Trabajo Fin de Grado
en Ingeniería de las Tecnologías de Telecomunicación

Implementación de un alcoholímetro mediante Raspberry Pi

Autor: Pablo Zurbano Canela

Tutor: Bernardo Palomo Vázquez

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2025

Trabajo Fin de Grado
en Ingeniería de las Tecnologías de Telecomunicación

Implementación de un alcoholímetro mediante Raspberry Pi

Autor:
Pablo Zurbano Canela

Tutor:
Bernardo Palomo Vázquez
Profesor titular

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2025

Trabajo Fin de Grado: Implementación de un alcoholímetro mediante Raspberry Pi

Autor: Pablo Zurbano Canela

Tutor: Bernardo Palomo Vázquez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2025

El Secretario del Tribunal

A mi familia,

A mis amigos,

A mis maestros,

A mí mismo.

Agradecimientos

A los profesores que han formado parte de mi educación académica, porque todos ellos, en mayor o menor medida, también son responsables de este documento, en especial a Bernardo, por confiar en mí para llevar a cabo este trabajo de fin de grado y acompañarme en esta última etapa de la carrera.

A mis compañeros de carrera, en especial a Adri y Chelo, gracias por todos los mensajes intercambiados intentando dar solución a problemas que parecía que no la tenían, gracias por el apoyo que habéis sido sobre todo en esas asignaturas que tantas convocatorias nos han supuesto, tenéis gran culpa de que estos años hayan sido más llevaderos.

A Sale, Macarena y Valle, por pensar siempre en mi en época de exámenes, por desearme siempre suerte y por demostrarme tanto cariño desde pequeño.

A mis amigos de toda la vida, Manu, Jorge, Trujillo, Laura y Josema, gracias por darme ese empujón necesario en los momentos duros, nada me hace más feliz que poder celebrar juntos cada uno de nuestros logros.

A mi padre, a mi madre y a mi hermano, gracias por la educación que me habéis dado, por construir la persona que soy y por soportar siempre con una sonrisa el esfuerzo económico y emocional que ha supuesto esta etapa, sin vosotros habría sido imposible.

A ti Pablo, por escoger este reto, por trabajar, por nunca tirar la toalla. El camino se nos ha hecho más largo de lo que esperábamos en un principio, pero al final lo hemos conseguido.

Gracias Pablo, por confiar en el proceso.

Pablo Zurbano Canela

Sevilla, 2025

Resumen

El presente Trabajo de Fin de Grado aborda el diseño e implementación de un alcoholímetro de bajo coste basado en una Raspberry Pi 4 Model B y un sensor de gas MQ-3. El objetivo principal ha sido desarrollar un sistema capaz de medir la concentración de alcohol en aire espirado, procesar la señal y mostrar los resultados en una interfaz web sencilla e intuitiva.

Para solventar la ausencia de entradas analógicas en la Raspberry Pi, se ha incorporado el conversor analógico-digital MCP3008, el cual permite digitalizar la señal analógica que aporta el sensor y transmitirla mediante el protocolo SPI. Tras la implementación del esquema eléctrico de conexiones, se ha desarrollado un código en Python que gestiona la calibración del sensor, el procesamiento de datos y la conversión de voltajes en concentraciones estimadas de etanol expresadas en mg/L. Los datos más relevantes de cada medición son almacenados en un fichero CSV, permitiendo así la consulta de un historial de mediciones en todo momento.

La interfaz web, implementada con el microframework web Flask, ofrece al usuario un entorno simple para realizar pruebas en tiempo real y visualizar los datos. Tras una primera calibración del sensor en aire limpio, se incluyen funcionalidades como: el registro del nombre del usuario, la medición de la cantidad de alcohol en aire y el acceso a un historial de resultados previos.

Los experimentos realizados con distintas bebidas alcohólicas han mostrado una respuesta coherente del sistema, aunque con limitaciones propias del sensor MQ-3, el cual se muestra sensible a factores externos como temperatura, humedad o vapores interferentes. Estos resultados, aunque no permiten un uso médico ni legal del prototipo, validan su utilidad como herramienta educativa y de concienciación, además de demostrar la viabilidad de la integración de sensores de bajo coste con plataformas de desarrollo accesibles como la Raspberry Pi.

En conclusión, este trabajo evidencia que se ha alcanzado el objetivo fundamental de poner en funcionamiento un sistema completo que combina electrónica, programación y desarrollo web, y que sienta las bases para futuras mejoras mediante sensores más avanzados, métodos de calibración más rigurosos y mecanismos de soplado controlados.

Abstract

This Final Degree Project addresses the design and implementation of a low-cost breathalyzer based on a Raspberry Pi 4 Model B and an MQ-3 gas sensor. The main objective has been to develop a system capable of measuring alcohol concentration in exhaled air, processing the signal, and displaying the results through a simple and intuitive web interface.

To overcome the absence of native analog inputs in the Raspberry Pi, the MCP3008 analog-to-digital converter has been incorporated, enabling the digitization of the sensor's analog signal and its transmission via the SPI protocol. After implementing the electrical connection scheme, a Python program was developed to handle sensor calibration, data processing, and the conversion of voltages into estimated ethanol concentrations expressed in mg/L. The most relevant data from each measurement are stored in a CSV file, allowing the consultation of a complete history of measurements at any time.

The web interface, implemented using the Flask micro-framework, provides users with a straightforward environment to perform real-time tests and visualize data. After an initial calibration of the sensor in clean air, the system includes functionalities such as user registration, alcohol measurement in exhaled air, and access to a history of previous results.

Experiments carried out with different alcoholic beverages showed consistent system responses, although with the inherent limitations of the MQ-3 sensor, which is sensitive to external factors such as temperature, humidity, and interfering vapors. While these limitations prevent the prototype from being used for medical or legal purposes, the results validate its utility as an educational and awareness tool, while also demonstrating the feasibility of integrating low-cost sensors with accessible development platforms such as the Raspberry Pi.

Finally, this work shows that the fundamental objective has been achieved: implementing a complete system that combines electronics, programming, and web development, and laying the groundwork for future improvements through more advanced sensors, stricter calibration methods, and controlled blowing mechanisms.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

MOS	Semiconductor de óxido metálico
$\propto$	Inversamente proporcional
ppm	Partes por millón
ADC	Conversor analógico digital
.csv	Comma Separated Values
SPI	Serial Peripheral Interface
SCLK	Serial Clock
MOSI	Master Out Slave In
MISO	Master In Slave Out
CS/SS	Chip Select / Slave Select
GPIO	General Purpose Input/Output
HDMI	High-Definition Multimedia Interface
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
VCC	Voltaje de corriente continua
GND	Ground
AOUT	Analogic output
DOUT	Digital output
VREF	Voltaje de referencia

1 INTRODUCCIÓN

He hecho esta carta más larga de lo usual porque no tengo tiempo para hacerla más corta.

- Blaise Pascal -

1.1 Contexto y motivación

El alcoholímetro es un instrumento que se utiliza para detectar el nivel de alcohol en aire espirado por una determinada persona, su uso es especialmente relevante en el ámbito de la seguridad vial, ya que la conducción bajo los efectos del alcohol representa una de las principales causas de accidentes de tráfico, lesiones graves y muertes en carretera. En España según una noticia publicada por la Dirección General de Tráfico (DGT) el 20 de marzo de 2025, el alcohol está presente entre el 30% y el 50% de los accidentes mortales, siendo uno de los factores de riesgo más significativos en siniestralidad vial [1].

Además, según datos de la OMS el consumo de alcohol está vinculado a más de 3 millones de muertes anuales en todo el mundo [2], lo que lo convierte en uno de los principales factores de riesgo en siniestralidad vial [3].

En la mayoría de los países europeos, existen límites legales definidos para regular así las concentraciones máximas permitidas de alcohol en aire espirado en diferentes tipos de conductores. En el caso de España, el límite para los conductores en general es de 0.25 mg/L, reduciéndose a 0.15 mg/L para conductores noveles y profesionales [1].

Tasa de alcoholemia

TIPO DE CONDUCTOR	EN SANGRE	EN AIRE ESPIRADO
Conductores en general	0,5 g/l	0,25 mg/l
Noveles y profesionales	0,3 g/l	0,15 mg/l

Figura 1. Tasas de alcoholemia de la DGT

La tasa de alcoholemia, que se corresponde con la concentración de alcohol en sangre, no es una constante universal y puede variar significativamente entre personas, incluso ante consumos similares. Estas diferencias hacen que confiar en las "sensaciones personales" sea arriesgado, especialmente en el momento en el que hay que ponerse al mando de un vehículo. La recomendación más segura, respaldada por los organismos de seguridad vial, es evitar completamente el consumo de alcohol si se tiene previsto conducir, lo que equivale a la

famosa tasa de 0,0 g/l. [1]

Una vez ingerido el alcohol, este comienza por absorberse en el aparato digestivo, comenzando la fase ascendente o de absorción. Se estima que entre un 20 % y un 25 % del alcohol se absorbe en el estómago, mientras que la mayor parte de este se incorpora al organismo a través del intestino delgado. A partir de ahí, el etanol pasa a la sangre, alcanzando su concentración máxima, conocido como fase meseta o como pico de alcoholemia, generalmente entre 30 y 90 minutos después de la ingesta, en función de varios factores fisiológicos y conductuales. Una vez alcanzado ese pico, comienza la fase de eliminación o fase descendente, en la que el nivel de alcohol en sangre disminuye progresivamente a medida que el hígado lo metaboliza. Este proceso se produce a un ritmo relativamente constante, aunque puede oscilar ligeramente en función de la actividad hepática y el estado de salud del individuo. Como referencia, una persona con una tasa de alcoholemia de 1,0 g/L puede necesitar entre 6 y 10 horas para reducirla por debajo del límite legal máximo permitido.

Hay una gran cantidad de variables que influyen sobre la tasa de alcoholemia y también en la velocidad a la que se alcanza la misma. A continuación, se muestra una tabla con la tasa de alcoholemia aproximada de las bebidas más habituales:

TIPO DE BEBIDA	CANTIDAD	HOMBRE 70-90Kg	MUJER 50-70Kg
Cerveza 330ml 5º	1 Tercio	0,21-0,28	0,34-0,48
	2 Tercios	0,43-0,55	0,68-0,95
	3 Tercios	0,64-0,83	1,02-1,43
Vino/Cava 100ml 12º	1 Vaso	0,16-0,20	0,25-0,35
	2 Vaso	0,31-0,40	0,50-0,69
	3 Vaso	0,47-0,60	0,74-1,04
Vermú 70ml 17º	1 Vaso	0,15-0,20	0,25-0,34
	2 Vaso	0,31-0,40	0,49-0,69
	3 Vaso	0,47-0,60	0,49-1,03
Licor 45ml 23º	1 Vaso	0,13-0,17	0,21-0,30
	2 Vaso	0,27-0,35	0,43-0,60
	3 Vaso	0,40-0,52	0,64-0,90
Brandi 45ml 38º	1 Vaso	0,22-0,29	0,35-0,49
	2 Vaso	0,44-0,57	0,71-0,99
	3 Vaso	0,67-0,86	1,06-1,48
Combinado 50ml 38º	1 Vaso	0,25-0,32	0,39-0,55
	2 Vaso	0,49-0,63	0,78-1,10
	3 Vaso	0,74-0,95	1,18-1,65

Figura 2. Tasas de alcoholemia de las bebidas más habituales

En los controles de alcoholemia se utiliza la determinación del alcohol en aire espirado utilizando una relación que es constante y conocida (2001/1) entre el nivel de alcohol en sangre y el nivel en aire espirado [1].

El proceso de absorción y eliminación del alcohol en el cuerpo humano sigue una evolución característica que puede representarse gráficamente mediante la denominada curva de alcoholemia (o curva de Widmark). Esta curva muestra cómo varía la concentración de alcohol en sangre en función del tiempo, desde el momento que lo ingerimos hasta su eliminación completa. Sin embargo, esta curva no es idéntica para todas las personas. Existen variables como el sexo, el peso corporal, la edad, la genética, el estado del hígado o incluso el nivel de hidratación influyen notablemente en su forma. De este modo, dos personas que consumen la misma cantidad de alcohol pueden presentar tasas de alcoholemia muy distintas.

La comprensión de la curva de Widmark resulta especialmente útil en el contexto de la seguridad vial, ya que

permite concienciar sobre cuánto tiempo puede permanecer una persona por encima del límite legal tras consumir alcohol. [4]

En la curva se pueden observar 3 fases diferenciadas, la fase ascendente, la fase de meseta (que se corresponde con el ya comentado pico de alcoholemia) y la fase descendente.

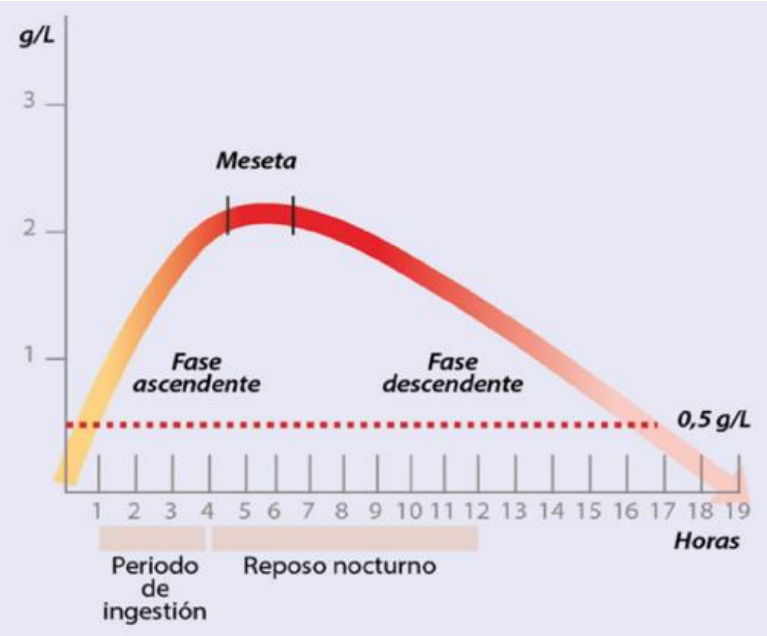

Figura 3. Curva de Widmark

1.2 Planteamiento del problema

Actualmente, los alcoholímetros homologados utilizados por las autoridades competentes se basan en tecnologías avanzadas como sensores electroquímicos o espectroscopía por infrarrojo. Debido a la importancia clave que supone la ingesta de alcohol en la seguridad vial, estos dispositivos están diseñados para ofrecer una elevada precisión, fiabilidad y cumplimiento de estándares de calibración. Sin embargo, su coste tan elevado, complejidad de fabricación y necesidad de mantenimiento periódico los hacen poco accesibles para aplicaciones de carácter didáctico o experimental.

Ante los problemas especificados en el párrafo anterior, surge la idea del desarrollo de un alcoholímetro de bajo coste, cuyo enfoque no pretende igualar la precisión ni fiabilidad de los equipos homologados, pero sí permite un acercamiento práctico, en el entorno educativo, a conceptos como son la obtención y tratamiento de señales con su posterior procesamiento, y la interpretación de esos datos en tiempo real.

1.3 Objetivo

El presente Trabajo de Fin de Grado propone el diseño e implementación de un sistema de medición de alcohol en aire espirado mediante una Raspberry Pi 4, ayudándose de un sensor de gas tipo MOS (el MQ-3) para la detección de etanol, y un conversor analógico-digital (el MCP3008) para la interpretación de señales analógicas, ya que la Raspberry Pi carece de entradas analógicas propias.

