

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN  
UNIVERSIDAD POLITÉCNICA DE CARTAGENA



Universidad  
Politécnica  
de Cartagena

MIEMBRO DE



EUROPEAN  
UNIVERSITY OF  
TECHNOLOGY



**Trabajo Fin de Máster**  
**MASTER UNIVERSITARIO EN INGENIERÍA**  
**TELEMÁTICA**

**Diseño y desarrollo de un servidor IoT  
edge-computing para monitorización de  
una vivienda inteligente**



AUTORA: Alba Martínez Meroño

DIRECTOR: Alejandro S. Martínez Sala

CODIRECTOR: Francisco Miguel Moral Moreno

Septiembre / 2022





Universidad  
Politécnica  
de Cartagena

MIEMBRO DE



EUROPEAN  
UNIVERSITY OF  
TECHNOLOGY

<b>Autor</b>	Alba Martínez Meroño
<b>E-mail del autor</b>	alba_bvb@hotmail.com
<b>Director</b>	Alejandro S. Martínez Sala
<b>E-mail del director</b>	alejandros.martinez@upct.es
<b>Codirector</b>	Francisco Miguel Moral Moreno
<b>E-mail del codirector</b>	fmoral@ctcon-rm.com
<b>Título del TFM</b>	Diseño y desarrollo de un servidor IoT edge-computing para monitorización de una vivienda inteligente
<b>Resumen</b>	<p>El objetivo de este proyecto es diseñar e implementar un servidor IoT que integra la telemetría de varios sistemas y monitoriza parámetros varios de una vivienda inteligente: datos detallados de temperatura y humedad del aire, estación meteorológica, estado de puertas, ventanas y persianas, ocupación de la vivienda, estado máquinas climatización, consumo eléctrico y producción energía solar. Los datos son recopilados de forma eficiente y fiable y son organizados y presentados de forma accesible y amigable para facilitar la monitorización y análisis del comportamiento climático de dicha vivienda.</p>
<b>Titulación</b>	Máster Universitario en Ingeniería de Telemática.
<b>Departamento</b>	Tecnología de la Información y las Comunicaciones.
<b>Fecha de presentación</b>	Septiembre - 2022

## Agradecimientos

Me gustaría agradecerle a mi familia, sobre todo a mis padres, el haber conseguido finalizar tanto el grado como el máster durante estos años. Me han apoyado y se han sacrificado para que llegue a tener una gran formación y por ello un buen trabajo. Les doy las gracias, pues siempre han estado ahí para ayudarme y no han dudado sobre las acciones que he ido tomando.

Agradecer también a mis amigos, que han intentado que saliera y me divirtiera con ellos siempre que pudiese para así despejarme y animarme.

De la misma forma, a mis compañeros de carrera, con los que he podido aprender sobre muchas cosas y nos hemos ayudado mutuamente. Menciono especialmente a mi compañero de prácticas Daniel Ros con el que he pasado muy buenos momentos y a mi parecer, ambos hemos crecido como personas y hemos llegado muy lejos.

## Índice

Capítulo 1. Introducción.....	8
1.1 Antecedentes y objetivos.....	8
1.2 Herramientas software y elementos usados .....	10
1.3 Estructura y organización del proyecto .....	11
Capítulo 2. Tecnologías empleadas .....	12
2.1 Sistemas instalados que se integran en el servidor .....	12
2.2 Conceptos básicos del protocolo MQTT .....	20
2.3 Conceptos básicos del protocolo Modbus.....	22
2.4 Conceptos básicos sobre el servicio REST.....	23
Capítulo 3. Diseño e implementación de la arquitectura del servidor .....	27
3.1 Arquitectura global del servidor .....	27
3.2 Monitorización Gateway-Servidor .....	29
3.3 Recepción y procesado de datos temperatura y humedad.....	31
3.4 Recepción de mensajes mediante Bot Telegram.....	34
3.5 Recepción y procesado de datos centralita domótica .....	35
3.6 Recepción y procesado de datos estación meteorológica .....	37
3.7 Gestión y monitorización del servicio con PM2.....	39
3.8 Diseño Backend y BBDD del servidor .....	40
3.9 Diseño Frontend y descarga de archivos .....	41
3.10 Configuración del acceso al servidor desde Internet .....	43
Capítulo 4. Pruebas y resultados .....	44
4.1 Pruebas de la Base de Datos .....	44
4.2 Pruebas del Bot de Telegram.....	48
4.3 Pruebas de descarga de archivos CSV .....	51
Capítulo 5. Conclusiones y trabajos futuros.....	59
5.1 Conclusiones .....	59
5.2 Grado de consecución de los objetivos y formación adquirida .....	60
5.3 Trabajos futuros .....	61
Anexos.....	63
Listado código sensores IoT.....	63
Direcciones KNX y Modbus.....	65
Estructura GitHub .....	68
Bibliografía y referencias.....	69

## Índice de figuras

Figura 1. Esquema simplificado servidor IoT de almacenamiento y visualización de datos .....	9
Figura 2. Acceso router - información de internet .....	13
Figura 3. Acceso router - clientes comunicación inalámbrica 1 .....	14
Figura 4. Acceso router - clientes comunicación inalámbrica 2 .....	14
Figura 5. Acceso router - clientes comunicación por cable.....	14
Figura 6. Acceso router - configuración DHCP.....	15
Figura 7. Acceso router - port forwarding - NAT .....	15
Figura 8. Módulo sensor.....	17
Figura 9. Plano distribución sensores IoT.....	18
Figura 10. Módulo sensor con pértiga .....	18
Figura 11. Cámara del suelo .....	19
Figura 12. Colocado del módulo sensor en cámara del techo .....	19
Figura 13. Esquema general MQTT.....	20
Figura 14. Cronograma funcionamiento MQTT.....	21
Figura 15. Ejemplo de uso comando modpoll .....	22
Figura 16. Esquema obtención datos estación meteorológica .....	23
Figura 17. Acceso página estación meteorológica.....	24
Figura 18. Visualización datos estación meteorológica .....	24
Figura 19. Obtención API KEY estación meteorológica.....	25
Figura 20. Consulta a Postman .....	25
Figura 21. Respuesta de Postman.....	26
Figura 22. Arquitectura del sistema implementado .....	28
Figura 23. Cronograma envío de mensajes monitorización Gw-Servidor.....	29
Figura 24. Ejemplo envío real Gw-servidor .....	30
Figura 25. Esquema obtención datos sensores IoT .....	31
Figura 26. Diagrama de flujo recepción datos sensores IoT.....	32
Figura 27. Diagrama de flujo procesado datos sensores IoT.....	33
Figura 28. Ejemplo funcionamiento Bot Telegram .....	34
Figura 29. Esquema obtención datos centralita domótica .....	35
Figura 30. Diagrama de flujo recepción y procesado datos centralita domótica .....	35
Figura 31. Diagrama de flujo recepción y procesado datos estación meteorológica .....	37
Figura 32. Visualización servidor con PM2.....	39
Figura 33. Diagrama de bloques e interconexión módulos del servidor .....	40
Figura 34. Visualización página web .....	41
Figura 35. Generación archivo CSV .....	42
Figura 36. Descarga archivo CSV.....	42
Figura 37. Ventana VCN Viewer.....	43
Figura 38. Autenticación VCN Viewer .....	43
Figura 39. Tablas de la Base de Datos .....	44
Figura 40. Tabla activos.....	44
Figura 41. Tabla datos .....	45
Figura 42. Tabla datosKNX.....	45
Figura 43. Tabla datosMeteo.....	46
Figura 44. Tabla datosPotAct.....	46
Figura 45. Tabla tyh.....	47
Figura 46. Tamaño Base de Datos.....	47

Figura 47. Ejemplo Bot sin fallo .....	48
Figura 48. Ejemplo Bot con fallo .....	49
Figura 49. Ejemplo Bot con fallo espontáneo.....	50
Figura 50. Mensaje incoherencia temporal en horas .....	51
Figura 51. Mensaje incoherencia temporal en fechas .....	52
Figura 52. Mensaje inserción hora inicio .....	52
Figura 53. Mensaje inserción hora fin.....	52
Figura 54. Mensaje inserción fecha inicio .....	53
Figura 55. Mensaje inserción fecha fin .....	53
Figura 56. Generación CSV global.....	53
Figura 57. Descarga CSV sensores .....	54
Figura 58. Visualización archivo Datos_sensores.csv .....	54
Figura 59. Tamaño archivo Datos_sensores.csv .....	55
Figura 60. Descarga CSV KNX.....	55
Figura 61. Visualización archivo Datos_KNX.csv.....	55
Figura 62. Tamaño archivo Datos_KNX.csv .....	56
Figura 63. Descarga CSV potencias activas.....	56
Figura 64. Visualización archivo Datos_PotenciasActivas.csv .....	57
Figura 65. Tamaño archivo Datos_PotenciasActivas.csv .....	57
Figura 66. Descarga CSV estación meteorológica.....	58
Figura 67. Visualización archivo Datos_Meteo.csv .....	58
Figura 68. Tamaño archivo Datos_Meteo.csv .....	58
Figura 69. Estructura Github .....	68

# Capítulo 1. Introducción

## 1.1 Antecedentes y objetivos

Se parte de un caso práctico y real donde el CTCON -Centro Tecnológico de la Construcción de la Región de Murcia- está realizando un proyecto de I+D de análisis de la climatización y eficiencia energética de una vivienda unifamiliar. Este proyecto, denominado proyecto Coolchamber [1], forma parte del programa encuentro TF en colaboración con el CTCON.

La vivienda está domotizada y sensorizada pero el objetivo es integrar la telemetría de varios sistemas y facilitar su explotación y análisis de datos para el estudio de la eficiencia de su climatización. El objetivo es diseñar y desarrollar un servidor IoT edge-computing que esté funcionando 24x7 durante un año e integre datos de temperatura y humedad en numerosos puntos de la vivienda, las condiciones climáticas exteriores, el estado de puertas, ventanas y persianas, el estado y consumo de las máquinas de climatización y el consumo global de electricidad. El servidor IoT tiene que ser robusto, fiable y con una arquitectura modular escalable y reutilizable para implantar en otras viviendas inteligentes.

La finalidad de este proyecto es diseñar e implementar un servidor IoT que integra la telemetría de varios sistemas y monitoriza parámetros varios de una vivienda inteligente: datos detallados de temperatura y humedad del aire, estación meteorológica, estado de puertas, ventanas y persianas, ocupación de la vivienda, estado máquinas climatización, consumo eléctrico y producción energía solar. Los datos son recopilados de forma eficiente y fiable y son organizados y presentados de forma accesible y amigable para facilitar la monitorización y análisis del comportamiento climático de dicha vivienda.

El objetivo principal es el diseño de un servidor de recogida de datos masivos de sensores para facilitar el análisis por parte de los investigadores del proyecto Coolchamber.

Se ha diseñado un sistema IoT a medida para las necesidades del proyecto CoolChamber para tener datos de sensorización masivos durante varios meses de medidas. Un requisito importante ha sido generar datos de calidad y en cantidad, generando ficheros exportables intuitivos para los investigadores, es decir simplificando su generación y gestión. Se ha escogido el formato CSV por la facilidad de extraer datos y poder procesarlos en otros programas y herramientas.

Mediante una página web, los datos obtenidos del servidor son fácilmente consultables y exportables a un formato estándar de una tabla tabulada tipo .csv. Este formato es la solución más simple para, sin conocimientos informáticos, los investigadores de la UGR, UPCT y la empresa puedan descargar el histórico de datos y analizarlos en Excel.



El servidor, denominado **Servidor CoolChamber**, se encarga de, en una vivienda:

- Registrar datos de sensores IoT de temperatura y humedad distribuidos por la vivienda.
- Registrar datos de una estación meteorológica ubicada en el tejado.
- Registrar datos de la centralita domótica.
- Procesar todos esos datos en BBDD para descarga y visualización en una página web de fácil acceso y uso para los investigadores de la UGR y UPCT.

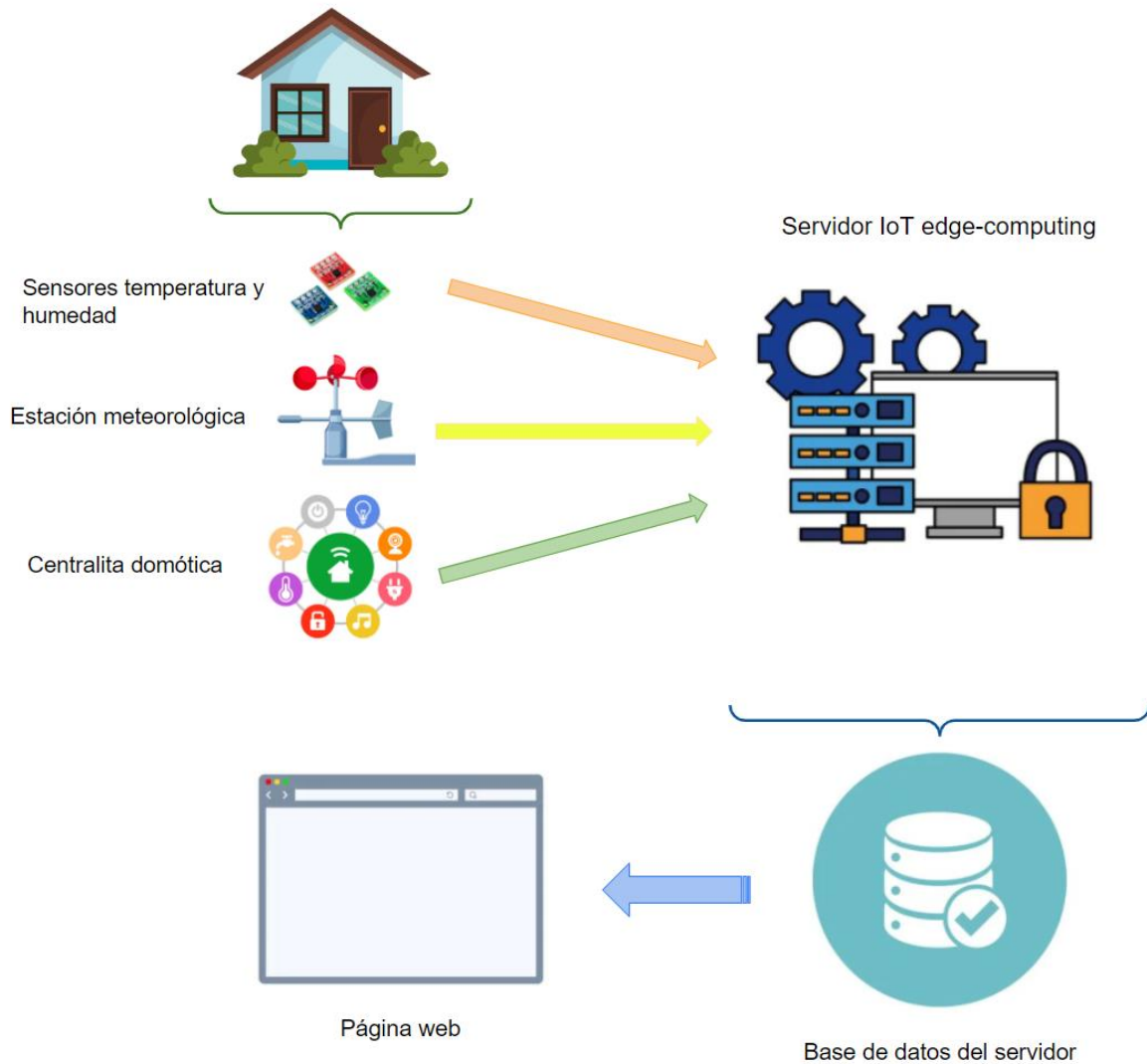


Figura 1. Esquema simplificado servidor IoT de almacenamiento y visualización de datos

## 1.2 Herramientas software y elementos usados

Para el desarrollo de este proyecto se han utilizado los siguientes programas, entornos de desarrollo y librerías:

- \* DB Browser for SQLite [2]. Herramienta de código abierto, visual y de alta calidad para crear, diseñar y editar archivos de Base de Datos compatibles con SQLite.
- \* GIT [3]. Sistema de control de versiones distribuido de código abierto y gratuito, diseñado para manejar proyectos con velocidad y eficiencia.
- \* Modbus v3.10 [4]. Protocolo de comunicación abierto, utilizado para transmitir información a través de redes en serie entre dispositivos electrónicos.
- \* Mosquitto v3.1.1 [5]. Programa que permite iniciar un servicio MQTT y monitorizar las tramas que aparecen.
- \* NodeJS LTS [6]. Framework del lenguaje de programación JavaScript especializado en la creación de Backends. Utilizado para desarrollar el servidor que obtiene, procesa, filtra y guarda los datos en la Base de Datos para su posterior descarga, además del diseño del Frontend. Librerías y dependencias utilizadas:

- Bootstrap	v5.1.3
- Bootstrap-icons	v1.8.1
- Csv-express	v1.2.2
- Dotenv	v10.0.0
- Express	v4.17.1
- Mqtt	v4.2.8
- Telegram-bot-api	v0.56.0
- Request	v2.88.2
- Sqlite3	v5.0.2

- \* Npm v8.1.0 [7]. Sistema de gestión de paquetes por defecto para NodeJS, un entorno de ejecución para JavaScript.
- \* Postman v9.0.9 [8]. Software para realizar peticiones al servidor y servicios API REST.
- \* Visual Studio Code [9]. Editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. Este es el entorno de desarrollo en el que se realizan los programas.
- \* Ubuntu 18.04.6 LTS [10]. Distribución de Linux utilizada para desarrollar el servidor.

## 1.3 Estructura y organización del proyecto

A lo largo de este proyecto vamos a estudiar diferentes conceptos, protocolos, tecnologías y software para la monitorización de una vivienda inteligente.

El proyecto se divide en varias fases:

- 1) Análisis de requisitos, especificación y definición de la arquitectura del sistema.
- 2) Desarrollo continuo aplicando metodologías ágiles:
  - a) Implementación de prototipo de Backend y BBDD para recepción masiva de datos de temperatura y humedad del aire.
  - b) Implementación sistema de alerta y avisos de mantenimiento mediante chatbot.
  - c) Configuración estación meteorológica e integración de datos.
  - d) Integración de datos relevantes de centralita KNX.
  - e) Estudio y pruebas de sensores y sistema para monitorizar apertura y cierre de puertas y ventanas. Integración en servidor.
  - f) Desarrollo de Frontend de visualización y descarga de datos a medida de los usuarios.
- 3) Pruebas sistemas y mejora continua en vivienda real.
  - Mecanismos de robustez, fiabilidad y tolerancia a fallos.
- 4) Revisión de la documentación -código, especificación integración, manuales instalación y uso-.

# Capítulo 2. Tecnologías empleadas

## 2.1 Sistemas instalados que se integran en el servidor

- Router Wifi → conexión internet.
- Sensores T/HR → con un ID de sensor y un código según su ubicación.
- Gateway sensores → 2-3 unidades para asegurar cobertura y redundancia de datos sensores.
- Estación meteorológica → en el tejado de la casa.
- Centralita domótica → con datos accesibles consumo eléctrico, estado persianas y control On/Off y temperatura consigna climatización.
- Placas solares → con acceso a datos visualización producción electricidad y consumo global electricidad de la casa.

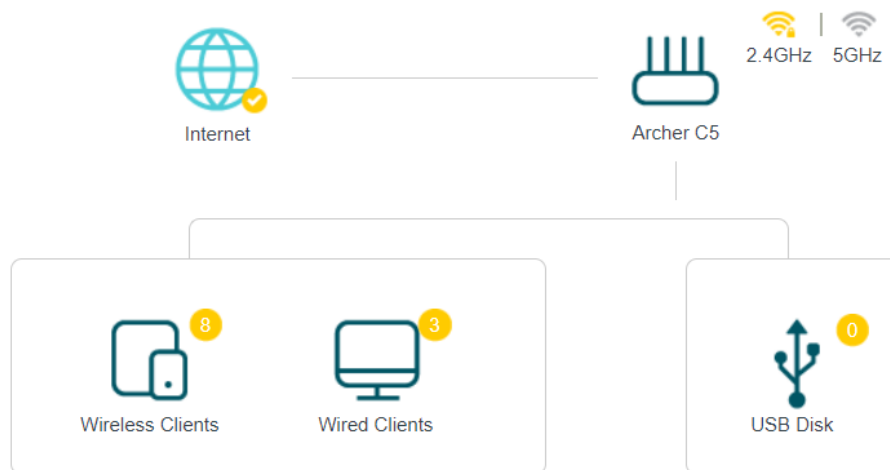
### Enlaces y cuentas de acceso a servicios

- \* Acceso red Wifi vivienda → a través de Smartphone u ordenador
- \* Acceso remoto server Coolchamber → a través de VNC  
Se especifica la dirección asignada y puerto: 194.169.184.32:1997
- \* Acceso al server descarga CSV → Frontend de visualización  
<http://194.169.184.32:27190/>
- \* Acceso a centralita KNX domótica → para ver configuración  
<http://194.169.184.32:20000>
- \* Enlace web paneles solares → producción placas y consumo global casa  
<https://region01eu5.fusionsolar.huawei.com/>
- \* Acceso datos estación meteorológica  
Estación meteorológica Coolchamber (Instalada 12/1/2022)  
<https://www.wunderground.com/dashboard/pws/ICARTA94>  
Configuración: <https://www.wunderground.com/login>

\* Acceso administración Router

<http://194.169.184.32:3030/>

Accediendo a él podemos ver información detallada de la configuración Wifi y dispositivos conectados que se detallan a continuación en las siguientes figuras. Cabe destacar que se utiliza port forwarding - NAT.



