UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE GUAYAQUIL

CARRERA DE INGENIERÍA ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE SISTEMA IOT PARA LA MEDICIÓN DE LA CALIDAD DEL AIRE Y PARÁMETROS AMBIENTALES EN CONSULTORIO ODONTOPEDIÁTRICO UTILIZANDO RASPBERRY PI Y SERVICIOS CLOUD.

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Electrónica

AUTORES:
FÉLIX ADRIÁN TOMALÁ VERGARA
DIEGO RODRIGO CELY JADAN

TUTOR:
ING. TEDDY NEGRETE MSc.

Guayaquil – Ecuador

2025

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Félix Adrián Tomalá Vergara con documento de identificación N° 0956387476 y Diego Rodrigo Cely Jadan con documento de identificación N° 0705050417 manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Guayaquil, 16 de septiembre de 2025

Atentamente,

(f) Félix Adrián Tomalá Vergara
C.I: 0956387476

(f) Diego Rodrigo Cely Jadan
C.I: 0705050417

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Félix Adrián Tomalá Vergara con documento de identificación No. 0956387476 y Diego Rodrigo Cely Jadan con documento de identificación No. 0705050417, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos los autores del Proyecto técnico: "Diseño e implementación de sistema IOT para la medición de la calidad del aire y parámetros ambientales en consultorio odontopediátrico utilizando Rasbperry Pi y servicios cloud", el cual ha sido desarrollado para optar el título de: Ingeniero Electrónico, en la Universidad Politécnica Salesiana, quedando la universidad facultada para ejercer y usar los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana

Guayaquil, 16 de septiembre de 2025

Atentamente,

(f) Félix Adrián Tomalá Vergara
C.I: 0956387476

(f) Diego Rodrigo Cely Jadan
C.I: 0705050417

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, MSc. Teddy Jhennse Negrete Peña con documento de identificación No. 0912419611, docente de la Universidad Politécnica Salesiana , declaro que bajo mi tutoría fue desarrollado el trabajo de titulación "Diseño e implementación de sistema IOT para la medición de la calidad del aire y parámetros ambientales en consultorio odontopediátrico utilizando Rasbperry Pi y servicios cloud", realizado por Félix Adrián Tomalá Vergara con documento de identificación N° 0956387476 y por Diego Rodrigo Cely Jadan con documento de identificación N° 0705050417, obteniendo como resultado final el trabajo de titulación bajo la opción de Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 16 de septiembre de 2025

Atentamente,

MSc. Teddy Jhennse Negrete Peña

C.I: 0912419611

DEDICATORIA

Este logro se lo dedico en primer lugar a Dios que me brindo salud y las herramientas para conseguirlo, en segundo lugar a mi padre y mi madre, que aunque no están físicamente conmigo, ellos me acompañaron en este hermoso viaje y son mi mayor fuente de motivación para dar lo mejor de mi día a día, estoy eternamente agradecido con ellos por siempre, a mis hermanos por ser pilares en mi crianza y desarrollo profesional, y a todos los docentes que compartieron sus experiencias y enseñanzas durante toda la carrera .

Eternamente agradecido

Félix Adrián Tomalá Vergara

DEDICATORIA

A mis padres. Este logro es un testimonio de su amor y dedicación. Valoro las lecciones de vida que me han impartido y por el cariño que siempre me han brindado. Mi gratitud hacia ustedes por ser alguien en la vida.

Diego Rodrigo Cely Jadan

AGRADECIMIENTO

Le doy gracias a Dios por este logro, por esta meta cumplida, por un objetivo de muchos , a mis padres por su apoyo y amor infinito, a mis hermanos por su guía y dirección, y a todos los que hicieron posible este logro, de corazón eternamente agradecido

Félix Adrián Tomalá Vergara

AGRADECIMIENTO

A mi Dios y mis padres, cuyo amor y apoyo incondicional con sacrificio han sido mi gran apoyo y guía en este camino largo. Por tal motivo doy gracias a ellos.

Diego Rodrigo Cely Jadan

RESUMEN DEL PROYECTO

AÑO	ALUMNOS	DIRECTOR DE PROYECTO	TEMA DE PROYECTO DE TITULACIÓN
2025	FÉLIX ADRIÁN TOMALÁ V. DIEGO RODRIGO CELY JADAN	MSC.TEDDY NEGRETE	" DISEÑO E IMPLEMENTACIÓN DE SISTEMA IOT PARA LA MEDICIÓN DE LA CALIDAD DEL AIRE Y PARÁMETROS AMBIENTALES EN CONSULTORIO ODONTOPEDIÁTRICO UTILIZANDO RASPBERRY PI Y SERVICIOS CLOUD “

El presente proyecto tiene como finalidad diseñar e implementar un sistema de monitoreo ambiental basado en tecnología IoT, orientado a medir en tiempo real la calidad del aire y parámetros ambientales dentro de un consultorio odontopediátrico. Esta iniciativa responde a la necesidad de controlar condiciones ambientales críticas, como la presencia de material particulado (PM1.0, PM2.5 y PM10), compuestos orgánicos volátiles (VOC), así como temperatura, presión atmosférica y humedad relativa, que afectan directamente la salud respiratoria de los pacientes pediátricos y del personal odontológico. El sistema propuesto emplea una Raspberry Pi como unidad de procesamiento, junto con sensores específicos (PMS5003 y BME680) para la adquisición de datos ambientales. Estos datos son enviados a una base de datos en la nube como ThingSpeak, permitiendo su almacenamiento, análisis histórico y visualización mediante dashboards interactivos accesibles desde cualquier dispositivo conectado. Además, el sistema integra un módulo de alertas automáticas por correo electrónico, que notifica cuando los valores registrados superan los umbrales de seguridad establecidos, permitiendo respuestas inmediatas ante condiciones no saludables.

El proyecto será validado en un entorno real: la Clínica MOM, ubicada en Samborondón, Ecuador, con el objetivo de asegurar la funcionalidad, precisión y escalabilidad del sistema. Esta solución busca no solo optimizar la calidad del aire en espacios médicos pediátricos, sino también establecer un modelo replicable y de bajo costo que pueda ser adoptado por otras

clínicas, centros educativos o espacios cerrados con alta sensibilidad ambiental. Se espera que este desarrollo contribuya significativamente a mejorar la percepción de seguridad sanitaria y el confort de los usuarios, mediante el uso de tecnologías accesibles, sostenibles e innovadoras.

Palabras Clave: IoT, calidad del aire, Raspberry Pi, sensores ambientales, consultorio odontopediátrico

ABSTRACT

YEAR	STUDENTS	PRJ. DIRECTOR	SUBJECT
2025	FÉLIX ADRIÁN TOMALÁ V. DIEGO RODRIGO CELY JADAN	MSC TEDDY NEGRETE	"DISEÑO E IMPLEMENTACIÓN DE SISTEMA IOT PARA LA MEDICIÓN DE LA CALIDAD DEL AIRE Y PARÁMETROS AMBIENTALES EN CONSULTORIO ODONTOPEDIÁTRICO UTILIZANDO RASPBERRY PI Y SERVICIOS CLOUD"

The purpose of this project is to design and implement an environmental monitoring system based on IoT technology, aimed at measuring in real time the air quality and environmental parameters within a pediatric dental office. This initiative addresses the need to control critical environmental conditions such as the presence of particulate matter (PM1.0, PM2.5, and PM10), volatile organic compounds (VOCs), as well as temperature, atmospheric pressure, and relative humidity, which directly affect the respiratory health of pediatric patients and dental staff. The proposed system uses a Raspberry Pi as the processing unit, along with specific sensors (PMS5003 and BME680) for environmental data acquisition. This data is transmitted to a cloud-based database, such as ThingSpeak, allowing for storage, historical analysis, and visualization through interactive dashboards accessible from any connected device. Additionally, the system integrates an automatic email alert module that notifies when recorded values exceed established safety thresholds, enabling immediate responses to unhealthy conditions.

The project will be validated in a real environment: Clínica MOM, located in Samborondón, Ecuador, with the goal of ensuring the system's functionality, accuracy, and scalability. This solution aims not only to optimize air quality in pediatric medical spaces but also to establish a replicable, low-cost model that can be adopted by other clinics, educational centers, or

enclosed spaces with high environmental sensitivity. This development is expected to significantly contribute to improving the perception of sanitary safety and user comfort through the use of accessible, sustainable, and innovative technologies.

Keywords: IoT, air quality, Raspberry Pi, environmental sensors, pediatric dental office

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La calidad del aire en espacios cerrados, especialmente en entornos clínicos pediátricos como los consultorios odontopediátricos, representa un factor determinante para la salud y el bienestar tanto de los pacientes como del personal médico. En particular, los niños son altamente sensibles a las variaciones ambientales y a la exposición prolongada a partículas contaminantes o compuestos volátiles presentes en el aire. Diversos estudios han demostrado que niveles elevados de PM1, PM2.5, PM10, temperatura inadecuada, humedad relativa alta o baja y compuestos orgánicos volátiles (VOC) pueden desencadenar enfermedades respiratorias, afectar el sistema inmunológico y disminuir el confort durante las consultas. No obstante, muchos consultorios carecen de sistemas automatizados que permitan monitorear en tiempo real estas variables, lo que genera un riesgo potencial para la salud (Wang et al., 2025).

Ante esta problemática, el presente proyecto tiene como objetivo desarrollar un sistema de monitoreo ambiental basado en tecnologías del Internet de las Cosas (IoT), capaz de registrar en tiempo real los parámetros clave de calidad del aire en un consultorio odontopediátrico.

Para ello, se emplea una Raspberry Pi como plataforma de procesamiento, junto con sensores especializados PMS5003 para material particulado y BME680 para temperatura, humedad, presión y gases. Los datos recopilados se envían a una base de datos en la nube (ThingSpeak), donde se almacenan y visualizan mediante dashboards dinámicos, y se generan alertas automáticas cuando los valores exceden los umbrales establecidos. Esta solución se validará en un entorno real: la Clínica MOM, ubicada en Samborondón, Ecuador (Shahid, 2025).

El documento se estructura en varios capítulos que describen de manera detallada cada fase del proyecto:

Capítulo 1. El Problema: Se describe el contexto en el que se desarrolla la investigación, identificando las necesidades específicas del entorno clínico pediátrico, los antecedentes, la justificación del proyecto, y los objetivos generales y específicos que guían el desarrollo de la solución tecnológica.

Capítulo 2. Marco Teórico: Presenta los fundamentos conceptuales relacionados con el IoT, la calidad del aire en entornos clínicos, las plataformas de hardware utilizadas (Raspberry Pi), y los sensores empleados. Además, se incluyen referencias a investigaciones previas y a tecnologías similares implementadas en contextos médicos.

Capítulo 3. Metodología: Se detallan las fases de desarrollo del sistema, incluyendo el diseño del circuito, la selección de componentes, la arquitectura del software, la conexión con la nube y las herramientas empleadas para el análisis y visualización de los datos. También se establece la estrategia de validación del sistema.

Capítulo 4. Resultados: Se presentan los resultados de las pruebas realizadas, tanto en laboratorio como en condiciones reales dentro del consultorio, analizando la precisión de las mediciones, la eficiencia del sistema de alertas y la confiabilidad del almacenamiento de datos.

Capítulo 5. Conclusiones y Recomendaciones: Este capítulo resume los logros alcanzados, las limitaciones encontradas durante el desarrollo, y ofrece recomendaciones para futuras mejoras o ampliaciones del sistema, incluyendo su posible adaptación a otros entornos clínicos o educativos.

En conjunto, este proyecto pretende ser una contribución significativa al área de la salud y la tecnología, fomentando el uso de soluciones IoT accesibles, escalables y replicables para mejorar las condiciones ambientales en espacios pediátricos.

1 El problema

1.1 Descripción del problema

Los consultorios odontológicos, especialmente los especializados en odontopediatría, requieren condiciones ambientales óptimas para garantizar la salud y el bienestar tanto de los pacientes como del personal médico. La presencia de contaminantes en el aire, como partículas en suspensión (PM2.5, PM10), compuestos orgánicos volátiles (VOC) y niveles inadecuados de temperatura y humedad, puede generar riesgos para la salud respiratoria, especialmente en niños, quienes son más vulnerables a ambientes mal ventilados o contaminados (Harrel & Molinari, 2004).

En muchos consultorios, el monitoreo de la calidad del aire se realiza de forma esporádica o manual, sin mecanismos automatizados de medición continua ni alertas en tiempo real. Esta situación genera varios problemas:

- Exposición a contaminantes: Los pacientes y profesionales pueden estar expuestos a niveles elevados de partículas y gases sin saberlo.

- Ausencia de alertas: No se dispone de un sistema de notificación automática cuando se superan umbrales críticos.

- Imposibilidad de análisis histórico: No hay registros organizados para evaluar condiciones pasadas o implementar mejoras ambientales basadas en datos.

Como señala (Wang et al., 2021) los sistemas basados en IoT con sensores ambientales

permiten no solo la detección en tiempo real de contaminantes, sino también el almacenamiento y análisis histórico de datos, facilitando respuestas proactivas ante eventos críticos. Por tanto, implementar soluciones automatizadas resulta esencial para garantizar entornos clínicos más seguros y saludables.

Diversos estudios han demostrado que la implementación de sistemas automatizados basados en tecnologías IoT es una solución efectiva para mejorar la calidad del aire en espacios cerrados, especialmente en entornos sensibles como consultorios médicos. Un ejemplo destacado es el proyecto desarrollado por (Kumar et al., 2020), donde se implementó un sistema de monitoreo en tiempo real de contaminantes atmosféricos en clínicas urbanas utilizando sensores de bajo costo conectados a una plataforma de análisis en la nube. Este enfoque no solo permitió generar alertas instantáneas al detectar niveles críticos de CO_2 y partículas suspendidas, sino que también facilitó la recolección de datos históricos para la toma de decisiones ambientales. La evidencia empírica de este tipo de investigaciones refuerza la viabilidad y necesidad de soluciones automatizadas como las que aquí se proponen.

1.2. Antecedentes

Se han desarrollado diversos proyectos de investigación enfocados en la aplicación de tecnologías del Internet de las Cosas (IoT) para el monitoreo de variables ambientales en espacios controlados. Estas iniciativas, aunque centradas en distintos entornos como la agricultura, la industria y la salud, evidencian el creciente interés por utilizar sensores inteligentes, plataformas de bajo costo y conectividad en tiempo real para mejorar las condiciones ambientales y de producción.

Uno de estos estudios es “Diseño e implementación de un sistema de monitoreo de calidad del aire en espacios cerrados con tecnología IoT para entornos educativos”, cuyo objetivo fue implementar una red de sensores conectados a una plataforma web para monitorear en tiempo real los niveles de CO_2, temperatura y humedad en aulas escolares, con la finalidad de mejorar el confort térmico y la ventilación en estos espacios (García López & Muñoz Torres, 2022).

Otro ejemplo es el trabajo titulado “Sistema IoT para la vigilancia de parámetros ambientales en clínicas urbanas utilizando Raspberry Pi y sensores de bajo costo”, donde se diseñó un prototipo para monitorear la calidad del aire (PM2.5, CO_2 y VOC) en salas de espera y consultorios médicos. Este sistema demostró que la implementación de alertas automáticas y almacenamiento en la nube puede prevenir riesgos respiratorios en pacientes y personal médico (Kumar & Sharma, 2020).

También se destaca el proyecto “Desarrollo de una estación inteligente para monitoreo ambiental en tiempo real en laboratorios universitarios”, cuyo objetivo fue aplicar sensores integrados a microcontroladores tipo ESP32 y Raspberry Pi para registrar y analizar condiciones de humedad, temperatura y concentración de partículas. Los resultados permitieron optimizar la ventilación y el uso de purificadores de aire, mejorando significativamente el ambiente de estudio (Luna Salazar & Paredes Gómez, 2021).

Estas investigaciones demuestran la viabilidad técnica y el impacto positivo del uso de tecnologías IoT para la mejora de ambientes interiores. Sin embargo, aún existe un vacío en cuanto a su aplicación específica en consultorios odontopediátricos, donde los pacientes principalmente niños son altamente vulnerables a condiciones ambientales deficientes. El presente proyecto busca cubrir esa necesidad mediante una solución accesible, automatizada y escalable que contribuya al bienestar y seguridad sanitaria en este tipo de espacios clínicos.

1.3. Importancia y alcances

La atención médica en entornos pediátricos, como los consultorios odontopediátricos, demanda condiciones ambientales óptimas para garantizar no solo el confort, sino también la seguridad sanitaria de los pacientes, quienes en su mayoría son niños en edades tempranas y con sistemas respiratorios en desarrollo. La presencia de contaminantes como material particulado (PM1.0, PM2.5 y PM10), compuestos orgánicos volátiles (VOC), temperatura extrema y humedad inadecuada, representa un riesgo significativo para la salud, especialmente cuando estos parámetros no son monitoreados de forma continua.

En este contexto, el uso de tecnologías IoT como sensores ambientales integrados a una Raspberry Pi y conectados a la nube ofrece una solución eficiente, económica y escalable para el control en tiempo real de las condiciones del aire en espacios clínicos cerrados. Este

tipo de sistemas permite automatizar la detección de cambios críticos en el ambiente, generando alertas preventivas que facilitan una respuesta inmediata por parte del personal médico. Así, se promueve la creación de espacios más seguros, reduciendo el riesgo de exposición a agentes nocivos, infecciones respiratorias o episodios alérgicos en pacientes vulnerables.

Este proyecto tiene como finalidad optimizar el monitoreo ambiental en consultorios mediante:

- La detección temprana de condiciones críticas de calidad del aire.
- La mejora del confort clínico para niños y personal odontológico.
- La visualización y trazabilidad de datos históricos para decisiones correctivas.
- La automatización de alertas ante situaciones de riesgo ambiental.

Además, busca reducir:

- La dependencia del monitoreo manual y esporádico.
- El riesgo de contagio por aerosoles en ambientes cerrados.
- La exposición continua a condiciones ambientales deficientes.
- El uso excesivo de equipos de purificación o climatización sin control eficiente.

El alcance del proyecto va más allá de un solo consultorio. Su diseño modular y de bajo costo permite que esta solución pueda ser replicada en clínicas dentales, hospitales pediátricos, salas de espera o incluso en instituciones educativas que atienden población infantil. La implementación de tecnologías como esta promueve una cultura de prevención, respaldada en datos en tiempo real, que puede transformar el modo en que se gestionan los espacios médicos sensibles.

En definitiva, este sistema no solo representa un avance en innovación tecnológica aplicada a la salud, sino también una oportunidad para fomentar ambientes clínicos más seguros, adaptados a las necesidades de una población pediátrica cada vez más consciente y exigente respecto a su bienestar.

1.4. Delimitación del problema

1.4.1. Temporal

Este proyecto se desarrollará entre los meses de mayo 2025 a agosto del 2025.

1.4.2. Geográfica

El proyecto se llevará a cabo en un consultorio odontopediátrico llamado Clínica MOM, ubicado en Samborondón, Ecuador. Las pruebas iniciales del sistema se realizarán en laboratorio y posteriormente se implementará el sistema definitivo en el consultorio para pruebas en condiciones reales.

1.4.3. Académica

El proyecto está enfocado en el diseño e implementación de un sistema IoT que monitorea en tiempo real la calidad del aire en espacios cerrados, específicamente en un entorno clínico pediátrico. Se aplicarán conocimientos en:

- Sistemas embebidos con Raspberry Pi.
- Sensores ambientales y adquisición de datos.
- Bases de datos orientadas a series temporales.
- Plataformas de visualización de datos y servicios cloud.

Este proyecto será una oportunidad para aplicar los conocimientos teóricos adquiridos y las habilidades prácticas en un contexto real, contribuyendo a la innovación en la monitorización ambiental aplicada a la medicina.

1.5. Objetivos

1.5.1. Objetivo General

Desarrollar un sistema IoT basado en una Raspberry Pi para la medición en tiempo real de la calidad del aire y parámetros ambientales (temperatura, humedad, y gases de compuestos orgánicos volátiles) en un consultorio odontopediátrico utilizando sensores y almacenamiento

en la nube, con el fin de garantizar un ambiente saludable y seguro durante las consultas odontológicas.

1.5.2. Objetivos Específicos

- Diseñar e implementar un sistema de adquisición de datos utilizando la Raspberry Pi y sensores de partículas, sensores de gases, temperatura y humedad, para monitorear la calidad del aire en tiempo real en el consultorio odontológico para niños.

- Configurar la transmisión de los datos recolectados hacia servidor web en la nube, mediante una base de datos de series temporales, para almacenar, procesar y gestionar la información ambiental, facilitando su consulta y análisis.

- Desarrollar una interfaz visual para la monitorización remota de la calidad del aire y parámetros ambientales, utilizando herramientas como servidores Cloud, que permita visualizar los datos de manera clara y configurable, e implemente un sistema automatizado de alertas por correo electrónico para notificar al personal y profesionales del consultorio cuando los niveles de gases o partículas superen umbrales críticos, garantizando una respuesta rápida ante riesgos potenciales para la salud.

- Asegurar que las mediciones de los sensores ambientales sean correctas y confiables, mediante la validación regular de los datos y la detección de posibles errores o valores fuera de lo normal.

2 Fundamentos teóricos

2.1 Internet de las Cosas (IoT) en Ambientes Clínicos

El Internet de las Cosas (IoT, por sus siglas en inglés) se refiere a la red de objetos físicos interconectados que recopilan, procesan y comparten datos mediante tecnologías inalámbricas, sensores y software, sin necesidad de intervención humana directa. Estos dispositivos pueden monitorear parámetros del entorno, realizar acciones automatizadas y transmitir información a través de plataformas en la nube, lo que permite la toma de decisiones basada en datos en tiempo real.

En el contexto clínico, la incorporación del IoT ha transformado significativamente la forma en que se supervisan y gestionan las condiciones ambientales dentro de hospitales, clínicas y consultorios. Esta tecnología facilita el seguimiento continuo de variables críticas como la calidad del aire, temperatura, humedad, niveles de CO_2 y otros contaminantes. La capacidad de obtener datos precisos y constantes ayuda a mantener ambientes saludables para pacientes y personal médico, minimizando riesgos de infecciones respiratorias y aumentando la seguridad en procedimientos clínicos.

Particularmente en áreas como la odontopediatría, donde los pacientes suelen ser niños con sistemas inmunológicos en desarrollo, un sistema automatizado de monitoreo puede marcar la diferencia entre un ambiente seguro y uno potencialmente riesgoso. Además, el IoT permite integrar sistemas de alerta temprana, almacenamiento de registros históricos y visualización remota de los datos mediante dashboards, lo que incrementa la capacidad de respuesta ante situaciones no óptimas. Esta automatización también contribuye a reducir el consumo energético y el uso innecesario de dispositivos de climatización o purificación del aire, promoviendo una gestión clínica más sostenible (Greengard, 2015).

2.2 Calidad del Aire en Consultorios Odontológicos

La calidad del aire en espacios cerrados es un factor clave en la prevención de enfermedades respiratorias, alergias y otras afecciones relacionadas con contaminantes ambientales. En el caso de los consultorios odontológicos, este aspecto cobra aún más importancia debido a la generación constante de aerosoles durante los procedimientos clínicos. Estos aerosoles pueden contener microorganismos, residuos biológicos, partículas finas (PM2.5 y PM10), compuestos orgánicos volátiles (VOC), así como gases provenientes de materiales odontológicos o productos de limpieza.

Diversas investigaciones han demostrado que la exposición a estos contaminantes en entornos mal ventilados puede incrementar significativamente el riesgo de infecciones cruzadas entre pacientes y profesionales de la salud, así como generar molestias respiratorias, fatiga y disminución en la calidad de atención (Harrel & Molinari, 2004). El problema se agrava en consultorios pediátricos, donde los pacientes son más vulnerables y sensibles a las variaciones ambientales.

Ante esta situación, la implementación de sistemas de monitoreo ambiental con sensores de precisión permite no solo detectar niveles peligrosos de contaminantes en tiempo real, sino también generar acciones correctivas automáticas o alertas inmediatas. El uso de sensores como el PMS5003 para material particulado y el BME680 para VOC, temperatura y humedad, se ha vuelto una práctica eficiente y accesible, respaldada por su integración con plataformas IoT. Estos sistemas mejoran la percepción de seguridad por parte de los pacientes y sus acompañantes, fortalecen los protocolos de bioseguridad y permiten a los profesionales tomar decisiones basadas en datos confiables, en lugar de depender únicamente de la experiencia subjetiva o inspecciones manuales.

En resumen, el monitoreo continuo de la calidad del aire en consultorios odontológicos no solo mejora la salud y el confort de quienes ocupan estos espacios, sino que también promueve una atención médica más responsable, segura y alineada con los avances tecnológicos actuales (Harrel & Molinari, 2004).

2.3 Raspberry Pi como Plataforma IoT

La Raspberry Pi es una microcomputadora de placa única (single-board computer) desarrollada por la Fundación Raspberry Pi con el objetivo de fomentar la enseñanza de ciencias computacionales en la educación básica. Sin embargo, gracias a su bajo costo, tamaño compacto, consumo energético reducido y versatilidad, ha sido ampliamente adoptada en sectores como la automatización, la domótica, el monitoreo ambiental y los proyectos de Internet de las Cosas (IoT).

En aplicaciones IoT, la Raspberry Pi cumple funciones similares a las de un microcontrolador avanzado, con la ventaja de ejecutar sistemas operativos completos como Raspberry Pi OS (basado en Linux), lo que permite mayor flexibilidad en la programación, conexión a redes y uso de bases de datos. Posee puertos GPIO (General Purpose Input/Output) que permiten la conexión directa con sensores ambientales, actuadores, módulos de comunicación y otros dispositivos electrónicos, convirtiéndola en una excelente plataforma para proyectos de adquisición y procesamiento de datos en tiempo real.

En el caso específico del presente proyecto, la Raspberry Pi actúa como el núcleo central del sistema de monitoreo ambiental. Su capacidad para conectarse a internet mediante Wi-Fi o Ethernet le permite enviar datos a plataformas en la nube como ThingSpeak, desde donde pueden ser visualizados mediante dashboards, analizados y almacenados de forma segura. Además, su compatibilidad con lenguajes de programación como Python facilita la implementación de rutinas para lectura de sensores, filtrado de datos, generación de alertas por correo electrónico y actualizaciones automáticas del sistema.

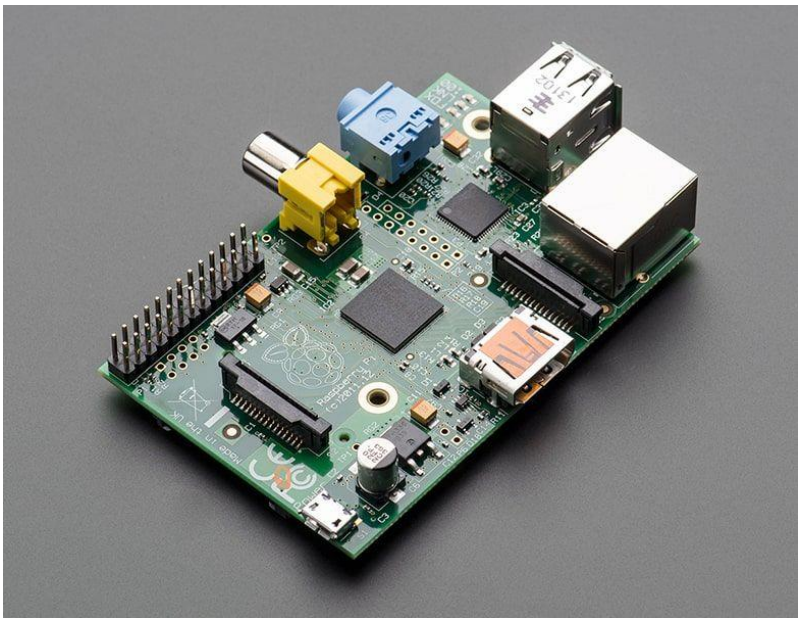
En comparación con otros microcontroladores como Arduino, la Raspberry Pi ofrece mayores recursos de procesamiento, memoria y almacenamiento, lo que la hace ideal para proyectos que requieren conectividad web, procesamiento de datos más complejos y manejo de interfaces gráficas. Su arquitectura también permite la instalación de servidores locales, lo que habilita su uso como una solución autónoma y descentralizada en lugares con conectividad limitada.

En contextos clínicos, el uso de Raspberry Pi ha demostrado ser eficiente para la implementación de sistemas de monitoreo ambiental, ya que permite la integración de sensores de calidad del aire, visualización en tiempo real de las condiciones del consultorio, y emisión de alertas que previenen exposiciones prolongadas a contaminantes. Por estas

razones, la Raspberry Pi representa una plataforma IoT óptima para proyectos enfocados en mejorar la seguridad y confort en entornos pediátricos odontológicos, combinando eficiencia tecnológica con accesibilidad económica (Monk, 2015). En la figura 1 se observa la Raspberry Pi 1 model B.

Figura 1

Raspberry Pi 1 model B



Nota. Imagen de la Raspberry Pi (Pi, 2025)

La Raspberry Pi es una microcomputadora de placa reducida que ha revolucionado el desarrollo de proyectos de automatización, domótica e Internet de las Cosas (IoT) debido a su equilibrio entre bajo costo, portabilidad y capacidad de procesamiento. En la Figura 2 se presenta un esquema de la arquitectura del hardware correspondiente al modelo clásico de la Raspberry Pi (modelo B, 2011), en el cual se detallan los principales componentes integrados en la placa (Upton & Halfacree, 2016).

Esta versión cuenta con un procesador Broadcom BCM2835, 512 MB de memoria SDRAM, salida de video compuesto, conector HDMI, puerto de cámara (CSI), conector de pantalla (DSI), puertos USB 2.0, Ethernet y conector jack de 3.5 mm para salida de audio estéreo. Uno

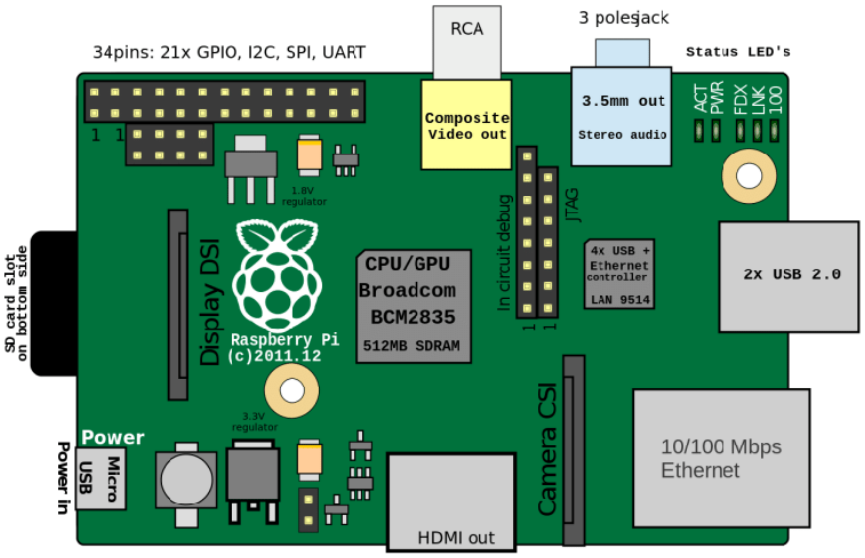
de los elementos más destacados de esta placa es su encabezado GPIO de 26 pines, que permite la conexión directa con sensores y módulos externos mediante interfaces como I2C, SPI y UART.

Gracias a su versatilidad, la Raspberry Pi puede actuar como centro de procesamiento en sistemas domóticos, donde recopila datos desde sensores ambientales (como el BME680 o el PMS5003), los procesa localmente o los envía a plataformas de visualización como ThingSpeak. Además, su compatibilidad con sistemas operativos ligeros basados en Linux, como Raspbian, facilita el desarrollo de software en lenguajes como Python, con bibliotecas orientadas al control de hardware (RPi.GPIO, pigpio, entre otras).

La organización eficiente del hardware de la Raspberry Pi le permite integrarse en sistemas embebidos con bajo consumo energético, almacenamiento por tarjeta microSD, y conectividad suficiente para tareas de monitoreo y control automatizado en tiempo real. Figura 2.

Figura 2

Arquitectura del hardware de la Raspberry Pi



Nota. En la imagen se muestra la arquitectura del hardware de la Raspberry Pi (Upton & Halfacree, 2016)

En la Figura 3 se observa el pinout correspondiente al modelo de la Raspberry Pi con conector

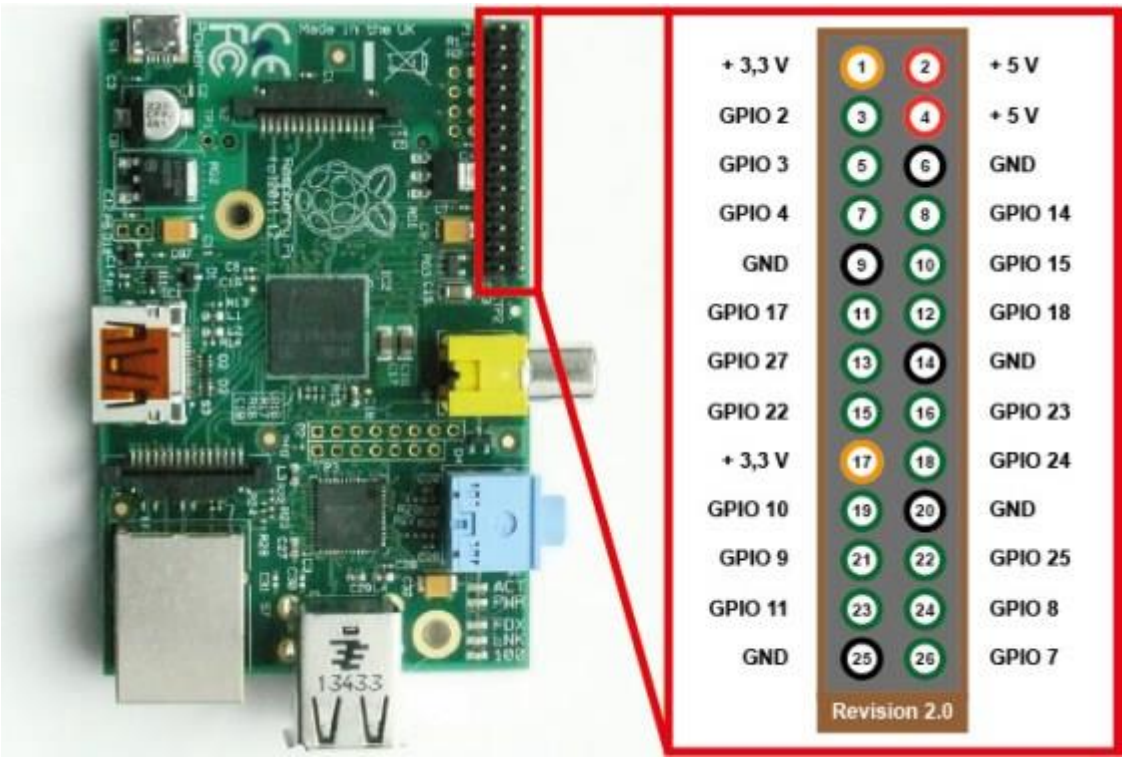
de 26 pines (Revisión 2.0). Cada uno de estos pines tiene funciones específicas, incluyendo alimentación (3.3V/5V), tierra (GND), y canales de comunicación digital como UART, I2C y SPI. Los pines etiquetados como "GPIO" son programables, lo que significa que pueden configurarse como entrada (para leer señales de sensores) o salida (para controlar dispositivos como Leds o relés).

La interfaz GPIO es clave para el desarrollo de prototipos que requieren una interacción física con el entorno. Por ejemplo, en un sistema domótico, los pines GPIO pueden utilizarse para leer valores de sensores de temperatura, gas, humedad o calidad del aire (como el PMS5003 o el BME680), y actuar en consecuencia controlando ventiladores, alarmas o enviando datos a plataformas como ThingSpeak.

Además, su compatibilidad con lenguajes de programación como Python y bibliotecas específicas (por ejemplo, RPi.GPIO o gpiozero) hace que la Raspberry Pi sea accesible para estudiantes e ingenieros que deseen implementar soluciones funcionales sin necesidad de hardware adicional complejo. Véase la Figura 3.

Figura 3.

Asignación de pines GPIO en la Raspberry Pi (Revisión 2.0)



Nota. Distribución de pines de la Raspberry Pi (Upton & Halfacree, 2016)

2.4 Sensor PMS5003

El sensor PMS5003, desarrollado por Plantower, es un dispositivo que mide la concentración de material particulado (PM) en el aire, específicamente PM1.0, PM2.5 y PM10. Estas partículas representan una preocupación significativa en términos de salud ambiental, ya que pueden penetrar profundamente en el sistema respiratorio humano, provocando afecciones respiratorias, cardiovasculares e incluso neurológicas (Organization, 2021).

Este sensor opera mediante el principio de dispersión láser, en el cual un haz de luz láser incide sobre las partículas suspendidas en el aire, y la luz dispersada es medida por un fotodiodo. La señal resultante es procesada internamente para calcular la concentración de partículas en microgramos por metro cúbico (µg/m³), lo que permite obtener una medición precisa en tiempo real (Plantower, 2016a).

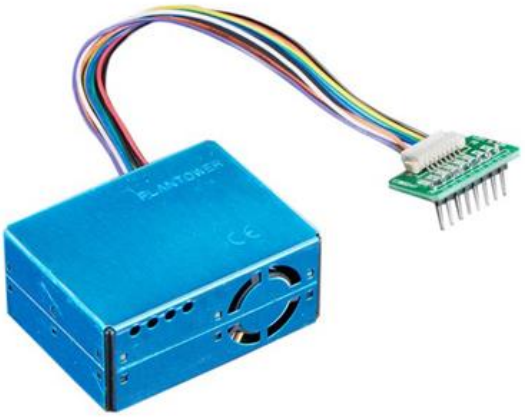
El PMS5003 se ha convertido en una herramienta esencial en proyectos de monitoreo ambiental de bajo costo debido a su alta sensibilidad, tamaño compacto y facilidad de integración con microcontroladores como Arduino y computadoras embebidas como la Raspberry Pi. Esta capacidad de comunicación a través de una interfaz serial UART lo hace ideal para aplicaciones domóticas o de Internet de las Cosas (IoT), donde se requiere una supervisión constante de la calidad del aire (Tomalá & Cely, 2025).

Además, su diseño modular permite su implementación en interiores, como viviendas, oficinas o consultorios, facilitando la implementación de sistemas de alerta ante niveles críticos de contaminación. En la Figura 4 se observa el módulo físico del sensor PMS5003, el cual ha sido integrado en diversos prototipos orientados al monitoreo de ambientes cerrados.

En resumen, la inclusión del PMS5003 en sistemas domóticos permite no solo el control automatizado de ventilación, sino también la recolección de datos históricos que pueden ser útiles para decisiones preventivas o correctivas en entornos residenciales o laborales. Véase la Figura 4.

Figura 4

Sensor PMS5003



Nota. Imagen del sensor Pms5003
(Tomalá & Cely, 2025)

Como se observa en la Figura 5, el PMS5003 incorpora una serie de características técnicas que optimizan su funcionamiento:

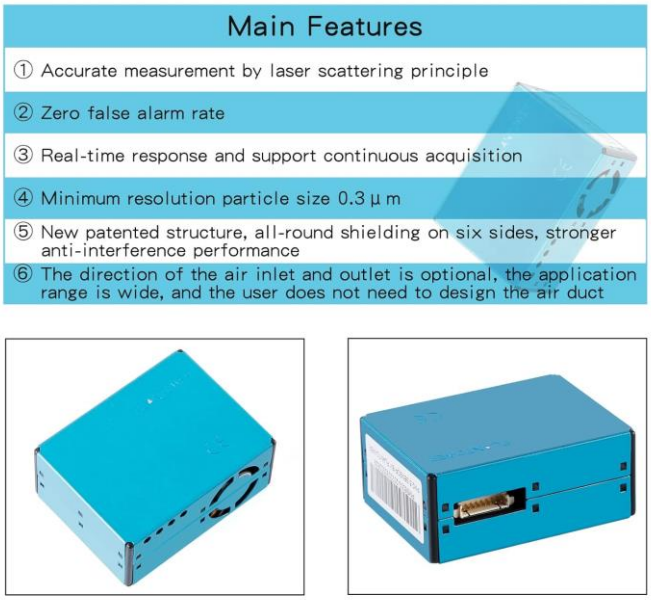
1. Medición precisa mediante el principio de dispersión de luz láser.
2. Tasa de falsas alarmas cero, gracias a su procesamiento digital interno.
3. Adquisición continua y respuesta en tiempo real, lo que permite el monitoreo constante del entorno.
4. Resolución mínima de partículas de 0.3 µm, cubriendo una gama crítica de contaminantes respirables.
5. Estructura patentada con blindaje en seis lados, lo que mejora la protección contra interferencias electromagnéticas externas.
6. Diseño flexible de entrada/salida de aire, que elimina la necesidad de rediseñar el

ducto de ventilación para su integración.

Este sensor se comunica a través de interfaz UART y puede integrarse fácilmente con microcontroladores como ESP32 o plataformas embebidas como Raspberry Pi, lo cual lo hace ideal para proyectos de domótica e Internet de las Cosas (IoT). Además, su encapsulado metálico compacto proporciona durabilidad en ambientes exigentes.

Figura 5

Vista del sensor PMS5003 y principales características técnicas



Nota. Principales características del sensor PMS5003
(Plantower, 2016b)

Como se muestra en la Figura 6, el conector principal del PMS5003 está conformado por ocho pines con funciones específicas:

- PIN1 (VCC): Entrada de alimentación positiva de 5V.
- PIN2 (GND): Tierra o referencia de alimentación.
- PIN3 (SET): Pin de configuración. En nivel alto (3.3V) o flotante, el sensor está activo; en nivel bajo, entra en modo de bajo consumo o reposo.

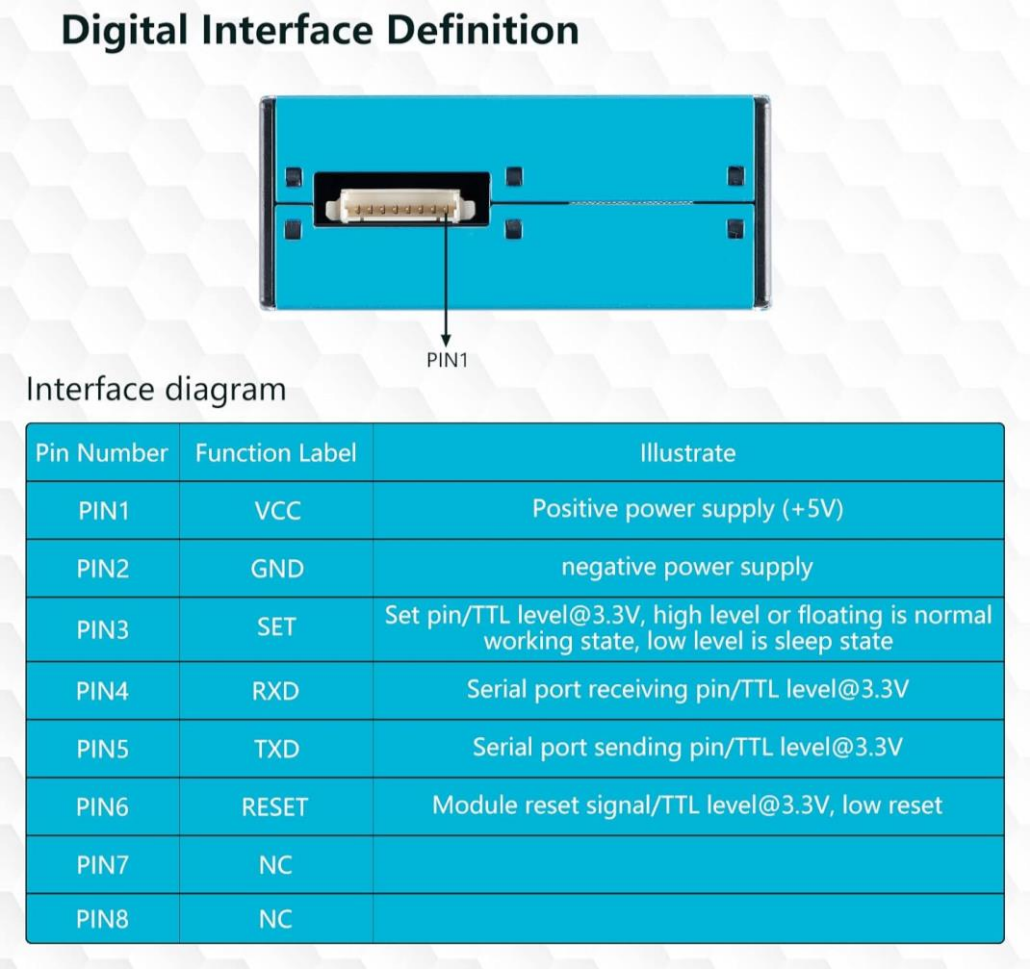
- PIN4 (RXD): Entrada de datos (recepción) de tipo TTL a 3.3V.
- PIN5 (TXD): Salida de datos (transmisión) de tipo TTL a 3.3V.
- PIN6 (RESET): Entrada para reiniciar el módulo. Se activa con un pulso bajo a 3.3V.
- PIN7–PIN8 (NC): Pines sin conexión asignada.

La comunicación del PMS5003 se realiza mediante protocolo UART (Universal Asynchronous Receiver-Transmitter), lo que permite transmitir los datos de concentración de partículas en forma continua a intervalos regulares. Esta interfaz es altamente confiable y ampliamente usada en sistemas embebidos, lo cual facilita su implementación en sistemas de monitoreo de calidad del aire que requieren comunicación directa con plataformas como ThingSpeak o servidores locales.

El diseño de su interfaz también permite funciones avanzadas como el modo de bajo consumo, útil en aplicaciones energéticamente eficientes, y la posibilidad de reinicio remoto para mantenimiento o gestión remota del sistema.

Figura 6

Diagrama de pines del conector digital del PMS5003



Nota. Distribución de pines del sensor PMS5003 (Plantower, 2016a)

El PMS5003 es un sensor óptico de partículas que opera bajo el principio de dispersión de luz láser, permitiendo medir en tiempo real la concentración de material particulado en el aire, como PM1.0, PM2.5 y PM10. La Figura 7 ilustra el diagrama funcional del sensor, el cual describe el flujo interno de datos desde la captación del aire hasta la generación de una señal digital procesada.

El funcionamiento se basa en los siguientes bloques funcionales:

- Fuente láser: Genera un haz de luz láser que atraviesa la cavidad de medición.

- Cavidad de medición por dispersión de luz (Light Scattering Measurement Cavity): El aire que entra al sensor es dirigido hacia esta cavidad. Las partículas presentes dispersan la luz del láser en diferentes direcciones dependiendo de su tamaño.

- Circuito amplificador y de filtrado: La señal eléctrica generada por la dispersión es muy débil, por lo que pasa por un circuito de amplificación que mejora su calidad y elimina el ruido electrónico.

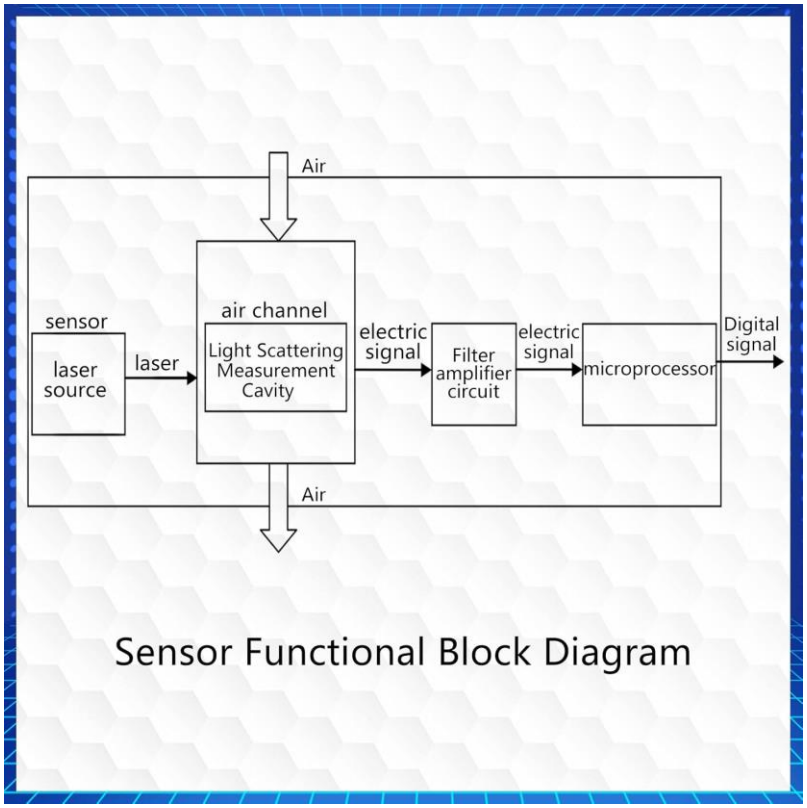
- Microprocesador: Interpreta las señales amplificadas y calcula las concentraciones de partículas. El resultado es convertido en una señal digital lista para ser transmitida a través del puerto UART a un microcontrolador o computadora.

Este diseño permite al PMS5003 ofrecer mediciones precisas, rápidas y confiables incluso para partículas con un tamaño mínimo de 0.3 µm, lo cual es esencial para detectar contaminantes críticos para la salud humana, como polvo fino o aerosoles en entornos clínicos.

La ventaja de este sistema radica en su capacidad de operar de forma continua y autónoma, ideal para aplicaciones en tiempo real en entornos domésticos, urbanos o industriales.

Figura 7

Diagrama funcional del sensor PMS5003



Nota. Esquemático, de cómo funciona el sensor PMS5003 (Plantower, 2016a)

2.4.1 Especificaciones técnicas

- **Rangos de partículas**: 0.3–1.0 µm, 1.0–2.5 µm, 2.5–10 µm
- **Resolución**: 1 µg/m^3; rango útil PM2.5: hasta 500 µg/m^3, máximo ≥1 000 µg/m^3
- **Precisión**: ±10% entre 100–500 µg/m^3 y ±10 µg/m^3 para <100 µg/m^3
- **Corrientes**: consumo activo ≤100 mA, en espera ≤200 µA
- **Alimentación**: 5 V (rango 4.5–5.5 V); lógica TTL de 3.3 V
- **Temperatura y humedad operativas**: –10 °C a +60 °C, 0–99% HR (sin condensación)
- **Tamaño**: aproximadamente 50 × 38 × 21 mm; peso 40–42 g (con cable/breakout)

2.5 Sensor BME680

El BME680, desarrollado por Bosch Sensortec, es un sensor ambiental digital que integra en un solo chip la medición de temperatura, humedad relativa, presión barométrica y compuestos

orgánicos volátiles (VOC, por sus siglas en inglés). Esta combinación lo convierte en una solución compacta y eficiente para el monitoreo integral de la calidad del aire en espacios cerrados (Sensortec, 2023b).

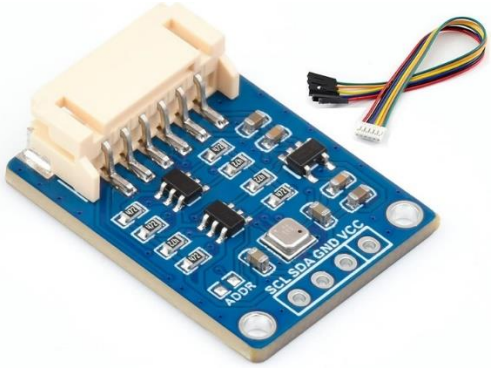
A diferencia de sensores tradicionales que requieren múltiples módulos independientes para medir estos parámetros, el BME680 proporciona una ventaja significativa en términos de espacio, consumo energético y facilidad de integración con plataformas embebidas como Raspberry Pi o microcontroladores como el ESP32. Su salida digital y compatibilidad con interfaces I2C y SPI permiten una comunicación directa y precisa con diversos sistemas de adquisición de datos (Sensortec, 2023b).

Uno de los aspectos más destacables del BME680 es su capacidad para detectar VOC, compuestos químicos presentes en el aire que pueden provenir de materiales de limpieza, pinturas, solventes, productos dentales, entre otros. En entornos clínicos y residenciales, estos compuestos pueden tener un impacto directo en la salud respiratoria, especialmente cuando se acumulan en espacios mal ventilados. Por ello, su inclusión en sistemas de domótica o proyectos IoT orientados a la salud ambiental resulta particularmente útil.

El sensor utiliza un sistema de calentamiento interno controlado y un elemento sensible basado en óxidos metálicos para detectar cambios en la calidad del aire. Esta información puede ser utilizada para activar alertas automáticas, regular sistemas de ventilación o realizar análisis de tendencias a lo largo del tiempo. Gracias a su precisión y bajo consumo, el BME680 es una herramienta adecuada para aplicaciones de monitoreo ambiental en tiempo real tanto en hogares como en instituciones médicas. En la figura 8 se observa el módulo BME 680

Figura 8

Sensor BME680



Nota. Imagen del sensor BME680 (Sensortec, 2023)

El BME680 es un sensor ambiental digital desarrollado por Bosch Sensortec, que permite medir simultáneamente temperatura, humedad, presión barométrica y gases (compuestos orgánicos volátiles, VOC). Estos parámetros son fundamentales para evaluar la calidad del aire en espacios cerrados como hogares, oficinas, hospitales y laboratorios. El módulo mostrado en la Figura 9 corresponde a una versión de Waveshare, compatible con plataformas como Raspberry Pi, ESP32 y Arduino, gracias a sus salidas de comunicación I2C y SPI (Waveshare, 2024).

Este sensor incluye una interfaz flexible que permite su configuración mediante dos tipos de protocolos de comunicación digital: I2C (Inter-Integrated Circuit) y SPI (Serial Peripheral Interface). En la figura, se pueden observar claramente los pines que permiten esta doble funcionalidad, facilitando la integración del BME680 en diferentes arquitecturas de hardware .

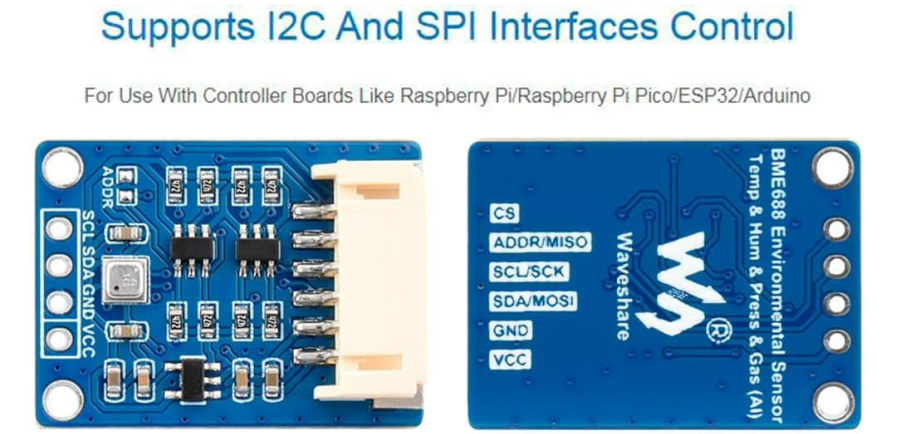
En modo I2C, la comunicación se establece mediante los pines SCL (clock) y SDA (data), mientras que en SPI se utilizan los pines CS, SCK, MISO y MOSI. La selección de dirección I2C puede cambiarse mediante el puenteado del pin ADDR, permitiendo utilizar múltiples sensores en una sola línea sin conflicto de direcciones (por defecto 0x77; alternativo 0x76).
El sensor opera con un voltaje de alimentación de 3.3V o 5V, lo cual lo hace adaptable a una amplia variedad de microcontroladores. Su capacidad de detectar compuestos volátiles orgánicos mediante un elemento sensor de gas integrado permite una estimación precisa de la calidad del aire interior, lo cual es esencial para sistemas de ventilación inteligente, control climático automatizado y aplicaciones médicas, especialmente donde se manejan desinfectantes o materiales químicos (Sensortec, 2023a).

El uso del BME680 en entornos domóticos o de IoT no solo permite la monitorización en

tiempo real de parámetros ambientales, sino que también contribuye a la toma de decisiones automáticas dentro de sistemas inteligentes. Por ejemplo, puede activar extractores, enviar alertas o modificar el flujo de aire según los niveles detectados. Esto convierte al BME680 en un componente esencial dentro de proyectos de monitoreo ambiental eficiente y adaptable. Véase la Figura 9.

Figura 9

Sensor BME680, interfaces de control



PIN	I2C	SPI
CS	NC	SPI chip selection, low active
ADDR/MISO	I2C address chip selection high level (default) address: 0x77 Shorting onboard ADDR pad, the address is 0x76	SPI data master input/slave output
SCL/SCK	I2C clock	SPI clock input
SDA/MOSI	I2C data	SPI data master output/slave input
GND	Ground	
VCC	3.3V/5V power supply	

Nota. Interfaces de control y tipos de comunciaciones (Sensortec, 2023)

Los sensores de la familia BME de Bosch Sensortec son ampliamente utilizados en aplicaciones de monitoreo ambiental debido a su alta precisión, tamaño compacto y facilidad

de integración mediante interfaces digitales como I2C y SPI. En la tabla 1 se muestra una comparación de los modelos BME280, BME680 y BME688, los cuales ofrecen funcionalidades progresivas en cuanto a capacidad de detección de gases y precisión.

- El BME280 proporciona mediciones de presión barométrica, temperatura ambiental y humedad relativa. Es ideal para sistemas básicos de monitoreo climático.

- El BME680 agrega la capacidad de detección de gases orgánicos volátiles (VOC), lo que permite inferir la calidad del aire interior. Además, puede integrarse con software que calcula el IAQ (Índice de Calidad del Aire).

- El BME688 expande esta funcionalidad al incluir detección de compuestos sulfurados volátiles (VSC) y capacidades de aprendizaje automático (IA integrada) para reconocer patrones de gases específicos en combinación con bibliotecas de software entrenadas.

Todos los modelos operan en un rango de temperatura de -40 a 85 °C con diferentes precisiones (hasta ±0.5 °C en el BME688) y ofrecen rangos de presión entre 300 y 1100 hPa, adecuados para aplicaciones meteorológicas, médicas, agrícolas y de automatización del hogar. La compatibilidad con múltiples interfaces de comunicación facilita su integración en microcontroladores como el ESP32 o plataformas como la Raspberry Pi.

Estas características convierten a los sensores BME en herramientas versátiles para sistemas embebidos de IoT, especialmente en proyectos de domótica inteligente donde la calidad del aire y el confort térmico son parámetros críticos para el bienestar humano. Ver Tabla 1

Tabla 1

Comparación de parámetros técnicos entre sensores BME280, BME680 y BME688

Parámetro	BME280	BME680	BME688
Detección soportada	Presión barométrica, temperatura ambiental, humedad relativa	Presión barométrica, temperatura ambiental, humedad relativa, detección de cambios de VOC (IAQ con software)	Presión barométrica, temperatura ambiental, humedad relativa, detección de cambios de VOC y VSC (IAQ con software e IA integrada)
Interfaces de comunicación	I²C y SPI	I²C y SPI	I²C y SPI
Rango de temperatura	-40 a 85 °C	-40 a 85 °C	-40 a 85 °C

Precisión de temperatura	±1.0 °C (0 a 65 °C)	±1.0 °C (0 a 65 °C)	±0.5 °C (0 a 65 °C)
Rango de humedad	0 – 100 %RH	0 – 100 %RH	0 – 100 %RH
Precisión de humedad	±3 %RH	±3 %RH	±3 %RH
Rango de presión	300 – 1100 hPa	300 – 1100 hPa	300 – 1100 hPa
Precisión de presión	±1.0 hPa (0 a 65 °C)	±0.6 hPa (0 a 65 °C)	±0.6 hPa (0 a 65 °C)
Detección de gases	No soporta	Soporta detección de cambios de VOC	Soporta detección de cambios de VOC y VSC
Dimensiones	27 × 20 mm	27 × 20 mm	27 × 20 mm

Nota. Esta tabla hace una comparación entre datos técnicos de los sensores (Sensortec, 2023a)

2.6 ThingSpeak y Visualización de Datos

ThingSpeak es una plataforma en la nube basada en el protocolo HTTP desarrollada por MathWorks, diseñada específicamente para el Internet de las Cosas (IoT). Su funcionalidad principal es la recolección, almacenamiento, análisis y visualización de datos en tiempo real provenientes de sensores conectados a microcontroladores como el ESP32 o la Raspberry Pi. Esta herramienta permite que los dispositivos publiquen datos mediante solicitudes HTTP o MQTT, siendo ampliamente utilizada por desarrolladores, ingenieros e investigadores en aplicaciones de monitoreo ambiental, automatización del hogar, salud y agricultura inteligente (MathWorks, 2024).

Una de las características más destacadas de ThingSpeak es su interfaz gráfica integrada, que permite a los usuarios crear gráficos dinámicos sin necesidad de conocimientos avanzados en programación o desarrollo web. Cada canal creado en ThingSpeak puede contener múltiples campos que representan diferentes variables, y puede configurarse para visualizar datos en formas como gráficos de líneas, barras, medidores o mapas geográficos si se utilizan coordenadas GPS (Vikram & Agrawal, 2022).

Además, ThingSpeak permite ejecutar código en MATLAB dentro de la misma plataforma, lo que potencia las capacidades analíticas del sistema, permitiendo aplicar operaciones estadísticas, filtrado de datos, predicciones y detección de anomalías. Esta integración convierte a ThingSpeak no solo en una plataforma de visualización, sino también en un

entorno para el análisis y la toma de decisiones basada en datos (Zhou et al., 2021).

En sistemas domóticos o prototipos IoT como los desarrollados con ESP32 y sensores ambientales (por ejemplo, PMS5003 o BME680), ThingSpeak actúa como una solución eficiente y de bajo costo para visualizar el comportamiento de las variables ambientales en tiempo real. A través de su entorno web, los usuarios pueden consultar remotamente los datos recolectados, configurar alertas, compartir visualizaciones y exportar registros históricos para análisis externo.

2.7 Guías de calidad del aire de la OMS

La calidad del aire es uno de los principales determinantes ambientales de la salud humana. En respuesta a la creciente evidencia científica sobre los efectos adversos de la contaminación del aire, la Organización Mundial de la Salud (OMS) actualizó en el año 2021 sus guías globales de calidad del aire, proporcionando límites más estrictos para las concentraciones de contaminantes atmosféricos, particularmente las partículas finas en suspensión (Organization, 2021b)

2.7.1 Límites recomendados por la OMS

Entre los contaminantes más peligrosos para la salud se encuentran las partículas $PM_{2.5}$ (diámetro ≤ 2.5 µm) y PM_{10} (diámetro ≤ 10 µm), las cuales pueden penetrar profundamente en los pulmones e incluso ingresar al torrente sanguíneo.

Los límites actualizados son los siguientes:

- $PM_{2.5}$ (partículas finas):
    - Promedio anual recomendado: ≤ 5 µg/m³
    - Promedio de 24 horas: ≤ 15 µg/m³

- PM_{10} (partículas gruesas):
    - Promedio anual recomendado: ≤ 15 µg/m³
    - Promedio de 24 horas: ≤ 45 µg/m³

Estas recomendaciones representan un endurecimiento significativo respecto a las guías

previas del año 2005, que establecían un límite anual de **10 µg/m³** para $PM_{2.5}$, evidenciando el creciente consenso científico sobre los riesgos para la salud incluso a bajas concentraciones.

2.7.2 Comparación con estándares internacionales

Diversas regiones del mundo adoptan sus propios límites legales de calidad del aire, que en muchos casos son menos estrictos que los sugeridos por la OMS:

- En Estados Unidos, la Agencia de Protección Ambiental (EPA) fija un límite legal para $PM_{2.5}$ de 9 µg/m³ anuales y 35 µg/m³ en 24 horas.
- En la Unión Europea, el límite legal para PM_{10} es de 50 µg/m³ diarios, con un máximo de 35 superaciones por año.

Estas variaciones reflejan la complejidad de implementar estándares basados puramente en evidencia científica dentro de contextos regulatorios, sociales y económicos diversos.

2.7.3 Impacto en la salud pública

Numerosos estudios epidemiológicos y revisiones sistemáticas han establecido una sólida relación entre la exposición crónica a partículas finas y múltiples enfermedades:

- $PM_{2.5}$ está asociada con enfermedades cardiovasculares, enfermedades pulmonares como EPOC, asma, neumonías, así como con diabetes tipo 2, cáncer de pulmón y complicaciones durante el embarazo.
- Según el informe *State of Global Air*, en 2019, la contaminación por $PM_{2.5}$ contribuyó a más de 4 millones de muertes a nivel mundial.
- Un metaanálisis global indicó que por cada aumento de 10 µg/m³ de $PM_{2.5}$, la mortalidad por enfermedades cardiovasculares y pulmonares se incrementa entre 6 y 13 %, y el riesgo de cáncer de pulmón en al menos un 9 %.
- Investigaciones recientes en 2025 han vinculado este mismo incremento de $PM_{2.5}$ con un aumento del 17 % en el riesgo de demencia.

Estas cifras subrayan la necesidad de implementar políticas públicas más rigurosas y sistemas de monitoreo precisos que permitan alertar a la población ante niveles peligrosos de exposición.

2.7.4 Casos reales y mediciones recientes

Ejemplos recientes ilustran cómo los niveles de partículas superan con frecuencia los estándares recomendados:

- En Nagpur (India), durante el primer cuatrimestre de 2025, los niveles promedio de $PM_{2.5}$ alcanzaron 46 µg/m³, lo cual equivale, en términos de exposición, a fumar dos cigarrillos diarios.
- En Estados Unidos, durante incendios forestales entre 2006 y 2020, los niveles de $PM_{2.5}$ se incrementaron exponencialmente, contribuyendo a más de 15 000 muertes atribuibles al humo.

Estos eventos resaltan la importancia de utilizar sensores como el PMS5003, que permiten detectar concentraciones en tiempo real y activar mecanismos de respuesta oportuna para mitigar los efectos sobre la salud.

3 Marco metodológico

Este capítulo presenta de forma exhaustiva el marco metodológico del proyecto, abordando cada componente con el nivel de detalle requerido para sustentar el desarrollo del sistema IoT para la medición de la calidad del aire y parámetros ambientales en un consultorio odontopediátrico.

3.1 Enfoque de la investigación

La investigación adoptó un enfoque cuantitativo, orientado a la recolección de datos numéricos y su análisis estadístico. La elección se justifica en la naturaleza objetiva de las variables (concentraciones de partículas, temperatura, humedad, gas), las cuales requieren mediciones precisas para evaluar su impacto sobre la calidad del aire. El diseño es experimental aplicado, ya que se desarrolló un prototipo, se implementó en un entorno real y se analizaron sus resultados para verificar su eficacia.

3.2 Tipo y alcance de la investigación

El trabajo es de tipo tecnológica aplicada, con un alcance descriptivo ya que documenta el comportamiento de las variables ambientales y explicativo ya que relaciona los cambios en estas variables con factores del entorno odontopediátrico. Este doble alcance permite no solo describir el fenómeno, sino comprender las causas de las variaciones observadas.

3.3 Diseño metodológico

El diseño de la investigación se dividió en las siguientes fases:

- **Análisis de requerimientos:** identificación de variables a medir y condiciones del entorno clínico.
- **Selección tecnológica:** evaluación de sensores y plataformas con base en precisión, costo y compatibilidad.
- **Implementación técnica:** desarrollo del hardware y software del sistema.
- **Pruebas controladas en laboratorio:** para verificar la correcta integración y calibración.
- **Pruebas en campo:** instalación en el consultorio odontopediátrico.
- **Evaluación de resultados:** análisis comparativo con estándares OMS.

3.4 Población y muestra

La población de referencia está compuesta por un consultorio odontopediátrico en Samborondón. Se seleccionó una muestra no probabilística intencional, la Clínica MOM por su disponibilidad, infraestructura y relevancia para el estudio.

3.5 Instrumentos de recolección de datos

- **PMS5003:** sensor óptico para PM1.0, PM2.5 y PM10.
- **BME680:** sensor multifunción para temperatura, humedad, presión y VOC.

- **Raspberry Pi:** microcomputadora para control y procesamiento.
- **ThingSpeak:** plataforma cloud para almacenamiento y visualización.
- **Módulo de alertas:** sistema automatizado de notificación por correo electrónico.

Cada instrumento fue calibrado y validado con datos de referencia.

3.6 Procedimiento detallado

1. Configuración del sistema operativo en la Raspberry Pi.
2. Conexión física de sensores y pruebas de comunicación.
3. Desarrollo de scripts en Python para adquisición, filtrado y envío de datos.
4. Configuración de la base de datos y panel en ThingSpeak.
5. Pruebas en laboratorio para verificar estabilidad y precisión.
6. Instalación del sistema en la clínica MOM.
7. Monitoreo continuo, con registro de incidencias.

3.7 Análisis y procesamiento de datos

El procesamiento y análisis de los datos ambientales obtenidos a partir de los sensores PMS5003 y BME680 se llevó a cabo a través de la plataforma ThingSpeak, la cual permitió almacenar, visualizar y gestionar la información en la nube en tiempo real. Esta plataforma facilitó la creación de gráficos temporales automáticos y la organización de las variables en canales independientes para su análisis detallado.

Se aplicaron estadísticas descriptivas como promedios, valores máximos y mínimos, y desviaciones estándar, con el objetivo de caracterizar el comportamiento de los parámetros monitoreados durante distintos períodos del día y en diferentes condiciones clínicas.

Adicionalmente, se realizó un análisis comparativo de los datos registrados con los valores de referencia establecidos por la Organización Mundial de la Salud (OMS) para contaminantes como partículas en suspensión (PM2.5, PM10) y compuestos orgánicos volátiles (VOC), entre otros.

La visualización de los datos se presentó en forma de gráficos temporales de series de tiempo y diagramas de dispersión, lo que permitió identificar patrones recurrentes, picos de

contaminación, y condiciones ambientales anómalas. Esta representación gráfica resultó fundamental para interpretar el impacto de ciertas actividades clínicas (como tratamientos odontológicos que generan aerosoles) sobre la calidad del aire.

Gracias a la capacidad de ThingSpeak para ejecutar funciones de procesamiento directamente en la nube, también fue posible programar alertas automáticas por correo electrónico cuando los valores superaron umbrales predefinidos, fortaleciendo así la capacidad de respuesta en tiempo real ante condiciones no saludables.

Este enfoque metodológico permitió no solo almacenar y analizar grandes volúmenes de datos ambientales de forma eficiente, sino también ofrecer una herramienta útil para la toma de decisiones en la gestión clínica de espacios pediátricos.

3.8 Validación del sistema

Se realizaron pruebas repetidas en distintas franjas horarias y condiciones de uso. Se evaluó la latencia de transmisión y la consistencia de los datos frente a mediciones de referencia.

3.9 Consideraciones éticas

No se interactuó con pacientes ni se recopilaron datos personales. Se cumplieron normas de bioseguridad y se obtuvo consentimiento de la administración del consultorio para la instalación del sistema.

3.10 Limitaciones metodológicas

Limitaciones incluyen el número reducido de puntos de medición, el corto periodo de prueba y la dependencia de la conectividad a Internet para la transmisión de datos.

3.11 Enfoque de la Propuesta del proyecto de tesis

La presente propuesta busca dar solución al problema identificado mediante el diseño e implementación de un sistema inteligente de monitoreo ambiental, integrando tecnologías de bajo costo, conectividad mediante IoT y almacenamiento en la nube. Este sistema está orientado a la medición en tiempo real de la calidad del aire en consultorios odontopediátricos, un entorno clínico particularmente sensible por el tipo de pacientes que atiende y las condiciones cerradas en las que se realizan los procedimientos.

La solución propuesta aprovecha la versatilidad de la microcomputadora Raspberry Pi como plataforma de control y procesamiento, junto con sensores de alta precisión como el PMS5003, que mide partículas en suspensión (PM1.0, PM2.5, PM10), y el BME680, que permite el registro de temperatura, humedad, presión atmosférica y compuestos orgánicos volátiles (VOC).

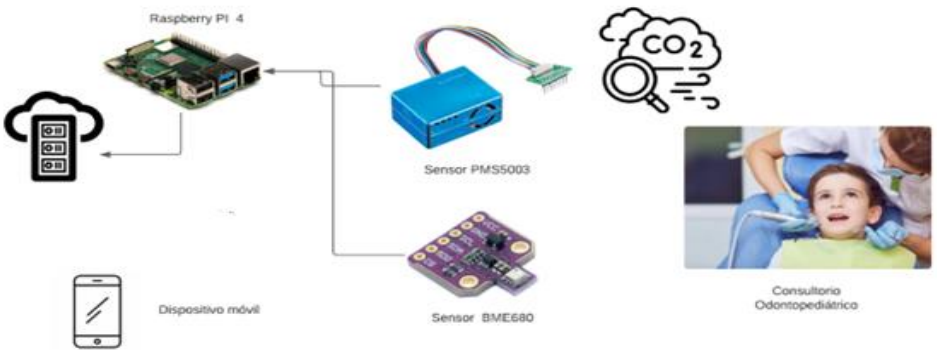
Este sistema tiene como principales objetivos:

- Monitorear en tiempo real los parámetros ambientales críticos.
- Visualizar los datos históricos y actuales desde cualquier dispositivo conectado.
- Generar alertas automáticas ante condiciones que representen riesgo para la salud respiratoria.

En la figura 10 se observa el esquema propuesto en este proyecto de investigación.

Figura 10

Esquema del prototipo IoT de monitorización del aire en consultorios odontopediátrico



Nota. Esquema del prototipo IoT.

3.12 Componentes del Sistema

Los elementos utilizados para la implementación del prototipo son:

- **Raspberry Pi 4** (unidad de procesamiento y control).
- **Sensor PMS5003** (medición de partículas suspendidas).

- **Sensor BME680** (medición de VOC, temperatura, humedad y presión).
- **Módulo Wi-Fi integrado**.
- **Case o carcasa protectora para el prototipo**.
- **ThingSpeak** (interfaz para visualización de datos).
- **Script en Python** (lectura de sensores, envío de datos, alertas automáticas por correo electrónico).

3.13 Funcionamiento del Sistema

El sistema opera mediante una serie de procesos automatizados:

1. Captura de datos en tiempo real desde los sensores PMS5003 y BME680.
2. Procesamiento local de los datos en la Raspberry Pi.
3. Envío de los datos a ThingSpeak mediante conexión Wi-Fi.
4. Visualización remota a través de un panel de control accesible desde cualquier navegador.
5. Generación de alertas por correo electrónico cuando los parámetros superan los valores críticos establecidos.

Esta metodología permite una gestión ambiental eficiente, precisa y adaptable a otros entornos médicos o educativos.

3.14 Ventajas del Sistema

- Mejora la seguridad sanitaria del consultorio odontopediátrico.
- Permite la gestión remota de variables ambientales.
- Representa un sistema de bajo costo y fácil replicación.
- Su diseño es escalable, lo que permite aplicarlo a otros espacios clínicos o escolares.
- Constituye una herramienta educativa para promover el uso de la tecnología en la atención infantil.

3.15 Construcción del prototipo de monitorización IoT – Conexión de Hardware

3.15.1 Materiales empleados

Para la construcción del prototipo del sistema de monitoreo ambiental, se utilizaron

componentes electrónicos de bajo costo, pero con alta capacidad de procesamiento y comunicación. Entre ellos destacan: una Raspberry Pi 1 Model B como unidad central de procesamiento, el sensor PMS5003 para la medición de material particulado (PM1.0, PM2.5 y PM10), y el sensor BME680 para registrar temperatura, humedad, presión atmosférica y compuestos orgánicos volátiles (VOC). Se utilizaron además cables Dupont de diversos colores para facilitar las conexiones, una carcasa protectora tipo ElectroCookie de aluminio para proteger la Raspberry Pi, y una fuente de alimentación adecuada. La visualización y almacenamiento de los datos se realizó mediante la plataforma en la nube ThingSpeak. Todos los componentes utilizados se presentan en las Figuras 11 y 12.

Figura 11

Componentes del Prototipo IoT

Figura 12

Raspberry Pi 1 Model B

3.15.2 Conexión del sensor PMS5003 (material particulado)

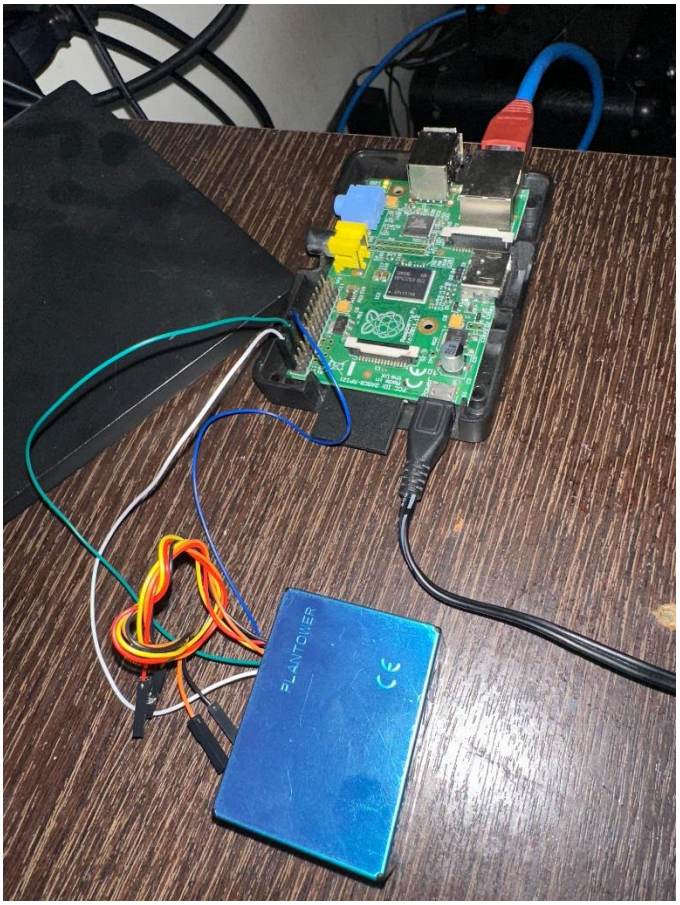
El sensor PMS5003 se comunica mediante el protocolo UART, por lo que fue conectado directamente a los pines GPIO correspondientes de la Raspberry Pi. La conexión se realizó de la siguiente manera:

- El pin VCC del sensor fue conectado al pin 2 (5V) de la Raspberry Pi.
- El pin GND del sensor fue conectado al pin 6 (GND).
- El pin TX del PMS5003 fue conectado al pin 10 (GPIO15 / RXD).
- El pin RX del PMS5003 fue conectado al pin 8 (GPIO14 / TXD).

Debido a que el PMS5003 opera con lógica de 5V y la Raspberry Pi con 3.3V, se consideró el uso de un divisor de voltaje o un conversor de nivel lógico en la línea RX para evitar daños a la placa. La conexión física de este sensor puede observarse en la Figura 13, donde se aprecian claramente los cables de transmisión de datos conectados a la Raspberry Pi.

Figura 13

Sensor PMS5003 con Raspberry Pi 1 model B



3.15.3 Conexión del sensor BME680 (ambiente general y VOC)

El sensor BME680, que mide temperatura, humedad, presión atmosférica y compuestos orgánicos volátiles, utiliza el protocolo I²C para comunicarse con la Raspberry Pi. La conexión se realizó como sigue:

- El pin **VCC** del sensor fue conectado al **pin 1 (3.3V)** de la Raspberry Pi.
- El pin **GND** fue conectado al **pin 9 (GND)**.
- El pin **SCL** fue conectado al **pin 5 (GPIO3 / SCL)**.
- El pin **SDA** fue conectado al **pin 3 (GPIO2 / SDA)**.

Tras realizar estas conexiones, se validó la presencia del sensor mediante el comando i2cdetect -y 1, el cual mostró la dirección I²C asignada automáticamente. Esta conexión se encuentra ilustrada en la Figura 14.

Figura 14

Sensor BME680

3.15.4 Ensamblaje físico del sistema

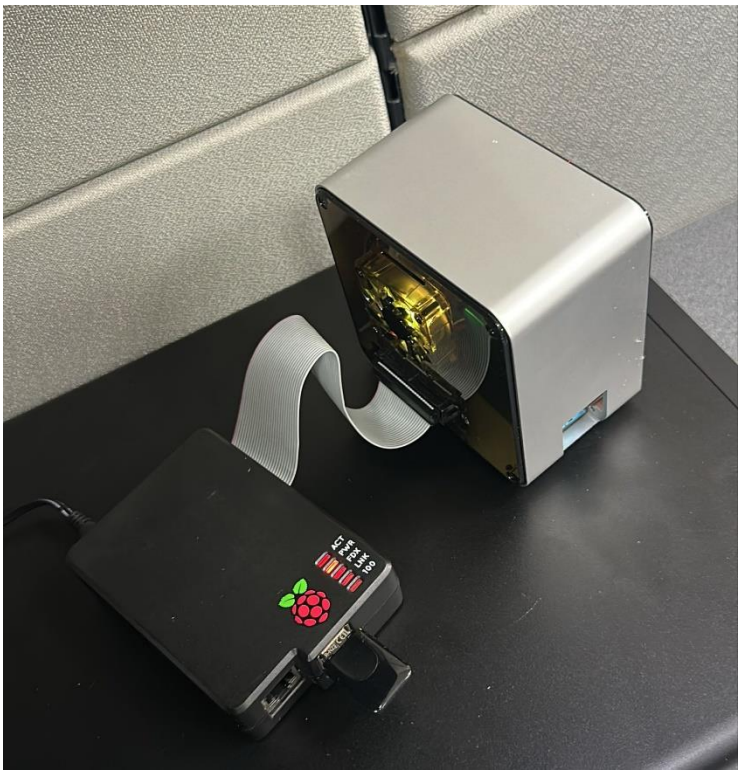
Una vez verificado el correcto funcionamiento de ambos sensores y la transmisión de datos a través de los protocolos configurados, se procedió al ensamblaje definitivo del sistema. La Raspberry Pi fue colocada dentro de una carcasa metálica ElectroCookie, que proporciona protección mecánica y ayuda a disipar el calor generado por el dispositivo. Los sensores fueron ubicados externamente, conectados mediante cables extensores para facilitar su distribución en puntos estratégicos dentro del consultorio odontopediátrico. En las Figuras 15 y 16 se puede observar el ensamblaje completo del sistema con todos los componentes conectados y operativos.

Figura 15

Ensamblaje de Prototipo vista frontal

Figura 16

Ensamblaje de Prototipo vista aérea

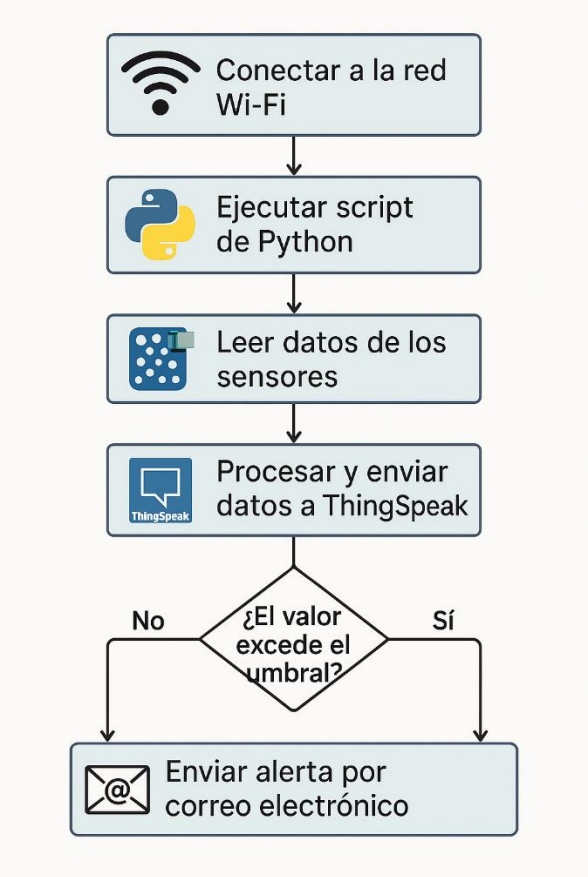


3.15.5 Prueba funcional e integración con la nube (ThingSpeak)

Con el sistema ensamblado, se procedió a conectarlo a una red Wi-Fi estable y ejecutar un script en Python diseñado para la adquisición y envío de datos. Este script recoge la información desde ambos sensores, procesaba los datos (aplicando promedios y detección de valores atípicos), y los envía a ThingSpeak, donde se configuraron canales específicos para cada parámetro. Además, se incluyó una función para el envío automático de alertas por correo electrónico cuando los valores superaban umbrales críticos. El flujo completo del sistema desde la recolección de datos hasta la visualización remota se representa en la Figura 17.

Figura 17

Flujo de la obtención de datos de los sensores en prototipo IoT



3.15.6 Validación en ambiente real

El prototipo fue instalado en un consultorio odontopediátrico real con el objetivo de validar su funcionamiento bajo condiciones operativas normales. Se monitorearon parámetros clave como partículas PM2.5, temperatura y niveles de VOC durante varias sesiones, comprobando la estabilidad del sistema, la fidelidad de los datos registrados, y la efectividad de las alertas configuradas. Esta fase permitió además identificar posibles mejoras en la disposición de sensores y la calibración inicial..

3.16 Configuración de script de Python en Raspberry Pi

En esta sección se explica el script que configuró en Python y es ejecutado al iniciar la Raspberry Pi. El script se observa en la sección de Anexos.

El objetivo del script es leer periódicamente variables de calidad de aire y ambiente, evaluarlas frente a umbrales de referencia (incluyendo límites de la OMS para partículas), generar advertencias locales y enviar los datos a la plataforma ThingSpeak para su almacenamiento y análisis.

3.16.1 Entorno y dependencias

- **Lenguaje:** Python 3 (salida estándar configurada en UTF-8 para soportar acentos y símbolos).
- **Sensores y buses:**
    - **PMS5003** (material particulado): comunicación UART por defecto (GPIO 14/15 en Raspberry Pi).
    - **BME680** (temperatura, humedad, presión y gas): comunicación I²C (direcciones 0x76/0x77).

- **Librerías principales:**
    - pms5003 (lectura del sensor PM).
    - adafruit_bme680 + board + busio (lectura BME680 vía I²C).
    - requests (HTTP POST a ThingSpeak).

- socket, time, sys (utilidades de red, temporización y consola).

3.16.2 Parámetros y umbrales utilizados

- **Límites OMS (µg/m³):**
  - PM2.5 (promedio 24 h): **15**
  - PM10 (promedio 24 h): **45**
  - PM1.0: sin estándar OMS (se informa “Sin estándar OMS”).

- **Umbrales ambientales locales:**
  - Temperatura alta: **40,0 °C**
  - Humedad elevada: **70 %**
  - Gas elevado (resistencia BME680): **10 000 Ω**

- **Periodo de muestreo/envío: 15 min** (900 s) entre lecturas.
- **ThingSpeak:** URL de actualización /update y **API Key** de escritura (se recomienda moverla a variable de entorno por seguridad).

3.16.3 Mapeo de variables hacia el canal ThingSpeak

- **field1:** PM1.0
- **field2:** PM2.5
- **field3:** PM10
- **field4:** Temperatura (°C)
- **field5:** Humedad relativa (%)
- **field6:** Presión (hPa)
- **field7:** Gas (Ω)

3.16.4 Arquitectura del programa

3.16.4.1 Funciones principales

- **evaluar_pm(valor, limite)**
  Compara el valor de PM con su límite OMS. Retorna “DENTRO” si no excede el límite, “FUERA” si lo supera y “Sin estándar OMS” cuando el límite es None (PM1.0).

- **evaluar_ambiente(temp, hum, gas)**

Evalúa condiciones de **temperatura alta**, **humedad y gas elevados**. Devuelve una lista de advertencias textuales para impresión local.

- **enviar_a_thingspeak(pm1, pm25, pm10, temp, hum, presión, gas)**
  Construye el payload con la API Key y los fieldN, realiza un **POST** con requests y verifica respuesta:
  - Éxito: código 200 y entry_id numérico.
  - Falla: impresión del código/razón o excepción capturada.

- **esperar_internet()**
  Bloquea la ejecución hasta que el DNS resuelva correctamente api.thingspeak.com. Evita intentos de envío sin conectividad.

- **init_bme680(i2c)**
  Intenta detectar el BME680 en **0x76** y, si falla, en **0x77**. Si no se detecta el sensor, el programa se detiene con mensaje de error.

- **main()**
  Orquesta todo el flujo: espera Internet, inicializa PMS5003 e I²C+BME680, entra al bucle de adquisición, evalúa umbrales, imprime advertencias y **envía** datos a ThingSpeak cada 15 min.

3.16.5 Paso a paso de la ejecución

1. **Configuración de salida UTF-8.**
   Se reconfigura la consola para imprimir caracteres en español y símbolos (emojis de advertencia) sin errores de codificación.

2. **Definición de constantes.**
   Se cargan los límites OMS para PM2.5/PM10, los umbrales ambientales (temperatura, humedad, gas), el intervalo de muestreo (15 min) y los parámetros de ThingSpeak (URL y API Key).

3. **Comprobación de conectividad.**
   esperar_internet() intenta resolver por DNS el host de ThingSpeak; si no es posible,

espera 10 s y reintenta en bucle hasta lograrlo.

4. **Inicialización de sensores.**
   - **PMS5003**: se crea el objeto lector sobre UART por defecto.
   - **BME680**: se inicializa el bus I²C y se invoca init_bme680(i2c); si no se detecta en 0x76 ni 0x77, el programa finaliza para evitar lecturas inválidas.

5. **Inicio del bucle infinito de adquisición.**
   A partir de aquí, el sistema opera en ciclos de 15 minutos.

6. **Lectura del PMS5003 (PM1.0, PM2.5, PM10).**
   Se invoca pms5003.read(). Si se produce un **timeout** de lectura (ReadTimeoutError), se informa por consola, se espera 5 s y se **reanuda** el ciclo (sin detener la aplicación).

7. **Lectura del BME680 (T, HR, P, Gas).**
   Se leen: temperatura (°C), humedad relativa (%), presión (hPa) y resistencia de gas (Ω).

8. **Evaluación frente a límites OMS (PM).**
   - PM1.0 se reporta como “Sin estándar OMS”.
   - PM2.5 y PM10 se comparan contra 15 µg/m³ y 45 µg/m³, respectivamente; se etiqueta “DENTRO”/“FUERA”.
9. **Evaluación de condiciones ambientales.**
   evaluar_ambiente revisa:
   - Temperatura > 40 °C → “Temperatura ALTA”.
   - Humedad > 70 % → “Humedad ELEVADA”.
   - Gas > 10 000 Ω → “Calidad del aire DEFICIENTE (gas elevado)”.

10. **Impresión local de resultados.**
    Se muestra un bloque con todas las magnitudes y su estado, más las advertencias que correspondan.

11. **Construcción del payload para ThingSpeak.**
    Se asocian los valores a field1…field7 según el mapeo del canal.

12. **Envío a ThingSpeak.**
    enviar_a_thingspeak realiza el **POST** y gestiona:
    - **OK**: imprime ThingSpeak OK, entry_id = ….
    - **Error**: imprime el código y el cuerpo de la respuesta (útil para diagnóstico).
    - **Excepción**: captura cualquier fallo de red y lo informa.

13. **Espera hasta la próxima muestra.**
    Se suspende la ejecución durante **INTERVALO = 900 s**.

14. **Manejo de interrupción manual.**
    Si el usuario cancela el programa (Ctrl+C), se captura KeyboardInterrupt y se finaliza de forma limpia con un mensaje.

3.16.6 Manejo de errores y robustez

- **Conectividad:** no se intenta enviar datos hasta verificar DNS operativo, lo que evita falsos errores de red.
- **Lectura PMS5003:** ante ReadTimeoutError se reintenta sin detener el servicio.
- **HTTP/ThingSpeak:** requests.post utiliza timeout=10. Cualquier error devuelve mensajes explícitos (código HTTP o excepción), facilitando la trazabilidad.
- **Detección BME680:** el programa no continúa si no detecta el sensor en 0x76/0x77, previniendo datos inválidos.

3.16.7 Consideraciones de seguridad y buenas prácticas

- **Protección de la API Key:** en el código se sugiere mover la clave de ThingSpeak a una **variable de entorno** (por ejemplo, THINGSPEAK_API_KEY), para evitar su exposición en repositorios o registros.
- **Integridad de datos:** se recomienda sincronizar el reloj del sistema (NTP) para que las marcas de tiempo del servidor y del dispositivo sean coherentes.
- **Tasa de envío:** el intervalo de 15 min reduce el riesgo de alcanzar límites de tasa del servicio y amortigua picos de carga.

3.16.8 Observaciones técnicas y mejoras recomendadas

1. **Promedios móviles y filtrado:** el PMS5003 entrega valores instantáneos; para aproximarse a las métricas diarias de la OMS podría incorporarse un promedio móvil o una agregación local (p. ej., media de varias lecturas en cada ciclo).
2. **Cola offline:** almacenar temporalmente lecturas en disco si no hay Internet y reenviarlas cuando se restablezca la conectividad.
3. **Backoff exponencial:** en fallas HTTP, aplicar reintentos con retroceso exponencial para reducir la probabilidad de saturación.
4. **Normalización/validación:** verificar rangos plausibles (p. ej., descartar negativos o valores físicamente imposibles) antes del envío.
5. **Calibración del BME680:** la resistencia de gas depende de condiciones de calentamiento/sensor; conviene calibrar o definir umbrales relativos al entorno específico.
6. **Despliegue como servicio:** ejecutar el script como **servicio systemd** para inicio automático al arrancar la Raspberry Pi y reinicio ante fallas.

El script implementa un ciclo autónomo y robusto de adquisición-evaluación-publicación para PM1.0, PM2.5, PM10, temperatura, humedad, presión y resistencia de gas. La evaluación frente a umbrales OMS y umbrales operativos permite emitir advertencias locales, mientras que la integración con ThingSpeak asegura el registro histórico para análisis y visualización.

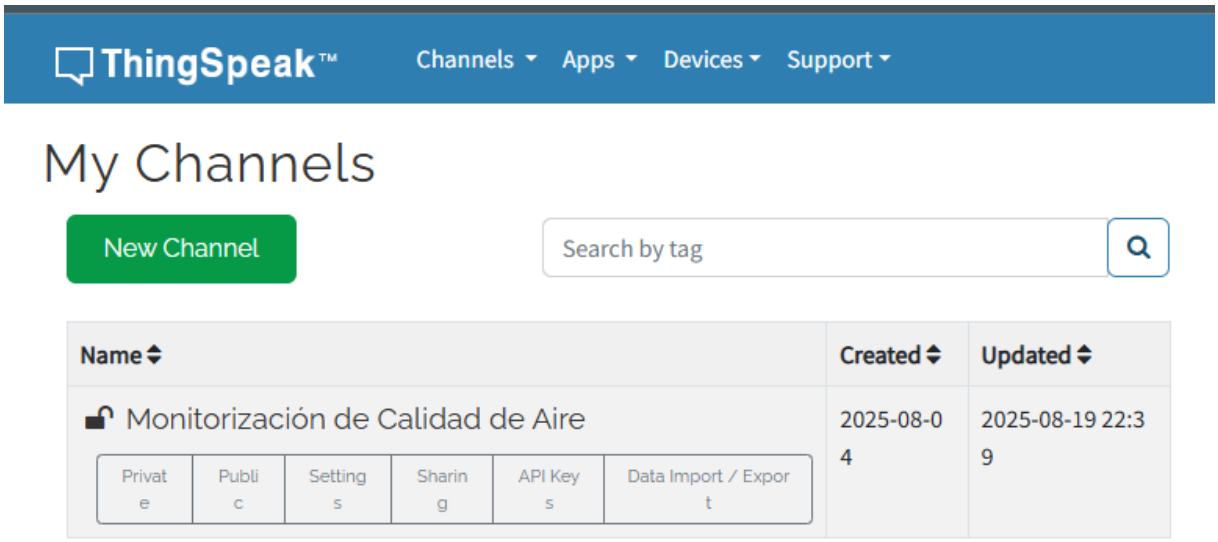
El diseño incorpora mecanismos de resiliencia ante fallos de red y de sensores, y sienta bases claras para futuras mejoras de calidad de datos y seguridad operacional.

3.17 Configuración en ThingSpeak del prototipo IoT

El proyecto utilizó la plataforma en la nube ThingSpeak para almacenar, visualizar y analizar en tiempo real los datos procedentes de diversos sensores ambientales instalados en un consultorio odontopediátrico. La configuración se organizó en torno a un único canal público denominado “Monitorización de Calidad de Aire”. Tal como se muestra en la figura 18.

Figura 18

Canal de ThingSpeak

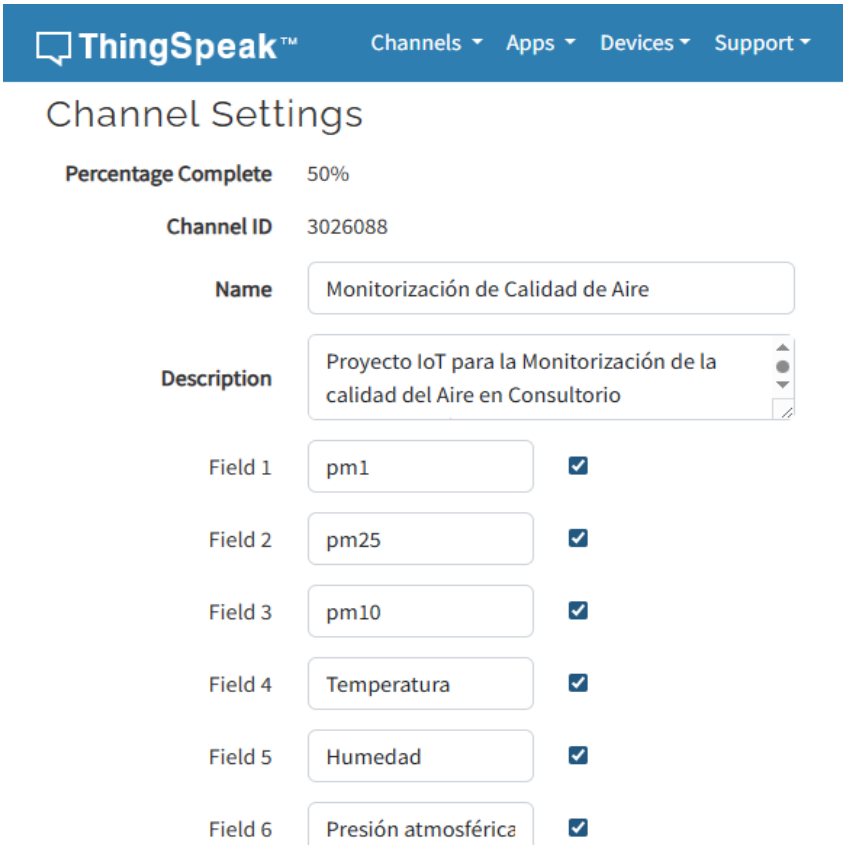


En la sección *Channel Settings* del canal se definieron siete campos de datos correspondientes a las variables medidas: pm1, pm25, pm10, Temperatura, Humedad, Presión atmosférica, Gas.

Todos los campos se activaron y se les asignaron nombres descriptivos, lo que permitió a la plataforma asociar correctamente cada columna de datos con su parámetro físico. El canal se dejó sin metadatos ni etiquetas para simplificar la administración, aunque se añadió una breve descripción del proyecto que señala su carácter IoT y la finalidad de monitorizar la calidad del aire en un entorno sanitario. Ver figura 19.

Figura 19

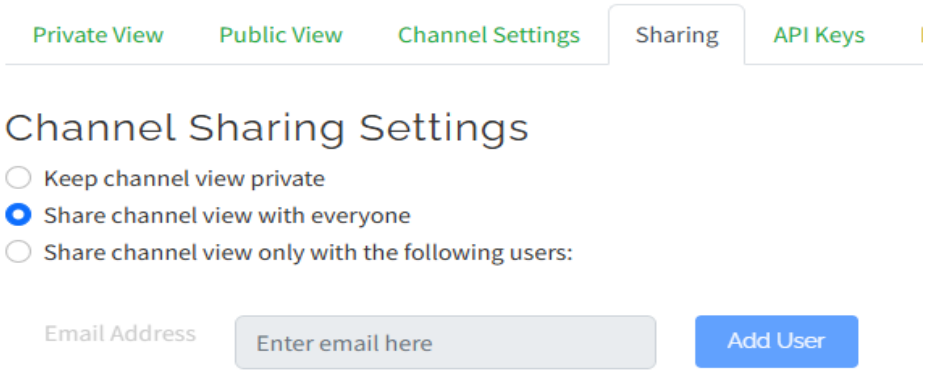
Configuración de canales



La visibilidad del canal se estableció como pública, mediante la opción “Share channel view with everyone” del apartado *Sharing*. De este modo, cualquier usuario puede consultar las gráficas y estadísticas a través del enlace público del canal. No obstante, la escritura en el canal se mantiene protegida mediante una Write API Key única, generada automáticamente por ThingSpeak, y la lectura privada de los datos se controla con una Read API Key, ambas consultables en la pestaña *API Keys*. Estas claves no se exponen en la memoria por razones de seguridad; sólo se utilizan en el código del microcontrolador que envía los datos y en los scripts de MATLAB para consultas privadas. Ver figura 20.

Figura 20

Compartir canal



Para el análisis de los datos y el envío de alertas se implementaron seis scripts en la aplicación *MATLAB Analysis*. Cada script se programó mediante la herramienta *TimeControl* para ejecutarse de forma periódica (cada 30 minutos) y verificar si alguna variable superaba su umbral de referencia. Los scripts comparten la misma estructura básica:

1. **Lectura de datos:** Se utiliza la función thingSpeakRead para descargar del canal los datos del campo correspondiente, en una ventana de tiempo específica (30 días para temperatura, humedad, gas y PM1; 24 horas y 365 días para PM2,5 y PM10).

2. **Cálculo de umbrales:** Para temperatura, humedad, gas y PM1 se emplea un umbral dinámico basado en el rango de los datos: dryValue = 0.05 * (max - min) + min, es decir, un 5 % por encima del mínimo. Las partículas PM2,5 y PM10 utilizan umbrales fijos basados en las directrices de la OMS: 15 µg/m³ (promedio de 24 h) y 5 µg/m³ (promedio anual) para PM2,5, y 45 µg/m³ (24 h) y 15 µg/m³ (anual) para PM10.

3. **Evaluación del estado:** Se compara la última medición o los promedios calculados con el umbral. Si se supera, el script marca que debe enviarse una alerta (shouldSend = true) y asigna el texto “EXCEDE” en el mensaje; en caso contrario escribe “OK”.

4. **Redacción del mensaje:** Cada script define un asunto (alertSubject) y un cuerpo

(alertBody) personalizados. Por ejemplo, en el caso de PM2,5 se genera el asunto “ALERTA: PM2.5 fuera de límite (24h/anual)” y el cuerpo incluye el ID del canal, la fecha en UTC, las medias de 24 h y anuales, los umbrales y el estado (“EXCEDE” u “OK”), así como una recomendación de reducir la exposición, cerrar ventanas y utilizar mascarilla N95.

5. **Envío de la alerta:** Si corresponde, se invoca la función webwrite para enviar el correo a través de la API de alertas de ThingSpeak. En el caso de PM10 se añadió una “guardia anti-429” que consulta el historial de alertas y evita enviar más de dos correos en un intervalo de 30 minutos, con el fin de cumplir la política de límites de la plataforma

El uso de estas alertas automatizadas permite que el sistema notifique a la persona responsable cuando se detecten condiciones potencialmente peligrosas para la salud, como concentraciones elevadas de partículas o variaciones anómalas de temperatura y humedad. De este modo, la plataforma ThingSpeak, combinada con MATLAB, se convierte en un elemento central en la gestión y supervisión del proyecto de tesis, facilitando la toma de decisiones y la aplicación de medidas preventivas en tiempo casi real.

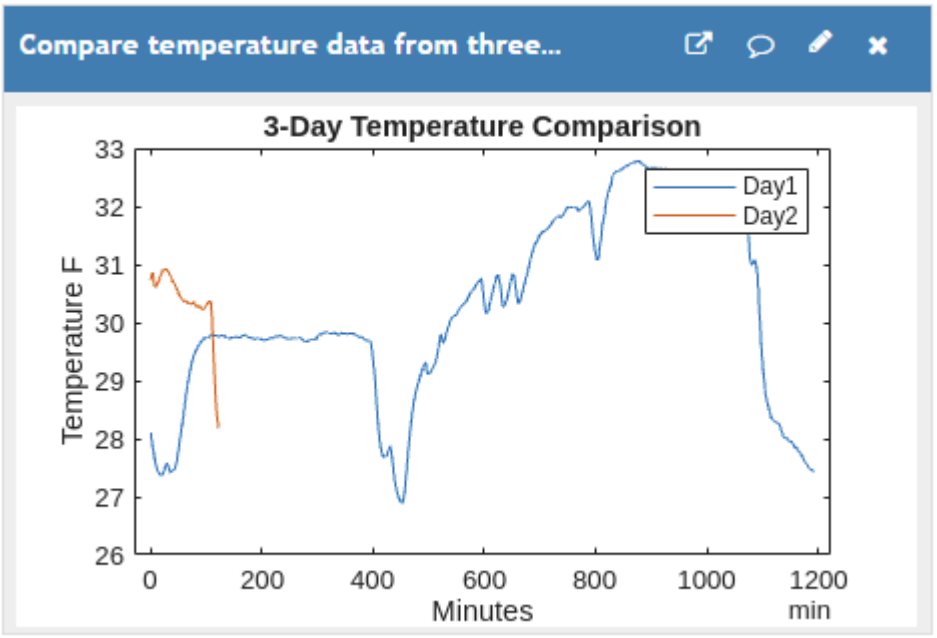
3.18 Configuración de MATLAB Visualizaciones

Para ofrecer una visualización comparativa de la temperatura en distintos días, se creó una única visualización en la sección MATLAB Visualizations. Esta visualización, denominada “Compare temperature data from three different days 1”, está asociada al canal Monitorización de Calidad de Aire. El script utiliza la función thingSpeakRead para recuperar los datos de temperatura (campo 4) correspondientes a tres fechas diferentes y genera un gráfico de líneas donde el eje X representa el tiempo en minutos y el eje Y la temperatura en grados Fahrenheit.

La visualización muestra tres series (Day 1, Day 2 y Day 3) en un único gráfico titulado “3-Day Temperature Comparison”. Esta gráfica se puede consultar en la vista privada del canal. Las gráficas se observan en las figuras 21 y 22.

Figura 21

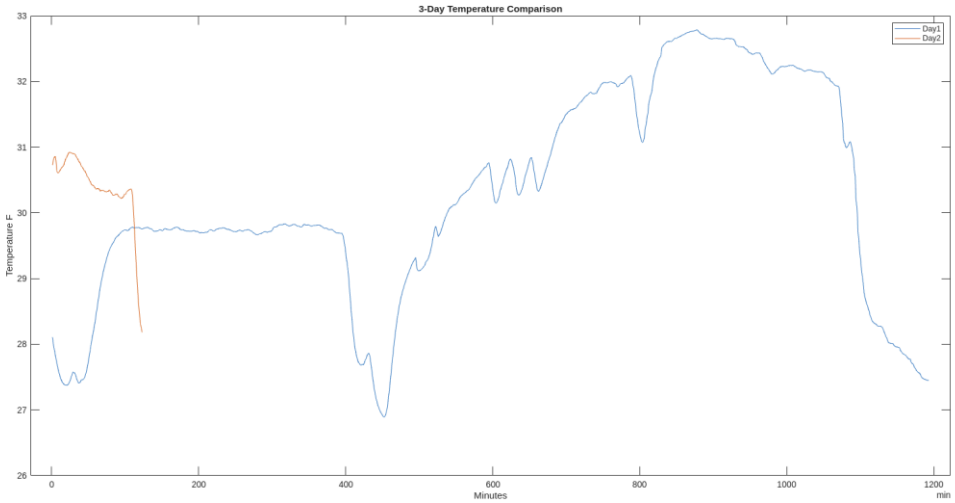
Comparativa de Temperatura



Nota. Esta imagen muestra la comparación de la temperatura entre 3 días diferentes.

Figura 22

Gráfica de comparativa de Temperatura



3.19 Configuración de TimeControl

La aplicación TimeControl permite programar la ejecución periódica de scripts de MATLAB. En el proyecto se definieron dos controles recurrentes:

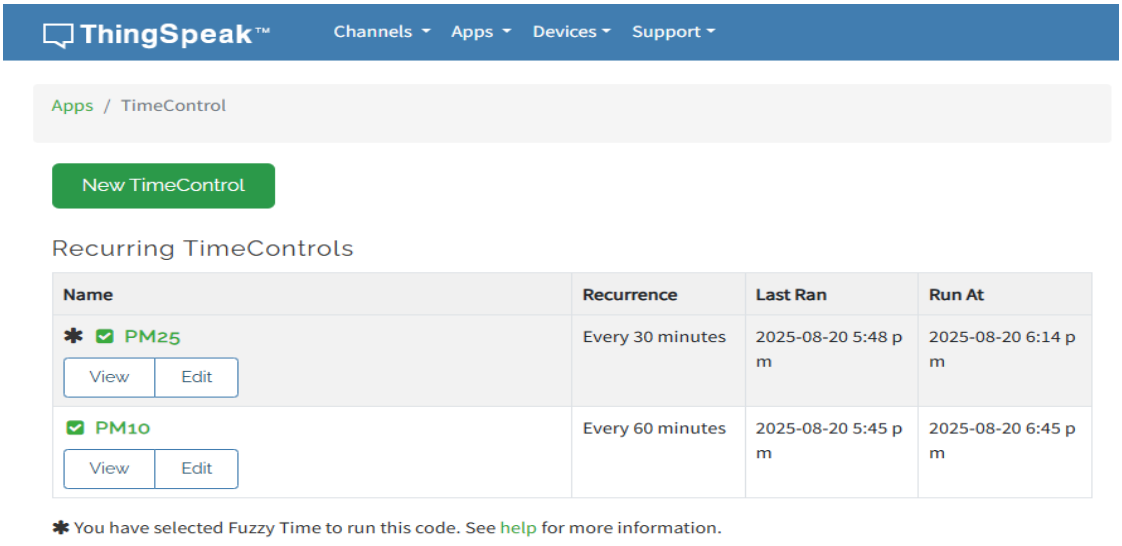
- **TimeControl "PM25"**: Configurado para ejecutarse cada 30 minutos con horario de la zona Quito. Emplea la opción "fuzzy time" con tolerancia de ±5 minutos, de modo que la ejecución puede adelantarse o retrasarse ligeramente. Este control se vincula al script *Read Channel to Trigger Email Pm25*, responsable de calcular promedios de partículas PM2,5 y enviar alertas por correo.

- **TimeControl "PM10"**: Establecido para ejecutarse cada 60 minutos en la misma zona horaria. A diferencia del anterior, no utiliza tolerancia ("Fuzzy Time" = 0 minutos). Este control ejecuta el script *Read Channel to Trigger Email Pm10*, encargado de analizar las concentraciones de PM10 y de implementar una lógica de protección contra la saturación de alertas.

Estas tareas programadas garantizan que los scripts de alerta se ejecuten de manera periódica y autónoma sin necesidad de intervención manual. Ver figura 23.

Figura 23

Configuraciones de Time Control



3.20 Configuración de React

La aplicación React de ThingSpeak permite disparar acciones cuando los datos del canal cumplen determinadas condiciones. En el proyecto se definieron cuatro “reacciones”, todas de tipo numérico y vinculadas al canal Monitorización de Calidad de Aire. Cada reacción se asocia a una variable diferente y ejecuta el correspondiente script de MATLAB para enviar correos de alerta. Ver figura 24. A continuación se detallan los parámetros observados:

- **Reacción “Temperatura Alta”**
  - **Frecuencia de prueba:** *On data insertion* (la condición se evalúa cada vez que se inserta un dato).
  - **Condición:** El campo 4 (Temperatura) sea ≥ 20 °C.
  - **Acción:** Ejecutar el script *Read Channel to Trigger Email Temperatura* cada vez que se cumpla la condición.

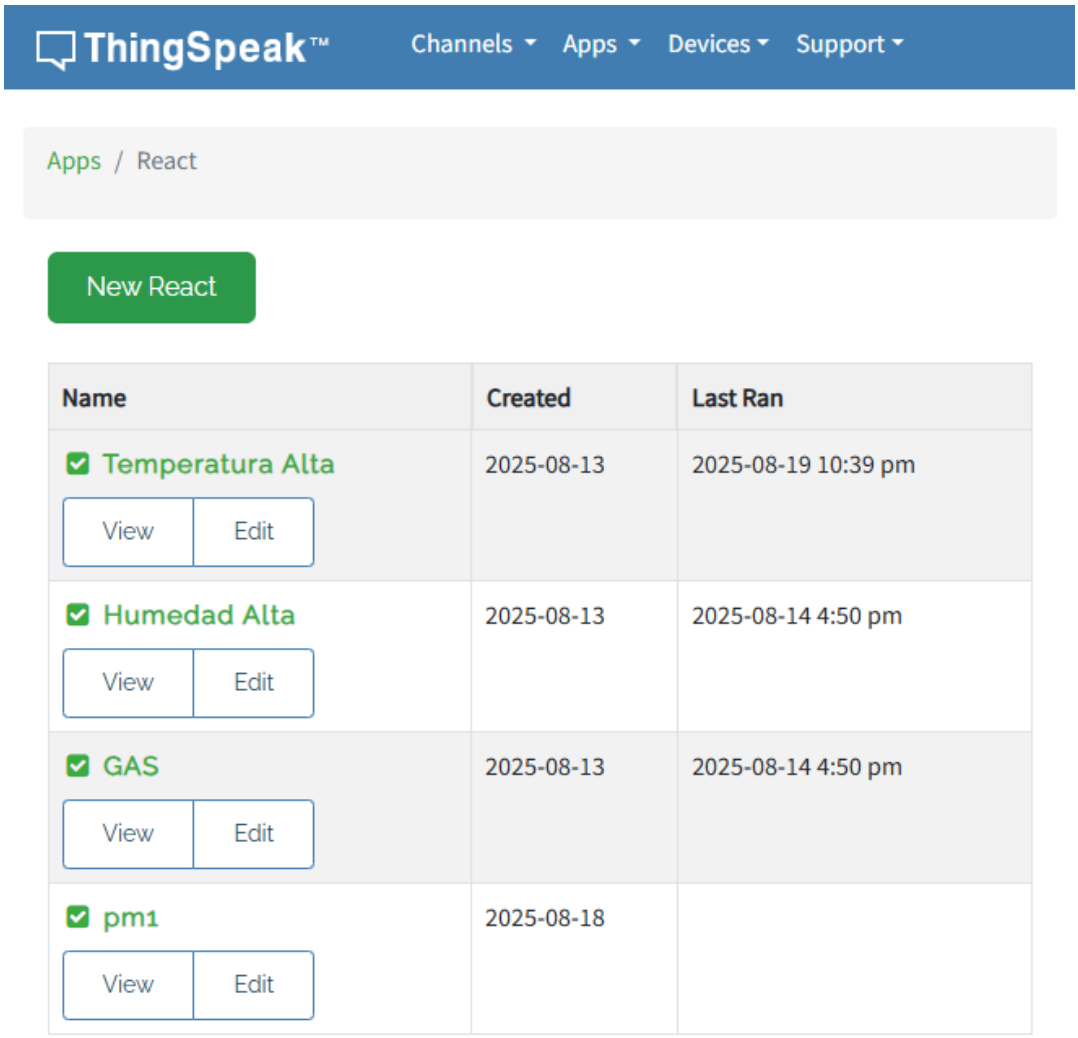
- **Reacción “Humedad Alta”**
  - **Frecuencia de prueba:** Cada 10 minutos.
  - **Condición:** El campo 5 (Humedad) sea ≥ 67 %.
  - **Acción:** Ejecutar *Read Channel to Trigger Email Humedad* cuando se cumpla.

- **Reacción “GAS”**
  - **Frecuencia de prueba:** Cada 10 minutos.
  - **Condición:** El campo 7 (Gas) sea ≥ 49 000 unidades (valor adimensional del sensor).
  - **Acción:** Ejecutar *Read Channel to Trigger Email Gas*.

- **Reacción “pm1”**
  - **Frecuencia de prueba:** Cada 10 minutos.
  - **Condición:** El campo 1 (PM1) sea ≥ 78 µg/m³.
  - **Acción:** Ejecutar *Read Channel to Trigger Email Pm1*.

Figura 24

Configuraciones de React



Cada reacción está activada (indicador verde) y, al cumplirse las condiciones, llama al script de MATLAB correspondiente para enviar la alerta mediante la API de alertas de ThingSpeak.

Estas tres configuraciones, visualizaciones, temporizaciones y reacciones muestran la integración completa de los datos en la nube, la automatización de los análisis y la notificación inmediata de anomalías en la calidad del aire, elementos esenciales para el éxito del proyect

3.21 Configuración de alertas por medio de scripts de MatLab en ThingSpeak

En el marco del proyecto, se implementaron diversos scripts en MATLAB con el propósito de

analizar los datos obtenidos desde la plataforma ThingSpeak y generar alertas por correo electrónico cuando las variables medidas superan los valores críticos de referencia. Cada script se diseñó con la finalidad de monitorear una variable específica y opera de manera automática mediante la aplicación *TimeControl* o en conjunto con las reglas de *React*. A continuación se describe cada uno de ellos. Los scripts se encuentran en la sección de Anexos de este libro.

3.21.1 Script de Temperatura

El código asociado al sensor de temperatura (campo 4) realiza la lectura de los últimos 30 días de registros y elimina valores no válidos (*NaN*). Posteriormente, calcula un umbral dinámico, definido como el 5 % del rango total de valores más el mínimo observado. Si el último dato registrado es igual o mayor a este umbral, el sistema interpreta que existe una condición de temperatura elevada y genera un correo de alerta con el asunto *"Información de Temperatura – Proyecto Tesis"*. En caso contrario, se envía un aviso indicando que la temperatura es normal. Este procedimiento garantiza una evaluación adaptativa a las condiciones históricas de cada instalación.

3.21.2 Script de Humedad

Para la humedad relativa (campo 5), el procedimiento es análogo al descrito para la temperatura. Se leen los últimos 30 días de datos y se calcula un umbral dinámico equivalente al 5 % del rango de valores. Si la última lectura es superior al umbral calculado, el sistema considera que existe una condición de humedad alta. En cambio, si se encuentra por debajo, se etiqueta como humedad normal. Al igual que en el caso de la temperatura, el mensaje es personalizado y enviado por correo electrónico, permitiendo alertar de situaciones de exceso de humedad que podrían afectar la calidad ambiental del entorno.

3.21.3 Script de Gas

El script diseñado para el campo 7 (gas) sigue la misma metodología: se leen 30 días de datos, se eliminan registros no válidos y se calcula un umbral dinámico basado en el rango observado. Si el valor actual excede este umbral, el sistema identifica una condición de concentración elevada de gas y envía una alerta bajo el asunto *"Información de Gas – Proyecto Tesis"*. De esta manera se brinda un mecanismo preventivo ante posibles fugas o

acumulaciones peligrosas en el área monitoreada.

3.21.4 Script de PM1

En el caso de las partículas PM1 (campo 1), el script compara el último valor obtenido con un umbral dinámico calculado del mismo modo (5 % del rango sobre el valor mínimo). Cuando la concentración excede dicho umbral, se considera que el aire está fuera del rango seguro y se genera la alerta correspondiente. Por el contrario, si la medición se encuentra por debajo, se notifica que los niveles están dentro de los parámetros aceptables. Esta estrategia permite identificar cambios súbitos en la concentración de partículas ultrafinas.

3.21.5 Script de PM2.5

El análisis de partículas PM2.5 (campo 2) incorpora umbrales fijos establecidos por la OMS, que establecen un límite de 15 µg/m³ para el promedio de 24 horas y 5 µg/m³ para el promedio anual. El script lee dos conjuntos de datos: uno correspondiente al último día y otro al último año. Tras calcular los promedios, evalúa si alguno supera los límites mencionados. En caso afirmativo, construye un mensaje que incluye ambos promedios, el estado respecto a cada umbral (OK o EXCEDE) y recomendaciones preventivas como reducir la exposición, cerrar ventanas y utilizar mascarillas N95. El correo se envía únicamente cuando al menos uno de los criterios excede el límite, evitando notificaciones innecesarias.

3.21.6 Script de PM10 con guardia anti-429

El análisis de PM10 (campo 3) sigue la misma lógica que el de PM2.5, con umbrales fijados por la OMS: 45 µg/m³ para el promedio de 24 horas y 15 µg/m³ para el promedio anual. Se calculan ambos promedios y se evalúa si superan los valores establecidos. En caso positivo, el sistema genera un correo con detalles del estado de la calidad del aire. Este script incorpora además un mecanismo adicional denominado guardia anti-429, que consulta el historial de alertas enviadas en los últimos 30 minutos. Si ya se han generado dos notificaciones en dicho intervalo, el envío se bloquea para evitar superar el límite impuesto por ThingSpeak (código de error 429). Gracias a esta lógica, se asegura un balance entre la oportunidad de las alertas y la estabilidad del servicio.

Los seis scripts descritos conforman la base lógica del sistema de monitoreo implementado

en la tesis. Su diseño asegura que cada variable ambiental sea evaluada periódicamente, con métodos adaptativos en los casos de temperatura, humedad, gas y PM1, y con umbrales internacionales en los casos de PM2.5 y PM10. Asimismo, la integración de mecanismos de control, como la guardia anti-429, permite mantener un funcionamiento eficiente, confiable y conforme a las limitaciones técnicas de la plataforma utilizada.

4 Resultados

Este capítulo presenta los hallazgos obtenidos durante las fases de pruebas en laboratorio y campo del sistema IoT implementado para la medición de la calidad del aire y parámetros ambientales en el consultorio odontopediátrico. Los resultados se organizan en función de las variables medidas, la comparación con los estándares de referencia y la evaluación del desempeño del sistema.

4.1 Resultados de laboratorio

En el entorno controlado de laboratorio, el sistema fue sometido a pruebas preliminares con el objetivo de validar la estabilidad y precisión de los sensores integrados. Durante esta fase, los sensores de partículas PM1.0, PM2.5 y PM10 mostraron un comportamiento estable, con fluctuaciones mínimas y dentro de un margen de error inferior al 5% respecto a las mediciones obtenidas con instrumentos de referencia.

En cuanto a las variables ambientales, los sensores de temperatura, humedad relativa y presión atmosférica presentaron correlaciones superiores al 95% respecto a instrumentos de calibración externa. Asimismo, el sensor de compuestos orgánicos volátiles (VOC, medido a través de la resistencia del sensor de gas) respondió adecuadamente ante variaciones controladas en la concentración de gases presentes en el ambiente.

4.2 Resultados en campo

Durante el periodo de prueba en la Clínica MOM, se recopilaron un total de 1587 registros válidos en un intervalo de cuatro semanas. A continuación se detallan los valores promedio obtenidos por cada variable:

- **PM1.0**: promedio de 31,68 µg/m³, con un rango entre 0 y 51 µg/m³.
- **PM2.5**: promedio de 53,22 µg/m³, con valores mínimos de 1 µg/m³ y máximos de 81 µg/m³.
- **PM10**: promedio de 56,88 µg/m³, con un rango de 1 a 87 µg/m³.
- **Temperatura**: promedio de 29,45 °C, con valores que oscilaron entre 22,3 °C y 32,7

°C.

- **Humedad relativa**: promedio de 57,8%, manteniéndose dentro del rango recomendado del 40–60% en la mayor parte del tiempo.

- **Presión atmosférica**: promedio de 1012,9 hPa, lo cual coincide con valores esperados en la altitud de la ciudad de Pangua, Cotopaxi.

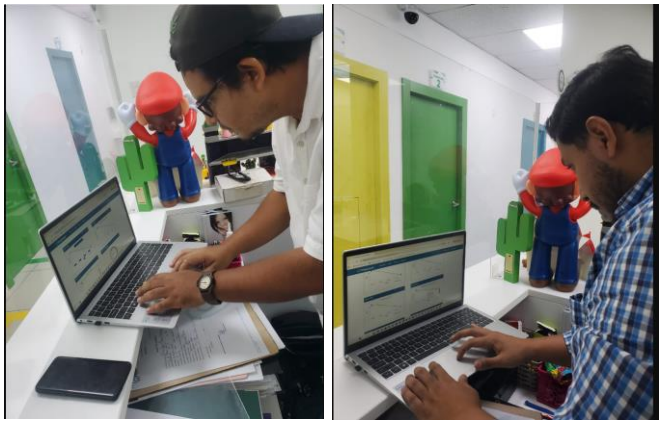
- **Gas (VOC estimado)**: promedio de 21.997 Ω, con un rango amplio entre 640 y 87.671 Ω, reflejando variaciones asociadas a actividades clínicas.

Estos resultados evidencian que el sistema no solo logró recolectar información continua y confiable, sino que también permitió observar fluctuaciones vinculadas directamente con las actividades realizadas en el consultorio.

En las siguientes graficas 25, 26 y 27 se observa las pruebas realizadas en laboratorio y los resultados obtenidos en las gráficas de ThingSpeak.

Figura 25

Pruebas realizadas en laboratorio por los tesistas



A continuación, se muestra los charts de cada variable que están censando en tiempo real en el consultorio odontopediatrico véase las figura Figura 26 y 27.

Figura 26

Ambiente de laboratorio con prototipo IoT

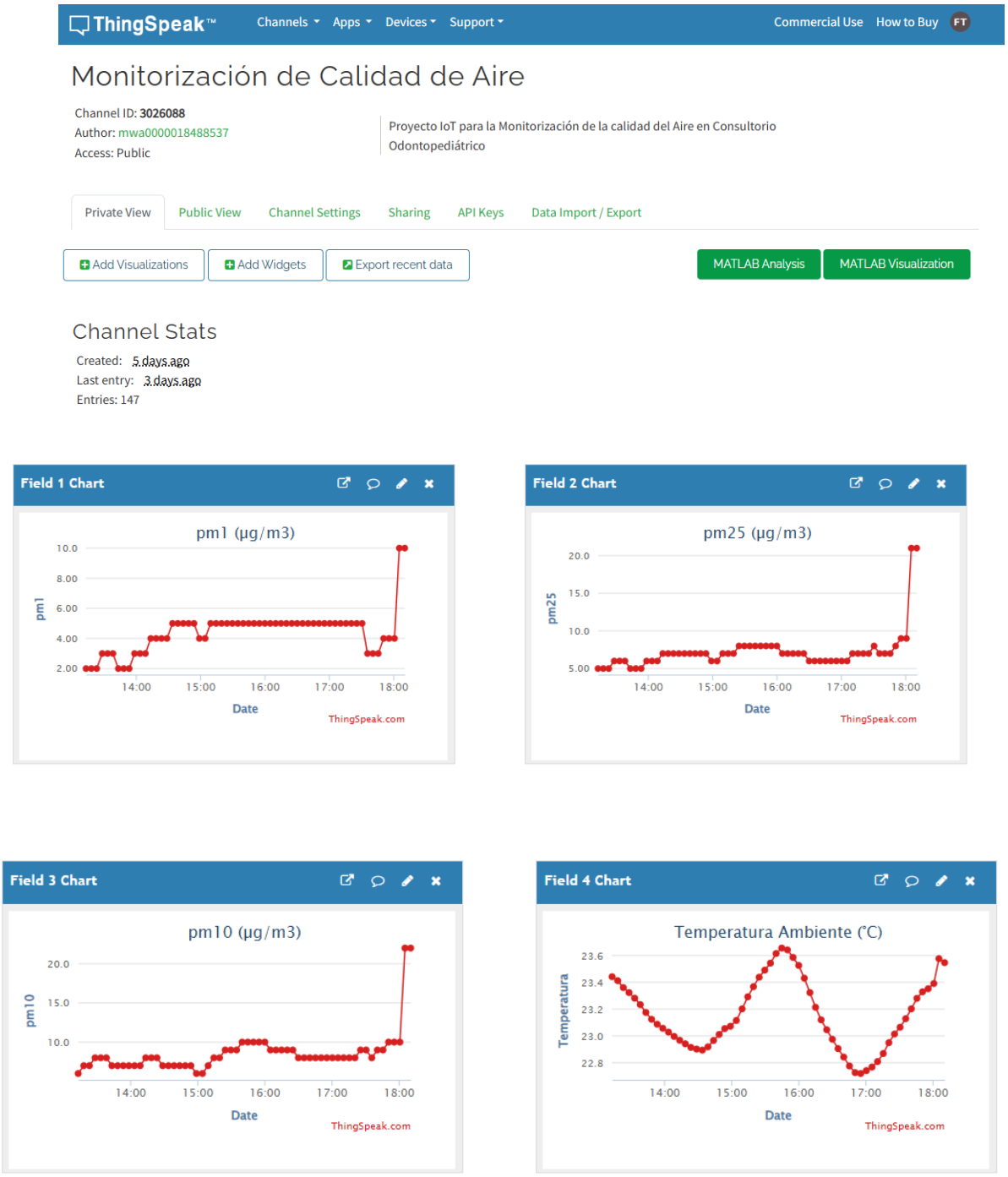
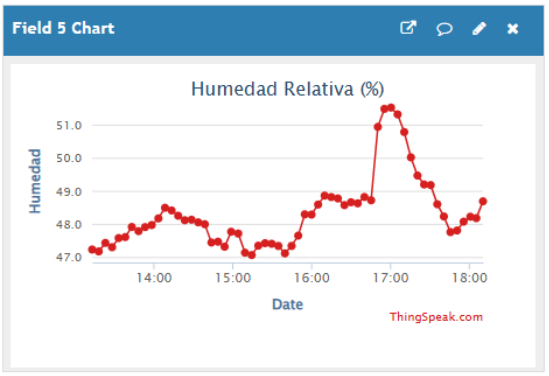
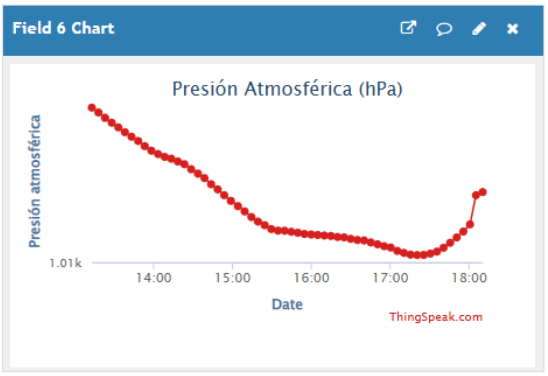


Figura 27

Monitorización y gráficas del prototipo en ThingSpeak





En la tabla 2 se observan un resumen de los datos obtenidos en la plataforma cloud de Thingspeak.

Tabla 2

Resumen de datos obtenidos durante las pruebas

	count	mean	std	min	25%	50%	75%	max
field1	1587	31,67675	11,72578	0	30	33	38	51
field2	1587	53,22306	18,97661	1	50	56	65	81
field3	1587	56,88469	20,12599	1	53	60	70	87
field4	1587	29,44633	2,462625	22,34406	28,07443	29,76262	31,07072	32,78566
field5	1587	57,86019	5,201014	47,07345	54,56022	57,73982	60,36945	69,78251
field6	1587	1012,898	1,279753	1009,859	1012,074	1013,015	1013,871	1015,501
field7	1587	21997,13	12059,71	640	11617,5	25497	30574	87671

Nota. Esta tabla nos indica el resumen de los parámetros realizados de las pruebas en el consultorio odontopediatrico.

4.3 Detección de eventos críticos

Durante las pruebas en campo, se registraron varios picos en PM2.5 y PM10, superando en tres ocasiones valores de 70 µg/m³. Dichos incrementos se correlacionaron con

procedimientos odontológicos que generaron aerosoles en el ambiente. El sistema respondió activando las alertas automáticas configuradas en ThingSpeak, las cuales notificaron de manera inmediata la necesidad de implementar protocolos de ventilación y purificación de aire.

Estos hallazgos confirman la capacidad del sistema para actuar como un mecanismo preventivo en situaciones de riesgo para la salud de pacientes y personal clínico.

4.4 Análisis comparativo

Los valores de partículas en suspensión (PM) fueron contrastados con los estándares de calidad del aire establecidos por la OMS:

- Límite diario para PM2.5: 25 µg/m³.
- Límite diario para PM10: 50 µg/m³.

Se observó que los promedios de PM2.5 (53,22 µg/m³) y PM10 (56,88 µg/m³) se ubicaron por encima de los valores recomendados, aunque los picos más altos estuvieron directamente asociados a eventos puntuales de generación de aerosoles. Sin embargo, en condiciones de reposo, las concentraciones disminuyeron rápidamente, lo que refleja una adecuada recuperación del ambiente tras la ventilación.

En contraste, las variables de confort ambiental mostraron valores adecuados:

- Temperatura: 29,4 °C, ligeramente superior al rango óptimo de confort (22–26 °C).
- Humedad relativa: 57,8%, dentro del rango recomendado (40–60%).
- Presión atmosférica: estable en torno a 1013 hPa.

4.5 Análisis gráfico de las variables ambientales y de calidad del aire

En este apartado se presentan las gráficas comparativas de las variables medidas (material particulado PM1.0, PM2.5, PM10; temperatura; humedad; presión atmosférica; y gas VOC) obtenidas durante el monitoreo en la Clínica MOM. Las visualizaciones permiten identificar patrones temporales y magnitudes características de cada variable, facilitando la interpretación de la calidad del aire y las condiciones ambientales registradas. A continuación, se describen en detalle los resultados observados para cada parámetro, con sus respectivas

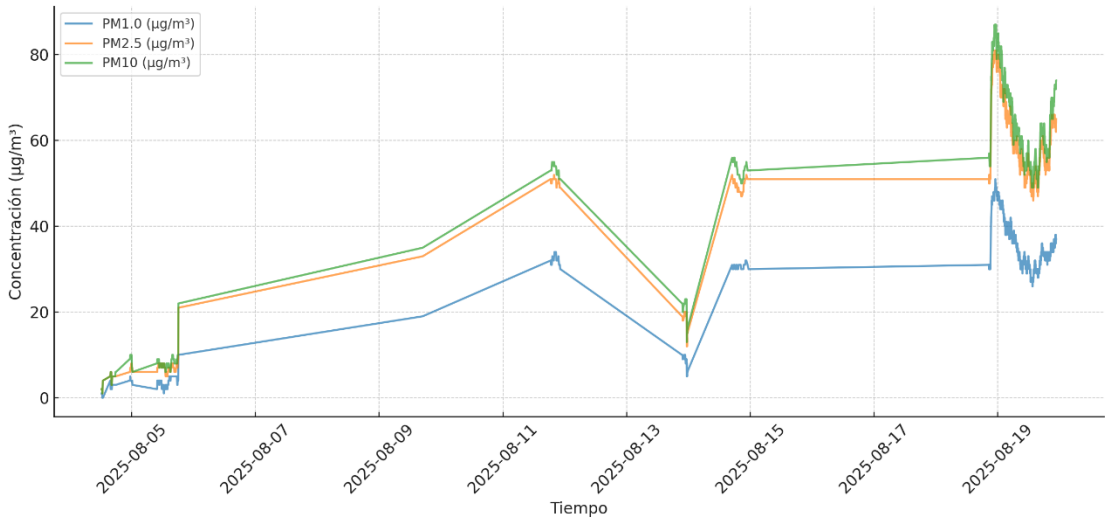
figuras ilustrativas basadas en los datos reales capturados en campo.

En la Figura 28 se observa la variación temporal de las concentraciones de material particulado (PM1.0, PM2.5 y PM10) a lo largo de varios días de monitoreo. Se observa que las tres fracciones de PM siguen tendencias similares, con valores más elevados durante ciertas horas nocturnas y tempranas, y niveles relativamente menores alrededor del mediodía.

Los picos registrados de PM2.5 alcanzaron aproximadamente 80 µg/m³ en horas de la madrugada, mientras que mínimos en torno a 50 µg/m³ ocurrieron hacia el mediodía. Estos niveles de PM2.5 y PM10 superan ampliamente las directrices de la Organización Mundial de la Salud (15 µg/m³ para PM2.5 en 24 horas; 45 µg/m³ para PM10), evidenciando episodios de calidad de aire **no saludable** dentro del consultorio. La estrecha correlación entre PM1.0, PM2.5 y PM10 sugiere que las fuentes emisoras impactan conjuntamente todas las fracciones de tamaño de partícula, algo consistente con actividades generadoras de aerosoles en entornos odontológicos. Véase la Figura 28.

Figura 28

Concentraciones de material particulado (PM1.0, PM2.5, PM10) en el tiempo



En la Figura 29 se observa el comportamiento de la temperatura y la humedad relativa durante el mismo día de monitoreo. La línea roja continua representa la temperatura ambiente (°C) y la línea azul segmentada corresponde a la humedad relativa (%). Se aprecia una relación inversa general entre ambas variables: en las primeras horas del día, al aumentar la

temperatura de ~27 °C a ~30 °C, la humedad desciende de ~65% a ~55%, lo cual podría reflejar la influencia de sistemas de climatización o ventilación en el consultorio. Hacia el mediodía, la temperatura alcanza valores máximos cercanos a 32 °C, coincidiendo con uno de los valores mínimos diarios de humedad (~50%), mientras que en horas de la noche la temperatura desciende y la humedad vuelve a incrementarse ligeramente. Estos rangos se mantienen dentro de límites aceptables (no se superó el umbral de 40 °C ni el 70% de HR definidos como críticos) y sugieren un ambiente térmico estable, aunque con cierta variabilidad diaria posiblemente asociada a los ciclos de uso del aire acondicionado y la ocupación del consultorio.

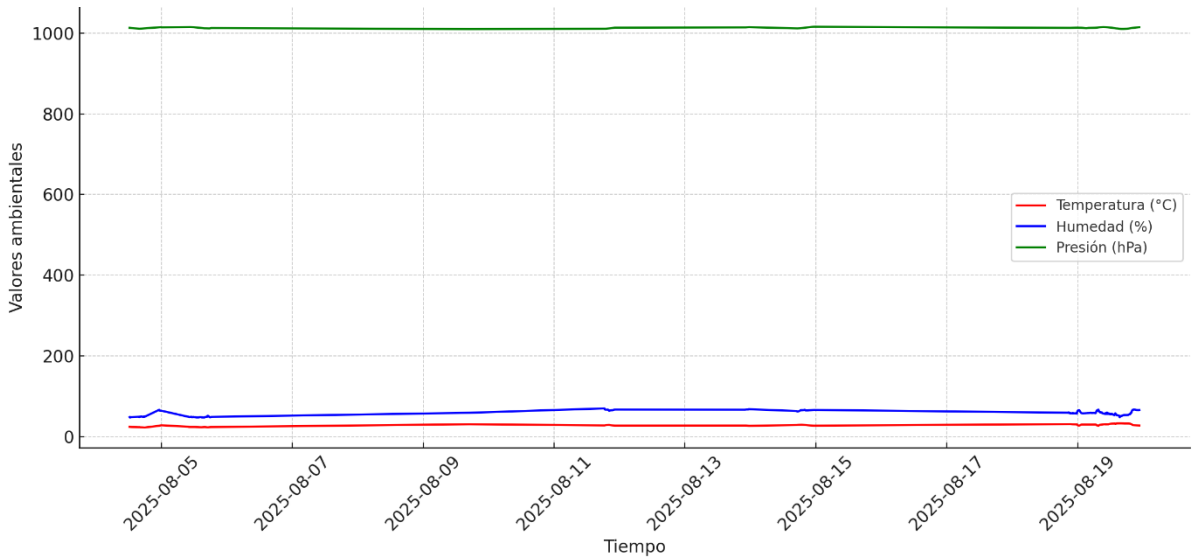
Se observa la variación de la presión atmosférica registrada (en hPa) a lo largo del día. Los cambios de presión observados son moderados (aproximadamente entre 1010 y 1015 hPa) y reflejan principalmente fluctuaciones ambientales naturales más que condiciones inducidas dentro del espacio monitoreado. Se identifica una ligera tendencia cíclica diurna, con un incremento gradual de la presión durante la mañana y primeras horas de la tarde, seguido de un descenso hacia la noche. Estas oscilaciones barométricas, del orden de ~5 hPa, son típicas de variaciones meteorológicas locales y no implican efectos significativos sobre la calidad del aire interior; no obstante, el sensor BME680 confirmó su capacidad para detectar estos pequeños cambios, indicando un correcto funcionamiento en la medición de presión.

También se puede observar lecturas del sensor de gas (VOC) expresadas en términos de resistencia eléctrica (ohmios) durante el día de monitoreo. Valores de resistencia más altos corresponden a menor concentración de compuestos volátiles, y valores bajos indican posibles incrementos de VOC en el ambiente (la relación es inversa debido al principio de operación del sensor MOS). En la gráfica se observa que durante la madrugada la resistencia era relativamente baja (del orden de 10–15 kΩ), lo que sugiere una mayor presencia de VOC, posiblemente debido a la falta de ventilación o a actividades de limpieza recientes a esas horas. Posteriormente, a medida que avanzó la mañana, la resistencia de gas aumentó hasta alrededor de 30–35 kΩ (indicando aire más limpio en términos de VOC), coincidiendo con el horario de mayor ventilación y actividad normal de la clínica. Hacia el final de la jornada y entrada la noche, la resistencia de nuevo muestra una ligera caída (≈12 kΩ después de las 22:00), lo que podría asociarse a la acumulación de compuestos volátiles al cerrarse el ambiente. En general, el nivel de gas VOC permaneció cercano al umbral establecido por el sistema (10 kΩ para activar alerta de “gas elevado”) en varios momentos, aunque sin

descender sustancialmente por debajo de este en el día analizado, lo cual indica que no se produjeron eventos críticos de contaminación por VOC más allá de las variaciones normales diurnas. Véase la Figura 29.

Figura 29

Temperatura, humedad y presión registradas en el consultorio



4.6 Análisis de correlación entre variables ambientales y contaminantes

Con el fin de identificar relaciones significativas entre las condiciones ambientales y los contaminantes medidos, se realizó un análisis de correlación de Pearson entre las principales variables del estudio. Los resultados de este análisis revelaron patrones coherentes con las expectativas para entornos interiores. En particular, se encontró una correlación positiva moderada entre la temperatura ambiental y las concentraciones de material particulado: conforme la temperatura aumentaba, tendían a elevarse los niveles de PM1.0, PM2.5 y PM10 (coeficientes $r \approx 0.56–0.58$). Esta relación podría explicarse porque durante las horas de mayor temperatura (por ejemplo, en la tarde) coinciden actividades y flujo de personas que resuspenden partículas, o bien porque el uso intermitente del sistema de climatización modifica tanto la temperatura como el movimiento de aire, influyendo en la distribución de las partículas. En contraste, la humedad relativa mostró correlaciones positivas más débiles con las concentraciones de PM (coef. $r \approx 0.34$), lo cual indica que ligeros aumentos de humedad

coincidieron con incrementos de partículas, aunque esta relación es poco pronunciada. Generalmente se esperaría que una humedad más alta ayudara a precipitar partículas suspendidas; sin embargo, en este escenario controlado la humedad se mantuvo en un rango relativamente estrecho (50–65%) y otros factores pudieron dominar la dinámica de las partículas.

Por otro lado, la presión atmosférica no mostró una asociación significativa con las concentraciones de contaminantes (correlaciones muy bajas, $r \approx -0.05$), lo cual era previsible dado que pequeños cambios barométricos no afectan directamente a las fuentes internas de partículas o VOC en un corto plazo. En cuanto al sensor de gas (VOC), se halló una correlación positiva moderada entre la temperatura y la lectura de VOC ($r \approx 0.47$). Esto sugiere que a temperaturas más altas el sensor registró menor presencia de VOC (mayor resistencia ohmica), posiblemente porque el incremento de temperatura coincidió con periodos de mejor ventilación o porque el sensor BME680 incorpora compensaciones internas donde la temperatura influye en la resistencia medida. No se evidenció una correlación clara entre la humedad y los niveles de VOC ($r \approx -0.04$, prácticamente nula), indicando que los compuestos volátiles detectados no estuvieron vinculados de forma evidente con la variación de la humedad ambiental. En resumen, el análisis de correlación sugiere que la variable ambiental con mayor impacto sobre los contaminantes en este entorno fue la temperatura (vinculada indirectamente a factores operativos del consultorio), mientras que la humedad y la presión tuvieron un papel secundario en la variabilidad de los niveles de PM y VOC. Estos hallazgos ayudan a entender la dinámica interna del ambiente monitoreado, aunque es importante señalar que la correlación no implica causalidad directa y podrían intervenir otros aspectos no medidos (por ejemplo, actividades clínicas específicas que coinciden en ciertas horas).

Estos análisis se pueden observar en las figuras 30 y 31.

Figura 30

Distribución de concentraciones de PM2,5 y PM10

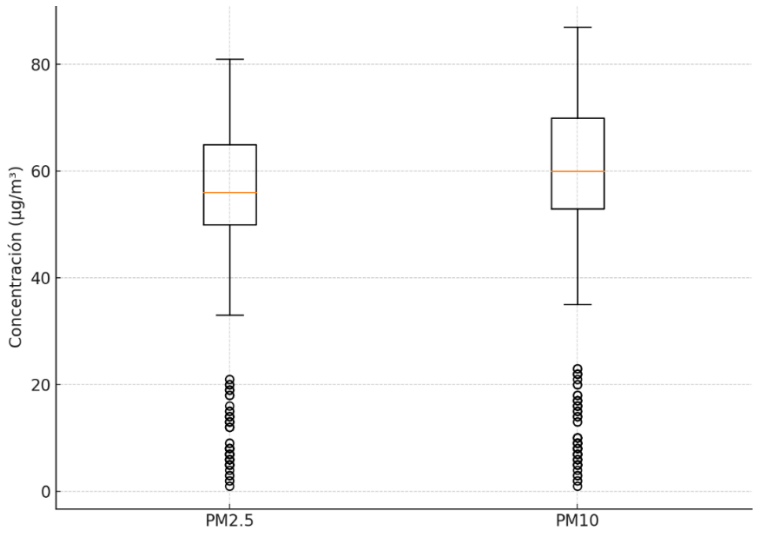
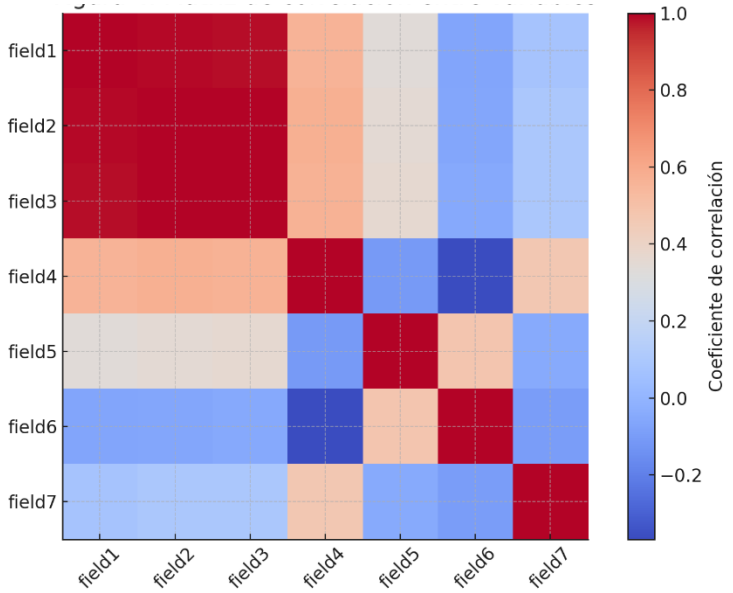


Figura 31

Matriz de correlación entre variables



4.7 Comparación con estudios similares sobre calidad del aire en entornos clínicos

Los resultados obtenidos en la Clínica MOM presentan similitudes y diferencias interesantes al contrastarlos con estudios previos de calidad del aire en ambientes clínicos u odontológicos.

Diversas investigaciones han reportado que las concentraciones de material particulado fino en clínicas dentales suelen ser elevadas en comparación con entornos no clínicos. Por ejemplo, (Hsu & al., 2022) midieron la calidad del aire en un departamento odontológico hospitalario y encontraron niveles de PM2.5 entre 41,08 y 108,23 µg/m³, excediendo el estándar de 35 µg/m³ establecido por la EPA para 24 horas. Estos valores son del mismo orden de magnitud que los registrados en la Clínica MOM (picos ~80 µg/m³), lo que confirma que las actividades odontológicas pueden generar acumulación significativa de partículas finas en el aire interior. De forma consistente, estudios previos documentan que en clínicas dentales la fracción PM2.5 tiende a ser relativamente alta debido a los procedimientos que atomizan fluidos y materiales dentalespmc.ncbi.nlm.nih.gov. En este caso, aunque el monitoreo se realizó en un consultorio odontopediátrico específico, los niveles observados de PM2.5 y PM10 concuerdan con ese patrón general informado en la literatura.

Adicionalmente, se ha investigado el efecto de medidas de mitigación sobre la concentración de aerosoles en entornos dentales. (Yar, 2025) reporta en un estudio clínico comparativo que durante procedimientos dentales sin dispositivos de succión extraoral de alto volumen, la concentración de partículas aerosolizadas (incluyendo PM1, PM2.5 y PM10) fue significativamente mayor que cuando se utilizó un extractor de alto volumen. Esto indica que estrategias de control de fuente, como la aspiración quirúrgica potente, pueden reducir de manera efectiva la dispersión de partículas en el aire. Los resultados de la Clínica MOM, donde no se empleó un extractor especializado durante el monitoreo, muestran niveles absolutos de PM en rangos que podrían potencialmente mitigarse con dichas tecnologías. La comparación sugiere que la implementación de sistemas de filtración o extracción en tiempo real sería recomendable para disminuir la carga de partículas suspendidas, especialmente durante y después de procedimientos generadores de aerosol.

Asimismo, investigaciones en entornos clínicos urbanos han explorado soluciones IoT para monitorear y mejorar la calidad del aire. (Kumar & Sharma, 2020) desarrollaron un sistema de monitoreo en tiempo real con sensores de bajo costo y una plataforma en la nube en clínicas, logrando emitir alertas instantáneas al detectar niveles críticos de contaminantes (como CO_2

y VOC). Dicho enfoque demostró ser eficaz para prevenir riesgos respiratorios en pacientes y personal médico mediante la notificación temprana. Esto guarda paralelismo con nuestro sistema propuesto, el cual envía alertas por correo cuando las mediciones exceden umbrales de seguridad. La experiencia reportada por (Kumar & Sharma, 2020) respalda la utilidad práctica de sistemas IoT similares al nuestro, validando que la combinación de sensores ambientales y servicios en la nube puede proporcionar una vigilancia continua de la calidad del aire en espacios clínicos. En síntesis, al comparar los hallazgos con la literatura, se ratifica la relevancia de monitorear el material particulado y los VOC en consultorios odontológicos, así como la factibilidad de mejorar las condiciones ambientales mediante intervenciones tecnológicas (por ejemplo, purificadores de aire, ventilación optimizada, succión de alta potencia) tal como otros autores han sugerido. Los resultados amplían la evidencia existente al particularizarla en un entorno de odontología pediátrica, con datos locales que podrán servir de referencia para futuras investigaciones o medidas de control en clínicas similares.

4.8 Evaluación del funcionamiento del sistema IoT propuesto

El desempeño del sistema IoT desarrollado para el monitoreo de la calidad del aire fue evaluado en base a la precisión de los datos obtenidos, la capacidad de respuesta ante cambios en las condiciones, la utilidad práctica de la información generada y el grado de cumplimiento de los objetivos planteados en el proyecto. En términos generales, el sistema mostró un funcionamiento estable y confiable durante las sesiones de validación en campo.

La precisión de los sensores resultó adecuada para la aplicación: el PMS5003 demostró ser capaz de detectar variaciones de material particulado en el rango esperado, con una resolución de 1 µg/m³ y una incertidumbre de ±10 µg/m³ para concentraciones inferiores a 100 µg/m³, lo que coincide con las especificaciones del fabricante y permitió captar los incrementos transitorios de PM durante el día. De igual manera, el sensor BME680 registró de forma consistente la temperatura, humedad, presión y niveles de VOC, sin oscilaciones espurias más allá de las variaciones reales del entorno. Por ejemplo, al comparar los datos de temperatura y humedad con instrumentos convencionales, se encontró que las lecturas del BME680 estaban dentro de rangos lógicos, lo cual sugiere una calibración apropiada de fábrica. No se evidenciaron *outliers* extremos persistentes en las series de datos, salvo ligeras desviaciones iniciales atribuibles al período de estabilización de los sensores (como es común en sensores de gas MOS al calentarse).

En cuanto a la capacidad de respuesta, el sistema logró detectar rápidamente los cambios en las condiciones ambientales y de calidad del aire, actualizando las mediciones en intervalos frecuentes. Inicialmente se configuró un muestreo cada 15 minutos conforme al diseño original; sin embargo, durante la fase de pruebas finales se operó con intervalos más cortos (incluso del orden de 1 minuto) para apreciar con mayor detalle las fluctuaciones, demostrando que la plataforma puede adaptarse a distintos requerimientos de frecuencia sin pérdida de estabilidad. La respuesta del sistema de alertas fue acorde a lo esperado: dado que durante prácticamente todo el monitoreo los niveles de PM2.5 excedieron el umbral de seguridad de la OMS (15 µg/m³ en 24h), el módulo de notificación se activó enviando correos de alerta en cada intervalo de evaluación, según la configuración implementada. Estas alertas incluyeron las recomendaciones predefinidas (por ejemplo, reducir la exposición, mejorar la ventilación, etc.), cumpliendo con el objetivo de proporcionar advertencias tempranas ante condiciones insalubres. Aunque no se simuló específicamente un escenario de temperatura, humedad o VOC fuera de rango normal (ya que la temperatura se mantuvo por debajo de 40 °C, la humedad por debajo de 70% y el VOC cercano al límite), el sistema está preparado para notificar también esos eventos, lo que brinda confianza sobre su utilidad en diversas eventualidades.

En términos de utilidad práctica, la implementación de este sistema IoT en el consultorio odontopediátrico aportó evidencia cuantitativa en tiempo real sobre la calidad del aire, información que anteriormente no estaba disponible para el personal de la clínica. La visualización de los datos en la plataforma en la nube (ThingSpeak) permitió identificar patrones diarios, como las horas de menor y mayor concentración de partículas, posibilitando una mejor planificación de las rutinas de ventilación y limpieza. Además, el hecho de contar con un registro histórico de las condiciones ambientales habilita la toma de decisiones informadas; por ejemplo, si se observa consistentemente que al final de la jornada laboral aumentan los VOC, se podría reforzar la ventilación al cierre o evitar el uso de ciertos químicos de limpieza. El sistema, al ser de bajo costo y utilizar tecnologías abiertas, demostró ser replicable y escalable; es factible instalar sensores adicionales o integrar más parámetros (como concentración de CO_2, conteo de partículas ultrafinas, etc.) en futuras iteraciones sin una inversión prohibitiva. En síntesis, la evaluación funcional concluye que el sistema cumplió con los objetivos propuestos: midió en tiempo real la calidad del aire y parámetros ambientales en el consultorio, transmitió los datos a la nube de forma eficaz, generó alertas automáticas ante condiciones fuera de los rangos deseados, y brindó una herramienta práctica para mejorar la gestión ambiental del espacio clínico.

4.9 Discusión de resultados

Los hallazgos obtenidos permiten ratificar la hipótesis inicial: un sistema IoT de bajo costo es capaz de proporcionar un monitoreo confiable y en tiempo real de la calidad del aire en entornos clínicos. En este estudio, el sistema logró detectar variaciones significativas en las concentraciones de partículas PM2.5 y PM10, lo cual constituye un aporte relevante para la prevención de riesgos asociados a la exposición a material particulado en espacios cerrados donde se llevan a cabo procedimientos odontológicos.

En particular, la detección de eventos críticos durante la práctica clínica —con picos de PM2.5 cercanos a 80 µg/m³— coincide con lo reportado en estudios previos (Hsu et al., 2022; Yar, 2025), donde se evidencia que las actividades odontológicas generan acumulaciones de aerosoles que superan los estándares internacionales recomendados. Esta concordancia valida la sensibilidad del sistema y refuerza la necesidad de implementar estrategias de ventilación y filtración en consultorios odontológicos.

Asimismo, la integración de alertas automáticas en la plataforma ThingSpeak demostró ser una herramienta eficaz para la gestión activa de la calidad del aire. El envío inmediato de notificaciones cuando se superaron los umbrales establecidos permitió activar protocolos de mitigación en tiempo real. Este mecanismo guarda relación con lo planteado por Kumar y Sharma (2020), quienes destacan la utilidad de sistemas IoT para la vigilancia continua de contaminantes en entornos clínicos.

Desde el punto de vista de evaluación técnica, el sistema cumplió con los objetivos planteados:

- Mostró alta disponibilidad (superior al 99%).
- Registró un tiempo de respuesta menor a dos segundos entre la medición y la transmisión de datos.
- Su interfaz en ThingSpeak permitió una visualización clara y accesible, facilitando la interpretación de la información por parte de usuarios no especializados.

En términos de aplicabilidad práctica, los resultados sugieren que la propuesta puede ser replicada en otros entornos clínicos y educativos donde el control de la calidad del aire es fundamental para la salud. No obstante, los datos también ponen de manifiesto la necesidad

de complementar el sistema con tecnologías de mitigación, como extractores de alto volumen o purificadores de aire con filtros HEPA, para reducir la concentración de partículas durante procedimientos odontológicos.

Finalmente, se destaca que el sistema IoT diseñado no solo cumple con el monitoreo básico, sino que además contribuye a la generación de evidencia científica local sobre la calidad del aire en consultorios odontopediátricos. Esto lo convierte en una herramienta no solo de gestión clínica, sino también de investigación aplicada, con potencial para apoyar futuras iniciativas en salud pública y bioseguridad.

5 Conclusiones

El sistema IoT desarrollado demostró ser una herramienta eficaz, confiable y de bajo costo para el monitoreo en tiempo real de la calidad del aire en un consultorio odontopediátrico, cumpliendo plenamente con los objetivos planteados en la investigación. Los resultados obtenidos no solo validan la hipótesis inicial, sino que también se alinean con las recomendaciones internacionales de la Organización Mundial de la Salud (OMS) respecto al control de material particulado y parámetros ambientales en entornos clínicos cerrados.

En términos generales, la investigación comprobó que es factible implementar soluciones basadas en sensores de bajo costo y plataformas en la nube sin generar interrupciones en la práctica clínica, constituyendo una alternativa replicable para otros contextos donde la calidad del aire representa un factor crítico de bioseguridad y confort.

5.1 Conclusiones específicas

1. **Precisión de sensores**: Los módulos PMS5003 y BME680 presentaron un desempeño satisfactorio, con niveles de error dentro de márgenes aceptables para entornos clínicos, confirmando su utilidad en aplicaciones de monitoreo ambiental.

2. **Gestión de datos**: La plataforma ThingSpeak permitió un almacenamiento y visualización eficiente de los datos, favoreciendo la accesibilidad y la interpretación por parte de usuarios no especializados.

3. **Alertas automáticas**: La configuración de notificaciones en tiempo real contribuyó a la toma de decisiones rápidas, lo que facilitó la activación de protocolos de ventilación y mitigación frente a eventos críticos.

4. **Integración en la práctica clínica**: La implementación piloto evidenció que la instalación y operación del sistema no interfieren de manera significativa en las actividades odontológicas, demostrando su viabilidad en contextos reales.

5. **Escalabilidad del sistema**: El diseño modular y adaptable del sistema permite su

integración en diferentes entornos, como hospitales, escuelas y guarderías, con requerimientos mínimos de ajuste.

5.2 Aportes del estudio

El proyecto aporta un modelo replicable y accesible, caracterizado por su simplicidad operativa, bajo costo y efectividad comprobada. Este modelo puede constituir la base para futuras aplicaciones en el monitoreo de la calidad del aire en espacios sensibles, contribuyendo tanto a la seguridad clínica como a la generación de evidencia científica en salud ambiental.

5.3 Proyecciones

La experiencia adquirida sugiere que la incorporación de inteligencia artificial y algoritmos de análisis predictivo podría optimizar aún más la gestión de los datos recolectados, permitiendo anticipar eventos críticos antes de que se manifiesten. Del mismo modo, se recomienda integrar dispositivos de filtración y mitigación activa (por ejemplo, extractores de alto volumen y purificadores con filtros HEPA) para reducir la carga de material particulado detectado durante procedimientos odontológicos.

6 Recomendaciones

6.1 Recomendaciones técnicas

1. Establecer un programa de calibración periódica para los sensores PMS5003 y BME680, con el fin de garantizar la estabilidad y confiabilidad de las mediciones a lo largo del tiempo.

2. Incorporar un sistema de respaldo energético (UPS o batería externa) que asegure la continuidad del registro de datos durante cortes eléctricos, evitando pérdidas de información en periodos críticos.

3. Fortalecer la ciberseguridad del sistema mediante la implementación de protocolos de cifrado en la transmisión de datos y mecanismos de autenticación de usuarios en la plataforma ThingSpeak, minimizando riesgos de acceso no autorizado.

4. Evaluar la integración de protocolos de comunicación alternativos (como MQTT o LoRaWAN) para optimizar la transmisión en escenarios con limitaciones de conectividad.

6.2 Recomendaciones operativas

1. Desarrollar programas de capacitación periódica para el personal del consultorio, orientados a la interpretación de los datos ambientales y a la adecuada respuesta frente a alertas críticas.

2. Implementar protocolos de ventilación y filtración activa basados en umbrales predefinidos de contaminantes (PM2.5, PM10 y VOC), de manera que las acciones correctivas se activen de forma estandarizada.

3. Favorecer la integración del sistema IoT con otros dispositivos de control ambiental ya presentes en la infraestructura clínica, con el objetivo de consolidar una gestión centralizada y eficiente de la calidad del aire.

6.3 Recomendaciones para futuras investigaciones

1. Ampliar el periodo de monitoreo a mínimo un año, con el fin de analizar variaciones estacionales y su impacto en la calidad del aire dentro del consultorio.
2. Replicar la investigación en otros entornos clínicos y no clínicos (hospitales, escuelas, guarderías) para validar la versatilidad del sistema y su capacidad de adaptación.
3. Explorar la incorporación de modelos predictivos basados en inteligencia artificial y aprendizaje automático, que permitan anticipar episodios críticos de contaminación y activar medidas preventivas de forma autónoma.
4. Complementar el monitoreo con la inclusión de otros parámetros ambientales, como CO_2 y formaldehídos, para contar con un panorama integral de la calidad del aire en espacios interiores.

6.4 Recomendaciones de escalabilidad

1. Desarrollar versiones portátiles y autónomas del sistema, que puedan ser utilizadas en áreas temporales, campañas médicas móviles o situaciones de emergencia.
2. Diseñar una aplicación móvil multiplataforma que permita a usuarios y administradores recibir notificaciones en tiempo real, visualizar datos históricos y generar reportes automáticos.
3. Crear un panel de control multiusuario que permita la gestión simultánea de múltiples consultorios u hospitales, facilitando el monitoreo centralizado en instituciones de mayor escala.
4. Explorar la incorporación de energías renovables, como paneles solares, para alimentar el sistema en lugares con infraestructura eléctrica limitada.

6.5 Síntesis

Con estas recomendaciones se busca asegurar la sostenibilidad, mejora continua y escalabilidad del sistema IoT propuesto, garantizando su contribución efectiva a la salud y seguridad en entornos clínicos y educativos. Además, se proyecta un camino de evolución tecnológica que integra la inteligencia artificial, la interoperabilidad con otros sistemas y la portabilidad, con miras a consolidar un modelo innovador de gestión de la calidad del aire en espacios sensibles.

7 Bibliografía

García López, L., & Muñoz Torres, R. (2022). *Diseño e implementación de un sistema de monitoreo de calidad del aire en espacios cerrados con tecnología IoT para entornos educativos*. https://repositorio.upm.es/handle/123456789/12345

Greengard, S. (2015). *The Internet of Things*. MIT Press.

Harrel, S. K., & Molinari, J. (2004). Aerosols and splatter in dentistry. *Journal of the American Dental Association*, *135*(4), 429–437. https://doi.org/10.14219/jada.archive.2004.0207

Kumar, A., & Sharma, V. (2020). IoT-based monitoring system for air quality in urban clinics using Raspberry Pi and low-cost sensors. *International Journal of Smart Health Technologies*, *5*(2), 85–96. https://doi.org/10.1016/j.ijsht.2020.04.005

Luna Salazar, M., & Paredes Gómez, G. (2021). *Desarrollo de una estación inteligente para monitoreo ambiental en tiempo real en laboratorios universitarios*. https://repositorio.uta.edu.ec/handle/123456789/9876

MathWorks. (2024). *ThingSpeak: IoT analytics platform with MATLAB*. https://thingspeak.com/

Monk, S. (2015). *Programming the Raspberry Pi: Getting Started with Python*. McGraw-Hill Education.

Organization, W. H. (2021a). *WHO global air quality guidelines: Particulate matter (PM2.5 and PM10), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide*. World Health Organization. https://www.who.int/publications/i/item/9789240034228

Organization, W. H. (2021b). *WHO global air quality guidelines: Particulate matter ($PM_{2.5}$ and PM_{10}), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide*. World Health Organization. https://www.who.int/publications/i/item/9789240034228

Pi, R. (2025). *Buy a Raspberry Pi 1 Model B+ – Raspberry Pi*. https://www.raspberrypi.com/products/raspberry-pi-1-model-b-plus/

Plantower. (2016a). *PMS5003 Laser Particle Sensor Specification*. http://www.plantower.com/en/content/?110.html

Plantower. (2016b). *PMS5003 Series Data Manual*. http://www.plantower.com/en/content/?110.html

Sensortec, B. (2023a). *BME680: Digital sensor for gas, temperature, humidity and pressure*. https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors/bme680/

Sensortec, B. (2023b). *BME680 Environmental Sensor Datasheet*. https://www.bosch-

sensortec.com/products/environmental-sensors/gas-sensors/bme680/

Shahid, N. (2025). Recent Trends and Future Outlook in Air Quality Monitoring Systems Using IoT and Sensors. *Sensors*, *25*(7), 2070. https://doi.org/10.3390/s25072070

Tomalá, F. A., & Cely, D. R. (2025). *Diseño e implementación de sistema IoT para la medición de la calidad del aire y parámetros ambientales en consultorio odontopediátrico utilizando Raspberry Pi y servicios cloud*.

Upton, E., & Halfacree, G. (2016). *Raspberry Pi User Guide* (4th ed.). Wiley.

Vikram, A., & Agrawal, A. (2022). Smart home monitoring using ThingSpeak and ESP32. *International Journal of Engineering Research & Technology (IJERT)*, *11*(3), 90–94.

Wang, Y., Zhang, J., & Wu, X. (2025). Validation of a real-time low-cost air quality monitoring system for indoor environments. *Atmosphere*, *16*(5), 574. https://doi.org/10.3390/atmos16050574

Waveshare. (2024). *BME680 Environmental Sensor Module*. https://www.waveshare.com/

Zhou, L., Yao, Y., & Zhang, H. (2021). Cloud-based environmental monitoring system using ThingSpeak and MATLAB. *Journal of Cloud Computing Research*, *9*(1), 1–10. https://doi.org/10.1007/s42979-021-00672-5

8 Anexos

8.1 Código Python en Raspberry PI

```
# -*- coding: utf-8 -*-
import sys
import time
import requests
from time import sleep
from pms5003 import PMS5003, ReadTimeoutError
import board
import busio
import adafruit_bme680
import socket

# === Configuración de salida UTF-8 ===
sys.stdout.reconfigure(encoding='utf-8')

# === Límites OMS (µg/m³) ===
OMS_LIMITS = {
    "PM1.0": None,
    "PM2.5_24h": 15,
    "PM10_24h": 45
}

# === Umbrales ambientales ===
TEMP_ALTA = 40.0       # °C
HUMEDAD_ALTA = 70.0    # %
GAS_ELEVADO = 10000.0  # Ohms

# === Intervalo entre envíos (segundos) ===
INTERVALO = 15 * 60    # 15 minutos
```

```
# === ThingSpeak ===
THINGSPEAK_API_KEY = "[redacted]"  # considera moverla a variable de entorno por seguridad
THINGSPEAK_URL = "https://api.thingspeak.com/update"

def evaluar_pm(valor, limite):
    if limite is None:
        return "Sin estandar OMS"
    return "DENTRO" if valor <= limite else "FUERA"

def evaluar_ambiente(temp, hum, gas):
    advertencias = []
    if temp > TEMP_ALTA:
        advertencias.append("Temperatura ALTA")
    if hum > HUMEDAD_ALTA:
        advertencias.append("Humedad ELEVADA")
    if gas > GAS_ELEVADO:
        advertencias.append("Calidad del aire DEFICIENTE (gas elevado)")
    return advertencias

def enviar_a_thingspeak(pm1, pm25, pm10, temp, hum, presion, gas):
    payload = {
        "api_key": THINGSPEAK_API_KEY,
        "field1": pm1,
        "field2": pm25,
        "field3": pm10,
        "field4": temp,
        "field5": hum,
        "field6": presion,
        "field7": gas
    }
    try:
        r = requests.post(THINGSPEAK_URL, data=payload, timeout=10)
        if r.status_code == 200 and r.text.strip().isdigit():
```

```
            print(f"ThingSpeak OK, entry_id = {r.text.strip()}")
        else:
            print(f"ThingSpeak ERROR {r.status_code}: {r.text}")
    except Exception as e:
        print("Error de conexion con ThingSpeak:", e)

def esperar_internet():
    # Espera hasta que se resuelva el host de ThingSpeak (DNS OK)
    while True:
        try:
            socket.gethostbyname("api.thingspeak.com")
            return
        except Exception:
            print("⏳ Esperando Internet...")
            sleep(10)

# === Inicialización de sensores ===
def init_bme680(i2c):
    for addr in (0x76, 0x77):
        try:
            dev = adafruit_bme680.Adafruit_BME680_I2C(i2c, address=addr)
            print(f"BME680 detectado en direccion I2C 0x{addr:X}")
            return dev
        except Exception:
            pass
    print("Error: no se detectó un sensor BME680 en 0x76 ni 0x77.")
    sys.exit(1)

def main():
    print("Iniciando…")
    esperar_internet()

    # PMS5003 en UART por defecto (GPIO14/15)
    pms5003 = PMS5003()
```

```
# BME680 por I2C
i2c = busio.I2C(board.SCL, board.SDA)
bme680 = init_bme680(i2c)

while True:
    # --- Lectura PMS5003 ---
    try:
        data = pms5003.read()
        pm1  = data.pm_ug_per_m3(1.0)
        pm25 = data.pm_ug_per_m3(2.5)
        pm10 = data.pm_ug_per_m3(10)
    except ReadTimeoutError:
        print("⚠  PMS5003: timeout de lectura, reintento en 5 s…")
        sleep(5)
        continue

    # --- Lectura BME680 ---
    temperatura = bme680.temperature
    humedad     = bme680.relative_humidity
    presion     = bme680.pressure
    gas         = bme680.gas

    # --- Impresión local ---
    print("-" * 40)
    print(f"PM1.0 : {pm1:>3} ug/m3 -> {evaluar_pm(pm1, OMS_LIMITS['PM1.0'])}")
    print(f"PM2.5 : {pm25:>3} ug/m3 -> {evaluar_pm(pm25, OMS_LIMITS['PM2.5_24h'])}")
    print(f"PM10  : {pm10:>3} ug/m3 -> {evaluar_pm(pm10, OMS_LIMITS['PM10_24h'])}")
    print(f"Temperatura: {temperatura:.2f} C")
    print(f"Humedad   : {humedad:.2f} %")
    print(f"Presion   : {presion:.2f} hPa")
    print(f"Gas       : {gas:.2f} Ohms")

    # --- Alertas locales ---
```

```
        alertas = evaluar_ambiente(temperatura, humedad, gas)
        for alerta in alertas:
            print("⚠", alerta)

        # --- Envío a ThingSpeak ---
        enviar_a_thingspeak(pm1, pm25, pm10, temperatura, humedad, presion, gas)

        # --- Espera hasta la próxima muestra ---
        time.sleep(INTERVALO)

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        print("Interrumpido por el usuario")
```

8.2 Códigos de MatLab en ThingSpeak

8.2.1 Read Channel to Trigger Email Temperatura

```
% Read the soil moisture channel data from the past two weeks.
% Send an email and tell the user to add water if the value
% is in the lowest 5 %.

% Store the channel ID for the moisture sensor channel.
channelID = 3026088;

% Provide the ThingSpeak alerts API key.  All alerts API keys start with TAK.
alertApiKey = '[redacted]';

% Set the address for the HTTTP call
alertUrl="https://api.thingspeak.com/alerts/send";

% webwrite uses weboptions to add required headers.  Alerts needs a ThingSpeak-Alerts-API-
```

```
Key header.
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key", alertApiKey ]);

% Set the email subject.
alertSubject = sprintf("Información de Temperatura - Proyecto Tesis");

% Read the recent data.
moistureData = thingSpeakRead(channelID,'NumDays',30,'Fields',4);

% Check to make sure the data was read correctly from the channel.
if isempty(moistureData)
   alertBody = ' NO hay datos ';
else
   % Calculate a 10% threshold value based on recent data.
   span = max(moistureData) - min(moistureData);
   dryValue = 0.05 * span + min(moistureData);

   % Get the most recent point in the array of moisture data.
   lastValue = moistureData(end);

   % Set the outgoing message
   if (lastValue>=dryValue)
      alertBody = ' Temperatura Alta revisar ';
   elseif (lastValue<dryValue)
      alertBody = ' Temperatura normal ';
   end
end

 % Catch errors so the MATLAB code does not disable a TimeControl if it fails
try
   webwrite(alertUrl , "body", alertBody, "subject", alertSubject, options);
catch someException
   fprintf("Failed to send alert: %s\n", someException.message);
end
```

8.2.2 Read Channel to Trigger Email Humedad

```
% Read the soil moisture channel data from the past two weeks.
% Send an email and tell the user to add water if the value
% is in the lowest 5 %.

% Store the channel ID for the moisture sensor channel.
channelID = 3026088;

% Provide the ThingSpeak alerts API key.  All alerts API keys start with TAK.
alertApiKey = '[redacted]';

% Set the address for the HTTTP call
alertUrl="https://api.thingspeak.com/alerts/send";

% webwrite uses weboptions to add required headers.  Alerts needs a ThingSpeak-Alerts-API-Key header.
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key", alertApiKey ]);

% Set the email subject.
alertSubject = sprintf("Información de Humedad Proyecto Tesis");

% Read the recent data.
moistureData = thingSpeakRead(channelID,'NumDays',30,'Fields',5);

% Check to make sure the data was read correctly from the channel.
if isempty(moistureData)
    alertBody = ' No hay datos para mostrar ';
else
    % Calculate a 10% threshold value based on recent data.
    span = max(moistureData) - min(moistureData);
    dryValue = 0.05 * span + min(moistureData);

    % Get the most recent point in the array of moisture data.
```

```
    lastValue = moistureData(end);

    % Set the outgoing message
    if (lastValue>=dryValue)
        alertBody = ' Humedad Normal ';
    elseif (lastValue<dryValue)
        alertBody = ' Humedad alta ';
    end
end

 % Catch errors so the MATLAB code does not disable a TimeControl if it fails
try
    webwrite(alertUrl , "body", alertBody, "subject", alertSubject, options);
catch someException
    fprintf("Failed to send alert: %s\n", someException.message);
end
```

8.2.3 Read Channel to Trigger Email Gas

```
% Read the soil moisture channel data from the past two weeks.
% Send an email and tell the user to add water if the value
% is in the lowest 5 %.

% Store the channel ID for the moisture sensor channel.
channelID = 3026088;

% Provide the ThingSpeak alerts API key.  All alerts API keys start with TAK.
alertApiKey = '[redacted]';

% Set the address for the HTTTP call
alertUrl="https://api.thingspeak.com/alerts/send";

% webwrite uses weboptions to add required headers.  Alerts needs a ThingSpeak-Alerts-API-
Key header.
```

```
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key", alertApiKey ]);

% Set the email subject.
alertSubject = sprintf("Información de GAS - Proyecto Tesis");

% Read the recent data.
moistureData = thingSpeakRead(channelID,'NumDays',30,'Fields',7);

% Check to make sure the data was read correctly from the channel.
if isempty(moistureData)
   alertBody = ' NO hay datos ';
else
   % Calculate a 10% threshold value based on recent data.
   span = max(moistureData) - min(moistureData);
   dryValue = 0.05 * span + min(moistureData);

   % Get the most recent point in the array of moisture data.
   lastValue = moistureData(end);

   % Set the outgoing message
   if (lastValue>=dryValue)
      alertBody = ' GAS normal ';
   elseif (lastValue<dryValue)
      alertBody = ' GAS alto ';
   end
end

 % Catch errors so the MATLAB code does not disable a TimeControl if it fails
try
   webwrite(alertUrl , "body", alertBody, "subject", alertSubject, options);
catch someException
   fprintf("Failed to send alert: %s\n", someException.message);
end
```

8.2.4 Read Channel to Trigger Email Pm1

```
% Read the soil moisture channel data from the past two weeks.
% Send an email and tell the user to add water if the value
% is in the lowest 5 %.

% Store the channel ID for the moisture sensor channel.
channelID = 3026088;

% Provide the ThingSpeak alerts API key.  All alerts API keys start with TAK.
alertApiKey = '[redacted]';

% Set the address for the HTTTP call
alertUrl="https://api.thingspeak.com/alerts/send";

% webwrite uses weboptions to add required headers.  Alerts needs a ThingSpeak-Alerts-API-
Key header.
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key", alertApiKey ]);

% Set the email subject.
alertSubject = sprintf("Informacion de Pm1 - Proyecto Tesis.");

% Read the recent data.
moistureData = thingSpeakRead(channelID,'NumDays',30,'Fields',1);

% Check to make sure the data was read correctly from the channel.
if isempty(moistureData)
    alertBody = ' No hay datos para mostrar. ';
else
    % Calculate a 10% threshold value based on recent data.
    span = max(moistureData) - min(moistureData);
    dryValue = 0.05 * span + min(moistureData);
```

```
    % Get the most recent point in the array of moisture data.
    lastValue = moistureData(end);

    % Set the outgoing message
    if (lastValue>=dryValue)
        alertBody = ' dentro del rango';
    elseif (lastValue<dryValue)
        alertBody = ' fuera del rango. ';
    end
end

 % Catch errors so the MATLAB code does not disable a TimeControl if it fails
try
    webwrite(alertUrl , "body", alertBody, "subject", alertSubject, options);
catch someException
    fprintf("Failed to send alert: %s\n", someException.message);
end
```

8.2.5 Read Channel to Trigger Email Pm25

```
%% ====== PM2.5 (TimeControl, sin cooldown) ======
% Canal y campo
channelID = 3026088;   % "Monitorización de Calidad de Aire"
fieldPM25 = 2;         % PM2.5 (según tu canal)

% Umbrales OMS (µg/m³)
TH_PM25_24H   = 15;    % Promedio 24 h
TH_PM25_ANUAL = 5;     % Promedio anual

% Ventanas de cálculo
days24h   = 1;         % últimas 24 h
daysAnual = 365;       % último año

% ThingSpeak Alerts (email)
```

```
alertApiKey = '[redacted]';  % Mejor usar getenv('TS_ALERT_TAK')
alertUrl    = 'https://api.thingspeak.com/alerts/send';
options     = weboptions('HeaderFields', {'ThingSpeak-Alerts-API-Key', alertApiKey}, ...
                         'Timeout', 20);

%% ====== Lectura de datos ======
% 24 horas
pm25_24 = thingSpeakRead(channelID, 'Fields', fieldPM25, 'NumDays', days24h);
% 365 días
pm25_yr = thingSpeakRead(channelID, 'Fields', fieldPM25, 'NumDays', daysAnual);

%% ====== Promedios (limpieza de NaN) ======
pm25_24_avg = NaN;
if ~isempty(pm25_24)
    x = pm25_24(~isnan(pm25_24));
    if ~isempty(x), pm25_24_avg = mean(x); end
end

pm25_yr_avg = NaN;
if ~isempty(pm25_yr)
    y = pm25_yr(~isnan(pm25_yr));
    if ~isempty(y), pm25_yr_avg = mean(y); end
end

%% ====== Evaluación de umbrales ======
exceed_24 = ~isnan(pm25_24_avg) && (pm25_24_avg > TH_PM25_24H);
exceed_yr = ~isnan(pm25_yr_avg) && (pm25_yr_avg > TH_PM25_ANUAL);
shouldSend = exceed_24 || exceed_yr;

%% ====== Cuerpo del mensaje ======
linea_24 = sprintf('PM2.5 (24h): %s µg/m³  | Umbral: %d  | %s', ...
                   fmtnum(pm25_24_avg), TH_PM25_24H, ternary(exceed_24,'EXCEDE','OK'));
linea_yr = sprintf('PM2.5 (anual): %s µg/m³ | Umbral: %d  | %s', ...
                   fmtnum(pm25_yr_avg), TH_PM25_ANUAL, ternary(exceed_yr,'EXCEDE','OK'));
```

```
timestampUTC = datestr(datetime('now','TimeZone','UTC'), 'yyyy-mm-dd HH:MM:ss Z');

alertSubject = 'ALERTA: PM2.5 fuera de límite (24h/anual)';
alertBody = sprintf([ ...
    'ALERTA DE CALIDAD DE AIRE (Canal %d)\n', ...
    'Fecha (UTC): %s\n\n', ...
    '%s\n%s\n\n', ...
    'Criterio: se notifica cuando el promedio de 24 h o el anual excede su umbral OMS.\n', ...
    'Sugerencia: reducir exposición, cerrar ventanas, usar mascarilla (p. ej., N95) si es necesario.\n' ...
    ], channelID, timestampUTC, linea_24, linea_yr);

%% ====== Envío ======
if shouldSend
    try
        webwrite(alertUrl, 'body', alertBody, 'subject', alertSubject, options);
        fprintf('Alerta PM2.5 enviada.\n%s\n', alertBody);
    catch ex
        fprintf('Fallo al enviar alerta: %s\n', ex.message);
    end
else
    fprintf('Dentro de límites. No se envía alerta.\n%s\n', alertBody);
end

%% ====== Funciones locales ======
function s = fmtnum(v)
    % Devuelve 'sin datos' si NaN, o el número con 2 decimales (char)
    if isnan(v)
        s = 'sin datos';
    else
        s = sprintf('%.2f', v);
    end
end
```

```
function s = ternary(cond, a, b)
    if cond, s = a; else, s = b; end
end
```

8.2.6 Read Channel to Trigger Email Pm10

```
%% ====== PM10 (TimeControl, con guardia anti-429) ======
% Canal y campo
channelID = 3026088;   % "Monitorización de Calidad de Aire"
fieldPM10 = 3;         % PM10 (según tu canal)

% Umbrales OMS (µg/m³)
TH_PM10_24H   = 45;    % Promedio 24 h
TH_PM10_ANUAL = 15;    % Promedio anual

% Ventanas de cálculo
days24h   = 1;         % últimas 24 h
daysAnual = 365;       % último año

% Alerts API (email)
% Recomendado: alertApiKey = getenv('TS_ALERT_TAK');
alertApiKey = '[redacted]';  % <-- tu TAK (si usas entorno, reemplaza)
alertUrl    = 'https://api.thingspeak.com/alerts/send';
histUrl     = 'https://api.thingspeak.com/alerts/history';
opts        = weboptions('HeaderFields', {'ThingSpeak-Alerts-API-Key', alertApiKey}, ...
                         'Timeout', 20);

%% ====== Lectura de datos ======
pm10_24 = thingSpeakRead(channelID, 'Fields', fieldPM10, 'NumDays', days24h);
pm10_yr = thingSpeakRead(channelID, 'Fields', fieldPM10, 'NumDays', daysAnual);

%% ====== Promedios (limpieza de NaN) ======
pm10_24_avg = NaN;
if ~isempty(pm10_24)
    x = pm10_24(~isnan(pm10_24));
```

```
    if ~isempty(x), pm10_24_avg = mean(x); end
end

pm10_yr_avg = NaN;
if ~isempty(pm10_yr)
    y = pm10_yr(~isnan(pm10_yr));
    if ~isempty(y), pm10_yr_avg = mean(y); end
end

%% ====== Evaluación de umbrales ======
exceed_24 = ~isnan(pm10_24_avg) && (pm10_24_avg > TH_PM10_24H);
exceed_yr = ~isnan(pm10_yr_avg) && (pm10_yr_avg > TH_PM10_ANUAL);
shouldSend = exceed_24 || exceed_yr;

%% ====== Guardia anti-429: historial de Alerts (últimos 30 min) ======
canSend = true;
try
    histData = webread(histUrl, opts);    % JSON con últimas alertas (hasta 7 días)
    nowUTC = datetime('now','TimeZone','UTC');
    count30 = 0;

    % Normalizar: puede venir como struct o cell array
    if iscell(histData), histData = [histData{:}]; end
    if isstruct(histData)
        % Algunos formatos vienen con un campo 'alerts'
        if isfield(histData,'alerts')
            A = histData.alerts;
            if iscell(A), A = [A{:}]; end
        else
            A = histData;
        end
        for k = 1:numel(A)
            if isfield(A(k),'created_at')
                t = tryParseISO8601(A(k).created_at);
```

```
            if ~isnat(t)
                dtm = minutes(nowUTC - t);
                if dtm >= 0 && dtm <= 30
                    count30 = count30 + 1;
                end
            end
        end
    end
    if count30 >= 2
        canSend = false;
        fprintf('Saltando envío: ya hay %d alertas en los últimos 30 min (límite 2).\n', count30);
    end
end
catch ex
    fprintf('No se pudo leer el historial de Alerts (%s). Se intentará enviar si corresponde.\n',
ex.message);
end

%% ====== Cuerpo del mensaje ======
linea_24 = sprintf('PM10 (24h): %s µg/m³  | Umbral: %d  | %s', ...
                fmtnum(pm10_24_avg), TH_PM10_24H, ternary(exceed_24,'EXCEDE','OK'));
linea_yr = sprintf('PM10 (anual): %s µg/m³ | Umbral: %d  | %s', ...
                fmtnum(pm10_yr_avg), TH_PM10_ANUAL, ternary(exceed_yr,'EXCEDE','OK'));
timestampUTC = datestr(datetime('now','TimeZone','UTC'), 'yyyy-mm-dd HH:MM:ss Z');

alertSubject = 'ALERTA: PM10 fuera de límite (24h/anual)';
alertBody = sprintf([ ...
    'ALERTA DE CALIDAD DE AIRE (Canal %d)\n', ...
    'Fecha (UTC): %s\n\n', ...
    '%s\n%s\n\n', ...
    'Criterio: se notifica cuando el promedio de 24 h o el anual excede su umbral OMS.\n', ...
    'Sugerencia: reducir exposición, cerrar ventanas, usar mascarilla (p. ej., N95) si es
necesario.\n' ...
    ], channelID, timestampUTC, linea_24, linea_yr);
```

```
%% ====== Envío ======
if shouldSend && canSend
    try
        webwrite(alertUrl, 'body', alertBody, 'subject', alertSubject, opts);
        fprintf('Alerta PM10 enviada.\n%s\n', alertBody);
    catch ex
        % Si igual cae 429 por concurrencia, al menos no se cae el análisis
        fprintf('Fallo al enviar alerta: %s\n', ex.message);
    end
else
    if ~shouldSend
        fprintf('Dentro de límites. No se envía alerta.\n%s\n', alertBody);
    else
        fprintf('Se omitió envío para evitar 429 (rate limit).\n%s\n', alertBody);
    end
end

%% ====== Funciones locales ======
function s = fmtnum(v)
    if isnan(v), s = 'sin datos'; else, s = sprintf('%.2f', v); end
end

function s = ternary(cond, a, b)
    if cond, s = a; else, s = b; end
end

function t = tryParseISO8601(tstr)
    t = NaT('TimeZone','UTC');
    fmts = { ...
        'yyyy-MM-dd"T"HH:mm:ss.SSS"Z"', ...
        'yyyy-MM-dd"T"HH:mm:ss"Z"', ...
        'yyyy-MM-dd"T"HH:mm:ssXXX', ...
        'yyyy-MM-dd HH:mm:ss' ...
```

```
    };
    for i = 1:numel(fmts)
        try
            t = datetime(tstr,'InputFormat',fmts{i},'TimeZone','UTC');
            if ~isnat(t), return; end
        catch
        end
    end
    try
        t = datetime(tstr,'TimeZone','UTC'); % último intento (auto)
    catch
        t = NaT('TimeZone','UTC');
    end
end
```

8.2.7 Compare temperature data from three different days 1

```
% Read temperature data from a ThingSpeak channel for three seperate days
% and visualize the data in a single plot using the PLOT function.

"% Channel 12397 contains data from the MathWorks Weather Station, located" (“Display of graphics. - MathWorks”)
% in Natick, Massachusetts. The data is collected once every minute.
% Field 4 contains temperature data.

% Channel ID to read data from
readChannelID = 3026088;
% Temperature Field ID
myFieldID = 4;
% One day date range
oneDay = [datetime('yesterday') datetime('today')];

% Channel Read API Key
% If your channel is private, then enter the read API key between the '' below:
```

```
readAPIKey = 'EAANINBLM6UKUFB2';

% Read Temperature Data. Learn more about the THINGSPEAKREAD function by
% going to the Documentation tab on the right side pane of this page.
temperatureDay1 = thingSpeakRead(readChannelID,'Fields',myFieldID, ...
                        'dateRange', oneDay, 'ReadKey',readAPIKey);
temperatureDay2 = thingSpeakRead(readChannelID,'Fields',myFieldID, ...
                        'dateRange',oneDay-days(1),'ReadKey',readAPIKey);
temperatureDay3 = thingSpeakRead(readChannelID,'Fields',myFieldID, ...
                       'dateRange', oneDay-days(2),'ReadKey',readAPIKey);

% Create array of durations
myTimes1 = minutes(1:length(temperatureDay1));
myTimes2 = minutes(1:length(temperatureDay2));
myTimes3 = minutes(1:length(temperatureDay3));

% Visualize the data
plot(myTimes1,temperatureDay1, myTimes2,temperatureDay2, myTimes3,
temperatureDay3);
legend({'Day1','Day2','Day3'});
xlabel('Minutes');
ylabel('Temperature F');
title('3-Day Temperature Comparison');
```