UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

TEMA:

Implementación de sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT

AUTOR:

Vargas Castro Frank Allan

Trabajo de titulación previo a la obtención del título de Ingeniero en electrónica y automatización

TUTOR:

Ing. Quezada Calle, Edgar Raúl. M. Sc.

Guayaquil, Ecuador

20 de febrero del 2025

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

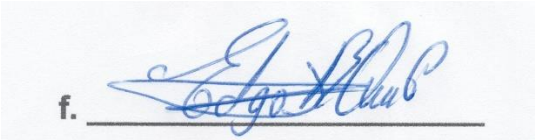
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

CERTIFICACIÓN

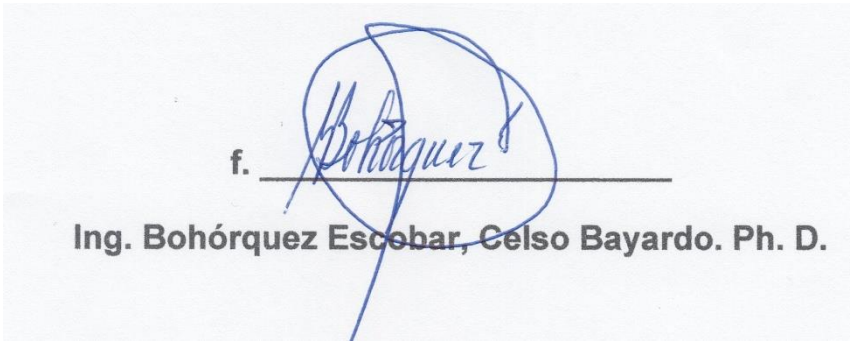
Certificamos que el presente trabajo de titulación fue realizado en su totalidad por **Vargas Castro Frank Allan**, como requerimiento para la obtención del título de **ingeniero en electrónica y automatización**.

TUTOR



Ing. Quezada Calle, Edgar Raúl. M. Sc.

DIRECTOR DE LA CARRERA



Guayaquil, a los 20 del mes de febrero del año 2025

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Vargas Castro Frank Allan**

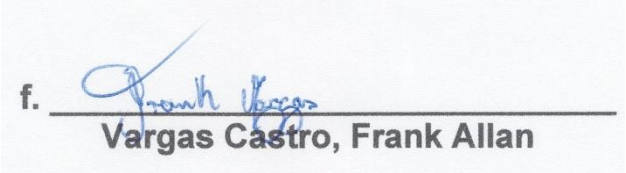
DECLARO QUE:

El Trabajo de Titulación, **Implementación de sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT** previo a la obtención del título de **ingeniero en electrónica y automatización**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 20 del mes de febrero del año 2025

EL AUTOR



UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

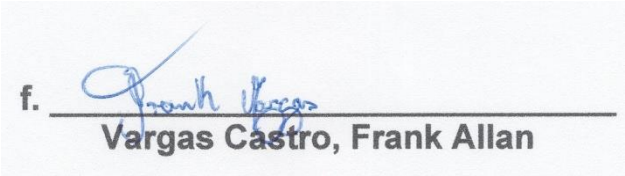
AUTORIZACIÓN

Yo, **Vargas Castro, Frank Allan**

Autorizo a la Universidad Católica de Santiago de Guayaquil a la **publicación** en la biblioteca de la institución del Trabajo de Titulación, **Implementación de sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT** cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 20 del mes de febrero del año 2025

EL AUTOR:



UNIVERSIDAD CATÓLICA

DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y AUTOMATIACIÓN

TRIBUNAL DE SUSTENTACIÓN

f. _____

ING. BOHÓRQUEZ ESCOBAR, CELSO BAYARDO, Ph. D.

DECANO O DIRECTOR DE CARRERA

f. _____

ING. UBILLA GONZÁLEZ, RICARDO XAVIER, M. Sc.

COORDINADOR DEL ÁREA O DOCENTE DE LA CARRERA

f. _____

ING. MERO VALLAS, ALEXANDER RONALD, M. Sc.

OPONENTE

REPORTE COMPILATIO

Certifico que después de revisar el documento final del trabajo de titulación "**Implementación de sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT**", presentado por el estudiante Vargas Castro Frank Allan, fue enviado al Sistema Anti-Plagio COMPILATIO, presentando un porcentaje de similitud correspondiente al 4%, por lo que se aprueba el trabajo para que continúe con el proceso de titulación.

TUTOR

f. _____

Ing. Quezada Calle, Edgar Raúl. M. Sc.

DEDICATORIA

Este trabajo de titulación está dedicado a mis padres, Juan Alberto Vargas García Y Ana Margarita Castro Sande; quienes han sido un pilar fundamental a lo largo de mi vida inculcándome valores y brindándome su amor y educación. Son quienes siempre han estado en cada etapa de mi vida apoyándome para superar cada reto, y esta no es una excepción ya que me han su apoyo incondicional ha sido base en toda mi vida académica hasta el punto de poder lograr mis estudios superiores y poder tener un futuro brillante y digno del pueda sentirme orgulloso.

Así mismo, a mis hermanos quienes también han brindado su apoyo en diferentes circunstancias de en las cuales se han presentado problemas de los cuales estoy agradecido de tenerlos.

EL AUTOR

VARGAS CASTRO, FRANK ALLAN

AGRADECIMIENTOS

Mi agradecimiento eterno a mi padre que es quien ha velado y se ha sacrificado todos los días para que yo pueda tener una educación digna, es quien ha visto que no me falta nada a lo largo de mi vida.

Agradecimiento a mi madre, quien incondicionalmente ha velado por mi bienestar y salud desde el día que me trajo al mundo.

Agradecimiento a todos los docentes y tutores que han formado parte de mi formación académica y profesional para permitirme llegar al final de esta etapa.

Agradezco a las empresas CNEL EP, SIEMAV y ENERGY CONTROL quienes me brindaron la oportunidad de adquirir conocimiento y desarrollar habilidades en calidad de pasante y poder convertirme en un profesional de la república del ecuador.

EL AUTOR

VARGAS CASTRO, FRANK ALLAN

Índice General

Índice de figuras

Índice de tablas

Resumen

El desarrollo de un sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT, utilizará el microcontrolador ESP-WROOM-32 integrando diferentes sensores de uso domótico en una tarjeta electrónica de tamaño reducido. En cuanto a su firmware este microcontrolador alojará una página web desarrollada en código HTML, CSS y Javascript; este resultado se aplica en forma de HTML embebido alojado en espacio de memoria del dispositivo, adicional a este el ESP32 captará las mediciones de los sensores para mostrar los resultados en indicadores contenidos en la página web. Con la comunicación wifi-integrada utilizará los protocolos WebServer y WebSocket para el manejo de múltiples clientes locales y comunicación en tiempo real para la visualización de la información obtenida. Debido a las propiedades de CSS la página tiene la particularidad de poseer un diseño responsive por lo que su visualización se ajusta a pantallas más pequeñas como celulares o tablets desde el navegador web. Con la implementación de este sistema de monitorización IoT se espera que sea una herramienta educativa de aplicaciones prácticas para estudiantes de las carreras técnicas de la Facultad de Educación Técnica para el Desarrollo de la UCSG puesto que el módulo electrónico propuesto permite el uso de los GPIOS no utilizados en el presente trabajo de integración.

Palabras claves: ESP32, IoT, Embebidos, HTML, CSS, Javascript, WebServer, WebSockets.

Abstract

The development of a system for reading variables with sensors using embedded systems in IoT, will use the ESP-WROOM-32 microcontroller integrating different home automation sensors in a small electronic card. Regarding its firmware, this microcontroller will host a web page developed in HTML, CSS and Javascript code; this result is applied in the form of embedded HTML hosted in the device's memory space, in addition to this, the ESP32 will capture the sensor measurements to show the results in indicators contained in the web page. With integrated wifi communication, it will use the WebServer and WebSocket protocols for the management of multiple local clients and real-time communication for the visualization of the information obtained. Due to the properties of CSS, the page has the particularity of having a responsive design, so its visualization adjusts to smaller screens such as cell phones or tablets from the web browser. With the implementation of this IoT monitoring system, it is expected to be an educational tool with practical applications for students of technical courses at the Faculty of Technical Education for Development of the UCSG, since the proposed electronic module allows the use of GPIOS not used in the present integration work.

Keywords: ESP32, IoT, Embedded, HTML, CSS, Javascript, WebServer, WebSockets.

Capítulo 1: Generalidades Del Trabajo De Titulación

1.1 Introducción

En el rubro de los sistemas embebidos o microcontrolados que se encuentran en constante crecimiento y demanda, es vital el diseño y programación de estas soluciones concretas que requieren del conocimiento necesario para su correcto desarrollo y funcionamiento por lo que las carreras técnicas de la UCSG brindan estos conocimientos a sus estudiantes.

Sin embargo, el conocimiento no es lo único que se requiere. La práctica es un punto importante para el desarrollo de las habilidades necesarias y la solidificación de las aptitudes. Este tipo de circunstancias nos demanda tener los recursos necesarios para progresar. En consecuencia, es fundamental poder tener acceso a nuevas tecnologías y herramientas de trabajo para poder estar en constante mejora, desarrollo y aprendizaje.

1.2 Definición del problema

El creciente interés de los estudiantes de las carreras técnicas de la Facultad de Educación Técnica de la Universidad Católica de Santiago de Guayaquil, especialmente en el aprendizaje de nuevas tecnologías y sistemas, ha generado un desafío para la facultad, el mismo radica en la falta de instrumentación idónea en los laboratorios de dichas carreras, para realizar prácticas con sensores en las cátedras de sistemas microcontrolados, programación y lenguaje de programación avanzado lo cual limita la capacidad de la institución para satisfacer plenamente el entusiasmo y las necesidades educativas de los estudiantes.

1.3 Justificación del problema

La creación de un módulo educativo basado en el microcontrolador ESP32 permitirá al alumnado el uso de una herramienta de uso práctico mejorando su conocimiento en sensores. Además, con el uso de esta, podrán mejorar habilidades y lógica de programación de sistemas embebidos haciendo uso de adquisición y visualización de datos de sensores.

1.4 Objetivos del Problema de Investigación

1.4.1 Objetivo general

Crear un módulo educativo en esp32 para la implementación de prácticas y visualización de las propiedades de sensores.

1.4.2 Objetivos específicos

1. Diseñar un sistema embebido con esp32, que integre múltiples sensores.
2. Crear un módulo educativo de múltiples sensores aplicados a la enseñanza.

1.5 Antecedentes

A lo largo de los años el registro de la información ha sido vital para la toma de decisiones, comprender sucesos y realizar conclusiones. Si bien en esto se realizaba con la observación de los fenómenos y variables siendo en su totalidad una persona o varias personas los principales actores, esto en la actualidad ha cambiado debido a la evolución de los sistemas, siendo los sensores y microcontroladores los principales encargados de medir variables del entorno y manipular la información obtenida, de manera respectiva.

Así los sistemas electrónicos desempeñan un papel muy importante en la adquisición, visualización y almacenamiento de datos para la detección de problemas, mejoramiento procesos, solución a errores, ejecución de respuestas ante determinados resultados.

Asimismo, el procesamiento de la información va ligado de los microcontroladores por lo que es vital saber cómo programarlos puesto que esta son las instrucciones que recibe el cerebro del sistema para su correcto funcionamiento con los diversos tipos de sensores ya que es este quien realiza ejecuciones con las variables medidas.