Internet

Internet Status:	Connected
Connection Type:	Dynamic IP
IP Address:	192.168.100.2
DNS Server:	8.8.8.8 8.8.4.4
Gateway:	192.168.100.1

Figura 2. Acceso router - información de internet

Wireless Clients

ID	Name	IP Address	MAC Address
1	ESP-E47842	192.168.1.100	E8-DB-84-E4-78-42
2	android-1c6b5d392498a8fa	192.168.1.106	30-95-87-EB-DC-C1
3	espressif	192.168.1.101	7C-9E-BD-CF-0C-CC
4	espressif	192.168.1.102	7C-9E-BD-CF-0D-34
5	SUN2000-102140103823	192.168.1.108	48-2C-D0-F6-00-B0



Figura 3. Acceso router - clientes comunicación inalámbrica 1

Wireless Clients

ID	Name	IP Address	MAC Address
6	Samsung	192.168.1.109	7C-0A-3F-8B-59-B2
7	Wibeee clima interior	192.168.1.23	BC-FF-4D-E9-BB-C0
8	Wibeee clima envolvente	192.168.1.22	00-1E-C0-2C-5E-03

Figura 4. Acceso router - clientes comunicación inalámbrica 2

Wired Clients

ID	Name	IP Address	MAC Address
1	espressif	192.168.1.103	7C-9E-BD-30-2E-43
2	Coolchamber server	192.168.1.19	00-D8-61-7F-93-3F
3	LogicMachine KNX	192.168.1.20	00-1B-C5-00-47-B8

Figura 5. Acceso router - clientes comunicación por cable

## DHCP Server

IP Version:  IPv4  IPv6

MAC Address: D8-07-B6-95-BB-97

IP Address:

Subnet Mask:

IGMP Snooping:  Enable

Second IP:  Enable

DHCP:  Enable

DHCP Server  DHCP Relay

IP Address Pool:  -

Address Lease Time:  minutes. (1-2880. The default value is 1440.)

Default Gateway:  (Optional)

Default Domain:  (Optional)

Primary DNS:  (Optional)

Secondary DNS:  (Optional)

Figura 6. Acceso router - configuración DHCP

## Virtual Servers

+ Add - Delete

<input type="checkbox"/>	ID	Service Type	External Port	Internal IP	Internal Port	Protocol	Status	Modify
<input type="checkbox"/>	1	SSH	19270	192.168.1.19	22	TCP		
<input type="checkbox"/>	2	HTTP	27190	192.168.1.19	8080	TCP		
<input type="checkbox"/>	3	vnc	1997	192.168.1.19	5901	TCP		
<input type="checkbox"/>	4	HTTP	20000	192.168.1.20	80	TCP		

Figura 7. Acceso router - port forwarding - NAT

## Elementos en la red y su configuración IP

Red 192.168.1.0/24  
DNS 192.168.1.1

### IPs asignadas:

- \* 192.168.1.1 → router - gateway
  - \* 192.168.1.19 → servidor Coolchamber
- Rango DHCP: 192.168.1.100 - 192.168.1.199
- Gateways Coolchamber usan DHCP.
- GW1: 192.168.1.101
  - GW2: 192.168.1.102
  - GW3: 192.168.1.103
- 
- \* 192.168.1.20 → centralita KNX
  - \* 192.168.1.21 → inversor Huawei -producción placas solares y consumo global vivienda-
  - \* 192.168.1.22 → analizador red climatización envolvente -medidor consumo máquina climatización cámara envolvente-
  - \* 192.168.1.23 → analizador red climatización casa -medidor consumo máquina climatización interior de la casa-
  - \* 194.169.184.32 → IP fija

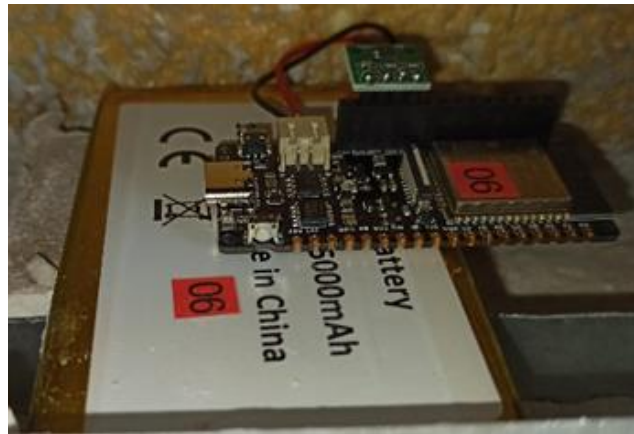
## Sensores IoT instalados

Los principales datos de sensores en el interior de la vivienda son Temperatura y Humedad relativa del aire. Para evaluar y validar el modelo de transferencia de calor de la vivienda y de la envolvente térmica, los investigadores establecieron un requisito de muestreo de temperatura y humedad relativa cada 5 minutos y una duración objetivo de la batería del módulo sensor de hasta 12 meses, con un error en la precisión menor a 0,3 °C y 3% de humedad relativa. Se ha diseñado un módulo sensor ad-hoc formado por un microcontrolador, batería de lipo de alta capacidad y un sensor digital de temperatura y humedad relativa del aire de alta precisión, alta repetibilidad y mínimo error. Se ha escogido un sensor de gama alta de Sensirion SHT35 [11].

El maximizar el tiempo de la batería es para simplificar la carga de trabajo de mantenimiento y operativa del sistema. Las comunicaciones inalámbricas han sido clave para recolectar los datos de los módulos sensores y se ha escogido el estándar Bluetooth Low Energy por tener un alcance de decenas de metros con un mínimo consumo energético.



En la siguiente figura se observa un **módulo sensor** instalado en un punto de la cámara envolvente donde se distingue la batería LIPO (carga de 4000-5000 mAh), microcontrolador con comunicación Bluetooth Low Energy y sensor de temperatura y humedad SHT35. Al ser un entorno interior controlado, se ha podido tener el sensor SHT35 expuesto al aire, lo que mejora la capacidad de medir con mayor fiabilidad la temperatura y humedad del aire con un menor tiempo de respuesta ante cambios. Todos los módulos sensores están etiquetados con un ID para gestionar su configuración, pruebas y lugar de instalación.



*Figura 8. Módulo sensor*

La empresa Bujercal contrató una conexión de internet -típica del hogar- y se instaló un router Wifi en una zona protegida de la vivienda. Hay un PC industrial -robusto para funcionamiento 24x7- con el rol de servidor para recibir y registrar datos de los módulos sensores y de otros elementos del sistema. Para recopilar datos de los módulos sensores se han implementado gateways Bluetooth Low Energy que recopilan los datos de los módulos sensores, mediante Bluetooth, y reenvían los datos al servidor, mediante la red Wifi de la casa. En la siguiente imagen se muestra una fase de las pruebas iniciales de puesta en marcha con el router Wifi, el server y dos Gateways BLE. Se han instalado 3 Gateways para asegurar la cobertura con todos los módulos sensores y que no haya pérdida de datos.

Para validar el modelo de transferencia de calor y comportamiento térmico, se estudió la colocación óptima de módulos sensores y el número de unidades necesarias. Se acordó un total de 44 módulos sensores en puntos clave. En el siguiente plano se muestra el layout 2D de la vivienda con los módulos sensores instalados. Se ha establecido una codificación para facilitar la gestión, instalación y manipulación de datos. El código empieza con el nombre de la habitación (GA= Garaje, LA=Lavadero, CS= Cocina-Salón, etc.). El triángulo azul representa sensores que están dentro de la envolvente -subcódigo W de Wall- pero pueden estar arriba (U=Up) o abajo (D=Down). El icono círculo con una cruz representa los sensores que están accesibles dentro de una habitación. El triángulo rojo representa los sensores que están en la cámara envolvente del techo o de la pared.

Ver Anexo para el listado de códigos.

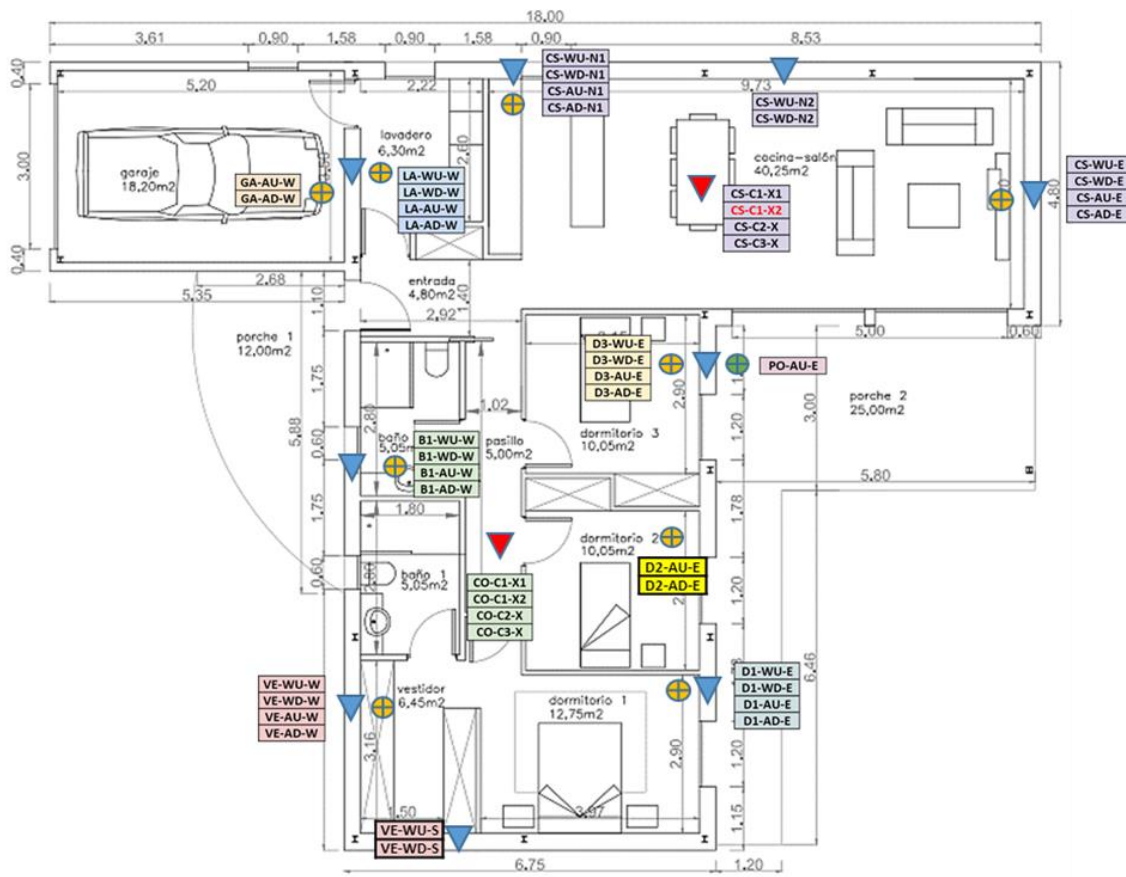


Figura 9. Plano distribución sensores IoT

Los módulos sensores situados en la cámara del suelo son de difícil acceso y ha tenido cierta dificultad su instalación. En las siguientes figuras se muestra una pértiga con una parte final con el módulo sensor, que se introduce por una arqueta para posicionar el módulo en el punto óptimo de medida.



Figura 10. Módulo sensor con pértiga



*Figura 11. Cámara del suelo*

La siguiente figura muestra el colocado del módulo sensor en la cámara del techo



*Figura 12. Colocado del módulo sensor en cámara del techo*

## 2.2 Conceptos básicos del protocolo MQTT

El protocolo Message Queing Telemetry Transport, más conocido como MQTT [12] es un protocolo de comunicación máquina a máquina, basado en TCP/IP.

Las conexiones de MQTT se mantienen abiertas y se reutilizan para cada comunicación, a diferencia de una petición con HTTP, en la que las transmisiones se realizan a través de conexión.

MQTT se define como un servicio de mensajería Publicador/Suscriptor, donde los clientes se conectan a un servidor central llamado Broker.

Los mensajes que se envían a cada cliente se organizan en Topics, de esta forma un cliente puede publicar un mensaje en un determinado Topic sin que afecte al resto. Los demás clientes pueden suscribirse a este Topic publicado, y el Broker se encargará de hacerle llegar los mensajes suscritos bajo ese Topic.

Como hemos mencionado, los clientes inician una conexión TCP/IP que se mantiene abierta hasta que el cliente decide desconectarse con el servidor o Broker, que conserva un registro de los clientes conectados. Normalmente MQTT trabaja bajo el puerto 1883, pero si funciona sobre TLS empleará el puerto 8883.

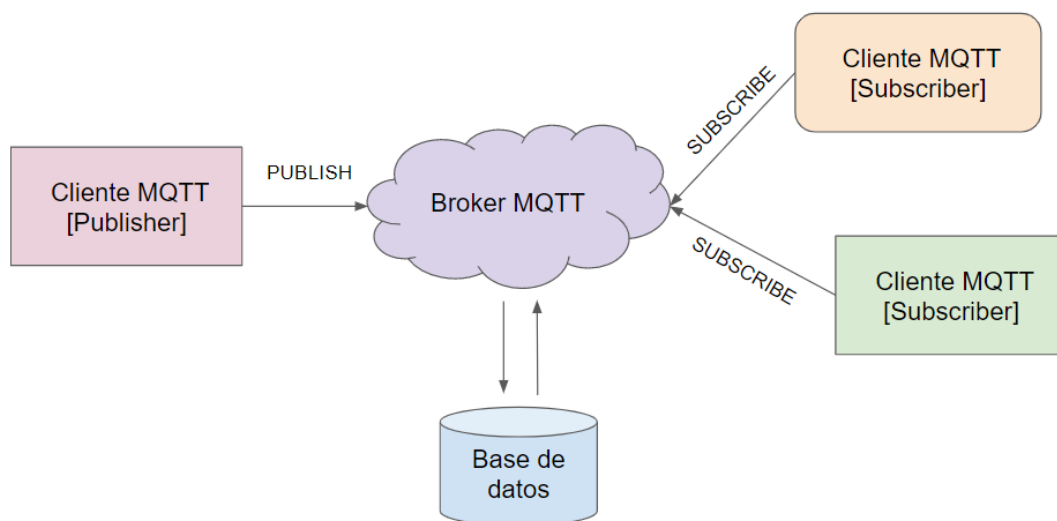


Figura 13. Esquema general MQTT

Como vemos en el esquema general de MQTT, tenemos el Broker encargado de distribuir la información a través de diferentes Topics a los clientes conectados -suscritos- a estos.

El Topic se define como el nombre del mensaje, en él se suscriben o publican los clientes.

Los clientes pueden ser Subscribers -se suscriben a un Topic para recibir los mensajes que se publican en él- o Publishers -envían la información bajo cierto Topic al Broker y este la distribuye a los clientes suscritos a él-

También debemos tener en cuenta la QoS -calidad de servicio- pues cada conexión específica qué calidad de servicio necesita al Broker. Ésta tiene un valor de 0, 1 o 2.

- El valor 0 precisa una vez y sólo una vez, sin necesidad de obtener ACK.
- El valor 1 precisa al menos una vez, de esta forma el mensaje es enviado varias veces hasta recibir un ACK, pudiendo llegar los mensajes duplicados o más tarde.
- El valor 2 precisa una vez exactamente, así tanto el publicador como el suscriptor se aseguran de recibir solamente una copia del mensaje. Esto se define como entrega asegurada.

MQTT utiliza primordialmente los siguientes mensajes:

- ✳ Mensaje CONNECT: el cliente envía este mensaje con la información necesaria de éste -nombre de usuario, id...-
- ✳ Mensaje CONNACK: el Broker responde al cliente con este mensaje, el cual lleva la resolución de la conexión -aceptada, rechazada...-
- ✳ Mensaje PUBLISH: el cliente emplea este mensaje para enviar el Topic y Payload del mensaje en cuestión.
- ✳ Mensaje SUBSCRIBE/UNSUBSCRIBE: mensajes para suscribirse o desuscribirse al Topic.
- ✳ Mensaje SUBACK/UNSUBACK: el Broker responde con uno de estos mensajes dependiendo de cuál es la acción que se desea.
- ✳ Mensaje PINGREQ/PINGRESP: mensajes para asegurar la conexión. Los clientes se aseguran de ello mandándolos periódicamente y el Broker los responde.
- ✳ Mensaje DISCONNECT: cuando el cliente desea finalizar la conexión envía este mensaje.

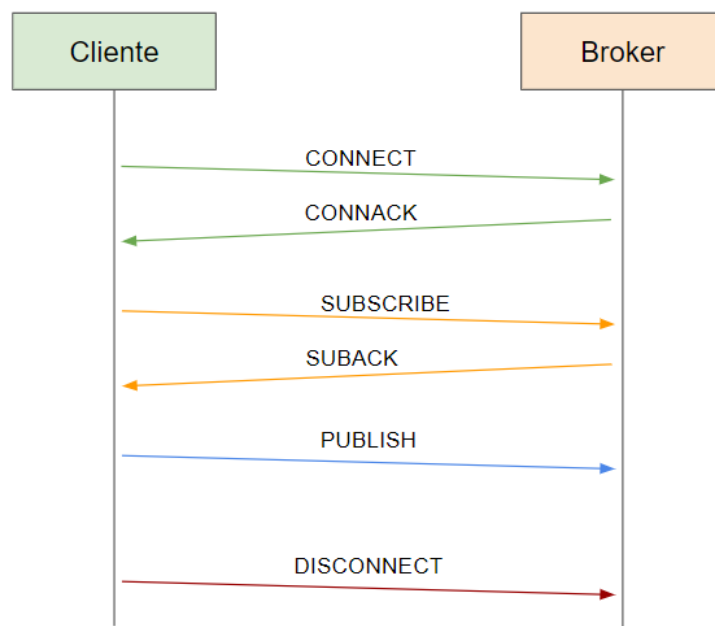


Figura 14. Cronograma funcionamiento MQTT

## 2.3 Conceptos básicos del protocolo Modbus

Modbus es un protocolo de comunicación Maestro/Esclavo, usado para transmitir información entre dispositivos electrónicos por medio de redes en serie.

El protocolo Modbus permite que se pueda comunicar un equipo de control de estado de puertas, persianas, temperatura, humedad, presencia, etc -como es nuestro propósito- con un sistema de recolección de datos u ordenador a través de comunicación serie.

Es destacable su uso para comunicarse con dispositivos IoT y monitorizar y programar otros dispositivos, así como para aplicaciones RTU en las que se precisa la comunicación inalámbrica.

El dispositivo que actúa como Maestro es quien pide la información, y el dispositivo Esclavo es quien la proporciona. De esta forma, el Esclavo no puede facilitar la información hasta que el Maestro no se la pide.

Normalmente en una red Modbus tenemos un Maestro y varios dispositivos Esclavos enviando información, cada uno con una dirección única.

Para empezar a utilizar Modbus primero nos debemos descargar un software de Modbus [13].

En nuestro caso tenemos la centralita KNX que actúa como Esclavo devolviendo la información de las variables de la pasarela sobre la dirección IP 192.168.1.20.

Para leer las variables desde un terminal se utiliza el siguiente comando

```
modpoll -c C -r R -m tcp 192.168.1.20
```

Siendo -r desde donde queremos leer -respecto a direcciones-, dirección R y -c hasta donde queremos leer, desde dirección R leemos C direcciones, utilizando conexión TCP sobre la IP precisa.

