



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería  
Informática**

**Diseño de un sistema  
económico IoT de  
monitorización de  
invernaderos de cannabis  
medicinal**



Presentado por José Luis Caballero  
Martínez-Quintanilla  
en Universidad de Burgos — 16 de febrero de  
2024

Tutor: Alejandro Merino Gómez  
Tutor: Carlos Cambra Baseca







UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



D. Alejandro Merino Gómez, profesor del departamento de Digitalización, área de Systems Engineering and Automation y D. Carlos Cambra Baseca, profesor del departamento de Digitalización del área de Computer Science and Artificial Intelligence,

Exponen:

Que el alumno D. José Luis Caballero Martínez-Quintanilla, con DNI 48471169-A, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Diseño de un sistema económico IoT de monitorización de invernaderos de cannabis medicinal.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 16 de febrero de 2024

Vº. Bº. del Tutor:

D. Merino Gómez, Alejandro

Vº. Bº. del co-tutor:

D. Cambra Baseca, Carlos





## Resumen

El presente trabajo de fin de grado aborda el diseño de un sistema económico basado en el Internet de las cosas (IoT) para la monitorización de invernaderos destinados al cultivo de cannabis medicinal. El objetivo principal es mejorar la eficiencia y la calidad del cultivo mediante la implementación de sensores y dispositivos conectados que permitan la recolección y análisis de datos en tiempo real.

El hardware seleccionado para este proyecto incluye la Raspberry Pi Pico W como unidad central, una pantalla OLED 128x64 para la visualización de información, el sensor DHT22 para la medición de la temperatura y humedad ambiente, el sensor BH1750 para evaluar la intensidad lumínica, y un sensor de humedad de suelo para monitorear las condiciones de la tierra.

A lo largo del trabajo, se detalla el proceso de integración de estos componentes, se describen las tecnologías utilizadas para la comunicación y el manejo de datos, y se presenta la interfaz de usuario diseñada para la visualización de información relevante. Se destacan también las consideraciones económicas que han llevado a la elección de cada componente, buscando una solución asequible sin comprometer la calidad de los resultados.

Los resultados obtenidos demuestran la viabilidad y eficacia del sistema propuesto, ofreciendo a los agricultores de cannabis medicinal un instrumento práctico y accesible para mejorar la gestión de sus invernaderos. Este trabajo contribuye al campo emergente de la agricultura inteligente y sostenible, abriendo posibilidades para futuras investigaciones y aplicaciones en el ámbito de la monitorización agrícola basada en IoT.

## Descriptores

Raspberry Pi Pico W, Micropython, Autónomo, Sistema Domótico, Bot, Telegram, Python ...

## Abstract

This project deals with the design of an economic system based on the Internet of Things (IoT) for the monitoring of greenhouses for the cultivation of medical cannabis. The main objective is to improve the efficiency and quality of the crop through the implementation of sensors and connected devices that allow the collection and analysis of data in real time.

The hardware selected for this project includes the Raspberry Pi Pico W as the central unit, a 128x64 OLED screen for displaying information, the DHT22 sensor for measuring ambient temperature and humidity, the BH1750 sensor for evaluating light intensity, and a soil moisture sensor for monitoring soil conditions.

Throughout the paper, the integration process of these components is detailed, the technologies used for communication and data management are described, and the user interface designed for the visualization of relevant information is presented. It also highlights the economic considerations that led to the choice of each component, seeking an affordable solution without compromising the quality of the results.

The results obtained demonstrate the feasibility and effectiveness of the proposed system, offering medical cannabis farmers a practical and accessible tool to improve the management of their greenhouses. This work contributes to the emerging field of smart and sustainable agriculture, opening possibilities for future research and applications in the field of IoT-based agricultural monitoring.

## Keywords

Raspberry Pi Pico W, Micropython, Autonomous, Domotic System, Bot, Telegram, Python ...

---

# Índice general

---

Índice general	iii
Índice de figuras	iv
Índice de tablas	v
<b>5. Aspectos relevantes del desarrollo del proyecto</b>	<b>1</b>
5.1. Motivación del proyecto . . . . .	1
5.2. Formación necesaria . . . . .	2
5.3. Metodología . . . . .	3
5.4. Desarrollo del proyecto . . . . .	8
<b>Bibliografía</b>	<b>19</b>

---

# Índice de figuras

---

5.1.	Pantalla oled conectada a la Raspberry Pi Pico W. . . . .	8
5.2.	Alertas del bot de Telegram. . . . .	9
5.3.	Servicios instalados y operativos en el servidor LAMP. . . . .	10
5.4.	Se ha implementado la librería umqtt [21] en la Raspberry Pi Pico W. . . . .	10
5.5.	Data almacenada en la base de datos. . . . .	11
5.6.	Consulta a la base de datos mediante comandos usando Telegram. . . . .	11
5.7.	Aplicación de escritorio llamado InverIoT. . . . .	12
5.8.	Página Web InverIoT desarrollada con Node.js. . . . .	13
5.9.	Conexiones. . . . .	14
5.10.	Aplicación de escritorio <b>InverIoT</b> . . . . .	15
5.11.	Historial de los datos. . . . .	16
5.12.	Gráfica mostrando los datos en un intervalo de fechas. . . . .	16
5.13.	Intensidad de luz superando los umbrales. . . . .	17
5.14.	Vista del historial: al cargar, muestra los datos del día actual. . . . .	17
5.15.	MQTT con Node.js para almacenar datos de los sensores. . . . .	18

---

## **Índice de tablas**

---

5.1. Características del servidor HP ProLiant. . . . .	9
5.2. Umbrales ideales para un invernadero de cannabis medicinal. . . .	13



---

## **5. Aspectos relevantes del desarrollo del proyecto**

---

En esta sección, se destacarán los elementos más significativos del desarrollo, proporcionando una perspectiva detallada de las decisiones adoptadas para alcanzar los objetivos del proyecto. Se resumirá la experiencia práctica, detallando cómo se abordaron los desafíos específicos, y se evaluará la importancia de estas soluciones en el contexto general del alcance del proyecto.

### **5.1. Motivación del proyecto**

La motivación subyacente a este proyecto surge de la creciente importancia y demanda en el ámbito de la monitorización de invernaderos, especialmente en el contexto del cultivo de cannabis medicinal. La necesidad de implementar soluciones tecnológicas eficientes para garantizar condiciones óptimas de cultivo y maximizar la producción ha impulsado la elección de este tema.

