SISTEMA TELEMÁTICO PARA EL SEGUIMIENTO Y CONTROL DE LA HUMEDAD DEL SUELO EN INVERNADEROS

AUTORES

MICHAEL JESUS UBAQUE AVENDAÑO

Código 20221678028

CRISTIAN FELIPE MOLINA BROCHERO

Código 20221678002

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

FACULTAD TECNOLÓGICA

INGENIERÍA EN TELEMÁTICA

BOGOTÁ D.C

2026

SISTEMA TELEMÁTICO PARA EL SEGUIMIENTO Y CONTROL DE LA HUMEDAD DEL SUELO EN INVERNADEROS

AUTORES

MICHAEL JESUS UBAQUE AVENDAÑO

Código 20221678028

CRISTIAN FELIPE MOLINA BROCHERO

Código 20221678002

DIRECTOR

MIGUEL ÁNGEL LEGUIZAMÓN PÁEZ

MODALIDAD

MONOGRAFÍA

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

FACULTAD TECNOLÓGICA

INGENIERÍA EN TELEMÁTICA

BOGOTÁ D.C

2026

NOTAS DE ACEPTACIÓN

_____

_____

_____

_____

_____

_____

TUTOR

_____

JURADO

_____

JURADO

RESUMEN

Este trabajo de grado presenta el diseño, desarrollo e implementación de un sistema telemático para la monitorización remota de la humedad del suelo en cultivos dentro de invernaderos, con el propósito de proporcionar información en tiempo real que apoye a los agricultores en la toma de decisiones al momento de realizar riegos.

La solución incluye un prototipo funcional encargado de capturar las lecturas mediante un sensor de humedad del suelo, así como una aplicación web para la visualización de la información y la generación de alertas automáticas dirigidas a los usuarios mediante correo electrónico y mensajes SMS cuando se detectan valores de humedad por fuera de los parámetros óptimos configurados.

El sistema se fundamenta en una arquitectura orientada al Internet de las cosas (IoT), utilizando el protocolo MQTT como mecanismo principal de comunicación entre el prototipo y el servidor. Asimismo, se emplean WebSockets para permitir la actualización y representación de las lecturas en tiempo real en la interfaz web. El backend del sistema fue desarrollado utilizando Laravel, mientras que el frontend fue desarrollado con el framework Vue.js. La aplicación web permite visualizar las lecturas mediante gráficos, tablas y componentes interactivos, facilitando el seguimiento de las condiciones de humedad del suelo.

La solución fue desplegada en una infraestructura en la nube mediante la configuración de un VPS (Servidor Privado Virtual), uso de APIs externas y diferentes mecanismos de comunicación, lo que permitió disponer del sistema en un entorno de producción y realizar pruebas en un ambiente simulado y controlado.

Palabras clave: Monitorización remota, Internet de las cosas, sensor de humedad del suelo, MQTT, ESP32, Invernaderos, sistema telemático.

ABSTRACT

This thesis presents the design, development, and implementation of a telematic system for the remote monitoring of soil moisture in greenhouse crops, with the purpose of providing real-time information to support farmers in decision-making regarding irrigation.

The solution includes a functional prototype responsible for collecting measurements using a soil moisture sensor, as well as a web application for data visualization and the generation of automatic alerts sent to users via email and SMS when moisture values fall outside the configured optimal parameters.

The system is based on an Internet of Things (IoT) architecture, using the MQTT protocol as the primary communication mechanism between the prototype and the server. Additionally, WebSockets are used to enable the real-time updating and display of measurements in the web interface. The system backend was developed using Laravel, while the frontend was developed with the Vue framework. The web application allows measurements to be displayed through charts, tables, and interactive components, facilitating the monitoring of soil moisture conditions.

The solution was deployed on a cloud infrastructure through the configuration of a VPS (Virtual Private Server), the use of external APIs, and different communication mechanisms. This made it possible to operate the system in a production environment and conduct tests in a simulated and controlled environment.

Keywords: Remote monitoring, Internet of Things, soil moisture sensor, MQTT, ESP32, greenhouses, telematic system.

Tabla de contenido

1. Planeación 13
1.1. Planteamiento del problema 13
1.1.1. Descripción 13
1.1.2. Formulación 14
1.2. Justificación 14
1.3. Objetivos 15
1.3.1. General 15
1.3.2. Específicos 15
1.4. Alcance y delimitación 15
1.4.1. Alcance 15
1.4.2. Delimitación 16
1.5. Metodología 17
1.5.1. Scrum 17
1.5.2. Roles 19
1.5.3. Sprints 20
1.5.4. Product Backlog 20
1.6. Marco de referencia 25
1.6.1. Revisión Literaria 25
1.6.2. Marco teórico 27
1.6.3. Marco conceptual 31
1.7. Factibilidad 39
1.7.1. Factibilidad Técnica 39
1.7.2. Factibilidad Operativa 40
1.7.3. Factibilidad Económica 40
1.7.4. Factibilidad legal 43
1.8. Cronograma 44
2. Tecnologías y procesos para implementar sistemas de monitorización remota 45
2.1. Descripción de la solución 46
2.2. Arquitectura IoT 46
2.3. Protocolo de comunicación 47
2.3.1. Uso del Protocolo MQTT 47
2.3.2. Uso del protocolo HTTP 48
2.4. Framework de desarrollo 48
2.4.1. Laravel 48

2.4.2. Vue 48
2.5. Microcontrolador y Sensor de Humedad de Suelo 48
2.5.1. Microcontrolador 49
2.5.2. Sensor de humedad 49
3. Diseño de la arquitectura 50
3.1. Diagrama general del sistema 50
3.2. Diagrama de comunicación 53
3.3. Entorno de implementación 54
3.3.1. Servidor 54
3.3.2. Microcontrolador 55
3.3.3. Comunicación 55
3.3.4. Software 56
4. Desarrollo del prototipo para la medición de la humedad del suelo 58
4.1. Diseño 58
4.1.1. Esquema 58
4.1.2. Diagrama de flujo del prototipo 60
4.2. Desarrollo 61
4.3. Montaje e implementación 62
5. Desarrollo del sistema web 65
5.1. Análisis 66
5.1.1. Requerimientos 66
5.1.2. Actores del sistema 67
5.1.3. Casos de uso 68
5.2. Diseño 68
5.2.1. Base de datos 68
5.2.2. Backend 74
5.2.3. Frontend 77
5.3. Desarrollo 78
5.3.1. Gestión de usuarios y autenticación 79
5.3.2. Gestión de invernaderos y gestión de sensores 82
5.3.3. Captura y representación de datos 85
5.4. Implementación 91
5.4.1. Configuración del servidor 91
5.4.2. Instalación de dependencias 92
5.4.3. Despliegue 93
5.4.4. Configuración de procesos en segundo plano 93

6. Integración del sistema 95
6.1. Integración del prototipo con el broker MQTT 95
6.2. Integración del broker MQTT con el backend 96
6.3. Integración del sistema de notificaciones 96
6.4. Integración de la página web con el backend 97
7. Pruebas y validación 98
7.1. Prototipo 99
7.2. Backend 100
7.3. Frontend 101
7.4. Generación de alertas 102
7.5. Pruebas en ambiente simulado 103
7.5.1. Prueba con un nivel medio de humedad del suelo. 104
7.5.2. Prueba con un nivel alto de humedad del suelo y notificaciones. 106
7.5.3. Prueba con nivel bajo de humedad del suelo. 109
7.5.4. Prueba de variación progresiva de la humedad del suelo 111
8. Conclusiones 113
9. Recomendaciones y trabajos futuros 114
10. Referencias bibliográficas 115

Índice de figuras

Figura 1. Cronograma 44
Figura 2. Diagrama general del sistema. Fuente: Elaboración propia 51
Figura 3. Modelo TCP/IP. Fuente: (Interpolados, 2017) 52
Figura 4. Diagrama de comunicación del sistema. 53
Figura 5. Esquema de conexión de referencia. Fuente: (Cytron Technologies, 2019)..... 59
Figura 6. Diagrama de flujo del prototipo. Fuente: Elaboración propia 60
Figura 7. Montaje e implementación del prototipo. 63
Figura 8. Encendido del prototipo. Fuente: Elaboración propia 63
Figura 9. Salida monitor serial prototipo. Fuente: Elaboración propia 64
Figura 10. Diagrama de casos de uso. 68
Figura 11. Diagrama entidad-relación. Fuente: Elaboración propia 73
Figura 12. Diagrama general del backend. Fuente: Elaboración propia 74
Figura 13. Esquema Publicador/Subscriptor. 75
Figura 14. Arquitectura MVC de Laravel. Fuente: (Monocubed, 2025) 76
Figura 15. Esquema de utilización de colas. Fuente: Elaboración propia 77
Figura 16. Stack Laravel Vue. Fuente: (Thomson, P. J., 2020) 77
Figura 17. Módulos del frontend por roles. Fuente: Elaboración propia 78
Figura 18. Interfaz inicio de sesión del sistema a web. 80
Figura 19. Interfaz recuperación de contraseña. 80
Figura 20. Código manejo de rutas en Laravel. 81
Figura 21. Módulo gestión de usuarios. 81
Figura 22. Interfaz para la creación de usuarios. 82
Figura 23. Código modelo cultivo. Fuente: Elaboración propia 83
Figura 24. Interfaz listado de invernaderos. Fuente: Elaboración propia 83
Figura 25. Interfaz gestión de invernaderos. Fuente: Elaboración propia 84
Figura 26. Módulo gestión de sensores. Fuente: Elaboración propia 84
Figura 27. Interfaz de registro de sensores. Fuente: Elaboración propia 85
Figura 28. Comando Instalación Broker Mosquitto. 86
Figura 29. Código implementación del cliente MQTT en Laravel. 86
Figura 30. Código para la validación de lecturas y generación de alertas. Fuente: Elaboración propia 87
Figura 31. Código para el envío de los correos desde Laravel. 88
Figura 32. Panel rol administrador del sistema web. 89
Figura 33. Interfaz detalle de invernadero. 90
Figura 34. Interfaz detalle de sensor. 90
Figura 35. Interfaz de alertas generadas. 91
Figura 36. Contratación del servidor - plataforma OVHCloud. 92
Figura 37. Comando para despliegue de la aplicación Laravel. 93
Figura 38. Comando para optimización aplicación Laravel. 93
Figura 39. Procesos en ejecución en segundo plano. 94
Figura 40. Página principal del sitio web en producción. 94
Figura 41. Correos enviados visualizados en la plataforma Resend. Fuente: Elaboración propia 97
Figura 42. Macetas de prueba. Fuente: Elaboración propia 103
Figura 43. Macetas con cubierta plástica. Fuente: Elaboración propia 104
Figura 44. Maceta con nivel medio de humedad del suelo. 104

Figura 45. Visualización de información del sensor en el sitio web desarrollado. Fuente: Elaboración propia 105
Figura 46. Gráfica de lecturas con humedad media del suelo. 105
Figura 47. Histórico de lecturas obtenidas por el sensor. 106
Figura 48. Maceta con alto nivel de humedad del suelo. 107
Figura 49. Captura de valores de humedad del suelo alta. 107
Figura 50. Gráfica de lecturas con humedad alta del suelo. 108
Figura 51. Correo de notificación por humedad del suelo alta. 108
Figura 52. Mensaje de notificación SMS por humedad alta. 109
Figura 53. Maceta con bajo nivel de humedad del suelo. 110
Figura 54. Gráfica de lecturas con humedad baja del suelo. 110
Figura 55. Interfaz visualización de alertas generadas desde el sitio web. Fuente: Elaboración propia 111
Figura 56. Gráfica de lecturas tomadas durante 5 minutos. 111
Figura 57. Gráfica de lecturas tomadas durante una hora. 112

Índice de tablas

Tabla 1. Roles para el proyecto aplicando la metodología Scrum. 19
Tabla 2. Asignación de roles. Fuente: Elaboración propia 20
Tabla 3. Sprint del proyecto. Fuente: Elaboración propia 20
Tabla 4. Lista de actividades por sprint. Fuente: Elaboración propia 25
Tabla 5. Recursos de Hardware. Fuente: Elaboración propia 39
Tabla 6. Recursos de Software. Fuente: Elaboración propia 39
Tabla 7. Recursos Humanos. Fuente: Elaboración propia 40
Tabla 8. Lista de precios recursos de hardware. 41
Tabla 9. Costo recursos de software. Fuente: Elaboración Propia 41
Tabla 10. Precios recursos humanos. Fuente: Elaboración propia 42
Tabla 11. Detalle de horas de recursos humanos. 42
Tabla 12. Otros costos. Fuente: Elaboración propia 42
Tabla 13. Costo total del proyecto. Fuente: Elaboración propia 43
Tabla 14. Planificación de actividades del Sprint 1 45
Tabla 15. Planeación tareas para el diseño de la arquitectura. 50
Tabla 16. Requisitos mínimos del servidor. Fuente: Elaboración propia 54
Tabla 17. Requisitos del microcontrolador. Fuente: Elaboración propia 55
Tabla 18. Requisitos de conectividad. Fuente: Elaboración propia. 56
Tabla 19. requisitos de software. Fuente: Elaboración propia 57
Tabla 20. Planificación de actividades del Sprint 3. Fuente: Elaboración propia 58
Tabla 21. Conexión entre el ESP32 y el sensor de humedad del suelo YL-100. Fuente: Elaboración propia 59
Tabla 22. Parámetros de configuración implementados en el código del prototipo. Fuente: Elaboración propia 61
Tabla 23. Funciones implementadas en el código del prototipo. 62
Tabla 24. Planificación de actividades del Sprint 4 65
Tabla 25. Planificación de actividades del Sprint 5 65
Tabla 26. Requerimientos funcionales. 66
Tabla 27. Requerimientos no funcionales. 67
Tabla 28. Actores del sistema. 67
Tabla 29. Tabla usuarios de la base de datos. 69
Tabla 30. Tabla cultivos de la base de datos. Fuente: Elaboración propia 70
Tabla 31. Tabla tipos sensor de la base de datos. 70
Tabla 32. Tabla sensores de la base de datos. 71
Tabla 33. Tabla lecturas. Fuente: Elaboración propia 71
Tabla 34. Tabla de alertas generadas de la base de datos. 72
Tabla 35. Entorno de desarrollo. Fuente: Elaboración propia 79
Tabla 36. Parámetros de configuración del ESP32. 95
Tabla 37. Parámetros de configuración del cliente MQTT en Laravel. Fuente: Elaboración propia 96
Tabla 38. Planificación de actividades del Sprint 3 98
Tabla 39. Pruebas del prototipo. Fuente: Elaboración propia 99
Tabla 40. Pruebas del backend. Fuente: Elaboración propia 100
Tabla 41. Pruebas del frontend. Fuente: Elaboración propia 102
Tabla 42. Pruebas sistema de alertas y notificaciones. 102

Introducción

La agricultura constituye una actividad fundamental para el desarrollo de la sociedad, debido a que permite garantizar la producción de alimentos necesarios para el consumo diario y contribuye significativamente al desarrollo económico y social. En la actualidad, los avances tecnológicos han permitido transformar diferentes procesos relacionados con la producción agrícola, buscando mejorar el aprovechamiento de los recursos, optimizar las condiciones de los cultivos y aumentar la eficiencia de las actividades realizadas durante las diferentes etapas de producción.

En este contexto, la implementación de tecnologías de monitoreo representa una alternativa para obtener información precisa sobre las condiciones en las que se desarrollan los cultivos. Entre las variables que requieren mayor atención se encuentra la humedad del suelo, debido a que el agua es un recurso indispensable para el crecimiento y desarrollo adecuado de las plantas. Un nivel de humedad insuficiente puede afectar el desarrollo del cultivo, mientras que un exceso de agua puede generar problemas como el deterioro de las raíces, la aparición de enfermedades y el desperdicio del recurso hídrico. Por esta razón, conocer y controlar de manera oportuna los niveles de humedad presentes en el suelo puede contribuir a mejorar las condiciones de crecimiento de los cultivos y favorecer un uso más eficiente del agua.

A partir de esta necesidad, surge la posibilidad de incorporar herramientas tecnológicas que permitan realizar un seguimiento constante de las condiciones del suelo. Los sistemas de monitoreo basados en sensores pueden facilitar la recopilación de datos en tiempo real, permitiendo identificar variaciones en los niveles de humedad y disponer de información que sirva como apoyo para la toma de decisiones relacionadas con el riego.

Por lo anterior, se presenta el desarrollo e implementación de un sistema de monitoreo de la humedad del suelo para cultivos, orientado a medir y registrar los niveles de humedad presentes en el terreno mediante el uso de sensores y herramientas tecnológicas. La implementación de este sistema busca proporcionar información que permita conocer de manera más precisa las condiciones de humedad del suelo y, de esta manera, contribuir a un manejo más eficiente del recurso hídrico y al adecuado desarrollo de los cultivos.

1. Planeación

1.1. Planteamiento del problema

1.1.1. Descripción

La agricultura colombiana, caracterizada por su diversidad de cultivos y climas, enfrenta desafíos particulares en la gestión del riego, lo que se ve reflejado en la ineficiencia de los sistemas de riego tradicionales que no logran adaptarse a las variaciones climáticas y geográficas. En regiones de clima tropical, como la Amazonía y la Orinoquía, la abundancia de lluvias en ciertos períodos y la escasez en otros generan grandes fluctuaciones en la disponibilidad de agua, dificultando el uso adecuado de este recurso. A pesar de los avances tecnológicos en otros sectores, muchas regiones del país aún dependen de sistemas de riego tradicionales, los cuales presentan limitaciones significativas para garantizar la eficiencia y sostenibilidad de la producción agrícola (Ministerio de Agricultura y Desarrollo Rural, 2020).

La agricultura nacional, especialmente aquella desarrollada en regiones con condiciones climáticas adversas o en cultivos que requieren un control preciso de las variables ambientales, ha recurrido a la implementación de invernaderos, debido a su capacidad para ofrecer un ambiente controlado que mitiga los efectos de las variaciones climáticas. Los invernaderos permiten a los agricultores proteger sus cultivos de factores externos como las heladas, las lluvias intensas o la falta de agua, creando condiciones óptimas de temperatura, humedad y luz para maximizar la producción. Además, al posibilitar el uso eficiente del agua mediante sistemas de riego más precisos y menos dependientes de las fluctuaciones externas, los invernaderos representan una solución estratégica frente a la escasez de agua y la creciente presión por la sostenibilidad en la agricultura.

Uno de los mayores retos en los invernaderos se centra en la capacidad de monitorear y controlar las variables ambientales, como la temperatura, la humedad, la concentración de CO2 y la calidad del aire, que son cruciales para el crecimiento óptimo de los cultivos. Las fluctuaciones inesperadas en estas variables pueden afectar negativamente la salud de las plantas, generando un entorno menos óptimo y reduciendo su productividad. Según un estudio realizado por Rincón Vieda, Silva Plazas y Torres Camacho (2016) en Colombia, y en particular en algunos departamentos como el Huila, los invernaderos no cuentan con las capacidades tecnológicas necesarias para el desarrollo óptimo de cualquier tipo de cultivo. En este estudio se halló que son los agricultores quienes, mediante métodos empíricos, miden y controlan las variables ambientales, incluida la humedad del terreno, lo que puede derivar en controles poco precisos.

Diversos estudios, como el realizado por Wilson Martínez (2022), han demostrado que la humedad del suelo es un factor crítico para el desarrollo de las plantas. Un exceso o defecto de agua puede generar estrés hídrico, afectando el crecimiento, la producción y la calidad de los cultivos. Los métodos tradicionales de riego, basados en horarios fijos o en la experiencia del productor, no garantizan un suministro óptimo de agua, lo que puede llevar a pérdidas económicas significativas. Esta dependencia de métodos imprecisos resalta la creciente necesidad de adoptar enfoques más acertados y adaptados a las condiciones específicas de cada cultivo, que permitan monitorear y ajustar las condiciones

del suelo en tiempo real, asegurando una gestión eficiente de los recursos hídricos y mejorando los rendimientos agrícolas en los invernaderos.

1.1.2. Formulación

¿De qué manera apoyar la medición de la humedad del suelo en invernaderos, para ayudar en la toma de decisiones al momento de realizar el riego?

1.2. Justificación

La gestión eficiente del recurso hídrico en la agricultura representa un desafío crítico en un contexto de creciente variabilidad climática y demanda de alimentos. La optimización del riego se ha vuelto fundamental, dado que un manejo inadecuado del agua no solo disminuye los rendimientos agrícolas, sino que también puede deteriorar las propiedades del suelo. La degradación de los suelos y la pérdida de proteínas en el suelo son consecuencias directas de prácticas de riego ineficientes, lo que subraya la necesidad de adoptar métodos más sostenibles que garanticen la salud del ecosistema agrícola.

Además de la imprecisión en las prácticas de riego, la problemática del manejo del agua, especialmente en los invernaderos, se ve agravada por factores como la escasez de agua y la competencia por este recurso entre diferentes sectores. Las sequías prolongadas y los cambios climáticos han reducido los recursos hídricos disponibles, lo que obliga a los productores a enfrentar una presión creciente para utilizar el agua de manera más eficiente. En este contexto, la falta de sistemas avanzados para detectar y monitorizar las condiciones del ambiente y del suelo en invernaderos, limita la capacidad de los agricultores para ajustar el riego de manera precisa y oportuna. A pesar de la tecnología disponible, muchas regiones del país aún carecen de una infraestructura adecuada que permita integrar estos sistemas de monitoreo en invernaderos, lo que dificulta la adaptación a las condiciones cambiantes de disponibilidad de agua y del clima.

Ante esta problemática, se plantea el desarrollo de un sistema de monitoreo en tiempo real basado en tecnología IoT (Internet de las cosas), que permita la medición continua de la humedad del suelo dentro de los invernaderos. Mediante el uso de sensores conectados a un microcontrolador y una plataforma web para la visualización y el análisis de datos recopilados, el sistema propuesto busca apoyar la toma de decisiones relacionadas con el riego, contribuyendo al uso eficiente del recurso hídrico y a la mejora de la productividad de los cultivos.

1.3. Objetivos

1.3.1. General

Desarrollar un sistema telemático para hacer seguimiento de la humedad del suelo en tiempo real dentro de invernaderos, con el fin de apoyar la toma de decisiones y aportar en la optimización del riego en cultivos.

1.3.2. Específicos

- Realizar una búsqueda de las tecnologías y procesos disponibles para implementar sistemas de monitorización remota que puedan ser implementados en el sistema propuesto.

- Diseñar una arquitectura de comunicación que permita el envío de información desde los sensores hacia un sistema web y el envío de alertas mediante correo electrónico y mensajes SMS.

- Desarrollar un prototipo capaz de medir la humedad del suelo en cultivos dentro de invernaderos y enviar los datos recopilados a un servicio web a través de internet.

- Diseñar e implementar un sistema web para recibir, almacenar y procesar los datos de la humedad del suelo enviados por el o los prototipos.

- Realizar pruebas del sistema en conjunto para verificar su correcto funcionamiento, en una situación simulada.

1.4. Alcance y delimitación

Para apoyar la medición de la humedad del suelo en invernaderos, se plantea desarrollar una solución tecnológica basada en la recolección de datos en tiempo real. Esto permitirá proporcionar información precisa que ayude a la toma de decisiones, reduciendo la dependencia de métodos empíricos y poco precisos. Para lograrlo, se propone el desarrollo de un sistema de telemetría basado en sensores de humedad del suelo, tecnología IoT y una plataforma web para la gestión y visualización de datos.

1.4.1. Alcance

El desarrollo de este proyecto contempla la implementación de un sistema de monitorización de la humedad del suelo mediante la elaboración de como mínimo un prototipo con montaje físico para capturar los datos y un sistema web alojado en la nube para el almacenamiento y procesamiento de los datos.

El sistema web (backend y frontend) serán alojados en un servidor real en la nube con la configuración requerida para su correcto funcionamiento, además, se permitirá la generación de alertas mediante el envío de correo electrónico o mensajes SMS haciendo uso de APIs (Interfaz de programación de aplicaciones) externas. Los servicios y la plataforma web estarán disponibles únicamente durante las fases de desarrollo e

implementación del presente proyecto, hasta su presentación final, esto debido a los gastos concurrentes que trae mantener un sistema web en un ambiente de producción.

El prototipo contará con un microcontrolador con capacidad de conexión a internet mediante Wi-Fi, quien realizará el envío de los datos recopilados por un sensor de humedad del suelo al servidor en la nube.

1.4.2. Delimitación

El proyecto se limita al desarrollo e implementación de un sistema para el monitoreo de la humedad del suelo, sin tener en cuenta la monitorización de otras variables ambientales y sin contemplar la implementación de actuadores que modifiquen el flujo del riego. Así mismo, las pruebas hechas con el prototipo serán realizadas en un ambiente controlado y simulado de un invernadero sin llegar a realizarlas en un cultivo real.

A continuación, se detallan las principales limitaciones y restricciones que presenta el proyecto:

- **Tiempo:**
  El proyecto debe ser culminado y entregado en los plazos establecidos en el cronograma, en un tiempo no mayor a 6 meses desde su inicio.

- **Presupuesto:**
  El proyecto debe ajustarse al presupuesto y a los límites establecidos, sin exceder los topes definidos para su ejecución.

- **Restricciones en conectividad:**
  El funcionamiento integral del sistema depende de conexiones a internet de manera permanente. Los sensores deben estar conectados a una red Wi-Fi estable, sin embargo, en varias zonas del país, especialmente en áreas rurales donde se desarrollan actividades agrícolas y se ubican mayoritariamente los invernaderos, se dispone de una cobertura baja en conexión a internet.[1]

- **Capacidad de cómputo y almacenamiento:**
  La capacidad del servidor que soporta el sistema depende estrictamente del presupuesto destinado para su contratación, por lo que debe ser ajustado al presupuesto asignado. Una mayor capacidad de cómputo y almacenamiento depende de un mayor presupuesto para la contratación de un servidor con mayores capacidades.

[1] *Impacto TIC*. “Internet rural en Colombia: precios, empresas y cobertura en el país” (07 oct 2025). Disponible en: https://impactotic.co/innovacion/transformacion-digital/internet-en-zonas-rurales-en-colombia

1.5. Metodología

Para la elaboración de este proyecto, se utilizará la metodología Scrum, un enfoque ágil que permite desarrollar proyectos de manera rápida, adaptándose a entornos cambiantes y enfocándose en trabajar de forma iterativa e incremental, estructurando el trabajo en ciclos llamados sprints (Deemer, 2009). Esta metodología nos permitirá organizar el trabajo en fases llamadas sprints, lo que facilitará cumplir con los diferentes objetivos y requerimientos del proyecto mediante entregas parciales.

1.5.1. Scrum

Scrum es un marco que se usa para desarrollar proyectos, productos y aplicaciones. Se divide el trabajo en ciclos que se llaman Sprints y cada Sprint dura entre una y cuatro semanas, cada uno de ellos se suceden uno tras otro sin interrupción. Al iniciar cada uno el equipo elige los elementos más importantes de la lista de requisitos. Luego se compromete a entregar esos elementos al terminar el Sprint. Durante su ejecución el equipo trabaja en los elementos elegidos y los usuarios no pueden modificar los elementos seleccionados. La restricción garantiza estabilidad y enfoque en el trabajo (Deemer et al., 2009).

Uno de los principios centrales de Scrum es el de 'inspeccionar y adaptar': el equipo revisa continuamente tanto el producto construido como la eficacia de sus propias prácticas, y ajusta su rumbo en función de lo aprendido. Este principio convierte a Scrum en un mecanismo de mejora continua que visibiliza impedimentos y disfunciones, permitiendo que el equipo los aborde de manera sistemática y oportuna (Deemer et al., 2009).

- **Roles en Scrum**

Scrum define tres roles principales. El Dueño de Producto (DP) es el responsable de maximizar el retorno de inversión del producto, identificando las funcionalidades requeridas y priorizando las continuamente en la Pila de Producto. Interactúa de forma activa y frecuente con el equipo, revisando resultados al final de cada iteración. El Equipo, compuesto por entre cinco y nueve personas con perfiles multifuncionales, es quien construye el producto. Se autogestiona con un alto grado de autonomía, decide a qué trabajo se compromete en cada Sprint y determina la mejor manera de cumplir con ese compromiso (Deemer et al., 2009).

El Scrum Master, por su parte, es el facilitador del proceso. No actúa como jefe de equipo ni de proyecto, sino que sirve al equipo protegiéndolo de interferencias externas, guiando al DP y al equipo en la correcta aplicación de Scrum, y ayudando a eliminar los impedimentos que obstaculizan el avance. Es relevante destacar que en Scrum no existe el rol de jefe de proyecto en el sentido tradicional; su función se distribuye entre el DP, el equipo y el Scrum Master (Deemer et al., 2009).

- **Qué es un Sprint**

Scrum utiliza tres artefactos principales para gestionar y visualizar el trabajo. La Pila de Producto es una lista priorizada de todas las funcionalidades, mejoras, investigaciones y correcciones que podrían realizarse en el producto. Es mantenida y actualizada continuamente por el Dueño de Producto para reflejar los cambios en las necesidades del cliente, el mercado y el contexto técnico. La Pila del Sprint es el subconjunto de elementos de la Pila de Producto que el equipo se compromete a completar durante el Sprint en curso, desglosados en tareas con estimaciones de esfuerzo (Deemer et al., 2009).

Adicionalmente, la Gráfica de Trabajo Restante del Sprint es una herramienta visual que muestra diariamente la cantidad de trabajo pendiente en el Sprint, permitiendo al equipo evaluar su progreso y tomar decisiones oportunas si la tendencia indica que no se alcanzará el objetivo comprometido. A nivel de entrega, existe también la Gráfica de Trabajo Restante de la Entrega, que proporciona una visión de más largo plazo sobre el avance hacia la fecha de lanzamiento del producto (Deemer et al., 2009).

- **Eventos de Scrum**

El marco Scrum estructura el trabajo a través de eventos formales y recurrentes. La Planificación del Sprint, celebrada al inicio de cada ciclo, se divide en dos partes:
La primera orientada a comprender qué se construirá, y la segunda a definir cómo se llevará a cabo. El Scrum Diario es una reunión breve de 15 minutos en la que cada miembro del equipo informa sobre lo que hizo, lo que planea hacer y cualquier impedimento que enfrenta, promoviendo la coordinación y la autogestión del equipo (Deemer et al., 2009).

Al finalizar el Sprint, se realizan dos eventos complementarios. La Revisión del Sprint es un espacio de inspección y adaptación del producto, en el que el equipo presenta lo construido al DP y demás interesados, generando retroalimentación que orienta el siguiente ciclo. La Retrospectiva del Sprint, por su parte, se enfoca en inspeccionar y adaptar el proceso: el equipo reflexiona sobre lo que funcionó bien y lo que puede mejorarse, acordando cambios concretos para el siguiente Sprint. Estos dos eventos constituyen el mecanismo principal de aprendizaje organizacional en Scrum (Deemer et al., 2009).

- **Beneficios y Resultados del Uso de Scrum**

La implementación de Scrum ha demostrado mejoras sustanciales en múltiples dimensiones organizacionales. Con base en encuestas realizadas a equipos de Yahoo, empresa que migró aproximadamente 200 equipos a Scrum con un total de 2.000 personas, el 68% de los encuestados consideró que Scrum mejoró significativamente la productividad, el 81% reportó mejoras en la colaboración y cooperación, y el 63% destacó una mayor adaptabilidad ante los cambios. Asimismo, la productividad de los equipos mejoró en promedio un 36% según la percepción de los dueños de producto, y el 85% de los miembros de equipos manifestó que continuaría utilizando Scrum si la decisión dependiera de ellos (Deemer et al., 2009).

Estos resultados evidencian que Scrum no solo mejora la eficiencia operativa, sino que también incide positivamente en la moral del equipo, la responsabilidad individual y la

capacidad de respuesta ante entornos cambiantes. Su fortaleza radica en hacer visibles tanto los logros como las disfunciones del equipo, proporcionando un marco estructurado para la mejora continua y la toma de decisiones basada en evidencia empírica (Deemer et al., 2009).

- **Backlog**

Backlog hace referencia a una lista de actividades organizada y priorizada con unos requerimientos y funcionalidades pendientes que abarcan un proyecto. En el contexto de la metodología scrum, el backlog trabaja como un repositorio dinámico para llevar a cabo las necesidades del producto y darle prioridad en base a los objetivos del negocio. Hay dos tipos de backlog:

- **Product backlog**: Agrupa todos los requerimientos generales del producto
- **Sprint backlog:** Contiene las tareas seleccionadas para desarrollar durante un sprint en específico.

Backlog es un elemento esencial dentro de esta metodología, pues facilita en la organización y gestión de los recursos, con sus respectivos tiempos de ejecución en el proyecto lo que permite tener una visión más clara sobre las actividades pendientes y en general sobre el avance del desarrollo.

1.5.2. Roles

Gracias a la estructura del equipo de trabajo que presenta la metodología Scrum, se presenta la asignación de roles y tareas específicas a cada miembro del equipo:

Rol	Descripción
Scrum Master	● Encargado de asegurar que el marco de trabajo sea ejecutado correctamente. ● Identificar oportunidades de mejora y/o correcciones para darle mayor calidad a los resultados del proyecto. ● Revisar los avances y asegurar que las actividades se ejecuten en los tiempos establecidos
Product Owner	● Encargado de gestionar y promocionar el Product Backlog, organizando y priorizando las diferentes actividades. ● Valida que las entregas cumplan con las necesidades de los usuarios finales. ● Representa e interactúa con las partes interesadas en el proyecto.
Scrum Team	● Encargados de desarrollar y culminar las diferentes actividades asignadas para cumplir con las entregas parciales en cada sprint. ● Asegurar la mayor calidad en el desarrollo del producto final. ● Revisar los avances diariamente, adaptándose a entornos cambiantes y manteniendo un enfoque hacia el cumplimiento de los objetivos.

