



Sistema de monitoreo y gestión remota del clima en invernaderos

Martín Anibal Lacheski

Carrera de Especialización en Internet de las Cosas

Director: Leopoldo Alfredo Zimperz (FIUBA)

Jurados:

- Jurado 1 (pertenencia)
- Jurado 2 (pertenencia)
- Jurado 3 (pertenencia)

Ciudad de Montecarlo, Misiones, Junio de 2025



Sistema de monitoreo y gestión remota del clima en invernaderos

Lic. Martín Anibal Lacheski

Carrera de Especialización en Internet de las Cosas

Director: Mg. Lic. Leopoldo Alfredo Zimperz (FIUBA)

Jurados:

- Jurado 1 (pertenencia)
- Jurado 2 (pertenencia)
- Jurado 3 (pertenencia)

Ciudad de Montecarlo, Misiones, Junio de 2025

Resumen

La presente memoria describe el desarrollo de un prototipo para monitorear y controlar de manera remota las condiciones climáticas en los invernaderos de la Facultad de Ciencias Forestales de la Universidad Nacional de Misiones. La solución propuesta integra sensores y actuadores basados en el microcontrolador ESP32, conectados a un servidor IoT en la nube mediante Wi-Fi y el protocolo MQTT.

Este sistema permite la gestión y el monitoreo remoto a través de una aplicación web, optimizar el uso de recursos, mejorar la productividad y reducir costos operativos. Su implementación requirió conocimientos en sistemas embebidos, sensores, protocolos de comunicación, desarrollo de software, técnicas de seguridad y la implementación de soluciones cloud.

Resumen

La presente memoria describe el desarrollo de un prototipo para monitorear y controlar de manera remota las condiciones climáticas en los invernaderos de la Facultad de Ciencias Forestales de la Universidad Nacional de Misiones. La solución propuesta integra sensores y actuadores conectados a un servidor remoto a través de una red inalámbrica y un protocolo de mensajería ligero, así como una aplicación web que permite la supervisión y el control a distancia del sistema.

Este trabajo permite optimizar el uso de recursos, mejorar la productividad y reducir costos operativos. Su implementación requirió conocimientos en sistemas embebidos, sensores, protocolos de comunicación, desarrollo de software, técnicas de seguridad y la implementación de soluciones cloud.

Índice general

Resumen	I
1. Introducción general	
1.1. Problemática actual	1
1.2. Motivación	2
1.3. Estado del arte	2
1.4. Objetivos y alcance	3
1.4.1. Objetivo principal	3
1.4.2. Objetivos específicos	3
1.4.3. Alcance del trabajo	3
1.5. Requerimientos	4
2. Introducción específica	7
2.1. Protocolos de comunicación	7
2.1.1. Wi-Fi	7
2.1.2. MQTT	7
2.1.3. TLS	8
2.1.4. HTTP	8
2.1.5. WebSocket	8
2.2. Componentes de hardware	9
2.2.1. Microcontrolador	9
2.2.2. Sensor de temperatura ambiente, humedad relativa y presión atmosférica	9
2.2.3. Sensor de luz digital	9
2.2.4. Sensor de dióxido de carbono	10
2.2.5. Sensor de detección de pH	10
2.2.6. Sensor de conductividad eléctrica	10
2.2.7. Sensor de sólidos disueltos totales	10
2.2.8. Sensor de temperatura digital sumergible	11
2.2.9. Sensor ultrasónico	11
2.2.10. Sensor de medición de consumo eléctrico	11
2.2.11. Módulo Relay	11
2.3. Desarrollo de firmware	12
2.3.1. MicroPython	12
2.4. Desarrollo Backend y API	12
2.4.1. FastAPI	12
2.4.2. MongoDB	12
2.5. Desarrollo Frontend	12
2.5.1. React	12
2.6. Infraestructura y despliegue	13
2.6.1. Docker	13
2.6.2. AWS IoT Core	13

Índice general

Resumen	I
1. Introducción general	1
1.1. Problemática actual	1
1.2. Motivación	2
1.3. Estado del arte	2
1.4. Objetivos y alcance	3
1.4.1. Objetivo principal	3
1.4.2. Objetivos específicos	3
1.4.3. Alcance del trabajo	3
1.5. Requerimientos	4
2. Introducción específica	7
2.1. Protocolos de comunicación	7
2.1.1. Wi-Fi	7
2.1.2. MQTT	7
2.1.3. TLS	8
2.2. Componentes de hardware	8
2.2.1. Microcontrolador	8
2.2.2. Sensor de temperatura ambiente, humedad relativa y presión atmosférica	9
2.2.3. Sensor de luz digital	9
2.2.4. Sensor de dióxido de carbono	9
2.2.5. Sensor de detección de pH	10
2.2.6. Sensor de conductividad eléctrica	10
2.2.7. Sensor de sólidos disueltos totales	10
2.2.8. Sensor de temperatura digital sumergible	10
2.2.9. Sensor ultrasónico	11
2.2.10. Sensor de medición de consumo eléctrico	11
2.2.11. Módulo Relay	11
2.3. Desarrollo de firmware	12
2.3.1. MicroPython	12
2.4. Desarrollo Backend y API	12
2.4.1. FastAPI	12
2.4.2. MongoDB	12
2.5. Desarrollo Frontend	12
2.5.1. React	12
2.6. Infraestructura y despliegue	13
2.6.1. Docker	13
2.6.2. AWS IoT Core	13
2.6.3. AWS EC2	13
2.7. Herramientas de desarrollo	13

2.6.3. AWS EC2	13
2.7. Herramientas de desarrollo	13
2.7.1. Visual Studio Code	13
2.7.2. Postman	13
2.7.3. Git	14
2.7.4. Github	14
Bibliografía	15

2.7.1. Visual Studio Code	13
2.7.2. Postman	14
2.7.3. Github	14
3. Diseño e implementación	15
3.1. Arquitectura del sistema	15
3.1.1. Capa de percepción	16
3.1.2. Capa de red	16
3.1.3. Capa de aplicación	16
3.2. Modelo de datos	17
3.2.1. Pruebas iniciales de sensores	17
3.2.2. Diseño del modelo de datos	17
3.3. Servidor IoT	19
3.3.1. Tecnologías utilizadas	19
Backend	19
Base de datos	19
Frontend	19
Despliegue	19
3.3.2. Arquitectura del servidor	20
Backend	20
Capa de datos	21
Frontend	21
3.4. Desarrollo del backend	21
3.4.1. Diseño de la API	22
3.4.2. Autenticación y autorización	23
3.4.3. Persistencia de datos	24
3.4.4. Comunicación con el broker MQTT	25
Pasos en AWS IoT Core	25
Implementación de MQTT en FastAPI	26
Comunicación con MQTT en FastAPI	26
3.4.5. Implementación de WebSockets	28
3.5. Desarrollo del frontend	28
3.6. Desarrollo de nodos sensores y actuadores	28
3.7. Despliegue del sistema	28
A. Resumen de endpoints de la API	29
B. Autenticación con JWT	33
C. Conexión al broker MQTT con FastAPI y AWS IoT SDK para Python	35
Bibliografía	39

Índice de figuras

1.1. Diagrama en bloques del sistema.	4
2.1. Arquitectura del protocolo MQTT.	8
2.2. Microcontrolador ESP-WROOM-32.	9
2.3. Sensor BME280.	9
2.4. Sensor BH1750.	9
2.5. Sensor MHZ19C.	10
2.6. Sensor PH-4502C.	10
2.7. Sensor CE.	10
2.8. Sensor TDS.	10
2.9. Sensor de temperatura DS18B20.	11
2.10. Sensor HC-SR04.	11
2.11. Sensor de medición de consumo eléctrico.	11
2.12. Relay de 2 Canales 5v 10A	11

Índice de figuras

1.1. Diagrama en bloques del sistema.	4
2.1. Arquitectura del protocolo MQTT.	8
2.2. Microcontrolador ESP-WROOM-32.	9
2.3. Sensor BME280.	9
2.4. Sensor BH1750.	9
2.5. Sensor MHZ19C.	9
2.6. Sensor PH-4502C.	10
2.7. Sensor CE.	10
2.8. Sensor TDS.	10
2.9. Sensor de temperatura DS18B20.	11
2.10. Sensor HC-SR04.	11
2.11. Sensor de medición de consumo eléctrico.	11
2.12. Relay de 2 Canales 5 V 10 A	11
3.1. Arquitectura de la solución propuesta.	15
3.2. Modelo de datos implementado.	18
3.3. Arquitectura del servidor del sistema IoT.	20
3.4. Esquema de autenticación y autorización.	23
3.5. Ejemplo de políticas de acceso en AWS IoT Core.	25
3.6. Pruebas en Postman. Test de conexión al broker MQTT.	26
3.7. Pruebas en Postman. Publicación de un mensaje en un tópico.	27
3.8. Cliente MQTT en FastAPI.	27
3.9. Datos almacenados en MongoDB.	28

Índice de tablas

1.1. Características de la competencia.....	2
---	---

Índice de tablas

1.1. Características de la competencia.....	2
3.1. Principales sensores y librerías utilizadas	17
3.2. Resumen de principales endpoints de la API	22
A.1. Resumen de principales endpoints de la API	29
A.2. Resumen de principales endpoints de la API	30
A.3. Resumen de principales endpoints de la API	31

Capítulo 1

Introducción general

Este capítulo presenta una visión general de los sistemas de gestión y monitoreo en invernaderos, se abordan los desafíos actuales y las oportunidades de mejora en el ámbito de la agricultura. Se describe la problemática relacionada con la falta de optimización en los sistemas de cultivo tradicionales. Además, se describen la motivación, los objetivos, el alcance y los requerimientos asociados a los diferentes componentes del sistema.

