**UNIVERSIDAD POLITÉCNICA DE MADRID**

Escuela Técnica Superior de Ingeniería de Sistemas Informáticos

Logotipo

Descripción generada automáticamente

**Creación e implantación de un sistema IoT basado en ESP-32 para el análisis de espectros VIS-NIR**

**PROYECTO FIN DE GRADO**

**Álvaro Lameiro García**

Grado en Ingeniería de Computadores

Madrid, 2025

|  |  |  |
| --- | --- | --- |
|  | Logotipo  Descripción generada automáticamente | UNIVERSIDAD POLITÉCNICA DE MADRID  Escuela Técnica Superior de Ingeniería de Sistemas Informáticos |

**Grado en Ingeniería de Computadores**

**Creación e implantación de un sistema IoT basado en ESP-32 para el análisis de espectros VIS-NIR**

**PROYECTO FIN DE GRADO**

**Álvaro Lameiro García**

Grado en Ingeniería de Computadores

Bajo la dirección de:

Dr. Vicente García Alcántara

Madrid, 2025

Título: Creación e implantación de un sistema IoT basado en ESP-32 para el análisis de espectros VIS-NIR

Autor: Álvaro Lameiro García

Grado en Ingeniería de Computadores

Dirección:

Dr. Vicente García Alcántara

*<Dedicatoria (opcional) >*

Agradecimientos

<Página de agradecimientos (opcional)>

Abstract

<Abstract in English: maximum of 4000 characters, plain text (without symbols), structured summary of the thesis (introduction or motivation, objectives, findings and conclusions)>

Resumen

<Resumen en español: máximo de 4000 caracteres, texto plano (sin símbolos), resumen estructurado de la tesis (introducción o motivación, objetivos, hallazgos y conclusiones)>

1. Índice

[II. Índice de Figuras xi](#_Toc200799594)

[III. Lista de Tablas xii](#_Toc200799595)

[IV. Abreviaturas y Acrónimos xiii](#_Toc200799596)

[1. Introducción 1](#_Toc200799597)

[2. Estado de la Técnica 2](#_Toc200799598)

[2.1. Introducción 2](#_Toc200799599)

[2.2. Alcance del tema 2](#_Toc200799600)

[2.3. Revisión 3](#_Toc200799601)

[2.3.1. Aplicaciones del análisis VIS-NIR 3](#_Toc200799602)

[2.3.1.1. Agricultura de precisión y análisis de los suelos 3](#_Toc200799603)

[2.3.1.2. Industria alimentaria 4](#_Toc200799604)

[2.3.2. Dispositivos VIS/NIR 4](#_Toc200799605)

[2.3.2.1. Clasificación de dispositivos VIS/NIR 4](#_Toc200799606)

[2.3.2.2. Aplicaciones y desafíos 6](#_Toc200799607)

[2.3.2.3. Tabla comparativa 6](#_Toc200799608)

[2.3.2.4. Justificación de la elección del sensor 7](#_Toc200799609)

[2.3.3. Conclusiones 7](#_Toc200799610)

[3. Marco teórico 9](#_Toc200799611)

[3.1. Sistemas embebidos 9](#_Toc200799612)

[3.2. ESP32 10](#_Toc200799613)

[3.3. Espressif IoT Development Framework (ESP-IDF) 11](#_Toc200799614)

[3.4. AS7265x 12](#_Toc200799615)

[3.4.1. Introducción 12](#_Toc200799616)

[3.4.2. Arquitectura y composición 12](#_Toc200799617)

[3.4.3. Iluminación integrada 13](#_Toc200799618)

[3.4.4. Ventajas 13](#_Toc200799619)

[3.4.5. Inconvenientes 13](#_Toc200799620)

[3.5. I2C 13](#_Toc200799621)

[3.5.1. Descripción general 13](#_Toc200799622)

[3.5.2. Protocolo I2C 14](#_Toc200799623)

[3.6. Protocolo MQTT 15](#_Toc200799624)

[3.7. Internet of Things (IoT) 16](#_Toc200799625)

[3.8. Machine Learning 17](#_Toc200799626)

[4. Desarrollo del proyecto 19](#_Toc200799627)

[4.1. Objetivo 19](#_Toc200799628)

[4.2. Requisitos 19](#_Toc200799629)

[4.2.1. Funcionales 19](#_Toc200799630)

[4.2.2. No funcionales 22](#_Toc200799631)

[4.3. Metodología 23](#_Toc200799632)

[4.4. Hitos 25](#_Toc200799633)

[4.5. Realización del proyecto 28](#_Toc200799634)

[4.5.1. Hardware 28](#_Toc200799635)

[4.5.1.1. Componentes utilizados 28](#_Toc200799636)

[4.5.1.2. Esquema de conexión 29](#_Toc200799637)

[4.5.1.3. Consideraciones prácticas 30](#_Toc200799638)

[4.5.2. Software [42] 31](#_Toc200799639)

[4.5.2.1. ESP32 31](#_Toc200799640)

[4.5.2.1.1. Conexión Wi-Fi 31](#_Toc200799641)

[4.5.2.1.2. Comunicación con el sensor AS7265x 34](#_Toc200799642)

[4.5.2.1.3. Comunicación con la plataforma ThingsBoard 37](#_Toc200799643)

[4.5.2.1.4. Visualización en pantalla OLED 38](#_Toc200799644)

[4.5.2.2. ThingsBoard 41](#_Toc200799645)

[4.5.2.2.1. Visualización y control 41](#_Toc200799646)

[4.5.2.2.2. Cadenas de reglas 41](#_Toc200799647)

[Bibliografía 47](#_Toc200799648)

[Anexos 51](#_Toc200799649)

# Índice de Figuras

[*Ilustración 1. Espectro NIR del etanol líquido* [5] 2](#_Toc200390727)

[*Ilustración 2. El espectro electromagnético* [6] 3](#_Toc200390728)

[*Ilustración 3: Sensor SM245 de Ocean Optics con rango de 200 a 1050nm* 5](#_Toc200390729)

[*Ilustración 4: Analizador de Calidad de Producto (General) de Félix Instruments* 5](#_Toc200390730)

[*Ilustración 5: Componentes de un sistema empotrado* [18] 9](#_Toc200390731)

[*Ilustración 6: Algunos microcontroladores de la familia ESP32* [19] 10](#_Toc200390732)

[*Ilustración 7: Diagrama de pines de ESP32-WROOM-32* [21] 11](#_Toc200390733)

[*Ilustración 8: AS7265x en módulo Sparkfun* [24] 12](#_Toc200390734)

[*Ilustración 9: Visualización de las diferentes longitudes de onda que cubre el sensor* [24] 12](#_Toc200390735)

[*Ilustración 10: Ejemplo de sistema formado por un maestro y tres esclavos* [28] 14](#_Toc200390736)

[*Ilustración 11: Secuencia de inicio del protocolo I2C* [27] 14](#_Toc200390737)

[*Ilustración 12: Ejemplo completo del protocolo I2C* [29] 15](#_Toc200390738)

[*Ilustración 13: Ejemplo de protocolo MQTT* [31] 16](#_Toc200390739)

[*Ilustración 14: Modelo de creación de prototipos* [37]*.* 24](#_Toc200390740)

[*Ilustración 15: Logo de fritzing* [40] 29](#_Toc200390741)

[*Ilustración 16: Esquemático del sistema* 30](#_Toc200390742)

[*Ilustración 17: Web de configuración Wi-Fi (Fuente propia)* 32](#_Toc200390743)

# Lista de Tablas

[*Tabla 1: Comparación entre sensores para espectrometría VIS/NIR* 6](#_Toc200390744)

[*Tabla 2: Requisitos funcionales del sistema* 19](#_Toc200390745)

[*Tabla 3: Requisitos no funcionales del sistema* 22](#_Toc200390746)

# Abreviaturas y Acrónimos

|  |  |
| --- | --- |
| UPM  PFG | Universidad Politécnica de Madrid  Proyecto de Fin de Grado |
| NIR | *Near Infrared* (Infrarrojo cercano) |
| VIR | *Visible Infrared* (Infrarrojo visible) |
| CH | Carbono |
| PH | Potencial de Hidrógeno: Índice utilizado en química que expresa el grado de acidez o alcalinidad de una disolución aquosa |
| LED  IoT  CPU  SOC  SPI  I2C  I2S  UART  ML | *Light Emiting Diode* (Diodo emisor de luz)  Internet of Things (Internet de las cosas)  *Central Processing Unit* (Unidad Central de Procesamiento)  *System on Chip* (Sistema en Chip)  *Serial Peripheral Interface*  *Inter Integrated Circuit*  *Inter IC Sound*  *Universal Asynchronous Receiver Transmiter*  *Machine Learning* (Aprendizaje Automático) |
|  |  |

# Introducción

El agua y el suelo son dos recursos naturales esenciales para la vida en la Tierra. Su disponibilidad, calidad y estado determinan tanto la salud de los ecosistemas como el desarrollo de actividades básicas que sustentan el desarrollo humano, entre ellas la agricultura, la ganadería o la industria, entre otros. En un mundo afectado por el cambio climático, el sobrecrecimiento demográfico y la escasez de recursos, se vuelve prácticamente obligatorio implementar herramientas eficientes y sostenibles para su evaluación y su gestión.

Estos problemas no son ajenos a nuestro país, y aparecen en varias Comunidades Autónomas:

* En la Región de Murcia se advierte de estar “en el vórtice de la desertificación española al ser la provincia que está más cerca de convertirse en un desierto” debido principalmente al pobre contenido en nutrientes del suelo y a la mala calidad del agua de riego [1].
* Al mismo tiempo, la Comunidad de Madrid advierte de “la proliferación masiva de cianobacterias en las aguas del Parque Nacional de Monfragüe” que pueden “producir toxinas perjudiciales para la salud humana y ambiental” [2]

Tradicionalmente, el análisis de estos recursos se ha llevado a cabo mediante una serie de métodos de laboratorio los cuales, aunque precisos, resultan costosos tanto a nivel de equipamiento como de tiempo invertido. Es en este escenario donde surgen las técnicas espectroscópicas, especialmente las del rango visible (VIR) y del infrarrojo cercano (NIR). Estás técnicas permiten realizar análisis rápidos, no destructivos con la muestra e *in situ*, lo que facilita el control y monitoreo del recurso en tiempo real

# Estado de la Técnica

## Introducción

El objetivo de este apartado es presentar un análisis exhaustivo de las tecnologías existentes relacionadas con el análisis espectral en el rango visible e infrarrojo cercano (VIS y NIR), tanto a nivel de técnicas analíticas como de dispositivos. Con ello se pretende contextualizar la propuesta de este Proyecto de Fin de Grado en el marco de las soluciones tecnológicas actuales, así como encontrar las posibles limitaciones que justifiquen la realización de este estudio.