Este comando sirve para comprobar qué datos se leen en las direcciones solicitadas para posteriormente crear un programa que las pueda leer sin riesgo a equivocarse de dirección.

La siguiente figura muestra un ejemplo de uso del comando modpoll.

```
coolchamber@coolchamber-MS-9A65:~/Escritorio/servidor/servidor_actual$ modpoll -
c 9 -r 11 -m tcp 192.168.1.20
modpoll 3.10 - FieldTalk(tm) Modbus(R) Master Simulator
Copyright (c) 2002-2021 proconX Pty Ltd
Visit https://www.modbusdriver.com for Modbus libraries and tools.

Protocol configuration: MODBUS/TCP, FC3
Slave configuration...: address = 1, start reference = 11, count = 9
Communication.....: 192.168.1.20, port 502, t/o 1.00 s, poll rate 1000 ms
Data type.....: 16-bit register, output (holding) register table

-- Polling slave... (Ctrl-C to stop)
[11]: 100
[12]: 255
[13]: 255
[14]: 255
[15]: 255
[16]: 255
[17]: 255
[18]: 255
[19]: 255
Polling slave... (Ctrl-C to stop)
```

Figura 15. Ejemplo de uso comando modpoll

## 2.4 Conceptos básicos sobre el servicio REST

REST -REpresentational State Transfer- es una interfaz con la que podemos conectar varios sistemas basados en el protocolo HTTP -utilizado para páginas web- y obtener datos o generar operaciones sobre los datos en todos los formatos posibles, principalmente JSON y XML.

Un servicio REST no tiene estado, esto quiere decir que cada petición HTTP creada tiene toda la información necesaria para ser ejecutada. De esta forma tanto cliente como servidor no necesitan recordar nada para que se realice, es decir, el servicio pierde todos los datos entre una petición y otra.

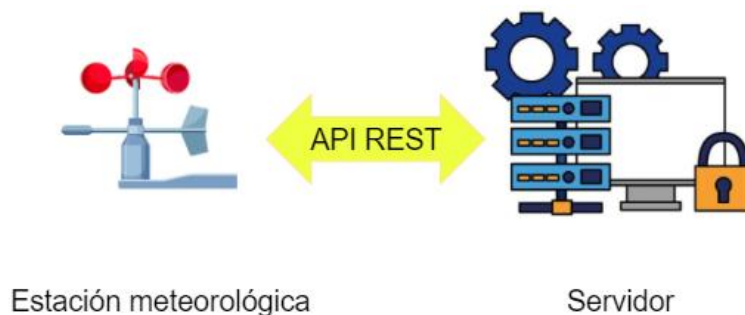
El cliente es quien mantiene el estado de la petición, si queremos que el servicio REST nos recuerde en cada petición debemos pasarle un nombre de usuario, contraseña o token. De esta forma podemos separar el cliente del servidor, proporcionándonos escalabilidad.

Un servicio REST posee cuatro operaciones más importantes:

- POST: crear un nuevo recurso.
- GET: leer y consultar un recurso.
- PUT: modificar/editar cierto recurso.
- DELETE: eliminar un recurso.

Cabe destacar que los objetos siempre se manipulan mediante URI -Uniform Resource Identifier- el cual actúa como identificador único de cada recurso.

En nuestro caso, nuestro servidor puede obtener los datos de la estación meteorológica ubicada en el tejado de la vivienda mediante peticiones a la API REST.



*Figura 16. Esquema obtención datos estación meteorológica*

A continuación se explica cómo se obtienen los datos.

Para obtener los datos de la estación meteorológica, se debe acceder a la página <https://www.wunderground.com/login> como mencionamos en el Capítulo 2.

Una vez accedemos a la página, seleccionamos My Profile → Member Settings → My devices

The screenshot shows the Wunderground website interface. At the top, there's a navigation bar with 'Sensor Network', 'Maps & Radar', 'Severe Weather', 'News & Blogs', and 'Mobile Apps'. Below that, a 'Member Settings' section is active, with sub-tabs for 'EMAIL & PASSWORD', 'HOME & FAVORITES', 'MY DEVICES', and 'API KEYS'. The 'MY DEVICES' tab is selected, showing a table of 3 devices. The table has columns for Name, Location, Status, ID, Key, Type, and Manage. The devices listed are:
 

- Raspberry Sense Hat**: Location: Cartagena (Cartagena), ES; Status: Offline; ID: ICARTA18; Key: 7En92eUf; Type: PWS.
- WH6000**: Location: Cartagena (Cartagena Casco), ES; Status: Online; ID: ICARTA58; Key: uKMsSeRq; Type: PWS.
- WH3000SE PRO Pruebas**: Location: Cartagena (Barriada de Villalba), ES; Status: Online; ID: ICARTA94; Key: BtBJLa3u; Type: PWS.

Figura 17. Acceso página estación meteorológica

Seleccionamos WH3000SE PRO Pruebas → ICARTA94

Desde ahí podemos visualizar los datos de nuestra estación meteorológica.

#### Weather History for ICARTA94

The screenshot shows the 'Weather History for ICARTA94' page. It includes a navigation bar with 'Daily Mode', 'February', '16', '2022', and a 'View' button. Below the navigation bar, there's a 'Summary' section for 'February 16, 2022'. The summary table shows:
 

	High	Low	Average
Temperature	22.1 °C	2.4 °C	11.0 °C
Dew Point	3.7 °C	-6.8 °C	-0.8 °C
Humidity	81 %	16 %	48 %
Precipitation	0.00 mm	--	--

 To the right of this table, there's another table with columns for 'High', 'Low', and 'Average' for various weather metrics:
 

	High	Low	Average
Wind Speed	14.3 km/h	0.0 km/h	3.1 km/h
Wind Gust	16.6 km/h	--	4.0 km/h
Wind Direction	--	--	East
Pressure	1,027.23 hPa	1,022.52 hPa	--

 Below the summary, there's a 'Graph' and 'Table' tab. The 'Table' tab is selected, showing a detailed table for 'February 16, 2022' with columns for Time, Temperature, Dew Point, Humidity, Wind, Speed, Gust, Pressure, Precip. Rate, Precip. Accum., UV, and Solar.
 

Time	Temperature	Dew Point	Humidity	Wind	Speed	Gust	Pressure	Precip. Rate	Precip. Accum.	UV	Solar
12:04 AM	6.3 °C	1.7 °C	73 %	NNW	1.2 km/h	1.5 km/h	1,026.72 hPa	0.00 mm	0.00 mm	0	0 w/m <sup>2</sup>
12:09 AM	5.8 °C	1.5 °C	74 %	WNW	0.2 km/h	0.6 km/h	1,026.92 hPa	0.00 mm	0.00 mm	0	0 w/m <sup>2</sup>
12:14 AM	5.4 °C	1.3 °C	75 %	WSW	0.0 km/h	0.0 km/h	1,026.72 hPa	0.00 mm	0.00 mm	0	0 w/m <sup>2</sup>
12:19 AM	5.2 °C	1.3 °C	76 %	NE	0.0 km/h	0.0 km/h	1,026.72 hPa	0.00 mm	0.00 mm	0	0 w/m <sup>2</sup>
12:24 AM	5.0 °C	1.2 °C	77 %	East	0.0 km/h	0.0 km/h	1,026.72 hPa	0.00 mm	0.00 mm	0	0 w/m <sup>2</sup>

Figura 18. Visualización datos estación meteorológica



Seleccionando My Profile → Member Settings → API KEYS obtenemos la clave de aplicación para obtener los datos a través de un servicio REST.

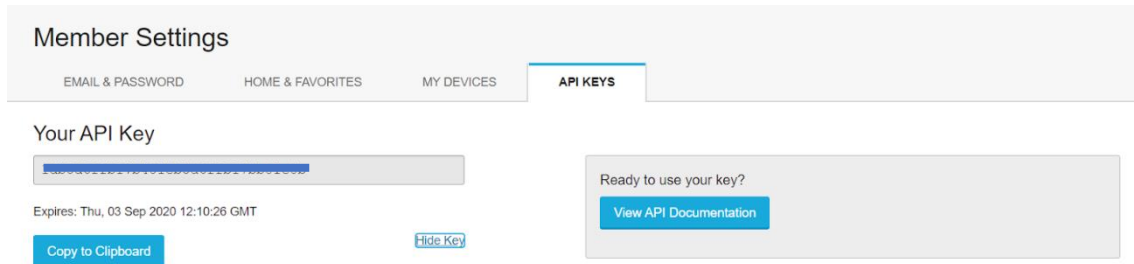


Figura 19. Obtención API KEY estación meteorológica

A modo de ejemplo y para visualizar los datos que vuelca la estación meteorológica utilizamos Postman.

Postman es una herramienta que se utiliza para el testing de API REST. Gracias a esta herramienta, además de testear, consumir y depurar API REST, podemos monitorizarlas, escribir pruebas automatizadas para ellas, documentarlas, simularlas, etc.

Gracias a la documentación proporcionada en la página [14] podemos consultar al servicio REST y así obtener los datos que necesitamos.

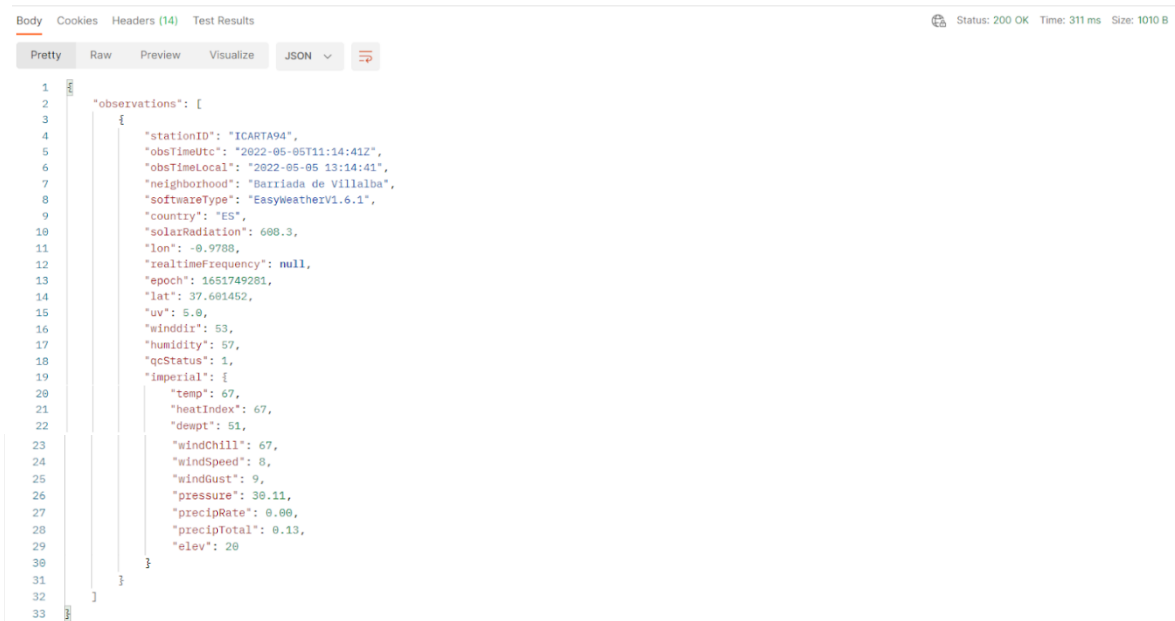
De esta forma, en Postman debemos ejecutar e incorporar los siguientes datos haciendo uso del método GET:

***`https://api.weather.com/v2/pws/observations/current?stationId=[nuestra_estacion]&format=json&units=e&apiKey=[nuestraApiKey]`***



Figura 20. Consulta a Postman

Para la visualización de los datos de nuestra estación pulsamos sobre el botón Send y observamos el resultado de nuestra petición:



```
1  "observations": [  
2  
3    {  
4      "stationID": "ICARTA94",  
5      "obsTimeUtc": "2022-05-05T11:14:41Z",  
6      "obsTimeLocal": "2022-05-05 13:14:41",  
7      "neighborhood": "Barriada de Villalba",  
8      "softwareType": "EasyweatherV1.6.1",  
9      "country": "ES",  
10     "solarRadiation": 688.3,  
11     "lon": -0.9788,  
12     "realtimeFrequency": null,  
13     "epoch": 1651749281,  
14     "lat": 37.681452,  
15     "uv": 5.0,  
16     "winddir": 53,  
17     "humidity": 57,  
18     "qcStatus": 1,  
19     "imperial": {  
20       "temp": 67,  
21       "heatIndex": 67,  
22       "dewpt": 51,  
23       "windChill": 67,  
24       "windSpeed": 8,  
25       "windGust": 9,  
26       "pressure": 38.11,  
27       "precipRate": 0.00,  
28       "precipTotal": 0.13,  
29       "elev": 20  
30     }  
31   }  
32 ]  
33
```

Figura 21. Respuesta de Postman

Aquí tenemos toda la información volcada por nuestra estación meteorológica.

Podemos visualizar el estado de la petición → Status: 200 OK.

Este código de respuesta 200 OK indica que la solicitud ha tenido éxito y, al utilizar el método de solicitud GET, significa que el recurso ha sido recuperado y el mensaje se transmite en el body como muestra en la figura 21.

En el siguiente capítulo se explica detalladamente el proceso de obtención de los datos de la estación meteorológica mediante peticiones a la API REST, así como la obtención de los datos de los sensores IoT y de la centralita KNX.

# Capítulo 3. Diseño e implementación de la arquitectura del servidor

## 3.1 Arquitectura global del servidor

Para el desarrollo de esta arquitectura se han utilizado como elementos principales:

- Programa Mosquitto, Broker al que los Gateways IoT envían la información de los sensores de Temperatura y Humedad relativa.
- Backend Servidor, desarrollado con el framework de JavaScript NodeJS, encargado de recibir datos de diversas formas. También aloja un servicio REST.
- Base de Datos SQLite para el almacenamiento de los datos.
- Frontend de visualización de datos para el usuario, es la parte que se ejecuta en el dispositivo del cliente donde es posible la descarga de los datos.
- Alta y configuración de sensores IoT activos.

La principal funcionalidad del servidor es la recepción y almacenamiento de datos de los sensores de Temperatura y Humedad relativa reenviados por los Gateways, de la estación meteorológica que posee la vivienda en el tejado y de la centralita domótica.

Aparte de recoger la información, el servidor la filtra y procesa para guardarla en la Base de Datos.

Además tenemos creado un Bot de Telegram conectado al servidor que envía mensajes de información/fallo de los sensores IoT y de la presencia en la vivienda.

A su vez, el servidor gestiona la visualización de una página web para la descarga de todos los datos recogidos sobre la vivienda para los investigadores de forma sencilla.

Cabe destacar que podemos dar de alta y configurar los sensores IoT para marcarlos como activos o no y que así el servidor sólo reciba mensajes de los sensores activos que se han considerado. El proceso de dar de alta es el siguiente: el servidor posee una tabla en la Base de Datos con la información de todos los sensores y una columna de “activo”, basta con marcarlo como Y si queremos que esté activo este sensor o como N si no lo deseamos. El programa del servidor posee una función para detectar los sensores marcados como activos y así sólo recibir los datos de estos.

A continuación se muestra la figura 22 con la arquitectura global del sistema.

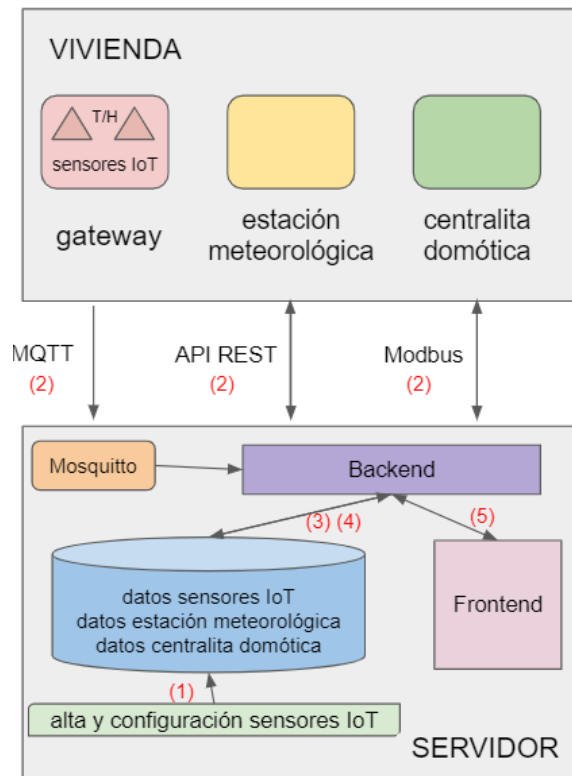


Figura 22. Arquitectura del sistema implementado

1. En primer lugar, se comprueban y marcan como activas en la Base de Datos del servidor las IDs de los sensores IoT que van a enviar la información de Temperatura y Humedad.
2. Comienzan a transmitir datos cada módulo de la vivienda de una forma distinta:
  - Los Gateways mandan la información captada de cada sensor IoT al servidor cada 10 minutos por medio de MQTT. Simultáneamente lo recibe Mosquitto, el cual está suscrito a los topics necesarios y lo envía al Backend.
  - El Backend solicita los datos de la estación meteorológica mediante llamadas API REST cada 6 minutos.
  - El Backend obtiene los datos volcados sobre los registros de la centralita domótica con Modbus cada 4 minutos y cada 30 segundos para obtener ciertas variables que se especificarán más adelante.
3. Una vez obtenido, filtrado y procesado cada dato, el Backend lo guarda en la Base de Datos a la que está conectado, quedando accesible para acciones futuras.
4. La Base de Datos almacena la información y el Backend puede pedirle los datos para mandarlos al Frontend.
5. Cuando un usuario/investigador desea descargar datos, ya sean de los sensores IoT, de la estación meteorológica o de la centralita domótica, el Frontend se encarga de comunicarse con el Backend para que éste obtenga los datos solicitados de la Base de Datos y se los pase al Frontend para así poder descargar el CSV de ellos.

## 3.2 Monitorización Gateway-Servidor

Los Gateways captan la información enviada de los sensores IoT de Temperatura y Humedad relativa por medio de Bluetooth Low Energy y la reenvían mediante MQTT a nuestro servidor, el cual posee Mosquitto. MQTT es un protocolo de envío de mensajes en el que un cliente puede suscribirse a un topic -mensaje- que tiene el Bróker -servidor- y publicar información bajo este, de esta forma solo los usuarios suscritos a ese topic podrán recibir la información.

Para la recepción de los datos que envían los Gateways por MQTT es necesario que el servidor esté conectado a Mosquitto y suscrito bajo los topics que hemos creado.

Cuando se recibe un mensaje por MQTT, se comprueba bajo qué topic ha sido enviado y:

- Si el topic es tyh → obtiene la información y la inserta en la tabla tyh de la BBDD.
- Si el topic es admin → se está comprobando si hay comunicación entre GW-server.

Los Gateways, además de enviar la información de los sensores IoT bajo el topic “tyh”, cada ciertas muestras, envían por MQTT bajo el topic “admin” su nombre de Gateway al servidor como mensaje de estado. Si el servidor no responde al mensaje enviado por los Gateways, estos se reiniciarán. De esta forma el servidor garantiza la comunicación entre él y el Gateway, asegurando el envío y recepción de los datos.

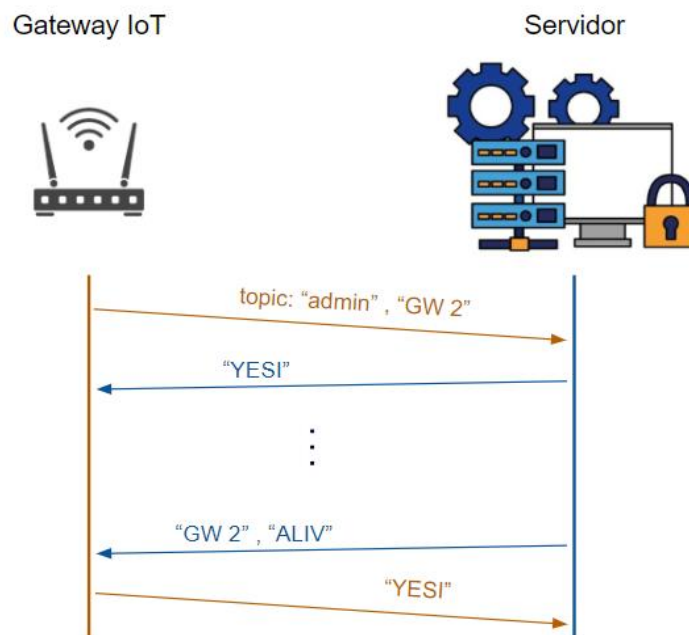


Figura 23. Cronograma envío de mensajes monitorización Gw-Servidor

En la figura 23 podemos observar cómo el Gateway envía su nombre bajo el topic admin y el servidor le responde.

