaminación.

Si llegan a estropearse en algún caso es importante desecharlos de forma adecuada ya que tienen componentes electrónicos y otros tipos de materiales que pueden ser perjudiciales para el medio.

También se han elegido y planeado los sensores y la infraestructura para tener el mínimo consumo de energía posible.

Implementación

A continuación, se detalla el despliegue y desarrollo de todos los componentes del proyecto, se configurarán todos los servidores y servicios necesarios, se creará la base de datos, el código Arduino y el código PHP.

Despliegue del servidor Proxmox y las dos máquinas virtuales

Para empezar, se va a desplegar y configurar la base de todo el servicio, es decir, el hipervisor Proxmox VE con las dos máquinas virtuales Ubuntu server.

Los sistemas hipervisores tipo 1 están diseñados para correr en hardware dedicado, pero en este caso se virtualizará el propio hipervisor por comodidad y con propósitos demostrativos.

Se usa la función de virtualización anidada, que esencialmente permite correr máquinas virtuales dentro de otra máquina virtual. Se va a usar una máquina virtual con 8GB de RAM, una CPU de cuatro núcleos y 100GB de almacenamiento principal además de otros dos discos de 5GB cada uno para almacenar las copias de seguridad, aunque se han realizado pruebas y se ha comprobado que después de la instalación el servicio puede funcionar con unos recursos muy reducidos, 4GB de RAM y 2 núcleos de CPU serían suficientes. La instalación sería igual en hardware real con características similares.

Para comenzar la instalación del servidor Proxmox VE lo primero es obtener la imagen ISO del sistema desde la web oficial.

En un entorno real se instalaría esta imagen ISO en una unidad USB, convirtiéndola en un medio desde el que la máquina pueda iniciarse y se configuraría la máquina desde la BIOS para que arrancara desde este dispositivo, en la máquina virtual se añade la imagen ISO y la máquina iniciara desde esta.

Aparece la pantalla inicial del instalador



Se debe aceptar la licencia de uso y elegir en que disco se va a realizar la instalación, se elige el disco de 100GB.



Proxmox Virtual Environment (PVE)

The Proxmox Installer automatically partitions your hard disk. It installs all required packages and makes the system bootable from the hard disk. All existing partitions and data will be lost.

Press the Next button to continue the installation.

- **Please verify the installation target**
The displayed hard disk will be used for the installation.
Warning: All existing partitions and data will be lost.
- **Automatic hardware detection**
The installer automatically configures your hardware.
- **Graphical user interface**
Final configuration will be done on the graphical user interface, via a web browser.

Target Harddisk:

Después habrá que configurar otros parámetros como la zona horaria, el teclado o la contraseña para “root”. También habrá que configurar la red, es decir la dirección IP del servidor y la URL.



Management Network Configuration

Please verify the displayed network configuration. You will need a valid network configuration to access the management interface after installing.

After you have finished, press the Next button. You will be shown a list of the options that you chose during the previous steps.

- **IP address (CIDR):** Set the main IP address and netmask for your server in CIDR notation.
- **Gateway:** IP address of your gateway or firewall.
- **DNS Server:** IP address of your DNS server.

Management Interface:

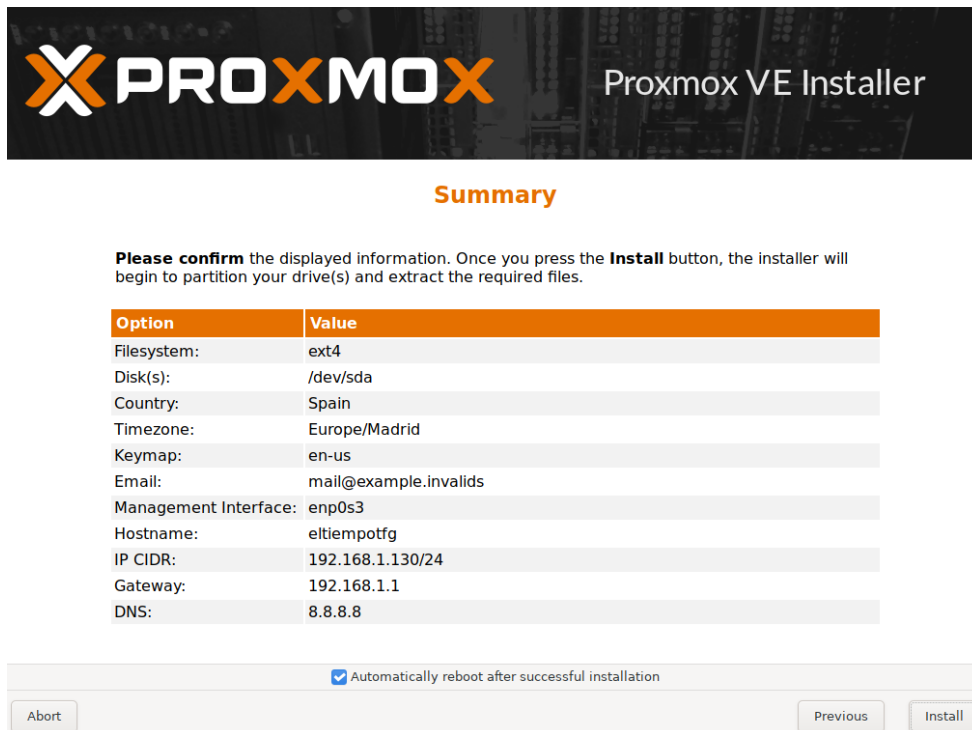
Hostname (FQDN):

IP Address (CIDR): /

Gateway:

DNS Server:

Se revisa toda la configuración y se proceder a la instalación.



The image shows the Proxmox VE Installer Summary screen. At the top, there is a header with the Proxmox logo and the text "Proxmox VE Installer". Below this, the word "Summary" is centered. A paragraph of text states: "Please confirm the displayed information. Once you press the **Install** button, the installer will begin to partition your drive(s) and extract the required files." Below this text is a table with two columns: "Option" and "Value". The table contains the following data:

Option	Value
Filesystem:	ext4
Disk(s):	/dev/sda
Country:	Spain
Timezone:	Europe/Madrid
Keymap:	en-us
Email:	mail@example.invalids
Management Interface:	enp0s3
Hostname:	eltiempotfg
IP CIDR:	192.168.1.130/24
Gateway:	192.168.1.1
DNS:	8.8.8.8

Below the table, there is a checkbox labeled "Automatically reboot after successful installation" which is checked. At the bottom, there are three buttons: "Abort", "Previous", and "Install".

Al finalizar la instalación se retira el medio de instalación, tras hacerlo el sistema se reiniciará y el servidor Proxmox ya estar instalado y funcionando.

```
-----
Welcome to the Proxmox Virtual Environment. Please use your web browser to
configure this server - connect to:

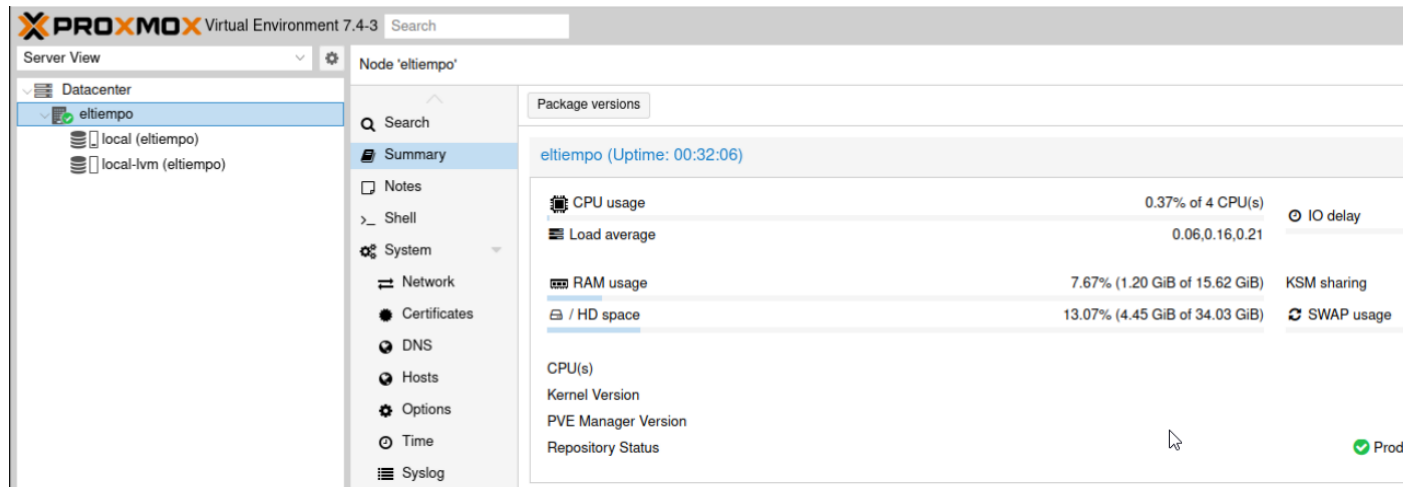
https://192.168.1.130:8006/

-----
eltiempotfg login: _
```

Una vez completada la instalación del servidor, se puede acceder a su interfaz desde cualquier navegador web, accediendo a la dirección del servidor y un puerto específico que se indican al finalizar la instalación, en este caso la dirección es 192.168.1.130:8006.

Se inicia sesión con la contraseña que creada durante la instalación y el usuario root.

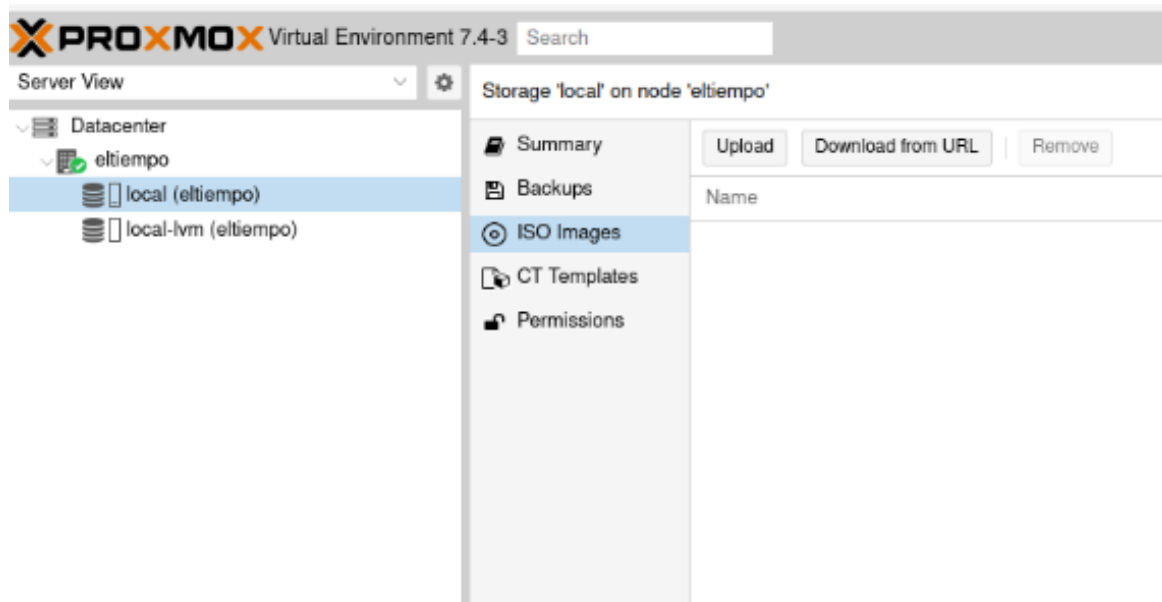
Una vez iniciada la sesión, se puede ver la interfaz gráfica del servicio.