1.1. Problemática actual

La agricultura enfrenta desafíos crecientes en la optimización de la productividad y la eficiencia, especialmente en regiones con condiciones climáticas adversas y variables. Según la FAO (*Food and Agriculture Organization of the United Nations*) [1], para el año 2050, se estima que la población superará los 9 mil millones de personas, lo que demandará un aumento del 60 % en la producción de alimentos. Para abordar este desafío, es fundamental optimizar el uso del agua, mejorar la productividad agrícola y fomentar prácticas que contribuyan a la sostenibilidad ambiental.

Ante estos retos, los cultivos hidropónicos han surgido como una solución prometedora debido a su capacidad para utilizar los recursos de manera más eficiente. Entre sus principales ventajas se destacan la reducción en el consumo de agua [2], la posibilidad de cultivar durante todo el año en entornos controlados y un aumento significativo en la productividad, gracias a la mayor velocidad de crecimiento y rendimiento de los cultivos.

En la provincia de Misiones, la producción hidropónica ha experimentado un crecimiento notable en los últimos años [3], [4]. No obstante, persisten desafíos en la gestión eficiente de los recursos esenciales. Actualmente, la mayoría de los productores emplean sistemas de control basados en temporizadores programables, los cuales no consideran las variaciones ambientales. Esto implica la necesidad de intervenciones manuales frecuentes y mediciones directas, limitando la eficiencia del proceso.

La ausencia de un monitoreo en tiempo real impacta negativamente en la calidad y el rendimiento de los cultivos, aumentando los costos operativos y afectando la sostenibilidad ambiental debido a la implementación de prácticas poco optimizadas.

Capítulo 1

Introducción general

Este capítulo presenta una visión general de los sistemas de gestión y monitoreo en invernaderos, se abordan los desafíos actuales y las oportunidades de mejora en el ámbito de la agricultura. Se describe la problemática relacionada con la falta de optimización en los sistemas de cultivo tradicionales. Además, se describen la motivación, los objetivos, el alcance y los requerimientos asociados a los diferentes componentes del sistema.

1.1. Problemática actual

La agricultura enfrenta desafíos crecientes en la optimización de la productividad y la eficiencia, especialmente en regiones con condiciones climáticas adversas y variables. Según la FAO (*Food and Agriculture Organization of the United Nations*) [1], para el año 2050, se estima que la población superará los 9 mil millones de personas, lo que demandará un aumento del 60 % en la producción de alimentos. Para abordar este desafío, es fundamental optimizar el uso del agua, mejorar la productividad agrícola y fomentar prácticas que contribuyan a la sostenibilidad ambiental.

Ante estos retos, los cultivos hidropónicos han surgido como una solución prometedora debido a su capacidad para utilizar los recursos de manera más eficiente. Entre sus principales ventajas se destacan la reducción en el consumo de agua [2], la posibilidad de cultivar durante todo el año en entornos controlados y un aumento significativo en la productividad, gracias a la mayor velocidad de crecimiento y rendimiento de los cultivos.

En la provincia de Misiones, la producción hidropónica ha experimentado un crecimiento notable en los últimos años [3], [4]. No obstante, persisten desafíos en la gestión eficiente de los recursos esenciales. Actualmente, la mayoría de los productores emplean sistemas de control basados en temporizadores programables, los cuales no consideran las variaciones ambientales. Esto implica la necesidad de intervenciones manuales frecuentes y mediciones directas, limitando la eficiencia del proceso.

La ausencia de un monitoreo en tiempo real impacta negativamente en la calidad y el rendimiento de los cultivos, aumentando los costos operativos y afectando la sostenibilidad ambiental debido a la implementación de prácticas poco optimizadas.

1.2. Motivación

La motivación de este trabajo radica en el desarrollo e implementación de un sistema basado en IoT (*Internet of Things*) y de bajo costo, que permite monitorear en tiempo real y controlar de manera remota los invernaderos de la Facultad de Ciencias Forestales (FCF) de la Universidad Nacional de Misiones (UNaM).

Este sistema posibilita el registro continuo de diversas variables de interés, como temperatura ambiente, humedad relativa, dióxido de carbono (CO_2), niveles de nutrientes, y consumo de agua y energía, entre otros. Los datos generados están disponibles para docentes, estudiantes e investigadores, para su uso en la realización de tesis, investigaciones y trabajos académicos.

Así, el proyecto no solo tiene un impacto directo en la producción, sino también en la formación académica y el avance científico. Proporciona una plataforma de datos para el análisis y el desarrollo de nuevas soluciones tecnológicas, alineadas con las demandas actuales de sostenibilidad ambiental y seguridad alimentaria [5].

1.3. Estado del arte

En el mercado actual, existen diversas empresas que ofrecen soluciones comerciales para optimizar la gestión de invernaderos. Estas herramientas permiten el control automatizado de variables clave como temperatura, humedad, ventilación y circulación de nutrientes o riego. La tabla 1.1 presenta una comparación de algunas de las soluciones disponibles y sus características más relevantes.

TABLA 1.1. Características de la competencia.

Empresa	Características
Hidropónia FIL [6]	Ofrece servicios en comodato de sensores y actuadores para monitorear y controlar en tiempo real variables críticas como temperatura ambiente, humedad relativa, conductividad eléctrica, pH, riego e iluminación.
Hidrosense [7]	Ofrece productos para automatizar la inyección de nutrientes en el sistema de riego a través del control del nivel de la conductividad eléctrica, la temperatura y el nivel de pH. Ofrece una plataforma para la visualización del estado, reportes y el envío de alertas.
iPONIA [8]	Ofrece productos y una plataforma para monitorear y controlar el invernadero hidropónico. Integra sensores para medir el nivel de pH, conductividad eléctrica, temperatura de la solución, temperatura ambiente y humedad relativa del aire. También ofrece dosificadores para inyectar los fertilizantes a la solución nutritiva.
Growcast [9]	Ofrece productos y una plataforma para controlar cultivos a través de sensores y actuadores que procesan y reportan datos en tiempo real. Integra sensores para medir temperatura ambiente, humedad relativa y CO_2 . Realiza el control del riego, la iluminación y la ventilación.

1.2. Motivación

La motivación de este trabajo radica en el desarrollo e implementación de un sistema basado en IoT (*Internet of Things*) y de bajo costo, que permite monitorear en tiempo real y controlar de manera remota los invernaderos de la Facultad de Ciencias Forestales (FCF) de la Universidad Nacional de Misiones (UNaM).

Este sistema posibilita el registro continuo de diversas variables de interés, como temperatura ambiente, humedad relativa, dióxido de carbono (CO_2), niveles de nutrientes, y consumo de agua y energía, entre otros. Los datos generados están disponibles para docentes, estudiantes e investigadores, para su uso en la realización de tesis, investigaciones y trabajos académicos.

Así, el proyecto no solo tiene un impacto directo en la producción, sino también en la formación académica y el avance científico. Proporciona una plataforma de datos para el análisis y el desarrollo de nuevas soluciones tecnológicas, alineadas con las demandas actuales de sostenibilidad ambiental y seguridad alimentaria [5].

1.3. Estado del arte

En el mercado actual, existen diversas empresas que ofrecen soluciones comerciales para optimizar la gestión de invernaderos. Estas herramientas permiten el control automatizado de variables clave como temperatura, humedad, ventilación y circulación de nutrientes o riego. La tabla 1.1 presenta una comparación de algunas de las soluciones disponibles y sus características más relevantes.

TABLA 1.1. Características de la competencia.