1.6 Metodología de la investigación

El desarrollo del proyecto posee como objetivo primario la implementación una herramienta educativa para realizar prácticas empleando una metodología con un enfoque experimental, este no tiene la necesidad y de un análisis poblacional como objeto de logro sino más bien en brindar una

solución tangible al problema para cumplir con las necesidades educativas de los estudiantes en los laboratorios de las carreras técnicas de la FETD.

La metodología implementada tiene una alineación paralela al proyecto, donde la respuesta al problema es vital y la validez de esta está ligada a su uso. La validez es definida por los resultados obtenidos a través de la implementación práctica.

1.7 Hipótesis

La implementación del módulo educativo permitirá depurar el nivel de conocimientos y mejora de habilidades de programación con las nuevas tecnologías ESP32 y microcontroladores.

Capítulo 2: Marco Teórico

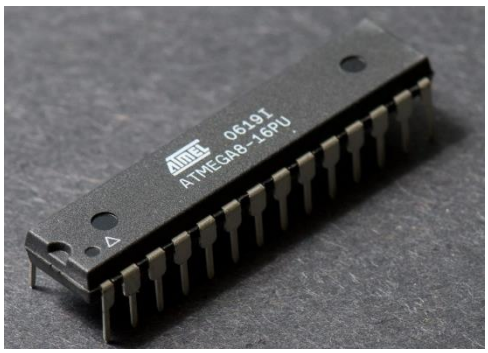
2.1 Microcontroladores

Es un dispositivo electrónico con la capacidad de ser programado y ejecutar instrucciones alojadas en la memoria; su composición consta de varios bloques funcionales que realizan determinadas operaciones. Las principales partes de un microcontrolador son similares a las partes de una computadora, puesto que estas son:

- Unidad de procesamiento.
- Memoria.
- Periféricos de entrada/Salida.

Figura 1

Microcontrolador ATmega8 en formado DIP TTH.



Fuente: (Perera, 2020).

El uso de los microcontroladores depende de la aplicación en la que se desea utilizar, esto debido al rendimiento y poder computacional intrínseco para realizar tareas. Cabe resaltar que ante elevados requerimientos los microcontroladores tienden a ser considerados como DPS (procesadores de señal digital) lo que a su vez necesita grandes velocidades de reloj y consumo energético elevado (Lima & Edmundo, 2024).

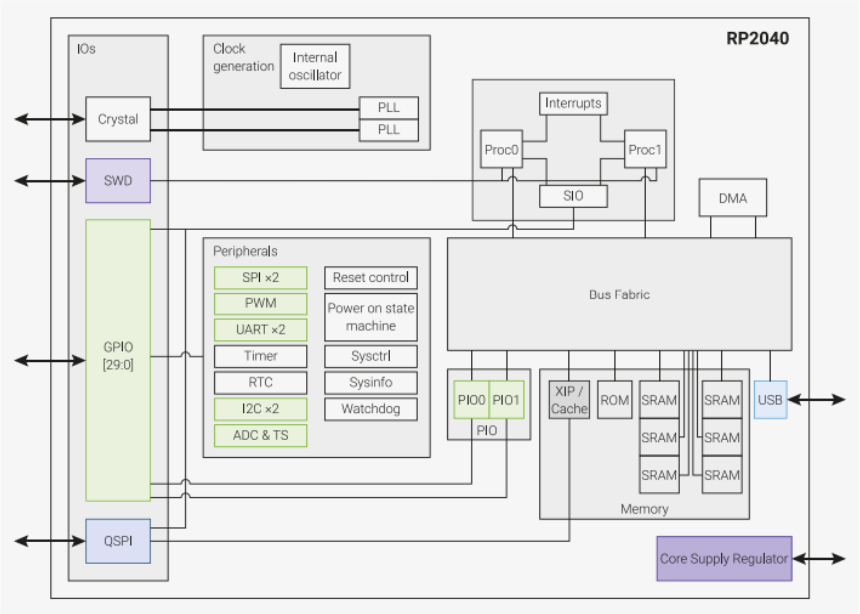
Toda la programación es contenida dentro del espacio de memoria llamada EEPROM la cual es borrable eléctricamente, siendo esta programación personalizable para cada caso de aplicación que se desee sin tener la necesidad de cargar el programa cada vez que se energiza el dispositivo (*¿Qué es un microcontrolador?*, 2024).

2.2 Partes de un microcontrolador

Las principales partes de un microcontrolador son:

Figura 2

Estructura interna de un microcontrolador contemporáneo.



Fuente: (Silicon - Raspberry Pi Documentation, 2024).

2.3 Registros

Direcciones de memorias capaces de almacenar datos para futuros procesos llevados a cabo en sectores diferentes del microcontrolador. También tienen uso en el guardado de información procedente de una operación, instrucción, datos exportados de una o varias memorias externas o de datos ajenos al sector de guardado. El tamaño de los registros es definido por el tamaño de la información que fue procesada previamente, dentro de los tamaños más comunes tenemos desde 4 bits hasta los 64 bits de información. El dimensionamiento de la información es decir el tamaño del registro es vital ya que con este se infiere una característica más ligada al potencial del microcontrolador (Xu & Peng, 2023).

2.3.1 Unidad de control

Se encarga de procesar las instrucciones y de coordinar el tráfico de datos en el interior de la CPU y otros elementos. Esta parte de la principal, puesto que se la considera como el cerebro del microcontrolador y es el

portador de toda la lógica requerida para decodificar pues osee elementos que convierten estas instrucciones a un tipo de información que es extraída de las unidades de almacenamiento de información y las convierte en micro operaciones procesables y ejecutables para CPU. Además, direcciona componentes de la cpu para el procesamiento de operaciones necesarias (*¿Qué es una CPU?*, 2024).

2.3.2 Unidad aritmético-lógica (ALU)

Es el sitio encargado de llevar las operaciones matemáticas y algebra booleana teniendo como principal tarea estas determinadas labores con grado de dependencia a otras actividades y sectores. De las operaciones realizada aquí se encuentras las más básicas: suma y resta; además cálculos con booleanos. Hoy en día, los dispositivos conocidos como microcontroladores son capaces de tener en un espacio reducido muchas ALU's con el fin de obtener un mador poder de operación para realizar procesos complejos como tareas con números de coma flotante. Esto se traduce a que la potencia del microcontrolador está ligada a la población de unidades de aritméticas lógicas que posea debido a que en esta radica la capacidad de un ampliar el abanico de tareas y operaciones en el menor tiempo de ejecución y resolución (Advances in Arithmetic Logic Units, 2024).

2.3.3 Buses

Son los canales destinados para la comunicación entre los diferentes elementos del procesador (periféricos, memorias, cpu, etc.), los cuales también realizan transferencia de información entre sí. Su utilización radica en entradas y salidas de uso general y periféricos (*Arquitectura de computadoras*, 2023).

Según la ficha 3 de microprocesadores y microcontroladores del INET de Argentina (Filmus et al., s. f.), entre los tipos de buses tenemos:

- **Dirección:** Sirve para la selección de memoria o dispositivo con el cual se trabajará, un uso frecuente es la selección de datos en memoria.

- **Datos:** Se usa para el cambio de información del microprocesador con la memoria o entradas y salidas seleccionadas en los buses de memoria.
- **Control:** Útiles para la gestión de procesos de lectura y escritura, así como para el control de operaciones.

2.3.4 Periféricos

Son circuitos de tipo digital los cuales brindan la capacidad de poder trabajar con señales externas. Entre los tipos de periféricos tenemos: entradas y salidas, temporizadores y contadores y conversores ADC y DAC, puertos de comunicación, moduladores PWM.

- **Entradas y salidas:** Tienen la capacidad individual de trabajar como canal receptor o emisor de información o señales para trabajar con elementos externos. Las entradas digitales leen la información de estado alto y bajo, es decir, 1 o 0 siendo 5V y 0V para los respectivos valores lógicos. De igual manera las salidas digitales operan con los mismos valores lógicos de 1 y 0 y los mismos valores de voltaje que las entradas. Por otro lado, las entradas y salidas analógicas operan con valores variables que van dentro de un rango de 0V a 5V haciendo referencia a valores variables del mundo real que van también dentro un rango, como la velocidad de un motor o la luminosidad de un foco (Instituto Superior Universitario Sucre et al., 2024).
- **Temporizadores y contador:** Son elementos utilizados para llevar cuenta del tiempo y eventos respectivamente (¿Qué es y cómo funciona un microcontrolador?, 2024).
- **Conversores Analógico Digital:** Circuitos encargados de la conversión de señales tanto analógicas como digitales siendo se principal utilidad en aplicaciones con sensores y manipulación de actuadores (¿Qué es y cómo funciona un microcontrolador?, 2024).

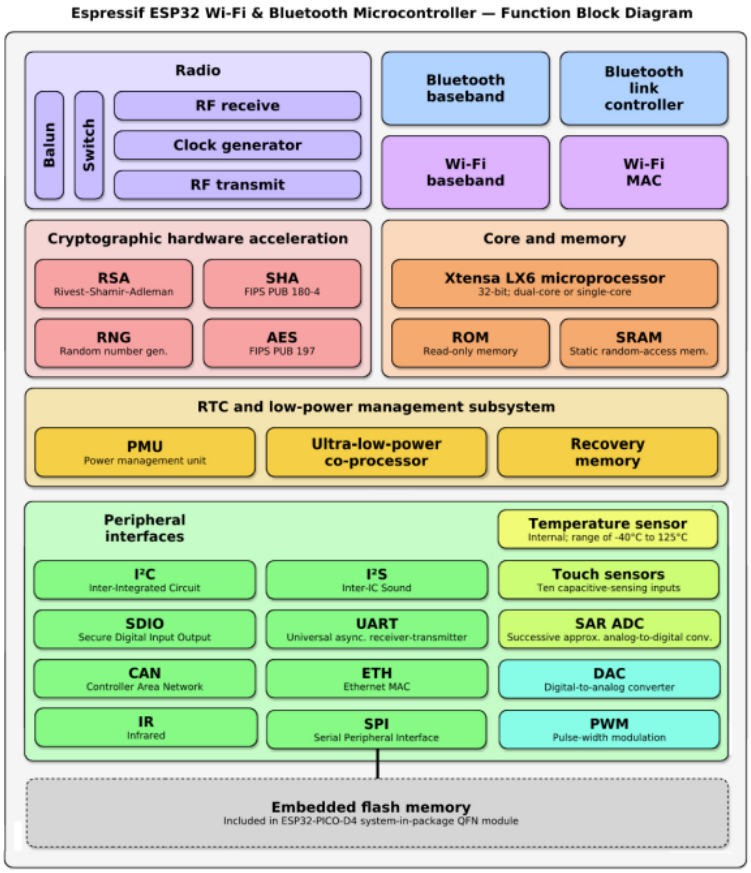
- **Puerto de comunicación:** Puertos de transmisión y recepción de información con el microcontrolador, para estos los protocolos más comunes son: Puerto serie, SPI, I2C, USB y ethernet (IEEE, 2020).
- **Modulador de ancho de pulso (PWM):** tiene aplicación para modificar la cantidad de corriente que se suministra a un actuador como un motor por ejemplo variando el ancho relativo de cada pulso acorde a determinada frecuencia (Instituto Superior Universitario Sucre et al., 2024).

2.4 ESP-WROOM-32

Es común en su versión de sistema de procesamiento integrado en placa conocido también como SoM (System-on-Modules) Este microcontrolador contiene dos núcleos de procesador (dual-core) a 32 bits y seis bloques de memoria. El procesador es el Xtensa LX6 dual-core o single-core con su facilidad de realizar dos tareas a la vez. Considerados como dispositivos de bajo consumo energético con capacidad de deep sleep, además de su fácil adquisición por su bajo precio (Sanz Martín, 2022).

