

**CARRERA DE ESPECIALIZACIÓN EN
INTERNET DE LAS COSAS**

MEMORIA DEL TRABAJO FINAL

**Monitoreo y gestión remota de
controladores de clima y riego**

Autor:
Ing. Martín A. Brocca

Director:
Mg. Ing. Juan Carlos Brocca

Jurados:
A ser designado
A ser designado
A ser designado

*Este trabajo fue realizado en la Ciudad Autónoma de Buenos Aires,
entre junio de 2022 y mayo de 2023.*

Resumen

En la presente memoria se describe el diseño, desarrollo e implementación de un sistema de control y monitoreo de clima y riego para un invernadero de tipo hogareño. El trabajo se compone de una colección de sensores y actuadores basados en hardware de bajo costo y una plataforma de software de código abierto que permite automatizar el proceso y adaptarlo a diferentes cultivos.

Para completar este proyecto se utilizaron técnicas de selección e integración de circuitos, desarrollo de firmware, evaluación de software y diseño de automatizaciones.

Índice general

Resumen	I
1. Introducción general	1
1.1. Introducción	1
1.2. Motivación	1
1.3. Estado del arte	2
1.4. Objetivos y alcance	5
2. Introducción específica	7
2.1. Tecnologías de comunicación	7
2.1.1. Protocolo MQTT	7
2.1.2. Protocolo HTTP	8
2.1.3. Protocolo SSL/TLS	8
2.1.4. Tecnologías Wi-Fi	9
2.2. Componentes de hardware utilizado	9
2.2.1. Raspberry Pi	9
2.2.2. Módulos ESP	10
2.2.3. Sensores y actuadores	12
2.3. Tecnologías de software aplicadas	12
2.3.1. ThingsBoard	12
2.3.2. Arduino IDE	13
2.3.3. Telegram	13
2.4. Requerimientos	13
3. Diseño e implementación	15
3.1. Arquitectura de la solución	15
3.1.1. Componentes del sistema	15
3.1.2. Protocolos de comunicación	16
3.2. Detalle de los módulos de hardware	17
3.2.1. Módulos sensores de humedad del suelo	18
3.2.2. Módulo controlador del riego	19
3.2.3. Módulo sensor de temperatura y humedad	20
3.2.4. Módulo controlador de clima	21
3.3. Desarrollo del firmware	23
3.3.1. Módulos sensores de humedad del suelo	23
3.3.2. Módulo controlador del riego	24
3.3.3. Módulo sensor de temperatura y humedad	26
3.3.4. Módulo de control de clima	27
3.4. Selección y configuración del software	27
3.4.1. Servidor de la aplicación	28
3.4.2. Configuración de la aplicación	28
3.4.3. Manejo de dispositivos	29

3.4.4. Reglas de automatización	29
3.4.5. Interfaz de manejo remoto	31
3.5. Ciberseguridad del sistema	31
4. Ensayos y resultados	33
4.1. Banco de pruebas	33
4.2. Pruebas individuales	35
4.2.1. Pruebas a módulos sensores	35
4.2.2. Pruebas a módulos actuadores	36
4.2.3. Pruebas de acceso concurrente a la aplicación	36
4.3. Pruebas de sistema	37
4.3.1. Control de clima	38
4.3.2. Control de riego	39
4.4. Comparativa con el estado de arte	41
5. Conclusiones	43
5.1. Resultados obtenidos	43
5.2. Trabajo futuro	43
Bibliografía	45

Índice de figuras

1.1. Invernadero hogareño	2
1.2. Invernadero inteligente controlado por IoT	3
1.3. Sistema de control de irrigación de Growlink	4
1.4. Sistema multisensor de Grodan	4
2.1. Ubicación de los protocolos de IoT en la pila TCP/IP	7
2.2. Arquitectura del protocolo MQTT	8
2.3. Proceso de autenticación de TLS	9
2.4. Raspberry Pi	10
2.5. Módulos de desarrollo ESP empleados en el proyecto	11
2.6. Principales sensores y actuadores empleados en el invernadero . .	12
3.1. Arquitectura del sistema.	16
3.2. Protocolos de comunicación entre módulos.	17
3.3. Esquema de conexión de módulos sensores de humedad del suelo .	18
3.4. Módulos de sensores de humedad del suelo empleados en el pro- yecto	19
3.5. Conexión del módulo de control de riego	20
3.6. Módulo de control de riego	20
3.7. Conexión del sensor de temperatura y humedad	21
3.8. Módulo completo en su caja protectora	21
3.9. Conexión del módulo de control de clima	22
3.10. Módulo de control de clima	22
3.11. Diagrama de flujo del firmware de los módulos sensores de hume- dad del suelo	24
3.12. Diagrama de flujo del firmware del módulo de control de riego . .	25
3.13. Diagrama de flujo del firmware del módulo de control de riego - control de válvulas	26
3.14. Diagrama de flujo del firmware del módulo de control de riego - control de bomba	26
3.15. Diagrama de flujo del firmware del módulo sensor de temperatura y humedad	27
3.16. Diagrama de flujo del firmware del módulo de control de clima . .	28
3.17. Pantallas de manejo de dispositivos creados en el proyecto	29
3.18. Fragmento de regla de automatización para el control de riego . .	30
3.19. Uso del <i>bot</i> de Telegram	31
4.1. Modelo completo del invernadero	33
4.2. Modelo de pruebas del invernadero	34
4.3. Dashboard de visualización de telemetría	35
4.4. Atributos de sensores de humedad del suelo	36
4.5. Prueba de resistencia al agua	36
4.6. Pruebas unitarias sobre controladores	37

4.7. Pruebas unitarias sobre acceso concurrente	37
4.8. Regla de automatización genérica	38
4.9. Regla de control de clima	38
4.10. Generación de alarmas por exceso de temperatura	39
4.11. Prueba del sistema de riego	40

Índice de tablas

1.1.	Análisis del estado del arte	4
2.1.	Especificaciones técnicas de la Raspberry Pi 4B	10
2.2.	Especificaciones de los microcontroladores ESP	11
4.1.	Pruebas de sistema de riego	40

Capítulo 1

Introducción general

En este capítulo se introducen conceptos asociados al uso de Internet de las Cosas (IoT) en invernaderos, la motivación del cliente y cuál es el estado del arte. Asimismo, se explican los objetivos y el alcance establecidos.

1.1. Introducción

El término Internet de las Cosas (IoT) se refiere a escenarios en los que la conectividad de red y la capacidad de cómputo se extienden a objetos, sensores y artículos de uso diario que habitualmente no se consideran computadoras, permitiendo que estos dispositivos generen, intercambien y consuman datos con una mínima intervención humana [1]. IoT tiene una amplia variedad de campos de aplicación, entre los cuales se destaca la agricultura inteligente aplicada a invernaderos.

Los invernaderos modernos son sistemas de cultivo intensivo diseñados para alcanzar una alta eficiencia y productividad. Debido a su capacidad para mantener las condiciones ambientales en niveles óptimos o subóptimos, facilitan la producción de plantas a lo largo de todo el año en forma independiente a las condiciones climáticas externas [2].

La aplicación de IoT a invernaderos ha demostrado una mejora sustancial en la eficiencia de la gestión de los cultivos al mismo tiempo que ha logrado acelerar la producción y reducir sus costos [3]. Además de las ventajas mencionadas, los invernaderos pueden impactar de forma positiva a los entusiastas de la jardinería, ya que proporcionan beneficios tanto físicos como anímicos especialmente durante las temporadas invernales o de baja temperatura [4].

1.2. Motivación

El cliente de este proyecto es un jubilado reciente que decidió comenzar una nueva vida en una finca rural. Su intención es convertir la propiedad en una residencia sustentable, capaz de producir diferentes tipos de plantas: por un lado, hortalizas para abastecer el consumo familiar y por otro, especies de árboles para reforestación.

Para lograrlo con mínima intervención humana en el proceso, surge la necesidad de instalar un invernadero que automatice el control de los cultivos y además tenga la capacidad de adaptar las condiciones climáticas y de riego a los diversos tipos de plantas a sembrar.

En la figura 1.1 se observa un vivero hogareño típico de producción y dimensiones similares al propuesto.



FIGURA 1.1. Invernadero hogareño¹.

1.3. Estado del arte

Los invernaderos son estructuras diseñadas para controlar y proteger a las plantas del clima y otros factores ambientales adversos. Tradicionalmente, la temperatura, humedad e iluminación se controlaban de forma manual, lo que requería una gran cantidad de mano de obra y recursos.

Sin embargo, debido a los avances tecnológicos se ha popularizado el desarrollo de invernaderos capaces de ajustar las condiciones ambientales mediante el uso de sensores, actuadores y controladores. Estos dispositivos responden en función de configuraciones preprogramadas o a partir de datos en tiempo real.

Así el despliegue de este tipo de sistemas, denominados invernaderos inteligentes, se ha extendido enormemente en los últimos años debido a la eficiencia obtenida durante la producción y al incremento en la resiliencia de los cultivos [5].

Algunas posibles aplicaciones de IoT en viveros incluyen:

- Monitoreo y control del clima: distintos sensores miden la temperatura, humedad, iluminación y otros factores ambientales en el invernadero, para que luego diferentes actuadores automáticos ajusten el clima y creen las condiciones óptimas para el crecimiento de las plantas.
- Riego a demanda: la medición continua de la humedad del suelo permite activar sistemas de riego automatizados para mantener los niveles óptimos de humedad. Se logra así reducir el desperdicio de agua y disminuir los costos asociados.
- Seguimiento del crecimiento de las plantas: los sensores IoT pueden medir el crecimiento de las plantas y proporcionar información útil para la gestión del cultivo. Esto puede ayudar a identificar problemas de crecimiento temprano y a tomar medidas que eviten problemas mayores a futuro.

¹Imagen bajo licencia de <https://www.istockphoto.com/>

- Control de plagas y enfermedades: monitorear sus niveles en el vivero y activar sistemas de control cuando se detecten problemas ayuda a reducir el uso de pesticidas y otros productos químicos.
- Fertirrigación: el sistema puede administrar fertilizantes o nutrientes al suelo de forma optimizada y precisa a través del riego, en base a configuraciones acordes a la plantación en curso o mediante sensores que midan las características del agua, el pH o la conductividad eléctrica entre otras.
- Automatización de tareas: los sistemas IoT pueden automatizar muchas tareas en el vivero, como la siembra, el trasplante y la recolección de plantas. Esto puede reducir los costos de mano de obra y mejorar la eficiencia de la producción.

En la figura 1.2 se representa un invernadero inteligente con sus respectivos sensores y actuadores.



FIGURA 1.2. Invernadero inteligente controlado por IoT².

En el mercado internacional se encuentran diferentes proveedores que ofrecen soluciones para el desarrollo de invernaderos inteligentes. A la hora de comparar distintas opciones es necesario considerar los niveles de automatización requeridos, la facilidad de uso y las opciones de personalización. Adicionalmente, se debe considerar el costo y la compatibilidad con la infraestructura existente.

En la tabla 1.1 se observa una breve comparación entre los principales proveedores comerciales de servicios. Allí se observa que en general las soluciones presentadas ofrecen características similares siendo el costo el mayor diferenciador.

²Imagen adaptada de *Internet of Things Empowered Smart Greenhouse Farming* [6].

TABLA 1.1. Análisis del estado del arte.

Funcionalidad	Argus Controls [7]	Grodan [8]	Growlink [9]
Gestión de clima	Sí	Sí	Sí
Control de riego	Sí	Sí	Sí
Fertirrigación	Sí	Sí	Sí
Gestión de energía	No	Sí	Sí
Tamaño de mercado	Grande	Grande	Pequeño
Costo	\$\$\$\$	\$\$\$	\$\$

A manera de ejemplo, algunas soluciones comerciales ofrecen kits que pueden costar más de USD 200 por sensor, con gastos adicionales asociados al transporte y almacenamiento de datos. De esta forma, las redes inalámbricas de sensores pueden llegar a requerir presupuestos mayores a USD 10 000 por invernadero [10]. Las figuras 1.3 y 1.4 muestran diferentes soluciones que se encuentran en el mercado.

FIGURA 1.3. Sistema de control de irrigación de Growlink³.FIGURA 1.4. Sistema multisensor de Grodan⁴.

Una solución alternativa es utilizar kits de IoT como los provistos por Arduino

³Imagen bajo licencia de <https://www.growlink.com/>

⁴Imagen bajo licencia de <https://www.grodan.com/>

[11], una plataforma de hardware y software de código abierto. Además, es necesario contar con una comunidad de usuarios que colaboren en el proceso de desarrollo. En contrapartida, para aplicar esta solución se requieren conocimientos de programación y electrónica no siempre disponibles para el cultivador promedio [10].

1.4. Objetivos y alcance

El propósito de este trabajo es el desarrollo de una plataforma capaz de controlar el clima y riego de un invernadero mediante el uso de sensores y actuadores. Estos dispositivos se comunican con una aplicación instalada en un servidor local que administra los parámetros y las alarmas del sistema.

Durante el proyecto se construyó un prototipo completo de invernadero con los siguientes elementos:

- Aplicación para el monitoreo, control de dispositivos, gestión de alarmas y automatización.
- Control de usuarios, permisos y accesos a la plataforma.
- Interfaz gráfica para acceso y control de la plataforma.
- Análisis, investigación y elección del hardware para los sensores y actuadores.

El trabajo no incluyó:

- Instalación en sitio de los sistemas desarrollados.
- Implementación de métodos de control basados en condiciones climatológicas externas.
- Desarrollo o implementación de modelos analíticos o predictivos de las condiciones del vivero.
- Diseño o instalación de conexiones que no sean por Wi-Fi (LTE/5G).

Capítulo 2

Introducción específica

En este capítulo se describen las tecnologías y los protocolos de comunicación, los componentes de hardware, las herramientas de software y los requerimientos para la realización del proyecto.