Tabla 1. Roles para el proyecto aplicando la metodología Scrum.
Fuente: Elaboración propia.

A continuación, se definen los roles de los miembros del equipo de acuerdo con las capacidades y responsabilidades de cada uno.

Rol	Asignado a
Scrum Master	Michael Jesus Ubaque
Product Owner	Cristian Felipe Molina
Scrum Team	Michael Jesus Ubaque Cristian Felipe Molina

Tabla 2. Asignación de roles. Fuente: Elaboración propia

1.5.3. Sprints

Teniendo en cuenta la metodología, los sprint son fases o ciclos iterativos e incrementales de duración fija, que terminan con una entrega parcial y funcional del proyecto, para este proyecto se plantean las siguientes fases:

Sprint	Descripción	Duración
S1	Análisis	3 semanas
S2	Diseño	4 semanas
S3	Desarrollo del prototipo para la medición de la humedad del suelo	3 semanas
S4	Desarrollo backend	4 semanas
S5	Desarrollo frontend	4 semanas
S6	Pruebas e implementación	3 semanas

Tabla 3. Sprint del proyecto. Fuente: Elaboración propia

1.5.4. Product Backlog

A continuación, se presenta la lista de actividades necesarias para completar cada Sprint y generar como resultados entregables incrementales:

No.	Sprint	Actividad	Estimación en tiempo	Descripción
1	S1	Identificación y definición del problema	7 días	Es fundamental en cualquier proyecto contar con un punto de partida claro que permita identificar y comprender la problemática que se busca resolver a lo largo del desarrollo del proyecto. Este análisis inicial establece las bases para diseñar soluciones efectivas y orientadas a abordar los desafíos específicos de manera eficiente.
2	S1	Contextualización de la problemática (Investigación)	6 días	Otro factor clave para entender y proporcionar una solución adecuada al planteamiento establecido es analizar los escenarios en los que se desarrolla la problemática. Esta comprensión permite generar propuestas más precisas y efectivas, que aborden de manera óptima los desafíos identificados.
3	S1	Identificación de requerimientos	7 días	Los requerimientos son los elementos esenciales que permiten comprender plenamente la problemática y asegurar que la solución propuesta sea adecuada y oportuna, basada en el planteamiento inicial. Estos aspectos son cruciales para guiar el desarrollo de la solución de manera eficiente y efectiva.
4	S2	Arquitectura y comunicación	3 días	Es crucial detallar la solución tecnológica, describiendo su composición e interacciones, para poder comprender cómo funcionará el sistema en su conjunto.
5	S2	Base de datos	3 días	Toda la información procesada en el sistema es controlada por la base de datos, que se encarga de almacenar, gestionar y facilitar su consulta de manera eficiente, siguiendo una estructura definida que permite su acceso y manipulación adecuada según las necesidades del sistema.

6	S2	Módulos	5 días	Es crucial clasificar las funcionalidades del sistema en módulos específicos, cada uno encargado de una tarea concreta. Esto incluye un módulo para la captura de datos tomados en el cultivo del invernadero por medio del circuito integrado, encargado de recolectar la información del entorno. Un módulo de bases de datos, responsable de almacenar y gestionar la información de manera estructurada y eficiente. Y un módulo de interfaz de la aplicación, que permitirá visualizar los datos procesados y ofrecer una interacción intuitiva con el usuario. Esta organización modular asegura un flujo de información claro y un funcionamiento eficiente del sistema.
7	S2	Interfaces	5 días	Las interfaces son la capa visual del sistema, encargada de mostrar la información de manera clara, exacta y precisa cada vez que se realice una consulta, asegurando una experiencia de usuario eficiente y comprensible.
8	S2	Prototipo	7 días	Este prototipo permite evaluar cómo interactúan los diferentes elementos del sistema, garantizando su ejecución correcta y su capacidad para cumplir con los objetivos del proyecto en condiciones reales de operación.

9	S3	Requisitos	2 días	Los requisitos necesarios para asegurar el correcto funcionamiento del sistema incluyen la integración adecuada de todos sus componentes, la precisión en la captura y procesamiento de los datos, y la fiabilidad en la comunicación entre el circuito integrado y el aplicativo. Además, es fundamental contar con una infraestructura de almacenamiento de datos eficiente, una interfaz de usuario intuitiva, y un sistema de control que permita realizar ajustes en tiempo real. Cumplir con estos requisitos garantiza que el sistema opere de manera eficiente y que los datos obtenidos sean precisos y útiles para la toma de decisiones.
10	S3	Definición de funcionalidades	2 días	Se detallan las distintas partes que componen el sistema, describiendo la funcionalidad específica que cada una aporta para asegurar el correcto funcionamiento del conjunto. Cada componente, desde los sensores de captura de datos hasta la base de datos y la interfaz de usuario, juega un papel crucial en la recopilación, procesamiento y visualización de la información, permitiendo una operación eficiente y coherente del sistema.
11	S3	Montaje y configuración	14 días	Se lleva a cabo el ensamblaje de todos los componentes del prototipo, incluyendo el Arduino, la base de datos y el servidor encargado de almacenar la información. Además, se realiza la configuración detallada de cada parte para asegurar su correcta integración y funcionamiento, garantizando que todos los elementos del sistema trabajen de manera conjunta y eficiente.

12	S3	Pruebas	7 días	Se realizan pruebas exhaustivas para evaluar la funcionalidad del sistema, con el objetivo de identificar posibles fallos o carencias. A partir de los resultados obtenidos, se proporciona retroalimentación que permite ajustar y mejorar el sistema, garantizando su desempeño óptimo.
13	S4	Implementación de base de datos	3 días	Instalación y configuración de la base de datos según la estructura diseñada. Configuración de usuarios y credenciales de acceso.
14	S4	Creación de servicios web	15 días	Desarrollo de los servicios y endpoints de acuerdo al diseño de la arquitectura de comunicación, teniendo en cuenta el flujo de la información y la gestión de la seguridad.
15	S4	Pruebas	7 días	Verificar el correcto funcionamiento de los servicios.
16	S5	Desarrollo Interfaz web	17 días	Se desarrollarán las interfaces gráficas con las que los usuarios finales podrán interactuar, implementando los módulos definidos en el diseño.
17	S5	Representación gráfica de datos	3 días	Se representarán los datos capturados por los sensores en tablas y gráficos para facilitar su análisis.
18	S5	Configuración de hosting y servidor	3 días	Se preparará el entorno donde se alojará y ejecutará el sistema web en un servicio de hosting o VPS (servidor virtual privado)
19	S5	Pruebas	5 días	Se comprobará la funcionalidad y rendimiento de la interfaz web para garantizar una experiencia óptima para los usuarios.
20	S6	Integración del sistema	5 días	Se conectarán los diferentes componentes del sistema (back-end, front-end, base de datos, prototipo y servicios externos) para que funcionen de manera coordinada.

21	S6	Pruebas de funcionalidad	5 días	Se verificará el correcto funcionamiento de todo el sistema en conjunto, asegurando que se cumple con la totalidad de los objetivos y los requerimientos.
22	S6	Documentación	10 días	Se recopila la información sobre cómo se desarrolló e implementó el sistema.

Tabla 4. Lista de actividades por sprint. Fuente: Elaboración propia

1.6. Marco de referencia

1.6.1. Revisión Literaria

Partiendo del enfoque hacia la promoción del conocimiento científico y generación de soluciones tecnológicas que posee la Universidad Distrital Francisco José De Caldas, se realizó una investigación en sus repositorios con el fin de identificar proyectos de grados hechos por estudiantes, sobre sistemas de telemetría y monitorización remota, que puedan aportar bases técnicas y teóricas para la realización del presente proyecto.

En este sentido se encontró el proyecto de grado titulado “Implementación, Instrumentación, Monitoreo y Control de un Invernadero a Escala”, elaborado en año 2023 por los estudiantes Andrés Ariza y David Corredor, como modalidad de grado para para obtener su titulación como Ingenieros en Electrónica. En este proyecto se diseñó e implementó un invernadero semi automatizado. Para este fin, utilizaron diversos sensores y dispositivos para alterar o modificar las variables climáticas dentro del invernadero, como microcontrolador se utilizó un módulo ESP32 ya que integra tecnología WIFI. El reto de la comunicación y envío de información de manera remota, fue solucionado haciendo uso del protocolo HTTP y websockets, mediante peticiones a un servidor backend elaborado en Spring Boot junto con una base de datos PostgreSQL. Finalmente, de este proyecto destacamos los módulos, sensores y protocolos usados, que pueden adaptarse a las necesidades de nuestra propuesta.

Por otra parte, también se encontraron proyectos de grado en otras universidades de Colombia que abordan y dan solución a la problemática de falta de sistemas de monitorización de variables ambientales en cultivos, uno de ellos es un proyecto de grado titulado “Diseño e implementación de un sistema para el monitoreo de variables agroclimáticas (temperatura, humedad, brillo solar) y control remoto de riego con módulo esp8266 en cultivos cítricos en la región caribe”, este proyecto elaborado por los estudiantes Santiago López y Jeisson Rojas de la universidad Antonio Nariño, tiene como finalidad desarrollar un sistema de control de riego remoto y monitoreo de variables agroindustriales para la transmisión de datos usando IoT. En este proyecto se utilizaron sensores de humedad de suelo anticorrosivo YL100, Tarjetas programables Arduino Uno y módulos Esp-01s (para dotar de acceso a internet a las tarjetas programables), como elementos principales para la elaboración del prototipo que será instalado en los cultivos. En este proyecto se obtuvieron resultados satisfactorios en la medición de datos y el envío

de estos mediante internet, usando el protocolo IEEE802.11, además se resalta la conveniencia de usar Internet de las cosas, dicha tecnología puede servir de base para la arquitectura de nuestro proyecto.

Otro proyecto relacionado con el control y monitorización remota, fue elaborado como proyecto de grado en la pontificia Universidad Javeriana por el estudiante César Andrés Gaviria Cuevas, este proyecto denominado “Sistema IoT aplicado en la gestión de riego en cultivos de papa mediante censado y control remoto”, tiene como objetivo el diseño e implementación de un sistema IoT de censado y control remoto en cultivos. En este proyecto, aparte de implementar un prototipo con sensores de humedad y módulos ESP8266 para realizar conexiones a internet por medio de una WLAN, se implementó un servicio en cloud en Amazon Azure, que incluía un backend, un frontend y un sistema de actuación para realizar la recepción, el procesamiento y el envío de los datos hacia y desde los nodos sensores, Algunos de los servicios utilizados son Azure IoT Hub y Azure Events Hub. Este proyecto puede brindar bases para la implementación del servicio web en nube planteado en los objetivos específicos. Como conclusiones se obtuvo que fue posible implementar el sistema de monitorización remoto usando tarjetas Raspberry Pi y servicios en la nube basados en Amazon Azure.

Un artículo publicado en la revista Ingeniería Investiga en el año 2022, titulado “Sistema IoT basado en ESP32 para el control y monitoreo de cultivos en invernadero con enfoque de Agricultura 4.0” y elaborado por Sebastián Berrios y Hugo Javier, también presenta un diseño de un sistema IoT orientado al monitoreo y control de variables agrícolas dentro de un invernadero, empleando sensores de temperatura, humedad relativa, humedad del suelo y nivel de agua conectados a nodos basados en el microcontrolador ESP32. Para la comunicación inalámbrica entre los sensores y el nodo central se implementó el protocolo ESPNOW, mientras que el almacenamiento de datos y el acceso remoto se gestionaron mediante Firebase Realtime Database y una aplicación móvil desarrollada para Android. Los autores detallan el proceso de diseño, la configuración de los sensores y actuadores, así como las pruebas realizadas en un cultivo de lechugas, donde se comprobó el correcto funcionamiento del sistema para mantener los parámetros ambientales dentro de los rangos óptimos definidos para el tipo de cultivo. Debido a su flexibilidad y bajo costo, la solución tecnológica basada en microcontroladores ESP32 representa una alternativa viable para ser adaptada en el proyecto en desarrollo, debido a las múltiples formas de comunicación que integra. Además, usar los servicios de firebase para el almacenamiento ayuda a una implementación más rápida y sencilla.

En la revista Vector, de la Universidad de Caldas, se publicó el artículo “Monitoreo y control de variables ambientales mediante una red inalámbrica para agricultura de precisión en invernaderos”, elaborado por Ossa Duque en 2017. Este trabajo presenta el desarrollo de un sistema de bajo costo basado en nodos Arduino equipados con sensores de temperatura, humedad relativa y humedad del suelo, diseñado para recopilar información del ambiente en tiempo real dentro de un invernadero. Los nodos se comunican mediante una red inalámbrica que permite transmitir los datos hacia un módulo central encargado de procesarlos y activar actuadores cuando las variables superan los rangos establecidos. El estudio describe la metodología de implementación, la arquitectura de red, la calibración de los sensores y las pruebas realizadas en un entorno agrícola controlado, donde se comprobó que la solución es capaz de mantener condiciones adecuadas para el desarrollo

del cultivo mediante el monitoreo continuo y la actuación automática del sistema. Por su enfoque práctico y la utilización de tecnologías accesibles como Arduino y redes inalámbricas de corto alcance, este proyecto ofrece elementos valiosos que pueden ser considerados en el diseño de sistemas de monitoreo similares, especialmente en lo relacionado con la integración de sensores económicos y la comunicación eficiente entre nodos.

1.6.2. Marco teórico

Agricultura de precisión:

La agricultura de precisión constituye un enfoque de gestión agrícola que integra herramientas digitales y tecnologías avanzadas para mejorar la eficiencia en la producción de los cultivos. Su fundamento radica en reconocer que las condiciones del suelo y el comportamiento de las plantas varían dentro de un mismo terreno, por lo que las decisiones relacionadas con el manejo agronómico deben ajustarse a las características específicas de cada área. Esta estrategia favorece una administración más eficiente de los recursos, optimiza el rendimiento de los cultivos y contribuye a una producción agrícola más sostenible.

Para la implementación de este enfoque se emplean diferentes tecnologías, entre ellas sistemas de posicionamiento global, sensores remotos, imágenes satelitales, cámaras multiespectrales y maquinaria equipada con sistemas de aplicación variable. Estas herramientas permiten obtener información detallada sobre las condiciones del terreno y del cultivo, facilitando la ejecución de actividades como el levantamiento topográfico, la preparación y nivelación del suelo, la siembra mecanizada, la aplicación diferenciada de fertilizantes y la cosecha de acuerdo con las necesidades particulares de cada sector del lote.

Como parte de estas iniciativas, el grupo de investigación GAMMA ha desarrollado una metodología que integra datos obtenidos mediante sensores remotos con modelos estadísticos y herramientas de apoyo para la toma de decisiones. A través de este procedimiento es posible estimar los requerimientos nutricionales de los cultivos, identificar la presencia de plagas mediante el procesamiento de imágenes y generar recomendaciones técnicas que contribuyen a mejorar el manejo agronómico y la productividad de los sistemas agrícolas.

Entre las principales aplicaciones de la agricultura de precisión se encuentran la elaboración de levantamientos topográficos, la fertilización de dosis variable, la cosecha de precisión y la generación de mapas de productividad que permiten analizar el comportamiento de los cultivos dentro de cada lote.

Asimismo, este enfoque ofrece beneficios importantes para los productores agrícolas, ya que facilita el análisis de información proveniente de imágenes satelitales y sensores especializados, incrementa la precisión en la ejecución de las labores agrícolas, mejora la utilización de insumos y proporciona información técnica y económica que respalda la toma de decisiones en el manejo de los cultivos (Universidad de Antioquia, s. f.).

Humedad del suelo:

La humedad del suelo corresponde a la cantidad de agua que hay en un terreno, puede expresarse como porcentaje de agua por peso o por volumen, o pulgadas de agua por pie de suelo. La humedad del suelo depende de varios factores como el clima, el tipo de terreno y el tipo de plantas.

La humedad del suelo impacta directamente a los organismos y plantas que habitan un terreno, la salud de las plantas depende de la cantidad de agua presente. Además, la humedad del suelo impacta a:

- el contenido de aire, la salinidad y la cantidad de sustancias tóxicas
- su estructura y espesor del terreno
- la temperatura y capacidad calorífica

Humedad ideal del suelo:

La humedad ideal del suelo depende de dos parámetros fundamentales: la capacidad de campo (FC) y el punto de marchitamiento permanente (PWP).

- Capacidad de campo (FC): es la cantidad de agua que el suelo puede contener después de que el exceso se ha drenado. Representa el equilibrio adecuado entre agua y oxígeno en los poros del suelo. Si la humedad supera este nivel, disminuye el oxígeno disponible y puede afectar negativamente a las plantas.

- Punto de marchitamiento permanente (PWP): es el nivel de humedad por debajo del cual las plantas no pueden absorber suficiente agua, empiezan a marchitarse y pueden morir.

Ambos valores varían según el tipo de suelo, por lo que es esencial analizarlo para determinar el intervalo óptimo de humedad. La deficiencia de humedad del suelo se define como la diferencia entre la cantidad de agua presente y la cantidad máxima que el suelo puede retener.

Cómo se mide la humedad del suelo:

La humedad del suelo puede medirse mediante distintos métodos que varían según la fuente de datos utilizada. Los principales son: medición gravimétrica, medición con uso de sensores y la teledetección.

El método gravimétrico es un procedimiento directo y manual que consiste en extraer el agua de una muestra de suelo mediante evaporación, lavado y una reacción química.

Los sensores de humedad del suelo utilizan dos enfoques principales para la medición, los cuales dependen de la tecnología utilizada:

- Contenido Volumétrico de Agua (VWC): representa el volumen de agua respecto al volumen total de suelo. Se expresa en porcentaje y permite estimar el déficit hídrico del campo, ayudando a planificar el riego.

  VWC (%) = [volumen de agua / volumen de suelo] × 100

- Tensión del Agua del Suelo (SWT): indica la energía necesaria para que la planta absorba el agua. A mayor tensión, menor disponibilidad hídrica. Se mide en centibares y permite identificar estados de estrés hídrico para ajustar el riego.

Invernadero:

Un invernadero es una estructura diseñada para generar un ambiente controlado que favorece el crecimiento y desarrollo de los cultivos. Generalmente está construido con materiales transparentes o translúcidos que permiten el paso de la radiación solar y contribuyen a mantener condiciones internas adecuadas de temperatura, humedad y luminosidad. Gracias a este sistema es posible disminuir la influencia de factores climáticos adversos, como las lluvias intensas, las heladas, los vientos fuertes o las temperaturas extremas, proporcionando un entorno más estable para la producción agrícola (Castillo Arnedo, s. f.).

El funcionamiento de un invernadero se fundamenta en la creación de un microclima que favorece el desarrollo de las plantas. La energía proveniente del sol atraviesa la cubierta de la estructura y se transforma en calor, el cual permanece parcialmente retenido en el interior. Este proceso puede complementarse con sistemas de ventilación, calefacción, sombreo y riego que permiten mantener las condiciones ambientales requeridas por cada cultivo, optimizando su crecimiento y rendimiento (Castillo Arnedo, s. f.).

En la agricultura moderna, los invernaderos representan una alternativa para incrementar la productividad y garantizar una oferta continua de alimentos durante todo el año. Su implementación facilita la producción fuera de las temporadas tradicionales, reduce las pérdidas ocasionadas por fenómenos climáticos y mejora el aprovechamiento de recursos como el agua y los fertilizantes. Asimismo, al mantener un ambiente más controlado, disminuye la incidencia de plagas y enfermedades, lo que puede reducir la necesidad de aplicar productos fitosanitarios y contribuir a prácticas agrícolas más sostenibles (Castillo Arnedo, s. f.).

Los avances tecnológicos han impulsado el desarrollo de invernaderos cada vez más eficientes, incorporando sensores para el monitoreo ambiental, sistemas automatizados de riego, control climático y tecnologías de cultivo como la hidroponía y la aeroponía. Estas innovaciones permiten optimizar el uso de los recursos disponibles, mejorar la calidad de los productos obtenidos y aumentar la rentabilidad de la producción agrícola, convirtiendo al invernadero en una herramienta fundamental para la agricultura de precisión y la seguridad alimentaria (Castillo Arnedo, s. f.).

Telemetría:

Es un proceso automatizado para la monitorización de datos de manera remota, mediante el cual se recopilan datos de forma remota en lugares remotos y de difícil acceso permanente, este proceso se realiza mediante sensores que capturan los datos y los envían a equipos receptores para el correspondiente procesamiento y almacenamiento de los datos. La información se envía a través de conexiones de red, que pueden ser cableadas o por radiofrecuencia.

Los componentes principales de un sistema de telemetría son:

- Transductor:
  También puede denominarse sensor, es un dispositivo que convierte un estímulo físico (temperatura, vibración, presión, humedad) en una señal eléctrica, esta señal eléctrica se envía a un circuito que adapta o corrige la señal para su posterior transmisión.

- Medio de transmisión:
  Es el medio físico por el cual se transportan los datos entre los sensores y el receptor, pueden ser conexiones cableadas o conexiones inalámbricas. Generalmente se usan protocolos de comunión como HTTP o MQTT para Internet de las cosas (IoT).

- Receptor:
  Es el encargado de recibir las señales de las mediciones y de representarlas gráficamente de forma inteligible par que puedan ser interpretadas por una persona

IoT:

Internet de las Cosas (IoT) es un conjunto de dispositivos físicos equipados con sensores, actuadores y capacidad de conectividad mediante redes, que pueden comunicarse entre sí y con sistemas externos para intercambiar datos y ejecutar acciones automáticas sin necesidad estricta de intervención humana.

Estos dispositivos pueden variar en sus usos, pueden ser desde dispositivos domésticos simples como termostatos inteligentes, lámparas inteligentes o relojes, hasta complejas máquinas industriales utilizadas para la monitorización en fábricas, el control de máquinas y procesos, el seguimiento de variables ambientales, entre otros.

El IoT tiene aplicaciones muy amplias en distintos sectores como la industria, el transporte, la salud y la agricultura. Con el aumento de los dispositivos conectados, esta tecnología está cambiando la forma en que vivimos y trabajamos.

1.6.3. Marco conceptual

Sistema Web:

Un sistema web es una aplicación informática que funciona a través de un navegador y permite a los usuarios acceder a la información o ejecutar diferentes procesos mediante una conexión a Internet, sin necesidad de instalar programas en cada equipo. Su arquitectura centralizada facilita que tanto los datos como la lógica del sistema se administren en un servidor, lo que simplifica las tareas de mantenimiento, actualización y seguridad (Crea System, 2022).

Una de las principales ventajas de los sistemas web es su capacidad para operar en diferentes dispositivos y sistemas operativos, permitiendo el acceso desde computadores, tabletas o teléfonos inteligentes. Además, las actualizaciones se realizan de forma centralizada, garantizando que todos los usuarios trabajen con la misma versión del sistema y reduciendo la posibilidad de errores ocasionados por versiones desactualizadas (Crea System, 2022).

En el ámbito organizacional, los sistemas web contribuyen a la automatización de procesos, la gestión eficiente de la información y el apoyo a la toma de decisiones. Su implementación favorece la disponibilidad de datos en tiempo real, optimiza los flujos de trabajo y mejora la productividad al facilitar el acceso seguro a la información desde cualquier ubicación con conexión a Internet (Crea System, 2022).

MQTT:

Es un protocolo de comunicación que se utiliza para transferir información de un equipo a otro, generalmente usado por dispositivos inteligentes y por dispositivos del Internet de las Cosas que tienen que transferir mensajes en una red con recursos y ancho de banda limitados. Una de sus mayores ventajas es que es un protocolo ligero (utiliza pocos recursos de red) y fácil de implementar.

El protocolo MQTT utiliza un modelo de publicación – suscripción, con el fin de desacoplar el emisor del mensaje (publicador) del receptor (suscriptor) mediante un tercer competente denominado agente MQTT, quien filtra y distribuye los mensajes de los publicadores a los suscriptores correspondientes.

Componentes MQTT

- **Cliente MQTT:**

Es un dispositivo de cualquier tipo (dispositivo inteligente, microcontrolador, computador, etc.) que integra una biblioteca MQTT para la comunicación con un agente MQTT. Si el cliente envía mensajes se le denomina publicador y si recibe mensajes se denomina suscriptor.

- **Agente MQTT:**

Es un sistema o servicio corriendo en un servidor actuando como back-end, es el encargado de coordinar los mensajes entrantes enviados por los clientes y re-enviarlos nuevamente a los dispositivos suscritos a un “tema” o “tópico” específico. Adicionalmente se encarga de la autorización, autenticación y control de pérdida de mensajes.

Un tema se refiere a palabras que el cliente MQTT utiliza como referencia para recibir, filtrar y enviar los mensajes a los dispositivos correspondientes. Un tema tiene similitud a una dirección URL y pueden estar organizados jerárquicamente.

Funcionamiento:

Primero, un cliente establece una conexión con el agente MQTT, una vez conectado el cliente puede actuar como publicador y/o como suscriptor. Si el cliente actúa como publicador, este puede enviar o publicar mensajes a un tema específico, por ejemplo, al tema “dispositivos/invernadero/riego”. Si el cliente actúa como suscriptor, el cliente se suscribe a un tema, por ejemplo, al tema “dispositivos/invernadero/riego”. En este caso todos los suscriptores a dicho tema recibirán todos los mensajes enviados por los clientes que están enviando o publicando mensajes en el mismo tema. Los mensajes pueden ser enviados y recibidos en diferentes formatos como texto plano, datos binarios, XML o JSON.

HTTP:

HTTP o el “Protocolo de Transferencia de Hipertexto” es un protocolo para la transmisión de documentos de hipermedia que funciona en la capa de aplicación del modelo TCP/IP, diseñado para permitir la comunicación entre los navegadores web y los servidores.

Este protocolo funciona bajo el modelo cliente-servidor para permitir la comunicación entre dispositivos de la red. El cliente es un dispositivo o software que realiza una petición a un servidor solicitando o enviando algún recurso (texto, página, imagen, archivos, etc.). El servidor es el dispositivo encargado de almacenar y entregar los recursos solicitados por los clientes.

El protocolo HTTP establece diferentes métodos que indican cual es la acción que se quiere realizar en cada solicitud, los métodos las comunes son:

- POST: Para enviar datos al servidor.
- GET: Para solicitar un recurso al servidor.
- PUT: Indica que se va a actualizar un recurso existente en el servidor.
- DELETE: Para eliminar un recurso.
- HEAD: Se usa para solicitar la cabecera de la solicitud hecha al servidor.

WebSocket:

WebSocket es un protocolo de comunicación que permite el intercambio continuo de información entre un cliente y un servidor mediante una conexión persistente. A diferencia del modelo tradicional basado en solicitudes y respuestas de HTTP, esta tecnología establece un canal bidireccional que posibilita el envío y la recepción de datos en tiempo real, sin necesidad de crear una nueva conexión para cada intercambio de información (IBM, s. f.).

Una de las principales características de WebSocket es su capacidad para mantener una comunicación permanente durante toda la sesión, lo que reduce la latencia y el consumo de recursos de red. Gracias a este comportamiento, es una tecnología ampliamente utilizada en aplicaciones que requieren actualizaciones instantáneas, como sistemas de monitoreo, plataformas colaborativas, aplicaciones de mensajería, videojuegos en línea y soluciones de Internet de las Cosas (IoT), donde la información debe transmitirse de forma inmediata entre dispositivos y servidores.

En proyectos de automatización y agricultura inteligente, WebSocket facilita la comunicación en tiempo real entre sensores, dispositivos de control y aplicaciones web. Esto permite supervisar variables ambientales, visualizar datos actualizados, enviar alertas y ejecutar acciones de manera remota, mejorando la capacidad de respuesta del sistema y apoyando la toma de decisiones basada en información obtenida en tiempo real (IBM, s. f.).

Back-end:

El término back-end proviene del inglés y significa “parte de atrás”. En informática, especialmente en el ámbito de la programación, se refiere a la sección de un sistema, página web o aplicación que no es visible para el usuario final, pero que resulta fundamental para su funcionamiento. En esta capa se configuran los aspectos lógicos del desarrollo, como el almacenamiento de datos, los mecanismos de seguridad y los procesos necesarios para que todas las acciones solicitadas por el usuario se ejecuten correctamente.

Front-end:

Las aplicaciones web modernas se estructuran principalmente en dos componentes: el frontend y el backend, los cuales trabajan de manera integrada para ofrecer una experiencia funcional al usuario. El frontend corresponde a la parte visible de la aplicación, es decir, la interfaz con la que interactúan los usuarios. En este componente se desarrollan los elementos gráficos, la navegación y la presentación de la información mediante tecnologías como HTML, CSS y JavaScript, buscando proporcionar una interacción intuitiva y eficiente (Amazon Web Services [AWS], s. f.).

Por otra parte, el backend comprende el conjunto de procesos que se ejecutan en el servidor y que permiten el funcionamiento interno de la aplicación. Este componente se encarga de procesar las solicitudes enviadas por el frontend, gestionar la lógica del negocio, administrar la autenticación de usuarios y realizar las operaciones sobre las bases

de datos. Gracias a esta arquitectura, la información puede almacenarse, procesarse y recuperarse de forma segura y eficiente (AWS, s. f.).

La comunicación entre el frontend y el backend se realiza mediante protocolos y servicios que permiten el intercambio de información entre ambos componentes. Esta separación de responsabilidades facilita el mantenimiento, la escalabilidad y la actualización de las aplicaciones, además de permitir que cada parte evolucione de manera independiente. En proyectos de monitoreo y automatización, como los sistemas de control para invernaderos inteligentes, esta arquitectura posibilita la visualización de datos en tiempo real, el almacenamiento de registros históricos y la gestión remota de dispositivos conectados, favoreciendo una administración eficiente del sistema (AWS, s. f.).

ESP32:

El ESP32 es un microcontrolador de alto rendimiento desarrollado por Espressif Systems, ampliamente utilizado en proyectos de Internet de las Cosas (IoT) debido a que integra conectividad Wi-Fi y Bluetooth en un solo dispositivo. Estas características permiten que el microcontrolador establezca comunicación con otros equipos, sensores y plataformas en la nube sin requerir módulos de comunicación adicionales, convirtiéndose en una solución eficiente y de bajo costo para aplicaciones de automatización y monitoreo (Carranza, 2021).

Entre sus principales características se encuentran su procesador de doble núcleo, bajo consumo energético, múltiples pines de entrada y salida (GPIO) y el soporte para diversos protocolos de comunicación, como UART, SPI e ²C. Asimismo, incorpora convertidores analógicos y digitales, salidas PWM y entradas capacitivas, lo que facilita la conexión de una amplia variedad de sensores y actuadores utilizados en sistemas inteligentes. Estas capacidades permiten desarrollar soluciones con un alto grado de flexibilidad y escalabilidad.

El ESP32 es compatible con diferentes entornos de desarrollo, entre ellos ESP-IDF y Arduino IDE, siendo este último uno de los más utilizados por su facilidad de uso y su amplia comunidad de desarrolladores. Esta compatibilidad simplifica el desarrollo de aplicaciones para monitoreo remoto, automatización y adquisición de datos en tiempo real, reduciendo el tiempo de implementación de nuevos proyectos.

En el ámbito de la agricultura inteligente, el ESP32 desempeña un papel fundamental como unidad de adquisición y procesamiento de datos. Su capacidad para recibir información de sensores ambientales, controlar dispositivos como bombas de agua, ventiladores o sistemas de iluminación y transmitir datos hacia aplicaciones web o plataformas en la nube permite implementar sistemas de supervisión y control remoto que optimizan la gestión de los cultivos y mejoran la eficiencia en el uso de los recursos.

Sensor de humedad:

El sensor de humedad es un dispositivo electrónico diseñado para medir la cantidad de humedad presente en un ambiente o en un medio específico y convertir esa información en una señal que puede ser procesada por un sistema de control. Su función principal es proporcionar datos precisos sobre las condiciones ambientales, permitiendo supervisar y mantener niveles adecuados de humedad según los requerimientos de cada aplicación (Internet República, 2023).

El funcionamiento de estos sensores se basa en la detección de cambios en la humedad relativa del entorno mediante elementos sensibles que responden a las variaciones del vapor de agua. La información capturada se transmite a un controlador o sistema automatizado, el cual puede generar alertas o activar dispositivos como extractores, ventiladores, humidificadores o sistemas de riego cuando los valores registrados se encuentran fuera de los rangos establecidos. Esta capacidad favorece la automatización de procesos y el control continuo de las condiciones ambientales (Internet República, 2023).

En aplicaciones de agricultura inteligente e Internet de las Cosas (IoT), los sensores de humedad desempeñan un papel fundamental al proporcionar información en tiempo real para la toma de decisiones. Integrados con microcontroladores como el ESP32 y plataformas de monitoreo web, permiten supervisar las condiciones del cultivo, optimizar el uso del agua y automatizar sistemas de riego, contribuyendo a mejorar la productividad y la eficiencia en el manejo de los recursos.

PHP:

PHP (Hypertext Preprocessor) es un lenguaje de programación de código abierto ampliamente utilizado para el desarrollo de aplicaciones web dinámicas. Su funcionamiento se basa en la ejecución del código en el servidor, donde procesa la información solicitada por el usuario y genera el contenido que posteriormente es enviado al navegador. Esta característica permite crear aplicaciones capaces de gestionar datos, procesar formularios, administrar sesiones y ofrecer contenido dinámico de manera eficiente (PHP Documentation Group, s. f.).