Figura 3

Diagrama de bloques acerca de la estructura del microcontrolador.



Fuente: (Sanz Martín, 2022).

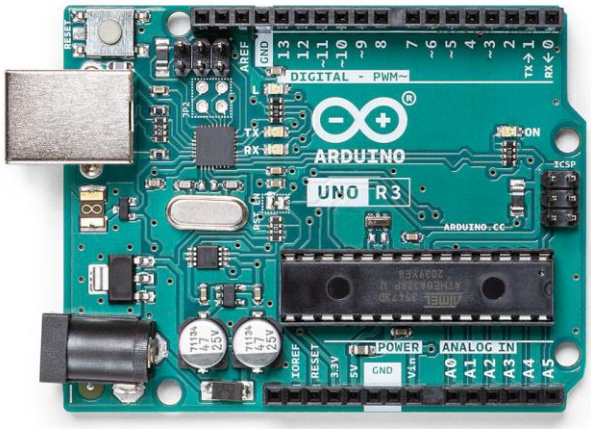
Contiene comunicaciones inalámbricas que integra diferentes sistemas amplificadores como de potencia, receptores de bajo ruido, filtros y módulos de administración de energía para un mejor aprovechamiento energético durante su uso. Por otro lado, existen muchas versiones de la placa ESP32 siendo ESP-WROOM-32 la versión más común, teniendo también las versiones, ESP32-CAM, ESP32-DEVKITC, ESP32-WROVER, etc. Las placas hacen uso de un mismo modelo base, el cual es el microprocesador, siendo las periferias la principal diferencia entre un modelo y otro por lo que el uso y propósito de cada modelo radica en estas (Casellas, 2020).

Con respecto a las comunicaciones, el ESP32 integra de conectividad Wi-Fi y Bluetooth y también posee la disponibilidad utilizar protocolos como UART, I2C o SPI para el trabajo con señales y digitales a través de los diferentes GPIOS de la placa (Sampedro et al., 2022).

2.5 Arduino UNO R3

Figura 4

Arduino Uno Rev3



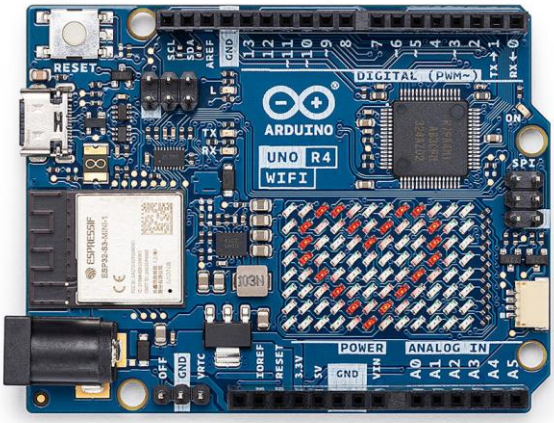
Fuente: (Arduino Uno Rev3, s. f.).

Es una tarjeta basada en el microcontrolador ATmega328P. Pone a disposición una cantidad de 14 pines digitales de entrada o salida de los cuales 6 pueden ser utilizados para señales PWM (Pulse width modulation), 6 entradas analógicas. Utiliza un resonador cerámico de 16MHz. Posee conexión usb para alimentación y programación, además de otros conectores y pines para alimentación. Su voltaje de operación es de 5V y su alimentación va de 7V a 12V siendo este último es más recomendado para mejor rendimiento. Sus GPIO pueden consumir hasta 20mA o 50mA a 3V. Posee interfaces I2C, SPI que con útiles para la comunicación con sensores y otros periféricas adicionales. Puede ser programada en lenguaje C++ en el IDE oficial de Arduino (*Arduino Uno Rev3*, s. f.).

2.6 Arduino UNO R4 WiFi

Figura 5

Arduino Uno R4 Wifi



Fuente: (Gordillo & Michel, 2024).

Es parte de la nueva generación de tarjetas electrónicas de Arduino que integra el microprocesador R4AM1 de Renesas fusionado con el microcontrolador ESP32-S3, teniendo como resultado una potente herramienta todo en uno. Dentro de sus comunicaciones dispone de conectividad Wi-Fi y bluetooth. Integra led colocados en orden matricial de dimensión 12x8 y un conector de tipo Qwiic para la integración de sensores y módulos I2C para fácil manipulación. La placa integra más de una interfaz de comunicación (UART, I2C, SPI, CAN), un regulador de voltaje, protecciones y cristal oscilador. Puede ser programado en lenguaje C/C++ y Python através del puerto usb-c que también sirve de alimentación. Ofrece 14 pines digitales de entrada/salida, 6 pines de entrada analógica, 1 DAC y 6 pines PWM (Gordillo & Michel, 2024).

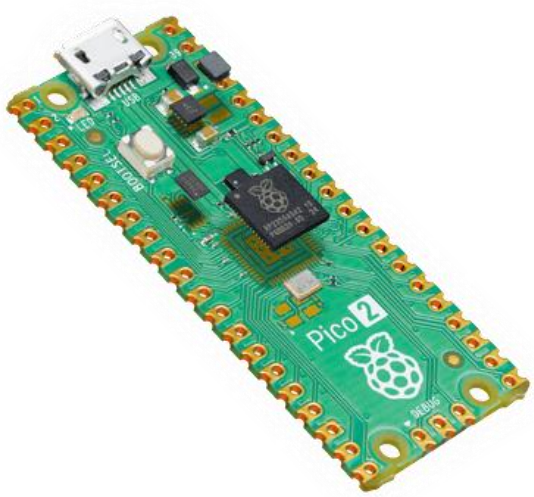
2.7 Raspberry Pi Pico 2

Según la documentación oficial de Raspberry (*Pico-Series Microcontrollers - Raspberry Pi Documentation*, s. f.): Esta nueva tarjeta posee un bajo precio y alto performance en microcontrolador con flexibilidad de interfaz de comunicación. Esta placa posee el microcontrolador RP2350 con una SRAM de 520kB y 4MB de memoria flash, conexión a USB 1.1 y modo de baja potencia. Ofrece 26 GPIO de los cuales 3 pueden ser utilizados como

DAC, de sus interfaces de comunicación tenemos dobles en SPI, I2C, UART. De los 26 GPIO tan solo 24 pueden ser utilizados como canales PWM. Integra un sensor de temperatura, comuniones inalámbricas WiFi 2.4GHzy Bluetooth. Es programable en lenguajes C/C++ y Python (*Silicon - Raspberry Pi Documentation*, 2024).

Figura 6

Raspberry pi pico 2



Fuente: (Raspberry Pi, 2024).

2.8 Sensores

Son elementos capaces de transformar un tipo de energía de tipo físico en otra se tipo eléctrica, otra forma de llamarlos avece es como “transductores”, estos contienen pequeños circuitos y/o integrados que perciben la señal a medir (García & Alberto, 2022).

2.8.1 Sensor de flujo

Es un elemento de medición para cuantificar el flujo de líquidos, comúnmente agua. La composición de este tipo de sensor es comúnmente una carcasa de material plástico incorporando un sensor de efecto hall. Este último es el emisor de pulsos a través de uno de los conductores del sensor, siendo los otros GND y VCC. Entre los principales sensores de este tipo tenemos a los modelos: YF-S201 de conexión ½ in y caudal de 1-30L/min, FS300A con ¾ in de conexión y caudal de 1-60 L/min y el FS400A de conexión de 1in y caudal de 1-60L/min (Ccora & Jarlin, 2020).

Figura 7

Sensor YF-S201

Fuente: (Sigma Elektronik, 2024).

2.8.2 Sensor de temperatura DS18B20

Es un termómetro digital capaz de brindar mediciones de temperatura en la escala de grados Celsius en un tamaño de 9 a 12 bits y con función de alarma con rango de activación no volátil que es ajustable en la interfaz de programación, es decir, se puede programar. Este sensor envía la información a través de un único cable de comunicación lo que se conoce como 1-wire bus y una línea común de tierra para la comunicación con el MCU. Posee un modo de energía parásita en el que se elimina la necesidad de una fuente de alimentación externa. Cada sensor de este tipo posee un numero serial único de identificación de 64 bits almacenado en una ROM integrada que permite funcionar varios DS18B20 con un único 1-wire bus, es decir, estos elementos pueden funcionar de manera paralela distribuidos en un área grande y ser identificador por el cerebro computacional que se esté utilizando. El rango de temperatura medible va desde -55ºC a +125ºC (-67 °F a +257 °F) con una precisión de ±0,5 °C de -10 °C a +85 °C. Se puede alimentar con una fuente de 3V a 5.5V con un consumo activo típico de 1mA y en Standby de 750nA. Además, tiene uso sumergible ya que en su versión de sonda de temperatura está sellado en un envoltorio estanco (Armando, 2021).
Este sensor tiene aplicación en controles ambientales HVAC, sistemas de monitoreo de temperatura dentro de edificios, equipos o maquinaria y sistemas de control y monitoreo de procesos (*DS18B20 - Programmable Resolution 1-Wire Digital Thermometer*, s. f.).

Figura 8

Sonda de temperatura DS14B20

Fuente: (Novatronic, 2020).

2.8.3 Sensor de temperatura y humedad relativa DHT11

Es uno de los más reconocidos debido a su cómodo precio y nivel de dificultad el cual es un sensor muy fácil de usar lo que lo convierte en una opción muy útil para la mayoría de los escenarios. Además de temperatura puede medir la humedad relativa, con un rango de 0 a 50 grados Celsius y del 20% al 90%, respectivamente (Flores & Guadalupe, 2024).

El voltaje de operación va dentro de los 3V-5V DC con una precisión de medición de temperatura de 2.0 ºC y resolución de 0.1ºC con respecto a la humedad la precisión es de 5% RH y una resolución de 1% RH, tiene un característico tiempo de censado de 1 segundo. Tiene un consumo máximo de 2.5mA en estado activo y 50uA en standby (Industries, s. f.).

Figura 9

Sensor de humedad y temperatura relativa DHT11

Fuente: (Fonseca Yupa & Soria Badillo, 2020).

2.8.4 Sensor de distancia ultrasónico HC-SR04

El modo de operar de este sensor es emitiendo ondas ultrasónicas y recibiendo la onda reflejada que retorna desde el objetivo midiendo la

distancia con relación al tiempo de emisión y recepción. La distancia que provee este sensor es una medida precisa siendo su rango de entre 2 a 400cm. Posee baja sensibilidad a la radiación electromagnética o energía solar. Su voltaje de alimentación es de 5V DC y corriente de operación es de 15mA, su ángulo de medición es de 30º es de 15º y una resolución de 0,3cm. Respecto a su pinout; 1 pin de VCC, 1 pin de GND, 1 pin ECHO y 1 pin TRIGGER siendo estas últimas una entrada y una salida respectivamente (Valenzuela-Ramírez et al., 2024).

Figura 10

Sensor ultrasónico HC-SR04.

Fuente: (Fonseca Yupa & Soria Badillo, 2020).