Empresa	Características
Hidropónia FIL [6]	Ofrece servicios en comodato de sensores y actuadores para monitorear y controlar en tiempo real variables críticas como temperatura ambiente, humedad relativa, conductividad eléctrica, pH, riego e iluminación.
Hidrosense [7]	Ofrece productos para automatizar la inyección de nutrientes en el sistema de riego a través del control del nivel de la conductividad eléctrica, la temperatura y el nivel de pH. Ofrece una plataforma para la visualización del estado, reportes y el envío de alertas.
iPONIA [8]	Ofrece productos y una plataforma para monitorear y controlar el invernadero hidropónico. Integra sensores para medir el nivel de pH, conductividad eléctrica, temperatura de la solución, temperatura ambiente y humedad relativa del aire. También ofrece dosificadores para inyectar los fertilizantes a la solución nutritiva.
Growcast [9]	Ofrece productos y una plataforma para controlar cultivos a través de sensores y actuadores que procesan y reportan datos en tiempo real. Integra sensores para medir temperatura ambiente, humedad relativa y CO_2 . Realiza el control del riego, la iluminación y la ventilación.

1.4. Objetivos y alcance

1.4.1. Objetivo principal

Diseñar y desarrollar un prototipo de sistema para el monitoreo y control remoto de las condiciones climáticas en invernaderos, mediante sensores y actuadores conectados a través de Wi-Fi, un servidor IoT en la nube y una aplicación web, con el fin de optimizar el uso de los recursos, reducir costos operativos y mejorar la sostenibilidad ambiental, además de servir como plataforma de datos para la investigación académica y científica.

1.4.2. Objetivos específicos

- Implementar una arquitectura IoT basada en Wi-Fi para monitorear sensores y actuadores en tiempo real.
- Desarrollar un servidor IoT en la nube para la recolección, almacenamiento y procesamiento de los datos obtenidos.
- Diseñar una aplicación web que permita la visualización en tiempo real y el control remoto de las condiciones del invernadero.
- Facilitar el acceso a los datos generados para su uso en investigaciones académicas, trabajos finales y estudios específicos.

1.4.3. Alcance del trabajo

El alcance del trabajo incluyó las siguientes tareas:

- Diseño e implementación de nodos IoT.
 - Selección de sensores, actuadores y microcontroladores.
 - Configuración de conexión Wi-Fi en nodos sensores y actuadores.
 - Desarrollo de firmware para la adquisición de datos de los sensores y el control de los actuadores.
- Comunicación y Protocolos.
 - Configuración de un servidor IoT para gestión de mensajes entre nodos y aplicaciones.
 - Transmisión de datos al servidor IoT mediante MQTT (*Message Queue Telemetry Transport*).
 - Cifrado de comunicaciones mediante TLS (*Transport Layer Security*).
- Desarrollo de Software.
 - Diseño e implementación de una base de datos para almacenar los datos recolectados por los sensores y permitir su consulta y análisis.
 - Diseño y desarrollo de una API (*Application Programming Interface*) REST (*Representational State Transfer*) que permita la comunicación con el sistema utilizando HTTP (*Hypertext Transfer Protocol*), MQTT y WebSockets.

1.4. Objetivos y alcance

1.4.1. Objetivo principal

Diseñar y desarrollar un prototipo de sistema para el monitoreo y control remoto de las condiciones climáticas en invernaderos, mediante sensores y actuadores conectados a través de Wi-Fi, un servidor IoT en la nube y una aplicación web, con el fin de optimizar el uso de los recursos, reducir costos operativos y mejorar la sostenibilidad ambiental, además de servir como plataforma de datos para la investigación académica y científica.

1.4.2. Objetivos específicos

- Implementar una arquitectura IoT basada en Wi-Fi para monitorear sensores y actuadores en tiempo real.
- Desarrollar un servidor IoT en la nube para la recolección, almacenamiento y procesamiento de los datos obtenidos.
- Diseñar una aplicación web que permita la visualización en tiempo real y el control remoto de las condiciones del invernadero.
- Facilitar el acceso a los datos generados para su uso en investigaciones académicas, trabajos finales y estudios específicos.

1.4.3. Alcance del trabajo

El alcance del trabajo incluyó las siguientes tareas:

- Diseño e implementación de nodos IoT.
 - Selección de sensores, actuadores y microcontroladores.
 - Configuración de conexión Wi-Fi en nodos sensores y actuadores.
 - Desarrollo de firmware para la adquisición de datos de los sensores y el control de los actuadores.
- Comunicación y Protocolos.
 - Configuración de un servidor IoT para gestión de mensajes entre nodos y aplicaciones.
 - Transmisión de datos al servidor IoT mediante MQTT (*Message Queue Telemetry Transport*).
 - Cifrado de comunicaciones mediante TLS (*Transport Layer Security*).
- Desarrollo de Software.
 - Diseño e implementación de una base de datos para almacenar los datos recolectados por los sensores y permitir su consulta y análisis.
 - Diseño y desarrollo de una API (*Application Programming Interface*) REST (*Representational State Transfer*) que permita la comunicación con el sistema utilizando HTTP (*Hypertext Transfer Protocol*), MQTT y WebSockets.

1.5. Requerimientos

5

- 1) Nodos ambientales: temperatura ambiente, humedad relativa, presión atmosférica, nivel de luminosidad y nivel de CO_2 .
- 2) Nodos de solución nutritiva: valores de pH (potencial de Hidrógeno), conductividad eléctrica (CE) y TDS (*Total Dissolved Solids*); nivel y temperatura de la solución.
- 3) Nodos de consumos: agua, nutrientes y energía eléctrica.
- g) Los nodos actuadores deben transmitir al servidor IoT:
 - 1) Configuración remota de parámetros por cada canal.
 - 2) Reporte del estado de cada canal.
- h) Los nodos actuadores deben recibir desde el servidor IoT:
 - 1) Comandos de activación/desactivación remota de canales.
2. Broker MQTT:
 - a) Soportar conexiones cifradas mediante TLS.
 - b) Poseer comunicación bidireccional (publicación/suscripción).
 - c) Implementar QoS (*Quality of Service*) para garantizar entrega de mensajes.
3. Frontend (aplicación web)
 - a) Interfaz intuitiva y responsive (accesible desde móviles y escritorio).
 - b) Autenticación de usuarios mediante credenciales.
 - c) Realización de las operaciones CRUD (Crear, Leer, Actualizar, Eliminar).
 - d) Visualización en tiempo real de datos de sensores y actuadores.
 - e) Envío remoto de comandos y configuraciones.
 - f) Acceso a datos históricos mediante gráficos y tablas.
 - g) Tablero interactivo para monitoreo y control centralizado.
4. Backend:
 - a) Soportar conexiones seguras mediante TLS.
 - b) Implementar JWT (*JSON Web Token*).
 - c) Persistencia de los datos.
 - d) Soporte para métodos HTTP (CRUD y reportes), WebSockets (datos en tiempo real) y MQTT (interacción con dispositivos).
5. Requerimientos de documentación:
 - a) Se entregará el código del sistema, que incluye todos los componentes desarrollados (sensores, actuadores, broker MQTT, frontend, backend y API).
 - b) Se entregarán las guías y diagramas de instalación, configuración y operación.

1.5. Requerimientos

5

- 1) Nodos ambientales: temperatura ambiente, humedad relativa, presión atmosférica, nivel de luminosidad y nivel de CO_2 .
- 2) Nodos de solución nutritiva: valores de pH (potencial de Hidrógeno), conductividad eléctrica (CE) y TDS (*del inglés, Total Dissolved Solids*); nivel y temperatura de la solución.
- 3) Nodos de consumos: agua, nutrientes y energía eléctrica.
- g) Los nodos actuadores deben transmitir al servidor IoT:
 - 1) Configuración remota de parámetros por cada canal.
 - 2) Reporte del estado de cada canal.
- h) Los nodos actuadores deben recibir desde el servidor IoT:
 - 1) Comandos de activación/desactivación remota de canales.
2. Broker MQTT:
 - a) Soportar conexiones cifradas mediante TLS.
 - b) Poseer comunicación bidireccional (publicación/suscripción).
 - c) Implementar QoS (*del inglés, Quality of Service*) para garantizar entrega de mensajes.
3. Frontend (aplicación web)
 - a) Interfaz intuitiva y responsive (accesible desde móviles y escritorio).
 - b) Autenticación de usuarios mediante credenciales.
 - c) Realización de las operaciones CRUD (Crear, Leer, Actualizar, Eliminar).
 - d) Visualización en tiempo real de datos de sensores y actuadores.
 - e) Envío remoto de comandos y configuraciones.
 - f) Acceso a datos históricos mediante gráficos y tablas.
 - g) Tablero interactivo para monitoreo y control centralizado.
4. Backend:
 - a) Tener conexiones seguras mediante TLS.
 - b) Implementar JWT (*del inglés, JSON Web Token*).
 - c) Realizar la persistencia de los datos.
 - d) Soportar métodos HTTP (CRUD y reportes), WebSockets (datos en tiempo real) y MQTT (interacción con dispositivos).
5. Requerimientos de documentación:
 - a) Se entregará el código del sistema, que incluye todos los componentes desarrollados (sensores, actuadores, broker MQTT, frontend, backend y API).
 - b) Se entregarán las guías y diagramas de instalación, configuración y operación.