Una de las principales ventajas de PHP es su facilidad de integración con tecnologías web como HTML, CSS y JavaScript, además de su compatibilidad con diferentes sistemas operativos, servidores web y sistemas gestores de bases de datos. Gracias a esta flexibilidad, se ha convertido en una de las tecnologías más empleadas para el desarrollo de plataformas web, sistemas de información y aplicaciones empresariales.

En proyectos de Internet de las Cosas (IoT), PHP suele utilizarse en el desarrollo del backend, donde se encarga de recibir la información enviada por los dispositivos, procesarla, almacenarla en bases de datos y suministrarla a las interfaces de usuario para su visualización. En aplicaciones orientadas a la agricultura inteligente, esta tecnología facilita la administración de datos provenientes de sensores ambientales y permite implementar funcionalidades como el monitoreo en tiempo real, la generación de reportes y el control remoto de los sistemas de automatización (PHP Documentation Group, s. f.).

Laravel:

Laravel es un framework de código abierto desarrollado para facilitar la creación de aplicaciones web utilizando el lenguaje de programación PHP. Su arquitectura incorpora herramientas y componentes que simplifican el desarrollo de software, permitiendo organizar el código de manera estructurada y aplicar buenas prácticas de programación. Entre sus principales características se encuentran el sistema de enrutamiento, la autenticación de usuarios, la gestión de bases de datos mediante un mapeo objeto-relacional (ORM), la validación de datos y el procesamiento de tareas en segundo plano, lo que favorece el desarrollo de aplicaciones escalables y de fácil mantenimiento (Laravel, s. f.).

Una de las ventajas más importantes de Laravel es que implementa el patrón de arquitectura Modelo–Vista–Controlador (MVC), el cual separa la lógica del negocio, la interfaz de usuario y el acceso a los datos. Esta organización mejora la legibilidad del código, facilita el mantenimiento del sistema y permite que varios desarrolladores trabajen de forma simultánea sobre un mismo proyecto sin afectar su funcionamiento (Laravel, s. f.).

En aplicaciones orientadas al Internet de las Cosas (IoT), Laravel puede utilizarse como la base del backend, encargándose de recibir la información enviada por dispositivos como el ESP32, procesar los datos obtenidos por los sensores, almacenarlos en la base de datos y suministrarlos a la interfaz web para su consulta en tiempo real. Además, facilita la implementación de mecanismos de autenticación, administración de usuarios, generación de reportes y comunicación con otros servicios, convirtiéndose en una plataforma robusta para el desarrollo de sistemas de monitoreo y automatización.

Vue:

Vue.js es un framework progresivo de JavaScript diseñado para el desarrollo de interfaces de usuario dinámicas e interactivas. Su principal característica es permitir la construcción de aplicaciones web mediante una arquitectura basada en componentes reutilizables, lo que facilita la organización del código, mejora la mantenibilidad del software y favorece el desarrollo de aplicaciones de diferentes niveles de complejidad (Vue.js, s. f.).

Una de las fortalezas de Vue.js es su sistema de reactividad, el cual actualiza automáticamente la interfaz cuando los datos de la aplicación cambian, evitando la manipulación manual del Documento de Objetos del Modelo (DOM). Asimismo, ofrece una sintaxis declarativa que simplifica la creación de interfaces y permite integrar de forma eficiente HTML, CSS y JavaScript dentro de un mismo entorno de desarrollo. Estas características contribuyen a mejorar la experiencia del usuario y optimizan el rendimiento de las aplicaciones web.

Otra ventaja importante de Vue.js es su flexibilidad, ya que puede utilizarse tanto para incorporar funcionalidades dinámicas en sitios web existentes como para desarrollar aplicaciones de una sola página (Single Page Applications - SPA). Además, es compatible con diversas herramientas del ecosistema moderno de desarrollo web y puede integrarse

con servicios backend, APIs y bases de datos, permitiendo construir soluciones escalables y de alto desempeño.

En el contexto de este proyecto, Vue.js se emplea para el desarrollo del frontend del sistema de monitoreo del invernadero inteligente. Su utilización facilita la visualización en tiempo real de la información proveniente de los sensores, la interacción con los usuarios mediante una interfaz intuitiva y la comunicación con el backend desarrollado en Laravel, proporcionando una experiencia de uso eficiente y una administración centralizada de los datos del sistema.

MariaDB:

MariaDB es un sistema de gestión de bases de datos relacionales de código abierto que permite almacenar, organizar y administrar grandes volúmenes de información de manera eficiente. Su diseño está orientado a garantizar un alto rendimiento, confiabilidad y escalabilidad, lo que ha favorecido su adopción en aplicaciones web, sistemas empresariales y plataformas que requieren un procesamiento continuo de datos. Además, mantiene una alta compatibilidad con MySQL, facilitando la migración de aplicaciones desarrolladas sobre este motor de base de datos sin necesidad de realizar modificaciones significativas (MariaDB Foundation, s. f.).

Entre sus principales características se destacan el soporte para transacciones, mecanismos de seguridad, múltiples motores de almacenamiento y la compatibilidad con el lenguaje SQL para la consulta y manipulación de datos. Asimismo, MariaDB ofrece herramientas que favorecen la disponibilidad de la información, el procesamiento eficiente de grandes cantidades de registros y la integración con diferentes lenguajes de programación y frameworks de desarrollo web, lo que la convierte en una alternativa ampliamente utilizada para el desarrollo de aplicaciones modernas.

En proyectos de Internet de las Cosas (IoT) y automatización, MariaDB desempeña un papel esencial como repositorio central de la información generada por los dispositivos conectados. En el caso de un invernadero inteligente, la base de datos permite registrar las mediciones capturadas por sensores de temperatura, humedad, luminosidad y otras variables ambientales, facilitando su consulta, análisis y visualización mediante aplicaciones web. De esta manera, contribuye al monitoreo en tiempo real, al almacenamiento de datos históricos y a la toma de decisiones orientadas a optimizar las condiciones del cultivo (MariaDB Foundation, s. f.).

Eventos:

Los eventos constituyen un mecanismo que permite a las aplicaciones web detectar y responder a las acciones realizadas por los usuarios o a los cambios que ocurren durante la ejecución del sistema. Estas acciones pueden incluir, entre otras, hacer clic sobre un botón, ingresar información en un formulario, mover el cursor, presionar una tecla o finalizar la carga de una página. Cuando alguno de estos sucesos ocurre, el navegador genera un evento que puede ser capturado por el programa para ejecutar una acción determinada, favoreciendo la interacción dinámica entre el usuario y la aplicación (MDN Web Docs, s. f.).

La gestión de eventos se realiza mediante funciones conocidas como manejadores o controladores de eventos, las cuales se asocian a elementos específicos de la interfaz utilizando mecanismos como addEventListener(). Este enfoque permite desarrollar aplicaciones con un comportamiento interactivo y organizado, facilitando la implementación de funcionalidades como la validación de formularios, la actualización de información en tiempo real y la ejecución de procesos en respuesta a las acciones del usuario.

En el desarrollo de aplicaciones modernas, especialmente aquellas construidas con frameworks como Vue.js, el manejo de eventos resulta esencial para establecer la comunicación entre la interfaz y la lógica de la aplicación. En proyectos de monitoreo y automatización, como un invernadero inteligente, los eventos permiten actualizar la información proveniente de los sensores, responder a las acciones del operador y controlar dispositivos de manera inmediata, proporcionando una experiencia de uso más dinámica y eficiente.

Servidor Privado Virtual (VPS):

Un Servidor Privado Virtual (VPS, por sus siglas en inglés *Virtual Private Server*) es un entorno de alojamiento que proporciona recursos de procesamiento, memoria y almacenamiento dedicados dentro de un servidor físico mediante técnicas de virtualización. A diferencia del alojamiento compartido, un VPS ofrece un mayor nivel de independencia y control sobre la configuración del sistema operativo, las aplicaciones y los servicios instalados, permitiendo adaptar el entorno a los requerimientos específicos de cada proyecto (Hostinger, s. f.).

Una de las principales ventajas de un VPS es la posibilidad de administrar el servidor con acceso completo, lo que facilita la instalación de diferentes tecnologías, lenguajes de programación, bases de datos y herramientas de desarrollo. Además, este tipo de infraestructura ofrece un mejor desempeño, mayor estabilidad y capacidad de escalamiento, convirtiéndose en una alternativa adecuada para aplicaciones que requieren un procesamiento continuo de datos o un número creciente de usuarios. Estas características hacen que los VPS sean ampliamente utilizados para alojar aplicaciones web, servicios de bases de datos, APIs y plataformas de monitoreo en tiempo real (Hostinger, s. f.).

En proyectos de Internet de las Cosas (IoT), un VPS actúa como la infraestructura central donde se ejecutan los servicios que reciben, procesan y almacenan la información transmitida por los dispositivos conectados. En el caso de un invernadero inteligente, el servidor puede alojar el backend desarrollado en Laravel, la base de datos MariaDB, los servicios de comunicación mediante WebSocket y la interfaz web implementada con Vue.js, permitiendo el monitoreo remoto de las variables ambientales, el almacenamiento histórico de las mediciones y la administración del sistema desde cualquier ubicación con acceso a Internet.

1.7. Factibilidad

1.7.1. Factibilidad Técnica

El desarrollo del proyecto requiere unos recursos mínimos de software y hardware, así como de recursos humanos para la implementación de los requerimientos y el cumplimiento de los objetivos del proyecto. A continuación, se describen los recursos necesarios para el desarrollo e implementación del proyecto:

Recurso	Descripción	Cant.
Computador	Ryzen 5 16 GB RAM Sistema Operativo Incluido	2
Servidor / Hosting	VPS OVH Cloud VPS - 1	1
Tarjeta de desarrollo	Tarjeta De Desarrollo Esp32 Wifi Bluetooth 30 Pines	1
Sensor de humedad	Sensor de Humedad de Suelo Capacitivo V1.2	1
Prototipo	Cables, pines, soldadura, etc.	1

Tabla 5. Recursos de Hardware. Fuente: Elaboración propia

En la Tabla 6 se listan los elementos de software necesarios para el desarrollo e implementación del sistema.

Recurso	Descripción	cantidad
Lenguajes de programación	PHP, JavaScript	1
Frameworks	Laravel, Vue	1
Base de datos	MariaDB	1
IDE de desarrollo de software	Visual Studio Code.	2
Pruebas	PostMan.	2

Tabla 6. Recursos de Software. Fuente: Elaboración propia

En la siguiente tabla se detallan los recursos humanos necesarios para el desarrollo del proyecto

Rol	Descripción	Cantidad
Tutor	Asesorías para la realización del proyecto.	60 horas
Desarrolladores	Dos programadores que realicen el desarrollo y la implementación del proyecto asegurando los objetivos del mismo.	384 horas

Tabla 7. Recursos Humanos. Fuente: Elaboración propia

1.7.2. Factibilidad Operativa

La implementación del sistema de monitoreo se realizará de tal manera que el uso y el acceso a la aplicación sea realizada mediante la web (Navegadores web), esto permite que el personal y los usuarios requieran de un conocimiento básico en el manejo de herramientas digitales y conceptos mínimos de humedad del suelo y riego en cultivos. Adicionalmente y debido a que la recolección de datos y la generación de alertas se hace de manera automatizada, solo se requiere que los usuarios finales se encarguen de la visualización de los datos y de atender de manera autónoma las alertas que genera el sistema.

Así mismo, la operación de todo el sistema en conjunto no demanda la presencia de personal técnico permanente, debido a que los mantenimientos pueden hacerse de manera periódica. Por último, la implementación de todo el sistema, asegurando su correcto funcionamiento estará a cargo de los miembros del equipo enmarcado en la metodología Scrum.

1.7.3. Factibilidad Económica

Para garantizar que el proyecto sea económicamente viable se contempla la utilización de herramientas de código abierto para reducir los gastos en licencias de software, así como la adquisición de elementos de hardware mínimos que se aseguren el cumplimiento de los requerimientos y los objetivos del proyecto, a continuación, se detallan los elementos necesarios y sus precios.

Recurso	Descripción	Costo unitario	Cant.	Total
Computador	Ryzen 5 16 GB RAM S.O Windows	$ 2.500.000	2	$ 5.000.000
Servidor	VPS OVH Cloud VPS - 1	$ 17.939 / mes	6	$ 107.634
API SMS	API para el envío de mensajes SMS.	$ 19	1200	$ 22800
API correo	API plataforma Resend para el envío de correo electrónico (Free)	$ 0	3000	$ 0
Tarjeta de desarrollo	Tarjeta De Desarrollo Esp32 Wifi Bluetooth 30 Pines	$ 24.000	1	$ 24.000
Sensor de humedad	Sensor de Humedad de Suelo Capacitivo V1.2	$ 15.000	1	$ 15.000
Otros	Cables, pines, soldadura, etc.	$ 30.000	1	$ 40.000
Total				**$ 5.209.434**

Tabla 8. Lista de precios recursos de hardware.
Fuente: Elaboración propia

En la tabla 9 se detallan los elementos de software de código abierto necesarios para el proyecto.

Tipo	Descripción	Costo unitario	cantidad
Lenguajes de programación	PHP, JavaScript	De uso libre	1
Base de datos	MariaDB	De uso libre	1
IDE de desarrollo de software	Visual Studio Code.	De uso libre	2

Tabla 9. Costo recursos de software. Fuente: Elaboración Propia

Rol	Descripción	Valor Hora	Cantidad	Total
Tutor	Asesorías para la realización del proyecto.	$ 40.000	60 horas	$ 2.400.000
Desarrolladores	Dos programadores que realicen el desarrollo y la implementación del proyecto asegurando los objetivos del mismo.	$ 30.000	384 horas	$ 11.520.000
Total Recursos Humanos				**$ 13.920.000**

Tabla 10. Precios recursos humanos. Fuente: Elaboración propia

En la Tabla 11 se detallan los cálculos hechos para la obtención de las horas necesarias:

Rol	Horas por semana	Semanas (6 meses)	Total Horas
Tutor	2.5	24	60
Desarrollador 1	8	24	192
Desarrollador 2	8	24	192
Total			**384**

Tabla 11. Detalle de horas de recursos humanos.
Fuente: Elaboración propia

Tipo	Descripción	Costo unitario	Cantidad	Total
Internet	Conexión a internet operador Tigo	$ 70.000	12	$ 840.000
Servicios Luz	Codensa	$ 30.000	12	$ 360.000
Total				**$ 1.200.000**

Tabla 12. Otros costos. Fuente: Elaboración propia

La cantidad de cada elemento corresponde a 12 teniendo en cuenta que los miembros del equipo encargados de desarrollar el proyecto son 2 personas y el tiempo estimado para su ejecución son 6 meses.

En la Tabla 13 se muestra el costo total del proyecto.

Recurso	Valor
Recurso Humanos	$ 13.920.000
Recursos técnicos	$ 5.209.434
Otros	$ 1.200.000
Imprevistos (10%)	$ 2.032.943
TOTAL	**$ 22.362.377**

Tabla 13. Costo total del proyecto. Fuente: Elaboración propia

Teniendo en cuenta los costos planteados previamente se puede concluir que el proyecto es económicamente factible, dado que el equipo cuenta con algunos de los elementos requeridos y que la mayoría de las herramientas de software necesarias son de libre uso.

1.7.4. Factibilidad legal

El desarrollo e implementación del proyecto contempla el cumplimiento de la normatividad vigente en Colombia en materia de propiedad intelectual, protección de datos personales y uso de licencias de software, garantizando que todas las actividades y procesos se ajusten a las normas legales disponibles.

- **Ley de protección de datos personales**

En materia de protección de datos personales, el sistema cumple con lo dispuesto en la Ley 1581 de 2012, los cuales establecen el régimen general de protección de datos personales en Colombia. Por lo cual, toda la información personal recopilada en los módulos de administración y de usuario serán tratados bajo los principios que establece dicha ley.

- **Ley de derechos de autor**

En lo referente a la propiedad intelectual, el proyecto se acoge a lo establecido en la Ley 23 de 1982, que regula los derechos de autor en Colombia, garantizando el respeto y la debida atribución de las obras literarias, científicas y de software que sean utilizadas para la ejecución del proyecto.

- **Uso de licencias**

Para la implementación del sistema para la monitorización de la humedad del suelo y generación de alertas, se requieren de diversas licencias de uso, entre ellas, licencias de software y licencias de uso de servidor. El proyecto respeta las condiciones establecidas en las licencias de las herramientas de software, frameworks y librerías utilizadas, las cuales serán en su totalidad de uso libre. Asimismo, se cumplirán las licencias y condiciones de uso definidas por los proveedores del servidor virtual privado (VPS) y por los servicios o APIs externas contratadas para la emisión de correos electrónicos y mensajes SMS.

1.8. Cronograma

Id	Modo de tarea	Nombre de tarea	Duración	Duración restante
1		Analisis	20 días	20 días
2		1. Identificación y definición del problema	7 días	7 días
3		2. Contextualización de la problemática(Investigación)	6 días	6 días
4		3. Identificación de requrimientos	7 días	7 días
5		Diseño	23 días	23 días
6		1. Arquitectura y comunicación	3 días	3 días
7		2. Bases de datos	3 días	3 días
8		3. Modulos	5 días	5 días
9		4. Interfaces	5 días	5 días
10		5. Prototipo	7 días	7 días
11		Desarrollo Prototipo	25 días	25 días
12		1. Requisitos	2 días	2 días
13		2. Definición de funcionalidades	2 días	2 días
14		3. Montaje y configuración	14 días	14 días
15		4. Pruebas	7 días	7 días
16		Desarrollo Back-End	28 días	28 días
17		1. Configuración de hosting y servidor	3 días	3 días
18		2. Implementación de base de datos	3 días	3 días
19		3. Creación servicios web	15 días	15 días
20		4. Pruebas	7 días	7 días
21		Desarrollo Front-End	23 días	23 días
22		1. Desarrollo interface web	15 días	15 días
23		2. Representación grafica de los datos	3 días	3 días
24		3. Pruebas	5 días	5 días
25		Pruebas e Implementación	20 días	20 días
26		1. Integración del sistema	5 días	5 días
27		2. Pruebas de funcionalidad	5 días	5 días
28		3. Documentación	10 días	10 días
29			1 día?	1 día?

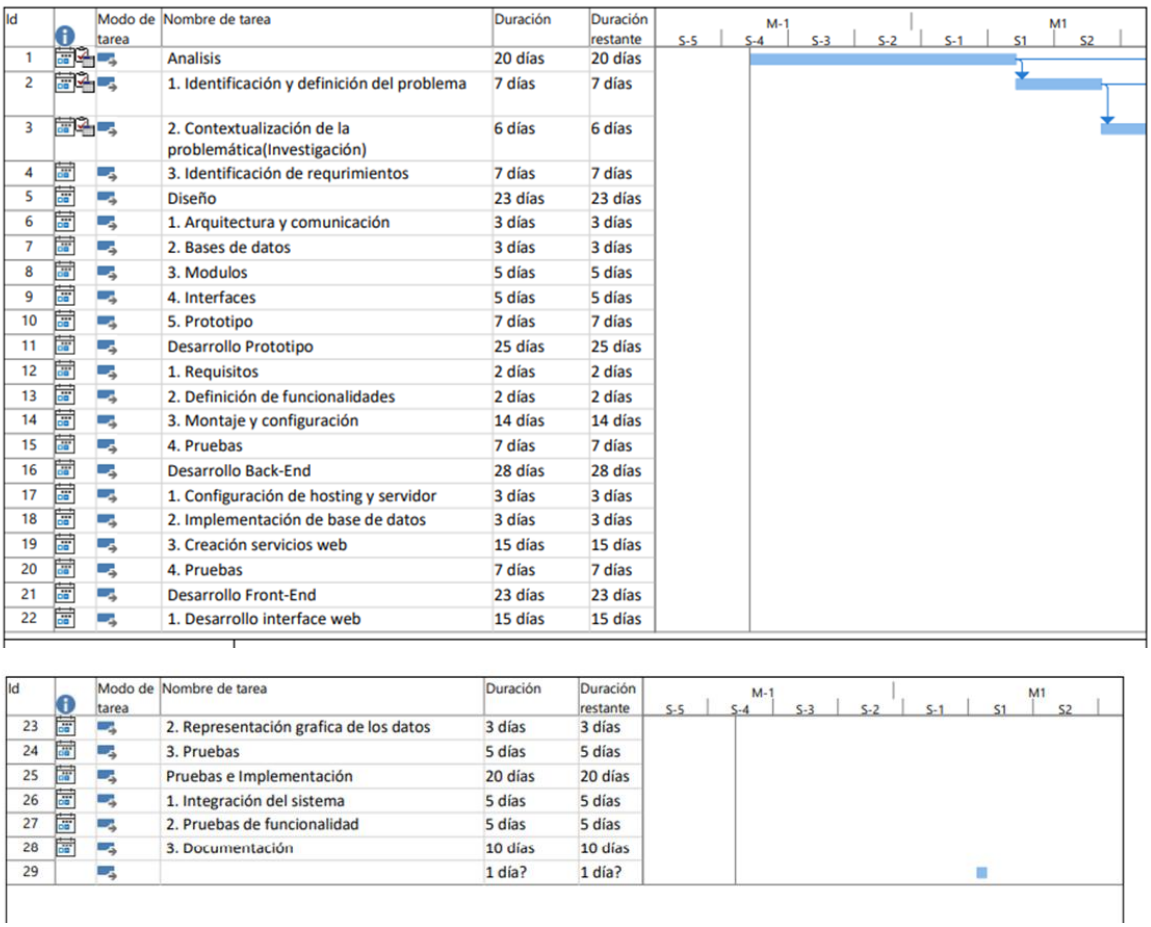


Figura 1. Cronograma

2. Tecnologías y procesos para implementar sistemas de monitorización remota

Para el desarrollo de este sistema se busca establecer el conjunto de procesos, tecnologías y herramientas necesarias y más adecuadas, que permitan implementar una solución orientada a monitorizar remotamente y en tiempo real la humedad de suelo en invernaderos. Mediante esta actividad se pretende garantizar que el sistema tenga capacidades de comunicación entre sus diferentes componentes, procesamiento central de la información, generación de alertas, captura de lecturas de humedad del suelo y representación de los datos al usuario final.

Para ello y de acuerdo con la metodología Scrum seleccionada para el desarrollo del proyecto, esta primera parte comprende el primer Sprint, correspondiente a la fase de análisis, en el cual se establecen los principales requerimientos y elementos necesarios para la definición de la solución. De acuerdo con lo anterior, se planificó el sprint con las siguientes actividades:

No.	Actividad	Descripción	Responsable
1	Descripción de la solución	Describir la solución propuesta identificando sus principales componentes y funcionalidades.	Michael Jesus Ubaque
2	Definición de la arquitectura de comunicación	Seleccionar y describir un modelo de arquitectura que permita la comunicación entre los sensores y un servidor central.	Michael Jesus ubaque
3	Definición de protocolos de comunicación	Describir los protocolos necesarios para permitir la comunicación entre los componentes del sistema.	Cristian Felipe Molina
4	Selección de frameworks de desarrollo	Seleccionar los frameworks de desarrollo necesarios para la implementación del backend y frontend.	Cristian Felipe Molina
5	Selección del sensor y microcontrolador	Analizar las alternativas disponibles y seleccionar el sensor de humedad del suelo y el microcontrolador que conformarán el prototipo.	Cristian Felipe Molina

Tabla 14. Planificación de actividades del Sprint 1.
Fuente: Elaboración propia

2.1. Descripción de la solución

Para apoyar el proceso de medición de la humedad del suelo realizado por agricultores y administradores de cultivos en invernaderos, se planteó el diseño de una solución tecnológica basada en la recolección de datos en tiempo real. Para lograrlo, se propuso el diseño de un sistema de telemetría basado en sensores de humedad del suelo, tecnología IoT y una plataforma web para la gestión y visualización de datos.

Las principales funcionalidades que se tuvieron en cuenta para el diseño del sistema telemático propuesto fueron las siguientes:

- La medición de la humedad del suelo en tiempo real mediante la captura de datos obtenidos directamente en los invernaderos con prototipos ubicados en el terreno, conectados a internet para el envío de la información en remoto.

- Capturar los datos por medio de la medición de la humedad en el suelo en tiempo real obtenido directamente con los prototipos ubicados en el terreno de los invernaderos, los cuales están conectados a internet para el envió de la información de forma remota.

- Almacenamiento y procesamiento de los datos recopilados en un servidor en la nube y acceso a los datos mediante una plataforma web, que permita su análisis y visualización en tiempo real.

- Generación de alertas automáticas, mediante correo electrónico y mensajes SMS, que notifiquen a los usuarios sobre lecturas fuera de rangos permitidos y previamente configurados.

- Gestión de usuarios, gestión de dispositivos de medición y configuración desde una plataforma web.

Teniendo en cuenta las funciones del sistema telemático, fue necesario establecer cuáles son las tecnologías, herramientas y procesos más adecuados para ser implementados en la solución propuesta. Por lo anterior, a continuación, se presentan y analizan diferentes alternativas de arquitectura, protocolos de comunicación, frameworks y sensores que fueron utilizados para el desarrollo de este proyecto.

2.2. Arquitectura IoT

El propósito de este proyecto es desarrollar un sistema que permita conocer el estado de la humedad del suelo en invernaderos de manera continua y desde cualquier lugar. Para lograrlo, es necesario contar con una arquitectura tecnológica que facilite la comunicación entre los sensores instalados en el cultivo y la plataforma donde se visualizará la información. En este sentido, la arquitectura basada en el Internet de las Cosas (IoT) representa la opción más adecuada, ya que permite conectar dispositivos físicos a Internet para recopilar, transmitir y procesar información de forma automática.

A diferencia de los métodos tradicionales, en los que las mediciones deben realizarse manualmente y dependen de la disponibilidad del personal, una arquitectura IoT hace posible que los datos sean capturados y enviados de forma permanente. Esto permite conocer el comportamiento de la humedad del suelo en tiempo real, facilitando un seguimiento constante de las condiciones del cultivo y brindando información confiable para apoyar la toma de decisiones relacionadas con el riego.

Otro aspecto que hace apropiada esta arquitectura es su flexibilidad. Aunque el sistema está orientado al monitoreo de la humedad del suelo, en el futuro podrá ampliarse fácilmente para incorporar otros sensores que miden variables como la temperatura, la humedad relativa o la intensidad lumínica. De esta manera, el sistema puede evolucionar sin necesidad de modificar completamente su estructura, lo que representa una ventaja importante para futuras mejoras o ampliaciones.

Para que todo el proceso funcione de manera organizada, la arquitectura propuesta se divide en varias etapas. Primero, los sensores obtienen la información directamente del suelo. Posteriormente, esos datos son enviados a través de la red hacia un servidor, donde son procesados y almacenados. Finalmente, la información es presentada al usuario mediante una plataforma web que permite consultar el estado del cultivo desde cualquier dispositivo con acceso a Internet.

2.3. Protocolo de comunicación

La comunicación entre los diferentes componentes es fundamental para el funcionamiento del sistema, ya que permite el intercambio de información entre los dispositivos y las diferentes partes que conforman la solución. Para lograr esta interacción, se emplean mecanismos de comunicación que permiten transportar y procesar los datos generados durante el funcionamiento del sistema, facilitando su posterior gestión y presentación al usuario. La integración de estos mecanismos permite establecer un flujo de información entre los distintos componentes de la arquitectura, de manera que cada uno pueda cumplir con la función asignada y contribuir al funcionamiento conjunto de la solución. A continuación, se presentan los principales protocolos de comunicación utilizados en el proyecto y su participación dentro de la arquitectura propuesta.

2.3.1. Uso del Protocolo MQTT

Una vez obtenidas las mediciones, es necesario transmitirlas al servidor utilizando un protocolo que sea eficiente y que no genere un consumo excesivo de recursos. Por esta razón se eligió MQTT, un protocolo desarrollado especialmente para aplicaciones de telemetría e Internet de las Cosas.

Su funcionamiento se basa en un modelo de publicación y suscripción, donde los dispositivos envían la información a un intermediario denominado broker MQTT, encargado de distribuir los datos a las aplicaciones que los necesitan. Este mecanismo reduce el tráfico en la red y permite que la comunicación sea rápida y estable, incluso cuando varios dispositivos transmiten información al mismo tiempo.

2.3.2. Uso del protocolo HTTP

Aunque MQTT es ideal para enviar las mediciones desde los sensores, la plataforma web necesita un protocolo diferente para atender las solicitudes realizadas por los usuarios. En este proyecto esa función la cumple HTTP, ya que es el protocolo utilizado por la mayoría de las aplicaciones web.

Mediante HTTP se desarrollan los servicios que permiten iniciar sesión, consultar el historial de mediciones, registrar dispositivos y administrar la información almacenada en la base de datos. De esta manera, MQTT y HTTP cumplen funciones distintas, pero complementarias dentro del sistema.

2.4. Framework de desarrollo

2.4.1. Laravel

El procesamiento de toda la información se desarrolla utilizando Laravel, un framework que facilita la construcción de aplicaciones web organizadas, seguras y fáciles de mantener. Su estructura permite administrar usuarios, gestionar la base de datos, desarrollar servicios web y controlar el acceso a la información de manera eficiente.

Además, Laravel ofrece herramientas que simplifican el desarrollo del sistema y ayudan a proteger la aplicación frente a problemas comunes de seguridad, lo que aporta mayor confiabilidad al funcionamiento de la plataforma.

2.4.2. Vue

Para la interfaz de usuario se seleccionó Vue.js debido a que permite desarrollar aplicaciones web modernas y dinámicas. Su principal ventaja es que actualiza automáticamente la información mostrada en pantalla cuando llegan nuevos datos desde el servidor, evitando recargas constantes de la página.

Esto hace posible que el usuario visualice gráficas, indicadores y registros que cambian de forma inmediata conforme se reciben nuevas mediciones del sensor, facilitando el seguimiento continuo del estado del cultivo.

2.5. Microcontrolador y Sensor de Humedad de Suelo

Los componentes utilizados para la adquisición de información representan una parte fundamental dentro de la arquitectura del sistema, ya que permiten obtener los datos necesarios para realizar el seguimiento de las condiciones del suelo. Para este proyecto se emplean elementos de hardware que trabajan de manera conjunta para captar, procesar y transmitir la información obtenida durante el funcionamiento del prototipo. La integración de estos componentes permite establecer el punto inicial del flujo de datos que posteriormente es procesado y gestionado por los demás elementos de la arquitectura. A continuación, se presentan el microcontrolador y el sensor de humedad utilizado en el desarrollo del proyecto, así como su función dentro de la solución propuesta.

2.5.1. Microcontrolador

El ESP32 fue seleccionado como dispositivo principal para la implementación del prototipo debido a que reúne varias características que lo hacen apropiado para proyectos de Internet de las Cosas. Este microcontrolador incorpora conectividad Wi-Fi, posee suficiente capacidad de procesamiento y permite conectar diferentes tipos de sensores sin necesidad de utilizar hardware adicional.

Además de ser un dispositivo económico, ofrece un buen desempeño y un bajo consumo de energía, aspectos importantes cuando se pretende mantener un sistema funcionando de manera continua. Gracias a estas características, el ESP32 puede recibir las mediciones del sensor de humedad y enviarlas al servidor de forma rápida y confiable.

2.5.2. Sensor de humedad

Para la medición de la humedad del suelo se seleccionó el sensor YL-100 con características anticorrosivas, esto debido a que está diseñado para realizar mediciones directamente sobre el suelo y presenta una mayor resistencia a la corrosión ocasionada por la exposición prolongada a la humedad. Esta característica resulta adecuada para el sistema propuesto, ya que el sensor debe permanecer en contacto con el suelo durante las actividades de monitoreo.

Además, este sensor opera con un voltaje de alimentación de 3,3 voltios y dispone de una salida analógica que permite obtener los datos para su posterior procesamiento en el microcontrolador ESP32. Esto permite una conexión compatible con el nivel de alimentación utilizado por el microcontrolador y facilita la obtención de las lecturas para su posterior procesamiento y transmisión.

Con el desarrollo de las anteriores actividades, fue posible establecer el conjunto de procesos, herramientas y tecnologías necesarias para la implementación de la solución propuesta, y de igual forma, cumplir con el primer objetivo del proyecto orientado a realizar una búsqueda de las tecnologías y procesos disponibles para el desarrollo sistemas de monitorización remota que puedan ser implementados en el sistema propuesto.

3. Diseño de la arquitectura

Esta etapa comprendió el diseño de una arquitectura adecuada para la implementación del sistema telemático propuesto, estableciendo los componentes necesarios que conformarán el sistema y los mecanismos de comunicación requeridos para permitir el intercambio de información entre ellos. Lo anterior con el fin de garantizar un correcto flujo de la información y establecer una base sólida para el desarrollo e implementación de todo el sistema.

Esta fase se desarrolló mediante la ejecución del Sprint número 2 (diseño), más específicamente mediante la actividad 4 (Arquitectura y comunicación) de acuerdo con la metodología seleccionada para el desarrollo del proyecto. En la Tabla 15 se presenta la planeación mediante las tareas a realizar y su asignación a los miembros del equipo.

No.	Actividad	Descripción	Responsable
1	Diagrama general del sistema	Establecer y representar los diferentes componentes del sistema y las relaciones entre ellos.	Michael Jesus Ubaque
2	Diagrama de comunicación	Representar el flujo de la información en el sistema y los métodos de comunicación entre los componentes.	Michael Jesus Ubaque
3	Entorno de implementación	Describir las características de los componentes que integran el sistema.	Cristian Felipe Molina

Tabla 15. Planeación tareas para el diseño de la arquitectura.
Fuente: Elaboración propia

3.1. Diagrama general del sistema

El sistema está estructurado a través de una arquitectura distribuida orientada al Internet de las cosas (IoT) conformada por sensores de humedad, microcontroladores, un servidor en la nube, APIs externas y una plataforma web disponible para los usuarios y accesible desde cualquier lugar.

