Índice general

V

Índice general

Resumen I

1. Introducción general 1
1.1. Hogares inteligentes . . . 1
1.1.1. Aplicaciones comunes . . . 1
1.2. Motivación . . . 2
1.3. Estado del arte . . . 2
1.4. Objetivos y alcance . . . 3

2. Introducción específica 5
2.1. Protocolos de comunicación . . . 5
2.1.1. Modelo OSI . . . 5
2.1.2. Protocolo HTTP . . . 6
2.1.3. Protocolo MQTT . . . 7
Calidad de servicio . . . 8
Protocolos SSL y TLS . . . 8
2.2. Componentes de hardware . . . 8
2.2.1. Raspberry Pi . . . 8
2.2.2. Sistema en chip . . . 9
Familia ESP32 . . . 10
2.3. Herramientas de software . . . 11
2.3.1. Visual studio code . . . 11
2.3.2. Marco de desarrollo ESP . . . 11
2.3.3. Lenguajes JavaScript y TypeScript . . . 11
2.3.4. Angular y Ionic . . . 12
2.3.5. Bases de datos relacionales MySQL y MariaDB . . . 12
2.3.6. Contenedores con Docker . . . 12

3. Diseño e implementación 15
3.1. Arquitectura del sistema . . . 15
3.1.1. Especificaciones técnicas del servidor . . . 16
3.2. Modelo de datos . . . 16
3.3. Desarrollo del frontend . . . 16
3.4. Desarrollo del backend . . . 16
3.5. Nodos, sensores y actuadores . . . 16
3.6. Comunicación del sistema . . . 16

4. Ensayos y resultados 17

5. Conclusiones 19

Bibliografía 21

Índice de figuras

VII

Índice de figuras

1.1. Ejemplo de sistema de domótica 2
1.2. Matter y Home Assistant 3
1.3. Esquema básico 4

2.1. Modelo OSI 5
2.2. Arquitectura MQTT 7
2.3. Raspberry Pi 9
2.4. Familia ESP32 10
2.5. Docker 13

3.1. Arquitectura cliente-servidor 15

Índice de tablas

IX

Índice de tablas

Calidad de servicio

MQTT está diseñado para ser simple y minimizar el ancho de banda, lo que lo convierte en una buena opción para muchos escenarios, aunque esta simplicidad hace que la interpretación de los mensajes dependa completamente del diseñador del sistema. Para mitigar este tipo de inconvenientes se soportan distintos niveles de calidad de servicio.

Estos niveles determinan cómo se entrega cada mensaje y es necesario especificar un QoS (siglas en inglés para *quality of service*) para cada *topic* MQTT. Es importante elegir el valor de QoS adecuado para cada mensaje que está determinado de la siguiente manera:

- QoS 0 (a lo sumo una vez): los mensajes se mandan sin tener en cuenta si llegan o no. Puede haber pérdida de mensajes. No se hacen retransmisiones.
- QoS 1 (al menos una vez): se asegura que los mensajes lleguen pero se pueden producir duplicados. El receptor recibe el mensaje por lo menos una vez. Si el receptor no confirma la recepción del mensaje o se pierde en el camino se reenvía el mensaje hasta que recibe por lo menos una confirmación.
- QoS 2 (exactamente una vez): se asegura que el mensaje llegue exactamente una vez. Eso incrementa la sobrecarga en la comunicación pero es la mejor opción cuando la duplicación de un mensaje no es aceptable.

Protocolos SSL y TLS

SSL y TLS son protocolos criptográficos, que proporcionan comunicaciones seguras por una red. Se usa criptografía asimétrica para autentificar a la contraparte con quien se está comunicando y para intercambiar una llave simétrica, iniciando una sesión entre las partes intervinientes. Esta sesión es luego usada para cifrar el flujo de datos entre las partes. Esto permite la confidencialidad de la información transmitida y la autenticación de los mensajes.

Antes de que un cliente y el servidor pueden empezar a intercambiar información protegida por TLS, deben intercambiar en forma segura o acordar una clave de cifrado y una clave para usar cuando se cifren los datos. Existen varios métodos utilizados para el intercambio y acuerdo de claves [11].

2.2. Componentes de hardware

2.2.1. Raspberry Pi

Las Raspberry Pi son computadoras de placa simple (en inglés *Single Board Computer* o su sigla SBC). Actualmente en el mercado existen distintas marcas y variantes de este tipo de placas, siendo elegidas tanto como computadoras de escritorio como para algunas aplicaciones específicas. A continuación, se describen las principales características de la familia Raspberry Pi:

- Posee distintos puertos, permitiendo conectarla a periféricos como pueden ser una pantalla, un medio de almacenamiento e incluso cualquier dispositivo que tenga interfaz USB.
- Contiene un procesador gráfico integrado, lo que permite la reproducción de vídeo, incluso en alta definición.