Capítulo 2

Introducción específica

En este capítulo se presentan los protocolos de comunicación, componentes de hardware y herramientas de software utilizados en el desarrollo del trabajo. Se detallan las características y sus especificaciones técnicas.

2.1. Protocolos de comunicación

En esta sección se describen los diferentes protocolos de comunicación utilizados en el desarrollo del trabajo.

2.1.1. Wi-Fi

Wi-Fi es el nombre comercial propiedad de la Wi-Fi Alliance para designar su familia de protocolos de comunicación inalámbrica basados en el estándar IEEE 802.11 para redes de área local sin cables [10].

El estandar identifica dos modos principales de topología de red: infraestructura y ad-hoc.

- Modo infraestructura: los dispositivos se conectan a una red inalámbrica a través de un router o AP (*Access Point*) inalámbrico, como en las WLAN. Los AP se conectan a la infraestructura de la red mediante el sistema de distribución conectado por cable o de manera inalámbrica.
- Modo ad-hoc: los dispositivos se conectan directamente entre sí sin necesidad de un punto de acceso.

2.1.2. MQTT

MQTT es un protocolo de mensajería estándar internacional OASIS [11] para el Internet de las Cosas (IoT). Está diseñado como un transporte de mensajería de publicación/suscripción extremadamente ligero, ideal para conectar dispositivos remotos con un consumo de código reducido y un ancho de banda de red mínimo.

MQTT es un protocolo ligero basado en TCP/IP [12] que sigue un modelo de publicación/suscripción, donde:

- Broker: **Funciona** como un servidor central que recibe los mensajes de los clientes y los distribuye a los suscriptores correspondientes, **actúa** como intermediario en la comunicación.
- Cliente: **Puede** ser un dispositivo que publica mensajes en un tópico o que recibe mensajes al estar suscrito a un tópico.

Capítulo 2

Introducción específica

En este capítulo se presentan los protocolos de comunicación, componentes de hardware y herramientas de software utilizados en el desarrollo del trabajo. Se detallan las características y sus especificaciones técnicas.

2.1. Protocolos de comunicación

En esta sección se describen los diferentes protocolos de comunicación utilizados en el desarrollo del trabajo.

2.1.1. Wi-Fi

Wi-Fi es el nombre comercial propiedad de la Wi-Fi Alliance para designar a su familia de protocolos de comunicación inalámbrica basados en el estándar IEEE 802.11 para redes de área local sin cables [10].

El estandar identifica dos modos principales de topología de red: infraestructura y ad-hoc.

- Modo infraestructura: los dispositivos se conectan a una red inalámbrica a través de un router o AP (*del inglés, Access Point*) inalámbrico, como en las WLAN. Los AP se conectan a la infraestructura de la red mediante el sistema de distribución conectado por cable o de manera inalámbrica.
- Modo ad-hoc: los dispositivos se conectan directamente entre sí sin necesidad de un punto de acceso.

2.1.2. MQTT

MQTT es un protocolo de mensajería estándar internacional OASIS [11] para Internet de las Cosas (IoT). Está diseñado como un transporte de mensajería de publicación/suscripción extremadamente ligero, ideal para conectar dispositivos remotos con un consumo de código reducido y un ancho de banda de red mínimo.

MQTT es un protocolo ligero basado en TCP/IP [12] que sigue un modelo de publicación/suscripción, donde:

- Broker: **funciona** como un servidor central que recibe los mensajes de los clientes y los distribuye a los suscriptores correspondientes, **actúa** como intermediario en la comunicación.

- Tópico: Es la dirección a la que se envían los mensajes en MQTT. El broker se encarga de distribuirlos a los clientes suscritos. Los temas se organizan en una estructura jerárquica de tópicos.

La figura 2.1 muestra la arquitectura del protocolo MQTT.

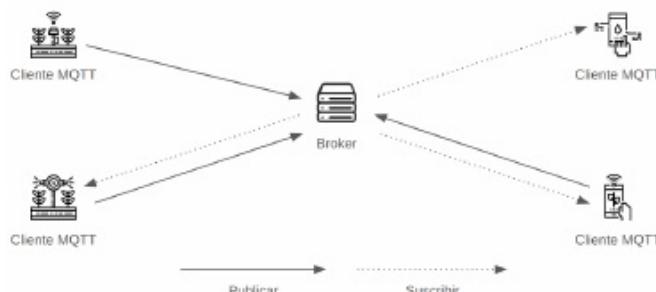


FIGURA 2.1. Arquitectura del protocolo MQTT.

2.1.3. TLS

TLS es un protocolo de seguridad criptográfica diseñado para garantizar la privacidad y la integridad de los datos en comunicaciones sobre redes, como Internet [13]. Opera sobre la capa de transporte y permite autenticación, cifrado de datos y protección contra manipulación.

TLS se utiliza para garantizar la confidencialidad de los protocolos de aplicación (MQTT, HTTP y WebSocket) [14].

2.1.4. HTTP

HTTP es un protocolo a nivel de aplicación que opera sobre TCP/IP [15] y está diseñado para sistemas de información distribuidos, colaborativos e hipermédia. Está basado en el modelo cliente-servidor, diseñado para la transferencia de recursos web [16].

Este protocolo es asíncrono, lo que significa que un cliente puede enviar una petición sin necesidad de mantener una conexión activa mientras espera la respuesta. Esto optimiza el uso de recursos en la red [17] y permite que cada interacción sea independiente, gracias a su diseño sin estado.

2.1.5. WebSocket

WebSocket es un protocolo de comunicación bidireccional y simultánea (full-duplex) que mantiene una conexión persistente entre un cliente y un servidor sobre una única conexión TCP [18]. Para establecer la conexión, utiliza una secuencia de solicitud-respuesta HTTP estándar. Una vez conectados, la API WebSocket proporciona una interfaz para la lectura y escritura de datos de manera asíncrona y en modo dúplex [19]. Esto lo hace ideal para aplicaciones que requieren baja latencia y actualizaciones en tiempo real, como chats, juegos en línea y sistemas de monitoreo.

- Cliente: puede ser un dispositivo que publica mensajes en un tópico o que recibe mensajes al estar suscrito a un tópico.

- Tópico: es la dirección a la que se envían los mensajes en MQTT. El broker se encarga de distribuirlos a los clientes suscritos. Los temas se organizan en una estructura jerárquica de tópicos.

La figura 2.1 muestra la arquitectura del protocolo MQTT.

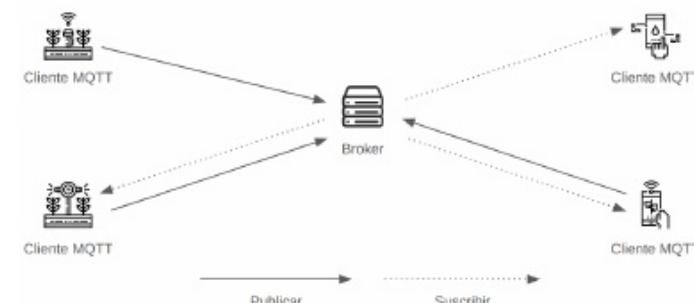


FIGURA 2.1. Arquitectura del protocolo MQTT.

2.1.3. TLS

TLS es un protocolo de seguridad criptográfica diseñado para garantizar la privacidad y la integridad de los datos en comunicaciones sobre redes, como Internet [13]. Opera sobre la capa de transporte y permite autenticación, cifrado de datos y protección contra manipulación.

TLS se utiliza para garantizar la confidencialidad de los protocolos de aplicación (MQTT [11], HTTP [14] y WebSocket [15]) [16].

2.2. Componentes de hardware

En esta sección se describen los diferentes elementos de hardware utilizados en el desarrollo del trabajo.