El objetivo principal radica en diseñar un sistema económico y eficaz que permita a los cultivadores de cannabis medicinal supervisar las condiciones ambientales de sus invernaderos de manera remota. Este proyecto busca proporcionar una herramienta valiosa para mejorar la eficiencia, la calidad y la consistencia en la producción de cannabis medicinal, al tiempo que se abordan los desafíos específicos asociados con la monitorización de variables críticas como temperatura, humedad y luz.

## 5.2. Formación necesaria

El proyecto ha requerido una formación interdisciplinaria que abarca diversas áreas. En primer lugar, la comprensión profunda de los principios de la programación y el desarrollo de software ha sido esencial. La elección de la Raspberry Pi Pico W como plataforma y la programación en MicroPython para controlar los dispositivos hardware involucra un conocimiento sólido en programación embebida y desarrollo en entornos limitados.

Además, la utilización de sensores especializados, como el DHT22 [16], el BH1750 [6] y el sensor de humedad del suelo [4], ha demandado una comprensión detallada de los principios de operación de estos dispositivos y cómo integrar sus lecturas en el sistema general. Esto implica una formación técnica en electrónica y sensores.

La implementación de conceptos relacionados con el Internet de las cosas (IoT) y la comunicación inalámbrica mediante la Raspberry Pi Pico W [5] también ha requerido conocimientos específicos en redes y protocolos de comunicación.

La integración de MQTT [9] en el código MicroPython para la Raspberry Pi Pico W ha requerido conocimientos específicos sobre este protocolo de mensajería ligero diseñado para la eficiente comunicación entre dispositivos en redes IoT.

En resumen, la formación necesaria abarca habilidades en programación embebida, electrónica, redes, IoT y desarrollo de software. La combinación de estas competencias ha sido esencial para la ejecución exitosa del proyecto, demostrando la importancia de una formación integral y multidisciplinaria en el ámbito de la ingeniería y la tecnología.

### 5.3. Metodología

Se ha optado por la metodología del **Modelo de Prototipos** [7] debido a su capacidad para proporcionar retroalimentación temprana, clarificar requisitos, adaptarse a cambios, validar prácticamente funcionalidades, reducir riesgos, comprender la interfaz de usuario y facilitar la mejora continua. Este enfoque iterativo permite identificar y corregir problemas desde las primeras etapas, garantizando la eficiencia y el éxito en el desarrollo del sistema IoT de monitorización de invernaderos de cannabis medicinal.

#### Identificación de Requisitos Iniciales:

##### Hardware:

- Raspberry Pi Pico W [5] como la plataforma central.
- Sensores DHT22 [16] para medir temperatura y humedad ambiente.
- Sensor BH1750 [6] para evaluar la intensidad de luz.
- Sensor de humedad de suelo [4] para monitorizar las condiciones del suelo.
- Pantalla OLED [22] para visualización de datos.
- LEDs RGB [2] para indicar condiciones críticas.

##### Software:

- Desarrollo de un sistema en MicroPython para la Raspberry Pi Pico W 5.4.
- Implementación de un servidor LAMP para almacenar y gestionar datos.
- Integración de la API de Telegram [19] para recibir alertas.
- Aplicación de escritorio para Windows 5.4, desarrollada en Visual Studio 2022 en lenguaje C#. Incluye su instalador.
- Dashboard web 5.4, realizado con Node.js, accesible desde internet.

##### Funcionalidades clave:

- Captura de datos ambientales como temperatura, humedad, luz y humedad del suelo.
- Representación gráfica de los datos en una pantalla OLED.
- Envío de alertas a través de Telegram en caso de valores fuera de los umbrales ideales.
- Interacción bidireccional para activar mecanismos de respuesta a condiciones ambientales adversas.

### **Desarrollo del Primer Prototipo:**

En esta etapa, se enfocó en la integración de los componentes hardware definidos, como la Raspberry Pi Pico W [5] y los sensores de humedad, temperatura [16], luminosidad [6] y humedad del suelo [4]. Se estableció la conexión con el servidor LAMP utilizando MQTT [9] para la transmisión de datos en tiempo real.

Durante la fase de desarrollo, se configuraron y probaron los mecanismos de alerta a través del bot de Telegram y la activación de los LEDs RGB para indicar condiciones críticas. También se diseñó e implementó la lógica de umbrales para verificar si los valores ambientales están dentro de los límites ideales.

### **Evaluación del Primer Prototipo:**

El objetivo principal fue validar la efectividad de las funcionalidades implementadas y su capacidad para cumplir con los requisitos planteados en la fase inicial del proyecto.

- **Pruebas de Mecanismos de Alerta y Retroalimentación Visual:** Se realizaron pruebas del sistema de alertas a través del bot de Telegram. Se verificó la capacidad del sistema para enviar notificaciones en tiempo real sobre condiciones ambientales críticas, como temperaturas extremas o niveles de humedad fuera de los umbrales establecidos.

Además, se evaluó la respuesta de los LEDs RGB conectados, los cuales indican visualmente las condiciones del entorno. Cada color del LED se asoció con un estado específico, permitiendo una retroalimentación instantánea sobre el estado actual del invernadero.

- **Verificación de la Lógica de Umbrales:** La lógica de umbrales, diseñada para verificar si los valores ambientales se mantienen dentro de los límites ideales, está en fase de pruebas en un entorno no controlado. En este contexto, se observa que los valores experimentan salidas fuera de los umbrales establecidos. Sin embargo, se ha verificado con éxito que la lógica implementada para activar las alertas funciona de manera efectiva cuando los valores se desvían de los límites ideales predefinidos.

### **Retroalimentación y Mejoras:**

Tras la evaluación del primer prototipo, se recopiló retroalimentación valiosa para orientar mejoras futuras. Se identificaron áreas de oportunidad, como la necesidad de ajustar los umbrales para adaptarse a condiciones específicas del entorno y mejorar la precisión de las alertas. La retroalimentación del usuario sobre la interfaz y la usabilidad también se tuvo en cuenta para realizar ajustes en el diseño del dashboard web y la aplicación de escritorio. Estas observaciones se utilizarán como guía para la siguiente iteración del prototipo, buscando una mayor eficiencia y adecuación a los requisitos específicos del sistema.