8 Capítulo 2. Introducción específica

Calidad de servicio

MQTT está diseñado para ser simple y minimizar el ancho de banda, lo que lo convierte en una buena opción para muchos escenarios, aunque esta simplicidad hace que la interpretación de los mensajes dependa completamente del diseñador del sistema. Para mitigar este tipo de inconvenientes se soportan distintos niveles de calidad de servicio.

Estos niveles determinan cómo se entrega cada mensaje y es necesario especificar un QoS (siglas en inglés para *quality of service*) para cada *topic* MQTT. Es importante elegir el valor de QoS adecuado para cada mensaje que está determinado de la siguiente manera:

- QoS 0 (a lo sumo una vez): los mensajes se mandan sin tener en cuenta si llegan o no. Puede haber pérdida de mensajes. No se hacen retransmisiones.
- QoS 1 (al menos una vez): se asegura que los mensajes lleguen pero se pueden producir duplicados. El receptor recibe el mensaje por lo menos una vez. Si el receptor no confirma la recepción del mensaje o se pierde en el camino se reenvía el mensaje hasta que recibe por lo menos una confirmación.
- QoS 2 (exactamente una vez): se asegura que el mensaje llegue exactamente una vez. Eso incrementa la sobrecarga en la comunicación pero es la mejor opción cuando la duplicación de un mensaje no es aceptable.

Protocolos SSL y TLS

SSL y TLS son protocolos criptográficos, que proporcionan comunicaciones seguras por una red. Se usa criptografía asimétrica para autentificar a la contraparte con quien se está comunicando y para intercambiar una llave simétrica, iniciando una sesión entre las partes intervinientes. Esta sesión es luego usada para cifrar el flujo de datos entre las partes. Esto permite la confidencialidad de la información transmitida, códigos de autenticación de mensajes y como resultado secundario, autenticación de los mensajes.

Antes de que un cliente y el servidor pueden empezar a intercambiar información protegida por TLS, deben intercambiar en forma segura o acordar una clave de cifrado y una clave para usar cuando se cifren los datos. Existen varios métodos utilizados para el intercambio y acuerdo de claves [11].

2.2. Componentes de hardware

2.2.1. Raspberry Pi

Las Raspberry Pi son computadoras de placa simple (en inglés *Single Board Computer* o su sigla SBC). Actualmente en el mercado existen distintas marcas y variantes de este tipo de placas, siendo elegidas tanto como computadoras de escritorio como para algunas aplicaciones específicas. A continuación, se describen las principales características de la familia Raspberry Pi:

- Posee distintos puertos, permitiendo conectarla a periféricos como pueden ser una pantalla, un medio de almacenamiento e incluso cualquier dispositivo que tenga interfaz USB.

- Permite la conexión a la red a través del puerto de Ethernet y algunos modelos permiten conexión Wi-Fi y Bluetooth.
- Consta de una ranura microSD que permite instalar y ejecutar sistemas operativos a través de una tarjeta de memoria.

El diseño de la Raspberry Pi fue evolucionando con el correr del tiempo. En la actualidad se encuentran los modelos Raspberry Pi 4 y Raspberry Pi 400, siendo los más modernos y robustos lanzados por la marca. La familia de Raspberry Pi 4 cuenta con 4 modelos de placa con distinta capacidad de memoria RAM, yendo desde 1 GB hasta 8 GB. La Raspberry Pi 400 es una variante de la Raspberry Pi 4. También existen otros modelos más pequeños diseñados para otro tipo de aplicaciones.

En la figura 2.3 se pueden ver los modelos Raspberry Pi 4 y Raspberry Pi 400.

FIGURA 2.3. Raspberry Pi 4 y Raspberry Pi 400. [3]

Algunas de las funciones y aplicaciones más comunes de este tipo de computadoras se describen a continuación:

- Navegar en la red, utilizar aplicaciones de oficina para la edición de documentos y emplearla como si fuese una computadora de escritorio.
- Crear un centro multimedia y ver los archivos guardados en su memoria.
- Se puede utilizar como un servidor privado dentro de una red local.
- Conectar los puertos del microprocesador desde un conector de 40 pines a distintos circuitos y dispositivos.

2.2.2. Sistema en chip

Un sistema en chip o SoC (del inglés *System on a Chip*) es aquel dispositivo que posee integrados todos o gran parte de los módulos que componen un sistema informático o electrónico en un único circuito integrado o chip. El diseño de estos sistemas puede estar basado en circuitos de señal digital, señal analógica y a menudo módulos o sistemas de radiofrecuencia. Un ámbito común de aplicación de la tecnología SoC son los sistemas embebidos.