Además de la parte electrónica, para aportar valor a la simple obtención de datos desde el sensor, se le ha incorporado al sistema una interfaz web construida con el framework Flask, a través de la cual el usuario, tras una calibración previa del sensor, puede introducir su nombre, realizar una prueba de soplado cronometrada y

visualizar el resultado de forma gráfica e intuitiva, indicando los valores legales impuestos por la Dirección General de Tráfico. Por último, también se ha añadido un sistema que emula el funcionamiento de una base de datos, almacenando el resultado de cada medición en un archivo .csv y mostrándolo en cualquier momento si el usuario lo solicita.

Los objetivos específicos de este trabajo son los siguientes:

1. Diseñar e implementar el hardware de un sistema de detección de alcohol en aire espirado, integrando una Raspberry Pi 4 con un sensor de etanol MQ-3 y un conversor analógico-digital MCP3008 para la lectura de señales analógicas.
2. Desarrollar el software necesario para la lectura, procesamiento y conversión de las señales del sensor, incluyendo la calibración inicial del sistema y la interpolación logarítmica para estimar la concentración de alcohol.
3. Crear una aplicación web interactiva que permita al usuario introducir su nombre, gestionar el proceso de soplado mediante una cuenta atrás, visualizar el resultado estimado de concentración de alcohol en mg/L y comparar dicho resultado con los límites legales establecidos en España.
4. Implementar un sistema de registro de datos que almacene automáticamente la información relevante (nombre del usuario, fecha, hora, voltaje leído y estimación en mg/L) en un archivo .csv para su posterior análisis.
5. Verificar la funcionalidad y sensibilidad del prototipo mediante la realización de pruebas experimentales con diferentes tipos de alcoholes, documentando los resultados obtenidos y evaluando las limitaciones del sistema propuesto.

El punto clave de la medición reside en el sensor MQ-3, un componente electroquímico que varía su resistencia interna en presencia de vapores de etanol. Sin embargo, la Raspberry Pi, al ser una plataforma digital, carece de entradas analógicas directas para leer la señal continua que produce este sensor. Para evitar esta limitación, se ha incorporado el conversor MCP3008. Este conversor analógico-digital (ADC) actúa como intermediario entre el dominio analógico del sensor y el digital de la Raspberry Pi, traduciendo las variaciones de voltaje de salida del sensor MQ-3 en datos que la Raspberry Pi pueda interpretar. Para la correcta conexión de todo este conjunto de hardware (sensor, conversor, Raspberry Pi) se ha utilizado una placa de pruebas (breadboard) y los correspondientes cables DuPont.

La calibración inicial del sistema es un paso importante para asegurar la precisión de las mediciones. Este primer paso sirva para calcular un valor base de resistencia del sensor en aire limpio (Rs/Ro), de modo que en las siguientes mediciones el sistema calcula la resistencia del sensor (Rs) y su relación con la resistencia base. La conversión de esta relación a una concentración de alcohol se logra mediante una interpolación logarítmica basada en las curvas características proporcionadas en el datasheet del sensor MQ-3, un proceso que traduce la respuesta del sensor a una métrica comprensible.

Más allá de la adquisición de datos, el proyecto se compone de una aplicación web intuitiva utilizando el *framework* Flask en Python 3. Esta interfaz permite al usuario interactuar directamente con el sistema: introducir su nombre, seguir una cuenta atrás para soplar y visualizar en tiempo real una estimación de su concentración de alcohol en miligramos por litro (mg/L). La aplicación compara este resultado con los límites legales de alcohol en sangre establecidos en España, proporcionando una imagen visual inmediata mediante un sistema de colores (verde, amarillo o rojo) dependiendo del nivel de alcohol detectado por el sensor. Como se ha comentado anteriormente, estos resultados se registran automáticamente en un archivo .csv, el cual simula el funcionamiento de una base de datos.

Este prototipo sienta las bases para el desarrollo de prototipos de bajo coste en el ámbito del (Internet de las Cosas" (IoT), abriendo puertas a la exploración de diversas aplicaciones, promoviendo así la experimentación y el aprendizaje en el campo de la electrónica y la programación. Gracias a la combinación de componentes accesibles como la Raspberry Pi, el sensor MQ-3 y el conversor MCP3008, el sistema puede servir como base para futuras mejoras o adaptaciones, ya sea mejorando el sensor utilizado o añadiendo otros sensores adicionales que midan variables del entorno, optimizando el tratamiento de señales para mejorar así la precisión de las lecturas realizadas, o bien añadiendo funcionalidades de conectividad como la transmisión de datos a través de una red local o a una plataforma web.

Cabe destacar que este proyecto se compone de diversas competencias adquiridas a lo largo del Grado en Ingeniería de las Tecnologías de Telecomunicación como pueden ser el diseño electrónico, la programación de sistemas embebidos, el tratamiento de señales, el desarrollo web, y el análisis de datos. Asimismo, supone un caso práctico de como la ingeniería puede aplicarse a un problema real y socialmente relevante como es el impacto del consumo de alcohol frente a la seguridad vial, haciendo también un ejercicio de responsabilidad social acercando la tecnología a situaciones cotidianas que afectan directamente a las personas.

2 MARCO TEÓRICO

El diseño y la implementación de un sistema de detección de alcohol en aire espirado fundamentado en una Raspberry Pi exige una comprensión de diversos principios en los campos de la electrónica, la sensórica, el procesamiento de señales y la informática. En este capítulo se van a desglosar los principales fundamentos teóricos que han sido necesarios para llevar a cabo las decisiones de diseño adoptadas a lo largo del proyecto.

2.1 Sensores de detección de gas

La detección de gases es un campo crucial en diversas aplicaciones, desde la seguridad industrial hasta el monitoreo ambiental. Para llevar a cabo esta tarea, existen diferentes tipos de sensores, cada uno con sus propias ventajas según el tipo de gas que se quiera detectar y las condiciones en las que se trabaje. El funcionamiento básico de estos consiste en una parte sensora, dedicada a interactuar con el gas a medir, y una parte electrónica, que transforma la señal a un formato legible. A continuación, se listan los principales tipos de sensores:

- Sensores electroquímicos (EC), como por ejemplo el Alphasense CO-B4.
- Sensores ópticos (OC), como por ejemplo el MiniPID 2.
- Sensores catalíticos (CS), como por ejemplo el Figaro TGS 6812.
- Sensores de infrarrojos (IR), como por ejemplo el Senseair S8.
- Sensores semiconductores (MOS), como por ejemplo el MQ-3.

2.1.1 Fundamentos de los sensores MOS

Los sensores de óxido metálico semiconductor (MOS) se suelen utilizar por su bajo coste y su alta sensibilidad. Estos operan mediante la interacción entre la superficie activa de un material semiconductor, que suele ser normalmente dióxido de estaño (SnO_2), y los gases que se encuentren presentes en el entorno.

La sensibilidad de los sensores MOS ha sido ampliamente estudiada, identificándose factores como la temperatura de operación, la humedad relativa y la morfología superficial del material como determinantes en su rendimiento [5][6].

Estos sensores se componen de un micro calentador que mantiene la superficie a temperaturas elevadas (200–400 °C) para que se produzcan las reacciones de adsorción y las posteriores de oxidación o reducción en la superficie del material [7].

En aire limpio, las moléculas de oxígeno se adsorben sobre la superficie del SnO_2, esto provoca que se capturen los electrones libres de su banda de conducción y se aumente la resistencia eléctrica del material. En cambio, cuando un gas reductor, como el etanol, entra en contacto con la superficie, esta reacciona con el oxígeno adsorbido y libera esos electrones, disminuyendo así la resistencia del sensor. El resultado de esta variación es proporcional a la concentración del gas en el ambiente, la cual se puede estimar mediante la relación entre resistencia en presencia del gas (Rs) y resistencia en aire limpio (Ro) [8].

Esta relación, propia del sensor, se expresa como:

$$\frac{Rs}{Ro} \propto (concentración)^{-k_{GAS}}$$

donde k es una constante empírica directamente relacionada con el gas que se está midiendo.

Dicha relación, aunque a simple vista no pueda parecerlo, dibuja una curva logarítmica, ya que, si la concentración de alcohol aumenta, Rs disminuye, por lo que Rs/Ro también disminuye. Además, la caída no es

uniforme porque al principio con pequeños incrementos de concentración se provocan grandes cambios en la resistencia, pero conforme va aumentando, los cambios se van suavizando, lo que caracteriza a una curva logarítmica. Esto explica el por qué los sensores MOS son muy sensibles a bajas concentraciones, pero menos sensible a concentraciones más altas.

2.1.2 El sensor MQ-3

El sensor MQ-3, empleado en este trabajo, está optimizado para detectar vapores de etanol. Su rango de detección se sitúa entre 0.05 mg/L y 10 mg/L de alcohol en aire, cubriendo el rango habitual de interés para aplicaciones de concienciación vial o demostración educativa [9].

Figura 4. Parte frontal del sensor MQ-3

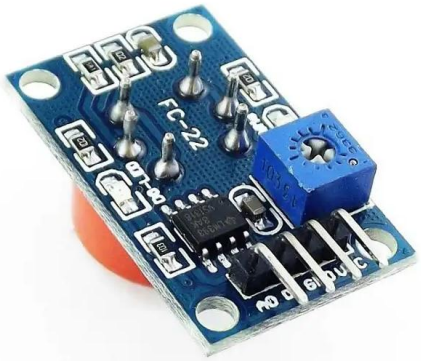

Figura 5. Parte trasera del sensor MQ-3

En la imagen de la parte frontal se pueden observar los 4 pines de los que dispone el sensor MQ-3.

- **VCC:** Pin destinado a la alimentación del dispositivo.
- **GND:** Pin destinado a la conexión a tierra.
- **AOUT:** Pin destinado a la salida analógica del sensor, su voltaje depende de la resistencia interna del sensor Rs.
- **DOUT:** Pin destinado a la salida digital del sensor, indica 0 en ausencia de alcohol y 1 en cuanto la concentración de este supera un umbral configurable por el potenciómetro situado en la parte trasera del dispositivo.

Centrándonos en la señal analógica continua (AOUT), esta representa el comportamiento de la resistencia interna del SnO2 frente a una resistencia de carga fija (RL), formando así un divisor de tensión. El voltaje de salida resultante nos sirve para interpretar la concentración del gas.

Esta señal, aunque no proporciona una precisión comparable a la de un alcoholímetro homologado, es suficientemente sensible para mostrar diferencias evidentes entre mediciones en aire limpio y en aire cargado de etanol.

Entre sus principales características se encuentran:

- Buena sensibilidad a etanol y alcohol etílico.
- Tiempo de respuesta rápido (5–15 segundos tras exposición).
- Larga vida útil en condiciones normales de uso.
- Bajo coste y amplia disponibilidad en el mercado.

No obstante, también presenta limitaciones importantes:

- Es poco selectivo: puede reaccionar ante otros compuestos orgánicos volátiles (COV) como acetona o gasolina.
- Es bastante sensible a factores ambientales como humedad o temperatura.
- Necesita un período de precalentamiento antes de ofrecer lecturas fiables.
- Su lectura requiere calibración frente a muestras conocidas para aproximar ppm o mg/L.

La salida analógica de este sensor se manifiesta como un voltaje (Vout), el cual varía en relación directa con la resistencia del propio sensor (Rs). Siendo Vcc el voltaje de alimentación, la relación entre Rs y Vout se puede describir aplicando la Ley de Ohm según la fórmula del divisor de tensión:

$$V_{out} = V_{cc} \cdot \frac{R_L}{R_L + R_s}$$

De esta expresión se puede obtener la resistencia del sensor Rs a partir del voltaje de salida medido para así determinar la concentración de alcohol mediante la relación Rs/Ro:

$$R_s = R_L \cdot \left(\frac{V_{cc} - V_{out}}{V_{out}}\right)$$

El datasheet del sensor proporciona las curvas de sensibilidad que establecen una correlación entre el cociente Rs/Ro y la concentración de alcohol, expresada en partes por millón (ppm). Estas curvas son fundamentales para el proceso de calibración y posterior conversión de la lectura del sensor en una concentración de alcohol interpretable. En esta figura se pueden observar las curvas de sensibilidad del C_2H_5OH (etanol) en color azul, el aire en color rojo, el CO (monóxido de carbono) en color verde y el H_2 (hidrógeno) en color celeste, comprobándose claramente la diferencia de reacción que provoca el etanol en el sensor con la practicamente inexistente reacción que provocan los otros gases.

Para el uso en el código python de esta curva típica representada en el datasheet del sensor, se han tomado dos puntos X(50,0.18) e Y(500,0.022). Con estos dos puntos, se consigue la ecuación de la recta, que teniendo en cuenta que nos encontramos en una escala logarítmica sería así:

$$log10(Y) = m \cdot log10(X) + b$$

Donde X representa la concentración en ppm e Y la relación Rs/R0.

La pendiente m y la ordenada b se calcularía del siguiente modo:

$$m = \frac{\log_{10}(y1) - \log_{10}(y0)}{\log_{10}(x1) - \log_{10}(x0)} \qquad b = \log_{10}(y0) - m \cdot \log_{10}(x0)$$

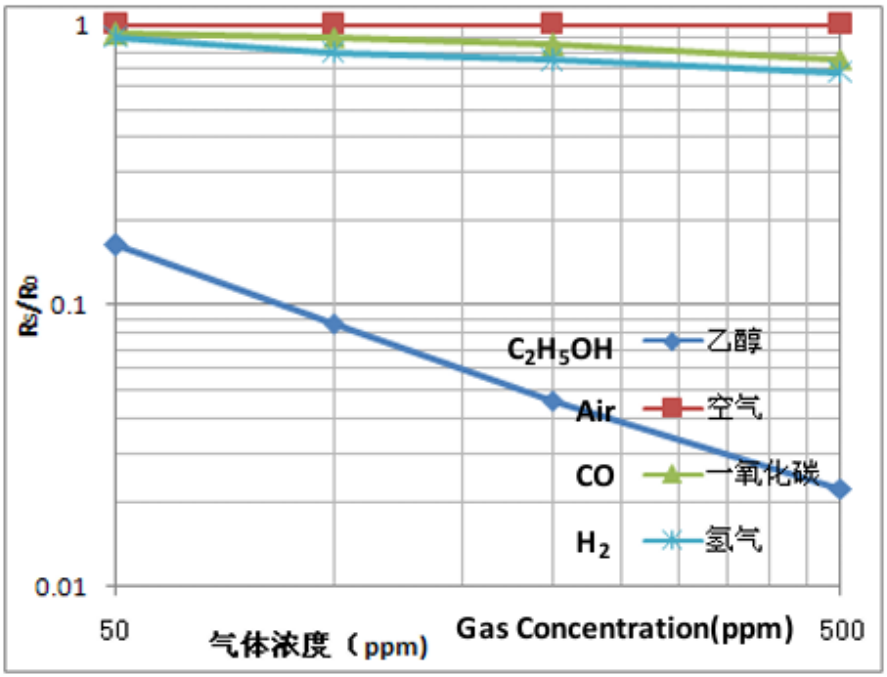

Figura 6. Curva típica de sensibilidad

Una vez obtenida la concentración en ppm, para realizar la conversión a mG/L se utiliza la ley de los gases ideales:

$$PV = nRT$$

Donde P es la presión, V el volumen, n número de moles, R es la constante de los gases ideales (0.08205746mol·KL·atm) y T es la temperatura en Kelvin (K).