2.1. Tecnologías de comunicación

Los principales protocolos de IoT empleados en el trabajo son: MQTT [12], HTTP [13], SSL/TLS [14] e IEEE 802.11 [15]. Según el modelo TCP/IP [16] los tres primeros se encuentran en la capa de aplicación mientras que el cuarto se ubica en la capa de enlace. Esto se esquematiza en la figura 2.1.

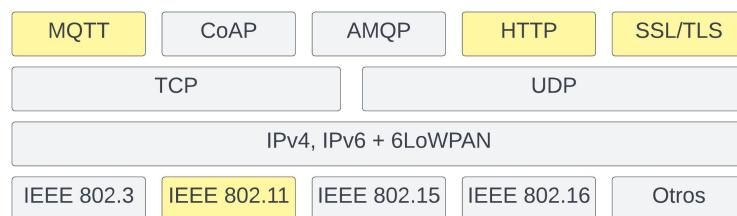


FIGURA 2.1. Ubicación de los protocolos de IoT en la pila TCP/IP.

2.1.1. Protocolo MQTT

MQTT fue desarrollado en 1999 con el objetivo principal de crear un protocolo muy eficiente desde el punto de vista del uso del ancho de banda y de muy bajo consumo de energía. Por estas razones es adecuado para el uso en IoT [17].

Su funcionamiento se basa en el paradigma de publicación-suscripción [18], el cual consiste en desvincular un cliente que publica un mensaje (publicador) de otros clientes que reciben el mensaje (suscriptores). Otra característica importante a mencionar es que, al ser un protocolo asincrónico, el cliente puede seguir operando mientras espera un nuevo mensaje.

Un componente principal del protocolo es el *broker* cuya función primaria es la de recibir los mensajes de los publicadores y enviarlos a los suscriptores. Para realizar esta tarea, el *broker* utiliza temas o *topics* para agrupar clientes que necesitan recibir los mismos mensajes. De esta manera el *topic* es un canal virtual que conecta a los publicadores con sus suscriptores [17].

En la figura 2.2 se observa la arquitectura del protocolo.

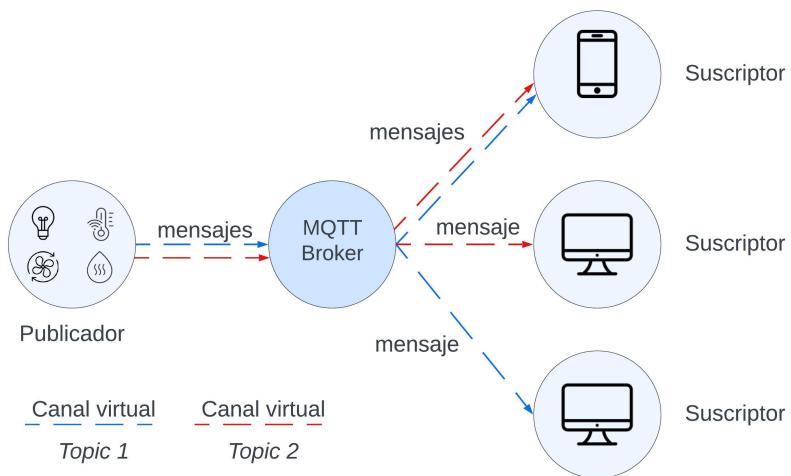


FIGURA 2.2. Arquitectura del protocolo MQTT.

2.1.2. Protocolo HTTP

El *Hypertext Transfer Protocol* (HTTP)[\[19\]](#) es un protocolo utilizado en la Web para el desarrollo de aplicaciones y está basado en el paradigma cliente-servidor. Aquí el cliente emplea un agente intermediario (por lo general un *browser*) para realizar un pedido de información y el servidor proporciona una respuesta. Esto se conoce con el nombre de modelo *request/response*.

HTTP es un protocolo que no guarda información de estado, esto significa que el servidor no es capaz de reconocer la relación entre múltiples pedidos de un mismo usuario [\[20\]](#).

En la actualidad, HTTP se utiliza en conjunto con la arquitectura REST (*Representational State Transfer*) [\[21\]](#) para facilitar la interacción entre distintas entidades sobre servicios basados en red. Esta asociación permite que los dispositivos interactúen mediante funciones estándares de tipo CRUD (*create, read, update, delete*) [\[22\]](#). Dichas funciones a su vez se traducen en los métodos HTTP POST, GET, PUT y DELETE respectivamente [\[23\]](#).

2.1.3. Protocolo SSL/TLS

Secure Socket Layer/Transport Layer Security (SSL/TLS) es un protocolo criptográfico que proporciona seguridad de extremo a extremo de los datos enviados entre aplicaciones a través de Internet. TLS evolucionó a partir de *Secure Socket Layer* (SSL), que fue desarrollado originalmente por Netscape Communications Corporation en 1994 para proteger las sesiones web.

Cabe señalar que TLS no protege los datos en los sistemas finales, simplemente garantiza la entrega segura de datos a través de Internet y al mismo tiempo evita posibles escuchas y/o alteraciones del contenido. TLS normalmente se implementa sobre TCP [\[24\]](#) para cifrar los protocolos de la capa de aplicación, como por ejemplo HTTP.

TLS utiliza una combinación de criptografía simétrica y asimétrica que proporciona un buen compromiso entre rendimiento y seguridad al momento de transmitir

la información [25]. Para mayor protección es deseable que un cliente que se conecta a un servidor pueda validar la veracidad de la clave pública ofrecida por este. Normalmente dicha verificación se lleva a cabo por medio de un certificado digital X.509 [26] emitido por un tercero de confianza denominado Autoridad Certificadora (CA). Dicha CA está encargada de afirmar la autenticidad de la clave pública. En los casos en los que no se dispone de una CA, un servidor puede usar un certificado autofirmado en el que el cliente debe confiar explícitamente [25].

En la figura 2.3 se detalla el esquema de autenticación y verificación con certificados e inicio de una conexión segura.



FIGURA 2.3. Proceso de autenticación de TLS¹.

2.1.4. Tecnologías Wi-Fi

El estándar IEEE 802.11 para redes inalámbricas de área local (WLAN) es conocido comercialmente como Wi-Fi y presenta dos modos de operación [27]:

- Infraestructura: uno o más *access points* (AP) actúan como puente entre la red cableada y la red inalámbrica. Todas las comunicaciones entre los dispositivos conectados a la red se realizan a través de los APs.
- Ad-hoc: cada nodo puede realizar una conexión directa con otro, sin necesidad de un AP central. En este caso, los nodos se organizan en una red donde todos son capaces de enrutar los paquetes.

2.2. Componentes de hardware utilizado

2.2.1. Raspberry Pi

Se denomina así a una serie de computadoras monoplaca o computadoras de placa simple (SBC, por *Single Board Computer*) de bajo costo desarrolladas por la Raspberry Pi Foundation [28]. Una de sus principales características es proveer

¹Gráfico creado en base a una imagen tomada de <http://www.herongyang.com/PKI/HTTPS-Communication-Data-Encryption.html>

un conjunto de pines de GPIO (*general purpose input/output*) que permiten controlar componentes electrónicos y otros dispositivos en el ámbito de Internet de las Cosas. A pesar de su reducido tamaño, la Raspberry Pi ofrece una capacidad de procesamiento comparable a una computadora de escritorio y es por ello que su uso se ha expandido en proyectos que incluyen domótica, *edge computing* y aplicaciones industriales [29].

En la figura 2.4 se muestra una Raspberry Pi modelo 4B similar a la utilizada en el trabajo y en la tabla 2.1 se listan sus principales características:

TABLA 2.1. Especificaciones técnicas de la Raspberry Pi 4B.

Categoría	Especificación[30]
Procesador	Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64 bits SoC @ 1,8 GHz
Memoria SDRAM	1, 2, 4 u 8 GB LPDDR4-3200
Wi-Fi	2,4 GHz y 5,0 GHz IEEE 802.11ac
Bluetooth	5.0 y BLE
Ethernet	Gigabit, con soporte opcional para POE
USB	2 puertos 3.0 y 2 puertos 2.0
GPIO	Conector de 40 pines
HDMI	2 puertos micro-HDMI
Alimentación	5 V USB y GPIO
Temperatura de operación	0 °C a 50 °C



FIGURA 2.4. Raspberry Pi².

2.2.2. Módulos ESP

ESP es una familia de microcontroladores de baja potencia desarrollada por la empresa china Espressif. Estos chips cuentan con una amplia variedad de usos

²Imagen tomada de <https://datasheets.raspberrypi.com/>.

en IoT tanto en el ámbito profesional o industrial como en el de los aficionados [31] [32].

En la tabla 2.2 se observa una comparación entre los modelos ESP32-WROOM-32 y ESP8266 utilizados en este trabajo.

TABLA 2.2. Especificaciones de los microcontroladores ESP.

Características	ESP32 [31]	ESP8266 [32]
Procesador	2 x Xtensa 32 bits LX6	Tensilica L106 32 bits
Memoria ROM	448 KB	No dispone
Memoria SRAM	520 KB	160 KB
Wi-Fi	802.11 b/g/n	802.11 b/g/n
Bluetooth	4.2, BLE	No dispone
GPIOs	34 pines	17 pines
Consumo normal	~500 mA	~80 mA
Temperatura de operación	-40 °C a 85 °C	-40 °C a 125 °C
Sistema operativo	freeRTOS	freeRTOS
Costo en USD	\$ 6 a \$ 12	\$ 3 a \$ 6

El modelo ESP8266 es una versión anterior y con menores prestaciones que el ESP32. Constituye una opción económica para soluciones en las que los requerimientos de conectividad, cantidad de periféricos y demandas computacionales no son tan exigentes.

En la figura 2.5a se observa un módulo de desarrollo de la familia ESP32, mientras que el ESP8266 puede verse en la figura 2.5b.



(A) Módulo de desarrollo ESP32.



(B) Módulo de desarrollo ESP8266.

FIGURA 2.5. Módulos de desarrollo ESP empleados en el proyecto.

2.2.3. Sensores y actuadores

Los principales sensores y actuadores empleados en el sistema del invernadero inteligente son los siguientes:

- DHT22: módulo básico y económico para determinar los valores de temperatura y humedad en forma digital. Utiliza un sensor de humedad capacitivo y un termistor para medir el aire circundante y entrega una señal digital en el pin de datos de acuerdo al valor calculado. Funciona con una alimentación de 3,3 a 6 VDC y su rango de medición es de -40 °C a 80 °C y de 0 a 100 % de humedad relativa [33].
- Sensor capacitivo de humedad del suelo: módulo analógico compuesto de un material resistente a la corrosión que mide la humedad del suelo indirectamente por medio de la capacitancia observada. Opera con una alimentación de 3,3 a 5,5 VDC y entrega un valor de tensión que varía entre 0 V para un suelo seco a aproximadamente 3,15 V en un suelo completamente húmedo [34].
- Válvula solenoide de dos vías: dispositivo neumático para controlar el flujo de líquidos o gases que se acciona eléctricamente. Para poder operarlo, se utiliza un relé que es un instrumento electromecánico que actúa como interruptor controlado por un circuito eléctrico [35][36].

La figura 2.6 muestra imágenes de los componentes listados previamente.



FIGURA 2.6. Principales sensores y actuadores empleados en el invernadero.

2.3. Tecnologías de software aplicadas

2.3.1. ThingsBoard

ThingsBoard es una plataforma de código abierto que permite el desarrollo, administración y expansión de proyectos de IoT. Esta aplicación permite la gestión de las comunicaciones, el almacenamiento y la visualización de los datos que provienen de los sensores u otros dispositivos que forman parte del sistema [37]. Se ofrece en las siguientes versiones:

- Professional Edition: es la versión comercial, con varios tipos de licenciamiento y costos según sea el sistema a implementar. Esta edición posee soporte técnico, ilimitada cantidad de dispositivos a conectar y soporte para almacenamiento híbrido entre otras funcionalidades.

- Cloud: similar a la edición profesional, pero alojada en la nube de Things-Board. En este caso el proveedor se encarga del manejo de los componentes de la plataforma.
- Community edition: se encuentra bajo licencia Apache 2.0 [38] y es la que se utilizó en este trabajo. Si bien no tiene limitaciones en cuanto a la cantidad de dispositivos a conectar, carece de ciertas funcionalidades como por ejemplo la descarga de datos de dispositivos desde la interfaz web o la ejecución programada de tareas (*scheduler*).

2.3.2. Arduino IDE

Arduino IDE (*Integrated Development Environment*) es un software de código abierto que se utiliza para escribir y cargar código a placas Arduino. Sin embargo por medio de la instalación de paquetes de expansión, el IDE soporta hardware de terceros entre los que encuentran las módulos ESP32/ESP8266 [39] empleados en este proyecto.

El código de los programas se realiza en lenguaje C o C++ y los archivos resultantes se denominan *sketches*. Estos son compilados y cargados en las placas desde el mismo IDE [40].

Este software es una herramienta fácil de utilizar tanto por usuarios experimentados como por principiantes. Es frecuentemente empleada por aquellos que se inician en la programación electrónica y la robótica o al momento de construir prototipos interactivos [41].

2.3.3. Telegram

Telegram es una aplicación de mensajería multiplataforma rápida, simple y gratuita. Está basada en la nube y cuenta con sincronización constante, lo que significa que se puede acceder a los mensajes desde diferentes dispositivos simultáneamente.

Tanto el código de los clientes de Telegram como el de su API tienen licencias abiertas, lo que permite crear otras aplicaciones a partir de ellos. Adicionalmente cuenta con una API para *bots* que facilita la implementación de herramientas basadas en Telegram, la integración de servicios o la realización pagos [42].

2.4. Requerimientos

A continuación se listan los principales requerimientos funcionales, no funcionales y de documentación del proyecto:

- Requerimientos funcionales:
 1. El estado del sistema podrá ser consultado desde Internet.
 2. La aplicación soportará múltiples usuarios de forma concurrente.
 3. La aplicación permitirá crear roles de usuarios con diferentes permisos.
- Requerimientos no funcionales:
 1. El rango de tensión de alimentación de los nodos será de 3,3 a 5 VDC.