2.2.1. Microcontrolador

El microcontrolador ESP-WROOM-32 (figura 2.2), es un chip de tipo SoC (del inglés, *System on Chip*) de bajo costo y bajo consumo de energía que integra WiFi, Bluetooth y Bluetooth LE en un solo paquete. El ESP-WROOM-32 [17] es un microcontrolador de 32 bits con una arquitectura Xtensa LX6 de doble núcleo, lo que le permite ejecutar dos hilos de ejecución simultáneos. Además, cuenta con una amplia gama de periféricos, como UART, I2C, SPI y ADC, que lo hace ideal para aplicaciones de IoT.

2.2. Componentes de hardware

En esta sección se describen los diferentes elementos de hardware utilizados en el desarrollo del trabajo.

2.2.1. Microcontrolador

El microcontrolador ESP-WROOM-32 (figura 2.2), es un chip de tipo SoC (*System on Chip*) de bajo costo y bajo consumo de energía que integra Wi-Fi, Bluetooth y Bluetooth LE en un solo paquete. El ESP-WROOM-32 [20] es un microcontrolador de 32 bits con una arquitectura Xtensa LX6 de doble núcleo, lo que le permite ejecutar dos hilos de ejecución simultáneos. Además, cuenta con una amplia gama de periféricos, como UART, I2C, SPI y ADC, que lo hace ideal para aplicaciones de IoT.



FIGURA 2.2. Microcontrolador ESP-WROOM-32.

2.2.2. Sensor de temperatura ambiente, humedad relativa y presión atmosférica

El BME280 (figura 2.3) es un sensor digital de alta precisión para la medición de temperatura ambiente, humedad relativa y presión atmosférica. Se comunica a través de las interfaces I2C y SPI y ofrece una precisión de $\pm 1^{\circ}\text{C}$ para la temperatura ambiente, $\pm 3\%$ para la humedad relativa y $\pm 1 \text{ hPa}$ para la presión atmosférica [21].



FIGURA 2.3. Sensor BME280.

2.2.3. Sensor de luz digital

El BH1750 (figura 2.4) es un sensor digital de intensidad luminosa que mide la iluminación ambiental en lux. Utiliza la interfaz I2C para la comunicación y puede medir niveles de luz en un rango de 1 a 65.535 lux, con una precisión de 1 lux [22].



FIGURA 2.4. Sensor BH1750.



FIGURA 2.2. Microcontrolador ESP-WROOM-32.

2.2.2. Sensor de temperatura ambiente, humedad relativa y presión atmosférica

El BME280 (figura 2.3) es un sensor digital de alta precisión para la medición de temperatura ambiente, humedad relativa y presión atmosférica. Se comunica a través de las interfaces I2C y SPI y ofrece una precisión de $\pm 1^{\circ}\text{C}$ para la temperatura ambiente, $\pm 3\%$ para la humedad relativa y $\pm 1 \text{ hPa}$ para la presión atmosférica [18].



FIGURA 2.3. Sensor BME280.

2.2.3. Sensor de luz digital

El BH1750 (figura 2.4) es un sensor digital de intensidad luminosa que mide la iluminación ambiental en lux. Utiliza la interfaz I2C para la comunicación y puede medir niveles de luz en un rango de 1 a 65.535 lux, con una precisión de 1 lux [19].



FIGURA 2.4. Sensor BH1750.

2.2.4. Sensor de dióxido de carbono

El sensor MH-Z19C (figura 2.5) es un detector de CO_2 por NDIR (del inglés, *Non Dispersive Infrared Detector*). Se comunica a través de la interfaz UART y es capaz de medir la concentración de CO_2 en un rango de 0 a 5000 ppm con una precisión de 50 ppm [20].



FIGURA 2.5. Sensor MHZ19C.

2.2.4. Sensor de dióxido de carbono

El sensor MH-Z19C (figura 2.5) es un detector de CO_2 por NDIR (*Non Dispersive Infrared Detector*). Se comunica a través de la interfaz UART y es capaz de medir la concentración de CO_2 en un rango de 0 a 5000 ppm con una precisión de 50 ppm [23].



FIGURA 2.5. Sensor MHZ19C.

2.2.5. Sensor de detección de pH

El sensor PH-4502C (figura 2.6) mide la acidez o alcalinidad del líquido mediante un electrodo de vidrio. Se comunica a través de la interfaz analógica y es capaz de medir el pH en un rango de 0 a 14 [24].



FIGURA 2.6. Sensor PH-4502C.

2.2.6. Sensor de conductividad eléctrica

El sensor de CE (figura 2.7) mide la capacidad de una solución para conducir electricidad, lo cual depende de la presencia de iones. A mayor concentración de iones, mayor es la conductividad [25]. Este sensor se comunica a través de una interfaz analógica y puede medir la conductividad en un rango de 0 a 20 mS/cm [26].



FIGURA 2.7. Sensor CE.

2.2.7. Sensor de sólidos disueltos totales

El sensor TDS (figura 2.8) mide la cantidad de sales, minerales y metales que se encuentran disueltos en el líquido [27]. Se comunica a través de la interfaz analógica y es capaz de medir la concentración de TDS en un rango de 0 a 1000 ppm [28].



FIGURA 2.8. Sensor TDS.

2.2.5. Sensor de detección de pH

El sensor PH-4502C (figura 2.6) mide la acidez o alcalinidad del líquido mediante un electrodo de vidrio. Se comunica a través de la interfaz analógica y es capaz de medir el pH en un rango de 0 a 14 [21].



FIGURA 2.6. Sensor PH-4502C.

2.2.6. Sensor de conductividad eléctrica

El sensor de CE (figura 2.7) mide la capacidad de una solución para conducir electricidad, lo cual depende de la presencia de iones. A mayor concentración de iones, mayor es la conductividad [22]. Este sensor se comunica a través de una interfaz analógica y puede medir la conductividad en un rango de 0 a 20 mS/cm [23].



FIGURA 2.7. Sensor CE.

2.2.7. Sensor de sólidos disueltos totales

El sensor TDS (figura 2.8) mide la cantidad de sales, minerales y metales que se encuentran disueltos en la solución [24]. Se comunica a través de la interfaz analógica y es capaz de medir la concentración de TDS en un rango de 0 a 1000 ppm [25].



FIGURA 2.8. Sensor TDS.

2.2.8. Sensor de temperatura digital sumergible

El DS18B20 (figura 2.9) es un sensor digital de temperatura sumergible. Se comunica a través de la interfaz OneWire y puede medir la temperatura en un rango de -55 °C a 125 °C con una precisión de ±0.5 °C [26].

2.2.8. Sensor de temperatura digital sumergible

El DS18B20 (figura 2.9) es un sensor digital de temperatura sumergible en líquidos. Se comunica a través de la interfaz 1-Wire y puede medir la temperatura en un rango de -55°C a 125°C con una precisión de ±0.5°C [29].



FIGURA 2.9. Sensor de temperatura DS18B20.

2.2.9. Sensor ultrasónico

El sensor HC-SR04 (figura 2.10) mide distancias por ultrasonido en un rango de 2 cm a 400 cm con una precisión de 3 mm. Se comunica a través de la interfaz GPIO [30].



FIGURA 2.10. Sensor HC-SR04.

2.2.10. Sensor de medición de consumo eléctrico

El sensor PZEM-004T (figura 2.11) es un módulo de medición de parámetros eléctricos que mide la tensión, corriente, potencia activa y energía consumida. Se comunica a través de la interfaz UART y es capaz de medir la tensión en un rango de 80 a 260 V, la corriente en un rango de 0 a 100 A, y la potencia en un rango de 0 a 22 kW [31].



FIGURA 2.11. Sensor de medición de consumo eléctrico.

2.2.11. Módulo Relay

El módulo Relay (figura 2.12) es un actuador eléctrico de dos canales optocoplados que permite el control de encendido y apagado de dispositivos eléctricos. Se comunica a través de la interfaz GPIO y es capaz de controlar dispositivos de hasta 10 A y 250 VAC [32].



FIGURA 2.12. Relay de 2 Canales 5V 10A



FIGURA 2.9. Sensor de temperatura DS18B20.

2.2.9. Sensor ultrasónico

El sensor HC-SR04 (figura 2.10) mide distancias por ultrasonido en un rango de 2 cm a 400 cm con una precisión de 3 mm. Se comunica a través de la interfaz GPIO [27].



FIGURA 2.10. Sensor HC-SR04.

2.2.10. Sensor de medición de consumo eléctrico

El sensor PZEM-004T (figura 2.11) es un módulo de medición de parámetros eléctricos que mide la tensión, corriente, potencia activa y energía consumida. Se comunica a través de la interfaz UART y es capaz de medir la tensión en un rango de 80 a 260 V, la corriente en un rango de 0 a 100 A, y la potencia en un rango de 0 a 22 kW [28].



FIGURA 2.11. Sensor de medición de consumo eléctrico.