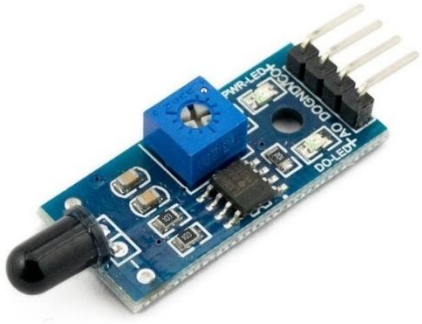
Sensor de flama yg1006

El principio de funcionamiento de este sensor está basado en un fototransistor del tipo NPN de silicio con una alta sensibilidad y velocidad para la detección de fuentes de flujo de luz en el rango de longitud de onda de 760nm a 1100nm. Una desventaja es que este sensor es muy sensible a altas temperaturas por lo que puede fácilmente dañarse, debido a esto es aconsejable tener una distancia considerable con relación a la llama como medida de precaución. Puede operar a una distancia de 100cm. La información de salida de este elemento puede ser del tipo digital o analógica manejable a conveniencia (Danish, 2023).

Este sensor opera con un voltaje de 3.3V -5V DC a 15mA, con un rango de temperatura de - 25°C hasta 85°C y rango de detección de 60º, además posee un potenciómetro que permite regular la sensibilidad de operación (Armando, 2021).

Figura 11

Sensor de flama YG10006.



Fuente: (Naylamp Mechatronics, 2023).

2.8.5 Sensor de corriente no invasivo HSTS016I

Figura 12

Toroide de sensor de corriente no invasivo.

Fuente: (Sigma Elektronik, 2024).

Es un sensor de tipo Hall el cual es empleado para censar la corriente de tipo alterna y continua, AC y DC respectivamente, posee alta precisión y un buen aislamiento eléctrico. Tiene aplicación industrial, automotriz y sistemas de monitoreo de energía debido a sus propiedades. Su modo de operación es usando el principio de efecto hall para medir la corriente através de un conductor, este conductor pasa a través del sensor debido a su forma toroidal. Hay muchas versiones de este sensor con un rango de medida específico que pueden ser de ±50A, ±100A, ±300A, etc., según la versión. La señal de salida depende de la señal de alimentación la cual puede ser de 3.3VDC a 5VDC. Su consumo es mejor de 16mA lo cual lo hace ideal para aplicaciones de bajo consumo energético y aprovechamiento de la energía (Beijing Yaohuadechang Electronic Co., Ltd., s.f.).

2.8.6 Sensor de movimiento HC-SR501

Es un tipo de sensor PIP (piroeléctrico) el cual posee dos agentes detectores distanciados entre sí, obteniendo así una diferencia entre señales capturadas entre estos dos agentes lo que produce la señal de detección de movimiento. Esta repuesta generada por este sensor es envía a un opamp para luego ser llevada a la interfaz electrónica extra. Posee dos resistencias variables que permiten modificar la sensibilidad de detección de objetos, tiempo de activación y tiempo de respuesta. Dentro de los valores nominales de operación tenemos un voltaje de entrada de 5V-20V DC a menos de 1mA, rango de distancia de detección de 3 a 7 metros, ángulo de detección de 110º, salida de alarma de 3 segundo ajustables hasta a 5 minutos. La salida de alarma es a 3.3v 5mA por lo que se puede conectar un led de carga o un relevador para control de carga (Bravo & Manuel, 2023).

Figura 13

Sensor de movimiento PIR HC-SR501

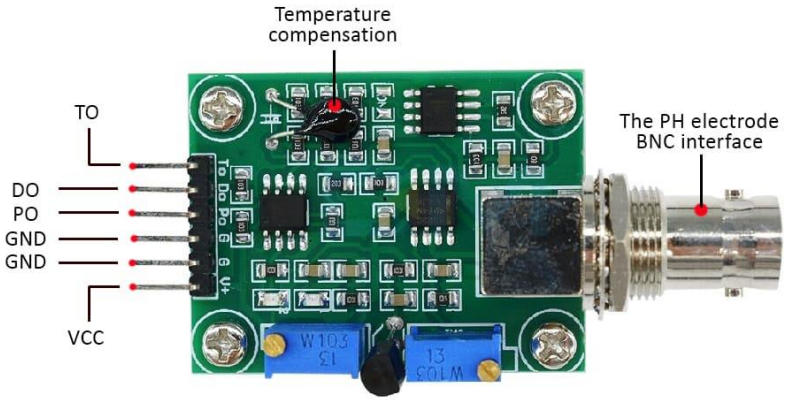
Fuente: (Sigma Elektronik, 2024).

2.8.7 Sensor de PH PH-4502C

Es un sensor útil para la medición del pH en líquidos el cual es una tarjeta acondicionadora de 6 pines el cual se alimenta a 5V con una señal de salida de voltaje de hasta 5V con potenciómetros ajustables para esta señal y facilidad de escala y calculada en programación. Por ejemplo, puede asignársele 0V para pH de 0 y 5V para pH de 14 para luego. La tarjeta tiene un conector para la sonda o electrodo de medición de pH los cuales trabajan siempre en conjunto (*Arduino pH-Meter Using PH-4502C[EN]*, 2020).

Figura 14

Pinout de la tarjeta del sensor PH-4502C y elementos



Fuente: (Cimpleo, 2020).

2.8.8 Sensor de gas metano MQ-4

Figura 15

Modulo sensor de gas metano.

Fuente: (Grupo Electrostore, 2023).

Es un sensor capaz de detectar la presencia de gas metano CH_4 y gas natural que tenga la presencia de metano. Su alimentación es de 5V con salida de 0V a 5V, es decir puede funcionar como sensor de señal digital o de señal analógica, en cuanto esta ultima la señal es relativa a la contaminación con salida de aproximadamente 4V. Su consumo energético es de 150mA. La característica más llamativa es su precio relativamente bajo (Grupo Electrostore, 2023)

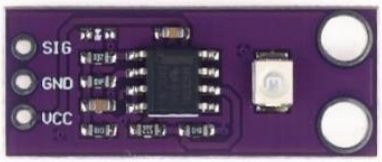
2.8.9 Sensor UV Ultravioleta GUVA-S12SD

Es un módulo sensor capaz de medir la energía visible que se encuentra dentro del espectro electromagnético de la radiación ultravioleta

con una longitud de onda entre 240 a 370 nm lo que se traduce a UV-B y UV-A, el tipo de señal que emite es analógico que va a estar en relación con el índice UV detectado. Es útil para aplicaciones de monitoreo del medio ambiente. Dentro del voltaje de alimentación este puede ser de 2.7VDC a 5.5VDC con una mejor eficiente para este último y un consumo energético muy bajo de microamperios dependiendo del índice UV (*Módulo Sensor de luz ultravioleta (UV) GUVA-S12SD*, 2023).

Figura 16

Modulo sensor Ultravioleta UV



Fuente: (Nuñez & Bryan, 2024).

2.8.10 Sensor Monóxido de carbono MQ-7 CO

Tiene uso en aplicaciones de detección de concentraciones de CO que se encuentren dentro del rango de 20 a 2000 partes por millón (ppm), es altamente sensible y bajo tiempo de reacción, dentro de sus características destaca su tamaño reducido, precio cómodo y fácil conexión a diferencia de otros tipos de sensores de CO. Trabaja con un voltaje de 5VDC, con un voltaje de calentamiento de 5V en alto y de 1.4V en bajo el cual es cuando detecta CO y en su módulo integra un potenciómetro para regular la resistencia de carga mientras que posee una resistencia de calentamiento de 33 Ohm. Puede trabajar con una señal de salida tanto digital como analógica siendo la primera ajustable para mejor aplicación (Armas et al., 2023).

Figura 17

Sensor de monóxido de carbono CO MQ-7

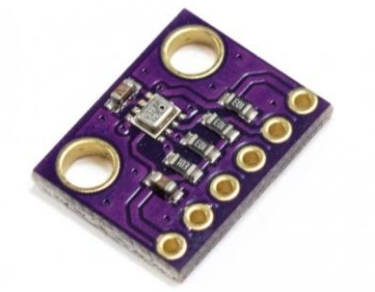
Fuente: (Naylamp Mechatronics, 2024).

2.8.11 Sensor de presión atmosférica

Posee un rango de medición de entre 300 a 1100 hPa (hecto Pascales) o 0.3-1.1 bar. Integra tecnología piezorresistiva de Bosch por lo es acreedor de alta robustez y gran precisión. Se comunica mediante interfaz I2C o SPI a microcontroladores. A diferencia de su antecesor el BMP180 este posee características que lo posicionan más por encima dentro de estas están: Mejor precisión, filtros digitales más eficientes, mejor resolución de temperatura y presión acompañados de un eficiente consumo energético. Es ideal para aplicaciones con drones para determinar la altitud con relación a la presión medida teniendo una precisión de hasta un metro. Puede ser alimenta a 1.8V-3.3V DC. Tiene como rango de altura medible de 0 a 9100 metros (Naylamp Mechatronics, 2024).

Figura 18

Sensor de presión BMP280



Fuente: (Naylamp Mechatronics, 2024).

2.9 Entorno de programación

Es el software que contiene varias herramientas, con las que se es posible escribir código de determinado lenguaje de programación, probar y depurar los programas escritos. Esta herramienta disminuye el nivel de complejidad en el proceso de desarrollo de firmware y software al dotar de una interfaz que usualmente es intuitiva y variada en funciones que facilitan el trabajo del desarrollador (Eras et al., 2024).

2.9.1 Arduino IDE

Figura 19

Interfaz gráfica del IDE de Arduino.

Fuente: (Getting Started with Arduino IDE 2 | Arduino Documentation, 2024).

Es el entorno de desarrollo de código abierto empleado para la programación de proyectos electrónicos. Es una herramienta libre de costo y compatible con los diferentes sistemas operativos (SO) tales como Window, Mac OS y linux (Perez Campoy, 2023).

Arduino IDE permite al usuario utilizar diferentes lenguajes de programación, así como C y C++. Este entorno dispone de librerías predeterminadas por instalación de software que brindan funciones de uso frecuente para la programación de microcontroladores. Estas librerías contienen funciones para leer/escribir entradas y salidas, comunicación entre diferentes sensores, actuadores y/o otros dispositivos (Perez Campoy, 2023).

2.9.1.1 Ventanas y herramientas

Según la documentación oficial para Arduino IDE (*Getting Started with Arduino IDE 2 | Arduino Documentation*, 2024) tenemos las siguientes ventanas y herramientas principales:

- **Skecthbook:** Ventana en la cual se guardan los archivos de código cuya extensión es ".ino" y deben ser almacenados bajo el mismo nombre de la carpeta.
- **Boards Manager:** La cual es una ventana en la que se gestiona los paquetes de tarjetas de desarrollo instalados y los que se deseen instalar.
- **Library Manager:** Sitio donde se gestiona las librerías instaladas y las que ofrece el IDE para poder descargar e instalar.
- **Serial Monitor:** Permite visualizar la información procesada por las placas, el modo de acceder a esta es utilizando el comando Serial.print() como parte del código.
- **Serial Plotter:** Ventana destinada para la visualización de datos empleando gráficas de dos ejes.
- **Example Sketches:** Apartado en el que la interfaz ofrece programas de ejemplo para su ejecución rápida.
- **Debugging:** Empleada para la depuración de código navegando por el programa con el control adecuado.

2.9.2 Visual studio code

Comúnmente llamado VScode, programa desarrollado por la empresa Microsoft siendo en concreto un editor de código gratuito de gran uso para desarrolladores en diversas áreas. Su amplio abanico de herramientas, extensiones y personalización lo convierten en una herramienta idónea para proyecto que impliquen el uso de microcontroladores (Microsoft, 2024).