Un SoC estándar está constituido por [12]:

[3]Imagen tomada de: https://all3dp.com/2/raspberry-pi-400-vs-raspberry-pi-4-differences/

- Contiene un procesador gráfico integrado, lo que permite la reproducción de vídeo, incluso en alta definición.
- Permite la conexión a la red a través del puerto de Ethernet y algunos modelos permiten conexión Wi-Fi y Bluetooth.
- Consta de una ranura microSD que permite instalar y ejecutar sistemas operativos a través de una tarjeta de memoria.

El diseño de la Raspberry Pi fue evolucionando con el correr del tiempo. En la actualidad se encuentran los modelos Raspberry Pi 4 y Raspberry Pi 400, siendo los más modernos y robustos lanzados por la marca. La familia de Raspberry Pi 4 cuenta con 4 modelos de placa con distinta capacidad de memoria RAM, yendo desde 1 GB hasta 8 GB. La Raspberry Pi 400 es una variante de la Raspberry Pi 4. También existen otros modelos más pequeños diseñados para otro tipo de aplicaciones.

En la figura 2.3 se pueden ver los modelos Raspberry Pi 4 y Raspberry Pi 400.

FIGURA 2.3. Raspberry Pi 4 y Raspberry Pi 400. [3]

Algunas de las funciones y aplicaciones más comunes de este tipo de computadoras se describen a continuación:

- Navegar en la red, utilizar aplicaciones de oficina para la edición de documentos y emplearla como si fuese una computadora de escritorio.
- Crear un centro multimedia y ver los archivos guardados en su memoria.
- Se puede utilizar como un servidor privado dentro de una red local.
- Conectar los puertos del microprocesador desde un conector de 40 pines a distintos circuitos y dispositivos.

2.2.2. Sistema en chip

Un sistema en chip o SoC (del inglés *System on a Chip*) es aquel dispositivo que posee integrados todos o gran parte de los módulos que componen un sistema informático o electrónico en un único circuito integrado o chip. El diseño de estos sistemas puede estar basado en circuitos de señal digital, señal analógica y a menudo módulos o sistemas de radiofrecuencia. Un ámbito común de aplicación de la tecnología SoC son los sistemas embebidos.

[3]Imagen tomada de: https://all3dp.com/2/raspberry-pi-400-vs-raspberry-pi-4-differences/

- Un microcontrolador con el núcleo de la CPU. Algunos son construidos con microprocesadores dotados de varios núcleos.
- Módulos de memoria ROM (memoria de sólo lectura), RAM (memoria de acceso aleatorio), EEPROM (memoria de sólo lectura programable y borrable electrónicamente) y Flash (memorias de acceso muy rápido).
- Generadores de frecuencia fija.
- Componentes periféricos como contadores, temporizadores y relojes en tiempo real o RTC (en inglés *Real Time Clock*).
- Controladores de comunicación con interfaces externas normalmente estándar como USB, Ethernet, UART, o SPI.
- Controladores de interfaces analógicas, incluyendo conversores analógico a digital (ADC) y digital a analógico (DAC).
- Reguladores de voltaje y circuitos de gestión eficaz de la energía.

Familia ESP32

ESP32 es la denominación de una familia de chips SoC de bajo costo y consumo de energía, con tecnología Wi-Fi y Bluetooth de modo dual integrada. Emplea un microprocesador Tensilica Xtensa LX6 en sus variantes de simple y doble núcleo e incluye interruptores de antena, balun de radiofrecuencia, amplificador de potencia, amplificador receptor de bajo ruido, filtros y módulos de administración de energía. El ESP32 fue creado y desarrollado por Espressif Systems.

En la figura 2.4 se pueden ver algunos de los módulos de desarrollo que contienen ESP32.

FIGURA 2.4. Módulos de la familia ESP32. [4]

[4]Imagen tomada de: https://www.electrodaddy.com/esp32/

10 Capítulo 2. Introducción específica

Un SoC estándar está constituido por [12]:

- Un microcontrolador con el núcleo de la CPU. Algunos son construidos con microprocesadores dotados de varios núcleos.
- Módulos de memoria ROM (memoria de sólo lectura), RAM (memoria de acceso aleatorio), EEPROM (memoria de sólo lectura programable y borrable electrónicamente) y Flash (memorias de acceso muy rápido).
- Generadores de frecuencia fija.
- Componentes periféricos como contadores, temporizadores y relojes en tiempo real o RTC (en inglés *Real Time Clock*).
- Controladores de comunicación con interfaces externas normalmente estándar como USB, Ethernet, UART, o SPI.
- Controladores de interfaces analógicas, incluyendo conversores analógico a digital (ADC) y digital a analógico (DAC).
- Reguladores de voltaje y circuitos de gestión eficaz de la energía.