2.2.11. Módulo Relay

El módulo Relay (figura 2.12) es un actuador eléctrico de dos canales optocoplados que permite el control de encendido y apagado de dispositivos eléctricos. Se comunica a través de la interfaz GPIO y es capaz de controlar dispositivos de hasta 10 A y 250 VAC [29].



FIGURA 2.12. Relay de 2 Canales 5V 10A

2.3. Desarrollo de firmware

En esta sección se describe la herramienta de software utilizada para la programación de los microcontroladores ESP32.

2.3.1. MicroPython

MicroPython es una implementación optimizada de Python 3 para microcontroladores y sistemas embebidos. Está diseñado para ejecutarse en dispositivos con recursos limitados, como el ESP32, y proporciona una forma sencilla de programar microcontroladores utilizando un lenguaje de alto nivel como Python [33].

Su facilidad de uso, la amplia disponibilidad de bibliotecas y la reducción del tiempo de desarrollo lo convierten en una opción eficiente. Además, al ser un lenguaje interpretado, posibilita la ejecución interactiva de pruebas y depuración, facilitando la identificación y corrección de errores en el código [34].

2.4. Desarrollo Backend y API

En esta sección se presentan las herramientas de software utilizadas en el desarrollo del backend y la API REST.

2.4.1. FastAPI

FastAPI es un framework moderno para la construcción de APIs REST rápidas y escalables en Python. Está diseñado para ser fácil de usar, rápido de desarrollar y altamente eficiente en términos de rendimiento. FastAPI utiliza Python 3.6+ y aprovecha las características de tipado estático de Python para proporcionar una API autodocumentada y con validación de tipos integrada [35].

2.4.2. MongoDB

MongoDB es una base de datos NoSQL (*Not Only SQL*) de código abierto y orientada a documentos que proporciona una forma flexible y escalable de almacenar y recuperar datos. Utiliza un modelo de datos basado en documentos BSON (*Binary JavaScript Object Notation*) que permite almacenar datos de forma anidada y sin esquema fijo, lo que facilita la manipulación y consulta de datos no estructurados [36].

2.5. Desarrollo Frontend

2.5.1. React

React es una biblioteca de JavaScript de código abierto para construir interfaces de usuario interactivas y reutilizables. Desarrollada por Facebook, React permite crear componentes de interfaz de usuario que se actualizan de forma eficiente cuando cambian los datos, lo que facilita la creación de aplicaciones web rápidas y dinámicas [37].

2.3. Desarrollo de firmware

En esta sección se describe la herramienta de software utilizada para la programación de los microcontroladores ESP32.

2.3.1. MicroPython

MicroPython es una implementación optimizada de Python 3 para microcontroladores y sistemas embebidos. Está diseñado para ejecutarse en dispositivos con recursos limitados, como el ESP32, y proporciona una forma sencilla de programar microcontroladores con un lenguaje de alto nivel como Python [30].

Su facilidad de uso, la amplia disponibilidad de bibliotecas y la reducción del tiempo de desarrollo lo convierten en una opción eficiente. Además, al ser un lenguaje interpretado, posibilita la ejecución interactiva de pruebas y depuración, facilitando la identificación y corrección de errores en el código [31].

2.4. Desarrollo Backend y API

En esta sección se presentan las herramientas de software utilizadas en el desarrollo del backend y la API REST.

2.4.1. FastAPI

FastAPI es un framework moderno para la construcción de APIs REST rápidas y escalables en Python. Está diseñado para ser fácil de usar, rápido de desarrollar y altamente eficiente en términos de rendimiento. FastAPI utiliza Python 3.6+ y aprovecha las características de tipado estático de Python para proporcionar una API autodocumentada y con validación de tipos integrada [32].

2.4.2. MongoDB

MongoDB es una base de datos NoSQL (*del inglés, Not Only SQL*) de código abierto y orientada a documentos que proporciona una forma flexible y escalable de almacenar y recuperar datos. Utiliza un modelo de datos basado en documentos que almacena datos en un formato similar a JSON (*del inglés, JavaScript Object Notation*) llamado BSON (*del inglés, Binary JSON*) que permite almacenar datos de forma anidada y sin esquema fijo, lo que facilita la manipulación y consulta de datos no estructurados [33].

2.5. Desarrollo Frontend

En esta sección se presentan las herramientas de software utilizadas en el desarrollo del frontend.

2.5.1. React

React es una librería de JavaScript de código abierto para construir interfaces de usuario interactivas y reutilizables. Desarrollada por Facebook, React permite crear componentes de interfaz de usuario que se actualizan de forma eficiente

2.6. Infraestructura y despliegue

2.6.1. Docker

Docker es una plataforma de código abierto que permite a los desarrolladores y a los equipos de operaciones construir, empaquetar y desplegar aplicaciones en contenedores. Los contenedores son unidades de software ligeros y portátiles que incluyen todo lo necesario para ejecutar una aplicación, incluidas las **bibliotecas**, las dependencias y el código [38].

Docker facilita la creación de entornos de desarrollo y despliegue consistentes y reproducibles, lo que garantiza que las aplicaciones se ejecuten de la misma manera en cualquier entorno.

2.6.2. AWS IoT Core

AWS IoT Core es un servicio de AWS (*Amazon Web Services*) que permite a los dispositivos conectarse de forma segura a la nube y comunicarse entre sí a través de protocolos de comunicación estándar como MQTT y HTTP. Proporciona una infraestructura escalable y segura para la gestión de dispositivos, la recopilación de datos y la integración con otros servicios de AWS [39]. Utiliza TLS para cifrar la comunicación entre los dispositivos y la nube, garantizando la confidencialidad y la integridad de los datos.

2.6.3. AWS EC2

Amazon EC2 (*Elastic Compute Cloud*) es un servicio de AWS que proporciona **capacidad informática escalable** en la nube. Permite a los usuarios **lanzar** instancias virtuales en la nube con diferentes configuraciones de CPU, **memoria**, **almacenamiento** y red, lo que facilita la implementación de aplicaciones escalables y de alta disponibilidad [40].

2.7. Herramientas de desarrollo

2.7.1. Visual Studio Code

Visual Studio Code, comúnmente abreviado como VS Code, es un entorno de desarrollo integrado (IDE, *Integrated Development Environment*) de **código abierto**, altamente extensible y multiplataforma compatible con Windows, macOS y Linux [41].

VS Code es un editor de código ligero y rápido con soporte para cientos de lenguajes de programación y extensiones que permiten personalizar y mejorar la funcionalidad del editor. Además, cuenta con herramientas de depuración integradas, control de versiones y terminal integrada.

2.7.2. Postman

Postman es una plataforma de colaboración para el desarrollo de APIs que permite a los desarrolladores diseñar, probar y documentar de forma rápida. Proporciona una interfaz gráfica intuitiva para enviar solicitudes HTTP a un servidor y visualizar las respuestas, lo que facilita la depuración y el desarrollo de APIs [42].

cuando cambian los datos, lo que facilita la creación de aplicaciones web rápidas y dinámicas [34].

2.6. Infraestructura y despliegue

En esta sección se presentan las herramientas de software utilizadas en la infraestructura y despliegue del sistema.

2.6.1. Docker

Docker es una plataforma de código abierto que permite a los desarrolladores y a los equipos de operaciones construir, empaquetar y desplegar aplicaciones en contenedores. Los contenedores son unidades de software ligeros y portátiles que incluyen todo lo necesario para ejecutar una aplicación, incluidas las **librerías**, las dependencias y el código [35].

Docker facilita la creación de entornos de desarrollo y despliegue consistentes y reproducibles, lo que garantiza que las aplicaciones se ejecuten de la misma manera en cualquier entorno.

2.6.2. AWS IoT Core

AWS IoT Core es un servicio de AWS (*Amazon Web Services*) que **permite** a los dispositivos conectarse de forma segura a la nube y comunicarse entre sí a través de protocolos de comunicación estándar como MQTT y HTTP. **Proporciona** una infraestructura escalable y segura para la gestión de dispositivos, la recopilación de datos y la integración con otros servicios de AWS [36]. Utiliza TLS para cifrar la comunicación entre los dispositivos y la nube, para garantizar la confidencialidad y la integridad de los datos.

2.6.3. AWS EC2

Amazon EC2 (*Elastic Compute Cloud*) es un servicio de AWS que **proporciona** **capacidad informática escalable** en la nube. Permite a los usuarios **lanzar** instancias virtuales en la nube con diferentes configuraciones de CPU, **memoria**, **almacenamiento** y red, lo que facilita la implementación de aplicaciones escalables y de alta disponibilidad [37].

2.7. Herramientas de desarrollo

En esta sección se presentan las herramientas de software utilizadas en el desarrollo del sistema.