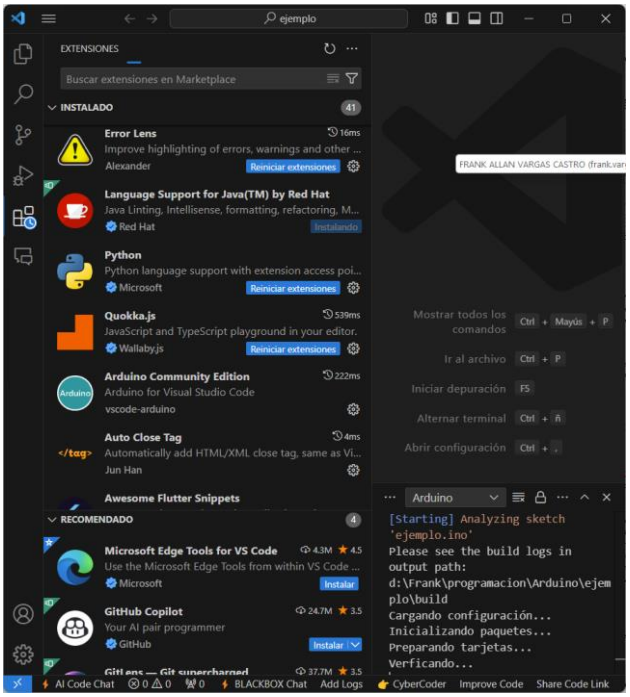
Principales características según Microsoft (2024):

- Compatible con diversos lenguajes de programación.
- Incorporación de la terminal para correr comandos.

- Librería de extensiones que incorporan funcionalidad y mejor experiencia de trabajo.
- Depurador de código integrado y control de versiones.
- Compatible para Window, MacOS, Linux.

Figura 20

Interfaz gráfica de Visual Studio Code apartado de extensiones y herramientas.



Fuente: Elaboración propia.

2.10 Internet de las cosas

Es una alternativa tecnológica moderna en la que propone un nuevo modelo de interacción del mundo físico y digital. Es una red de equipos inteligentes dotados de sensores y conexión a internet, los cuales captan datos del entorno real de forma masiva para diferentes opciones de trabajo con la información, es decir, optimización de procesos, automatización de tareas, etc (Gómez & Daniel, 2024).

Con el internet de las cosas los dispositivos suelen trabajar comunicados de manera inalámbrica y todo el almacenamiento de la información se suele llevar a cabo en la nube (Goudarzi et al., 2021).

Respecto al control de la comunicación de dispositivos IOT, es común ver utilizar tipos de interfaces para que el usuario monitoree las variables

medidas y ejerza control sobre los actuadores en caso de ser necesario, estas lecturas y escrituras se dan en tiempo real por lo que los datos obtenidos y el control ejercido se dan al mismo tiempo de ejecución (Behnke & Austad, 2024).

2.10.1 Funcionamiento del IoT

El término se refiere a los dispositivos físicos que conforman un sistema, que reciben y emiten datos en una red inalámbrica con la ausencia de una parte humana. Esto se logra con la integración de dispositivos electrónicos en diferentes objetos cotidianos. Por ejemplo, un termostato inteligente, dispositivo IoT, permite percibir la temperatura desde cualquier ubicación. El apartado visual ofrece permisos de calibración y control sobre la temperatura utilizando los diferentes elementos de la arquitectura IoT. Su Funcionamiento se basa en el envío, recepción y análisis de la información de manera cíclica retroalimentada. Dependerá de los recursos y agentes que intervienen en el sistema para el analisis de esta información de manera inmediata o en periodos de tiempo, estos agentes pueden ser las personas o sistemas basados en inteligencia artificial y machine learning. Por ejemplo, en el caso del termostato, el sistema IoT puede estar programado para consumir una API de google maps para indicar cuando es ideal controlar la temperatura deseada dependiendo el flujo vehicular. Por otro lado, la información obtenida puede usarse también para determinar hábitos de conducción (*¿Qué es el Internet de las cosas y cómo funciona?*, s. f.).

2.10.2 Arquitectura IoT

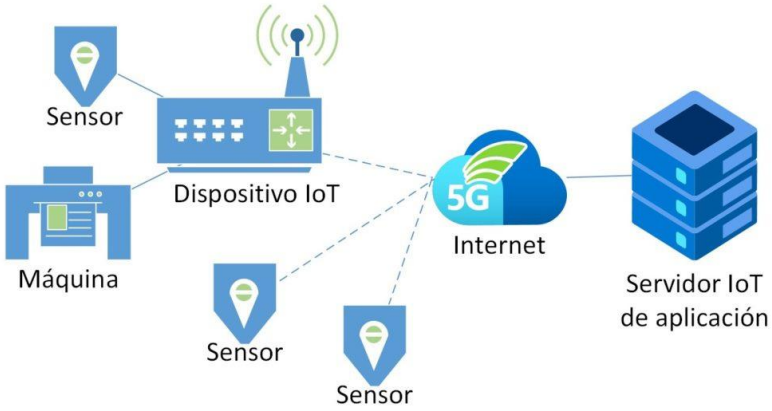
Según Darío Herrera (Herrera Chávez, 2020), los componentes claves que forman parte de una arquitectura IoT son:

- Dispositivos IoT: Los cuales son sensores, actuadores o centrales de procesamiento que pueden ser conectados inalámbricamente o vía cableada. Su fin es la interacción con el entorno circundante para poder producir o trabajar con información para llevar a cabo determinada actividad, siendo estos dispositivos inteligentes.

- Redes: Son la encargada del tráfico de la información producida o consumida por los dispositivos y nube. Esta es la que sostiene la comunicación entre los elementos IoT.
- Nube: Es un conjunto de servidores ubicados estratégicamente en diferentes partes del mundo. Su función se basa en el almacenamiento de la información, con los cumplimientos de seguridad requeridos, para ser solicitada y visualizada por el usuario desde cualquier parte del mundo y cuantas veces sean necesarias.

Figura 21

Arquitectura simple de una red de aplicación IoT.



Fuente: IoT (IoT Projects, 2024)

2.10.3 IoT en domótica

El IoT y la domótica son áreas tecnologías que se encuentran en constante transformación a velocidades increíbles cambiando la manera de vivir. Esta automatización inteligente del hogar mejora significativamente la comodidad, la eficiencia energética, la seguridad u cuidado de las personas. Esto se traduce en un gran impacto tecnológico hacia la sociedad. Una de las aplicaciones de esta tecnología en la domótica se da en la reducción de consumo energético en las casas lo que se traduce como una reducción en la huella de carbono y preservación del medio ambiente. Además, mejora la manera de vivir de los usuarios de maneras como accesibilidad, control y monitoreo remoto en aplicaciones de salud y bienestar, aplicaciones que se adaptan a las necesidades individuales (Ochoa & Jeanella, 2024).

2.10.4 IoT Empresarial

El IoT no solo abarca el mundo de las casas, puede ser aplicado también en el ámbito empresarial, esto con el fin de facilitar muchas de las tantas tareas de rutina que se realizan diariamente o aportar información adicional que no es fácil de obtener (*Dispositivos IoT en el entorno empresarial | Empresas | INCIBE*, 2021).

2.10.5 IoT Industrial

Hace referencia a los elementos que conforman el sistema IoT, sensores, instrumentos y dispositivos autónomos que establecen conexión mutua utilizando el internet. Esto permite la recolección de información útil para establecer parámetros de productividad y eficiencia en los diferentes procesos industriales y prestación de servicios. El IoT industrial o IIoT es una rama más del internet de las cosas, siendo vital para tareas principales para la producción como: Resolución de conflictos desde cualquier lugar del mundo, administración de suministro y recursos para dosificación de manera eficiente, búsqueda de posibles fallas para mantenimiento predictivo, control de inventario, etc. Es por esto por lo que su uso se da principalmente en sectores como la manufactura, transporte y energía, por lo que no es de extrañar que su uso se extienda más hacia los sistemas de control y mantenimiento (*IoT Industrial y sus principales aplicaciones*, 2022).

2.11 Tecnologías de desarrollo web

Las tecnologías para el desarrollo de páginas web son principalmente tres: HTML en su estructura, CSS en sus estilos y Javascript en su lógica y funcionalidad.

2.11.1 HTML

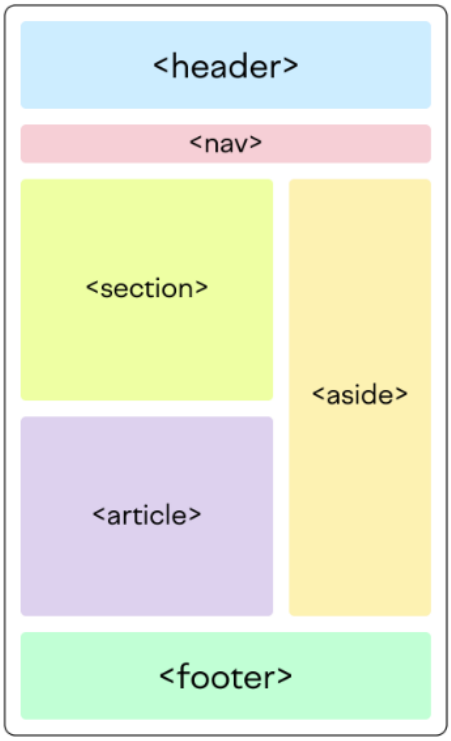
HyperText MarkUP Language, abreviado como HTML y traducido al español como Lenguaje de marcado de hipertexto, es un lenguaje de marcado que se utiliza para crear la estructura y contenido de páginas web, suele ser traducido como un lenguaje de formato de documentos. Este estándar está a manos de W3C, esta es una organización cuyo objetivo es estandarizar tecnologías destinada a la web (W3C, 2020).

HTML tiene la labor de indicar como los textos y estructura se visualizan en una página web. Debido que es un estándar trata de funcionar en cualquier página web sin darle importancia a la versión implementada para que cualquier navegador pueda trabajar con ella (MDN Web Docs, 2021).

La versión más actual es HMTL5, misma que contiene etiquetas, atributos y demás implementaciones tales como un amplio repertorio de funciones que dan la capacidad a los sitios web y aplicaciones tener una mayor personalización y funcionalidad para alcance (Mahedero Biot, 2020).

Figura 22

Estructura semántica con etiquetas HTML para mejor accesibilidad



Fuente: (HTML Semántico, 2023).

2.11.2 CSS

Describe el diseño y la presentación de documentos HTML o XML. Su principal función es definir cómo se muestra el contenido estructurado, lo que incluye elementos como colores, fuentes, márgenes, líneas y disposición de elementos en la pantalla. La separación de contenido (HTML) y presentación (CSS) promueve la flexibilidad y eficiencia en la gestión del diseño web (O'Neill, 2020, p. 45).

2.11.2.1 Principales características de CSS.

Existe una gran brecha entre páginas desarrolladas con hojas de estilos y con las que no, Un sitio sin CSS no es más que texto y demás elementos colocados en un lienzo blanco pareciendo que la página aún sigue cargando su contenido. Este tipo de lenguaje permite añadir estilo hasta al más minúsculo elemento dentro del árbol HTML haciendo todo este trabajo desde un directorio diferente al de la estructura obteniendo así un resultado escalable y mantenible, es una ventaja sete hecho debido a que previamente se trabajaba en el mismo documento HTML debido a que era el estándar y no había otra manera. Además, es permitido poseer diferentes y múltiples estilos en una misma página por lo que es imposible que existe páginas con estilos idénticos debido a la creatividad y posibilidades infinitas de diseño (B. Gustavo, 2019).

