UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

PROYECTO PARA LA MONITORIZACIÓN DE LA ACTIVIDAD DE UNA PERSONA CON DEMENCIA SENIL

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Zapata Romero, Francisco José

Tutor/a: Romero Martínez, José Oscar

CURSO ACADÉMICO: 2022/2023

Resumen

Este proyecto presenta el diseño e implementación de un sistema de monitorización de bajo costo para pacientes con demencia senil utilizando una red compuesta por sensores, actuadores y microcontroladores ESP32, e interconectada con el protocolo de comunicación MQTT vía Wi-Fi. La demencia senil es una condición que afecta principalmente a las personas mayores, y una de las principales dificultades que conlleva es la necesidad de una observación constante para garantizar la seguridad y bienestar del paciente.

Los sensores se emplean para recoger datos de diversas fuentes, como el movimiento del paciente, sonidos a la hora de comunicarse, la temperatura de la habitación y la humedad del colchón a la hora de posibles incontinencias. Estos datos son luego procesados y analizados utilizando el microcontrolador ESP32, que es capaz de realizar tareas complejas de procesamiento de datos a un costo relativamente bajo.

Se pretende que las personas a cargo del paciente reciban los avisos de las constantes medidas a través de una interfaz controlada por un microcontrolador. Este sistema tiene el propósito de mejorar la calidad del cuidado proporcionado a estos pacientes y reducir la carga para los cuidadores.

Resum

Aquest projecte presenta el disseny i implementació d'un sistema de monitoratge de baix cost per a pacients amb demència senil utilitzant una xarxa composta per sensors, actuadors i microcontroladors ESP32, i interconnectada amb el protocol de comunicació MQTT via Wi-Fi. La demència senil és una condició que afecta principalment les persones majors, i una de les principals dificultats que comporta és la necessitat d'una observació constant per a garantir la seguretat i benestar del pacient.

Els sensors s'empren per a recollir dades de diverses fonts, com el moviment del pacient, sons a l'hora de comunicar-se, la temperatura de l'habitació i la humitat del matalàs a l'hora de possibles incontinències. Aquestes dades són després processats i analitzats utilitzant el microcontrolador ESP32, que és capaç de fer tasques complexes de processament de dades a un cost relativament baix.

Es pretén que les persones a càrrec del pacient reben els avisos de les constants mesurades a través d'una interfície controlada per un microcontrolador. Aquest sistema té el propòsit de millorar la qualitat de la cura proporcionada a aquests pacients i reduir la càrrega per als cuidadors.

Abstract

This project presents the design and implementation of a low-cost monitoring system for patients with senile dementia using a network composed of sensors, actuators and ESP32 microcontrollers, and interconnected with the MQTT communication protocol via Wi-Fi. Senile dementia is a condition that mainly affects older people, and one of the main difficulties it brings is the need for constant observation to ensure the patient's safety and well-being.

Sensors are used to collect data from various sources, such as the patient's movement, sounds when communicating, the temperature of the room and the humidity of the mattress for possible incontinence. This data is then processed and analysed using the ESP32 microcontroller, which is capable of performing complex data processing tasks at relatively low cost.

It is intended that caregivers will receive alerts of measured constants via a microcontroller-controlled interface. This system is intended to improve the quality of care provided to these patients and reduce the burden on caregivers.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1. Introducción

Los sistemas IoT son una red de dispositivos interconectados a través de Internet y entre ellos recopilan, transmiten y procesan datos a tiempo real, lo que los hace en la actualidad ser una forma económica de automatizar procesos, por ejemplo, dentro de una red doméstica o en la red de una empresa.

Basándose en el concepto de automatizar procesos y con las grandes posibilidades que proporciona la tecnología con un bajo coste, como obtener datos mediante sensores y procesarlos para conseguir un resultado, a la hora de solucionar un problema.

En este proyecto se pretende desarrollar un sistema que ayude a una persona que ha sido diagnosticada con demencia senil y está en un proceso avanzado de la enfermedad. Consiguiéndose monitorear ciertos parámetros que serán explicados más adelante. Se pretende conseguir que sus familiares o las personas que se encargan de cuidarlo estén informados acerca de la situación de su actividad en cada momento.

1.1 Motivación

La idea de este proyecto surge de la búsqueda para desarrollar algo que fuese una aplicación funcional e interesante, todo esto en colaboración con el tutor, el cual oriento sobre los distintos posibles ámbitos que había para desarrollar un proyecto dentro de la especialidad de telemática.

El primer paso fue decidir en qué ámbito se quería realizar el proyecto, la opción elegida fue desarrollar una idea dentro del IoT [11], ya que englobaba distintas características que motivaban para desarrollar un proyecto, como son:

- Poder realizar un proyecto con una parte de electrónica, poder montar y diseñar circuitos, para construir un dispositivo.

- Trabajar con comunicaciones entre dispositivos a través de internet.

- La motivación de construir un proyecto fácilmente aplicable en el día a día.

Una vez esto claro, se plantearon distintas opciones como podía ser realizar la domótica de un domicilio para ahorrar energía o esto mismo en un comercio, pero finalmente tras observar que esa parte ya estaba bastante explotada en el sector comercial y por circunstancias alrededor de la realización del proyecto a la hora de hablar con familiares, se decidió implementar un sistema IoT para monitorizar una persona de avanzada edad con demencia senil, ya que las opciones comerciales son limitadas.

1.2 Objetivos

El objetivo principal de este proyecto es conseguir crear funcionalidades para monitorizar la actividad de una persona que está la mayor parte del tiempo en una cama, de forma autónoma dentro de una vivienda de forma sencilla, para que las personas encargadas de cuidarlo sean capaces de estar en todo momento sabiendo la situación de la persona. Todo esto estructurado con sensores de montaje sencillo y económicos.

Se plantea montar una red de dispositivos con dos extremos, en una parte para que los sensores capten datos de la actividad de la persona, en la otra parte actuadores para notificar de posibles cambios en la actividad de la persona y poder controlar de forma automática parámetros. En ambos extremos se ha conseguido realizar la comunicación utilizando dispositivos ESP32, que son microcontroladores de bajo coste. Se ha decidido utilizar estos dispositivos gracias a que cuentan con conectividad inalámbrica Wi-Fi, ya que esto ahorra tener que incorporar un módulo auxiliar para la conectividad.

En este caso, ya que el objetivo es desarrollar un prototipo, la idea es utilizar un módulo de desarrollo que contiene un microcontrolador ESP32, junto con una serie de componentes adicionales para facilitar su programación y conexión a otros dispositivos. Tienen pines de entrada y salida (GPIO) que permiten la conexión a otros elementos, así como interfaces para la programación y depuración del microcontrolador.

Figura 1. Módulo de desarrollo ESP32 NodeMCU

La interconexión de estos microcontroladores se ha hecho mediante la tecnología de comunicación MQTT, el cual es un protocolo de comunicación M2M de tipo cola de mensajes y se basa en el modelo de publicador/subscriptor que permite la comunicación asíncrona entre el editor y los clientes suscriptores. También destaca por ser ligero y sencillo, por lo cual se puede usar fácilmente con microcontroladores con capacidad de procesamiento limitado.

1.3 Estructura de la memoria

La estructura de esta memoria está compuesta por distintos capítulos. El primer capítulo, se utiliza para hacer una breve introducción sobre el proyecto en distintos puntos, entre los que esta esté mismo apartado.

En el segundo capítulo, se contextualiza teóricamente toda la tecnología que se ha utilizado en este proyecto y se introduce porque se ha escogido esta frente a otra. Se explican conceptos teóricos sobre internet de las cosas, el protocolo de comunicación MQTT y sobre los dispositivos hardware utilizados.

En el tercer y cuarto capítulo, se explica cómo se hacen las configuraciones básicas necesarias a nivel software para comenzar a desarrollar las comunicaciones y se desarrollan las funcionalidades que se quieren desarrollar para controlar al paciente, respectivamente.

En el quinto capítulo, se explica el código con el que se han programado los microcontroladores ESP32, este código se explica a nivel pseudocódigo, detallando cada función que se realiza en el lenguaje de programación Arduino. El explicar el código de esta forma es para detallar mejor lo que hace este directamente y proteger la propiedad intelectual.

En el sexto capítulo, se describe el montaje de los tres circuitos necesarios para que el sistema IoT funcione, cada uno de estos tiene como elemento central un microcontrolador ESP32. Entre los conceptos que se describen, están principalmente las conexiones entre los dispositivos y su explicación.

En el séptimo capítulo, se detalla el funcionamiento general del sistema, donde se pretende hacer un resumen de cómo se montaría en una vivienda, una descripción de su funcionamiento y una explicación de como visualizar los datos en la interfaz de usuario.

En el octavo capítulo, se muestra un presupuesto sencillo de todos los dispositivos que se necesitan adquirir para formar el sistema.

Por último, en el noveno y décimo capítulo, se dan unas conclusiones de lo que ha sido desarrollar este proyecto y la bibliografía para justificar de donde se han sacado los conocimientos teóricos, respectivamente.

Capítulo 2. Contexto de la tecnología utilizada

2.1 Internet de las cosas.

2.1.1 Introducción

El Internet de las cosas [1] se podría definir como “Conexión de dispositivos físicos y objetos cotidianos a Internet y entre sí, permitiéndoles recopilar y compartir datos”. Estos dispositivos pueden incluir desde electrodomésticos inteligentes, como refrigeradores y termostatos, hasta vehículos conectados y equipos industriales.

La conectividad de estos objetos permite recopilar y transmitir datos en tiempo real, lo que permite monitorear y controlar diversos aspectos del entorno, automatizar procesos y mejorar la eficiencia en una variedad de industrias, como la salud, la agricultura, la energía y el transporte. Para que los dispositivos se comuniquen entre sí, utilizan una variedad de tecnologías de conectividad, como Wi-Fi, Bluetooth, NFC, RFID y redes móviles.

El uso de IoT ha llevado a la creación de ciudades inteligentes, en las que los datos recopilados por los dispositivos IoT se utilizan para mejorar la gestión de recursos públicos y mejorar la calidad de vida de los ciudadanos.

2.1.2 Contexto histórico del IoT

El concepto de Internet de las cosas (IoT) se originó en la década de 1990 con el objetivo de conectar dispositivos electrónicos y hacer que intercambiaran información a través de Internet. Sin embargo, el término "Internet de las cosas" fue creado por Kevin Ashton en 1999, quien fue uno de los fundadores del Centro de Auto-ID del MIT.

En las primeras etapas de IoT, los dispositivos conectados eran principalmente sensores y medidores utilizados para recopilar datos en entornos industriales. A medida que avanzó la tecnología, se ido haciendo posible cada vez conectar dispositivos más diversos y de una forma más económica.

El rápido crecimiento del IoT se ha visto impulsado por el aumento de la conectividad inalámbrica, cada vez se consigue que los dispositivos sean de menor tamaño, el coste de los componentes es más asequible y la pertenencia casi imprescindible de tener un teléfono móvil por la población. Según algunos estudios, se espera que para 2025 haya más de 75 mil millones de dispositivos IoT en todo el mundo.

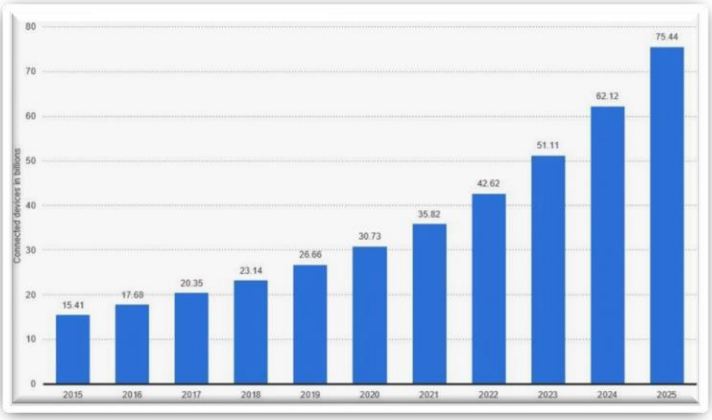

Figura 2. Previsión de dispositivos IoT hasta 2025 [2]

2.1.3 Protocolos de comunicación y tecnologías de transmisión.

Entre los diversos protocolos de comunicación y tecnologías de transmisión en IoT, en la actualidad destacan los enumerados en las siguientes tablas:

Protocolos de comunicación	Características
MQTT	Protocolo ligero de mensajería diseñado para transmitir datos en redes de baja potencia y ancho de banda. Permite la comunicación bidireccional entre dispositivos IoT y se utiliza en aplicaciones de IoT que requieren alta escalabilidad y eficiencia energética.
CoAP	Protocolo de aplicaciones web de bajo costo para sistemas IoT, diseñado para ser utilizado en dispositivos con recursos limitados. Utiliza UDP como protocolo de transporte y se enfoca en la eficiencia energética y la interconexión entre dispositivos.
HTTP	Protocolo utilizado para la comunicación en internet, también se utiliza en las IoT para la transmisión de datos a través de la web. Es muy utilizado en aplicaciones de IoT que requieren una gran cantidad de datos o que necesitan integrarse con otros sistemas web.
AMQP	Protocolo de mensajería avanzado que permite la comunicación entre sistemas IoT de manera eficiente y segura. Se enfoca en la interconexión entre dispositivos y es muy utilizado en aplicaciones de IoT empresariales y de alta seguridad.