El siguiente diagrama representa la arquitectura general del sistema, mostrando los componentes principales, su distribución en diferentes capas y los mecanismos de comunicación entre ellos.

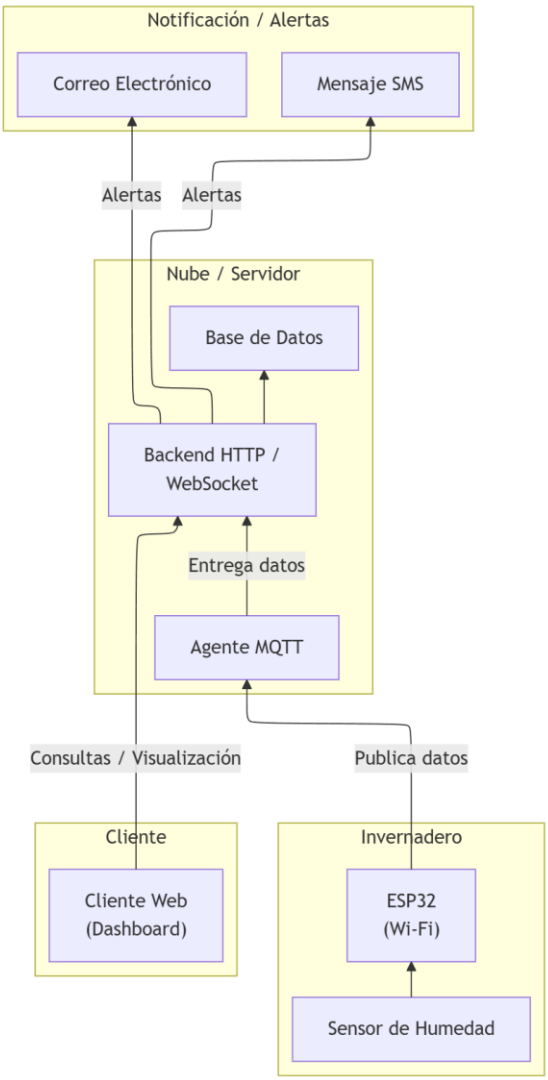


Figura 2. Diagrama general del sistema. Fuente: Elaboración propia

Se emplea una arquitectura dividida en cuatro niveles, se tiene un primer nivel de captura y recolección de los datos de la humedad del suelo, conformado los sensores o prototipos; Un segundo nivel encargado del procesamiento y almacenamiento de los datos, conformado por un servidor central; Un tercer nivel con la que el usuario final interactúa directamente, que corresponde al sitio web de visualización de datos y las alertas generadas que serán notificadas por correo y mensajes SMS. Por último, se tiene un nivel de comunicación que abarca dispositivos de red y protocolos de comunicación.

El primer nivel de captura de datos se conforma por un microcontrolador ESP32, que integra conectividad mediante Wi-Fi para facilitar el envío de los datos a través de internet, un sensor de humedad anticorrosivo, encargado de medir el contenido de agua (volumétrico o potencial) en la tierra, y el software embebido necesario para establecer una conexión directa con el backend, esto último se hará utilizando el protocolo MQTT.

Los datos serán recibidos en un servidor central (segundo nivel), esta capa está compuesta por un agente MQTT, encargado de coordinar la recepción y envío de los datos enviados por los dispositivos IoT (prototipo); Un backend que valida y procesa toda la información

recibida, haciendo uso de los protocolos HTTP y WebSockets para el envío de los datos en tiempo real a los usuarios y, por último, una base de datos relacional donde se almacenan las lecturas y las configuraciones del sistema.

Con la recepción de datos en tiempo real se permitirá la generación de notificaciones, mediante mensajes enviados a los usuarios por correo electrónico y mensajes SMS, que alerten sobre lecturas fuera de rangos preestablecidos y que indiquen que está habiendo un sub-riego o sobre-riego en el cultivo. Los servicios de correo electrónico y mensajes SMS serán contratados externamente y consumidos mediante una API.

El tercer nivel dispondrá de una interfaz web (frontend) accesible para los usuarios finales. A través de esta interfaz, los usuarios podrán monitorizar en tiempo real los niveles de humedad del suelo, visualizar gráficos interactivos, configurar alertas personalizadas según los rangos de humedad deseados, gestionar los sensores y dispositivos conectados, y acceder a reportes históricos para su análisis que ayuden en la toma de decisiones.

Por último, el nivel de comunicación, permite la integración de todos los componentes del sistema, garantizando la transmisión eficiente, confiable y segura de la información. Esto se logra utilizando la red de internet y protocolos diseñados para diferentes propósitos dentro del sistema. Por un lado, se emplea el protocolo MQTT para la comunicación entre los sensores y el agente MQTT, el protocolo HTTP se utiliza para interacción entre el cliente web y el backend y el protocolo WebSocket posibilita la comunicación bidireccional en tiempo real entre el servidor y el cliente, lo que facilita la actualización dinámica de las lecturas sin necesidad de recargar la página.

Todos estos mecanismos de comunicación operan bajo el modelo de referencia TCP/IP, que define cómo se comunican las computadoras en una red o en internet. Gracias a este modelo, el prototipo, el broker MQTT y el sistema web pueden comunicarse de manera interoperable a través de Internet utilizando protocolos como MQTT, HTTP/HTTPS y WebSockets [2].

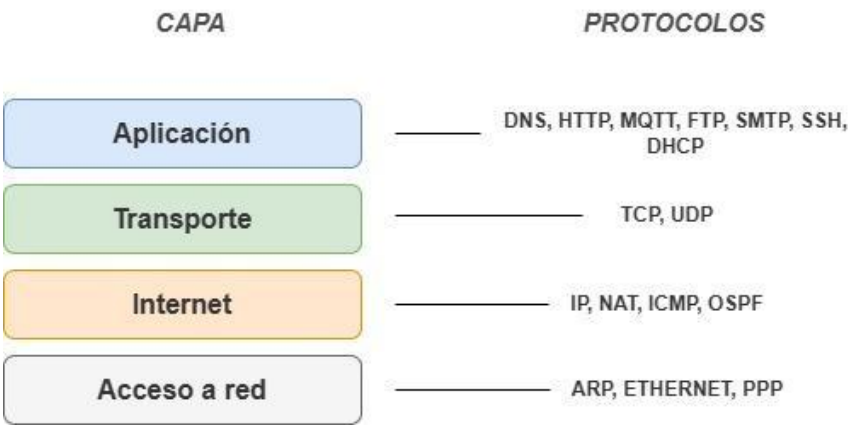


Figura 3. Modelo TCP/IP. Fuente: (Interpolados, 2017)

[2] Cisco Systems. *Internetworking Basics: TCP/IP*. Disponible en: https://www.cisco.com/E-Learning/bulk/public/tac/cim/cib/using_cisco_ios_software/linked/tcpip.htm.

La arquitectura distribuida presentada es utilizada debido a su capacidad para separar responsabilidades funcionales y permitir la operación independiente y desacoplada de cada componente del sistema. Esto permite garantizar requisitos fundamentales de seguridad, confiabilidad, rendimiento, escalabilidad e interoperabilidad. Además, es una arquitectura adecuada y ampliamente utilizada en proyectos que involucren entornos del internet de las cosas (IoT).

3.2. Diagrama de comunicación

El diagrama de comunicación muestra la manera en que interactúan los diferentes componentes del sistema a través de la red, utilizando diferentes protocolos y capas del modelo TCP / IP. Algunos de los protocolos de la capa de aplicación utilizados son HTTP (Protocolo de Transferencia de Hipertexto), MQTT (Message Queuing Telemetry Transport), WebSocket y se emplean protocolos de mensajería como SMTP (Protocolo Simple de Transferencia de Correo) para el envío de notificaciones por correo electrónico, sin embargo, este último no es implementado directamente en el sistema, sino que es gestionado por servicios externos encargados del envío de correos, a los cuales se accede mediante una API REST. De igual manera, se utilizan servicios externos para el envío de mensajes SMS, mediante la integración de otra API REST.

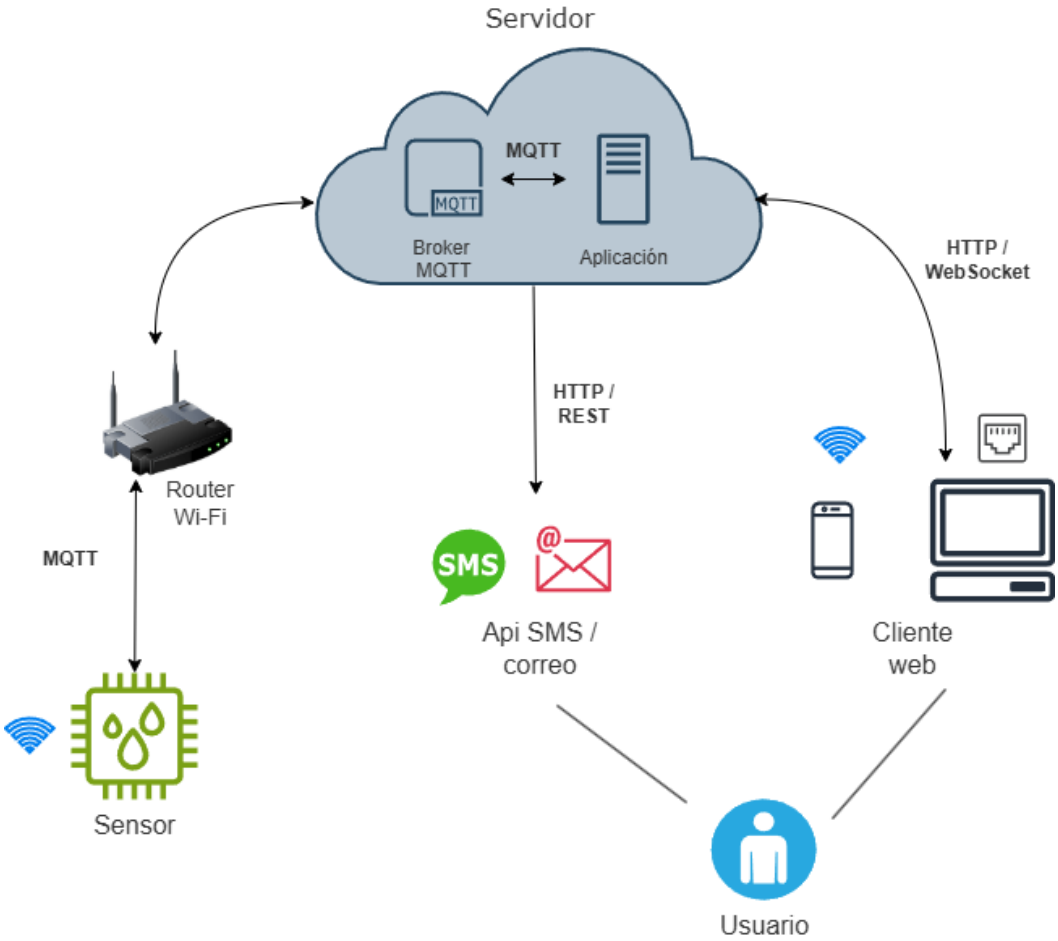


Figura 4. Diagrama de comunicación del sistema.
Fuente: Elaboración propia

3.3. Entorno de implementación

3.3.1. Servidor

Para la implementación integral del sistema, fue necesario definir las capacidades técnicas y configuraciones con las que debe contar el servidor central, el cual será el responsable de soportar todas las funcionalidades del sistema de monitorización y la generación de alertas. Dicho servidor es responsable de la recepción de datos provenientes de los dispositivos IoT (sensores), el procesamiento de la información, el almacenamiento en la base de datos, la gestión de alertas y la atención de solicitudes realizadas por los clientes web.

Se utilizó un servidor virtual privado (VPS), ya que corresponde a criterios de eficiencia y de costo-beneficio, este tipo de servidor permite acceder a recursos de cómputo decentes a un bajo costo, permitiendo garantizar el cumplimiento de los requerimientos del sistema y los objetivos del proyecto.

Por lo anterior se establecieron los siguientes parámetros mínimos con las que debe contar el servidor VPS:

Componente	**Requisito mínimo**	**Descripción**
CPU	2 vCPU	Procesamiento suficiente para soportar las cargas de las peticiones HTTP de los usuarios, la gestión del broker MQTT, la base de datos y los demás servicios derivados de la operación del sistema en el servidor.
RAM	2 GB	Para permitir la ejecución simultánea de todos los servicios requeridos.
Almacenamiento	30 GB SSD	Espacio de almacenamiento para guardar todos los archivos del programa, bases de datos y la información proveniente de usuarios y sensores.
Sistema Operativo	Linux ubuntu 22.04	Sistema operativo estable, con bajo consumo de recursos y compatible con la mayoría de módulos y bibliotecas.
Comunicación	IP pública	Necesaria para permitir la comunicación remota con dispositivos IoT y clientes web a través de internet.
Ancho de banda	100 MB	Velocidad de internet suficiente para garantizar transmisión eficiente de datos.
Accesibilidad	Acceso por SSH	Acceso remoto seguro al servidor para la configuración, administración y despliegue de la solución.
Seguridad	Firewall y SSH	El servidor debe contar con protección contra accesos no autorizados.
Disponibilidad	> 99%	El proveedor de servicio debe asegurar una disponibilidad superior al 99%.

Tabla 16. Requisitos mínimos del servidor. Fuente: Elaboración propia

3.3.2. Microcontrolador

Para la implementación del prototipo es necesario establecer los requisitos del microcontrolador, el cual es el responsable de procesar las señales provenientes del sensor de humedad y transmitir periódicamente las lecturas al servidor central. Esta comunicación se realizará mediante el protocolo MQTT, utilizando un agente o broker MQTT instalado en el servidor como intermediario para la recepción de los datos.

En la Tabla 17, se detallan los requisitos que debe cumplir el microcontrolador para garantizar la correcta funcionalidad del prototipo.

Componente	Requisito	Descripción
Microcontrolador	ESP32	Dispositivo (placa programable) con Wi-Fi integrado, óptimo para su implementación en entornos IoT
Frecuencia de CPU	240 MHz	Capacidad para el procesamiento de las lecturas del sensor y la transmisión de datos.
Memoria RAM	520 KB SRAM	Permite ejecutar tareas de lectura de sensores y comunicación MQTT simultáneamente.
Conectividad	Wi-Fi 802.11	Conectividad necesaria para la comunicación con el servidor.
Voltaje de operación	3.3 V	Consumo estándar para la mayoría de sensores de humedad.
Consumo energético	Bajo	Adecuado para aplicaciones embebidas o alimentadas por fuentes limitadas.
Periféricos	15 pines	Para las conexiones necesarias con los demás elementos físicos del prototipo.
Protocolos	MQTT sobre TCP/IP	Debe permitir establecer conexiones mediante el protocolo MQTT, haciendo uso de librerías.

Tabla 17. Requisitos del microcontrolador. Fuente: Elaboración propia

3.3.3. Comunicación

Para el funcionamiento integral del sistema y la comunicación entre todos sus componentes de manera eficiente y confiable, es necesario establecer los requisitos de red que garanticen la conectividad entre el microcontrolador, el servidor, las APIs externas y el cliente web. Estos requisitos contemplan el uso de Internet como medio de interconexión y la implementación de protocolos como MQTT, HTTP y WebSocket, operando bajo el modelo TCP/IP.

En la Tabla 18 se detallan los requisitos de comunicación necesarios para el sistema de monitoreo.

Componente	Requisito	Descripción
Conectividad a internet	Conexión a internet estable y permanente	Permite a los usuarios acceder el sitio web del sistema.
	Punto de acceso Wi - Fi en invernaderos	Necesario para conectar el o los sensores (ESP32) al servidor mediante la red de internet.
Protocolo IoT	MQTT	Permite una comunicación ligera y eficiente bajo el modelo publicador-suscriptor, entre los sensores y el servidor
Protocolo web	HTTP	Permite la comunicación bajo el modelo cliente-servidor, entre el cliente web y el backend.
Comunicación en tiempo real	WebSocket	Fundamental para la actualización en tiempo real de los datos y gráficos de las lecturas de humedad del suelo, en el sistema web por parte de los usuarios finales.
API de correo	Conexión a servicio externo de email	Permite el envío automático de alertas por correo electrónico cuando se detectan valores fuera de rango.
API de SMS	Conexión a servicio externo de mensajería SMS	Permite el envío de notificaciones críticas a través de mensajes de texto.
Disponibilidad	> 99%	Asegura continuidad en la monitorización y generación de alertas.

Tabla 18. Requisitos de conectividad. Fuente: Elaboración propia.

3.3.4. Software

El procesamiento y la gestión de los datos se realizan mediante distintos componentes de software, los cuales deben ser adecuados para establecer comunicación con los diferentes elementos del sistema y garantizar un flujo de información adecuado. Asimismo, se requieren herramientas que faciliten las tareas de desarrollo, configuración e implementación de la solución, asegurando un entorno de trabajo organizado y funcional.

En la Tabla 19, se detallan los componentes de software necesarios para el proyecto.

Componente	Requisito	Descripción
Base de datos	Sistema de gestión de bases de datos relacionales (SQL)	Permite almacenar y gestionar los datos de las lecturas de humedad, usuarios y todas las demás entidades del sistema.
broker MQTT	Software de mensajería	Gestiona la recepción de datos enviados por el microcontrolador.
Servidor web	Software de servidor web (ej. Apache o Nginx)	Se requiere un servidor web que permita gestionar las solicitudes HTTP/HTTPS provenientes del cliente, servir la aplicación web y actuar como intermediario entre el frontend y el backend.
Backend	Framework para el desarrollo backend	Necesario para implementar la lógica del sistema, gestionar la información proveniente de los sensores, exponer servicios mediante API REST, integrar el broker MQTT y administrar la base de datos.
Frontend	Framework para el desarrollo frontend	Proporciona visualización de datos, configuración de parámetros y monitoreo en tiempo real.
Librerías	Software cliente MQTT para el backend y el microcontrolador.	Permite establecer comunicación entre el broker MQTT, el servidor backend y los sensores.
Entorno de desarrollo	Entorno para el desarrollo de software (ej. Visual Studio Code, Arduino IDE)	Facilita la programación, depuración y pruebas del sistema.
Control de versiones	Sistema para el control de versiones	Necesario para gestionar cambios en el código y mantener trazabilidad del desarrollo.
Seguridad	Autenticación, cifrado	Todos los elementos de software deben contar con mecanismos de autenticación para el acceso a la información y cifrado cuando sea necesario.

Tabla 19. requisitos de software. Fuente: Elaboración propia

Con la ejecución de las tareas establecidas, fue posible definir y representar una arquitectura acorde a las necesidades del sistema telemático planteado, estableciendo los diferentes componentes y métodos de comunicación entre ellos. De esta manera, se da cumplimiento al objetivo establecido para el diseño de la arquitectura, proporcionando una base estructural para el desarrollo del sistema.

4. Desarrollo del prototipo para la medición de la humedad del suelo

A continuación, se presenta el diseño e implementación del prototipo funcional para la medición de la humedad del suelo, que posee la capacidad de establecer comunicación remota para la transmisión de las lecturas obtenidas a un servidor central.

Siguiendo la metodología Scrum, se implementará el Sprint número 3 (Desarrollo del prototipo para la medición de la humedad del suelo) de la metodología adoptada para el proyecto. A continuación, se muestra la planeación de las actividades que se realizaron.

No.	Actividad	Descripción	Responsable
1	Diseño	Definir los componentes y la estructura del circuito del prototipo.	Cristian Felipe Molina
2	Diagrama de flujo	Representar mediante un diagrama de flujo el funcionamiento y procesamiento de la información.	Michael Jesus Ubaque
3	Desarrollo	Implementar el software que estará embebido (firmware) en el prototipo.	Cristian Felipe Molina
4	Montaje e Implementación	Realizar la integración, configuración y puesta en funcionamiento del prototipo.	Michael Jesus Ubaque

Tabla 20. Planificación de actividades del Sprint 3. Fuente: Elaboración propia

4.1. Diseño

Se diseñó un prototipo funcional capaz de medir la humedad del suelo mediante un sensor de humedad, además de establecer una conexión a internet a través de una red Wi-Fi, permitiendo la transmisión de las lecturas al servidor central para su procesamiento y almacenamiento.

Para esto, se utilizó el sensor de humedad de suelo anticorrosivo YL-100 y el microcontrolador ESP32 DevKit V1, el cual incorpora un módulo de comunicación Wi-Fi compatible con el estándar IEEE 802.11, lo que permite transmitir las lecturas de humedad al sistema backend mediante el protocolo MQTT, facilitando el monitoreo remoto y en tiempo real de las condiciones del suelo.

4.1.1. Esquema

El sensor de humedad de suelo YL-100 se conecta a un módulo adaptador basado en el circuito integrado LM393, el cual acondiciona la señal generada por el sensor y permite obtener tanto una salida analógica como una salida digital. La conexión entre el sensor y el módulo se realiza mediante los cables suministrados por el fabricante, formando un único conjunto de medición.

Por otra parte, la comunicación entre el módulo adaptador y el microcontrolador se establece a través de los pines de alimentación (3.3 V y GND) y de los pines de entrada analógica (GPIO). En este proyecto se emplea la salida analógica del módulo, ya que proporciona una lectura proporcional del nivel de humedad del suelo, permitiendo realizar mediciones con mayor precisión.

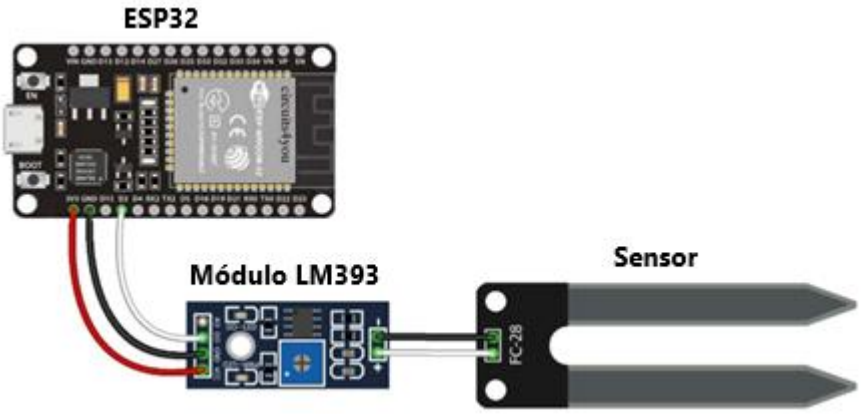


Figura 5. Esquema de conexión de referencia. Fuente: (Cytron Technologies, 2019)

La conexión entre el sensor de humedad de suelo y el microcontrolador ESP32 se realiza mediante cables Dupont, conectando los pines VCC, GND y AO del módulo adaptador LM393 a los respectivos pines de alimentación, tierra y entrada analógica del ESP32, según la siguiente tabla.

Pin Microcontrolador	**Pin Sensor (módulo adaptador)**	**Descripción**
GND	GND	Tierra
3V3	VCC	Alimentación 3.3 voltios
GPIO34	AO	Salida analógica de lecturas

Tabla 21. Conexión entre el ESP32 y el sensor de humedad del suelo YL-100. Fuente: Elaboración propia

La alimentación eléctrica del prototipo se realiza mediante la conexión USB integrada en el microcontrolador ESP32, permitiendo disponer de una fuente de energía continua dependiente solo del suministro eléctrico del invernadero. Esta alternativa evita la utilización de baterías desechables o recargables que requieren mantenimiento y reemplazo periódico, garantizando así un funcionamiento más estable y prolongado del sistema. La alimentación por USB permite utilizar un cable conectado a un cargador convencional o a una extensión eléctrica, facilitando la ubicación del sensor en zonas alejadas de la toma de corriente principal dentro del invernadero.

4.1.2. Diagrama de flujo del prototipo

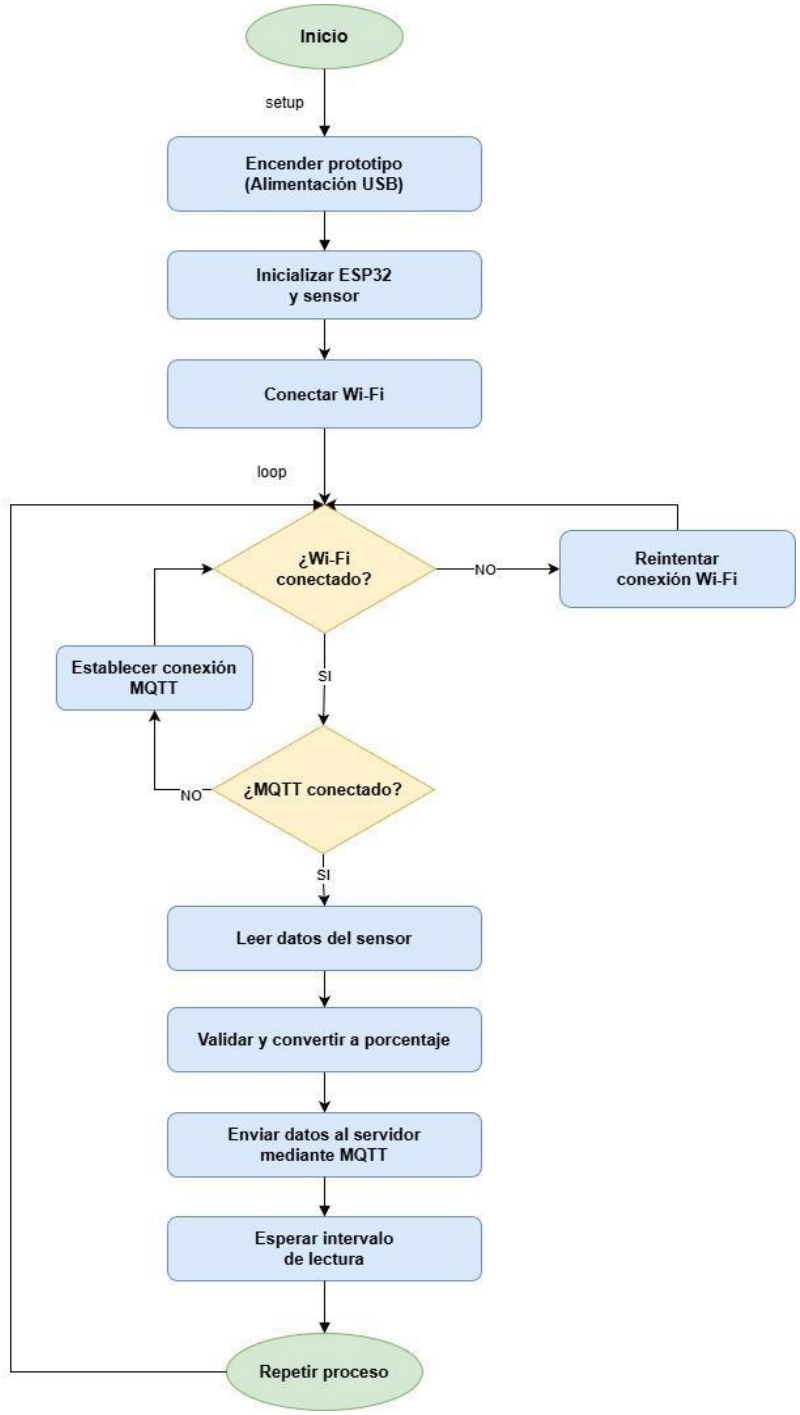


Figura 6. Diagrama de flujo del prototipo. Fuente: Elaboración propia

El firmware o programación integrada en el prototipo, permite establecer una conexión Wi-Fi automática mediante la configuración previa del SSID (nombre público de la red) y la contraseña establecida en el código. Posteriormente, el prototipo realiza de manera iterativa la lectura de los valores de la humedad del suelo, válida los datos obtenidos y establece una comunicación mediante el protocolo MQTT para el envío de los datos al servidor.

Una vez enviada la información al servidor, el prototipo permanece en espera hasta el siguiente intervalo de lectura para repetir el proceso. Adicionalmente, durante cada ciclo el prototipo verifica el estado de la conexión Wi-Fi y MQTT, en caso de fallas en la conexión, el sistema intenta restablecer la conexión automáticamente.

4.2. Desarrollo

Para el desarrollo o programación del prototipo, se utilizó el entorno de desarrollo Arduino IDE y el lenguaje de programación C++. Para la implementación de las funcionalidades de conexión Wi-Fi e intercambio de mensajes mediante el protocolo MQTT, se emplearon las librerías WiFi, PubSubClient y ArduinoJson.

La configuración de los parámetros que permiten la funcionalidad de los diferentes componentes, como las credenciales de la red Wi-Fi, las credenciales y host del broker MQTT y el pin utilizado para las lecturas del sensor, se establecen utilizando constantes que se inicializan al principio del programa. Además, para mayor organización y legibilidad del código, se desarrollaron funciones para diferentes necesidades como se detalla en las tablas 22 y 23.

Tipo	**Nombre**	**Descripción**
constante	ssid	Nombre público de la red Wi-Fi.
constante	password	Contraseña de la red Wi-Fi
constante	mqtt_server	Dirección IP del servidor MQTT (broker).
constante	mqtt_port	Puerto de conexión del servidor MQTT.
constante	mqtt_user	Usuario MQTT establecido en el servidor.
constante	mqtt_password	Contraseña del usuario MQTT.
constante	mqtt_topic	Topic o “tema” del broker al que se suscribe el prototipo.
constante	sensorPin	Pin del ESP32 de donde se tomarán las lecturas analógicas de la humedad del suelo.
constante	sensor_id	Identificador único del sensor o prototipo.
constante	intervalo	Intervalo de lectura.

Tabla 22. Parámetros de configuración implementados en el código del prototipo. Fuente: Elaboración propia

Tipo	Nombre	Descripción
función	conectarWiFi	Establecer y/o reintentar conexión a la red Wi-Fi.
función	conectarMQTT	Establecer y/o reintentar conexión al broker MQTT.
función	leerSensor	Leer los datos del sensor (lectura analogica)
función	convertirHumedad	Convertir los valores de la lectura en porcentaje.
función	enviarLectura	Enviar los datos al servidor mediante un mensaje en formato JSON.

Tabla 23. Funciones implementadas en el código del prototipo.
Fuente: Elaboración propia

El seguimiento y control de errores durante la ejecución del programa cargado al prototipo, se realizan mediante el monitor serial del entorno de desarrollo Arduino IDE. Esta herramienta permite visualizar en tiempo real mensajes de depuración relacionados con el estado de la conexión Wi-Fi, la conexión al broker MQTT, las lecturas del sensor de humedad del suelo y el envío de datos.

4.3. Montaje e implementación

En esta fase, se integraron los diferentes componentes del prototipo y se puso en funcionamiento mediante el encendido por primera vez, con el fin de validar su comportamiento en situaciones controladas y simuladas de un invernadero, representadas en una maceta con tierra humedecida. Además, se realiza el cargue del código desarrollado.

A continuación, se presenta la implementación del prototipo con las conexiones entre los diferentes componentes realizada mediante cables Dupont. En la parte derecha de la Figura 7 se evidencia el microcontrolador ESP32, montado sobre una placa de expansión para facilitar la conexión de los cables y la manipulación del dispositivo. En el medio se encuentra el módulo adaptador LM393, encargado de entregar la lectura mediante una señal analógica y a la izquierda el sensor de humedad del suelo, responsable de realizar las mediciones del nivel de humedad del terreno:

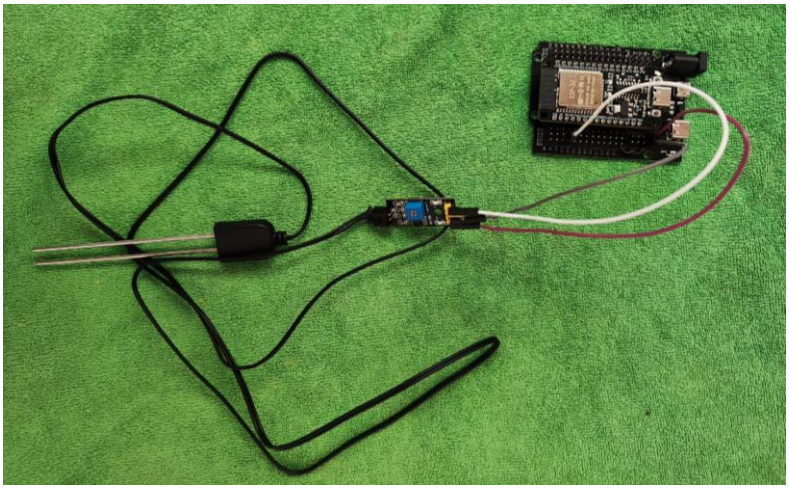


Figura 7. Montaje e implementación del prototipo.
Fuente: Elaboración propia

Una vez energizado el prototipo mediante un cable USB conectado directamente al computador, y cargado el código al microcontrolador, se hace una primera verificación mediante la validación de los indicadores LED encendidos. Los LED rojos del prototipo indican que existe alimentación de energía en el sistema, mientras el LED azul encendido indica que el prototipo estableció conexión correctamente a la red Wi-Fi y al broker MQTT.