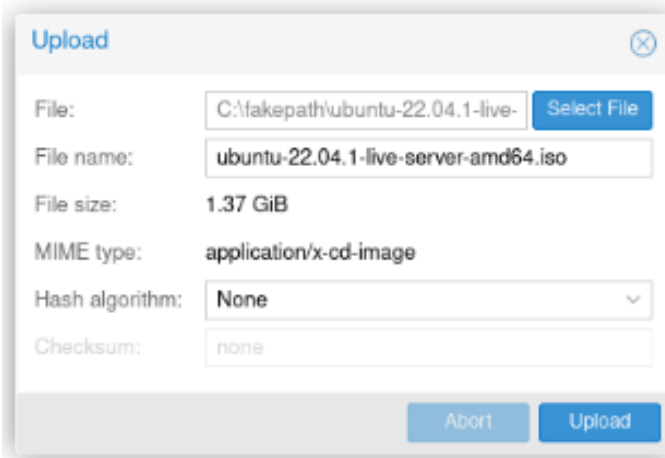
Ahora, lo primero que se hace es añadir la imagen ISO necesaria para los dos sistemas Ubuntu Server. Esta imagen se obtiene desde la web oficial de Ubuntu:

<https://ubuntu.com/download/server>

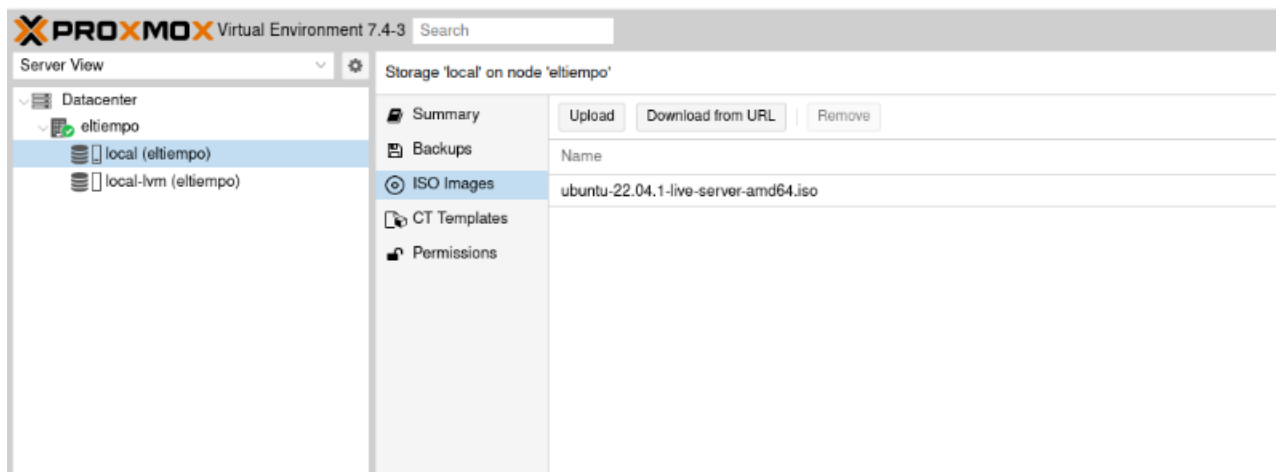
Una vez descargada la imagen, se accede a la sección de almacenamiento local y al apartado de “ISO Images”. Aquí con la opción “Upload” se pueden añadir nuevas imágenes ISO.



Se abrirá un cuadro de dialogo en el que se sube el archivo que se acaba de descargar y se confirmar su subida.



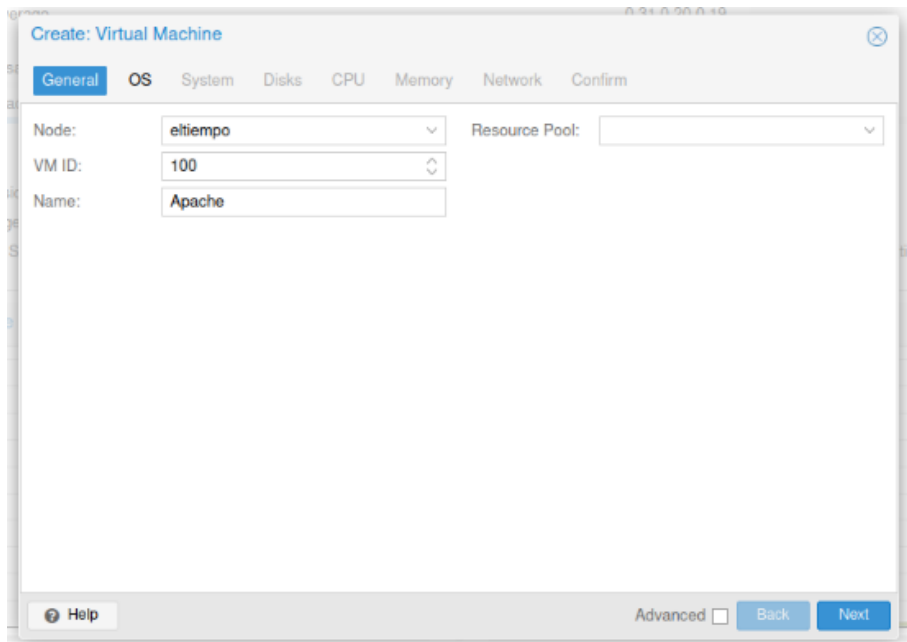
Después el archivo se procesará y almacenará en el servidor y la imagen ya estará disponible.



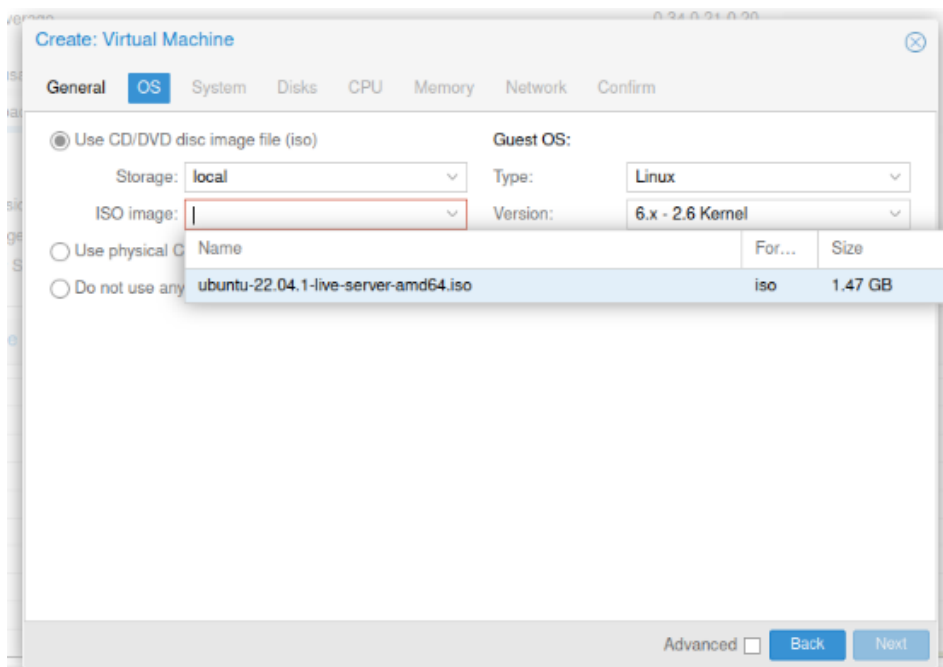
Ahora desde el servidor principal ya se puede comenzar con la creación de las máquinas virtuales, después de pulsar el botón “Create VM” se abrirá un dialogo en el que se irán configurando todos los aspectos de la máquina virtual.

El Tiempo En Casa - ASIR

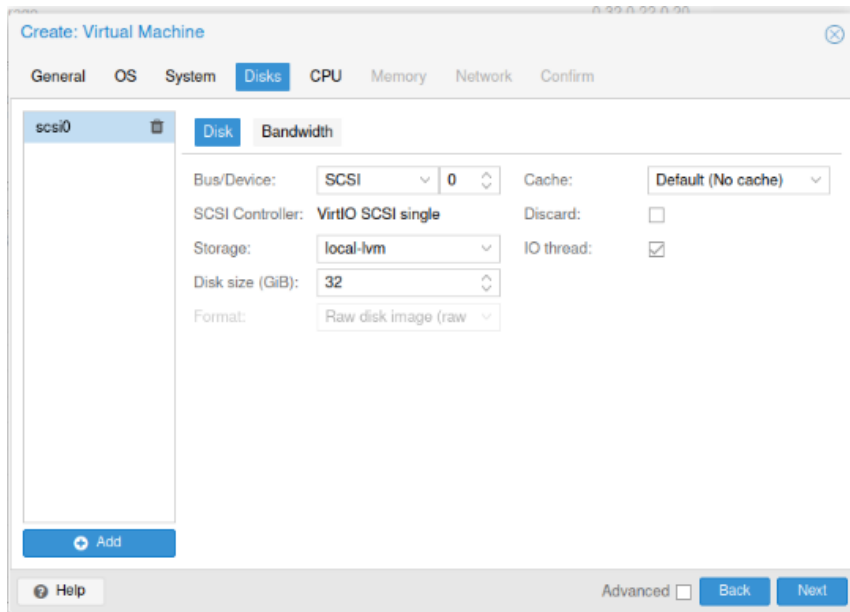
Primero se introduce el nombre y se elige en que servidor crearla



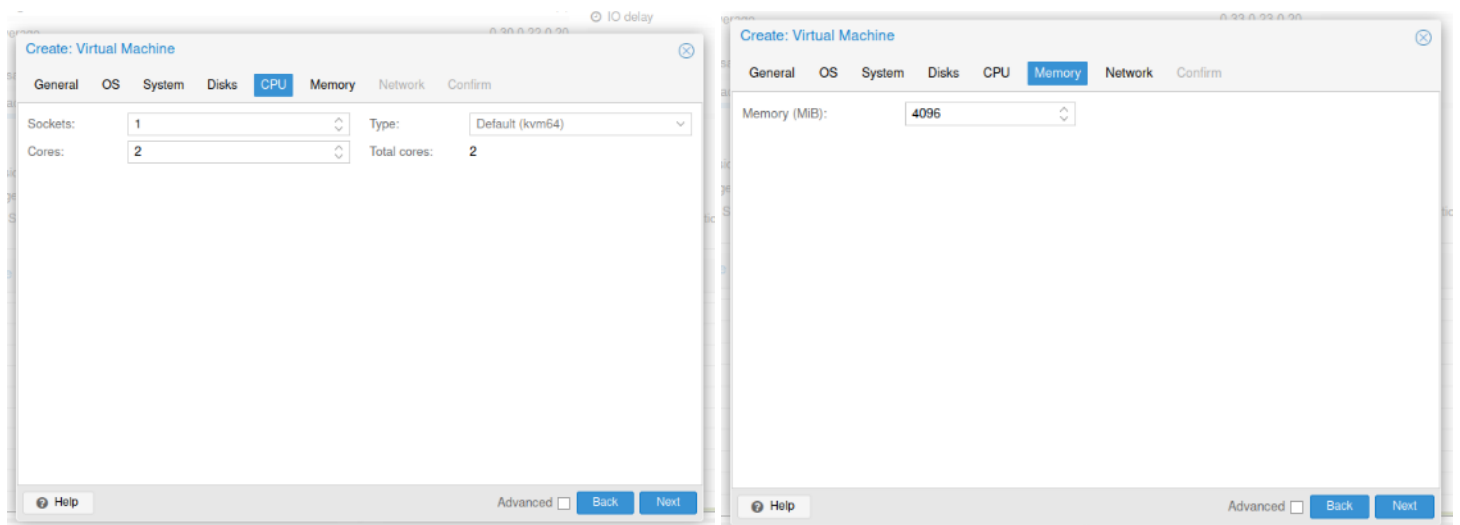
Luego hay que configurar la imagen de inicio, en este caso se elegirá la imagen Ubuntu Server añadida anteriormente, también se configura el tipo de sistema, aunque se suele configurar automáticamente dependiendo de la imagen ISO elegida.