Tabla 1. Protocolos de comunicación en IoT [3]

Tecnologías de transmisión	Características
Wi-Fi	Tecnología de transmisión inalámbrica de alta velocidad que se utiliza en dispositivos IoT con una fuente de energía constante. Ofrece alta velocidad y buena cobertura, pero consume mucha energía, lo que limita su uso en dispositivos alimentados por batería. Es ampliamente utilizado en entornos domésticos y empresariales.
Bluetooth	Tecnología de transmisión inalámbrica de corto alcance que se utiliza en dispositivos IoT para la comunicación entre dispositivos cercanos. Entre sus características destaca su bajo consumo de energía, alcance limitado, capacidad limitada para manejar grandes cantidades de datos, interferencias de otras señales inalámbricas.
ZigBee	Tecnología de transmisión inalámbrica de baja potencia y alcance medio utilizada en dispositivos IoT para la comunicación entre dispositivos de red de área personal.
LoRaWAN	Tecnología de transmisión de baja potencia y largo alcance que utiliza una red de sensores para transmitir datos a través de grandes áreas. Sus ventajas son su largo alcance, bajo consumo de energía, adecuado para aplicaciones de IoT en áreas remotas.

Tabla 2. Tecnologías de transmisión en IoT.

2.2 Protocolo MQTT

2.2.1 Introducción

El protocolo MQTT [4],[5],[6] es un protocolo de comunicación ligero basado en el patrón de publicación-suscripción, diseñado específicamente para sistemas de Internet de las cosas (IoT) y aplicaciones de mensajería de baja potencia y latencia. Desarrollado por IBM en 1999, MQTT es ampliamente utilizado en aplicaciones donde la eficiencia de la red y la entrega de mensajes en tiempo real son cruciales.

Una de las principales razones para utilizar MQTT en este proyecto es su eficiencia en términos de ancho de banda y recursos del sistema. Al ser un protocolo ligero, ocupa menos espacio en los dispositivos y consume menos energía en comparación con otros protocolos más pesados. Esto es especialmente útil en esta aplicación, donde los dispositivos son potentes, pero tienen los recursos limitados y se quiere que este la posibilidad de usarlos con baterías.

Otra ventaja de MQTT es su capacidad para garantizar la entrega de mensajes a través de diferentes niveles de calidad de servicio (QoS). Dependiendo de la importancia y el tipo de datos, se pueden seleccionar diferentes niveles de QoS para asegurar la entrega de mensajes y minimizar la posibilidad de pérdida de datos.

2.2.2 Modelo de publicador/subscriptor

El patrón de comunicación publicador/subscriptor en el protocolo MQTT, permite la interacción entre dispositivos mediante el uso de "topics", en lugar de establecer conexiones directas entre ellos. En este contexto, los dispositivos pueden funcionar como publicadores, subscriptores o ambos. Los componentes de esta comunicación son los tres siguientes:

- **Publicadores**: Estos dispositivos son los encargados de generar y transmitir datos o información a través de mensajes. Los mensajes se asocian a “topics” específicos, que son cadenas de texto jerárquicas utilizadas para organizar y clasificar los mensajes según su contenido o finalidad. Por ejemplo, en el sistema diseñado para este proyecto, el dispositivo que tiene conectado los sensores, una vez recopilados los datos, los publica en el “topic = TFGdemencia”.

- **Subscriptores**: Son dispositivos interesados en obtener mensajes de determinados “topics”. Para ello, se suscriben a temas concretos y reciben todos los mensajes publicados en dichos temas. Por ejemplo, en el caso de este proyecto tanto la interfaz de usuario, como un dispositivo encargado de controlar la temperatura, se subscriben al “topic = TFGdemencia” para recibir y actuar con la medición de los sensores.

- **Broker MQTT**: Este es el componente central en la arquitectura MQTT se encarga de gestionar la comunicación entre publicadores y subscriptores. El broker recibe los mensajes de los publicadores, los procesa y los distribuye a los dispositivos subscriptores apropiados. Además, el broker controla las suscripciones y la autenticación de los dispositivos, y puede garantizar la entrega de mensajes según los niveles de calidad de servicio (QoS) definidos.

2.2.3 Calidad de servicio

La calidad de servicio en el protocolo MQTT es al nivel de garantía en la entrega de mensajes entre publicadores y subscriptores. Existen tres niveles de calidad de servicio en MQTT, que permiten a los desarrolladores elegir el nivel de garantía de entrega de mensajes según las necesidades de su aplicación:

- **Calidad de servicio 0**: Este nivel de QoS garantiza que los mensajes se entregan a los subscriptores a lo sumo una vez, pero no se garantiza la entrega. Los mensajes pueden perderse si hay problemas de conexión o si el receptor no está disponible. Este nivel de QoS es adecuado para aplicaciones en las que no es crítica la pérdida ocasional de datos, y es preferible un menor consumo de recursos.

- **Calidad de servicio 1**: Este nivel de QoS asegura que los mensajes se entregan a los subscriptores al menos una vez, pero es posible que se entreguen múltiples copias del mismo mensaje. El publicador almacena y retransmite el mensaje hasta que el receptor confirme la recepción.

- **Calidad de servicio 2**: Este nivel de QoS garantiza que los mensajes se entregan a los subscriptores exactamente una vez, sin pérdida ni duplicación. Se utiliza un proceso de intercambio de cuatro mensajes entre el publicador y el subscriptor para asegurar la entrega única y confirmada. Este nivel de QoS es el más seguro, pero también el más lento y consume más recursos.

Cada mensaje enviado en MQTT tiene un nivel de QoS asociado, y tanto el publicador como el subscriptor pueden especificar su nivel de QoS preferido para cada tema. El broker MQTT es responsable de gestionar y aplicar los niveles de QoS en la entrega de mensajes.

2.2.4 Estructura básica de un sistema IoT

La estructura básica [12] para poder funcionar de un sistema IoT que utiliza el protocolo MQTT como método de comunicación, consta de varios elementos clave para procesar y actuar con los datos que recopila. Estos componentes son los siguientes:

- **Sensores y actuadores**: Los dispositivos IoT y sensores son fundamentales en cualquier sistema IoT. Incluyen una amplia gama de sensores (temperatura, humedad, presión, luz, distancia, etc.) y actuadores (motores, relés, bombillas, etc.) que interactúan con el entorno para recopilar datos y realizar acciones.

- **Conectividad**: La conectividad en un sistema IoT basado en MQTT como es este proyecto, se centra en la comunicación entre dispositivos y el bróker utilizando el protocolo MQTT.

- **Plataforma IoT**: La plataforma IoT recibe y procesa los datos recopilados por los dispositivos y sensores a través del broker MQTT. Algunas plataformas IoT populares con soporte para MQTT incluyen AWS IoT, Microsoft Azure IoT y Google Cloud IoT, en el caso de este proyecto se usará la plataforma AWS IoT.

- **Aplicaciones y servicios**: Las aplicaciones y servicios se construyen sobre la plataforma IoT y utilizan los datos procesados y analizados para ofrecer funcionalidades específicas al usuario final. Este proyecto se basa en el uso del servicio IoT Core dentro de AWS para conseguir realizar las distintas interacciones entre microcontroladores una vez recopilados los datos.

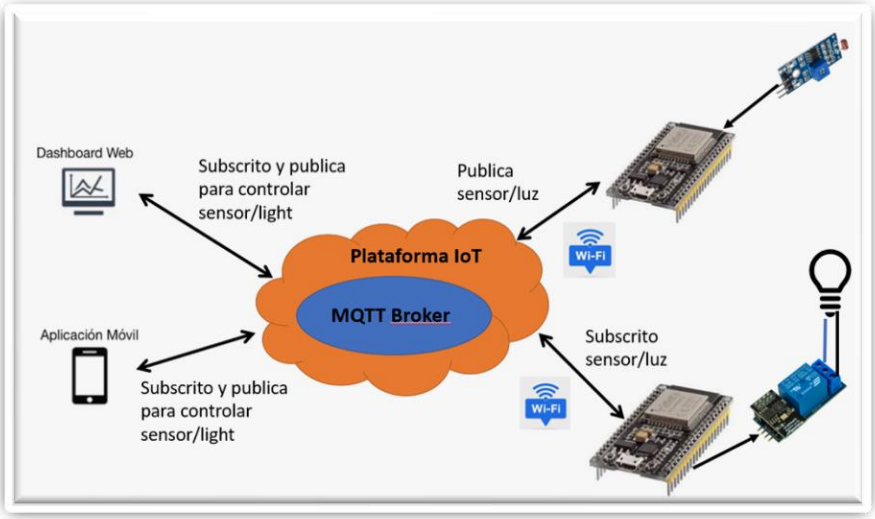

Figura 3. Esquema de una estructura básica de un sistema IoT

2.3 ESP32 ESP-WROOM-32 NodeMCU

2.3.1 Introducción

El ESP32[7] es un microcontrolador económico y bajo consumo energético desarrollado por Espressif Systems, una empresa china de tecnología y lanzado en septiembre de 2016. Cuenta con una arquitectura de doble núcleo basada en el procesador Tensilica Xtensa LX6, que puede funcionar a velocidades de hasta 240 MHz. Además, el ESP32 incorpora conectividad Wi-Fi 802.11 b/g/n y Bluetooth 4.2.

Es un microcontrolador versátil que ofrece diferentes métodos y entornos de programación para adaptarse a las necesidades y preferencias de los desarrolladores. Algunos de los más utilizados son Arduino IDE, MicroPython, ESP-IDF y PlatformIO, en el caso de este proyecto se ha usado Arduino IDE ya que es uno de los más populares y tiene una fácil instalación en Windows 11.

Se ha elegido este microcontrolador frente a los que podrían ser otras opciones viables como el ESP8266, una placa del propio fabricante Arduino o incluso un Raspberry Pi, porque tiene alguna ventaja para este diseño frente a cada una de las opciones con las que compite.

Frente al ESP8266, este es el modelo anterior del mismo fabricante y el ESP32 es más potente a la hora de procesar, por disponer de un mejor procesador. Respecto a los modelos del fabricante Arduino, son más fáciles de utilizar en un principio, pero están más limitados tanto a la hora de la conectividad, teniendo que utilizar módulos extra para esta función, como en el poder de procesamiento por estar destinados a aplicaciones más sencillas. Ya, por último, se ha elegido este microcontrolador respecto una Raspberry Pi, porque el ESP32 es de un coste bastante inferior y está destinado a aplicaciones más enfocadas al bajo consumo.

2.3.2 Características técnicas

Las especificaciones del módulo de desarrollo utilizado en este proyecto basándonos en la ficha técnica del distribuidor [8], son las siguientes:

Categoría	Especificaciones
Procesador	Doble núcleo Xtensa LX6, que funciona a una velocidad de reloj de hasta 240 MHz.
Estándar inalámbrico	FCC/CE/IC/TELEC/KCC/SRRC/NCC.
Conectividad	802.11 b/g/n/d/e/I/k/r Bluetooth 4.2 BR/EDR y BLE
Rango de frecuencias	2.4 - 2.5 GHz
Puertos de entrada y salida	36 pines GPIO, 18 canales de ADC de 12 bits, 2 canales de DAC de 8 bits, 2 puertos I2C, 3 puertos UART, 2 puertos SPI
Especificaciones bluetooth	Receptor NZIF con -98dBm de sensibilidad. Transmisor de Clase-1, Clase-2 y Clase-3 AFH, CVSD y SBC
Memoria	4 MB de memoria flash y 520 KB de memoria SRAM
Tipo de antena	Antena de PCB integrada
Características eléctricas	3.3 V de operación 15 mA de corriente de salida por pin GPIO La media de la corriente de trabajo es 80 mA
Temperatura de operación	Entre -40°C y +125 °C
Modos de operación	Station / SoftAP / SoftAP + Station / P2P
Tipo de seguridad	WPA / WPA2 / WPA2-Enterprise / WPS
Tipo de encriptación	AES / RSA/ ECC / SHA
Actualización de firmware	UART Download / OTA / Host
Protocolo de red	IPv4, IPv6, SSL, TCP / UDP / FTP / HTTP / MQTT.
Configuración de usuario	AT + Order Set, Web Android / iOS, Cloud Server

Tabla 3. Resumen de las características del módulo de desarrollo ESP32 [8]

2.3.3 Esquema de pines

El esquema de pines de la ESP32 ESP-WROOM-32 NodeMCU se puede observar en la "Figura 4", es uno de los aspectos más importantes a tener en cuenta al utilizar esta placa de desarrollo. Los 38 pines que contiene se pueden clasificar en los siguientes tipos según su función:

- **Pines GPIO**: que se pueden utilizar para conectar una amplia variedad de dispositivos externos, como sensores, actuadores, pantallas, entre otros.