2. El sistema de riego operará con una tensión de alimentación de 12 VDC.
 3. El sistema estará basado en software de código abierto.
 4. El firmware deberá desarrollarse en plataformas de código abierto.
 5. El trabajo se realizará sobre dispositivos de bajo costo y fácil reposición.
 6. Los datos se almacenarán localmente.
 7. La aplicación soportará MQTT.
 8. Los sensores de humedad del suelo tendrán una protección IP65 [43].
- Requerimientos de documentación:
 1. Los manuales y/o guías estarán redactados en inglés.

Capítulo 3

Diseño e implementación

En este capítulo se presentan los detalles del diseño de los nodos sensores y actuadores que conforman el trabajo, como así también los del software seleccionado.

3.1. Arquitectura de la solución

Para la implementación del prototipo propuesto en el trabajo se requirió la construcción de diferentes subsistemas encargados de las múltiples funciones dentro del invernadero inteligente. Cada uno de ellos opera en forma independiente del resto y todos se comunican con una aplicación central mediante una red inalámbrica. Para garantizar el acceso de los usuarios desde Internet se desarrolló una interfaz de acceso remoto.

3.1.1. Componentes del sistema

En la figura 3.1 se observa el diagrama en bloques de la arquitectura diseñada, que está compuesta por los siguientes elementos:

- Sistema de monitoreo y control de clima, formado por dos módulos:
 - Sensores de temperatura y humedad con sus correspondientes microcontroladores.
 - Unidad de control de temperatura y humedad comprendida por un microcontrolador, relé y ventiladores.

Los sensores miden la temperatura y humedad ambiente en el invernadero y envían estos valores a la aplicación central por medio del microcontrolador. De acuerdo con los datos recibidos, la aplicación determina si es necesario emitir una señal para que la unidad de control encienda los ventiladores.

- Sistema de control de riego, dividido en dos partes:
 - Conjunto de sensores de humedad de suelo con sus respectivos microcontroladores.
 - Unidad de control de riego constituida por un microcontrolador, relés, bomba de agua y válvulas.

Los sensores envían las mediciones a la aplicación central que se encarga de procesarlas. En caso de ser necesario, se disparan las señales de encendido a través de la unidad de control. Primero se activa la válvula que corresponda

y luego se enciende la bomba de agua para comenzar el riego. El orden de estas actividades es importante para evitar daños en la bomba o las cañerías.

- Aplicación central: constituye el cerebro del invernadero y es la encargada de almacenar los parámetros de configuración de los diversos sensores y actuadores, procesar los mensajes recibidos, disparar acciones y alertas y visualizar el estado general.
- Sistema de acceso remoto: permite a los usuarios obtener reportes del sistema en forma segura desde Internet.

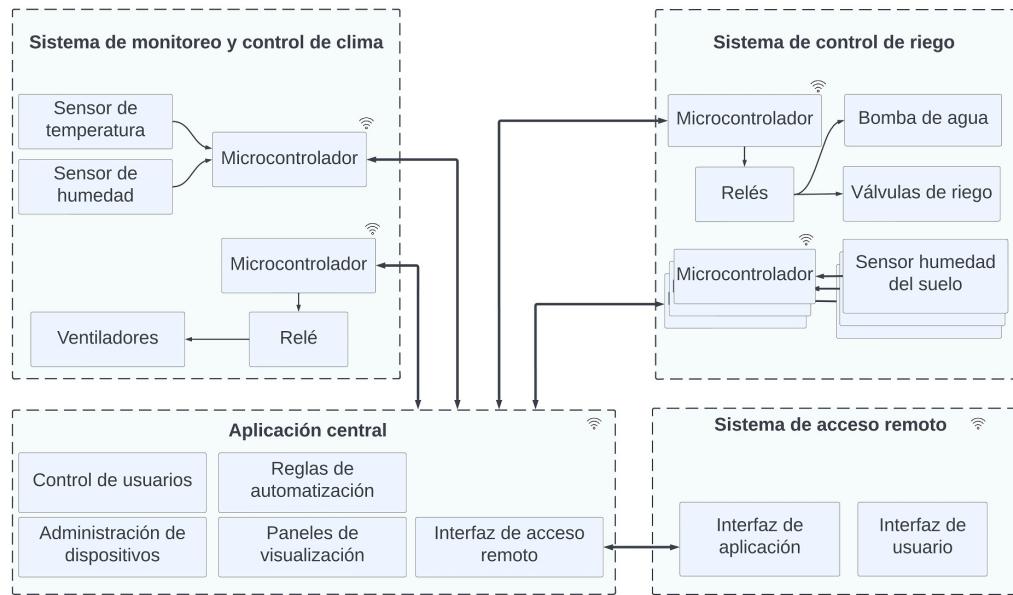


FIGURA 3.1. Arquitectura del sistema.

3.1.2. Protocolos de comunicación

En esta sección se describe cómo se comunican los sistemas con la aplicación central y los protocolos usados en cada caso. En la figura 3.2 se aprecia un esquema simplificado que ilustra dichas interacciones.

Si bien los módulos de hardware y el software soportan una gran variedad de protocolos, se implementó MQTT en la mayoría de los casos conforme a los requerimientos. En algunas situaciones donde esto no fue técnicamente posible, se utilizó HTTP. Adicionalmente, para garantizar la seguridad de las comunicaciones, se incorporó un certificado autofirmado TLS en el servidor.

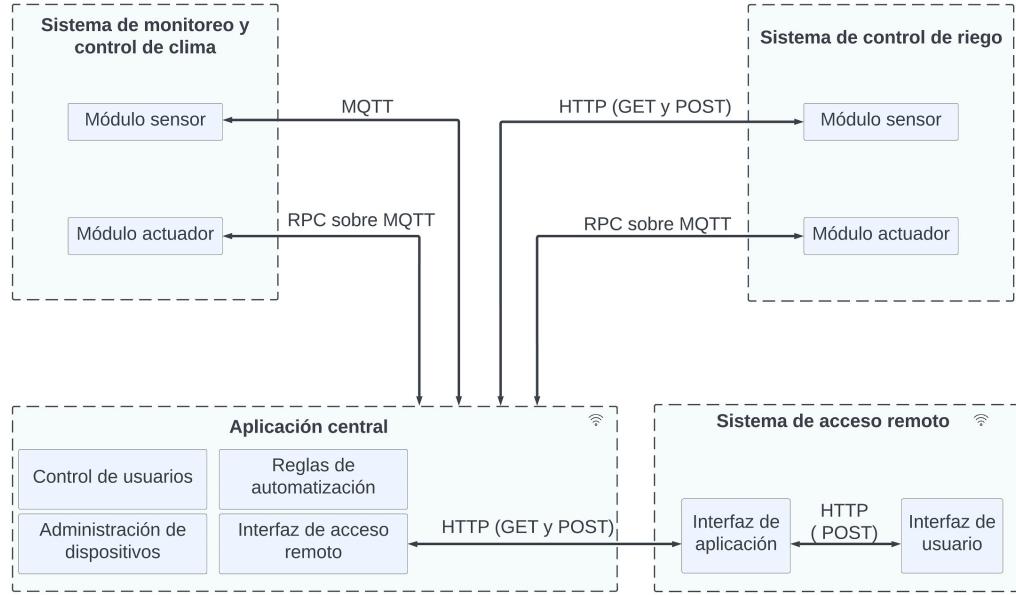


FIGURA 3.2. Protocolos de comunicación entre módulos.

Las interacciones principales entre los bloques componentes son:

- Sistema de monitoreo y control de clima: las comunicaciones se realizan exclusivamente con la aplicación central. El módulo sensor envía las mediciones realizadas por medio de MQTT y en caso de requerir una acción, la aplicación central comanda el encendido de los ventiladores por medio de un mensaje enviado por RPC [13] sobre MQTT.
- Sistema de control de riego: las comunicaciones se realizan con la aplicación central de manera bidireccional. El módulo sensor efectúa dos conexiones, una para el envío de las mediciones y otra para recibir valores de atributos tales como la duración del tiempo de riego. Debido a limitaciones en la configuración de la persistencia de los mensajes en las colas de MQTT, se optó por utilizar llamadas HTTP (GET y POST) para realizarlas. Al igual que en el control de clima, la aplicación central inicia el riego por medio de mensajes RPC sobre MQTT hacia el controlador de la bomba y de las válvulas.
- Sistema de acceso remoto: la interfaz se comunica con la aplicación por medio de pedidos HTTP GET y POST para consultar el reporte de estado de los diferentes componentes. A continuación este se envía hacia la interfaz de usuario por medio de una solicitud HTTP POST.

3.2. Detalle de los módulos de hardware

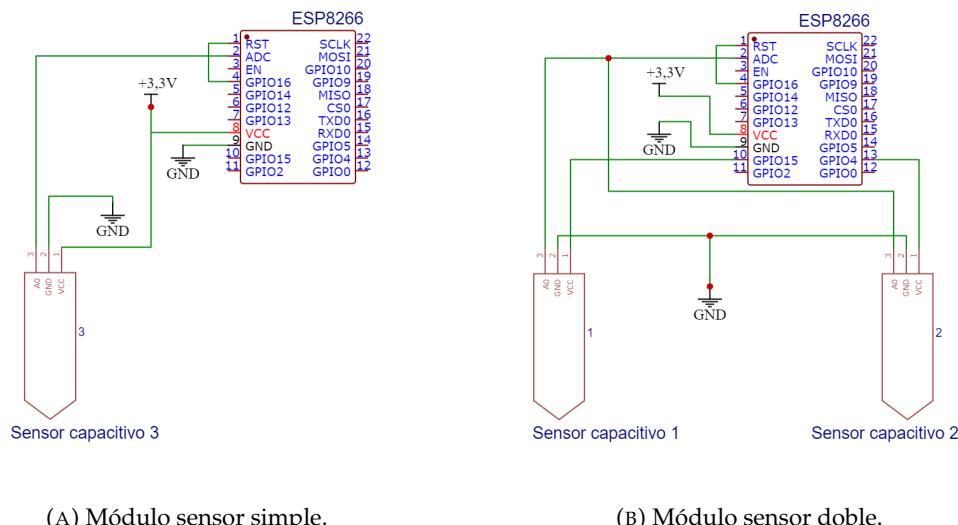
En esta sección se describen en detalle los esquemas de conexión de los distintos módulos y las consideraciones de diseño y construcción empleadas.

3.2.1. Módulos sensores de humedad del suelo

En el proyecto se desarrollaron dos configuraciones diferentes de módulos para medir la humedad del suelo en macetas de diverso tipo y tamaño. Ambas opciones utilizan el microcontrolador ESP8266, pero incorporan distintas cantidades de sensores. En la figura 3.3a se muestra el esquema de conexión para la versión simple (con un único sensor) y en la figura 3.3b se ilustra la configuración doble.

Si bien en el prototipo los sensores se conectaron a una fuente de alimentación, la configuración y conexión del sistema está optimizada para el uso de baterías. Esto se debe a que las sondas pueden estar desplegadas en múltiples ubicaciones dentro del invernadero y no siempre es posible conectarlas a la red eléctrica.

El ahorro de energía necesario para permitir el uso de baterías se logra a través de ciclos de apagado en los períodos donde no se realizan lecturas. A este mecanismo se lo que conoce como *deep sleep* y una vez que el microcontrolador se encuentra en este estado, se requiere un pulso eléctrico en el pin de *reset* para que retorne al modo activo. Para habilitar la reactivación se interconectan los pines D0 (GPIO16) y RST del chip ESP8266. Se logra así una reducción del consumo de energía a valores muy pequeños que rondan los 0,3 mA durante el período de inactividad.



(A) Módulo sensor simple.

(B) Módulo sensor doble.

FIGURA 3.3. Esquema de conexión de módulos sensores de humedad del suelo.

La integración de los componentes de los sensores se realizó en forma manual por medio de placas de circuitos impresos (PCB) experimentales. En las figuras 3.4a y 3.4b se muestran los componentes empleados y un módulo ensamblado.

Dado que los sensores están expuestos a salpicaduras, se los recubrió con tubos adhesivos termocontraíbles que, al aplicarles calor, generan una protección a prueba de agua. Para el resguardo del chip ESP8266 se utilizó una caja de polipropileno transparente sellada. En las figuras 3.4c y 3.4d se muestran los componentes y sus protecciones.

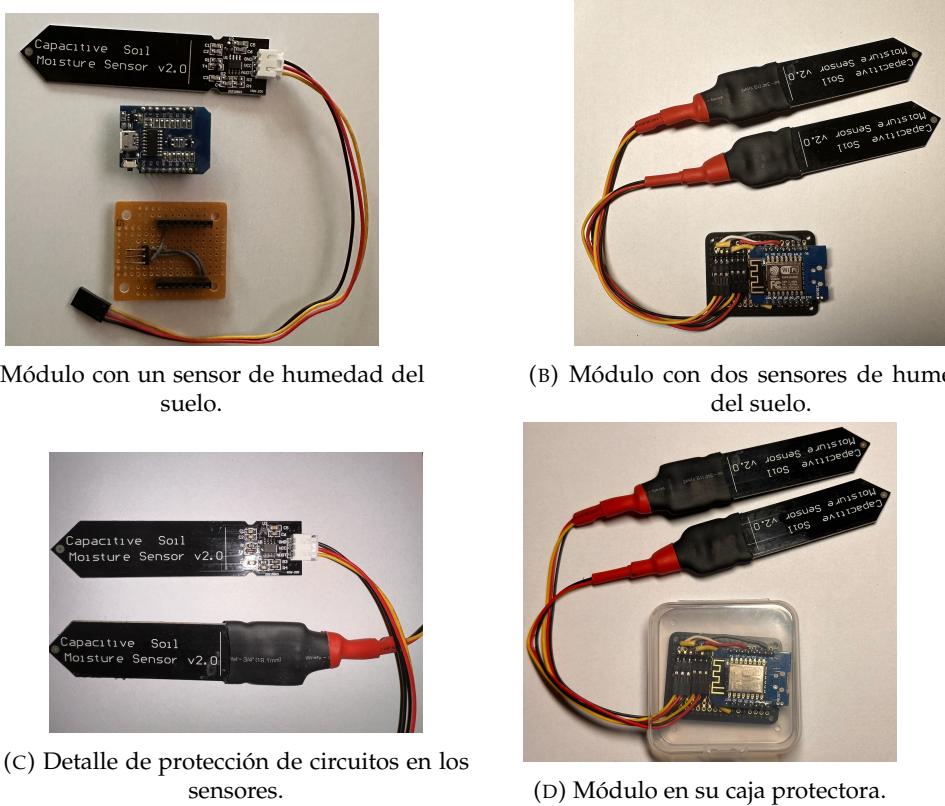


FIGURA 3.4. Módulo de sensores de humedad del suelo empleados en el proyecto.