## Alcance del tema

La espectroscopíaestudia la cantidad de luz que absorbe, refleja o dispersa un objeto. En otras palabras, fragmenta el haz de luz y mide las diferentes longitudes de onda visible y no visible con el objetivo de identificar o estudiar diferentes sustancias químicas presentes en la muestra analizada[3].

Dicho método de análisis se apoya en el hecho de que las moléculas absorben ciertas frecuencias características de su estructura [4], como se puede ver en la siguiente figura.

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

*Ilustración 1. Espectro NIR del etanol líquido* [5]

Esta técnica aplica a las longitudes de onda desde los rayos gamma (0.01nm) hasta las microondas (> 30 µm) y radiofrecuencia (> 100 µm), pero para este proyecto el estudio se va a enfocar en el rango visible (380 a 750 nm) y el infrarrojo cercano (≈ 780nm a 3000nm).

Imagen que contiene Gráfico en cascada

El contenido generado por IA puede ser incorrecto.

*Ilustración 2. El espectro electromagnético* [6]

## Revisión

Esta revisión se ha llevado a cabo principalmente utilizando la herramienta Google Scholar [7] para la búsqueda de libros y artículos científicos, así como el navegador Google [8]para la búsqueda de productos y sensores relacionados con el análisis explicado en el punto anterior.

## Aplicaciones del análisis VIS-NIR

## Agricultura de precisión y análisis de los suelos

La espectroscopía VIS-NIR se ha utilizado con éxito para predecir propiedades fisicoquímicas del suelo, tales como el contenido de CH orgánico, la textura o el PH. Con esto se permite analizar las propiedades del suelo en base a su reflectancia, reduciendo así la necesidad de análisis químicos de laboratorio. En el estudio *On-line measurement of some selected soil properties using a VIS–NIR sensor* se demostró que mediante modelos calibrados con espectros VIS-NIR es posible predecir con alta precisión propiedades químicas, físicas y biológicas de los suelos, facilitando así una caracterización más eficiente y económica para aplicaciones agronómicas [9].

## Industria alimentaria

Otra prometedora aplicación del análisis espectroscópico se encuentra en el estudio *Analyzing the quality parameters of apples by spectroscopy from Vis/NIR to NIR region: A comprehensive review* donde se utilizan los dos rangos de onda comentados anteriormente para evaluar una serie de parámetros clave, con el objetivo de asegurar la calidad de las manzanas. Se destaca la capacidad de esta técnica para realizar mediciones rápidas y no destructivas de la fruta, lo que lo convierte en un procedimiento a tener en cuenta para el control de calidad en la industria frutícola [10].

La espectroscopía VIS/NIR ha sido utilizada en diversas etapas de la cadena de producción de las manzanas, permitiendo así evaluar las siguientes cualidades:

* **Evaluación interna:** Gracias al control de diversos elementos químicos (Éster, Alhendín, Cetona y alcohol) se han conseguido predecir ciertos parámetros como la calidad del sabor, el buen aroma, la textura interna o el contenido nutricional, entre otros.
* **Evaluación externa:** Mediante el análisis NIR, se pueden realizar predicciones del color y tamaño que tendrán las manzanas inmaduras una vez que finalicen su proceso de maduración.
* **Otros parámetros:** Esta técnica también se puede utilizar para optimizar las condiciones de almacenamiento de las manzanas, adivinar su origen o controlar el grado de madurez de la muestra, permitiendo así establecer una fecha de consumo preferente.

## Dispositivos VIS/NIR

En este apartado se va a presentar un panorama general de los dispositivos disponibles para la espectroscopía VIS/NIR explorando sus características, aplicaciones y limitaciones con el objetivo de determinar que sensor se va a utilizar para esta propuesta de Proyecto de Fin de Grado.

## Clasificación de dispositivos VIS/NIR

Los dispositivos espectroscópicos se pueden clasificar en tres categorías principales:

* Equipos de laboratorio: Como por ejemplo los de **Ocean Optics**, que ofrecen una altísima resolución espectral y precisión, pero son extremadamente costosos y requieren de condiciones controladas.

Imagen que contiene electrónica

El contenido generado por IA puede ser incorrecto.

*Ilustración 3: Sensor SM245 de Ocean Optics con rango de 200 a 1050nm*

* Dispositivos portátiles: Como el **Felix F-750**, ofrecen un análisis detallado sin la necesidad de hacerse dentro de un laboratorio, lo que permite realizar controles *in situ.*



*Ilustración 4: Analizador de Calidad de Producto (General) de Félix Instruments*

* Sistemas hiperespectrales por imagen: Una herramienta extremadamente cara y poderosa, que permite recopilar y procesar información a lo largo de **todo** el espectro electromagnético.

Todos estos dispositivos utilizan técnicas de detección en el rango de 400 a 2500 nm, con sensores basados en silicio, en su mayoría. Sus fuentes de iluminación pueden ser halógenas, LEDs o láseres.

## Aplicaciones y desafíos

Este tipo de sensores se emplea ampliamente en la agricultura de precisión, farmacología, industria química, medioambiente y más. Sin embargo, su integración en proyectos IoT de bajo coste se ve limitada por factores como:

* Elevado precio de muchos espectrómetros comerciales (> 1000€)
* Dimensiones físicas que dificultan su integración en sistemas embebidos
* Necesidad de complejos modelos de calibración

## Tabla comparativa

*Tabla 1: Comparación entre sensores para espectrometría VIS/NIR*

| **Dispositivo / Sensor** | **Rango espectral** | **Nº de bandas** | **Resolución espectral** | **Interfaz** | **Tamaño / Integración** | **Precio aprox.** | **Observaciones** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Ocean Optics Flame** [11] | 350–1000 nm | Continuo | ~1.5 nm | USB | Voluminoso (de laboratorio) | > 2000 € | Alta precisión. Requiere PC. |
| **Felix F-750** [12] | 729–975 nm | Continuo | - | Propietario | Portátil, autónomo | ~4000 € | Diseñado para frutas. Muy fiable. |
| **NIRScan Nano (TI)** [13] | 900–1700 nm | Continuo | ~10 nm | I²C / USB | Compacto | ~1000 € | Versátil, pero caro. Requiere software. |
| **SCiO (Consumer Physics)** [14] | 740–1070 nm | Pocos puntos | - | Bluetooth | Muy compacto (de consumo) | ~300–500 € | Uso limitado. Plataforma cerrada. |
| **AS7265x** [15] | 410–940 nm | 18 bandas (fijas) | ~20 nm (por banda) | I²C / UART | Muy compacto. Ideal para IoT | ~80–120 € | Buena cobertura VIS-NIR. Integrable en ESP32. |

## Justificación de la elección del sensor

En el contexto de este trabajo, que busca desarrollar un sistema IoT de bajo coste y portátil para el análisis espectral VIS/NIR, se ha optado por el sensor **AS7265x**, fabricado por AMS (Austria Microsystems).

Las razones clave para la elección han sido:

* **Compromiso entre prestaciones y precio:** Ofrece una resolución espectral algo discreta, pero suficiente para muchas actividades prácticas y a un coste muy inferior al del resto de espectrómetros (< 90 €).
* **Fácil integración**: Se comunica mediante I2C, por lo que es compatible con microcontroladores ESP32 y puede integrarse en dispositivos portátiles.
* **Iluminación integrada:** Reduce así el número de dispositivos necesarios para el funcionamiento del sistema y supone una reducción del coste energético.

Si bien no alcanza la resolución de los espectrómetros de laboratorio, el AS7265x representa una solución robusta, accesible y flexible para sistemas IoT como el que se va a realizar en este proyecto.

## Conclusiones

A lo largo de este apartado se ha evidenciado el amplio potencial de la espectroscopía del rango visible e infrarrojo cercano (VIS/NIR) como herramienta de análisis no destructivo en múltiples ámbitos. Los estudios revisados reflejan cómo esta técnica puede reducir la dependencia de análisis químicos tradicionales ofreciendo una alternativa más rápida, sostenible y económicamente viable.

En cuanto a los dispositivos disponibles, se ha evidenciado que, si bien los equipos comerciales de alta gama ofrecen resultados muy fiables, su elevado coste y dimensiones limitan su uso en la creación de aplicaciones IoT.

Frente a estas limitaciones, sensores como el AS7265x ofrecen una solución equilibrada y fácilmente integrable con microcontroladores, lo cual abre un nuevo marco de investigación y desarrollo en el que se basará este proyecto.

# Marco teórico

## Sistemas embebidos

Un sistema embebido es un sistema de computación compuesto por una combinación de hardware y software diseñado para realizar una o algunas pocas funciones específicas.

Destacan por su fiabilidad, eficiencia energética y bajo coste de fabricación. Además, se pueden programar tanto en lenguajes de bajo nivel (p.ej. ensamblador) como en C o C++, cuando el tiempo de respuesta del sistema no es un factor crítico.

Los componentes principales de un sistema embebido suelen ser la CPU, puertos de comunicaciones, módulos de Entrada/Salida y conversores Analógico/Digital y Digital/Analógico, como se puede comprobar en la imagen [16][17].

Imagen de la pantalla de un celular con texto e imágenes

El contenido generado por IA puede ser incorrecto.

*Ilustración 5: Componentes de un sistema empotrado* [18]

## ESP32

Los ESP32 son una familia de microcontroladores realizados por Espressif Systems ampliamente utilizados en el mundo de IoT por su bajo coste, versatilidad y eficiencia energética. Estos dispositivos integran en un único chip capacidades de procesamiento, conectividad inalámbrica y una amplia gama de interfaces de comunicación, lo que los convierte en una solución idónea para el desarrollo de sistemas embebidos.