Familia ESP32

ESP32 es la denominación de una familia de chips SoC de bajo costo y consumo de energía, con tecnología Wi-Fi y Bluetooth de modo dual integrada. Emplea un microprocesador Tensilica Xtensa LX6 en sus variantes de simple y doble núcleo e incluye interruptores de antena, balun de radiofrecuencia, amplificador de potencia, amplificador receptor de bajo ruido, filtros y módulos de administración de energía. El ESP32 fue creado y desarrollado por Espressif Systems.

En la figura 2.4 se pueden ver algunos de los módulos de desarrollo que contienen ESP32.

FIGURA 2.4. Módulos de la familia ESP32. [4]

[4]Imagen tomada de: https://www.electrodaddy.com/esp32/

Capítulo 3

Diseño e implementación

En el presente capítulo se presentan los detalles del diseño y los criterios adoptados para el desarrollo del trabajo junto con los pasos seguidos para su implementación.

3.1. Arquitectura del sistema

El sistema tiene una configuración de arquitectura del tipo cliente-servidor. Está constituido por dos nodos y un servidor, los cuales están conectados a la red local y se comunican a través del protocolo MQTT. El servidor recibe los parámetros actuales de estado y cambios desde cada dispositivo, los procesa y almacena en la base de datos. También envía mensajes hacia los dispositivos para cambiar el estado de las salidas o el parámetro que se desee cambiar. Los usuarios pueden consultar y modificar el estado de los dispositivos desde un navegador web móvil o desde una computadora.

En la figura 3.1 se puede observar la arquitectura cliente-servidor del sistema implementado en el trabajo.

FIGURA 3.1. Arquitectura cliente-servidor. [1]

[1]Imagen tomada de: https://www.bujarra.com/raspberry-pi-servidor-vpn-con-pptp/

15

Capítulo 3

Diseño e implementación

En el presente capítulo se presentan los detalles del diseño y los criterios adoptados para el desarrollo del trabajo junto con los pasos seguidos para su implementación.

3.1. Arquitectura del sistema

El sistema tiene una configuración de arquitectura del tipo cliente-servidor. Está constituido por dos nodos y un servidor, los cuales están conectados a la red local y se comunican a través del protocolo MQTT. El servidor recibe los parámetros actuales de estado y cambios desde cada dispositivo, los procesa y almacena en la base de datos. También envía mensajes hacia los dispositivos para cambiar el estado de las salidas o el parámetro que se desee cambiar. Los usuarios pueden consultar y modificar el estado de los dispositivos desde un navegador web móvil o desde una computadora.

En la imagen 3.1 se puede observar la arquitectura cliente-servidor del sistema implementado en el trabajo.

FIGURA 3.1. Arquitectura cliente-servidor. [1]

[1]Imagen tomada de: https://www.bujarra.com/raspberry-pi-servidor-vpn-con-pptp/

Uno de los nodos tiene como función sensar y controlar de temperatura de un recinto, y el otro controlar la iluminación. Originalmente el proyecto estaba pensado para que un solo nodo implemente estas dos funciones, pero durante el desarrollo se optó por la implementación separada. Este cambio se basó en la idea de modularizar los nodos y que sus funciones sean específicas. De esta forma es más amigable para el usuario visualizar y modificar el estado en pantalla e implementar el alta de nuevos dispositivos en el sistema.

El servidor está montado sobre una Raspberry Pi 400 con un sistema operativo Raspbian con interfaz gráfica. Este sistema operativo es la versión oficial ofrecida por la fundación Rasperry Pi y está basado en Debian versión 11 (*bullseye*). En esta etapa de desarrollo se optó por una Raspberry Pi 400 por una cuestión de costos y practicidad a la hora de desarrollar y hacer las pruebas, aunque tiene un mayor volumen que los otros modelos de la familia. Al momento de ofrecer una solución definitiva está pensado que sea implementado en una placa con el formato más pequeño como cualquiera de las Raspberry Pi 4. La conexión del servidor a la red local es por cable Ethernet.

3.1.1. Especificaciones técnicas del servidor

El sistema operativo del servidor está instalado y se ejecuta desde un disco de estado sólido por USB. Este tipo de discos tienen una mayor capacidad de escrituras y lecturas que una memoria microSD, lo que resulta favorable al momento de hacer modificaciones y pruebas de ejecución de software. En la versión final del sistema todo el software estará instalado en una tarjeta microSD para que todo el conjunto sea lo más pequeño posible.