3.2.2. Módulo controlador del riego

Se compone de un microcontrolador ESP32, una placa de interfaz de relé de cuatro canales, una pantalla LCD/OLED SSH1106 y un regulador de voltaje DC-DC *step down* LM2596. El esquema de conexiones entre estos componentes se detalla en la figura 3.5.

El módulo se alimenta con una fuente de 12 VDC y para energizar a los circuitos electrónicos el regulador LM2596 reduce la tensión a 5 VDC.

Para la construcción del prototipo se realizó la integración del microcontrolador con la pantalla LCD mediante una placa PCB experimental. Tanto para el módulo regulador de tensión como para el conjunto de relés, se emplearon circuitos preensamblados. En la figura 3.6 se muestran los componentes, su conexionado y la versión final de la unidad dentro de una caja protectora.

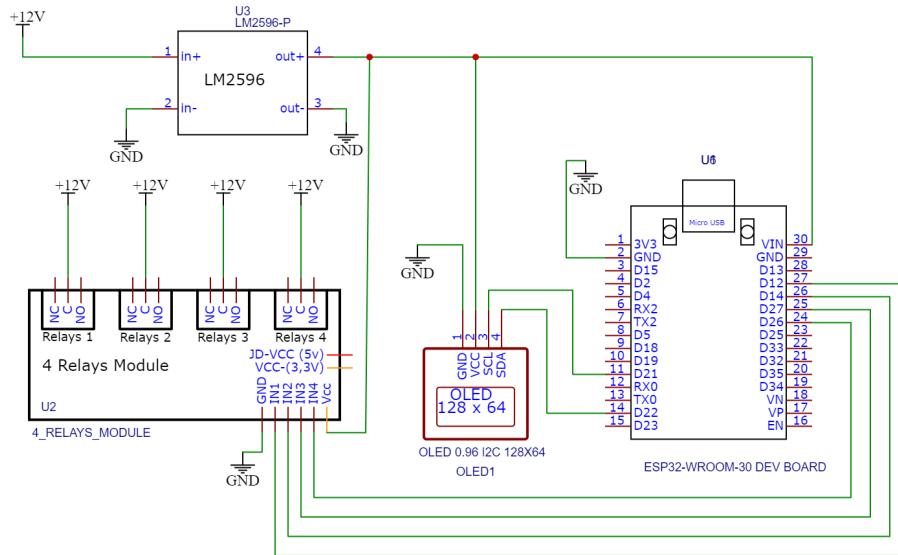
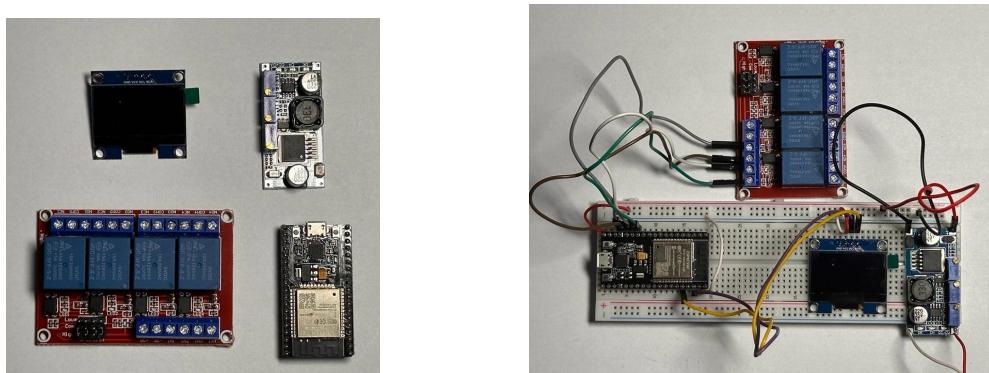
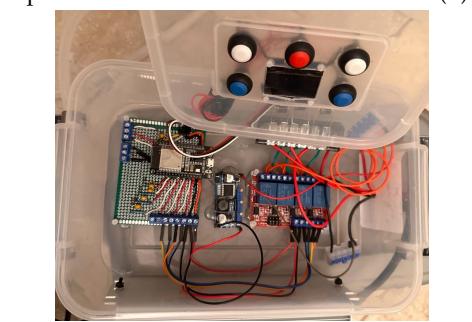


FIGURA 3.5. Conexión del módulo de control de riego.



(A) Detalle de los componentes.

(B) Conexionado.



(C) Módulo finalizado en su caja protectora.

FIGURA 3.6. Módulo de control de riego.

3.2.3. Módulo sensor de temperatura y humedad

Está compuesto por un microcontrolador ESP8266, un sensor DHT22 y una pantalla LCD/OLED SSH1106 para visualizar los valores de temperatura y humedad *in situ* en tiempo real. El esquema de conexión de los componentes se puede ver en la figura 3.7.

A diferencia de las sondas de humedad del suelo, el sensor de temperatura y humedad está pensado para instalarse en una ubicación fija con acceso a la red

eléctrica. Por este motivo no se consideró configurarlo para soportar *deep sleep*.

La construcción del módulo se realizó sobre placa PCB experimental en forma similar a los demás sistemas. Para proteger los componentes se utilizó una caja de polipropileno transparente. El sensor DHT22 quedó expuesto para medir las condiciones ambientales como muestra la figura 3.8.

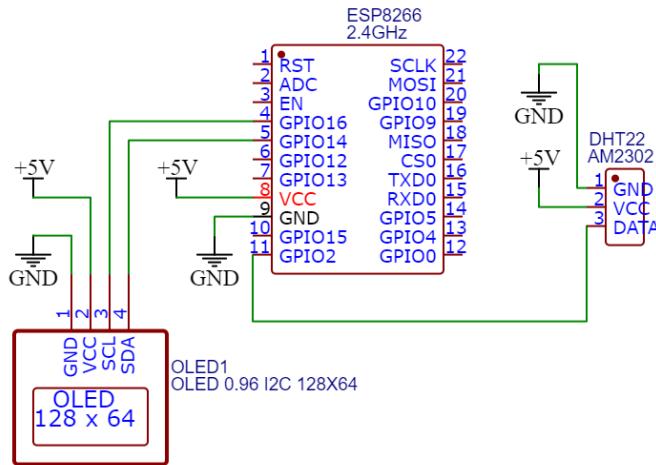


FIGURA 3.7. Conexión del sensor de temperatura y humedad.



FIGURA 3.8. Módulo completo en su caja protectora.

3.2.4. Módulo controlador de clima

Es el responsable de accionar los ventiladores en el invernadero. Para el diseño se utilizó un chip ESP8266 conectado a un relé de una vía como se muestra en la figura 3.9.

Dado que la salida del microcontrolador es de 3,3 V para el prototipo se seleccionó un relé que pueda ser accionado con ese valor de tensión, para evitar el uso de componentes adicionales tal como un convertidor de tensión DC-DC *step up*.

En la figura 3.10 se ilustra el proceso de construcción del módulo.

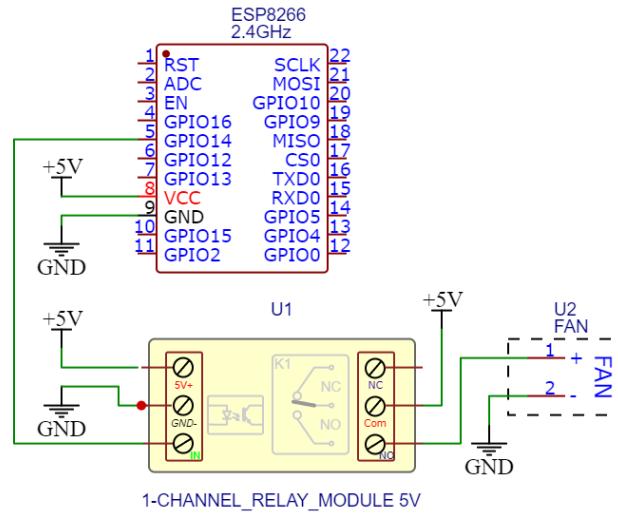


FIGURA 3.9. Conexión del módulo de control de clima.

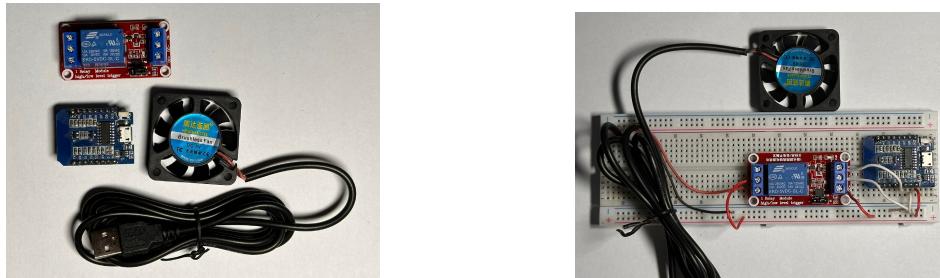


FIGURA 3.10. Módulo de control de clima.

3.3. Desarrollo del firmware

Como el soporte y la expansión del sistema quedará a cargo del cliente, se optó por desarrollar el firmware en C++ mediante la aplicación Arduino IDE. Esta elección se fundamenta en la baja curva de aprendizaje de la herramienta, la amplia disponibilidad de librerías y ejemplos para el uso de componentes, además de contar con una vasta comunidad de entusiastas de IoT en Internet.

En líneas generales, un programa escrito mediante esta herramienta respeta una estructura de dos bloques:

1. Setup: función que se ejecuta una única vez al comienzo del programa y se utiliza para inicializar las variables, configurar los pines de entrada y salida y establecer las comunicaciones necesarias, tales como la velocidad del puerto serial o la conexión a la red Wi-Fi.
2. Loop: función que se ejecuta continuamente en un bucle que, por lo general, solo se interrumpe al apagar el dispositivo. En esta sección se encuentra el código principal del firmware.

Adicionalmente, el código puede contener variables, funciones y bibliotecas.

Para facilitar la conexión a la aplicación ThingsBoard se utilizó el kit de desarrollo Arduino ThingsBoard SDK [44] que, entre otras funciones, provee mecanismos para el manejo de los protocolos MQTT y RPC, a la vez que permite la actualización OTA [45] del firmware.

Desde el mismo Arduino IDE se transfiere el código generado a la memoria no volátil (*flash*) del dispositivo. Al iniciar (o reiniciar) el microcontrolador, el programa se carga desde la memoria *flash* a la memoria RAM y comienza su ejecución.

3.3.1. Módulos sensores de humedad del suelo

En este caso, debido a que ThingsBoard no permite ajustar los períodos de retención de las colas de MQTT para soportar configuraciones prolongadas de *deep sleep*, se resolvió utilizar el protocolo HTTP para las comunicaciones entre el módulo y la aplicación central.

Luego del inicio, el programa realiza una llamada HTTP para obtener el valor del tiempo de hibernación y a continuación lleva a cabo la lectura de los sensores según el caso:

- Sensor simple: lee el valor del pin de conversión analógica a digital (ADC).
- Sensor doble: como el chip ESP8266 posee un único pin ADC compartido por ambas sondas, se procede al energizado secuencial de los sensores al activar la salida GPIO correspondiente, como se mostró en la figura 3.3b.

Para que el módulo pueda reportar un medición que represente la humedad del suelo, es necesario llevar a cabo una conversión del valor leído en el pin ADC. La relación entre cuentas valor obtenido y humedad se obtiene con una variante respecto del método descrito por Joshua Hrisko [46] que consiste en realizar mediciones en suelo seco para luego ir agregando cantidades precisas de agua sobre las que se evalúa la diferencia de potencial observada en cada caso. Del proceso resulta una expresión que permite estimar el contenido volumétrico de agua presente.

Una vez obtenido el o los valores, se los reporta a la aplicación por medio de una llamada HTTP POST y se da inicio al período de *deep sleep* durante el lapso de tiempo establecido.

El flujo de ejecución del código para un módulo doble se visualiza en la figura 3.11.

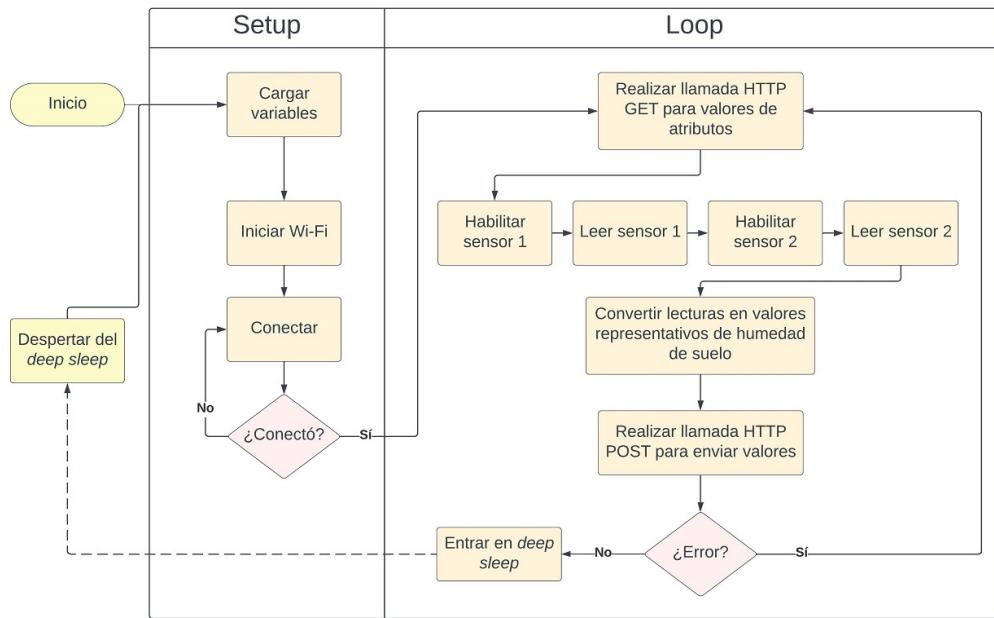


FIGURA 3.11. Diagrama de flujo del firmware de los módulos sensores de humedad del suelo.

