MoniGas: Prototipo de monitoreo remoto de gases para el hogar

Edilberto Chara Mejia

Asesor

Jairo Luis Gutiérrez Torres

Universidad Nacional Abierta y a Distancia UNAD

Escuela de Ciencias Básicas, Tecnología e Ingeniería ECBTI

Ingeniería Electrónica

2025

Agradecimiento

Agradezco a Dios por darme la fortaleza y sabiduría para alcanzar este logro, a mi familia por su amor y apoyo incondicional, y a mis amigos por su compañía y palabras de aliento durante todo este proceso. Asimismo, agradezco a los tutores por compartir su conocimiento y brindarme su valiosa orientación, siendo todos ellos fundamentales para alcanzar este objetivo académico.

Resumen

El proyecto de monitoreo remoto de calidad de aire y gases peligrosos para el hogar se basa en la detección temprana de situaciones de riesgo que puedan poner en peligro la seguridad y salud de los habitantes de una vivienda. Para ello, se utilizarán tarjetas de desarrollo electrónico integrados, tecnología IoT, API para el envío de notificaciones a través de WhatsApp, sensores de temperatura, humedad, calidad de aire, humo, gases tóxicos e inflamables para monitorear remotamente el ambiente del hogar y detectar situaciones de riesgo.

Palabras clave: Monitoreo remoto, Calidad del aire, Gases peligrosos, Seguridad en el hogar, Tecnología IoT, Notificaciones por WhatsApp.

Abstract

The remote monitoring project for air quality and gases for the home is based on the early detection of risk situations that could endanger the safety and health of the inhabitants of a residence. For this purpose, integrated electronic development boards, IoT technology, an API for sending notifications via WhatsApp, sensors for temperature, humidity, air quality, smoke, toxic, and flammable gases will be used to remotely monitor the home environment and detect risk situations.

Keywords: Remote monitoring, Air quality, gases, Home safety, IoT technology, WhatsApp notifications.

Tabla de Contenido

Lista de Tablas

Lista de Figuras

Lista de Apéndices

Introducción

La seguridad y protección en nuestros hogares son aspectos fundamentales que todos deseamos garantizar. En este contexto, la implementación de un prototipo de monitoreo de incendio y gases se convierte en una solución clave para prevenir riesgos y asegurar un entorno seguro. En este proyecto, se presenta la experiencia de implementar exitosamente dicho sistema utilizando la plataforma Blynk IoT, la tarjeta ESP32 con Wifi integrado, La API CallMeBot para el envío de notificaciones por WhatsApp y los sensores de gas MQ-5, MQ-135 y sensor de temperatura y humedad DHT22.

La plataforma Blynk IoT demostró ser una elección acertada, permitiendo una interacción intuitiva y un acceso remoto a las variables del sistema en tiempo real. Asimismo, la tarjeta ESP32, con su conectividad Wifi, fue fundamental para lograr la comunicación efectiva entre los dispositivos y la plataforma IoT. Los sensores de gas MQ-5, MQ-135 y DHT22, por su parte, brindaron una monitorización precisa de la temperatura, humedad y concentraciones de gases inflamables y contaminantes en el aire del hogar.

A través de este informe, se presentan los resultados obtenidos, destacando la eficacia en la detección y prevención de incendios y gases peligrosos, la accesibilidad y facilidad de uso del sistema, así como la integración de múltiples sensores para una monitorización completa del ambiente en el hogar. En conjunto, estos elementos ofrecen a los usuarios una solución confiable y efectiva para mejorar la seguridad y tranquilidad en sus hogares.

Planteamiento del Problema

Durante las visitas a familiares y amigos se ha notado que la gran mayoría de los hogares y, en general, en hogares de Colombia, no cuentan con un sistema de detección de incendios ni otro sistema capaz de detectar concentraciones peligrosas de gas natural, propano, humo o, cualquier otra contaminación del aire doméstico, que según la Organización Mundial de la Salud "La contaminación del aire doméstico causa enfermedades no transmisibles, en particular accidente cerebrovascular, cardiopatía isquémica, neumopatía obstructiva crónica y cáncer de pulmón".

A pesar de la existencia de sistemas de detección de incendios y calidad del aire, muchos de estos son costosos, de difícil acceso o no ofrecen un monitoreo en tiempo real ni alertas inmediatas que permitan actuar con prontitud ante emergencias.

En este contexto, la falta de soluciones integrales que combinen la detección de variables críticas como temperatura, humedad y concentraciones de gases inflamables y contaminantes, junto con la capacidad de enviar alertas tanto local como remotamente, representa una brecha significativa en la protección y monitoreo ambiental de los hogares. Además, la implementación de tecnologías IoT accesibles y eficientes no ha sido suficientemente explorada para atender estas necesidades, especialmente en contextos donde los recursos son limitados.

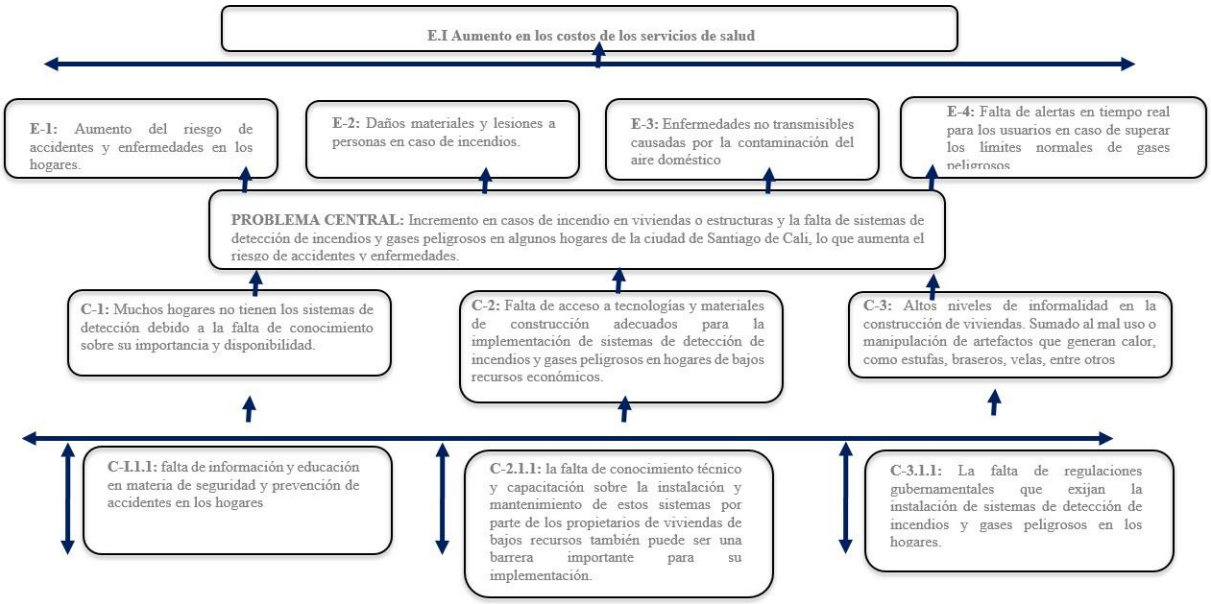
Dado este panorama, surge la necesidad de implementar un prototipo que combine sensores de bajo costo y tecnología IoT para monitorear en tiempo real la calidad del aire y detectar posibles riesgos de incendio, alertando a los usuarios de forma eficiente mediante notificaciones locales (pantalla LCD, alarma sonora) y remotas (Blynk IoT y WhatsApp).

Árbol causa – Efecto del Problema

En la figura 1, se muestra el árbol de causa y efecto del problema identificado, donde se identifican y se exponen las diferentes causas y sus posibles efectos en relación a los problemas generados por el incremento en casos de incendio en viviendas o estructuras y en la falta de sistemas de detección de incendios y gases peligrosos en los hogares, lo cual aumenta el riesgo de accidentes y enfermedades en los hogares de la comunidad de Santiago de Cali, de acuerdo con las cifras estadísticas del Cuerpo Oficial de Bomberos de Santiago de Cali, donde se observa un gran número de viviendas afectadas por este tipo de emergencias, llegando aproximadamente a los 500 casos en el último año y lo corrido de este.

Figura 1

Árbol de Problema



Ante la problemática expresada anteriormente, nace la siguiente pregunta de investigación: ¿Cómo diseñar e implementar un sistema basado en tecnología IoT que permita el monitoreo en tiempo real de la calidad del aire y la detección de gases inflamables para el hogar, enviando alertas locales y remotas para una respuesta oportuna ante posibles emergencias?

Justificación

La implementación de un sistema prototipo que permita la detección de incendios y concentraciones peligrosas de gases en el hogar es una necesidad urgente debido a los riesgos crecientes de accidentes y enfermedades asociados a estas condiciones. Este proyecto tiene un impacto social directo al ofrecer una solución accesible que mejora la seguridad y calidad de vida en los hogares colombianos, donde la informalidad en las construcciones y la falta de sistemas de monitoreo son factores recurrentes.

Al identificar esta necesidad y realizar un análisis sobre los incendios en viviendas y estructuras en la ciudad de Cali, se encontró que, según las estadísticas del Cuerpo Oficial de Bomberos de Santiago de Cali, se produjeron 421 incendios en el año 2021, 435 en el año 2022 y 462 en el año 2023. Esta problemática ha mostrado un incremento constante año tras año, reflejando un aumento del 9% en la ocurrencia de incendios durante el período analizado. Este aumento se traduce en un promedio de 97 casos adicionales en comparación con el año anterior, evidenciando la importancia de implementar soluciones efectivas de detección y prevención de incendios en los hogares y estructuras de la ciudad.

El diseño de este prototipo utiliza tecnologías IoT, como la tarjeta ESP32 y sensores como el MQ-135 y MQ-5, para proporcionar monitoreo en tiempo real y envío de alertas mediante plataformas como Blynk IoT y WhatsApp. Este enfoque permite una respuesta oportuna ante emergencias, minimizando riesgos de seguridad y reduciendo potenciales daños materiales y humanos.

Además de prevenir accidentes, el proyecto fomenta el desarrollo académico y tecnológico en el ámbito de la ingeniería electrónica. Los estudiantes y profesionales pueden aplicar sus conocimientos para crear soluciones que beneficien a la sociedad, reforzando su

compromiso con el bienestar colectivo y promoviendo el uso responsable de la tecnología. Finalmente, el sistema no solo beneficia a los usuarios domésticos, sino que también facilita una mejor respuesta por parte de bomberos y servicios de emergencia, al proporcionar información oportuna sobre las condiciones de riesgo en el hogar.

En resumen, este proyecto no solo tiene el potencial de salvar vidas y reducir costos asociados a incendios y contaminación del aire, sino que también contribuye al desarrollo tecnológico del país al abordar un problema crítico con soluciones innovadoras y accesibles.

Objetivos

Objetivo General

Diseñar e implementar un prototipo de monitoreo de calidad de aire y detección de gases inflamables para el hogar, que opere de manera automática y envíe alertas a los usuarios a través de una interfaz remota IoT y por WhatsApp.

Objetivos Específicos

Diseñar y configurar el prototipo de monitoreo utilizando la Tarjeta ESP32 y los sensores MQ-5, MQ-135 y DHT22 para monitorear cambios en la temperatura, humedad, calidad de aire y la presencia de gases inflamables en el ambiente del hogar.

Programar e implementar la comunicación entre los componentes del sistema y la Plataforma Blynk IoT, permitiendo el acceso remoto y el monitoreo en tiempo real de los datos recopilados por los sensores.

Configurar la API de CallMeBot para habilitar el envío de notificaciones por WhatsApp, garantizando una comunicación efectiva y oportuna con los usuarios en situaciones de emergencia.

Realizar pruebas al sistema, así como la correcta transmisión de datos hacia la plataforma Blynk IoT, asegurando su correcto funcionamiento.

Marco Referencial

Marco

Conceptual

Incendio

Un incendio es una reacción química de combustión que se caracteriza por la liberación de calor, humo, gases y llamas. Las causas más comunes de los incendios en el hogar son las fallas eléctricas, la cocina, el tabaquismo, las velas, entre otros (Cuerpo Oficial de Bomberos de Cali, 2023).

El proyecto de detección de incendio y gases peligrosos para el hogar se basa en la detección temprana de situaciones de riesgo que puedan poner en peligro la seguridad y salud de los habitantes de una vivienda. Para ello, se utilizarán tarjetas de desarrollo electrónico integrados, tecnología IoT, sensores de temperatura, humedad, calidad de aire, humo, gases tóxicos e inflamables para monitorear remotamente el ambiente del hogar y detectar situaciones de riesgo.

Tarjetas de Desarrollo Electrónico Integrado

Las tarjetas de desarrollo electrónico integrado son placas o dispositivos que integran componentes electrónicos esenciales, como microcontroladores, sensores, actuadores y puertos de comunicación, en un solo paquete. Estas tarjetas proporcionan una plataforma de desarrollo para prototipado rápido, pruebas y proyectos de electrónica.

Las tarjetas de desarrollo electrónico integrado ofrecen una serie de ventajas, como la facilidad de uso, la compatibilidad con diversos entornos de programación y la posibilidad de ampliación mediante la conexión de otros módulos y componentes. Estas tarjetas suelen estar diseñadas para ser programadas y configuradas a través de software, lo que permite a los

desarrolladores escribir código y controlar los componentes integrados para realizar tareas específicas.

Algunas de las tarjetas de desarrollo electrónico integrado más populares incluyen Arduino, Raspberry Pi, ChipKIT. y ESP32. Estas tarjetas han ganado gran popularidad debido a su accesibilidad, amplia comunidad de usuarios y la disponibilidad de bibliotecas y recursos de programación que facilitan el desarrollo de proyectos electrónicos.

Arduino

Es una plataforma de desarrollo electrónico de código abierto basada en hardware y software de fácil uso para cualquier persona interesada en desarrollar proyectos didácticos o de uso general. Esta plataforma integra circuitos periféricos y puertos en una sola placa. La tecnología Arduino le permite capturar información del entorno al recibir entradas analógicas y/o digitales de varios sensores diferentes e influir en su entorno mediante el control de dispositivos. El microcontrolador de la placa se programa utilizando un entorno de desarrollo integrado - IDE (Arduino, 2023).

El mismo autor dice que los proyectos de Arduino pueden ser independientes o comunicarse con software IDE que se ejecuta en sistemas operativos Windows, Macintosh OSX y Linux. Los módulos pueden ensamblarse a mano o comprarse ensamblados; el software se puede descargar de forma gratuita desde el mismo servidor web, lo que permite conocer todos los dispositivos Arduino y brindar información y soporte para los módulos existentes. Además, ofrece ciertas ventajas a profesores, alumnos y otros interesados con respecto a economía, multiplaforma, entorno de programación simple y claro y software extensible y de código abierto. Lo que significa que se puede usar libremente para desarrollo de proyectos.

En la tabla 1, se visualiza la comparación de las características básicas de las principales tarjetas Arduino existentes en el mercado: como Arduino uno, Arduino mega, Arduino nano, entre otros. Permitiendo establecer puntos de comparación, que conduzca establecer la plataforma adecuada para desarrollo del proyecto.

Tabla 1

Características de las Principales Tarjetas Arduino Del Mercado

Tarjeta	Voltaje De Operación	Voltaje De Alimentación	Flash [KB]	SRAM [KB]	I/O digitales /PWM	Pines	Voltaje De Operación	Voltaje De Alimentación	Flash [KB]
Uno	5v	7-12v	32	2	14/6	6/0	1	Excelente	
Pro	5v	5-12v	32	2	14/6	6/0	1	Excelente	Requiere FTDI para programar
Pro Mini	5v	3.35 - 12v	32	2	14/6	6/0	1	N/A	
Leonardo	5v	7-12v	32	2.5	20/7	12/0	1	Decente (diferencias de pines)	USB nativo
Micro	5v	5v	32	2.5	20/7	12/0	1	N/A	Compatible con protoboards
Mega 2560	5v	7-12v	256	8	54/15	16/0	4	Buena (Algunas diferencias de pines)	
Mega ADK	5v	7-12v	256	8	54/15	16/0	4	Buena (Algunas diferencias de pines)	Funciona con ADK (Android)
Due	3.3v	7-12v	512	96	54/12	12/2	4	Mala (Diferencias de pines y voltaje)	El procesador más rápido
Esplora	5v	5v	32	2.5 -	N/A	N/A	-	N/A	Integración nativa con sensores

Nota. Tabla comparativa de las placas Arduino. Adaptado de 5HERTZ ELECTRONICA, 2016, Cual Arduino comprar. [en línea]. < http://5hertz.com/tutoriales/?p=571>

Raspberry Pi

Es un pequeño ordenador de placa única (SBC, por sus siglas en inglés) diseñado para ser utilizado en proyectos de computación y electrónica de bajo costo y bajo consumo de energía. Fue desarrollado por la Fundación Raspberry Pi en el Reino Unido con el objetivo de fomentar la enseñanza de la informática y la programación en las escuelas, pero su facilidad de uso y bajo costo lo han convertido en una herramienta popular para la creación de proyectos de electrónica y computación a nivel personal y profesional (Raspberry Pi, 2023). Uno de los módulos más utilizados es el Raspberry Pi modelo B+.

ChipKIT

Es una plataforma desarrollada por la compañía Digilent en conjunto con Microchip. Cuenta con un microcontrolador de 32 bits PIC32, es totalmente compatible con la plataforma Arduino y usa software de código abierto. Entre los módulos más destacados, de esta plataforma, está el módulo ChipKIT uno32.

Este dispositivo es de fácil uso para el desarrollo de aplicaciones basadas en microcontroladores. Utiliza entorno de desarrollo chipKIT núcleos y Arduino IDE para la compatibilidad en la programación del código, contando con soporte, muy amplio, por medio de ejemplos y tutoriales. Es compatible con muchos módulos de Arduino que pueden funcionar a 3.3V (chipKIT, 2016). Posee las siguientes características: Procesador PIC32MX795F512L, memoria 512K Flash, 128K de RAM, Velocidad de funcionamiento de 80MHz, Controlador USB 2.0 OTG, 10/100 Ethernet MAC, Dos controladores CAN, 83 disponibles líneas de E / S y USB o alimentación externa.

Tarjeta ESP32

La Tarjeta ESP32 es una tarjeta de desarrollo electrónica integrada basada en el chip ESP32. El ESP32 es un microcontrolador de bajo consumo de energía y alta capacidad de procesamiento, diseñado para aplicaciones de Internet de las cosas (IoT) y proyectos de electrónica.

Este dispositivo ofrece una amplia gama de características y funcionalidades, incluyendo conectividad Wifi y Bluetooth, múltiples puertos de entrada/salida (GPIO), interfaces de comunicación como UART, SPI e I2C, conversor analógico-digital (ADC) y conversor digital-analógico (DAC), entre otros.

La Tarjeta ESP32 es muy popular en la comunidad de desarrollo de IoT debido a su versatilidad y rendimiento. Es compatible con diferentes entornos de programación, como el entorno de desarrollo Arduino y el lenguaje de programación MicroPython, lo que facilita su programación y configuración para diversos proyectos.

Entorno de Desarrollo Integrado

El Entorno de Desarrollo integrado (IDE, por sus siglas en ingles). Se refiere a un software que proporciona herramientas y funcionalidades integradas para facilitar el desarrollo de software.

Un IDE generalmente incluye un editor de código, un compilador o intérprete, un depurador y otras características que ayudan a los programadores a escribir, probar y depurar su código de manera más eficiente. Proporciona una interfaz unificada y coherente para el desarrollo de software, lo que simplifica el proceso y mejora la productividad (Arduino Docs, 2023). Uno de los más populares es el Arduino IDE.

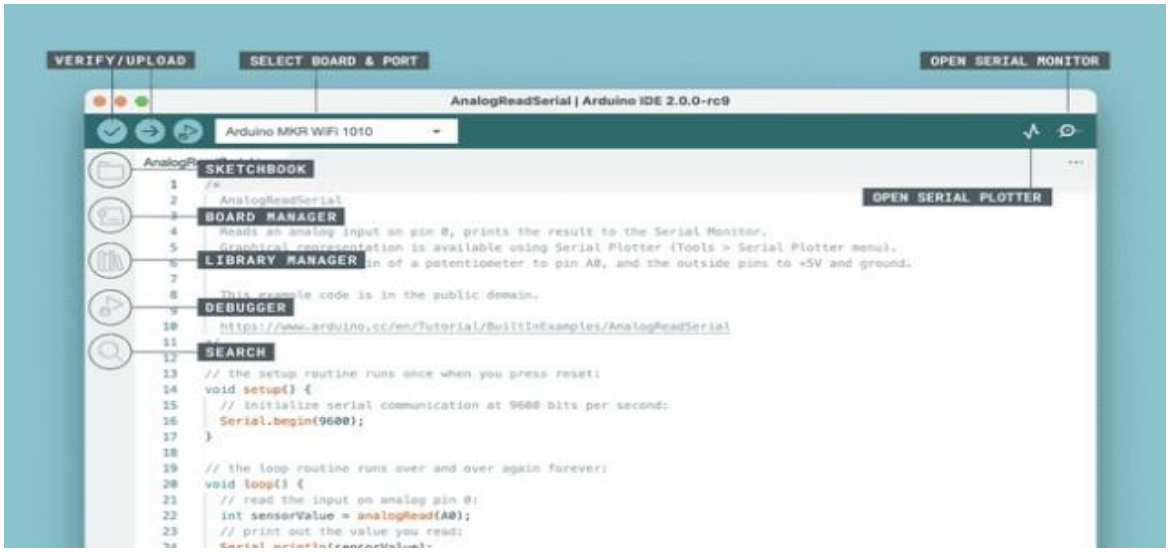
Arduino IDE

El Entorno de Desarrollo Integrado es el encargado de la interacción de la conexión entre el computador y el hardware de Arduino, con el fin de establecer una comunicación entre ellos por medio de configuraciones. Además, se encarga de otras funciones, que se muestra en la figura 2, donde se describen los partes más significativos de la herramienta de programación.

El IDE de Arduino se compone de: un editor de texto, donde se escribe el código del programa; un área de mensajes, en la cual el usuario tiene constancia en todo momento de los procesos que se ejecutan, errores en código, problemas de comunicación y mensajes de interés; una ventana de texto, mediante la que se puede establecer comunicación con el hardware Arduino y viceversa; una barra de herramientas, donde puede acceder a una serie de menús y botones con acceso directo a otras funciones de Arduino (Castro, 2013).