En la tabla 3.1 pueden verse las especificaciones técnicas de hardware más importantes del modelo Raspberry Pi 400.

TABLA 3.1. Especificaciones de Raspberry Pi 400.

Procesador	Broadcom BCM2711 quad-core Cortex-A72(ARM v8) 64-bit SoC @ 1.8GHz
Memoria	4GB LPDDR4-3200
Conectividad	2.4GHz and 5.0GHz 802.11b/g/n/ac wireless LAN Bluetooth 5.0, BLE Gigabit Ethernet
Alimentación	5V DC vía USB-C

3.2. Modelo de datos

En esta sección se describen las diferentes tablas dentro de la base de datos de tipo relacional MariaDB llamada *Domotica*. Con el objetivo de mostrar una representación visual fácil de comprender se muestran las imágenes representadas en la página phpMyadmin. Las tablas que forman dicha base son:

- Dispositivos.
- Usuarios.
- Mediciones.

16 Capítulo 3. Diseño e implementación

Uno de los nodos tiene como función sensar y controlar de temperatura de un recinto, y el otro controlar la iluminación. Originalmente el proyecto estaba pensado para que un solo nodo implemente estas dos funciones, pero durante el desarrollo se optó por la implementación separada. Este cambio se basó en la idea de modularizar los nodos y que sus funciones sean específicas. De esta forma es más amigable para el usuario visualizar y modificar el estado en pantalla e implementar el alta de nuevos dispositivos en el sistema.

El servidor está montado sobre una Raspberry Pi 400 con un sistema operativo Raspbian con interfaz gráfica. Este sistema operativo es la versión oficial ofrecida por la fundación Rasperry Pi y está basado en Debian versión 11 (*bullseye*). En esta etapa de desarrollo se optó por una Raspberry Pi 400 por una cuestión de costos y practicidad a la hora de desarrollar y hacer las pruebas, aunque tiene un mayor volumen que los otros modelos de la familia. Al momento de ofrecer una solución definitiva está pensado que sea implementado en una placa con el formato más pequeño como cualquiera de las Raspberry Pi 4. La conexión del servidor a la red local es por cable Ethernet.

3.1.1. Especificaciones técnicas del servidor

El sistema operativo del servidor está instalado y se ejecuta desde un disco de estado sólido por USB. Este tipo de discos tienen una mayor capacidad de escrituras y lecturas que una memoria microSD, lo que resulta favorable al momento de hacer modificaciones y pruebas de ejecución de software. En la versión final del sistema todo el software estará instalado en una tarjeta microSD para que todo el conjunto sea lo más pequeño posible.

En la tabla 3.1 pueden verse las especificaciones técnicas de hardware más importantes del modelo Raspberry Pi 400.

TABLA 3.1. Especificaciones de Raspberry Pi 400.

Procesador	Broadcom BCM2711 quad-core Cortex-A72(ARM v8) 64-bit SoC @ 1.8GHz
Memoria	4GB LPDDR4-3200
Conectividad	2.4GHz and 5.0GHz 802.11b/g/n/ac wireless LAN Bluetooth 5.0, BLE Gigabit Ethernet
Alimentación	5V DC vía USB-C

3.2. Modelo de datos

3.3. Desarrollo del frontend

3.4. Desarrollo del backend

3.5. Nodos, sensores y actuadores

3.6. Comunicación del sistema

Debe tenerse en cuenta que todo el contenido que se ejecuta del lado del servidor se encuentra dentro de un contenedor Docker. Dicho contenido corresponde a las imágenes de los servicios de Ionic, MariaDB, phpMyAdmin, backend con Node y mosquitto. En la figura 3.2 se observa el contenido del archivo *docker-compose.yml* pertinente a la configuración de los servicios de MariaDB y phpMyAdmin.

FIGURA 3.2. Configuración de Docker de la base de datos. [2]

3.2.1. Tabla Dispositivos

Esta tabla contiene los datos de los dispositivos dados de alta en el sistema. Dichos datos son el ID del dispositivo, el nombre, la ubicación, la dirección MAC, el tipo, el valor de la alarma y el estado de la alarma (0 para desactivada y 1 para activada).