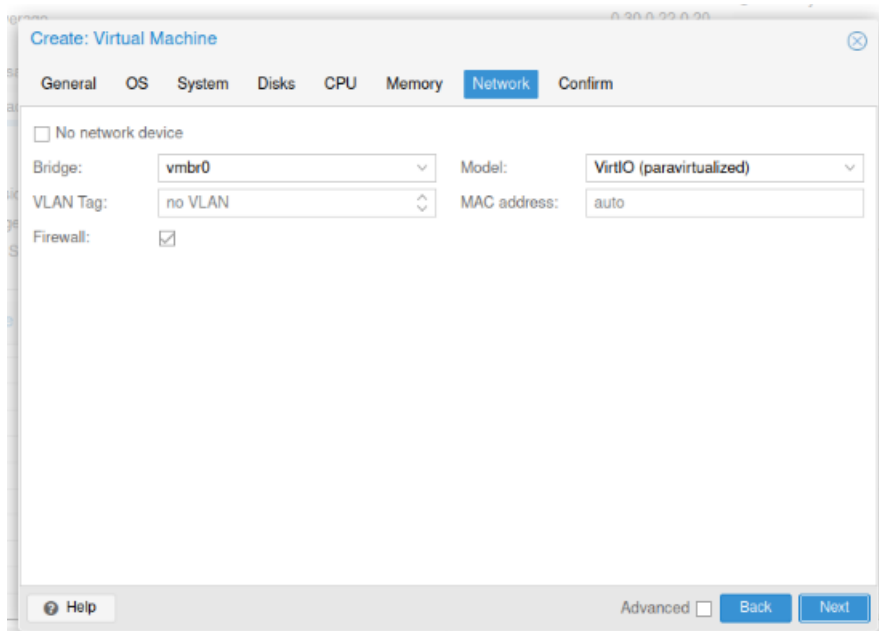
Se configura el disco virtual, en este caso se usará el valor por defecto y se creará un disco virtual de 32 GB, que es más que suficiente para el sistema y los servicios necesarios.



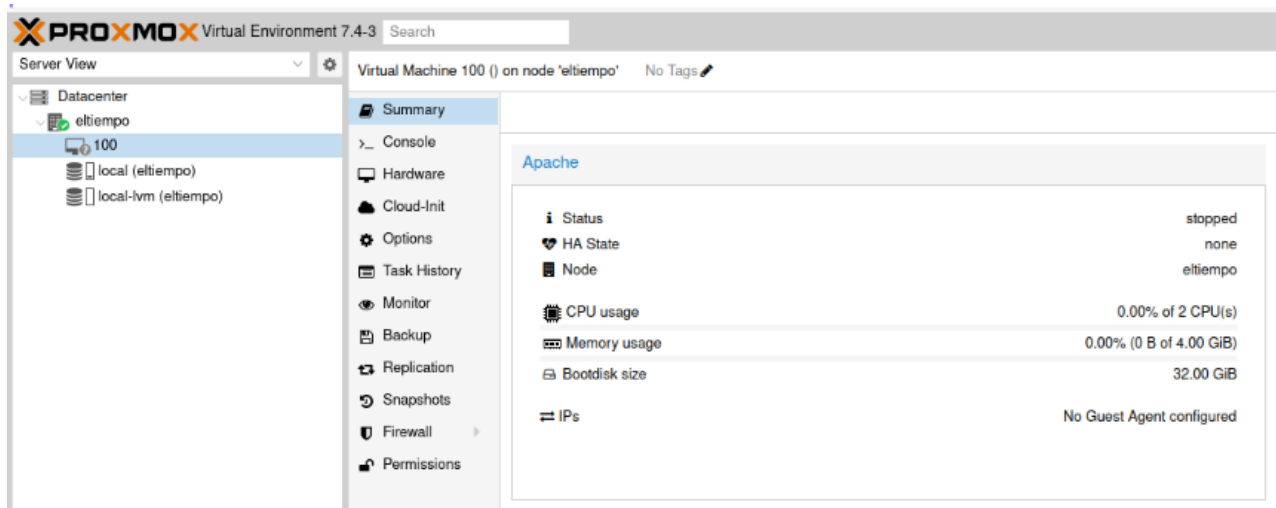
Después se configura la CPU y la memoria RAM. En este caso se asignarán 2 núcleos y 4GB de RAM a cada máquina.



Se configura la red. Aquí se usa la configuración básica por defecto para que las maquinas tengan conexión por el momento para realizar la configuración e instalación, pero más adelante se configurara la red en su forma final.



Por último, se revisan y confirman todos los parámetros, ya se puede ver que la máquina virtual se ha creado correctamente.



Ahora se inicia la máquina, y desde la pestaña “Console” se puede ver e interactuar con ella. El instalador de Ubuntu Server se iniciará, habrá que configurar primero algunos parámetros básicos como el idioma o la distribución de teclado.

Una vez realizadas estas configuraciones básicas se debe configurar la red, en este caso se usará una IP fija 192.168.1.140.

Network connections [Help]

Configure at least one interface this server can use to talk to other machines, and which preferably provides sufficient access for updates.

NAME	TYPE	NOTES
ens18	eth	-
DHCPv4 192.168.1.134/24		
ba:1c:24:89:73:af / Red Hat, Inc. / Virtio network device		

[Create bond >]

Edit ens18 IPv4 configuration

IPv4 Method: [Manual]

Subnet: 192.168.1.0/24

Address: 192.168.1.140

Gateway: 192.168.1.1

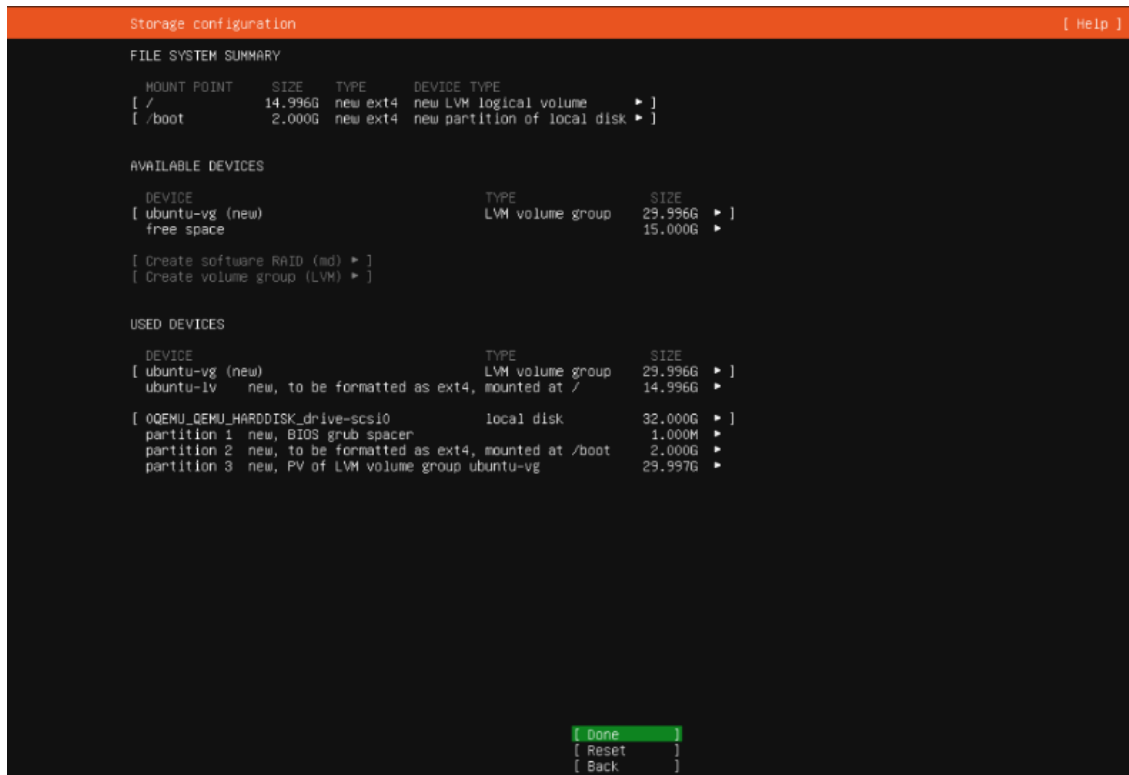
Name servers: 8.8.8.8
IP addresses, comma separated

Search domains:
Domains, comma separated

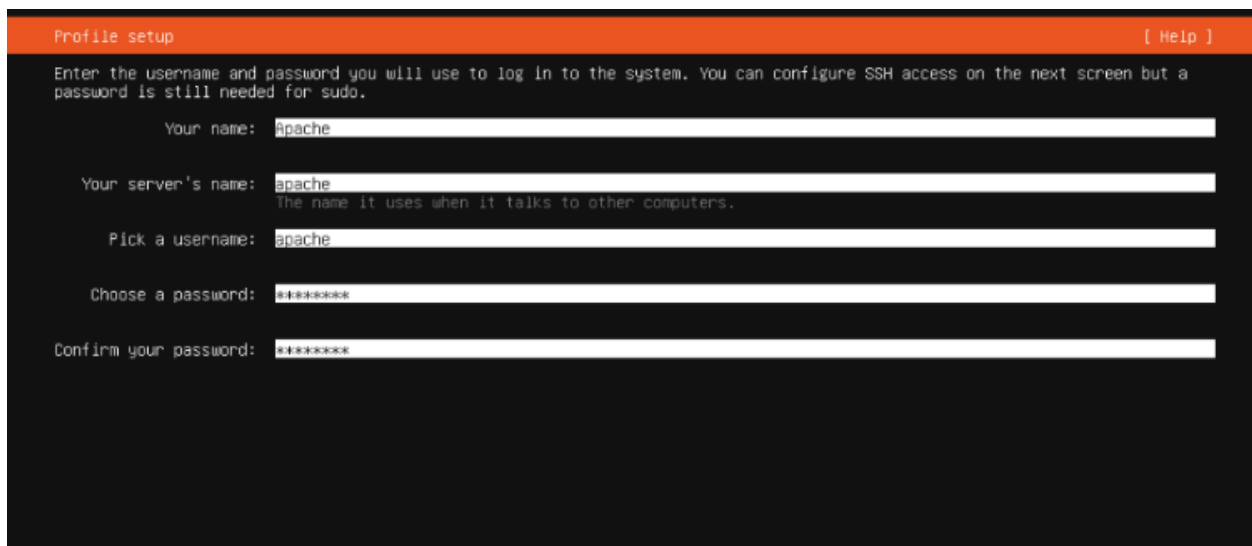
[Save]
[Cancel]

[Done]
[Back]

Se deben configurar las opciones de almacenamiento, en este caso se usará todo el disco virtual para la instalación.