- **Pines de alimentación**: la placa tiene varios pines de alimentación, incluyendo VCC, que se utiliza para alimentar la placa con una fuente de alimentación externa, y VIN, que se puede utilizar para alimentar la placa mediante el conector micro-USB.

- **Pines de comunicación**: tiene varios pines de comunicación, incluyendo dos puertos I2C, dos puertos SPI y tres puertos UART, que se pueden utilizar para conectar dispositivos de comunicación externos, como sensores y módulos GPS.

- **Pines ADC y DAC**: la placa tiene canales ADC de 12 bits, que se pueden utilizar para medir señales analógicas, como la temperatura y la humedad. También cuenta con canales DAC de 8 bits, que se pueden utilizar para generar señales analógicas.

- **Otros pines**: también cuenta con otros pines, como el pin EN, que se utiliza para activar o desactivar la placa, y el pin RST, que se utiliza para reiniciar la placa.

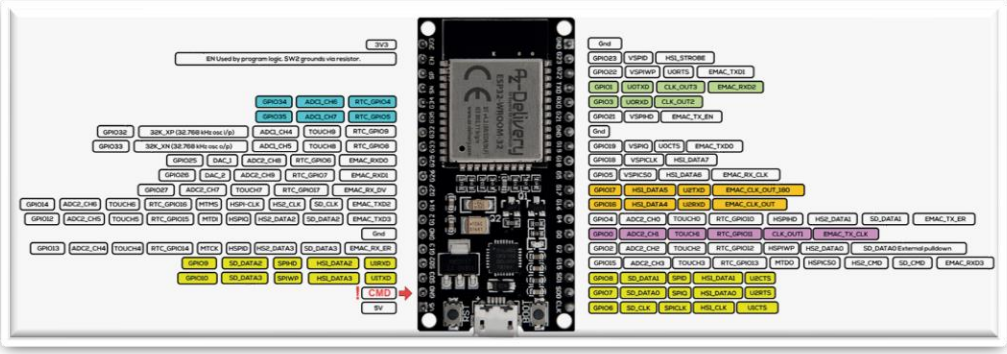

Figura 4. Esquema de pines del distribuidor.

2.4 Sensores y actuadores

2.4.1 Introducción

Un sensor es un dispositivo que convierte una magnitud física, como la temperatura, la humedad o la luz, en una señal eléctrica que puede ser procesada y analizada por un microcontrolador como el ESP32. Los actuadores, por otro lado, son dispositivos que realizan una acción física, como mover un motor o encender una luz, en función de las señales recibidas del microcontrolador. La combinación de sensores y actuadores permite al ESP32 interactuar con el entorno físico y realizar tareas automatizadas y controladas de forma remota a través de aplicaciones IoT.

2.4.2 Sensores para ESP32

Con un ESP32 se pueden utilizar muchos sensores de distintos tipos gracias a sus múltiples interfaces de comunicación y capacidades de entrada/salida analógica y digital. Entre los muchos posibles, a continuación, se resumen las categorías más importantes relacionadas con este proyecto.

2.4.2.1 Sensores de temperatura y humedad

Estos sensores permiten medir la temperatura y la humedad del entorno. Son útiles en aplicaciones como sistemas de control de clima, estaciones meteorológicas y monitoreo de invernaderos.

- **DHT11**: Estos sensores ofrecen mediciones básicas de temperatura y humedad a bajo costo. En el caso de este proyecto este modelo será el utilizado para medir la humedad por posibles incontinencias en el colchón, se elige este modelo frente a la competencia, porque no necesitando más precisión de la que nos ofrece este sensor, ya que vamos a pasar de una situación con baja humedad a alta en poco tiempo, este sensor es el mejor en su rango de precio.

- **BME280**: Este sensor proporciona mediciones de temperatura, humedad y presión barométrica con alta precisión.

- **DS18B20**: Este sensor digital de temperatura es fácil de conectar y tiene una alta precisión y resolución. Este modelo será utilizado para controlar la temperatura de la habitación, ya que se puede conseguir de forma económica y con un formato en el que es una sonda, por lo cual, será más resistente a cualquier accidente que le ocurra.

2.4.2.2 Sensores de luz

Los sensores de luz detectan la intensidad de la luz en el entorno y son útiles en aplicaciones como control de iluminación automático, sistemas de seguimiento solar y monitoreo del crecimiento de las plantas.

- **LDR**: Es un componente simple y económico que cambia su resistencia según la intensidad de la luz. Este sensor se puede regular para captar una intensidad lumínica en concreto y será el utilizado para captar si la luz de la habitación esta encendida o apagada durante la noche. Se va a utilizar un modelo en concreto que incorpora un led para indicar cuando está funcionando y un potenciómetro para regular su precisión de manera manual, se elige este frente a un sensor de luz ambiente, que puede ser la alternativa más clara, por ser una opción regulable sin cambiar el código.

- **BH1750** y **TSL2561**: Estos sensores digitales de luz ambiente ofrecen mediciones precisas y comunicación a través de protocolos I2C.

2.4.2.3 Sensores de movimiento y presión.

Estos sensores detectan cambios en la posición, velocidad y orientación de los objetos y se utilizan en aplicaciones como robótica, sistemas de seguridad y vehículos autónomos. Algunos ejemplos incluyen:

- **NEO-6M**: Este módulo GPS permite determinar la posición geográfica, la velocidad y la altitud en tiempo real.

- **HC-SR04**: Este sensor de distancia por ultrasonidos mide la distancia a objetos cercanos mediante el tiempo de vuelo de las ondas ultrasónicas. Se usará en este proyecto para detectar el cambio en la distancia cuando esta la cabeza del paciente sobre la almohada y cuando no, se ha elegido este frente a la competencia por su rango de distancia, más detallado en el capítulo 4, es adecuado para esta función.

- **SR602**: Es un sensor de movimiento de infrarrojos pasivo de bajo costo y compacto, que detecta la presencia de seres humanos u objetos en movimiento al medir los cambios en la radiación infrarroja. Este sensor es ideal para aplicaciones como sistemas de seguridad, automatización del hogar, control de iluminación… En este proyecto se va a utilizar para detectar el movimiento de la persona en el caso que se levante de la cama, ya que consigue abarcar gran parte de la zona cerca de la cama.

- **Sensores de Presión de Fuerza Resistiva (FSR)**: Estos sensores se activan cuando se aplica una fuerza sobre su superficie. Cambian su resistencia en función de la presión que se les aplica.

- **SW-420**: Es un módulo de detección de vibraciones, este sensor se basa en el principio de la resistencia conductora y es muy sensible a los cambios en la vibración o el movimiento. Dentro de este proyecto se va a utilizar para detectar posibles golpes en la cama/paredes, se elige este frente a otros que podrían ser una opción, ya que este aparte de detectar el golpe, detecta también si la cama simplemente se mueve y tiene la ventaja como el LDR de poder regularse de forma externa con un potenciómetro.

- **KY-002**: Es un módulo de interruptor de vibraciones, tiene un pequeño resorte metálico en su interior que completa un circuito cuando se agita o se somete a vibración, lo que hace que la señal de salida cambie de estado.

2.4.2.4 Sensores de audio.

Este tipo de sensores son fundamentales para capturar y transmitir sonido. Se utilizan en asistentes de voz inteligentes, sistemas de seguridad y automatización del hogar para detectar comandos de voz o sonidos inusuales.

- **Sensor de Sonido KY-037**: Este sensor tiene dos salidas, una analógica y una digital. La salida analógica proporciona una señal de voltaje proporcional a la intensidad del sonido detectado, y la salida digital proporciona una señal binaria que indica si la intensidad del sonido ha superado un cierto umbral. Este sensor va a ser el utilizado en este proyecto

por su bajo coste y ya que la salida digital se puede regular con un potenciómetro. Va a ser empleado para detectar voz humana.

- **SPH0645LM4H**: Este es un micrófono digital MEMS de alta calidad que utiliza el bus I2S para la comunicación, lo que facilita la obtención de datos de audio de alta calidad desde la ESP32.

2.4.3 Actuadores para la ESP32

Los actuadores son dispositivos que convierten una señal de control en un movimiento o acción física. Los tipos más utilizados junto a un ESP32 son los siguientes:

- **Motores**: los motores utilizados pueden ser de distintos tipos, por ejemplo, de corriente continua o alterna, motores paso a paso, servomotores.

- **Zumbadores**: son dispositivos electromecánicos que se utilizan para producir sonido o tono como resultado de una señal eléctrica.

- **Relés**: Los relés son dispositivos electromecánicos o de estado sólido que permiten controlar circuitos de alta potencia utilizando señales de control de baja potencia. Para este proyecto se emplean para controlar el encendido y apagado de aparatos que controlen la temperatura de la habitación.

- **Solenoides**: Los solenoides son actuadores electromagnéticos que convierten la energía eléctrica en movimiento lineal.

- **Pantallas**: las pantallas pueden ser otro dispositivo que sirva como actuador, reproducir algo sobre ella a raíz de que se envié una señal eléctrica. En la interfaz de usuario del sistema IoT de este proyecto, se usa un modelo de pantalla LCD con 2 filas y 16 columnas.

2.4.4 Implementación de los sensores y actuadores con el ESP32.

Para conectar sensores y actuadores al ESP32, se deben seguir las especificaciones eléctricas y de conexión proporcionadas por los fabricantes de los dispositivos. Esto implica conectar correctamente las señales de alimentación, las señales de datos y las señales de control a los pines GPIO del ESP32. El diseño de la conexión física debe tener en cuenta la compatibilidad de voltaje, las limitaciones de corriente de los pines GPIO y los dispositivos conectados. Todas estas cosas se deben tener muy en cuenta ya que conectar por ejemplo el voltaje con la polaridad al contrario podría estropear el sensor fácilmente. Todo esto se explicará más en detalle en el sexto capítulo.

Es necesario instalar y configurar el entorno de desarrollo, asegurando que se incluyan las bibliotecas necesarias para los ciertos sensores/actuadores. Las bibliotecas son un conjunto de funciones y métodos que simplifican la comunicación y el control de los sensores y actuadores.

Capítulo 3. Configuraciones iniciales

Antes de plantear empezar a montar la red de sensores y conectarlos entre sí, hay que decidir de qué forma configurar los dispositivos y conectarlos entre ellos. En el caso de este proyecto donde se van a usar como microcontroladores los ESP32, se decide que se van a programar con el entorno de desarrollo Arduino IDE y la interconexión entre ellos se va a hacer con el protocolo de comunicación MQTT, a través de la plataforma Amazon Web Service con ayuda de su herramienta IoT Core. En este capítulo, se va a explicar la forma en la que se han realizado las configuraciones iniciales.

3.1 Instalación y configuración de Arduino IDE

Lo primero necesario para comenzar a programar el ESP32 es instalar el entorno de desarrollo y configurarlo. Para instalarlo es necesario seguir los siguientes pasos (Se va a explicar la instalación para Windows 11, ya que es el sistema operativo que se dispone para realizar este proyecto):

1. El primer paso será descargar la aplicación de “Microsoft Store”, se descargará automáticamente la última versión del Legacy IDE.

Figura 5. Logotipo dentro de la Microsoft Store

2. En este segundo paso habrá que abrir el IDE y configurarlo para que se puedan seleccionar placas con el chip ESP32, esto se conseguirá siguiendo los siguientes pasos:

   I. Archivo > Preferencias > Pegar dentro del apartado Gestor de URLs Adicionales de tarjetas el siguiente texto: https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

   II. Ahora, tocaría ir a Herramientas > Placa > Gestor de tarjetas.

   III. En el cuadro de búsqueda del Gestor de tarjetas, habrá que escribir "esp32" y darle a buscar. Aparecerá un resultado llamado "esp32” desarrollado por Espressif System. Habrá que instalar la última versión.

   IV. Por último, solo habrá que seleccionar la placa que vamos a utilizar, Herramientas > Placa > ESP32 Arduino > ESP32-WROOM-DA Module, en este caso esta porque es la compatible con nuestro modelo de placa de desarrollo.

3.2 Programación del ESP32

Con el entorno de programación preparado, lo primero será conectar la ESP32 a un puerto USB de la maquina donde se vaya a programar con un cable Micro-USB al módulo de desarrollo y configurar el puerto Herramientas > Puerto > "Puerto COM que corresponda con el que está conectado a la placa"

Para empezar a programar lo primero será crear un nuevo sketch, Archivo > Nuevo (File > New). En la ventana del sketch, Arduino IDE ha generado automáticamente dos funciones, la función void setup() que se ejecuta una vez al inicio del programa, se utiliza para inicializar variables, configurar pines y establecer comunicaciones, y la función void loop() que se ejecuta en bucle después de setup(), es donde se coloca el código principal del programa.