### **Desarrollo de Subsiguentes Prototipos:**

Con base en la retroalimentación obtenida del primer prototipo, se inició el desarrollo de subsiguientes prototipos con el objetivo de implementar mejoras significativas. Se priorizó la optimización de los umbrales de alerta, considerando las condiciones específicas del entorno de monitoreo. También se abordaron ajustes en la interfaz del dashboard web y la aplicación de escritorio, respondiendo a las sugerencias de los usuarios para mejorar la usabilidad y la experiencia general. Cada nuevo prototipo se sometió a pruebas exhaustivas para validar las mejoras implementadas y garantizar un funcionamiento más eficiente del sistema en su conjunto. Este proceso iterativo permitió refinamientos continuos, enfocados en lograr un sistema de monitoreo más preciso y adaptado a las necesidades específicas del usuario.

### **Validación Continua:**

Se llevaron a cabo pruebas exhaustivas para verificar el funcionamiento del sistema, evaluando diversos escenarios y condiciones ambientales para confirmar la precisión de los sensores y la respuesta del sistema. Se realizaron pruebas de estrés para evaluar la capacidad de respuesta en situaciones extremas.

Durante un período prolongado, se recopilaron datos para analizar el rendimiento a lo largo del tiempo, comparándolos con los umbrales ideales establecidos. Algunos valores se salieron del rango ideal debido a las pruebas en un entorno no controlado, pero se verificó que el sistema funcionó correctamente y generó alertas de manera adecuada al encontrarse en un ambiente controlado.

La retroalimentación continua de las pruebas y la validación de los resultados se utilizaron para ajustar el sistema, mejorando su robustez y confiabilidad. Este ciclo de validación y ajuste continuo contribuyó a optimizar el sistema para lograr un monitoreo preciso y eficiente de las condiciones del invernadero.

### **Integración de Componentes:**

Durante esta etapa, se procedió a la integración de los diferentes componentes del sistema. Se conectaron y configuraron la Raspberry Pi Pico W [5] como plataforma central, junto con los sensores DHT22, BH1750 [6] y el sensor de humedad del suelo [4]. Se aseguró la comunicación efectiva entre estos elementos, permitiendo la captura precisa de datos ambientales.

El desarrollo de un sistema en MicroPython [3] para la Raspberry Pi Pico W se integró con éxito con el servidor LAMP, facilitando la gestión y almacenamiento de datos. La implementación de la API de Telegram [19] se conectó con el sistema para recibir alertas en tiempo real.

La aplicación de escritorio para Windows 5.4, desarrollada en Visual Studio 2022 con lenguaje C# [10], se integró sin problemas, permitiendo la visualización de datos en tiempo real y la interacción bidireccional con el sistema.

El dashboard web 5.4, creado con Node.js, se integró para ofrecer una interfaz accesible a través de internet, proporcionando una visualización completa y en tiempo real de los datos capturados. Esta etapa de integración aseguró la cohesión y el funcionamiento conjunto de todos los componentes del sistema IoT para el monitoreo de invernaderos.

### **Ajustes Finales:**

Estos ajustes se centraron en mejorar la precisión de los sensores, la eficacia de las alertas y la respuesta de los mecanismos de control. Se llevaron a cabo modificaciones en el código del sistema para perfeccionar la lógica

de umbrales y asegurar que los valores se mantuvieran dentro de los límites ideales.

Además, se realizaron ajustes en la configuración de la pantalla OLED y los LEDs RGB para una visualización más clara y distintiva de las condiciones del invernadero. La interfaz de usuario se mejoró para facilitar la comprensión de los datos mostrados en tiempo real. También se realizaron pruebas adicionales para verificar la estabilidad del sistema en condiciones diversas.

Este proceso de ajustes finales contribuyó a la optimización global del sistema, asegurando un monitoreo preciso y una respuesta efectiva a las variaciones ambientales en el invernadero. La retroalimentación continua y la iteración fueron fundamentales en esta fase para lograr un prototipo final robusto y funcional.

### **Documentación:**

Se elaboró una documentación completa que abarca aspectos técnicos, funcionales y de implementación del sistema desarrollado. La documentación detallada incluye manuales de usuario para las distintas interfaces, descripciones técnicas de hardware y software, así como instrucciones para la instalación y configuración del sistema.

Además, se generaron documentos específicos para el mantenimiento del sistema, facilitando futuras actualizaciones y mejoras.

Esta documentación sirve como recurso integral para cualquier persona que interactúe con el sistema, desde usuarios finales hasta desarrolladores o técnicos de mantenimiento. Su elaboración garantiza la comprensión y el uso efectivo del sistema, contribuyendo a su sostenibilidad a lo largo del tiempo.

### **Implementación Final:**

La implementación final del sistema se llevó a cabo tras la validación y ajustes continuos a lo largo de las fases de desarrollo y prototipado. En esta etapa, se consolidaron todas las funcionalidades y características planificadas, integrando los componentes hardware y software de manera coherente.

Se realizó una verificación exhaustiva de la interacción entre los sensores, la Raspberry Pi Pico W, la pantalla OLED, y los LEDs RGB. Se mejoró la respuesta del sistema ante condiciones ambientales diversas y se afinaron los algoritmos de control para garantizar un monitoreo preciso y eficiente del invernadero.

Además, se implementaron las últimas medidas de seguridad y se realizaron pruebas exhaustivas para garantizar la estabilidad del sistema en distintos escenarios. Se documentaron detalladamente los pasos necesarios para replicar la implementación final, asegurando una fácil reproducción y mantenimiento.

La implementación final refleja la evolución del sistema desde su concepción hasta la etapa de producción, consolidando todas las lecciones aprendidas durante el desarrollo y prototipado.

## 5.4. Desarrollo del proyecto

El propósito inicial radicaba en la recopilación de datos ambientales, tales como humedad, temperatura, intensidad de luz y humedad del suelo, con el fin de almacenarlos y presentarlos de manera gráfica.

Una vez definida la idea del TFG y los componentes hardware, surgía la interrogante: "¿Dónde enviar los datos? ¿Cómo gestionarlos?". La primera medida fue mostrar los datos en una pantalla OLED [22] conectada a la protoboard.

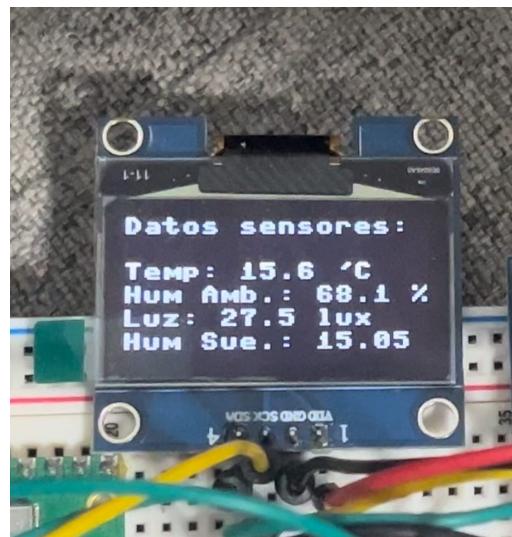


Figura 5.1: Pantalla oled conectada a la Raspberry Pi Pico W.