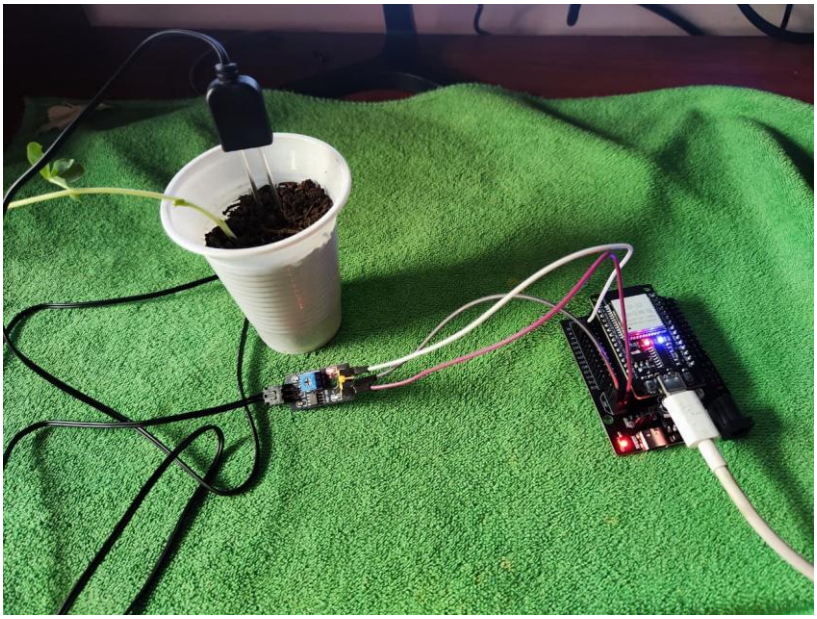


Figura 8. Encendido del prototipo. Fuente: Elaboración propia

Adicionalmente, se puede realizar una primera verificación del funcionamiento del prototipo revisando el monitor serial del entorno de desarrollo Arduino IDE, el cual permite visualizar el estado del prototipo en tiempo real y los mensajes salientes enviados desde el microcontrolador hacia el servidor.

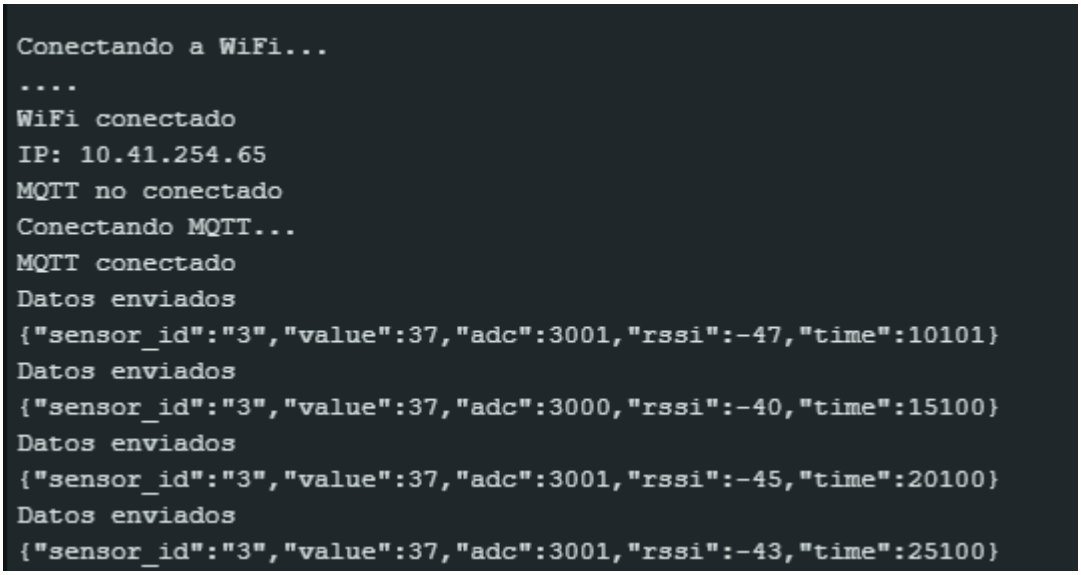


Figura 9. Salida monitor serial prototipo. Fuente: Elaboración propia

La Figura 9 muestra el correcto funcionamiento del prototipo desarrollado, en ella se observa que el dispositivo es capaz de establecer de forma autónoma conexión a una red Wi-Fi y al broker MQTT, así como de restablecer la comunicación automáticamente ante eventuales fallas de conexión.

Además, se evidencia la capacidad del prototipo de capturar las lecturas de humedad del suelo, convertir los datos a porcentaje e incluir el valor en un mensaje en formato JSON, incluyendo metadata, como el valor original capturado, el identificador único del prototipo y el tiempo activo.

Gracias al desarrollo de esta fase, se logró implementar un prototipo funcional, capaz de medir la humedad del suelo y transmitir las lecturas a un broker MQTT a través de internet, en un entorno simulado de un invernadero. De esta manera se da cumplimiento al objetivo de “desarrollar un prototipo capaz de medir la humedad del suelo en cultivos dentro de invernaderos y enviar los datos recopilados a un servicio web a través de internet”.

5. Desarrollo del sistema web

Se desarrolló e implementó un sistema web, conformado por un backend y un frontend capaz de recibir, procesar, almacenar y representar la información de las lecturas de humedad del suelo dentro de invernaderos a los usuarios finales. Para ello, se tomarán como referencia las principales fases del ciclo de vida del desarrollo de software, entre ellas el análisis, diseño, desarrollo, implementación y pruebas.

Para implementar este componente del sistema telemático, se implementaron los Sprint número 4 y 5, denominados “desarrollo del backend” y “desarrollo del frontend” respectivamente, de la metodología Scrum planteada para el proyecto.

No.	Actividad	Descripción	Responsable
1	Implementación de base de datos	Instalación y configuración de la base de datos según la estructura diseñada.	Michel Jesus ubaque
2	Creación de servicios web	Desarrollo de los servicios del backend.	Michael Jesus Ubaque y Cristian Felipe Molina
3	Instalación del broker MQTT	Instalación y configuración del broker MQTT para permitir conexión con el prototipo.	Michel Jesus ubaque
4	Pruebas	Verificar el correcto funcionamiento de los servicios.	Cristian Felipe Molina

Tabla 24. Planificación de actividades del Sprint 4.
Fuente: Elaboración propia

No.	Actividad	Descripción	Responsable
1	Desarrollo Interfaz web	Desarrollo de las interfaces web de los diferentes módulos.	Michel Jesus ubaque
2	Representación gráfica de datos	Representación de los datos de las lecturas en gráficas y tablas.	Michael Jesus Ubaque y Cristian Felipe Molina
3	Configuración de hosting y servidor	Preparación del entorno de implementación y despliegue del sistema web.	Michael Jesus Ubaque y Cristian Felipe Molina
4	Pruebas	Comprobación de la funcionalidad y correcto funcionamiento de los módulos del sistema web.	Cristian Felipe Molina

Tabla 25. Planificación de actividades del Sprint 5.
Fuente: Elaboración propia

5.1. Análisis

5.1.1. Requerimientos

Teniendo en cuenta las funcionalidades que debe tener el sistema en conjunto, se describen los siguientes requerimientos funcionales y no funcionales para cumplir con las expectativas de cómo debe funcionar y bajo qué condiciones.

5.1.1.1. Requerimientos funcionales

Código	Descripción	Prioridad
RF 01	La aplicación permitirá crear y administrar usuarios en el sistema.	Alta
RF 02	Se debe permitir la autenticación de usuarios mediante credenciales válidas.	Alta
RF 04	La aplicación debe permitir crear y administrar invernaderos, incluyendo los sensores de humedad para cada invernadero.	Alta
RF 05	El sistema debe permitir asignar usuarios a cada invernadero, asegurando que cada usuario solo pueda ver la información asociada a él.	Media
RF 06	El sistema debe recibir y almacenar las lecturas enviadas por los sensores.	Alta
RF 07	El sistema deberá generar alertas automáticas a los usuarios cuando los valores de los sensores sobrepasen los valores configurados.	Alta
RF 08	Se debe permitir configurar parámetros como correos de notificación y parámetros mínimos y máximos para la generación de alertas.	Media
RF 09	La aplicación debe permitir la visualización de los datos, tanto el tiempo real como el historial de lecturas.	Alta

Tabla 26. Requerimientos funcionales.
Fuente: Elaboración propia

5.1.1.2. Requerimientos no funcionales

Código	Descripción	Prioridad
RFN 01	Se debe permitir el acceso solo a usuarios autenticados.	Alta
RFN 02	El sistema debe ser accesible desde cualquier navegador moderno.	Alta
RFN 03	El sistema debe tener una interfaz web intuitiva y fácil de usar.	Media
RFN 04	El sistema debe tener la capacidad de manejar múltiples usuarios de manera simultánea.	Alta
RFN 05	Se debe garantizar la escalabilidad del sistema permitiendo crear y soportar más usuarios y sensores.	Alta
RFN 06	El sistema debe estar disponible de manera continua.	Alta

Tabla 27. Requerimientos no funcionales.
Fuente: Elaboración propia

5.1.2. Actores del sistema

Son los diferentes tipos de usuarios que podrán interactuar con el sistema, cada uno con responsabilidades distintas, según el rol que cumplen dentro del proceso de implementación, gestión y monitorización de los sensores.

Actor	Descripción
Administrador	Usuario con acceso total al sistema web, es el responsable de la gestión de los invernaderos y sensores, gestión de los demás usuarios del sistema y configuración general del sistema.
Técnico	Encargado de la instalación, configuración y el mantenimiento de los sensores o prototipos, responsable de asegurar una adecuada conexión y comunicación de los sensores con el servidor.
Usuario	Usuario final del sistema, son todos aquellos posibles clientes que utilizaran el sistema. Este tipo de usuario cuenta con acceso a la visualización de los datos recopilados por los sensores en los diferentes invernadero y cultivos, además serán quienes reciban las alertas generadas automáticamente.

Tabla 28. Actores del sistema.
Fuente Elaboración propia

5.1.3. Casos de uso

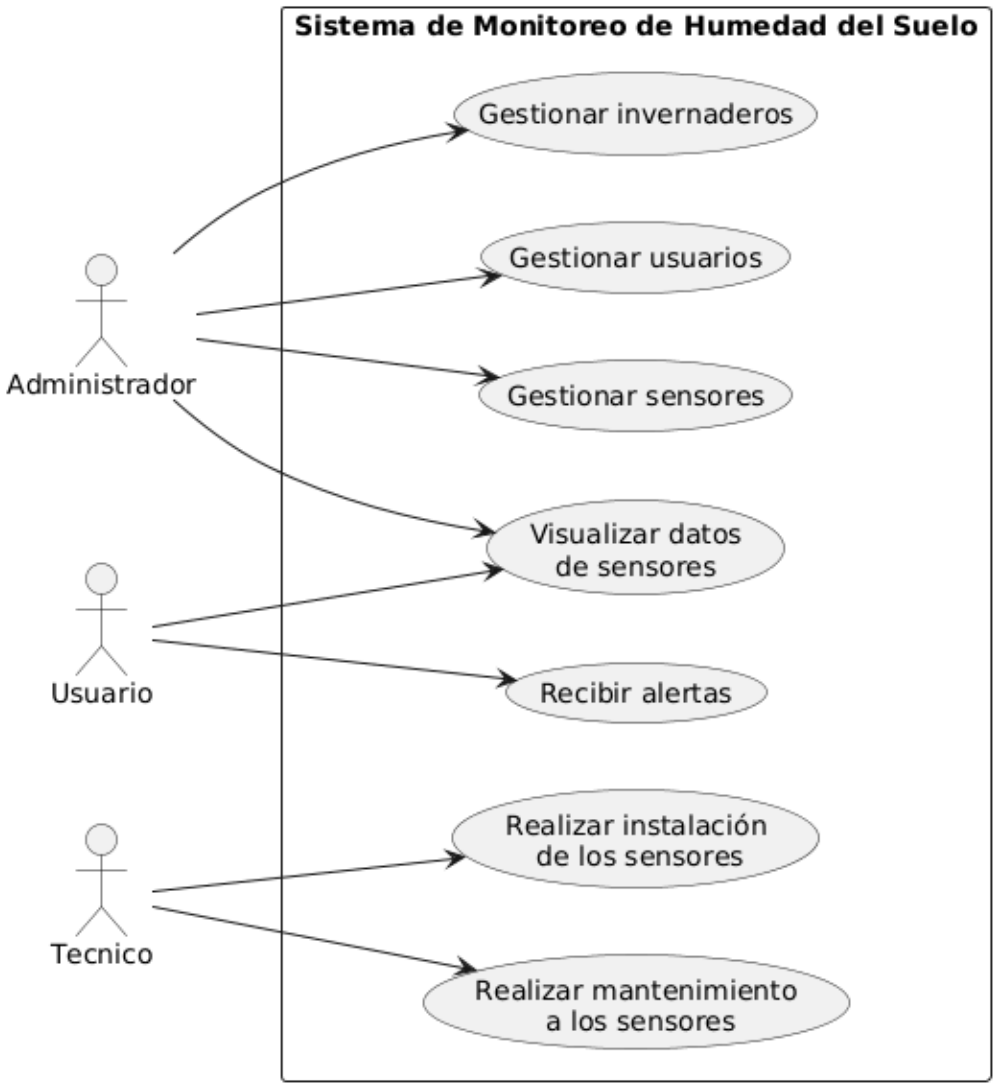


Figura 10. Diagrama de casos de uso.
Fuente: Elaboración propia

5.2. Diseño

5.2.1. Base de datos

Para el almacenamiento y gestión de la información de manera estructurada y organizada se optó por utilizar un modelo relacional de base de datos (SQL), lo que permite organizar la información en tablas relacionadas y una estructura fija.

5.2.1.1. Tablas y tipos de datos (Diccionario de datos)

Con el fin de representar la solución propuesta y dar cumplimiento a los requerimientos de la aplicación, se identificaron las siguientes entidades que conforman el modelo de datos del sistema:

- Usuarios: Usuarios que interactúan con el sistema.
- Invernaderos o cultivos: Cultivos o invernaderos que serán monitoreados.
- Sensores: Información de los sensores instalados en cada cultivo.
- Lecturas: Lecturas generadas por los sensores a lo largo del tiempo.
- Tipos sensor: Tipos de sensores disponibles y sus unidades de medición.
- Alertas: Alertas generadas cuando los valores de los sensores superan los límites establecidos.

A continuación, se describen cada una de las tablas necesarias, sus atributos y los tipos de datos asignados.

Tabla usuarios (users)		
Campo	**Tipo**	**Descripción**
id	bigint	Identificador del usuario
name	varchar(255)	Nombres del usuario
email	varchar(255)	Correo electrónico del usuario
phone_number	varchar(20)	Número celular del usuario
role	varchar(255)	Nombre del rol del usuario (“admin” o “customer”)
email_verified_at	time	Fecha de verificación de correo
password	varchar(100)	Contraseña hash del usuario
remember_token	varchar(100)	Token recordar datos de ingreso
created_at	datetime	Fecha de creación del usuario
updated_at	datetime	Fecha de la última actualización del usuario

Tabla 29. Tabla usuarios de la base de datos.
Fuente: Elaboración propia

Tabla cultivos (crops)		
Campo	**Tipo**	**Descripción**
id	bigint	Identificador del cultivo
name	varchar(255)	Nombre de cultivo
location	varchar(255)	Ubicación del cultivo
description	text	Descripción del cultivo
is_active	boolean	Estado (activo o inactivo) del cultivo en el sistema
created_at	datetime	Fecha de creación del registro
updated_at	datetime	Última modificación del registro

Tabla 30. Tabla cultivos de la base de datos. Fuente: Elaboración propia

Tabla tipos sensor (sensor_types)		
Campo	**Tipo**	**Descripción**
id	bigint	Identificador del tipo de sensor
name	varchar(50)	Nombre del tipo de sensor
unit	varchar(20)	Unidad de medida
symbol	varchar(5)	Símbolo de la medida
created_at	datetime	Fecha de creación del registro
updated_at	datetime	Última modificación del registro

Tabla 31. Tabla tipos sensor de la base de datos.
Fuente: Elaboración propia

Tabla sensores (sensors)		
Campo	**Tipo**	**Descripción**
id	bigint	Identificador del sensor
crop_id	bigint	Cultivo asociado al sensor
type_id	bigint	Tipo de sensor asociado
name	varchar(100)	Nombre del sensor
model	varchar(100)	Modelo del sensor
unit	varchar(20)	Unidad de medida del sensor
status	enum	Estado del sensor (activo, inactivo o en mantenimiento)
ralert_level	enum	Nivel de alerta del sensor, indica si el sensor está en modo alerta
reading_interval	int	Intervalo de toma de lecturas
min_value	decimal	Valor mínimo de lectura para generar alertas
max_value	decimal	Valor máximo de lectura para generar alertas
created_at	datetime	Fecha de creación del registro
updated_at	datetime	Última fecha de actualización del registro

Tabla 32. Tabla sensores de la base de datos.
Fuente: Elaboración propia

Tabla lecturas (readings)		
Campo	**Tipo**	**Descripción**
id	bigint	Identificador de la lectura
sensor_id	bigint	Identificador del sensor asociado
value	decimal	Valor de la lectura
recorded_at	datetime	Fecha y hora de toma de la lectura

Tabla 33. Tabla lecturas. Fuente: Elaboración propia

Tabla alertas (alerts)		
Campo	**Tipo**	**Descripción**
id	bigint	Identificador de alerta
sensor_id	bigint	Sensor que genera la alerta
value	decimal	Valor capturado que generó la alerta
type	enum	Tipo de alerta
message	text	Mensaje generado
triggered_at	datetime	Fecha de generación de la alerta
resolved_at	datetime	Fecha visualización de la alerta

Tabla 34. Tabla de alertas generadas de la base de datos.
Fuente: Elaboración propia

5.2.1.2. Diagrama Entidad - Relación (ER)

El diagrama entidad - relación permite identificar las entidades (tablas), sus atributos y las relaciones que deben existir en la base de datos para representar correctamente la información que se requiere almacenar de manera estructurada y sin redundancias. A continuación, se presenta el diagrama.

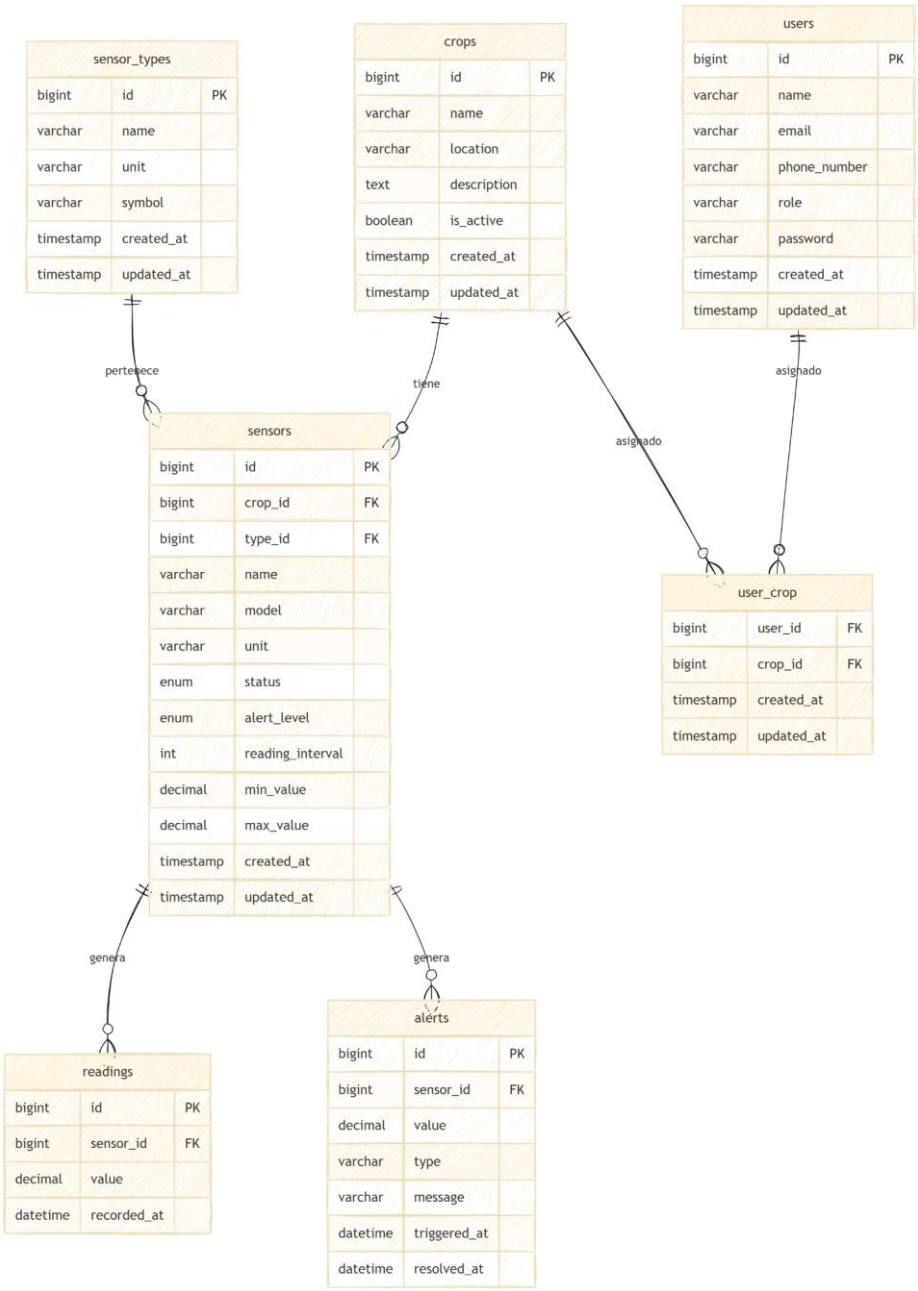


Figura 11. Diagrama entidad-relación. Fuente: Elaboración propia

5.2.1.3. Relaciones entre tablas

Las relaciones permiten establecer cómo interactúan y se enlazan las diferentes entidades del sistema. A continuación, se detallan las relaciones que se requieren implementar:

- Un cultivo puede tener varios sensores.
- Un tipo de sensor puede estar asociado a múltiples sensores.
- Un sensor puede generar múltiples lecturas.
- Un sensor puede generar múltiples alertas.
- Un usuario puede estar asignado a varios cultivos, y un cultivo puede tener varios usuarios asociados, lo cual se gestiona mediante la tabla intermedia “user_crop”.

5.2.1.4. Sistema de Gestión de Bases de Datos

El sistema de gestión de base de datos (SGBD) seleccionado para el desarrollo del sistema es MariaDB, el cual es un sistema de código abierto, ampliamente utilizado para el desarrollo web, ofrece un nivel alto de confiabilidad, rendimiento y compatibilidad con sistemas desarrollados con PHP. Esta elección es adecuada para el almacenamiento de las lecturas generadas por los sensores, la gestión de usuarios, cultivos y alertas generadas por el sistema de monitorización.

5.2.2. Backend

El backend está compuesto por tres elementos fundamentales: la base de datos, la aplicación desarrollada en Laravel y el broker MQTT. Estos tres elementos interactúan de manera coordinada para permitir un correcto flujo de información y garantizar el correcto funcionamiento de todo el sistema. El backend constituye el componente central del sistema, encargado de gestionar y procesar la información de las lecturas de humedad del suelo, así como de ponerla a disposición del usuario final a través de la aplicación, este componente permite integrar los diferentes elementos del sistema. Por su parte, el broker MQTT, se encarga de gestionar la comunicación directa con los dispositivos instalados en el invernadero (prototipo), siendo un punto intermedio entre los sensores y la aplicación desarrollada en Laravel.

Estos tres elementos serán alojados en un mismo servidor (VPS) para reducir gastos y así cumplir con el presupuesto destinado para el desarrollo del proyecto.

A continuación, se ilustra mediante un diagrama los diferentes componentes del backend y cómo interactúan entre sí.

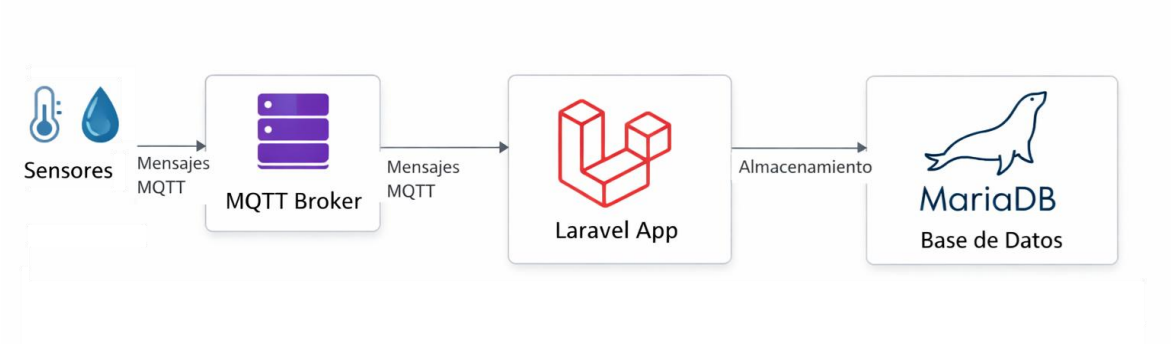


Figura 12. Diagrama general del backend. Fuente: Elaboración propia

5.2.2.1. Broker MQTT

El broker MQTT es un software que se encarga de gestionar la comunicación con los sensores y de transmitir la información hacia la aplicación Laravel.

Para poder establecer dicha comunicación, primero se deben tener en cuenta mecanismos de seguridad como la autenticación, con el fin de cumplir criterios de confidencialidad y seguridad de la información. Para esto, el broker se configura para permitir la autenticación mediante el uso de credenciales (usuario y contraseña), tanto los sensores, como la aplicación Laravel, deben autenticarse para comunicarse con el broker.

Los datos enviados por los sensores son transportados mediante mensajes MQTT utilizando un formato JSON directamente hacia el broker MQTT. Para que esto sea posible, los dispositivos sensores publican la información en un topic específico en el broker, para el desarrollo de este proyecto, el topic en el que el o los prototipos deberán publicar los datos es:

crop/sensor/humidity/

Mediante este modelo se asegura que toda la información llegue a un punto específico del sistema, garantizando que haya consistencia en la configuración de todos los dispositivos instalados en diferentes ubicaciones o invernaderos.

Por su parte, la aplicación Laravel también deberá establecer comunicación con el broker en un punto o topic específico, el cual debe ser el mismo en el que los dispositivos publican los datos:

crop/sensor/humidity/

En esta ocasión la aplicación Laravel tiene el papel de suscriptor, esto asegura que cada vez que el broker recibe información desde los sensores la retrasmita inmediatamente al suscriptor.

El broker MQTT debe ser instalado en el servidor y debe estar ejecutándose permanentemente como un servicio, teniendo en cuenta que el sistema operativo utilizado por el servidor es Ubuntu Linux, se optó por instalar Eclipse Mosquitto, un broker MQTT ligero y de código abierto (eG Innovations, s.f.)

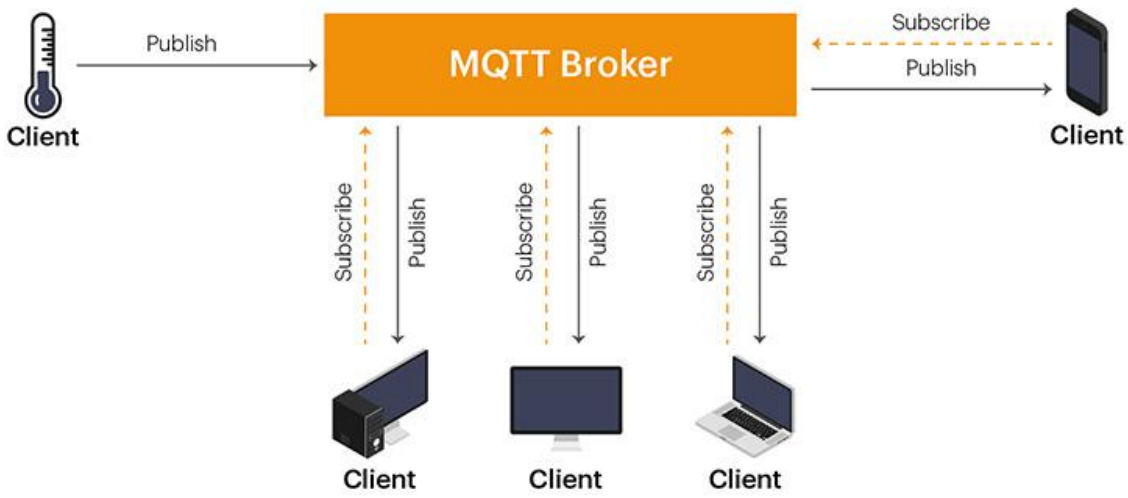


Figura 13. Esquema Publicador/Subscriptor.
Fuente: (eG Innovations, s.f.)

5.2.2.2. Aplicación Laravel

La aplicación desarrollada con el framework Laravel, permite crear los servicios backend adicionales necesarios para gestionar toda la información del sistema, así como de ponerla a disposición de los usuarios. Laravel trabaja con la arquitectura de software MVC (modelo, vista, controlador), lo que permite implementar el sistema telemático de monitorización con un código limpio, organizado y escalable. Además, integra un sistema de rutas y acceso a base de datos mediante su ORM integrado, facilitando el trabajo y optimizando el tiempo necesario para el desarrollo de la solución propuesta.

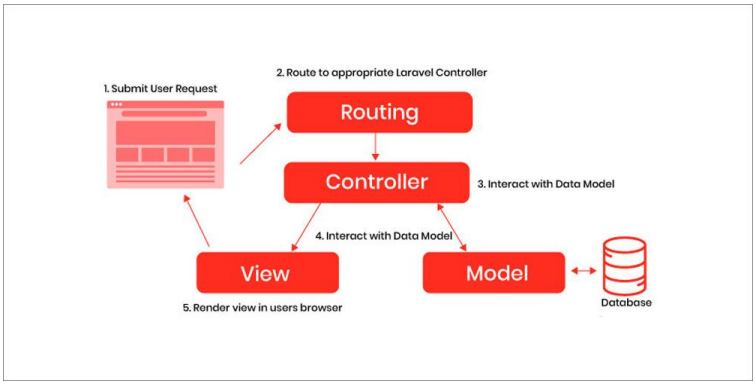


Figura 14. Arquitectura MVC de Laravel. Fuente: (Monocubed, 2025)

Una vez la información de las lecturas de humedad del suelo sean enviadas por los sensores hacia el broker MQTT, el broker las envía de nuevo hacia la aplicación Laravel. Para lograr esto, es necesario desarrollar un servicio independiente que siempre esté en ejecución en el sistema operativo y esté “escuchando” la información que envía el broker, este servicio se desarrolla mediante una herramienta que integra Laravel denominada “comandos”, un comando es una instrucción de consola, que permite automatizar tareas repetitivas dentro del sistema que se está desarrollando.

Para este caso, el comando ejecuta un bucle infinito en donde corre un cliente MQTT, quien será un suscriptor del broker MQTT. En este punto la aplicación recibe la información de las lecturas, las persiste en base de datos y las delega al siguiente componente del sistema backend, quien valida los rangos de los valores y genera las alertas a los usuarios si es necesario. Para desarrollar e implementar el cliente MQTT en Laravel, es necesario instalar dependencias y librerías adicionales, desarrolladas para tal fin.

5.2.2.3. Generación de alertas

Cada vez que la aplicación Laravel recibe información proveniente de los sensores, los datos son validados mediante un procedimiento independiente y desacoplado del flujo principal de la aplicación, con el fin de evitar bloqueos en el sistema de recepción de datos. Este procedimiento evalúa los valores recibidos con base en los rangos mínimos y máximos previamente configurados. En caso de que los datos se encuentren fuera de dichos rangos, se genera una alerta y se envía una notificación a los usuarios asociados al cultivo o invernadero desde el cual se originó la lectura.

El desacoplamiento del sistema se logra mediante el uso de colas en Laravel, las cuales permiten gestionar tareas de manera asíncrona. Este mecanismo recibe las tareas o procesos y los ejecuta de forma ordenada según su llegada, evitando que operaciones que pueden tardar, como el envío de notificaciones, afecten el flujo principal de la aplicación. Por esta razón, el uso de colas resulta una solución adecuada en estos casos.

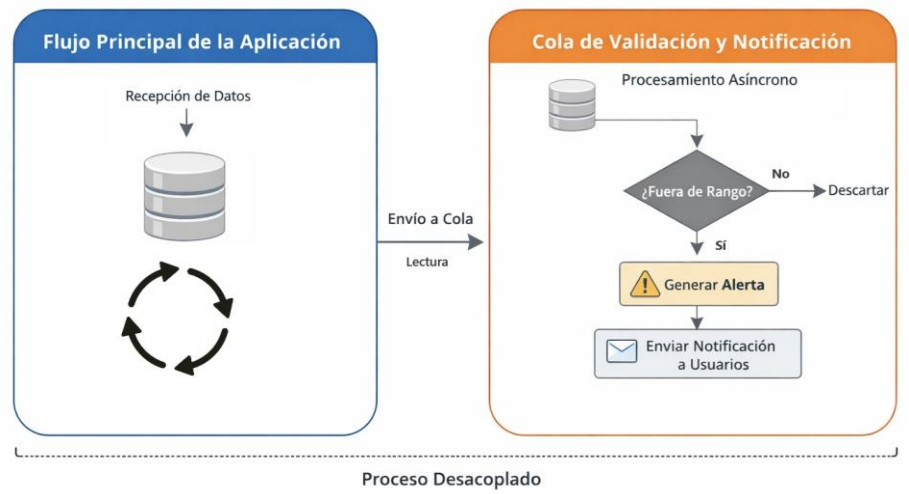


Figura 15. Esquema de utilización de colas. Fuente: Elaboración propia

Para el envío de notificaciones, se hace uso de APIs externas de mensajería, tanto para correo electrónico como para mensajes SMS, las cuales son contratadas durante el desarrollo del proyecto. Entre estas, se utiliza Resend para el envío de correos electrónicos.