Una vez tenemos el código ya escrito hay que verificarlo, se verifica haciendo clic en el ícono de verificación (✓) en la esquina superior izquierda de Arduino IDE. Esto compilará el código y mostrará posibles errores en la ventana de salida o el mensaje "Compilación terminada" si está correcto.

Para cargar el código en el ESP32, haga clic en el ícono de carga (→) en la esquina superior izquierda, justo a la derecha de él de verificación. Arduino IDE compilará el código y luego lo cargará en el módulo ESP32 a través del puerto USB seleccionado (En el caso del módulo de este proyecto adicionalmente habrá que mantener pulsado el botón físico de "BOOT" en la ESP32, en el momento que el IDE está intentando conectar con la placa para poder cargar el código).

3.2.1 Verificación de funcionamiento

Una vez que esta el código que se requiere para que el microcontrolador funcione correctamente, es el momento de comprobar que el código cumple con las especificaciones que se necesitan. Para verificar esto, el entorno de desarrollo contiene herramientas para ayudar en este paso, como el Monitor Serie "Herramientas > Monitor Serie" o el Serial Plotter si es necesario visualizar valores en forma de grafica "Herramientas > Serial Plotter", y otra opción en el caso que el código se encargue de cambiar el estado de elementos físicos, también se podría verificar a simple vista.

3.3 Plataforma IoT

3.3.1 Introducción

La plataforma de servicios en la nube usada para desarrollar este proyecto va a ser AWS, utilizando su servicio proporcionado para IoT, llamado IoT Core. Amazon Web Services IoT Core [9] es un servicio en la nube completamente administrado que permite a los dispositivos conectados interactuar de forma fácil y segura, con aplicaciones en la nube y otros dispositivos. Facilita la conexión, la comunicación y el control de dispositivos de IoT, proporcionando una infraestructura sólida para recopilar, procesar, analizar y actuar sobre los datos generados por estos dispositivos. En el caso de este proyecto se va a utilizar de forma limitada y gratuita, pero es una muy buena opción para si en un futuro se quiere escalar este prototipo a un producto de mercado, poder escalar el número de dispositivos y las conexiones entre ellos.

Entre las muchas funcionalidades que incluye para facilitar la administración de los dispositivos IoT, esta conectar dispositivos con protocolos y estándares eficientes como MQTT, HTTP y WebSocket, administrar las identidades con un sistema donde cada dispositivo tiene una única y con certificados X.509 para garantizar la autenticación, asegurar que las comunicaciones entre dispositivos y el servicio estén cifradas.

3.3.2 Conexión entre dispositivos y seguridad en la comunicación.

Una vez que ya está el entorno de desarrollo preparado para poder ejecutar un código dentro de un ESP32 y queremos que dos microcontroladores se comuniquen entre ellos, es el momento de configurar y preparar AWS IoT Core para recibir y procesar los datos. El primer paso para acceder al servicio de IoT Core será crear una cuenta dentro de AWS, una vez se dispone de la cuenta simplemente hay que buscar en el buscador de AWS el servicio “IoT Core”.

Una vez dentro del servicio de IoT Core solo habrá que seguir los siguientes puntos donde se describen los pasos necesarios para poder vincular una ESP32 de manera segura y de forma que cada dispositivo tenga un identificador único.

3.3.2.1 Crear una “Policy”

Una “Policy” es un documento JSON que define los permisos que un dispositivo o cliente tiene para realizar acciones específicas, como conectarse, publicar, suscribirse y recibir mensajes. Las políticas controlan el acceso a los recursos y operaciones dentro de AWS IoT Core. En este proyecto para conectar los microcontroladores a IoT core, vamos a crear una de la siguiente manera y utilizaremos la misma en todos los microcontroladores ya que necesitarán los mismos permisos.

Para crear una iremos a "Secure" en la barra lateral y habrá que seleccionar "Policies", después habrá que hacer clic en "Create a policy". Una vez se haya abierto el panel de configuración, habrá que asignar un nombre a la “Policy” y rellenar los campos de “Policy effect”, “Policy action” y “Policy resource”, esto habrá que rellenarlos para cada regla necesaria, para permitir las acciones deseadas en el dispositivo.

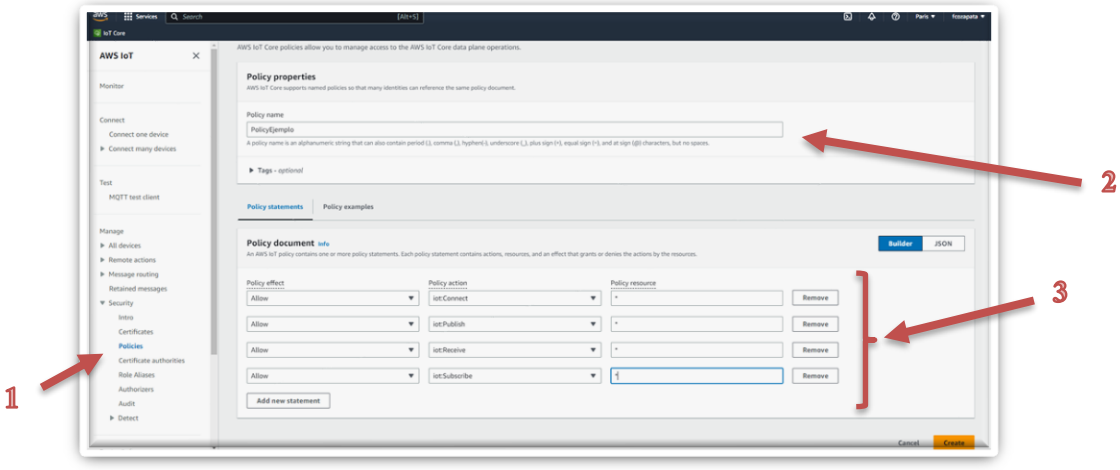

Figura 6. Pasos que seguir para crear una política.

3.3.2.2 Crear una "Thing"

El siguiente paso, para conseguir conectar dos o más microcontroladores entre sí, será crear lo que se llama dentro de la IoT Core "Thing", esto es lo que se conoce dentro de la plataforma como una representación virtual de un dispositivo físico y un registro de un dispositivo físico en la nube, sirven para organizar y administrar dispositivos conectados. Para crearlo hay que buscar el botón de "Manage" en la barra lateral y hacer clic en "Things", después hay que hacer clic en "Create a Thing" y selecciona "Create a single thing".

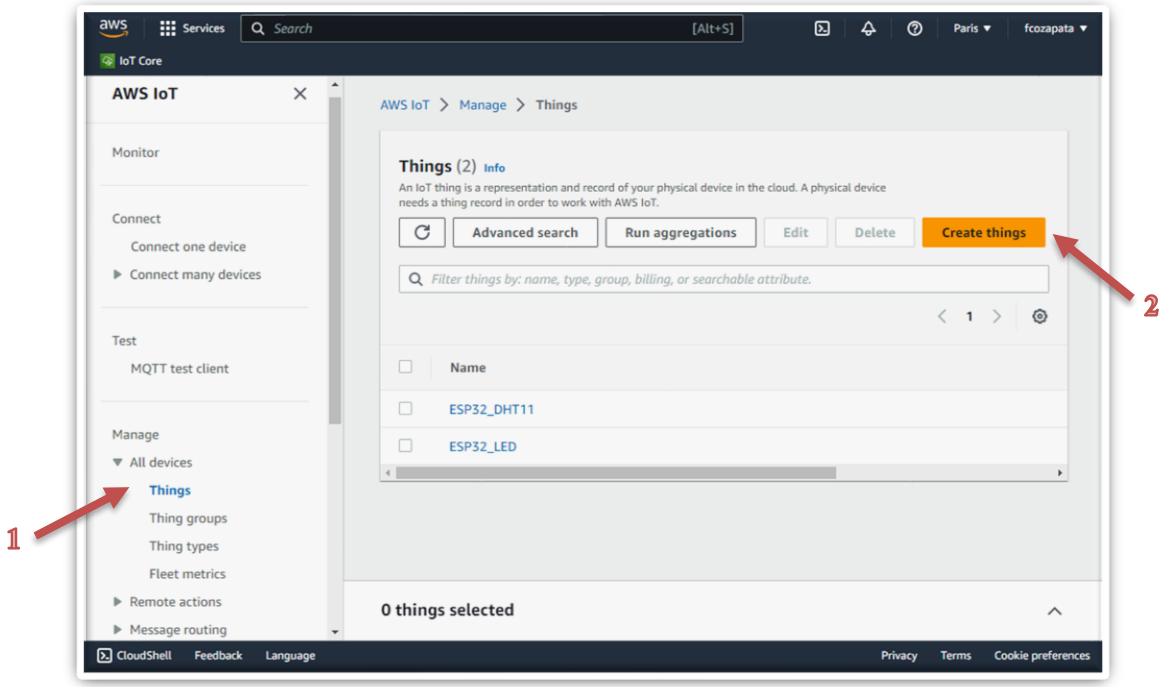

Figura 7. Primeros pasos para crear una "Thing"

Una vez hecho esto solamente se continua el proceso, siguiendo los pasos que te indica la aplicación, en primer lugar, se ha asignado el nombre que identificara a esa "Thing". Luego se ha de seleccionar como queremos hacer para generar los certificados, en nuestro caso le dejaremos la opción recomendada para que los autogenere y posteriormente añadírselos a nuestros dispositivos dentro del Arduino IDE. Por último, seleccionamos la "Policy" creada anteriormente que se amoldará a nuestras necesidades a la hora de utilizar el dispositivo.

Como último paso, le presionaremos sobre "créate thing" y nos dará la oportunidad de descargar los certificados autogenerados, la clave pública y privada serán obligatorias descargarlas en ese momento, ya que posteriormente no se podrá. En este proyecto cada uno de los microcontroladores ESP32 deberá tener una "Thing" única asignada, es un requisito ya que si se le asigna la misma a dos ESP32 se solaparían y solo uno estaría realmente conectado a IoT Core.

3.3.2.3 Certificados necesarios para conseguir el enlace entre la ESP32 y IoT Core

Los certificados y claves son componentes esenciales de la seguridad en la comunicación entre un dispositivo y AWS IoT Core. Estos archivos se utilizan para autenticar y cifrar la comunicación entre el dispositivo y el servicio en la nube. A continuación, se describen los utilizados para la comunicación que se va a enlazar entre dispositivo y servidor:

- **El certificado del dispositivo:** Este archivo contiene el certificado X.509 del dispositivo. Un certificado X.509 es un estándar de criptografía que define la estructura y el contenido de los certificados digitales. El certificado contiene información sobre el propietario del certificado, la clave pública y la entidad emisora del certificado. Este certificado se utiliza para autenticar al dispositivo cuando se comunica con AWS IoT Core.

- **Clave publica:** Este archivo contiene la clave pública del par de claves asimétricas generadas para el dispositivo. La clave pública la utiliza AWS IoT Core para el proceso de autenticación y cifrado de la comunicación, aunque nos obligue a descargarla no habrá que hacer nada con ella en el caso de este proyecto.

- **Clave privada:** Este archivo contiene la clave privada del par de claves asimétricas generadas, debe mantenerse en secreto y almacenarse de forma segura en el dispositivo. Se utiliza para descifrar los mensajes recibidos de AWS IoT Core y firmar digitalmente los mensajes enviados al servicio IoT Core.

- **Certificado de la Autoridad de Certificación:** Este archivo contiene el certificado de la Autoridad de Certificación (CA) raíz de Amazon. Una Autoridad de Certificación es una entidad confiable que emite y valida certificados digitales. El certificado CA raíz es necesario para establecer una cadena de confianza durante el proceso de autenticación y para verificar que el certificado del dispositivo fue emitido por una CA confiable. En el caso de AWS IoT Core, el certificado raíz de Amazon se utiliza para validar la conexión segura entre el dispositivo y el servicio.

Estos archivos se utilizan conjuntamente para establecer una conexión segura (TLS) entre el dispositivo ESP32 y AWS IoT Core, garantizando que la comunicación esté cifrada y autenticada. Almacenar y gestionar correctamente estos archivos es fundamental para mantener la seguridad en la comunicación entre el dispositivo y AWS IoT Core. En la siguiente imagen se muestra donde descargar estos archivos al acabar el proceso de crear una “Thing”, en el caso de nuestro proyecto habría que descargar todos los archivos marcados menos la clave pública ya que tendremos que introducirlos en el código que se introduce en el microcontrolador.