Sin embargo, al estar fuera de la red, surgieron nuevas incógnitas, ya que inicialmente se trató de un sistema unidireccional en el cual el sensor recolecta los datos y los muestra en la pantalla OLED.

Más tarde, se implementó la funcionalidad de enviar alertas a través de Telegram en caso de que los valores recopilados excedieran los umbrales ideales.

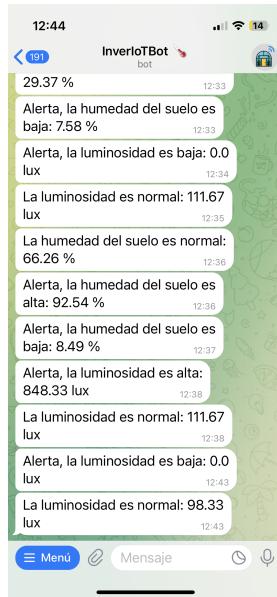
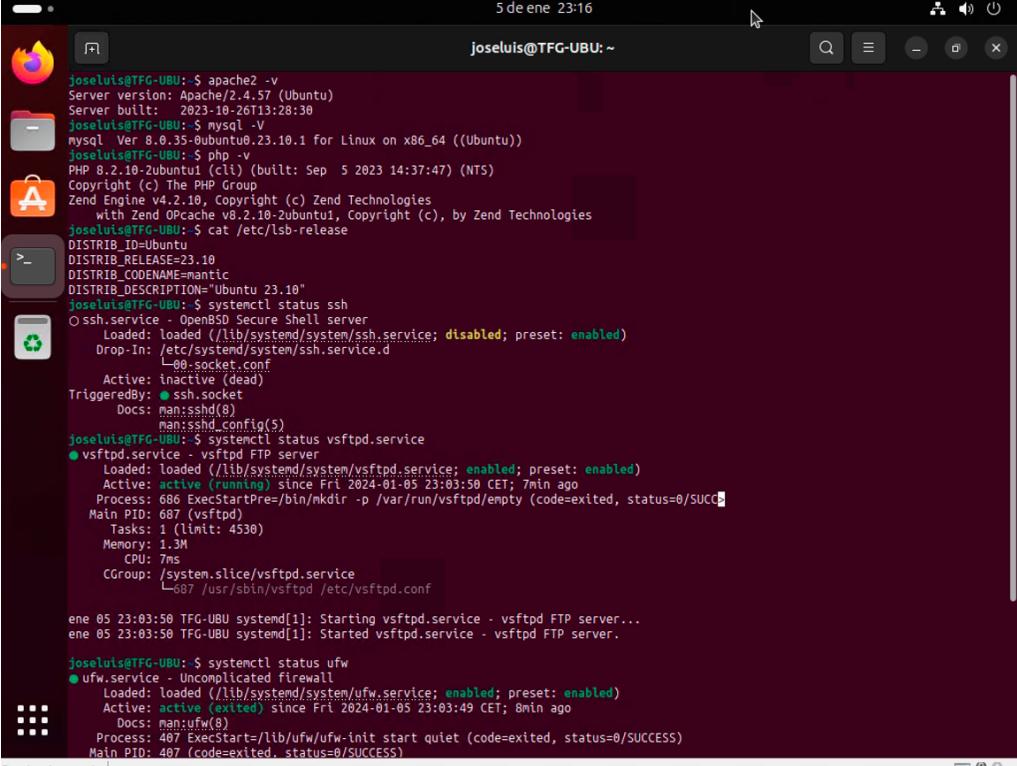


Figura 5.2: Alertas del bot de Telegram.

Posteriormente, opté por establecer un servidor LAMP para facilitar el envío y manejo de datos. La implementación de este servidor se llevó a cabo en el sistema operativo Ubuntu [20], el cual fue virtualizado utilizando Hyper-V [11]. En esta instancia, se emplea un servidor físico HP ProLiant [8] para la virtualización de los servicios Apache [1], MySQL [13], PHP [15] y SSH [17].

Tabla 5.1: Características del servidor HP ProLiant.

Característica	Descripción
Sistema operativo	Windows Server 2019 Standard
Fabricante	Hewlett Packard Enterprise
Modelo	Hewlett Packard Enterprise x64 Class PC
Procesador	Intel(R) Xeon(R) Silver 4210R CPU @ 2.4GHz 2.39GHz
Memoria instalada (RAM)	128 GB (128 GB utilizable)
Tipo de sistema	Sistema operativo de 64 bits, procesador x64



```

5 de ene 23:16
joseluis@TFG-UBU: ~

joseluis@TFG-UBU: $ apache2 -v
Server version: Apache/2.4.57 (Ubuntu)
Server built: 2023-10-26T13:28:30
joseluis@TFG-UBU: $ mysql -V
mysql Ver 8.0.35-0ubuntu0.23.10.1 for Linux on x86_64 ((Ubuntu))
joseluis@TFG-UBU: $ php -v
PHP 8.2.10-0ubuntu1 (cli) (built: Sep 5 2023 14:37:47) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.10, Copyright (c) Zend Technologies
with Zend OPcache v8.2.10-2ubuntu1, Copyright (c), by Zend Technologies
joseluis@TFG-UBU: $ cat /etc/lsb-release
DISTRIB_ID=ubuntu
DISTRIB_RELEASE=23.10
DISTRIB_CODENAME=natty
DISTRIB_DESCRIPTION="Ubuntu 23.10"
joseluis@TFG-UBU: $ systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; disabled; preset: enabled)
  Drop-In: /etc/systemd/system/ssh.service.d
            └─00-socket.conf
    Active: inactive (dead)
      Docs: man:sshd(8)
             man:sshd_config(5)
joseluis@TFG-UBU: $ systemctl status vsftpd.service
● vsftpd.service - vsftpd FTP server
  Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; preset: enabled)
  Active: active (running) since Fri 2024-01-05 23:03:50 CET; 7min ago
    Process: 686 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited, status=0/SUCCESS)
   Main PID: 687 (vsftpd)
      Tasks: 1 (limit: 4530)
        CPU: 7ms
       Memory: 1.3M
          CPU: 7ms
        CGroup: /system.slice/vsftpd.service
                 └─687 /usr/sbin/vsftpd /etc/vsftpd.conf

ene 05 23:03:50 TFG-UBU systemd[1]: Starting vsftpd.service - vsftpd FTP server...
ene 05 23:03:50 TFG-UBU systemd[1]: Started vsftpd.service - vsftpd FTP server.

joseluis@TFG-UBU: $ systemctl status ufw
● ufw.service - Uncomplicated firewall
  Loaded: loaded (/lib/systemd/system/ufw.service; enabled; preset: enabled)
  Active: active (exited) since Fri 2024-01-05 23:03:49 CET; 8min ago
    Docs: man:ufw(8)
  Process: 407 ExecStart=/lib/ufw/ufw-init start quiet (code=exited, status=0/SUCCESS)
   Main PID: 407 (code=exited, status=0/SUCCESS)

Estado: ejecutando [  ]

```

Figura 5.3: Servicios instalados y operativos en el servidor LAMP.