Pasado un tiempo, si el servidor no encuentra tramas enviadas por un Gateway en el momento de procesado, el servidor enviará un mensaje a ese Gateway preguntando si sigue “con vida” o se ha reiniciado/apagado y este le responderá si está encendido y funcionando.

```
Received ID: 33, from Gateway: 3
Received ID: 33, from Gateway: 2
message from admin
On topic: admin : <Buffer 47 57 32>
Received ID: 33, from Gateway: 1
Received ID: 20, from Gateway: 3
Received ID: 20, from Gateway: 1
Received ID: 20, from Gateway: 2
Received ID: 43, from Gateway: 3
Received ID: 43, from Gateway: 1
Received ID: 43, from Gateway: 2
Received ID: 17, from Gateway: 3
Received ID: 17, from Gateway: 1
Received ID: 17, from Gateway: 2
Received ID: 27, from Gateway: 3
Received ID: 27, from Gateway: 2
Received ID: 27, from Gateway: 1
message from admin
On topic: admin : <Buffer 47 57 31>
```

*Figura 24. Ejemplo envío real Gw-servidor*

En la figura 24 vemos un ejemplo de envío de mensajes del Gateway al servidor.

Cuando se recibe un mensaje bajo el topic admin, este es mostrado en el terminal donde se está ejecutando el código del servidor, seguido del mensaje que el gateway envía.

Comprobamos en la figura 24 que hemos recibido dos mensajes de dos Gateways distintos, pues el último número que aparece en el mensaje <Buffer 47 57 3x> corresponde con la ID del Gateway, habiendo recibido así dos mensajes de estado: primero del Gateway 2 y después del Gateway 1.

### 3.3 Recepción y procesado de datos temperatura y humedad

Primero debemos hablar sobre el funcionamiento de los sensores de temperatura y humedad relativa.

Disponemos de 44 sensores IoT ubicados en partes superiores e inferiores de la vivienda como hemos comentado en el capítulo anterior.

Los sensores IoT obtienen la temperatura y humedad relativa del ambiente cada 5 minutos. Cada **10 minutos** mandan esa información a los Gateways que la reenviarán al servidor.

Su funcionamiento es el siguiente:

1ª muestra:



2ª muestra:



Pasados 20 minutos, habremos recolectado dos nuevas mediciones que ocuparán las primeras posiciones del buffer -T1, H1 y T2, H2- y junto a ellas se enviarán al gateway las dos mediciones anteriormente enviadas, en las siguientes posiciones -T3, H3 y T4, H4- para aportar redundancia.



Disponemos de 3 Gateways que reciben los datos de los sensores IoT y los mandan al servidor mediante MQTT:

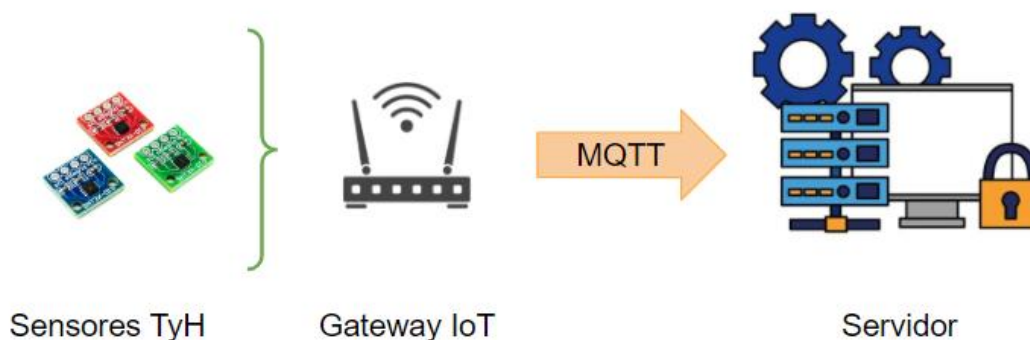


Figura 25. Esquema obtención datos sensores IoT

El funcionamiento del servidor ante la recepción de los datos de los sensores IoT es el siguiente:



Figura 26. Diagrama de flujo recepción datos sensores IoT

- \* En primer lugar, se llama a la función `updateActives()`. Esta función se llama al principio de la ejecución del servidor para que compruebe las IDs activas -sensores activos-, de esta forma el servidor sólo guardará los datos de los sensores activos.
- \* Se obtienen los datos de los sensores enviados por los Gateways. Los Gateways envían al servidor cada **10 minutos** la información en bruto de cada sensor -ID- mediante MQTT.
- \* Esos datos se guardan en la tabla `tyh` de la base de datos para su posterior procesado.
- \* Se procesan los datos siguiendo el esquema desarrollado en la siguiente página y se guardan en la base de datos, tabla `datos`.
- \* Una vez procesados los datos, cada hora se envía un mensaje de información de estado a nuestro dispositivo móvil mediante Bot de Telegram -si todo OK- o cada 15 minutos un mensaje de aviso de fallo:
  - Sensor X no se ha detectado
  - Temperatura anómala en sensor X
  - Batería baja en sensor X



\*Una vez guardados los datos en bruto en la Base de Datos, tabla tyh, se deben procesar los datos de temperatura y humedad en el servidor para poder enviar mensajes por Telegram:

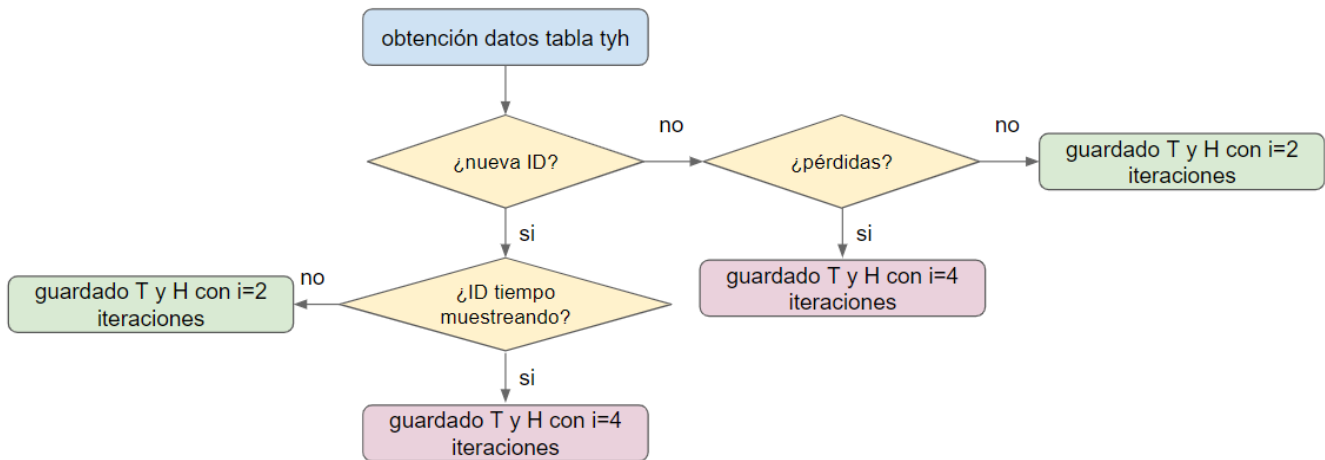


Figura 27. Diagrama de flujo procesado datos sensores IoT

Este programa lo realiza todo en conjunto.

Esta explicación se ha centrado en los datos de Temperatura para simplificar, pero se realiza tanto con los datos de Temperatura como con los de Humedad relativa.

- \* Cada **15 minutos** salta un temporizador que obtiene los datos de la tabla tyh y llama a la función procesado(). Esta comprueba:
- \* Si servidor toma esa ID por primera vez:
  - Guardamos T2 y T1, con fechas: T2 actual y T1 hace 5 minutos, siendo 2 iteraciones al guardar.
  - También es posible que el sensor lleve tiempo muestreando y el servidor lo encuentra por primera vez, en ese caso deberá guardar T1, T2, T3 y T4, por lo que debemos guardar todas las muestras con su respectiva fecha: hace 5 minutos, ahora, hace 10 minutos y hace 15 minutos, siendo 4 iteraciones al guardar.
  - Se deberán hacer varias iteraciones, guardando las respectivas T, H y fechas, junto con los demás datos -ID\_alf, ID\_num, nseq, T, H, batería, timestamp, fecha, hora-.
- \* Si servidor ya conoce ese ID:
  - Se realizan varias comprobaciones para ver las pérdidas, por ejemplo si el número de secuencia pasa de 254 a 0, si el número de secuencia que llega es el siguiente al recibido o si ya lo tengo registrado -enviado por otro gw-.
  - Dependiendo de si se han producido pérdidas o no:
    - Si no hay pérdidas: se realizan 2 iteraciones guardando T2 y T1.
    - Si hay pérdidas: 4 iteraciones, guardando así también T3 y T4, los cuales provienen del muestreo anterior.
  - Se realizarán las correspondientes iteraciones guardando los valores como en el caso anteriormente mencionado.

### 3.4 Recepción de mensajes mediante Bot Telegram

Como ya hemos mencionado antes, hemos añadido un Bot de Telegram que está incorporado en el servidor, el cual posee una ID de chat y un token con el cual podemos enviar mensajes desde el servidor a la aplicación de Telegram.

Este Bot envía mensajes de información/fallo de la siguiente forma:

- sOK → indica los sensores que están funcionando, en total tenemos 44 activos.
- Fail → indica la ID del sensor que está fallando.
- P → 1 si hay presencia en la vivienda, 0 si no.
- TW → Temperature Warning, indica la ID del sensor que dé temperatura anómala, por encima de 50º.
- LB → Low Battery, indica la ID del sensor que tenga batería baja, por debajo de 60%

En la siguiente figura podemos ver un ejemplo del funcionamiento de envío de mensajes:



Figura 28. Ejemplo funcionamiento Bot Telegram

- \* Si no hay ningún fallo, el Bot de Telegram enviará cada hora un mensaje de información de estado.
- \* Si durante el procesado de datos -realizado cada 15 minutos- se produce un fallo, el Bot de Telegram enviará un mensaje de fallo en ese instante y cada 15 minutos mientras el fallo persista.
- \* Si se produce un fallo espontáneo, este se mostrará, pero si no vuelve a fallar el servidor esperará una hora para enviar su mensaje de información de estado.

### 3.5 Recepción y procesado de datos centralita domótica

Tenemos ubicados dispositivos KNX por toda la vivienda y, mediante Modbus se han obtenido los datos que vuelca la centralita domótica.

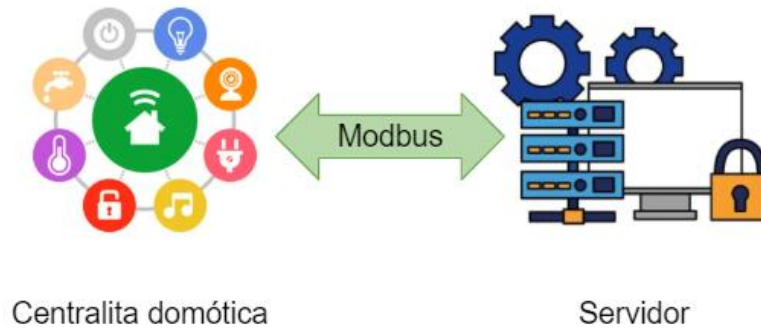


Figura 29. Esquema obtención datos centralita domótica

En el Anexo se muestran las variables de la pasarela que vamos a obtener y procesar.

En la siguiente figura 30 vemos el funcionamiento del programa para la recepción y procesado de datos de la centralita domótica.

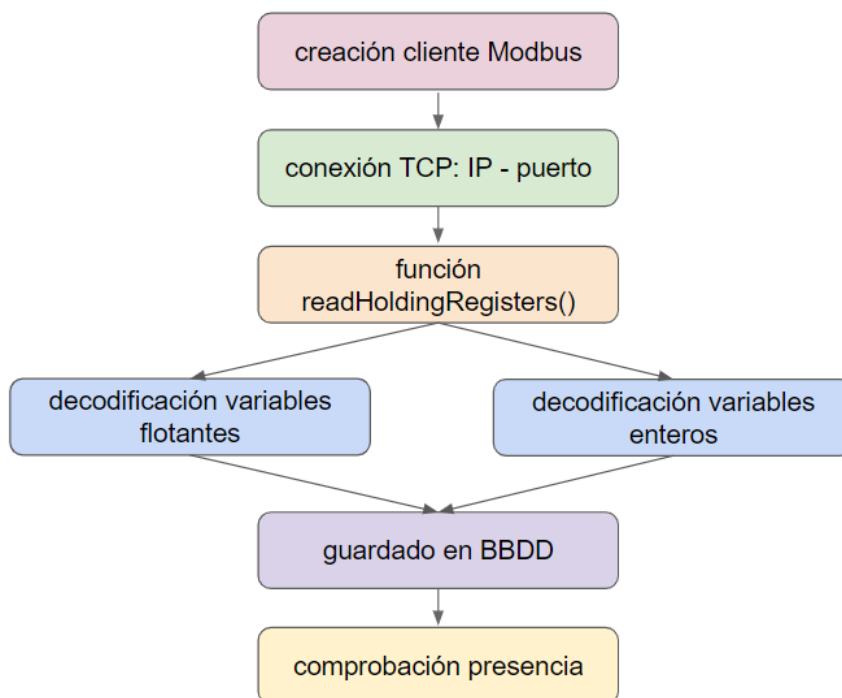


Figura 30. Diagrama de flujo recepción y procesado datos centralita domótica

- \* En el servidor tenemos un script modbus.js que, en primer lugar, crea un cliente ModbusRTU().
  - \* El cliente se conecta a la dirección IP 192.168.1.20 donde se está devolviendo la información, con el puerto 502.
  - \* Cada **4 minutos** se llama a la función readHoldingRegisters() que se encarga de leer las direcciones de los registros, en nuestro caso desde el registro 0 hasta el registro 58.
  - \* Se obtienen los datos de las variables que deseamos, estas son:
    - Estado interior: On-Off.
    - Estado posición persianas.
    - Estados máquina 1: On-Off, modo, velocidad ventilador, temperatura y temperatura entorno.
    - Estados máquina 2: On-Off, modo, velocidad ventilador, modo ventilador, temperatura y temperatura entorno.
    - Recuperador de calor.
    - Salon: luminosidad, presencia, CO2, humedad y temperatura.
    - Dormitorio principal: luminosidad, presencia, CO2, humedad y temperatura.
    - Dormitorio 2: luminosidad y presencia.
    - Dormitorio 3: luminosidad y presencia.
  - \* Se decodifican las variables obtenidas por modbus, ya sean flotantes o enteros.
  - \* Se guardan los datos en la Base de Datos, tabla datosKNX.
  - \* Tenemos una variable Presencia la cual tomará el valor 1 si al menos en un punto de la vivienda hay presencia, 0 si no. Esta variable se mostrará en nuestro dispositivo móvil cada vez que el servidor mande un mensaje de información -cada hora- o de fallo -cada 15 minutos-.
- De la misma forma, tenemos otro script potAct.js que, cada **30 segundos** llama a la función readHoldingRegisters() que lee las direcciones de dos registros y obtiene los datos de potencias activas de las máquinas de climatización 1 y 2 -máquina envolvente y máquina interior vivienda- midiendo el consumo energético de cada máquina. Estos datos sirven para monitorizar los picos instantáneos y se guardarán en la Base de Datos, tabla datosPotAct.

### 3.6 Recepción y procesado de datos estación meteorológica

En el capítulo anterior se ha explicado cómo obtener la información de la API REST, a continuación se muestra el funcionamiento de recepción de datos de la estación meteorológica en el servidor para contrastar los datos ambientales exteriores con los del interior de la vivienda:

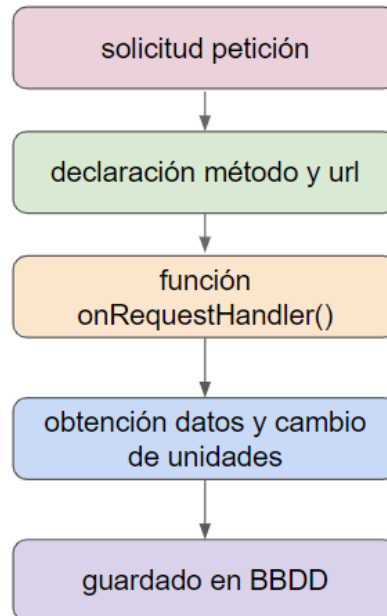


Figura 31. Diagrama de flujo recepción y procesado datos estación meteorológica

- \* Creamos un script `meteo.js` en nuestro servidor que, en primer lugar, realiza una petición a la API REST de la estación meteorológica.
- \* Se declara el método para adquirir los datos -GET- y la url donde se obtiene la información -url utilizada en Postman-
- \* Cada **6 minutos** se llama a la función `onRequestHandler()` que se encarga de leer los datos volcados de la estación meteorológica.
- \* Se obtienen los datos de las variables que deseamos, estas son:
  - Temperature → temperatura
  - Dew Point → punto de rocío
  - Humidity → humedad
  - WindDir → dirección del viento
  - Lat → latitud
  - Lon → longitud

- WindSpeed → velocidad del viento
  - WindGust → ráfaga de viento
  - Pressure → presión
  - PrecipRate → tasa de precipitación
  - PrecipTotal → precipitación total
  - UV → radiación ultravioleta
  - SolarRadiation → radiación solar
- \* Se realiza un cambio en las unidades de ciertas variables obtenidas:
- Temperatura y punto de rocío:  
°F -grados Fahrenheit- → °C -grados Celsius-
  - Velocidad del viento y ráfaga de viento:  
mph -millas por hora- → km/h -kilómetros por hora-
  - Presión:  
in -pulgadas- → hPa -hectopascales-
- \* Se guardan todos los datos en la base de datos, tabla [datosMeteo](#).

## 3.7 Gestión y monitorización del servicio con PM2

PM2 [15] es un administrador de procesos para el tiempo de ejecución de JavaScript NodeJS que nos ayuda a administrar y mantener aplicaciones en línea.

Se ofrece como una CLI -interfaz de línea de comandos- simple e intuitiva, que se puede instalar a través de npm.

Algunas de las características de PM2 son:

- Equilibrio automático de carga de aplicaciones.
- Configuración de aplicaciones declarativas.
- Sistema de implementación.
- Supervisión.

PM2 permite mantener siempre activas las aplicaciones y volverlas a cargar evitando tiempos de inactividad, a su vez facilita tareas comunes de administrador del sistema.

Esto proporciona robustez en nuestro código, pues PM2 se encarga de monitorizar y levantar el servicio si este se cae.

Gracias a PM2, el programa principal server.js se ejecutará nada más iniciar el ordenador.

Si hubiera un corte de luz o se reinicia el servidor, PM2 ejecutará el script de forma autónoma, evitando así la menor pérdida de información.

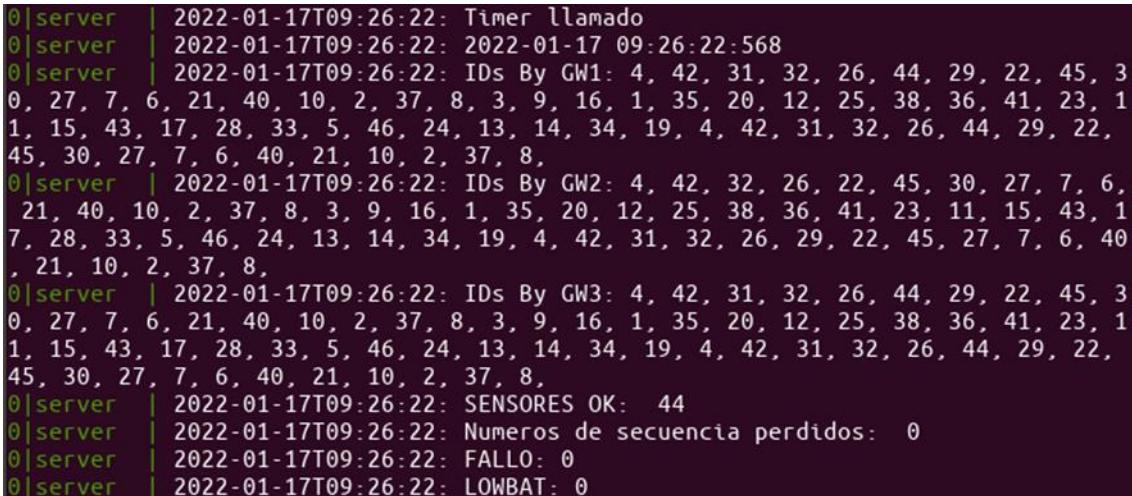
Además genera mensajes sobre cualquier evento anómalo en el sistema.