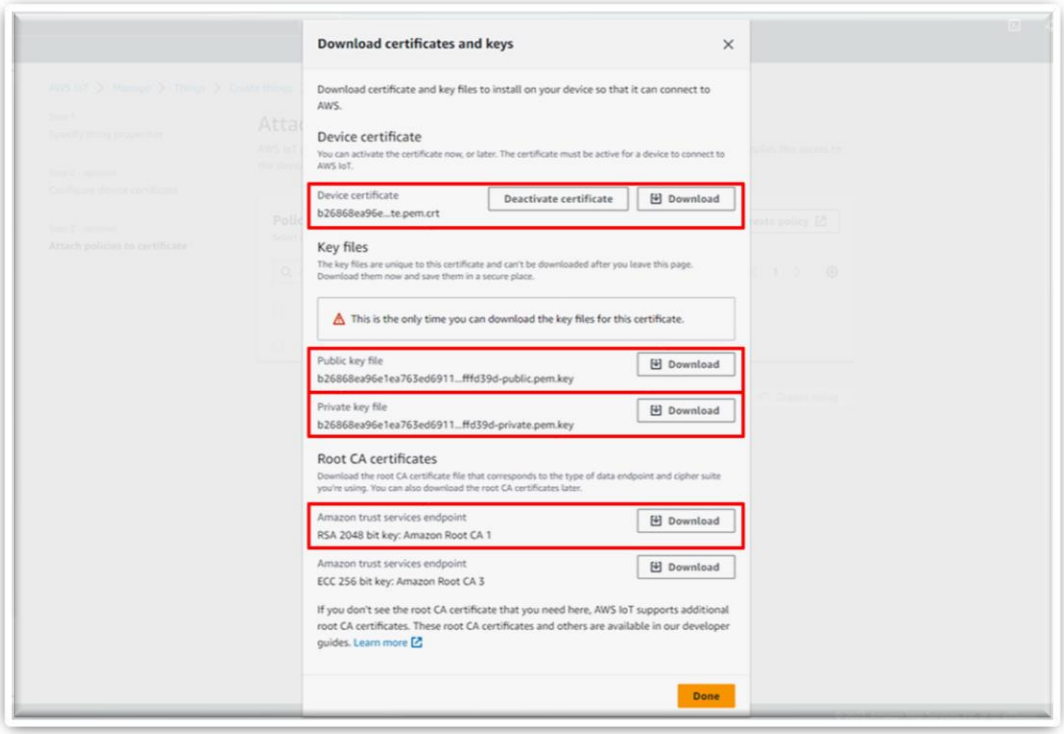

Figura 8. Pantalla donde descargar los certificados.

Capítulo 4. Tipos de funcionalidades a monitorear

Este proyecto tiene como propósito diseñar un sistema para monitorizar distintos parámetros sobre la actividad de una persona que sufre demencia senil, todo esto en el momento que esta con la enfermedad en una etapa avanzada. Los efectos de la enfermedad [10] en la persona van haciéndose más significativos con el tiempo, entre los muchos que afectan al paciente los que han motivado a diseñar este prototipo son los siguientes:

- **Deterioro cognitivo**: Entre los distintos efectos que engloba, los importantes para este proyecto son problemas con la memoria a corto plazo (a medida que la enfermedad avanza también afecta la memoria a largo plazo) y la orientación a la hora de ubicarse en el tiempo.

- **Cambios en el comportamiento**: Estos pueden incluir cambios de personalidad como agitación, irritabilidad o ansiedad, que son efectos que pueden aparecer sin control del paciente y ser necesarios de tratar en el momento.

- **Dificultades motoras**: Algunas formas de demencia pueden afectar la coordinación física y el equilibrio, por lo que en etapas avanzadas de la enfermedad impide al paciente andar sin una persona que le asiste al lado por peligro de caerse.

- **Autocuidado y funciones diarias**: A medida que la demencia avanza, puede interferir con la capacidad de la persona para cuidar de sí misma, por ejemplo, olvidar comer o beber, no mantener la higiene personal o tener dificultades para vestirse.

Por estos efectos, que van aumentando con el deterioro del paciente a causa de la enfermedad, este proyecto quiere conseguir un sistema IoT realmente útil, para monitorear en todo momento cambios en la actividad y poder prevenir posibles problemas por no conseguir estar una persona todo el rato vigilando al paciente.

4.1 Indicadores sobre la actividad del paciente

Contextualizando la situación de la persona que se pretende monitorear, se trata de una persona, la cual, los efectos de la enfermedad hacen que pase la mayor parte del tiempo en cama, aunque aún sería capaz de andar, necesita de hacerlo con otra persona al lado, por riesgo de caída, ya que la musculatura se debilita como consecuencia de que la enfermedad se agrava con el tiempo. El estado de su memoria a corto plazo está muy deteriorado y no se sabe orientar ni en la edad que tiene, ni en el año en el que vive. Estos efectos hacen que no recuerde gran parte de sus recuerdos, se pueda poner agresivo en ciertos momentos y necesitar de una persona que pueda asistirlo por un cambio en su actividad en cualquier momento. Respecto a las características descritas se han desarrollado las siguientes ideas, para monitorear al paciente cuando esta acostado en la cama.

4.1.1 Controlar si el paciente esta tumbado

El primer paso para comprobar la actividad del paciente es controlar que está acostado y no se ha incorporado por estar inquieto o por la necesidad de levantarse. Para ello la mejor forma de monitorizar esta situación, después de evaluar las diferentes opciones para este proyecto, ha sido medir la distancia entre un sensor de ultrasonidos y una pared, donde al estar acostado el paciente, la cabeza se interpondrá entre el sensor y la pared alterando la medición y pudiendo controlar que está con la cabeza apoyada sobre la almohada. Se utilizará el sensor de ultrasonidos HC-SR04, el cual tiene como especificaciones técnicas principales las de la siguiente tabla:

Características	
Voltaje de operación	5V DC
Rango de medición	2cm a 450cm
Precisión	+- 3mm
Frecuencia de ultrasonido	40KHz

Tabla 4. Especificaciones técnicas del sensor HC-SR04

Figura 9. Imagen del sensor HC-SR04

4.1.2 Medir el movimiento en un lateral de la cama

En el momento que una persona está con un estado avanzado de la enfermedad, es necesario controlar que no ande solo, ya que la desorientación puede hacer que salga de casa. Incluso a la hora de andar por la habitación podría tropezar, y al que pasar la mayoría del tiempo en la cama hace que se debiliten los músculos y la probabilidad de aguantar una caída es mucho menor.

Para monitorizar que la persona no se levante y ande se ha colocado un sensor de movimiento por infrarrojos pasivo, con este se pretende detectar movimientos en la zona de la habitación del lateral de la cama. Para evitar que detecte los que se produzcan por el paciente mientras está durmiendo o despierto, pero sin estar levantado, se coloca en un lateral de la cama por debajo de la altura donde duerme el paciente. El modelo elegido es el SR602, algunas de sus características más importantes son:

Características	
Distancia de detección	Hasta 5 metros.
Distancia recomendada	Entre 0 y 3,5 metros.
Salida	Nivel alto, H = 3.3V, L = 0V
Fuente de alimentación de CC	3.3V-15V
Corriente inactiva	20uA

Tabla 5. Especificaciones técnicas del sensor SR602

Figura 10. Imagen del sensor SR602

4.1.3 Medir el sonido para detectar al paciente hablar

Midiendo el sonido dentro de la habitación cuando el paciente está solo, se pretende identificar los momentos en los que se despierta y está hablando porque necesita algo o la atención de la persona que está a su cargo. Se ha decidido utilizar el modelo KY-037 del distribuidor AZ-Delivery, es un micrófono de detección de sonido de alta sensibilidad e irá colocado en un lateral de la cama cerca de la cabeza del paciente. Las características más importantes del dispositivo son las siguientes:

Características	
Rango del voltaje de operación	Entre 3,3V y 5V DC
Salida	Digital y analógica
Rango de frecuencias	Desde 100Hz hasta 10.000Hz
Sensibilidad	-46 ± 2.0 (0 dB = 1V/Pa) at 1kHz
Sensibilidad de ruido	58dB

Tabla 6. Especificaciones técnicas del micrófono KY-037

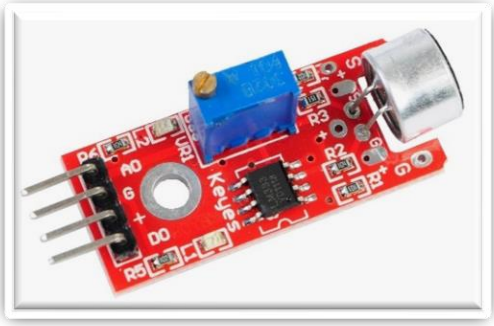

Figura 11. Imagen del sensor KY-037

4.1.4 Medir la humedad para detectar incontinencias en el colchón

Con esta utilidad se pretende que la persona a cargo del paciente no tenga que estar preocupada de las incontinencias. En una etapa de la enfermedad donde aún no es necesario que el paciente lleve una medida protectora para no orinarse encima, esta funcionalidad del sistema pretende detectar las posibles incontinencias que puedan ser producidas aleatoriamente, por ejemplo, por la desorientación o el estar mucho tiempo de forma continuada dormido. Se pretende incorporar un sensor DHT11 del distribuidor AZ-Delivery entre el colchón y la sabana bajera de la cama, a la altura de la cadera del paciente a la hora de estar acostado. Es un sensor digital de bajo costo y confiable, que mide temperatura y humedad ambiental. Sus principales características son las siguientes:

Características	
Rango del voltaje de operación	Entre 3,3V y 5V DC
Máxima corriente de operación	2.5mA máximo
Rango de humedad	20% - 90% con una precisión del 5%
Rango de temperatura	0°C - 50°C con una precisión de ±2°C
Frecuencia de muestreo	1Hz

Tabla 7. Especificaciones técnicas del sensor DHT11

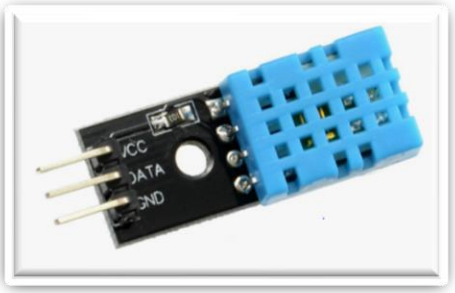

Figura 12. Imagen del sensor DHT11

4.1.5 Medir temperatura de la habitación para regularla

La idea de la implementación de un sensor para medir la temperatura de la habitación, es que el paciente esté siempre en un ambiente con la temperatura perfecta, para poder prevenir posibles enfermedades a raíz de excesivo frío o calor. Se pretende implementar esta función con un sensor de temperatura DS18B20, es un sensor que está introducido dentro de una sonda y se colocará cerca de la cama del paciente y a su misma altura para medir la temperatura de la habitación lo más precisa posible. La utilidad de medir la temperatura es mandar esa información a otro microcontrolador que encienda y apague con un relé un aparato capaz de regular la temperatura, un ejemplo en invierno puede ser un radiador y en verano un ventilador. Las principales características de este sensor son las siguientes:

Características	
Voltaje de funcionamiento	3,3V - 5 V
Fuente de alimentación	3,0 V a 5,5 V
Rango de temperatura	-55°C a 125°C con una precisión de ±0,5°C

Tabla 8. Especificaciones técnicas del sensor DS18B20

Figura 13. Imagen del sensor DS18B20

4.1.6 Medir la intensidad lumínica para detectar si se enciende alguna luz

La idea de esta función en el proyecto es muy sencilla, simplemente es detectar si se enciende la luz de la habitación durante la noche, esto significa que el paciente se ha despertado y necesita algo ya que ha encendido la luz, la utilidad va a ser detectarlo automáticamente y notificar a la persona que está a su cargo de esto en el caso de que no duerma en la habitación. El sensor que se ha utilizado para esto es un fotorresistor, ajustado mediante una resistencia variable para que detecte la intensidad lumínica cuando se enciende una luz de la habitación. El modelo de sensor que se ha utilizado es el LDR5528, montado en un módulo con la resistencia variable preparado para conectar directamente al microcontrolador del distribuidor Aliexpress. Las características principales del sensor son las siguientes:

Características	
Voltaje de funcionamiento	3,3 V-5 V
Salida	Salida de conmutación digital (0 y 1)

Tabla 9. Especificaciones técnicas del sensor LDR5528

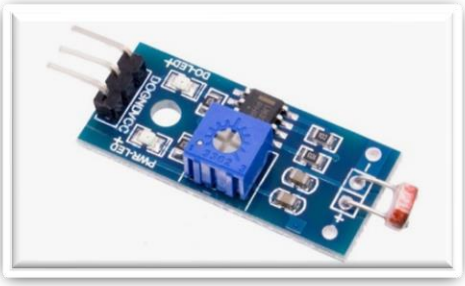

Figura 14. Imagen del sensor LDR5528

4.1.7 Medir vibraciones en la cama.

En muchos casos las personas que padecen demencia senil tienden a adoptar un carácter agresivo a causa de la pérdida de memoria, por lo que la idea de esta función es detectar posibles momentos en los que el paciente golpeé la cama o también la posible vibración que puede hacer sobre un lateral de la cama al intentar salir de ella para levantarse. La idea es colocar un sensor capaz de detectar vibraciones en el lateral de la cama, se va a utilizar el modelo SW420 del distribuidor AZ-Delivery. Tiene incorporada una resistencia variable con la que se puede cambiar la sensibilidad a la hora de medir una vibración y emitir por la salida de datos el resultado de esta. Sus características más importantes son las siguientes:

Características	
Rango del voltaje de operación	Entre 3,3V y 5V DC
Corriente de operación	15mA
Salida de datos	Digital

Tabla 10. Especificaciones técnicas del sensor SW420

Figura 15. Imagen del sensor SW420

4.1.8 Medir el estado de un pulsador de ayuda

Esta función es simplemente medir el estado de un pulsador colocado a la vista del paciente con un cartel de ayuda, para que lo pueda interpretar de la forma correcta, aunque se encuentre en una parte avanzada de la enfermedad. Simplemente se mandarán los datos del pulsador al microcontrolador que controla la interfaz de usuario y en el momento que este pulsado se notificara durante un periodo suficiente, como para que se pueda alertar de la situación al responsable del paciente. El pulsador se ha colocado en la cama de forma que sea intuitivo y accesible para el paciente.