De esta fórmula podemos despejar la concentración molar, que se encuentra en unidades de mol/L:

$$\frac{n}{V} = \frac{P}{RT}$$

Para convertirla en una concentración de masa (gramos/litro), la multiplicamos por el peso molecular (PM) del etanol. El peso molecular es la masa de un mol de una sustancia, expresada en gramos por mol (g/mol).

$$\text{Concentración (g/L)} = \text{Concentración(mol/L)} \cdot PM_{etanol}\text{(g/mol)}$$

Sustituyendo la expresión que obtuvimos de la ley de los gases ideales, asumiendo una presión de 1 atm y dividiendo por 1000 para obtener mg/L, la fórmula queda así:

$$\frac{mg}{L} = \frac{ppm \cdot PM_{etanol}}{R \cdot T \cdot 1000}$$

2.2 Conversión analógico-digital (ADC) y comunicación SPI

En los sistemas electrónicos modernos, especialmente aquellos basados en microcontroladores o microprocesadores como pueden ser Arduino, ESP32 o Raspberry Pi, la lectura de señales externas es una tarea muy común. Estas señales, en muchos casos, son analógicas, ya que se utilizan para representar fenómenos físicos (temperatura, luz, sonido o, en este caso, concentración de gases) mediante valores continuos en el tiempo y en amplitud. Sin embargo, las computadoras y los sistemas digitales sólo pueden interpretar datos binarios. Por este motivo, para poder trabajar con señales analógicas, es necesario transformarlas en valores digitales mediante un dispositivo llamado conversor analógico-digital, conocido por sus siglas en inglés: ADC (Analog-to-Digital Converter).[10]

A diferencia de otras plataformas como Arduino, la Raspberry Pi no dispone de entradas analógicas en sus pines GPIO (General Purpose Input/Output). Lo que quiere decir que no puede leer directamente el voltaje variable analógico proporcionado por sensores como el MQ-3 a través de su salida AOUT. [12]

Esta limitación crea la necesidad de utilizar un componente adicional que actúe como puente entre el mundo analógico del sensor MQ-3 y el mundo digital de la Raspberry Pi: el ADC.

2.2.1 El conversor MCP3008-I/P

En este trabajo de fin de grado, se ha utilizado el MCP3008-I/P como la solución a la falta de entradas analógicas en la Raspberry Pi. Se trata de un conversor analógico-digital (ADC) de 10 bits de resolución que ofrece 8 canales de entrada analógica independientes, esto permite digitalizar señales provenientes de uno o varios sensores, en este caso solo recibirá la salida AOUT del MQ-3, de forma simultánea o secuencial.

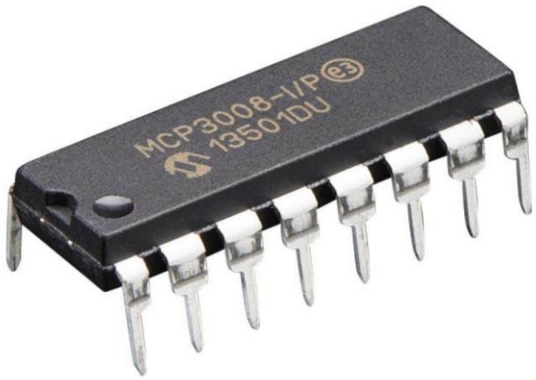

Figura 7. Conversor MCP3008 I/P

La resolución de 10 bits del MCP3008-I/P supone que puede dividir la señal analógica en 2^{10} niveles, lo que equivale a 1024 valores discretos distintos, abarcando un rango de voltajes entre 0 V y una tensión de referencia definida por el usuario (Vref). En este trabajo de fin de grado, el conversor se alimenta con 3.3 V, valor habitual para que sea compatible con el mismo voltaje de alimentación de la Raspberry Pi. Esto significa que el menor cambio que el conversor puede detectar en esta configuración, conocido como LSB (Least Significan Bit), es:

$$LSB = \frac{V_{ref}}{2^{10}} = \frac{3.3\,V}{1024} \approx 3.22mV$$

Esto proporciona una resolución suficiente para detectar pequeñas variaciones en la salida analógica del sensor MQ-3, que pueden reflejar cambios sutiles en la concentración de etanol.

Además, en términos de consumo energético, el MCP3008-I/P es un componente eficiente y de bajo consumo,

ideal para aplicaciones alimentadas por batería o que requieren una huella energética reducida. [10]

2.2.2 Comunicación SPI

La comunicación entre el MCP3008-I/P y la Raspberry Pi se realiza de forma eficiente mediante el protocolo Serial Peripheral Interface (SPI). Este es un protocolo de comunicación síncrono, full-duplex y de alta velocidad, adoptado normalmente en sistemas embebidos para la transferencia rápida de datos entre un dispositivo maestro, que en este caso sería la Raspberry Pi, y uno o varios dispositivos esclavos, en este caso el MCP3008-I/P. Fue desarrollado originalmente por Motorola y se ha convertido en una de las interfaces más empleadas en electrónica digital.

El protocolo SPI se caracteriza por utilizar cuatro líneas principales:

- **SCLK:** Es la línea de reloj, generada por el maestro (Raspberry Pi), que sincroniza todas las operaciones de transferencia de datos.
- **MOSI:** Es la línea que permite transmitir datos desde el maestro hacia el esclavo.
- **MISO:** Es la línea que permite transmitir datos desde el esclavo hacia el maestro.
- **CS/SS:** Es una línea que el maestro utiliza para seleccionar con qué esclavo específico desea comunicarse cuando hay múltiples dispositivos conectados al mismo bus.

En una comunicación SPI, por cada bit que el maestro envía a través de la línea MOSI, simultáneamente recibe un bit desde el esclavo por MISO, todo ello sincronizado por el reloj generado en SCLK. Cuando la línea CS está activa (en low mode, o estado bajo), el esclavo seleccionado responde a las instrucciones del maestro.

En este trabajo de fin de grado, la Raspberry Pi actúa como maestro SPI, enviando una secuencia de 3 bytes. De este modo se solicita la conversión del canal correspondiente (en este caso, el canal 0 del MCP3008 al que está conectado el pin AOUT del MQ-3), y recibiendo un valor de 10 bits que representa la medida analógica.

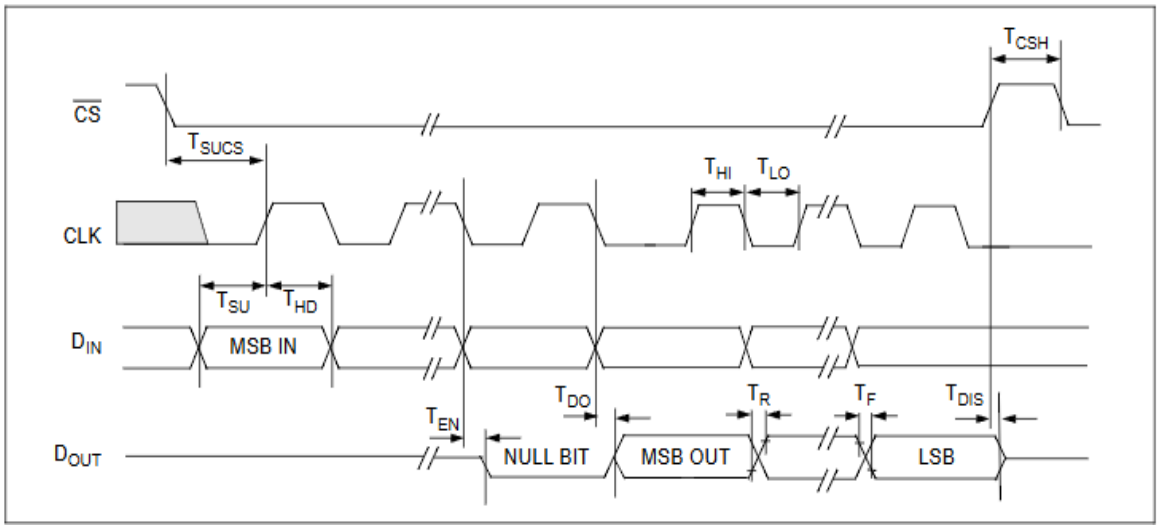

Figura 8. Diagrama de tiempos del protocolo SPI del MCP3008-I/P

En la figura 8 se puede observar el diagrama de una comunicación entre maestro y esclavo, extraída del datasheet del MCP3008-I/P.

1. El CS se pone en nivel bajo para habilitar la comunicación.
2. El CLK marca el ritmo de la transferencia
3. D_{IN} representa a la entrada de datos, desde la Raspberry Pi hacia el MCP3008-I/P se envían 3 bytes:
    - El primer byte representa un bit de inicio (00000001) indicando al MCP3008-I/P que

se va a realizar una lectura.

- El segundo byte contiene codificados el modo de lectura y el número de canal a leer.
- El tercer byte se envía vacío. Es un byte de relleno o "dummy" que mantiene el reloj SPI activo para permitir que el MCP3008-I/P pueda enviar la respuesta.

4. D_{OUT} representa a la salida de datos, desde el MCP3008-I/P hacia la Raspberry Pi responde también con 3 bytes:
   - El primer byte se envía vacío, ya que al MCP3008-I/P todavía no le ha dado tiempo a iniciar la conversión.
   - El segundo byte contiene los dos bits más significativos del resultado de 10 bits, los cuales se ubican en las posiciones de bits 0 y 1, que equivalen a los 2 bits menos significativos del conjunto del byte.
   - El tercer byte contiene los 8 bits menos significativo del resultado de 10 bits,

2.3 La Raspberry Pi

2.3.1 Hardware

La Raspberry Pi es un ordenador de placa única (SBC, por sus siglas en inglés, Single Board Computer), lo que quiere decir que sus componentes están todos integrados en una sola placa. Aunque los primeros diseños fueron de 2006, basados en el microcontrolador Atmel ATmega644, la fundación Raspberry Pi fue fundada en mayo de 2009, teniendo como objetivo la promoción de la educación de los adultos y los niños, particularmente en el campo de las computadoras, ciencias de la computación y temas relacionados sin que estos se preocuparan por dañarla [11].

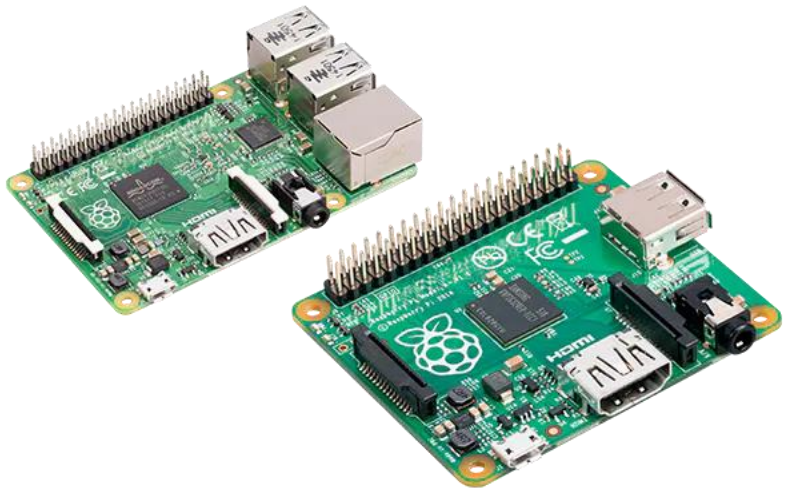

Figura 9. Raspberry Pi 1 Model A & Model B

En la figura superior se puede observar el primer modelo de Raspberry, cuyo lanzamiento comercial se produjo en el año 2012 al precio de 40€. Esta versión inicial tenía un diseño básico en comparación con sus sucesoras, carecía de puerto Ethernet, por lo que para su conexión a Internet requería de un adaptador Wi-Fi externo que se conectaba por USB. Poseía 26 conectores GPIO, una salida de vídeo vía HDMI y un conector RCA para vídeo compuesto, una salida de audio con un conector Jack de tipo 3.5 mm, un conector USB tipo A, un conector específico para cámara y su alimentación era mediante cable micro-usb con una tensión de 5V. Respecto al hardware, incluía un procesador Broadcom BCM2835 de un solo núcleo (Single-Core a 700MHz). Su memoria

RAM era de 256 MB y su gráfica era una VideoCore IV [11].

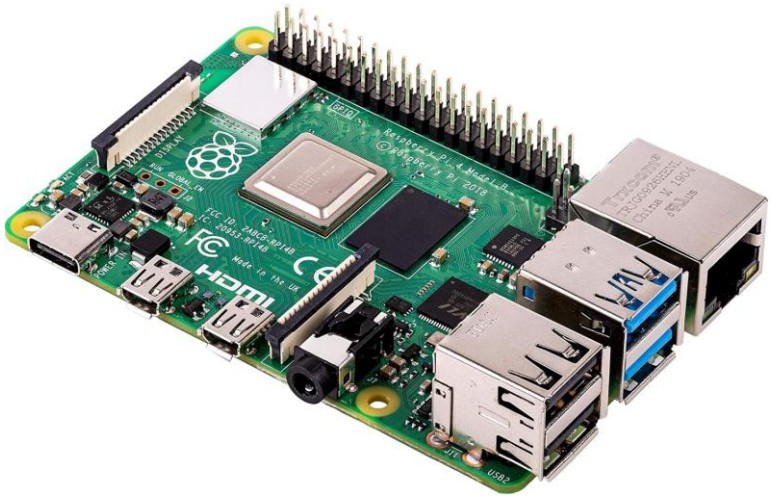

Figura 10. Raspberry Pi 4 Model B

Figura 11 . Carcasa protectora de la Raspberry Pi 4

En la figura 10 se puede observar la Raspberry Pi 4 Model B, el modelo que ha sido utilizado en este trabajo de fin de grado, la cual fue presentada en junio de 2019 desde 35€ hasta 75€. Esta edición del dispositivo supone un salto significativo tanto en rendimiento como en funcionalidades, convirtiéndola en una solución realista para aplicaciones más exigentes que vayan más allá del ámbito educativo para el que se ideó en un principio. Su arquitectura se basa en un procesador ARM de cuatro núcleos a 1.5GHz, se encuentra disponible en configuraciones de memoria RAM de 2GB. 4GB y 8GB. Respecto a la conectividad, incluye dos puertos USB 3.0 y dos USB 2.0, una entrada RJ45, Wi-Fi doble banda (de 2.4GHz y 5GHz), Bluetooth 5.0, y dos salidas de vídeo micro-HDMI con soporte para resoluciones de hasta 4K, posee 40 pines GPIO y un conector para cámaras, se alimenta mediante un puerto USB-C con una tensión de 5V, en cuanto a la gráfica, posee una VideoCore VI [11]. En la figura 11 se ha incluido una imagen de la carcasa que se incluye con la propia Raspberry Pi para proteger a esta de golpes cubriéndola por completo.

2.3.2 Software

Más allá de sus múltiples mejoras en el apartado de hardware, es fundamental destacar una característica clave como es el papel que juegan los propios usuarios del dispositivo, generando nuevas soluciones ingeniosas tanto de hardware como de software basadas en la Raspberry Pi.

Una de las mayores fortalezas de la Raspberry Pi reside en su versátil ecosistema de software. Este dispositivo no se limita a una única distribución, sino que es compatible con una amplia gama de sistemas operativos basados en Linux, adaptados específicamente para su arquitectura ARM. Esta flexibilidad de software permite que la Raspberry Pi se pueda adaptar a una amplia gama de aplicaciones y necesidades de proyecto de todo tipo.

En el ámbito de la programación, la Raspberry Pi se ha convertido en una herramienta accesible especialmente para aquellos que se inician en el desarrollo de sistemas embebidos y aplicaciones de software. Python, en particular, se ha convertido en el lenguaje de programación por excelencia para la Raspberry Pi. Su sintaxis sencilla y su curva de aprendizaje suave, combinadas con una gran oferta de librerías y módulos preexistentes (como RPi.GPIO para el control de pines GPIO, spidev para la comunicación SPI, o matplotlib para la visualización de datos, todos ellas utilizadas en este trabajo de fin de grado), simplifican enormemente la interacción con el hardware y el desarrollo de la lógica de la aplicación.