Una vez instalado PM2 en nuestro ordenador, si queremos iniciar el programa del servidor debemos escribir en la ventana de comandos:

```
> pm2 start server.js --time  
> pm2 logs
```

El primer comando iniciará el script de nuestro servidor y, añadiendo --time podremos ver el tiempo en el cual se producen cambios o se muestra información.

Con el comando pm2 logs se visualiza en la pantalla la información y los procesos del programa iniciado.



```
0|server | 2022-01-17T09:26:22: Timer llamado  
0|server | 2022-01-17T09:26:22: 2022-01-17 09:26:22:568  
0|server | 2022-01-17T09:26:22: IDs By GW1: 4, 42, 31, 32, 26, 44, 29, 22, 45, 3  
0, 27, 7, 6, 21, 40, 10, 2, 37, 8, 3, 9, 16, 1, 35, 20, 12, 25, 38, 36, 41, 23, 1  
1, 15, 43, 17, 28, 33, 5, 46, 24, 13, 14, 34, 19, 4, 42, 31, 32, 26, 44, 29, 22,  
45, 30, 27, 7, 6, 40, 21, 10, 2, 37, 8,  
0|server | 2022-01-17T09:26:22: IDs By GW2: 4, 42, 32, 26, 22, 45, 30, 27, 7, 6,  
21, 40, 10, 2, 37, 8, 3, 9, 16, 1, 35, 20, 12, 25, 38, 36, 41, 23, 11, 15, 43, 1  
7, 28, 33, 5, 46, 24, 13, 14, 34, 19, 4, 42, 31, 32, 26, 29, 22, 45, 27, 7, 6, 40  
, 21, 10, 2, 37, 8,  
0|server | 2022-01-17T09:26:22: IDs By GW3: 4, 42, 31, 32, 26, 44, 29, 22, 45, 3  
0, 27, 7, 6, 21, 40, 10, 2, 37, 8, 3, 9, 16, 1, 35, 20, 12, 25, 38, 36, 41, 23, 1  
1, 15, 43, 17, 28, 33, 5, 46, 24, 13, 14, 34, 19, 4, 42, 31, 32, 26, 44, 29, 22,  
45, 30, 27, 7, 6, 40, 21, 10, 2, 37, 8,  
0|server | 2022-01-17T09:26:22: SENSORES OK: 44  
0|server | 2022-01-17T09:26:22: Numeros de secuencia perdidos: 0  
0|server | 2022-01-17T09:26:22: FALLO: 0  
0|server | 2022-01-17T09:26:22: LOWBAT: 0
```

Figura 32. Visualización servidor con PM2

### 3.8 Diseño Backend y BBDD del servidor

El Backend es la parte dentro del servidor conectada a la Base de Datos, contiene la lógica de la aplicación y no es accesible directamente por los usuarios.

Posee un servicio REST para la inserción y obtención de datos de la Base de Datos SQLite.

A continuación se muestra el diagrama de bloques del funcionamiento del Backend que obtiene y procesa todos los datos e interactúa con el Frontend para poder descargarlos.

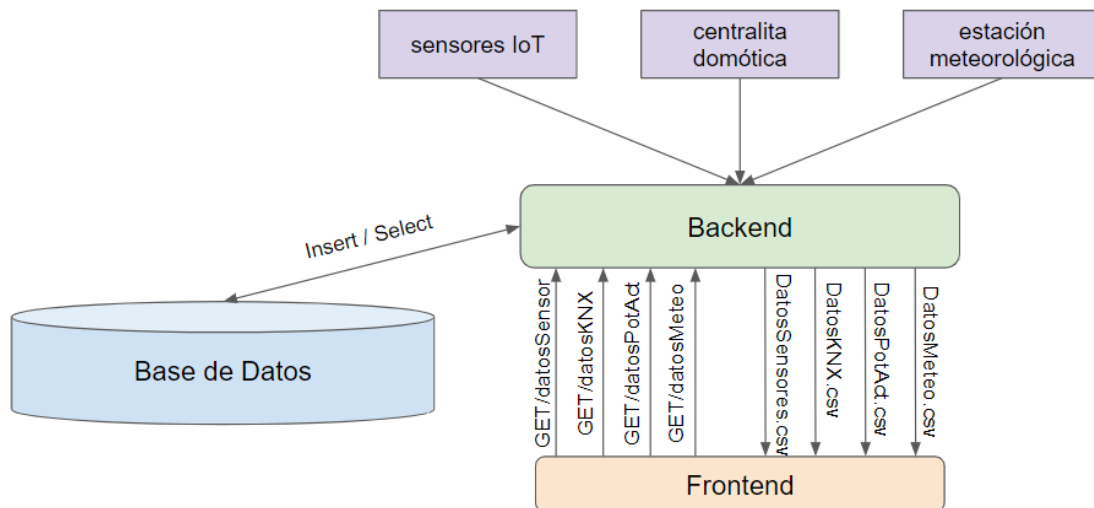


Figura 33. Diagrama de bloques e interconexión módulos del servidor

Nuestro Backend obtiene los siguientes datos:

- Datos en bruto de los sensores IoT de temperatura y humedad relativa enviados por los Gateways a través de MQTT.
- Datos de la centralita domótica, tanto el estado de las persianas, presencia, modos de las máquinas, temperatura, etc, como las potencias activas de las máquinas de climatización mediante Modbus.
- Datos de la estación meteorológica a través de llamadas a la API REST.

Todos estos datos son procesados en el Backend e insertados en la Base de Datos SQLite en su correspondiente tabla gracias a nuestro programa server.js.

Cuando el usuario/investigador desee obtener la información de nuestros datos recogidos, el Frontend mandará la petición al Backend pidiendo los datos necesarios y éste devolverá la información solicitada en la petición al Frontend.

La petición deberá llevar la fecha y hora del intervalo que se desea recibir la información, el Backend realizará una consulta en su Base de Datos para encontrar los datos demandados con el intervalo de fecha y hora indicados en la petición, creará un archivo .csv y devolverá los datos al Frontend que serán descargados.



### 3.9 Diseño Frontend y descarga de archivos

El Frontend es la parte que se ejecuta en el dispositivo del cliente, contiene el código para visualizar la página web, en la que el usuario puede descargarse diversos archivos .csv de los datos que el servidor ha obtenido y procesado, seleccionando la fecha y hora en la que se deseen esos datos.

El diseño del Frontend se basa en el código index.html que incluye el esqueleto de la página principal, con varias imágenes: proyecto Coolchamber y ubicación sensores en el plano de la vivienda, y enlaces de interés para el usuario: ubicación detallada de los sensores con sus códigos y un enlace temporal a la página donde se pueden visualizar los datos de la estación meteorológica.

Además contiene un script interno que se encarga de comprobar las peticiones del usuario: fecha y hora de inicio y fin de la búsqueda que se requiere para así, mediante un botón, seleccionar qué datos desea descargar proporcionando un archivo CSV de éste.

Este script se comunica con el servicio REST del servidor para obtener los datos de la Base de Datos y convertirlos en formato .csv para su descarga.

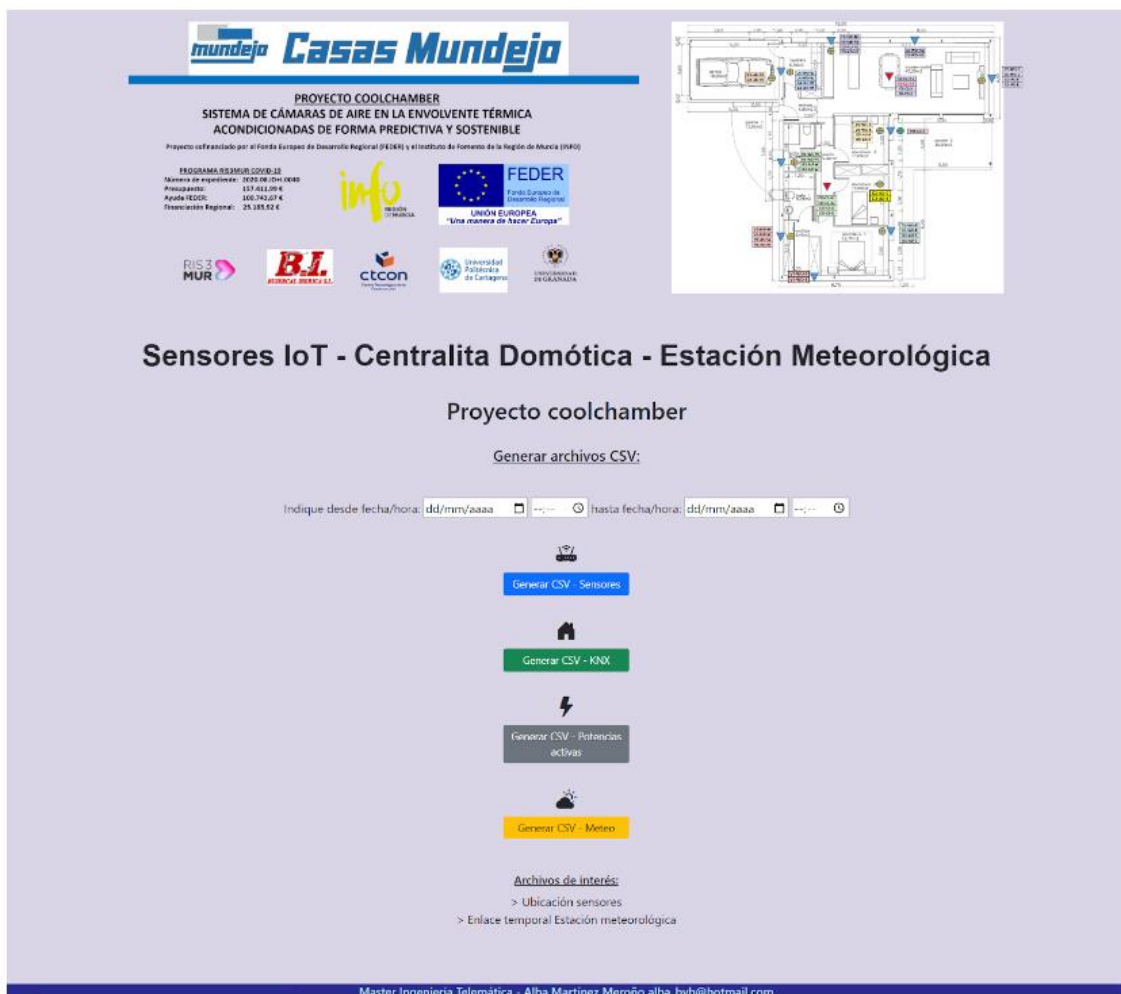


Figura 34. Visualización página web

Podemos observar que la funcionalidad para generar el archivo CSV permite seleccionar la fecha y hora de inicio que se desea, y la fecha y hora de final de la búsqueda.

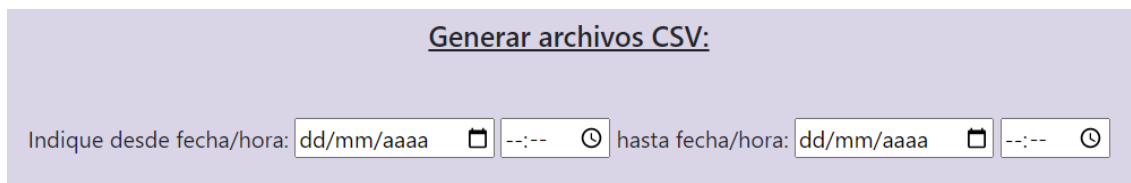


Figura 35. Generación archivo CSV

Para acceder a la página debemos poner en el navegador <http://194.169.184.32:27190/>, de esta forma podremos generar archivos csv y descargar todos los datos que hemos obtenido:

- Datos sensores IoT
- Datos KNX
- Datos potencias activas
- Datos estación meteorológica

Para descargar los datos, como ya hemos dicho, primero debemos seleccionar el rango de fechas y horas que se desean. Una vez seleccionado pulsamos sobre el botón de Generar CSV -de los datos que necesitamos, por ejemplo de los sensores- y a continuación aparecerá otro botón de Descargar CSV el cual permitirá su descarga.



Figura 36. Descarga archivo CSV

El motivo de los dos botones -generar y descargar- es el siguiente:

Al pulsar sobre el botón Generar se llama a una función validar() que comprueba que los rangos de fechas y horas sean correctos, por ejemplo, no se puede seleccionar una fecha de fin que sea anterior a la fecha de inicio -de igual manera con las horas-, y a su vez no se puede dejar ningún campo vacío.

### 3.10 Configuración del acceso al servidor desde Internet

El acceso al servidor, desde el punto de vista del cliente -investigadores-, se realiza como hemos descrito anteriormente, accediendo a la página web de descarga de archivos.

`http://194.169.184.32:27190/`

Desde el punto de vista de los programadores -administradores-, el acceso al servidor se realiza para llevar a cabo servicios de mantenimiento de éste. Para acceder al servidor en sí -ordenador ubicado en la vivienda- debemos utilizar VNC Viewer [16].

VNC Viewer es un programa que permite tomar el control del servidor de forma remota por medio de ordenadores o smartphones. Es muy parecido a AnyDesk.

Para hacer uso de VNC Viewer debemos descargarnos su programa.

Una vez lo tenemos en nuestro ordenador de casa, lo ejecutamos y aparecerá una ventana como la siguiente

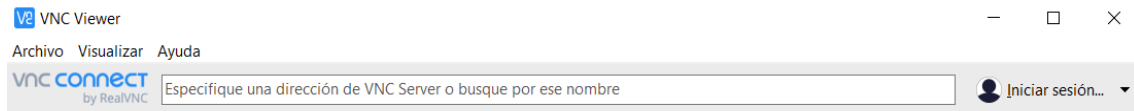


Figura 37. Ventana VCN Viewer

En el buscador que aparece en la figura 37 tenemos que especificar la dirección IP pública del servidor seguido del puerto asignado para la conexión por VNC, en nuestro caso se debe escribir lo siguiente:

`194.169.184.32:1997`

Una vez especificada la dirección y puerto podremos entrar al servidor de forma remota, autenticándonos con la contraseña establecida.

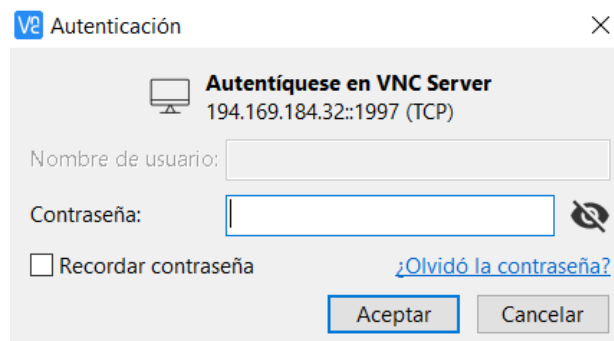


Figura 38. Autenticación VCN Viewer

## Capítulo 4. Pruebas y resultados

Cabe destacar que el servidor lleva funcionando desde Noviembre de 2021, integrando datos desde entonces.

### 4.1 Pruebas de la Base de Datos

Como hemos descrito en los capítulos anteriores, usamos una Base de Datos SQLite, en la cual tenemos 6 tablas que detallamos a continuación:

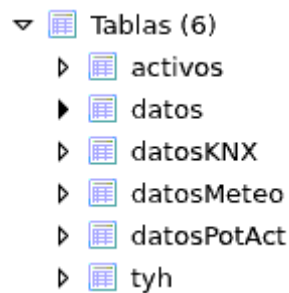


Figura 39. Tablas de la Base de Datos

→ Tabla activos: muestra los sensores activos y así el servidor sabe cuáles debe escuchar

ID	ID_alf	ID_bateria	Fecha_instalacion	Fecha_recarga	Activo
Fi...	Filtro	Filtro	Filtro	Filtro	Filtro
1	1 LA-WU-W	1	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
2	2 LA-WD-W	2	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
3	3 GA-AU-W	3	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
4	4 GA-AD-W	4	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
5	5 CS-WU-N1	5	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
6	6 CS-WD-N1	6	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
7	7 CS-WU-N2	7	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
8	8 CS-WD-N2	8	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
9	9 CS-AU-N1	9	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
10	10 CS-AD-N1	10	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
11	11 CS-WU-E	11	2021-11-29 23:34:52	2021-11-29 23:34:52	Y
12	12 CS-WD-E	12	2021-11-29 23:34:52	2021-11-29 23:34:52	Y

Figura 40. Tabla activos

→ Tabla datos: contiene los datos de los sensores IoT una vez filtrados y procesados

	ID_alf	ID_num	nseq	Temperatura	Humedad	Bateria	Timestamp	Fecha	Hora
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
2791098	CS-AD-E	14	58	22.68359375	56.6171875	83	2022-07-18...	202...	20:...
2791099	B1-WD...	22	81	23.59375	47.109375	84	2022-07-18...	202...	20:...
2791100	B1-WD...	22	81	23.60546875	47.24609375	84	2022-07-18...	202...	20:...
2791101	D3-AD-E	40	2	23.75390625	50.73828125	74	2022-07-18...	202...	20:...
2791102	D3-AD-E	40	2	23.75390625	50.84375	74	2022-07-18...	202...	20:...
2791103	CS-C2-X	16	90	27.234375	44.92578125	81	2022-07-18...	202...	20:...
2791104	CS-C2-X	16	90	27.2109375	44.95703125	81	2022-07-18...	202...	20:...
2791105	B1-WU-W	21	84	26.9609375	39.8515625	83	2022-07-18...	202...	20:...
2791106	B1-WU-W	21	84	27.00390625	39.8671875	83	2022-07-18...	202...	20:...
2791107	B1-AD-W	24	223	23.62109375	53.30078125	81	2022-07-18...	202...	20:...
2791108	B1-AD-W	24	223	23.609375	53.265625	81	2022-07-18...	202...	20:...
2791109	CO-C3-X	20	181	22.3359375	54.9921875	45	2022-07-18...	202...	20:...
2791110	CO-C3-X	20	181	22.3046875	55.05859375	45	2022-07-18...	202...	20:...
2791111	D3-AU-E	46	18	25.9765625	45.37109375	96	2022-07-18...	202...	20:...
2791112	D3-AU-E	46	18	25.94921875	45.3515625	96	2022-07-18...	202...	20:...

Figura 41. Tabla datos

→ Tabla datosKNX: contiene los datos obtenidos de la centralita domótica

	Timestamp	Fecha	Hora	Estado Interior OnOff	Estado Posicion P1
	Filtro	Filtro	Filtro	Filtro	Filtro
57667	2022-07-18 19:38:58:819	2022-07-18	19:38:58:819	1	100,00
57668	2022-07-18 19:42:58:764	2022-07-18	19:42:58:764	1	100,00
57669	2022-07-18 19:46:58:825	2022-07-18	19:46:58:825	0	100,00
57670	2022-07-18 19:50:58:837	2022-07-18	19:50:58:837	0	100,00
57671	2022-07-18 19:54:58:817	2022-07-18	19:54:58:817	0	100,00
57672	2022-07-18 19:58:58:766	2022-07-18	19:58:58:766	0	100,00
57673	2022-07-18 20:02:58:769	2022-07-18	20:02:58:769	0	100,00
57674	2022-07-18 20:06:58:767	2022-07-18	20:06:58:767	0	100,00
57675	2022-07-18 20:10:58:768	2022-07-18	20:10:58:768	0	100,00
57676	2022-07-18 20:14:58:787	2022-07-18	20:14:58:787	0	100,00
57677	2022-07-18 20:18:58:863	2022-07-18	20:18:58:863	0	100,00
57678	2022-07-18 20:22:58:769	2022-07-18	20:22:58:769	0	100,00
57679	2022-07-18 20:26:58:834	2022-07-18	20:26:58:834	0	100,00
57680	2022-07-18 20:30:58:771	2022-07-18	20:30:58:771	0	100,00

Figura 42. Tabla datosKNX

→ Tabla datosMeteo: contiene los datos obtenidos de la estación meteorológica

Tabla: datosMeteo							
	Timestamp	Fecha	Hora	Temperature	Dew Point	Humidity	WindDi
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
34764	2022-07-18 19:14:17:993	2022-07-18	19:14:17:993	33,33	15,00	33	9
34765	2022-07-18 19:20:18:300	2022-07-18	19:20:18:300	33,89	15,00	32	8
34766	2022-07-18 19:26:18:627	2022-07-18	19:26:18:627	33,33	15,00	33	7
34767	2022-07-18 19:32:19:110	2022-07-18	19:32:19:110	32,78	13,89	32	8
34768	2022-07-18 19:38:19:391	2022-07-18	19:38:19:391	32,78	13,89	32	1
34769	2022-07-18 19:44:19:729	2022-07-18	19:44:19:729	33,33	14,44	32	9
34770	2022-07-18 19:50:20:114	2022-07-18	19:50:20:114	32,78	13,89	32	9
34771	2022-07-18 19:56:20:478	2022-07-18	19:56:20:478	32,78	13,89	31	9
34772	2022-07-18 20:02:20:797	2022-07-18	20:02:20:797	32,78	12,78	30	2
34773	2022-07-18 20:08:21:060	2022-07-18	20:08:21:060	32,78	12,22	28	5
34774	2022-07-18 20:14:21:352	2022-07-18	20:14:21:352	32,78	12,22	29	10
34775	2022-07-18 20:20:21:683	2022-07-18	20:20:21:683	32,78	12,78	30	8
34776	2022-07-18 20:26:22:033	2022-07-18	20:26:22:033	32,22	12,78	31	8
34777	2022-07-18 20:32:22:359	2022-07-18	20:32:22:359	31,67	13,33	32	11

Figura 43. Tabla datosMeteo

→ Tabla datosPotAct: contiene los datos de las potencias activas de las máquinas 1 y 2 de la centralita domótica

Tabla: datosPotAct					
	Timestamp	Fecha	Hora	Potencia activa M1	Potencia activa M2
	Filtro	Filtro	Filtro	Filtro	Filtro
279787	2022-07-18 20:29:29:831	2022-07-18	20:29:29:831	30,85	7,03
279788	2022-07-18 20:29:59:831	2022-07-18	20:29:59:831	32,48	12,37
279789	2022-07-18 20:30:29:831	2022-07-18	20:30:29:831	0	4,62
279790	2022-07-18 20:30:59:978	2022-07-18	20:30:59:978	0	10,02
279791	2022-07-18 20:31:29:874	2022-07-18	20:31:29:874	12,85	7,75
279792	2022-07-18 20:31:59:833	2022-07-18	20:31:59:833	13,98	1,57
279793	2022-07-18 20:32:29:885	2022-07-18	20:32:29:885	0	8,41
279794	2022-07-18 20:32:59:834	2022-07-18	20:32:59:834	0	5,49
279795	2022-07-18 20:33:29:922	2022-07-18	20:33:29:922	0	12,53
279796	2022-07-18 20:33:59:835	2022-07-18	20:33:59:835	0	2,83
279797	2022-07-18 20:34:29:929	2022-07-18	20:34:29:929	0	12,51
279798	2022-07-18 20:34:59:911	2022-07-18	20:34:59:911	31,66	13,42
279799	2022-07-18 20:35:29:859	2022-07-18	20:35:29:859	0	6,9
279800	2022-07-18 20:35:59:909	2022-07-18	20:35:59:909	0	6,07
279801	2022-07-18 20:36:29:838	2022-07-18	20:36:29:838	31,04	14,78

Figura 44. Tabla datosPotAct

→ Tabla tyh: contiene todos los datos en bruto de los sensores IoT

ID_alf	ID	ID_gw	Timestamp	nseq	T1	H1	T2	H2	T3	H3	
2947855	D2-AD-E	32	1	2022-07-1...	69	22.87...	52.33...	22.8...	52.3...	22.8...	52.2...
2947856	CS-C3-X	17	2	2022-07-1...	93	30.39...	45.375	30.3...	45.4...	30.3...	45.5...
2947857	CS-C3-X	17	1	2022-07-1...	93	30.39...	45.375	30.3...	45.4...	30.3...	45.5...
2947858	D1-AD-E	36	1	2022-07-1...	204	23.89...	51.18...	23.9...	51.1...	23.8...	51.1...
2947859	D1-WD-E	34	1	2022-07-1...	104	23.90...	48.94...	23.9...	48.8...	23.8...	48.6...
2947860	D1-WD-E	34	2	2022-07-1...	104	23.90...	48.94...	23.9...	48.8...	23.8...	48.6...
2947861	D2-AU-E	31	1	2022-07-1...	89	25.51...	46.24...	25.5...	46.1...	25.4...	46.3...
2947862	D2-AU-E	31	2	2022-07-1...	89	25.51...	46.24...	25.5...	46.1...	25.4...	46.3...
2947863	CS-WU-N2	7	2	2022-07-1...	91	26.72...	46.73...	26.6...	46.8...	26.5...	47.3...
2947864	CS-WU-N2	7	1	2022-07-1...	91	26.72...	46.73...	26.6...	46.8...	26.5...	47.3...
2947865	CS-WD-N2	8	2	2022-07-1...	35	23.29...	54.85...	23.3...	54.7...	23.2...	54.4...
2947866	CS-WD-N2	8	1	2022-07-1...	35	23.29...	54.85...	23.3...	54.7...	23.2...	54.4...
2947867	LA-WD-W	2	2	2022-07-1...	74	23.98...	52.21...	23.9...	52.1...	23.9...	52.0...
2947868	LA-WD-W	2	1	2022-07-1...	74	23.98...	52.21...	23.9...	52.1...	23.9...	52.0...

Figura 45. Tabla tyh

A día de hoy, se han recogido grandes cantidades de datos, teniendo un tamaño de la Base de Datos de 673.4 MB



Figura 46. Tamaño Base de Datos

## 4.2 Pruebas del Bot de Telegram

Como ya hemos comentado en el capítulo anterior, tenemos configurado en el servidor un Bot de Telegram el cual realiza ciertas comprobaciones:

- Si no hay ningún fallo, enviará cada hora un mensaje de información de estado.

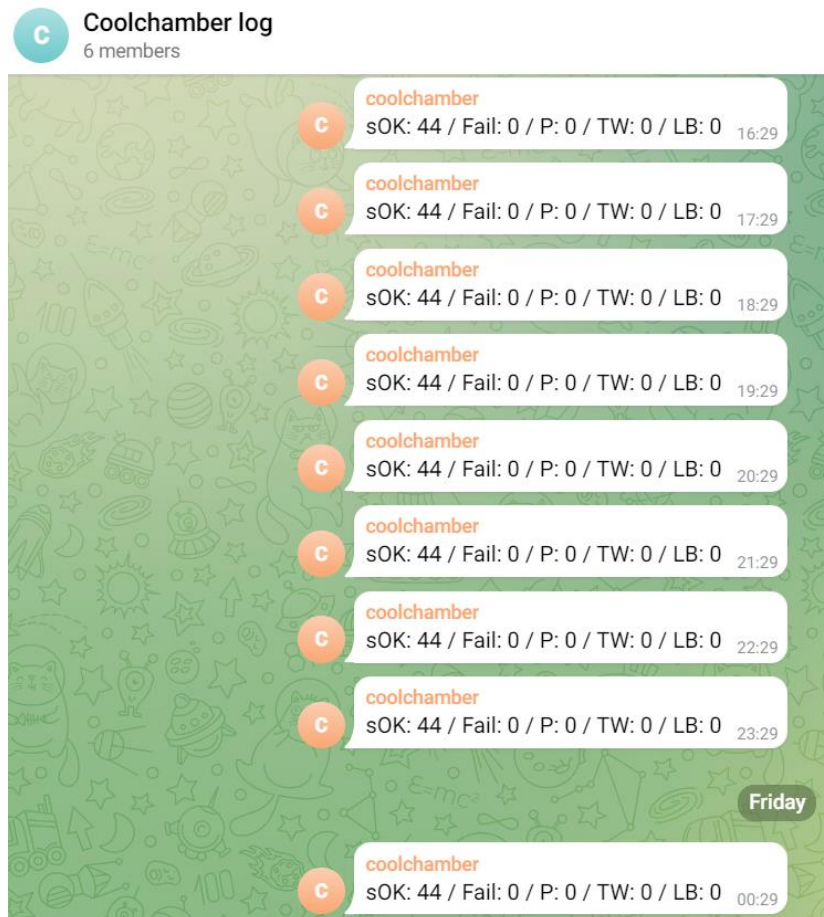


Figura 47. Ejemplo Bot sin fallo



→ Si durante el procesado de datos -realizado cada 15 minutos- se produce un fallo, enviará un mensaje de fallo en ese instante y cada 15 minutos.

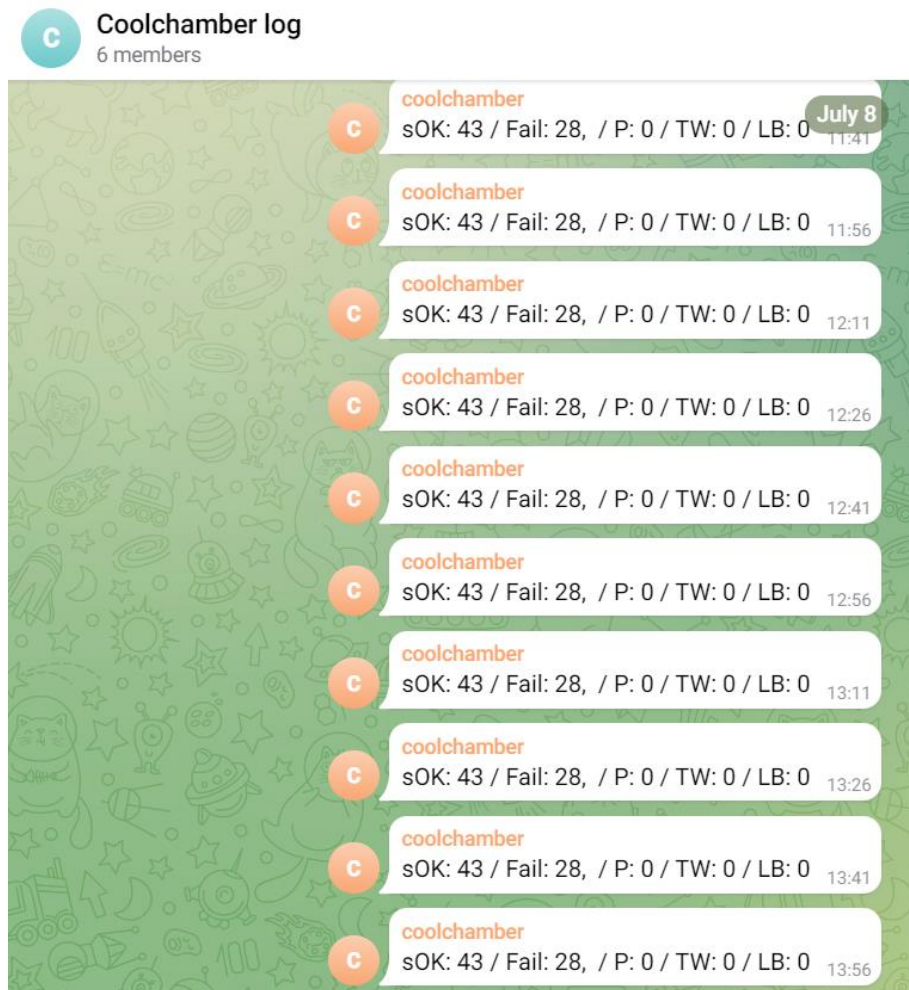


Figura 48. Ejemplo Bot con fallo

→ Si se produce un fallo espontáneo, este se mostrará, pero si no vuelve a fallar, el servidor esperará una hora para enviar su mensaje de información de estado.



Figura 49. Ejemplo Bot con fallo espontáneo

## 4.3 Pruebas de descarga de archivos CSV

Como hemos mencionado en el capítulo anterior, gracias al diseño del Frontend tenemos una página web donde podemos descargar archivos en formato .csv de todos los datos obtenidos y procesados en el servidor:

- Datos sensores IoT
- Datos KNX
- Datos potencias activas
- Datos estación meteorológica

Para ello, accedemos a la página poniendo en el navegador <http://194.169.184.32:27190/>

Para descargar los datos, como ya hemos comentado en el apartado de diseño Frontend y descarga de archivos, primero debemos seleccionar el rango de fechas y horas que se desean. Una vez seleccionado pulsamos sobre el botón Generar CSV y a continuación aparecerá otro botón Descargar CSV el cual permitirá su descarga.

El motivo de los dos botones -generar y descargar- es debido a que, al pulsar sobre el botón Generar CSV se llama a una función validar() que comprueba que los rangos de fechas y horas sean correctos, por ejemplo, no se puede seleccionar una fecha de fin que sea anterior a la fecha de inicio -de igual manera con las horas-, y a su vez no se puede dejar ningún campo vacío.

En las siguientes figuras se muestran ejemplos de selección de fecha/hora incorrecta, campos vacíos y el mensaje que se muestra por pantalla.

**Generar archivos CSV:**

Indique desde fecha/hora: 05/07/2022 19:00 hasta fecha/hora: 05/07/2022 18:00

194.169.184.32:27190 dice  
Incoherencia temporal en las horas

Aceptar

Figura 50. Mensaje incoherencia temporal en horas

**Generar archivos CSV:**

Indique desde fecha/hora:     hasta fecha/hora:

194.169.184.32:27190 dice  
Incoherencia temporal en las fechas

*Figura 51. Mensaje incoherencia temporal en fechas*

**Generar archivos CSV:**

Indique desde fecha/hora:     hasta fecha/hora:

194.169.184.32:27190 dice  
Debe insertar hora de inicio

*Figura 52. Mensaje inserción hora inicio*

**Generar archivos CSV:**

Indique desde fecha/hora:     hasta fecha/hora:

194.169.184.32:27190 dice  
Debe insertar hora de fin

*Figura 53. Mensaje inserción hora fin*

**Generar archivos CSV:**

Indique desde fecha/hora:     hasta fecha/hora:

194.169.184.32:27190 dice  
Debe insertar fecha de inicio

*Figura 54. Mensaje inserción fecha inicio*

**Generar archivos CSV:**

Indique desde fecha/hora:     hasta fecha/hora:

194.169.184.32:27190 dice  
Debe que insertar fecha de fin

*Figura 55. Mensaje inserción fecha fin*

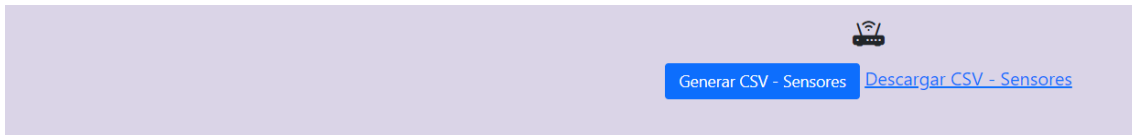
Ya hemos comprobado qué sucede si introducimos una fecha/hora incorrecta o dejamos un campo vacío. A continuación verificaremos la correcta descarga de los datos que necesitamos.

Vamos a seleccionar el siguiente rango de fechas y horas comprendidas en 3 meses, y realizaremos la descarga de los 4 archivos -sensores IoT, KNX, potencias activas y meteo- confirmando que se realiza correctamente.

**Generar archivos CSV:**

Indique desde fecha/hora:     hasta fecha/hora:

*Figura 56. Generación CSV global*



Datos\_sensores\_20....csv ^

Figura 57. Descarga CSV sensores

\* Abriendo el archivo Datos\_sensores podemos visualizar los datos en el rango solicitado

	A	B	C	D	E	F	G
1	Fecha	Hora	ID_num	ID_alf	replace(cast(Temperatura	replace(cast(Humedad	Bateria
2	18/04/2022	09:00:04:911	10	CS-AD-N1	22,74609375	53,7734375	89
3	18/04/2022	09:00:15:421	35	D1-AU-E	22,3203125	54,1328125	90
4	18/04/2022	09:00:34:261	34	D1-WD-E	21,44921875	58,51171875	84
5	18/04/2022	09:00:36:188	47	CS-AU-E	22,390625	55,953125	226
6	18/04/2022	09:00:37:783	24	B1-AD-W	21,76171875	57,23046875	91
7	18/04/2022	09:01:00:552	12	CS-WD-E	22,36328125	64,74609375	86
8	18/04/2022	09:01:04:811	25	VE-WU-W	21,38671875	66,203125	97
9	18/04/2022	09:01:10:039	40	D3-AD-E	21,9765625	55,6640625	84
10	18/04/2022	09:01:14:241	5	CS-WU-N1	21,8671875	64,69140625	92
11	18/04/2022	09:01:14:403	11	CS-WU-E	21,875	63,30078125	94
12	18/04/2022	09:01:17:871	41	PO-AU-E	15,02734375	75,15625	0
13	18/04/2022	09:01:33:451	51	VE-AU-W	22,2734375	55,26953125	100
14	18/04/2022	09:01:36:761	31	D2-AU-E	22,28125	55,77734375	88
15	18/04/2022	09:01:52:931	42	LA-AU-W	24,38671875	52,0390625	88
16	18/04/2022	09:01:56:749	7	CS-WU-N2	21,26953125	62,3203125	94
17	18/04/2022	09:02:03:231	2	LA-WD-W	22,1640625	62,88671875	216
18	18/04/2022	09:02:05:841	16	CS-C2-X	21,52734375	61,91796875	90
19	18/04/2022	09:02:08:718	23	B1-AU-W	22,1015625	55,8671875	91
20	18/04/2022	09:02:38:140	3	GA-AU-W	19,25	67,640625	214
1048562	04/07/2022	23:18:48:202	14	CS-AD-E	23,1796875	69,5625	85
1048563	04/07/2022	23:18:49:816	26	VE-WD-W	23,80859375	55,43359375	80
1048564	04/07/2022	23:18:50:541	16	CS-C2-X	26,55859375	59,44140625	83
1048565	04/07/2022	23:18:50:818	26	VE-WD-W	23,80859375	55,43359375	80
1048566	04/07/2022	23:18:51:137	22	B1-WD-W	23,90234375	52,92578125	213
1048567	04/07/2022	23:18:51:603	16	CS-C2-X	26,55859375	59,44140625	211
1048568	04/07/2022	23:18:52:339	22	B1-WD-W	23,90234375	52,92578125	85
1048569	04/07/2022	23:19:03:786	1	LA-WU-W	26,49609375	52,6484375	81
1048570	04/07/2022	23:19:05:393	1	LA-WU-W	26,49609375	52,6484375	80
1048571	04/07/2022	23:19:21:705	35	D1-AU-E	25,60546875	58,49609375	80
1048572	04/07/2022	23:19:22:941	35	D1-AU-E	25,60546875	58,49609375	80
1048573	04/07/2022	23:19:30:404	15	CS-C1-X1	20,6328125	64,2890625	83
1048574	04/07/2022	23:19:31:480	15	CS-C1-X1	20,6328125	64,2890625	83
1048575	04/07/2022	23:19:32:907	51	VE-AU-W	24,8125	64,69140625	97
1048576	04/07/2022	23:19:34:193	51	VE-AU-W	24,8125	64,69140625	97

Figura 58. Visualización archivo Datos\_sensores.csv

Este archivo ocupa 82.3MB debido a su gran cantidad de datos recogidos durante esos 3 meses. Recordemos que los sensores IoT toman muestras cada 5 minutos y las envían por medio de Gateways al servidor cada 10 minutos con la fecha y hora tomada del muestreo.

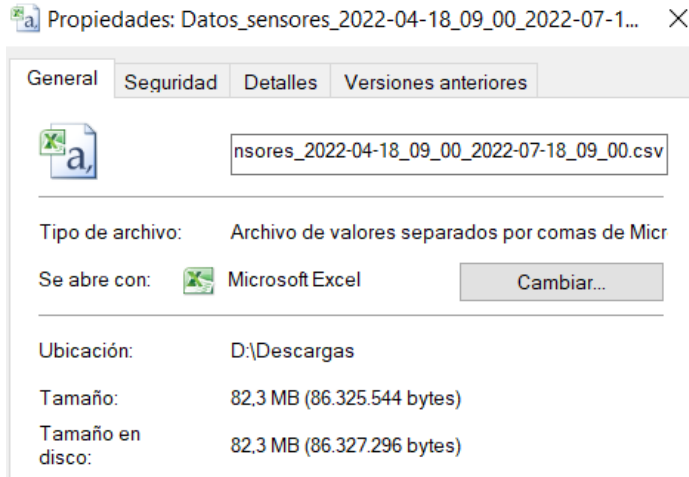
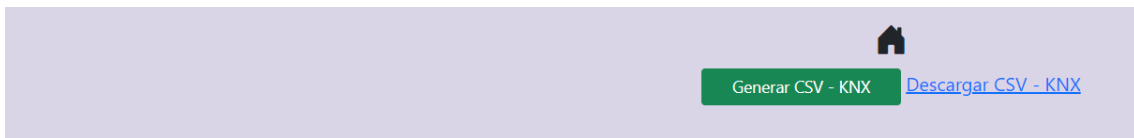


Figura 59. Tamaño archivo Datos\_sensores.csv



Datos\_KNX\_2022-0....csv

Figura 60. Descarga CSV KNX

✳ De la misma forma, abriendo el archivo Datos\_KNX podemos visualizar los datos en el rango solicitado

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	
1	Timestamp	Fecha	Hora	Estado Interic	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Pos	Estado Or	M1 Estado	M1 Estado	Ve M1 Estado	Te M1 Temperat	M2 Estado Or
2	2022-04-18 00:00:34:689	18/04/2022	09:00:34:689	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
3	2022-04-18 00:04:34:739	18/04/2022	09:04:34:739	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
4	2022-04-18 00:08:34:690	18/04/2022	09:08:34:690	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
5	2022-04-18 00:12:34:767	18/04/2022	09:12:34:767	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
6	2022-04-18 00:16:34:691	18/04/2022	09:16:34:691	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
7	2022-04-18 00:20:34:692	18/04/2022	09:20:34:692	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
8	2022-04-18 00:24:34:692	18/04/2022	09:24:34:692	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
9	2022-04-18 00:28:34:824	18/04/2022	09:28:34:824	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
10	2022-04-18 00:32:34:706	18/04/2022	09:32:34:706	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
11	2022-04-18 00:36:34:694	18/04/2022	09:36:34:694	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
12	2022-04-18 00:40:34:698	18/04/2022	09:40:34:698	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
13	2022-04-18 00:44:34:695	18/04/2022	09:44:34:695	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
14	2022-04-18 00:48:34:696	18/04/2022	09:48:34:696	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
15	2022-04-18 00:52:34:697	18/04/2022	09:52:34:697	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
16	2022-04-18 00:56:34:757	18/04/2022	09:56:34:757	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
17	2022-04-18 01:00:34:719	18/04/2022	10:00:34:719	0	0	0	0	0	0	0	0	0	0	0	0	1	33,33	24	19	0
32202	2022-07-18 00:07:30:58:708	18/07/2022	07:30:58:708	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32203	2022-07-18 00:07:34:58:685	18/07/2022	07:34:58:685	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32204	2022-07-18 00:07:38:58:685	18/07/2022	07:38:58:685	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32205	2022-07-18 00:07:42:58:686	18/07/2022	07:42:58:686	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32206	2022-07-18 00:07:46:58:734	18/07/2022	07:46:58:734	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32207	2022-07-18 00:07:50:58:687	18/07/2022	07:50:58:687	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32208	2022-07-18 00:07:54:58:688	18/07/2022	07:54:58:688	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32209	2022-07-18 00:07:58:58:689	18/07/2022	07:58:58:689	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32210	2022-07-18 00:08:02:58:832	18/07/2022	08:02:58:832	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32211	2022-07-18 00:08:06:58:691	18/07/2022	08:06:58:691	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32212	2022-07-18 00:08:10:58:710	18/07/2022	08:10:58:710	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32213	2022-07-18 00:08:14:58:766	18/07/2022	08:14:58:766	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32214	2022-07-18 00:08:18:58:793	18/07/2022	08:18:58:793	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32215	2022-07-18 00:08:22:58:762	18/07/2022	08:22:58:762	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32216	2022-07-18 00:08:26:58:762	18/07/2022	08:26:58:762	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32217	2022-07-18 00:08:30:58:709	18/07/2022	08:30:58:709	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32218	2022-07-18 00:08:34:58:763	18/07/2022	08:34:58:763	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32219	2022-07-18 00:08:38:58:697	18/07/2022	08:38:58:697	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32220	2022-07-18 00:08:42:58:697	18/07/2022	08:42:58:697	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32221	2022-07-18 00:08:46:58:697	18/07/2022	08:46:58:697	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32222	2022-07-18 00:08:50:58:703	18/07/2022	08:50:58:703	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32223	2022-07-18 00:08:54:58:783	18/07/2022	08:54:58:783	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0
32224	2022-07-18 00:08:58:58:762	18/07/2022	08:58:58:762	0	100	100	100	100	100	100	100	100	100	100	0	1	12,94	24	19	0

Figura 61. Visualización archivo Datos\_KNX.csv

Este archivo ocupa 7.7MB, bastante menos que el archivo anterior, pues tiene menos datos guardados en el rango de 3 meses. Recordemos que estos datos se guardan en la Base de Datos cada 4 minutos.

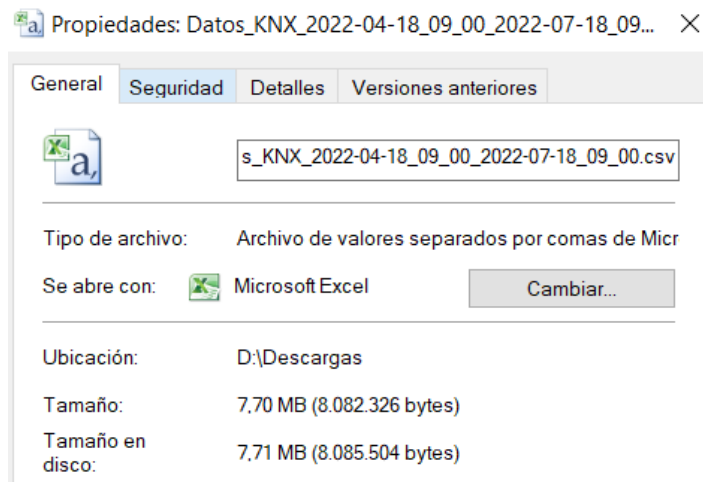


Figura 62. Tamaño archivo Datos\_KNX.csv



Figura 63. Descarga CSV potencias activas

- ✳ De igual manera, abriendo el archivo Datos\_PotenciasActivas podemos visualizar los datos en el rango solicitado



	A	B	C	D	E
1	Timestamp	Fecha	Hora	Potencia activa M1	Potencia activa M2
2	2022-04-18 09:00:06	18/04/2022	09:00:06:219	10,45	15,29
3	2022-04-18 09:00:36	18/04/2022	09:00:36:220	0	20,23
4	2022-04-18 09:01:06	18/04/2022	09:01:06:220	0	16,07
5	2022-04-18 09:01:36	18/04/2022	09:01:36:220	0	54,4
6	2022-04-18 09:02:06	18/04/2022	09:02:06:317	0	19,49
7	2022-04-18 09:02:36	18/04/2022	09:02:36:237	0	38,76
8	2022-04-18 09:03:06	18/04/2022	09:03:06:306	15,85	52,54
9	2022-04-18 09:03:36	18/04/2022	09:03:36:241	7,37	51,3
10	2022-04-18 09:04:06	18/04/2022	09:04:06:333	0	51,31
11	2022-04-18 09:04:36	18/04/2022	09:04:36:241	0	19,28
12	2022-04-18 09:05:06	18/04/2022	09:05:06:389	11,52	31,67
13	2022-04-18 09:05:36	18/04/2022	09:05:36:238	17,82	17,2
14	2022-04-18 09:06:06	18/04/2022	09:06:06:221	16,5	32,15
15	2022-04-18 09:06:36	18/04/2022	09:06:36:316	0	39,23
16	2022-04-18 09:07:06	18/04/2022	09:07:06:299	0	17,81
17	2022-04-18 09:07:36	18/04/2022	09:07:36:221	10,05	31,59
18	2022-04-18 09:08:06	18/04/2022	09:08:06:300	0	37,88
19	2022-04-18 09:08:36	18/04/2022	09:08:36:221	0	32,79
20	2022-04-18 09:09:06	18/04/2022	09:09:06:229	0	51,46
21	2022-04-18 09:09:36	18/04/2022	09:09:36:240	0	31,17
258378	2022-07-18 08:50:29	18/07/2022	08:50:29:270	12	37,34
258379	2022-07-18 08:50:59	18/07/2022	08:50:59:270	6,17	14,7
258380	2022-07-18 08:51:29	18/07/2022	08:51:29:271	0	32,91
258381	2022-07-18 08:51:59	18/07/2022	08:51:59:272	13,67	31,65
258382	2022-07-18 08:52:29	18/07/2022	08:52:29:297	0	51,67
258383	2022-07-18 08:52:59	18/07/2022	08:52:59:272	0	32,48
258384	2022-07-18 08:53:29	18/07/2022	08:53:29:273	0	34,66
258385	2022-07-18 08:53:59	18/07/2022	08:53:59:272	14,4	38,45
258386	2022-07-18 08:54:29	18/07/2022	08:54:29:272	0	35,93
258387	2022-07-18 08:54:59	18/07/2022	08:54:59:352	0	38,36
258388	2022-07-18 08:55:29	18/07/2022	08:55:29:283	9,84	32,82
258389	2022-07-18 08:55:59	18/07/2022	08:55:59:272	0	37,72
258390	2022-07-18 08:56:29	18/07/2022	08:56:29:316	12,14	17,43
258391	2022-07-18 08:56:59	18/07/2022	08:56:59:428	0	36,9
258392	2022-07-18 08:57:29	18/07/2022	08:57:29:367	0	51,33
258393	2022-07-18 08:57:59	18/07/2022	08:57:59:363	0	15,83
258394	2022-07-18 08:58:29	18/07/2022	08:58:29:273	0	40,86
258395	2022-07-18 08:58:59	18/07/2022	08:58:59:369	0	14,25
258396	2022-07-18 08:59:29	18/07/2022	08:59:29:273	0	19,01
258397	2022-07-18 08:59:59	18/07/2022	08:59:59:272	0	33,22

Figura 64. Visualización archivo Datos\_PotenciasActivas.csv

Este archivo ocupa 16.4MB, más del doble que el archivo anterior, pues se han registrado más cantidad de datos de este tipo en el rango de 3 meses. Recordemos que estos datos se guardan en la Base de Datos cada 30 segundos.

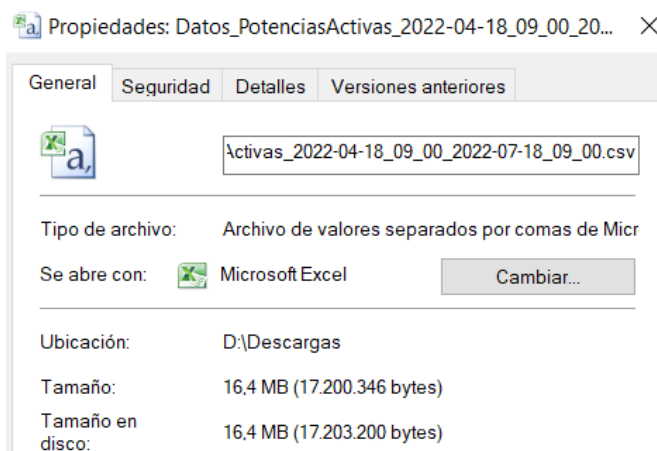


Figura 65. Tamaño archivo Datos\_PotenciasActivas.csv



Datos\_Meteo\_2022....csv

Figura 66. Descarga CSV estación meteorológica

Finalmente, abriendo el archivo Datos\_Meteo podemos visualizar los datos en el rango solicitado

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Timestamp	Fecha	Hora	Temperature	Dew Point	Humidity	WindDir	Lat	Lon	WindSpeed	WindGust	Pressure	PrecipRate	PrecipTotal	UV	SolarRadiation
2	2022-04-18 00	18/04/2022 09:00:27:663	13,89	10,56	79	86	37,601452	-0,9788	4,827	4,827	1015,58	0	0	1	132,1
3	2022-04-18 00	18/04/2022 09:06:27:919	13,89	10,56	79	91	37,601452	-0,9788	4,827	4,827	1015,58	0	0	1	136,9
4	2022-04-18 00	18/04/2022 09:12:28:186	13,89	10,56	79	46	37,601452	-0,9788	4,827	4,827	1015,58	0	0	1	159,8
5	2022-04-18 00	18/04/2022 09:18:28:516	13,89	10,56	78	58	37,601452	-0,9788	4,827	4,827	1015,58	0	0	1	172,6
6	2022-04-18 00	18/04/2022 09:24:28:795	14,44	10	76	62	37,601452	-0,9788	3,218	3,218	1015,58	0	0	1	168,5
7	2022-04-18 00	18/04/2022 09:30:29:049	14,44	10,56	76	60	37,601452	-0,9788	4,827	8,045	1015,58	0	0	0	98,8
8	2022-04-18 00	18/04/2022 09:36:29:319	14,44	10	76	84	37,601452	-0,9788	4,827	4,827	1015,58	0	0	2	219,6
9	2022-04-18 00	18/04/2022 09:42:29:568	15	10,56	74	58	37,601452	-0,9788	6,436	8,045	1015,58	0	0	2	253,5
10	2022-04-18 00	18/04/2022 09:48:29:809	15,56	11,11	76	144	37,601452	-0,9788	3,218	3,218	1015,58	0	0	2	224,3
11	2022-04-18 00	18/04/2022 09:54:30:113	15,56	11,11	74	87	37,601452	-0,9788	1,609	3,218	1015,58	0	0	2	245,9
12	2022-04-18 00	18/04/2022 10:00:30:448	16,11	11,11	72	46	37,601452	-0,9788	4,827	4,827	1015,58	0	0	2	279,4
13	2022-04-18 00	18/04/2022 10:06:30:772	16,67	11,11	70	68	37,601452	-0,9788	1,609	3,218	1015,58	0	0	3	318,4
14	2022-04-18 00	18/04/2022 10:12:31:029	17,22	11,11	68	27	37,601452	-0,9788	1,609	3,218	1015,58	0	0	3	348,1
15	2022-04-18 00	18/04/2022 10:18:31:270	17,22	11,11	68	97	37,601452	-0,9788	1,609	1,609	1015,58	0	0	3	342
16	2022-04-18 00	18/04/2022 10:24:31:608	18,33	11,67	65	338	37,601452	-0,9788	1,609	3,218	1015,58	0	0	2	244,6
17	2022-04-18 00	18/04/2022 10:30:31:844	17,22	11,11	66	80	37,601452	-0,9788	4,827	8,045	1015,58	0	0	2	298,2
18	2022-04-18 00	18/04/2022 10:36:32:136	17,78	11,11	65	42	37,601452	-0,9788	3,218	3,218	1015,58	0	0	3	367,3
19	2022-04-18 00	18/04/2022 10:42:32:694	18,33	11,11	63	348	37,601452	-0,9788	4,827	4,827	1015,58	0	0	4	416,4
20	2022-04-18 00	18/04/2022 10:48:32:971	18,33	11,11	62	244	37,601452	-0,9788	0	0	1015,58	0	0	3	344,7
21	2022-04-18 00	18/04/2022 10:54:33:262	19,44	11,67	59	48	37,601452	-0,9788	3,218	3,218	1015,58	0	0	3	358,8
22	2022-04-18 00	18/04/2022 11:00:33:504	18,89	10,56	59	56	37,601452	-0,9788	1,609	1,609	1015,24	0	0	4	430,8
23	2022-04-18 00	18/04/2022 11:06:33:788	20	11,11	57	2	37,601452	-0,9788	1,609	3,218	1015,24	0	0	4	417,8
24	2022-04-18 00	18/04/2022 11:12:34:039	20	10,56	56	91	37,601452	-0,9788	0	0	1015,58	0	0	4	417
21438	2022-07-18 00	18/07/2022 06:49:42:976	21,11	11,11	52	47	37,601452	-0,9788	6,436	8,045	1017,61	0	0	0	0,9
21439	2022-07-18 00	18/07/2022 06:55:43:231	21,11	11,11	53	58	37,601452	-0,9788	1,609	3,218	1017,61	0	0	0	2
21440	2022-07-18 00	18/07/2022 07:01:43:500	21,11	11,11	53	56	37,601452	-0,9788	1,609	3,218	1017,61	0	0	0	3,9
21441	2022-07-18 00	18/07/2022 07:07:43:733	21,11	11,11	53	47	37,601452	-0,9788	0	1,609	1017,61	0	0	0	6,5
21442	2022-07-18 00	18/07/2022 07:13:43:977	20,56	11,11	54	227	37,601452	-0,9788	0	0	1017,61	0	0	0	9,7
21443	2022-07-18 00	18/07/2022 07:19:44:214	20,56	11,11	54	40	37,601452	-0,9788	0	0	1017,61	0	0	0	13,9
21444	2022-07-18 00	18/07/2022 07:25:44:523	20,56	11,11	53	26	37,601452	-0,9788	3,218	3,218	1017,95	0	0	0	18,5
21445	2022-07-18 00	18/07/2022 07:31:44:810	20,56	11,11	53	50	37,601452	-0,9788	8,045	9,654	1017,61	0	0	0	24,1
21446	2022-07-18 00	18/07/2022 07:37:45:055	21,11	11,11	52	47	37,601452	-0,9788	8,045	11,263	1017,61	0	0	0	30,3
21447	2022-07-18 00	18/07/2022 07:43:45:286	21,11	11,11	52	54	37,601452	-0,9788	6,436	8,045	1017,61	0	0	0	37,2
21448	2022-07-18 00	18/07/2022 07:49:45:533	21,67	11,11	52	27	37,601452	-0,9788	4,827	4,827	1017,95	0	0	0	44,6
21449	2022-07-18 00	18/07/2022 07:55:45:817	21,67	11,11	51	55	37,601452	-0,9788	6,436	8,045	1017,61	0	0	0	52,1
21450	2022-07-18 00	18/07/2022 08:01:46:106	21,67	11,11	51	50	37,601452	-0,9788	4,827	4,827	1017,61	0	0	0	59,9
21451	2022-07-18 00	18/07/2022 08:07:46:467	22,22	11,11	50	49	37,601452	-0,9788	6,436	9,654	1017,61	0	0	0	69,2
21452	2022-07-18 00	18/07/2022 08:13:46:692	22,22	11,11	49	58	37,601452	-0,9788	6,436	8,045	1017,95	0	0	0	79
21453	2022-07-18 00	18/07/2022 08:19:46:965	22,22	11,11	49	65	37,601452	-0,9788	4,827	8,045	1017,61	0	0	0	88,9
21454	2022-07-18 00	18/07/2022 08:25:47:239	22,22	11,11	48	50	37,601452	-0,9788	11,263	12,872	1017,95	0	0	0	98,9
21455	2022-07-18 00	18/07/2022 08:31:47:509	22,22	11,11	48	46	37,601452	-0,9788	9,654	9,654	1017,95	0	0	1	109,4
21456	2022-07-18 00	18/07/2022 08:37:47:756	23,33	11,11	46	64	37,601452	-0,9788	6,436	9,654	1017,95	0	0	1	120,4
21457	2022-07-18 00	18/07/2022 08:43:47:981	23,33	11,11	46	69	37,601452	-0,9788	8,045	9,654	1017,95	0	0	1	130,7
21458	2022-07-18 00	18/07/2022 08:49:48:244	23,33	10,56	45	55	37,601452	-0,9788	8,045	11,263	1018,28	0	0	1	142,1
21459	2022-07-18 00	18/07/2022 08:55:48:478	23,89	11,11	45	65	37,601452	-0,9788	3,218	3,218	1017,95	0	0	1	154,1

Figura 67. Visualización archivo Datos\_Meteo.csv

Este archivo ocupa 2.79MB, mucho menos que los archivos anteriores, pues se han registrado menos cantidad de datos de este tipo en el rango de 3 meses. Recordemos que estos datos se guardan en la Base de Datos cada 6 minutos.

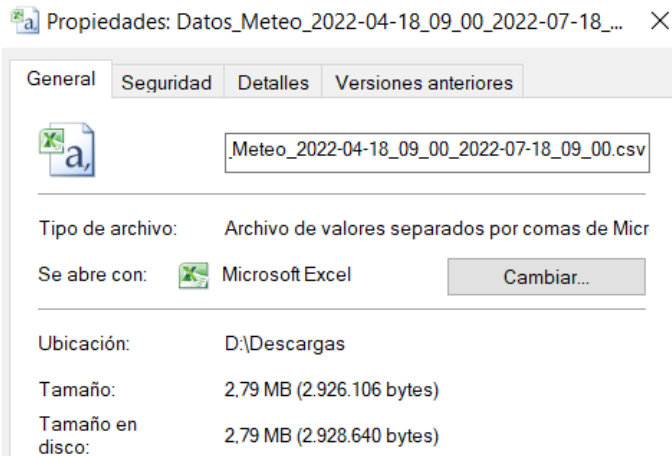


Figura 68. Tamaño archivo Datos\_Meteo.csv

# Capítulo 5. Conclusiones y trabajos futuros

## 5.1 Conclusiones

Este proyecto ha servido de ayuda para un sistema IoT de monitorización y recopilación de parámetros para el estudio del comportamiento térmico y de gasto de energía de una vivienda real en un proyecto de investigación CoolChamber. Se ha cumplido satisfactoriamente el objetivo principal de facilitar y simplificar el trabajo de los investigadores porque todos los datos están centralizados y fácilmente accesible en el servidor desarrollado.