3.3.2. Módulo controlador del riego

Es el módulo de mayor complejidad, tal como se refleja en las figuras 3.12, 3.13 y 3.14.

La ejecución inicia con la carga de variables, entre las que se destacan:

- Tiempo de riego: define la duración de encendido de la bomba en segundos.
- Bomba lista: variable lógica que indica el pedido de encendido de la bomba.
- Estado de válvulas: controlan las GPIOs de las válvulas.
- Estado de bomba: controla la GPIO de la bomba.

Luego de la conexión a la red Wi-Fi, el firmware se registra con la aplicación y se suscribe a los *getter* y *setter topics* para reportar los estados a la aplicación o recibir comandos.

En el caso de recibir un mensaje de pedido de información, el código reporta el o los valores de las variables solicitadas en el *topic* de respuesta. Por otro lado, si el mensaje recibido indica realizar un cambio de estado, se pueden presentar las siguientes situaciones:

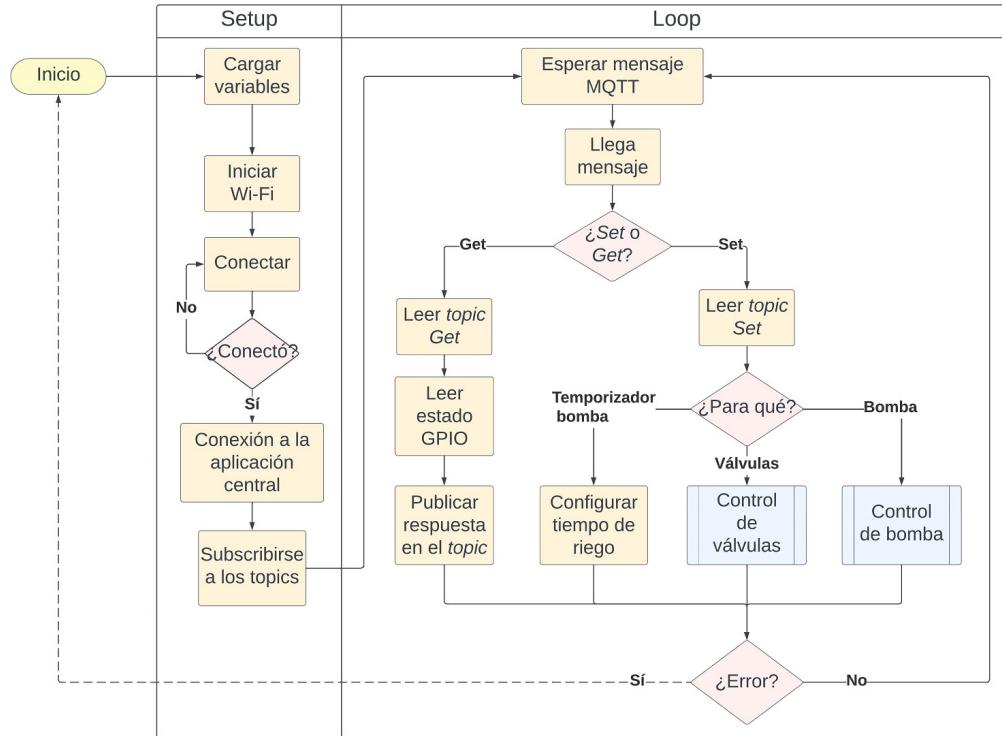


FIGURA 3.12. Diagrama de flujo del firmware del módulo de control de riego.

- Pedido de reconfigurar el tiempo de riego: el código actualiza la variable y reinicia el bucle.
- Pedido de apertura o cierre de válvula:
 - Apertura: procede a abrir la válvula seleccionada y comprueba si la bomba se encuentra en estado de pedido de encendido (bomba lista), en cuyo caso ordena el encendido.
 - Cierre: verifica si otras válvulas están abiertas y en caso afirmativo procede con el cierre. En caso contrario, si es la única, ordena el apagado de la bomba y luego el cierre de la válvula.
- Pedido de encendido o apagado de la bomba:
 - Encendido: en el caso de haber al menos una válvula abierta, procede a encender la bomba durante el tiempo que indique la variable de duración de riego. Una vez finalizado, procede ordenadamente al apagado de la bomba y al cierre de las válvulas. De no haber válvulas abiertas, se configura la variable de bomba lista en positivo, indicando que el riego fue pedido pero aún no están dadas las condiciones para habilitarlo.
 - Apagado: en el caso de requerir una interrupción en el riego, se ordena el apagado de la bomba y el cierre de las válvulas.

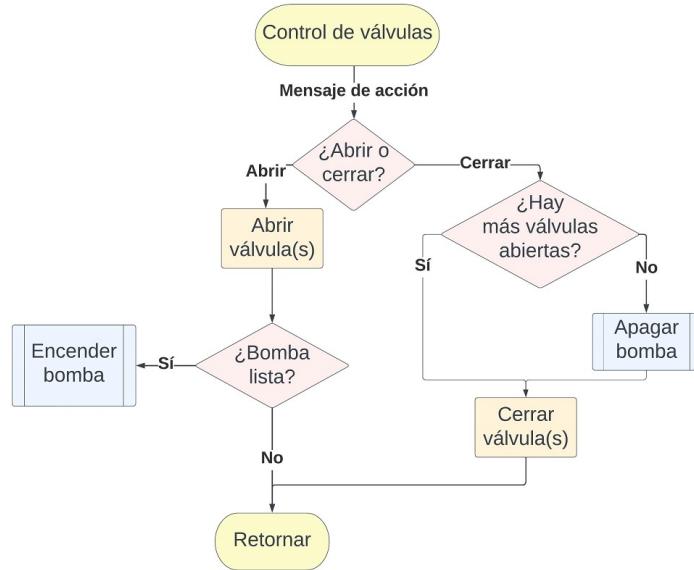


FIGURA 3.13. Diagrama de flujo del firmware del módulo de control de riego - control de válvulas.

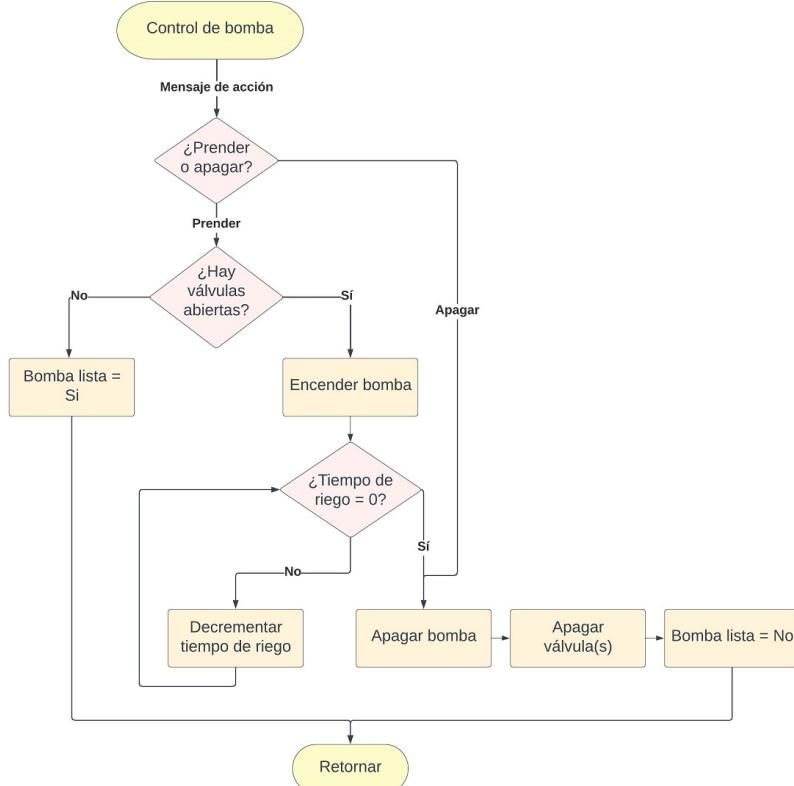


FIGURA 3.14. Diagrama de flujo del firmware del módulo de control de riego - control de bomba.

3.3.3. Módulo sensor de temperatura y humedad

La figura 3.15 describe el funcionamiento de este módulo. Luego del ciclo de inicio y del setup, el código se encarga de obtener las medidas de temperatura y

humedad, valores que despliega en la pantalla del módulo. Periódicamente estos valores se envían a la aplicación de acuerdo con un contador de ciclos.

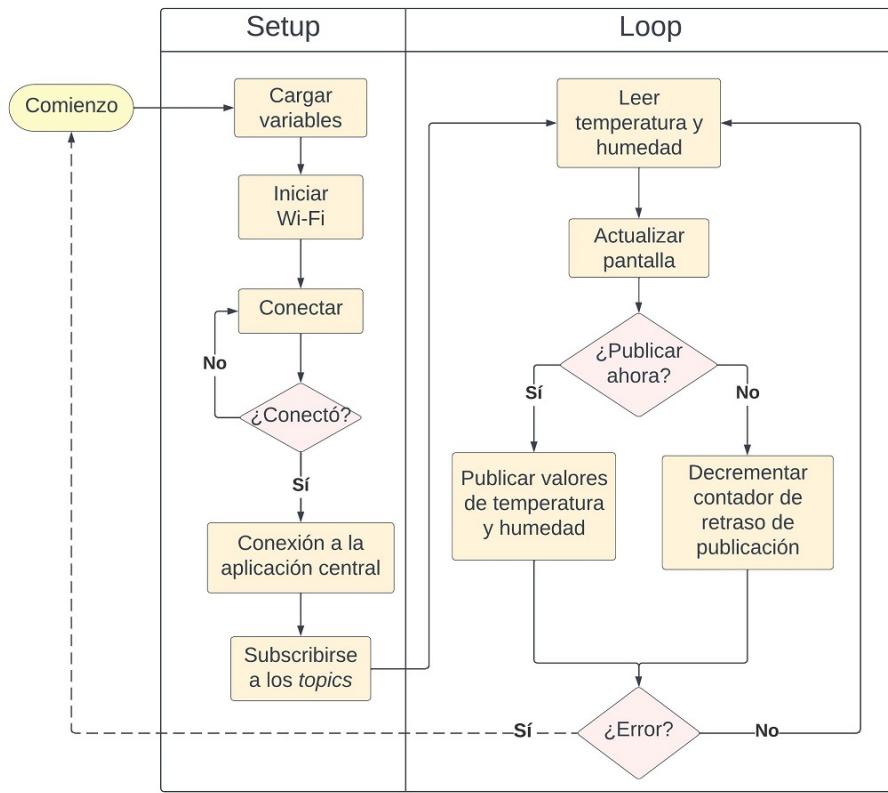


FIGURA 3.15. Diagrama de flujo del firmware del módulo sensor de temperatura y humedad.

3.3.4. Módulo de control de clima

Es el encargado del encendido y apagado de los ventiladores para controlar el clima. Luego del inicio, el programa se conecta a la red, se subscribe en los *topics* correspondientes e ingresa en la función de loop a la espera de mensajes. En caso de recibir un pedido de reporte, responde con el estado del pin GPIO asociado con el ventilador. Si recibe una orden de encendido o apagado, realiza la acción correspondiente y retorna al inicio del bucle.

En la figura 3.16 se describe el flujo del código para este módulo.

3.4. Selección y configuración del software

La propiedad del cliente se encuentra en una zona rural con acceso a internet limitado, por tal motivo se implementó una solución local basada en ThingsBoard Community Edition. Esta plataforma tiene bajos requerimientos de hardware, buen diseño de seguridad y ofrece amplias posibilidades de expansión.

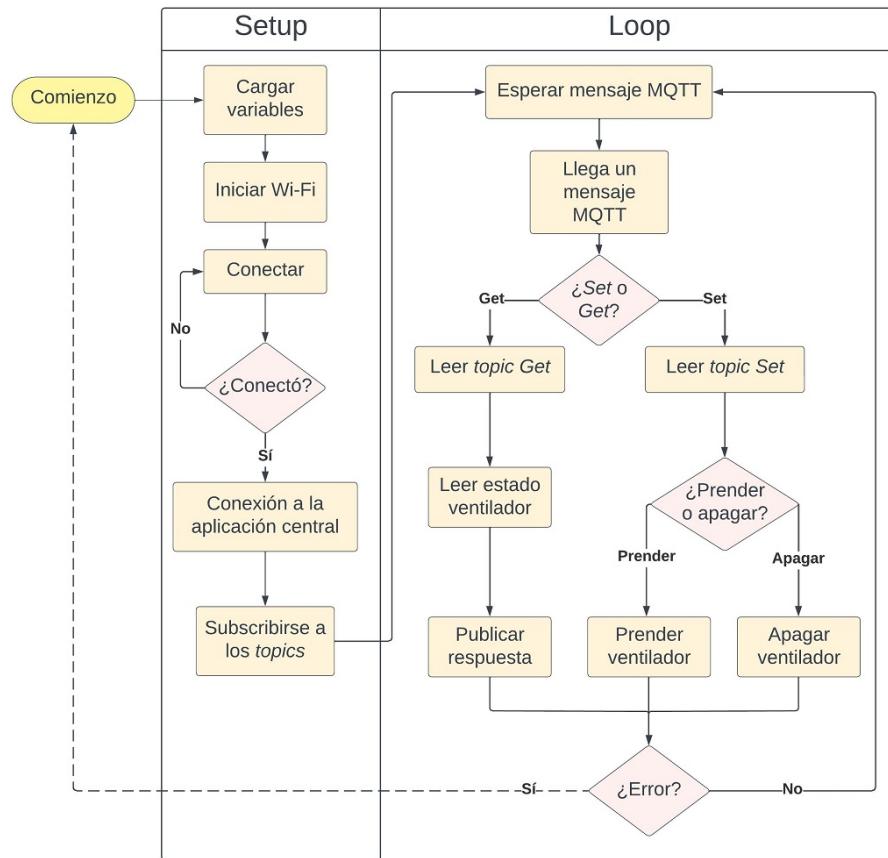


FIGURA 3.16. Diagrama de flujo del firmware del módulo de control de clima.