Figura 2

Arduino IDE 2.0



Nota. Área de trabajo de Arduino IDE con las herramientas más utilizadas. Adaptado de docs.arduino.cc, 2023, Getting Started with Arduino IDE 2. https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2

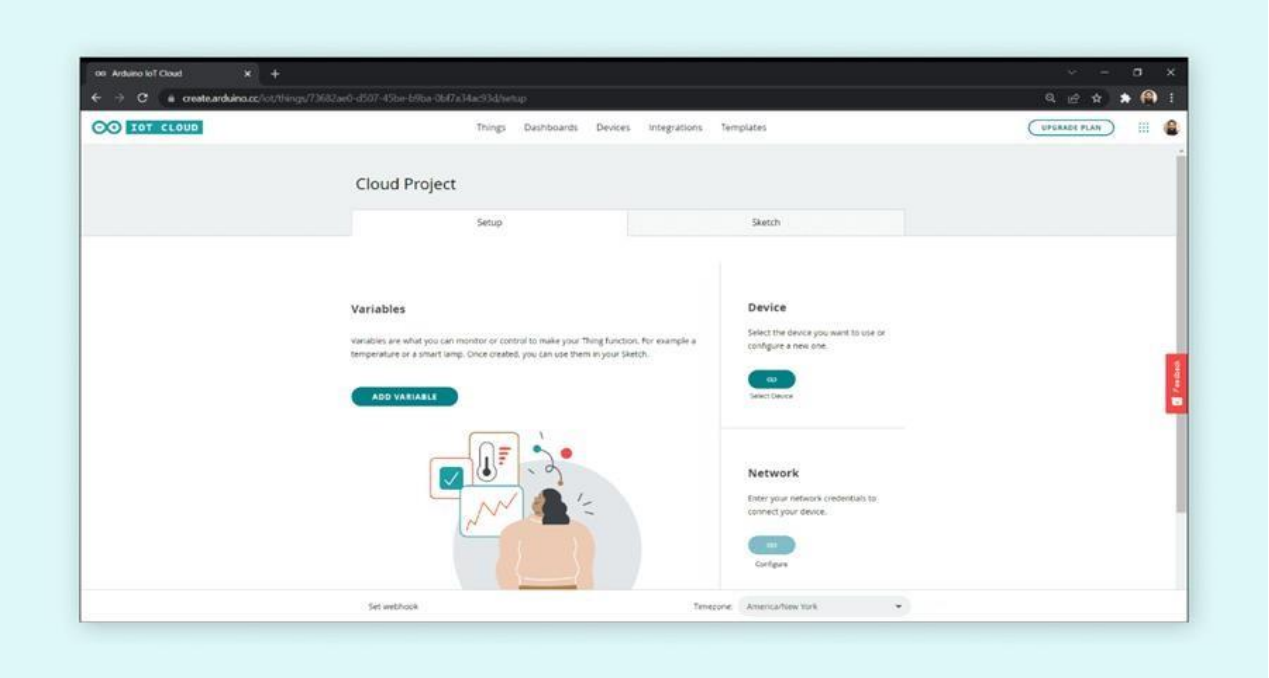
Arduino IoT Cloud

Arduino IoT Cloud es una plataforma en la nube desarrollada por Arduino que permite la conexión y control de dispositivos basados en Arduino a través de Internet (Arduino Docs, 2023). Proporciona una interfaz intuitiva y fácil de usar para monitorear y controlar los dispositivos conectados de forma remota.

Con esta plataforma, los usuarios pueden crear aplicaciones IoT (Internet de las cosas) sin la necesidad de realizar una programación compleja. La plataforma ofrece una amplia gama de características y funcionalidades, como la visualización de datos en tiempo real, la configuración de reglas y notificaciones, y la integración con otros servicios en la nube. En la figura 3, se presenta la plataforma IoT Cloud.

Figura 3

Arduino IoT Cloud



Nota. Adaptado docs.arduino.cc, 2023, Getting Started With the Arduino IoT Cloud. https://docs.arduino.cc/arduino-cloud/getting-started/iot-cloud-getting-started

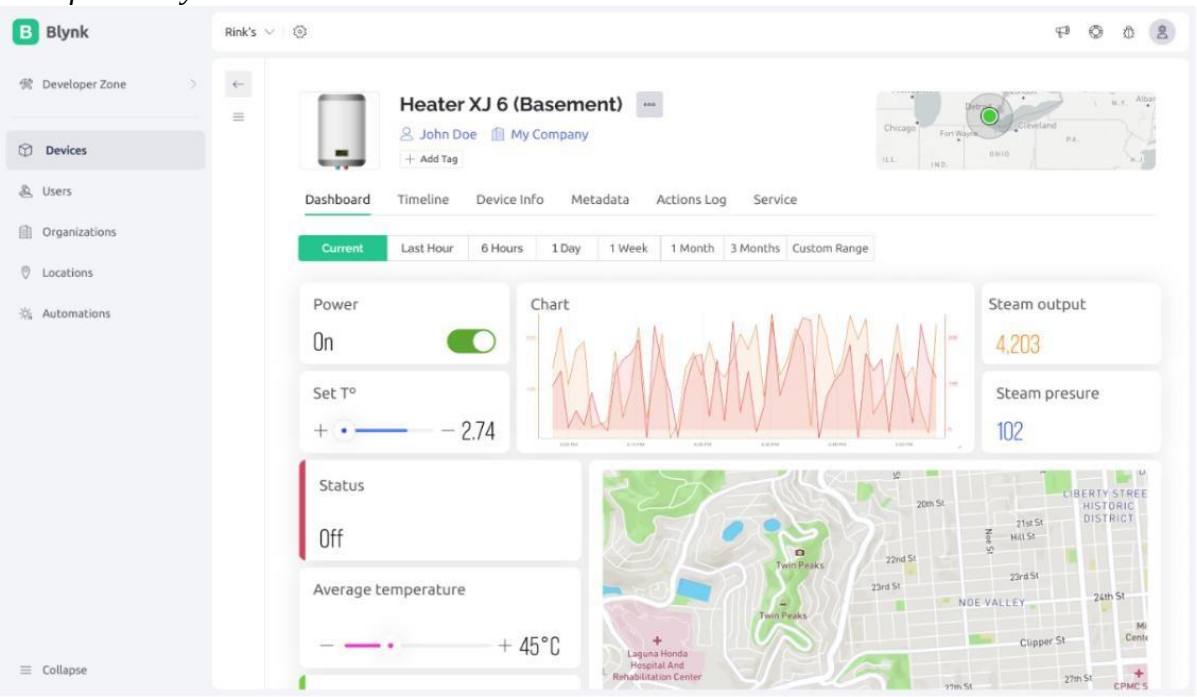
Además, utilizando Arduino IoT Cloud, es posible crear proyectos de IoT personalizados, como sistemas de monitoreo y control para el hogar, sistemas de automatización, dispositivos de medición y muchos otros. La plataforma proporciona una forma conveniente de conectar los dispositivos Arduino u otros dispositivos a Internet y aprovechar las capacidades de la nube para mejorar la funcionalidad y accesibilidad de los proyectos.

Blynk IoT

Blynk IoT es una plataforma de desarrollo que permite crear fácilmente proyectos de Internet de las cosas (IoT) de manera rápida y eficiente (Blynk.Documentation, 2024). Permite, para este caso, conectar dispositivos hardware, como la Tarjeta ESP32 y los sensores MQ-5, MQ-135 y DHT22 a la nube para monitorear y controlar diferentes aspectos del entorno desde cualquier lugar a través de una aplicación móvil o web. Ver figura 4.

Figura 4

Plataforma Blynk IoT



Nota. Adaptado de docs.blynk, 2024, Componentes de la plataforma Blynk IoT. https://docs.blynk.io/en#components-of-the-blynk-iot-platform

Una de las principales características de esta plataforma es su interfaz intuitiva y amigable, que facilita la creación de interfaces personalizadas para visualizar datos en tiempo real, recibir notificaciones y controlar dispositivos de manera remota. Además, ofrece una gran variedad de *widgets* y herramientas que simplifican el proceso de diseño y configuración de proyectos IoT.

Blynk también proporciona funciones avanzadas, como la integración con servicios en la nube, la capacidad de enviar comandos a dispositivos remotamente, y la posibilidad de crear reglas y automatizaciones para responder automáticamente a eventos específicos.

En el contexto del proyecto de detección de incendios y gases peligrosos para el hogar, Blynk IoT se utilizará para establecer la comunicación entre los sensores y la plataforma en la nube, permitiendo a los usuarios monitorear y recibir alertas sobre la calidad del aire, la temperatura y la presencia de gases, así como controlar el sistema de manera remota desde sus dispositivos móviles o computadoras.

Protocolo MQTT

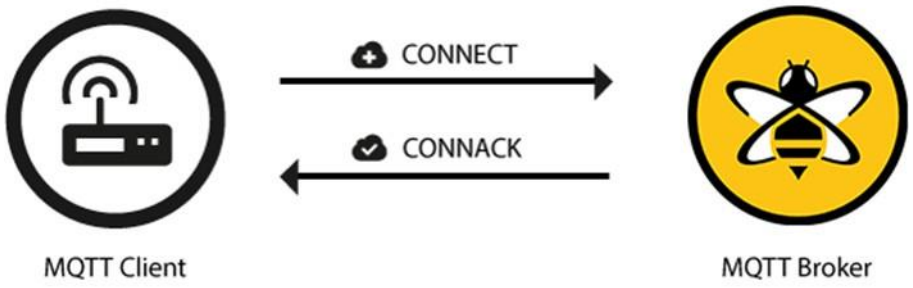
MQTT son las siglas en ingles de “Message Queuing Telemetry Transport” es un protocolo de comunicación ligero y de bajo consumo de ancho de banda diseñado para transmitir datos de forma eficiente en entornos de IoT (Internet de las cosas) y telemetría (HiveMQ, 2023). Fue desarrollado por IBM en 1999 y es un protocolo de comunicación basado en TCP/IP.

Este protocolo se utiliza comúnmente en sistemas de telemetría para enviar y recibir datos de sensores y dispositivos remotos. Su diseño ligero y su bajo consumo de ancho de banda lo hacen ideal para aplicaciones de telemetría que necesitan transmitir datos en redes de baja velocidad o limitadas en ancho de banda, como redes celulares y satelitales.

Según HiveMQ (2023), el protocolo MQTT utiliza un modelo de publicación/suscripción en el que los dispositivos publican información a través de un "broker" (un servidor central) y los dispositivos suscritos reciben la información. Esto permite que los dispositivos remotos puedan enviar datos sin la necesidad de mantener una conexión continua con el servidor central, lo que reduce la sobrecarga de la red. En la figura 4, se muestra el modelo de comunicación entre cliente y servidor.

Figura 5

Modelo Publicación/Suscripción



Nota. Adaptado de explicación del cliente y agente MQTT, por The HiveMQ Team, 2019, https://www.hivemq.com/blog/mqtt-essentials-part-3-client-broker-connection-establishment/

Además, en cuanto a la seguridad de la información, el MQTT proporciona mecanismos para garantizar la entrega de mensajes, lo que significa que los mensajes se pueden enviar de forma fiable incluso en condiciones de red inestables. También admite la encriptación de extremo a extremo para asegurar que la información se transmita de manera segura y privada.

Por lo tanto, el MQTT es un protocolo de comunicación eficiente y fiable que se utiliza comúnmente en sistemas de telemetría para transmitir datos de forma segura y fiable en entornos de IoT.

El Internet de las Cosas

El Internet de las cosas (IoT, por sus siglas en inglés) se refiere a la interconexión de objetos físicos cotidianos a través de internet, permitiendo que se comuniquen y compartan datos entre sí sin la necesidad de intervención humana directa. Estos objetos, también conocidos como dispositivos inteligentes, están equipados con sensores, actuadores y conectividad a internet, lo que les permite recopilar información, tomar decisiones y realizar acciones automatizadas (Oracle, s.f).

Esta tecnología tiene el potencial de transformar diversos sectores, como el hogar, la salud, la industria, la agricultura y la movilidad. Permite la creación de entornos inteligentes en los que los dispositivos pueden interactuar y colaborar para mejorar la eficiencia, la comodidad y la productividad. Por ejemplo, el IoT puede permitir el monitoreo remoto de la salud de los pacientes, la gestión automatizada de la energía y ambiente en los hogares, la optimización de la cadena de suministro en la industria y la implementación de sistemas de transporte inteligentes.

El IoT se basa en la capacidad de los objetos físicos para recopilar datos y comunicarse a través de redes de internet, lo que permite su integración en sistemas más amplios. Esto crea oportunidades para el análisis de datos en tiempo real, la toma de decisiones inteligentes y la automatización de procesos.

Interfaz de Programación de Aplicaciones (API)

Una API (Application Programming Interface, por sus siglas en inglés) es un conjunto de definiciones y protocolos que permite que diferentes aplicaciones de software se comuniquen entre sí (Amazon Web Services, Inc. 2023). Estas aplicaciones permiten que los desarrolladores integren y utilicen las funciones de otros servicios, aplicaciones o plataformas sin tener que

conocer los detalles internos de su implementación. En otras palabras, una API actúa como un intermediario que facilita la interacción entre diferentes sistemas de software.

Sus principales características son:

Interfaz de comunicación: Proporciona un medio para que diferentes aplicaciones se comuniquen entre sí, generalmente a través de llamadas HTTP en el caso de las APIs web.

Abstracción: Oculta la complejidad de las operaciones internas de una aplicación, ofreciendo funciones fáciles de usar para realizar tareas específicas.

Estandarización: Define un conjunto de reglas y formatos que deben seguirse para asegurar la compatibilidad entre los diferentes sistemas que interactúan.

Interoperabilidad: Permite la integración de diferentes sistemas y aplicaciones, facilitando la creación de servicios y aplicaciones más complejas y funcionales.

Seguridad: Las APIs pueden incluir mecanismos de autenticación y autorización para controlar el acceso a los datos y funciones de una aplicación.

CallMeBot

Es un servicio que permite a los desarrolladores y usuarios enviar mensajes y notificaciones a través de aplicaciones de mensajería populares como WhatsApp, Telegram, Facebook Messenger, entre otros. Esta aplicación proporciona una interfaz de programación de aplicaciones (API) que facilita la integración de estas capacidades de mensajería en sistemas y aplicaciones personalizadas (CallMeBot, 2021). Entre las principales características se tiene:

Notificaciones por Mensajería Instantánea: Permite enviar mensajes de texto a través de plataformas como WhatsApp, Signal y Telegram, facilitando la comunicación rápida y directa con los usuarios. Además, es ideal para enviar alertas, recordatorios, actualizaciones de estado y notificaciones de emergencia.

Fácil Integración: Proporciona una API sencilla y bien documentada, siendo fácil de integrar funcionalidades de mensajería en aplicaciones y sistemas. Por otra parte, incluye ejemplos de código y documentación detallada para una implementación rápida y efectiva.

Automatización: Permite la automatización de mensajes, lo que es particularmente útil en sistemas de monitoreo y alerta donde se requiere notificar a los usuarios de manera inmediata en caso de eventos específicos, como detección de incendios o gases peligrosos.

Compatibilidad: Funciona con múltiples plataformas de mensajería, ofreciendo flexibilidad y adaptabilidad según las preferencias de los usuarios y las necesidades del proyecto.

Seguridad: CallMeBOT implementa medidas de seguridad para proteger las comunicaciones y garantizar que los mensajes sean enviados únicamente a los destinatarios autorizados.

Sensor

Los sensores son dispositivos electrónicos diseñados para detectar y medir un fenómeno físico o químico. Estos dispositivos convierten una magnitud física o química, como la temperatura, la presión, la humedad, la luz, la posición, la aceleración, la fuerza, el sonido, entre otros, en una señal eléctrica que puede ser procesada y analizada por otros componentes electrónicos u ordenadores (MecatrónicaLATAM, 2021).

Existen diversos tipos de sensores para medir un gran número de variables físicas, entre ellos los sensores de temperatura, los sensores de luz, los sensores de sonido, los sensores de presión, los sensores de humedad, los sensores de movimiento, los sensores de proximidad, los sensores de nivel, los sensores de gas, los sensores de fuerza, los sensores de vibración, entre otros. Cada tipo de sensor tiene diferentes aplicaciones y métodos de funcionamiento, lo que los

hace útiles en una amplia variedad de dispositivos y sistemas, incluyendo la domótica, la robótica, la industria, la medicina, la seguridad, entre otros.

Temperatura y Humedad

Las variables de temperatura y humedad son dos factores fundamentales que inciden en la detección temprana de un incendio en el hogar, así como en la identificación de la presencia de gases peligrosos que puedan poner en riesgo la salud de las personas. En el proyecto de dispositivo de detección de incendio y gases peligrosos para el hogar, la variable de temperatura se refiere a una magnitud o medida a través de la cual se expresa la cantidad de calor o energía térmica que hay en el ambiente al interior de la vivienda (Lifeder, 2021). Con el objetivo de poder detectar de forma temprana cualquier incremento anormal de la misma que pudiera indicar la presencia de un incendio.

Por otro lado, la variable de humedad se refiere a la cantidad de vapor de agua presente en el aire, y su medición es importante porque puede ser un indicador de condiciones previas al incendio. Por ejemplo, si la humedad relativa es baja, los combustibles se secarán más rápido y, por lo tanto, serán más inflamables, aumentando el riesgo de incendio (El Servicio Nacional de Manejo del Fuego-SNMF, s.f). Además, la presencia de humedad excesiva puede favorecer el desarrollo de moho y otros problemas de salud en los residentes de las viviendas. En conjunto, estas variables permiten una monitorización precisa del ambiente en el hogar y contribuyen a la prevención de situaciones de riesgo.

A continuación, se menciona el principal sensor de temperatura y humedad que se ha encontrado en el mercado y que son totalmente compatibles con las distintas versiones de la plataforma Blynk IoT y Arduino.

Calidad del Aire

Según la OMS (2021), se refiere a la medida de la presencia de contaminantes en el aire que pueden afectar la salud humana, el medio ambiente y la vida en general. Estos contaminantes pueden ser gases como el dióxido de carbono, el monóxido de carbono, el dióxido de azufre y el ozono, así como partículas finas, gases tóxicos, compuestos orgánicos volátiles entre otros. Dicha calidad del aire se mide utilizando el Índice de Calidad del Aire (ICA), que es un indicador numérico que describe de manera clara, sencilla y unificada la calidad del aire en una determinada área geográfica.

La Resolución 2254 de 2017 del Ministerio de Ambiente y Desarrollo Sostenible adoptó en Colombia el Índice de Calidad del Aire (ICA), el cual es una metodología desarrollada por la Agencia de Protección Ambiental de los Estados Unidos. En la figura 9 se puede apreciar el ICA en Colombia, el cual establece los niveles de calidad del aire en función de la concentración de contaminantes como PM2.5, PM10, CO, SO2 y O3. Esta herramienta permite evaluar la calidad del aire de forma sencilla y comprensible para la población, facilitando la toma de decisiones y la implementación de medidas para mejorarla calidad del aire en el país.

Existen diferentes métodos para medir la calidad del aire, pero la mayoría de las mediciones se realizan mediante sensores electrónicos que miden la concentración de ciertos contaminantes en el aire. Los sensores pueden medir tanto la concentración de partículas como la concentración de gases.

Figura 6

Índice de Calidad de Aire en Colombia

Rango	Color	Estado	Efectos
0 - 50	Verde	Buena	La contaminación atmosférica supone un riesgo bajo para la salud.
51 - 100	Amarillo	Aceptable	Posibles síntomas respiratorios en grupos poblacionales sensibles.
101 - 150	Naranja	Dañina a la salud de grupos sensibles	Los grupos poblacionales sensibles pueden presentar efectos sobre la salud. 1) Ozono Troposférico: las personas con enfermedades pulmonares, niños, adultos mayores y las que constantemente realizan actividad física al aire libre, debe reducir su exposición a los contaminantes del aire. 2) Material particulado: Las personas con enfermedad cardiaca o pulmonar, los adultos mayores y los niños se consideran sensibles y por lo tanto en mayor riesgo.
151 - 200	Rojo	Dañina para la salud	Todos los individuos pueden comenzar a experimentar efectos sobre la salud. Los grupos sensibles pueden experimentar efectos más graves para la salud
201 - 300	Púrpura	Muy dañina para la salud	Estado de alerta que significa que todos pueden experimentar efectos más graves para la salud
301 - 500	Marrón	Peligrosa	Advertencia sanitaria. Toda la población puede presentar efectos adversos graves en la salud humana y están propensos a verse afectados por graves efectos sobre la salud

Nota. Adaptado de MADS, 2017, Resolución 2254 del 2017, Norma de calidad del aire ambiente. https://www.minambiente.gov.co/wp-content/uploads/2021/10/Resolucion-2254-de-2017.pdf

El Sistema de Información Ambiental de Colombia (SIAC)

El Sistema de Información Ambiental de Colombia (SIAC) es una plataforma desarrollada por el Ministerio de Ambiente y Desarrollo Sostenible de Colombia, cuyo objetivo principal es consolidar, integrar, administrar y divulgar información ambiental del país. Este sistema actúa como un mecanismo centralizado y de acceso público que permite reunir datos, indicadores, reportes y análisis relacionados con el estado del medio ambiente en Colombia (Sistema de Información Ambiental de Colombia [SIAC], 2024).

El SIAC sirve como una herramienta estratégica para la toma de decisiones, planificación y gestión ambiental, tanto por parte de las autoridades gubernamentales como por las

organizaciones privadas, académicas y la ciudadanía en general. Proporciona información clave para evaluar la calidad de los recursos naturales, el impacto de las actividades humanas y las estrategias de conservación y mitigación.

Componentes principales del SIAC:

Módulos temáticos: El SIAC organiza la información en diferentes áreas temáticas como biodiversidad, calidad del aire, agua, suelo, cambio climático, gestión de residuos y otros aspectos relevantes del ambiente.

Indicadores ambientales: Contiene indicadores específicos que reflejan el estado y la evolución de los recursos naturales y los factores ambientales en el país.

Mapas y herramientas geográficas: Integra mapas y sistemas de información geográfica (SIG) para facilitar la visualización de datos espaciales y georreferenciados.

Datos abiertos: Permite acceder a datos descargables para su análisis y uso en investigaciones o proyectos relacionados con el ambiente.

Información normativa: Incluye leyes, normativas y políticas relacionadas con la gestión ambiental.

Gases Inflamables

Los gases inflamables también conocidos como gases combustibles son aquellos que pueden prender o explotar en presencia de una fuente de ignición, de aire o de un oxidante (Gasex, 2021). Estos gases tienen un punto de inflamación bajo, lo que significa que pueden encenderse con una pequeña chispa, llama o calor. Los principales gases inflamables incluyen el gas natural, el propano, el butano, el hidrógeno, el acetileno, propileno, el metano, monóxido de carbono y vapores de petróleo. La presencia de gases inflamables en el aire puede ser peligrosa

en áreas confinadas o sin ventilación adecuada, ya que pueden acumularse y provocar una explosión o incendio.

Los gases más comunes en el hogar son el gas natural, el propano, el dióxido de carbono, el monóxido de carbono y el humo.

¿Y por qué es importante medir las concentraciones de estos gases en los hogares?

Los gases como el gas natural y el propano se utilizan como combustibles para cocinar, calentar y generar electricidad. El dióxido de carbono es un subproducto de la combustión de los combustibles fósiles, mientras que el monóxido de carbono es un subproducto peligroso de la combustión incompleta de combustibles como la madera, el carbón y el gas natural. Ejemplo, el calentador de agua a gas, cual utiliza la combustión de gas natural para calentar el agua. Por otro lado, el humo es un subproducto común de la combustión y puede ser causado por la quema de alimentos, la combustión de madera y otros materiales.

En la table 3, se muestra las principales características de la familia de sensores de gas electroquímicos MQ, que detectan la presencia de diferentes gases en el aire, y que son totalmente compactible con las tarjetas Arduino o ESP32.

Tabla 2

Familia sensor MQ

Sensor de Gas	**Rango de detección**	**Voltaje de operación**	**Consumo corriente**	**Tiempo respuesta**	**Sensibilidad**
MQ-2 humo y gas natural	Gas inflamable,	5V	150 mA	<10 s	300 - 10000 ppm

MQ-5 LPG, CH4, CO, Alcohol	Gas inflamable, H2	5V	150 mA	<10 s	200 - 10000 ppm
MQ-7	CO	5V	150 mA	<10 s	20 - 2000 ppm
MQ-135 C6H6, CO, CO2, NO2	Calidad de aire, NH3	5V	150 mA	<10 s	10 - 1000 ppm

Nota. Principales características de los sensores MQ

Estos sensores, incluido el MQ-5 y el MQ-135, funcionan mediante la variación de la resistencia eléctrica en función de la concentración de gas en el aire. La salida es analógica a través del pin Aout y es fácilmente accesible por medio de la tarjeta Arduino o ESP32.

Estado del arte

Los sistemas de detección de incendios y gases peligrosos para el hogar han evolucionado significativamente en la última década. Se han desarrollado varios sistemas que utilizan tecnologías similares a las que se plantean en el proyecto, como sensores catalíticos y temperatura, altavoces y luces LED para alertar a los usuarios de posibles emergencias, así como aplicaciones móviles para monitorear y recibir alertas en tiempo real. Entre los sistemas más destacados en el mercado se encuentran:

En el estudio realizado por Pérez Galvis, Harold (2022), titulado "Detección y monitoreo de gases mediante sensores catalíticos IoT", se aborda la problemática de identificar y

monitorear gases inflamables como GLP, propano, monóxido de carbono, hidrógeno, metano y humo. El proyecto destaca el uso de sensores catalíticos IoT, reconocidos por su alta sensibilidad y capacidad para realizar mediciones en tiempo real en rangos de 300 a 10,000 ppm, proporcionando un monitoreo efectivo en áreas urbanas e industriales.

El autor logro construir un prototipo funcional de detector de gases (CO, H2, CH4, GLP, propano y humo), diseñado para monitorear y transmitir datos a la nube mediante la red de comunicación Sigfox. Este dispositivo está pensado para instalarse cerca de fuentes de gas a baja altura, aprovechando la lógica de que los gases inflamables, por su mayor densidad, pueden detectarse con mayor rapidez en estas ubicaciones. El estudio también resalta la importancia de realizar pruebas en entornos controlados para garantizar una adecuada calibración y el correcto funcionamiento del dispositivo, asegurando resultados precisos y confiables. Además de su aporte técnico, el proyecto permitió desarrollar nuevas habilidades en diseño e implementación de proyectos basados en sensores catalíticos, así como mejorar la gestión de tiempo y recursos. Este avance representa un paso significativo en la prevención de riesgos asociados a gases inflamables, contribuyendo tanto al desarrollo académico como a la seguridad en entornos vulnerables.

El Google Nest (2022), que utiliza sensores de humo y monóxido de carbono, además de una aplicación móvil para monitorear el estado del sistema en tiempo real. El First Alert Onelink (2022) también es un sistema de detección de humo y monóxido de carbono que utiliza sensores y una aplicación móvil para recibir alertas y monitorear el estado del sistema. El Kidde Wireless (2022), por su parte, utiliza sensores de humo y monóxido de carbono, así como una aplicación móvil para recibir alertas y monitorear el estado del sistema. Finalmente, el Smartthings (2022) es un sistema de automatización del hogar que cuenta con sensores de humo y monóxido de

carbono, así como otros sensores de temperatura y movimiento para monitorear el hogar y recibir alertas en caso de emergencias.

En comparación con estos sistemas, el proyecto actual se enfoca en la utilización de componentes de bajo costo y accesibles para implementar un sistema de detección y alerta de emergencias en hogares de Colombia, especialmente en la ciudad de Santiago de Cali, donde este tipo de dispositivos son menos comunes. Este enfoque permitirá una mayor accesibilidad y cobertura en la detección y alerta temprana de emergencias en hogares de bajos recursos.

De ahí que, el sistema de detección de incendios que envía alertas a usuarios a través de mensaje de texto es una solución innovadora y eficiente para prevenir y controlar incendios en hogares en este grupo poblacional. En la literatura científica se han reportado diversos proyectos que utilizan la tarjeta Arduino para desarrollar sistemas de este tipo.

Por ejemplo, Cifuentes y García (2020) desarrollaron un sistema de monitoreo y alerta de incendios basado en Arduino y sensores de temperatura y humo. El sistema envía alertas por mensaje de texto y utiliza una aplicación móvil para controlar y monitorear el estado del sistema en tiempo real.

Por su parte, Pérez et al. (2021) diseñaron un sistema de detección de incendios basado en Arduino y sensores de gas, temperatura y humo. El sistema utiliza un módulo GSM para enviar alertas por mensaje de texto y una aplicación móvil para monitorear y controlar el sistema.

En otro estudio, Díaz et al. (2018) desarrollaron un sistema de detección de incendios y monitoreo remoto basado en Arduino y sensores de gas y temperatura. El sistema envía alertas por mensaje de texto y utiliza una plataforma web para monitorear el estado del sistema en tiempo real.

Por otra parte, González, J. C., & González, V. (2018), en el artículo "Desarrollo de un sistema de detección y control de incendios con notificación mediante mensajes de texto" describen el desarrollo de un sistema de detección y control de incendios para uso en hogares y edificios comerciales. El sistema utiliza sensores de humo y calor para detectar incendios y activar una alarma sonora y visual. Además, el sistema utiliza un módulo GSM para enviar mensajes de texto a los números de teléfono de los usuarios en caso de una emergencia.

Y, por último, Castro (2013), enseña a explorar las posibilidades de desarrollo de un sistema de control de temperatura basado en la plataforma Arduino. Para lograr este objetivo se implementó un sistema que permite visualizar y controlar la temperatura del aire circundante a través de la red móvil. Luego del análisis previo de varias placas Arduino, se evalúa una serie de módulos de expansión (shields) compatibles con la mencionada plataforma, que permite ampliar sus funciones al dotar al dispositivo de un sistema de comunicación basado en tecnología GPRS/GSM.

Metodología

Para el desarrollo del presente proyecto se utilizó una metodología estructurada, la cual permitió dividir el trabajo en fases claramente definidas, asegurando un enfoque sistemático y organizado, entendido como un método que permite gestionar cada fase del proyecto de manera lógica y ordenada para optimizar los resultados (Hernández Sampieri, Fernández y Baptista, 2014). Esta metodología incluyó cuatro fases fundamentales: análisis del problema, diseño, implementación, pruebas y validación del sistema, cada una con objetivos específicos y actividades detalladas. A continuación, se detallan cada uno de ellas:

Análisis del Problema y Planeamiento

En esta etapa, se realizó un análisis del problema, identificando la necesidad de monitorear de manera remota las condiciones ambientales en el hogar para prevenir riesgos asociados a incendios y enfermedades. A partir de este análisis, se definieron las principales variables a monitorear, enfocadas en la detección de gases peligrosos, la calidad del aire, la temperatura y la humedad. Las variables críticas consideradas incluyen: temperatura y humedad, factores claves que inciden en la detección temprana de un incendio en el hogar y confort del mismo; Gases inflamables, como gas natural, propano, butano y metano, cuya detección temprana es esencial para evitar explosiones o intoxicaciones; y Gases contaminantes que afectan la calidad del aire, como monóxido de carbono, amoníaco, benceno y humo, los cuales pueden tener efectos adversos en la salud, especialmente en personas con enfermedades respiratorias preexistentes.

Además, en esta etapa se definieron los requisitos funcionales del sistema, incluyendo la visualización local de las variables medidas, la capacidad de monitoreo remoto a través de una aplicación móvil y el envío de alertas en tiempo real a los usuarios en caso de detectar condiciones peligrosas en el hogar. Para ello, el sistema debe basarse en un dispositivo embebido con conexión Wi-Fi, que actúe como el núcleo del sistema y sea responsable de recopilar los datos de los sensores, así como de gestionar la activación de alarmas o respuestas automáticas.

Asimismo, el dispositivo embebido debe estar conectado tanto física como virtualmente a distintos dispositivos de visualización, entre los que se incluyen una pantalla LCD y una plataforma de monitoreo IoT. Esto permitirá que los residentes accedan a la información en tiempo real, ya sea de manera local o remota, y reciban alertas automáticas ante cualquier condición peligrosa detectada en el hogar.

Basado en lo anterior, se estableció un cronograma de actividades que incluyo las tareas a desarrollar en cada una de las etapas o fases al igual que los tiempos estimados para cada una de ellas. Ver tabla 3.

Tabla 3

Cronograma de Actividades para Desarrollo del Proyecto

ACTIVIDAD	MES 1	MES 2	MES 3	MES 4	MES 5	MES 6
Fase 1: Identificar las necesidades del sistema. -Revisión de la literatura científica y tecnológica -Evaluación de los requisitos del sistema	X					
Fase 2: Diseño del sistema - Selección y especificación técnica de los componentes - Diseño del circuito electrónico - Programación la tarjeta ESP32 - Diseño y programación de Interfaz Remota Blynk IoT -Programación de la API CallMeBot al proyecto para el envió de notificaciones por WhatsApp		X	X			
Fase 3: Implementación del sistema. -Impresión del circuito en plaqueta - Montaje de componentes en plaqueta y en carcasa - Implementar la comunicación entre los componentes del sistema y la Interfaz Blynk IoT para permitir el acceso remoto y el monitoreo en tiempo real de los datos recopilados por los sensores. -Implementación de la API CallMeBot			X	X	X	
Fase 4: Pruebas y validación del sistema - Pruebas en diferentes escenarios - Validación de los resultados obtenidos a través de las pruebas y se identificarán oportunidades de mejora					X	X

Diseño de la Solución

El diseño del sistema de detección de incendios y gases para el hogar se ha estructurado en cuatro etapas principales para asegurar una correcta implementación del proyecto. En la primera etapa, se establecieron las especificaciones técnicas necesarias para los dispositivos y sensores que serán implementados en el sistema. En la segunda etapa, se procedió al diseño del plano del circuito para integrar los diferentes componentes del sistema y su correcta interconexión.

La tercera etapa se enfocó en el algoritmo del sistema, donde se desarrolló el código necesario para la detección de incendios y gases, la medición de la temperatura y humedad, y el envío de alertas al dispositivo móvil en caso de alguna anomalía. Esta etapa también incluyó la programación de la interfaz de usuario, para que el usuario final pueda interactuar con el sistema de manera sencilla e intuitiva.

Por último, en la etapa cuatro se llevó a cabo la simulación del sistema para verificar su correcto funcionamiento. Se utilizaron herramientas de simulación como Proteus para verificar el correcto funcionamiento del circuito.

Etapa 1 –Selección y Módulo ESP32 Especificaciones Técnicas de Componentes

El ESP32 es un microcontrolador de bajo costo y alto rendimiento. Está basado en el microcontrolador Xtensa® Dual-Core LX6 de 32 bits y debido a su capacidad de conectarse a internet a través de la tecnología Wifi y Bluetooth integrada, la convierte en un dispositivo ideal para proyectos de Internet de las Cosas (IoT). Además, de su fácil y diversas formas de programar usando para ello Arduino IDE, como también su compatibilidad con diversos

sensores. Siendo estas características fundamentales para la implementación del proyecto del curso.

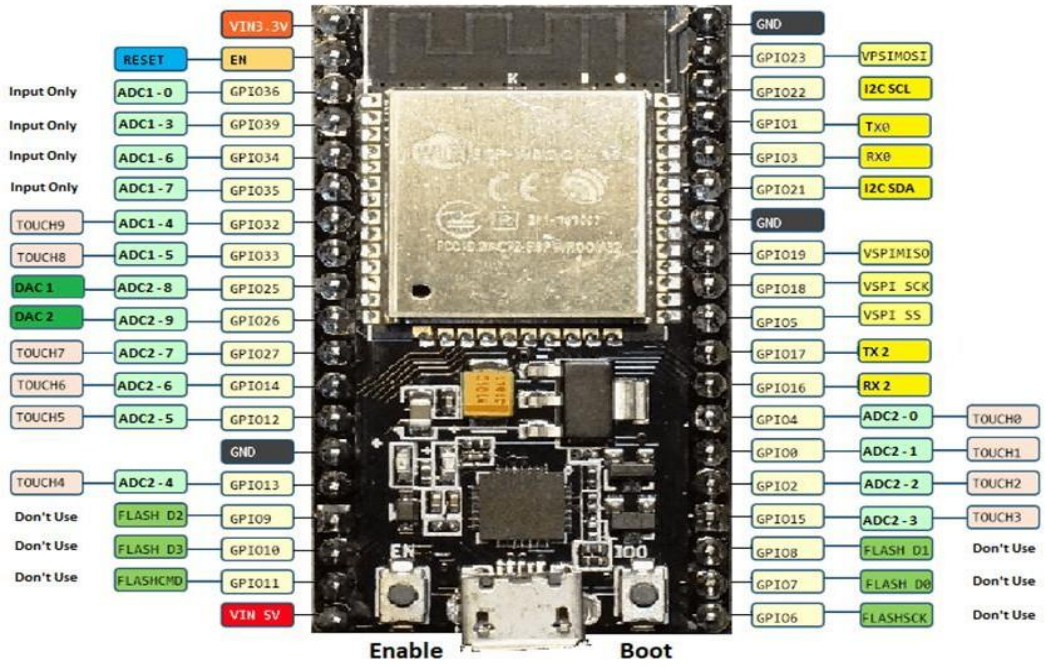
Entre sus principales características, se encuentra:

- CPU: Xtensa® Dual-Core LX6 de 32 bits;
- Memoria ROM: 448 KBytes;
- Reloj máximo: 240MHz;
- Memoria RAM: 520 Kbytes;
- Memoria flash: 4 MB;
- Estándar inalámbrico 802.11 b / g / n;
- Conexión Wifi de 2.4Ghz (máximo 150 Mbps);
- Antena integrada la placa;
- Conector micro USB para comunicación y alimentación a hasta 5Vdc;
- Wi-Fi Direct (P2P), P2P Discovery, modo P2P Group Owner y P2P Power Management;
- Modos de funcionamiento: STA / AP / STA + AP;
- Bluetooth BLE 4.2;
- Puertos GPIO: 11;
- GPIO con PWM, I2C, funciones SPI, etc.
- Voltaje de operación; 3.3 Vdc
- Convertidores analógicos a digital (ADC) y digital a analógico (DAC)

Dado que el objetivo principal del proyecto es notificar situaciones de emergencia a los usuarios, la tarjeta ESP32 actuará como dispositivo central para recopilar los datos de los sensores y controlar la transmisión de la información a través de la RED por medio de la conexión Wifi integrado que posee. Esto permitirá conectarse remotamente a la interfaz móvil IoT de Blynk para que los usuarios interactúen y a su vez monitoreen las variables en caso de detectar concentraciones peligrosas de gases, humo u otra contaminación del aire doméstico. Además, para una respuesta rápida y efectiva, el sistema estará equipado con dispositivo sonoro local que se emitirá en caso de una situación de emergencia para avisar a los residentes que se encuentren en la vivienda. En la figura 7, se muestra el PINOUT del ESP32.

Figura 7

PINOUT del ESP32



Nota. De Circuitos electrónicos, s.f, ESP32 – Especificaciones y diseños. [en línea]. https://www.circuitos-electricos.com/esp32-especificaciones-y-disenos/

Sensores

En el proyecto descrito, se requiere de sensores que permitan la captura de datos de las variables de temperatura y humedad, así como la detección de diferentes gases potencialmente peligrosos en el aire doméstico. Dichos sensores deben cumplir los siguientes requisitos:

• Total compactibilidad con la tarjeta ESP32 y Arduino
• Disponibilidad en el mercado
• Bajo costo
• Librerías, Ejemplos de programación y documentación de soporte de libre acceso.

Luego de evaluar y analizar mencionados requerimientos, los sensores que cumplen estas condiciones son:

Sensor MQ-135

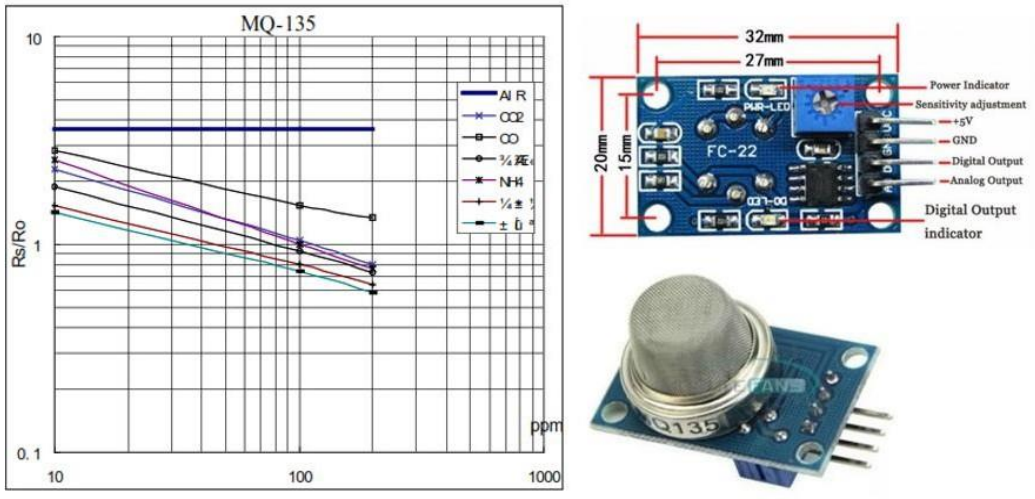
Es un sensor de calidad del aire capaz de detectar diferentes concentraciones de gases contaminantes, incluyendo amoníaco, benceno, humo, dióxido de carbono y monóxido de carbono. Su bajo costo, tamaño reducido y facilidad de uso lo hacen adecuado para aplicaciones en el hogar. Sus principales características son:

- Rango de detección: 10 a 300 ppm (partes por millón)
- Sensibilidad: 1.5 a 2.5 mA/ppm
- Tiempo de respuesta: <10 segundos
- Consumo de energía: 800 mW
- Temperatura de trabajo: -10 a 50 °C
- Humedad de trabajo: <95% RH
- Dimensiones: 35 x 22 x 21 mm
- Alimentación a 5Vdc
- Aplicación de mediciones de gas en interiores

En la figura 8 se muestra el sensor y las características sensitivas del MQ-135.

Figura 8

Sensor MQ-135



a) Características sensitivas b) Aspecto físico

Nota. a) de datasheet MQ-135 y b) de Mercado Libre

Sensor MQ-5

Es un sensor de gas que detecta la presencia de gas natural, gas licuado de petróleo (GLP), butano y otros gases inflamables en el aire. Estos gases pueden ser peligrosos si se acumulan en una concentración peligrosa en los hogares, por lo que es importante contar con un sensor que pueda detectarlos y alertar a los usuarios.

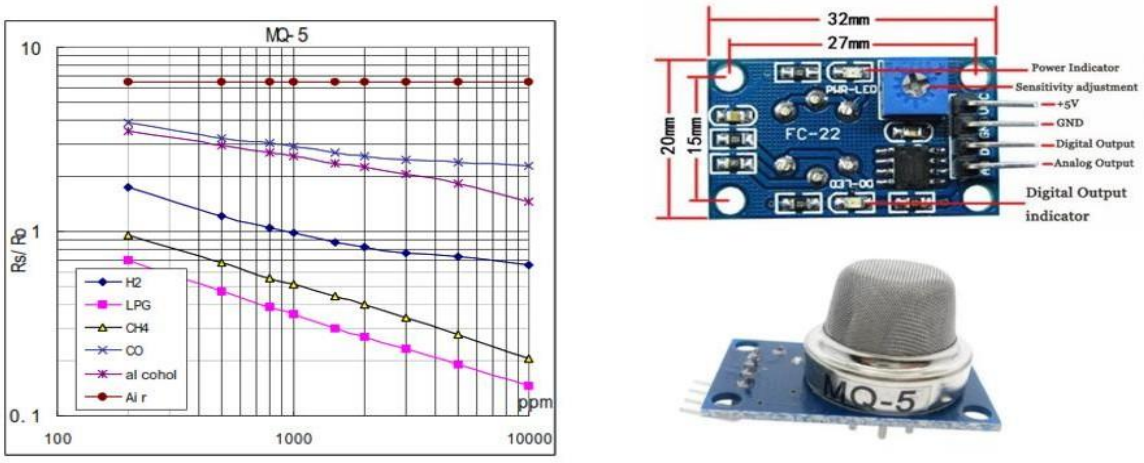
Al igual que el MQ-135, el MQ-5 tiene un bajo costo y tamaño. Además, el MQ-5 es fácil de usar lo hace adecuado para aplicaciones como detección de fugas de gas y monitoreo ambiental en el hogar. Sus principales características son:

- Rango de detección: 200 a 10.000 ppm
- Sensibilidad: 0.6 a 1.9 mA/ppm
- Tiempo de respuesta: <10 segundos
- Consumo de energía: 900 mW // Alimentación a 5Vdc
- Temperatura de trabajo: -20 a 50 °C
- Humedad de trabajo: <95% RH

En la figura 9 se muestra el sensor y las características sensitivas del MQ-5.

Figura 9

Sensor MQ-5



a) Características b) Aspecto físico

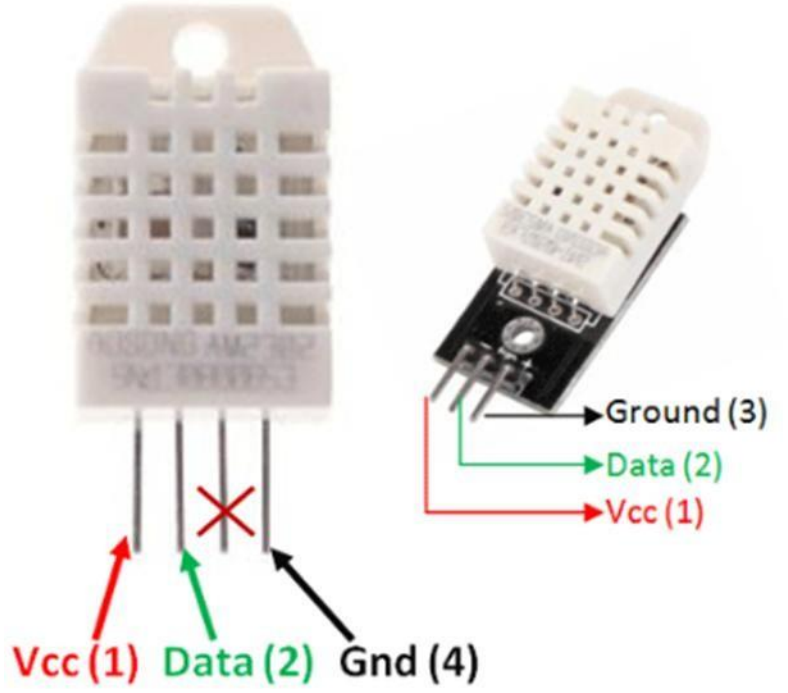
Nota. a) de datasheet MQ-5 y b) de MercadoLibre

Sensor DHT22

Es un sensor digital de temperatura y humedad de alta precisión, compactible con la tarjeta ESP32 y de bajo costo. Es capaz de medir la temperatura en un rango de -40 a 80 grados Celsius con una precisión de ±0.5 grados Celsius. Además, es capaz de medir la humedad relativa en un rango del 0 al 100% con una precisión de ±2-5%. Se alimenta con una tensión de entrada de 3,3 - 5 V CC. Estas características lo hacen apto para su implementación en el proyecto, dado que proporciona una excelente relación costo beneficio. En la figura 10, se muestra la distribución de pines de este sensor.

Figura 10

Sensor DHT22/AMT2302



Nota. De Circuits-elec, 2023, chipKIT Uno32. [en línea]. https://circuits-elec.com/products/temperature-humidity-sensor-module-dht22

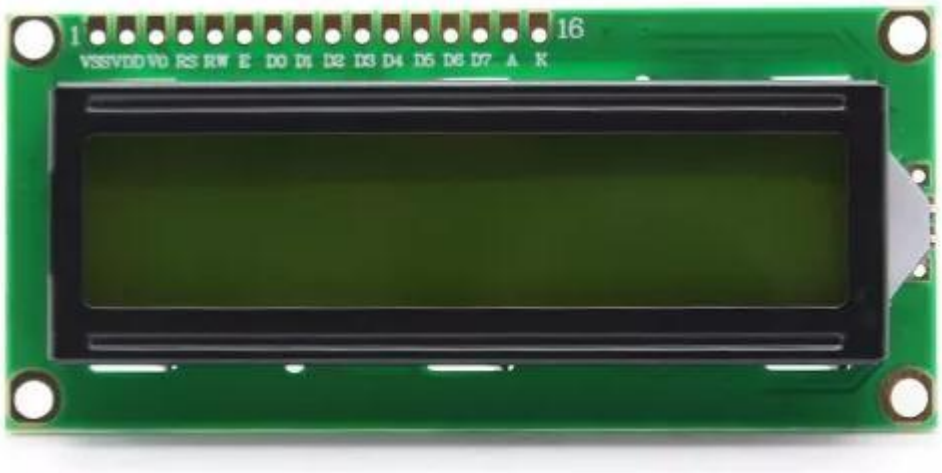
Por otra parte, El DTH22 proporciona una señal de salida digital (Pin 2) en la que los datos de temperatura y humedad se codifican en un solo paquete. Este sensor utiliza una señal digital de un solo cable para comunicarse con el microcontrolador o tarjeta ESP32.

Pantalla LCD 16x2

El display LCD 16x2 es un componente clave en el prototipo de detección de incendios y gases peligrosos para el hogar, ya que proporciona una interfaz local para visualizar información crítica en tiempo real. A continuación, se detalla la descripción y características de este dispositivo, así como su integración en el proyecto. Ver la figura 11.

Figura 11

Pantalla LCD 16x2



Nota. De Mercado Libre, 2024, Display Lcd 16x2 Con Backlight Amarillo. [en línea]. https://articulo.mercadolibre.com.co/MCO-586699799-display-lcd-16x2-con-backlight-amarillo-_JM

Características del Display LCD 16x2

- El display tiene 16 columnas y 2 filas, permitiendo la visualización de 32 caracteres en total.

- Cada carácter es representado por una matriz de puntos de 5x8 píxeles.
- El LCD 16x2 puede comunicarse con la Tarjeta ESP32 a través de una interfaz paralela de 8 bits o de 4 bits.
- En este proyecto, se utiliza la configuración de 4 bits para optimizar el uso de pines GPIO en la ESP32.
- Pines de Conexión:
- RS (Register Select): Selecciona el registro de datos o comandos.
- E (Enable): Habilita la escritura de datos.
- RW (lectura/escritura, conectado a GND para solo escritura)
- D4 a D7: Pines de datos para la comunicación en modo de 4 bits.
- VCC y GND: Pines de alimentación (5V y tierra).
- VEE: Pin para ajustar el contraste, conectado a un potenciómetro.
- Backlight: Pines para la iluminación de fondo del display.

Integración en el Proyecto

Este display se utilizará para mostrar información relevante sobre las condiciones del hogar, como los niveles de gases detectados, la temperatura y la humedad. Asimismo, proporciona alertas visuales inmediatas sobre la calidad del aire y la presencia de gases peligrosos. Facilitando la monitorización local sin necesidad de un dispositivo móvil o conexión a internet. Su integración se detalla a continuación:

Conexión de Pines:

- RS: Conectado al pin GPIO 15 del ESP32.
- E: Conectado al pin GPIO 4 del ESP32.
- D4: Conectado al pin GPIO 5 del ESP32.
- D5: Conectado al pin GPIO 18 del ESP32.

- D6: Conectado al pin GPIO 19 del ESP32.
- D7: Conectado al pin GPIO 21 del ESP32.
- VCC: Conectado a la fuente de alimentación de 5V.
- GND: Conectado a tierra.
- VEE: Conectado a un potenciómetro de 10kΩ para ajustar el contraste.
- Backlight: Alimentado desde la fuente de 5V para la iluminación de fondo.

Programación y Control

La programación del display se realizará mediante una serie de librerías específicas para LCD en el entorno de desarrollo Arduino IDE. Posteriormente, el control estará a cargo de la tarjeta ESP32 que dará las funcionalidades descriptas anteriormente.

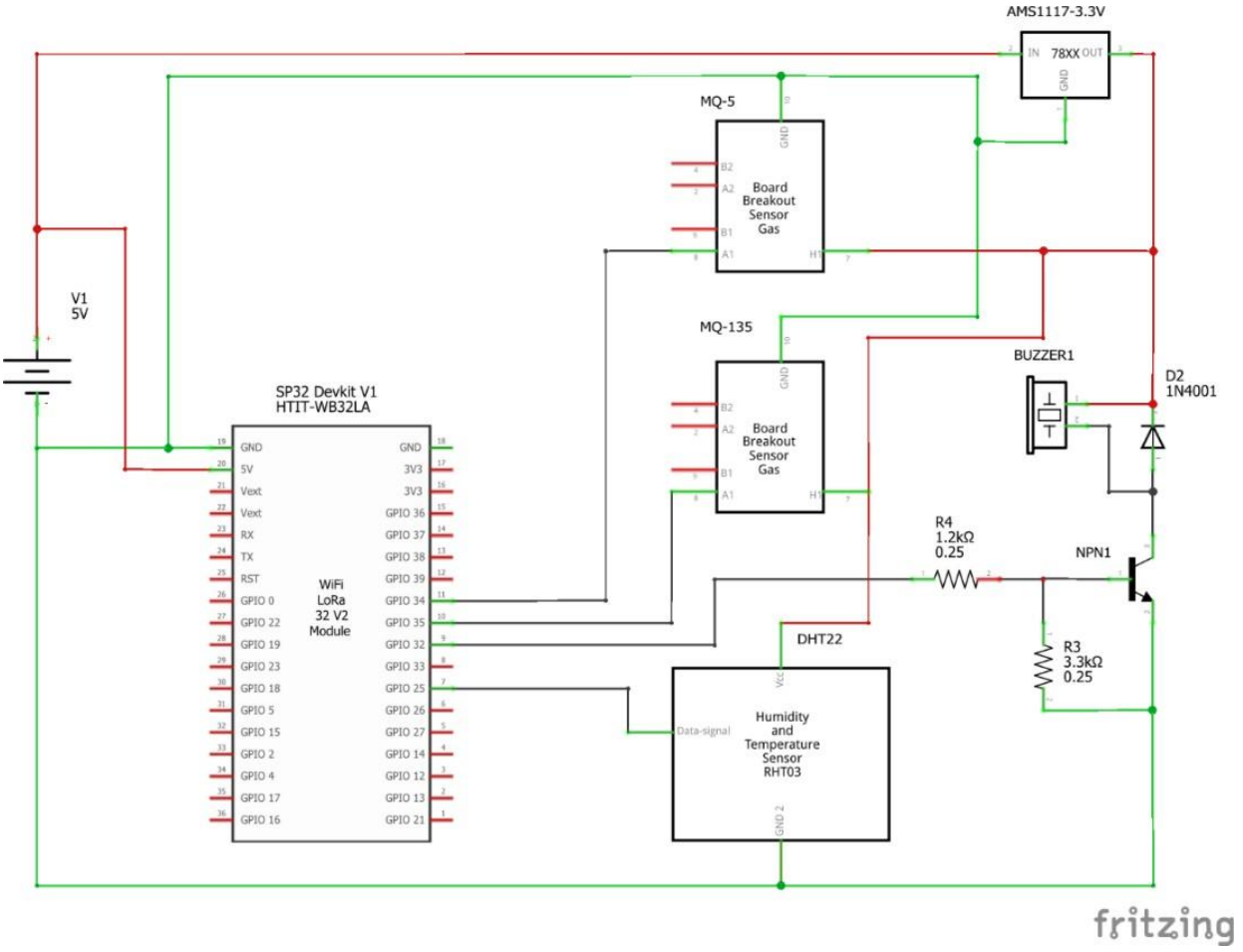
Etapa 2 – Diseño del Plano Electrónico

La Figura 12 presenta el diagrama detallado del sistema de detección de incendios y gases diseñado principalmente para el uso en el hogar. Este sistema opera a 5 voltios de corriente continua, lo que asegura una operación confiable y segura.

Además, el sistema está diseñado para funcionar automáticamente, lo que significa que no requiere intervención del usuario para su operación. El sistema está equipado con sensores de detección de incendios y gases que, cuando detectan una situación de emergencia, envían alertas al usuario por medio de una interfaz remota a través de la red. Adicionalmente, en caso de una situación de emergencia, se emitirá una alerta sonora para notificar a los ocupantes de la casa y garantizar una respuesta rápida.

Figura 12

Diseño Electrónico del Sistema

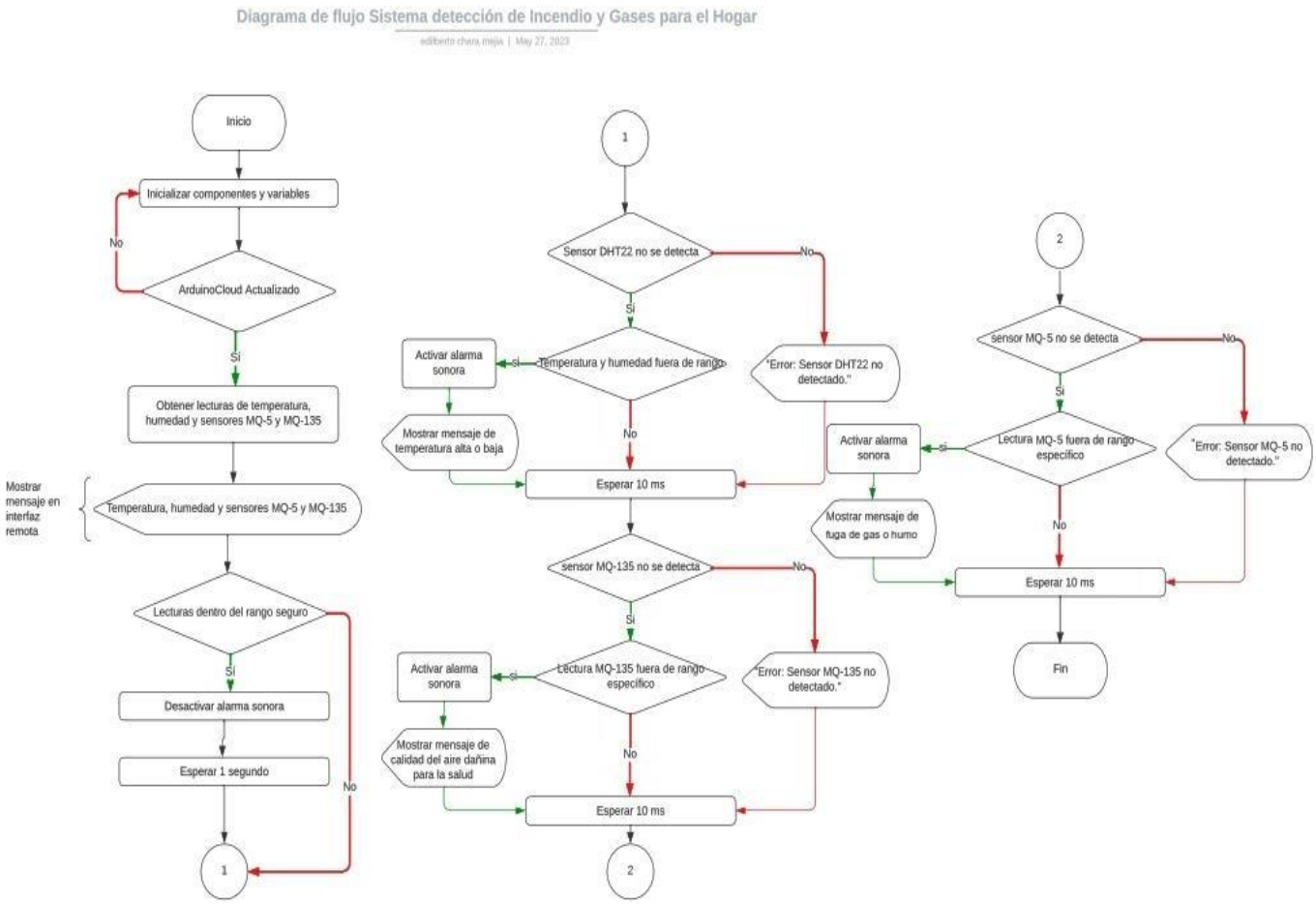


Etapa 3 – Algoritmo

A continuación, se presentará el algoritmo, el cual consiste en un conjunto de pasos específicos que se deben seguir para lograr el objetivo del proyecto. Para representar visualmente el algoritmo, se utilizará un diagrama de flujo, que permitirá una comprensión más clara y sencilla del proceso. Además, de permitir identificar posibles problemas o errores en el proceso antes de que ocurran, lo que ayudará a mejorar la eficiencia del proyecto.

Figura 13

Diagrama De Flujo Para el Prototipo de Monitoreo de Gases para el Hogar

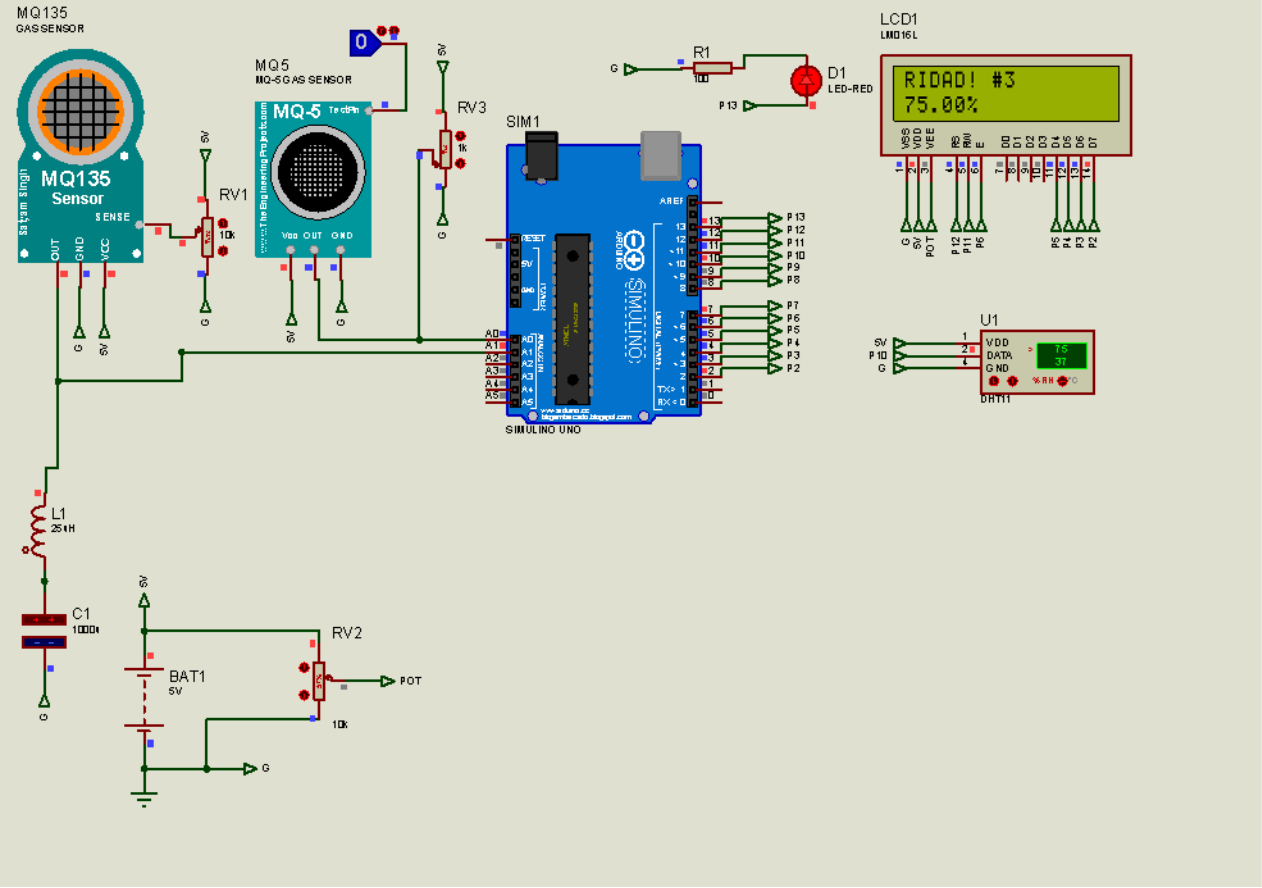


Etapa 4 – Simulación

En la figura 14 se puede observar el circuito simulado en Proteus del dispositivo diseñado para la detección de incendios y gases en el hogar. Actualmente se encuentra en la etapa de diseño y programación. Una vez finalizado el proceso de diseño y programación, se procederá a la implementación del dispositivo para su uso en el hogar.

Figura 14

Simulación del Circuito Diseñado en Proteus



Implementación del Sistema

A continuación, se detalla el proceso de implementación del proyecto:

Montaje de Dispositivos en Placa de Circuito Impreso (PCB)

Para el montaje de los dispositivos, se consideraron los seleccionados en la Etapa 1 – Selección y Especificaciones Técnicas de Componentes. Dichos dispositivos fueron integrados inicialmente en una protoboar, siguiendo las especificaciones detalladas a continuación:

- Tarjeta ESP32: Esta tarjeta es responsable de procesar los datos recolectados por los sensores y transmitirlos a la interfaz Blynk IoT. Además, envía notificaciones a WhatsApp y

activa las alarmas visuales en la pantalla LCD y sonoras a través del Buzzer. Su alimentación se hizo a través del conector mini USB de la placa a 5V del adaptador para teléfonos móviles y el resto de componentes a través de los pines GPIO correspondientes, que se detallan a continuación.

- Sensores MQ-5 y MQ-135: Los sensores electroquímicos de detección de gases inflamables y calidad de aire se conectan al ESP32 a través de los pines GPIO correspondientes. El sensor MQ-5, con su salida analógica, se conecta al pin GPIO 34, y el sensor MQ-135 al pin GPIO 35. Los pines de alimentación de ambos sensores se conectan a una fuente de 3.3 voltios. Estos sensores recolectan datos de temperatura, humedad y concentraciones de gases inflamables y contaminantes en el aire.

- Sensor DHT22: Este sensor de temperatura y humedad se conecta al pin GPIO 25 del ESP32 a través de su pin de salida Data (2). Los pines Vcc y GND se conectan a una fuente de 3.3 V.

- Pantalla LCD 16x2: Para la comunicación entre el LCD y el controlador, se configuró el BUS de datos en modo de 4 bits, además de los pines de control RS (chip select), RW (lectura/escritura, conectado a GND para solo escritura), y E (enable). La conexión se realiza de la siguiente manera: RS = 15, EN = 4, D4 = 5, D5 = 18, D6 = 19, y D7 = 21. Para el control del contraste del LCD, pin VEE, se utilizó un potenciómetro de 10k. El resto de los pines de alimentación y Backlight se conectan a la fuente de 5V. Este dispositivo permitirá la monitorización local de las condiciones del hogar.

- Transistor 2N2222 y Buzzer: Estos dos dispositivos sirven como alarma sonora local para el prototipo. El transistor está configurado como interruptor y su base está conectada al

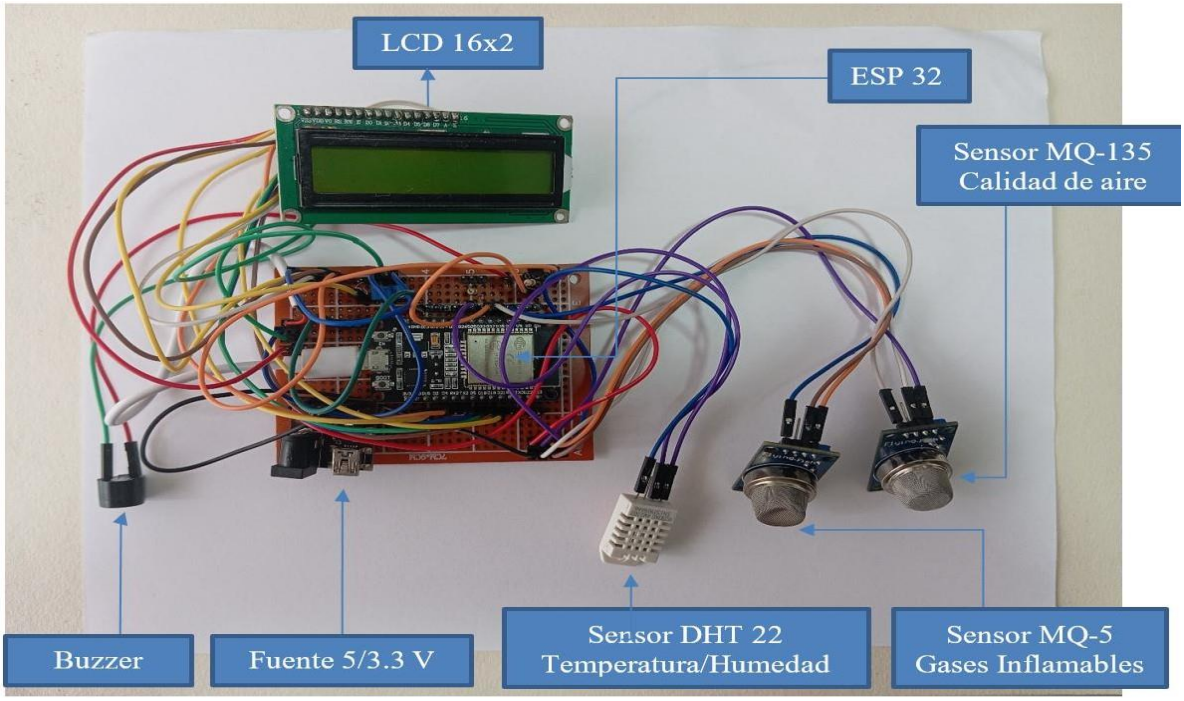
GPIO 32 del ESP32, el colector a un extremo del Buzzer y el emisor a tierra. Así, cuando llegue la señal del ESP32, el Buzzer producirá un sonido que indicará la presencia audible de una alarma.

- Fuentes de Voltaje: Se utilizan dos fuentes de voltaje: la primera es un adaptador para teléfonos móviles de 5 VCC a 2 A y la segunda es una fuente regulada dual de 5 y 3.3 V. La primera alimenta el ESP32 a 5 V a través de un conector mini USB, la pantalla LCD y la fuente regulada. La fuente regulada alimenta a los sensores a 3.3 V, dado que ese es el voltaje de operación del ESP32.

Una vez fue verificado su correcto funcionamiento de cada uno de los componentes, se procedió a montarlos en la PCB, como se observa en la figura 15.

Figura 15

Montaje de Dispositivos en PCB



Calibración de Sensores MQ

La calibración de los sensores MQ-5 y MQ-135 es esencial para garantizar la precisión y confiabilidad en la detección de gases peligrosos y la calidad del aire en el hogar. Para realizar la calibración de las mediciones de estos gases a través de los sensores MQ, se empleó una relación matemática con base en la información proporcionada por los fabricantes de los sensores a través de sus hojas de datos, utilizando el modelo de regresión matemática que permitió obtener valores en partes por millón (ppm) a partir del valor de la resistencia leída en cada uno de los sensores.

A continuación, se describe el proceso de calibración de estos sensores.

a. Sensores Utilizados:

MQ-5: Sensor de gases inflamables (GLP, metano, hidrógeno).

MQ-135: Sensor de calidad del aire (amoníaco, dióxido de carbono, tolueno, alcohol, humo, etc.).

b. Modelo matemático

Para obtener el modelo matemático, en primer lugar, se digitalizaron los datos del grafico por medio del software WebPlotDigitizer y se exportaron en una tabla de Excel.

En la figura 16, se muestra la obtención de los puntos de tipos de gases que pueden ser medidos por el sensor MQ-5. Estos puntos describen la relación de PPM utilizando los términos Rs/Ro, donde Rs es la resistencia interna del sensor en el tiempo y Ro es el valor de Rs en aire limpio.

En segundo lugar, con ayuda de Excel se grafican los datos por cada tipo de gas capaz de ser sensado por el MQ-5 y se obtiene la ecuación característica de cada uno de ellos. Por ejemplo, para el gas LPG del sensor MQ-5 se obtuvieron los siguientes datos, que se muestran en la figura 17.

Figura 16

Digitación de Datos del Sensor MQ-5 con WebPlotDigitizer

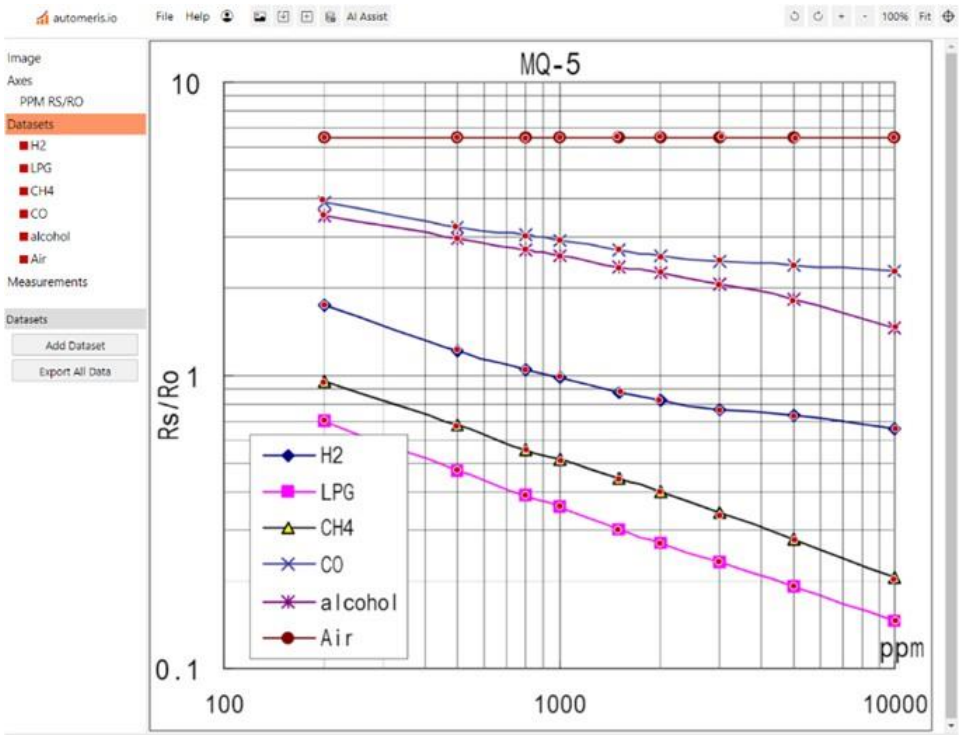
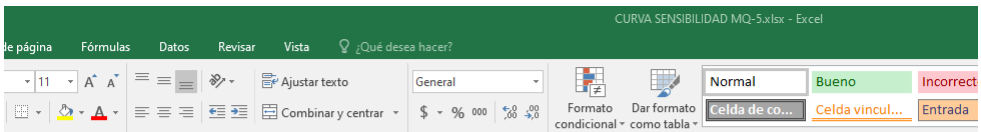
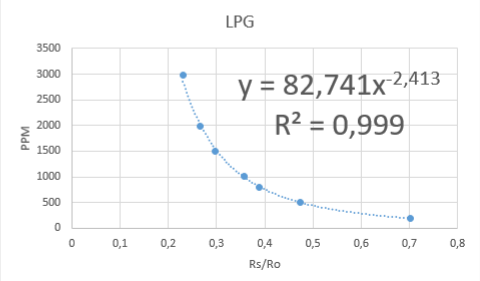


Figura 17

Obtención de Ecuación del Grafico Gas LPG del sensor MQ-5 en Excel.



LPG		PPM	Error	Error
RS/RO	PPM	Calculado		porcentual
0,70152502	198,893149	194,633955	4,25919447	0,02141449
0,47452489	495,571874	499,928408	4,3565342	0,00879092
0,3889807	791,350504	807,652167	16,3016628	0,0205998
0,35687653	1002,89324	994,248321	8,64492198	0,00861998
0,29644385	1494,18377	1555,69209	61,5083175	0,04116516
0,26662403	1994,68595	2009,21218	14,5262223	0,00728246
0,23045701	2989,05048	2856,21746	132,833025	0,04443987
0,19016767	4998,8576	4541,12996	457,72764	0,09156645
0,14492759	10057,9486	8747,08227	1310,86629	0,13033138
		Error promedio		0,04157895



Nótese que la ecuación que representa el grafico es $y = 82{,}741x^{-2{,}413}$, siguiendo un tipo de línea de tendencia potencial. A partir de ahí, se puede utilizar la regresión potencial como modelo matemático para modelar y predecir relaciones entre variables (gas y resistencia) en las que se observa un crecimiento o decrecimiento no lineal.

En una regresión potencial, el modelo matemático que se utiliza es de la forma: $y = ax^b$

Donde:

- y es la variable dependiente, es decir, PPM del gas a medir.
- x es la variable independiente, es decir, la relación Rs/Ro.
- a y b son constantes que deben ser estimadas para ajustar la curva potencial a los datos por cada gas a medir.

A continuación, se presenta la tabla 6, que muestra las constantes a y b de la expresión de regresión para los valores del gas H2, LPG, CH4, CO y Alcohol en el sensor MQ-5. Lo anterior, siguiendo el procedimiento descrito.

Tabla 4

Expresión de Regresión Para Medición de Gases Del MQ-5

GAS	A	B	ECUACIÓN
H2	1172,9	-3,865	$y = 1172{,}9x^{-3{,}865}$
LPG	82,741	-2,413	$y = 82{,}741x^{-2{,}413}$
CH4	179,8	-2,548	$y = 179{,}8x^{-2{,}548}$
CO	645556	-6,013	$y = 645556x^{-6{,}013}$
Alcohol	91440	-4,829	$y = 91440x^{-4{,}829}$

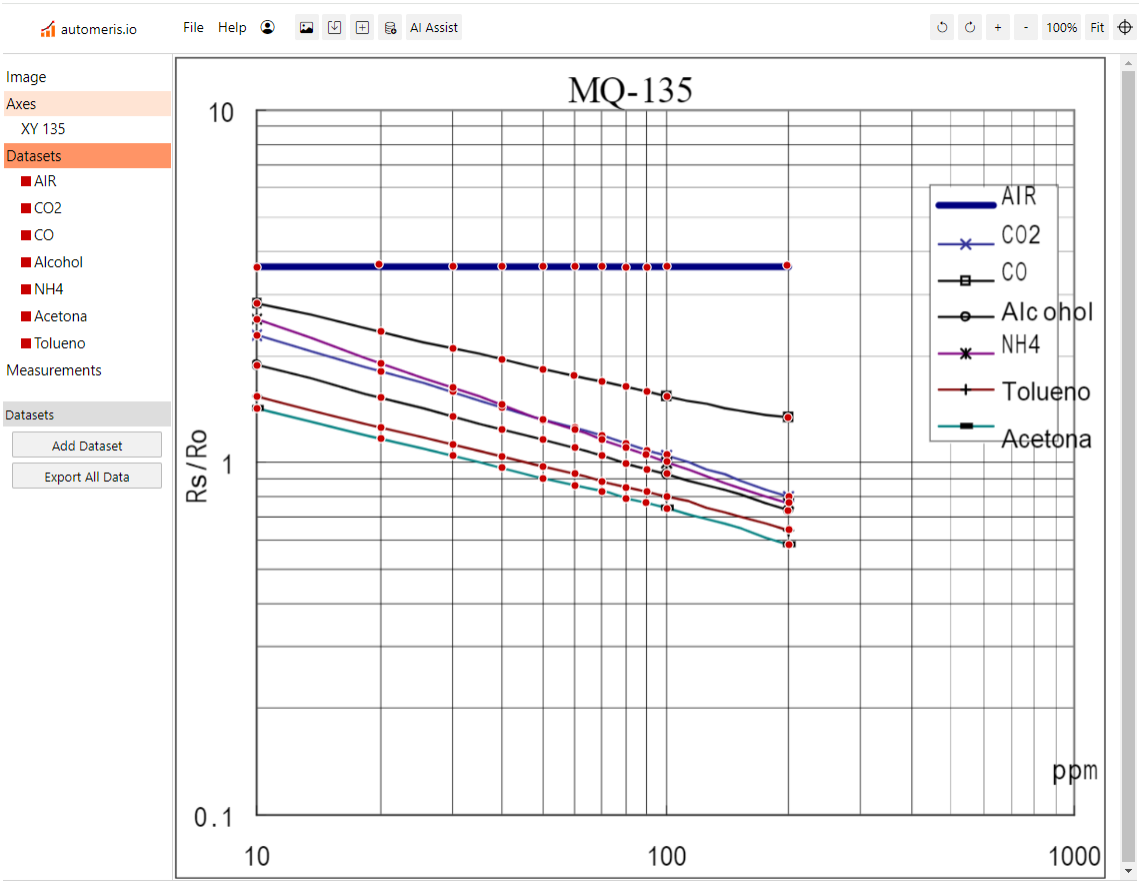
Para la calibración del MQ-135, se repite el procedimiento anterior. Primero, digitalizando la gráfica de sensibilidad del comportamiento del sensor con ayuda del software

WebPlotDigitizer, el cual permitirá obtener información sobre cómo medir cada tipo de gas y; segundo con ayuda de Excel, se obtiene el modelo matemático que permite predecir, de forma teórica, la relación de concentración de gas vs Rs/Ro.

En la figura 18, se muestra la digitalización de la gráfica de sensibilidad del MQ-135. Digitalizando los datos teóricos de la relación entre cada gas y la Rs/Ro.

Figura 18

Digitación de datos del sensor MQ-135 con WebPlotDigitizer



Asimismo, en la figura 19 se presenta las ecuaciones obtenidas con ayuda de Excel y en la tabla 5, el resumen de las constantes a y b de la expresión de regresión para los valores de los gases del sensor MQ-135.

Figura 19

Obtención de ecuaciones del MQ-135 en Excel para cada tipo de gas.

CO2	
PPM	Rs/Ro
10	2,2950675
20,116368	1,8135216
30,185902	1,5911259
39,788283	1,4330126
50,132851	1,3161888
60,042363	1,2572795
69,912244	1,1931761
80,039575	1,1323411
90,097422	1,0816603
100,84909	1,053725
200,59753	0,800588

CO	
PPM	RS/RO
10	2,829466429
20,116368	2,355912144
30,185902	2,107966168
39,788283	1,961614379
50,132851	1,837403526
59,704875	1,766684875
69,912244	1,698688069
80,039575	1,64402755
90,097422	1,591125901
100,84909	1,539926526
199,47001	1,342273169

ALCOHOL	
PPM	RS/RO
10	1,8861151
20,116368	1,5298861
30,185902	1,3510823
39,788283	1,2409378
50,132851	1,1623607
60,042363	1,1030968
69,912244	1,0468546
80,039575	0,9934799
90,097422	0,9552425
100,84909	0,930572
199,47001	0,7305272

NH4	
PPM	RS/RO
10	2,5482967
20,116368	1,910953
30,185902	1,6333083
39,788283	1,4614122
50,132851	1,3248268
60,042363	1,2409378
69,912244	1,1623607
80,039575	1,1030968
89,591	1,053725
100,84909	1,0065629
200,59753	0,7697747

TOLUENO	
PPM	RS/RO
10	1,5399265
20,116368	1,2572795
30,185902	1,1249581
39,788283	1,040029
50,132851	0,9741736
60,042363	0,930572
69,912244	0,8831261
80,039575	0,849136
90,097422	0,8272059
100,84909	0,800588
200,59753	0,6451476

ACETONA	
PPM	RS/RO
10	1,4236692
20,116368	1,1699891
30,185902	1,0468546
39,788283	0,9678219
50,132851	0,900628
60,042363	0,8603182
69,912244	0,8272059
80,039575	0,7901823
89,591	0,7697747
100,84909	0,7401473
200,59753	0,5848513

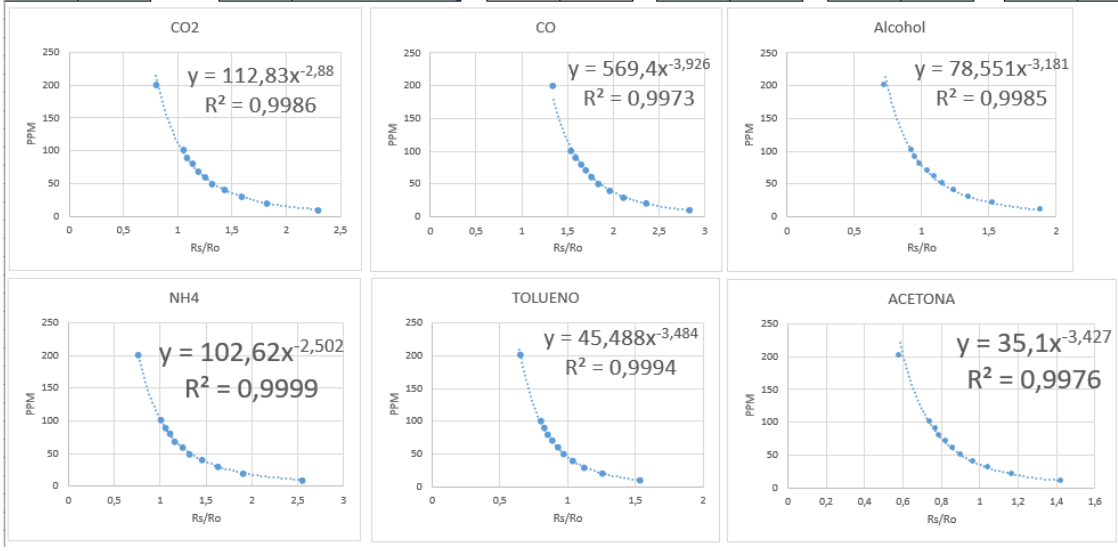


Tabla 5

Expresión de Regresión para Medición de Gases del MQ-135

GAS	A	B	ECUACIÓN
CO2	112,83	-2,88	y = 112,83x-2,88
CO	569,4	-3,926	y = 569,4x-3,926
ALCOHOL	78,551	-3,181	y = 78,551x-3,181
NH4	102,62	-2,502	y = 102,62x-2,502
TOLUENO	45,488	-3,488	y = 45,488x-3,488
ACETONA	35,1	-3,427	$y = 35,1x^{-3,427}$

Implementación Interfaz Remota IoT

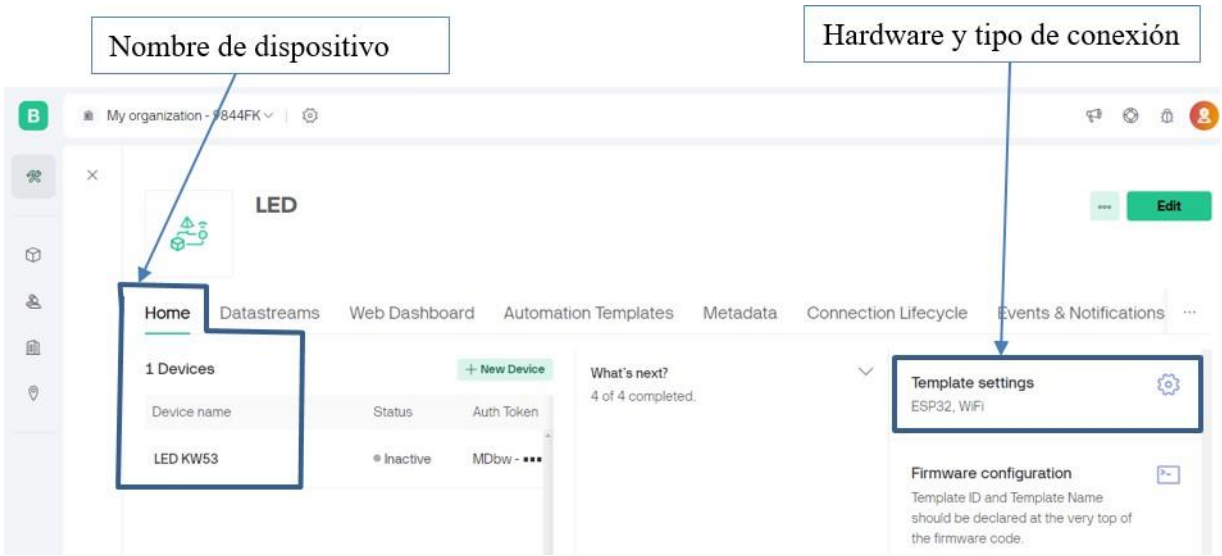
En el contexto del proyecto de monitoreo de calidad de aire y detección de gases para el hogar, Blynk IoT se utilizó para establecer la comunicación entre los sensores y la plataforma en la nube, permitiendo a los usuarios monitorear y recibir alertas sobre la calidad del aire, la temperatura y la presencia de gases, así como monitorear el sistema de manera remota desde dispositivos móviles o computadoras.

Para ello, se configura la interfaz en tres pasos fundamentales:

Creación de Dispositivo: En este punto se define el nombre del dispositivo a utilizar, Hardware y tipo de conexión. Utilizando para ello, el nombre de LED, el Hardware ESP32 y el tipo de conexión WiFi. En la figura 20, se muestra el paso inicial.

Figura 20

Creación de dispositivo en Blynk IoT

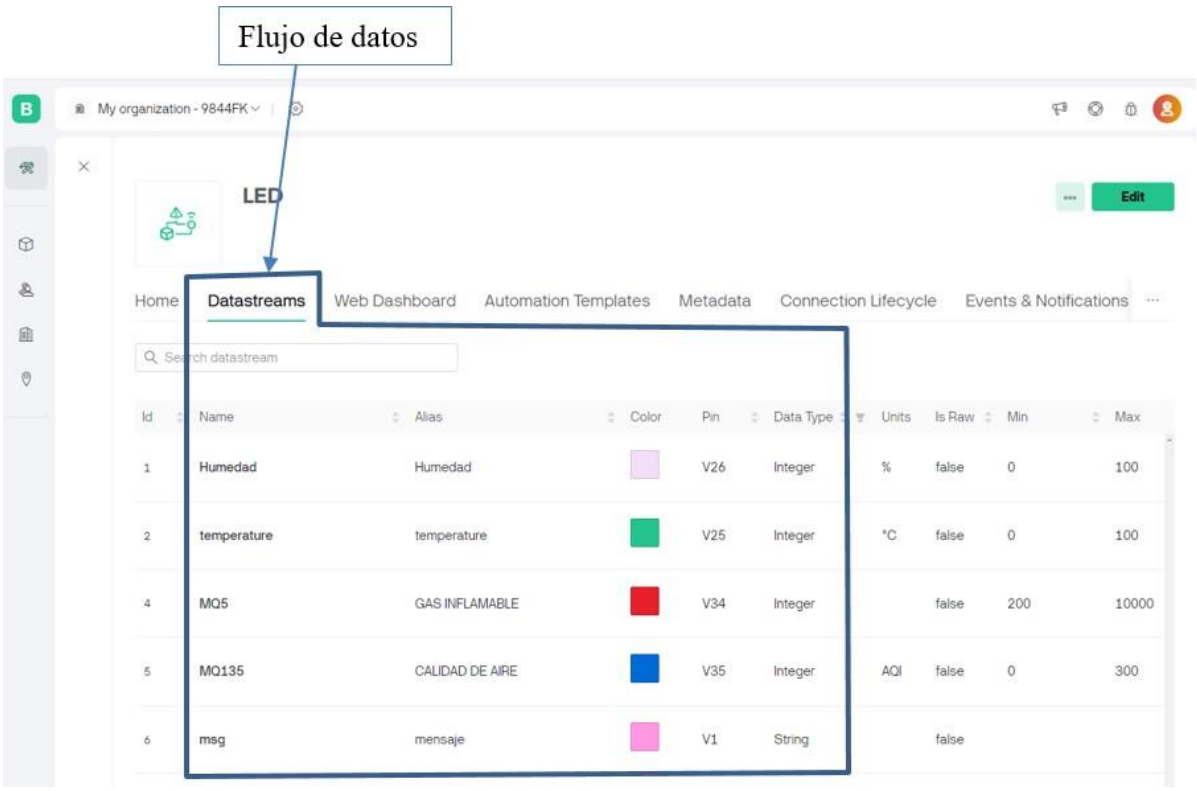


Definición de Variables o Flujo de Datos. estos son canales para datos con marca de tiempo transmitidos entre el dispositivo y la nube (Blynk, 2024). Aquí se asocian los datos de los

sensores a una variable, que a su vez se relaciona a un pin virtual del canal de datos de la nube de Blynk. Para el prototipo, se crearon cinco variables que corresponden a los datos de los sensores de calidad del aire, la temperatura y la presencia de gases. Como se observa en la figura 21.

Figura 21

Creación de Variables del Proceso



Nota. Variables del proceso asociadas a pines virtuales de Blynk IoT, donde cada valor recibe una marca de tiempo automáticamente y se almacena en la base de datos de Blynk.Cloud.

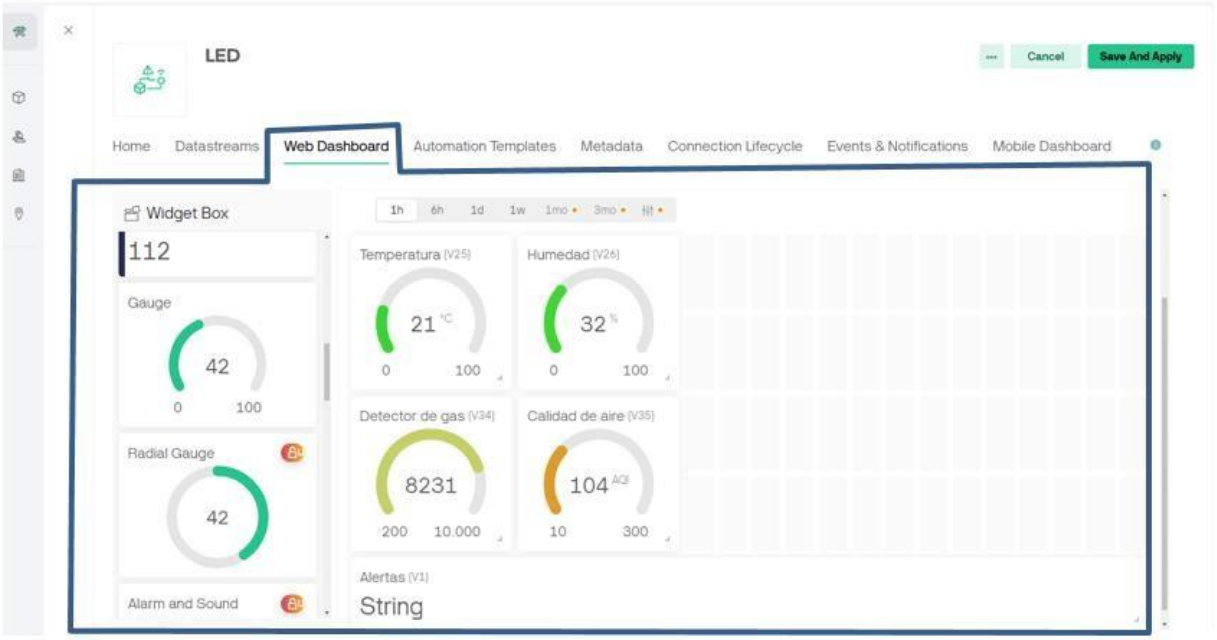
Creación de Panel

En este apartado, se crea el panel para la visualización de los datos procesados de los sensores y realizar el monitoreo del prototipo de manera remota desde los dispositivos móviles o computadoras.

Panel Web. Para el proyecto se utilizan cuatro indicadores tipo circular para mostrar los valores de temperatura, humedad, Gas inflamable y Calidad de aire; además, de uno tipo String para mostrar las alertas cuando dichos valores superen las condiciones normales. Cada uno de estos indicadores está asociado a una variable y un pin virtual creada previamente. En la figura 22, se observa el diseño del panel para el proyecto.

Figura 22

Panel del Prototipo MoniGas



Nota. Panel para visualización de datos de los sensores para el monitoreo remoto en computadoras.

Panel para Dispositivos Móviles. Para el panel móvil se utilizan los mismos flujos de datos que en panel web, para acceder a cada uno de los datos. Como se podrá observar en la figura 23.

Figura 23

Panel para Dispositivos Móviles

Nota. Panel para visualización de datos de los sensores para el monitoreo remoto en dispositivos móviles.

Configuración E Integración de CallMeBot

El prototipo de monitoreo de gases para el hogar utilizó CallMeBot para enviar notificaciones inmediatas a los usuarios a través de WhatsApp. La integración y uso de CallMeBot se describen a continuación:

Configuración. Se configura la API de CallMeBot para poder enviar mensajes desde el sistema de detección a WhatsApp. Para ello, se debe registrar el número de teléfono y obtener las credenciales necesarias, como una API key y una URL, para hacer las peticiones que permitan el envío de mensajes. Ver figura 24.

Figura 24

Configuración de API CallMeBot

Nota. Se muestran las credenciales necesarias para la configuración de notificaciones por WhatsApp en CallMeBOT, incluyendo la API key y la URL de solicitud.

Integración con la Tarjeta ESP32 e Interfaz IoT. La funcionalidad de envío de mensajes de CallMeBOT se integra con la plataforma Blynk IoT para asegurar que todas las alertas relevantes sean comunicadas de manera efectiva y en tiempo real. En el código de programación de la tarjeta ESP32, que se programará a través de Arduino IDE, se incluirán tres funciones: la primera se encargará de conectar la tarjeta ESP32 a la red WiFi; la segunda, de

preparar y codificar el mensaje para la URL de CallMeBot; y la tercera, de crear el cliente HTTP y enviar las peticiones a la API CallMeBot. Ver figura 25.

Figura 25

Integración de CallMeBot con ESP32

```
void conectNetwork()
{
 WiFi.begin(ssid, password);
 Serial.println("Intentando Conectar a la RED WiFi");
 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
  }
  Serial.println("");
  Serial.print("Conectado a la Red WiFi! ");
  Serial.println(WiFi.localIP());
}


void sendMessage(String message)
  {
  // Data to send with HTTP POST
  String url = "https://api.callmebot.com/whatsapp.php?phone=" + phoneNumber + "&apikey=" + apiKey + "&text=" + urlEncode(message);
  postData(url);

   }

void postData(String url) //funcion para hacer la llada a la API con post
{
  int httpCode;
  HTTPClient http;   // se declara un objeto  HTTPClient
  http.begin(url);
  httpCode = http.POST(url);
  if (httpCode == 200) {  // verificamos respuesta
    Serial.println("Mensaje enviado Correctamente.");
  } else {
    Serial.print("Error: ");
    Serial.println(httpCode);
  }
 // Free resources
  http.end();
}
```

Nota. La figura ilustra el flujo de integración entre la tarjeta ESP32 y CallMeBot. Se destacan las tres funciones principales en el código: conexión a la red Wifi, preparación y codificación del mensaje para CallMeBot, y la creación del cliente HTTP para enviar las peticiones a la API de CallMeBot.

Programación Tarjeta ESP32

En el apéndice A, se presenta el código que da funcionalidad a la tarjeta ESP32 e integrara el sistema con los sensores, la interfaz remota IoT y CallMeBot.

Este programa permitirá monitorear y alertar en tiempo real la calidad del aire y la presencia de gases peligrosos en el hogar. Cuando uno de los sensores detecte una condición

fuera de los parámetros normales, como una temperatura alta, humedad baja o la presencia de gases peligrosos, la tarjeta ESP32 procesará esta información y enviará una alerta a través de Blynk IoT y un mensaje de WhatsApp por CallMeBot. Además, de una alerta sonora y mensajes continuos en la pantalla LCD del prototipo.

Resultados

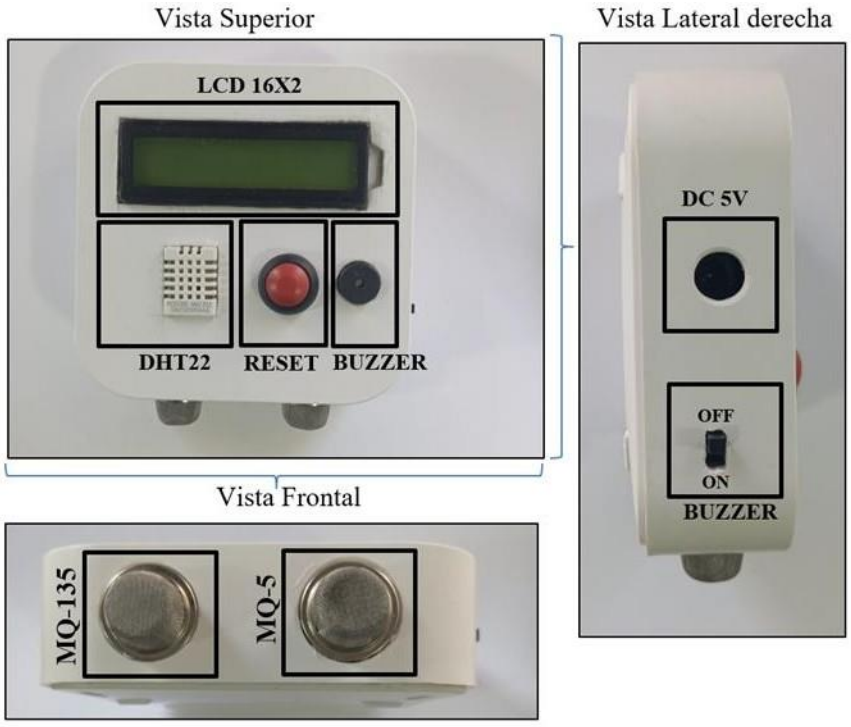
A continuación, se presenta los resultados obtenidos en el diseño e implementación del proyecto MoniGas: Prototipo de monitoreo remoto de gases para el hogar.

Construcción del Prototipo

El prototipo fue ensamblado con éxito en una placa PCB diseñada específicamente para integrar los componentes seleccionados, como se observa en la figura 26:

Figura 26

Prototipo de monitoreo de condiciones del hogar



Los principales dispositivos utilizados fueron:

- ESP32 como unidad central de monitoreo.
- Sensores MQ-5 y MQ-135 para la detección de gases inflamables y contaminantes.
- Sensor DHT22 para medir temperatura y humedad.
- Pantalla LCD 16x2 para la visualización local de datos en tiempo real.
- Buzzer y transistor 2N2222 para activar la alarma sonora.
- Fuente de alimentación a 5V DC

El montaje se realizó siguiendo el esquema de conexión previamente definido, logrando la correcta integración de los sensores y dispositivos de salida, así como una gestión óptima de la energía.

Monitorización Local en Tiempo Real

El prototipo mostró localmente en la pantalla LCD, los datos de: Temperatura (T), Humedad (H), Calidad de Aire (C) y Gas Inflamable (G) cuando las condiciones del hogar son normales, es decir, las variables se encuentran dentro de los parámetros. Además, en la pantalla LCD se visualizarán las alarmas de temperatura, calidad de aire y gases inflamables cuando estos superen los límites y simultáneamente se activará el zumbador, indicando la alerta sonora. Dichas visualizaciones se podrán ver en el apartado de pruebas al sistema. Ver figura 27.

Monitorización Remota en Tiempo Real

El prototipo se conectó exitosamente a la plataforma Blynk IoT, lo que permitió a los usuarios monitorear los datos de los sensores en tiempo real desde un dispositivo móvil. Los datos se actualizaron de manera constante, mostrando los valores de temperatura, humedad, niveles de gases y calidad de aire en la app de Blynk, lo cual facilitó la monitorización remota y visualización de alertas. Ver figura 28.

Figura 27

Monitorización local

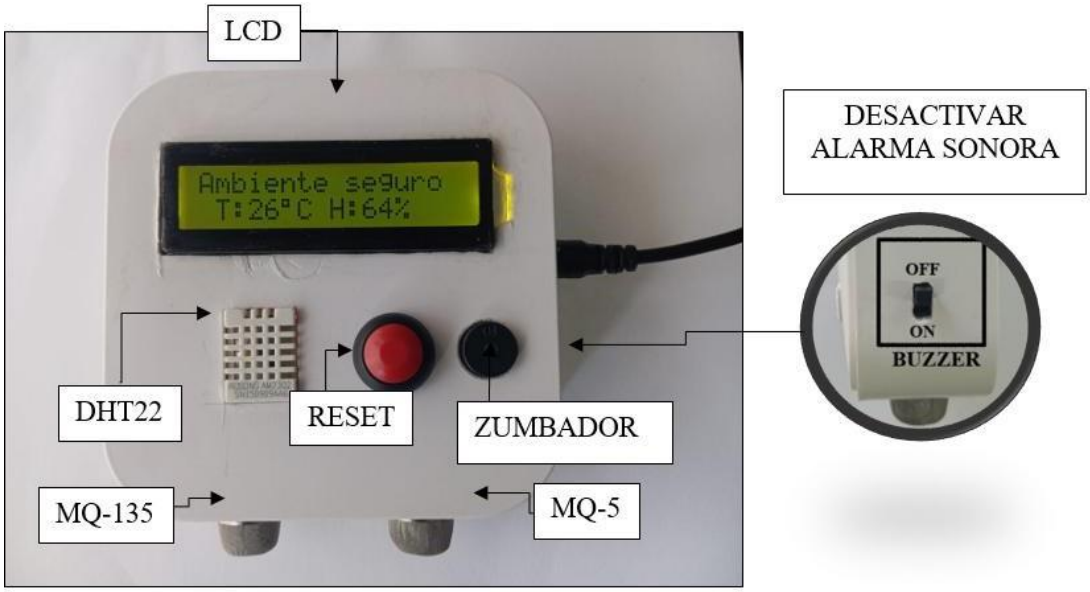
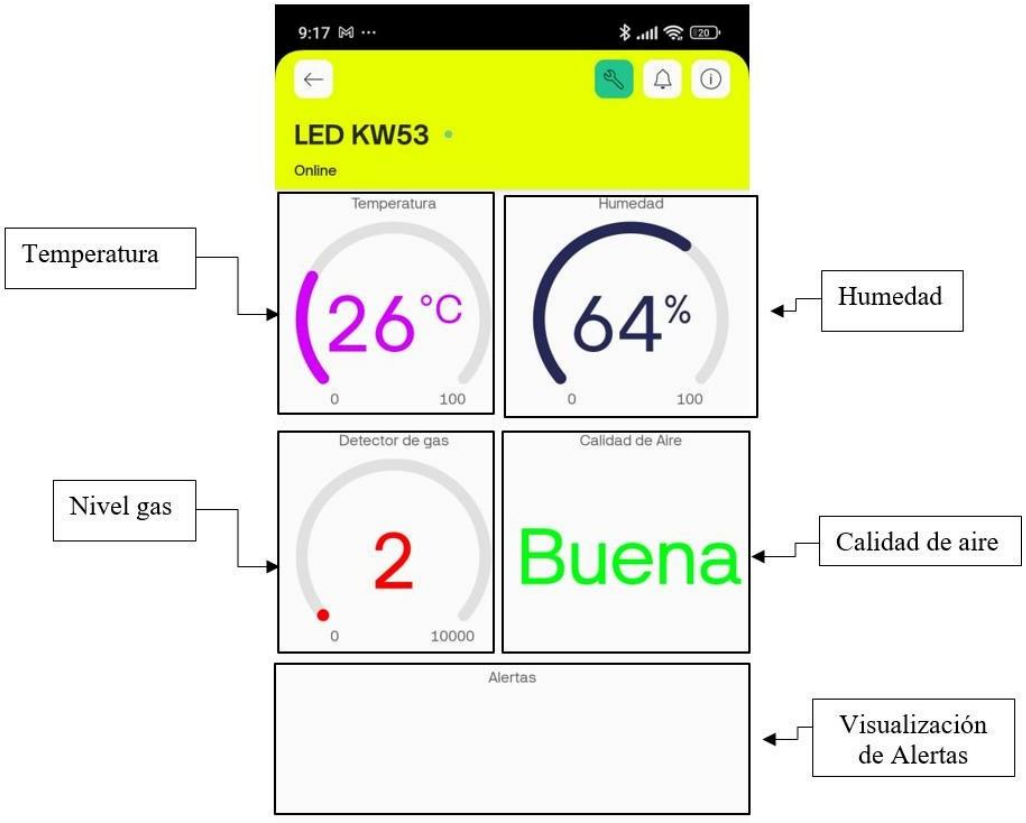


Figura 28

Monitorización Remota



Pruebas al sistema – Alertas

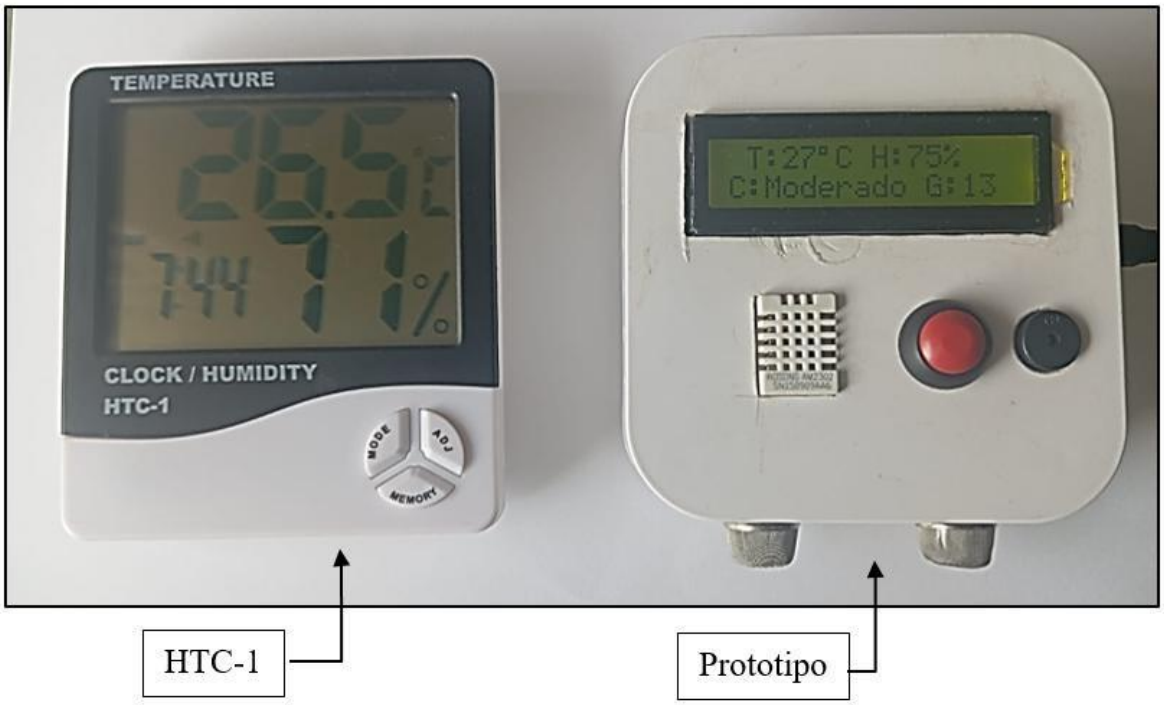
Una vez completada la fase de implementación del prototipo, se procedió a realizar las pruebas de funcionamiento. Estas pruebas se llevaron a cabo exponiendo los sensores, de manera controlada, a fuentes de gases inflamables, humo y al ambiente del hogar para evaluar su comportamiento.

Caso 1: Monitoreo de Temperatura y Humedad

En esta prueba se analizaron los valores de temperatura y humedad recogidos por el prototipo, los cuales se compararon con los datos obtenidos mediante un higrómetro digital de referencia (HTC-1). Este es un dispositivo ampliamente utilizado que mide la temperatura y humedad con una precisión de ±1°C y ±5% de humedad relativa, respectivamente, además de contar con una pantalla LCD que facilita la visualización de los parámetros en tiempo real. Ver figura 29.

Figura 29

Equipo de referencia HTC-1 vs Prototipo



El análisis se dividió en dos secciones: temperatura y humedad, evaluando la precisión, tendencia, y desviación entre ambos dispositivos.

Temperatura (°C)

Resumen de los Datos

- El rango de las temperaturas medidas por el prototipo osciló entre 23°C y 25°C.
- El higrómetro de referencia mostró valores ligeramente superiores, variando entre 23,5°C y 25,7°C.

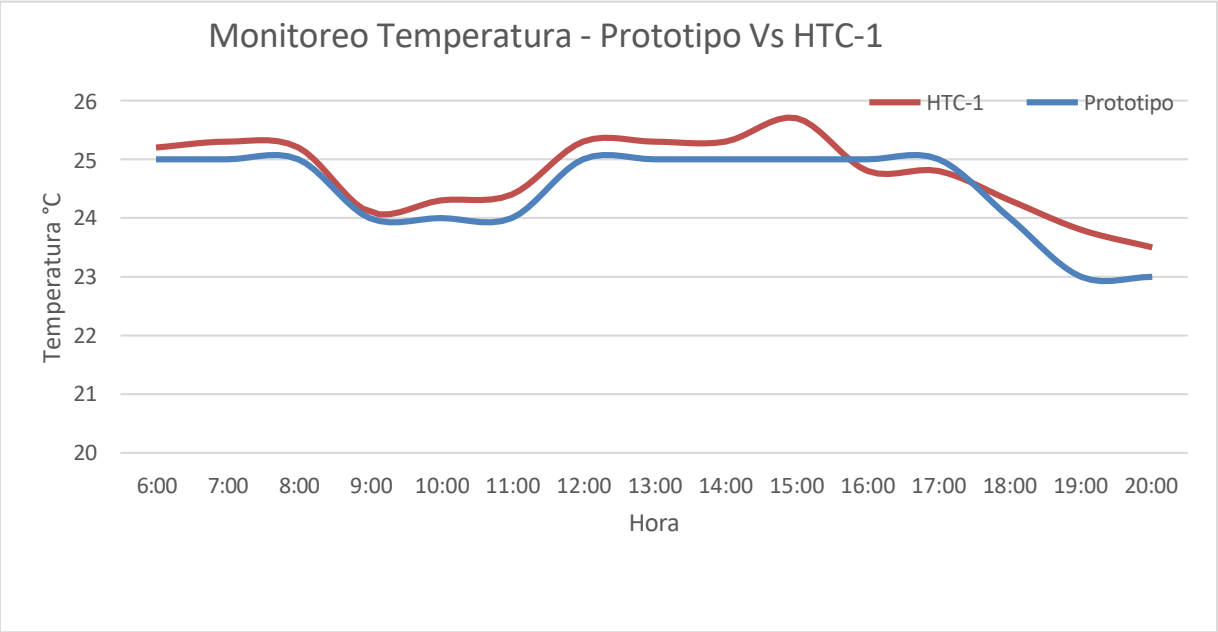
Análisis de Precisión y Consistencia

- En general, el prototipo ofreció mediciones muy cercanas a las del HTC-1, con una diferencia máxima de ±0,7°C (15:00).
- El prototipo y el higrómetro mantienen una tendencia similar durante todo el día, con las mayores diferencias observadas en las horas de la tarde.
- La diferencia promedio entre el prototipo y el HTC-1 es de 0,2°C, lo que indica una muy buena correlación entre ambos dispositivos. Esto sugiere que el prototipo puede ser confiable para la monitorización de la temperatura en el hogar.

En la figura 30, se muestra el comparativo de las mediciones de temperatura entre el prototipo y el Higrómetro digital HTC-1. En esta, se observa una diferencia de +/- 0.5 °C aproximados entre las dos mediciones. Comprobando el correcto funcionamiento del prototipo.

Figura 30

Comparativo – Temperatura



El prototipo se comportó de manera consistente y cercano al dispositivo de referencia. Las diferencias de temperatura observadas son mínimas y probablemente no afecten el uso práctico del prototipo en la monitorización del ambiente.

2. Humedad (%)

Resumen de los Datos

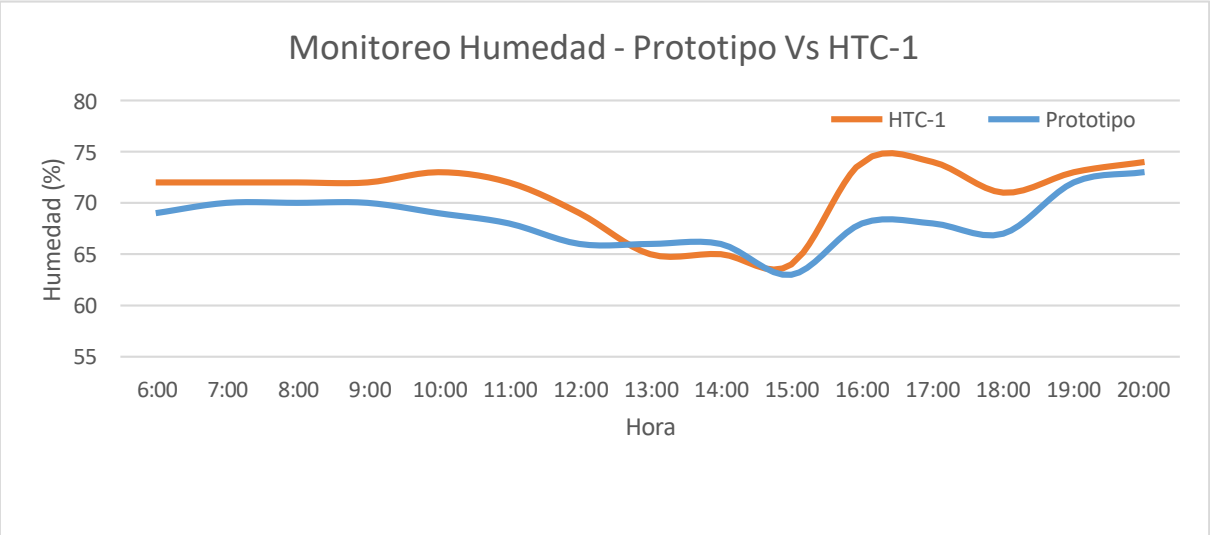
- El rango de las mediciones registradas por el prototipo varió entre 63% y 73%.
- El HTC-1 mostró un rango de humedad entre 64% y 74%, con ligeras discrepancias en ciertas horas.

Análisis de Precisión y Consistencia

- Se observó diferencias más notables entre las mediciones de humedad en comparación con las de temperatura.
- En general, el prototipo registró valores de humedad ligeramente inferiores a los del higrómetro, con una desviación máxima de -3% en algunas horas (6:00, 9:00, 16:00, y 17:00).
- La diferencia promedio de humedad entre el prototipo y el HTC-1 es de 1,73%, lo que sigue siendo aceptable para un dispositivo de monitoreo en tiempo real, aunque algo menos preciso que en la medición de temperatura

Figura 31

Comparativa humedad



Aunque el prototipo presentó una ligera subestimación de la humedad en comparación con el HTC-1, el margen de error es pequeño y aceptable. Es importante tener en cuenta que los sensores de humedad tienden a ser más sensibles a factores ambientales como la ventilación o la radiación directa, lo que puede haber causado parte de la variación observada.

Caso 2: Prueba de Gas Inflamable

En esta prueba, se utilizó un encendedor con una mezcla de gases inflamables comprimidos, principalmente butano y propano. El gas se liberó a una distancia de 5 centímetros del prototipo durante 30 segundos en distintos momentos del día, con el objetivo de evaluar la respuesta del sensor MQ-5.

El sensor MQ-5 mostró una correlación directa entre la concentración de gas y la activación de las alarmas locales, confirmando la eficacia del sistema de detección. La visualización local en la pantalla LCD y la visualización remota en la plataforma Blynk IoT son visibles en la figura 32, respectivamente.

Figura 32

Alerta de Detección de Gas Inflamable

a) Visualización Local

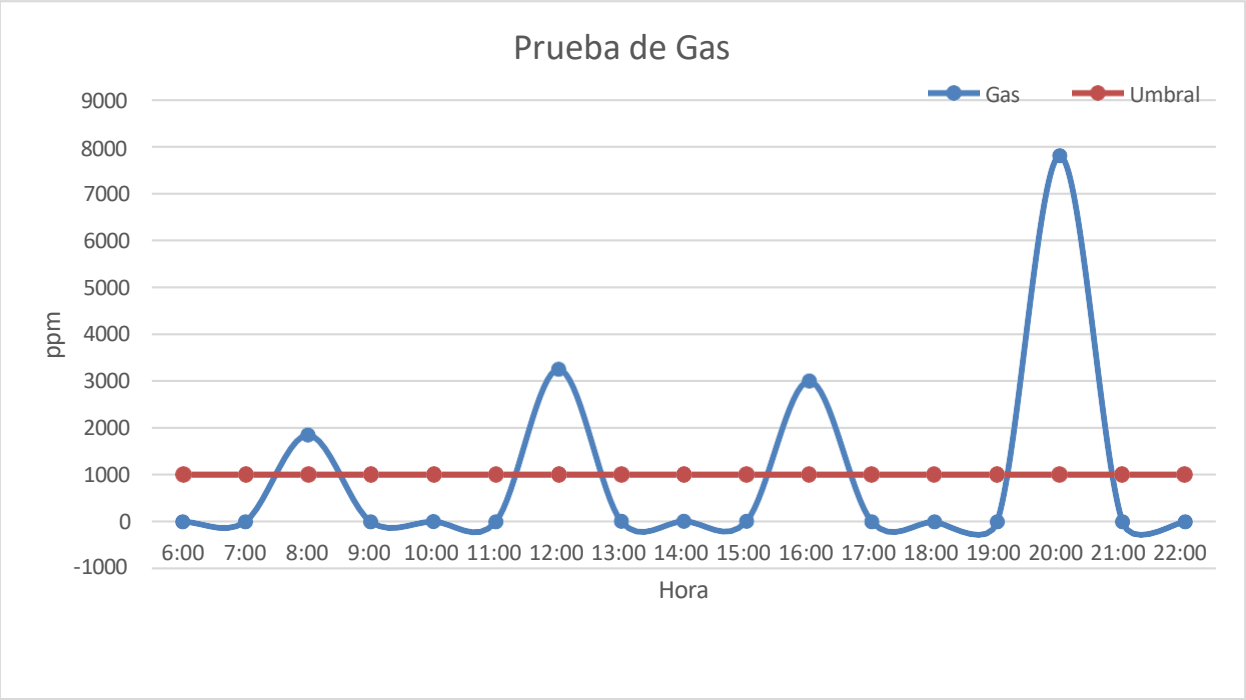
b) Visualización remota en Blynk IoT

Adicionalmente, cuando la concentración de gas alcanzó los 1000 ppm o mayor, el sistema activó la alarma remota, enviando notificaciones a través de Blynk IoT y un mensaje de alerta a WhatsApp.

En la figura 33, el eje X muestra las horas de las pruebas (8:00, 12:00, 16:00, y 20:00), mientras que el eje Y representa las concentraciones de gas en ppm. El sensor respondió rápida y eficazmente a las exposiciones, como se observa en los picos pronunciados de la señal.

Figura 33

Gráfica de Concentración de Gas Inflamable (Ppm) Durante las Pruebas

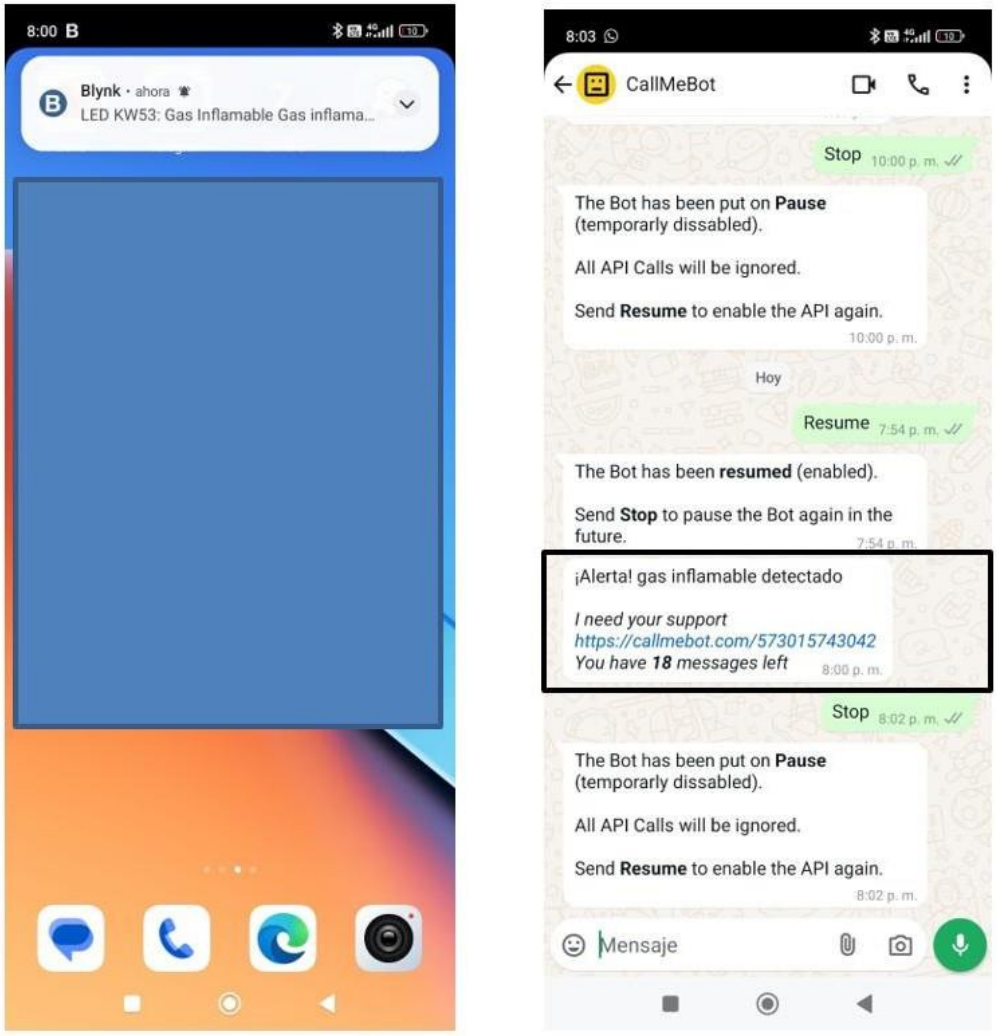


La señal del sensor (línea azul) experimentó picos pronunciados al momento de la exposición, alcanzando valores máximos de hasta 7500 ppm. Posteriormente, la señal decayó rápidamente a 0 debido a la ventilación del área y la calibración del sensor. El tiempo de

respuesta promedio del sistema, desde la detección del gas hasta el envío de la notificación, fue de hasta 5 segundos aproximadamente. Ver figura 34.

Figura 34

Notificaciones Remotas – Alertas



a) Notificación Push Blynk b) Notificación WhatsApp

Caso 3: Monitoreo de la Calidad del Aire

En esta prueba se analizaron los datos de calidad de aire recogidos por el prototipo a través del sensor MQ-135 calibrado y caracterizado de acuerdo a la curva de sensibilidad de la

ficha técnica del sensor (ver figura 18) , los cuales se compararon con los datos publicados por el Sistema de Información Ambiental de Colombia – SIAC, en su módulo temático de calidad de aire, que agrupa las diversas estaciones de monitoreo de Sistemas de Vigilancia de Calidad del Aire –SVCA del país; entre ella, la que es de interés particular para el proyecto debido a su ubicación, es el sistema de Vigilancia de Calidad del Aire de Cali – SVCAC.

A continuación, se muestran los resultados obtenidos de la prueba comparativa de monitoreo de la calidad del aire entre el prototipo desarrollado y el Sistema de Información Ambiental de Colombia (SIAC) como referencia. Los datos recopilados corresponden a un monitoreo durante un día completo (6:00 a.m. a 8:00 p.m.), expresados en partes por millón (ppm), una medida estándar utilizada para evaluar la concentración de gases y partículas contaminantes en el aire. Ver figura 35, 36 y 37.

Figura 35

Monitoreo de la Calidad del Aire

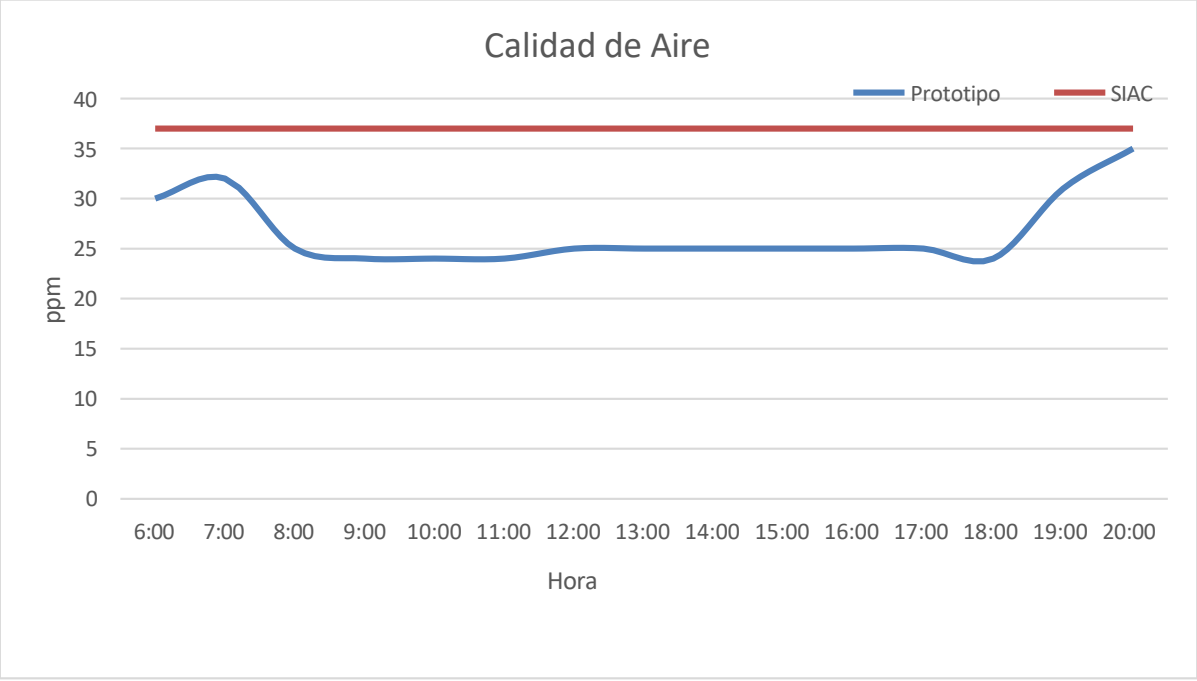
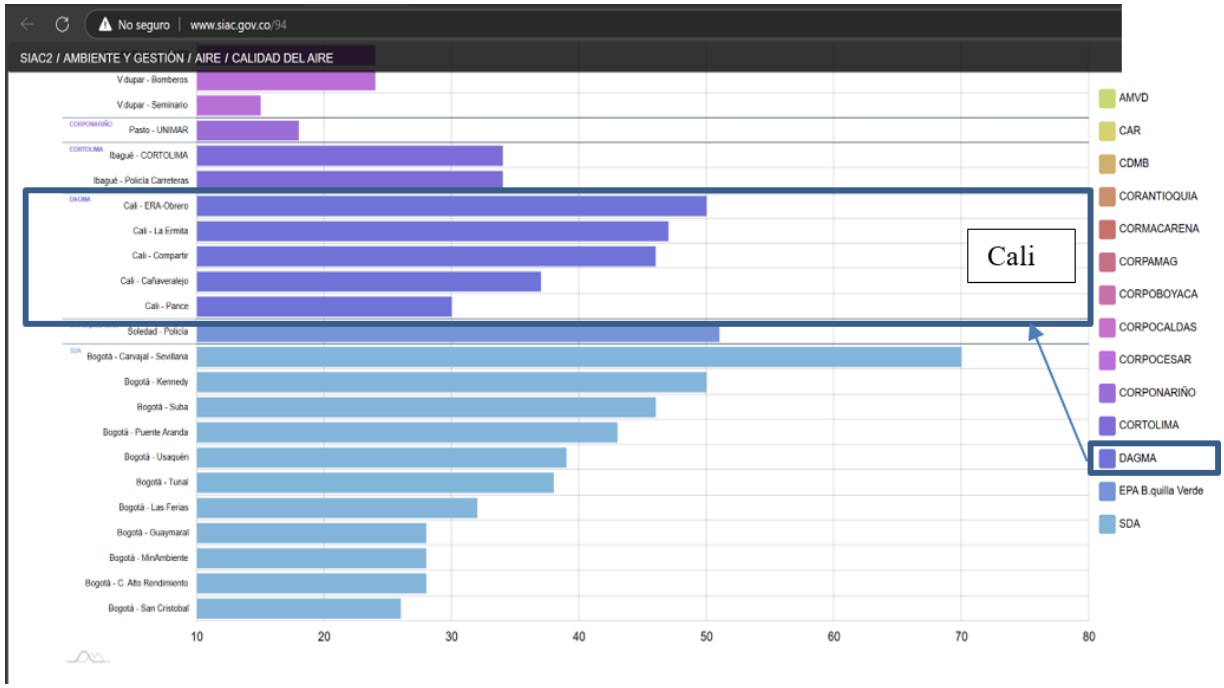


Figura 36

Monitoreo de la Calidad del Aire por SIAC



Nota. Del Sistema de Información Ambiental de Colombia – SIAC, 2024. Calidad del Aire. [en línea]. http://www.siac.gov.co/94

Figura 37

Monitoreo de la Calidad del Aire Local y Remoto

a) Remoto

b) Local

Rango	Color	Estado
0 - 50	Verde	Buena
51 - 100	Amarillo	Aceptable
101 - 150	Naranja	Dañina a la salud de grupos sensibles
151 - 200	Rojo	Dañina para la salud
201 - 300	Púrpura	Muy dañina para la salud
301 - 500	Marrón	Peligrosa

c) Clasificación MADS

Análisis del Comportamiento del Prototipo:

- El prototipo mostró lecturas que variaron entre 24 ppm y 35 ppm a lo largo del día. Lo que indica que la calidad del aire del hogar se encuentra en la clasificación de bueno.
- Las mediciones más bajas se registraron entre las 9:00 a.m. y 6:00 p.m., con valores estables de 24 – 25 ppm.
- En horas de la mañana y noche (6:00 a.m., 7:00 a.m., 19:00 y 20:00), se observó un

ligero incremento, con valores de hasta 35 ppm al final del día. Esto se debe que durante esos horarios son utilizados para preparar los alimentos y mayor concurrencia de personas al interior de la vivienda objeto de análisis.

Alerta Local y Remota de Calidad de Aire

Se realizaron pruebas para verificar el correcto funcionamiento del sistema de alertas locales y remotas del prototipo, específicamente al detectar niveles ambientales potencialmente peligrosos en la calidad del aire. Para este experimento, el sensor MQ-135 fue expuesto a humo (pequeñas partículas gaseosas y químicas que resultan de la combustión incompleta de un material combustible) durante un periodo de 5 minutos, con el objetivo de comprobar la activación de las alertas diseñadas cuando se detecten gases contaminantes en el ambiente del hogar.

Resultados de la Prueba:

Alerta Local: Durante la exposición al humo, el prototipo activó su sistema de alerta local, mostrando en la pantalla LCD el mensaje "CA DAÑINO" (Calidad de Aire Dañina), como

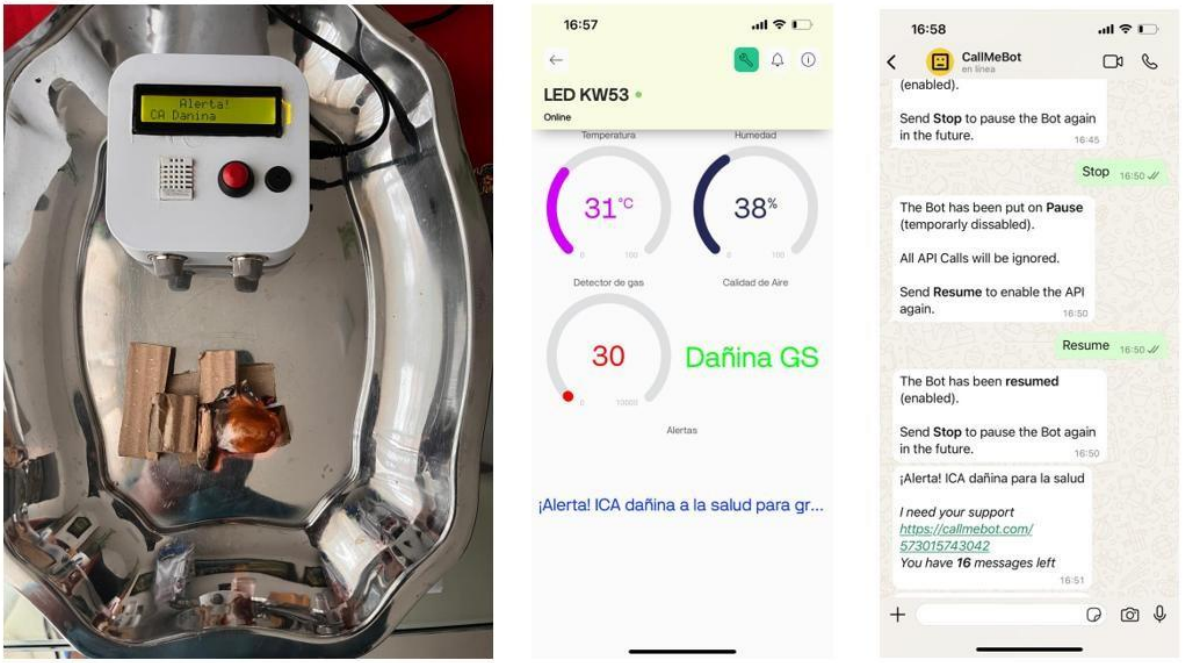
se observa en la Figura 38a. Este mensaje indica un nivel de contaminación que podría afectar la salud de las personas en el hogar.

Alerta Remota a través de Blynk IoT: Simultáneamente, la alerta fue enviada de forma remota a través de la aplicación móvil Blynk IoT, donde se visualizó un mensaje de advertencia en la interfaz del usuario. Esto permitió el monitoreo en tiempo real desde cualquier lugar, como se muestra en la Figura 38b.

Notificación a WhatsApp: Adicionalmente, el sistema envió una notificación inmediata a través de WhatsApp utilizando la integración con la API de CallMeBot. El mensaje contenía la alerta con la descripción del evento detectado, asegurando que los usuarios reciban información oportuna, como se presenta en la Figura 38c.

Figura 38

Alertas Calidad de Aire



a) Local b) Remota c) Notificación WhatsApp

Conclusiones

El prototipo de detección de incendios y gases peligrosos para el hogar demostró ser eficiente, preciso y fiable. Los datos se recopilaron y transmitieron en tiempo real, y las alarmas y notificaciones funcionaron dentro de los parámetros esperados.

La integración con la plataforma Blynk IoT y el servicio de notificaciones CallMeBot se llevó a cabo de manera satisfactoria, permitiendo una interacción remota y notificaciones instantáneas al usuario en situaciones de riesgo.

El sistema está preparado para ser implementado en hogares como una herramienta de seguridad adicional, proporcionando una capa extra de protección contra incendios y la exposición a gases peligrosos.

La alerta visual y auditiva en el prototipo permite a los habitantes del hogar reaccionar de manera inmediata ante la detección de gases dañinos, gracias al mensaje en la pantalla LCD y el zumbador.

Las alertas enviadas por Blynk IoT y WhatsApp garantizan que los usuarios sean notificados, incluso si no están físicamente presentes en el hogar, aumentando la seguridad y la capacidad de respuesta ante emergencias.

Las pruebas validaron la funcionalidad del sistema de alertas locales y remotas implementado en el prototipo. Las notificaciones claras y rápidas aseguran una respuesta efectiva ante condiciones ambientales potencialmente peligrosas, reforzando la utilidad del prototipo como herramienta de monitoreo y seguridad ambiental en el hogar.

Referencias Bibliográficas

Universidad Nacional Abierta y a Distancia - UNAD. (2017). Líneas de investigación de la Escuela de Ciencias Básicas, Tecnología e Ingeniería ECBTI. https://academia.unad.edu.co/investigacion-y-productividad-ecbti/lineas

Universidad Nacional Abierta y a Distancia - UNAD. (2017). Grupos de Investigación – ECBTI. https://academia.unad.edu.co/investigacion-y-productividad-ecbti/grupos

Organización mundial de la salud - OMS. (2022). Contaminación del aire doméstico y salud. https://www.who.int/es/news-room/fact-sheets/detail/household-air-pollution-and-health

Organización mundial de la salud - OMS. (2021). Directrices mundiales de la OMS sobre la calidad del aire. https://www.who.int/es/news-room/questions-and-answers/item/who-global-air-quality-guidelines

Cuerpo Oficial de Bomberos de Cali. (2023). Estadísticas de emergencia. https://bomberoscali.org/estadisticas-de-emergencia/

Arduino. (2023). Documentación de Arduino. https://docs.arduino.cc/?_gl=1*1xloapi*_ga*NTIzOTY2NTAzLjE2Nzk2MTk5MTg.*_ga_NEXN8H46L5*MTY4MDIyODQ2MC4zLjEuMTY4MDIzMTQzNS4wLjAuMA.

Castro Domínguez, Alberto. (2013). Sistema de control de temperatura a través de Arduino y la tecnología GPRS/GSM. Madrid, España: Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación, p. 103-111.

Openwebinars. (2015). Tutorial Arduino: IDE Arduino. https://openwebinars.net/tutorial-arduino-ide-arduino/

Campos, E. (2016). ¿Cuál Arduino comprar?. http://5hertz.com/tutoriales/?p=571

Raspberry Pi Foundation. (2016). Products. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

chipKIT. (2016). ChipKit uno32. [en línea]. http://chipkit.net/wpcproduct/chipkit-uno32/

First Alert. (2022). Onelink Safe & Sound. https://www.firstalert.com/onelink/

Google Nest. (2022). Nest Protect. https://store.google.com/us/product/nest_protect_2nd_gen

Kidde. (2022). Kidde Wireless Smoke & Carbon Monoxide Alarm. https://www.kidde.com/home-safety/en/us/products/fire-safety/combination-smoke-co-alarms/wireless/

SmartThings. (2022). SmartThings. https://www.smartthings.com/

González, J. C., & González, V. (2018). Desarrollo de un sistema de detección y control de incendios con notificación mediante mensajes de texto. Revista de Investigación Académica, 28, 1-10. https://doi.org/10.18273/revinv.v28n1-2017001.

Nagy, A. S., Polanco Risquet, A., Martínez de la Cotera, O. L., & Carralero Ibargollen, O. (2020). Medición simultánea de gases con sensores MQ. Ingeniería Electrónica, Automática y Comunicaciones, 41(1), 34–43. https://search-ebscohost-com.bibliotecavirtual.unad.edu.co/login.aspx?direct=true&db=asn&AN=142366625&lang=es&site=ehost-live.

Circuits-elec. (2023). Módulo sensor de temperatura y humedad (DHT22). https://circuits-elec.com/products/temperature-humidity-sensor-module-dht22

Lifeder. (2021). Temperatura. https://www.lifeder.com/temperatura/

El Servicio Nacional de Manejo del Fuego-SNMF. (s.f). ¿Cuáles son las variables y qué factores las afectan?. https://www.argentina.gob.ar/ambiente/fuego/conocemas/variables

Grajales B. E. (2016). Tecnología celular. https://sx-de-tx.wikispaces.com/Tecnologia+Celular

Moviles.info. (2023). Operadores en Colombia. https://moviles.info/frecuencias/colombia/

Cifuentes, R. & García, A. (2020). Sistema de monitoreo y alerta de incendios basado en Arduino. Revista de Tecnología - Journal of Technology, 19, 1-8. Recuperado de http://revistas.utp.ac.pa/index.php/tecnologia/article/view/2237/1981

Díaz, J., González, G., & Pérez, A. (2018). Diseño de un sistema de detección y monitoreo remoto de incendios basado en Arduino. Revista Internacional de Investigación en Ciencia y Tecnología, 11(1), 15-21. Recuperado de http://www.ute.edu.ec/revistas/revistas/riict/Vol11-1/05%20Diaz,%20Gonzalez,%20Perez.pdf

Pérez, J., Sánchez, V., & Reyes, E. (2021). Sistema de detección de incendios basado en Arduino y sensores de gas, temperatura y humo. Revista de Investigación Científica y Tecnológica, 4(1), 23-30. Recuperado de https://revistainvestigacion.uteq.edu.ec/index.php/ricyt/article/view/311/230

Ministerio de Ambiente y Desarrollo Sostenible – MADS. (01 de noviembre de 2017). Resolución 2254 del 2017. Norma de calidad del aire ambiente. https://www.minambiente.gov.co/wp-content/uploads/2021/10/Resolucion-2254-de-2017.pdf

MeacatrónicaLATAM. (2021). Sensores. https://www.mecatronicalatam.com/es/tutoriales/sensores/

Gasex. (2021). Gases inflamables ¿cuáles son y cómo tomar precauciones? https://gasex.cl/gases/gases-inflamables-cuales-son-y-como-tomar-precauciones/

HNC ELECTRIC. (2014). HT3000 IoT HMI. http://www.hncelectric.com/en_product_show.aspx?id=759

ITU-T Y.2060 (2012). Descripción general de Internet de los objetos. https://www.itu.int/rec/T-REC-Y.2060-201206-I/es

Arduino. (2023). Arduino IoT Cloud Remote App https://docs.arduino.cc/arduino-cloud/getting-started/iot-remote-app

Blynk. (2023). Blynk.io. https://blynk.io/

Seguridad IoT: riesgos en estos dispositivos | 2023 By CEUPE Container: Máster Ciberseguridad Year: 2021 URL: https://masterciberseguridadceupe.com/seguridad-iot-riesgos-en-estos-dispositivos/

Recomendaciones de seguridad en dispositivos IoT By IThinkUpc Container: ithinkweb Year: 2018 URL: https://www.ithinkupc.com/es/blog/recomendaciones-de-seguridad-en-dispositivos-iot

Fortinet. (2022). ¿Qué es la seguridad IoT Internet de las Cosas? https://www.fortinet.com/lat/resources/cyberglossary/iot-security

Human Machine Interfaces in the Internet of Things (IoT) - Latest Open Tech From Seeed By Container: Latest Open Tech From Seeed Year: 2021 URL: https://www.seeedstudio.com/blog/2021/04/19/human-machine-interfaces-in-the-internet-of-things-iot/

Pérez Galvis, H. (2022). Detección y monitoreo de gases mediante sensores catalíticos IoT. Universidad de los Andes. Disponible en: http://hdl.handle.net/1992/63425

ThingSpeak for Students and Educators - ThingSpeak IoT By Container: Thingspeak.com Year: 2023 URL: https://thingspeak.com/pages/education

Circuitos electrónicos. (s.f). ESP32 – Especificaciones y diseños. [en línea]. https://www.circuitos-electricos.com/esp32-especificaciones-y-disenos/

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2014). Metodología de la investigación (6ª ed.). McGraw-Hill Education. Recuperado de: http://repositorio.ucsh.cl/bitstream/handle/ucsh/2792/metodologia-de-la-investigacion.pdf?sequence=1

HiveMQ. (2023). Fundamentos de MQTT. Recuperado de https://www.hivemq.com/mqtt-essentials/

Arduino Docs (2023). Documentación de Arduino. https://docs.arduino.cc/

Oracle. (s.f). ¿Qué es el IoT?. https://www.oracle.com/es/internet-of-things/what-is-iot/

Alldatasheet. (s.f). Electronic Components Datasheet. https://www.alldatasheet.com/

Blynk.Documentation (2024). Documentación de Blynk. https://docs.blynk.io/en

Amazon Web Services, Inc. (2023). ¿Qué es una interfaz de programación de aplicaciones (API)?. https://aws.amazon.com/es/what-is/api/

CallMeBot. (2021). Free API to Send Whatsapp Messages. https://www.callmebot.com/blog/free-api-whatsapp-messages/

Sistema de Información Ambiental de Colombia. (2024). Inicio. Recuperado de https://www.siac.gov.co/

Apéndices

Apéndice A

Código Fuente

```
#define BLYNK_TEMPLATE_ID "[redacted]"

#define BLYNK_TEMPLATE_NAME "LED"

#define BLYNK_AUTH_TOKEN "[redacted]"

#define BLYNK_PRINT Serial

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include <HTTPClient.h>

#include <UrlEncode.h>

#include <DHT.h>

#include <LiquidCrystal.h>

const int rs = 15, en = 4, d4 = 5, d5 = 18, d6 = 19, d7 = 21;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

const char* ssid = [redacted];  // type your wifi name

const char* password = "[redacted]";  // type your wifi password

BlynkTimer timer;

#define DHTPIN 25 //Connect Out pin to D2 in NODE MCU
```

```
#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);

#define LED 2

const int buzzerPin = 32; // Pin del buzzer

#include <MQUnifiedsensor.h>

/**********Hardware Related Macros**********/

#define         Board                   ("ESP-32")

#define         MQ135_PIN               (35)  //Analog input A7 of your ESP32

#define         MQ5_PIN                 (34)  //Analog input A6 of your ESP32

//#define         PWMPin                  (5)   // Pin connected to mosfet

#define         PWMPin                  (14)   // Pin connected to mosfet

/**********Software Related Macros**********/

#define         RatioMQ5CleanAir          (6.5) //RS / R0 = 6.5 ppm

#define         RatioMQ135CleanAir        (3.6) //RS / R0 = 3.6 ppm

#define         ADC_Bit_Resolution        (12) // 12 bit ADC

#define         Voltage_Resolution        (3.3) // Volt resolution to calc the voltage

//#define         Type                    ("Arduino Mega 2560") //Board used

/**********Globals**********/
```

```
//Declare Sensor
MQUnifiedsensor MQ5(Board, Voltage_Resolution, ADC_Bit_Resolution, MQ5_PIN, "MQ-5");
MQUnifiedsensor MQ135(Board, Voltage_Resolution, ADC_Bit_Resolution, MQ135_PIN, "MQ-135");
// Definir variables
String msg;
String msg_MQ135;
float temperature;
float humidity;
int adc_MQ5;
int adc_MQ135;
int adc1_MQ135;
float MQ135calcR0;
float MQ5calcR0;
float CO;
float Alcohol;
float CO2;
float Toluen;
```

```
float NH4;

float Aceton;

float H2;

float LPG;

float CH4;

float CO_MQ5;

float Alcohol_MQ5;

int t;

int h;

int Gases_Inflamables;

int Calidad_Aire;

unsigned long currentMillis;

unsigned long previousMillis = 0; // Almacena el tiempo anterior

const long interval = 60000; // Intervalo de tiempo (1 minuto en milisegundos)

String phoneNumber = "+[redacted]";

String apiKey = "[redacted]";

void DesplaTexto() {

        for (int i = 0; i < 2; i++) { // Realizamos el ciclo dos veces
```

```
          for (int j = 0; j < 18; j++) { // Desplazamos el texto 34 posiciones hacia la izquierda
          lcd.scrollDisplayLeft();
          delay(350);
          }
        }
}
void LECTURASENSOR()
{
 float temperature_1 = dht.readTemperature();
float humidity_1 = dht.readHumidity();
temperature = temperature_1 ;
 humidity = humidity_1;
 t= int(temperature);
 h= int(humidity);
adc_MQ5 = analogRead(34); //Lemos la salida analógica  del MQ
 float voltajeMQ5 = adc_MQ5 * (3.3 / 1023.0);
 float RsMQ5=2000*((3.3-voltajeMQ5)/voltajeMQ5);  //Calculamos Rs con un RL de 2k
 adc_MQ135 = analogRead(35); //Lemos la salida analógica  del MQ
```

```
adc1_MQ135 = adc_MQ135 - 20;

float voltajeMQ135 = adc_MQ135 * (3.3 / 1023.0);

//  MQ-135 INICIO

 MQ135.update();

 MQ135.setA(605.18); MQ135.setB(-3.937);

 CO = MQ135.readSensor();

 MQ135.setA(77.255); MQ135.setB(-3.18);

 Alcohol = MQ135.readSensor();

 MQ135.setA(110.47); MQ135.setB(-2.862);

 CO2 = MQ135.readSensor();

 MQ135.setA(44.947); MQ135.setB(-3.445);

 Toluen = MQ135.readSensor();

 MQ135.setA(102.2 ); MQ135.setB(-2.473);

 NH4 = MQ135.readSensor();

 MQ135.setA(34.668); MQ135.setB(-3.369);

 Aceton = MQ135.readSensor();

 //  MQ-5 INICIO

 MQ5.update();
```

```
 MQ5.setA(1163.8); MQ5.setB(-3.874);

 H2 = MQ5.readSensor();

 MQ5.setA(80.897); MQ5.setB(-2.431);

 LPG = MQ5.readSensor();


 MQ5.setA(177.65); MQ5.setB(-2.56);

 CH4 = MQ5.readSensor();

 MQ5.setA(491204); MQ5.setB(-5.826);

 CO_MQ5 = MQ5.readSensor();

 MQ5.setA(97124 ); MQ5.setB(-4.918);

 Alcohol_MQ5 = MQ5.readSensor();


Gases_Inflamables = H2 + LPG + CH4 + CO_MQ5;

Calidad_Aire = CO + Alcohol + CO2 + Toluen + NH4 + Aceton;

Serial.print("CO: "+String(CO)+"\tCO2: "+String(CO2)+"\tAlcohol:
"+String(Alcohol)+"\tTolueno: "+String(Toluen)+"\tAmoníaco: "+String(NH4)+"\tAcetona:
"+String(Aceton));

Serial.println("\tH2: "+String(H2)+"\tGLP: "+String(LPG)+"\tMetano:
"+String(CH4)+"\tCO_MQ5: "+String(CO_MQ5)+"\talcohol_MQ5: "+String(Alcohol_MQ5));
```

```
 //Serial.println("Temperatura: "+ String (temperature)+ " °C\tHumedad: "+String(humidity)+"
%" + "\tDetector de Gas: "+String(Gases_Inflamables)+"\tCalidad de Aire:
"+String(Calidad_Aire));

 Serial.println("Temperatura: "+ String (temperature)+ " °C\tHumedad: "+String(humidity)+" %"
+ "\tDetector de Gas: "+String(Gases_Inflamables)+"\tCalidad de Aire: "+String(adc_MQ135));

 Blynk.virtualWrite(V25, temperature);

 Blynk.virtualWrite(V26, humidity);

 Blynk.virtualWrite(V34, Gases_Inflamables);

//if (temperature >= 14.0 && temperature <= 35.0 && adc_MQ135 <= 100 && CO < 9.4 &&
CO2 < 1000.0  && Alcohol <1000.0 && Toluen<100.0 && Aceton< 250.0 && NH4<25 &&
H2<10.0 && LPG<1.0 && CH4<2.0 && CO_MQ5<30.0 && Alcohol_MQ5<1000.0) {   //
Verifica si las variables están dentro del rango seguro

 if (temperature >= 14 && temperature <= 50 && adc1_MQ135 <= 100 && Gases_Inflamables
<= 80) {   // Verifica si las variables están dentro del rango seguro

    lcd.clear();                        // Limpia el LCD

    lcd.setCursor(0, 0);                    // Posiciona el cursor en la primera fila

    lcd.print(" T:" + String (t) + (char)223 + "C " + "H:" + String (h)+"%");

    if (adc1_MQ135 <= 50 ) {

    msg_MQ135 = "Buena";

    lcd.setCursor(0, 1);                    // Posiciona el cursor en la segunda fila
```

```
  lcd.print("C:Buena   " );             // Muestra un mensaje en el LCD
  Blynk.virtualWrite(V2, msg_MQ135);
  msg = " ";
  Blynk.virtualWrite(V1, msg);
  }
  if ( adc1_MQ135 >=51 && adc1_MQ135 <= 100 ) {
  msg_MQ135 = "Moderado";
  lcd.setCursor(0, 1);                    // Posiciona el cursor en la segunda fila
  lcd.print("C:Moderado" );            // Muestra un mensaje en el LCD
  Blynk.virtualWrite(V2, msg_MQ135);
  msg = " ";
  Blynk.virtualWrite(V1, msg);
  }
  lcd.setCursor(11, 1);                   // Posiciona el cursor en la segunda fila posicion 10
  lcd.print("G:"+ String (Gases_Inflamables));            // Muestra un mensaje en el LCD
 }
else
{
```

```
lcd.setCursor(0, 0);                    // Posiciona el cursor en la primera fila

lcd.print("    Alerta!     ");          // Muestra un mensaje en el LCD

Blynk.virtualWrite(V25, temperature);

Blynk.virtualWrite(V26, humidity);

Gases_Inflamables = H2 + LPG + CH4 + CO_MQ5 + Alcohol_MQ5;

Blynk.virtualWrite(V34, Gases_Inflamables);

 if (isnan(temperature) || isnan(humidity))

{

  activar_buzerr ();

  currentMillis = millis(); // Obtiene el tiempo actual

  lcd.setCursor(0, 1);

  lcd.print(" Error: DHT22   ");

  Serial.println("Error: Sensor DHT22 no detectado");

  if (currentMillis - previousMillis >= interval) {

  previousMillis = currentMillis; // Actualiza el tiempo anterior

  msg="Error: Sensor DHT22 no detectado";

  Blynk.virtualWrite(V1, msg);

  sendMessage(msg);
```

```
  }
}
else
{
  if (temperature < 14){
    currentMillis = millis(); // Obtiene el tiempo actual
    activar_buzerr ();
    lcd.setCursor(0, 1);
    lcd.print("Temp Baja: " + String (t) + (char)223 + "C");
    Serial.println("¡Alerta! Temperatura baja: " + String (temperature) + " °C");
    if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis; // Actualiza el tiempo anterior
    msg = "¡Alerta! Temperatura alta: " + String (temperature) + " °C";
    Blynk.virtualWrite(V1, msg);
    sendMessage(msg);
    }
   } else if (temperature > 40){
    currentMillis = millis(); // Obtiene el tiempo actual
```

```
    activar_buzerr ();
    lcd.setCursor(0, 1);
    lcd.print("Temp alta: " + String (t) + (char)223 + "C");
    Serial.println("¡Alerta! Temperatura alta: " + String (temperature) + " °C");
    Blynk.logEvent("temperatura", String("Alta Temperatura Detectada! Tº: ") + temperature);
    if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis; // Actualiza el tiempo anterior
    msg = "¡Alerta! Temperatura alta: " + String (temperature) + " °C";
    Blynk.virtualWrite(V1, msg);
    sendMessage(msg);
    }
   }
}
 if (CO >= 60 && Toluen >= 5 && Aceton >=  4 && NH4 >= 25 && adc_MQ135 >= 501)
 {
   activar_buzerr ();
   msg = "¡Alerta! Humo detectado";
   Blynk.virtualWrite(V1, msg);
```

```
  lcd.setCursor(0, 1);
  lcd.print(" Humo detectado ");
  Serial.println(" Humo detectado");
  Blynk.logEvent("humo");
  currentMillis = millis(); // Obtiene el tiempo actual
  if (currentMillis - previousMillis >= interval)
  {
    previousMillis = currentMillis; // Actualiza el tiempo anterior
    sendMessage(msg);
  }
}
if (adc1_MQ135 >=101 && adc1_MQ135 <= 150 )
{
 activar_buzerr ();
 currentMillis = millis(); // Obtiene el tiempo actual
 lcd.setCursor(0, 1);
 //lcd.print("CA Danina GS:" + String (adc_MQ135));
 lcd.print("CA Danina GS    ");
```

```
Serial.println("¡Alerta! ICA dañina a la salud para grupos sensibles: "+ String (adc_MQ135));
 msg = "¡Alerta! ICA dañina a la salud para grupos sensibles";
 Blynk.virtualWrite(V1, msg);
 msg_MQ135 = "Dañina GS";
 Blynk.virtualWrite(V2, msg_MQ135);
 Blynk.logEvent("alerta_calidad_aire", String("¡Alerta! ICA dañina a la salud de grupos sensibles: ") + adc1_MQ135);
 if (currentMillis - previousMillis >= interval)
 {
  previousMillis = currentMillis; // Actualiza el tiempo anterior
  msg = "¡Alerta! ICA dañina a la salud para grupos sensibles";
  sendMessage(msg);
 }
}
if (adc1_MQ135 >=151 && adc1_MQ135 <= 200 )
{
 activar_buzerr ();
 currentMillis = millis();
 lcd.setCursor(0, 1);
```

```
    lcd.print("CA Danina       ");
    Serial.println("¡Alerta! ICA dañina para la salud: "+ String (adc_MQ135));
    msg = "¡Alerta! ICA dañina para la salud";
    Blynk.virtualWrite(V1, msg);
    msg_MQ135 = "Dañina";
    Blynk.virtualWrite(V2, msg_MQ135);
    Blynk.logEvent("alerta_calidad_aire", String("¡Alerta! ICA dañina para la salud: ") +
adc1_MQ135);
    if (currentMillis - previousMillis >= interval)
    {
      previousMillis = currentMillis;
      msg = "¡Alerta! ICA dañina para la salud";
      sendMessage(msg);
    }
   }
   if (adc1_MQ135 >=201 && adc1_MQ135 <= 300 )
   {
    activar_buzerr ();
    currentMillis = millis();
```

```
    msg = "¡Alerta! ICA Muy dañina para la salud";
    Blynk.virtualWrite(V1, msg);
    msg_MQ135 = "Muy Dañina";
    Blynk.virtualWrite(V2, msg_MQ135);
    lcd.setCursor(0, 1);
    lcd.print("CA Muy Danina   ");
    Serial.println("¡Alerta! ICA Muy dañina para la salud: "+ String (adc1_MQ135));
    Blynk.logEvent("alerta_calidad_aire", String("¡Alerta! ICA Muy dañina para la salud: ") +
adc_MQ135);
    if (currentMillis - previousMillis >= interval)
    {
      previousMillis = currentMillis; // Actualiza el tiempo anterior
      sendMessage(msg);
    }
   }
   if (adc1_MQ135 >=301 && adc1_MQ135 <= 500 )
   {
    activar_buzerr ();
    currentMillis = millis(); // Obtiene el tiempo actual
```

```
 msg = "¡Alerta! ICA Pelgrosa";
 Blynk.virtualWrite(V1, msg);
 msg_MQ135 = "Peligrosa";
 Blynk.virtualWrite(V2, msg_MQ135);
 lcd.setCursor(0, 1);
 lcd.print("CA Peligrosa     ");
 Serial.println("¡Alerta! ICA peligrosa: "+ String (adc1_MQ135));
 Blynk.logEvent("alerta_calidad_aire", String("¡Alerta! ICA Pelgrosa: ") + adc1_MQ135);
 if (currentMillis - previousMillis >= interval)
 {
   previousMillis = currentMillis; // Actualiza el tiempo anterior
   sendMessage(msg);
 }
}
if (LPG >= 12.0 || CH4 >= 5.0 || H2 >= 7.0 )
{
 activar_buzerr ();
 currentMillis = millis(); // Obtiene el tiempo actual
```

```
    lcd.setCursor(0, 1);
    lcd.print(" Gas inflamable ");
    Serial.println("¡Alerta! Gas inflamable detectado");
     msg = "¡Alerta! gas inflamable detectado";
     Blynk.virtualWrite(V1, msg);
     Blynk.logEvent("Gas_Inflamable");
    if (currentMillis - previousMillis >= interval)
    {
      previousMillis = currentMillis; // Actualiza el tiempo anterior
      msg = "¡Alerta! gas inflamable detectado";
      sendMessage(msg);
    }
   }
  }
}
void setup()
{
  Serial.begin(115200);
```

```
lcd.begin(16, 2);

Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);

dht.begin();

timer.setInterval(1000L, LECTURASENSOR);

analogReadResolution(10);

pinMode(LED, OUTPUT);

pinMode(buzzerPin, OUTPUT);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print(" Dispositivo de");

lcd.setCursor(0, 1);

lcd.print("deteccion de gases");

delay(3000);

MQ5.setRegressionMethod(1);  //_PPM =  a*ratio^b

MQ135.setRegressionMethod(1); //_PPM =  a*ratio^b

MQ5.init();

MQ135.init();

 MQ5.setRL(2);  //RL de 2K
```

```
 MQ135.setRL(2);  //RL de 2K
Serial.print("Calibrating MQ5 please wait.");
lcd.clear();
lcd.print("Calibrating MQ5 ");
lcd.setCursor(0, 1);
lcd.print("Please wait!.");
MQ5calcR0 = 0;
for(int i = 1; i<=10; i ++)
{
  MQ5.update();
  MQ5calcR0 += MQ5.calibrate(RatioMQ5CleanAir);
  Serial.print(".");
  lcd.print(".");
  lcd.scrollDisplayLeft();
  delay(500);
}
MQ5.setR0(MQ5calcR0/10);
Serial.println("  done!.");
```

```
lcd.clear();
lcd.print(" MQ5 Calibrado! ");
Serial.print("Calibrating MQ135 please wait.");
lcd.clear();
lcd.print("Calibrating MQ135 ");
lcd.setCursor(0, 1);
lcd.print("Please wait!.");
MQ135calcR0 = 0;
for(int i = 1; i<=10; i ++)
{
  MQ135.update();
  MQ135calcR0 += MQ135.calibrate(RatioMQ135CleanAir);
  Serial.print(".");
  lcd.print(".");
  lcd.scrollDisplayLeft();
  delay(500);
}
MQ135.setR0(MQ135calcR0/10);
```

```
  Serial.println(" done!.");
  lcd.clear();
  lcd.print("MQ135 Calibrado!");
  while (isinf(MQ5calcR0) || MQ5calcR0 == 0 || isinf(MQ135calcR0) || MQ135calcR0 == 0) {
    if (isinf(MQ5calcR0)) {
      Serial.println("Advertencia: Problema de conexión en MQ-5, R0 es infinito (se detectó un circuito abierto), verifique su cableado y suministro");
    }
    if (MQ5calcR0 == 0) {
      Serial.println("Advertencia: Se encontró un problema de conexión en MQ-5, R0 es cero (el pin analógico se corta a tierra), verifique su cableado y suministro");
    }
    if (isinf(MQ135calcR0)) {
      Serial.println("Advertencia: Problema de conexión en MQ-135, R0 es infinito (se detectó un circuito abierto), verifique su cableado y suministro");
    }
    if (MQ135calcR0 == 0) {
      Serial.println("Advertencia: Se encontró un problema de conexión en MQ-135, R0 es cero (el pin analógico se corta a tierra), verifique su cableado y suministro");
```

```
    }
    delay(1000);
  }
  conectNetwork();
}
void loop()
{
 currentMillis = millis(); // Obtiene el tiempo actual
 Blynk.run();
 timer.run();
}
void activar_buzerr ()
{
    digitalWrite(LED, HIGH);  // turn the LED on (HIGH is the voltage level)
    digitalWrite(buzzerPin, HIGH);  // turn the buzzerPin on (HIGH is the voltage level)
    delay(1000); // Duración del tono
    digitalWrite(LED, LOW);  // turn the LED on (HIGH is the voltage level)
    digitalWrite(buzzerPin, LOW);  // turn the zumbador off
```

```
}
void conectNetwork()
{
 WiFi.begin(ssid, password);
 Serial.println("Intentando Conectar a la RED WiFi");
 while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Conectado a la Red WiFi! ");
  Serial.println(WiFi.localIP());
}
void sendMessage(String message)
  {
  String url = "https://api.callmebot.com/whatsapp.php?phone=" + phoneNumber + "&apikey="
+ apiKey + "&text=" + urlEncode(message);
  postData(url);
   }
```

```
void postData(String url) //función para hacer la llada a la API con post
{
  int httpCode;
  HTTPClient http;  // se declara un objeto  HTTPClient
  http.begin(url);
  httpCode = http.POST(url);
  if (httpCode == 200) {  // verificamos respuesta
    Serial.println("Mensaje enviado Correctamente.");
  } else {
    Serial.print("Error: ");
    Serial.println(httpCode);
  }
  http.end();
}
```