*Ilustración 6: Algunos microcontroladores de la familia ESP32* [19]

Para la realización de este PFG se ha escogido el **ESP32** **Dev Kit C V2** de AZ Delivey [20], uno de los módulos más populares de la familia basado en el SOC ESP32-WROOM 32 queincluye todo lo necesario para desarrollar aplicaciones IoT compactas y eficientes.

Sus principales características técnicas son:

* Fuente de alimentación y programación a través de micro USB-B.
* Procesador ESP-WROOM-32 con una CPU dual-core Tensica Xtensa LX6 con una frecuencia de reloj de hasta 240 MHz, Bluetooth 4.2 y WLAN-WIFI.
* 512 KB de SRAM integradas en un único chip controlador, así como 16 MB de memoria flash.
* 32 Pines E/S digitales, entre los cuales se encuentran interfaces SPI, I2C, I2S y UART como se muestra en la imagen siguiente:

Escala de tiempo

El contenido generado por IA puede ser incorrecto.

*Ilustración 7: Diagrama de pines de ESP32-WROOM-32* [21]

## Espressif IoT Development Framework (ESP-IDF)

ESP-IDF es el principal entorno proporcionado por Espressif Systems. Se trata de un framework de desarrollo en C y C++, de bajo nivel y altamente configurable, que permite acceder de forma bastante directa a las funcionalidades del microcontrolador.

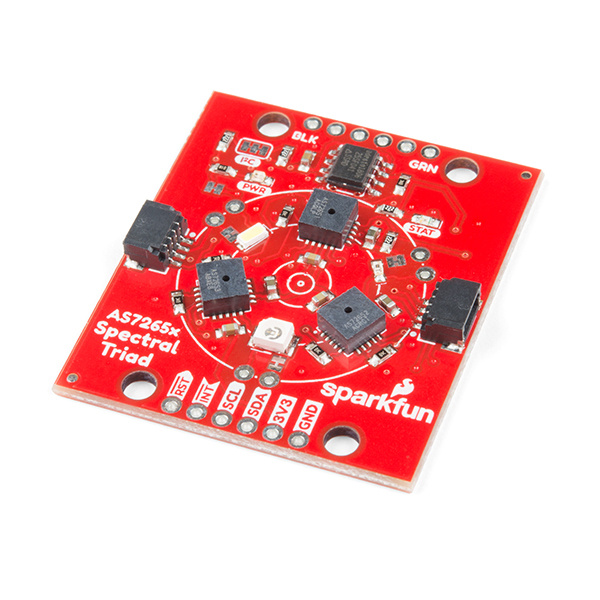
A diferencia de otros entornos, como Arduino IDE, el ESP-IDF ofrece una aproximación más profesional, robusta y pensada para aplicaciones profesionales.

En este proyecto, el Framework actúa como capa intermedia entre el hardware y las capas superiores de comunicación y gestión de datos, permitiendo así el control detallado de la adquisición espectral y el manejo del protocolo MQTT para la conexión con la plataforma Thingsboard [22].

## AS7265x

## Introducción

El AS7265x es un sensor multiespectral fabricado por AMS, diseñado para aplicaciones de espectroscopía en el rango visible e infrarrojo cercano. Es un dispositivo compacto y de coste reducido que permite capturar información espectral de 18 diferentes longitudes de onda entre los 410nm y los 940nm [23].

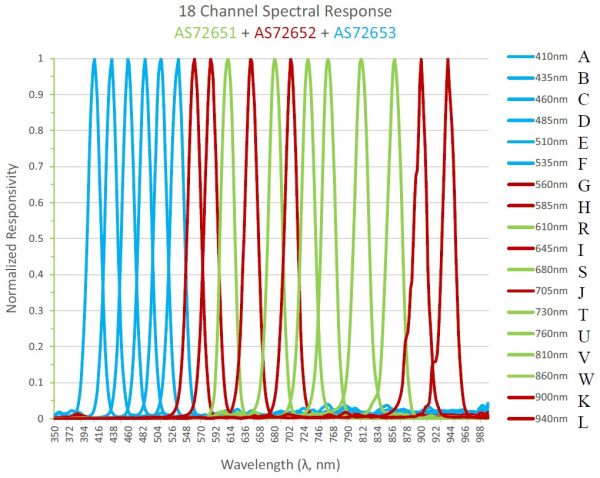


*Ilustración 8: AS7265x en módulo Sparkfun* [24]

## Arquitectura y composición

En realidad, el AS7265x no es un único sensor, sino una combinación de tres sensores individuales que trabajan de forma conjunta [25]:

* **AS72651:** Opera en el rango visible, con canales centrados en las siguientes longitudes: 410, 435, 460, 485, 510 y 535 nm.
* **AS72652:** Mezcla el rango visible y las primeras longitudes del NIR, incluyendo: 560, 585, 645, 705, 900 y 940 nm
* **AS72653:** Amplia la cobertura NIR con las siguientes bandas: 610, 680,730, 760, 810, 860



*Ilustración 9: Visualización de las diferentes longitudes de onda que cubre el sensor* [24]

Estos tres sensores están integrados en un único módulo que se comunica mediante el protocolo I2C, además, incorpora un microcontrolador interno que gestiona las mediciones, lo cual simplifica la interacción del sensor con el microcontrolador ESP32, ya que se accede a las mediciones desde registros virtuales.

## Iluminación integrada

El módulo anterior cuenta con un LED blanco de 5700k, una luz UV que emite en 405nm y un LED infrarrojo que emite en 875nm [24]. Esto permite iluminar las muestras de forma controlada, reduciendo la influencia de la luz ambiente y mejorando la precisión y repetibilidad de las mediciones.

## Ventajas

* Cobertura amplia del espectro VIS-NIR.
* Formato compacto y bajo consumo.
* Comunicación I2C estándar.

## Inconvenientes

* Resolución espectral discreta y limitada (18 bandas fijas), por lo que no se puede obtener un espectro continuo.
* La precisión depende de condiciones externas, como la distancia o la temperatura.

## I2C

## Descripción general

Circuito inter-integrado (*Inter Integrated Circuit*) es un bus serie desarrollado en 1982 y utilizado principalmente para la comunicación entre diferentes partes de un circuito, por ejemplo, entre un microcontrolador y otros periféricos integrados como sensores, pantallas, memorias EEPROM u otros dispositivos electrónicos [26].

El bus de datos I2C está diseñado con una arquitectura maestro/esclavo. La transferencia de datos se realiza por un maestro y de tal manera el esclavo reacciona. Dicho protocolo se caracteriza por ser síncrono, bidireccional y tener dos líneas de señales: SDA (*System Data*) es la línea por la que se mueven los datos entre los dispositivos y SCL (*System Clock*) es la línea que lleva los pulsos de reloj que sincronizan el sistema. Permite la comunicación entre muchos dispositivos a una velocidad aceptable de unos 100kbits por segundo (salvo casos especiales) [27].



*Ilustración 10: Ejemplo de sistema formado por un maestro y tres esclavos* [28]

## Protocolo I2C

Para comenzar la comunicación, cualquier dispositivo maestro tiene que establecer la condición de inicio (*Start*), donde se fuerza la línea SDA a nivel bajo mientras SCL permanece alta, como se puede ver en la imagen siguiente.

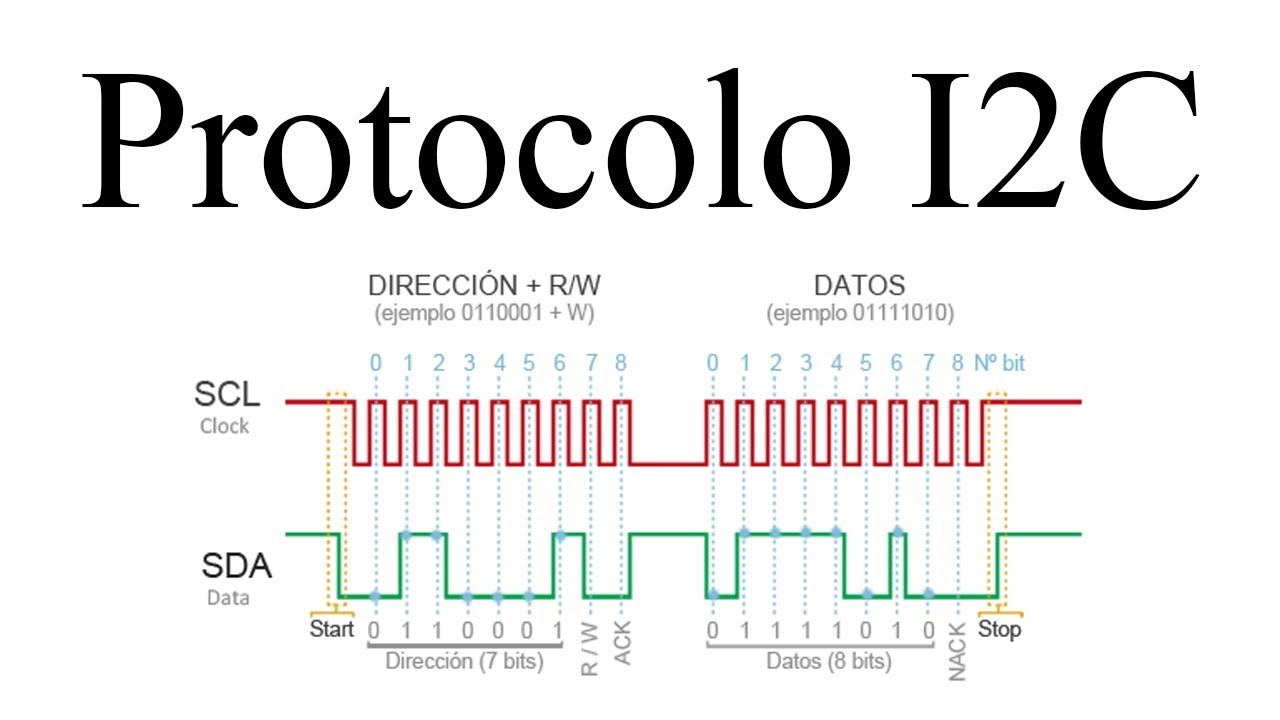
Imagen que contiene Gráfico de cajas y bigotes

El contenido generado por IA puede ser incorrecto.