3.4.1. Servidor de la aplicación

La instalación se realizó sobre una Raspberry Pi 4B de 8 GB de memoria RAM, con sistema operativo Linux Raspbian versión 11, todo instalado sobre una tarjeta de memoria SD de 64 GB de capacidad.

3.4.2. Configuración de la aplicación

La instalación de la aplicación se realizó conforme a las instrucciones provistas por el desarrollador en su página oficial [47]. ThingsBoard requiere de una base de datos para almacenar los valores de telemetría y atributos. Para una carga de hasta 5000 mensajes por segundo se recomienda el uso de PostgreSQL [48]. Dado que la cantidad de mensajes enviados es de aproximadamente 4 por minuto en el prototipo y no se espera un crecimiento desmedido en el pasaje a producción, la capacidad de memoria y procesamiento empleados resulta suficiente para la instalación conjunta de la base de datos y la aplicación.

3.4.3. Manejo de dispositivos

La baja cardinalidad de módulos creados para el proyecto permitió emplear la interfaz web para gestionarlos en lugar de la opción programática mediante la API REST. Al no contar con una CA para la generación de certificados, la autenticación de los dispositivos se resolvió mediante el empleo de tokens.

La figura 3.17a muestra la pantalla con los dispositivos creados, mientras que en la figura 3.17b se observan las características de un sensor de humedad del suelo doble. Desde allí se puede navegar a los atributos del dispositivo, sus alarmas y eventos e incluso desvincularlo.

	Created time	Name	Device profile	Label
<input type="checkbox"/>	2022-11-01 20:23:16	VentControl	default	Ventilators1
<input type="checkbox"/>	2022-09-04 16:32:52	Pump Control	default	
<input type="checkbox"/>	2022-08-31 20:51:51	Thermometer & Humidity Sensor	Thermometer	
<input type="checkbox"/>	2022-08-30 18:15:53	Soil Moisture Dual Sensor	SoilMoistureSensor	Zone2
<input type="checkbox"/>	2022-05-28 13:47:52	Single Sensor	SoilMoistureSensor	Zone1

(A) Listado de dispositivos del proyecto.

(B) Configuración de las características de un sensor.

FIGURA 3.17. Pantallas de manejo de dispositivos creados en el proyecto.

3.4.4. Reglas de automatización

ThingsBoard posee un motor de reglas [49] para la creación de flujos de trabajo basados en eventos. Una de sus principales ventajas es la de permitir el desarrollo de automatizaciones bajo la metodología *low-code/no-code* [50].

Las reglas se componen principalmente de:

- Mensaje: cualquier evento que ingrese, por ejemplo datos provenientes de un dispositivo, un pedido a través de la API REST (HTTP) o un requerimiento RPC.
- Nodo de regla: función para el filtrado, transformación o acción que se ejecuta sobre un mensaje entrante y que puede producir mensajes salientes.
- Conexión de nodo de regla: los nodos pueden vincularse entre sí mediante relaciones, de manera que la salida de uno es enviada a los próximos nodos conectados. Cada relación tiene una etiqueta que identifica su significado lógico, generalmente del tipo “éxito” o “fracaso”.
- Cadena de reglas: grupo lógico de nodos de reglas y sus conexiones.

El procesamiento de mensajes ofrece tres resultados posibles: éxito, error y *timeout* (tiempo excedido). Se obtiene un resultado exitoso cuando el último nodo de regla en la cadena completa correctamente el procesamiento. Se obtiene error cuando en el nodo se produce un fallo al operar sobre un mensaje y no existe un mecanismo para manejarlo. Por último, se establece un *timeout* cuando el tiempo total de procesamiento supera el umbral configurado.

Para el proyecto se desarrollaron una o más reglas por subsistema, de manera de conectar a las unidades de control con los respectivos sensores que informan sobre el estado del invernadero y para el manejo de alertas a los usuarios. La figura 3.18 muestra, a manera de ejemplo, un fragmento de la regla creada para el control de riego a partir de los mensajes recibidos desde los sensores de humedad del suelo. Allí se visualizan los nodos en diferentes colores de acuerdo a su función y las cadenas como las flechas que los unen.

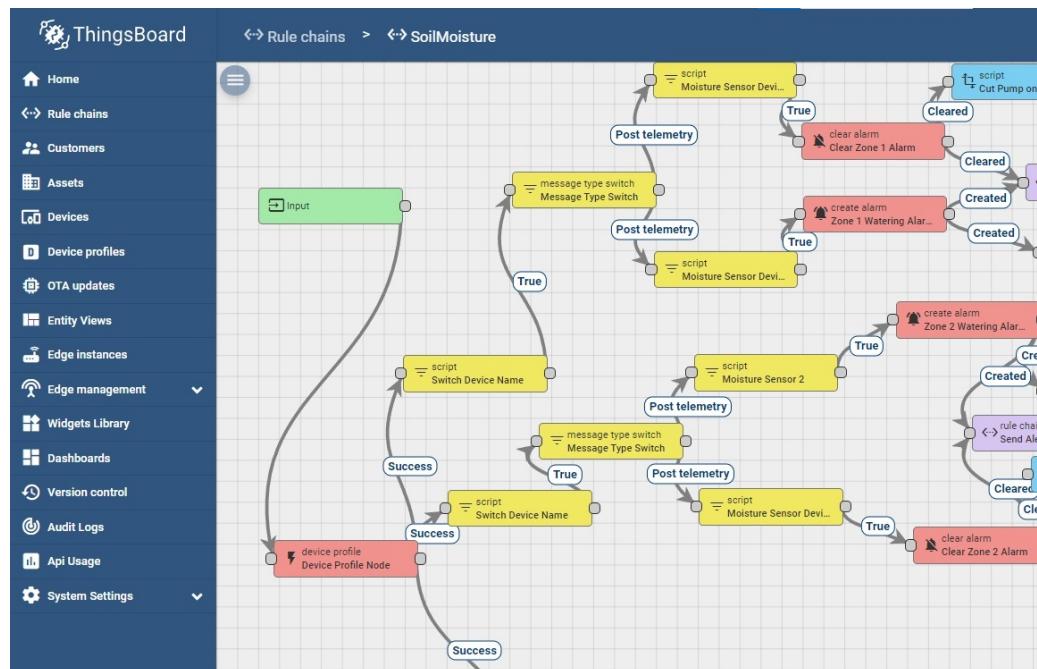


FIGURA 3.18. Fragmento de regla de automatización para el control de riego.

3.4.5. Interfaz de manejo remoto

Para permitir que los usuarios conozcan el estado del invernadero vía Internet, se desarrolló una aplicación en Python que se integra al *bot* de Telegram mediante la librería SDK python-telegram-bot. Este programa fue alojado en la misma Raspberry Pi donde corre ThingsBoard.

Se trata de un bucle que contacta al *bot* de Telegram periódicamente en busca de mensajes nuevos. Al detectar la llegada de alguno, el programa identifica si se corresponde con alguno de los comandos definidos y de ser así, envía una solicitud HTTP a la aplicación central. En base a la respuesta recibida, la interfaz confecciona un mensaje que es enviado al usuario por medio del *bot*. La figura 3.19a muestra un ejemplo de mensaje de estado del invernadero.

Además, el servicio de *bot* se utilizó para el envío de alertas a usuarios. Para ello se crearon nodos en las reglas de automatización de la aplicación central que, ante ciertos eventos, disparan una solicitud HTTP POST al *bot* con el mensaje de alerta como se ilustra en la figura 3.19b.

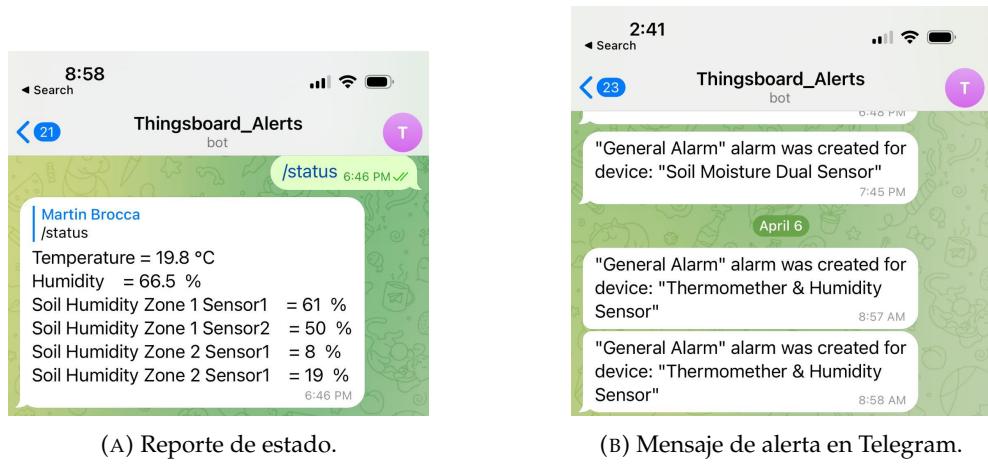


FIGURA 3.19. Uso del *bot* de Telegram .

3.5. Ciberseguridad del sistema

Por tratarse de un proyecto hogareño, el cliente no cuenta con la infraestructura de autenticación ni con una CA para proveer certificados a los dispositivos o a la aplicación central. Por este motivo se decidió utilizar un certificado autofirmado para encriptar las comunicaciones entre los módulos y ThingsBoard. Sin embargo, el uso de este tipo de certificados tiene la desventaja de carecer del respaldo de una entidad que garantice su autenticidad, por lo que es necesario realizar configuraciones explícitas para forzar la confianza.

Durante el proceso de desarrollo se siguieron las buenas prácticas para el manejo de contraseñas y tokens de autenticación, evitando divulgar estos valores en los repositorios públicos.

Capítulo 4

Ensayos y resultados

En este capítulo se explica la metodología de pruebas aplicada tanto a los componentes individuales como al sistema implementado para finalizar con una comparación con el estado del arte.

4.1. Banco de pruebas

La verificación del correcto funcionamiento de los módulos que componen el sistema se realizó mediante una maqueta que se muestra en la figura 4.1 y representa en escala reducida al invernadero del cliente.

El modelo cuenta con una bomba de agua conectada a tres válvulas para alimentar a los circuitos de riego. Se utilizaron mangueras y conexiones neumáticas de aluminio de acople rápido y el conjunto armado puede verse en la figura 4.2a.

El ensamblado se muestra en las figuras 4.2b, 4.2c y 4.2d donde se observa la implementación de dos circuitos de riego independientes. También se configuró un tercer circuito cerrado para pruebas de accionamiento de bomba y válvula a fin de evitar desperdicios durante las fases de calibración.

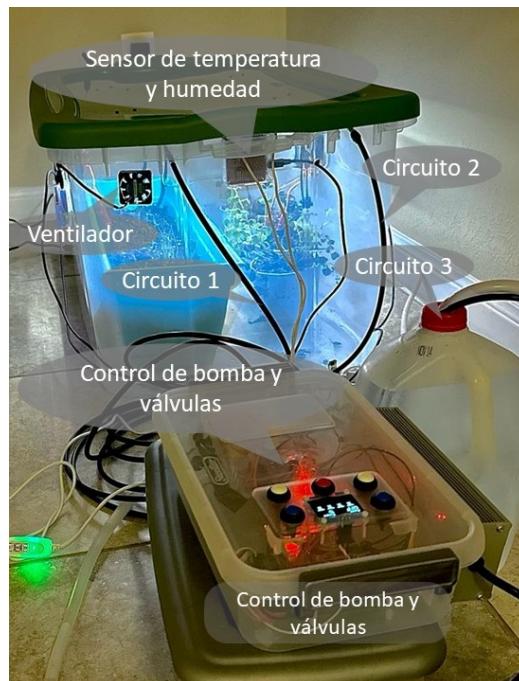


FIGURA 4.1. Modelo completo del invernadero.

Además se incorporaron a la maqueta dos módulos sensores de humedad del suelo, uno de temperatura y humedad y un actuador para el encendido de los ventiladores.

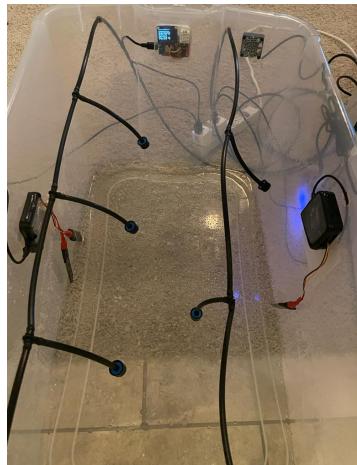
Para las pruebas manuales de accionamiento de los diferentes sistemas se empleó una computadora portátil y un celular para las pruebas de acceso concurrente a la aplicación central.



(A) Conjunto bomba y válvulas.



(B) Armado de mangueras de riego.



(C) Pruebas de riego.



(D) Ensamble general.

FIGURA 4.2. Modelo de pruebas del invernadero.

4.2. Pruebas individuales

Se detallan las principales pruebas realizadas a los módulos a fin de garantizar su correcto funcionamiento y/o el cumplimiento de requerimientos.