La manera de empezar a estilizar es escribiendo el selector del elemento, clase, id, pseudoclases o pseudoelemento a la que se desea aplicar estilo prosiguiendo de una apertura y cierre de llaves, dentro de estas llaves se colocarán las propiedades con los valores que se desean establecer, cabe mencionar que CSS maneja valores predefinidos para muchas propiedades como el tamaño de fuente con valor de 16px. El par propiedad-valor se lo conoce como declaración. CSS permite que escribir múltiples selectores para escribir una misma declaración para todos con el fin de reducir código (MDN contributors, 2024).

2.11.3 JavaScript

Es el lenguaje de programación que se utiliza para implementar funcionalidad en las páginas web para mejorar la experiencia de usuario. En comparación con HTML que no son lenguaje de programación como tal, JS permite la gestión de la información como en formularios, manipulación del DOM, BOM, consumo de API’s, etc (Flanagan, 2020).

Es bien valorado este lenguaje debido a su gran conveniencia para volver interactivos los sitios webs y mejorar la experiencia del usuario (Flanagan, 2020).

Según Crockford (Crockford, 2020), las ventajas que ofrece javascript

son:

- **Rapidez:** La carga se da del lado del cliente y no del servidor, siendo específicamente del navegador, dependiendo del dispositivo del cliente para la determinación del tiempo de carga.
- **Versatilidad:** Debido a que abarca una gran comunidad de desarrolladores, estos han sido aportadores con Frameworks y librerías que se pueden usar con mucha facilidad de implementación como Angular, React, Vue, Etc.

2.12 Servidor Web

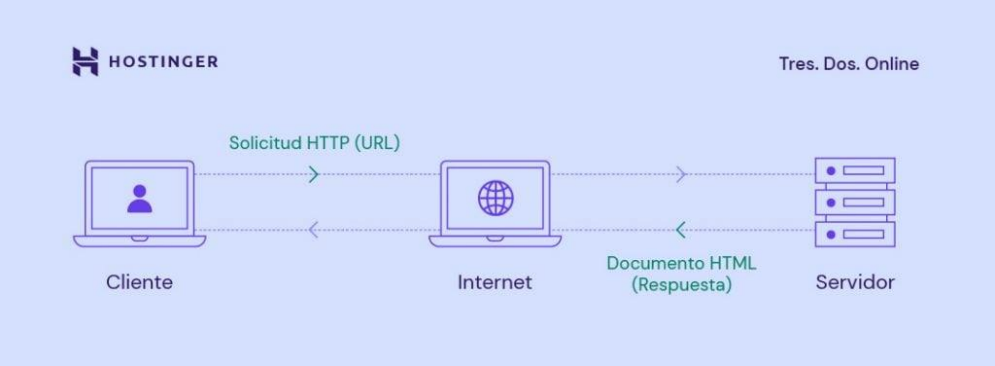
Es un dispositivo o programa que dota de funciones o herramientas a otros dispositivos o sistemas que se encuentran bajo la misma red. Dentro de sus capacidades está el alojamiento de sitios web, gestión y control de acceso a dispositivos remotos, control de base de datos, entre otras. Es común llamar servidor ya sea al programa en sí, como a el dispositivo o dispositivos que lo lleven a ejecución, aunque cabe recalcar que servidor es el software/firmware mientras que hosting es el hardware, en otras palabras, el programa es el servidor y el dispositivo es el hosting (Universitat Politècnica De València, 2014).

Un servidor utilizas varios protocolos dentro de ellos tenemos HTTP el cual es usado para gestionar peticiones del cliente por medio de la web, también existen otros protocolos como SMTP y FTP para la gestión de correos y archivos en la web, respectivamente. Además, es el software quien indica la forma de acceso a la información que se encuentra almacenada en el servidor, por lo que existe mínimo un servidor HTTP para el manejo de peticiones HTTP (get, put, post, delete) y URL existentes. Para las peticiones, los servidores utilizan el modelo cliente-servidor (Betania V, 2022).

2.12.1 Cliente y servidor

Figura 23

Representación simple de la relación Cliente-Servidor



Fuente: (Betania V, 2022).

Es un tipo de arquitectura comúnmente usada y fácil de aplicar con la que se puede poner en marcha un servidor. Esta arquitectura posee dos tipos de entidades, ya conocidas como el servidor y el cliente, aunque de este último pueden ser varios para un mismo servidor. La comunicación es llevada por medio de peticiones del cliente y respuestas del servidor, aunque esto puede variar dependiendo del tipo de tecnología en concreto a utilizarse como es el caso del uso de Websocket con el uso de servidor asíncrono (Universitat Politècnica De València, 2014).

2.12.2 Servidor Web estático y dinámico

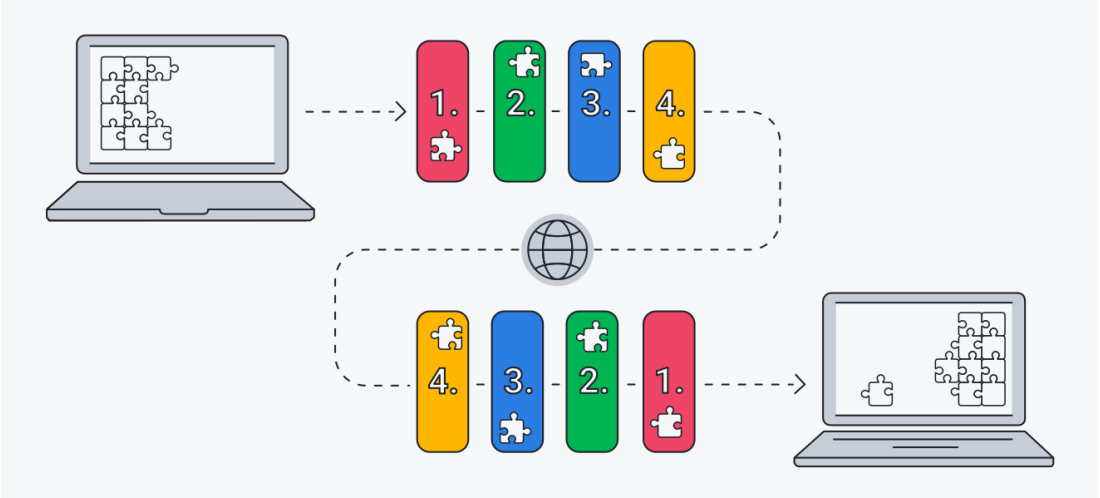
Según la documentación de la red de desarrolladores de mozilla un servidor estático, que puede llamarse pila, es un sistema computacional que aloja un servidor HTTP el cual este último es el agente llamado estático debido a que transfiere la información contenida en archivo sin realizar modificaciones sobre ella hacia el navegador, mientras que el servidor web dinámico es un SW estático con la implementación de un programa extra como puede ser el caso de bases de datos o una aplicación para poder modificar el contenido alojado antes de enviarlo hacia el navegador por medio del servidor HTTP (*¿Qué es un servidor WEB?*, 2024).

2.12.3 Protocolos de comunicación

2.12.3.1 TCP/IP

Figura 24

Modelo de tráfico de información por capas del protocolo TCP/IP.



Fuente: (Bodnar, 2021).

Es un protocolo de red encargado de llevar el tráfico de datos asegurados mediante una vía de comunicación definida. Tiene uso en la navegación en internet, mensajería electrónica, envío y gestión de archivos (Tapia & Enrique, 2024).

Con respecto a las ventajas y desventajas de TCP según Tapia y Enrique (Tapia & Enrique, 2024) tenemos:

Ventajas:

- **Fiabilidad**: TCP garantiza que los datos se entreguen de forma ordenada y segura.
- **Detección de errores**: Posee su propio sistema de detección de anomalías y errores, gestionando que los datos no se corrompan.
- **Control de flujo**: Normaliza el tráfico de información con el objetivo de mantener la red

- **Orientado a la conexión**: Realiza una conexión previa en modo seguro con el fin de tener una vía estable de comunicación para el envío de la información

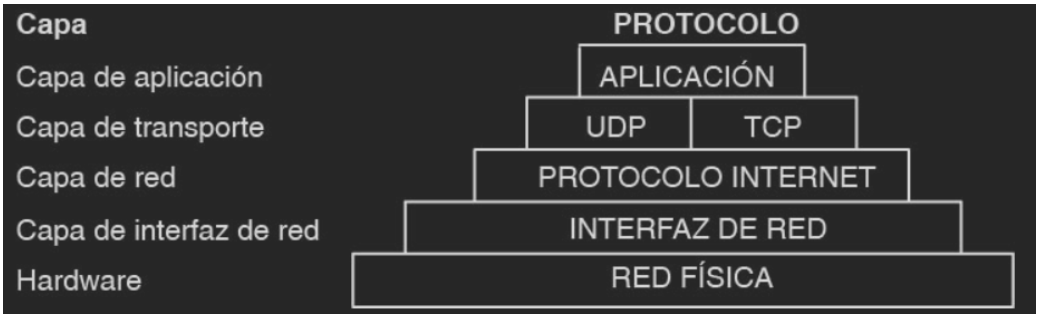
Desventajas:

- **Sobrecarga**: Los procesos que aseguran la entrega y el control del tráfico añaden carga adicional a la transmisión.
- **Latencia**: Debido a la necesidad de confirmar la recepción de los datos, el protocolo puede generar cierta demora.
- **Ineficiencia con datos pequeños**: Para cantidades reducidas de datos, el proceso de establecer una conexión puede resultar poco eficiente.

Capas del modelo TCP/IP

Figura 25

Capas del modelo TCP/IP, La Ultima capa refiere a los elementos físicos de la red



Fuente: (AIX 7.2, 2024).

Existen cuatro capas en el modelo TCP/IP las cuales trabajan de manera cooperativa enviándose la información entre sí con un determinado orden cada vez que se solicita el envío de información y con un orden inverso para la recepción de la información. Cuando estas capas están juntas forman lo que se conoce como un conjunto de protocolos (*¿Qué Es TCP/IP?*, 2021).

- **Capa 1 o Capa de acceso a la Red**: Aquí se define la forma en la que se permite el acceso físico a la red a los dispositivos que se encuentran conectados, así como los protocolos con los que se trabajará las comunicaciones además de la topología de la red y el límite de la red ya sea local o global (Castillo, 2020).

- **Capa 2 o Capa de internet:** Incluye el protocolo ip por lo que es quien controla el flujo de datos y el enrutamiento de información hacia el destino, su velocidad radica en el tráfico de red y clientes (*CAPAS DEL MODELO TCPIP | Universidad del Azuay*, s. f.).
- **Capa 3 o Capa de transporte:** Establece conexión entre el emisor y receptor enviando la información codificada para su envió por internet utilizando los protocolos TCP y UDP (Protocolo de datagrama de usuario) por la confiabilidad, control de flujo, simplicidad y velocidad (Micucci, 2024).
- **Capa 4 o Capa de aplicaciones:** Es el lugar donde la información se vuelve útil para el usuario. En esta capa interactúan mucho las aplicaciones que son utilizadas para la creación de datos. Por ejemplo, los navegadores que envían la información que obtienen hacia las demás capas por medio de un dominio (DNS) lo que a su vez está asociado a direcciones de protocolo IP (*Modelo TCP/IP vs. Modelo OSI | Similitudes y Diferencias*, 2022).

2.12.3.2 MQTT

Es un protocolo apropiado para microcontroladores, este establece una fácil comunicación entre los dispositivos y la nube, y nube y muchos dispositivos. Generalmente se aplica a redes de bajos recursos y ancho de banda limitado. Es un protocolo muy ligero y eficiente incluyendo los encabezados que utiliza para los mensajes transmitidos, por ejemplo, el mensaje más pequeño puede ser del tamaño de dos bytes. Por otro lago es escalable con facilidad al necesitar pocos cambios en el código y de bajo consumo energético (*¿Qué es el MQTT?*, s. f.).