2.7.1. Visual Studio Code

Visual Studio Code, comúnmente abreviado como VS Code, es un entorno de desarrollo integrado (IDE, *Integrated Development Environment*) de **código abierto**, altamente extensible y multiplataforma compatible con Windows, macOS y Linux [38].

2.7.3. Git

Git es un sistema de control de versiones distribuido de código abierto diseñado para gestionar proyectos de software de cualquier tamaño con rapidez y eficiencia. Permite a los desarrolladores trabajar en paralelo en un mismo proyecto, realizar seguimiento de los cambios, revertir a versiones anteriores y colaborar en el desarrollo de software [43].

2.7.4. Github

Github es una plataforma de alojamiento de repositorios Git que permite a los desarrolladores colaborar en proyectos de software de forma distribuida. Proporciona herramientas para gestionar el código fuente, realizar seguimiento de los cambios, revisar el código, realizar integración continua y despliegue automático [44].

VS Code es un editor de código ligero y rápido con soporte para muchos lenguajes de programación y extensiones que permiten personalizar y mejorar la funcionalidad del editor. Además, cuenta con herramientas de depuración integradas, control de versiones y terminal integrada.

2.7.2. Postman

Postman es una plataforma de colaboración para el desarrollo de APIs que permite a los desarrolladores diseñar, probar y documentar de forma rápida. Proporciona una interfaz gráfica intuitiva para enviar solicitudes HTTP a un servidor y visualizar las respuestas, lo que facilita la depuración y el desarrollo de APIs [39].

2.7.3. Github

Github es una plataforma de alojamiento de repositorios Git [40] que permite a los desarrolladores colaborar en proyectos de software de forma distribuida. Proporciona herramientas para gestionar el código fuente, realizar seguimiento de los cambios, revisar el código, realizar integración continua y despliegue automático [41].

Bibliografía

- [1] Global Agricultural Productivity (GAP). 2016 *Global Agricultural Productivity Report*. Inf. tén. Documento en línea. Global Agricultural Productivity, 2016. URL: https://globalagriculturalproductivity.org/wp-content/uploads/2019/01/2016_GAP_Report.pdf (visitado 20-03-2025).
- [2] Raquel Salazar-Moreno, Abraham Rojano-Aguilar, Ireneo Lorenzo López-Cruz. «La eficiencia en el uso del agua en la agricultura controlada». En: *Tecnología y ciencias del agua* 5.2 (2014). Documento en línea, págs. 177-183. URL: http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S2007-24222014000200012&lng=es&tlang=es (visitado 20-03-2025).
- [3] Misiones Online. *Horticultura en Misiones*. URL: <https://misionesonline.net/2024/06/14/horticultura-en-misiones-2/> (visitado 20-03-2025).
- [4] Primera Edición. *Misiones: la hidroponía, cada vez más presente*. Primera Edición. URL: <https://www.primeraedicion.com.ar/nota/100627758/misiones-la-hidroponia-cada-vez-mas-presente/> (visitado 20-03-2025).
- [5] Lucas A. Garibaldi et al. «Seguridad alimentaria, medio ambiente y nuestros hábitos de Consumo». En: *Ecología Austral* 28.3 (2018), págs. 572-580, URL: https://www.scielo.org.ar/scielo.php?script=sci_arttext&pid=S1667-782X201800400011&lng=es&tlang=es (visitado 20-03-2025).
- [6] Hidropónia FIL. URL: <https://hidroponiafil.com.ar/> (visitado 20-03-2025).
- [7] Hidrosense. URL: <https://www.hidrosense.com.br/> (visitado 20-03-2025).
- [8] iPonia. URL: <https://iponia.com.br/> (visitado 20-03-2025).
- [9] Growcast. URL: <https://www.growcast.io/> (visitado 20-03-2025).
- [10] Shanna Li. «Comparative analysis of infrastructure and Ad-Hoc wireless networks». En: *ITM Web of Conferences*. Proceedings of the International Conference on Intelligent Computing, Communication & Information Technologies (ICICCI 2018) 25 (2019). Article number 01009, pág. 01009. ISSN: 2271-2097. DOI: <https://doi.org/10.1051/itmconf/20192501009>. URL: https://www.itm-conferences.org/articles/itmconf/pdf/2019/02/itmconf_icicci2018_01009.pdf.
- [11] OASIS Open. *Foundational IoT Messaging Protocol MQTT Becomes International OASIS Standard*. OASIS Open. URL: <https://www.oasis-open.org/2014/11/13/foundational-iot-messaging-protocol-mqtt-becomes-international-oasis-standard/> (visitado 25-03-2025).
- [12] Amazon Web Services. ¿Qué es MQTT? AWS. URL: <https://aws.amazon.com/es/what-is/mqtt/> (visitado 25-03-2025).
- [13] E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. Internet Requests for Comments. RFC. DOI: <10.17487/RFC8446>. URL: <https://datatracker.ietf.org/doc/html/rfc8446>.
- [14] Amazon Web Services. *Transport Security in AWS IoT Core*. URL: <https://docs.aws.amazon.com/iot/latest/developerguide/transport-security.html> (visitado 25-03-2025).

Capítulo 3

Diseño e implementación

En este capítulo se describe el diseño y la implementación del sistema de monitoreo y control de invernaderos. Se detallan los componentes principales del sistema, las decisiones de diseño tomadas, y los pasos seguidos para su implementación.

3.1. Arquitectura del sistema

La figura 3.1 ilustra la arquitectura general del sistema y la interacción entre los diferentes componentes.

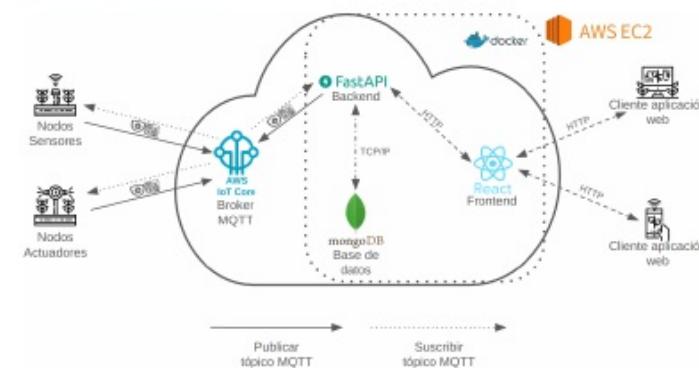


FIGURA 3.1. Arquitectura de la solución propuesta.

La arquitectura planteada para el desarrollo del trabajo sigue el modelo de tres capas típico de un sistema IoT: percepción, red y aplicación.

- Capa de percepción: formada por nodos sensores y actuadores que recopilan datos del entorno y ejecutan acciones de acuerdo con la configuración establecida.
- Capa de red: encargada de gestionar la comunicación entre los dispositivos IoT y el backend. Los sensores y actuadores transmiten datos a través de Wi-Fi, los cuales son gestionados por un broker MQTT.