Basándome en esta información y en mi experiencia previa trabajando con **Smart Cities**, tomo la decisión de enviar la información recopilada por los sensores al servidor mediante MQTT [9].

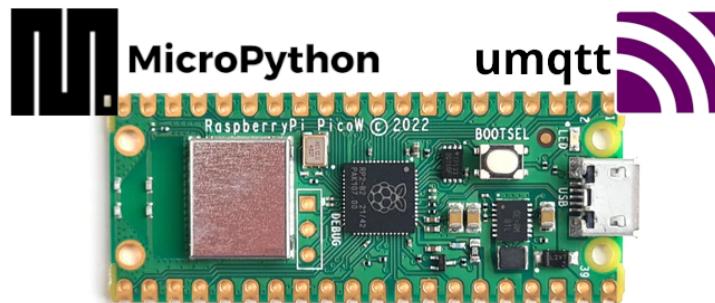


Figura 5.4: Se ha implementado la librería umqtt [21] en la Raspberry Pi Pico W.

MQTT permite la visualización en tiempo real de los datos en una página web alojada en el servidor LAMP cada vez que la placa junto con el sensor envíe datos. Además, posibilita el almacenamiento de los datos en la base de datos MySQL [13], permitiéndome mantener un historial y realizar consultas.

	<input type="checkbox"/>											
					<b>id</b>	<b>1</b>	<b>fecha</b>	<b>hora</b>	<b>temperatura</b>	<b>humedad</b>	<b>intensidad_luz</b>	<b>humedad_suelo</b>
					4118		2024-02-03	22:45:13	20.00	64.50	50.83	-1.03
					4117		2024-02-03	22:45:10	20.00	64.40	50.00	-0.67
					4116		2024-02-03	22:45:06	20.00	64.50	50.00	-0.67
					4115		2024-02-03	22:45:03	19.90	64.80	50.00	-0.55
					4114		2024-02-03	22:45:00	19.90	65.00	50.00	-0.97
					4113		2024-02-03	22:44:57	19.90	65.10	50.00	-0.97

Figura 5.5: Data almacenada en la base de datos.

Listo, he leído e incluso almacenado los datos que se envían. Además, con la implementación de un bot de Telegram [18], ahora puedo consultar en tiempo real la información proveniente de los sensores.

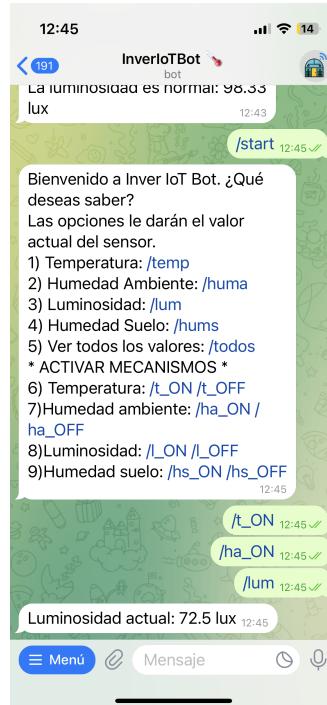


Figura 5.6: Consulta a la base de datos mediante comandos usando Telegram.

También optamos por desarrollar una aplicación de escritorio para Windows en C# [10] al que se ha llamado InverIoT 5.4, que permite la visualización en tiempo real de los datos. ¡Problema resuelto! Ahora, puedo recibir y monitorear la temperatura, luminosidad y otros valores del invernadero.

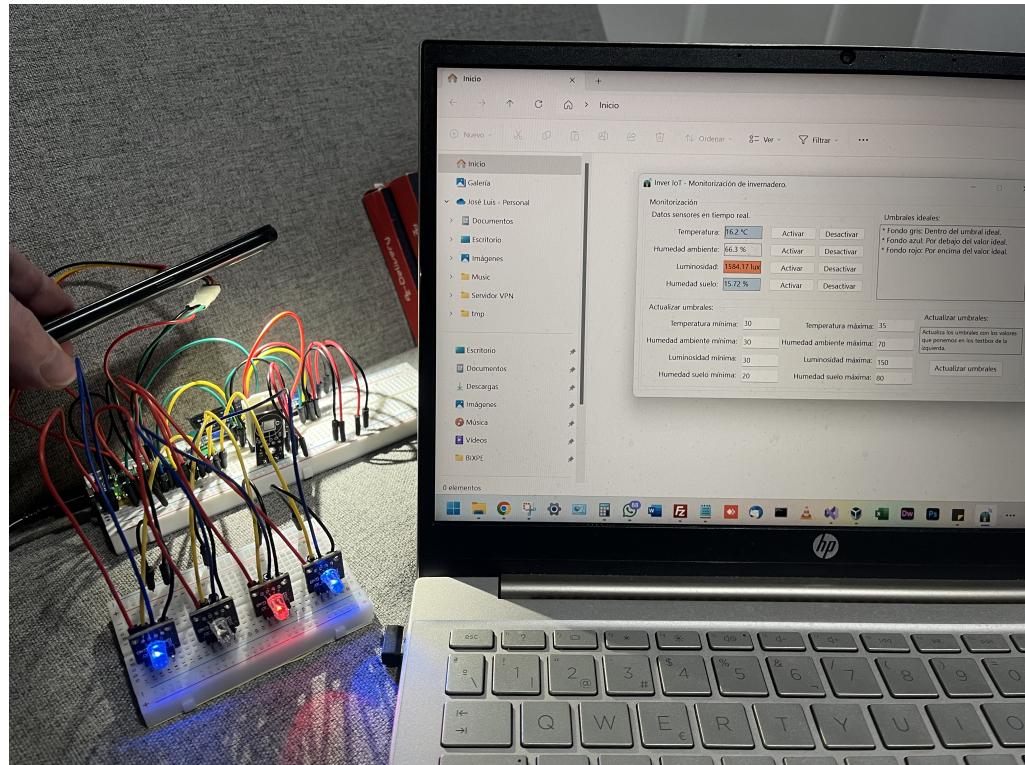


Figura 5.7: Aplicación de escritorio llamado InverIoT.