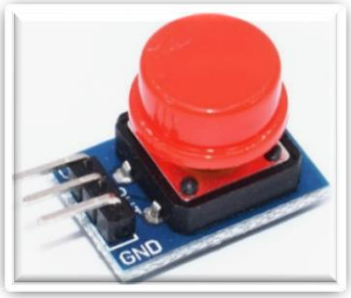

Figura 16. Imagen del botón utilizado

4.1.9 Diseño esquemático para la colocación de los sensores

En la siguiente imagen se resume la colocación de todos los sensores y actuadores dentro de una habitación, y el recorrido que harían los cables hasta su microcontrolador.

Figura 17. Esquema de la distribución de los sensores del sistema

Capítulo 5. Pseudocódigo para los microcontroladores

Para este proyecto se han utilizado tres microcontroladores ESP32 que han sido programados mediante el lenguaje de programación Arduino, cada uno de ellos cumple una función determinada dentro del sistema IoT, por lo cual tienen un código distinto. Las tres funciones implementadas, cada una realizada por uno de los microcontroladores, son las siguientes:

- Recoger y procesar los datos medidos por los sensores.
- Gestionar el estado de un relé, para que regule la temperatura de la habitación
- Gestionar una interfaz de usuario para notificar a la persona que está a cargo del paciente, del estado de todos los parámetros medidos por los sensores a través de diodos Leds y una pequeña pantalla LCD.

También dentro de cada microcontrolador hay una parte del código de estructura común, con la parte de código para la interconexión entre ellos a través de AWS IoT Core.

5.1 Pseudocódigo encargado de recoger los datos y procesarlos.

En el siguiente datagrama se ha descrito el pseudocódigo que describe el programa creado mediante Arduino, para hacer que el primer microcontrolador recoja datos de ocho sensores distintos, los procese y sea capaz de enviarlos todos al bróker del sistema IoT.

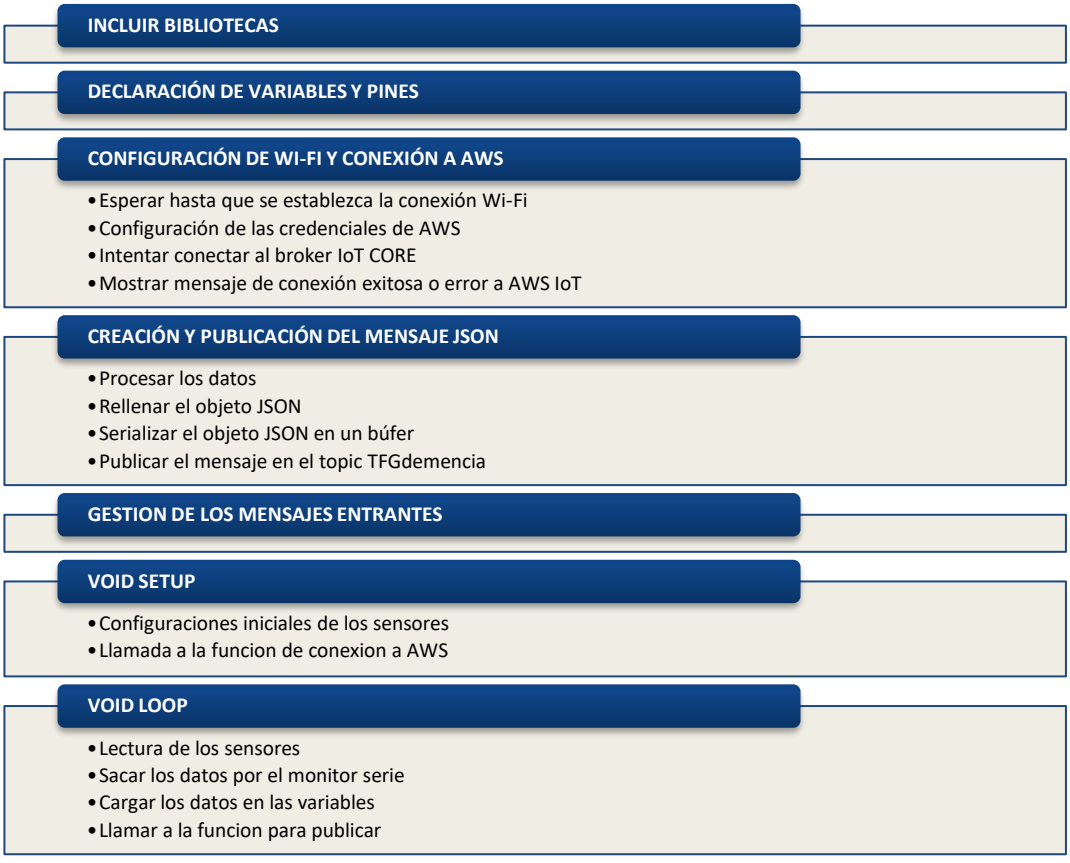

Figura 18. Pseudocódigo para leer los sensores

5.2 Pseudocódigo encargado de la interfaz de usuario.

En el siguiente datagrama se ha descrito el pseudocódigo que describe el programa creado mediante Arduino, para hacer que el segundo microcontrolador recoja los datos de la nube de AWS y los muestre en forma de señales lumínicas y a través de una pantalla LCD, siendo esto una interfaz para que el usuario pueda controlar al paciente.

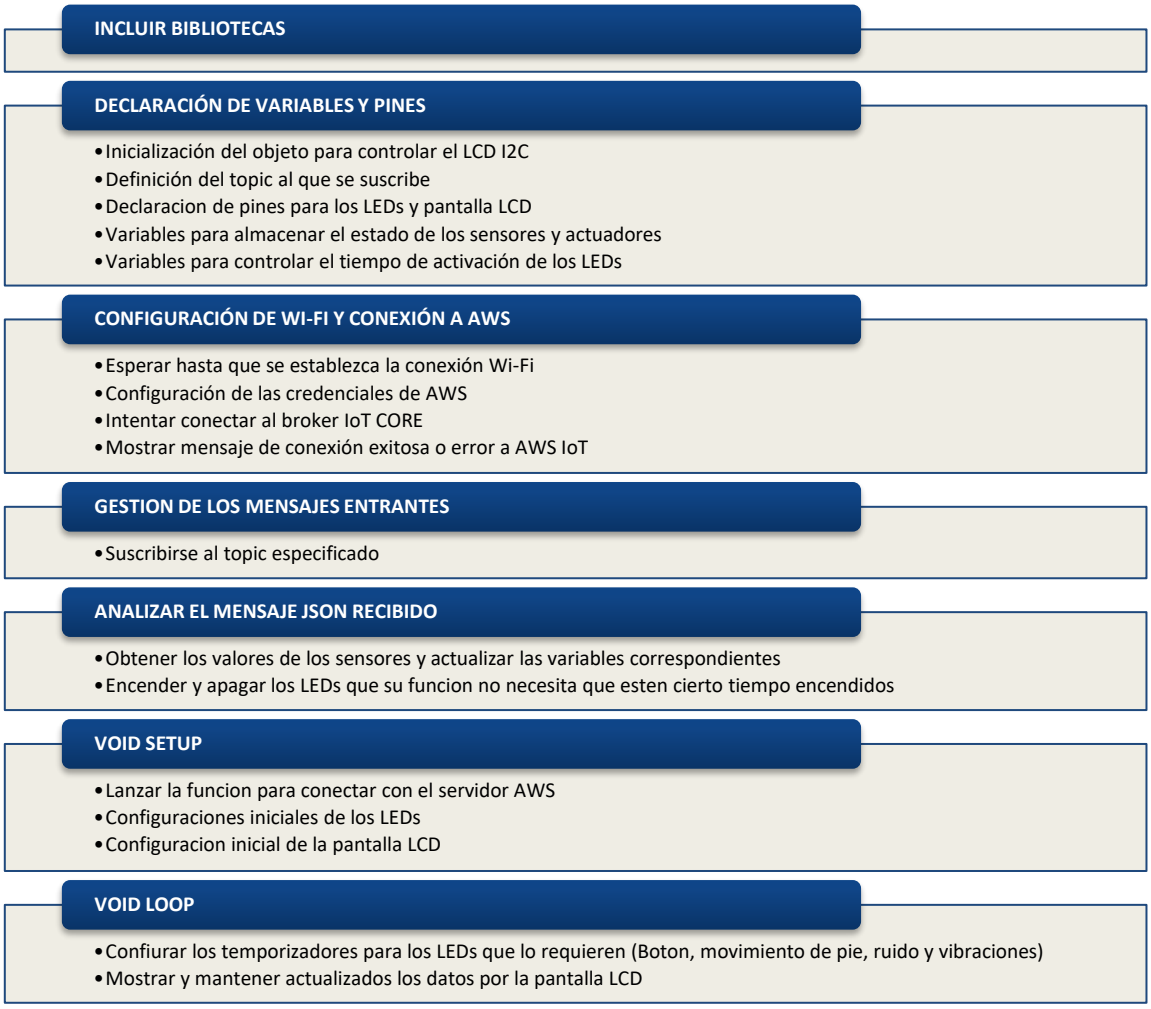

Figura 19. Pseudocódigo para controlar los sensores

5.3 Pseudocódigo encargado de regular la temperatura de la habitación.

En el siguiente datagrama se va a describir el pseudocódigo que describe el programa creado mediante Arduino para hacer que el tercer microcontrolador recoja los datos de la nube de AWS y sea capaz gracias a un relé, de regular la temperatura de la habitación con ayuda de aparatos adicionales.

Figura 20. Pseudocódigo para el control del relé

Capítulo 6. Montaje de los circuitos

El sistema IoT de este proyecto está formado por tres microcontroladores interconectados entre ellos, a cada uno de ellos se le conecta un cable de alimentación y una serie de sensores/actuadores para que cumplan su función. En este capítulo se va a describir como se distribuyen las conexiones en cada uno de los ESP32, según los componentes que llevan acoplados.

6.1 ESP32 encargado de recopilar los datos de los sensores

Este ESP32 tiene conectados 8 sensores que son los encargados de recopilar todos los datos de la habitación donde el paciente tiene la cama. Los sensores que se usan en este proyecto están pensados para una aplicación de IoT y se distribuyen para que su integración con el microcontrolador sea lo más sencilla posible, la gran mayoría de ellos solo necesitan tres pines de conexión, que son las dos conexiones para su alimentación en continua y un tercer pin para enviar los datos hacia el microcontrolador.

Las únicas dos excepciones que tienen más de tres pines son:

- El sensor KY-037, que es el encargado de detectar el ruido y tiene cuatro pines, pero el cuarto no le daremos utilidad porque es un pin para transmitir datos analógicos al microcontrolador y en este caso será suficiente con recibir un uno o un cero, regulando el nivel para recibir un uno con un potenciómetro a través del pin que emite señales digitales.

- El sensor de ultrasonidos HC-SR04, que tiene la funcionalidad de medir distancia para saber la posición de la cabeza del paciente. En este caso sí que utilizaremos los cuatro pines, ya que este sensor aparte de los pines de alimentación requiere de dos pines más, uno es el pin de “Trigger” que se encarga de enviar la señal para que se mande el pulso de ultrasonido para que se empiece a medir y el otro es el pin de “Echo” que es por donde se mandan los datos hacia el microcontrolador de la distancia medida.

Una vez que ya enumerados la cantidad de pines que hay que conectar al microcontrolador, se va a explicar cómo se han distribuido las conexiones. Dentro de las conexiones realizadas las podemos dividir en los tres siguientes grupos:

- Pines de tierra (GND): hay 8 conexiones a tierra, una por cada sensor y se dividen dentro de la placa de pruebas en dos sectores, uno en la parte superior del circuito y otro en la inferior. La elección de dividir las conexiones en dos partes es simplemente para que se haga de una forma más ordenada y cómoda.