5.2.3. Frontend

Las interfaces web y vistas del usuario se implementan utilizando el framework Vue.js, el cual es altamente compatible con Laravel. Este permite el desarrollo de interfaces mediante componentes reutilizables y facilita la comunicación con el backend de manera fácil y eficiente utilizando Inertia. Además, permite la utilización de un stack completo de desarrollo frontend y backend.

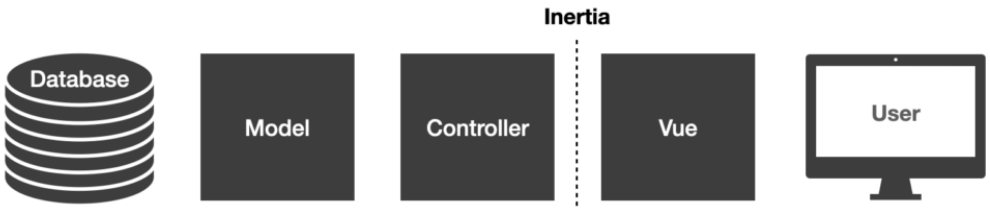


Figura 16. Stack Laravel Vue. Fuente: (Thomson, P. J., 2020)

En la Figura 17 se muestran los módulos que componen el frontend del sistema web, organizados de acuerdo con los perfiles de administrador y usuario final.

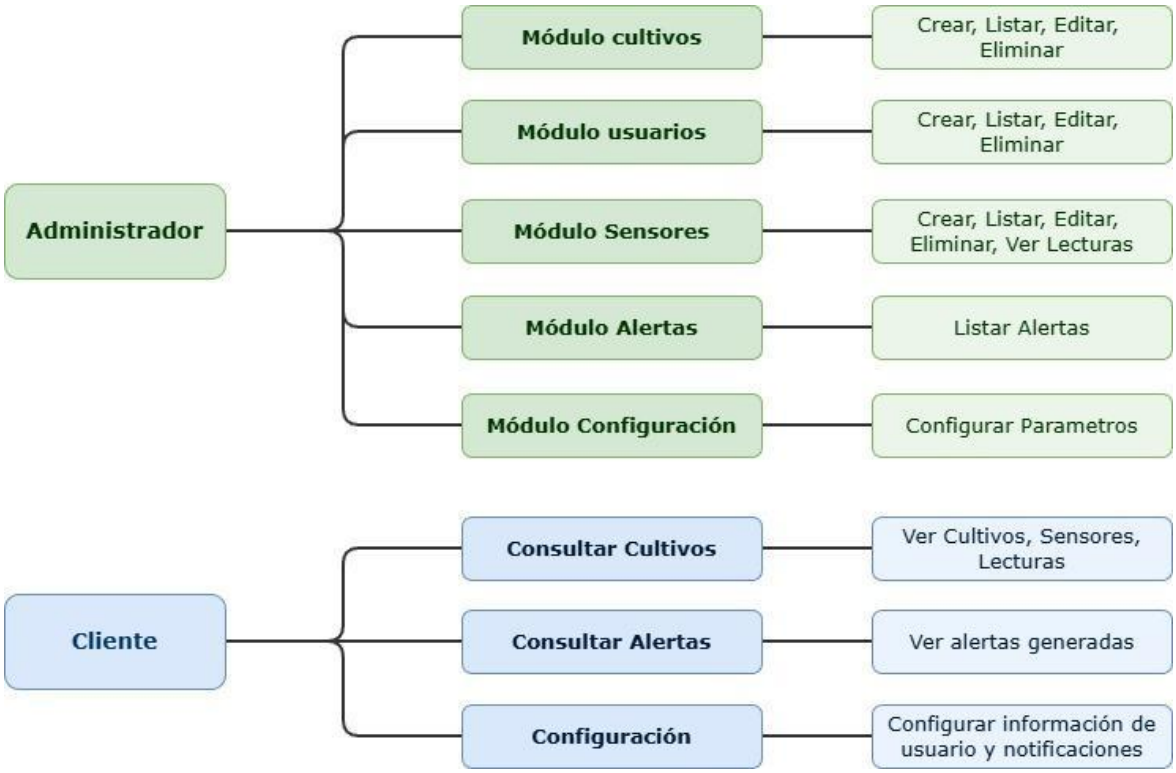


Figura 17. Módulos del frontend por roles. Fuente: Elaboración propia

Mediante la construcción e integración de estos módulos se garantiza la operatividad completa del sistema de monitorización remota. Así mismo facilitan la gestión de usuarios, cultivos y sensores, así como la visualización de alertas y configuraciones necesarias.

5.3. Desarrollo

En la fase de desarrollo se lleva a cabo la instalación y configuración de las herramientas necesarias para la ejecución del sistema web. Asimismo, se realiza la programación e implementación de los diferentes módulos del sistema de monitorización remota de la humedad del suelo, que ayudará en la toma de decisiones a los agricultores, brindando información precisa y en tiempo real sobre las condiciones de humedad que existen en el suelo.

En la Tabla 35 se listan las diferentes herramientas que se necesitan para la fase de desarrollo.

Herramienta	Versión
Visual Studio Code	1.114
PHP	8.4
npm	10.7.0
Node.js	22.2.0
Laravel	12.0
Git	2.44.0
Composer	2.7.6
MariaDB	10.11.6

Tabla 35. Entorno de desarrollo. Fuente: Elaboración propia

5.3.1. Gestión de usuarios y autenticación

Para la fase, se desarrollan los diferentes componentes del backend y frontend necesarios para la gestión de usuarios y autenticación en el sistema, de acuerdo con lo definido en la fase de diseño.

5.3.1.1. Creación del modelo de datos

En esta primera actividad se define y crea la tabla de usuarios utilizando las migraciones de Laravel, una vez definida es necesario ejecutar las migraciones en el entorno de desarrollo, lo que genera automáticamente la tabla correspondiente en el sistema gestor de bases de datos.

Para la gestión de la información en la base de datos se emplea Eloquent, un ORM integrado de Laravel que permite interactuar con la base de datos mediante modelos orientados a objetos, esto facilita la manipulación de datos sin necesidad de escribir consultas SQL.

Posteriormente, se define el modelo correspondiente a la tabla usuarios de la base de datos, con los atributos y métodos necesarios para representar la entidad usuarios. Este modelo es implementado utilizando el ORM Eloquent, que proporciona funcionalidades para la gestión de relaciones entre entidades.

5.3.1.2. Creación módulo login y autenticación de usuarios

En esta actividad se llevó a cabo el desarrollo y creación de las páginas y servicios necesarios para el login y la autenticación, así como un módulo para la recuperación de la contraseña. El módulo de autenticación redirige al usuario al panel o dashboard correspondiente según su tipo de usuario (Administrador y cliente).

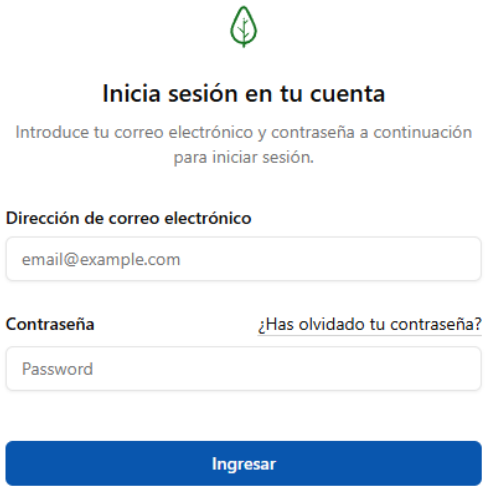


Figura 18. Interfaz inicio de sesión del sistema a web.
Fuente: Elaboración propia

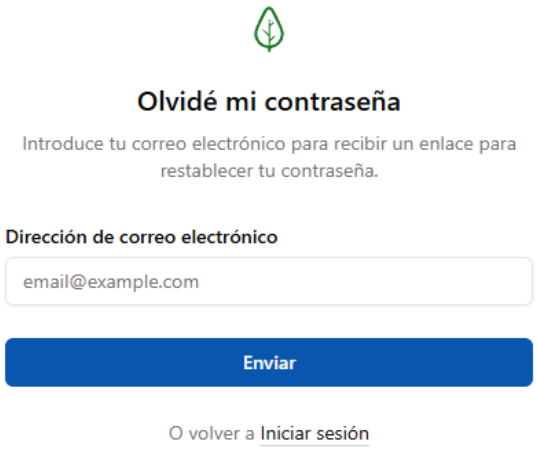


Figura 19. Interfaz recuperación de contraseña.
Fuente: Elaboración propia

5.3.1.3. Creación dashboard para administrador y cliente

Para la separación de responsabilidades de los diferentes usuarios del sistema, es necesario desarrollar dos paneles diferentes dentro del sistema web, teniendo en cuenta que existen dos tipos de usuario en el sistema. Mediante el uso de middlewares, de Laravel, es posible dirigir o redirigir usuarios a los módulos permitidos asegurando la confidencialidad de la información.

Cada panel o dashboard del sistema contiene módulos específicos definidos según el rol del usuario. El administrador cuenta con acceso a funcionalidades de gestión, incluyendo la administración de usuarios, invernaderos y sensores. Por otro lado, el usuario final o cliente dispone de módulos orientados a la consulta, permitiéndole visualizar sus cultivos y los sensores asociados. Esta diferenciación se implementa conforme al esquema de asignación de permisos y roles definido en la fase de diseño.

```php
Route::get('dashboard', function () {
    return Inertia::render('Dashboard');
})->middleware(['auth', 'verified', 'redirect_role'])->name('dashboard');

Route::middleware(['auth', 'role:admin'])->group(function () {
    Route::get('admin/dashboard', [AdminDashboardController::class, 'index'])->name('admin.dashboard');
    Route::resource('invernadero', CropController::class);
    Route::resource('users', UserController::class);
    Route::get('/user/search', [UserController::class, 'search'])->name('user.search');
    Route::post('/invernadero/{invernadero}/vincular', [CropController::class, 'attachUser']);
    Route::delete('/invernaderos/{invernadero}/users/{user}', [CropController::class, 'detachUser']);
    Route::resource('sensors', SensorController::class);
});
```

Figura 20. Código manejo de rutas en Laravel.
Fuente: Elaboración propia

5.3.1.4. Desarrollo módulo gestión de usuarios

En esta sección, se desarrollan las páginas y módulos que permiten la creación, modificación, eliminación y el listado de todos los usuarios del sistema telemático.

Para el desarrollo de los diferentes módulos del sistema se utilizó la arquitectura MVC (Modelo - Vista - Controlador), la cual permite estructurar la aplicación mediante la separación de responsabilidades entre la lógica de negocio, la gestión de datos y la presentación de la información. Este enfoque facilita la mantenibilidad, escalabilidad y la organización del código.

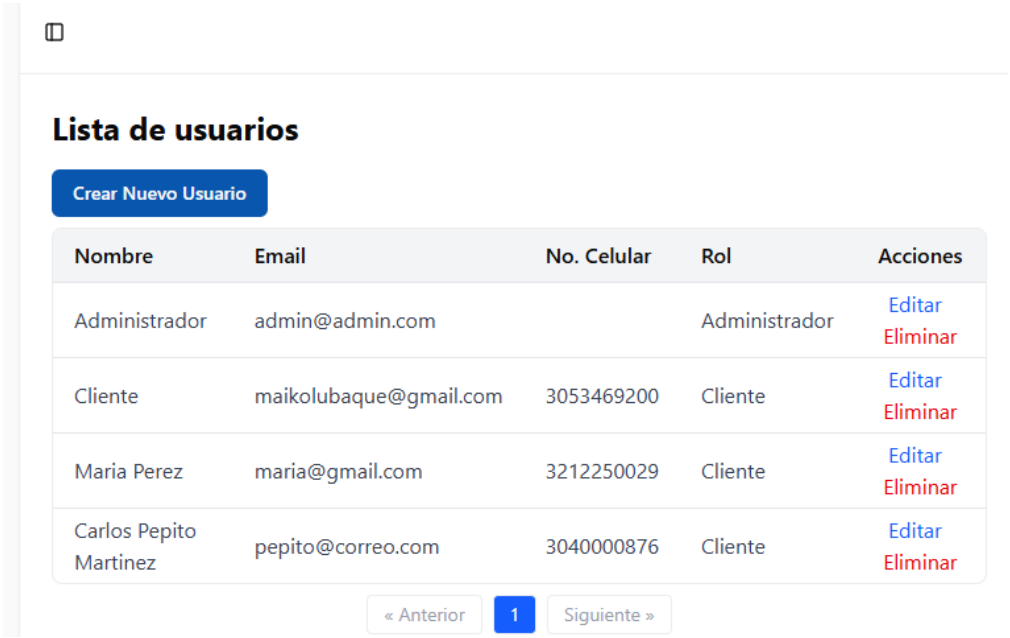


Figura 21. Módulo gestión de usuarios.
Fuente: Elaboración propia

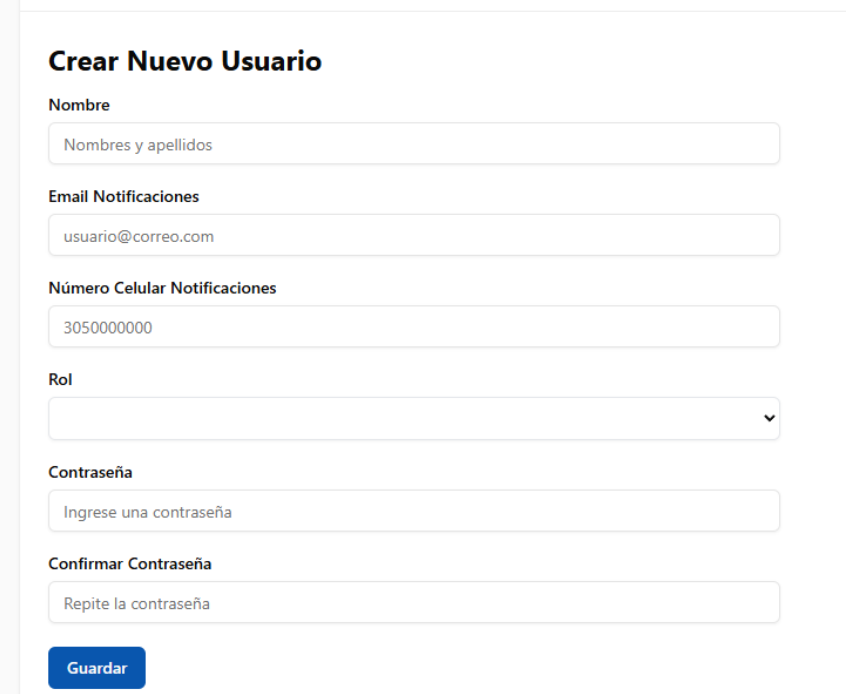


Figura 22. Interfaz para la creación de usuarios.
Fuente: Elaboración propia

El correo electrónico y el número de celular del usuario se utilizan como canales de notificación para el envío de las alertas generadas por el sistema telemático. Esto se debe a que pueden existir múltiples usuarios responsables de un mismo cultivo, garantizando así que la información sea recibida oportunamente por todas las personas involucradas.

5.3.2. Gestión de invernaderos y gestión de sensores

En esta fase, se implementaron los módulos necesarios para gestionar la información de los invernaderos y los sensores. Aquí se crearon las interfaces para que el administrador del sistema pueda listar, crear modificar y eliminar registros, así como asignar usuarios y sensores a invernaderos.

5.3.2.1. Modelo de datos y gestión de cultivos

Al igual que la creación del modelo de datos de usuario, se emplea Eloquent para la creación y gestión directa de los datos con la base de datos, así como para el desarrollo del modelo de cultivo. En la siguiente figura, se muestra un apartado del código que ilustra la manera en cómo se implementa Eloquent en un modelo de datos:

```php
class Crop extends Model
{
    protected $table = 'crops';

    protected $fillable = [
        'name',
        'location',
        'description',
        'is_active'
    ];

    /**
     * Relación muchos a muchos con usuarios
     */
    public function users(): BelongsToMany
    {
        return $this->belongsToMany(User::class, 'user_crop', 'crop_id', 'user_id');
    }

    /**
     * Sensores del invernadero
     */
    public function sensors(): HasMany
    {
        return $this->hasMany(Sensor::class, 'sensor_id');
    }
}
```

Figura 23. Código modelo cultivo. Fuente: Elaboración propia

Todas las interfaces web con las que los usuarios interactúan, se desarrollan utilizando el framework Vue. En la figura 24 se muestra la interfaz para la gestión de cultivos, indicando el nombre del cultivo, su ubicación y descripción.

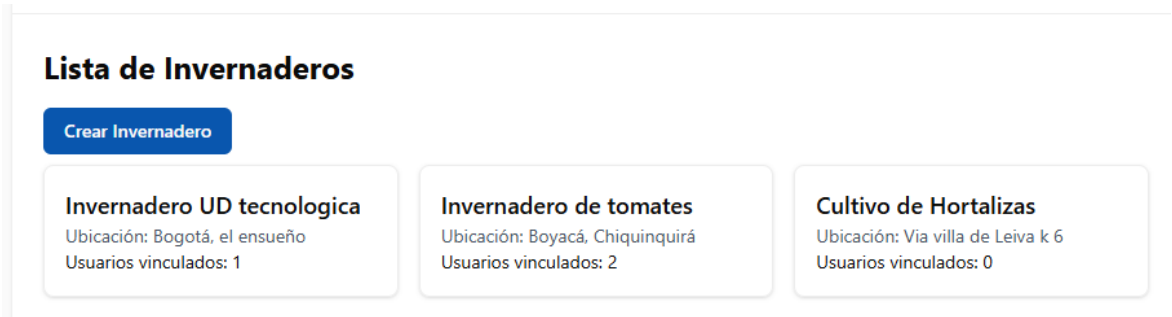


Figura 24. Interfaz listado de invernaderos. Fuente: Elaboración propia

La asignación de usuarios a un cultivo se implementó dentro de la misma vista destinada a la edición y eliminación del cultivo. Esta integración permite centralizar la gestión en un único punto, facilitando su administración y mejorando la usabilidad del sistema para el usuario administrador.

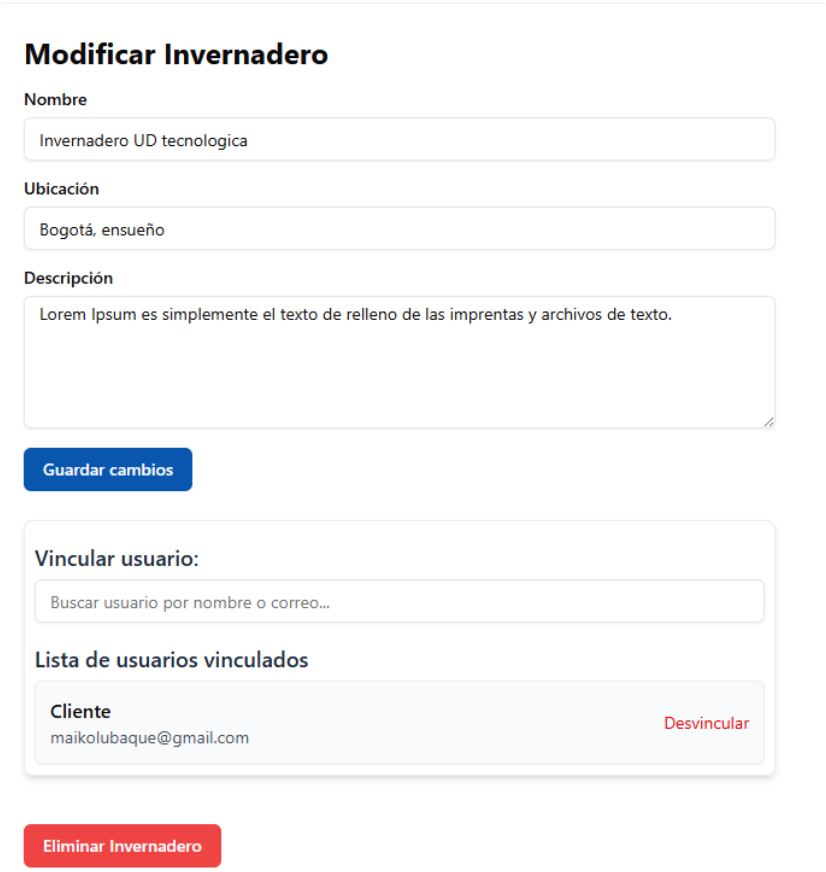


Figura 25. Interfaz gestión de invernaderos. Fuente: Elaboración propia

5.3.2.2. Modelo de datos y gestión de sensores

Se genera la migración y adicional se crea el modelo del sensor utilizando Eloquent para permitir la conexión y gestión de los datos con la base de datos. Además, se crean las interfaces y el controlador necesarios para asegurar el flujo completo de información.

Es importante destacar que el identificador del modelo sensor se incorpora en el prototipo del sensor de humedad, actuando como un campo único que permite asociar de forma inequívoca cada dispositivo físico con su representación en el sistema, facilitando su identificación y gestión dentro de los cultivos.



Figura 26. Módulo gestión de sensores. Fuente: Elaboración propia

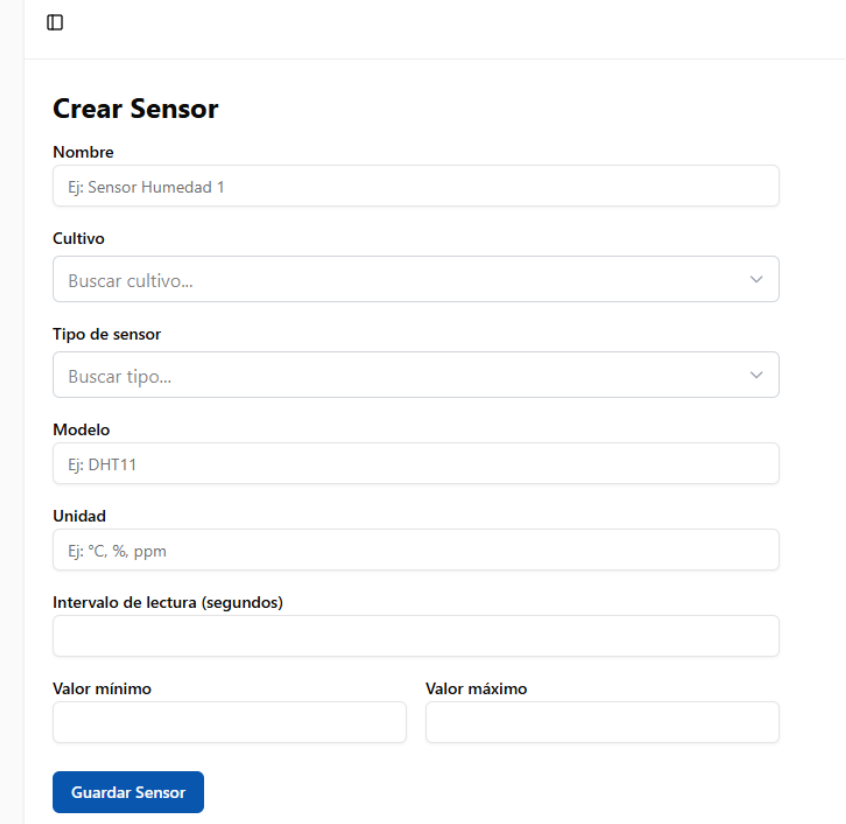


Figura 27. Interfaz de registro de sensores. Fuente: Elaboración propia

La configuración de los parámetros de los rangos de aceptación para las lecturas se realiza de forma individual para cada sensor. Esto se debe a que pueden existir diferentes modelos de sensores de un mismo tipo, por ejemplo, sensores de humedad de suelo de distintos fabricantes, los cuales pueden presentar la información utilizando escalas, unidades o niveles de precisión diferentes.

Por esta razón, se establece una configuración específica por sensor, permitiendo definir valores mínimos y máximos acordes a sus características particulares, garantizando así una correcta interpretación de los datos y una generación de alertas más precisa dentro del sistema.

5.3.3. Captura y representación de datos

Se desarrollaron los módulos y procesos necesarios para la gestión de las lecturas provenientes de los sensores, así como la generación y gestión de las alertas. Además, se implementan las interfaces de usuario para la representación de los datos al usuario final, mediante el uso de tablas y gráficos que permitan una interpretación clara de la información.

5.3.3.1. Creación de los modelos de datos para lecturas y alertas

Mediante la utilización de Eloquent de Laravel, en conjunto con las migraciones, se definen los modelos necesarios para establecer conexión con la base de datos y las tablas correspondientes para permitir la manipulación de la información.

5.3.3.2. Módulo de recepción de lecturas

Se establecen e implementan los procesos necesarios para la gestión de las lecturas enviadas por los sensores, mediante la instalación y configuración del broker MQTT en el servidor central, según los establecido en la fase de diseño.

La instalación del broker MQTT se realiza sobre un entorno Linux mediante la ejecución de comandos desde la terminal del sistema operativo. Posteriormente, la configuración se lleva a cabo mediante archivos de configuración, en los cuales se establecen parámetros relacionados con la autenticación y el control de acceso de los clientes MQTT.

```
vps-55369427:~$ sudo apt install mosquitto mosquitto-clients
```

Figura 28. Comando Instalación Broker Mosquitto.
Fuente: Elaboración propia

En el archivo de configuración ubicado en la ruta /etc/mosquitto/mosquitto.conf, se establecen parámetros relacionados con los registros del sistema (logs), el archivo de autenticación de usuarios, el puerto de comunicación y otras configuraciones necesarias para el funcionamiento del broker MQTT. Por otra parte, en el archivo /etc/mosquitto/pwfile se almacenan los usuarios y las contraseñas cifradas utilizadas para la autenticación de los clientes MQTT.

Además, se desarrolla e implementa un proceso de ejecución continua en el backend, que mantiene corriendo un cliente MQTT suscrito a un topic del broker MQTT, permitiendo la recepción de los mensajes emitidos, los cuales contienen las lecturas de los sensores y la información necesaria para validar la conexión. Este cliente MQTT es desarrollado utilizando las librerías y herramientas que proporciona el entorno de Laravel.

```php
    * Execute the console command.
    */
    public function handle()
    {
        $service = app(ReadingService::class);
        $server   = env('MQTT_HOST');
        $port     = env('MQTT_PORT', 1883);
        $clientId = 'laravel_mqtt_' . uniqid();

        $mqtt = new MqttClient($server, $port, $clientId);

        $settings = (new ConnectionSettings)
            ->setUsername(env('MQTT_USERNAME'))
            ->setPassword(env('MQTT_PASSWORD'))
            ->setKeepAliveInterval(60);

        $mqtt->connect($settings, true);

        $this->info('Conectado a MQTT');

        $mqtt->subscribe('crop/sensor/humidity/', function ($topic, $message) use ($service) {

            $this->info("Datos recibidos: " . $message . ' Topic: ' . $topic);
            $data = json_decode($message, true);

            if (!is_array($data)) {
                $this->info("Datos no guardados");
                return;
            }

            $service->store($data);

            $this->info("Datos guardados");
        }, 1);

        $mqtt->loop(true);
    }
}
```

Figura 29. Código implementación del cliente MQTT en Laravel.
Fuente: Elaboración propia

Una vez las lecturas son recibidas por el cliente MQTT ejecutado en el backend, estas son enviadas a un servicio encargado de validar la información y almacenarla en la base de datos. Posteriormente, mediante eventos y procesos desacoplados, la información es delegada a un módulo responsable de verificar los rangos óptimos definidos para la humedad del suelo y generar alertas al usuario final cuando se detectan valores fuera de los límites establecidos.

5.3.3.3. Generación de alertas.

La generación de alertas al usuario final se realiza mediante el envío de un correo electrónico y/o un mensaje SMS, indicando que las lecturas de la humedad del suelo recibidas están por fuera del rango óptimo, es decir, por fuera de los rangos óptimos establecidos para cada sensor. Este mensaje incluye el nombre del invernadero en donde se presenta la novedad, el sensor que está generando las lecturas, el valor recibido, la hora y la fecha.

En la Figura 30 se muestra el código PHP necesario para validar los datos, generar la alerta en base de datos y enviar la notificación al usuario:

```php
public function handle(): void
{
    $this->reading->load([
        'sensor.crop.users'
    ]);

    $sensor = $this->reading->sensor;
    $crop = $sensor->crop;

    $valMax = $sensor->max_value;
    $valMin = $sensor->min_value;

    $title = null;
    $type = null;

    if ($this->reading->value > $valMax) {
        $title = 'Humedad alta';
        $type = SensorAlertLevel::HIGH;
    }

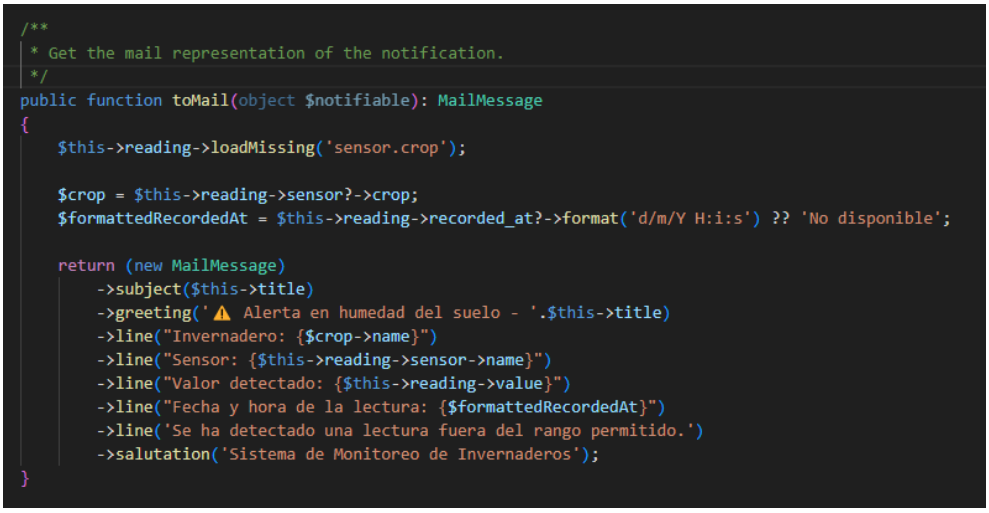
    if ($this->reading->value < $valMin) {
       $title = 'Humedad baja';
       $type = SensorAlertLevel::LOW;
    }

    Alert::create([
        'sensor_id' => $sensor->id,
        'value' => $this->reading->value,
        'type' => $type,
        'message' => $title,
        'triggered_at' => now(),
    ]);

    foreach ($crop->users as $usuario) {
        $usuario->notify(
            new AlertaLecturaNotification(
                $title,
                $this->reading
            )
        );
    }

}
```

Figura 30. Código para la validación de lecturas y generación de alertas.
Fuente: Elaboración propia



```php
/**
 * Get the mail representation of the notification.
 */
public function toMail(object $notifiable): MailMessage
{
    $this->reading->loadMissing('sensor.crop');

    $crop = $this->reading->sensor?->crop;
    $formattedRecordedAt = $this->reading->recorded_at?->format('d/m/Y H:i:s') ?? 'No disponible';

    return (new MailMessage)
        ->subject($this->title)
        ->greeting('⚠️ Alerta en humedad del suelo - '.$this->title)
        ->line("Invernadero: {$crop->name}")
        ->line("Sensor: {$this->reading->sensor->name}")
        ->line("Valor detectado: {$this->reading->value}")
        ->line("Fecha y hora de la lectura: {$formattedRecordedAt}")
        ->line('Se ha detectado una lectura fuera del rango permitido.')
        ->salutation('Sistema de Monitoreo de Invernaderos');
}
```

Figura 31. Código para el envío de los correos desde Laravel.
Fuente: Elaboración propia

Estos procesos son realizados utilizando eventos, listeners y colas de trabajo, con el objetivo de desacoplar la lógica de procesamiento, mejorar el rendimiento del sistema y evitar bloqueos en la ejecución de las peticiones principales. De esta manera la validación de las lecturas, la generación de alertas y el envío de notificaciones se ejecutan de forma asíncrona en segundo plano, permitiendo una mayor escalabilidad y eficiencia en el procesamiento de la información recibida desde los dispositivos/sensores IoT.

5.3.3.4. Interfaz de usuario final

Una vez desarrollados los diferentes módulos que conforman el sistema web, se integran en una única página mediante un menú de navegación. Para ello, se implementan dos paneles de control (dashboard), uno destinado al administrador y otro al usuario final, desde los cuales es posible acceder a las diferentes funcionalidades del sistema de acuerdo con el rol y los permisos asignados.

- **Panel de administración**

Permite la navegación entre los módulos de invernaderos, usuarios, sensores y alertas. Además, la página principal del panel, muestra un resumen general de la cantidad de registros almacenados en el sistema, así como un listado de las últimas alertas generadas por los sensores.