Por último, se elige el nombre del host, el nombre del usuario que se creará y su contraseña.



El Tiempo En Casa - ASIR

Después de realizar todas estas configuraciones el sistema comenzará a instalarse, y una vez se complete la instalación se apagará la máquina y se quitará la imagen ISO para que pueda iniciarse el nuevo sistema instalado.

Ahora al iniciar la maquina se puede ver que el sistema está instalado y funciona correctamente.

```
Ubuntu 22.04.1 LTS apache tty1
apache login: apache
Password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-71-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed May  3 06:09:55 PM UTC 2023

System load:        0.095703125
Usage of /:         39.3% of 14.66GB
Memory usage:       5%
Swap usage:         0%
Processes:          110
Users logged in:    0
IPv4 address for ens18: 192.168.1.140
IPv6 address for ens18: 2a0c:5a81:a30a:c800:b8c1:24ff:fe89:73af

136 updates can be applied immediately.
63 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

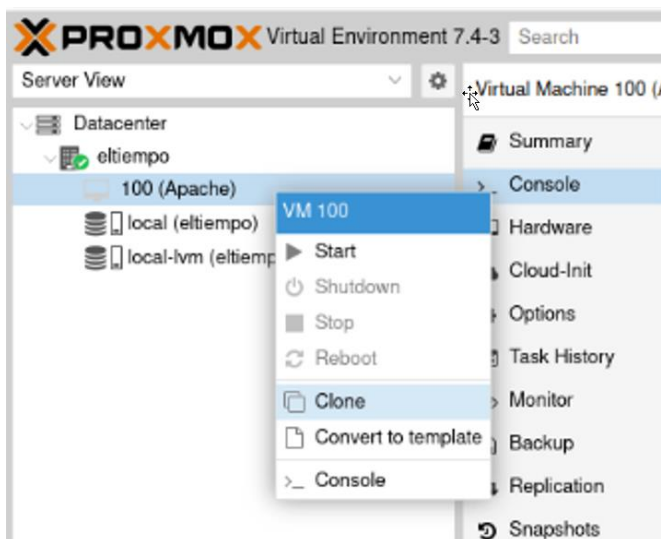
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

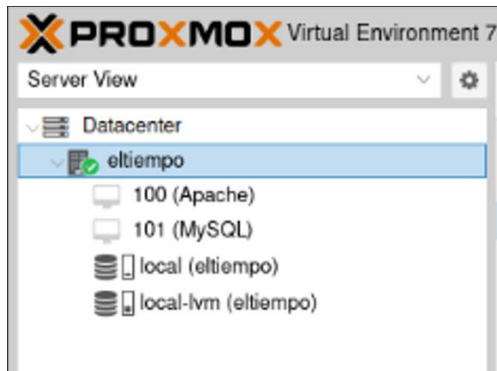
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

apache@apache:~$
```

A continuación, se configurará el servidor MySQL, como este va a usar el mismo sistema que se acaba de instalar y al que aún no se le ha realizado ningún cambio simplemente se clonara cambiando los parámetros que sean necesarios. Para esto primero se apaga la máquina, y después se le da clic derecho y se elige la opción “Clone”.



Se pedirá el nombre para la nueva máquina clonada, que en este caso será MySQL y se comenzará el proceso. Al finalizar se habrá creado una maquina idéntica a la de apache.



Ahora se van a añadir los dos discos que se usaran para la copia de seguridad a la maquina MySQL. Para esto, primero en el servidor se va a la sección “Disks”, aquí se pueden ver los dos discos que hay conectados al servidor Proxmox.

Updates

Repositories

Firewall

Disks

LVM

LVM-Thin

Directory

ZFS

ReloadShow S.M.A.R.T. valuesInitialize Disk with GPTWipe Disk

Device	Type	Usage	Size	GPT	Model
/dev/sda	unknown	partitions	107.37 GB	Yes	VBOX_HARDDISK
/dev/sda1	partition	BIOS boot	1.03 MB	Yes	
/dev/sda2	partition	EFI	536.87 MB	Yes	
/dev/sda3	partition	LVM	106.84 GB	Yes	
/dev/sdb	unknown	No	5.37 GB	Yes	VBOX_HARDDISK
/dev/sdc	unknown	No	5.37 GB	Yes	VBOX_HARDDISK

Dentro de “Disks” se elige “Directory” y se crea uno nuevo para cada disco.

Create: Directory

Disk:

/dev/sdb

Filesystem:

ext4

Name:

backup1

Add Storage:

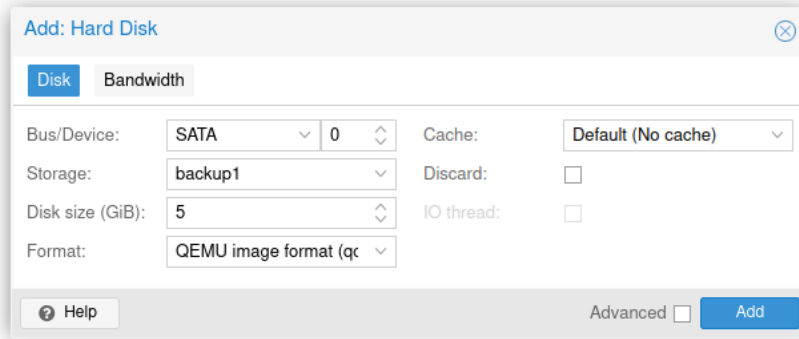
☒

Help

Create

Reload	Create: Directory
Path ↑	Device
/mnt/pve/backup1	/dev/disk/by-uuid/72058331-da18-4168-99a8-5e623dba0653
/mnt/pve/backup2	/dev/disk/by-uuid/3d2e0ffc-0183-4fe9-a7ce-96dcd17dceb

Con los dos “Directories” creados, en la maquina MySQL se añaden los dos nuevos discos en la sección “Hardware > “Hard disks”



Una vez añadidos los discos, se iniciará la máquina y se cambiarán algunos parámetros, primero se cambiará el nombre del host que, para esto se accede al archivo /etc/hostname y se cambiará el nombre de Apache a MySQL. Después se accede al archivo /etc/hosts y se cambia el nombre de la dirección local de la misma forma.

También se creará un nuevo usuario MySQL y se le dan permisos de root

```
sudo adduser mysql  
sudo usermod -s /bin/bash mysql.
```

Se reinicia la maquina y se puede ver que se puede iniciar sesión correctamente con el nuevo usuario y que el nombre del host ha cambiado.

```
Ubuntu 22.04.1 LTS mysql tty1  
mysql login: mysql  
Password:  
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-71-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Wed May 3 06:27:48 PM UTC 2023  
  
System load:        0.1240234375  
Usage of /:          39.7% of 14.66GB  
Memory usage:       5%  
Swap usage:         0%  
Processes:          111  
Users logged in:    0  
IPv4 address for ens18: 192.168.1.140  
IPv6 address for ens18: 2a0c:5a81:a30a:c800:10f1:40ff:fe45:cf41  
  
136 updates can be applied immediately.  
63 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
mysql@mysql:~$ _
```

Después se elimina el usuario apache y su directorio, ya que no es necesario en esta máquina

```
sudo deluser --remove-home apache
```

Por último, solo faltaría cambiar la dirección IP de este servidor, ya que al ser un clon tiene la misma que el servidor Apache. Como ya se ha visto antes la configuración de red final se hará más adelante, por lo que ahora solo se asigna una IP para que la maquina tenga conexión a internet y se puedan instalar y configurar los servicios.

Para cambiar la dirección se edita el archivo `/etc/netplan/00-installer-config.yaml` y en el apartado “adresses” se cambia la dirección 192.168.1.140 por 192.168.1.150

```
GNU nano 6.2                                00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    ens18:
      addresses:
        - 192.168.1.150/24
      nameservers:
        addresses:
          - 8.8.8.8
        search: []
      routes:
        - to: default
          via: 192.168.1.1
      version: 2
```

Después se aplica la nueva configuración de red

```
sudo netplan apply
```

Y ya se pueden iniciar los dos servidores correctamente.

```
Ubuntu 22.04.1 LTS mysql tty1
mysql login: mysql
Password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-71-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed May  3 06:33:19 PM UTC 2023

System load:  0.00244140625
Usage of /:   39.8% of 14.66GB
Memory usage: 5%
Swap usage:   0%
Processes:    103
Users logged in: 0
IPv4 address for ens18: 192.168.1.150
IPv6 address for ens18: 2a0c:5a81:a30a:c800:10f1:40ff:fe45:cf41

136 updates can be applied immediately.
63 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Last login: Wed May  3 18:30:25 UTC 2023 on tty1
mysql:mysql~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=118 time=2.06 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=118 time=1.84 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 1.844/1.951/2.059/0.107 ms
mysql:mysql~$ ping 192.168.1.140
PING 192.168.1.140 (192.168.1.140) 56(84) bytes of data.
64 bytes from 192.168.1.140: icmp_seq=1 ttl=64 time=0.793 ms
64 bytes from 192.168.1.140: icmp_seq=2 ttl=64 time=0.438 ms
^C
--- 192.168.1.140 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.438/0.615/0.793/0.177 ms
mysql:mysql~$ _

Ubuntu 22.04.1 LTS apache tty1
apache login: apache
Password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-71-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed May  3 06:33:26 PM UTC 2023

System load:  0.0048828125
Usage of /:   39.7% of 14.66GB
Memory usage: 5%
Swap usage:   0%
Processes:    109
Users logged in: 0
IPv4 address for ens18: 192.168.1.140
IPv6 address for ens18: 2a0c:5a81:a30a:c800:b8c1:24ff:fe89:73af

136 updates can be applied immediately.
63 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Last login: Wed May  3 18:29:46 UTC 2023 on tty1
To run a command as administrator (user 'root'), use "sudo <command>"
See "man sudo_root" for details.

apache@apache:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=118 time=2.36 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=118 time=2.40 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 2.360/2.380/2.401/0.020 ms
apache@apache:~$ ping 192.168.1.150
PING 192.168.1.150 (192.168.1.150) 56(84) bytes of data.
64 bytes from 192.168.1.150: icmp_seq=1 ttl=64 time=0.856 ms
64 bytes from 192.168.1.150: icmp_seq=2 ttl=64 time=0.324 ms
^C
--- 192.168.1.150 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.324/0.590/0.856/0.266 ms
apache@apache:~$
```