Sin embargo, al caer la noche, la luminosidad y la temperatura disminuyen, planteando un nuevo desafío. En este caso, necesitaría establecer una comunicación bidireccional con la placa Raspberry Pi Pico W para, por ejemplo, encender los focos y activar los calentadores.

Este cambio a una comunicación bidireccional se logró suscribiendo la Raspberry Pi Pico W a un topic MQTT. Ahora, puedo enviar órdenes para procesarlas según sea necesario. Sin embargo, la activación de mecanismos físicos como encender focos, activar calefacción, regar el suelo o humidificar el ambiente aún no se ha implementado, ya que corresponde a otras especialidades como telecomunicación industrial o mecánica.

El cannabis medicinal requiere condiciones ambientales constantes y dentro de un umbral específico para un crecimiento óptimo y delicado.

Definimos umbrales (valor mínimo/máximo) para mantener los valores de los sensores dentro de límites aceptables. En caso de detectar un valor anómalo fuera de estos umbrales, se genera una alerta. Para esta función, empleamos un bot de Telegram que envía mensajes de alerta, y se configura un LED RGB para cada medición (temperatura, humedad, luminosidad, humedad del suelo). Si la temperatura es baja, por ejemplo, el LED se ilumina en azul y envía una alerta por Telegram; si la temperatura es alta, el LED se enciende en rojo. Un LED apagado indica valores normales dentro del umbral.

Tabla 5.2: Umbrales ideales para un invernadero de cannabis medicinal.

Característica	Mínimo	Máximo
TEMPERATURA	30	35
HUMEDAD	30	70
LUMINOSIDAD	30	150
HUMEDAD DEL SUELO	20	80

La intervención manual también es posible, como activar ventiladores al observar un LED azul. Además, estos procesos pueden automatizarse. Por ejemplo, si la luminosidad cae por debajo del umbral, los focos se encienden hasta alcanzar el valor óptimo, por ahora eso no está implementado en este proyecto.

La información se visualiza en tiempo real en una página web desarrollada con Node.js, incluyendo un histórico de datos almacenados en la base de datos.

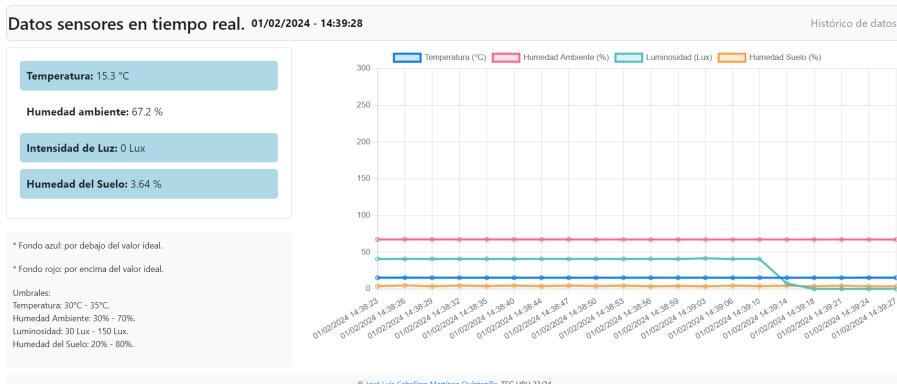


Figura 5.8: Página Web InverIoT desarrollada con Node.js.

## Hardware

- La Raspberry Pi Pico W estará situada en el invernadero para la recopilación de datos.
- Los LEDs RGB, instalados en la oficina del cliente, indicarán visualmente si los valores superan los umbrales establecidos.
- La pantalla OLED estará colocada en la puerta del invernadero para mostrar los valores actualizados.

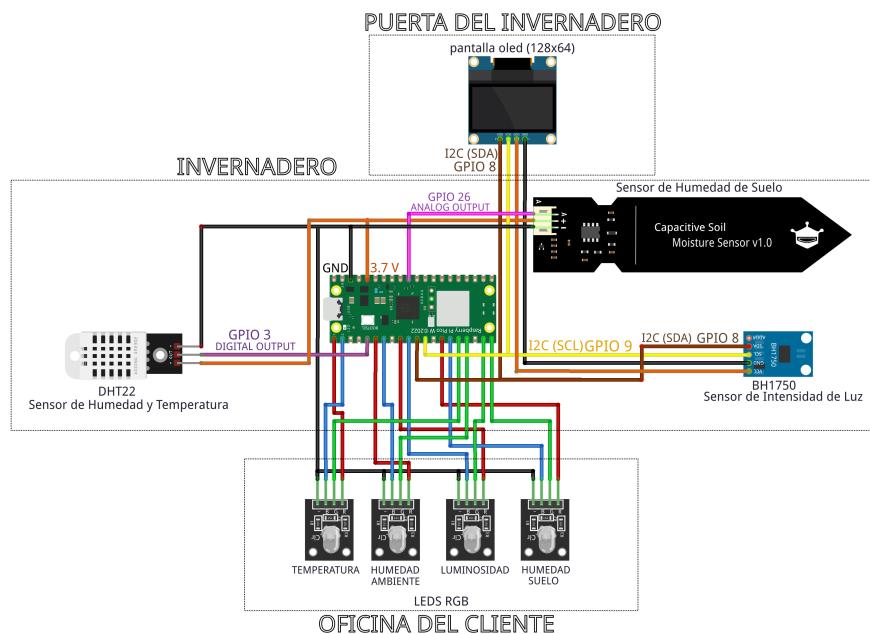


Figura 5.9: Conexiones.

## InverIoT

Se desarrolla una aplicación de escritorio para Windows llamada InverIoT, que permite al usuario monitorizar en tiempo real los datos provenientes de los sensores conectados a la placa. Utilizando el protocolo MQTT [9], la aplicación suscribe y recibe los datos publicados en el mismo tema. Estos datos son formateados y presentados en textboxes correspondientes, con unidades de medida agregadas. Se incorporan botones para activar o desactivar mecanismos, como un LED verde, que corrigen valores fuera de los umbrales ideales indicados en la parte derecha de la interfaz. Además, se añaden 8 textboxes en la parte inferior para ajustar manualmente los valores mínimos y máximos de cada parámetro. Los umbrales se actualizan desde la base de datos del servidor LAMP, proporcionando indicadores visuales de color azul para valores bajos, gris para valores normales y rojo para valores altos.