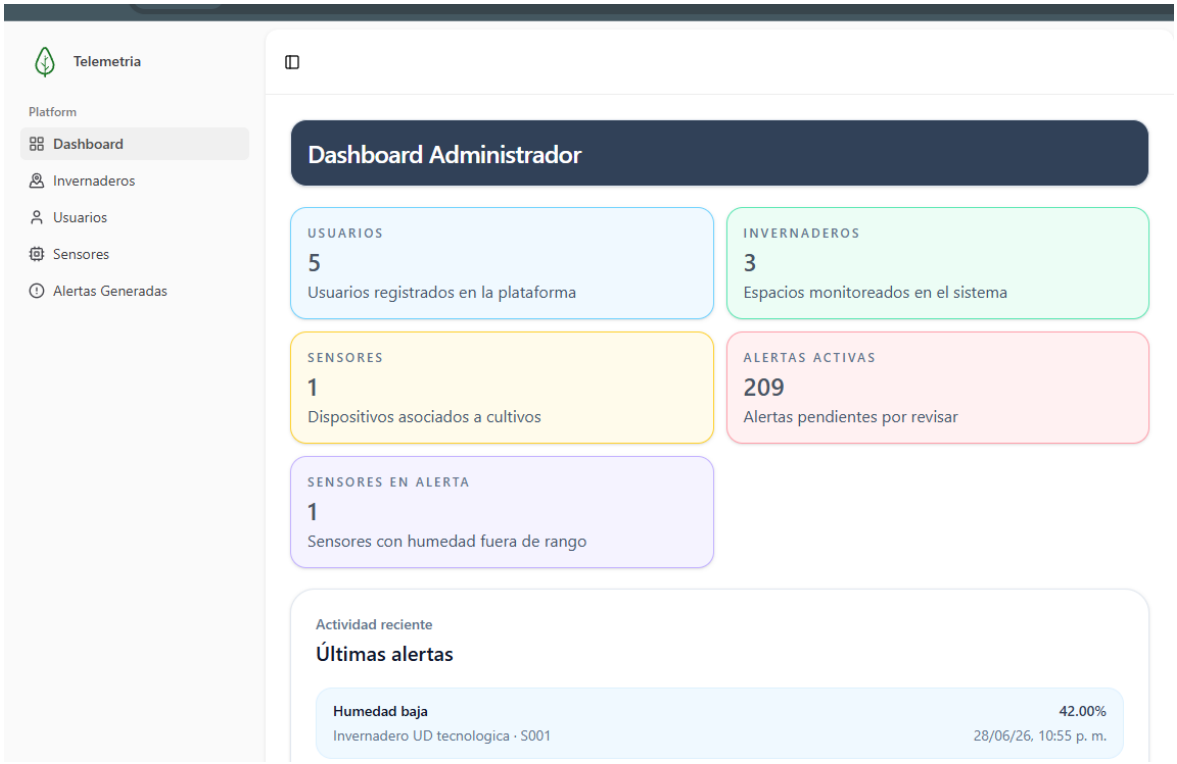


Figura 32. Panel rol administrador del sistema web.
Fuente: Elaboración propia

- **Panel de cliente**

Los usuarios finales o interesados en utilizar el sistema telemático para la monitorización de la humedad del suelo, podrán ingresar al sistema web para ver toda la información recopilada por los sensores en sus respectivos invernaderos. De esta manera se puede brindar información precisa, que ayude a los diferentes interesados en la toma de decisiones al momento de realizar el riego controlado en los cultivos.

Desde el panel del cliente, los usuarios podrán visualizar el estado actual de los sensores en tiempo real, incluyendo el nivel de lectura de humedad del suelo representado en porcentaje, la hora de la última lectura tomada y el nivel de alerta del sensor (si la lectura está por fuera del rango configurado).

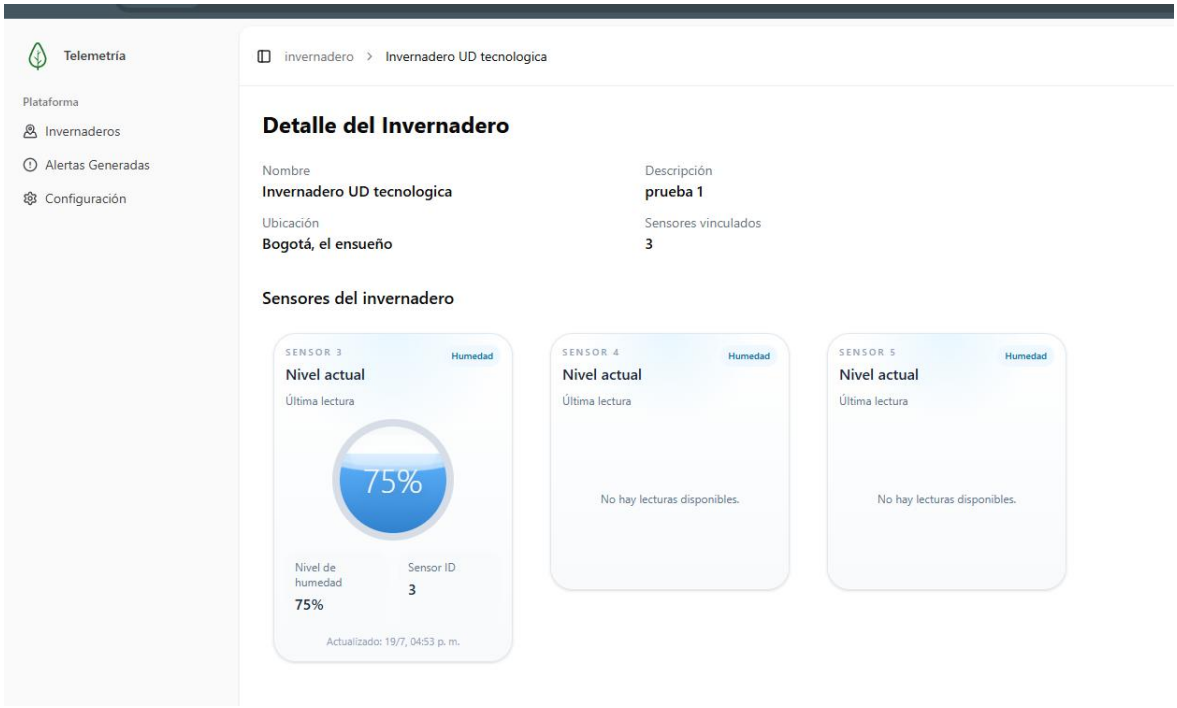


Figura 33. Interfaz detalle de invernadero.
Fuente: Elaboración propia

Adicionalmente, en el panel del cliente se permite la visualización del histórico de las alertas generadas, así como de las lecturas recopiladas por el sensor, mediante tablas y gráficas de líneas que permiten consultar en intervalos configurables, desde cinco minutos hasta un mes.

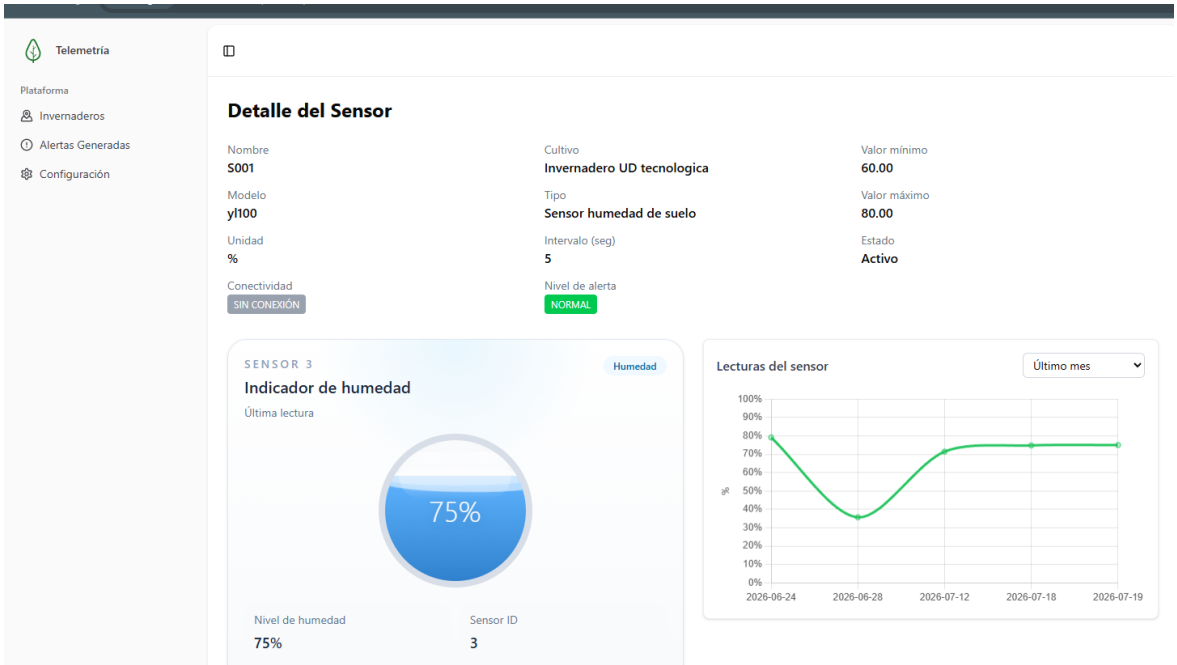


Figura 34. Interfaz detalle de sensor.
Fuente: Elaboración propia



Figura 35. Interfaz de alertas generadas.
Fuente: Elaboración propia

5.4. Implementación

En la fase de implementación del sistema web se pone en funcionamiento el sistema en un entorno de producción, con el propósito de permitir la recepción, procesamiento y almacenamiento de los datos generados por los sensores, así como de permitir el ingreso de los usuarios de manera remota desde cualquier navegador.

Esto se logra mediante la contratación de un servidor privado virtual (VPS), encargado de alojar el sistema web y todos los servicios asociados para el correcto funcionamiento de todos los componentes del backend, permitiendo el acceso remoto de los usuarios desde cualquier ubicación geográfica con conexión a internet.

5.4.1. Configuración del servidor

La contratación del servidor se realiza con la empresa OVHcloud[3] en su plan de contratación de servidores privados virtuales. El servidor se configura con el sistema operativo Ubuntu Server.

[3] OVHcloud. Virtual Private Server (VPS). Disponible en: https://www.ovhcloud.com/es/vps/. Consultado el 27 de mayo de 2026.

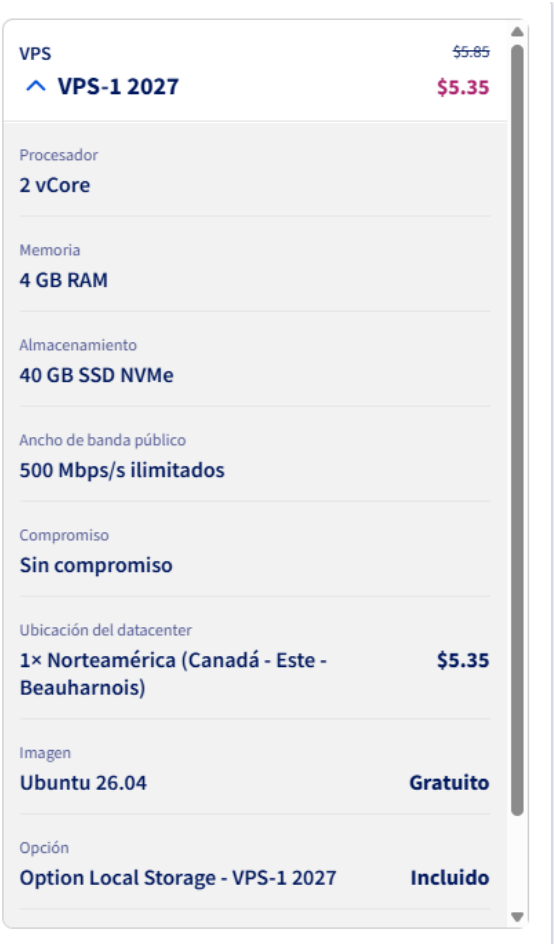


Figura 36. Contratación del servidor - plataforma OVHCloud.
Fuente: Elaboración propia

Una vez realizada la contratación del servidor, se procede a realizar las primeras configuraciones para su puesta en funcionamiento. Entre ellas la creación de la cuenta de usuario de Linux, aplicar actualizaciones del sistema, la configuración de los mecanismos básicos de seguridad y acceso SSH.

5.4.2. Instalación de dependencias

Las dependencias necesarias para el funcionamiento y operación del sistema web que se instalaron en el servidor son:

- PHP
- Composer
- Nginx
- MariaDB
- Mosquitto (broker MQTT)
- Supervisor
- Node.js
- Certbot
- Git

5.4.3. Despliegue

Una vez configurado el servidor e instaladas las dependencias necesarias, se procede a copiar los archivos del código fuente desde el entorno de desarrollo al entorno de producción (VPS). Debido a que en la fase de desarrollo del sistema web se utilizó un repositorio git para alojar el proyecto, esta operación se hace mediante el comando “clone” y “pull” de git.



Figura 37. Comando para despliegue de la aplicación Laravel.
Fuente: Elabora

Una vez copiado el proyecto al servidor es necesario configurar las variables de entorno definidas en el archivo “.env”, las cuales definen los parámetros requeridos para el correcto funcionamiento de la aplicación. Entre estas configuraciones se encuentran la dirección IP del broker MQTT, los datos de conexión a la base de datos, las credenciales de acceso a los servicios externos y los tokens de autenticación utilizados para la integración con las APIs de envío de correos electrónicos y mensajes SMS.

Por último y con el fin de mejorar el rendimiento del sistema backend, se realiza la optimización de la aplicación Laravel mediante el almacenamiento en caché de sus configuraciones, rutas y vistas.

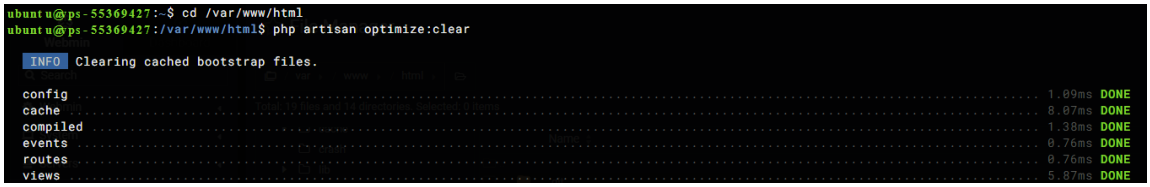


Figura 38. Comando para optimización aplicación Laravel.
Fuente: Elaboración propia

5.4.4. Configuración de procesos en segundo plano

Para garantizar el correcto funcionamiento del sistema, es necesario que algunos servicios se ejecuten de forma permanente en segundo plano, de manera paralela al servidor web. Entre estos se encuentran el cliente MQTT de Laravel, el procesamiento de las colas de trabajos y notificaciones, y el servidor de WebSockets Reverb. Estos procesos permiten el trabajo asíncrono y en tiempo real de los diferentes eventos que suceden dentro del sistema telemático.

Con el fin de asegurar la disponibilidad estos servicios se utiliza Supervisor, un gestor de procesos para Linux que permite iniciar, supervisar y reiniciar automáticamente los procesos en caso de fallos, interrupciones inesperadas o reinicios del servidor.

En la figura 39 se listan los procesos configurados y ejecutándose continuamente en el servidor:

```
ubuntu@ps-55369427:/var/www/html$ sudo supervisorctl status
laravel-mqtt                     RUNNING   pid 1011, uptime 19 days, 0:49:59
laravel-worker:laravel-worker_00 RUNNING   pid 1012, uptime 19 days, 0:49:59
reverb                           RUNNING   pid 97812, uptime 18 days, 4:15:18
```

Figura 39. Procesos en ejecución en segundo plano.
Fuente: Elaboración propia

Una vez terminado el proceso de despliegue, se comprueba que el sistema web responde correctamente, accediendo a la aplicación mediante la dirección IP pública del servidor o a través del nombre de dominio configurado, utilizando un navegador web. Con esto, se verifica que el sistema se encuentra disponible para su uso en un entorno de producción.

La siguiente figura muestra que el sistema web responde correctamente a las solicitudes realizadas por los usuarios a través de un navegador web.



Figura 40. Página principal del sitio web en producción.
Fuente: Elaboración propia

Mediante la ejecución de las actividades o fases del desarrollo de software descritas anteriormente, el sistema web, conformado por el backend y el frontend, fue implementado y desplegado en un entorno de producción, permitiendo su acceso remoto a los usuarios mediante internet.

Además, se garantiza la correcta ejecución de los servicios necesarios para la recepción, procesamiento y visualización de la información proveniente del prototipo, cumpliendo así con el objetivo de diseñar e implementar un sistema web capaz de recibir, almacenar y procesar los datos de humedad del suelo enviados por uno o varios prototipos.

6. Integración del sistema

Es fundamental garantizar que todos los elementos del sistema telemático funcionen de manera conjunta y coordinada, para asegurar el correcto flujo de la información, así como de la seguridad e integridad de los datos. Por esta razón, a continuación se describe y verifica el proceso de integración, comprobando la correcta comunicación entre los diferentes componentes del sistema.

El desarrollo de esta etapa se realiza de acuerdo con el Sprint número 6, actividad número 20, denominada “Integración del sistema”, establecida en la metodología adoptada para el proyecto.

6.1. Integración del prototipo con el broker MQTT

Después de desarrollado el prototipo para la medición de la humedad del suelo e instalado el broker MQTT en el servidor, se integran estos dos elementos mediante la configuración de canales de comunicación, con el fin de permitir el envío de las lecturas de humedad del suelo desde el prototipo a el broker MQTT.

Para esto, el código integrado en el microcontrolador ESP32 incorpora parámetros de conexión correspondientes a la red Wi-Fi y al broker MQTT, como se muestra a continuación.

Tipo	**Valor**
Host bróker MQTT	144.217.164.113
Usuario bróker MQTT	telemática
Contraseña bróker MQTT	contraseña configurada en el broker
Puerto	1883
topic	crop/sensor/humidity/
ssid Wi-Fi	Nombre de la red Wi-Fi
Contraseña Wi-Fi	Contraseña Wi-FI

Tabla 36. Parámetros de configuración del ESP32.
Fuente: Elaboración propia

Esta configuración permite establecer una comunicación directa entre el prototipo y el broker, utilizando internet mediante la conexión a una red Wi-Fi y haciendo uso del protocolo MQTT para el intercambio de mensajes.

6.2. Integración del broker MQTT con el backend

Como el broker MQTT y el backend son dos sistemas completamente separados y se ejecutan de manera independiente, es necesario establecer comunicación entre el backend, desarrollado en Laravel y el broker MQTT.

La comunicación entre estos dos elementos se logra, al igual que con el prototipo, mediante el protocolo MQTT. La aplicación Laravel también debe ser un cliente del broker, con la diferencia que es un cliente de tipo suscriptor, esto permite recibir las lecturas recibidas por el broker en el backend, para el respectivo procesamiento y almacenamiento de los datos.

A continuación, se muestran los parámetros necesarios para establecer conexión:

Tipo	**Valor**
Host bróker MQTT	144.217.164.113
Usuario bróker MQTT	telemática
Contraseña bróker MQTT	contraseña configurada en el bróker MQTT
Puerto	1883
topic	crop/sensor/humidity/

Tabla 37. Parámetros de configuración del cliente MQTT en Laravel. Fuente: Elaboración propia

6.3. Integración del sistema de notificaciones

Con el fin de notificar oportunamente a los usuarios sobre condiciones anómalas en la humedad del suelo, el sistema integra un servicio de envío de correos electrónicos mediante la plataforma Resend. La integración con la plataforma Resend se realiza a través de su API oficial. Para ello, se instala la librería compatible con Laravel y se configura el token de autenticación correspondiente en las variables de entorno de la aplicación.

A continuación, se evidencia que la plataforma Resend procesa y envía correctamente las notificaciones por correo electrónico a los usuarios.

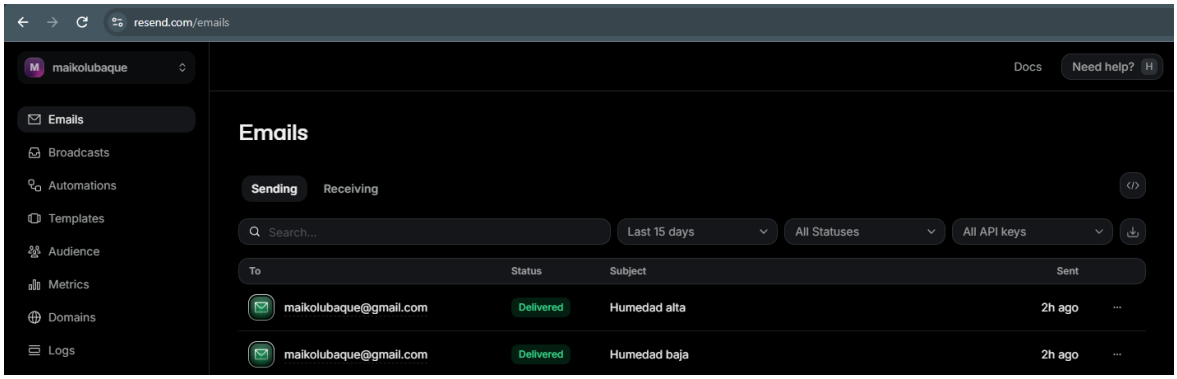


Figura 41. Correos enviados visualizados en la plataforma Resend.
Fuente: Elaboración propia

De forma complementaria, el sistema integra un servicio externo de mensajería SMS, el cual permite el envío de notificaciones a dispositivos móviles. Para ello, se utilizó la plataforma Textbelt, la cual proporciona una API REST que permite integrar servicios de mensajería con aplicaciones web. Mediante esta integración, el sistema envía automáticamente notificaciones al número de teléfono configurado cuando se detectan condiciones de humedad fuera de los rangos establecidos.

6.4. Integración de la página web con el backend

Una vez desarrollados el backend y frontend, se realiza la integración de ambos componentes mediante Inertia.js, una herramienta que actúa como puente o intermediario entre el framework Laravel y la biblioteca Vue.js. Esta integración permite que las solicitudes realizadas desde la interfaz web sean procesadas por el backend, el cual ejecuta la lógica de negocio, consulta la información almacenada en la base de datos y envía los datos requeridos a los componentes Vue para su representación en el navegador.

Gracias a la utilización de las anteriores herramientas, es posible implementar de manera fácil y segura la gestión de autenticación de usuarios basado en sesiones de Laravel y PHP.

En esta fase se lograron integrar correctamente todos los componentes que conforman el sistema telemático, haciendo que la solución propuesta sea completamente funcional. Durante esta etapa se comprobó que los componentes implementados puedan intercambiar información entre sí de manera correcta y que el flujo de la información y el procesamiento de los datos se haga según la arquitectura definida.

7. Pruebas y validación

Para asegurar y garantizar el correcto funcionamiento de todo el sistema en conjunto, es necesario realizar pruebas, tanto a los diferentes elementos que componen el sistema telemático de monitorización, así como validar el funcionamiento integral del sistema en un entorno simulado, sometiéndose a diferentes condiciones de humedad del suelo.

Esta fase del proyecto se realizó ejecutando el Sprint número 6, Actividad “Pruebas de funcionalidad” de la metodología Scrum planteada para el proyecto. A continuación, se detallan algunas de las tareas desarrolladas.

No.	Tarea	Descripción	Responsable
1	Pruebas del prototipo	Realizar pruebas individuales al prototipo desarrollado, verificando su correcto funcionamiento y operatividad.	Michael Jesus Ubaque
2	Pruebas del sistema web	Realizar pruebas al backend y el frontend, verificando que todos los procesos desarrollados funcionen según lo esperado.	Michael Jesus Ubaque
3	Pruebas de generación de alertas.	Realizar pruebas al módulo de recepción de lecturas para verificar su correcto funcionamiento y comprobar que las alertas sean generadas y los usuarios notificados de manera adecuada.	Cristian Felipe Molina
4	Pruebas en ambiente simulado	Evaluar el funcionamiento integral del sistema en un ambiente controlado que simule las condiciones de operación de un invernadero, bajo diferentes condiciones en la humedad del suelo.	Michael Jesus Ubaque y Cristian Felipe Molina

Tabla 38. Planificación de actividades del Sprint 3.
Fuente: Elaboración propia

7.1. Prototipo

Prueba	Descripción	Resultado esperado	Resultado
Encendido	Encendido del prototipo mediante la conexión de cable micro USB.	Se debe encender el prototipo, junto con los indicadores LED.	Exitoso
Conexión Wi-Fi	Establecer conexión Wi-Fi al punto de acceso configurado.	Se debe establecer conexión Wi-Fi y se debe encender el LED azul de prototipo, indicando una correcta conexión.	Exitoso
Conexión MQTT	Establecer conexión al broker MQTT	Se debe establecer conexión al broker MQTT. El broker debe mostrar una nueva conexión en en log.	Exitoso
Envío de datos	Envío de las lecturas al broker MQTT.	El prototipo debe enviar los datos en formato JSON al topic del broker configurado.	Exitoso
Recuperación ante pérdidas de conexión	Recuperar la conexión Wi-Fi ante una desconexión repentina por eventos como apagado de router.	El prototipo debe establecer conexión automáticamente a la red Wi-Fi, una vez la red esté nuevamente disponible. El Indicador LED azul de Wi-Fi debe encenderse de nuevo.	Exitoso

Tabla 39. Pruebas del prototipo. Fuente: Elaboración propia

7.2. Backend

Prueba	Descripción	Resultado esperado	Resultado
Conexión a la base de datos	Establecer conexión a la base de datos.	El backed se conecta correctamente con la base de datos.	Exitoso.
Conexión al broker MQTT	Verificar que el cliente MQTT de Laravel establece conexión al broker MQTT.	El cliente de Laravel se conecta correctamente al broker.	Exitoso
Recepción de mensajes MQTT	Suscripción al tópico correspondiente y recepción de lecturas en el cliente MQTT de Laravel.	El cliente Laravel se suscribe al topic y recibe las lecturas enviadas por el broker.	Exitoso
Validación de lecturas	Validar los datos de ingreso (en formato JSON) y comparar los valores con los rangos configurados.	Las lecturas se validan y se registra en la base de datos una alerta si la lectura está por fuera de los rangos establecidos.	Exitoso
Registro de lecturas	Almacenar la lectura en base de datos.	Se crea correctamente el registro de la lectura en la base de datos	Exitoso
Consulta de lecturas	Recuperación de datos mediante peticiones HTTP.	La información es retornada correctamente en formato JSON.	Exitoso
Consulta de lecturas en tiempo real	Recuperación de datos mediante una conexión WebSocket.	El backend abre una conexión persistente con el frontend mediante WebSockets y retorna las lecturas una vez son recibidas.	Exitoso.
Continuidad de los servicios	Verificar que todos los servicios de backend se inician correctamente ante reinicios del servidor.	Todos los servicios se inician automáticamente al reiniciar el servidor	Exitoso.

Tabla 40. Pruebas del backend. Fuente: Elaboración propia

7.3. Frontend

Prueba	Descripción	Resultado esperado	Resultado
Inicio de sesión	Los usuarios pueden ingresar al sistema con credenciales válidas.	El sistema valida que el usuario existe y lo dirige al panel de usuario correspondiente correctamente.	Exitoso.
Cierre de sesión	Verificar el cierre de sesión.	Los usuarios pueden cerrar sesión y son redirigidos a la interfaz de inicio de sesión.	Exitoso.
Registro de usuarios	El sistema permite crear nuevos usuarios	Los usuarios son creados y asignados a un invernadero correctamente.	Exitoso
Edicion de usuarios	Modificación de los datos de usuario, incluyendo el correo y número para notificaciones.	El sistema permite modificar los datos correctamente mediante la interfaz web	Exitoso
Gestión de cultivos	Crear, modificar, eliminar y visualizar los registros de los cultivos en la interfaz web.	El sistema permite gestionar correctamente la información de los cultivos reflejando los cambios en la interfaz.	Exitoso
Gestión de sensores	Crear, modificar, eliminar y visualizar los registros de los sensores en la interfaz web.	El sistema permite gestionar correctamente la información de los sensores reflejando los cambios en la interfaz.	Exitoso
Configuración de umbrales	Modificar los valores máximos y mínimos permitidos para la validación de las lecturas entrantes.	El sistema permite modificar correctamente los rangos máximos y mínimos permitidos.	Exitoso
Visualización de lecturas	Visualizar el histórico de las lecturas de un sensor.	Se visualizan correctamente las lecturas tomadas por un sensor mediante una tabla en la interfaz web.	Exitoso
Visualización de lecturas en tiempo real	Verificar que las nuevas lecturas aparezcan automáticamente mediante en la interfaz.	La información se actualiza sin necesidad de recargar la página.	Exitoso
Consulta de	Comprobar la generación	La gráfica representa	Exitoso

gráficas	de gráficas de líneas con los datos históricos.	correctamente las lecturas almacenadas.	
Consulta de alertas	Acceder al módulo de alertas registradas.	Se muestran las alertas con su información correspondiente.	Exitoso
Consultar el estado de los sensores.	Verificar que el estado del sensor se muestre correctamente en la interfaz.	El sistema refleja el estado actual del sensor de acuerdo con la información recibida.	Exitoso

Tabla 41. Pruebas del frontend. Fuente: Elaboración propia

7.4. Generación de alertas

Prueba	**Descripción**	**Resultado esperado**	**Resultado**
Registro de alertas	Al validar que una lectura está por fuera de los rangos establecidos se registra una alerta en base de datos y se activa el proceso de notificación.	Se crea un registro en base de datos con la información de la alerta y se llama al proceso de notificaciones.	Exitoso
Procesamiento en colas	Validar que el proceso de envío de notificaciones se crea y se pone en una cola de trabajo.	El proceso se crea correctamente, se pone en la cola y se ejecuta o despacha correctamente.	Exitoso
Envío de correo electrónico	Validar que los correos son enviados a los usuarios mediante el API de correo.	El correo llega correctamente a la bandeja de entrada del correo configurado.	Exitoso
Envío de mensaje SMS	Validar que los mensajes SMS son enviados a los usuarios mediante el API de mensajes SMS.	El mensaje SMS llega correctamente al celular del número configurado.	Exitoso
Consulta de alertas	Consulta del histórico de alertas generadas por parte de los usuarios.	Las alertas son representadas en la interfaz web correctamente.	Exitoso

Tabla 42. Pruebas sistema de alertas y notificaciones.
Fuente: Elaboración propia

7.5. Pruebas en ambiente simulado

Se realizaron diferentes pruebas para comprobar que todo el sistema telemático en conjunto funciona correctamente, para ello, se utilizaron macetas con tierra apta para el cultivo de plantas, junto con plantas sembradas y diferentes niveles de humedad del suelo.

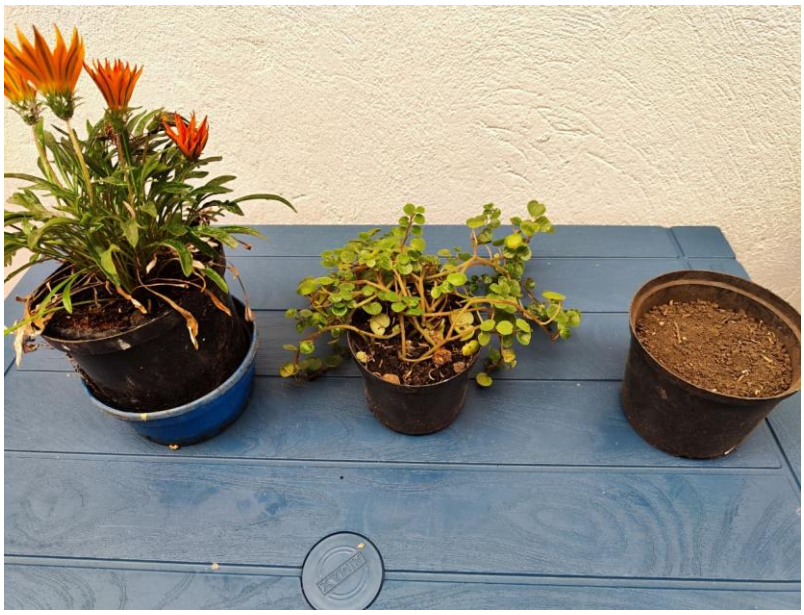


Figura 42. Macetas de prueba. Fuente: Elaboración propia

Las pruebas fueron hechas bajo condiciones controladas de humedad del suelo y en un ambiente cerrado, aislado de factores climáticos externos, como la lluvia y el viento, con el fin de simular las condiciones de operación de un invernadero. Para esto, se utilizó una cubierta plástica que permitió aislar el entorno de prueba de las condiciones ambientales externas.

Durante las pruebas se verificó el funcionamiento integral del sistema, incluyendo la captura de las lecturas por parte del prototipo, su transmisión al broker MQTT, el procesamiento y almacenamiento de la información en el backend, la visualización de los datos en la aplicación web y la generación de alertas y notificaciones cuando las condiciones de humedad se encontraban fuera de los rangos establecidos.

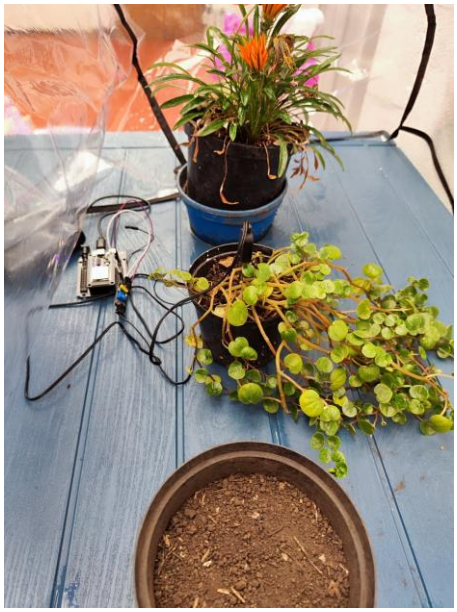


Figura 43. Macetas con cubierta plástica. Fuente: Elaboración propia

7.5.1. Prueba con un nivel medio de humedad del suelo.

Para esta prueba, se utilizó una maceta con tierra húmeda, como se muestra en la figura 44. Posteriormente, se insertó el sensor de humedad del suelo en la tierra, mientras el prototipo permanecía encendido y conectado a una red Wi-Fi y a un broker MQTT. La prueba se llevó a cabo durante un período de 10 minutos, con el fin de recopilar un número de lecturas suficiente que permita evaluar el comportamiento de todo el sistema.