Servicio web Apache

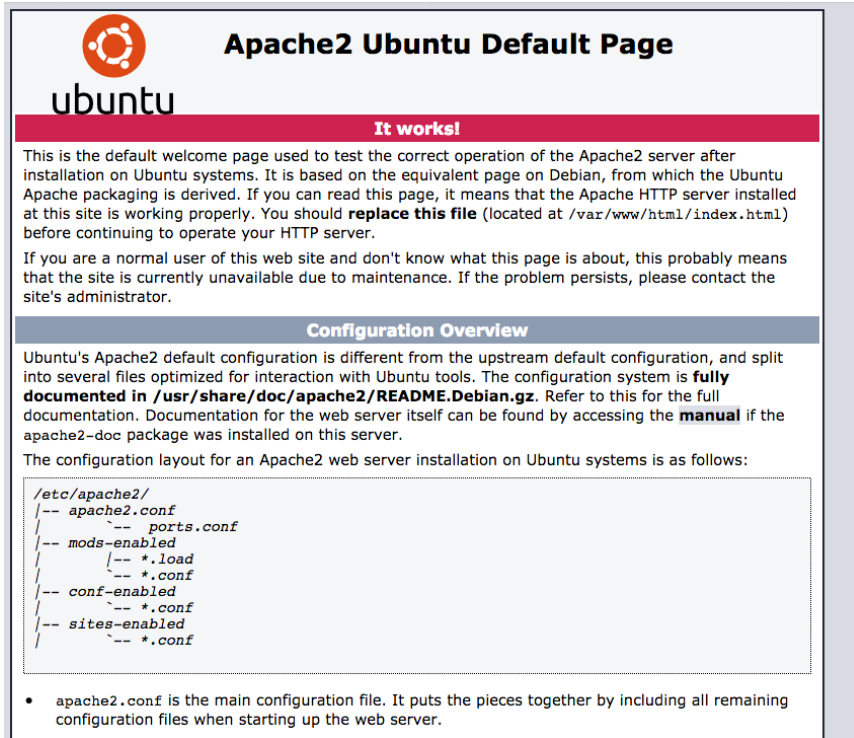
Para instalar y configurar el servicio Apache lo primero es instalar el paquete

```
sudo apt install apache2
```

Una vez se complete la instalación se comprueba que el servicio esta iniciado correctamente

```
sudo systemctl status apache2
```

Se puede acceder a la IP del servidor para comprobar que tenemos la página por defecto



Ahora se configura un nuevo virtual host para nuestro portal

Primero se crea el fichero que contendrá los archivos en /var/www

```
sudo mkdir /var/www/eltiempotfg.duckdns.org
```

después se hace que el fichero pertenezca al usuario local

```
sudo chown -R $USER:$USER /var/www/eltiempotfg.duckdns.org
```

Después se cambian los permisos del fichero

```
sudo chmod -R 755 /var/www/eltiempotfg.duckdns.org
```

Se crea un sitio simple para comprobar el funcionamiento que simplemente contendrá un título. Después se crea el archivo de configuración necesario en /etc/apache2/sites-available

```
sudo nano /etc/apache2/sites-available/eltiempotfg.duckdns.org.conf
```

```
GNU nano 6.2
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    ServerName eltiempotfg.duckdns.org
    ServerAlias eltiempotfg.duckdns.org
    DocumentRoot /var/www/eltiempotfg.duckdns.org
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    RewriteEngine on
    RewriteCond %{SERVER_NAME} =eltiempotfg.duckdns.org
    RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
</VirtualHost>
```

Ahora se activa el nuevo sitio y se desactiva el sitio por defecto

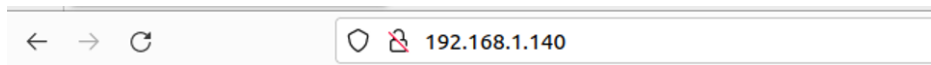
```
sudo a2ensite eltiempo.duckdns.org.conf
```

```
sudo a2dissite 000-default.conf
```

Y después se comprueba que la configuración es correcta

```
apache@apache:~$ sudo apache2ctl configtest
Syntax OK
```

Se puede comprobar que la página de prueba está disponible, después se cambiara esta por el portal web que se va a desarrollar.



La pagina funciona correctamente

Servicio de base de datos MySQL

Para instalar y configurar el servicio MySQL lo primero es instalar el paquete

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

Una vez se complete la instalación se comprueba que el servicio esta iniciado correctamente

```
sudo systemctl status mysql.service
```

después se comienza a configurar el servicio, para empezar, hay que establecer una contraseña para el usuario “root” de la base de datos

```
sudo mysql
```

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '*****';
```

Después, se ejecutará un script que se incluye con la instalación de MySQL que configurará algunos parámetros para reforzar la seguridad del sistema.

```
sudo mysql_secure_installation
```

También se debe cambiar un parámetro en los archivos de configuración, ya que por defecto la base de datos solo aceptara conexiones locales

Para esto, se accede al archivo /etc/mysql/mysql.conf.d/mysqld.cnf

Y se comenta la línea “bind-address = 127.0.0.1” añadiendo una almohadilla delante.

Una vez cambiada la configuración y completado el script la base de datos ya estará accesible y lista para los próximos pasos como crear los usuarios o las tablas.

Configuración de la red

Una vez los servicios están instalados y configurados se puede cambiar la configuración de red temporal, que ofrecía conexión a internet a las dos máquinas, a su forma final.

El servidor web tendrá dos interfaces, una conectada al router, y por lo tanto con salida a internet y otra conectada a una red interna.

El servidor de base de datos solo tendrá una interfaz conectada a la red interna, por lo que no tendrá salida a internet.

Primero se configuran las interfaces que se añadirán a las maquinas en la interfaz de Proxmox.

En la pestaña “Network” se pueden ver los puentes configurados, es decir, interfaces de red virtuales. Ahora mismo solo hay una, que es la que se crea por defecto, está conectada al router y ofrece acceso a internet ya que está vinculada a la interfaz real de la máquina.

Hay que crear otro puente, que no tendrá ninguna interfaz real asociada ya que solo conectará dos máquinas virtuales.

7.4-3 Search

Node 'eltiempo'

Create Revert Edit Remove Apply Configuration

Name ↑	Type	Active	Autostart	VLAN a...	Ports/Slaves	Bond Mode	CIDR	Gateway
enp0s3	Network Device	Yes	No	No				
vbr0	Linux Bridge	Yes	Yes	No	enp0s3		192.168.1.130/24	192.168.1.1

Search Summary Notes Shell System Network Certificates

Se crea la nueva interfaz y se le asigna una red distinta, mientras que la red domestica del router es la 192.168.1.0 esta red para la nueva interfaz será la 192.168.0.0. No se le asocia ninguna interfaz real.

Create: Linux Bridge

Name: vbr1 Autostart: ☒

IPv4/CIDR: 192.168.0.0/24 VLAN aware: ☐

Gateway (IPv4): Bridge ports:

IPv6/CIDR: Comment:

Gateway (IPv6):

Help Advanced Create

En el servidor apache ya está la interfaz por defecto, por lo que se añade esta nueva interfaz para que tenga las dos, la que lo conecta a la red doméstica y por lo tanto a internet y la que lo conecta a la red interna con la base de datos.

Network Device (net0)	virtio=BA:C1:24:89:73:AF,bridge=vmbr0,firewall=1
Network Device (net1)	virtio=B2:7F:5A:07:A8:5E,bridge=vmbr1,firewall=1

Ahora en el servidor de base de datos se cambia la interfaz por defecto que ofrecía conexión a la red domestica por la nueva interfaz, que hará que solo se pueda comunicar por la red interna con el servidor web.

Network Device (net1)	virtio=B2:7F:5A:07:A8:5E,bridge=vmbr1,firewall=1
-----------------------	--

Las interfaces ya están configuradas, pero hay que asignar las IP correspondientes a cada maquina y cada interfaz.

En el servidor Apache se mantiene la configuración de IP para la interfaz que conecta con la red doméstica, con la dirección 192.168.1.140. Además, se configura la nueva interfaz con una IP para la red interna que será 192.168.0.10.

En el servidor MySQL se cambia la IP, que anteriormente era 192.168.1.150 para la red domestica por una IP para la red interna que será 192.168.0.20

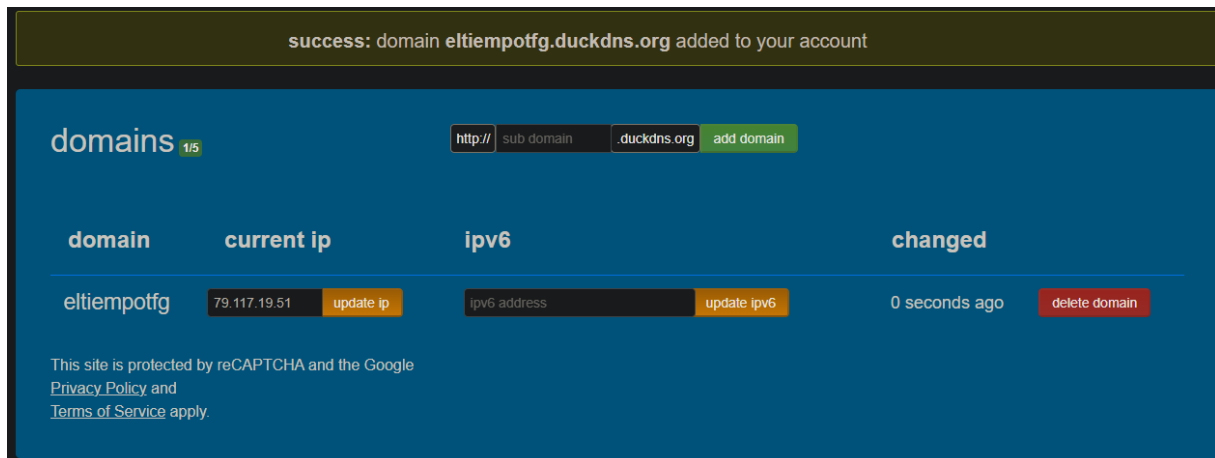
Se puede comprobar que el servidor apache tiene conexión con internet y también puede comunicarse con el servidor de base datos gracias a la red interna.