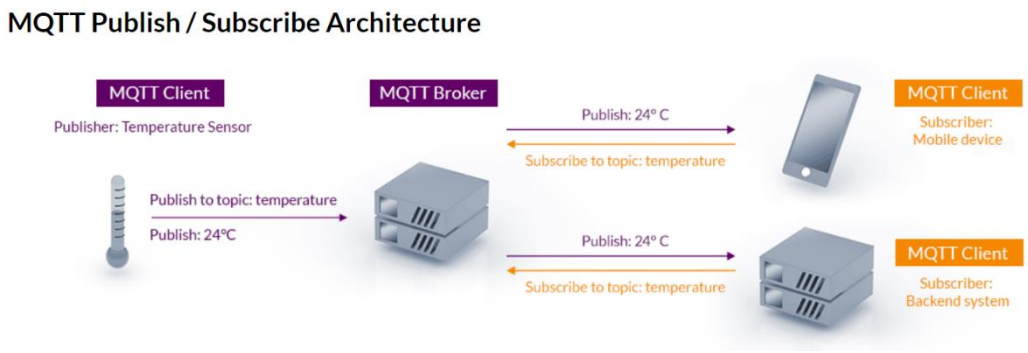
La base de ejecución de este protocolo es TCP/IP para la capa de transporte y recibe soporte sobre este y WebSocket. Por lo que cada vez que un usuario establece conexión lo hace con credenciales pertinentes como en los servicios de autenticación de API's (*IBM Maximo Monitor 8.8 to 9.0 continuous delivery*, 2024). Dentro de esta etapa de autenticación hay tres niveles: QoS 0 en el cual el mensaje es de envío único, QoS 1 en el cual se

envía mensaje hasta ser entregado y QoS 2 en el que se asegura el que mensaje haya sido entregado una única vez (Román et al., 2023).

Arquitectura de MQTT

Figura 26

Protocolo HTTP en funcionamiento de una red bajo la arquitectura Cliente-Servidor



Fuente: (MQTT - The Standard for IoT Messaging, 2024).

Solo existen dos entidades en la red para este protocolo: Los clientes y el Broker. Los clientes tienen el papel de suscribirse a su bróker y comunicarse con ellos en una relación de publicación y suscripción de mensajes categorizados por temas. Cuando el bróker percibe un tema de mensaje lo envía a los clientes que se hayan suscritos al mismo tema. Además, los clientes también pueden publicar, suscribirse a uno o varios temas. La estructura de los mensajes está conformada por el tema (topic), el cual es un nombre de identificación para el mensaje; y la carga (payload), que es la información por mensaje. El tema está en formato UTF-8 mientras que la carga está en bytes de información ya sea texto, datos binarios, etc (JESÚS, 2020).

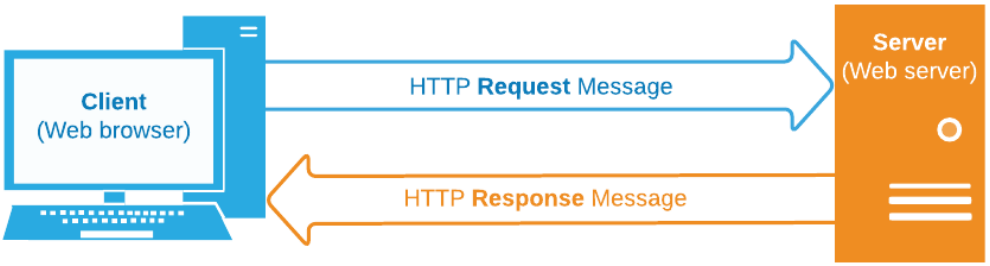
2.12.3.3 HTTP

Conocido como protocolo de transferencia de hipertexto o (Hypertext Transfer Protocol, por sus siglas en inglés) es un protocolo implementado para el envío de información a través de documentos hipermedia. Es utilizado para la comunicación entre servidores web y otros propósitos como comunicación entre navegadores. Emplea la arquitectura cliente-servidor. Además, es un protocolo sin estado puesto que no almacena ningún tipo de dato entre las peticiones. Es común que este protocolo se utilice en conexiones TCP/IP

también se puede utilizar en cualquier tecnología de comunicación siempre y cuando sea segura y fiable, sin perder información (*HTTP | MDN*, 2023). ´

Figura 27

Protocolo HTTP en funcionamiento de una red bajo la arquitectura Cliente-Servidor



Fuente: (Redes HTTP en JavaScript – Manual para Principiantes, 2024).

Arquitectura de HTTP

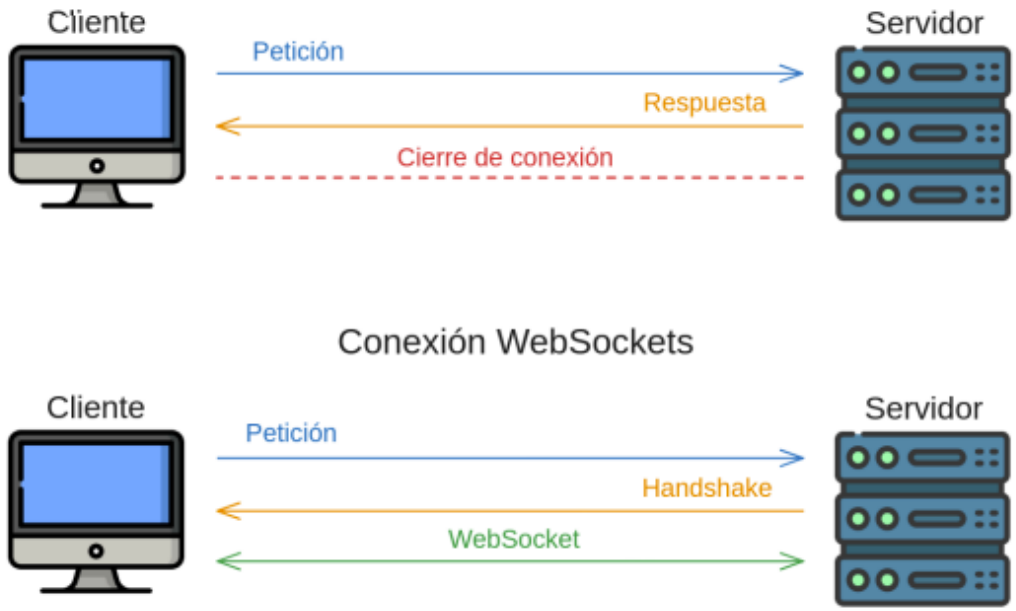
Una de las entidades de HTTP es el agente del usuario el cual es el encargado de enviar las peticiones. La mayoria de los casos este agente del usuario es un navegador web, pero también puede ser un bot explorando la web en busca de información. Por otro lado, cuando una petición individual es enviada al servidor existen múltiples agentes intermedios que manejan cada petición los cuales son los proxys que cumplen como gateways o caches. Aunque también pueden ser considerados agentes intermedios los módems, routers, etc (*Generalidades del protocolo HTTP - HTTP | MDN*, 2024).

2.12.3.4 WebSocket

Es un protocolo que permite comunicar interactivamente el navegador con el servidor. Esto se realiza por medio de una Application Programming Interfaz API misma que se utiliza para enviar y recibir información con el servidor de manera controlada a través de eventos. Actúa de manera similar al protocolo HTTP en la que se apertura la comunicación para petición y respuesta ante las solicitudes realizadas entre el cliente y el servidor con la diferencia de que el cliente no necesita volver a realizar solicitudes para la comunicación en tiempo real (Milla-Herrero, 2023).

Figura 28

Representación de las conexiones HTTP y WebSocket en arquitectura Cliente-Servidor.



Fuente: (Arriola Salazar, 2024).

Además, con websockets permite que el servidor envíe información al cliente cada vez que existen cambios sin la necesidad que el cliente realice solicitudes al servidor lo que permite que los valores esten en constante actualización, es decir, en tiempo real (Arriola Salazar, 2024). WebSockets y HTTP comunicación.

Dentro de las características principales según Arriola Salazar (2024), tenemos:

- **Bidireccional:** La comunicación se realiza de manera simultánea y en ambos sentidos, por lo que cual cualquier dispositivo puede iniciar la comunicación, enviar información y realizar solicitudes en cualquier momento.
- **Conexión persistente:** Destaca entre muchos protocolos debido a que este no demanda utilizar nuevas conexiones con futuras solicitudes, lo que hace en su lugar es sostener una vía de comunicación con cada uno de los clientes.

- **Bajo overhead:** Reduce la sobrecarga de encabezados HTTP que no son requeridos durante las comunicaciones lo que permite un bajo nivel de overhead.
- **Seguridad:** Con posibilidades de funcionar superpuesto a HTTPS, lo que se traduce a una capa más de seguridad para el cifrado de la información transmitida en la arquitectura utilizada.

Capítulo 3: Diseño, Implementación Y Resultados

3.1 Esquema conceptual

El presente trabajo de investigación posee como objetivo implementar un sistema IoT basado en el microcontrolador ESP32 para el uso de estudiantes de la institución, como material didáctico de practica de programación de sistemas embebidos, sistemas web, o demás aplicaciones que encuentre el alumnado como objeto de aplicación y práctica.

El proyecto se encuentra dividido las siguientes secciones:

- Sección de hardware: En esta parte del proyecto, se encuentra la parte de desarrollo del sistema con una descripción de funcionamiento y elección de los diferentes elementos físicos del sistema, tales como:
    - ESP32-DevKit es una placa de desarrollo asequible y de bajo consumo energético, equipada con conectividad Wi-Fi, Bluetooth, UART y ESP-NOW, lo que le permite comunicarse mediante diversos protocolos según las necesidades del sistema. En este proyecto, la ESP32 actuará como el cerebro del sistema, encargándose de recibir datos de los sensores y, simultáneamente, funcionando como un servidor web local. A través de su conexión Wi-Fi, permitirá visualizar la información en una página web IoT, desarrollada con tecnologías web, facilitando el acceso y monitoreo de los datos en tiempo real.
    - En cuanto a los sensores, se escogió sensores de aplicaciones comunes en domótica e inmótica siendo estos dispositivos los primeros elementos del funcionamiento del sistema ya que serán los encargados de censar variables del entorno mismas que serán enviadas al MCU para su procesamiento y visualización.
    - El sistema contará con su fuente de alimentación de voltaje constante para la alimentación del