4.2.1. Pruebas a módulos sensores

- Conexión a los sensores y valores obtenidos: se conectaron individualmente los módulos a la computadora mediante la consola serial de Arduino IDE y se constataron los valores obtenidos frente a cambios inducidos sobre la magnitud a medir.
- Conexión a la aplicación y envío de telemetría: se cargó el firmware en los módulos sensores conforme a lo descrito en las secciones 3.3.1 y 3.3.3, mientras que se configuró la instancia instalada de ThingsBoard de acuerdo a lo explicado en 3.4.2. Se construyó un *dashboard* (tablero de visualización) en donde se incluyeron los *widgets* (componentes visuales) necesarios para visualizar la telemetría en tiempo real. Desde la consola serial del módulo, se comprobó la conexión del microcontrolador a la red Wi-Fi. Posteriormente se verificó la vinculación del dispositivo con la aplicación y se constató la correcta recepción de los valores en el *dashboard*. La figura 4.3 muestra una captura de pantalla de la visualización de las series de tiempo correspondientes



FIGURA 4.3. Dashboard de visualización de telemetría.

- Sincronización de los sensores de humedad: como se indicó en la sección 3.1.2 la comunicación con la aplicación central es bidireccional. Para constatar la correcta recepción de mensajes por parte del módulo, se establecieron atributos compartidos capaces de ser modificados desde ThingsBoard y por medio de un *dashboard* se realizaron variaciones de los valores de calibración de los sensores y diferentes tiempos para el *deep sleep*. Se comprobó la recepción de los parámetros por consola serial y la acción en consecuencia por parte del módulo. En la figura 4.4 se observa el estado de humedad del suelo en dos sensores, los valores de ajuste y la configuración del tiempo de hibernación.

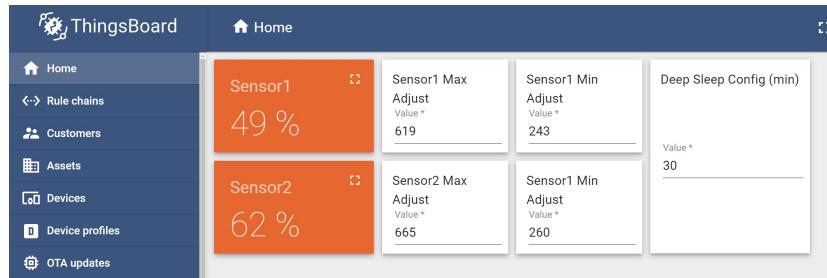


FIGURA 4.4. Atributos de sensores de humedad del suelo.

- Resistencia al agua y a salpicaduras: requerimiento no funcional explícito para los sensores de humedad del suelo. Consistió en comprobar la integridad y el funcionamiento del módulo al tiempo que se sumergía el sensor en un recipiente con agua o se lo sometía a aspersiones continuas por alrededor de 30 segundos con atomizadores desde diferentes ángulos. La figura 4.5 esquematiza parcialmente el proceso de prueba mencionado.



FIGURA 4.5. Prueba de resistencia al agua.

4.2.2. Pruebas a módulos actuadores

Se conectaron las salidas de los pines de GPIO a leds para visualizar el encendido y apagado del actuador.

La figura 4.6a muestra la interfaz desarrollada en ThingsBoard a fin de enviar las señales de comando al módulo por medio de mensajes RPC sobre MQTT. El microcontrolador recibe la orden y ejecuta la acción, reflejada en el estado de los leds correspondientes como se observa en la figura 4.6b.

4.2.3. Pruebas de acceso concurrente a la aplicación

Las pruebas de concurrencia se efectuaron mediante el acceso simultáneo de diferentes usuarios mediante una computadora portátil y un teléfono celular. Los ensayos consistieron en realizar distintas acciones desde un dispositivo como, por ejemplo, el accionamiento de un circuito de riego por parte de un usuario, y la confirmación del cambio de estado recibida por otro.

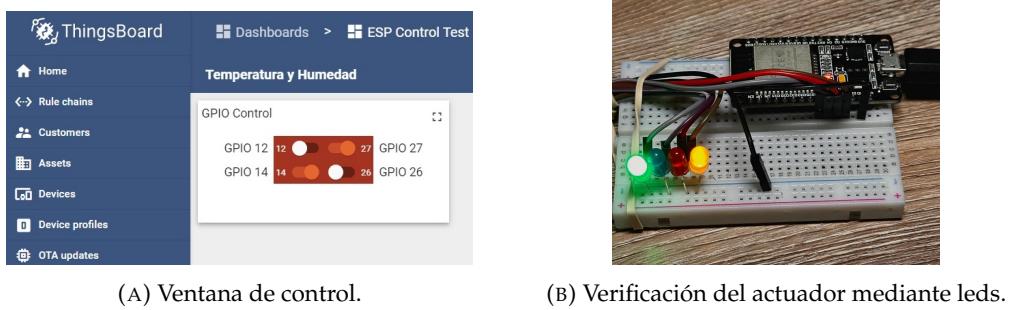


FIGURA 4.6. Pruebas unitarias sobre controladores.

La figura 4.7a muestra la vista obtenida mediante el uso del ordenador mientras que la figura 4.7b la del celular.

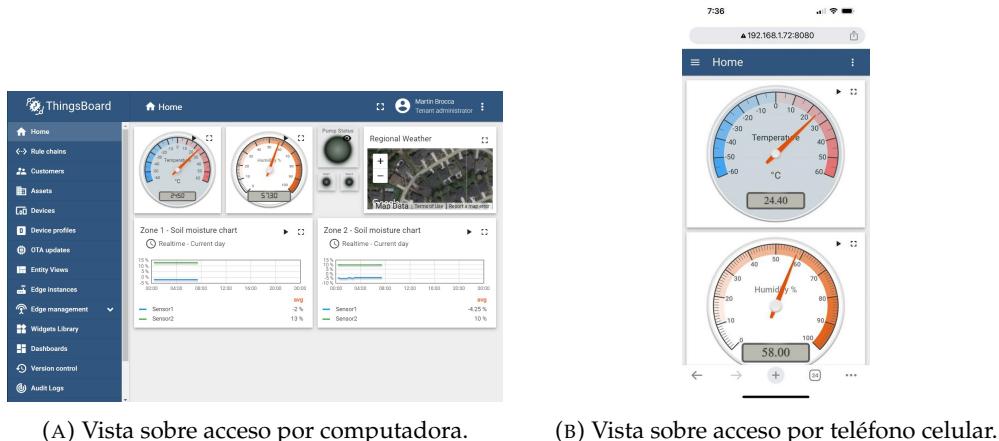


FIGURA 4.7. Pruebas unitarias sobre acceso concurrente.

4.3. Pruebas de sistema

El objetivo de las pruebas de integración es verificar el correcto funcionamiento e interacción de los sistemas que conforman el modelo. Con este propósito se desarrollaron reglas de automatización para el control de riego y de temperatura, además de las necesarias para el manejo de alarmas del sistema.

Estas reglas, cuyo esquema se muestra en la figura 4.8, son similares en construcción y constan de los siguientes nodos y relaciones:

- Recepción de mensajes desde los módulos sensores.
- Determinación del emisor en base a la metadata del mensaje.
- Lectura de la telemetría.
- Comparación con valor de referencia.

Si la comparación resulta exitosa, dispara una alerta de notificación al cliente e informa al microcontrolador que comience una acción. Caso contrario, limpia cualquier alerta presente y de ser necesario, detiene la acción en el actuador.

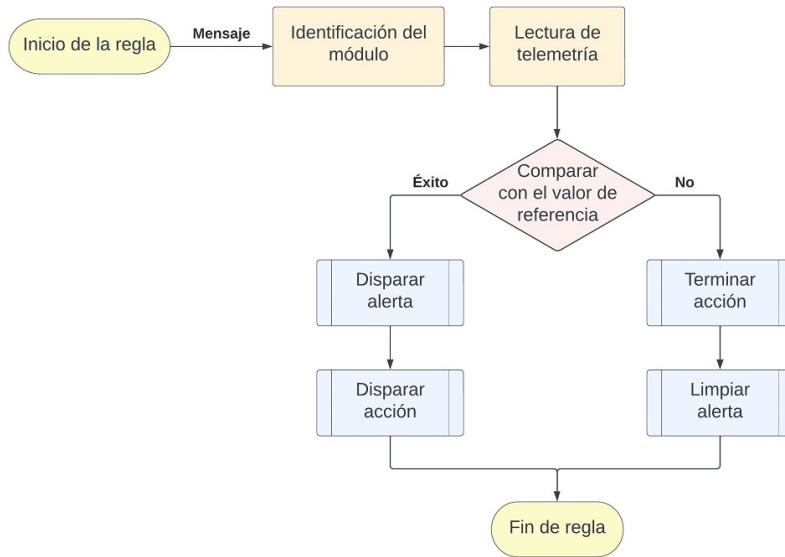


FIGURA 4.8. Regla de automatización genérica.

4.3.1. Control de clima

Se definió una regla de control con un límite máximo de 28 °C de temperatura que, cuando se supera, envía un mensaje de encendido de ventiladores e inicia el proceso de notificación al usuario. Por el contrario, si la temperatura se mantiene por debajo del umbral, se cancelan las alarmas y retorna al estado de espera de mensajes. La regla implementada se muestra en la figura 4.9.

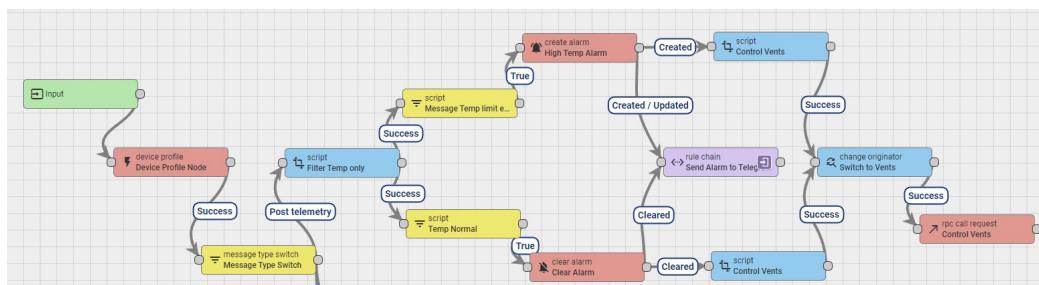
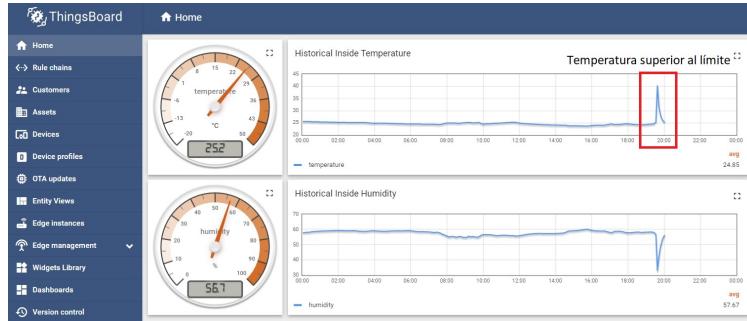
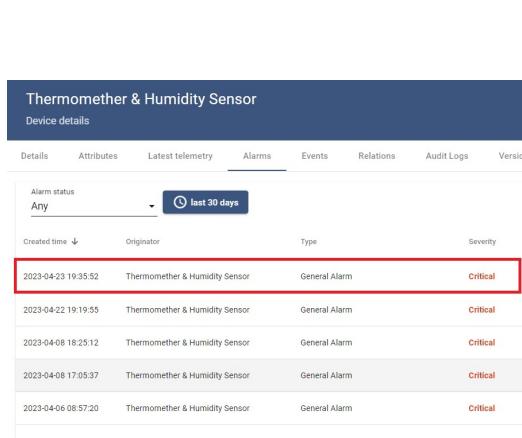


FIGURA 4.9. Regla de control de clima.

Para la prueba se instaló la maqueta en un entorno de ambiente controlado a 25 °C para luego incrementar la temperatura artificialmente por medio de una pistola de calor y observar la reacción del sistema. La figura 4.10a corresponde al diagrama histórico de temperatura y humedad del invernadero, en el recuadro rojo se muestra el instante en donde el cambio de temperatura dispara la acción del sistema y se encendieron los ventiladores, mientras que en las figuras 4.10b y 4.10c se observan las alarmas en el dispositivo y de usuario creadas por este evento.



(A) Vista del historial de temperatura.



(B) Alarmas en el dispositivo.



(C) Alarmas recibida por el usuario.

FIGURA 4.10. Generación de alarmas por exceso de temperatura.

4.3.2. Control de riego

La prueba de integración del sistema de riego se realizó sobre la maqueta descrita en la sección 4.1 empleando un módulo sensor doble por circuito de riego. La utilización de dos sondas por módulo pretende que el riego sea uniforme sobre diferentes disposiciones de macetas.

Se construyó una regla de control según lo descrito en la sección 4.3 donde se reciben las lecturas provenientes desde los dos módulos sensores. Como primer paso, se identifica la zona a la que pertenecen las sondas. Si el valor recibido es inferior a la referencia corresponde a un suelo seco, por lo tanto se debe iniciar el sistema de bomba y válvula correspondiente al circuito por el tiempo de riego predefinido.