*Ilustración 11: Secuencia de inicio del protocolo I2C* [27]

A continuación, el maestro envía la dirección del esclavo (7 u 8 bits) seguida de un bit que indica si va a realizar una lectura (1) o escritura (0). Si el esclavo reconoce su dirección responde con un bit de reconocimiento (ACK).

Seguido a la comunicación anterior se realiza la transferencia de datos en bloques de 8 bits, cada uno de estos seguidos por un bit ACK del receptor. Una vez finalizado el intercambio, el maestro genera una condición de parada (*Stop*) poniendo SDA en alto mientras SCL permanece en alto.

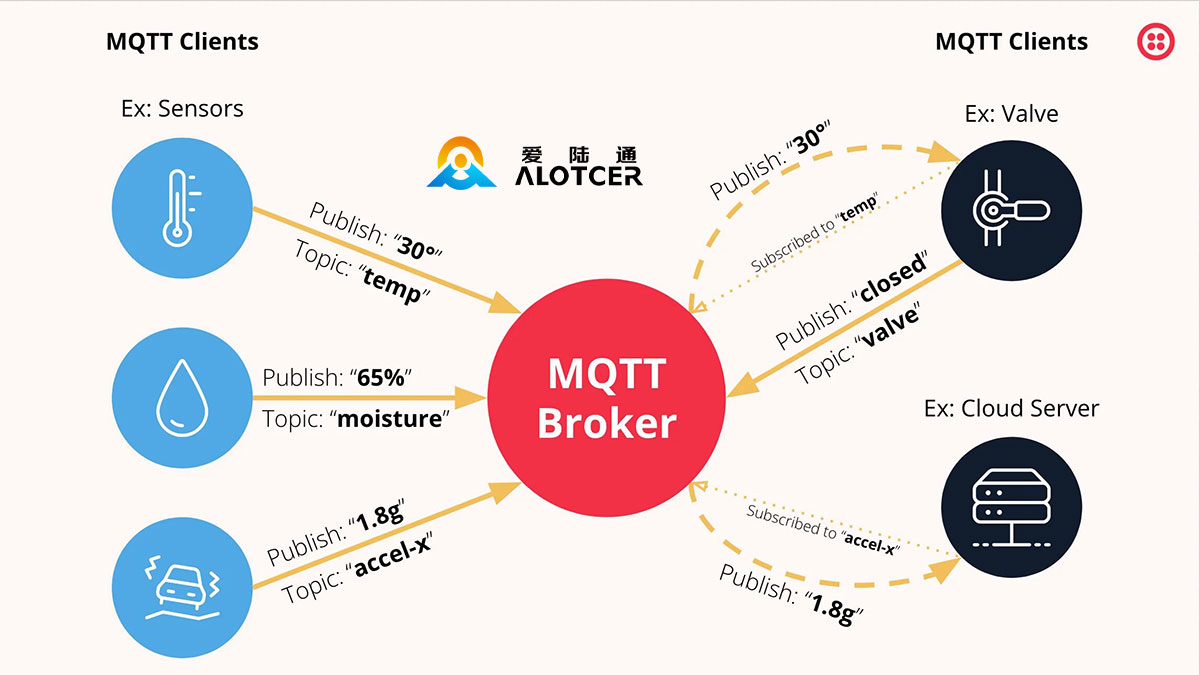


*Ilustración 12: Ejemplo completo del protocolo I2C* [29]

## Protocolo MQTT

MQTT (*Message Query Telemetry Transport*) es un protocolo ligero y eficiente diseñado para la transmisión de datos entre dispositivos a través de redes con recursos restringidos y ancho de banda limitado [30].

Está basado en la arquitectura publicador/suscriptor y suele operar bajo el protocolo TCP/IP, lo que permite una comunicación asíncrona, ordenada y bidireccional entre dispositivos, aplicaciones, sensores y servidores.



*Ilustración 13: Ejemplo de protocolo MQTT* [31]

La imagen muestra la arquitectura fundamental del protocolo MQTT. En el centro se encuentra el bróker MQTT , que actúa como intermediario entre los diferentes dispositivos conectados. A la izquierda se encuentran los publicadores, que envían los datos a varios *topics* mientras que los suscriptores de la derecha reciben automáticamente los mensajes publicados en esos *topics* y toman las acciones pertinentes*.* Es por esto por lo que este protocolo facilita la integración de múltiples sensores y receptores dentro de un mismo sistema IoT.

## Internet of Things (IoT)

El concepto de *Internet of Things* (IoT) hace referencia a una red de dispositivos físicos interconectados que recopilan, procesan y comparten datos a través de Internet u otras redes.

El objetivo principal del IoT es permitir una supervisión y control remoto de procesos, objetos o entornos físicos en tiempo real. Esto se traduce en aplicaciones variopintas como la agricultura inteligente, la automatización industrial o el análisis ambiental, donde los datos recopilados se convierten en información útil que facilita la optimización y la toma de decisiones [32].

En este proyecto, se implementa un sistema IoT combinando algunas tecnologías repasadas anteriormente en el marco teórico, como la ESP32 o el protocolo MQTT, con sensores y aplicaciones que se verán más adelante durante la descripción del Hardware y el Software utilizado.

## Machine Learning

El Machine Learning (Aprendizaje Automático) es una rama de la Inteligencia Artificial aplicada al desarrollo de técnicas de aprendizaje en computadores [33].

Dicha técnica consiste en la creación de algoritmos capaces de aprender patrones a partir de datos, sin estar específicamente programados para realizar tareas específicas. Gracias al uso de procesos de entrenamiento con conjuntos de datos representativos, los modelos de Machine Learning pueden llevar a cabo predicciones y clasificaciones con una precisión proporcional al grado de entrenamiento.

En la actualidad podemos distinguir tres tipos de métodos de Aprendizaje Automático [34]:

* **Aprendizaje supervisado:** Utiliza conjuntos de datos etiquetados para entrenar algoritmos con el objetivo de clasificar datos o predecir resultados con precisión.
* **Aprendizaje no supervisado:** Se trabaja con datos sin etiquetar para observar si un modelo ya entrenado es capaz de catalogar correctamente nuevas entradas.
* **Aprendizaje semi-supervisado:** Es la combinación de los dos métodos de aprendizaje anteriores.
* **Aprendizaje por refuerzo:** Es un método similar al aprendizaje supervisado, pero sin el uso de datos de ejemplo. Este modelo aprende sobre la marcha aplicando la técnica de prueba y error.

Para este PFG, se va a implementar un modelo de aprendizaje supervisado con el objetivo de añadir valor al sistema IoT al permitir no solo monitorizar datos sino también interpretarlos y tomar decisiones automáticas basadas en ellos.

# Desarrollo del proyecto

En este capítulo se va a elaborar todo lo relacionado con el desarrollo del proyecto, desde el objetivo hasta las pruebas realizadas, pasando por toda la realización del proyecto.

## Objetivo

El objetivo de este proyecto es diseñar e implementar un sistema IoT portátil capaz de adquirir datos espectrales en el rango VIS-NIR, transmitirlos a una plataforma en la nube para su control y analizarlos mediante técnicas de Aprendizaje Automático desarrolladas en una plataforma web personalizada. Todo esto, con el fin de realizar análisis de muestras físicas con el objetivo de extraer algunas de sus propiedades y características, sin la necesidad de equipo de laboratorio especializado ni procedimientos invasivos.

## Requisitos

## Funcionales

Los requisitos funcionales son especificaciones detalladas que definen las acciones, funcionalidades y comportamientos que se esperan del sistema para cumplir con su propósito previsto [35].

Para este proyecto se han tenido en cuenta:

*Tabla 2: Requisitos funcionales del sistema*

| **ID** | **Descripción** | **Prioridad** | **Justificación** |
| --- | --- | --- | --- |
| RF1 | Al iniciar, el sistema debe comprobar si hay credenciales Wi-Fi almacenadas en NVS. | Alta | Mejora la experiencia del usuario al evitar pasos redundantes tras cada reinicio. |
| RF2 | Si no hay una conexión válida, debe crear un punto de acceso (AP) y un servidor web para introducir el SSID y la contraseña de la red wifi a la que se quiera conectar. | Alta | Permite configurar el dispositivo sin necesidad de reprogramación, facilitando la instalación en entornos no técnicos. |
| RF3 | El sistema debe guardar las credenciales recibidas en la memoria NVS para futuros reinicios. | Media | Mejora la experiencia del usuario al evitar pasos redundantes tras cada reinicio. |
| RF4 | El sistema debe inicializar el sensor espectral AS7265x mediante el bus I2C. | Alta | Imprescindible para la realización del análisis. |
| RF5 | El sistema debe procesar los datos del sensor, manteniéndolos en una estructura con los 18 canales espectrales y la temperatura del sensor. | Alta | Base funcional del sistema: sin esa lectura no se puede realizar ningún análisis. |
| RF6 | Una vez conectado a una red Wi-Fi, el sistema debe iniciar el protocolo MQTT y conectarse a ThingsBoard usando un token de autenticación. | Alta | La conexión es necesaria para involucrar el control de los datos en tiempo real al proyecto. |
| RF7 | Los datos adquiridos por el sensor deben ser empaquetados en formato JSON y enviados a ThingsBoard mediante MQTT. | Alta | Es parte del proceso de envío de datos. |
| RF8 | La plataforma Thingsoard debe mostrar los datos en paneles y reenviarlos a un servidor propio a través de cadenas de reglas. | Media | Permite visualizar e integrar los datos sin modificar el dispositivo embebido. |
| RF9 | En caso de que la temperatura del sensor sea superior a 30º se enviará una notificación a un grupo de telegram desde ThingsBoard para que se tenga en consideración. | Baja | Mantener un control de la temperatura es crucial para conseguir unos datos fiables a pesar de que el sensor es de por sí resistente a temperaturas extremas. |
| RF10 | El servidor debe recibir datos desde ThingsBoard, transformarlos en un CSV y almacenarlos en una ubicación accesible para el análisis. | Alta | Los modelos de machine learning necesitan un conjunto de datos limpio y estructurado. |
| RF11 | Una aplicación web debe permitir al usuario cargar los datos .CSV y entrenar modelos de aprendizaje supervisado. | Alta | Imprescindible para el análisis inteligente de datos espectrales. |
| RF12 | La interfaz web debe ofrecer una forma sencilla de cargar nuevos datos y ver resultados de predicciones sobre las muestras. | Media | Mejora la interacción entre el usuario y la máquina, permitiendo validar resultados sin conocimientos técnicos. |
| RF13 | El microcontrolador debe ser capaz de reconectarse automáticamente al Wi-Fi y al MQTT bróker en caso de pérdida de conexión. | Alta | Aumenta la robustez frente a fallos en la red. |
| RF14 | El sistema debe proporcionar *feedback* sobre el estado de conexión, sensor o errores críticos mediante logs y/o mensajes. | Alta | Mejora la robustez general del sistema y la identificación de errores o problemas. |

## No funcionales

Los requisitos no funcionales definen las características generales del sistema y su comportamiento en diversas condiciones [35].