- Pines de suministro de voltaje (V5 y 3V3): cada sensor irá conectado a un pin de alimentación según los requerimientos del fabricante, gracias a que todos se alimentan o con 3,3 voltios o con 5 voltios, se pueden alimentar directamente del microcontrolador para su óptimo funcionamiento. En el caso de este circuito solo hemos conectado al voltaje de 3,3V el pulsador y la sonda para medir la temperatura, los demás sensores irán conectados a un voltaje de 5V.

- Pines de datos: para transmitir datos en el diseño de este sistema se utilizan los pines GPIO del microcontrolador, que son pines digitales tanto de entrada como de salida. Se selecciona un total de 9 pines GPIO ya que todos los sensores necesitan transmitir los datos captados hacia el microcontrolador y el sensor ultrasonidos necesita adicionalmente, recibir los datos por su pin de disparo para configurar cuando debe funcionar. Se configurarán 8 pines como entrada y el de disparo como salida. En la siguiente tabla se enumeran el número de pin GPIO en el microcontrolador correspondiente a cada sensor.

Asignación pines GPIO a cada sensor	
SR602	GPIO 14
SW420	GPIO 5
Pulsador	GPIO 4
DS18B20	GPIO 23
LDR5528	GPIO 34
DHT11	GPIO 15
KY-037	GPIO 19
HC-SR04	Pin de disparo GPIO 18
	Pin de eco GPIO 35

Tabla 11. Enumeración de los pines de datos

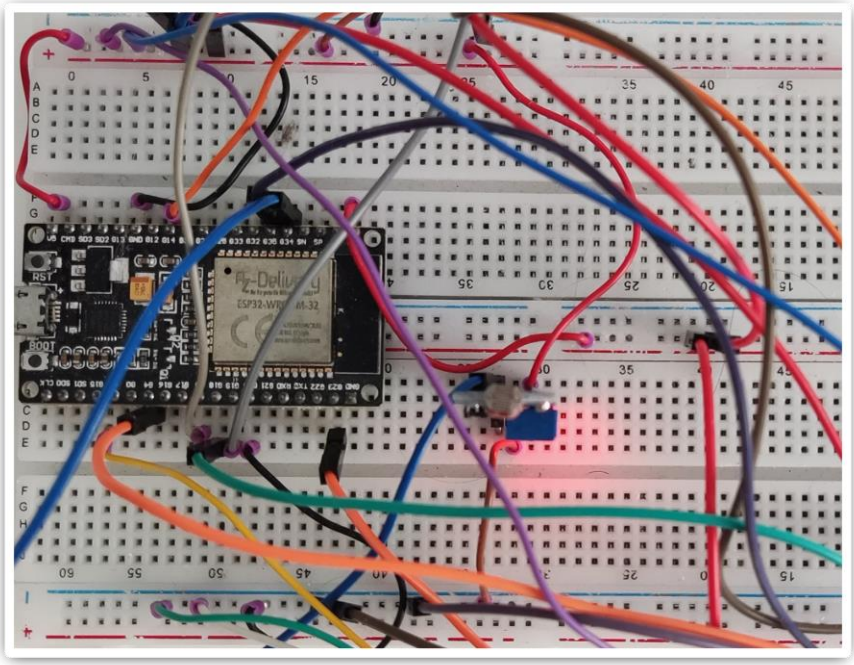

Figura 21. Conexiones al microcontrolador sobre placa de pruebas

6.2 ESP32 encargado de ser la interfaz de usuario

Para crear una interfaz de usuario que sea intuitiva para la persona responsable del enfermo se usa un ESP32, encargado mostrar los datos por un monitor LCD y de encender un led para notificar sobre una alerta de una de las funcionalidades desarrolladas en este proyecto, excepto la de regular la temperatura que se hará directamente con otro microcontrolador y un relé.

Para conectar los diodos LEDs y controlar su encendido, se conecta el ánodo de cada uno de ellos a un pin GPIO configurado como salida para que suministre 3,3V, este voltaje es el que necesitarán para funcionar y dar la señal luminosa. Los cátodos de todos los leds irán conectados en paralelo a un pin GND del microcontrolador y con eso será suficiente para programar las señales lumínicas ya que a través de los pines GPIO se enviarán las señales de encendido.

La pantalla LCD que será de un tamaño de matriz de dieciséis columnas y dos filas, donde cada carácter está compuesto por 5x8 píxeles. Esta pantalla se conecta al microcontrolador a través de un controlador para reducir el número de pines de conexión a cuatro y transmitir los datos más fácilmente a la pantalla gracias a la librería correspondiente dentro del código. Entre estos pines se queda el de alimentación que irá conectado al pin de 5V del microcontrolador, el GND que irá conectado al de tierra del microcontrolador para completar la alimentación y, finalmente, los pines SCL y SDA que son los comunes para transferir datos en una comunicación I2C que es el mecanismo de transferencia que incorpora el controlador.

El pin SCL se encarga de sincronizar la transferencia de datos hacia la pantalla emitiendo una señal de reloj y el pin SDA es la línea de datos serial por donde se envían hacia la pantalla, ambos pines llevan asignado un pin GPIO de salida del microcontrolador. En la siguiente tabla se muestra el resumen de todos los pines de datos GPIO y sus conexiones para esta interfaz de usuario.

Asignación pines GPIO a cada actuador	
LED ayuda	GPIO 13
LED luz	GPIO 14
LED paciente incorporado	GPIO 15
LED ruido	GPIO 16
LED golpes	GPIO 5
LED incontinencias	GPIO 18
LED paciente andando	GPIO 23
Pantalla LCD	SCL = GPIO 22
	SDA = GPIO 21

Tabla 12. Asignación de los pines de datos

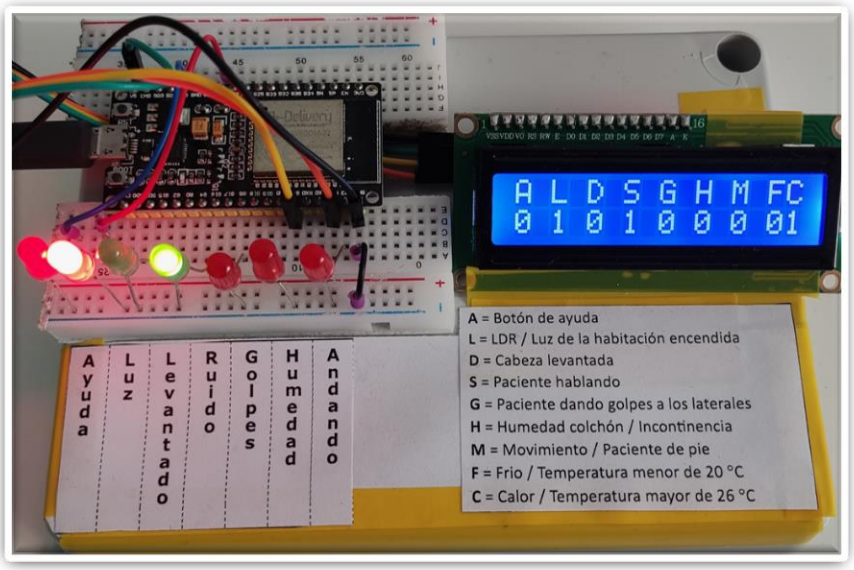

Figura 22. Circuito final de la interfaz de usuario

6.3 ESP32 encargado de regular la temperatura de la habitación

Para el dispositivo encargado de controlar la temperatura, se hace una configuración sencilla, donde simplemente se utiliza un microcontrolador ESP32, para controlar un relé capaz de controlar la tensión en dos enchufes. Uno de los enchufes va a ser el encargado de activarse y desactivarse con el frío en invierno, el otro enchufe va a estar destinado a todo lo contrario, controlar la temperatura en verano activándose o desactivándose según valor de temperatura.

El montaje de este circuito empieza por conectar los dos pines de alimentación del relé al ESP32 y un pin de control para activar/desactivar cada uno de los enchufes. Una vez conectado el relé, solo habría que conectar las dos tomas hembra de enchufe y la toma macho para alimentarlas, esto se hace conectando en paralelo uno de los cables de alimentación hacia los enchufes y el otro pasa por el relé hacia los enchufes hembra para que pueda cortar la alimentación de estos.

El relé esta alimentado por el pin de 5V del microcontrolador y la tierra está conectada a uno de los pines GND, los dos pines de control son el GPIO19 y el GPIO21.

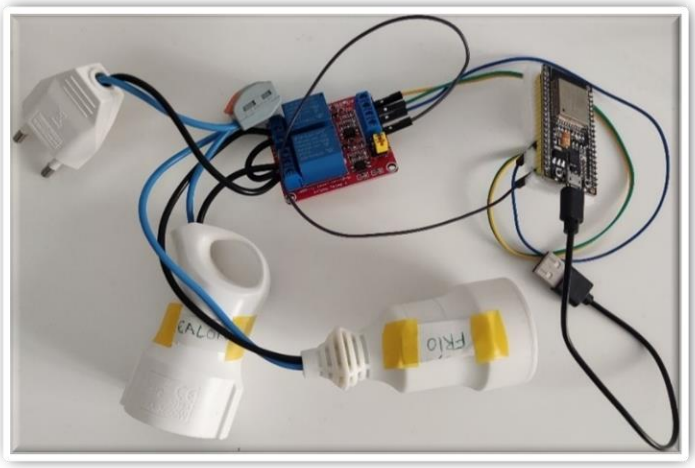

Figura 23. Dispositivo encargado de regular la temperatura de la habitación

En el montaje se obvia la conexión a tierra para los enchufes, ya que las pruebas se realizan conectando una bombilla, la cual no necesita tierra para funcionar de forma segura, como se aprecia en la siguiente figura. Se aprecia que está encendida al estar conectada al enchufe correspondiente a calor y en la pantalla LCD se ve que el valor correspondiente a temperatura alta es uno. Al cambiar la bombilla al enchufe correspondiente a regular la temperatura cuando la misma es baja, se ve apagada ya que la temperatura está por encima de la temperatura referencia necesaria para encender ese enchufe.

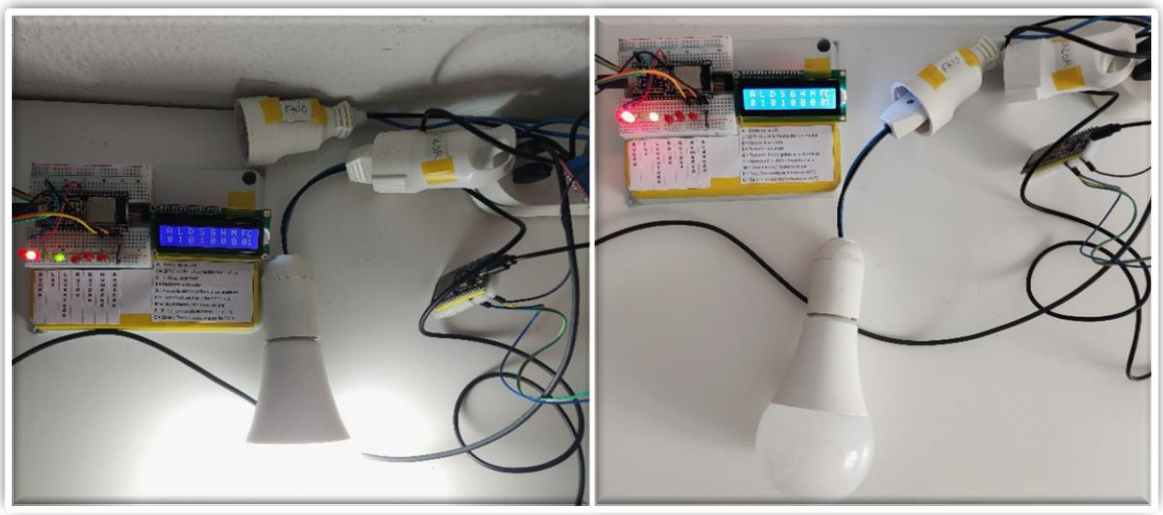

Figura 24. Funcionamiento de los enchufes encargados de regular la temperatura.

Capítulo 7. Funcionamiento general del sistema

En este capítulo se pretende recopilar todo lo explicado durante este informe, para explicar el funcionamiento del sistema IoT completo de manera que se entienda como manejarlo. A continuación, se enumera en orden los pasos a seguir hasta poder utilizar el sistema:

I. **Conectar los microcontroladores con sus sensores/actuadores:** la primera fase para conseguir los resultados del sistema es montar todo el hardware, donde habrá que tener en cuenta los siguientes puntos:

   a. **El tamaño y la posición de la cama:** Según la posición de la cama respecto a las paredes, la posición de los sensores puede cambiar y el ancho de esta también puede interferir en tener que regular ciertos sensores para que funcionen de la forma correcta.

   b. **Mobiliario cercano a la cama:** El mobiliario cerca de la cama es importante ya que sería adecuado que el microcontrolador este sobre una zona segura y en una posición cercana a la cama para reducir el tamaño del circuito.

   c. **Enchufes disponibles para alimentación:** Solo es dependiente de un enchufe de alimentación el relé para regular la temperatura, pero siempre que cada microcontrolador pueda ser alimentado directamente de la red eléctrica será útil para no tener que usar una batería portátil que se tenga que recargar.