Para evaluar la regla se plantearon los siguientes parámetros de configuración de sensor:

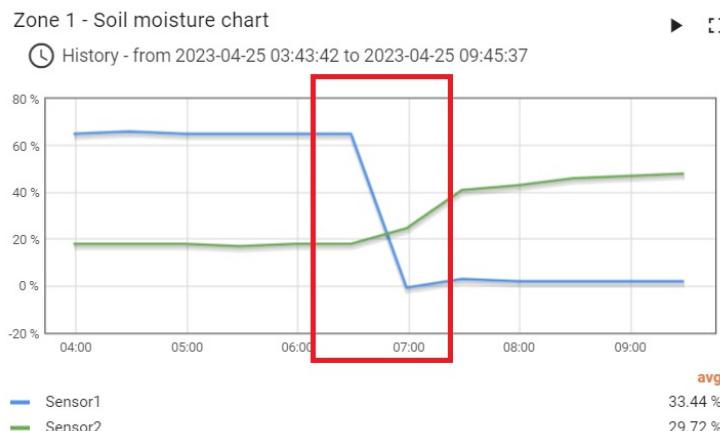
- Seco: Sonda expuesta al aire, limpia y sin rastros de humedad.
- Húmedo: Sensor introducido en tierra con una humedad aproximada del 35 %

En la tabla 4.1 se registró el resultado de las diferentes pruebas realizadas, considerando estas exitosas solamente cuando en ambos sensores del módulo se indique un suelo seco y se dispare el sistema de riego para el sector correspondiente. La figura 4.11a muestra el gráfico histórico de humedad de suelo en donde se

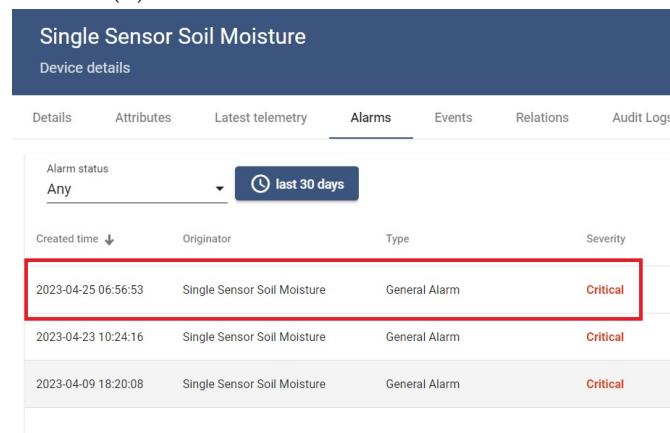
indica el instante en que ambas sondas se encuentran por debajo del umbral configurado y la correspondiente alarma generada se indica en la figura 4.11b.

TABLA 4.1. Pruebas de sistema de riego.

Zona 1		Zona 2		Riego zona 1	Riego zona 2	Resultado
Sensor 1	Sensor 2	Sensor 1	Sensor 2			
Seco	Seco	Seco	Seco	Activa	Activa	Ok
Seco	Seco	Seco	Húmedo	Activa	No	Ok
Seco	Seco	Húmedo	Húmedo	Activa	No	Ok
Seco	Húmedo	Seco	Seco	No	Activa	Ok
Seco	Húmedo	Seco	Húmedo	No	No	Ok
Seco	Húmedo	Húmedo	Húmedo	No	No	Ok
Húmedo	Húmedo	Seco	Seco	No	Activa	Ok
Húmedo	Húmedo	Seco	Húmedo	No	No	Ok
Húmedo	Húmedo	Húmedo	Húmedo	No	No	Ok



(A) Vista del historial de humedad del suelo.



(B) Alarmas generadas al superar los límites.

FIGURA 4.11. Prueba del sistema de riego.

4.4. Comparativa con el estado de arte

Esta tabla no me gusta para nada, la voy a pasar a bullets o algun otro formato, pero estaba viendo como quedaba. basicamente aca voy a poner algo como (copio porquerias de ChatGPT :

El invernadero inteligente tipo DIY es asequible, personalizable y ofrece una mayor flexibilidad de expansión, pero puede requerir un mayor conocimiento técnico. Las soluciones comerciales ofrecen una amplia gama de características integradas, son más fáciles de instalar y configurar, pero son más costosas y menos personalizables. En última instancia, la elección entre ambas opciones dependerá de las necesidades y objetivos específicos del usuario.

En el caso de las soluciones comerciales, estas suelen contar con una amplia gama de características integradas en el sistema, lo que puede hacerlas más atractivas para usuarios que buscan una solución completa y lista para usar. En cambio, en el caso de un invernadero inteligente tipo DIY, el usuario debe elegir y configurar los módulos de sensores y dispositivos a utilizar en el sistema, lo que puede requerir un mayor esfuerzo y conocimientos técnicos.

Característica	Trabajo realizado	Soluciones comerciales
Costo	Accesible, con gran variedad de productos disponibles en el mercado.	Costos medios a altos, con soluciones que dependen de la marca seleccionada.
Personalización	Altamente personalizable y ajustable a las necesidades del usuario.	Limitaciones en cuanto a la personalización.
Capacidad de expansión	Mayor flexibilidad para ser ampliado y personalizado con la incorporación de nuevos módulos de sensores y dispositivos.	Limitaciones en cuanto a la expansión y personalización, aunque puede ofrecer más opciones de integración con otros sistemas.
Dificultad de instalación y ajustes	Requiere conocimientos técnicos para su instalación y configuración, pero puede ser más fácil de ajustar y personalizar.	Más fáciles de instalar y configurar, aunque puede requerir una curva de aprendizaje para personalizar y ajustar a las necesidades específicas del usuario.
Funcionalidades e integración	Posee una limitada cantidad de funcionalidades integradas, aunque permite el desarrollo de las mismas.	Poseen una amplia gama de funcionalidades e integración con productos de la propia marca.

Capítulo 5

Conclusiones

En este capítulo se muestran las conclusiones sobre el trabajo realizado. A su vez se presentan algunas modificaciones o mejoras como posible trabajo futuro

5.1. Resultados obtenidos

5.2. Trabajo futuro

Acá se indica cómo se podría continuar el trabajo más adelante.

Bibliografía

- [1] Scott Eldridge y Lyman Chapin Karen Rose. «The Internet of Things: An Overview». En: (2015).
- [2] Krishna Nemali. «History of Controlled Environment Horticulture: Greenhouses». En: *HortScience* 57.2 (2022), págs. 239 -246. DOI: [10.21273/HORTSCI16160-21](https://doi.org/10.21273/HORTSCI16160-21). URL: <https://journals.ashs.org/hortsci/view/journals/hortsci/57/2/article-p239.xml>.
- [3] Chrysanthos Maraveas y Thomas Bartzanas. «Aplicación de internet de las cosas (IoT) para entornos de invernadero optimizados». En: *Magna Scientia UCEVA* 2 (dic. de 2022), págs. 253-268. DOI: [10.54502/msuceva.v2n2a11](https://doi.org/10.54502/msuceva.v2n2a11).
- [4] John W. Bartok. *Greenhouses for Homeowners and Gardeners*. Natural Resource, Agriculture, y Engineering Service (NRAES), 2000-06.
- [5] Intel. *¿Qué son los invernaderos inteligentes?* <https://agrofacto.com/invernaderos-inteligentes/m>. 2022. (Visitado 20-03-2023).
- [6] Rakiba Rayhana, Gaozhi Xiao y Zheng Liu. «Internet of Things Empowered Smart Greenhouse Farming». En: *IEEE Journal of Radio Frequency Identification* 4.3 (2020), págs. 195-211. DOI: [10.1109/JRFID.2020.2984391](https://doi.org/10.1109/JRFID.2020.2984391).
- [7] Argus Controls. <https://arguscontrols.com/>. (Visitado 20-03-2023).
- [8] Grodan. <https://www.grodan.com/>. (Visitado 20-03-2023).
- [9] Growlink. <https://www.growlink.com/>. (Visitado 20-03-2023).
- [10] Chet Udell y Alan Dennis Lloyd Nackley. <https://diggermagazine.com/the-smart-greenhouse/>. 2020. (Visitado 20-03-2023).
- [11] Arduino. <https://www.arduino.cc/>. (Visitado 20-03-2023).
- [12] OASIS. *MQTT Version 5.0*. Oasis Standard. URL: [\\url{https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html}](https://url{https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html}) (visitado 20-03-2023).
- [13] Inc. Sun Microsystems. *RPC: Remote Procedure Call Protocol Specification Version 2*. RFC 1057. RFC Editor, 1988. URL: <https://www.rfc-editor.org/rfc/rfc1057>.
- [14] E. Rescorla T. Dierks. *The Transport Layer Security (TLS) Protocol Version 1.2*. RFC. 2008. URL: <https://www.rfc-editor.org/rfc/rfc5280>.
- [15] C. Shao et al. *IEEE 802.11 Medium Access Control (MAC) Profile for Control and Provisioning of Wireless Access Points (CAPWAP)*. RFC. URL: [\\url{https://www.rfc-editor.org/rfc/rfc7494.html}](https://url{https://www.rfc-editor.org/rfc/rfc7494.html}) (visitado 20-03-2023).
- [16] T. Socolofsky et al. *A TCP/IP Tutorial*. RFC 1180. RFC Editor, 1991. URL: <https://www.rfc-editor.org/rfc/rfc1180>.
- [17] www.survivingwithandroid.com. <https://www.survivingwithandroid.com/mqtt-protocol-tutorial/>. (Visitado 20-03-2023).

- [18] Patrick Th. Eugster et al. «The Many Faces of Publish/Subscribe». En: *ACM Comput. Surv.* 35.2 (2003), 114–131. ISSN: 0360-0300. DOI: [10.1145/857076.857078](https://doi.org/10.1145/857076.857078). URL: <https://doi.org/10.1145/857076.857078>.
- [19] The Internet Society. *Hypertext Transfer Protocol – HTTP/1.1*. RFC. URL: \url{https://www.ietf.org/rfc/rfc2616.txt} (visitado 20-03-2023).
- [20] <https://www.oreilly.com/library/view/javaserver-pages-3rd/0596005636/ch02s01.html>. (Visitado 20-03-2023).
- [21] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. Inf. téc. 2000. Cap. Chapter 5: Representational State Transfer (REST).
- [22] Jasenka Dizdarević et al. «A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration». En: *ACM Comput. Surv.* 51.6 (ene. de 2019). ISSN: 0360-0300. DOI: [10.1145/3292674](https://doi.org/10.1145/3292674). URL: <https://doi.org/10.1145/3292674>.
- [23] Dimitrios Glaroudis, Athanasios Iossifides y Periklis Chatzimisios. «Survey, comparison and research challenges of IoT application protocols for smart farming». En: *Computer Networks* 168 (2020), pág. 107037. ISSN: 1389-1286. DOI: <https://doi.org/10.1016/j.comnet.2019.107037>. URL: <https://www.sciencedirect.com/science/article/pii/S1389128619306942>.
- [24] J. Postel. *Transmission Control Protocol*. RFC 793. Internet Engineering Task Force, 1981, pág. 85. URL: <http://www.rfc-editor.org/rfc/rfc793.txt>.
- [25] Internet Society. «TLS Basics». En: (). URL: <https://www.internetsociety.org/deploy360/tls/basics/>.
- [26] D. Cooper et. al. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC. 2008. URL: <https://www.rfc-editor.org/rfc/rfc5280>.
- [27] Shanna L. «Comparative Analysis of Infrastructure and Ad-Hoc Wireless Networks». En: *ITM Web of Conferences* 25, 01009 (2019).
- [28] https://es.wikipedia.org/wiki/Raspberry_Pi. (Visitado 20-03-2023).
- [29] <https://opensource.com/resources/raspberry-pi>. (Visitado 20-03-2023).
- [30] *Raspberry Pi Computer Model B*. Datasheet. Raspberry Pi. Ene. de 2021. URL: \url{https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf}.
- [31] *ESP32 Series*. Datasheet. V4.2. Espressif. Ene. de 2023.
- [32] *ESP8266EX*. Datasheet. V6.9. Espressif. Feb. de 2023.
- [33] *Digital output relative humidity and temperature sensor/module*. DHT22. Aosong Electronics Co.,Ltd. URL: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>.
- [34] <https://smartbitbn.com/product/soil-humidity-sensor/>. (Visitado 20-03-2023).
- [35] <https://www.amazon.com/dp/B07BW21Z5M/>. (Visitado 20-03-2023).
- [36] <https://es.wikipedia.org/wiki/Rel%C3%A1y>. (Visitado 20-03-2023).
- [37] <https://thingsboard.io/docs/getting-started-guides/what-is-thingsboard/>. (Visitado 20-03-2023).
- [38] Apache Software Foundation. *Apache License 2.0*. Contract. Apache Software Foundation, 2004. URL: <https://www.apache.org/licenses/LICENSE-2.0>.
- [39] *Soporte de Arduino IDE para hardware de terceros*. URL: <https://github.com/per1234/ino-hardware-package-list/blob/master/ino-hardware-package-list.tsv>.

- [40] *Arduino Integrated Development Environment (IDE) v1.* URL: <https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics>.
- [41] *What is arduino software (IDE), and how use it ?* URL: <https://andprof.com/tools/what-is-arduino-software-ide-and-how-use-it/>.
- [42] *Telegram - Preguntas Frecuentes.* URL: <https://telegram.org/faq?setln=es#p-que-es-telegram-que-puedo-hacer-aqui>.
- [43] <https://www.iec.ch/ip-ratings>. (Visitado 20-03-2023).
- [44] Thingsboard. *Thingsboard Arduino SDK.* Visitado el 2023-03-20. 2023. URL: <https://github.com/thingsboard/thingsboard-arduino-sdk>.
- [45] Jan Bauwens et al. «Over-the-Air Software Updates in the Internet of Things: An Overview of Key Principles». En: *IEEE Communications Magazine* 58.2 (2020), págs. 35-41. DOI: [10.1109/MCOM.001.1900125](https://doi.org/10.1109/MCOM.001.1900125).
- [46] Joshua Hrisko. <https://makersportal.com/blog/2020/5/26/capacitive-soil-moisture-calibration-with-arduino>. 2020. (Visitado 20-03-2023).
- [47] Thingsboard. *Thingsboard Arduino SDK.* Visitado el 2023-04-09. URL: <https://thingsboard.io/docs/user-guide/install/rpi/>.
- [48] PostgreSQL. *PostgreSQL - About.* Visitado el 2023-03-20. 2023. URL: <https://www.postgresql.org/about/>.
- [49] ThingsBoard. *Rule Engine Overview.* Visitado el 2023-03-21. 2023. URL: <https://thingsboard.io/docs/user-guide/rule-engine-2-0/overview/>.
- [50] Chris Johannessen y Tom Davenport. *When Low-Code/No-Code Development Works — and When It Doesn't.* Visitado el 2023-04-09. URL: <https://hbr.org/2021/06/when-low-code-no-code-development-works-and-when-it-doesnt>.