```
2: ens18: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether ba:c1:24:89:73:af brd ff:ff:ff:ff:ff:ff
    altname enp0s18
    inet 192.168.1.140/24 brd 192.168.1.255 scope global ens18
        valid_lft forever preferred_lft forever
    inet6 2a0c:5a81:a10a:1d00:b8c1:24ff:fe89:73af/64 scope global mngtmpaddr noprefixroute
        valid_lft forever preferred_lft forever
    inet6 fe80::b8c1:24ff:fe89:73af/64 scope link
        valid_lft forever preferred_lft forever
3: ens19: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether b2:7f:5a:07:a8:5e brd ff:ff:ff:ff:ff:ff
    altname enp0s19
    inet 192.168.0.10/24 brd 192.168.0.255 scope global ens19
        valid_lft forever preferred_lft forever
    inet6 fe80::b07f:5aff:fe07:a85e/64 scope link
        valid_lft forever preferred_lft forever
apache@apache:~$ ping 192.168.0.20
PING 192.168.0.20 (192.168.0.20) 56(84) bytes of data.
64 bytes from 192.168.0.20: icmp_seq=1 ttl=64 time=0.558 ms
64 bytes from 192.168.0.20: icmp_seq=2 ttl=64 time=0.578 ms
^C
--- 192.168.0.20 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1023ms
rtt min/avg/max/mdev = 0.558/0.568/0.578/0.010 ms
apache@apache:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=118 time=2.54 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=118 time=2.12 ms
```

Configuración del DDNS y el router

Para hacer posible el acceso al servicio desde internet, primero se configurará el servicio DDNS, lo primero que hay que hacer es crear la URL. Para ello, en la página del proveedor elegido hay que registrarse para poder usar el servicio, se ofrecen varias opciones, como iniciar sesión con Google, GitHub etc.

Una vez iniciada la sesión aparece el panel principal, que es muy sencillo y ofrece la posibilidad de crear el nuevo dominio. Solo hay que escribir la dirección, y si está libre se creará y estará listo para su uso.



Después, en la propia página se puede consultar como configurar la actualización automática de la URL, se va a crear un script que realizara esta tarea y se configurara en Crontab para que se ejecute cada 5 minutos.

Primero se crea el script:

```
nano duck.sh
echo url="https://www.duckdns.org/update?domains=eltiempotfg&token=***&ip=" | curl -k -o
~/duckdns/duck.log -K -
```

Se hace ejecutable

```
chmod 700 duck.sh
```

Y se abre el archivo crontab y se añade la línea que ejecuta el script cada 5 minutos

```
crontab -e
*/5 * * * * ~/duckdns/duck.sh >/dev/null 2>&1
```

Una vez configurado el servicio DDNS se configurará el router para abrir el puerto 443, usado para el tráfico de datos web HTTPs y dirigir todo el tráfico de este a la IP de nuestro servidor Apache, en este caso 192.168.1.140. (también se abre temporalmente el puerto 80 para realizar algunas pruebas)

No se entrará en demasiado detalle sobre este proceso ya que puede variar mucho dependiendo del fabricante o modelo del router, normalmente se accede a la dirección IP de la puerta de enlace desde el navegador (en este caso 192.168.1.1) y se inicia sesión con las credenciales que se pueden encontrar en el propio router para configurar los parámetros.

En este caso para configurar los puertos se accede a la sección “Internet”>”Seguridad”>”PortForwarding” y aquí se configura el puerto o rango de puertos a abrir y la IP de destino dentro de la red local.



The screenshot shows a web interface for configuring port forwarding. At the top, there's a tab labeled 'HTTP/s' and a status indicator with 'Activado' (checked) and 'Desactivado' (unchecked) radio buttons. Below this, the configuration fields are as follows:

Nombre	HTTP/s
Protocolo	TCP y UDP
Dirección IP del host WAN	0 . 0 . 0 . 0 ~ 0 . 0 . 0 . 0
LAN Host	192.168.1.140
Puerto WAN	79 ~ 450
Puerto de host LAN	79 ~ 450

At the bottom right, there are two buttons: 'Aplicar' (Apply) and 'Cancelar' (Cancel).

Ahora se puede comprobar que a través de internet con la URL se puede acceder al servidor apache



Configuración certificado HTTPS

Ahora se va a configurar el certificado para que se pueda acceder a los portales web de forma segura mediante HTTPS.

Para esto es necesario obtener un certificado para el dominio, instalarlo en el servidor y configurar Apache para que lo use y funcione con HTTPS, ya que como se ha visto antes, se ha configurado para funcionar con HTTP en el puerto 80.

Existe una herramienta llamada “certbot” muy sencilla de usar que realizara todas estas tareas de forma prácticamente automática.

Solicitará un certificado para nuestro dominio a la entidad certificadora LetsEncrypt, después lo instalará en el servidor y configurará Apache para trabajar con HTTPS.

Primero se instala la herramienta

```
sudo apt install certbot python3-certbot-apache
```

Y después simplemente se ejecuta

```
sudo certbot --apache
```

Aparecerán diferentes pantallas que solicitaran cierta información para obtener el certificado, después a partir de los dominios configurados en Apache se seleccionara cual va a funcionar con HTTPS, en este caso se elige eltiempo.duckdns.org.

```
apache@apache:~/duckdns$ sudo certbot --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): pablobras3@gmail.com

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.3-September-21-2022.pdf. You must
agree in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: Y

-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: N
Account registered.

Which names would you like to activate HTTPS for?
-----
1: eltiempotfg.duckdns.org
-----
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 1
Requesting a certificate for eltiempotfg.duckdns.org

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/eltiempotfg.duckdns.org/fullchain.pem
Key is saved at: /etc/letsencrypt/live/eltiempotfg.duckdns.org/privkey.pem
This certificate expires on 2023-08-09.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for eltiempotfg.duckdns.org to /etc/apache2/sites-available/eltiempotfg.duckdns.org-le-ssl.conf
Congratulations! You have successfully enabled HTTPS on https://eltiempotfg.duckdns.org
```

Automáticamente se configurará el certificado y un nuevo “virtualhost” preparado para la conexión HTTPS.

```
GNU nano 6.2 eltiempotfg.duckdns.org-le-ssl.conf
<IfModule mod_ssl.c>
<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    ServerName eltiempotfg.duckdns.org
    ServerAlias eltiempotfg.duckdns.org
    DocumentRoot /var/www/eltiempotfg.duckdns.org
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    SSLCertificateFile /etc/letsencrypt/live/eltiempotfg.duckdns.org/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/eltiempotfg.duckdns.org/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>
```

Se puede comprobar que ya se puede acceder al sitio correctamente mediante HTTPS con la URL



Estructura y comunicación de la base de datos

En la base de datos se necesita principalmente almacenar los registros meteorológicos, aunque también será necesario almacenar los datos de los diferentes sensores que existen.

Primero se almacenará en una tabla los datos de los sensores, estos estarán identificados por un ID único y tendrán nombre y ubicación. También se debe almacenar el código de seguridad que los sensores usaran para autentificar el envío de datos, se almacenara únicamente un hash que se usara para verificar el código enviado por los sensores.

Después en otra tabla se almacenarán los registros en sí. Aquí cada registro tendrá un ID único generado automáticamente, se almacenará la fecha y hora en la que el registro ha llegado a la base de datos de forma automática, la temperatura y humedad y la ID del sensor que ha generado el dato, como clave foránea relacionando esta tabla con la anterior. Esto hará más sencillo extraer los datos, y también hará que cuando se elimine o modifique un sensor se borren todos los datos relacionados con este.

Por lo tanto, la base de datos quedaría de la siguiente forma.

Registros			Estaciones		
Campo	Propiedades	Ejemplo	Campo	Propiedades	Ejemplo
ID	Number Autoincrement	27	ID	Number	2
ID_sensor	Number FK:Estaciones.ID	5	Nombre	Varchar	Salon
Fecha	Datetime Current timestamp	d/m/y- h:m:s	Ubicacion	Varchar	Interior
Temperatura	Number Float	23.5	Cod_seg	Varchar	\$2dfjvje
Humedad	Number Float	48.3			

En cuanto a la comunicación con la base de datos, para la inserción de los datos el código PHP recibirá del sensor el ID, temperatura, humedad y el código de seguridad. Con el ID, sacara de la base de datos el código hashado correspondiente a ese ID y lo verificara. Si es correcto, enviará a la base de datos la lectura de la temperatura y la humedad con la ID del sensor. La base de datos creara el id único y rellenara el campo de la fecha y la hora con la de ese momento.

Para consultar los datos, el cliente obtendrá los datos de la tabla registros y estaciones con una sentencia “join”, ya que el campo “ID_sensor” en los registros está relacionado con ID en estaciones.

El cliente también podrá añadir o borrar registros de la tabla estaciones y editar el nombre y ubicación de las existentes, así como eliminar registros.

Creación de la base de datos y sus usuarios

Se va a crear la base de datos y el usuario que va a usar el código PHP para conectarse a la base de datos y realizar sus funciones, así como las diferentes tablas.

Primero se crea la base de datos en sí y se selecciona.

```
CREATE DATABASE eltiempo;  
USE eltiempo;
```

Después se crea el usuario.

```
CREATE USER 'tfg' IDENTIFIED WITH mysql_native_password BY '****';
```

Es importante crear un nuevo usuario en lugar de usar “root” para mejorar la seguridad.

Se le asignaran los permisos necesarios para que pueda realizar sus funciones, en este caso son:

SELECT: permite extraer datos de la base de datos.

DELETE: permite borrar datos de una tabla.

INSERT: permite insertar datos en una tabla.

UPDATE: permite modificar los datos en una tabla.

Se dan estos permisos al usuario en la base de datos “eltiempo”.

```
GRANT INSERT, UPDATE, DELETE, SELECT on eltiempo.* TO 'tfg';
```

Después se crean las tablas y restricciones necesarias con la estructura planeada anteriormente

```
CREATE TABLE `estaciones` (  
  `ID` int(3) NOT NULL,  
  `Nombre` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish_ci NOT NULL,  
  `Ubicacion` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish_ci NOT NULL,  
  `Cod_seguridad` varchar(100) CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish_ci NOT  
  NULL  
)
```

```
CREATE TABLE `registros` (  
  `ID` int(6) NOT NULL,  
  `ID_sensor` int(3) NOT NULL,  
  `Temperatura` float NOT NULL,  
  `Humedad` float NOT NULL,  
  `Fecha` datetime NOT NULL DEFAULT current_timestamp()  
)
```

```
ALTER TABLE `estaciones`  
  ADD PRIMARY KEY (`ID`);
```

```
ALTER TABLE `registros`  
  ADD PRIMARY KEY (`ID`),  
  ADD KEY `sensor_id` (`ID_sensor`);
```

```
ALTER TABLE `registros`  
  ADD CONSTRAINT `sensor_id` FOREIGN KEY (`ID_sensor`) REFERENCES `estaciones` (`ID`) ON  
  DELETE  
  CASCADE ON UPDATE NO ACTION;
```


Copias de seguridad de la base de datos

A continuación, se va a configurar la copia de seguridad automática de la base de datos, que en este caso de realizar diariamente. Primero se va a preparar el medio en el que se almacenaran las copias, como se ha visto debe de ser un medio independiente y aislado, para esto el servidor Apache tiene dos discos dedicados, que se configuraran con un sistema RAID1, esto significa que la información se almacenara duplicada en ambos discos, por lo que si uno falla la información seguirá estando en el otro y se podrá recuperar.

Para crear esta unidad RAID1 se va a usar un servicio integrado en Linux llamado “mdadm”, primero se analizará que discos existen en el sistema y sus nombres con el comando “lsblk”.

```
mysql@mysql:~$ lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0                              7:0      0   62M  1 loop /snap/core20/1587
loop1                              7:1      0  79.9M  1 loop /snap/lxd/22923
loop2                              7:2      0   47M  1 loop /snap/snapd/16292
sda                                8:0      0   32G  0 disk
├─sda1                             8:1      0    1M  0 part
├─sda2                             8:2      0    2G  0 part /boot
├─sda3                             8:3      0   30G  0 part
│   └─ubuntu--vg-ubuntu--lv 253:0      0   15G  0 lvm  /
sdb                                8:16     0    5G  0 disk
sdc                                8:32     0    5G  0 disk
```

Ahora se crea el RAID con los dos discos de 5GB (sdb y sdc)

```
sudo mdadm --create --verbose /dev/md0--level=1 --raid-devices=2 /dev/sdb /dev/sdc
```

Con esto ya existe la nueva unidad que representa los dos discos en RAID. Se le da formato a esta nueva unidad, que en este caso será EXT4

```
sudo mkfs.ext4 -F /dev/md0
```

Se monta en un punto de montaje que se ha creado en /mnt

```
sudo mount /dev/md0 /mnt/md0
```

Ahora se actualiza la configuración de mdadm para que la unidad RAID esté disponible siempre al iniciar el sistema

```
sudo mdadm --detail --scan | sudo tee -a /etc/mdadm/mdadm.conf
```

Por último, se configura el archivo “Fstab” para que la unidad se monte siempre al iniciar el sistema, la unidad RAID se renombra automáticamente a “md127” ya que es detectada como unidad externa, por lo que se añade con este nombre. Se añade la siguiente línea al final del archivo etc/fstab

```
/dev/md127 /mnt/md0 ext4 defaults,nofail,discard 0 0
```

Con el almacenamiento listo solo queda crear las copias de seguridad en sí.

Primero, se crea un nuevo usuario SQL para realizar las operaciones de copia de seguridad, este necesitara únicamente los permisos SELECT y LOCK TABLES.