En la figura 3.3 puede verse la tabla cargada con 2 dispositivos funcionando.

dispositivoId	nombre	ubicacion	mac	tipo	alarma	act_al
0028192332001	ESP32+DHT22	Habitacion	94:B5:55:2B:FF:64	Temperatura	26	0
0128192332001	ESP32	Sala	B0:A7:32:DD:18:0C	Luz dimmer	0	0

FIGURA 3.3. Tabla de dispositivos. [3]

El ID del dispositivo es un número brindado por el fabricante del dispositivo que consta de 13 números y está formado por: los primeros 2 dígitos que corresponden al tipo de dispositivo (el sistema está pensado para cubrir hasta un total de 100 tipos de dispositivos, aunque en esta etapa de prototipo sólo se hayan desarrollado 2); 4 dígitos de seguridad fijos que en este caso son "2819z sirven para corroboración y como seguridad; y los últimos 5 dígitos corresponden al número de serie del equipo (2 para el año, 2 para la semana y 3 para el número de fabricación del equipo en esa semana, en ese orden). Como puede observarse, el sistema está pensado para que en un futuro sea parte de una producción en serie.

Capítulo 4

Ensayos y resultados

Tanto el nombre como la ubicación son campos alfanuméricos elegidos por el usuario para describir al dispositivo. La dirección MAC es enviada por el dispositivo la primera vez que se conecta al sistema y no se completa por el usuario. El tipo corresponde al tipo de sensor y para esta etapa del desarrollo puede ser "Temperatura.º "Luz dimmer".

El valor de la alarma es un campo numérico y sólo tiene efecto para dispositivos del tipo de temperatura. Se enviará una notificación por mail cada vez que el valor enviado por el dispositivo sea mayor o igual al seteado en este campo, siempre que dicha alarma esté activada en el campo.

3.2.2. Tabla Usuarios

La tabla de usuarios contiene todos los datos relacionados a aquellas personas que vayan a utilizar el sistema. En esta etapa está implementado con 3 usuarios, ya que al ser de tipo hogareño resultaría difícil que más de 3 personar dentro de una misma casa se registren. Pero con pocos cambios el sistema podría ser escalable a la cantidad de usuarios que se desee, generando una página de alta de usuarios.

Los valores almacenados en esta tabla son el ID de usuario (de 1 a 3), el nombre de usuario, la clave, el nombre de la persona y su apellido, su e-mail y un campo de tipo booleano que se modificará cuando se haya actualizado por primera vez. Cabe aclarar que cada vez que se ingrese al sistema con un usuario que no haya actualizado sus datos, se mostrará en pantalla un aviso para que este actualice sus datos.

En la figura 3.4 puede verse la tabla cargada con los 3 usuarios de los cuales hay 2 actualizados y el tercero tiene los valores por defecto.

userId	user	password	nombre	apellido	email	updated
1	javier	caiot	Javier	Fanelli	javifanelli@gmail.com	1
2	romina	user	Romina	Corrizo	rominamercorrizo@gmail.com	1
3	user3	user	Nombre	Apellido	user3@example.com	0

FIGURA 3.4. Tabla de usuarios. [4]

3.2.3. Tabla Mediciones

La tabla de mediciones contiene todos los datos referentes a las mediciones realizadas por los dispositivos y los datos de modo seleccionado. Los dispositivos reportan cada 5 minutos estos datos y se almacenan en esta tabla.

Los datos almacenados en esta tabla son: el ID de la medición (un valor autoincremental), el ID del dispositivo, el tipo de dispositivo, la fecha y hora de la medición, el valor de la medición, el modo de funcionamiento (manual o automático), el valor de la salida (si es de tipo temperatura es 0 o 100 y corresponde a encendido o apagado, y si es de tipo luz dimmer va de 0 a 100 con saltos de 10), y la hora y minutos de encendido y apagado para el modo automático.

En la figura 3.5 pueden verse algunas de las mediciones dentro de la correspondiente tabla, ordenadas por ID. Cabe aclarar que sólo se ven algunas ya que los dispositivos se encuentran reportando y llenando de datos la base.