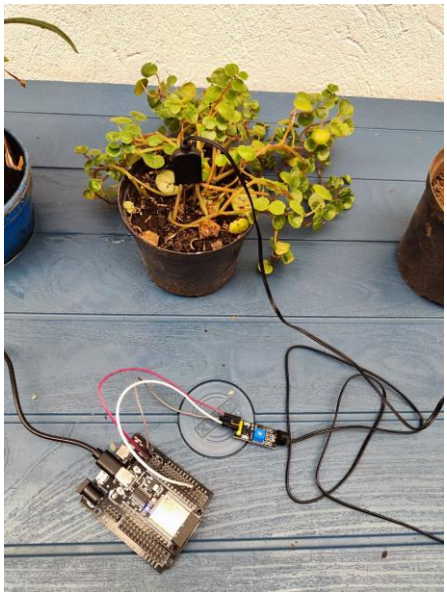


Figura 44. Maceta con nivel medio de humedad del suelo.
Fuente: Elaboración propia

Se comprobó que los datos son enviados y procesados correctamente en el backend, mediante la visualización de la información en la interfaz gráfica del sistema web implementado, correspondiente a la interfaz de detalle del sensor. Allí, se evidencia el

indicador de nivel de humedad de suelo, el gráfico lineal de lecturas en tiempo real y la tabla con el histórico de las lecturas almacenadas.

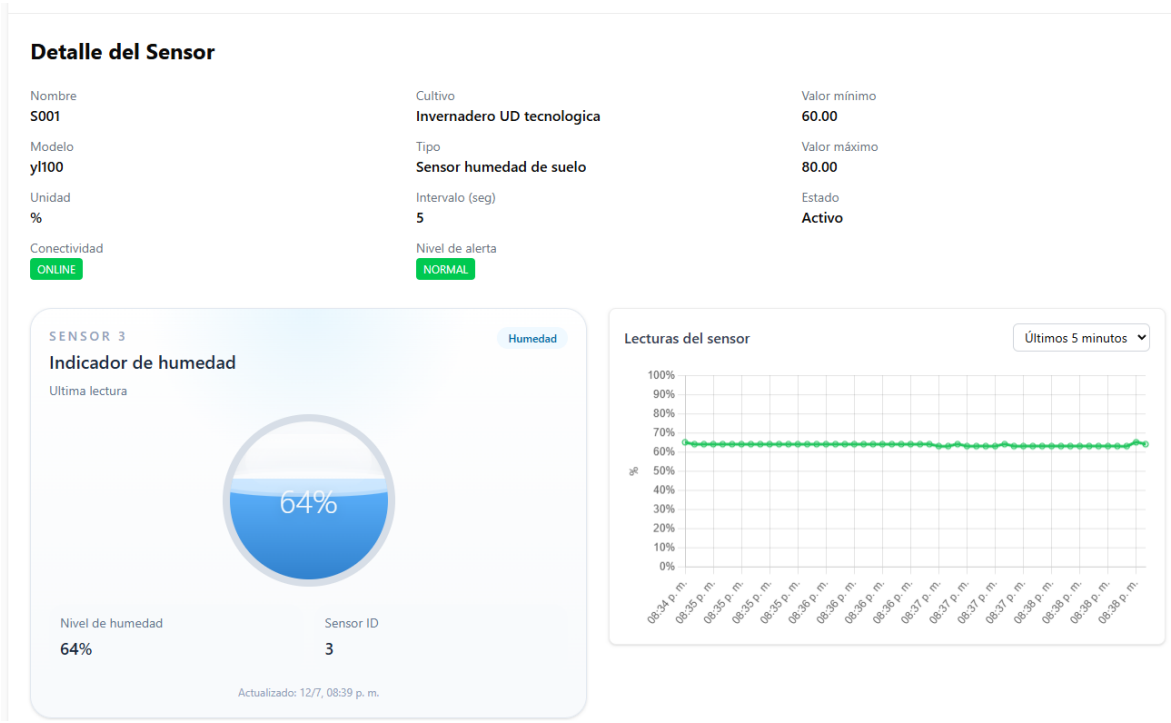


Figura 45. Visualización de información del sensor en el sitio web desarrollado. Fuente: Elaboración propia

Los valores de la humedad del suelo se muestran en la interfaz en tiempo real, esto quiere decir que, cada vez que el backend recibe una nueva lectura desde el sensor, esta es representa automáticamente e inmediatamente en la interfaz de la plataforma web, esto se logra gracias a la tecnología MQTT y WebSockets implementadas.

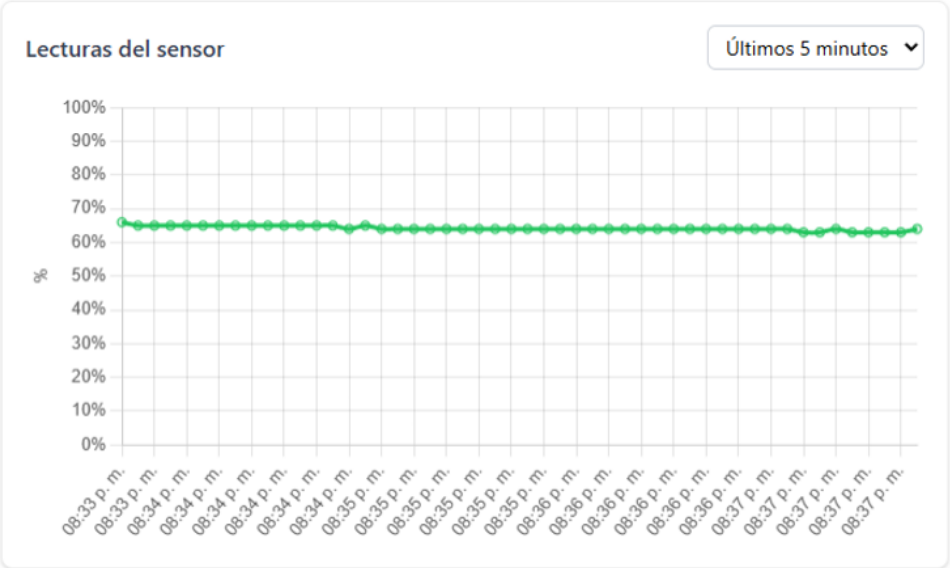


Figura 46. Gráfica de lecturas con humedad media del suelo. Fuente: Elaboración propia

La gráfica de líneas implementada en la interfaz web permite al agricultor o usuario final visualizar el histórico de las lecturas recopiladas por el sensor. Esto facilita el análisis sobre el comportamiento en nivel de humedad del suelo al pasar de tiempo y permite identificar tendencias o cambios significativos en las condiciones del cultivo. Además, la gráfica ofrece la posibilidad de seleccionar diferentes intervalos de visualización, que abarcan desde los últimos cinco minutos hasta una hora, un día o un mes, adaptándose a las necesidades de monitoreo del usuario.



Últimas lecturas

Id	Valor	Unidad	Fecha
12335	64.00	%	hace 1 seg
12334	64.00	%	hace 6 seg
12333	64.00	%	hace 11 seg
12332	64.00	%	hace 16 seg
12331	64.00	%	hace 21 seg
12330	64.00	%	hace 26 seg
12329	64.00	%	hace 31 seg
12328	64.00	%	hace 36 seg
12327	64.00	%	hace 41 seg
12326	65.00	%	hace 46 seg

Figura 47. Histórico de lecturas obtenidas por el sensor.
Fuente: Elaboración propia

Como resultado de la prueba realizada, se observa un nivel de humedad del suelo en la tierra de la maceta de un 64%, lo cual representa un valor dentro del rango óptimo de operación previamente establecido, este valor se interpreta como un nivel de humedad adecuado para la o las plantas y por lo tanto el sistema no genera ninguna alerta ni notificación al usuario.

Además, se evidencia que todas las lecturas permanecen estables en torno al mismo valor, con variaciones de tan solo 1% entre una lectura y otra, asimismo, la diferencia entre el valor máximo y el mínimo registrados, representada por ΔH, fue de un 3%, lo que demuestra la consistencia de las mediciones obtenidas por el prototipo.

$$\Delta H = H_{max} - H_{min}$$

$$\Delta H = 66\% - 63\% = 3\%$$

7.5.2. Prueba con un nivel alto de humedad del suelo y notificaciones.

Para esta prueba se siguió el mismo procedimiento de la prueba anterior, con la diferencia que se utilizó una maceta con tierra que presentaba un mayor nivel de carga de agua, justo después de recibir un riego controlado. Se aseguró que el nivel de humedad sea lo suficientemente alto como para que el sistema genere una alerta por humedad alta.



Figura 48. Maceta con alto nivel de humedad del suelo.
Fuente: Elaboración propia

Las lecturas obtenidas durante un periodo de 10 minutos se representan en la figura 49. Se evidencia un valor en el nivel de humedad en el suelo de un 85%, valor que supera el umbral máximo configurado, por lo cual el sistema genera una alerta y muestra automáticamente un indicador en color rojo, con el mensaje “HUMEDAD ALTA”.



Figura 49. Captura de valores de humedad del suelo alta.
Fuente: Elaboración propia

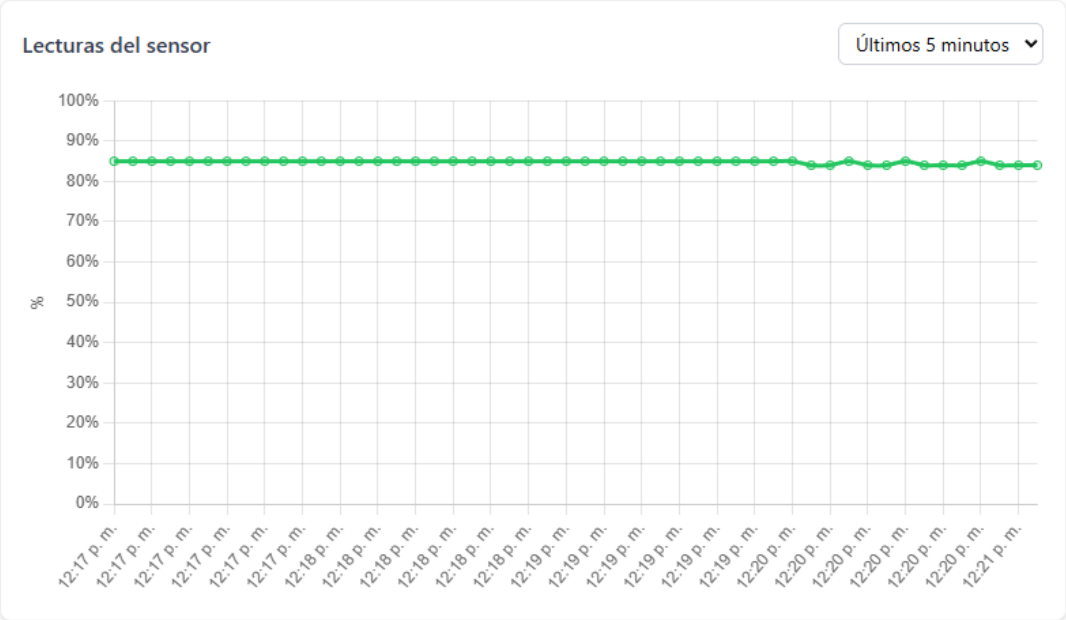


Figura 50. Gráfica de lecturas con humedad alta del suelo.
Fuente: Elaboración propia

Además, se realizó la verificación en la cuenta de correo electrónico configurado para recibir las notificaciones del usuario, y se evidencio que el sistema envía correctamente un mensaje con los detalles de la alerta por humedad alta en el suelo, confirmando que la integración con la plataforma de envío de correo electrónico funciona correctamente.

La generación de la alerta, así como el envío del correo electrónico y del mensaje SMS, se realiza automáticamente cuando el sistema detecta las primeras lecturas que superan los umbrales configurados.

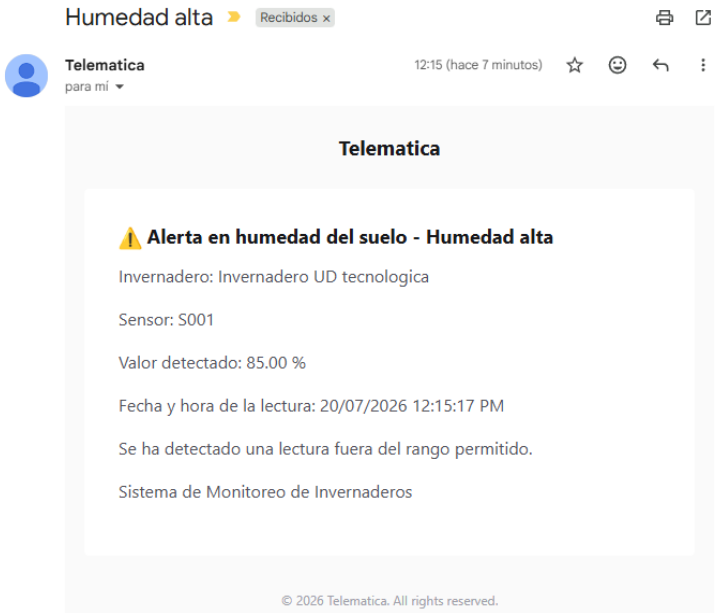


Figura 51. Correo de notificación por humedad del suelo alta.
Fuente: Elaboración propia

También se evidencia el correcto envío de la notificación al número de teléfono configurado mediante un mensaje SMS, confirmando la integración exitosa del servicio de mensajería con el sistema.



Figura 52. Mensaje de notificación SMS por humedad alta.
Fuente: Elaboración propia

7.5.3. Prueba con nivel bajo de humedad del suelo.

Esta prueba se realizó con una maceta con tierra seca, en condiciones de humedad extremadamente baja y que podrían comprometer el desarrollo de la mayoría de plantas o cultivos. Esto se hizo con el fin de evaluar el comportamiento del sistema ante un escenario en condiciones extremas de falta de humedad en el suelo y verificar así, la capacidad del sensor de detectar humedades mínimas en estas condiciones.



Figura 53. Maceta con bajo nivel de humedad del suelo.
Fuente: Elaboración propia

En esta prueba, las lecturas registradas por el sensor indicaron un nivel de humedad del suelo del 0 %. Este valor está por debajo del límite inferior de la escala configurada en el sensor y refleja que el suelo presenta un nivel de humedad extremadamente bajo, insuficiente para que el sensor detecte una cantidad mínima de humedad en la tierra.

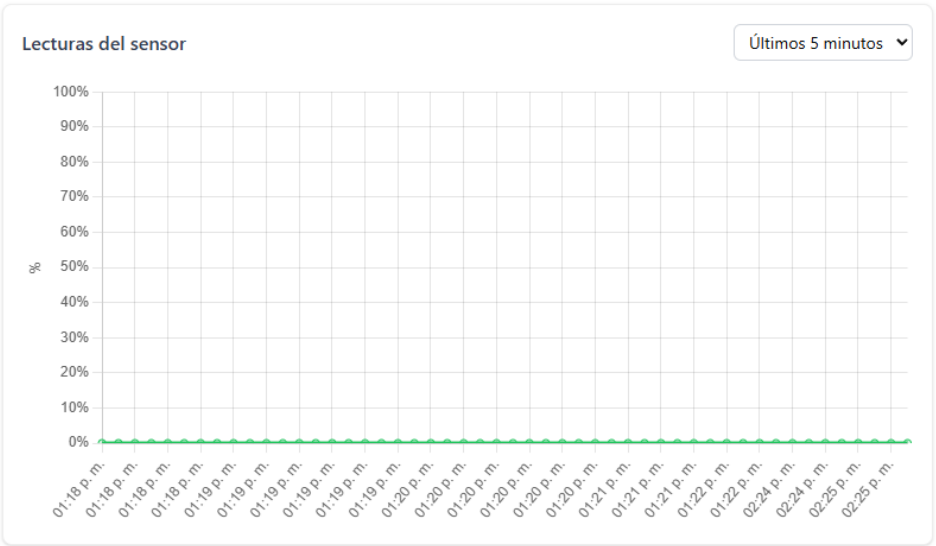


Figura 54. Gráfica de lecturas con humedad baja del suelo.
Fuente: Elaboración propia

En esta prueba, también se verificó que el sistema registra correctamente la alerta en la base de datos, la cual puede ser consultada en cualquier otro momento por el usuario correspondiente.

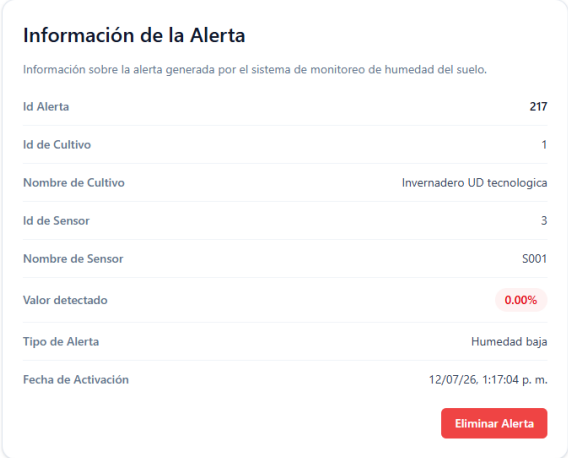


Figura 55. Interfaz visualización de alertas generadas desde el sitio web.
Fuente: Elaboración propia

7.5.4. Prueba de variación progresiva de la humedad del suelo

En esta prueba se busca evaluar el comportamiento del sistema y la variación de las lecturas de humedad del suelo a medida que se realizaron riegos controlados durante un período de tiempo determinado. Para ello, se tomó la maceta con tierra completamente seca y se realizaron varios riegos controlados con un atomizador durante una hora, con el propósito de verificar que las lecturas registradas por el sensor reflejaran correctamente el incremento de la humedad y que dicha información fuera transmitida, procesada y visualizada adecuadamente en el sistema web.

Inicialmente, se hicieron dos riegos con el atomizador con una ventana de tiempo de dos minutos, en la siguiente gráfica se evidencia el comportamiento de las lecturas generadas.

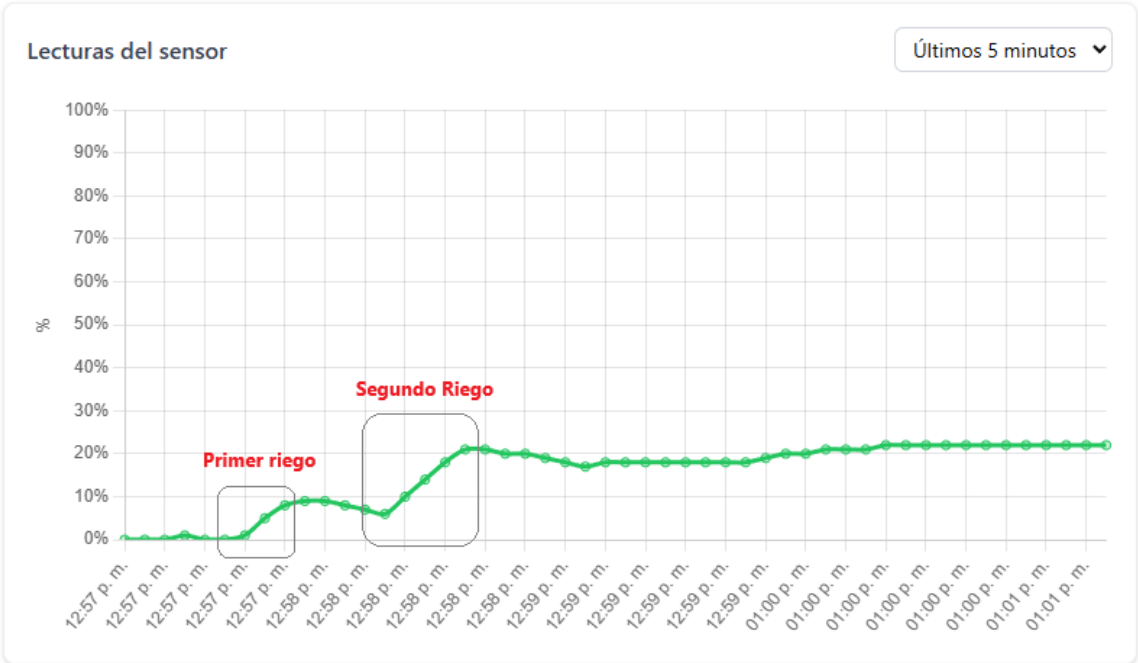


Figura 56. Gráfica de lecturas tomadas durante 5 minutos.
Fuente: Elaboración propia

Se observa que, inicialmente el sistema registra un nivel de humedad del suelo de 0%, posteriormente, al realizar el primer riego, la humedad del suelo detectada sube a 9% con una leve caída en los valores durante el siguiente minuto, esto puede atribuirse a la rápida infiltración y absorción del agua por parte de la tierra que inicialmente se encontraba seca. Después de aplicar el segundo riego el nivel de la humedad del suelo aumenta hasta 20% aproximadamente, manteniéndose en rangos similares durante los siguientes minutos.

Posteriormente, se realizaron cuatro riegos más sobre la maceta, separados por intervalos de entre 10 y 15 minutos. En la siguiente gráfica se observa el comportamiento de la humedad del suelo a lo largo de una hora, periodo en el cual se hicieron un total de seis riegos con el atomizador.

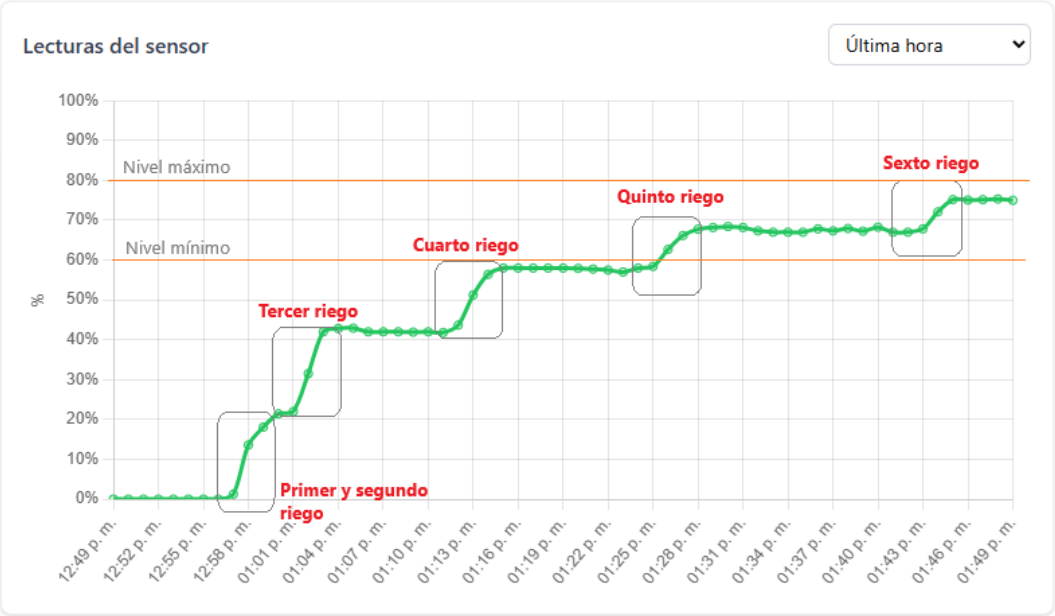


Figura 57. Gráfica de lecturas tomadas durante una hora.
Fuente: Elaboración propia

Se evidencia que, a medida que se realizaron los riegos, el nivel de humedad del suelo aumentó progresivamente hasta alcanzar un determinado valor. Posteriormente, las lecturas permanecieron estables hasta la aplicación del siguiente riego. Adicionalmente, se observa que el umbral mínimo de humedad configurado en el sistema se alcanzó después de la aplicación del quinto riego, a partir de ese momento, las lecturas permanecieron dentro del rango óptimo, por lo que el sistema dejó de generar alertas por baja humedad.

Una vez concluidas las pruebas, se evidenció el correcto funcionamiento de los diferentes elementos que componen el sistema, así como de su funcionamiento integral en un entorno simulado de un invernadero. Los resultados obtenidos evidencian un comportamiento consistente y estable del sistema telemático durante las diferentes pruebas realizadas. Las lecturas registradas por el sensor muestran poca variabilidad bajo condiciones de humedad constantes y responden correctamente ante variaciones de humedad del suelo. Además, los datos fueron procesados y representados correctamente en la interfaz web, según lo esperado, mientras que el sistema de alertas y notificaciones respondió de manera oportuna ante las condiciones de humedad fuera de los rangos establecidos.

8. Conclusiones

Se logró desarrollar e implementar un sistema telemático para la monitorización de la humedad del suelo en tiempo real utilizando sensores y desarrollando una plataforma web para la visualización de los datos recopilados. El sistema permite obtener información en tiempo real para su análisis y de esta manera apoyar la toma de decisiones en el momento de hacer el riego en cultivos dentro de invernaderos, cumplimento con los objetivos planteados para el proyecto.

El prototipo implementado demostró ser efectivo y funcional para capturar las lecturas de humedad del suelo y transmitirlas a un servidor para su procesamiento y almacenamiento. La utilización del microcontrolador ESP32 fue fundamental gracias a la conectividad Wi-Fi que integra, lo que permitió transmitir la información hacia un servidor mediante internet, así mismo, el sensor de humedad del suelo YL-100 respondió de manera correcta frente a diversas condiciones en la humedad de la tierra en las pruebas realizadas.

Se diseñó una arquitectura para el sistema telemático mediante la elaboración de un esquema, estableciendo los principales elementos y sus interacciones, lo que garantizó un correcto funcionamiento del sistema en conjunto en la etapa de implementación. La utilización del protocolo MQTT facilitó la comunicación entre el prototipo y el servidor, y junto con la integración con WebSockets se pudo transmitir y representar la información de las lecturas en la interfaz web en tiempo real.

Se desarrolló exitosamente una aplicación web estableciendo e implementando los requerimientos necesarios para garantizar la operatividad necesaria para el sistema. La utilización de frameworks facilitó el proceso de desarrollo de la aplicación web, para el backend se empleó Laravel, mientras que para el frontend se utilizó Vue. El sistema web permite visualizar el estado de los sensores y sus lecturas, mediante la representación de la información con gráficas y tablas. Además, la aplicación fue desplegada correctamente en un entorno de producción, utilizando un servidor VPS con las configuraciones y especificaciones técnicas mínimas para el funcionamiento del sistema, siendo accesible desde la IP pública del servidor o mediante el dominio configurado a través de la web.

Los resultados obtenidos durante las pruebas realizadas en los entornos simulados muestran como el sistema responde de manera acertada ante las diferentes condiciones de humedad que hay en el suelo. En especial los datos obtenidos en las pruebas de alta y baja humedad se pueden evidenciar que las lecturas presentan los respectivos cambios realizados en los riegos, en donde el sistema identifica los valores que están fuera del rango configurado permitido de la humedad, generando alertas en el sistema donde el usuario al darse cuenta toma decisiones que contribuyan en la estabilidad de la humedad en el cultivo por medio del riego.

9. Recomendaciones y trabajos futuros

Implementar mecanismos de almacenamiento local ya que el funcionamiento actual depende de la conexión por Wi-Fi, por lo cual se recomienda incorporar un almacenamiento adicional en el microcomponente ESP32 para conservar las lecturas que se realicen ante una posible interrupción en la conectividad y al retomarla posteriormente, transmitirlas nuevamente.

Automatizar el sistema de riego por medio de integrar más componentes como bombas de agua y electroválvulas que contribuyan el realizar el riego de manera automática en base a los niveles de humedad detectados por los sensores, de esta forma el sistema podría evolucionar a un control de riego automático monitorizado enfocado en la toma de decisiones para un riego más eficiente en los cultivos.

Implementar en el sistema una toma de decisión inteligente por medio de técnicas de inteligencia artificial o aprendizaje automático que puedan analizar y procesar las mediciones obtenidas y tomar decisiones convenientes con respecto al proceso de riego del cultivo tomando como base el comportamiento histórico y sus condiciones ambientales obtenidas.

10. Referencias bibliográficas

Deemer, P., Benefield, G., Larman, C., y Vodde, B. (2009). Información básica de Scrum (The Scrum Primer) (Versión 1.1; L. Antoli, trad.). ScrumTI. https://facialix.com/wp-content/uploads/2022/04/Informacion_Basica_Scrum.pdf

Interpolados. (2017, 1 de marzo). Conjunto del protocolo TCP/IP. https://interpolados.wordpress.com/2017/03/01/conjunto-del-protocolo-tcpip/

Cytron Technologies. (2019, octubre 4). Water notification for plants using Blynk and moisture sensor on ESP32. https://www.cytron.io/tutorial/water-notification-for-plants-using-blynk-and-moisture-sensor-on-esp32

eG Innovations. (s. f.). Mosquitto MQTT. https://www.eginnovations.com/glossary/mosquitto_mqtt

Monocubed. (2025, octubre 15). Advantages of Laravel framework. https://www.monocubed.com/blog/advantages-of-laravel-framework/

Thomson, P. J. (2020, septiembre). Laravel, Inertia, Livewire. https://www.peterjthomson.com/2020/09/laravel-inertia-livewire/

Martínez, Juan Pablo, Farías, Karen, Ruiz, Rubén (2020). Importancia del control de variables ambientales en invernaderos para la producción de hortalizas [en línea]. La Cruz: Informativo INIA La Cruz. no. 36. Disponible en: https://hdl.handle.net/20.500.14001/67176

Rincón Vieda, P. A., Plazas, J. A. S., & Torres Camacho, A. F. (2018). Automatización de invernadero para producción agrícola con tecnología de punta a bajo costo. Revista De Investigaciones Agroempresariales, 3, 9–23. https://doi.org/10.23850/25004468.1419

Redagrícola. (2023, 15 mayo). *Nivel tecnológico y desarrollo de los invernaderos en Colombia*. https://redagricola.com/nivel-tecnologico-y-desarrollo-de-los-invernaderos-en-colombia

Proain. (2020). Frecuencia y tiempos de riego para los cultivos agrícolas. Proain. https://proain.com/blogs/notas-tecnicas/frecuencia-y-tiempos-de-riego-para-los-cultivos-agricolas

Antonio, A. V. J. (2022, 30 agosto). Diseño e implementación de un sistema para el monitoreo de variables agroclimáticas (temperatura, humedad, brillo solar) y control remoto de riego con modulo esp8266 en cultivos cítricos en la región caribe. Universidad Antonio Nariño https://repositorio.uan.edu.co/items/fe372667-1ae0-4c66-9339-d7c3725b6819

Gaviria Cuevas, C. A. (2023). Sistema IoT aplicado en la gestión de riego en cultivos de papa mediante sensado y control remoto. Pontificia Universidad Javeriana. https://repository.javeriana.edu.co/bitstream/handle/10554/67475/attachment_0_Informe-Final_TG-2110-4_Maestria_IoT_Cesar-Gaviria.pdf?sequence=1&isAllowed=y

Camilo, A. P. A., & Gerardo, C. R. D. (2023). Implementación, instrumentación, monitoreo y control de un invernadero a escala. Universidad Distrital. https://repository.udistrital.edu.co/items/dbd6924f-97fc-4bb4-b004-710488998228

Zapata, E. D. Q. (2020). Sistema de monitoreo de humedad de suelo para apoyo en toma de decisiones del riego tecnificado de pitahaya en Mochumí. Universidad Nacional Pedro Ruiz Gallo. https://repositorio.unprg.edu.pe/handle/20.500.12893/10754

Martínez, E. (2010). Diseño e implementación de un sistema de monitoreo de humedad de suelo para cultivos utilizando tecnología de telemetría. Escuela Politécnica Nacional. https://bibdigital.epn.edu.ec/bitstream/15000/1308/1/CD-2028.pdf

Martins, J. (2025, 15 febrero). *Scrum: conceptos clave y cómo se aplica en la gestión de proyectos.* Asana. https://asana.com/es/resources/what-is-scrum

Britannica Editors (2025, abril 5). *telemetry*. *Encyclopedia Britannica*. https://www.britannica.com/technology/telemetry

IBM. (s. f.). *What is the Internet of Things (IoT)?* IBM Think. https://www.ibm.com/think/topics/internet-of-things

Amazon Web Services. (s.f.). *¿Qué es MQTT?* AWS. https://aws.amazon.com/es/what-is/mqtt/

Mozilla Contributors. (2025, 7 julio). *HTTP*. MDN Web Docs. https://developer.mozilla.org/es/docs/Web/HTTP

Sánchez Hernández, J. J. (2025). *Introducción al protocolo HTTP*. Despliegue de Aplicaciones Web. https://josejuansanchez.org/daw/introduccion_http/index.pdf

Arizbé, K*. (2023, 4 de octubre). Backend: ¿Qué es y para qué sirve? Gluo. https://www.gluo.mx/blog/backend-que-es-y-para-que-sirve*

IONOS Editorial Team. (2024, 7 de junio). *¿Qué es el Frontend?* IONOS DigitalGuide. https://www.ionos.com/es-us/digitalguide/paginas-web/creacion-de-paginas-web/que-es-el-frontend/

IONOS Editorial Team. (2020, 7 de agosto). WebSocket: Un canal de comunicación para la web en tiempo real. IONOS Digital Guide. https://www.ionos.com/es-us/digitalguide/paginas-web/desarrollo-web/que-es-websocket/

Cherlinka, V. (2025, 25 septiembre). Humedad del suelo: cómo medir y controlar su nivel. *EOS Data Analytics*. https://eos.com/es/blog/humedad-del-suelo/