```
CREATE USER 'backup' IDENTIFIED WITH mysql_native_password BY '****';  
GRANT LOCK TABLES, SELECT on eltiempo.* TO 'backup';
```

Después se crea un script de Bash que ejecutará el comando “mysqldump” y guardará la copia de la base de datos, también borrará las copias antiguas, en este caso de hace más de 30 días.

```
GNU nano 6.2                                backup.sh *  
#!/bin/bash  
  
backupfolder=/mnt/md0/backup  
user=backup  
password=Tfg54321.  
keep_day=30  
sqlfile=$backupfolder/eltiempo-$(date +%d-%m-%Y).sql  
  
mysqldump -u $user -p$password -y eltiempo > $sqlfile  
if [ $? == 0 ]; then  
    echo 'SQL dump created'  
else  
    echo 'mysqldump error'  
    exit  
fi  
find $backupfolder -mtime +$keep_day -delete
```

Este script se ejecutará con Crontab cada día, para esto abre el archivo Crontab

```
sudo crontab -e
```

Y se añade la siguiente línea al final del archivo.

```
30 23 * * * /home/mysql/backup.sh
```

Esto ejecutara el script todos los días a las 23:30.

Ejecutando el script manualmente se comprueba que la copia se crea correctamente

```
mysql@mysql:~$ ./backup.sh  
mysqldump: [Warning] Using a password on the command line interface can be insecure.  
SQL dump created  
mysql@mysql:~$ cd /mnt/md0/backup/  
mysql@mysql:/mnt/md0/backup$ ls  
eltiempo-23-05-2023.sql
```

Conexión a la base de datos con PHP

Antes de comenzar a desarrollar cualquier código PHP, se va a ver primero como establecer la conexión con la base de datos. Esto se hará en un archivo independiente, que después se incluirá en los demás archivos PHP para poder conectarse de una forma sencilla.

En este archivo simplemente se establece una conexión PDO indicando la base de datos, la dirección del host, el nombre de usuario y la contraseña.

Ahora se añadiría la línea

```
include("recursos/php/plantilla.php");
```

En los demás archivos PHP para tener conexión a la base de datos.

El código completo se puede encontrar en: [GitHub](#)

PHP para la recepción de datos e inserción en la BBDD

Ya se ha visto que los sensores enviarán los datos mediante POST, y el código PHP los recibirá e insertará en la base de datos. Ahora se va a desarrollar este código.

Lo primero es comprobar que se está haciendo un envío por POST, los sensores lo harán siempre, pero un usuario podría acceder a la URL por error, por lo que se comprueba con la estructura:

```
if ($_SERVER['REQUEST_METHOD'] == 'POST')
```

Si existe un formulario POST se continuará con la ejecución, sino se mostrará un mensaje de error para algún posible usuario informando de que esta es una página interna.

Una vez comprobado que se ha enviado un formulario post, se extraerá de este primero la ID del sensor que envía los datos y se consultará en la tabla sensores el hash del código de seguridad que corresponde a ese sensor. Se extraerá el código que ha enviado el sensor y se comparará con el de la base de datos con la función

```
password_verify($codigo, $hash)
```

Si es correcto se continuará la ejecución, sino se enviará un mensaje de error.

Una vez comprobado el código, se combinan la temperatura, humedad e ID del sensor recibido en una sentencia “insert” que podría ser como la siguiente:

```
INSERT INTO `registros` (`ID_sensor`, `humedad`, `temperatura`) VALUES ('5', 47.1, 21.4)
```

Se ejecuta la sentencia en el servidor de base de datos, que rellenará automáticamente el campo de la fecha y el de la ID del registro.

El código completo se puede encontrar en: [GitHub](#)

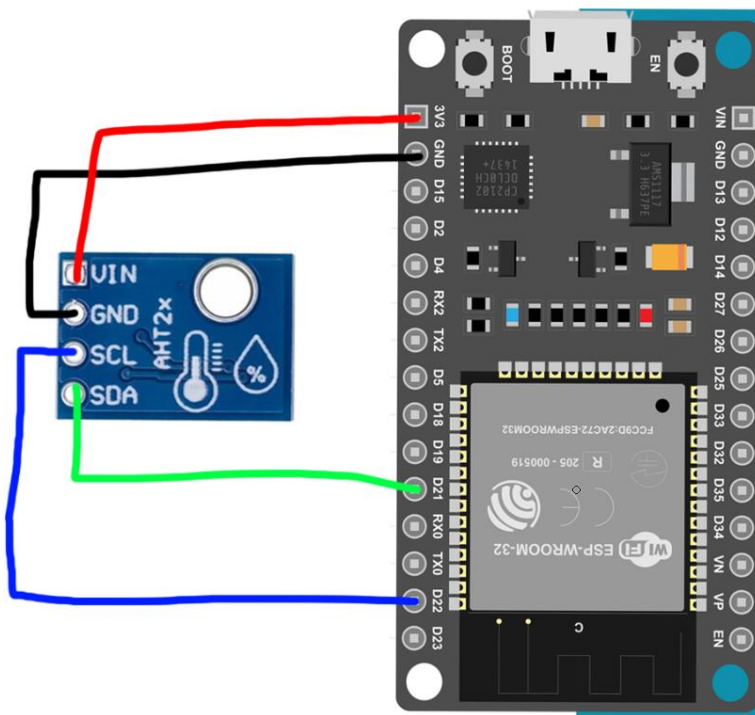
Montaje y código de los sensores

A continuación, se van a montar los propios sensores y desarrollar el código para que puedan enviar los datos con un formulario POST.

Como se ha visto, se va a usar la placa ESP32 y el sensor AHT21. Para conectarlos físicamente se podrán usar cables hembra-hembra para unir los pines directamente o un protoboard. También se pueden soldar con estaño para una solución más permanente.

Para unir correctamente los pines se consultan las fichas técnicas de ambos, el sensor dispone de 4 pines, que son VCC, GND, SDA y SCL. Estos tendrán que conectarse con la placa ESP32 de la siguiente forma:

- GND-GND** (Negativo de la alimentación y comunicación)
- VCC-3.3V** (Positivo de la alimentación, debe ser de 3.3 voltios para el sensor)
- SDA- D21** (D21 es el pin que corresponde a SDA para la comunicación IC2)
- SCL- D22** (D22 es el pin que corresponde a SCL para la comunicación IC2)



Una vez conectado el sensor con la placa ESP32 se desarrollará el código para que se realice el envío de datos. Al ser una placa basada en Arduino se usa un lenguaje basado en C++, y se usan librerías, que se incluyen para añadir funciones de forma sencilla al programa. En este caso se empleará una librería para conectar la placa a la red WiFi, una para configurar el certificado HTTPS para una conexión segura, una para comunicarse con el sensor AHT21 y una para realizar solicitudes web, como “POST”.

En el código, primero se definen las diferentes variables, algunas serán configuradas por la web al añadir un nuevo sensor, como la SSID y contraseña del WiFi o la frecuencia de envío de datos. También se incluye el certificado raíz de una entidad certificadora, que permitirá la conexión segura HTTPS y la URL a la que se enviarán las solicitudes.

Después se incluyen las librerías mencionadas anteriormente y se inicia la conexión WiFi, la comunicación con el sensor y se inicia una conexión segura usando el certificado almacenado en la variable.

Después un bloque de código se ejecuta de forma repetida indefinidamente, este obtiene los valores de temperatura y humedad del sensor y con las variables como la ID y el código de seguridad crea una cadena, después se inicia la conexión con la URL especificada y se envía la cadena en forma de solicitud “POST”.

Para realizar pruebas y poder controlar el proceso envío se comunica por el puerto serie el resultado de cada petición para poder verlo desde un ordenador.

Se espera el tiempo designado, que normalmente serán 10 minutos para repetir el proceso.

El código completo se puede encontrar en: [GitHub](#)

Estructura del portal web

A continuación, se va a planter la estructura y funcionalidad básica del portal web al que podrán acceder los clientes.

Primero, existirá una plantilla que se podrá incluir en todos los archivos PHP, igual que el archivo de conexión que se ha visto anteriormente. Esta contendrá la cabecera de la web, con un título y un menú que permitirá acceder a las diferentes secciones, que serán las siguientes:

-Sección principal: Se mostrará por defecto cuando el usuario acceda al portal.

Aquí se podrán consultar los datos más útiles, se mostrarán las diferentes ubicaciones (todo, interior, exterior y otros) todos los sensores individuales que engloba cada una.

Se podrá hacer click en las ubicaciones (donde se podrán ver las medias, máximas o mínimas) o en los sensores individuales (donde se podrá ver la temperatura y humedad actual, que será la última registrada, y las máximas, mínimas y medias)

También se podrán filtrar estos datos por periodo de tiempo.

-Sección registros: Se podrán ver todos los registros existentes en la base de datos. Se podrán filtrar por sensor, periodo de tiempo o los dos. Los datos se podrán eliminar, existiendo también una opción para eliminar múltiples datos a la vez.

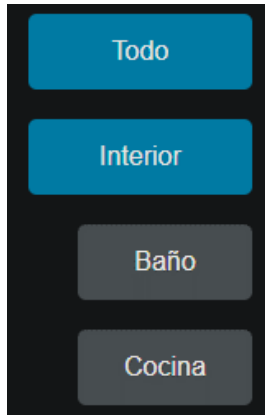
-Sección estaciones: Se podrán gestionar los sensores, todos ellos se listarán y se ofrecerá la opción de eliminarlos o editarlos. Existirá también una opción para añadir nuevas estaciones, aquí se introducirán todos los datos necesarios, se validarán y después se añadirá el nuevo sensor a la base de datos y se mostrará el diagrama de conexión y el código para este.

-Sección de información: No tendrá ninguna funcionalidad en especial, se podrán encontrar algunos datos, estadísticas e información sobre el proyecto y el servidor.

Funciones básicas del portal web

Se van a comenzar a desarrollar las funciones básicas de cada una de las secciones del portal web, más adelante se añadirán las más avanzadas.

-Página principal: se listarán las diferentes ubicaciones (todo, interior, exterior y otro) y se listarán todos los sensores individuales que engloba cada una. Las ubicaciones están directamente en el código, mientras que los sensores de cada ubicación se obtienen desde la tabla estaciones y se muestran con una estructura “foreach”. El resultado sería el siguiente:



Cada una de estas cajas será un enlace, que llevará a la misma página, pero cambiará un parámetro en la URL. Así, al hacer click en cualquiera de ellas, se refrescará la página y el código podrá saber que se ha elegido.

Si se elige un sensor individual, se obtendrá su ID de la tabla registros y se creará una sentencia SQL personalizada para obtener los datos únicamente del sensor elegido.

Por ejemplo, para obtener la temperatura máxima de un sensor en concreto:

```
SELECT Temperatura, Fecha FROM registros WHERE Temperatura = (SELECT MAX(temperatura) FROM registros WHERE ID_sensor=2)
```

Si se elige una ubicación, se obtendrán todas las ID de los sensores pertenezcan a esa ubicación, y se creará una sentencia SQL personalizada para obtener los datos entre los sensores de la ubicación elegida. Es importante saber de qué sensor en concreto es el dato, ya que existen varios en cada ubicación, por lo que la sentencia usará la estructura “join”. Por ejemplo, para obtener la temperatura mínima y el sensor que la ha generado de un grupo de sensores:

```
SELECT estaciones.Nombre, Temperatura, Fecha FROM registros INNER JOIN estaciones ON registros.ID_sensor = estaciones.ID AND Temperatura = (SELECT MIN(temperatura) FROM registros WHERE ID_sensor IN (1,2,3,4))
```

Si se elige la ubicación todos, se obtendrán los datos entre todos los sensores, sin ningún filtro.

-Sección registros: Se mostrarán todos los datos con el nombre del sensor que los ha creado. Para esto se crea una sentencia “select” de la tabla registros con “join” a la tabla estaciones como la siguiente:

```
SELECT registros.ID, ID_sensor, Temperatura, Humedad, Fecha, estaciones.Nombre, estaciones.Ubicacion FROM registros INNER JOIN estaciones ON registros.ID_sensor = estaciones.ID
```

Se obtienen los datos de temperatura y humedad, el nombre del sensor que ha creado el registro y la fecha, y se ejecuta en una estructura foreach para crear una fila en una tabla por cada registro. El resultado sería el siguiente:

ID	Sensor	Temperatura	Humedad	Fecha
144	Terraza	24.2	35.3	2023-05-11 16:31:36
143	Dormitorio	11.9	35.6	2023-05-11 16:31:33
142	Salon	24	37.4	2023-05-11 16:31:31
141	Cocina	22.1	51.3	2023-05-11 16:31:29

-Sección sensores: Se listarán todos los sensores existentes, esto se hará obteniendo la lista con una sentencia “select” de la tabla sensores que se ejecutará en una estructura “foreach” y creará una fila en una tabla por cada sensor existente. También se mostrará por cada estación un botón borrar, que será un enlace a la misma página, pero cambiará un parámetro en la URL, que será el ID de el sensor. Así, al hacer click la página se refrescará, y el código comprobará si este parámetro existe. Si es así ejecutara una sentencia SQL que borrara el sensor con esa ID.

ID	Nombre	Ubicacion	Opciones
1	Baño	Interior	<button>Borrar</button>
2	Cocina	Interior	<button>Borrar</button>
3	Salon	Interior	<button>Borrar</button>

-Sección información: Se mostrará información sobre el proyecto y el servidor, como la versión, el protocolo de conexión y el número total de registros procesados y el enlace al repositorio de GitHub con todo el código del proyecto.

El código completo se puede encontrar en: [GitHub](#)

Funciones avanzadas del portal web

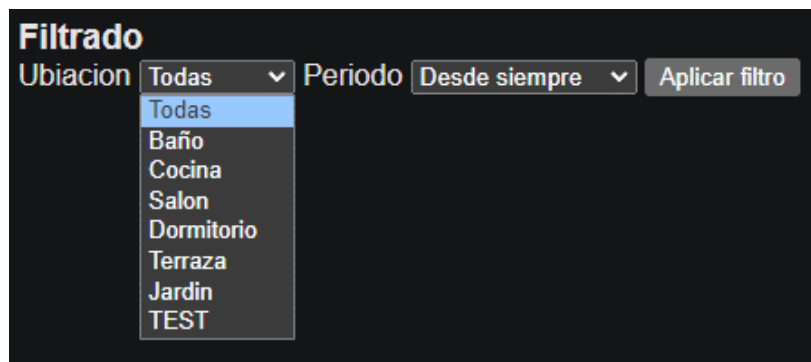
Una vez desarrollada la funcionalidad básica, se van a implementar las funciones más avanzadas

-Página principal: Se añade un filtro para poder ver los datos filtrados por periodo de tiempo. Para esto se añade un formulario con un desplegable en el que se puede elegir el periodo y un botón para aplicar el filtro. Al pulsar este botón se refrescará la página, y el código comprobará si hay datos en el formulario del filtro. Si es así, añadirá el filtro a la sentencia “select” con el periodo especificado. Por ejemplo, para obtener la temperatura mínima de un sensor en las últimas 24 horas:

```
SELECT estaciones.Nombre, Temperatura, Fecha FROM registros INNER JOIN estaciones ON registros.ID_sensor = estaciones.ID AND Temperatura = (SELECT MIN(temperatura) FROM registros WHERE ID_sensor IN (1,2,3,4,5,6,7) AND fecha > now() - interval 24 hour)
```

-Sección registros: Existirá un filtro de periodo, que funcionará igual que el anterior y uno de sensor, para este se añade un formulario en la que las opciones serán todos los sensores de la base de datos, que se obtendrán con una sentencia “select” y se listarán con una estructura “foreach”.

Existirá un botón para aplicar los filtros que refrescará la página, y el código comprobará si se está filtrando por periodo, por ubicación o por ambos y aplicará los filtros apropiados a la sentencia “select”.



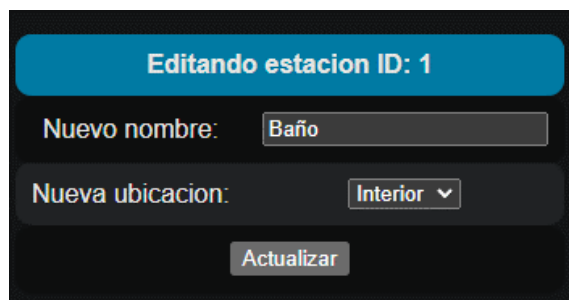
Por ejemplo, para obtener los registros de la ultima hora de un sensor en concreto:

```
SELECT registros.ID, ID_sensor, Temperatura, Humedad, Fecha, estaciones.Nombre, estaciones.Ubicacion FROM registros INNER JOIN estaciones ON registros.ID_sensor = estaciones.ID AND estaciones.Nombre='Baño' AND fecha > now() - interval 1 hour ORDER BY ID DESC
```

También se añade un formulario con una casilla seleccionable en cada registro para eliminarlo, y al final de la página una casilla para eliminar todos los datos mostrados y el botón de eliminar. Este será un enlace que llevara a la misma página, así al pulsarlo se recarga la página y se comprobaba si se ha marcado alguna casilla en algún registro o la casilla eliminar todo.

Si se ha marcado alguna casilla, se recorrerá la lista de registros seleccionados con una sentencia “foreach” y se ejecutará una sentencia “delete” en cada uno de ellos para eliminarlo, si se ha marcado la casilla eliminar todo se recorrerá la lista de todos los registros mostrados con una sentencia “foreach” y se ejecutará una sentencia “delete” para eliminar cada uno de ellos.

-Sección sensores: Se añadirá un botón editar en cada sensor, este será un enlace que llevará a una página independiente, pasando como parámetro la ID del sensor a editar. Se mostrará un formulario que se rellenará automáticamente con los datos actuales del sensor obtenidos de la base de datos, estos se podrán modificar.



Formulario para editar una estación. El título es "Editando estacion ID: 1". Hay dos campos de entrada: "Nuevo nombre:" con el valor "Baño" y "Nueva ubicacion:" con un menú desplegable que muestra "Interior". Hay un botón "Actualizar" al final.

Habrá un botón para confirmar la modificación, que ejecutará una sentencia “update” con los nuevos datos introducidos.

Además, existirá un botón para añadir nuevas estaciones, este será un enlace que llevará a otra página independiente, donde habrá un formulario en el que rellenar todos los datos necesarios para crear el sensor. Algunos parámetros se completarán automáticamente, por ejemplo, la URL, que será la de la web actual y la ID del nuevo sensor, que será la siguiente a la última existente. Habrá un botón para confirmar la creación, que será un enlace a otra página, esta obtendrá los datos del formulario y añadirá el nuevo sensor a la base de datos, además mostrará el código para el sensor ya configurado con los parámetros introducidos y el diagrama de conexión.

El código completo se puede encontrar en: [GitHub](#)



Formulario para registrar una nueva estación. El título es "Registra una nueva estación". Hay varios campos de entrada: "Estacion numero:" con el valor "8", "Nombre:" (vacío), "Ubicacion:" con un menú desplegable que muestra "-Seleccionar-", "Codigo seguridad:" (vacío), "SSID WiFi:" (vacío), "Contraseña WiFi:" (vacío), "URL:" con el valor "https://eltiempotfg.duckdns.org", "Frecuencia Actualizacion:" con un menú desplegable que muestra "10 minutos". Hay un botón "Crear" al final.

Conclusión

El desarrollo del proyecto ha sido bastante fluido y agradable, sin embargo, se han presentado ciertos problemas inesperados que ha sido necesario investigar y solventar, entre ellos:

- Se debe cambiar la contraseña del usuario “root” de la base de datos para poder ejecutar el script de configuración.

- La base de datos solo acepta conexiones locales por defecto y hay que cambiar los parámetros en el archivo de configuración.

- El nombre del disco RAID se cambia al reiniciar el equipo por ser detectado como externo.

- El usuario para realizar las copias de seguridad necesitaba algunos permisos específicos.

En cuanto a la dificultad, la instalación y configuración de los servicios ha sido relativamente sencilla a parte de los problemas mencionados anteriormente, sin embargo, la parte de programación ha sido más complicada.

Concretamente la formación de sentencias “select” para los filtros, ya que la sintaxis es delicada y se deben cerrar adecuadamente todos los paréntesis, comillas etc. Formar estas sentencias combinando variables es algo complejo y el código debe contemplar todos los casos posibles para que las sentencias, y por lo tanto los filtros, funcionen siempre.

El código Arduino para envío de formularios POST por HTTPS también ha presentado un reto, ya que no es algo demasiado común y no existe demasiada documentación ni información en línea sobre cómo hacerlo.

Finalmente, el resultado ha sido satisfactorio, ya que se han cumplido todos los requisitos que se plantearon, y se ha podido comprobar que el servicio funciona correctamente y es cómodo y conveniente de usar, además de viable económicamente.

Bibliografía

-Documentación oficial

[Proxmox](#)

[Apache](#)

[MySQL](#)

[DuckDNS](#)

[PHP](#)

[Arduino](#)

-Blogs y paginas informativas

[Sqlback](#)- Copia de seguridad de la base de datos.

[Randomnerdtutorials](#)- Conexión HTTPs con ESP32.

[IoTicos \(YouTube\)](#)- Conexión wifi de la placa ESP32 y POST de datos.

[Apalrd's adventures \(YouTube\)](#)-configuración de Porxmox (principalmente red).

DigitalOcean

[Instalación y configuración de Apache](#)

[Configuración HTTPs Apache](#)

[Instalación MySQL](#)

[Creación de la unidad RAID](#)

-Paginas de consulta y resolución de problemas, foros

[StackOverflow](#)

[W3Schools](#)