Para este sistema se han tenido en cuenta:

*Tabla 3: Requisitos no funcionales del sistema*

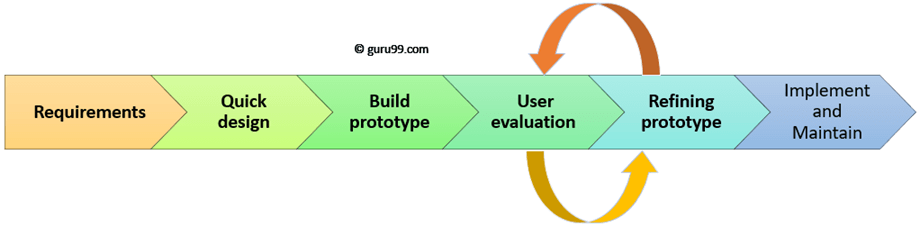
| **ID** | **Descripción** | **Prioridad** | **Justificación** |
| --- | --- | --- | --- |
| RNF1 | La transmisión de datos debe tener una latencia inferior a 5 segundos desde la lectura hasta su aparición en ThingsBoard. | Media | Asegura una experiencia cercana al tiempo real, lo cual mejora la experiencia de monitoreo. |
| RNF2 | El sistema debe utilizar MQTT con autenticación por token para asegurar un mínimo nivel de seguridad en la comunicación. | Alta | Garantiza que los datos no puedan ser interceptados y modificados fácilmente. |
| RNF3 | El sistema debe operar correctamente en una temperatura ambiente de entre 15ºC y 30 ºC. | Media | Cambios bruscos de la temperatura pueden afectar a las mediciones del sensor, por eso es necesario llevar un control de su temperatura. |
| RNF4 | El sistema debe estar diseñado de forma modular, permitiendo la integración de nuevos sensores, funciones o cambios adicionales. | Alta | Facilita la evolución del proyecto hacia nuevos escenarios y requerimientos. |
| RNF5 | El código de la programación del ESP32 y de la aplicación de Machine Learning deben estar documentados y estructurados, facilitando sus revisiones y mantenimiento. | Media | Mejora la comprensión por parte de personas externas al proyecto. |
| RNF6 | El sistema debe procesar y analizar los datos sin errores. | Alta | La fiabilidad del sistema es imprescindible para su introducción a un entorno profesional. |

## Metodología

Existen una gran y diversa cantidad de metodologías para el desarrollo de proyectos como: Ágil (Basada en los ciclos cortos e ideal para el trabajo en equipo y la adaptación constante), en Cascada (Secuencial y rígida con fases bien definidas), Iterativa (Donde el desarrollo se divide en pequeñas funcionalidades, mejorando el producto en cada ciclo), modelo en V (Aquí se valida cada etapa del desarrollo, con un foco fuerte en las fases de diseño y pruebas) … [36]

Sin embargo, para este Proyecto se ha optado por utilizar una metodología de prototipos complementada con la división del trabajo en Hitos, por tratarse de un enfoque flexible y adaptado a sistemas experimentales donde se deben validar componentes de hardware y de software de manera reiterativa.

La metodología consiste en construir versiones parciales del sistema, llamadas prototipos, que permiten verificar el comportamiento de cada módulo, identificar los errores que puedan surgir y orientar las mejoras, cambios y eliminaciones de cara a alcanzar la solución final [37].



*Ilustración 14: Modelo de creación de prototipos* [37]*.*

El modelado de prototipos sigue las siguientes 6 fases:

* Fase 1, Análisis de requisitos: Se definen los requisitos del sistema en base a una interacción con los usuarios del sistema.
* Fase 2, Diseño rápido: En esta etapa se crea un diseño simple del sistema, el cual da una breve idea del sistema al usuario.
* Fase 3, Construcción del prototipo: Se desarrolla un pequeño modelo funcional del sistema.
* Fase 4, Evaluación inicial del usuario: Se presenta el prototipo al cliente con el objetivo de recibir una evaluación inicial, la cual será recopilada por el desarrollador.
* Fase 5, Refinamiento del prototipo: En base a la evaluación del cliente, el sistema se perfeccionará siguiendo los comentarios y sugerencias del usuario.
* Fase 6, Implementación del producto: Una vez que se desarrolla el sistema final, se prueba meticulosamente y se implementa en la producción.

Además de la metodología de prototipos, se va a incluir una planificación por Hitos [38]. Un hito (o *milestone*) es un suceso significativo que tiene importancia para todo el proyecto. Es extremadamente útil para dividir el proyecto en etapas más manejables, fomentando una buena planificación, un control de plazos y una identificación temprana de posibles problemas o retrasos en el desarrollo, lo que se resume en un mayor control de las responsabilidades y la eficacia a la hora de progresar en el proyecto.

## Hitos

* Hito 1, Análisis y planificación inicial.
  + Descripción: Estudio de la información necesaria para asentar las bases conceptuales y técnicas del proyecto. Se investigan técnicas relevantes de espectroscopía, su alcance y productos ya existentes en el mercado relacionados.
  + Inicio: Semana 1.
  + Duración: 1 Semana.
  + Subhitos:
    1. Investigar como se realiza la espectroscopía infrarroja.
    2. Revisión y estudio de artículos relacionados con el tema.
    3. Esbozo de los objetivos del sistema.
* Hito 2, Búsqueda de componentes.
  + Descripción: Estudio de los diferentes sensores aptos para espectroscopía y componentes necesarios para crear un sistema IoT, buscando la mayor viabilidad tanto por coste como por disposición.
  + Inicio: Semana 2.
  + Duración: 3-4 Días.
  + Subhitos:
    1. Búsqueda y adquisición de componentes.
    2. Revisión y justificación de los componentes adquiridos.
* Hito 3, Diseño del sistema embebido.
  + Descripción: Definición de la arquitectura lógica y hardware del sistema. Planificación del sistema de alimentación, conectividad y sensores.
  + Inicio: Semana 3.
  + Duración: 1 Semana.
  + Subhitos:
    1. Elección de pines, protocolos de comunicación y creación de un diagrama de conexiones.
    2. Diseño preliminar de esquemáticos.
    3. Elección del tipo de alimentación.
* Hito 4, Desarrollo de software (ESP32).
  + Descripción: Creación de las distintas funciones software necesarias para el correcto funcionamiento del sistema IoT.
  + Inicio: Semana 4.
  + Duración: 4 semanas.
  + Subhitos:
    1. Implementación del entorno software en ESP-IDF.
    2. Desarrollo del código necesario para la conexión Wi-Fi.
    3. Desarrollo del código necesario para la recolección de datos del sensor.
    4. Desarrollo del código necesario para el envío de los datos a la plataforma ThingsBoard.
    5. Implementación de las visualizaciones en ThingsBoard.
    6. Realización de las pruebas de software pertinentes.
* Hito 5, Desarrollo de software (Machine Learning).
  + Descripción: Diseño e implementación de una aplicación web que permita al usuario cargar archivos CSV de telemetrías, entrenar modelos de aprendizaje supervisado y realizar predicciones sobre nuevas muestras espectrales.
  + Inicio: Semana 9.
  + Duración: 1-2 semanas.
  + Subhitos:
    1. Descarga de las telemetrías en formato CSV.
    2. Desarrollo del código necesario para aplicar Aprendizaje Automático a las telemetrías etiquetadas.
    3. Realización de las pruebas software pertinentes.
* Hito 6, Construcción física:
  + Descripción: Diseño de la carcasa y embalaje del dispositivo, integrando el hardware y el software.
  + Inicio: Semana 10.
  + Duración: 3 días.
  + Subhitos:
    1. Diseño de la carcasa y elección de materiales.
    2. Incorporación del resto de componentes a la carcasa.
* Hito 7, Pruebas:
  + Descripción: Todos los componentes desarrollados son integrados y probados de extremo a extremo. Se evalúa el flujo de trabajo completo y se simulan situaciones de error.
  + Inicio: Semana 11.
  + Duración: 1 Semana.
  + Subhitos:
    1. Validación del funcionamiento correcto del sensor.
    2. Comprobación de la correcta integración del sistema.
    3. Pruebas de extremo a extremo.
    4. Simulaciones de error.
* Hito 8, Evaluación y validación del sistema:
  + Descripción: Se evalúa el rendimiento del sistema tanto a nivel hardware (robustez de la conexión, estabilidad del firmware) como del software (rendimiento del sistema y precisión de los modelos de ML).
  + Inicio: Semana 12.
  + Duración: 1 Semana.
  + Subhitos:
    1. Verificación de los requisitos funcionales del sistema.
    2. Pruebas de rendimiento de la comunicación.
    3. Evaluación de los modelos de Machine Learning.
    4. Simulación de fallos y pruebas de robustez.
* Hito 9, Documentación:
  + Descripción: Recopilación del trabajo realizado a lo largo del proyecto mediante la redacción de la memoria.
  + Inicio: Semana 13.
  + Duración: 3 Semanas.
  + Subhitos:
    1. Redacción de la memoria técnica de PFG.
    2. Búsqueda de fuentes y figuras.
    3. Creación de tablas y diagramas.
    4. Verificación final y entrega.