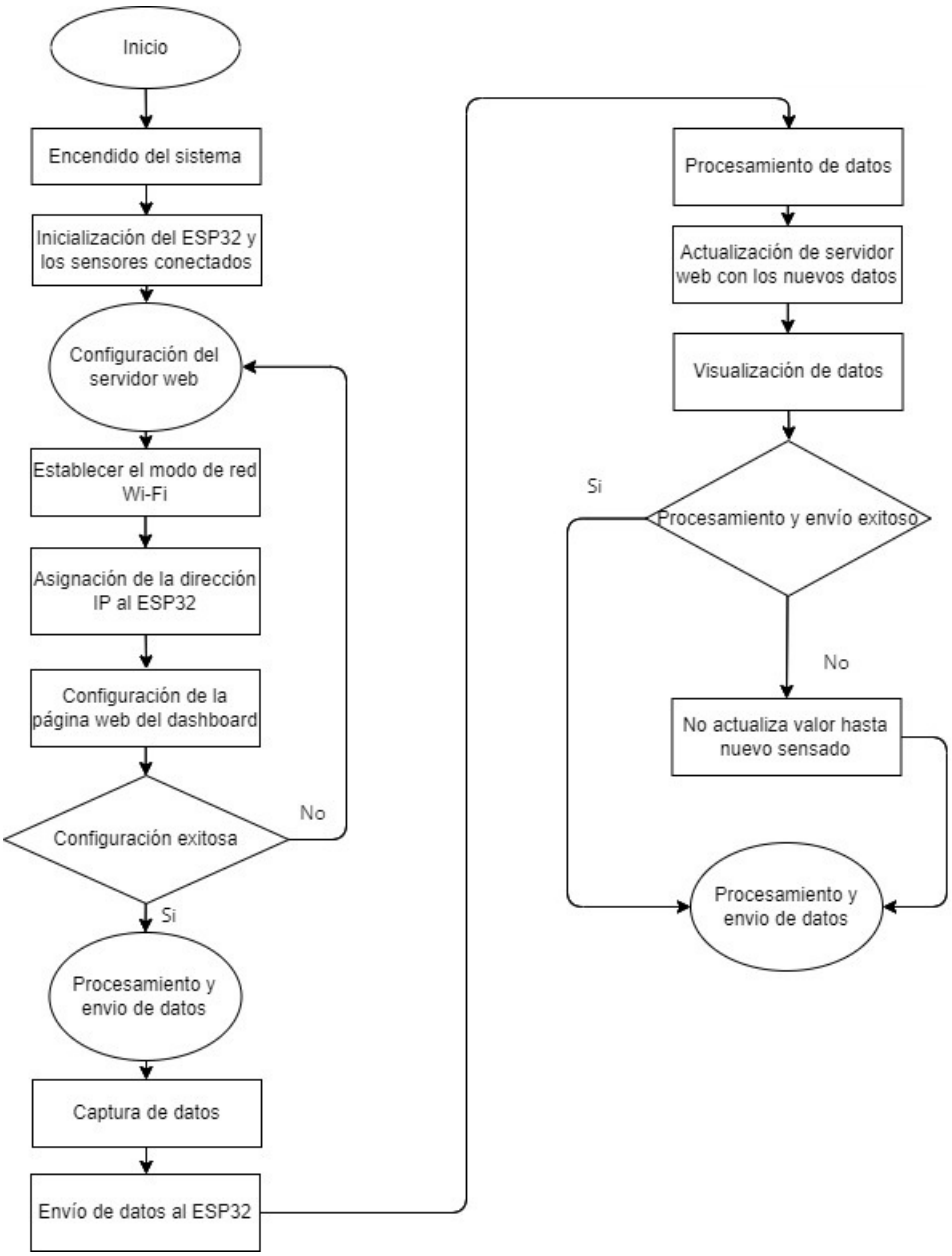
microcontrolador y de los sensores, además permitirá el uso de los GPIOS libres para futuras aplicaciones con sensores o actuadores por medio de relays.

- Sección de software: Con respecto a este punto, se detalla el paso a paso para la programación de los diferentes componentes antes mencionados:
    - La ESP32, será programada por medio de dos softwares de desarrollo, siendo ARDUINO IDE el principal programa a utilizar ya que aquí es donde se codificará el funcionamiento de los sensores, la implementación final de la página web IOT de visualización de la información censada y su configuración como servidor web. Por otro lado, el editor de código VS code siendo este una herramienta para el desarrollo de la página web debido a sus múltiples extensiones que facilitan del desarrollo de esta parte del proyecto.
    - La página web será el medio de visualización de datos en tiempo real cumpliendo la función de interfaz gráfica para los equipos conectados a la misma red wifi. El desarrollo contempla tecnologías básicas para páginas y sitios web siendo: HTML, CSS y Javascript.

3.2 Diagrama de flujo para el sistema

Figura 29

Diagrama de flujo para la inicialización, lectura de variables y envió de datos al servidor ESP.



Fuente: Elaboración propia.

3.2.1 Explicación del diagrama de flujo

El sistema tiene inicio con la puesta en marcha de la ESP32 y todos los sensores conectados a ella. Una vez activos estos elementos se inicializa la configuración y ajustes de parámetros del servidor web que utiliza la librería

WebServer y WebSockets para el ESP32, este elemento es quien permitirá la comunicación y el envío de datos a la página web desarrollada.

Una vez que el sistema haya inicializado correctamente se establece conexión con la red wi-fi configurada para el ESP32 la cual está establecida de manera local para realizar identificación y asignación por medio de una dirección IP. Esta dirección IP es el medio de acceso a la página web que servirá como una interfaz gráfica interactiva para la visualización de las variables medidas por medio de los sensores. Este “dashboard” permite una visualización de datos en tiempo real por lo que cada dato que capture cada sensor será mostrado en vivo en la página web.

Una vez la configuración del servidor y página web el sistema verifica si esta parte del proceso fue exitosa. En caso de no, la programación intentará todo el proceso anterior hasta tener éxito y poder continuar con el flujo del proceso, es decir, se retroalimentará hasta que todo esté correctamente establecido. Si la configuración resulta en un éxito se inicia el proceso envío de datos hacia la MCU. Por lo que la ESP32 empieza a recibir los datos de los sensores conectados. Cuando los datos ya se encuentren en el microcontrolador y hayan sido procesados serán mostrados en la página web.

Cuando toda la información ha sido procesada el servidor web se actualiza dejando así que la página muestre los datos de manera gráfica utilizando indicadores analógicos y digitales respectivamente para cada tipo de variable medida, esto en un dashboard dinámico. Esta parte del proceso también consta con retroalimentación. Para el caso que de los datos hayan tenido un correcto procesamiento envío y se hayan mostrados correctamente, el sistema continúa con su operación normal volviendo a tomar etapa de procesamiento y envío de datos de los sensores. En caso de que la información de uno o varios sensores no haya sido mostrada con existo se debido a problemas de lectura, el sistema intentará nuevamente la captura de datos hasta tener información nueva para mostrar, en el caso del DHT11 la consola del navegador mostrará un mensaje de aviso, para el DS18B20 se mostrará un valor de temperatura de -127º dando a entender que hubo problema con la lectura de datos.

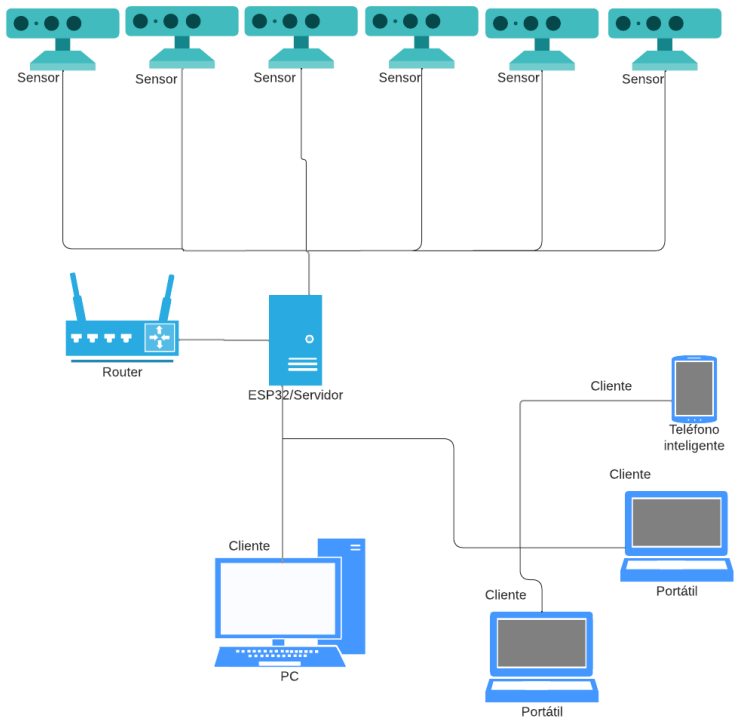
Así, el sistema sigue un flujo retroalimentado constante de captura, procesamiento y visualización de la información de los sensores garantizando que la página web esté actualizada todo el para el monitoreo de variables.

3.3 Diagrama de red del sistema

De acuerdo con la figura del diagrama de red, el ESP32 se conectará a una red de internet para poder servir como servidor web, implementando una arquitectura cliente-servidor en la que los dispositivos conectados bajo la misma red: laptops, pc's, celulares o cualquier equipo que pueda usar navegador web; podrá visualizar la página web que alojará el servidor y por ende monitorear las variables a medir con los sensores escogidos.

Figura 30

Diagrama simple de la conexión de los sensores al ESP32 como servidor y los clientes.



Fuente: Elaboración propia.

3.4 Selección de hardware

3.4.1 Consideraciones.

Para el desarrollo de la tarjeta electrónica de adquisición de datos se consideró el uso de elementos muy fácil de encontrar, de buena asequibilidad

y útiles en el mundo tecnológico actual como en aplicaciones inteligentes de domótica, inmótica y automatización, para una mejor familiarización con estos elementos y una baja dificultad de uso. Además, la tarjeta permitirá utilizar los GPIOS que no se consideraron, para demás prácticas y aplicaciones ya sea para trabajar con el mismo código de página web desarrollado o con otras opciones que desee la persona interesada en hacer uso de esta herramienta.

Placa de desarrollo

Este es el cerebro del sistema es la tarjeta ESP-32WROOM, la cual es común verla en su presentación de 38 pines, con versiones de bluetooth y antena para wifi incluida, lo que lo hace ideal para una gran variedad de aplicaciones. Posee interfaces UART, I2C, SPI, SDIO y PWM. Además, es programable en Arduino IDE, VS code y demás editores de códigos y entornos de programación que surgen con el pasar del tiempo. Los lenguajes de programación para este microprocesador son el c++ y Python en su versión para MCU llamada microPython. Servirá de servidor web que alojará una página web desarrollada con html para la estructura, css para el apartado estético y javascript para la funcionalidad. Así mismo, esp32 se encargará de obtener los datos por medio de los sensores para mostrarlos en la página web. Se opta por este microcontrolador debido a su bajo precio y fácil obtención en el mercado.

DS18B20

Es una sonda de temperatura que integra en su interior al ds18b20, este puede ser utilizado sumergido en agua debido a su encapsulamiento. Y de bajo consumo energético ideal en aplicaciones IoT donde se requiere monitoreo de la temperatura de líquidos y aplicaciones domóticas.

YF-S201

Es un sensor ideal para el censado de flujo siendo aplicable en monitoreo de consumo de recursos hídricos para un buen uso y aprovechamiento del agua, aplicaciones domóticas para determinar el consumo de agua, aplicaciones de dosificación de líquidos y demás usos. Es de bajo costo con relación a otros medidores de flujo y muy preciso.

DHT11

Es un sensor muy utilizado en todo tipo de aplicaciones de monitoreo/control de temperatura debido a que también es capaz de medir la humedad relativa del ambiente, por lo que también es aplicable para la agricultura y muchas más aplicaciones. Es uno de los sensores más baratos y utilizados del mercado por lo que no es de extrañar de encontrarlo de una tienda de electrónica.

HC-SR04

Es un sensor de aplicación muy requeridas para la determinación de distancias como el llenado de objetos, posicionamiento de objetos, detección de objetos, etc. También es utilizado en la rama de la robótica, domótica, agricultura y otras áreas más. Es cómodo precio y compatible con muchas placas de desarrollo debido a su amplio voltaje de operación.

C-SR501

Sensor muy utilizado mucho en la automatización domótica, inmótica, control de acceso, eficiencia energética de iluminación, etc. