
UNIVERSIDAD DE VALLADOLID
E.T.S.I DE TELECOMUNICACIÓN

TRABAJO FINAL DE GRADO
GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

# Plataforma IoT para convertir un frigorífico convencional en uno inteligente

Autor: Ignacio González Domínguez
Tutor: Alfonso Bahillo Martínez

11 de septiembre de 2023

| | |
|---|---|
| TÍTULO: | Plataforma IoT para convertir un frigorífico convencional en uno inteligente |
| AUTOR: | Ignacio González Domínguez |
| TUTOR: | Alfonso Bahillo Martínez |
| DEPARTAMENTO: | Teoría de la Señal y Comunicaciones e Ingeniería Telemática |

# TRIBUNAL

| | |
|---|---|
| PRESIDENTE: | D. Rubén M. Lorenzo Toledo |
| VOCAL: | Da. Noemí Merayo Álvarez |
| SECRETARIO: | D. Alfonso Bahillo Martínez |
| SUPLENTE: | D. Juan C. Aguado Manzano |
| SUPLENTE: | D. Ramón J. Durán Barroso |
| FECHA: | Día de la lectura |

| | |
|---|---|
| CALIFICACIÓN: | |

# Resumen

El monitoreo de los hábitos alimenticios es esencial para las personas con enfermedades mentales, ya que influye directamente en su tratamiento y en la mejora de su calidad de vida. En el mercado existen frigoríficos inteligentes que permiten un monitoreo avanzado, sin embargo, su elevado coste los hace inaccesibles para muchos. Frente a este problema, este Trabajo Fin de Grado ofrece una solución innovadora y económica: un sistema IoT que captura y almacena imágenes del contenido del frigorífico cuando se abre la puerta para su posterior análisis, permitiendo un seguimiento en remoto de los hábitos alimenticios de la persona a través de la evolución del contenido del mismo.

El diseño y desarrollo del sistema presentó desafíos, sobre todo en la integración de sus componentes para garantizar que cumplían con las funciones propuestas. Como resultado, se obtuvo un sistema que no solo cumple con las expectativas de monitoreo, sino que también representa una alternativa barata en comparación con las opciones existentes en el mercado.

A pesar de su eficacia, el sistema tiene áreas de mejora y se plantean futuras investigaciones. Estas incluyen la integración de inteligencia artificial para evaluar de forma automática el contenido de las imágenes, la creación de una interfaz de usuario y la mejora de la eficiencia energética.

PALABRAS CLAVE: Internet de las Cosas (IoT), Raspberry Pi, monitorización de frigoríficos, Captura de imágenes.

# Abstract

Monitoring eating habits is essential for people with mental illness, as it directly influences their treatment and improves their quality of life. In the market there are smart refrigerators that allow advanced monitoring, however, their high cost makes them inaccessible to many. Faced with this problem, this Final Degree Project offers an innovative and economical solution: an IoT system that captures and stores images of the refrigerator's contents when the door is opened for later analysis.

The design and development of the system presented challenges, especially in the integration of its components to ensure that they fulfilled the proposed functions. The result was a system that not only meets monitoring expectations, but also represents an inexpensive alternative compared to existing options on the market.

Despite its effectiveness, the system has areas for improvement and future research is planned. These include integrating artificial intelligence to evaluate image content, creating a user interface and improving energy efficiency.

KEYWORDS: Internet of the Things (IoT), mental illness, refrigerator monitoring, image caputure.

# Agradecimientos

En este momento me gustaría recordar y agradecer a todos aquellos que me han ayudado en esta etapa de mi vida.

En primer lugar me gustaría nombrar a la Universidad de Valladolid por ofrecerme las herramientas y el conocimiento para completar este desafío. También me gustaría agradecer a mi tutor de TFG Alfonso Bahillo Martínez por su paciencia y orientación que han sido vitales para el desarrollo de este TFG. Por último me gustaría agradecer el apoyo de mi familia y amigos durante todo el desarrollo del grado, que me han aconsejado y ayudado durante todo este tiempo, sin su ayuda no hubiera sido posible completar el grado.

# ÍNDICE GENERAL

**Índice de figuras** **xiii**

**1. Introducción** **1**
1.1. Contexto y motivación . . . . . 1
1.2. Importancia del monitoreo de hábitos alimenticios en personas con problemas de salud mental . . . . . 2
1.3. Relación entre las partes de IA e IoT . . . . . 3
1.4. Objetivos del proyecto . . . . . 3

**2. Estado del arte** **7**
2.1. Internet de las cosas (IoT) . . . . . 7
2.2. Aplicaciones de IoT en la industria de la alimentación y la salud . . . . 8
2.3. Beneficios potenciales del sistema propuesto . . . . . 9
2.4. Plataformas y tecnologías relacionadas . . . . . 10

**3. Diseño y desarrollo del sistema IoT** **11**
3.1. Componentes del sistema . . . . . 12
3.1.1. Raspberry Pi . . . . . 12
3.1.2. Cámara USB . . . . . 12
3.1.3. Sensor de puerta . . . . . 13
3.2. Arquitectura general . . . . . 13
3.3. Implementación . . . . . 15

**4. Desarrollo del software** **21**
4.1. Herramientas software utilizadas . . . . . 22
4.1.1. Raspberry Pi OS . . . . . 22
4.1.2. Debian . . . . . 22

4.1.3. deCONZ . . . 22
4.1.4. ZigBee . . . 23
4.1.5. API . . . 24
4.1.6. HTTP . . . 24
4.1.7. JSON . . . 24
4.1.8. Python . . . 25
4.1.9. VNC . . . 25
4.2. Script de control . . . 26
4.2.1. Script modo ráfaga . . . 27
4.2.2. Script modo vídeo . . . 30

**5. Pruebas y aprendizaje** **33**
5.1. Pruebas . . . 34
5.1.1. Pruebas usando Raspberry Pi . . . 34
5.1.2. Pruebas con la cámara: . . . 34
5.1.3. Pruebas con la API: . . . 35
5.1.4. Primer programa con el interruptor: . . . 35
5.1.5. Ubicación de la cámara: . . . 36
5.1.6. Sustitución de interruptor por sensor de puerta: . . . 36
5.1.7. Código para sensor de puerta: . . . 36
5.1.8. Pruebas del sistema completo: . . . 37
5.1.9. Pruebas de largo funcionamiento: . . . 37
5.1.10. Pruebas de frío: . . . 37
5.1.11. Pruebas de eficiencia energética: . . . 38

**6. Conclusiones** **41**

**7. Trabajo futuro** **43**
7.1. Integración con Inteligencia Artificial . . . 43
7.2. Desarrollo de una interfaz de usuario . . . 44
7.3. Optimización de la eficiencia energética . . . 44
7.4. Seguridad de los datos . . . 44

**A. Anexos** **45**
A.1. Código modo ráfaga . . . 45
A.2. Código modo vídeo . . . 45

**Bibliografía** **47**

# ÍNDICE DE FIGURAS


Figura 1.1. Esquema de las dos partes juntas . . . . . 5

Figura 2.1. IoT en la salud . . . . . 8

Figura 3.1. Raspberry Pi 4 Model B . . . . . 12
Figura 3.2. Logitech Brio 4K . . . . . 13
Figura 3.3. Sensor de puerta Aqara . . . . . 14
Figura 3.4. ConBee II . . . . . 14
Figura 3.5. Diagrama de la arquitectura . . . . . 15
Figura 3.6. Ubicación de la cámara . . . . . 16
Figura 3.7. Ubicación del sensor de puerta . . . . . 17
Figura 3.8. Problema con el cable . . . . . 18
Figura 3.9. Ubicación Raspberry Pi . . . . . 19
Figura 3.10. Sistema final . . . . . 19
Figura 3.11. Foto del interior del frigorífico . . . . . 20

Figura 4.1. Ejemplo de red deCONZ . . . . . 23
Figura 4.2. ZigBee . . . . . 23
Figura 4.3. Python . . . . . 25
Figura 4.4. VNC . . . . . 26
Figura 4.5. Diagrama de flujo software . . . . . 26
Figura 4.6. Conexión HTTP . . . . . 27
Figura 4.7. Solicitud de estado . . . . . 28
Figura 4.8. Análisis de la respuesta . . . . . 28
Figura 4.9. Captura de imágenes con fswebcam . . . . . 29
Figura 4.10. Configuración de vídeo . . . . . 31

Figura 4.11. Rotar frame . . . . . 31

Figura 5.1. Primera foto usando una Raspberry Pi . . . . . 34
Figura 5.2. Interruptor inteligente . . . . . 35
Figura 5.3. Secuencia de pruebas . . . . . 37

CAPÍTULO 1

# INTRODUCCIÓN


1.1. Contexto y motivación . . . . . 1
1.2. Importancia del monitoreo de hábitos alimenticios en personas con problemas de salud mental . . . . . 2
1.3. Relación entre las partes de IA e IoT . . . . . 3
1.4. Objetivos del proyecto . . . . . 3


La monitorización de los hábitos alimenticios es una de las variables más relevantes a la hora de observar la evolución en pacientes con problemas de salud mental. Este proyecto desarrolla una plataforma para conseguir esta monitorización, que tiene el potencial de ser usada en conjunto de otras tecnologías como la inteligencia artificial para conseguir un seguimiento automático del contenido del frigorífico. En esta memoria se desarrolla el diseño, desarrollo e implementación de la plataforma IoT.

## 1.1. Contexto y motivación

Los trastornos mentales afectan a una parte significativa de la población, en España al menos el 9 % de las personas lo padecen, y su seguimiento es clave para proporcionar una mejora en la calidad de vida de los afectados [34]. Tener hábitos de vida saludables, como por ejemplo una alimentación variada y equilibrada, y realizar actividad física de manera regular, es un aspecto determinante a la hora de prevenir y tratar esta

clase de trastornos [10]. Sin embargo, conseguir monitorear estos hábitos es una tarea complicada tanto para los profesionales sanitarios como para los propios pacientes [8]. Para conseguir una solución a este problema se puede recurrir a las tecnologías de la información y la comunicación (TIC) [9], como la IoT (Internet of the Things) que permite la interconexión y el intercambio de información entre dispositivos, lo que facilita la recopilación de datos del estilo vida y los hábitos alimenticios de las personas [3].

El objetivo principal de este proyecto es ayudar a las personas que padecen de problemas de salud mental a controlar sus hábitos alimenticios mediante la creación de una plataforma IoT que monitorice de forma automática el contenido del frigorífico a lo largo de los días y semanas [21]. Este sistema consta de un Raspbery Pi, una cámara USB y un sensor de puerta, que captura imágenes del interior del frigorífico cada vez que se abre la puerta.

## 1.2. Importancia del monitoreo de hábitos alimenticios en personas con problemas de salud mental

La esperanza de vida de las personas con trastornos mentales suele ser menor que la de un individuo sano [8], estos trastornos pueden estar ligados a estilos de vida poco saludables, los factores propios de la enfermedad (es decir si la una persona cumple las condiciones para desarrollar este problema) y un menor acceso a una atención médica continua y adecuada a sus necesidades. Por lo cual seguir unos hábitos alimenticios saludables que pueden afectar el curso de la enfermedad, ayudar a controlar los síntomas y hacer menos probable la aparición de nuevas enfermedades relacionadas como la diabetes, la obesidad y problemas cardiovasculares [9].

Hoy por hoy se usan estrategias cognitivo-conductuales para conseguir que una persona que no tiene uno estilo de vida saludable cambie su comportamiento [3]. Estas estrategias pueden verse limitadas cuando el comportamiento que se quiere cambiar es complejo, como es el caso de la alimentación, para ayudar a que estas estrategias sean más completas y personales se esta recurriendo a la tecnología, como puede ser las aplicaciones móviles o la incorporación de IoT y así conseguir una mayor interacción con los pacientes [21].

Teniendo en cuenta todo lo anterior se puede entender que una plataforma IoT que permita la monitorización del contenido de un frigorífico de una persona con una enfermedad mental puede aportar gran cantidad de datos útiles que tienen el potencial de ser utilizados por los profesionales sanitarios para conseguir progresos en el tratamiento de estas enfermedades y mejorar su calidad de vida.

## 1.3. Relación entre las partes de IA e IoT

Este proyecto, Plataforma de IoT para la monitorización del contenido de un frigorífico, está pensado para que trabaje en conjunto con otro proyecto de IA (inteligencia artificial) de tal manera que la solución final que se ofrezca sea mucho más completa.

El objetivo de la parte de IoT es conseguir capturar imágenes de forma automática, sin intervención por parte del usuario, (que pueden ser tanto fotos como vídeos) del interior de un frigorífico cada vez que se abre la puerta. Esta parte tiene que asegurar que los datos están disponibles para que la parte de IA pueda analizarlos y evaluar la evolución de los hábitos alimenticios.

La parte de IA es la responsable de procesar todas estas imágenes conseguidas por la parte de IoT. La IA es capaz de detectar los alimentos presentes en las imágenes capturadas y mediante el uso de algoritmos y aprendizaje automático detectar patrones de comportamiento que muestren de forma automática la evolución en el contenido del frigorífico, lo cual será reflejo de los hábitos alimenticios del paciente.

Para intentar ilustrar mejor cómo se relacionan las dos partes, en la figura 1.1 se muestra un esquema de cómo funcionan en conjunto.

## 1.4. Objetivos del proyecto

Los objetivos del proyecto son los siguientes:

- Desarrollar una plataforma que integre tecnologías para capturar imágenes y grabar vídeos del interior de un frigorífico cada vez que se abra la puerta de forma automática, sin intervención por parte del usuario, (usando una Raspberry Pi, un sensor de puerta y una cámara USB). Este es el objetivo principal y más importante del proyecto.

- Conseguir un dispositivo que se pueda adaptar a un tamaño de frigorífico estándar, para luego ampliar el proyecto de forma que se pueda adaptar a diferentes tamaños de frigoríficos. En este aspecto el reto principal sería donde posicionar la cámara de tal manera que se capture la mayor parte del contenido.

- Asegurar que las imágenes capturadas sean lo suficientemente claras y nítidas como para identificar los elementos del interior del frigorífico de forma precisa.

- Optimizar el almacenamiento asegurando que el tamaño de las imágenes sea el mínimo necesario manteniendo la calidad requerida.

- Diseñar el sistema para que su operación diaria sea intuitiva y no interfiera con las actividades cotidianas del usuario.

- Diseñar el sistema de modo que permita una integración directa y eficiente con módulos de IA dedicados al reconocimiento de objetos en las imágenes capturadas. Esta integración se considerará exitosa si, en etapas futuras, se puede implementar un módulo de IA que reconozca los alimentos en las fotografías con mínimas modificaciones al sistema original.

Con el cumplimiento de todos estos objetivos, el proyecto sentará unas bases muy sólidas para que en el futuro se pueda ir actualizando con facilidad. Cómo se puede observar al leer los objetivos este proyecto tiene dos partes bien diferenciadas, una de ingeniería general y otra de programación.

En la parte de ingeniería general se ha de encontrar una solución al problema de dónde colocar la cámara o conjunto de cámaras para obtener imágenes del contenido del frigorífico, y por otro lado en la parte de programación se ha de conseguir un código en Python que sea capaz de detectar la apertura del sensor de puerta, comandar a la cámara que comience a tomar instantáneas o grabar vídeos y almacenar estas imágenes en la memoria de la Raspberry Pi.

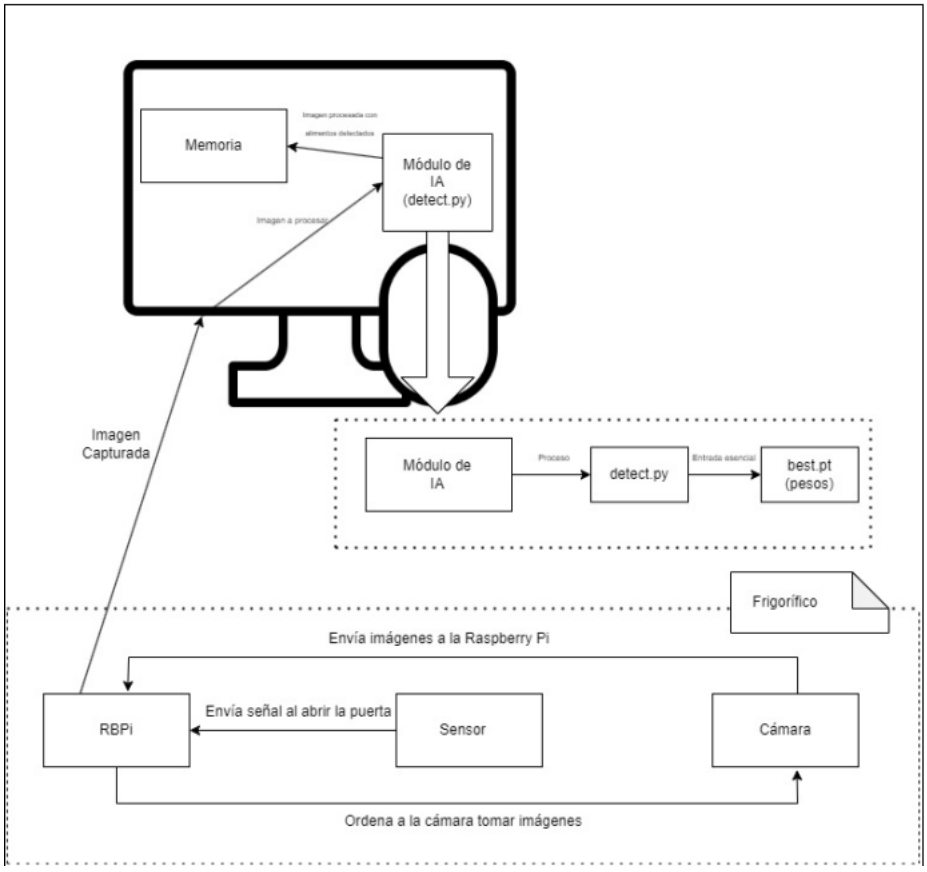


**Figura 1.1.:** Esquema de las dos partes juntas

CAPÍTULO 2

# ESTADO DEL ARTE


2.1. Internet de las cosas (IoT) . . . . . 7
2.2. Aplicaciones de IoT en la industria de la alimentación y la salud . . . . 8
2.3. Beneficios potenciales del sistema propuesto . . . . . 9
2.4. Plataformas y tecnologías relacionadas . . . . . 10


Se realizará un resumen de los cambios más recientes en los campos relacionados con este proyecto. En particular, nos concentraremos en el Internet de las cosas (IoT), sus aplicaciones en la industria de la alimentación y la salud, así como las posibles ventajas del sistema propuesto, y las plataformas y tecnologías relacionadas.

## 2.1. Internet de las cosas (IoT)

El término "Internet de las Cosas" o en inglés "Internet of the Things" define a un conjunto de dispositivos físicos, vehículos, electrodomésticos y objetos que tienen sensores, software, hardware y otras tecnologías que les permiten conectarse e interactuar con otros dispositivos o con Internet [35]. Estos dispositivos van desde elementos cotidianos como pueden ser frigoríficos, termostatos hasta sistemas complejos usados en la industria. Muchos dispositivos de Internet de las Cosas son capaces de tomar decisiones basadas en el procesamiento de los datos que previamente han recogido del entorno en el que se encuentran [7].

Gracias a la mejora de las capacidades de procesamiento y conectividad, así como a la reducción de los costos de los sensores y la transmisión de datos, Internet de las cosas ha experimentado un crecimiento exponencial en los últimos diez años [2]. Se prevé que para 2025, habrá más de 75 mil millones de dispositivos de Internet de las cosas en todo el mundo [32].

## 2.2. Aplicaciones de IoT en la industria de la alimentación y la salud

En el ámbito de la industria de la alimentación y la salud es una de las muchas que se han visto afectadas positivamente por la aparición y el desarrollo del internet de las cosas. En el ámbito de la alimentación, las cadenas de suministros están sufriendo una actualización importante, porque se están añadiendo sensores y dispositivos que son capaces de seguir las condiciones de los alimentos. Con esto se consigue una trazabilidad completa y se alcanza un mayor nivel de seguridad alimentaria [35].

**Figura 2.1.:** IoT en la salud

La industria de la salud también ha experimentado un salto de calidad desde la aparición de IoT, ya que se ha conseguido la prestación de atención medica más personal gracias a la monitorización remota de los pacientes y a la recopilación de datos en tiempo real. Un ejemplo de esto es el uso de dispositivos wearables en los pacientes, que no son más que dispositivos que las personas pueden llevar puestos (por ejemplo, una pulsera), permitiendo recolectar datos sobre indicadores de salud. Estos datos pueden ser utilizados para detectar enfermedades en etapas tempranas, antes

de que se vuelvan un problema mayor, manejar enfermedades crónicas y promover estilos de vida saludables [2].

Este proyecto se centra solo en la parte de IoT, pero el resultado final combinará IoT con IA, lo cual tiene el potencial de mejorar los hábitos alimenticios en las personas que sufren este tipo enfermedades mentales. Al monitorear y almacenar los hábitos alimenticios con una nevera inteligente y analizar estos datos con IA, se pueden ofrecer soluciones más adaptadas a cada persona que promuevan una alimentación equilibrada y ayuden a mejorar la salud mental de los pacientes, así como estudiar el desarrollo de la enfermedad y poder relacionarlo con los hábitos alimenticios.

## 2.3. Beneficios potenciales del sistema propuesto

Con lo expuesto anteriormente se puede intuir que este sistema tiene una serie de beneficios, los principales están resumidos en:

- Personalización y monitoreo regular: Este sistema recopila información cada vez que el usuario abre la puerta, lo que ofrece unos datos actualizados y muy precisos en cuanto a qué alimentos está consumiendo el paciente.

- Toma de decisiones: El sistema permite a los profesionales sanitarios ofrecer recomendaciones y tomar decisiones sobre el tratamiento en base a los datos adquiridos, consiguiendo una atención sanitaria más personal y un impacto positivo tanto en la salud física como mental del individuo [35][2].

- Convivencia: Esta plataforma está pensada para que actúe de manera transparente al usuario. Es decir, el paciente puede seguir realizando sus acciones cotidianas con normalidad.

- Investigación: Este proyecto puede ser de gran utilidad a la hora de investigar patrones de comportamiento en el ámbito de la alimentación en personas con enfermedades mentales [2].

## 2.4. Plataformas y tecnologías relacionadas

El avance y la incorporación de IoT en las vidas cotidianas de las personas se consigue gracias al desarrollo de una serie de tecnologías y plataformas. A continuación, se enumeran unas de las más importantes:

- Microcontroladores: En este proyecto se usa una Raspberry Pi, sus características principales son la versatilidad, la capacidad de computación y su gran comunidad de usuarios, como microcontrolador. En los sistemas IoT los microcontroladores son los encargados de la recopilación de datos de los sensores y la comunicación con otros dispositivos [20].

- Sensores: Son los encargados de detectar lo que está ocurriendo en el medio. En este proyecto se usa un sensor de puerta para detectar cuando la puerta del frigorífico se abre y se cierra [7].

- Tecnologías de comunicación inalámbrica: Estas tecnologías como pueden ser el Bluetooth o el Wi-Fi permiten la comunicación entre diferentes sistemas IoT. Además, estas tecnologías han experimentado un gran avance en términos de seguridad, alcance y velocidad últimamente [2].

- Plataformas en la nube: Aunque este proyecto este pensado para que se ejecute en su totalidad de forma local en la Raspberry Pi, existen otros muchos proyectos que utilizan los servicios de la nube porque ofrecen una gran capacidad de almacenamiento, así como de procesamiento y análisis de datos [35].

- Inteligencia Artificial: En los últimos años la IA está experimentando una evolución exponencial siendo capaz de hacer cada vez tareas más complejas, se puede usar para reconocer los elementos de una imagen. La integración de la IA a los sistemas IoT permite conseguir soluciones más completas [22].

CAPÍTULO 3

# DISEÑO Y DESARROLLO DEL SISTEMA IOT


3.1. Componentes del sistema . . . . . 12
3.1.1. Raspberry Pi . . . . . 12
3.1.2. Cámara USB . . . . . 12
3.1.3. Sensor de puerta . . . . . 13
3.2. Arquitectura general . . . . . 13
3.3. Implementación . . . . . 15


El diseño y desarrollo de un sistema IoT, es la arquitectura de un conjunto de dispositivos conectados mediante electrónica, software, hardware. La arquitectura de un sistema IoT incluye dispositivos de borde como sensores o actuadores, conectividad, procesamiento de datos, además una interfaz de usuario, la seguridad, la interoperabilidad y la escalabilidad también son aspectos importantes. Para conseguir un sistema óptimo se necesita un análisis de cada uno de los componentes del sistema.

Este proyecto crea un sistema para ayudar a monitorear los hábitos alimenticios en personas con problemas de salud mental. Se han seleccionado unos componentes que pueden realizar las tareas requeridas y se ha implementado una arquitectura que permita que estos componentes se adapten a un frigorífico y sean capaces de obtener imágenes de su interior

## 3.1. Componentes del sistema

### 3.1.1. Raspberry Pi

- Raspberry Pi: en este proyecto se usa la Raspberry Pi 4 Model B que cuenta con un procesador Quad-core Cortex-A72 de 64 bits y 4 GB de RAM [19]. Este dispositivo ofrece una buena relación coste-rendimiento y actua como el centro de procesamiento del sistema. Gestiona los datos recogidos por la cámara y el sensor de puerta y los almacena para su posterior análisis.

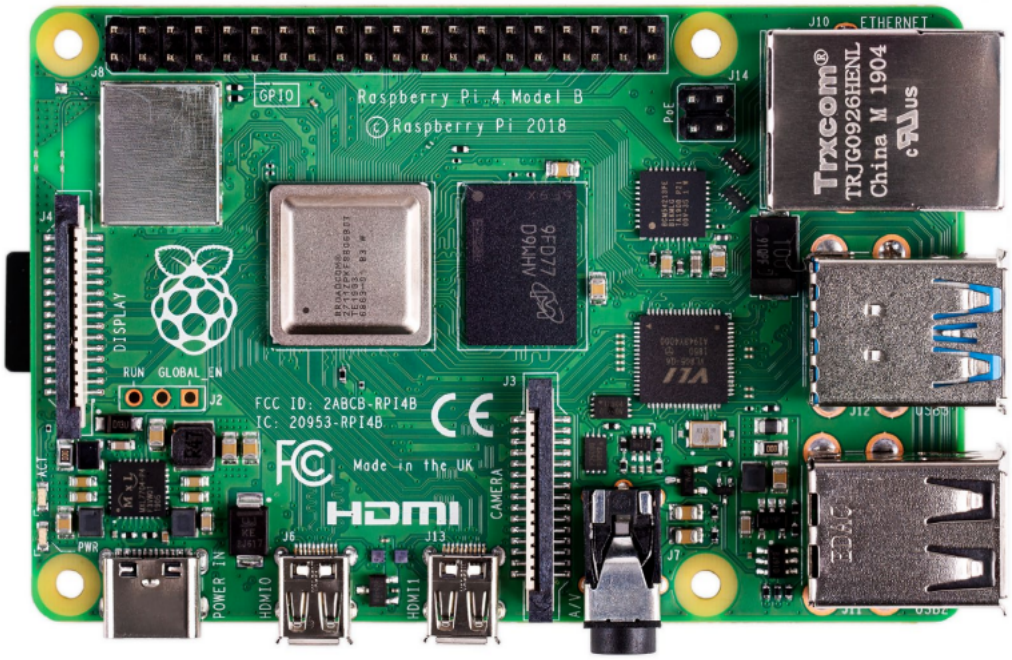


**Figura 3.1.:** Raspberry Pi 4 Model B

### 3.1.2. Cámara USB

- Cámara USB: en este caso se usa una cámara Logitech Brio 4K Stream Edition para conseguir imágenes o videos del interior del frigorífico [24]. Se usa esta cámara porque es la proporcionada por el tutor, ya que es la que se usa en el

laboratorio de la escuela. Es una cámara de alta definición perfectamente válida para tomar imágenes claras de los alimentos.


**Figura 3.2.:** Logitech Brio 4K

### 3.1.3. Sensor de puerta

- Para registrar la apertura y el cierre de la nevera se utiliza el Aqara Door and Window Sensor [5]. Este sensor de puerta es inalámbrico, y para conectarlo con la Raspberry Pi se usa un ConBee II de Zigbee, que me ha proporcionado el tutor. Se ha seleccionado este sensor porque es uno de los compatibles con Conbee II [26].

## 3.2. Arquitectura general

Como ya se ha especificado antes la arquitectura general de este proyecto consta de tres componentes: Raspberry pi, sensor de puerta y una cámara USB. Además, se

**Figura 3.3.:** Sensor de puerta Aqara

**Figura 3.4.:** ConBee II

utiliza un dispositivo Conbee II para facilitar la comunicación entre el sensor de puerta y la Raspberry Pi. Cada uno de estos componentes interactúa entre sí mediante una serie de tecnologías software, que permiten que el sistema recopile, procese y almacene datos de una manera coherente.

La Raspberry Pi, que actúa como el cerebro del sistema, está equipada con un sistema operativo en base Linux y ejecuta un software de orquestación diseñado en Python. Este software tiene dos funciones principales. En primer lugar, recoge y

procesa los datos del sensor de puerta y la cámara USB. En segundo lugar, registra y almacena estos datos para su posterior análisis. La señal del sensor de puerta se procesa a través de una librería Zigbee, en específico con la librería deCONZ, permitiendo su integración con la Raspberry Pi.

La cámara USB está configurada para capturar imágenes a través de OpenCV y fswebam, que son dos librerías Python que permiten interactuar con una cámara. Cuando la Raspberry Pi recibe una señal del sensor de puerta que indica que la puerta se ha abierto, envía un comando a la cámara para que capture imágenes del interior del frigorífico.

Los datos recogidos son almacenados en la Raspberry Pi, que además de las imágenes también registra la fecha y la hora a la que son tomadas, lo que ofrece información importante sobre los hábitos alimenticios de los pacientes. En versiones futuras los datos recogidos también serán procesadas en la Raspberry Pi, en la que se ejecutará una aplicación IA que es capaz de identificar y clasificar los elementos presentes en la imagen. Esta aplicación será la encargada de analizar los datos para entender y predecir el comportamiento del usuario.

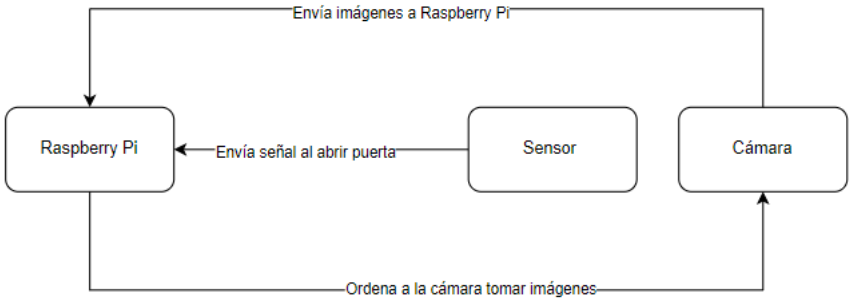


**Figura 3.5.:** Diagrama de la arquitectura

## 3.3. Implementación

Ahora que se han seleccionado los componentes y se ha explicado cómo van a integrarse unos con otros el siguiente paso es llevar a cabo su implementación física en un frigorífico real.

La implementación física del proyecto planteo una serie de desafíos que necesitaban ser resueltos para obtener un sistema eficaz. El reto principal del proyecto fue encontrar un lugar idóneo para instalar la cámara dentro del frigorífico. El objetivo era capturar la mayor parte del contenido del frigorífico con una sola cámara cada vez que el usuario abriera la puerta. Este proceso fue uno de los más largos del proyecto y se consiguió probando diferentes ubicaciones y ángulos. La solución final tiene un compromiso entre la visibilidad del interior del frigorífico y el hecho de tener una cámara montada en un electrodoméstico de uso regular.

**Figura 3.6.:** Ubicación de la cámara

El siguiente aspecto del que ocuparse era la ubicación del sensor de puerta. El sensor de puerta está constituido por dos piezas. Cuando estas dos piezas están en contacto o a una pequeña distancia, el sensor considera que la puerta está cerrada, y cuando se separan el sensor considera que la puerta se ha abierto. La ubicación de este sensor tenía que ser una que satisficiera que cuando la puerta este cerrada las dos piezas quedaran a una distancia muy próxima y que a la vez fuera un lugar

que el usuario no fuera a manipular. El resultado final montado en el frigorífico del laboratorio cumple con los requisitos.

**Figura 3.7.:** Ubicación del sensor de puerta

La cámara usada es una cámara USB, lo que significa que tiene un cable que une la cámara con la Raspberry Pi. Este cable debe estar en un sitio que no interfiera con la vida cotidiana del usuario y además debe salir al exterior del frigorífico. El problema principal de que tenga que salir es que el frigorífico puede perder su estanquidad, y tiene un sitio por el que entra calor, pero como se puede ver en la figura de abajo, el agujero que se genera es muy pequeño por lo que el aumento en el gasto energético sera mínimo. De todas formas este problema se produce por que el cable original de Logitech para la cámara BRIO es muy grueso y se podría solucionar implementando un cable USB plano.

Por último, también se necesita un lugar en el que colocar la Raspberry Pi. En esta ocasión se ha tenido en cuenta que sea una ubicación que no interfiera con la vida

**Figura 3.8.:** Problema con el cable

cotidiana del usuario. Se ha decido colocar en la parte superior, trasera del frigorífico, como se puede observar en la figura 3.9

Para tener una visión general más clara, en la figura 3.10 se adjuntan dos imágenes que reflejan cómo es el sistema.

Para finalizar se muestra en la figura 3.11 una foto del interior del frigorífico tomada con el sistema propuesto.

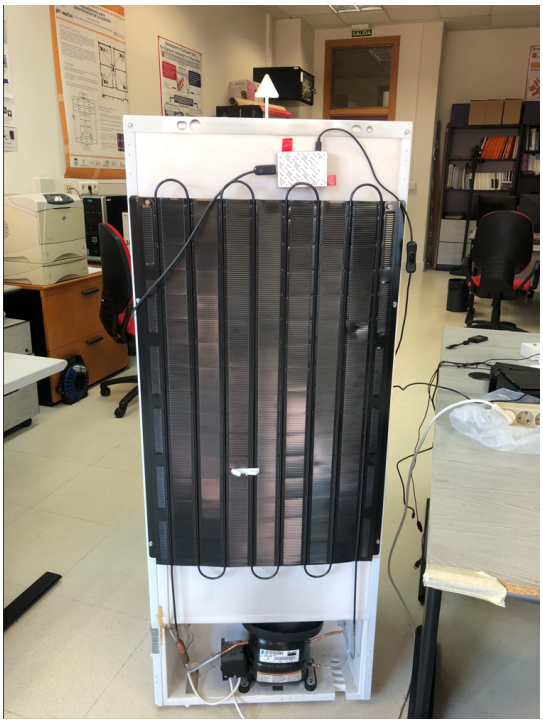

**Figura 3.9.:** Ubicación Raspberry Pi

**(a)** Parte delantera abierta

**(b)** Parte trasera

**Figura 3.10.:** Sistema final


**Figura 3.11.:** Foto del interior del frigorífico

CAPÍTULO 4

# DESARROLLO DEL SOFTWARE


4.1. Herramientas software utilizadas . . . . . 22

4.1.1. Raspberry Pi OS . . . . . 22

4.1.2. Debian . . . . . 22

4.1.3. deCONZ . . . . . 22

4.1.4. ZigBee . . . . . 23

4.1.5. API . . . . . 24

4.1.6. HTTP . . . . . 24

4.1.7. JSON . . . . . 24

4.1.8. Python . . . . . 25

4.1.9. VNC . . . . . 25

4.2. Script de control . . . . . 26

4.2.1. Script modo ráfaga . . . . . 27

4.2.2. Script modo vídeo . . . . . 30

## 4.1. Herramientas software utilizadas

### 4.1.1. Raspberry Pi OS

Raspberry Pi OS, antes conocido como Raspbian, es el sistema operativo oficial para la Raspberry Pi desarrollado por la Fundación Raspberry Pi. Raspberry Pi OS es una distribución de Linux basada en Debian, esto quiere decir que el sistema operativo esta optimizado para el hardware de la Raspberry Pi haciendo un uso eficiente de sus recursos. El sistema operativo ofrece una interfaz gráfica de usuario (GUI) que permite utilizar la Raspberry Pi con teclado y ratón, es decir cómo se interactúa con un ordenador tradicional, pero también ofrece acceso a la linea de comandos para los usuarios más avanzados [33].

### 4.1.2. Debian

Debian es un sistema operativo de código abierto muy versátil basado en Linux y herramientas del sistema GNU [25]. Debian es conocido por sus principios de software libre, es decir que los usuarios pueden usar, modificar y redistribuir el sofware según lo permitan las licencias correspondientes [31]. Se estableció por primera vez en 1993 y se ha ido desarrollando gracias a la comunidad de desarrolladores voluntarios bajo el nombre de Proyecto Debian [30]. Como se ha mencionado anteriormente Debian es la base de Raspberry Pi OS, entre muchas otras distribuciones de Linux.

### 4.1.3. deCONZ

deCONZ es un software desarrollado por dresden elektronik que facilita la interacion con redes y dispositivos ZigBee. Tiene una interfaz gráfica para el control y la visualización de los dispositivos asi como una API REST para la interacción programática [13]. El tutor del proyecto proporciono un dispositivo ConBee II, que es un coordinador de red ZigBee diseñado también por dresden elektronik [12]. Al estar diseñados por el mismo fabricante deCONZ es el software recomendado para trabajar con ConBee II.

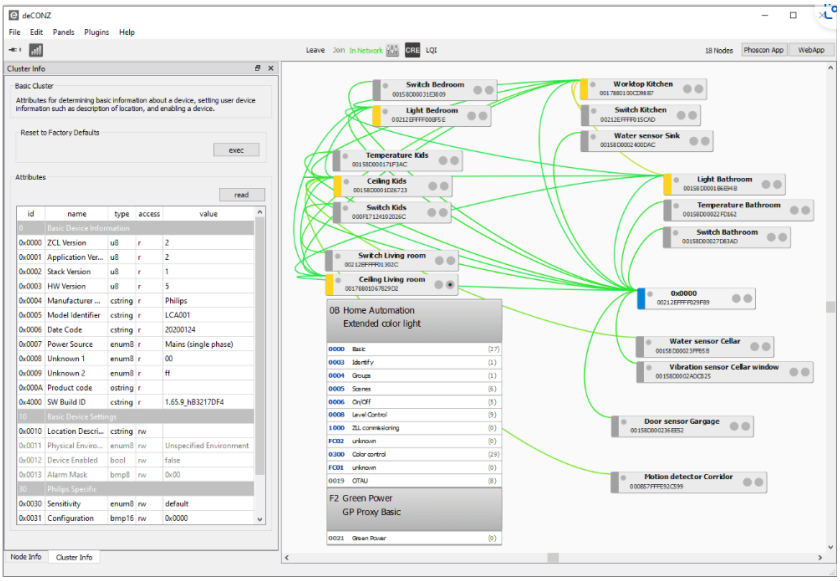


**Figura 4.1.:** Ejemplo de red deCONZ

### 4.1.4. ZigBee

ZigBee es un estándar de comunicación inalámbrica de alta confiabilidad y bajo consumo de energía. Fue diseñado para aplicaciones de control, monitoreo y automatización por lo que es muy útil en proyectos de Internet de las Cosas, donde los dispositivos necesitan comunicarse entre ellos [4].

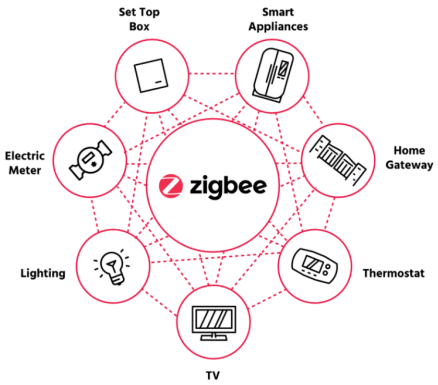


**Figura 4.2.:** ZigBee

Un dispositivo ZigBee es cualquier producto que incorpore la funcionalidad del estándar ZigBee, dentro de estos dispositivos se pueden incluir sensores de luz, temperatura o sensores de puerta, además de dispositivos de seguridad como cerraduras o detectores de humo.

### 4.1.5. API

API (Interfaz de Programación de Aplicaciones) es un conjunto de reglas que una aplicación sigue para comunicarse con otros softwares, por lo que permite que diferentes programas interactúen entre sí, compartiendo datos y funciones sin necesidad de tener los detalles de cómo esta implementado el otro software [27].

deCONZ usa una API REST para conseguir que otros programas interactúen con los dispositivos ZigBee de la red. API REST es un estilo de arquitectura de software que se usa para diseñar aplicaciones en red. REST usa un conjunto de principios basados en HTTP, como los métodos GET, POST, PUT, DELETE, entre otros para realizar operaciones con los datos [14].

Una API REST se usa normalmente para manejar las solicitudes del cliente y responder con datos que suelen estar en formato JSON. En comparación con otras arquitecturas API como puede ser SOAP, REST es más simple, más eficiente y adecuado para comunicaciones en internet.

### 4.1.6. HTTP

HTTP (protocolo de transferencia de hipertexto) es el protocolo en el que se basa la transferencia de datos en la web. Una petición HTTP es una solicitud de un cliente a un servidor para realizar una determinada acción [15]. En el código de este proyecto se usan peticiones GET para solicitar el estado del sensor de puerta, una petición GET es un tipo de petición HTTP que solicitas datos de un recurso especifico.

### 4.1.7. JSON

JSON (JavaScript Object Notation) es un formato estándar de intercambio de datos basado en texto, que se usa para representar datos estructurados de manera legible para los humanos [11]. Se suele usar para transmitir datos entre servidor y una aplicación web y viceversa.

### 4.1.8. Python

Python es un lenguaje de programación de alto nivel muy popular debido a su sintaxis clara y legible. Es muy versátil y potente lo que hace que sea adecuado para muchas tareas como por ejemplo análisis de datos o desarrollo web [17]. Python es un lenguaje interpretado que significa que el código se ejecuta línea por línea facilitando la depuración. Python cuenta con un sistema de tipos dinámico y con gestión automática de memoria, con lo cual no es necesario estar preocupado por la asignación de memoria para las variables y los objetos de datos. Además, Python está orientado a objetos, lo que significa que permite a los programadores definir tipos de datos utilizando clases, crear instancias de esos tipos de datos y llamadas a objetos [16].

Python también cuenta con bibliotecas que son un conjunto de funciones que permiten realizar muchas acciones sin tener que escribir su código. Algunas de estas bibliotecas son proporcionadas por Python como la biblioteca json usada en este proyecto, y otras son creadas por terceros.


**Figura 4.3.:** Python

### 4.1.9. VNC

VNC (Virtual Network Computing) es una tecnología que permite el control remoto de un ordenador a través de internet [28]. En este proyecto se usa VNC para conectarse a la Raspberry Pi desde un ordenador portátil de forma remota, lo que ha permitido mayor flexibilidad a la hora de desarrollar el código ya que no era necesario tener acceso físico a la Raspberry Pi. Se ha optado por este software porque viene ya preinstalado en Raspeberry Pi OS.

**Figura 4.4.:** VNC

## 4.2. Script de control

Como ya se ha mencionado anteriormente el objetivo principal del proyecto es conseguir imágenes del interior del frigorífico, para ello se han implementado dos metodologías. En la primera se toman fotos en modo ráfaga desde que se abre la puerta del frigorífico hasta que se cierra, y en la segunda se graba un video desde que se abre la puerta hasta que se cierra. Se ha tomado esta decisión de tal manera que cuando se implemente el proyecto final, añadiendo el proyecto con IA, se pueda elegir el script más conveniente. A continuación, se explican los dos scripts:

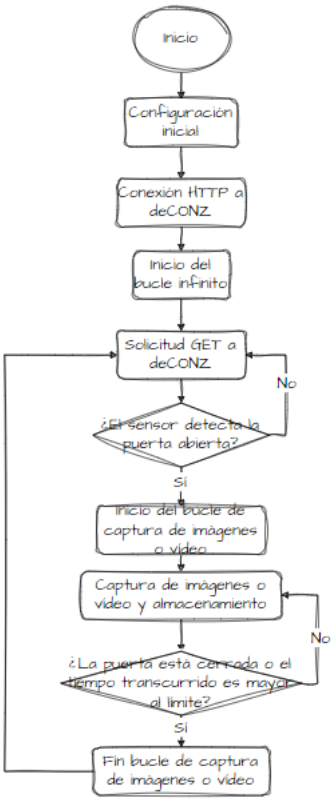


**Figura 4.5.:** Diagrama de flujo software

### 4.2.1. Script modo ráfaga

**Configuración inicial:**

En este apartado se inicializan las variables que se van a ir utilizando en el código, estas son: deconz_ip, api_key, sensor_id, folder_path e image_counter. Las dos primeras son necesarias para poder conectarse con el servidor de deCONZ, el sensor_id es la identificación del sensor utilizado, folder_path es la ruta de la carpeta donde se almacenan las imágenes capturadas y por último image_counter es una variable para llevar la cuenta de todas las fotos tomadas y dar un nombre distinto a cada imagen.

**Creación de carpeta de destino:**

Puede ocurrir que se seleccione una ruta o una carpeta que no exista, estas dos líneas de código verifican si la carpeta seleccionada existe, en caso negativo crea la carpeta. De esta manera no ocurrirá ningún error al intentar guardar las imágenes en una carpeta que no existe.

**Conexión HTTP:**

```
# configurar la conexión HTTP
conn = http.client.HTTPConnection(deconz_ip, port=8080)
```

**Figura 4.6.:** Conexión HTTP

Se establece una conexión HTTP con el servidor de deCONZ, para ello se utiliza la dirección IP del conbee II y el puerto, en este caso se usa el puerto 8080, el puerto por defecto para HTTP es el 80, pero se puede usar el 8080 para hacer pruebas. Esto se utilizará para enviar solicitudes al servidor.

**Monitorización del estado del sensor:**

Este apartado está compuesto por tres variables: last_sensor_state, se utiliza para saber cuál es el estado del sensor, inicialmente se inicializa a “None” que significa desconocido, time_limit, sirve para establecer un tiempo máximo de duración de la ráfaga de fotos, para que en caso de que la puerta se quede abierta no esté haciendo fotos de manera ilimitada, y por último wait_time que establece el tiempo entre fotos de la ráfaga.

**Bucle de comprobación del estado del sensor:**

Se necesita un bucle que se esté ejecutando indefinidamente, ya que se necesita que el sistema se consciente siempre del estado del sensor, hasta que se detenga la aplicación manualmente, por eso se ha optado por un while true, bucle infinito.

**Solicitud del estado del sensor:**

```
url = "/api/" + api_key + "/sensors/" + sensor_id
conn.request("GET", url)
response = conn.getresponse()
```

**Figura 4.7.:** Solicitud de estado

En cada iteración se construye una URL de la API con la clave API y el identificador del sensor. Después se envía una solicitud GET con la URL y la conexión HTTP creada anteriormente para conocer el estado del sensor, por último, se guarda la respuesta del servidor.

**Análisis de la respuesta:**

En esta parte del código se analiza la respuesta HTTP que se recibe después de hacer el GET a deCONZ

```
# analizar la respuesta JSON
if response.status == 200:
    response_json = json.loads(response.read().decode('utf-8'))
    sensor_state = response_json["state"]["open"]
```

**Figura 4.8.:** Análisis de la respuesta

Las respuestas HTTP tienen un código de estado para indicar si la solicitud tuvo éxito o por el contrario no, en caso negativo se proporciona información sobre el problema. El código 200 significa que la respuesta fue exitosa y que se pueden proporcionar los datos pedidos.

Si la respuesta es exitosa se procede a analizar los datos. Cuando la API de deCONZ envía una respuesta, la información se serializa en formato JSON, luego se codifica en bytes para poder transmitirla por la red. Por lo tanto, la respuesta que recibe el programa es en realidad una secuencia de bytes y para trabajar con esta respuesta se necesita convertir esta secuencia a un formato que Python sea capaz de entender. Para conseguir esto primero se usa response.read() para leer los bytes, después se decodifican estos bytes a una cadena de texto usando .decode('utf-8'). UTF-8 hace referencia al sistema que se usó para convertir los caracteres originales en bytes, es un sistema de codificación muy usado que se sirve para representar cualquier carácter en el estándar Unicode.

Los datos ya son una cadena de texto que se encuentra en formato JSON, el siguiente paso es convetir este formato a un objeto Python para poder usarlo en el programa, esto se hace con json.loads().

Por último, lo más importante es saber si el sensor está abierto o cerrado. Este dato se encuentra en el diccionario asociado a la clave state, por lo tanto, para obtener este valor primero se accede al diccionario usando response_json[“state”] y luego se obtiene el valor de la calve open con [“open]. De esta forma se asigna abierto o cerrado a la variable sensor_state. Un diccionario Python no es más que una estructura de datos que almacena pares de elementos clave-valor.

**Comparación del estado actual del sensor con el estado anterior:**

Como ya se ha explicado anteriormente la primera vez que se ejecuta el código el estado del sensor es desconocido por lo que no hay estado anterior con el que compararlo y se establece el estado anterior como el estado actual. En las siguientes iteraciones del bucle, el estado del sensor no va a ser desconocido y por lo tanto se compara el estado actual con el estado anterior y si son diferentes significa que el estado ha cambiado.

**Toma de fotos:**

Si el estado actual indica que la puerta está abierta se procede a tomar fotos. Cada foto se toma con una pausa wait_time, y se detiene la captura después de time_limit o cuando el sensor indica que la puerta se ha cerrado. Las fotos se toman haciendo uso de fswebcam y luego se guarda la foto en la ruta proporcionada anteriormente, cada foto tendrá un nombre que será imagenX.jpg donde X es image_counter. Después de cada foto se verifica el estado del sensor para ver si la puerta se ha cerrado.

```
subprocess.run(["fswebcam", "-r", "1280x720", "-S", "3", "--jpeg", "95", "--rotate", "270", "--save", image_path])
```

**Figura 4.9.:** Captura de imágenes con fswebcam

Fswebcam es un software simple de línea de comandos en sistemas Linux que usa una cámara para capturar imágenes [36]. Se puede usar con la función subprocess.run() que sirve para ejecutar comandos del sistema dentro de un script de Python [18]. En este script se usa fswebcam con una serie de argumentos:

- **-r 1280x720:** Con este argumento se establece la resolución de las imágenes.

- **-S 3:** Este argumento hace que fswebcam omita las tres primeras tramas. Esto se hace para evitar imágenes que estén borrosas o mal expuestas que se pueden producir cuando la cámara se enciende por primera vez.

- **–jpeg 95:** Este argumento sirve para configurar la calidad del archivo de salida. El numero hace referencia al porcentaje de calidad que tendrá. JPEG realiza una compresión con pérdidas para reducir el tamaño del archivo. 95 % proporciona la calidad de imagen necesaria para identificar objetos y mantiene el tamaño en un rango razonable.

- **–rotate 270:** Este argumento se usa para rotar la imagen 270 grados. Esto es necesario debido a como está montada la cámara en el frigorífico.

- **–save image_path:** Este argumento indica donde se van a guardar las imágenes.

**Actualización del estado del sensor:**

Al final de cada iteración se actualiza el estado anterior del sensor al estado actual para así preparar al script para la siguiente iteración.

**Espera antes de la siguiente comprobación del estado del sensor:**

Antes de volver a comprobar el estado del sensor se realiza una pequeña espera, en este caso de 1 segundo, para evitar que el servidor se sobrecargue con demasiadas solicitudes lo que podría llevar a que se bloque el acceso.

**Cierre de conexión HTTP:**

Finalmente se cierra la conexión HTTP. Aunque esta parte del código nunca se llegara a ejecutar debido a que el bucle es infinito es una buena práctica de programación cerrar las conexiones al final del programa.

### 4.2.2. Script modo vídeo

El script de video es muy similar al script de ráfaga de fotos. La única diferencia es que ahora se trabaja con video y la forma de tratar los datos es un poco distinta. Para ello es necesario usar el módulo cv2 de Python que cuenta con funciones para trabajar con video.

**Inicio de la grabación:**

Cuando se detecta que la puerta está abierta se genera un nombre del archivo de video usando la variable video_counter que cuenta cuantos videos se han gravado. El nombre del archivo quedaría: videoX.mp4, donde X es video_counter.

**Configuración de la captura de video:**

```
cap = cv2.VideoCapture(0, cv2.CAP_V4L2)
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(video_path, fourcc, 20.0, (720, 1280))
```

**Figura 4.10.:** Configuración de vídeo

La línea cv2.VideoCapture(0, cv2.CAP_V4L2) abre la cámara de video, 0 es el ID predeterminado de la cámara y cv2.CAP_V4L2 es el backend de captura, el backend se refiere al método que usa cv2 para interactuar con la cámara, en este caso es Video4Linux2 que suele ser el predeterminado para la mayoría de las configuraciones Linux. Las dos líneas de a continuación establecen el ancho a 1280 y el alto a 720, es decir una resolución de 1280x720. La siguiente línea define el codec usado en la grabación de video, el codec no es más que la forma que se comprimen y se descomprimen los archivos de video. Por último, con cv2.Videowriter se crea un objeto de escritura que guardara el video en video_path con el codec definido anteriormente, a 20 fotogramas por segundo y a una resolución de 720x1280. Es importante darse cuenta de que la resolución esta intercambiada con la definida previamente, esto está hecho así debido a que es necesario rotar el video por la posición de la cámara dentro del frigorífico [1].

**Grabación de video:**

A continuación, se comienza a grabar video, para ello se crea un bucle en el que se capturan frames de la cámara, si la lectura del frame es exitosa, se rota el frame 90 grados en sentido antihorario y se escribe en el archivo de video [1].

```
frame = cv2.rotate(frame, cv2.ROTATE_90_COUNTERCLOCKWISE)
```

**Figura 4.11.:** Rotar frame

Este bucle acabara cuando se llegue al tiempo límite de video establecido al comienzo del código o cuando se detecte el cierre de la puerta.

**Liberación de recursos de video:**

Las líneas cap.release(), out.release(), y cv2.destroyAllWindows sirven para cerrar todas las ventanas de visualización de videos y liberar los recursos de video. Este paso

es muy importante ya que de esta manera se evitan fugas de memoria y el hardware de la cámara se libera correctamente [1].

Estos dos scripts de control logran recoger, capturar y almacenar imágenes del interior del frigorífico. De tal manera que se consiguen unas bases sólidas para luego poder implementar la parte de inteligencia artificial y realizar el procesado de las imágenes conseguidas.

CAPÍTULO 5

# PRUEBAS Y APRENDIZAJE


5.1. Pruebas . . . . . 34
5.1.1. Pruebas usando Raspberry Pi . . . . . 34
5.1.2. Pruebas con la cámara: . . . . . 34
5.1.3. Pruebas con la API: . . . . . 35
5.1.4. Primer programa con el interruptor: . . . . . 35
5.1.5. Ubicación de la cámara: . . . . . 36
5.1.6. Sustitución de interruptor por sensor de puerta: . . . . . 36
5.1.7. Código para sensor de puerta: . . . . . 36
5.1.8. Pruebas del sistema completo: . . . . . 37
5.1.9. Pruebas de largo funcionamiento: . . . . . 37
5.1.10. Pruebas de frío: . . . . . 37
5.1.11. Pruebas de eficiencia energética: . . . . . 38


Normalmente la sección de pruebas es una de las más importantes en un proyecto de ingeniería. Este caso no es una excepción, pero el desarrollo de las pruebas ha sido un poco diferente de lo habitual debido a mi poca experiencia con las tecnologías utilizadas. Por eso según avanzaba el proyecto se hacían pruebas pequeñas que también han ayudado mucho a adquirir conocimientos y mejorar usando este tipo de tecnologías.

## 5.1. Pruebas

### 5.1.1. Pruebas usando Raspberry Pi

Al no tener nada de experiencia usando una Raspberry Pi todo era nuevo y el primer desafío era descargar el sistema operativo para luego poder familiarizarse con la Raspberry. Para ello se usó la documentación oficial de Raspberry Pi y otros recursos en línea.

### 5.1.2. Pruebas con la cámara:

El siguiente paso era conectar la cámara y probar el comando *fswebcam*. Esta vez se usó este programa desde la propia terminal de la Raspberry Pi, es decir no estaba integrada en ningún código de Python. Se probó a usar el comando *fswebcam* e ir cambiando los diferentes argumentos del comando para luego usarlos en el proyecto.


**Figura 5.1.:** Primera foto usando una Raspberry Pi

### 5.1.3. Pruebas con la API:

El siguiente paso era poder saber el estado del sensor en la Raspberry Pi. Esto se lleva a cabo usando la API de deCONZ. Este paso ha sido uno de los más complicado ya que no tenía experiencia en el uso de APIs. Antes de empezar con el sensor se usó un interruptor inteligente para familiarizarse con este proceso. Al principio este paso se atascó un poco, pero con la documentación de deCONZ y aprendiendo a hacer las peticiones GET y el uso de diccionarios en Python se consiguió realizar un programa que supiera el estado del sensor.

### 5.1.4. Primer programa con el interruptor:

El primer programa con el interruptor fue un programa que detectaba si el interruptor estaba en posición ON y si lo estaba hacia una foto haciendo uso de fswebcam. Poco a poco se fue mejorando este programa haciendo un bucle infinito y consiguiendo que se hiciera una foto cada vez que el interruptor estuviera en la posición ON.

**Figura 5.2.:** Interruptor inteligente

### 5.1.5. Ubicación de la cámara:

Ahora que ya tenía un programa que hacía fotos cada vez que yo accionaba un interruptor era el momento de pasar al reto principal del proyecto, donde ubicar la cámara dentro del frigorífico. Para lograrlo se llevó a cabo un proceso de prueba y error en que se iban intentando diferentes ubicaciones de la cámara, diferentes ángulos, por ejemplo, desde arriba del frigorífico con la cámara apuntando hacia abajo, hasta que se llegó a la solución final. Bajo mi punto de vista se ha llegado a una solución que tiene un compromiso entre capturar la mayor parte posible del interior del frigorífico y no estorbar mucho en la vida cotidiana de un usuario normal. Aunque la solución esté personalizada para el frigorífico disponible en la escuela, ya se ha determinado la zona óptima para ubicar la cámara. Por tanto, con unas ligeras modificaciones en la posición de la cámara, ya sea moviéndola un poco hacia arriba o hacia un lado, la solución podría adaptarse a cualquier frigorífico.

### 5.1.6. Sustitución de interruptor por sensor de puerta:

Una vez ya conocida la ubicación de la cámara era momento de sustituir el interruptor por un sensor de puerta. En esta parte también se probaron diferentes ubicaciones, todas tenían algún error como por ejemplo que se detectaba la apertura del congelador como apertura del frigorífico. Hasta que se consiguió esa ubicación.

### 5.1.7. Código para sensor de puerta:

Ahora era momento de hacer un código que se ajustase con los requisitos del sistema. La tarea era hacer un código que detectase la apertura del sensor y estuviera haciendo fotos o grabando un vídeo hasta que se cerrara la puerta o hasta que pasase un tiempo de seguridad. En esta parte tuvo mucha dificultad sobre todo a la hora de que se detectara bien cuando se cerraba la puerta antes de que se acabase el tiempo de seguridad.

### 5.1.8. Pruebas del sistema completo:

Una vez todo el sistema montado se hicieron pruebas para ver que todo funcionaba según lo previsto y se podía tener la Raspberry sin necesidad de estar conectada a un monitor, a un teclado y a un ratón por que se podían visualizar los resultados correctamente en un ordenador portátil haciendo uso de VNC.

**(a)** Inicio de la prueba **(b)** Recogida de alimento **(c)** Final de la prueba

**Figura 5.3.:** Secuencia de pruebas

### 5.1.9. Pruebas de largo funcionamiento:

Esta prueba consistió en probar el sistema un viernes ver que funcionaba, dejarlo encendido todo el finde semana y volverlo a probar un lunes. La prueba fue satisfactoria.

### 5.1.10. Pruebas de frío:

Estas pruebas son unas de las más importantes del sistema, pero no se han podido llevar a cabo porque el frigorífico no enfriaba bien. La cámara usada en el proyecto tiene muchas ventajas, una alta resolución pudiendo llegar hasta 4k, lo que proporciona imágenes de alta calidad, rango dinámico alto, esto quiere decir que puede

manejar una gama de iluminaciones muy amplia, un campo de visión de 90 grados, autoenfoque y la tecnología RightLight 3 de Logitech que ajusta automáticamente la exposición y balance de blancos para obtener las mejores imágenes posibles en diferentes condiciones de iluminación, pero se quería comprobar si se mantenían estas ventajas cuando se exponía a la cámara a temperaturas de 4 o 5 grados durante un tiempo prolongado. En la hoja de especificaciones oficial de Logitech no proporciona datos para la temperatura, por lo que he hecho una consulta al servicio de atención al cliente oficial [23]. En lo que recibo una respuesta he buscado una solución al problema. La solución podría ser utilizar cámaras de seguridad aptas para el uso en exteriores que son capaces de operar en condiciones de frío extremo. Las opciones son:

- Reolink Argus 2: Es una cámara de seguridad resistente a la intemperie que puede funcionar en el rango de temperaturas de -10 a 55 grados, además cuenta con un campo de visión de 130º, una resolución de 1080p Full HD y un tamaño contenido [29].

- Arlo Pro-3: Esta cámara puede trabajar en el rango de -20 a 45 grados. El campo de visión de esta cámara es de 160º, cuanta con una resolución de hasta 2K con HDR, pilas recargables con una duración de batería de hasta 6 meses y un tamaño contenido [6].

### 5.1.11. Pruebas de eficiencia energética:

Esta prueba tampoco la he podido llevar a cabo. En esta prueba me gustaría comprobar cuanto frío se pierde debido a que el cable USB que conecta la cámara con la Rapberry Pi sale del interior del frigorífico y provoca un pequeño agujero en la junta del frigorífico. Esto se podría comprobar con un Smart Plug que sea capaz de monitorizar el consumo del frigorífico con el sistema montado y sin el, y de esta manera saber si es ineficiente o no. De todas maneras como ya se menciono en el capítulo 3, el agujero es muy pequeño, por lo que la cantidad de calor que puede entrar al interior del frigorífico es mínima, y además, se puede solucionar fácilmente cambiando el cable USB, por uno plano.

Con todas estas pruebas se puede observar de forma resumida como ha ido siendo mi proceso de aprendizaje y familiarización con las diferentes tecnologías involucradas

y como el proyecto ha ido avanzando desde cero hasta la solución final pasando por varias etapas intermedias y realizando diferentes tipos de pruebas dependiendo de la fase en la que se encontraba el desarrollo.

CAPÍTULO 6

# CONCLUSIONES

Este Trabajo de Fin de Grado representa el desarrollo e implementación de un avanzado sistema de monitorización de frigoríficos, haciendo uso de tecnologías de Internet de las Cosas (IoT). En concreto, este proyecto integra una Raspberry Pi, una cámara USB, y un sensor de puerta, utilizando una variedad de softwares para lograr la funcionalidad deseada.

La solución propuesta ha demostrado ser efectiva y cumple con el objetivo principal propuesto. Este sistema es capaz de capturar imágenes o videos del interior de un frigorífico cada vez que se abre la puerta. Más aún, este trabajo sienta las bases para futuras ampliaciones, pudiendo incorporar tecnologías adicionales como la Inteligencia Artificial para enriquecer aún más la solución propuesta.

A lo largo del desarrollo de este proyecto, me he enfrentado a una serie de retos y desafíos, sobre todo relacionados con el manejo de diversas tecnologías y su implementación en un entorno real de frigorífico. Estos desafíos han reforzado la importancia de la habilidad para resolver problemas en ingeniería, y a la vez, han brindado la oportunidad de adquirir un conocimiento más profundo en una variedad de tecnologías. Me he familiarizado con el funcionamiento de una Raspberry Pi, he mejorado mi capacidad en el manejo de APIs para integrar los sensores con el sistema, y he ampliado mis habilidades en el lenguaje de programación Python.

Finalmente, este proyecto ha demostrado la versatilidad de las tecnologías IoT para ofrecer soluciones a problemas cotidianos. Adicionalmente, ha reforzado la importancia de promover una alimentación equilibrada, un aspecto crucial no sólo para personas con problemas de salud mental, sino para la población en general.

CAPÍTULO 7

# TRABAJO FUTURO

7.1. Integración con Inteligencia Artificial . . . . . 43
7.2. Desarrollo de una interfaz de usuario . . . . . 44
7.3. Optimización de la eficiencia energética . . . . . 44
7.4. Seguridad de los datos . . . . . 44

En este punto me gustaría resaltar aspectos que se podrían mejorar o ampliar para ofrecer un sistema más completo:

## 7.1. Integración con Inteligencia Artificial

Este es el aspecto más importante de todos, ya que este proyecto está pensado para que se pueda implementar en él la IA. Este proyecto consigue tomar imágenes del interior del frigorífico, pero es necesario analizarlas y procesarlas para observar la evolución en los hábitos alimenticios de una persona. Además, la IA ofrece muchas funciones con lo que se podría dotar al sistema de muchas aplicaciones.

## 7.2. Desarrollo de una interfaz de usuario

El sistema actual no tiene ninguna interfaz de usuario. Se podría crear una interfaz de usuario fácil de usar que ayude a los usuarios a interactuar con el sistema de manera más directa. Como ejemplo esta interfaz podría mostrar una lista de lo que hay en el frigorífico y de lo que se ha consumido.

## 7.3. Optimización de la eficiencia energética

Cada vez es más importante la sostenibilidad energética, por lo que habría que preocuparse de desarrollar un sistema que sea eficiente. Esta parte se puede mejorar revisando el código de programación, pero también cambiando como está integrado el hardware. Una solución a esto en el futuro podría ser un sistema con una mayor integración en el frigorífico de tal manera que el cable que alimenta la cámara USB este dentro de la propia instalación eléctrica del frigorífico, siendo el siguiente paso la integración de la cámara completa como una parte más del propio frigorífico.

## 7.4. Seguridad de los datos

Al estar trabajando con datos personales y tan sensibles como son los de la salud es muy importante asegurarse que se toman medidas para que los datos de las personas estén seguros y se cumplan las regulaciones pertinentes como el Reglamento General de Protección de Datos. Además, es muy importante saber cómo se almacenan y procesan los datos, que solo personas autorizadas tienen acceso a ellos y que se pueden eliminar de forma segura cuando ya no sean necesarios.

APÉNDICE A

# ANEXOS

En este apéndice se proporcionan los enlaces a los dos códigos Python desarrollados en este proyecto.

## A.1. Código modo ráfaga

https://github.com/nacho282000/Rafaga/tree/main

## A.2. Código modo vídeo

https://github.com/nacho282000/Video/tree/main

# BIBLIOGRAFÍA


[1] *Opencv (cv2) official documentation*. https://docs.opencv.org/master/, Accessed: June 23, 2023.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari y M. Ayyash: *Internet of things: A survey on enabling technologies, protocols, and applications*. IEEE Communications Surveys Tutorials, 17(4):2347–2376, 2015.

[3] C. Alcántara, L. M. Klesges, K. Resnicow, A. Stone y K. W. Davidson: *Enhancing the evidence for behavioral counseling*. Am J Prev Med, 49:S184–S193, 2015.

[4] Zigbee Alliance: *Zigbee*, n.d. https://zigbeealliance.org/solution/zigbee/, Retrieved from.

[5] Aqara: *Door and window sensor*, 2021. https://www.aqara.com/us/door_and_window_sensor.html, Retrieved from.

[6] Arlo: *Arlo pro 4 security camera official product page*. https://www.arlo.com/en_gb/cameras/arlo-pro-4, Accessed: July 7, 2023.

[7] K. Ashton: *That 'internet of things' thing*. RFiD Journal, 22(7):97–114, 2009.

[8] R. Ayesa Arriola, O. Martínez García, J. Mayoral Van Son, M. Rius Teixido y P. Suárez Pinilla: *Guía de hábitos saludables para personas con trastorno mental grave*. Informe técnico, Gobierno de Cantabria. Consejería de Sanidad y Servicios Sociales. Dirección General de Ordenación y Atención Sanitaria, 2014.

[9] M. D. Barros Albarrán, C. Esteban Ortega, M. I. Rivera Hidalgo y J. M. Caro Villanueva: *La salud física de los pacientes con trastorno mental grave: importancia de la prevención y promoción de la salud desde las unidades de salud mental comunitarias*. Hygia de Enfermería, (92):21–28, 2016.

[10] J. Carmona Calvo, P. García-Cubillana de la Cruz, A. Millán Carrasco, E. Huizing, G. Fernández Regidor, M. Rojo Villalba *y cols.*: *Iii plan integral de salud mental de andalucía 2016-2020*. Informe técnico, Junta de Andalucía. Consejería de Salud, 2016.

[11] D. Crockford: *The application/json media type for javascript object notation (json)*. Informe técnico, IETF, 2006. `https://tools.ietf.org/html/rfc4627`.

[12] dresden elektronik: *Conbee ii*, n.d. `https://phoscon.de/en/conbee2`, Retrieved from.

[13] dresden elektronik: *deconz*, n.d. `https://phoscon.de/en/conbee2/software#deconz`, Retrieved from: https://phoscon.de/en/conbee2/softwaredeconz.

[14] R. T. Fielding: *Architectural Styles and the Design of Network-based Software Architectures*. Tesis de Doctorado, University of California, Irvine, 2000.

[15] R. T. Fielding y J. Reschke: *Hypertext transfer protocol (http/1.1): Message syntax and routing*. Informe técnico, IETF, 2014. `https://tools.ietf.org/html/rfc7230`.

[16] Python Software Foundation: *9. classes. python documentation*. `https://docs.python.org/3/tutorial/classes.html`, Accessed: June 17, 2023.

[17] Python Software Foundation: *What is python? executive summary*. `https://www.python.org/doc/essays/blurb/`, Accessed: June 17, 2023.

[18] Python Software Foundation: *subprocess — subprocess management — python 3.10.0 documentation*, 2023. `https://docs.python.org/3/library/subprocess.html`, Accessed: July 6, 2023.

[19] Raspberry Pi Foundation: *Raspberry pi 4 model b specifications*, 2020. `https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/`, Retrieved from.

[20] Raspberry Pi Foundation: *What is a raspberry pi?*, 2021. `https://www.raspberrypi.org/help/what-is-a-raspberry-pi/`, Retrieved from.

[21] L. Jiménez-Muñoz, L. Gutiérrez-Rojas, A. Porras-Segovia, P. Courtet y E. Baca-García: *Mobile applications for the management of chronic physical conditions: a systematic review*. Intern Med J, 52:21–29, 2022. `http://dx.doi.org/10.1111/imj.15081`.

[22] Y. LeCun, Y. Bengio y G. Hinton: *Deep learning*. Nature, 521(7553):436–444, 2015.

[23] Logitech: *Brio 4k stream edition specifications sheet*. `https://www.logitech.com/content/dam/logitech/vc/en_ch/pdf/Brio-Datasheet.pdf`, Accessed: Juñy 7, 2023.

[24] Logitech: *Brio 4k stream edition*, 2021. `https://www.logitech.com/en-us/products/webcams/brio-4k-hdr-webcam.960-001178.html`, Retrieved from.

[25] Robert Love: *Linux Kernel Development*. Addison-Wesley, 2005.

[26] Phoscon: *Conbee ii*, 2021. `https://phoscon.de/en/conbee2`, Retrieved from.

[27] T. A. Puntambekar: *Web Services and APIs*, páginas 19–40. Springer, 2020.

[28] RealVNC: *How does vnc work?* `https://www.realvnc.com/en/b/`, Accessed: June 18, 2023.

[29] Reolink: *Argus 2 security camera official product page*. `https://reolink.com/product/argus-2/#overview`, Accessed: July 7, 2023.

[30] Peter H. Salus: *A Quarter Century of UNIX*. Addison-Wesley, 1994.

[31] Richard M. Stallman: *Free Software, Free Society: Selected Essays of Richard M. Stallman*. GNU Press, 2002.

[32] Statista: *Internet of things (iot) connected devices installed base worldwide from 2015 to 2025*, 2021. `https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/`, Retrieved from.

[33] Eben Upton y Gareth Halfacree: *Raspberry Pi User Guide*. Wiley, 2020.

[34] E. Utrera Caballero, G. Ruiz Guerrero, M. J. Aguilera Moreno, R. Guerrero Sánchez y J. A. Ariza Cot: *Efectividad de una intervención de enfermería en educación para la salud para la promoción de hábitos de vida saludables dentro del programa de atención a pacientes con tmg y apoyo a dispositivo residencial comunitario (casa-hogar)*. Rev. Paraninfo Digital, (25), 2016.

[35] O. Vermesan y P. Friess (editores): *Internet of things: converging technologies for smart environments and integrated ecosystems*. River Publishers, 2013.

[36] Debian Wiki: *fswebcam*, 2023. `https://wiki.debian.org/fswebcam`, Accessed: July 6, 2023.