En este trabajo se ha utilizado un servidor Linux con la distribución Ubuntu 18.04.6 LTS para llevar a cabo el diseño y desarrollo de la arquitectura del servidor.

Para instalar esta versión en nuestro ordenador debemos tener ciertos requisitos mínimos:

- Procesador de 64 bits de 700 MHz, idealmente de 1 GHz.
- 1 GB de memoria RAM, preferiblemente 2 GB o más.
- 10 GB de disco duro, pudiendo ser de 20 GB si se necesita.
- Lector de DVD o puerto USB para su instalación.

Como ventajas de esta distribución cabe destacar:

- Versión LTS -Long Term Support- para entornos de producción.
- Tiempo de arranque menor.
- Nueva versión de Apache y PHP, soportando el protocolo HTTP2 y mejorando la seguridad.

En este trabajo he programado tanto de forma remota en el servidor con Linux, como de forma local utilizando Visual Studio Code en Windows.

Realmente no ha supuesto un gran cambio utilizar un Sistema Operativo u otro, pues gracias a Visual Studio Code se pueden realizar pruebas y obtener los mismos resultados que llevándolo al desarrollo del servidor con Ubuntu.

El único problema que pudiera tener la programación del servidor de forma remota es, si en algún momento el router de la vivienda o cualquier dispositivo que tenemos en ella fallase o se apagara, debemos asistir de forma física a la vivienda para solucionarlo. Gracias a PM2 este problema no afecta al servidor en sí, pues si hubiera un corte de luz, PM2 reiniciaría el programa del servidor.

Gracias a la recogida de todos los datos en el servidor, los investigadores pueden:

- Contrastar los datos ambientales exteriores con los del interior de la vivienda.
- Estudiar el comportamiento térmico de la vivienda, pudiendo descargar los archivos .csv para relacionar con los registros de los 44 módulos sensores y datos exteriores de la estación meteorológica.
- Obtener fecha y hora de presencia en la vivienda o vivienda vacía. Que supone la apertura de alguna puerta principal o puerta de acceso al porche y modifica la circulación de aire y ventilación en la vivienda frente a estar completamente cerrada.
- Obtener fecha y hora con estado de las persianas -abiertas/cerradas- y grado de apertura para correlar con la entrada de radiación solar.
- Consultar el consumo instantáneo de la máquina de climatización envolvente y de la máquina de climatización interior de la vivienda.
- ✳ El objetivo principal se ha logrado con creces y gracias a él los investigadores tienen un punto centralizado de recogida de datos.

Además ha sido de gran ayuda la incorporación del Bot de Telegram, pues nos avisa de forma rápida cuando hay algún incidente y poder así llevar un seguimiento.

## 5.2 Grado de consecución de los objetivos y formación adquirida

### **Resumen de los objetivos propuestos y alcanzados**

Los objetivos técnicos logrados en este trabajo han sido los siguientes:

- Estudio y uso del protocolo MQTT para comunicación con los Gateways que interactúan con los sensores IoT de temperatura y humedad relativa.
- Estudio y configuración del protocolo Modbus para comunicación e integración de datos de los dispositivos KNX de la vivienda.
- Estudio, pruebas y uso de llamadas API REST para obtención de datos de la estación meteorológica.
- Diseño y desarrollo de Backend y Base de Datos para recepción y procesado de datos.
- Implementación de sistema de alerta y avisos de mantenimiento mediante Bot de Telegram.
- Integración de estudio y pruebas de sensores y sistema para monitorizar apertura y cierre de puertas y ventanas.
- Diseño y desarrollo de Frontend de visualización y descarga de datos a medida de los usuarios.

### **Curva de aprendizaje y principales dificultades encontradas**

La mayor dificultad de este proyecto ha sido entender el funcionamiento del protocolo industrial Modbus, que es ampliamente usado en dispositivos sensores en instalaciones industriales. Con los conocimientos del grado de Ingeniería Telemática he tenido los fundamentos teóricos para estudiar y comprender el protocolo y aprender a manejarlo.

Me instalé ciertas aplicaciones que simulaban la comunicación Modbus por medio de Maestro/Esclavo para poder comprender cómo se comunican y qué variables se deben utilizar. También pude hacer pruebas con una interfaz más amigable -Modbus Tools- que me facilitó la asimilación de conceptos.

Cabe destacar que, aunque no fuera del todo complicado, me llevó varios días recordar cómo funciona un servicio REST, tanto para la obtención de datos de la estación meteorológica, como para el propio servicio REST que posee el servidor para la inserción y obtención de datos de la Base de Datos. Además el uso de la herramienta Postman facilita la realización de pruebas y consultas a API REST para comprobar que funciona correctamente.

### **Resumen de los objetivos propuestos y alcanzados**

Para la realización de este trabajo ha sido necesario entender bien cómo funcionan los distintos protocolos que se han utilizado para la obtención de datos.

También se ha aprendido e indagado en conceptos sobre MQTT y Modbus, así como las peticiones a API REST.

Se ha familiarizado más sobre cómo diseñar y desplegar un servidor gracias a los conocimientos obtenidos en varias asignaturas del grado y máster, junto con la arquitectura desarrollada tras la realización del Trabajo de Fin de Grado.

Por último, se han adquirido nuevos conceptos y habilidades para la creación del Backend y Frontend con NodeJS, así como adquirir nuevas metodologías para su desarrollo.

## 5.3 Trabajos futuros

Como se ha indicado previamente, se ha cumplido satisfactoriamente el objetivo de ayudar a los investigadores del proyecto CoolChamber a tener datos de calidad y en cantidad de forma accesible, centralizada y simplificada en el servidor. Si espera que el proyecto pueda tener continuación y monitorizar otras viviendas y como trabajos futuros se propone:

- Evaluar el despliegue del servidor en la nube y simplificar la infraestructura a instalar en una vivienda.
- Evaluar y diseñar herramientas de reporting, como PowerBI, para facilitar la explotación y análisis de datos de distintas fuentes a los investigadores.



# Anexos

## Listado código sensores IoT

ID SENSOR	CODIGO	CODIGO SALA	POSICION	ORIENTACION
3	GA-AU-W	GA (Garaje)	AU (Pared Arriba)	W (Oeste)
4	GA-AD-W	GA (Garaje)	AD (Pared Abajo)	W (Oeste)
1	LA-WU-W	LA (Lavadero)	WU (Cámara Arriba)	W (Oeste)
2	LA-WD-W	LA (Lavadero)	WD (Cámara Abajo)	W (Oeste)
42	LA-AU-W	LA (Lavadero)	AU (Pared Arriba)	W (Oeste)
43	LA-AD-W	LA (Lavadero)	AD (Pared Abajo)	W (Oeste)
5	CS-WU-N1	CS (Cocina-Salon)	WU (Cámara Arriba)	N (Norte) Pos1
6	CS-WD-N1	CS (Cocina-Salon)	WD (Cámara Abajo)	N (Norte) Pos1
9	CS-AU-N1	CS (Cocina-Salon)	AU (Pared Arriba)	N (Norte) Pos1
10	CS-AD-N1	CS (Cocina-Salon)	AD (Pared Abajo)	N (Norte) Pos1
7	CS-WU-N2	CS (Cocina-Salon)	WU (Cámara Arriba)	N (Norte) Pos2
8	CS-WD-N2	CS (Cocina-Salon)	WD (Cámara Abajo)	N (Norte) Pos2
11	CS-WU-E	CS (Cocina-Salon)	WU (Cámara Arriba)	E (Este)
12	CS-WD-E	CS (Cocina-Salon)	WD (Cámara Abajo)	E (Este)
13	CS-AU-E	CS (Cocina-Salon)	AU (Pared Arriba)	E (Este)
14	CS-AD-E	CS (Cocina-Salon)	AD (Pared Abajo)	E (Este)
15	CS-C1-X1	CS (Cocina-Salon)	C1 (Cámara suelo)	X (SinOrient)
16	CS-C2-X	CS (Cocina-Salon)	C2 (Pared nivel 1)	X (SinOrient)
17	CS-C3-X	CS (Cocina-Salon)	C3 (Pared nivel 2)	X (SinOrient)
44	CO-C1-X1	CO (Corredor)	C1 (Cámara suelo)	X (SinOrient)
45	CO-C1-X2	CO (Corredor)	C1 (Cámara suelo)	
19	CO-C2-X	CO (Corredor)	C2 (Pared nivel1)	X (SinOrient)

20	CO-C3-X	CO (Corredor)	C3 (Pared nivel 2)	X (SinOrient)
21	B1-WU-W	B1 (Baño1)	WU (Cámara Arriba)	W (Oeste)
22	B1-WD-W	B1 (Baño1)	WD (Cámara Abajo)	W (Oeste)
23	B1-AU-W	B1 (Baño1)	AU (Pared Arriba)	W (Oeste)
24	B1-AD-W	B1 (Baño1)	AD (Pared Abajo)	W (Oeste)
25	VE-WU-W	VE (Vestidor)	WU (Cámara Arriba)	E (Este)
26	VE-WD-W	VE (Vestidor)	WD (Cámara Abajo)	E (Este)
27	VE-AU-W	VE (Vestidor)	AU (Pared Arriba)	E (Este)
28	VE-AD-W	VE (Vestidor)	AD (Pared Abajo)	E (Este)
29	VE-WU-S	VE (Vestidor)	WU (Cámara Arriba)	S (Sur)
30	VE-WD-S	VE (Vestidor)	WD (Cámara Abajo)	S (Sur)
33	D1-WU-E	D1 (Dormit. 1)	WU (Cámara Arriba)	E (Este)
34	D1-WD-E	D1 (Dormit. 1)	WD (Cámara Abajo)	E (Este)
35	D1-AU-E	D1 (Dormit. 1)	AU (Pared Arriba)	E (Este)
36	D1-AD-E	D1 (Dormit. 1)	AD (Pared Abajo)	E (Este)
31	D2-AU-E	D2 (Dormit. 2)	AU (Pared Arriba)	E (Este)
32	D2-AD-E	D2 (Dormit. 2)	AD (Pared Abajo)	E (Este)
37	D3-WU-E	D3 (Dormit. 3)	WU (Cámara Arriba)	E (Este)
38	D3-WD-E	D3 (Dormit. 3)	WD (Cámara Abajo)	E (Este)
46	D3-AU-E	D3 (Dormit. 3)	AU (Pared Arriba)	E (Este)
40	D3-AD-E	D3 (Dormit. 3)	AD (Pared Abajo)	E (Este)
41	PO-AU-E	PO (Porche)	AU (Pared Arriba)	E (Este)



## Direcciones KNX y Modbus

<b>Nombre</b>	<b>Dirección KNX</b>	<b>Permisos</b>	<b>Dirección Modbus*</b>
ON/OFF Interior	1/0/3	W	0
Estado ON/OFF Interior	1/0/4	R	1
Posición % Persiana 1	2/3/1	W	2
Posición % Persiana 2	2/3/2	W	3
Posición % Persiana 3	2/3/3	W	4
Posición % Persiana 4	2/3/4	W	5
Posición % Persiana 5	2/3/5	W	6
Posición % Persiana 6	2/3/6	W	7
Posición % Persiana 7	2/3/7	W	8
Posición % Persiana 8	2/3/8	W	9
Posición % Persiana 9	2/3/9	W	10
Estado Posición % Persiana 1	2/4/1	R	11
Estado Posición % Persiana 2	2/4/2	R	12
Estado Posición % Persiana 3	2/4/3	R	13
Estado Posición % Persiana 4	2/4/4	R	14
Estado Posición % Persiana 5	2/4/5	R	15
Estado Posición % Persiana 6	2/4/6	R	16
Estado Posición % Persiana 7	2/4/7	R	17
Estado Posición % Persiana 8	2/4/8	R	18
Estado Posición % Persiana 9	2/4/9	R	19
M1 ON/OFF	3/1/1	W	20
M1 Estado ON/OFF	3/1/2	R	21
M1 Modo 0=Aut/1=Hea/3=Coo/9=Fan/14=Dry	3/1/3	W	22

M1 Estado Modo 0=Aut/1=Hea/3=Coo/9=Fan/14=Dry	3/1/4	R	23
M1 Velocidad manual ventilador %	3/1/9	W	24
M1 Estado Velocidad manual ventilador %	3/1/10	R	25
M1 Modo Ventilador 0=MAN/1=AUTO	3/1/11	W	26
M1 Estado Modo Ventilador 0=MAN/1=AUTO	3/1/12	R	27
M1 Temperatura de consigna	3/1/13	W	28
M1 Estado Temperatura de consigna	3/1/14	R	29
M1 Temperatura de retorno	3/1/15	R	30
M2 ON/OFF	3/2/1	W	31
M2 Estado ON/OFF	3/2/2	R	32
M2 Modo 0=Aut/1=Hea/3=Coo/9=Fan/14=Dry	3/2/3	W	33
M2 Estado Modo 0=Aut/1=Hea/3=Coo/9=Fan/14=Dry	3/2/4	R	34
M2 Velocidad manual ventilador %	3/2/9	W	35
M2 Estado Velocidad manual ventilador %	3/2/10	R	36
M2 Modo Ventilador 0=MAN/1=AUTO	3/2/11	W	37
M2 Estado Modo Ventilador 0=MAN/1=AUTO	3/2/12	R	38
M2 Temperatura de consigna	3/2/13	W	39
M2 Estado Temperatura de consigna	3/2/14	R	40
M2 Temperatura de retorno	3/2/15	R	41
Recuperador de Calor - Valor de entrada %	4/1/1	W	42
Recuperador de Calor - Valor de salida %	4/1/2	R	43
Salón - Luminosidad (lux)	5/1/1	R	44
Salón - Presencia	5/1/2	R	45
Salón - CO2 [ppm]	5/1/3	R	46
Salón - Humedad relativa [%]	5/1/4	R	47
Dormitorio 3 - Luminosidad	5/2/1	R	48

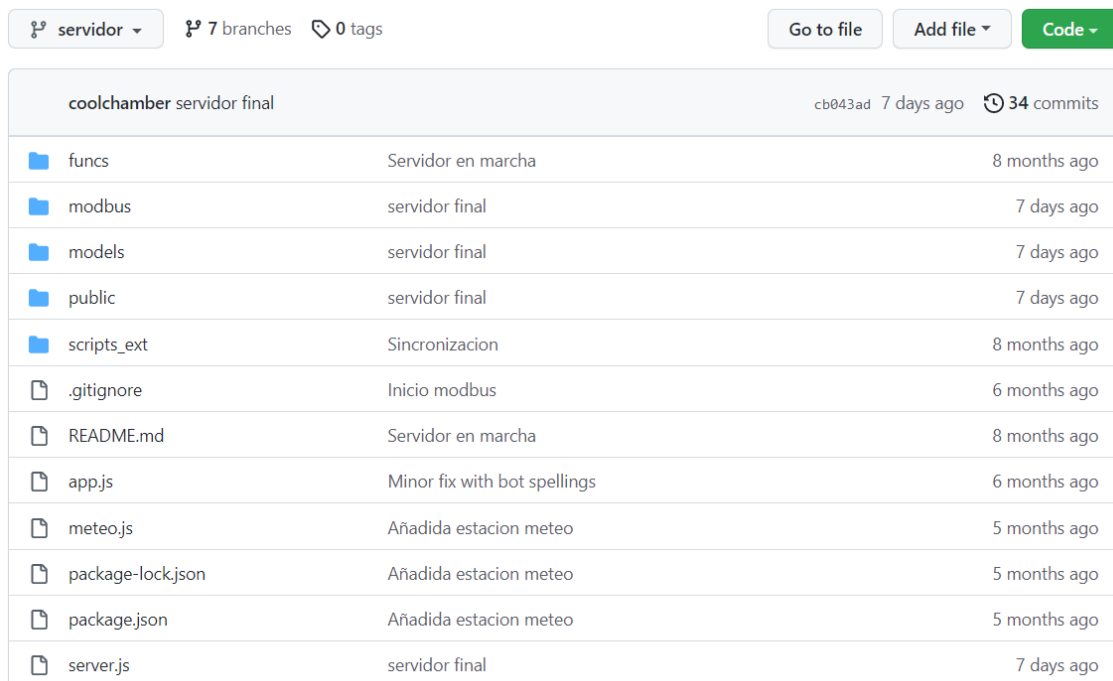
Dormitorio 3 - Presencia	5/2/2	R	49
Dormitorio 2 - Luminosidad	5/3/1	R	50
Dormitorio 2 - Presencia	5/3/2	R	51
Dormitorio principal - Luminosidad	5/4/1	R	52
Dormitorio principal - Presencia	5/4/2	R	53
Dormitorio principal - CO2 [ppm]	5/4/3	R	54
Dormitorio principal - Humedad relativa [%]	5/4/4	R	55

## Estructura GitHub

Se ha llevado un control de versiones del servidor por medio de GitHub, habiendo realizado 34 commits especificando qué se ha realizado en cada uno.

Podemos observar que tenemos 7 ramas -branches- en nuestro proyecto, se han utilizado distintas ramas para poder realizar pruebas y subir el código de forma estructurada para posteriormente realizar una unión -merge- del código final sin errores.

Por ejemplo, hay una rama llamada “modbus” en la que se ha subido todo el código referido a la parte de recepción, procesado y guardado de datos por modbus. También existe una rama “meteo” donde se ha subido todo el código relacionado a la estación meteorológica. A su vez, tenemos otras ramas “enprueba” y “tofix” en las que se ha subido cierta parte del código pero aún no es el final y se deben realizar varios cambios.



The screenshot shows the GitHub repository interface for 'servidor'. At the top, it indicates '7 branches' and '0 tags'. Below this is a table listing the repository's structure, including folders and files with their commit messages and dates.

coolchamber servidor final		cb043ad 7 days ago	🕒 34 commits
📁 funcs	Servidor en marcha		8 months ago
📁 modbus	servidor final		7 days ago
📁 models	servidor final		7 days ago
📁 public	servidor final		7 days ago
📁 scripts_ext	Sincronizacion		8 months ago
📄 .gitignore	Inicio modbus		6 months ago
📄 README.md	Servidor en marcha		8 months ago
📄 app.js	Minor fix with bot spellings		6 months ago
📄 meteo.js	Añadida estacion meteo		5 months ago
📄 package-lock.json	Añadida estacion meteo		5 months ago
📄 package.json	Añadida estacion meteo		5 months ago
📄 server.js	servidor final		7 days ago

Figura 69. Estructura Github

# Bibliografía y referencias

- [1] Proyecto Choolchamber: <https://www.casasmundejo.com/proyecto-coolchamber/>
- [2] DB Browser for SQLite: <https://sqlitebrowser.org/>
- [3] Git: <https://git-scm.com/>
- [4] Modbus: <https://modbus.org/>
- [5] Mosquitto: <https://mosquitto.org/>
- [6] NodeJS: <https://nodejs.org/en/>
- [7] Npm: <https://www.npmjs.com/>
- [8] Postman: <https://www.postman.com/>
- [9] Visual Studio Code: <https://code.visualstudio.com/>
- [10] Ubuntu 18.04.6 LTS: <https://releases.ubuntu.com/18.04/>
- [11] Dispositivo sensor Sensirion SHT35:  
<https://sensirion.com/products/catalog/SHT35-DIS-F/>
- [12] MQTT: The Standard for IoT Messaging <https://mqtt.org/>
- [13] Software Modbus: <https://www.modbusdriver.com/modpoll.html>
- [14] Documentación servicio REST:  
[https://docs.google.com/document/d/1eKCnKXI9xnoMGRRzOL1xPCBihNV2rOet08qpE\\_gArAY/edit](https://docs.google.com/document/d/1eKCnKXI9xnoMGRRzOL1xPCBihNV2rOet08qpE_gArAY/edit)  
<https://docs.google.com/document/d/1KGb8bTVYRsNgljnNH67AMhckY8AQT2FVwZ9urj8SWBs/edit>
- [15] PM2: <https://pm2.keymetrics.io/docs/usage/quick-start/>
- [16] VNC Viewer: <https://www.realvnc.com/es/connect/download/viewer/>