## Realización del proyecto

## Hardware

El dispositivo físico realizado en este proyecto está basado en una arquitectura sencilla y modular, diseñada para facilitar su prueba en el laboratorio como su posible puesta en marcha en entornos reales. Los componentes seleccionados se han elegido por su compatibilidad, disponibilidad y bajo coste, lo que permite replicar y/o escalar el sistema fácilmente.

## Componentes utilizados

* **ESP32-WROOM-32:** Microcontrolador principal que gestiona la lógica del sistema, la conexión Wi-Fi, la comunicación con el sensor espectral y la transmisión de datos vía MQTT. Se eligió este modelo por su versatilidad, conectividad integrada y bajo consumo.
* **Sensor multiespectral AS7265x:** Encargado de la adquisición de los datos espectrales en bandas en los rangos visible e infrarrojo. Se comunica mediante el protocolo I2C con la ESP32 y permite capturar un perfil espectral discreto de una muestra correctamente iluminada.
* **Pantalla OLED I2C:** Utilizada como interfaz de usuario básica para mostrar mensajes clave del sistema como: el estado de la conexión Wi-Fi, errores en la comunicación con el sensor o problemas durante el protocolo MQTT. Opera también mediante el bus I2C, compartiendo línea con el sensor anterior.
* **Protoboard:** Se utilizó una placa de pruebas para el montaje del sistema sin necesidad de soldaduras. Este montaje permite modificar conexiones fácilmente durante la fases de evolución del prototipo y facilita las pruebas individuales de cada módulo del sistema.

## Esquema de conexión

Con el objetivo de documentar la implementación del sistema, se ha optado por usar el software Fritzing [39].

Imagen que contiene Logotipo

El contenido generado por IA puede ser incorrecto.

*Ilustración 15: Logo de fritzing* [40]

Fritzing es un software de código abierto que permite a los usuarios diseñar y simular circuitos electrónicos utilizando una protoboard virtual, en la cual se pueden colocar e interconectar todo tipo de componentes, perfecto para tener una representación visual precisa del circuito físico real a implementar [41].

Como se va a poder observar a continuación, los componentes se conectan todos a la ESP32 mediante el mismo bus I2C, compartiendo pines SDA (ESP: 21) y SCL (ESP: 22) pero con diferentes direcciones lógicas. Además, se ha habilitado el pin G2 de la ESP para suministrar de energía y controlar la iluminación de los leds integrados en el AS7265x. Se ha respetado el rango de tensión recomendado de 3.3V para todos los módulos y se ha utilizado una alimentación USB estándar (Tipo B) para facilitar su posible portabilidad.

Imagen de la pantalla de un computador

El contenido generado por IA puede ser incorrecto.

*Ilustración 16: Esquemático del sistema*

## Consideraciones prácticas

Durante la construcción se han tenido en cuenta aspectos como la disposición física de los componentes, con el objetivo de asegurar la iluminación y análisis constante de la muestra (el sensor ha de estar en un sitio fijo) y la correcta visualización de los mensajes mostrados en la pantalla. Si bien para este esquemático se ha utilizado una estructura abierta, el montaje podría integrarse perfectamente en una carcasa cerrada o impresa en 3D para aumentar la estabilidad y la portabilidad.

## Software [42]

## ESP32

Uno de los elementos clave en el diseño del software correspondiente a la parte del sistema basada en ESP32 ha sido la modularidad del código [43], la cual consiste en subdividir una aplicación en diferentes bloques funcionales independientes y reutilizables. Esta estrategia facilita el desarrollo y mantenimiento del sistema a la vez que permite una escalabilidad más sencilla, la depuración de errores más eficiente y la incorporación de nuevas funcionalidades.

Cada módulo ha sido desarrollado en lenguaje C [44] e implementado como un archivo .c independiente, acompañado de su correspondiente encabezado .h. Esto permite establecer una separación entre la interfaz y la implementación de cada componente, lo que contribuye a mantener el código organizado y coherente.

## Conexión Wi-Fi

Una parte esencial del sistema es la capacidad del dispositivo de conectarse a la Wi-Fi sin necesidad de reprogramación ni intervención técnica. Para implementar esto, se ha implementado la lógica de conexión siguiente:

1. Conexión automática a redes previamente configuradas
2. Modo de punto de acceso (AP) con interfaz web para introducir las credenciales Wi-Fi
3. Almacenamiento persistente de las credenciales en memoria NVS
4. Conexión validada a internet antes de iniciar la comunicación con ThingsBoard

Cuando el sistema arranca, se ejecuta la función try\_auto\_connect() del módulo **wifi\_ap.c** para comprobar si hay credenciales wifi almacenadas en la memoria no-volátil de la ESP32 (NVS). Para ello se llama a la función load\_wifi\_credentials(char \*ssid, char \*password) que intenta recuperar ambas claves desde el espacio *storage* de la NVS. Si ambas están presentes, se inicia la conexión en modo STA o estación mediante la función connect\_to\_wifi().

Esta función configura la ESP32 como cliente, establece las credenciales y espera a que se obtenga una dirección IP válida mediante un sistema de eventos (IP\_EVENT\_STA\_GOT\_IP). Si transcurridos 10 segundos no ha conseguido conectar, el sistema activa el modo de punto de acceso como mecanismo de recuperación.

En caso de que no existan credenciales Wi-Fi o que la conexión falle, el sistema activa el modo AP ejecutando start\_wifi\_ap(). Esta función configura la ESP32 para crear una red Wi-Fi propia con SSID “ESP32-AP” y contraseña “12345678”. Se inicia además una interfaz de red en modo AP y se lanza un servidor embebido mediante la función start\_webserver().

El módulo **web\_server.c** proporciona una interfaz web integrada, accesible desde cualquier dispositivo que esté conectado al Punto de Acceso (AP). Al abrir la dirección del ESP32 (192.168.4.1), el usuario encuentra un formulario en HTML que permite introducir el SSID y la contraseña de su red Wi-Fi

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

*Ilustración 17: Web de configuración Wi-Fi (Fuente propia)*

El formulario realiza una petición POST a la ruta /connect, que es gestionada por la función post\_handler(). Esta función:

* Extrae las credenciales del cuerpo de la petición.
* Llama a safe\_wifi\_credentials() para almacenarlas en NVS .
* Envía la respuesta al navegador.
* Llama a connect\_to\_wifi() para reiniciar el proceso de conexión con las credenciales recién proporcionadas.

Por último, la gestión de eventos generados por la pila Wi-Fi de la ESP32 se realiza en la función wifi\_event\_handler(). Esta función es invocada por el sistema automáticamente cuando se produce un evento relacionado con la conectividad Wi-Fi, los cuales son:

* WIFI\_EVENT\_STA\_START: Indica que el sistema ha iniciado el intento de conexión como estación y se ejecuta esp\_wifi\_connect() para lanzar el proceso de conexión.
* WIFI\_EVENT\_STA\_CONNECTED: Señala que la conexión al punto de acceso ha sido establecida, aunque aún no se ha recibido una conexión IP.
* WIFI\_EVENT\_STA\_DISCONNECTED: Se genera cuando se pierde la conexión a la red Wi-Fi. En este caso se registra el error en la consola y se intenta reconectar automáticamente llamando a esp\_wifi\_connect().

A continuación se incluye un diagrama en pseudocódigo que resume de forma secuencial la lógica de los módulos **wifi\_ap.c** y **web\_server.c**

INICIO

**Inicializar NVS**

**try\_auto\_connect()**:

si existen credenciales guardadas en NVS:

connect\_to\_wifi(ssid, password) máx 10 segundos:

inicio en modo STA

conectar con las credenciales

si conexión OK:

comprobar acceso a internet

si hay Internet:

iniciar MQTT

si no:

mostrar error

si conexión falla:

start wifi\_ap()

si no hay credenciales:

start\_wifi\_ap()

**start\_wifi\_ap():**

iniciar modo AP

crear red Wi-Fi

iniciar servidor web

**En el navegador del usuario:**

acceder a IP local

introducir SSID y contraseña

**En el servidor web:**

extraer SSID y contraseña

guardar credenciales en NVS

cerrar servidor web

connect\_to\_wifi(ssid, password)

**wifi\_event\_handler():**

manejo constante de eventos

FIN

## Comunicación con el sensor AS7265x

El módulo **as7265x.c** es el encargado de gestionar la comunicación con el sensor homónimo, el cual es el componente central del sistema de adquisición de datos. Esta parte del sistema se encarga de la inicialización y configuración del sensor, así como de realizar lecturas secuenciales de los diferentes canales y devolver los valores al resto de sistema de forma estructurada y lista para su transmisión y análisis.

Inicialización

La inicialización del sensor es un paso fundamental para asegurar que las mediciones tomadas sean estables, reproducibles y adecuadas para su posterior análisis. Dado que este componente opera con múltiples parámetros configurables, es necesario establecer de forma explicita ciertas condiciones de funcionamiento antes de iniciar la adquisición. La función principal de inicialización es as7265\_init(), la cual desarrolla los siguientes pasos:

1. **Verificación de presencia del sensor:** Antes de cualquier configuración, se comprueba la correcta comunicación con el sensor maestro (AS72651), accediendo a sus registros de identificación. Esto se realiza extrayendo los valores contenidos en los registros HW y SW versión, contenidos en las direcciones 0x01 y 0x02, respectivamente. Este paso permite asegurar que el sensor está conectado correctamente y responde a comandos I2C.
2. **Configuración de ganancia:** La ganancia controla el factor de amplificación aplicado a la señal óptica capturada por los fotodetectores del sensor. El AS7265x permite uno de cuatro niveles de ganancia (1x, 3.7x, 16x y 64x), el cual hay que seleccionar escribiendo en los bits 4 y 5 del registro de configuración, ubicado en la dirección 0x04. Para este proyecto, se ha configurado la ganancia en 16x (= b10) ya que se ha tenido en cuenta que las muestras van a reflejar cantidades de luz relativamente bajas, por lo tanto se desea maximizar la sensibilidad del sensor sin saturar las lecturas [45].
3. **Configuración de canales:** En el mismo registro donde se realiza el control de ganancia, en los bits 2 y 3 se controla el modo de captura, para este proyecto se ha elegido el modo 2 (= b10) que permite la lectura continua de los 6 canales de los 3 sensores que conforman el AS7265x.
4. **Configuración del tiempo de integración:** El tiempo de integración define cuanto tiempo se expone el sensor a la luz antes de capturar una medición. El sensor permite configurarlo en pasos de 2.8ms, mediante el registro INTEGRATION\_TIME ubicado en la dirección 0x05, que acepta valores entre 0 y 255. En este caso, se ha utilizado un valor de 59, lo que equivale a un tiempo de integración de 165ms, el cual permite mantener tiempos de respuesta razonables a la vez que se obtiene suficiente intensidad en las lecturas.La integración es un aspecto delicado, ya que un tiempo muy corto puede dar lugar a espectros muy poco definidos mientras que un tiempo muy largo retrasa la adquisición y puede incrementar el ruido ocasionado por la temperatura.