Se desarrolló la aplicación de escritorio utilizando Windows Forms y programada en lenguaje C# [10].

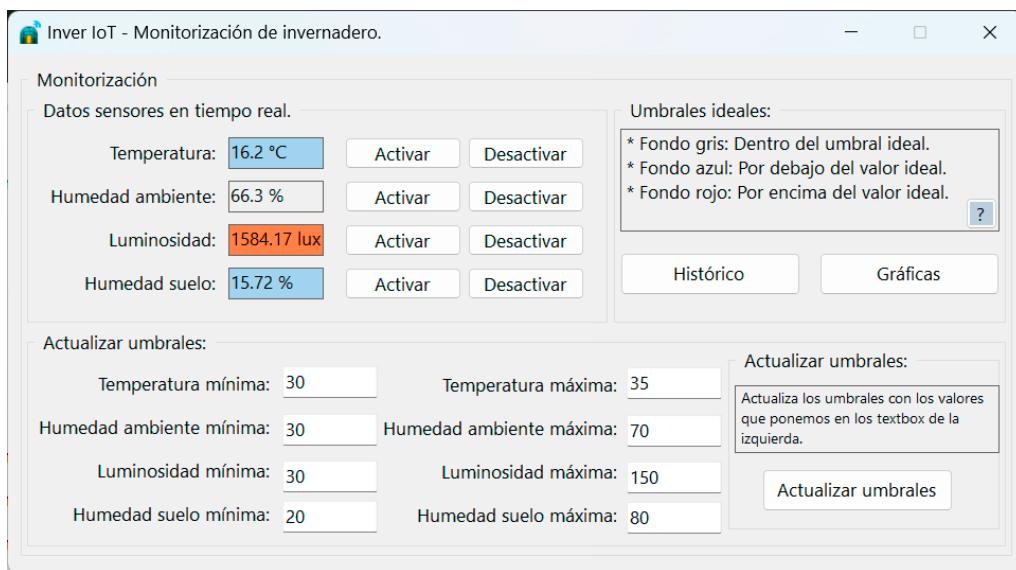


Figura 5.10: Aplicación de escritorio **InverIoT**.

	ID	Fecha	Hora	Temperatura	Humedad Ambiente	Luminosidad	Humedad Suelo
▶	2592	31/01/2024	12:26:28	15,60	68,70	20,83	3,64
	2593	31/01/2024	12:26:32	15,50	68,50	20,83	3,82
	2594	31/01/2024	12:26:36	15,50	68,40	20,83	5,22
	2595	31/01/2024	12:26:40	15,50	68,40	20,83	4,79
	2596	31/01/2024	12:26:44	15,50	68,30	21,67	4,85
	2597	31/01/2024	12:26:48	15,50	68,30	21,67	4,13
	2598	31/01/2024	12:26:52	15,50	68,30	20,83	5,22
	2599	31/01/2024	12:26:57	15,50	68,30	21,67	5,04
	2600	31/01/2024	12:27:01	15,50	68,30	21,67	4,49
	2601	31/01/2024	12:27:05	15,50	68,30	21,67	5,16
	2602	31/01/2024	12:27:09	15,50	68,30	21,67	5,46
	2603	31/01/2024	12:27:14	15,50	68,30	21,67	4,43
	2604	31/01/2024	12:27:18	15,50	68,30	21,67	5,16
	2605	31/01/2024	12:27:22	15,50	68,30	21,67	4,37
	2606	31/01/2024	12:27:26	15,50	68,30	21,67	4,13
	2607	31/01/2024	12:27:31	15,50	68,30	23,33	4,06
	2608	31/01/2024	12:27:35	15,50	68,30	27,50	4,73
	2609	31/01/2024	12:27:40	15,50	68,20	33,33	4,43
	2610	31/01/2024	12:27:45	15,50	68,20	33,33	4,55
	2611	31/01/2024	12:27:49	15,50	68,20	32,50	5,22
	2612	31/01/2024	12:27:54	15,50	68,20	33,33	4,55
	2613	31/01/2024	12:27:58	15,50	68,20	33,33	4,43

Figura 5.11: Historial de los datos.

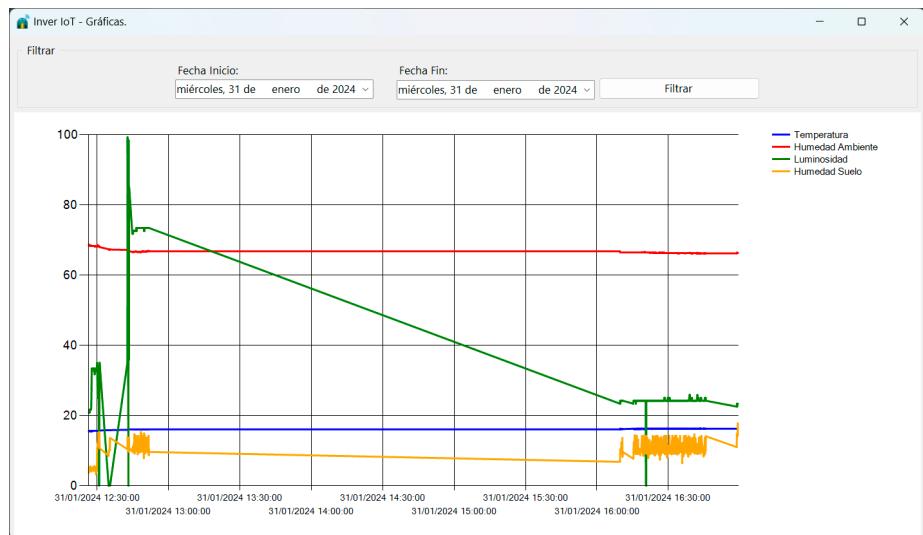


Figura 5.12: Gráfica mostrando los datos en un intervalo de fechas.

## Dashboard

Se desarrolla un dashboard que permite al usuario visualizar en tiempo real los datos capturados por los sensores, con una interfaz similar a la aplicación de escritorio. Los valores que exceden los umbrales establecidos se destacan mediante cambios de color. La plataforma incluye una gráfica en tiempo real y la capacidad de acceder a un historial con filtro de fecha. Los umbrales utilizados se extraen de la tabla **TFG\_\_UBU** en la base de datos MySQL [13].