- [15] IBM Corporation. *Protocolos TCP/IP*. International Business Machines (IBM). URL: <https://docs.aws.amazon.com/iot/latest/developerguide/transport-security.html> (visitado 25-03-2025).
- [16] Jeffrey Mogul et al. *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616. doi: 10.17487/rfc2616. URL: <https://rfc-editor.org/rfc/rfc2616.txt>.
- [17] Microsoft Corporation. *Interactuar con recursos HTTP/HTTPS de forma asíncrona*. URL: <https://learn.microsoft.com/es-es/power-apps/developer/model-driven-apps/best-practices/business-logic/interact-http-https-resources-asynchronously> (visitado 25-03-2025).
- [18] I. Fette y A. Melnikov. *The WebSocket Protocol*. Inf. t c. IETF. doi: 10.17487/RFC6455. URL: <https://tools.ietf.org/html/rfc6455>.
- [19] WebSocket Applications. URL: <https://www.ibm.com/docs/es/was/9.0.5?topic=applications-websocket> (visitado 25-03-2025).
- [20] Espressif Systems (Shanghai) Co., Ltd. *ESP32-WROOM-32 Datasheet*. URL: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf (visitado 26-03-2025).
- [21] Bosch Sensortec. *BME280 - Combined humidity, pressure and temperature sensor*. URL: <https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/> (visitado 26-03-2025).
- [22] ROHM Semiconductor. *BH1750 - Ambient Light Sensor (ALS) - Datasheet*. URL: https://www.mouser.com/catalog/specsheets/Rohm_11162017_ROHMS34826-1.pdf (visitado 26-03-2025).
- [23] Zhengzhou Winsen Electronics Technology Co., Ltd. *MH-Z19C NDIR CO₂ Sensor - Terminal Type Manual*. URL: https://www.mouser.com/datasheet/2/1398/Soldered_108994_co2_sensor_mh_z19-3532446.pdf (visitado 26-03-2025).
- [24] DFRobot. *Datos t cnicos del sensor de pH lquido PH-4502C*. URL: <https://image.dfrobot.com/image/data/SEN0161/PH%20composite%20electrode%20manual.pdf> (visitado 26-03-2025).
- [25] METTLER TOLEDO. *Sensores de Conductividad*. METTLER TOLEDO International Inc. URL: <https://www.mt.com/es/es/home/products/Process-Analytics/conductivity-resistivity-analyzers/conductivity-sensor.html> (visitado 26-03-2025).
- [26] EC Buying. *Datos t cnicos del sensor de CE*. URL: https://es.aliexpress.com/item/1005003479288815.html?spm=a2g0o.order_list.order_list_main.40.42e8194dYjUbr1&gatewayAdapt=glo2esp (visitado 26-03-2025).
- [27] Hach Company. *Solidos (totales y disueltos)*. URL: <https://es.hach.com/parameters/solids/> (visitado 26-03-2025).
- [28] DFRobot. *Datos t cnicos Sensor TDS*. URL: <https://www.dfrobot.com/product-1662.html?srsltd> (visitado 26-03-2025).
- [29] Dallas Semiconductor. *Datos t cnicos Sensor DS18B20*. URL: <https://cdn.sparkfun.com/datasheets/Sensors/Temp/DS18B20.pdf> (visitado 26-03-2025).
- [30] Sparkfun Electronics. *Datos t cnicos del sensor ultrasnico*. URL: <https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf> (visitado 26-03-2025).
- [31] Unit Electronics. *Datos t cnicos del sensor de energ a elctrica PZEM-004T*. URL: <https://uelectronics.com/wp-content/uploads/2024/06/AR4189-AR4190-Medidor-de-Energia-Electrica-AC-100A-Manual.pdf> (visitado 26-03-2025).
- [32] Songle Relay. *Datos t cnicos del Relay Songle*. URL: https://datasheet4u.com/pdf-down/S/R/D/SDR-12VDC-xx-x_ETC.pdf (visitado 26-03-2025).

- Capa de aplicaci n: plataforma en la nube responsable del procesamiento, almacenamiento y visualizaci n de datos. Facilita la interacci n con los dispositivos, la gesti n de la informaci n y la presentaci n de datos mediante una interfaz accesible para el usuario.

3.1.1. Capa de percepc n

La capa de percepc n est  constituida por los nodos sensores y actuadores, que se encargan de recopilar datos del entorno y ejecutar acciones espec icas en funci n de los par metros configurados.

Cada nodo sensor incluye un microcontrolador ESP-WROOM-32, el cual se conecta a diversos sensores que miden par metros como temperatura ambiente, humedad relativa, presi n atmosf rica, luminosidad, concentraci n de CO₂, pH, conductividad el ctrica, temperatura de la soluci n nutritiva, nivel de l quidos, consumo el ctrico, entre otros. Los nodos actuadores, por su parte, cuentan con rel s para controlar dispositivos como ventiladores, iluminaci n y sistemas de recirculaci n de nutrientes.

Los nodos est n conectados a una red Wi-Fi local, lo que les permite establecer comunicaci n con la red y transmitir los datos de los sensores hacia el servidor IoT. La transmisi n de datos se realiza con el protocolo MQTT.

3.1.2. Capa de red

La capa de red est  compuesta por la infraestructura que gestiona la comunicaci n entre los nodos sensores y actuadores y la plataforma de backend. Los nodos sensores y actuadores se conectan a la red Wi-Fi local, lo que les permite acceder a internet y a la infraestructura de la nube. Una vez conectados, los dispositivos transmiten los datos a trav s del protocolo MQTT.

La comunicaci n entre los nodos y el broker MQTT se asegura mediante el uso de certificados de seguridad, los cuales garantizan la autenticaci n de los dispositivos y el cifrado de los datos.

El broker MQTT utilizado en este trabajo es AWS IoT Core, un servicio completamente gestionado que permite establecer una conexi n segura y escalable entre los dispositivos IoT y la nube. Este broker act a como intermediario para la transmisi n de datos entre los nodos y la capa de aplicaci n.

3.1.3. Capa de aplicaci n

La capa de aplicaci n es responsable del procesamiento, almacenamiento y visualizaci n de los datos recopilados por los nodos. Para esta capa, se implement  el servidor IoT en la nube utilizando el servicio AWS EC2, que permite ejecutar aplicaciones y servicios en instancias virtuales.

El procesamiento y la gesti n de datos se realiza a trav s de un backend desarrollado con FastAPI, mientras que la base de datos MongoDB se utiliza para el almacenamiento de la informaci n. Adem s, se implement  una interfaz gr fica de usuario en React para la visualizaci n y gesti n de los datos. Todos estos servicios fueron desplegados a trav s de contenedores Docker.

- [33] Damien P. George y contributors. *MicroPython*. URL: <https://micropython.org/> (visitado 26-03-2025).
- [34] CTA Electronics. *MicroPython - Recursos y Guías*. URL: <https://www.ctaelectronics.com/es/micropython/> (visitado 26-03-2025).
- [35] Tiangolo. *FastAPI*. URL: <https://fastapi.tiangolo.com/> (visitado 26-03-2025).
- [36] MongoDB, Inc. *MongoDB Documentation*. URL: <https://www.mongodb.com/> (visitado 26-03-2025).
- [37] Meta (formerly Facebook) and contributors. *React: Biblioteca de JavaScript para interfaces de usuario*. URL: <https://es.react.dev/> (visitado 26-03-2025).
- [38] Docker, Inc. *Docker: Desarrollo acelerado de aplicaciones en contenedores*. URL: <https://www.docker.com/> (visitado 26-03-2025).
- [39] Amazon.com, Inc. *AWS IoT Core*. URL: <https://aws.amazon.com/es/iot-core/> (visitado 26-03-2025).
- [40] Amazon.com, Inc. *EC2, Nube de cómputo elástica de Amazon*.
- [41] Microsoft Corporation. *Visual Studio Code*. URL: <https://code.visualstudio.com/> (visitado 26-03-2025).
- [42] Postman, Inc. *Postman API Platform*. URL: <https://www.postman.com/> (visitado 26-03-2025).
- [43] Git. *Sistema de control de versiones Git*. URL: <https://git-scm.com/> (visitado 26-03-2025).
- [44] GitHub, Inc. *GitHub*. URL: <https://github.com/> (visitado 26-03-2025).

3.2. Modelo de datos

En esta sección se presenta el modelo de datos implementado en el sistema.

3.2.1. Pruebas iniciales de sensores

Para diseñar el modelo adecuado, se llevó a cabo una prueba inicial con los sensores y se registraron los datos generados por cada uno de ellos. La tabla 3.1 muestra los datos obtenidos de cada sensor, tal como lo devuelve la librería utilizada para su configuración y lectura.

TABLA 3.1. Principales sensores y librerías utilizadas

Componente	Datos
Sensor BME280	La librería utilizada [42] devuelve los valores de temperatura ambiente, humedad relativa y presión atmosférica como Float.
Sensor BH1750	La librería utilizada [43] devuelve el valor de lux como Float.
Sensor MH-Z19C	La librería utilizada [44] devuelve el valor de ppm de CO ₂ como Int.
Sensor PH-4502	La librería utilizada [45] devuelve el valor de TDS como Float.
Sensor de CE	La librería utilizada [46] devuelve el valor de TDS como Float.
Sensor de TDS	La librería utilizada [47] devuelve el valor de TDS como Float.
Sensor DS18B20	La librería utilizada [48] devuelve el valor de la temperatura como Float.
Sensor HC-SR04	La librería utilizada [49] devuelve el valor de distancia en centímetros como Int.
Sensor PZEM-004T	La librería utilizada [50] devuelve los valores de voltaje, corriente, potencia, cálculo de potencia y factor de potencia como Float.

3.2.2. Diseño del modelo de datos

El diseño del modelo de datos se desarrolló de acuerdo a los tipos de datos proporcionados por los sensores, así como los requerimientos técnicos establecidos para el sistema.

La estructura se organizó en colecciones dentro de MongoDB, donde cada colección representa un tipo de dato específico. Cada colección contiene documentos que almacenan las lecturas de los sensores y actuadores, usuarios, ambientes, tipos de ambientes, entre otros. Las colecciones se vinculan mediante identificadores únicos, lo que facilita la conexión entre las diferentes colecciones.

La figura 3.2 muestra el modelo de datos implementado en el sistema.