medicionId	dispositivoId	tipo	fecha	valor	set_point	modo	salida	hon	mos	hoff	moff
1	0128192332001	Luz dimmer	2023-10-01 21:32:56	0	50	Manual	0	20	0	8	0
2	0028192332001	Temperatura	2023-10-01 21:32:56	21	20	Manual	0	20	0	8	0
3	0128192332001	Luz dimmer	2023-10-01 21:37:53	0	50	Manual	0	20	0	8	0
4	0028192332001	Temperatura	2023-10-01 21:38:02	21	20	Manual	0	20	0	8	0
5	0128192332001	Luz dimmer	2023-10-01 21:42:57	0	50	Manual	0	20	0	8	0
6	0028192332001	Temperatura	2023-10-01 21:43:07	21	20	Manual	0	20	0	8	0
7	0128192332001	Luz dimmer	2023-10-01 21:48:00	0	50	Manual	0	20	0	8	0
8	0028192332001	Temperatura	2023-10-01 21:48:13	21	20	Manual	0	20	0	8	0
9	0128192332001	Luz dimmer	2023-10-01 21:53:03	0	50	Manual	0	20	0	8	0
10	0028192332001	Temperatura	2023-10-01 21:53:18	21	20	Manual	0	20	0	8	0
11	0128192332001	Luz dimmer	2023-10-01 21:58:08	0	50	Manual	0	20	0	8	0
12	0028192332001	Temperatura	2023-10-01 21:58:24	21	20	Manual	0	20	0	8	0
13	0128192332001	Luz dimmer	2023-10-01 22:03:08	0	50	Manual	0	20	0	8	0
14	0028192332001	Temperatura	2023-10-01 22:03:30	21	20	Manual	0	20	0	8	0
15	0128192332001	Luz dimmer	2023-10-01 22:08:13	0	50	Manual	0	20	0	8	0
16	0028192332001	Temperatura	2023-10-01 22:08:35	21	20	Manual	0	20	0	8	0
17	0128192332001	Luz dimmer	2023-10-01 22:13:16	0	50	Manual	0	20	0	8	0
18	0028192332001	Temperatura	2023-10-01 22:13:41	21	20	Manual	0	20	0	8	0
19	0128192332001	Luz dimmer	2023-10-01 22:18:19	0	50	Manual	0	20	0	8	0
20	0028192332001	Temperatura	2023-10-01 22:18:46	21	20	Manual	0	20	0	8	0

FIGURA 3.5. Tabla de mediciones. [5]

3.3. Desarrollo del frontend

El frontend del presente trabajo fue desarrollado en el lenguaje TypeScript con Angular como *framework* integrado con Ionic. El prototipo de aplicación está diseñado para acceder desde un navegador web tanto desde una computadora como un móvil, pero se optó por usar Ionic para un posterior desarrollo de una aplicación para sistemas operativos móviles. Es por esto que en esta instancia puede referirse a la aplicación web como una SPA y no como una PWA.

En la figura 3.6 se puede observar el fragmento de código correspondiente a la configuración de Ionic dentro del archivo *docker-compose.yml*.

```
ionic-ui:
  build:
    context: ./src/frontend/dam
    dockerfile: Dockerfile
  ports:
    - "8100:8100"
  container_name: ionic-ui
  volumes:
    - ./src/frontend/dam:/src/frontend/dam
    - /src/frontend/dam/node_modules
  command: ionic serve --external
```

FIGURA 3.6. Configuración de Docker de Ionic. [6]

La estructura de archivos de cada página esta diseñada de la misma forma como se muestra en la imagen 3.7. Allí se pueden ver como ejemplo los archivos que componen la página *home*.

Además se utilizaron interfaces para definir las estructuras de datos de los dispositivos, las mediciones y los usuarios, y servicios para el proceso de autenticación y de consultas HTTP al backend, que serán explicadas posteriormente.

Capítulo 5

Conclusiones

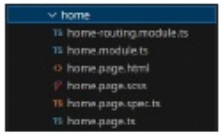

FIGURA 3.7. Estructura de archivos de página. [7]

El servicio de autenticación consta de el ingreso de un usuario y contraseña en la página de *login* que se comparan con los que están almacenados en la base de datos. Si dichos valores ingresados corresponden con alguno de los existentes, se genera un *token* desde el backend que se almacena en el dispositivo que está haciendo la consulta. Esto permite que se puedan acceder a las demás páginas de la aplicación.

En el código 3.1 puede verse el fragmento de la autenticación del archivo *auth.guard.ts*

```
export class AuthGuard {
  constructor(private _loginService: LoginService, private
     _router: Router) {}
  canActivate(
    route: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<boolean | UrlTree> |
       Promise<boolean | UrlTree> | boolean | UrlTree {
    if (!this._loginService.logIn) {
      this._router.navigate(['/login'])
      return false
    }
    return true;
  }
}
```

CÓDIGO 3.1. Autenticación de rutas

El *route guard* es una característica del Angular Router que permite ejecutar algún tipo de lógica cuando se solicita una ruta, y basado en esa lógica, se permite o denega el acceso al usuario a esa ruta. Comúnmente es utilizado para verificar si un usuario está logueado o no en el sistema para verificar si tiene autorización para acceder a esa URL.