2.4 Desarrollo de aplicaciones web y gestión de datos

2.4.1 Flask: Un microframework para interfaces web

Flask es una de las opciones para tener en cuenta a la hora de la creación de aplicaciones web. Se conoce como un "microframework" ya que proporciona las características necesarias para desarrollar aplicaciones web, otorgando a los desarrolladores elegir las bibliotecas y extensiones que más se adapten a sus necesidades. A diferencia de frameworks más robustos, Flask está escrito en Python y no impone estructuras ni dependencias excesivas, lo que lo convierte en una opción ideal para proyectos de pequeña a mediana escala y para sistemas embebidos (como los basados en Raspberry Pi), donde los recursos pueden ser limitados y donde además se valoran la simplicidad y el control total sobre la aplicación.

2.4.2 Almacenamiento de datos: el formato CSV

El almacenamiento de datos de diferentes mediciones es crucial para poder analizar diferentes tendencias de los sistemas a lo largo del tiempo. Para la persistencia de estos datos, y como alternativa a las bases de datos convencionales, se puede utilizar el formato CSV (Comma-Separated Values).

El formato CSV es un formato de texto plano donde cada fila es un registro y los valores pertenecientes a cada fila están separados por un delimitador que comúnmente suele ser una coma. Esta estructura simple facilita la lectura y escritura en el apartado de la programación. Por otra parte, dado que es un formato de texto plano, el almacenamiento en CSV requiere de recursos de procesamiento y memoria en sistema muy escasos. A todo esto, se suma su gran compatibilidad con Python, el cual ofrece módulos integrados que facilitan el trabajo con este tipo de archivos.

3 IMPLEMENTACIÓN DEL SISTEMA

En este capítulo se va a desglosar las distintas fases del proceso de diseño e implementación del proyecto, desde el diseño hasta la implementación práctica del mismo. El objetivo ha sido construir un prototipo funcional y educativo que integre aspectos de electrónica, programación, desarrollo web y tratamiento de señales.

3.1 Arquitectura general del sistema

El diseño de la arquitectura del sistema se ha gestionado dividiendo el problema en 3 distintos módulos. Esto, además de facilitar el desarrollo y la depuración, también permite la escalabilidad y adición de nuevas funcionalidades en un futuro. El sistema se puede dividir a su vez en los siguientes tres subsistemas interconectados: un módulo sensor, que es el que adquiere los datos, un módulo de procesamiento y lógica de control, y un módulo de interfaz, que es el que ofrece los resultados al usuario.

3.1.1 El módulo del sensor

El módulo del sensor es el punto de entrada de la información física. Para este módulo se ha seleccionado el sensor de gas MQ-3, un dispositivo de óxido de metal semiconductor (MOS) conocido por su alta sensibilidad al etanol y su bajo coste. Este sensor, al interactuar con el vapor de alcohol, varía su resistencia eléctrica. Para capturar esta variación, se ha incorporado un conversor analógico-digital (ADC) de 10 bits, el MCP3008, que traduce la señal analógica del sensor en un valor digital que la Raspberry Pi pueda interpretar.

3.1.2 El módulo del procesamiento

El módulo del procesamiento es el cerebro del sistema. Se ha optado por la Raspberry Pi 4 por su equilibrio entre potencia de procesamiento, bajo consumo de energía y amplio soporte de la comunidad. Su función es múltiple:

- Recepción de la señal: Lee los valores digitales del MCP3008 a través de su interfaz SPI.
- Procesamiento de datos: Convierte los valores brutos del sensor en una concentración de ppm y posteriormente esto a mg/L.
- Lógica de control: Gestiona el estado del sistema, controlando la cuenta atrás para el soplado o la comparación de la medición obtenida con los umbrales legales establecidos por la DGT.
- Gestión de la base de datos: Registra automáticamente cada medición en archivos CSV incluyendo varios datos de interés.

3.1.3 El módulo de la interfaz

Finalmente, el módulo de la interfaz facilita la interacción con el usuario y la visualización de los datos. Esta interfaz cumple dos funciones principales:

- Medición en tiempo real: Permite al usuario realizar una medición de alcohol en aire espirado y obtener los resultados en el momento.
- Historial de mediciones: Ofrece acceso a un registro de mediciones anteriores, permitiendo el seguimiento de los distintos resultados a lo largo del tiempo.

3.2 Esquema eléctrico y conexiones

A continuación, en este apartado se detalla el esquema eléctrico y las conexiones específicas entre los

dispositivos.

Se van a mostrar las dos distintas configuraciones de conexión que se han debido llevar a cabo para el correcto funcionamiento del sistema, diferenciando entre dos fases del trabajo de fin de grado:

3.2.1 Configuración básica

En primer lugar, antes de interactuar con los componentes electrónicos externos como son el MQ-3 y el MCP3008M es necesario programar y depurar el código del módulo de procesamiento y la interfaz web. Para ello, es necesaria una configuración básica que se presenta a continuación:

3.2.1.1 Componentes necesarios

- Raspberry Pi 4
- Teclado
- Ratón
- Monitor
- Cables para conexiones (USB, alimentación, HDMI y RJ45)

3.2.1.2 Interconexiones realizadas

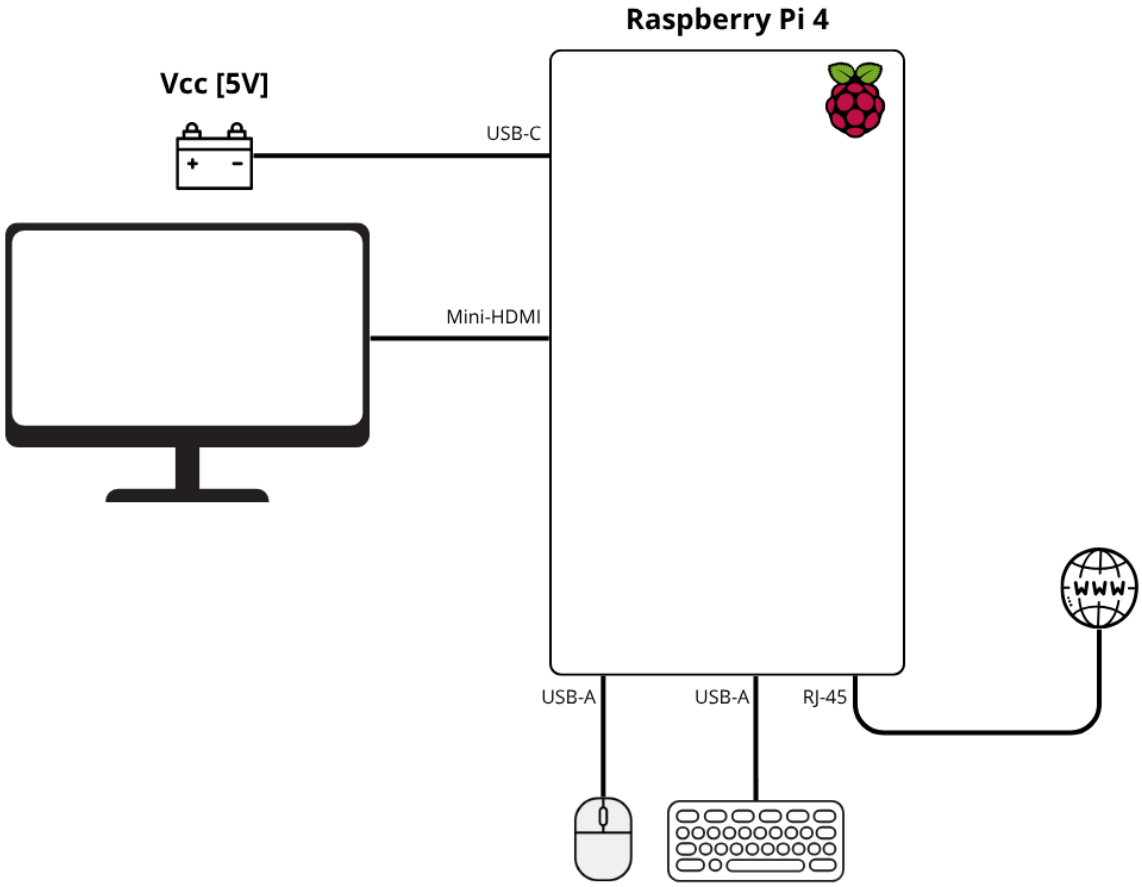

Figura 12. Configuración básica - Esquema de conexiones

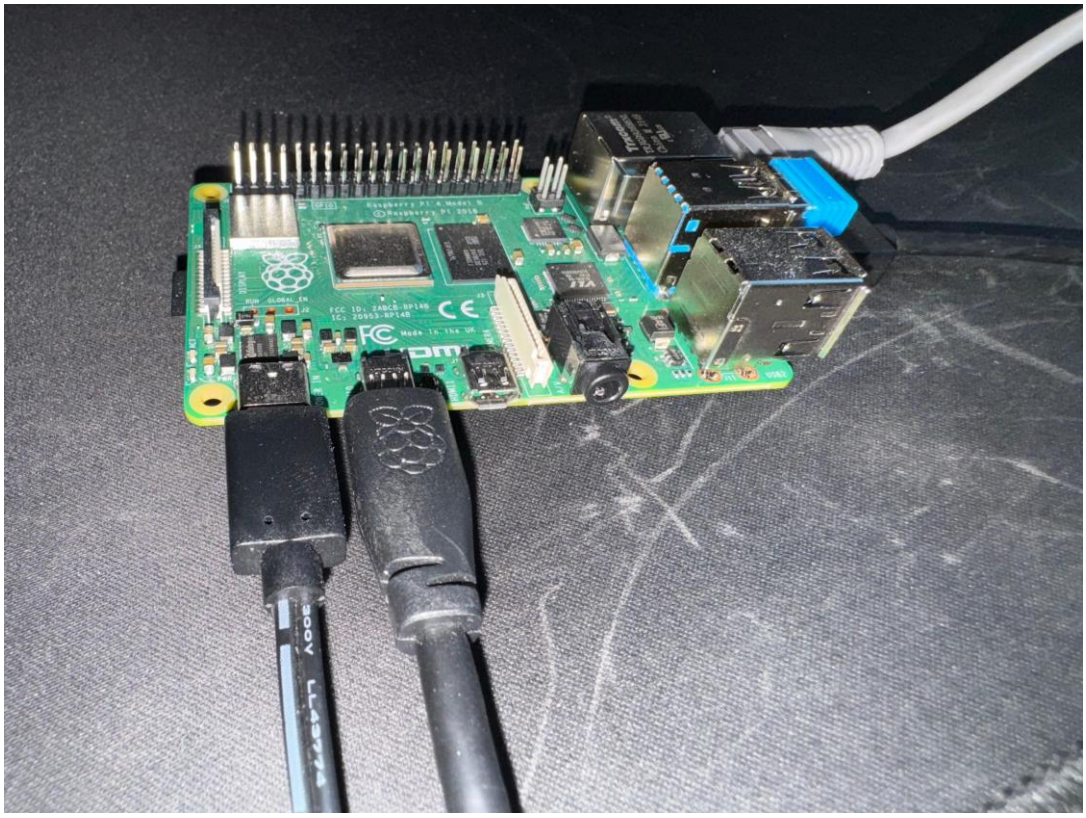

Figura 13. Configuración básica - Alimentación y HDMI

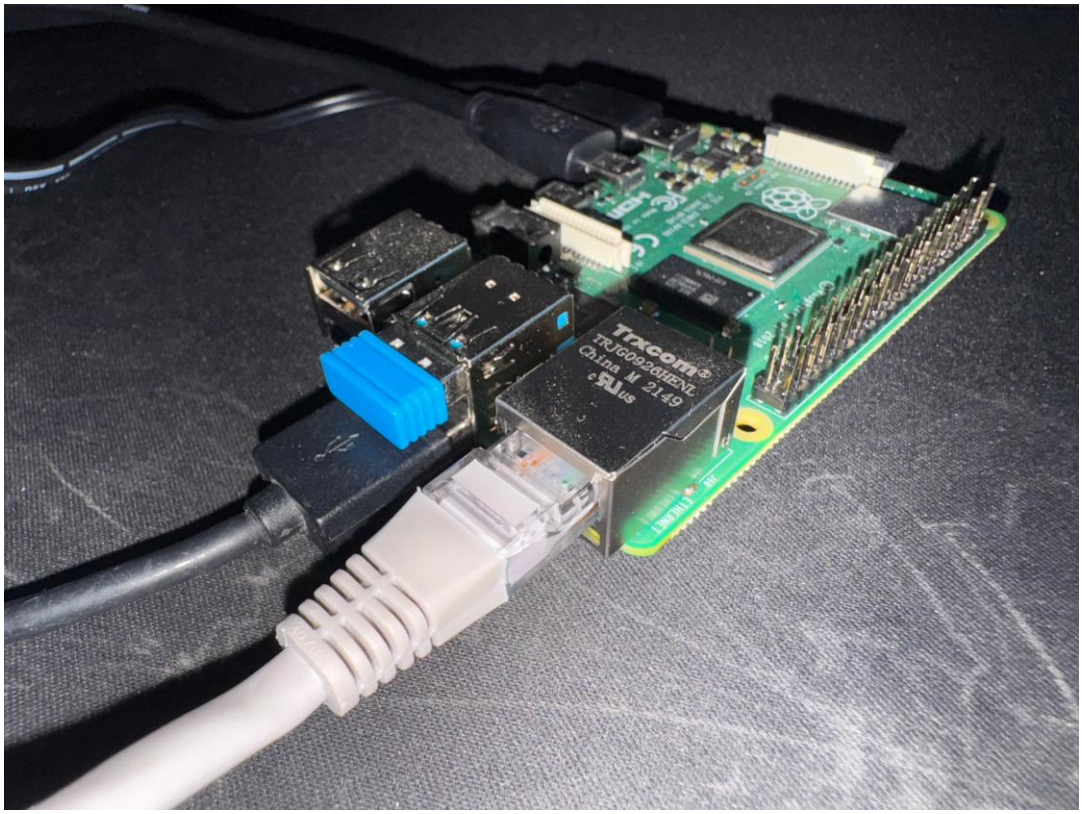

Figura 14. Configuración básica - Ratón, teclado y RJ45

Nota: La conexión con la red es necesaria en esta primera configuración para la descarga de las librerías utilizadas en el código Python.

3.2.2 Conexiones del sistema al completo

En segundo lugar, para el montaje del sistema al completo, se han realizado las siguientes interconexiones con los demás elementos.

- Interconexión con el Sensor MQ-3: El encapsulado del sensor MQ-3 tiene pines para VCC (alimentación), GND (tierra), DOUT (salida digital) y AOUT (salida analógica).
    - VCC: Se conecta al pin 2 (5V) de la Raspberry Pi.
    - GND: Se conecta al pin 6 (GND) de la Raspberry Pi.
    - AOUT: Proporciona una tensión variable en función de la cantidad de alcohol detectado en aire espirado y se conecta a la entrada analógica CH0 del conversor MCP3008.
    - DOUT: Se utilizó solo en la fase más temprana del proyecto por simplicidad, para comprobar que el sensor funcionaba y respondía frente a una concentración de alcohol. Una vez se confirmó que el sensor detectaba alcohol en aire espirado, este pin se dejó de utilizar para pasar a utilizar el AOUT, ya que la salida digital nos ofrece muy poca información (0 si no detecta alcohol o 1 si supera un umbral).
- Conexión del MCP3008: El MCP3008 se comunica con la Raspberry Pi a través del protocolo SPI, que utiliza cuatro pines para el intercambio de datos. Además, también recibe la salida analógica del sensor MQ-3.
    - VDD: Se conecta al pin 2 (5V) de la Raspberry Pi para alimentar al chip.
    - VREF: Se conecta al pin 1 (3.3V) de la Raspberry Pi para establecer la tensión de referencia para la conversión.
    - DGND: Se conecta al pin 6 (GND) de la Raspberry Pi para establecer la tierra digital.
    - AGND: Se conecta a la tierra de la breadboard para establecer la tierra analógica.
    - CLK (Clock): Se conecta al pin 23, GPIO 11 (SCLK) de la Raspberry Pi. Es el pin de reloj que sincroniza la transferencia de datos en el protocolo SPI.
    - DOUT (Data Out): Se conecta al pin 21, GPIO 9 (MISO) de la Raspberry Pi, por donde el MCP3008 envía los datos de la conversión a la Raspberry Pi.
    - DIN (Data In): Se conecta al pin 19, GPIO 10 (MOSI) de la Raspberry Pi, se utiliza por la Raspberry Pi para enviar comandos al MCP3008.
    - CS (Chip Select): Se conecta al pin 24, GPIO 8 (CE0) de la Raspberry Pi. Este permite a la Raspberry Pi seleccionar el MCP3008 para iniciar la comunicación en el protocolo SPI.
    - CH0 (Channel 0): Se conecta a la salida analógica (AOUT) del sensor MQ-3.
- Divisor resistivo a la entrada del MCP3008: Se ha optado por la implementación de un divisor resistivo con resistencias de 20kΩ a la entrada del MCP3008 para poder alimentar el sensor MQ-3 con 5V. De esta manera, el máximo valor que va a recibir el MCP3008 será 2.5V, protegiendo así la comunicación con la Raspberry Pi que tolera como máximo 3.3V en sus entradas.

A continuación, se muestran las conexiones realizadas en esta configuración:

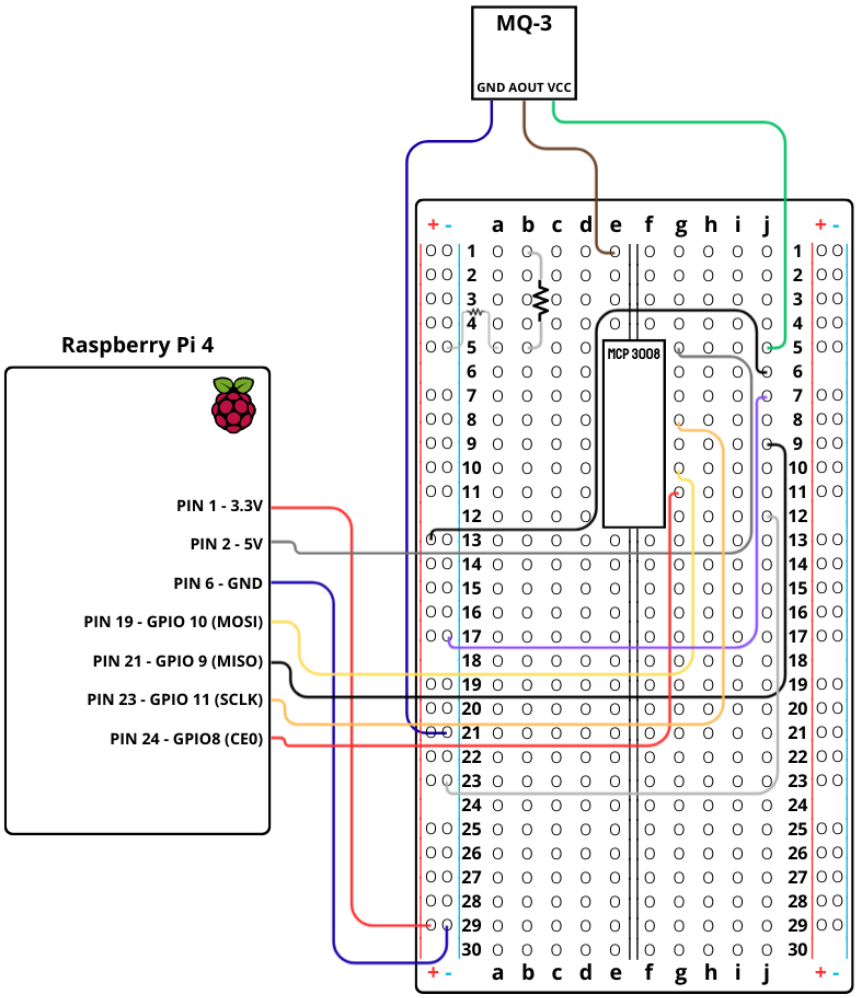

Figura 15. Configuración completa - Esquema de conexiones

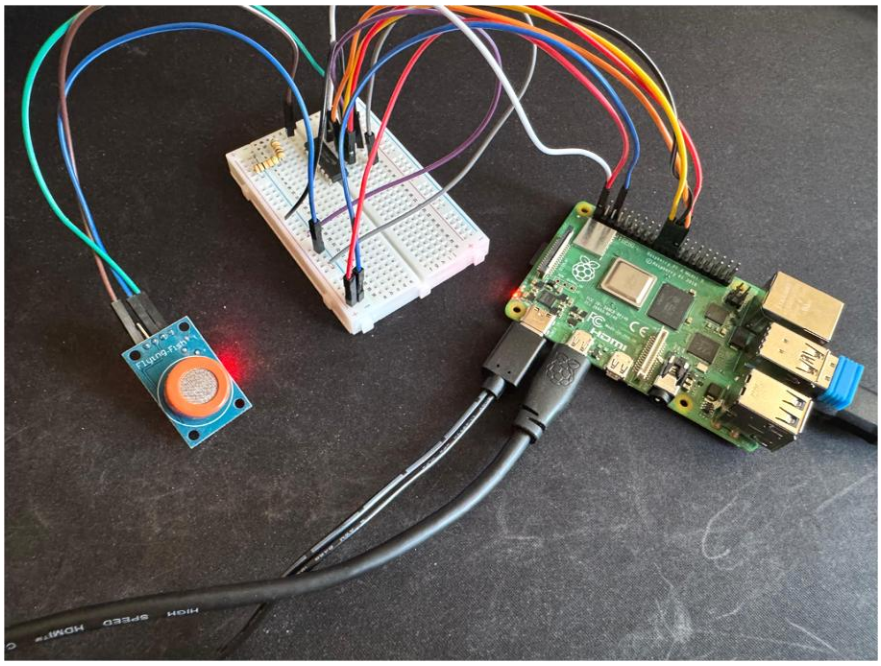

Figura 16. Configuración completa

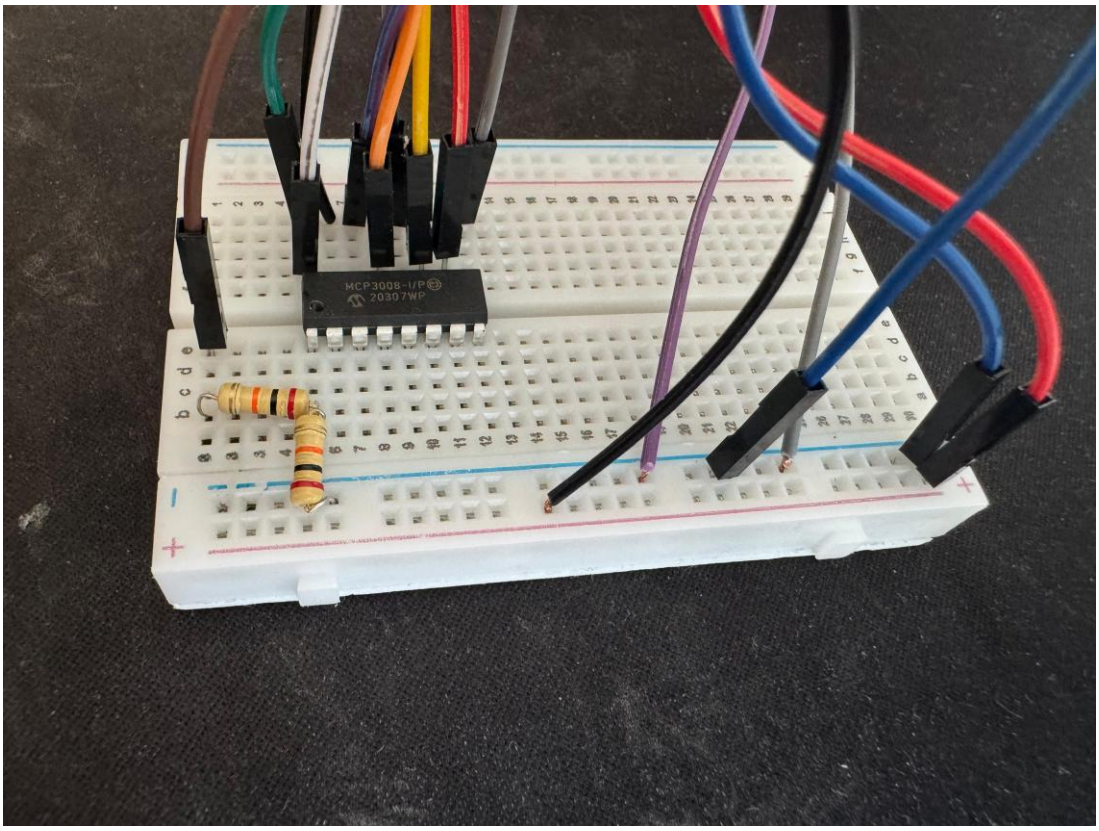

Figura 17. Configuración completa - Breadboard ampliada

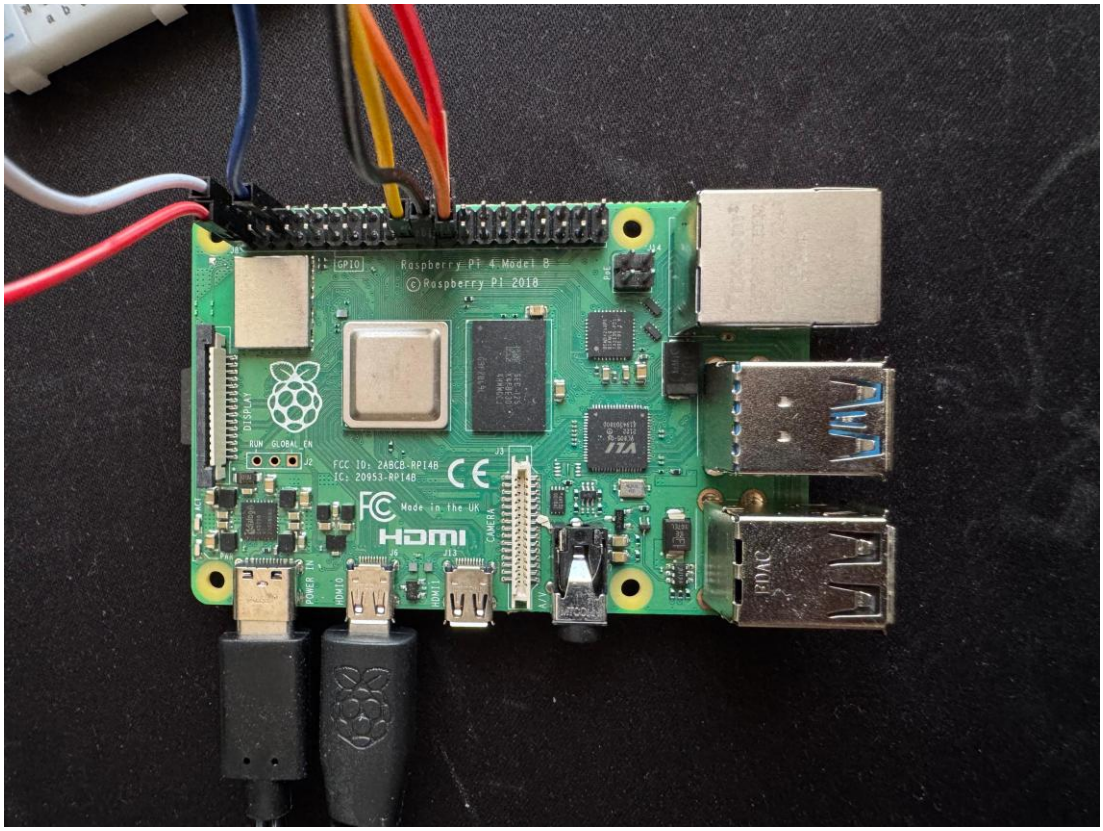

Figura 18. Configuración completa - Raspberry Pi ampliada

Nota: En el diagrama se obvian las conexiones con el ratón, el teclado e internet para simplificarlo.

3.3 Implementación del código e interfaz web

El código del proyecto se divide en dos archivos distintos llamados `calibrateTFG.py y appTFG.py`

3.3.1 calibrateTFG.py

En este archivo se encuentra la lógica relacionada con la calibración del sensor MQ-3 para la obtención del valor en ohmios de la resistencia de este en aire limpio. Este código se ejecuta mediante terminal de la siguiente manera:

```
python3 calibrateTFG.py
```

Este código permite obtener el parámetro fundamental R0, necesario para la correcta interpretación de las mediciones de alcohol en aire espirado.

El script utiliza la librería spidev para establecer la comunicación SPI con el convertidor analógico-digital MCP3008. Este ADC es el encargado de transformar la señal analógica generada por el sensor MQ-3 en valores digitales que pueden ser procesados por el sistema. La inicialización se realiza al comienzo del código, configurando el canal y la velocidad de transferencia.

Se definen constantes que determinan el número de muestras a tomar, el intervalo entre lecturas y los valores eléctricos del divisor resistivo. Estos parámetros aseguran que la calibración sea precisa y reproducible, adaptándose a las características del sensor y del circuito y representando las conexiones físicas realizadas en la breadboard.

El código implementa varias funciones que permiten:

- Leer el valor digital del ADC y, por tanto, el voltaje proporcionado por el sensor.
- Calcular el voltaje que cae en la resistencia del sensor y el correspondiente valor en ohmios de esta (Rs).
- Filtrar los valores atípicos mediante técnicas estadísticas, obteniendo así una estimación más fiel de la resistencia base (R0).

La función principal de calibración toma 50 muestras en aire limpio con un intervalo de 0.5 segundos entre muestras, calcula Rs en cada una y filtra los valores que se desvían más del 20% de la mediana. Finalmente, devuelve la media de Rs y el voltaje medio, que se emplearán como referencia en el código posterior. Al ejecutar el script, se inicia automáticamente el proceso de calibración y se muestran por consola los resultados obtenidos.

Una vez se obtiene el valor medio de R0, se guarda para utilizarlo como constante en el script de nombre `appTFG.py`

3.3.2 appTFG.py

Una vez calculado el valor de la resistencia del sensor en aire limpio, tenemos los datos necesarios para realizar el cálculo de alcohol en aire.

El archivo `appTFG.py` implementa una aplicación web basada en el framework Flask, cuyo objetivo es gestionar el proceso de medición de alcohol en aire espirado utilizando un sensor MQ-3 conectado a un convertidor analógico-digital MCP3008 mediante comunicación SPI. La aplicación permite la interacción con el usuario a través de una interfaz web, facilitando la toma de mediciones, la visualización de resultados y el almacenamiento de un historial de mediciones.

3.3.2.1 Inicialización y configuración

- Se importan las librerías necesarias para la gestión web (Flask), la comunicación con el hardware (spidev), el manejo de datos (csv, datetime) y los cálculos matemáticos (math).
- Se configura la aplicación Flask y la clave secreta para la gestión de sesiones.
- Se inicializa la comunicación SPI con el MCP3008.

- Se definen las constantes físicas y eléctricas necesarias para el funcionamiento del sensor y el circuito divisor de tensión.

3.3.2.2 Funciones de adquisición y procesamiento de datos

- Se implementan funciones para la lectura del sensor MQ-3 a través del MCP3008.
- Se desarrollan funciones para el cálculo de voltajes y resistencias en el circuito y también para la conversión de la señal eléctrica en una estimación de la concentración de alcohol (mg/L) mediante fórmulas físicas, pasando por el cálculo de ppm.
- Se incluye una función para interpretar el nivel de alcohol detectado, clasificándolo en diferentes categorías (sin alcohol, bajo, medio, alto) y asociando un color para su visualización.
- Los resultados de cada medición se almacenan en un archivo CSV, permitiendo la consulta posterior del historial.

3.3.2.3 Rutas y lógica web

- La aplicación define varias rutas que corresponden a las distintas etapas del proceso de medición:
  - Pantalla de inicio.
  - Solicitud del nombre del usuario.
  - Instrucción para soplar en el sensor.
  - Ejecución de la medición y procesamiento de resultados.
  - Visualización del resultado obtenido.
  - Consulta del historial de mediciones.
- Cada ruta está asociada a una plantilla HTML, las cuáles se almacenan en la carpeta "templates", que facilita la interacción con el usuario.

3.3.2.4 Ejecución de la aplicación

- El archivo incluye la instrucción para ejecutar el servidor Flask en modo local, permitiendo el acceso a la aplicación web desde un navegador.

3.3.3 Interfaz web

Para facilitar la interacción del usuario con el sistema, se ha desarrollado una interfaz web simple e intuitiva. Esta interfaz se ha creado utilizando HTML, CSS y JavaScript básico, y se despliega en un servidor web local basado en Flask ejecutado en la propia Raspberry Pi. La interfaz web sigue un diseño minimalista ya que el objeto de esta no es hacer grandes florituras en cuanto a la parte del frontend, si no que priorizar la construcción de una herramienta funcional, comprensible y accesible, que permita mostrar los resultados obtenidos en el sensor.

Nota: Aunque el datasheet del sensor recomienda un tiempo de precalentamiento de al menos 24 horas para alcanzar la máxima estabilidad térmica, en la práctica se ha optado por 8 horas de precalentamiento, habiéndose observado que, tras varias horas de funcionamiento continuo, el valor del voltaje base efectivamente disminuye, lo que permite al sensor una mayor precisión en las mediciones al disponer de un mayor rango de medición.

Una vez iniciada la aplicación, en la primera pantalla se puede observar el voltaje base que se ha registrado, que se obtiene de la media de todas las mediciones realizadas durante la ejecución del script `calibrateTFG.py`, y la R0, que hace referencia al valor que toma la resistencia del sensor en aire limpio, necesaria para posteriormente calcular la concentración de alcohol. Dicho valor de R0 se ha redondeado a 160000 Ω. También aparece un input para introducir el nombre del usuario que se va a someter al soplado y el botón "SOPLAR", que no se activa hasta que no se rellena dicho input. Como botón adicional, aparece el botón de "Ver historial de

mediciones", permitiéndonos la opción de acceder directamente al listado de mediciones anteriores.

Figura 19. Vista inicial previa al soplado

En cuanto se introduce el nombre y se pulsa el botón "SOPLAR", se muestra una pantalla con una cuenta regresiva de 3 segundos que nos indica que nos preparemos para soplar.

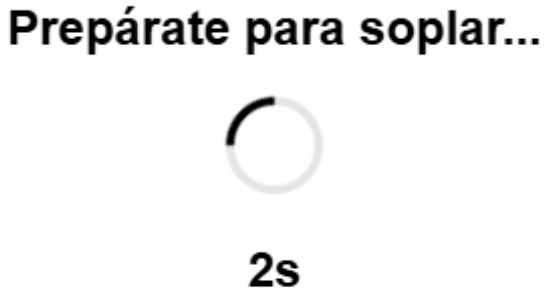

Figura 20. Vista de preparación para el soplado

Tras esta pantalla, se muestra una segunda cuenta regresiva, esta vez de 10 segundos, la cual nos indica que debemos soplar durante todo este tiempo. Internamente, el código le está indicando a la Raspberry Pi que recupere los valores de voltaje que le está enviando el sensor a razón de 100 muestras cada 0.1 segundos.

Figura 21. Vista de soplado

Una vez realizada la medición, se nos muestra una vista en la que nos devuelve el voltaje medido, que vuelve a ser la media de todos los valores recuperados en la cuenta regresiva de la vista anterior, comparado con el voltaje base registrado en la calibración inicial del sensor y acompañado de la concentración estimada en mg/L. Automáticamente cuando se realiza una medición, el programa almacena el nombre, fecha y hora, voltaje base, voltaje medido, concentración en mg/L y nivel de alcohol en un archivo .csv. Además, podemos observar una barra rellenada en función de la concentración medida. La barra dispone de dos líneas que determinan el límite legal para las personas nóveles y el límite legal general según la ley española. En esta última pantalla también se nos muestra el mismo botón de "Ver historial de mediciones".

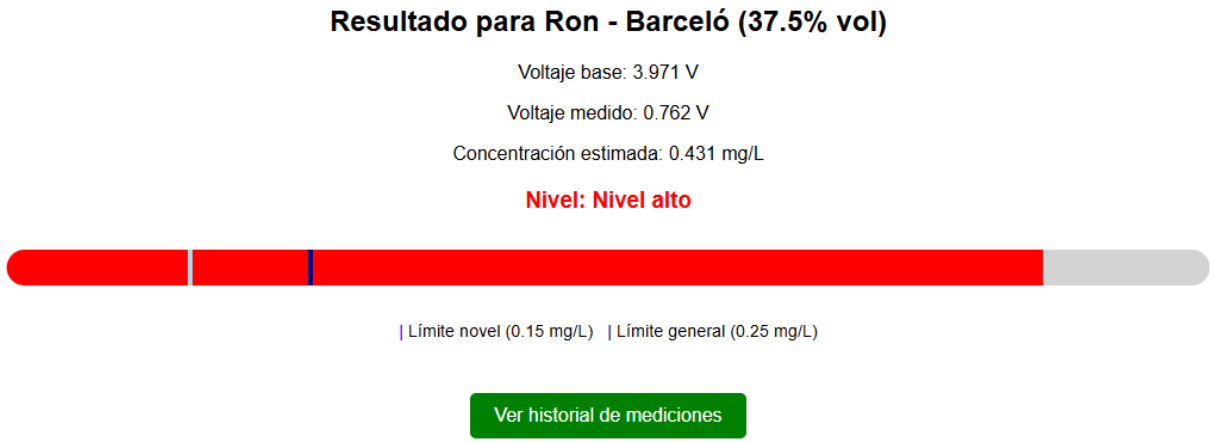

Figura 22 . Vista resultado medición

Si se pulsa el botón de "Ver historial de mediciones" en la primera o en la última pantalla, se redirigirá a una vista donde se visualiza una tabla con todo el historial de mediciones. Como los datos se almacenan en un archivo .csv, este historial de mediciones no se reinicia si el programa Python se parara por algún motivo.

Historial de Mediciones

Nombre	Fecha y hora	Voltaje base (V)	Voltaje (V)	Rs (Ω)	R0 (Ω)	Concentración (mg/L)	Nivel
Ron - Barceló (37.5% vol)	01/08/2025 20:34:32	3.971	0.762	7187.043	160000	0.431	Nivel alto
Anís - Marie Brizard (25% vol)	01/08/2025 20:39:50	3.971	1.128	11648.59	160000	0.254	Nivel medio
Licor - Barceló Cream (17% vol)	01/08/2025 20:44:53	3.971	1.127	11643.427	160000	0.254	Nivel medio
Cruzcampo - Pilsen (4.8% vol) tras 5 días abierta	01/08/2025 20:49:32	3.971	3.277	76091.825	160000	0.0	Sin alcohol
Vermouth Rojo - Gaztelu (15% vol)	01/08/2025 20:54:01	3.971	1.235	13114.024	160000	0.223	Nivel bajo
Vino tinto - Fidencio (13.5% vol)	01/08/2025 20:59:45	3.971	1.72	20981.607	160000	0.133	Nivel bajo
Ginebra - Exótica 1890 (38% vol)	01/08/2025 21:04:29	3.971	0.797	7586.154	160000	0.406	Nivel alto
Zumo de naranja exprimido	01/08/2025 21:09:33	3.971	3.94	148725.192	160000	0.0	Sin alcohol
Aire limpio	01/08/2025 21:14:09	3.971	3.954	151145.641	160000	0.0	Sin alcohol

Figura 23. Vista historial mediciones

4 RESULTADOS

Una vez finalizado el diseño, la implementación del sistema y su integración completa, tanto a nivel de hardware como de software, se procedió a llevar a cabo una fase de validación experimental. El objetivo principal de esta sección no es solamente comprobar que el sistema funcione, sino también evaluar en qué medida cumple con los objetivos definidos en el capítulo introductorio y cómo se comporta frente a distintas concentraciones de etanol.

4.1 Metodología experimental

El entorno de pruebas ha sido controlado lo máximo posible, intentando mantener una temperatura constante, evitando corrientes de aire, fuentes de calor cercanas y la presencia de otros vapores que pudieran interferir con el sensor de etanol MQ-3.

El primer paso ha consistido en asegurar un calentamiento suficiente del sensor. Como se explicó en el capítulo anterior, la calibración requiere obtener la resistencia del sensor en aire limpio (Ro), valor de referencia fundamental para las estimaciones posteriores. Esto se consigue realizando medidas en aire limpio una vez haya transcurrido un tiempo de calentamiento previo, el cual se estableció en 8 horas.

Tal y como se explicó en el capítulo anterior, en una fase muy temprana del trabajo de fin de grado, se conectó la salida digital del sensor directamente a una de las entradas de la Raspberry Pi (para ello el sensor se alimentó con 3.3V, evitando así dañar la Raspberry Pi) para comprobar el correcto funcionamiento de este. Tras acercar un tapón impregnado en alcohol al sensor se obtuvo un "1" en la Raspberry Pi, lo que significaba que el sensor detectaba correctamente el alcohol en aire.

Con esta primera prueba de aproximación hecha, se comenzó a trabajar para obtener valores provenientes de la salida analógica del sensor, para lo cual hubo que conectar antes de la Raspberry Pi el MCP3008 ya que como se ha comentado también anteriormente, la Raspberry Pi carece de entradas analógicas. Además, se protegió la entrada del MCP3008 con un divisor resistivo tal y como se ha comentado en el capítulo anterior

Con todos los elementos del sistema interconectados, se pasó a la siguiente fase de pruebas, que consistía en la validación de la salida analógica del sensor MQ-3. Para validar dicha salida se realizaron distintas pruebas, las cuales consistían en acercar un tapón impregnado en varios tipos de bebidas, que fueron:

- Cruzcampo Pilsen
- Ron Barceló
- Vino tinto Fidencio
- Ginebra Exótica 1890
- Anís Marie Brizard
- Licor Barceló Cream
- Vermouth Rojo Gaztelu
- Zumo de naranja exprimido
- Aire limpio

4.2 Observaciones en aire limpio

Antes de realizar cualquier medición en presencia de alcohol, se registraron datos de referencia en condiciones de aire completamente limpio. El sistema fue probado en una habitación sin ventilación directa de productos químicos, perfumes o vapores.

El valor de voltaje medido por el sensor MQ-3 en estas condiciones se situó alrededor de los 0.5 V, que se correspondía con unos 4V en la resistencia de este, lo que se tradujo, tras el proceso de conversión mediante el ADC MCP3008 y el uso de la interpolación logarítmica basada en el datasheet del sensor, en una concentración estimada de 0.00 mg/L. Esta lectura se mantuvo estable con ligeras oscilaciones (de unas pocas décimas de mV), lo cual es coherente con las especificaciones de sensibilidad del sensor y evidencia una buena respuesta frente a un entorno sin presencia de alcohol. Estos valores indican que el valor de RS en aire limpio se situaba en torno a los 160000**Ω**, valor que se utilizó redondeado para las mediciones posteriores de alcohol.

Estos datos se obtuvieron tras ejecutar el archivo `calibrateTFG.py`. Los resultados se exportaron a un txt y se presentan a continuación:

```
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 0.99 | Voltaje que cae en el sensor 4.01 V, por lo que RS = 161298.70 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.52 | Voltaje en B: 1.03 | Voltaje que cae en el sensor 3.97 V, por lo que RS = 153750.00 Ω
Voltaje medido en MCP3008: 0.51 | Voltaje en B: 1.03 | Voltaje que cae en el sensor 3.97 V, por lo que RS = 154968.55 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.00 | Voltaje que cae en el sensor 4.00 V, por lo que RS = 160000.00 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω
Voltaje medido en MCP3008: 0.50 | Voltaje en B: 1.01 | Voltaje que cae en el sensor 3.99 V, por lo que RS = 158717.95 Ω


[160000.00, 160000.00, 160000.00, 160000.00, 160000.00, 160000.00, 160000.00, 160000.00, 160000.00, 161298.70, 161298.70,
161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 160000.00,
161298.70, 161298.70, 160000.00, 160000.00, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 161298.70, 160000.00,
160000.00, 153750.00, 154968.55, 158717.95, 158717.95, 158717.95, 158717.95, 158717.95, 158717.95, 158717.95, 158717.95,
160000.00, 160000.00, 160000.00, 160000.00, 158717.95, 158717.95]
Media r0: 159933.545Ω | Media v en sensor: 3.997 V
```

Figura 24 . Salida del archivo calibrateTFG.py

4.3 Respuesta frente a diferentes tipos de alcohol

Una vez comprobada la estabilidad del sistema en aire limpio, se procedió a realizar pruebas reales exponiendo el sensor a vapores de diferentes bebidas alcohólicas comunes. Las mediciones se realizaron colocando los tapones de las botellas cerca del sensor, a una distancia inferior a 5 cm, manteniendo la exposición durante 10 segundos. Durante ese tiempo, el sistema adquirió muestras consecutivas y generó una media final de concentración en aire espirado.

Se han utilizado distintos tipos de bebidas las cuales fueron cerveza (Cruzcampo - Pilsen, 4.8% vol.), vino tinto (Fidencio, 13.5% vol.), vermouth rojo (Gaztelu, 15% vol.), licor destilado (Ron - Barceló, 37.5% vol.), ginebra (Exótica 1890, 38% vol.), anís (Marie Brizard, 25% vol.) y licor crema (Barceló Cream, 17% vol.). Además, se realizaron mediciones con aire limpio y zumo de naranja como control.

Los resultados obtenidos muestran una respuesta clara y creciente en función de la concentración alcohólica del líquido. El voltaje de salida del sensor MQ-3 descendía notablemente en presencia de alcohol, lo que implica una reducción en la resistencia Rs y, por tanto, un aumento en la estimación de etanol presente.

A continuación, se muestra una tabla con los resultados obtenidos:

Nombre	Fecha y hora	Voltaje base (V)	Voltaje (V)	Rs (Ω)	R0 (Ω)	Concentración (mg/L)	Nivel
Ron - Barceló (37.5% vol)	01/08/2025 20:34:32	3,997	0,762	7187,043	160000	0,431	Nivel alto
Anís - Marie Brizard (25% vol)	01/08/2025 20:39:50	3,997	1,128	11648,59	160000	0,254	Nivel medio
Licor - Barceló Cream (17% vol)	01/08/2025 20:44:53	3,997	1,127	11643,43	160000	0,254	Nivel medio
Cruzcampo - Pilsen (4.8% vol) tras 5 días abierta	01/08/2025 20:49:32	3,997	3,277	76091,83	160000	0	Sin alcohol
Vermouth Rojo - Gaztelu (15% vol)	01/08/2025 20:54:01	3,997	1,235	13114,02	160000	0,223	Nivel bajo
Vino tinto - Fidencio (13.5% vol)	01/08/2025 20:59:45	3,997	1,72	20981,61	160000	0,133	Nivel bajo
Ginebra - Exótica 1890 (38% vol)	01/08/2025 21:04:29	3,997	0,797	7586,154	160000	0,406	Nivel alto
Zumo de naranja exprimido	01/08/2025 21:09:33	3,997	3,94	148725,2	160000	0	Sin alcohol
Aire limpio	01/08/2025 21:14:09	3,997	3,954	151145,6	160000	0	Sin alcohol

Tabla 1. Resultados de pruebas experimentales

5 CONCLUSIONES Y MEJORAS

5.1 Análisis de la precisión y limitaciones

Aunque el sistema mostró ser funcional y coherente con sus respuestas, es importante señalar que existen diferentes limitaciones técnicas evidentes que afectan la precisión de las mediciones. El sensor MQ-3, como se ha detallado anteriormente, está basado en tecnología MOS, lo que implica una sensibilidad alta pero también una notable variabilidad frente a factores como temperatura, humedad o presencia de gases interferentes.

Además, al tratarse de un sensor de bajo coste, el MQ-3 no está calibrado de fábrica para lecturas legales ni médicas, lo que significa que la conversión entre Rs/Ro y concentración en mg/L se basa en aproximaciones obtenidas mediante interpolación logarítmica a partir de su hoja técnica. Esta metodología es válida en términos educativos y experimentales, pero no sustituye a un procedimiento de calibración legal con patrones certificados.

Otra de las limitaciones observadas es la ausencia de control preciso del soplado. A diferencia de un alcoholímetro profesional, que guía al usuario sobre la fuerza y duración del soplo, en este caso el proceso depende de la regularidad con la que se expongan los vapores al sensor, lo que puede introducir variabilidad en las mediciones.

Si nos fijamos en la ***Tabla 1. Resultados de pruebas experimentales*** *¡Error! No se encuentra el origen de la referencia.*, podemos observar que en el caso de la Cruzcampo - Pilsen (4.8% vol) tras 5 días abierta, en concreto, se probó con un botellín que llevaba abierto varios días, esto sumado al calor de Sevilla, la ciudad en la que se realizaron las pruebas nos hace sospechar que el alcohol de este se habría evaporado durante el tiempo que transcurrió entre que se abrió dicho botellín y cuando se hizo la medición.

También, en la misma ***Tabla 1. Resultados de pruebas experimentales***, podemos observar un caso donde a priori, la concentración de alcohol medida no tiene correlación con la graduación de alcohol, y es el caso del Licor - Barceló Cream (17% vol) y Anís - Marie Brizard (25% vol). Donde el segundo de ellos, con más graduación, reporta prácticamente los mismos valores que el primero. Esta posible diferencia se achaca al tiempo que puedan llevar abiertas ambas botellas, así como el dosificador de cada una de ellas, los cuales podrían provocar que se traspasara menos aire al tapón con el que se realizaron las medidas y por tanto, repercutiría negativamente en las mediadas realizadas por el sensor, pudiendo detectar menor cantidad de etanol en un caso u en otro.

5.2 Futuras mejoras

Por todo lo comentado en el punto anterior, se propone como futuras mejoras la utilización de un sensor de mayor calidad que permitiera la detección más exacta y sin tanta variabilidad del etanol en aire. Adicionalmente, para optimizar el diseño y la fiabilidad del sistema, se plantean las siguientes propuestas de mejora:

- **Sustitución del sensor utilizado**: El MQ-3 demuestra ser una opción adecuada para un prototipo de bajo coste y fines educativos. Sin embargo, su funcionamiento basado en óxidos metálicos lo hace bastante sensible a factores externos como la temperatura o la humedad. Esto introduce variaciones en las lecturas, lo que limita su fiabilidad en contextos donde se requiera una precisión de mayor nivel. Una primera línea de mejora consistiría en sustituir este sensor por otros de mayor calidad, como los sensores electroquímicos específicos de etanol. Este cambio permitiría obtener mediciones mucho más estables y reproducibles, con un margen de error considerablemente menor, aunque evidentemente no podemos olvidarnos de que el coste del sistema aumentaría considerablemente.
- **Implementación de un sistema de soplado controlado:** A diferencia del acercamiento manual del tapón al sensor, donde se carece de control sobre la intensidad, duración o caudal del soplo, factores que influyen directamente en la cantidad de aire y vapor de etanol que alcanzan la superficie del sensor, otra posible mejora sería incorporar una boquilla similar a la de los alcoholímetros profesionales. Para evitar

el tener que ingerir alcohol en cada prueba, esta boquilla se acoplaría a un sistema de ventilación forzada, como un pequeño ventilador que garantizaría un flujo de aire constante y a un volumen determinado conocido. Este método simularía de mejor manera las condiciones de soplado de un alcoholímetro real, reduciendo así la variabilidad de cantidad de aire medido en cada prueba.

- **Condiciones controladas:** Para eliminar las variables observadas en las pruebas, como la evaporación por el tiempo o el calor, las futuras pruebas se deberían en un entorno de laboratorio con temperatura y humedad controladas. Si se usara el mismo sensor MQ-3 y fuera posible, se debería calibrar con el tiempo de precalentamiento recomendado por el datasheet, de 24-48 horas para conseguir una menor variabilidad en las mediciones. Además, se usarían botellas de bebidas alcohólicas homologadas y precintadas que se abrirían justo antes de cada medición, garantizando así que la concentración de alcohol de cada muestra sea la declarada por el fabricante y que las condiciones de las pruebas sean idénticas para todos los casos.
- **Gestión y almacenamiento de datos:** En este proyecto, los resultados se almacenan en archivos CSV y se visualizan en una interfaz web sencilla. Si bien esto es suficiente para el prototipo inicial y suma funcionalidades al objetivo inicial de realizar una simple medición de alcohol, en el futuro podría plantearse una base de datos (por ejemplo, SQLite). Si se mejoraran las condiciones de tal manera que estuvieran controladas, se podría también almacenar no solo los resultados de las mediciones, sino también las condiciones experimentales bajo las que se llevaron a cabo (temperatura, humedad, caudal de soplado) proporcionando así datos más relevantes para un posterior análisis si fuese necesario.

5.3 Conclusión final

El sistema resultante, aunque no alcanza la exactitud de un alcoholímetro profesional, considero que consigue lograr el objetivo planteado al inicio del trabajo de fin de grado, el cual consistía desarrollar e implementar un sistema completo y funcional, integrando un sensor MQ-3 con una Raspberry Pi 4 para realizar mediciones de etanol en aire, quedando demostrada la viabilidad de un sensor económico y una plataforma de bajo coste para construir un prototipo operativo útil en el entorno educativo.

El trabajo de fin de grado no ha finalizado con la consecución del objetivo principal del mismo, sino que, gracias a los conocimientos previos sobre programación adquiridos tanto en la carrera como en la vida laboral, se le han añadido funcionalidades como una aplicación web en la que representar cada paso de la medición y un registro de datos en .csv, el cual se podía consultar en la misma aplicación.

Finalmente, se concluye que se ha conseguido no solo poner en marcha el sistema, sino también ampliar conocimientos en Python, la electrónica de sensores y la programación de módulos para la Raspberry Pi.

ANEXO A: CÓDIGO CALIBRATETFG.PY

```
# Importaciones de librerías necesarias
import spidev, time
import statistics

# Inicialización de la comunicación SPI para el ADC MCP3008
spi = spidev.SpiDev()
spi.open(0, 0)
spi.max_speed_hz = 1350000

# Parámetros del sensor MQ-3 y del divisor resistivo
MUESTRAS_CALIBRACION = 50  # Número de muestras para calibrar
INTERVALO_CALIBRACION = 0.5  # Intervalo entre muestras (segundos)
RELACION_RS_RO_AIRE_LIMPIO = 1  # Relación Rs/R0 en aire limpio según la hoja de datos del MQ-3

R1 = 20000.0  # Resistencia 1 del divisor (Ohmios)
R2 = 20000.0  # Resistencia 2 del divisor (Ohmios)

VREF_ADC = 3.3  # Voltaje de referencia del ADC (V)
VCC_SENSOR = 5.0  # Voltaje de alimentación del divisor (V)

def read_adc(channel):
    adc = spi.xfer2([1, (8 + channel) << 4, 0])
    data = ((adc[1] & 3) << 8) + adc[2]
    return data

def leer_mq3():
    """
    Lee el valor analógico del sensor MQ-3 a través del ADC MCP3008.
    Devuelve el valor digital leído (0-1023).
    """
    adc = spi.xfer2([1, (8 + 0) << 4, 0])
    data = ((adc[1] & 3) << 8) | adc[2]
    return data
```

```
def leer_media(muestras, delay):
    """
    Realiza varias lecturas del sensor y devuelve la media.
    Parámetros:
        muestras: número de lecturas a realizar
        delay: tiempo de espera entre lecturas (segundos)
    """
    total = 0
    for _ in range(muestras):
        total += leer_mq3()
        time.sleep(delay)
    return total / muestras

# Llamamos B al punto medio del divisor resistivo (entre R1 y Rs)
def calcular_vB(vA):
    return 2 * vA

def calculate_rs(vA):
    return R1 * (VCC_SENSOR - 2 * vA) / vA

# Llamamos Vsensor al voltaje que cae en la resistencia del sensor MQ-3 (Rs)
def calculate_v_sensor(adc_value):
    """
    Calcula el voltaje que cae en la resistencia del sensor MQ-3 (Rs) y el voltaje en nodo B (v_adc_5v)
    considerando el divisor con Rs (asumido o medido).
    """
    v_adc = adc_value * (VREF_ADC / 1023.0)  # Voltaje que ve el ADC (0-3.3V)
    v_adc_5v = calcular_vB(v_adc)
    v_rs = VCC_SENSOR - v_adc_5v  # Voltaje que cae en Rs
    return v_rs, v_adc, v_adc_5v



def calculate_r0():
    """
    Realiza la calibración del sensor en aire limpio para obtener R0 (resistencia base).
    Devuelve el valor de R0 calculado y el voltaje medio en Rs.
    """
```

```
    rs_readings = []
    voltajes_rs = []
    for _ in range(MUESTRAS_CALIBRACION):
        adc_value = leer_mq3()
        v_sensor,v_adc,vb = calculate_v_sensor(adc_value)
        rs = calculate_rs(v_adc)
         print(f"Voltaje medido en MCP3008: {v_adc:.2f} | Voltaje en B {vb:.2f} | Voltaje que cae en el sensor {v_sensor:.2f} V , por lo que RS = {rs:.2f} Ω")
        if rs is None or rs == 0.000:
            continue
        rs_readings.append(rs)
        voltajes_rs.append(v_sensor)
        time.sleep(INTERVALO_CALIBRACION)
    if not rs_readings:
        return None, None
    median_rs = statistics.median(rs_readings)
    threshold = 0.2 * median_rs
    filtered_indices = [i for i, x in enumerate(rs_readings) if abs(x - median_rs) <= threshold]
    # Se realiza filtrado eliminando los valores que estén fuera del 20% de la mediana
    filtered_rs = [rs_readings[i] for i in filtered_indices]
    filtered_voltajes = [voltajes_rs[i] for i in filtered_indices]

    # Imprimimos los valores filtrados
    print(rs_readings)
    if not filtered_rs:
        filtered_rs = rs_readings
        filtered_voltajes = voltajes_rs
    rs_mean = statistics.mean(filtered_rs)
    voltaje_mean = statistics.mean(filtered_voltajes)

    return rs_mean, voltaje_mean

if __name__ == "__main__":
    ro_mean,v_mean = calculate_r0()
    print(f"Media r0: {ro_mean:.3f}Ω | Media v en sensor: {v_mean:.3f}")
```

ANEXO B: CÓDIGO APPTFG.PY

```
from flask import Flask, render_template, request, redirect, url_for, session
import spidev, time
import csv
import math
from datetime import datetime

# Importaciones de librerías necesariasa

# Inicialización de la aplicación Flask
app = Flask(__name__)
app.secret_key = 'secret-key'

# Inicialización de la comunicación SPI para el ADC MCP3008
spi = spidev.SpiDev()
spi.open(0, 0)
spi.max_speed_hz = 1350000

# Parámetros del sensor MQ-3 y del divisor resistivo
RL = 20000.0  # Resistencia de carga (Ohmios)
MUESTRAS_CALIBRACION = 300  # Número de muestras para calibrar
INTERVALO_CALIBRACION = 0.1  # Intervalo entre muestras (segundos)

R1 = 20000.0  # Resistencia 1 del divisor (Ohmios)
R2 = 20000.0  # Resistencia 2 del divisor (Ohmios)

VREF_ADC = 3.3  # Voltaje de referencia del ADC (V)
VCC_SENSOR = 5.0  # Voltaje de alimentación del divisor (V)

# Constantes para la conversión de ppm a mg/L de aire espirado
R_GAS = 0.08205746  # Constante de gases (L·atm)/(mol·K)
PESO_MOLECULAR_ETANOL = 46.07  # g/mol
TEMP_C = 25  # Temperatura ambiente (°C)
```

```
X0 = 50
Y0 = 0.18
X1 = 500
Y1 = 0.022

# Calcular los puntos de la curva
PUNTO0 = (math.log10(X0), math.log10(Y0))
PUNTO1 = (math.log10(X1), math.log10(Y1))

SCOPE = (PUNTO1[1] - PUNTO0[1])/ (PUNTO1[0]-PUNTO0[0])
COORD = PUNTO0[1] -PUNTO0[0] * SCOPE

R0 = 160000
V_BASE = 3.997

# --- ESQUEMA DEL CIRCUITO MQ-3 + MCP3008 ---
# VCC: 5V
# Conexiones:
#   - VCC (5V) -> AO (sensor MQ-3)
#   - AO (internamente posee Rs que es la resistencia del sensor MQ-3) -> NodoB
#   - NodoB -> R1 (20kOhm) -> NodoA
#   - NodoA -> R2 (20kOhm) -> GND
#   - NodoA -> CH0 (entrada MCP3008)
#
# El divisor resistivo está formado por R1 y R2 (ambas de 20kOhm).
# El MCP3008 mide el voltaje en NodoA.
# Rs es la resistencia INTERNA variable del sensor MQ-3 (detecta alcohol).

# --- FUNCIONES PRINCIPALES Y AUXILIARES ---

def leer_mq3():
    """
    Lee el valor analógico del sensor MQ-3 a través del ADC MCP3008.
    Devuelve el valor digital leído (0-1023).
    """
    adc = spi.xfer2([1, (8 + 0) << 4, 0])
    adc = spi.xfer2([1, (8 + 0) << 4, 0])
    data = ((adc[1] & 3) << 8) | adc[2]
```

```
    return data


def leer_media(muestras, delay):
    """
    Realiza varias lecturas del sensor y devuelve la media.
    Parámetros:
        muestras: número de lecturas a realizar
        delay: tiempo de espera entre lecturas (segundos)
    """
    total = 0
    print(total)
    for _ in range(muestras):
        total += leer_mq3()
        print(total)
        time.sleep(delay)
    return total / muestras


def calculate_v_sensor(adc_value):
    """
    Calcula el voltaje que cae en la resistencia del sensor MQ-3 (Rs).
    El ADC mide el voltaje en el punto medio del divisor (entre R1 y Rs).
    Devuelve el voltaje en Rs (V).
    """
    v_adc = adc_value * (VREF_ADC / 1023.0)  # Voltaje que ve el ADC (0-3.3V)
    v_b = v_adc * 2  # Por el divisor resistivo 20k/20k
    v_rs = VCC_SENSOR - v_b  # Voltaje que cae en Rs
    return v_rs,v_adc


def calculate_rs(vA):
    return R1 * (VCC_SENSOR - 2 * vA) / vA


def measure(voltaje,channel=0):
    """
    Realiza la medición de alcohol:
    - Calcula Rs con el voltaje actual
    - Calcula la relación Rs/R0
    - Convierte a ppm y mg/L
    Devuelve la concentración de alcohol en mg/L.
```

```
    """
    rs = calculate_rs(voltaje)
    session['rs'] = rs
    r0 = session['r0']
    if rs<0 or rs>r0:
        rs=r0
        ratio=Y0
    else:
        ratio = rs/r0

    print(f"ratio: {ratio:.3f}")
    ppm = getConcentration(ratio)
    if ppm<X0:
        ppm=0
        mgL=0.0
    else:
        mgL = ppm_to_mgL_aire_espirado(ppm)

    return mgL

def ppm_to_mgL_aire_espirado(ppm, temp_c=TEMP_C):
    """
    Convierte ppm de etanol a mg/L de aire espirado usando constantes físicas.
    """
    T = temp_c + 273.15
    mgL = ppm * (PESO_MOLECULAR_ETANOL) / (R_GAS * T*1000)
    print(f"ppm: {ppm:.3f}")
    print(f"mgL: {mgL:.3f}")
    return mgL

def getConcentration(rs_ro_ratio):

    print(f"COORD: {COORD:.3f}")
    print(f"SCOPE: {SCOPE:.3f}")

    print(f"mathlog: {math.log10(rs_ro_ratio):.3f}")
    mathlog = math.log10(rs_ro_ratio)
    exponente = (mathlog-COORD)/SCOPE
```

```
    result=10 ** exponente
    print(f"result: {result:.3f}")
    return result


def interpretar_nivel(mgL):
    """
    Interpreta el nivel de alcohol según la concentración en mg/L.
    Devuelve una etiqueta y un color para mostrar en la interfaz.
    """
    if mgL < 0.1:
        return "Sin alcohol", "green"
    elif mgL < 0.25:
        return "Nivel bajo", "gold"
    elif mgL < 0.4:
        return "Nivel medio", "orange"
    else:
        return "Nivel alto", "red"


def guardar_datos(nombre, voltajeBase, voltaje, rs, r0, mgL, nivel):
    """
    Guarda los datos de la medición en un archivo CSV para el historial.
    """
     fila = [nombre, datetime.now().strftime('%d/%m/%Y %H:%M:%S'), round(voltajeBase, 3), round(voltaje,
3), round(rs,3) ,round(r0,3), round(mgL, 3), nivel]
    archivo = "historial_alcoholimetro.csv"
    try:
        with open(archivo, "a", newline='') as f:
            writer = csv.writer(f)
            writer.writerow(fila)
    except Exception as e:
        print("Error al guardar los datos:", e)


# --- FLUJO WEB (RUTAS DE LA APP) ---


# Ruta principal: redirige a la pantalla de calibración
@app.route('/')
def redirectInicio():
    return redirect(url_for('inicio'))
```

```
# Realiza la calibración en aire limpio y muestra los valores obtenidos
@app.route('/inicio')
def inicio():
    session['nombre'] = ""
    session['voltaje_base'] = V_BASE
    session['r0'] = R0
    return render_template('inicio.html', voltaje_base=session['voltaje_base'], r0=session['r0'])

# Recibe el nombre del usuario y pasa a la siguiente pantalla
@app.route('/pedir_nombre', methods=['POST'])
def pedir_nombre():
    nombre = request.form.get("nombre")
    session['nombre'] = nombre
    return render_template('prepararse.html')

# Pantalla para que el usuario sople en el sensor
@app.route('/soplar')
def soplar():
    return render_template('soplar.html')

@app.route('/medir')
def medir():
    valor = leer_media(MUESTRAS_CALIBRACION, INTERVALO_CALIBRACION)
    v_rs, v_adc = calculate_v_sensor(valor)
    mgL = measure(v_adc, 0)
    nivel, color = interpretar_nivel(mgL)

    guardar_datos(
        session['nombre'],
        session['voltaje_base'],
        v_rs,
        session['rs'],
        session['r0'],
        mgL,
        nivel
    )

    session['resultado'] = {
```

```
        'nombre': session['nombre'],
        'voltaje': v_rs,
        'base': session['voltaje_base'],
        'mgL': mgL,
        'nivel': nivel,
        'color': color
    }

    return {"ok": True}

@app.route('/resultado')
def resultado():
    return render_template('resultado.html', **session['resultado'])

# Muestra el historial de mediciones almacenadas en el archivo CSV
@app.route('/historial')
def historial():
    datos = []
    try:
        with open("historial_alcoholimetro.csv", newline='') as f:
            reader = csv.reader(f)
            for fila in reader:
                datos.append(fila)
    except Exception as e:
        print("Error leyendo historial:", e)
    return render_template('historial.html', registros=datos)

# Punto de entrada principal de la aplicación Flask
if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8080)
```

ANEXO C: CARPETA TEMPLATES

En esta carpeta se alojan los archivos html de los que se servirá la aplicación web para mostrar el contenido.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>TFG alcoholímetro - Inicio</title>
    <style>
        body { font-family: Arial, sans-serif; text-align: center; margin-top: 50px; }
        input { padding: 10px; font-size: 16px; }
       button { padding: 10px 20px; font-size: 18px; background: green; color: white; border: none; border-radius: 5px; cursor: pointer; }
        button:disabled { background: gray; }
    </style>
    <script>
        function validarNombre() {
            const nombre = document.getElementById('nombre').value.trim();
            document.getElementById('btn').disabled = (nombre === "");
        }
    </script>
</head>
<body>
    <h1>Bienvenido al alcoholímetro</h1>
     <h2>Voltaje base registrado en el sensor en la calibración: {{ voltaje_base | round(3)}} V y R0: {{ r0 | round(3)}} Ω</h2>
    <h3>Introduzca su nombre para comenzar con la medición</h3>
    <form action="/pedir_nombre" method="POST">
        <input type="text" id="nombre" name="nombre" placeholder="Nombre" oninput="validarNombre()">
        <br></br>
        <button id="btn" disabled>SOPLAR</button>
    </form>
    <h3>También puede consultar el historial de mediciones</h3>
    <a href="/historial">
        <button style="padding: 10px 20px; font-size: 16px;">Ver historial de mediciones</button>
    </a>
```

```
</body>
</html>
```

Código a. inicio.html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>TFG alcoholímetro - Preparación</title>
  <style>
    body { font-family: Arial, sans-serif; text-align: center; margin-top: 50px; }
    #cuenta { font-size: 3em; color: #333; }
  </style>
  <script>
    let tiempo = 3;
    function cuentaAtras() {
      if (tiempo <= 0) {
        window.location.href = "/soplar";
      } else {
        document.getElementById("cuenta").innerHTML = tiempo + 's';
        tiempo--;
        setTimeout(cuentaAtras, 1000);
      }
    }
    window.onload = cuentaAtras;
  </script>
</head>
<body>
  <h2>Prepárate para soplar...</h2>
  <div id="cuenta">3s</div>
</body>
</html>
```

Código b. prepararse.html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>TFG alcoholímetro - Realizando medición</title>
  <style>
    body { font-family: Arial, sans-serif; text-align: center; margin-top: 50px; }
    .spinner {
      margin: 30px auto;
      width: 50px;
      height: 50px;
      border: 6px solid #ccc;
      border-top: 6px solid #333;
      border-radius: 50%;
      animation: spin 1s linear infinite;
    }
    @keyframes spin {
      0% { transform: rotate(0deg); }
      100% { transform: rotate(360deg); }
    }
    #cuenta { font-size: 3em; color: #333; margin-top: 20px; }
  </style>
<script>
  let tiempo = 10;

  function cuentaAtras() {
    if (tiempo >= 0) {
      document.getElementById("cuenta").innerHTML = tiempo + 's';
      tiempo--;
      setTimeout(cuentaAtras, 1000);
    } else {
      window.location.href = "/resultado";
    }
  }

  function iniciarMedicion() {
    fetch("/medir")
      .then(r => r.json())
```

```
        .then(data => {
            console.log("Medición ejecutada:", data);
        })
        .catch(err => console.error(err));
    }

    window.onload = () => {
        iniciarMedicion(); // arranca la medición en paralelo
        cuentaAtras();     // arranca la cuenta atrás visible
    };
</script>

</head>
<body>
    <h2>Sople ahora</h2>
    <div class="spinner"></div>
    <div id="cuenta">10s</div>
</body>
</html>
```

Código c. soplar.html

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <title>TFG alcoholímetro - Resultado</title>
    <style>
        body { font-family: Arial, sans-serif; text-align: center; margin-top: 40px; }
        .barra-container {
            width: 80%;
            height: 30px;
            background: lightgray;
            margin: 30px auto;
            position: relative;
            border-radius: 15px;
        }
```

```
.limite-novel {
    position: absolute;
    left: calc(0.15 * 100%) ;
    top: 0;
    bottom: 0;
    width: 4px;
    background: lightblue;
    z-index: 2;
}
.limite-general {
    position: absolute;
    left: calc(0.25 * 100%);
    top: 0;
    bottom: 0;
    width: 4px;
    background: darkblue;
    z-index: 2;
}
.medicion {
    position: absolute;
    top: 0;
    bottom: 0;
    left: 0;
    border-radius: 15px 0 0 15px;
    z-index: 1;
}
.leyenda {
    font-size: 0.9em;
    margin-top: 10px;
}

button {
    padding: 10px 20px;
    font-size: 18px;
    background: green;
    color: white;
    border: none;
    border-radius: 5px;
```

```
        cursor: pointer;
      }
  </style>
</head>
<body>
  <h2>Resultado para {{ nombre }}</h2>
  <p>Voltaje base: {{ base | round(3)}} V</p>
  <p>Voltaje medido: {{ voltaje | round(3)}} V</p>
  <p>Concentración estimada: {{ mgL | round(3)}} mg/L</p>
  <h3 style="color: {{ color }}">Nivel: {{ nivel }}</h3>

  <div class="barra-container">
    <div class="limite-novel" title="Límite novel (0.15 mg/L)"></div>
    <div class="limite-general" title="Límite general (0.25 mg/L)"></div>
    <div class="medicion"
      style="width: calc({{ (mgL if mgL < 0.5 else 0.5) / 0.5 * 100 }}%);
          background: {{ color }};"
      title="Medición actual">
    </div>
  </div>

  <div class="leyenda">
    <p><span style="color: blue">|</span> Límite novel (0.15 mg/L)  
      <span style="color: darkblue">|</span> Límite general (0.25 mg/L)</p>
  </div>
  <a href="/historial">
        <button style="margin-top: 30px; padding: 10px 20px; font-size: 16px;">Ver historial de mediciones</button>
  </a>

</body>
</html>
```

Código d. resultado.html

```
<!DOCTYPE html>
```

```
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Historial de mediciones</title>
  <style>
    body { font-family: Arial, sans-serif; text-align: center; padding: 30px; }
    table { width: 90%; margin: auto; border-collapse: collapse; }
    th, td { border: 1px solid #ccc; padding: 10px; }
    th { background: #eee; }
    .Sin { background: #c8e6c9; }     /* Verde */
    .Bajo { background: #fff9c4; }    /* Amarillo */
    .Medio { background: #ffe0b2; }   /* Naranja */
    .Alto { background: #ffcdd2; }    /* Rojo */
  </style>
</head>
<body>
  <h1>Historial de Mediciones</h1>
  <table>
    <tr>
      <th>Nombre</th>
      <th>Fecha y hora</th>
      <th>Voltaje base (V)</th>
      <th>Voltaje (V)</th>
      <th>Rs (Ω)</th>
      <th>R0 (Ω)</th>
      <th>Concentración (mg/L)</th>
      <th>Nivel</th>
    </tr>
    {% for fila in registros %}
      {% set clase = '' %}
      {% if 'Sin' in fila[7] %}
        {% set clase = 'Sin' %}
      {% elif 'bajo' in fila[7] %}
        {% set clase = 'Bajo' %}
      {% elif 'medio' in fila[7] %}
        {% set clase = 'Medio' %}
      {% elif 'alto' in fila[7] %}
        {% set clase = 'Alto' %}
```

```
        {% endif %}
        <tr class="{{ clase }}">
          <td>{{ fila[0] }}</td>
          <td>{{ fila[1] }}</td>
          <td>{{ fila[2] }}</td>
          <td>{{ fila[3] }}</td>
          <td>{{ fila[4] }}</td>
          <td>{{ fila[5] }}</td>
          <td>{{ fila[6] }}</td>
          <td>{{ fila[7] }}</td>
        </tr>
      {% endfor %}
  </table>
</body>
</html>
```

Código e. historial.html

REFERENCIAS

[1] DIRECCIÓN GENERAL DE TRÁFICO (DGT). Consumo de alcohol [en línea]. Madrid: DGT. Disponible en: https://www.dgt.es/muevete-con-seguridad/evita-conductas-de-riesgo/consumo-de-alcohol/

[2] WORLD HEALTH ORGANIZATION (WHO), 2018. Global status report on alcohol and health 2018 [en línea]. Ginebra: WHO. ISBN 978-92-4-156563-9. Disponible en: https://www.who.int/publications/i/item/9789241565639

[3] TAVERNINI, D., ROSSI, R. y GASTALDI, M., 2018. Effect of alcohol use on driving performance: A simulator study. Accident Analysis & Prevention, vol. 115, pp. 162–169. DOI: 10.1016/j.aap.2018.03.017.

[4] FILLMORE, C.J., 1981. Pragmatics and the description of discourse. Journal of Studies on Alcohol [en línea], vol. 42, no. 1, pp. 547-565. ISSN 0096-882X. DOI: 10.15288/jsa.1981.42.547. Disponible en: https://www.jsad.com/doi/10.15288/jsa.1981.42.547

[5] LIU, X., CHENG, S., LIU, H., HU, S., ZHANG, D. y NING, H., 2012. A survey on gas sensing technology. Sensors [en línea], vol. 12, no. 7, pp. 9635–9665. DOI: 10.3390/s120709635.

[6] WANG, C., YIN, L., ZHANG, L., XIANG, D. y GAO, R., 2010. Metal oxide gas sensors: Sensitivity and influencing factors. Sensors [en línea], vol. 10, no. 3, pp. 2088–2106. DOI: 10.3390/s100302088.

[7] KOROTCENKOV, G. (ed.), 2007. Handbook of Gas Sensor Materials: Vol. 1. Conventional Approaches. New York: Springer. ISBN 978-1-4419-1220-2.

[8] SBERVEGLIERI, G. (Ed.), 1992. Gas Sensors: Principles, Operation and Developments. Springer. ISBN 978-0-7923-1433-8.

[9] MQ303B Datasheet (PDF) [en línea]. Zhengzhou Winsen Electronics Technology Co., Ltd. Disponible en: https://www.alldatasheet.es/datasheet-pdf/pdf/1307695/WINSEN/MQ303B.html

[10] MICROCHIP TECHNOLOGY INC., 2006. *MCP3008 10-Bit ADC with SPI Interface – Datasheet* [en línea]. Disponible en: https://ww1.microchip.com/downloads/en/devicedoc/21295d.pdf

[11] RASPBERRY PI FOUNDATION. Raspberry Pi – Official Website [en línea]. Disponible en: https://www.raspberrypi.org/

[12] UPTON, E. y HALFACREE, G., 2021. Raspberry Pi User Guide. 4ª ed. Chichester: Wiley. ISBN 978-1-119-23595-3.