Adquisición

Una vez que este paso se ha realizado correctamente comienza la fase de lectura de los canales espectrales. Este sensor opera, como se ha mencionado anteriormente, sobre un sistema de registros virtuales, lo cual implica que las operaciones de lectura y escritura no se realizan directamente sobre los registros físicos del I2C, sino que deben de seguir un protocolo especial a través de registros intermedios. Esto obliga a:

1. Escribir primero la dirección del registro virtual deseado en un registro físico (0x01).
2. Esperar a que el bit de estado TX\_VALID esté en bajo.
3. Leer o escribir el valor desde el registro 0x02, cuando RX\_VALID esté en alto.

Este protocolo se realiza mediante las funciones write\_virtual\_register(registro, valor) y read\_virtual\_register(registro), las cuales garantizan un acceso seguro y sincronizado al sensor.

Dado que el sensor AS7265x contiene tres dispositivos lógicos, es necesario seleccionar manualmente cada uno antes de iniciar una lectura. Esto se hace mediante la función select\_sensor(sensor), que escribe en el registro DEV\_SEL (Selección de dispositivo) para activar uno de los tres bloques:

* **Sensor 1:** RSTUVW (AS72651)
* **Sensor 2**: GHIJKL (AS72652)
* **Sensor 3**: ABCDEF (AS72653)

Despues de cada cambio de sensor se aplica una pequeña espera de 10 ms para asegurar que los datos están estabilizados antes de leer.

La función read\_sensor\_values(sensor, \*valores), se encarga de activar el sensor deseado, leer los 6 canales mediante la función read\_raw\_value() (la cual lee y combina los registros alto y bajo de cada canal) e imprimir los valores con etiquetas (R:, S:, T: …).

Temperatura

El sensor también permite leer la temperatura interna, muy útil para monitorizar el estado operativo y asegurarse de que opera en unas condiciones adecuadas. La función read\_temperature() accede al registro 0x06 y devuelve su valor en grados Celsius.

Tarea y estado del sensor

Todas estas funciones permiten realizar la tarea cíclica del sensor sensor\_task(). Dicha función es una tarea de FreeRTOS [46], un sistema operativo de tiempo real para dispositivos embebidos, que se mantiene en un bucle constante donde cada segundo:

1. Lee los 18 valores espectrales
2. Obtiene la temperatura interna
3. Verifica los parámetros del sensor
4. Muestra el estado del sensor en la pantalla OLED (Se verá más adelante)
5. Envía los datos a ThingsBoard por MQTT (Se verá más adelante)

Este ciclo asegura una monitorización continua de las muestras con una latencia baja y fiabilidad alta.

## Comunicación con la plataforma ThingsBoard

La comunicación con la plataforma IoT ThingsBoard se realiza en el módulo **thingsboard\_control.c**, el cual se encarga de establecer la conexión con el bróker MQTT, publicar los datos adquiridos durante el proceso anterior y recibir comandos remotos mediante Remote Procedure Calls (RPC).

Conexión MQTT

La función principal que inicializa la comunicación MQTT es mqtt\_app\_start(), la cual se invoca desde **wifi\_ap.c** una vez que la ESP32 se ha conectado a la Wi-Fi. Aquí se define la configuración del cliente mediante una estructura que incluye la URL del bróker de ThingsBoard y el token de acceso.

Este cliente se inicializa mediante esp\_mqtt\_client\_init() y se registra un manejador de eventos (mqtt\_event\_handler()) para responder a sucesos como la conexión, la desconexión o la recepción de los datos.

Una vez iniciado con esp\_mqtt\_client\_start(), el cliente se mantiene activo en segundo plano, manteniendo la sesión abierta con el servidor de ThingsBoard.

Publicación de datos

La función send\_data\_to\_thingsboard\_mqtt() es invocada desde la tarea del sensor cada vez que se realiza una adquisición espectral completa. Su propósito es:

1. Construir un objeto JSON que contiene los 18 canales espectrales etiquetados y la temperatura del sensor.
2. Publicar el JSON al tópico v1/devices/telemetry usando esp\_mqtt\_client\_publish().

Solo se incluyen en el mensaje aquellos canales con valor mayor a cero, lo cual ayuda a reducir el tamaño del payload.

Recepción de comandos remotos RPC

Otra funcionalidad del sistema es la capacidad de controlar remotamente el dispositivo desde la interfaz de Thingsboard. Esto se consigue mediante suscripciones al tópico v1/devices/me/rpc/request/+.

Cuando el bróker recibe un mensaje en este tópico, se dispara el evento MQTT\_EVENT\_DATA, que es manejado por mqtt\_event\_handler\_cb().

El callback analiza el mensaje entrante, lo interpreta como un JSON, y actúa según el método indicado. En el proyecto se ha implementado un método setLed, que permite encender o apagar el LED conectado al GPIO 2 remotamente.

Manejador de eventos MQTT

Por último, la función mqtt\_event\_handler() se encarga de gestionar todos los eventos relevantes del cliente MQTT. Entre ellos:

* MQTT\_EVENT\_CONNECTED: Se registra la conexión y se suscribe automáticamente al tópico de RPC.
* MQTT\_EVENT\_DISCONNECTED: Se muestra un mensaje de error en la pantalla OLED y se registra el evento.
* MQTT\_EVENT\_DATA: Se llama a mqtt\_event\_handler\_cb() para procesar comandos RPC.

## Visualización en pantalla OLED

El módulo oled.c implementa la interfaz gráfica básica para el usuario mediante la pantalla OLED mencionada durante el apartado de hardware. Esta pantalla, basada en el controlador SSD1306, proporciona una visualización en tiempo real del estado del sistema, incluyendo el tiempo de ejecución y el estado de la conexión Wi-Fi, del MQTT y del sensor.

Inicialización

La inicialización se realiza mediante la función oled\_init(), que llama a ssd1306\_init(), la cual envía al controlador una secuencia de comandos siguiendo las especificaciones del datasheet del SSD1306 [47]. Una vez inicializado, se llama a ssd1306\_clear() para borrar completamente el contenido de la pantalla.

Comunicación I2C

La comunicación con la pantalla se realiza a través del bus I2C, utilizando funciones de bajo nivel para:

* Enviar comandos (ssd1306\_send\_cmd) identificados por el byte de control 0x00
* Enviar datos gráficos (ssd1306\_send\_data) con el byte de control 0x40

Estos comandos y datos se estructuran en bloques utilizando el protocolo i2c\_master de ESP-IDF. La pantalla está segmentada por páginas (de 8 pixeles verticales), lo que permite enviar cadenas de bytes que representen caracteres o imágenes.

Escritura de texto

La visualización de caracteres se realiza mediante una fuente fija de 8x8 píxeles, definida en font8x8\_basic [48].

* La función ssd1306\_draw\_char(offset, página, char) permite posicionar y dibujar un carácter individual.
* La función ssd1306\_draw\_string(offset\_inicio, página, cadena) permite imprimir cadenas completas.

La pantalla se divide en 8 páginas verticales, que pueden contener hasta 16 caracteres de 8 píxeles de alto.

Interfaz de usuario (HUD)

El sistema proporciona una capa para mostrar distintos estados a través de funciones específicas:

* Hud\_display\_message(msg, página): Muestra un mensaje personalizado en la línea (entera) indicada.
* Hud\_display\_time(h,m,s): Muestra el tiempo de ejecución en la esquina superior izquierda.
* Hud\_display\_wifi(connected): Indica si la conexión Wi-Fi está activa (Wi-Fi ON) o caída (Wi-Fi OFF) en la página 5.
* Hud\_display\_sensor\_status(ok): Indica si el sensor AS7265x está correctamente configurado, mostrando SENSOR OK o SENSOR ERROR en la página 6.

Estas funciones se apoyan en las funciones de escritura comentadas en el punto anterior, borrando primero el área antes de escribir el nuevo contenido para evitar superposiciones gráficas.

Tarea de actualización periódica

Con el objetivo de actualizar periódicamente el contenido visual, la tarea independiente de FreeRTOS oled\_hud\_task() se lanza desde **main.c**. Esta tarea muestra una línea divisoria entre el tiempo de ejecución (el cual se actualiza cada segundo gracias a la función gmtime\_r()) y el resto de los mensajes. Permitiendo al usuario tener una indicación continua del funcionamiento del dispositivo sin necesidad de interfaz gráfica compleja.

Mano sosteniendo un aparato electrónico

El contenido generado por IA puede ser incorrecto.

*Ilustración 18: Visualización del HUD (Fuente propia*

## ThingsBoard

## Visualización y control

La interfaz de Thingsboard permite crear dashboards personalizados, lo cual ha sido aprovechado para mostrar en tiempo real:

* Los valores de cada canal espectral en un diagrama temporal y un gráfico de bloques.
* La temperatura interna del sensor.
* Un botón virtual para activar o desactivar el LED del dispositivo mediante RPC.

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.

*Ilustración 19: Dashboard del sistema en ThingsBoard* [49]

## Cadenas de reglas

Las Rule Chains (Cadenas de Reglas) en ThingsBoard son mecanismos de automatización que permiten procesar y reaccionar a los datos que llegan desde los diferentes dispositivos IoT. Se trata de una arquitectura visual basada en nodos que definen cómo deben de tratarse los diferentes mensajes entrantes [50]

En este proyecto, las Rule Chains se han utilizado para:

* Automatizar un envío de alertas a Telegram cuando se detecta una temperatura elevada en el sensor
* Enviar telemetrías a un servidor externo para su procesamiento y evaluación (Se verá más adelante).