El panel de control está disponible para su acceso a través del siguiente enlace: [InverIoT Dashboard](#)



Figura 5.13: Intensidad de luz superando los umbrales.

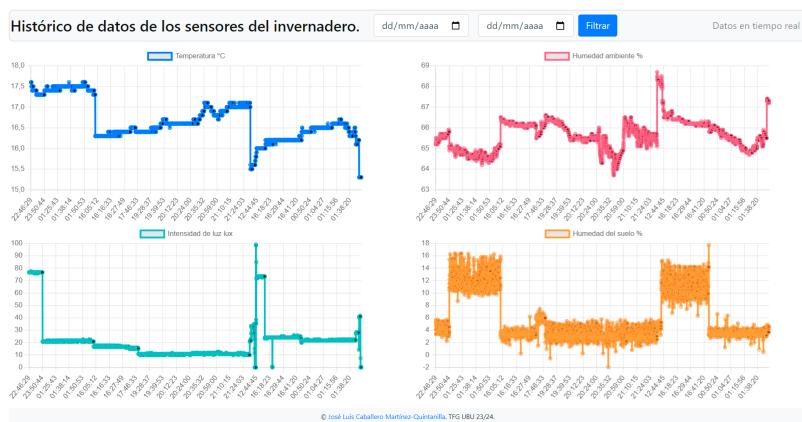


Figura 5.14: Vista del historial: al cargar, muestra los datos del día actual.

## NodeMqtt

**nodeMqtt** es un servicio en Node.js [14] que escucha los topics **invernadero/sensores** e **invernadero/umbrales**. Utiliza la librería MQTT.js [12]. Los datos son enviados por la placa Raspberry Pi Pico W RP2040 [5], que recopila valores de sensores. El servicio **nodeMqtt** captura, formatea y luego inserta estos datos en la base de datos MySQL [13] para los valores de sensores, además de actualizar los umbrales correspondientes.

Está conformado por los siguientes archivos:

- **index.js:** Es el punto de entrada del código de la aplicación.
- **package.json:** Es un archivo de configuración que describe la aplicación y sus dependencias.

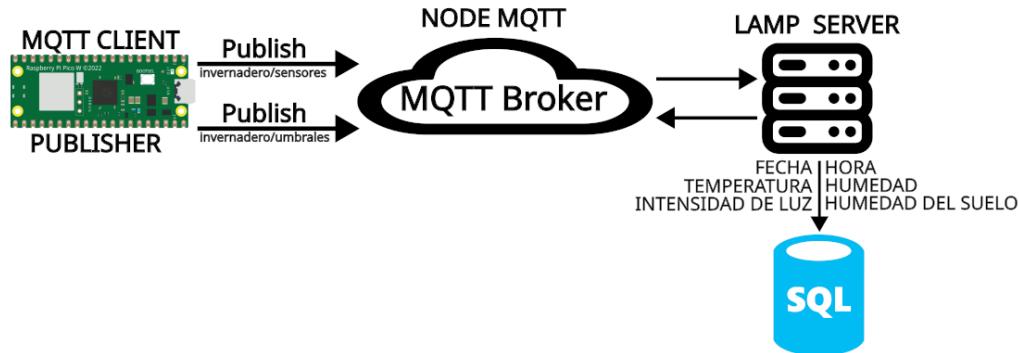


Figura 5.15: MQTT con Node.js para almacenar datos de los sensores.

---

# Bibliografía

---

- [1] Apache. Http server project. <https://httpd.apache.org/>.
- [2] datasheetspdf. *KY-016 RGB 5mm LED module.* <https://datasheetspdf.com/mobile/1402027/Joy-IT/KY-016/1>.
- [3] Equipo de Desarrollo de MicroPython. Micropython documentation, 1 2024. <https://github.com/micropython/micropython/wiki>.
- [4] DFROBOT. Capacitive soil moisture sensor, 1 2024. [https://wiki.dfrobot.com/Capacitive\\_Soil\\_Moisture\\_Sensor\\_SKU\\_SEN0193](https://wiki.dfrobot.com/Capacitive_Soil_Moisture_Sensor_SKU_SEN0193).
- [5] Raspberry Pi Foundation. Raspberry pi pico w. <https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html#raspberry-pi-pico-w>.
- [6] Handsontec. *BH1750 Ambient Light Sensor Module.* <https://www.handsontec.com/dataspecs/sensor/BH1750%20Light%20Sensor.pdf>.
- [7] Hostingplus. Modelo de prototipos: ¿qué es y cuáles son sus etapas? <https://www.hostingplus.com.es/blog/modelo-de-prototipos-que-es-y-cuales-son-sus-etapas/>.
- [8] HPE. Hp proliant servers. [https://www.hpe.com/emea\\_europe/en/hpe-proliant-servers.html](https://www.hpe.com/emea_europe/en/hpe-proliant-servers.html).
- [9] IEEEEXPLORE. *MQTT Protocol: Fundamentals, Tools and Future Directions.* <https://ieeexplore.ieee.org/document/8931137>.
- [10] Microsoft. *C# documentation.* <https://learn.microsoft.com/en-us/dotnet/csharp/>.

- [11] Microsoft. *Introduction to Hyper-V*. <https://learn.microsoft.com/en-us/virtualization/hyper-v-on-windows/about/>.
- [12] MQTT.js. client library for the mqtt protocol. <https://www.npmjs.com/package/mqtt>.
- [13] MySQL. The world's most popular open source database. <https://www.mysql.com/>.
- [14] Node.js. Open source cross platform javascript runtime environment. <https://nodejs.org/en>.
- [15] PHP. A popular general-purpose scripting language. <https://www.php.net/>.
- [16] Sparkfun. *DHT22 Digital-output relative humidity & temperature sensor/module*. <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>.
- [17] SSH. Communications security for humans, systems, and networks. <https://www.ssh.com/>.
- [18] Telegram. Telegram. <https://telegram.org/>.
- [19] Telegram. Telegram api. <https://core.telegram.org/api>.
- [20] Ubuntu. Ubuntu. <https://ubuntu.com/>.
- [21] umqtt. Micropython. <https://github.com/micropython/micropython-lib/tree/master/micropython/umqtt.simple/umqtt>.
- [22] waveshare. *Pantalla oled sh1106 128x64 1.3 pulgadas datasheet*. <https://www.waveshare.com/w/upload/e/e3/1.3inch-SH1106-OLED.pdf>.