II. **Programar el software de cada microcontrolador:** una vez montado y colocado en cada posición cada microcontrolador, será el momento de cargar los programas con ayuda del IDE de Arduino, para que se puedan conectar al bróker de AWS e interaccionar entre ellos. En importante que este paso vaya después del anterior, por si hay que ajustar algún parámetro del código para medir correctamente los valores de los sensores y configurar la red WiFi adecuadamente, para que funcione en el lugar que se instala el sistema.

III. **Comprobar todos los sensores/actuadores:** este paso se centra en que todos los sensores midan adecuadamente, simulando todas las propuestas a monitorizar por el sistema, que los enchufes para la temperatura se enciendan y apaguen en su momento correspondiente, que las siete señales luminosas se enciendan y que la pantalla marque los valores adecuados, durante las ocho posibles situaciones enumeradas a continuación:

   a. Testear que el pulsador de ayuda marca correctamente sus dos estados.
   b. Ajustar el sensor de vibraciones para que detecte golpes en la pared/cama y comprobar que se envían los datos correctamente.
   c. Ajustar el potenciómetro del micrófono para que detecte la voz del paciente al hablar.
   d. Comprobar que el sensor de ultrasonidos tiene la distancia configurada perfecta para medir la cabeza del paciente en cualquier parte de la almohada, detectar cuando la cabeza está apoyada y cuando no.
   e. Ajustar el potenciómetro del LDR para que cambie de estado con cualquier lampara de la habitación durante la noche.

f. Colocar el sensor de humedad sobre la cama y comprobar con una toalla húmeda, por ejemplo, que cambia de estado cuando se la coloca sobre él.
g. Probar que la sonda de temperatura envía correctamente los datos en sus tres posibles estados, temperatura baja, media y alta. También habrá que comprobar que los enchufes encargados de controlar la temperatura se activen correctamente en cada uno de los estados.
h. Comprobar que el sensor PIR detecta el movimiento de una persona que está de pie por la habitación moviéndose.

Para todas estas situaciones se comprueba que se envían los datos correctamente al servidor AWS, visualizando en la pantalla LCD de la interfaz de usuario los valores adecuados en cada momento, si los diodos LEDs se encienden y si los enchufes cambian de estado con la temperatura. En la siguiente tabla se muestran todos los datos que se pueden visualizar entre las diferentes combinaciones de los estados en la interfaz de usuario.

Sensor	**Estado**	**Valor en pantalla LCD**	**LED**
Pulsador	Pulsado	A = 1	Encendido
Pulsador	No pulsado	A = 0	Apagado
SW420	Detecta golpes	G = 1	Encendido
SW420	No detecta golpes	G = 0	Apagado
KY-037	Detecta ruido	S = 1	Encendido
KY-037	No detecta ruido	S = 0	Apagado
HC-SR04	Cabeza detectada	D = 0	Apagado
HC-SR04	No se detecta nada	D = 1	Encendido
LDR5528	Detecta luz	L = 1	Encendido
LDR5528	Hay oscuridad	L = 0	Apagado
DHT11	Detecta un porcentaje de humedad alto	H = 1	Encendido
DHT11	El porcentaje de humedad es bajo	H = 0	Apagado
SR602	Detecta movimiento	M = 1	Encendido
SR602	No detecta movimiento	M = 0	Apagado
Sensor	**Estado**	**Valor en pantalla LCD**	**Enchufe**
DS18B20	Temperatura alta	F = 0, C = 1	Enchufe de calor activo y de frio apagado
DS18B20	Temperatura óptima	F = 0, C = 0	Ambos enchufes apagados
DS18B20	Temperatura baja	F = 1, C = 0	Enchufe de frio activo y de calor apagados

Tabla 13. Tabla de control para la interfaz de usuario.

Si estas tres fases se logran hacer con éxito el sistema IoT, ya estará montado y listo para funcionar de forma ininterrumpida, solo habrá que mantener alimentados los tres microcontroladores, ya sea a través un transformador conectado a la red eléctrica o con baterías portátiles. En la siguiente figura se muestra el sistema IoT completamente conectado y funcionando.

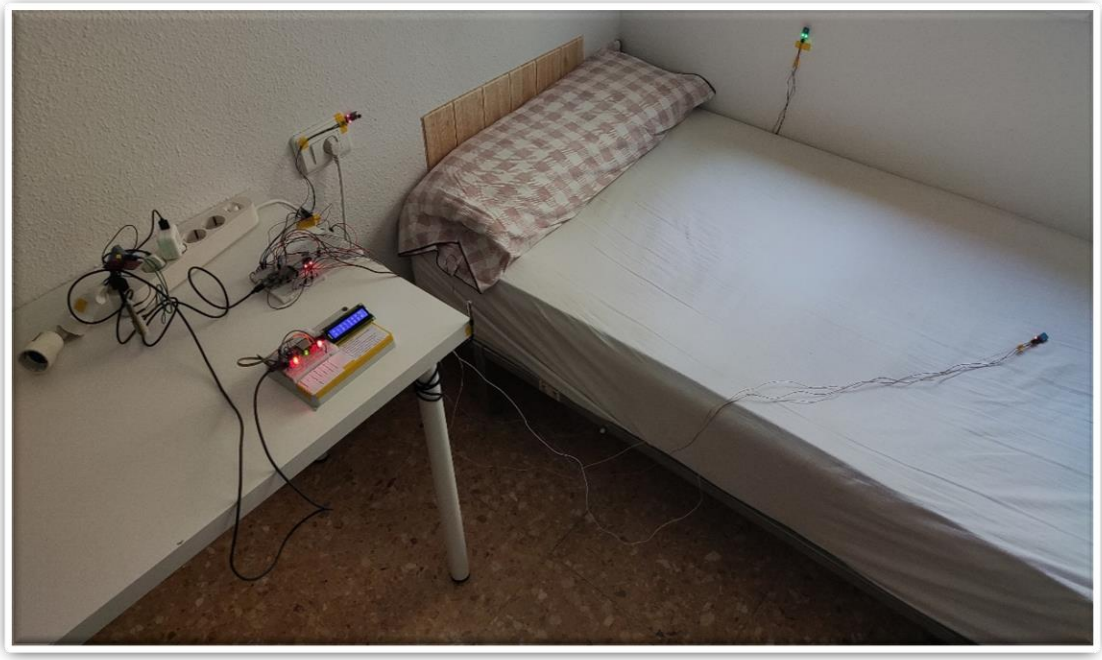

Figura 25. Sistema montado para monitorizar a un paciente.

Capítulo 8. Presupuesto

En este capítulo se detalla mediante el siguiente presupuesto el coste de todos los componentes hardware utilizados (todo el software utilizado es de uso gratuito):

Concepto	Distribuidor	Importe
3 x ESP32 NodeMCU Módulo WLAN WiFi	AZ-Delivery	27,99 €
KY-037 Micrófono detección de Sonido	AZ-Delivery	4,49 €
SW420 Módulo Sensor de Vibración	AZ-Delivery	4,49 €
KY-015 Módulo del Sensor de Temperatura	AZ-Delivery	5,99 €
Módulo de botón grande	Aliexpress	0,39 €
Kit de módulo de Sensor de temperatura DS18B20	Aliexpress	1,71 €
Sensor piroeléctrico infrarrojo SR602	Aliexpress	0,67 €
3 x Placa de pruebas	Aliexpress	8,27 €
Cableado para placa de pruebas	Aliexpress	13,42 €
Sensor HC-SR04	Aliexpress	0,74 €
Módulo de sensor fotosensible	Aliexpress	0,47 €
Pantalla LCD1602C Blue	Aliexpress	2,49 €
Gastos de envió	Aliexpress	2,62 €
Diodos LED	Comercio local	3 €
	Total:	**76,74 €**

Tabla 14. Presupuesto del hardware utilizado.

Capítulo 9. Conclusiones

Tras finalizar la implementación del sistema IoT completo, se ha conseguido cumplir con el objetivo del mismo, el cual es explicado en el capítulo uno. Como conclusión se ha podido resolver el problema planteado en una situación real de una persona con demencia senil, esto se ha obtenido mediante un sistema IoT, montando un circuito con componentes económicos interconectados entre sí.

Con el paso de los años y el avance de la tecnología, los sistemas IoT van a ser más explotados en el sector comercial. A través de este proyecto, el cual se centra en mejorar la calidad de vida y el bienestar de las personas, se ha pretendido motivar a que se pueden llegar a automatizar muchos procesos ahorrando esfuerzo. En este caso la automatización implementada en la cama del paciente permite un monitoreo continuo las 24 horas del día, con una respuesta rápida ante situaciones de emergencia.

En lo que respecta a la tecnología, a nivel software queda detallado como implementarlo y todos los elementos utilizados han sido gratuitos, incluso AWS IoT Core para este tipo de utilidades es gratuito por ser pocos dispositivos los interconectados a través de la nube. A nivel hardware ha supuesto un gasto de cerca de 80 euros, coste que se podría reducir en un futuro, comprando todos los componentes a proveedores asiáticos si se pudiese esperar los plazos de entrega.

En conclusión, tras una fase de investigación teórica sobre IoT, el protocolo de comunicación MQTT y la enfermedad de Demencia Senil, se ha logrado desarrollar y montar un prototipo con microcontroladores de bajo coste y bajo consumo, para asegurar que el cuidador va a poder estar tranquilo moviéndose por la vivienda, manteniéndose informado de la situación del paciente, por lo que se ha conseguido el objetivo principal del proyecto. De esta forma he podido profundizar en los conocimientos tratados a lo largo de la carrera de una forma práctica y aplicada a la vida real.

9.1 Ampliaciones futuras.

Como forma de ampliar el trabajo realizado, pudiendo llevarlo a un nivel superior o incluso al sector comercial, se proponen las siguientes líneas de trabajo:

- Desarrollar una aplicación para la interfaz de usuario: la idea podría ser crear una aplicación móvil o web para visualizar los datos de usuario conectándose a AWS.

- Mejorar la interfaz de usuario a nivel dispositivo hardware: en esta idea se propone mejorar la interfaz de usuario, un ejemplo podría ser que disponga de una pantalla de mayor tamaño donde se vean los datos con más detalles y no tener que mostrar las señales lumínicas.

- Desarrollar un prototipo para desarrollarlo a nivel comercial: este caso sería desarrollar un sistema, donde no se usen placas de desarrollo y los componentes fuesen directamente soldados para que el cliente no se preocupe por la instalación en la vivienda.

Capítulo 10. Bibliografía.

[1] La historia detrás de la internet de las cosas, María Alejandra Medina (2017). Ultimo acceso marzo 2023, https://www.elespectador.com/tecnologia/la-historia-detras-de-la-internet-de-las-cosas-article-716678/

[2] Las grandes estadísticas del Internet de las Cosas (IoT), IoT World Online (2020). Ultimo acceso marzo 2023, https://www.iotworldonline.es/las-grandes-estadisticas-del-internet-de-las-cosas-iot/

[3] Protocolos de redes inalámbricas del IoT, Farnell (2022). Ultimo acceso marzo 2023, https://es.farnell.com/iot-wireless-network-protocols?ICID=I-CT-TECH-WIR-FC-JAN22-TC-000020

[4] ¿Qué es MQTT? Su importancia como protocolo IoT, Luis Llamas (2019). Ultimo acceso mayo de 2023, https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/

[5] MQTT: The Standard for IoT Messaging (2022). Ultimo acceso junio de 2023, https://mqtt.org/

[6] ¿Qué es MQTT? El protocolo de comunicación para IoT, Solectro (2022). Ultimo acceso abril 2023,https://solectroshop.com/es/blog/que-es-mqtt-el-protocolo-de-comunicacion-para-iot-n117

[7] Datasheet del microcontrolador ESP32, en la web del fabricante (2023). Ultimo acceso marzo de 2023, https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[8] Datasheet del módulo de desarrollo ESP32 NodeMCU Module, en la web del distribuidor. Ultimo acceso junio de 2023, https://cdn.shopify.com/s/files/1/1509/1638/files/ESP_-_32_NodeMCU_Developmentboard_Datenblatt_AZ-Delivery_Vertriebs_GmbH_10f68f6c-a9bb-49c6-a825-07979441739f.pdf?v=1598356497

[9] Documentación de AWS IoT Core (2023). Ultimo acceso junio de 2023, https://docs.aws.amazon.com/es_es/iot/?icmpid=docs_homepage_iot

[10] Demencia senil: fases, síntomas y tratamiento, Allegra (2020). Ultimo acceso junio de 2023, https://allegra.cat/es/demencia-senil-fases/

[11] Universidad Politécnica de Valencia (2022-2023). Apuntes de la asignatura Diseño de Servicios Telemáticos.

[12] Universidad Politécnica de Valencia (2022-2023). Apuntes de la asignatura Servicios y Sistemas de Transmisión por Radio.