Alertas a Telegram

La comunicación con Telegram se realiza mediante un bot personalizado, creado con el servicio oficial de Telegram llamado BotFather. El proceso para su creación es el siguiente [51]:

1. Acceder a @BotFather en telegram.
2. Ejecutar el comando /newbot y proporcionar un nombre y nombre de usuario único (tiene que acabar en \_bot).
3. Guardar el token de autenticación que genera el bot
4. Crear un canal e invitar al bot añadiéndolo como administrador.
5. Ir a la dirección “https://api.telegram.org/bot{{TOKEN}}/getUpdates” sustituyendo el token por el token que ha generado el bot.
6. Publicar un mensaje en el canal para que el bot pueda ver que está añadido.

Una vez se ha creado y añadido el bot a un canal, hay que crear una cadena de reglas en ThingsBoard que envie un mensaje a la api de telegram (con el token del bot) cuando la temperatura es demasiado alta como para garantizar un buen funcionamiento del sensor (>30ºC).

Diagrama

El contenido generado por IA puede ser incorrecto.

*Ilustración 20: Cadena de Reglas para el control de la temperatura (Fuente Propia)*

* **Control Temp** sirve de filtro lógico para que únicamente continue la cadena de reglas cuando el mensaje de temperatura es superior a 30ºC.
* **Telegram MSG** crea el cuerpo del mensaje, donde se incluye la temperatura actual del sensor, que va a ser enviado a Telegram.
* **Envio Mensaje** realiza la publicación a la api de telegram “https://api.telegram.org/bot{{TOKEN}}/sendMessage”.

Tras configurar la cadena de reglas, las alertas llegan a Telegram, creando así un sistema de avisos en tiempo real que funciona en segundo plano.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

*Ilustración 21: Canal de difusión de alertas creado en Telegram (Fuente propia)*

Envio de telemetrías a servidor externo

asdasd

# Bibliografía

|  |  |
| --- | --- |
| [1] | J. A. Sanchez, «.laopiniondemurcia,» 05 Diciembre 2024. [En línea]. Available: https://www.laopiniondemurcia.es/comunidad/2024/12/05/murcia-riesgo-abandono-tierras-sequia-112335615.html. |
| [2] | UAM, «madrimasd,» 11 03 2025. [En línea]. Available: https://www.madrimasd.org/notiweb/noticias/cianobacterias-toxicas-en-parque-nacional-monfrague-un-desafio-salud-ecologica. |
| [3] | F. y. B. Instituto Europeo de Química, «ieqfb,» [En línea]. Available: https://ieqfb.com/que-es-la-espectroscopia-tipos-y-tecnicas/. |
| [4] | N/A, «Espectroscopía infraroja,» [En línea]. Available: https://es.wikipedia.org/wiki/Espectroscop%C3%ADa\_infrarroja. |
| [5] | Wikipedia, «Espectroscopía del infrarojo cercano,» [En línea]. Available: https://es.wikipedia.org/wiki/Espectroscop%C3%ADa\_del\_infrarrojo\_cercano. |
| [6] | D. D. v. Staveren, «NIR vs IR: What is the difference?,» [En línea]. Available: https://www.metrohm.com/en\_gb/discover/blog/2024/nir-vs-ir.html. |
| [7] | «Google Scholar,» [En línea]. Available: https://scholar.google.es/. |
| [8] | «Google,» [En línea]. Available: https://www.google.com/. |
| [9] | M. M. J. D. B. H. R. A.M. Mouazen, «On-line measurement of some selected soil properties using a VIS–NIR sensor,» Marzo 2007. [En línea]. Available: https://www.sciencedirect.com/science/article/pii/S0167198706000663. |
| [10] | K. B. B. N. U. C. W. H. Justyna Grabska, «MDPI,» 10 Mayo 2023. [En línea]. Available: https://www.mdpi.com/2304-8158/12/10/1946. |
| [11] | [En línea]. Available: https://www.oceanoptics.com/spectrometers/. |
| [12] | [En línea]. Available: https://felixinstruments.com/food-science-instruments/nir-spectroscopy/f-750-produce-quality-meter/. |
| [13] | [En línea]. Available: https://www.ti.com/tool/DLPNIRNANOEVM#description. |
| [14] | [En línea]. Available: https://sciolatam.com/. |
| [15] | [En línea]. Available: https://www.sparkfun.com/sparkfun-triad-spectroscopy-sensor-as7265x-qwiic.html. |
| [16] | «Sistema embebido,» [En línea]. Available: https://es.wikipedia.org/wiki/Sistema\_embebido. |
| [17] | N/A, «¿Que es un sistema embebido?,» [En línea]. Available: https://www.ceupe.com/blog/sistema-embebido.html. |
| [18] | «ResearchGate,» [En línea]. Available: https://www.researchgate.net/figure/Componentes-de-la-tarjeta-del-sistema-embebido-Fuente-Autores-Los-tres-sistemas\_fig3\_319148610. |
| [19] | [En línea]. Available: https://www.raspberryme.com/premiers-pas-avec-la-carte-de-developpement-esp32/. |
| [20] | [En línea]. Available: https://www.az-delivery.de/es/products/esp32-developmentboard. |
| [21] | TeachMeMicro. [En línea]. Available: https://www.teachmemicro.com/esp32-pinout-diagram-wroom-32/. |
| [22] | «Página Home de ESP-IDF,» [En línea]. Available: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/index.html. |
| [23] | «Portfolio del sensor AS7265x,» [En línea]. Available: https://ams-osram.com/products/sensor-solutions/ambient-light-color-spectral-proximity-sensors/ams-as7265x-smart-spectral-sensor. |
| [24] | «Hookup Guide AS7265x Sparkfun,» [En línea]. Available: https://learn.sparkfun.com/tutorials/spectral-triad-as7265x-hookup-guide. |
| [25] | «Documento de producto del AS7265x,» [En línea]. Available: https://look.ams-osram.com/m/98f8b5adf940b726/original/AS7265x\_UG000337\_2-00.pdf. |
| [26] | «Bus I2C,» [En línea]. Available: https://es.wikipedia.org/wiki/I%C2%B2C. |
| [27] | «Descripción y funcionamiento del bus I2C,» [En línea]. Available: https://robots-argentina.com.ar/didactica/descripcion-y-funcionamiento-del-bus-i2c/. |
| [28] | «Fotografía bus I2C,» [En línea]. Available: https://www.prometec.net/bus-i2c/. |
| [29] | E. avanzada, «youtube,» [En línea]. Available: https://www.youtube.com/watch?v=uyCvC4aeqEI. |
| [30] | AWS, «¿Que es MQTT?,» [En línea]. Available: https://aws.amazon.com/es/what-is/mqtt/. |
| [31] | «Alotcer,» [En línea]. Available: https://www.alotceriot.com/es/revelando-el-mundo-de-mqtt-una-guia-completa-para-el-protocolo-de-mensajeria-ligero/. |
| [32] | «IoT (Internet de las Cosas),» [En línea]. Available: https://es.wikipedia.org/wiki/Internet\_de\_las\_cosas. |
| [33] | «Aprendizaje Automático (Machine Learning),» [En línea]. Available: https://es.wikipedia.org/wiki/Aprendizaje\_autom%C3%A1tico. |
| [34] | «What Is Machine Learning? IBM,» [En línea]. Available: https://www.ibm.com/think/topics/machine-learning. |
| [35] | «Requisitos funcionales y no funcionales,» [En línea]. Available: https://visuresolutions.com/es/requirements-management-traceability-guide/functional-vs-non-functional-requirements/. |
| [36] | T. Asana, «Metodologías para la gestión de proyectos,» [En línea]. Available: https://asana.com/es/resources/project-management-methodologies. |
| [37] | L. Bennet, «¿Qué es el modelo de creación de prototipos?,» [En línea]. Available: https://www.guru99.com/es/software-engineering-prototyping-model.html. |
| [38] | A. Stepanets, «Qué es un hito y cuáles son los ejemplos de hitos de un proyecto,» [En línea]. Available: https://blog.ganttpro.com/es/que-es-un-hito-y-que-ejemplos-de-hitos-hay/. |
| [39] | «Home Page de Fritzing,» [En línea]. Available: https://fritzing.org/. |
| [40] | «Logo de fritzing,» [En línea]. Available: https://commons.wikimedia.org/wiki/File:Fritzing\_logo\_%28new%29.png. |
| [41] | «Fritzing en Wikipedia,» [En línea]. Available: https://en.wikipedia.org/wiki/Fritzing. |
| [42] | Á. Lameiro, «Repositorio de Github donde se encuentra todo el código,» [En línea]. Available: https://github.com/ALameiroGarcia/TFG-2. |
| [43] | «Modularidad del software,» [En línea]. Available: https://es.wikipedia.org/wiki/Modularidad\_(inform%C3%A1tica). |
| [44] | «Lenguaje C,» [En línea]. Available: https://es.wikipedia.org/wiki/C\_(lenguaje\_de\_programaci%C3%B3n). |
| [45] | «Datasheet AS7265x,» [En línea]. Available: https://cdn.sparkfun.com/assets/c/2/9/0/a/AS7265x\_Datasheet.pdf. |
| [46] | «FreeRTOS,» [En línea]. Available: https://www.freertos.org/. |
| [47] | «Datasheet SSD1306,» [En línea]. Available: https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf. |
| [48] | dhepper, «Código de font8x8\_basic,» [En línea]. Available: https://github.com/dhepper/font8x8/blob/master/font8x8\_basic.h. |
| [49] | «Página demo de Thingsboard,» [En línea]. Available: https://demo.thingsboard.io/. |
| [50] | «Rule Chains en ThingsBoard,» [En línea]. Available: https://thingsboard.io/docs/user-guide/ui/rule-chains/. |
| [51] | «Crear un Bot en Telegram usando BotFather,» [En línea]. Available: https://help.nethunt.com/en/articles/6253243-how-to-make-an-api-call-to-the-telegram-channel. |

Anexos