El *route guard* se puede agregar implementando la interfaz *CanActivate* disponible en *@angular/router* y allí implementar el método *canActivate()* que contendrá la lógica para denegar o permitir el acceso a la ruta.[22]

En todas las páginas de la aplicación se utilizan los métodos *ngOnInit* y *ngOnDestroy*. Ambos son métodos de Angular en el ciclo de vida de un componente. Sus funciones son la inicialización de variables y configuración de un componente y la liberación de recursos antes que el componente se destruya.[23]

3.3.1. Rutas y páginas destacadas

Las rutas disponibles en la aplicación se describen en la tabla 3.2. A través de ellas se navega dentro de la aplicación y se accede a las distintas pantallas. Estas rutas se encuentran definidas en el archivo *app-routing.module.ts* donde además se hace

referencia al módulo de la aplicación que debe abrirse al acceder a cada una de las rutas listadas.

TABLA 3.2. Rutas de la aplicación

Ruta	Descripción
/login	Pantalla de inicio de sesión
/home	Pantalla principal con funciones y listado de dispositivos
/usuario/:userId	Pantalla de edición de datos de usuario
/ayuda	Pantalla de manual de usuario
/dispositivos/:id	Pantalla de visualización de estado del dispositivo
/medicion/:id	Pantalla de visualización de las mediciones del
/grafico/:id	Pantalla de visualización del gráfico de las mediciones
/config/:id	Pantalla de configuración del dispositivo
/modificar/:id	Pantalla de edición de datos de un dispositivo existente
/agregar	Pantalla para agregar un dispositivo nuevo

En los casos en los que se encuentra el campo *:id*, este valor se completa con el valor de identificador del elemento en cuestión. En el caso de los usuarios, cada uno de ellos tiene un identificador, lo mismo que para los dispositivos. Para las rutas *medicion*, *grafico*, *config* y *modificar* el *:id* al que se hace referencia es el del dispositivo al que se quiere leer o modificar.

Pantalla de *login*

En la figura 3.8 se muestra la página de inicio de sesión en la aplicación.

FIGURA 3.8. Pantalla de *login*. [8]

El sistema trae la posibilidad de configurar 3 usuarios distintos para que utilicen, visualicen y configuren la aplicación. Todos poseen el mismo rol y pueden configurar cualquier campo que corresponda a dicho usuario ingresando en la opción Usuario dentro de la pantalla principal.

Pantalla principal *home*

En la figura 3.9 puede verse la pantalla principal de la aplicación. En la parte superior se encuentran los botones principales de la aplicación tales como el de

Bibliografía

[1] *Domótica ¿Qué es la domótica? ¿Cómo funciona?* 2023. URL: https://e-ficiencia.com/domotica-que-es-y-como-funciona/.

[2] *Domótica - Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/Domotica.

[3] *Matter*. 2023. URL: https://csa-iot.org/all-solutions/matter/.

[4] *Home assistant*. 2023. URL: https://www.home-assistant.io/.

[5] *Domotic*. 2018. URL: https://www.sistemasdomotic.com.ar/.

[6] *Commax*. 2023. URL: http://domotica.com.ar/.

[7] *Reactor*. 2023. URL: https://www.reactor.com.ar/.

[8] *Modelo OSI, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/Modelo_OSI.

[9] *Protocolo HTTP, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto.

[10] *Protocolo MQTT, Goto IoT*. 2021. URL: https://www.gotoiot.com/pages/articles/mqtt_intro/index.html.

[11] *Seguridad de la capa de transporte, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/Seguridad_de_la_capa_de_transporte.

[12] *Sistema en un chip, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/Sistema_en_un_chip.

[13] *Lenguaje de programación C, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/C_(lenguaje_de_programaciÃşn).

[14] *Lenguaje de programación JavaScript, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/JavaScript.

[15] *Lenguaje de programación TypeScript, Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/TypeScript.

[16] *¿Qué es Angular?, Hubspot*. 2022. URL: https://blog.hubspot.es/website/que-es-angular.

[17] *Qué es Ionic: ventajas y desventajas de usarlo para desarrollar apps móviles híbridas, profile*. 2021. URL: https://profile.es/blog/que-es-ionic/.

[18] *¿Cuál es la diferencia entre MariaDB y MySQL?, AWS*. 2023. URL: https://aws.amazon.com/es/compare/the-difference-between-mariadb-vs-mysql/#:~:text=MariaDB%20is%20more%20scalable%20and,multiple%20engines%20in%20one%20table..

[19] *¿Qué son los contenedores?, NetApp*. 2021. URL: https://www.netapp.com/es/devops-solutions/what-are-containers/.

[20] *Docker (software), Wikipedia*. 2023. URL: https://es.wikipedia.org/wiki/Docker_(software).

[21] *Descripción general de Docker Compose, Docker*. 2023. URL: https://docs.docker.com/compose/.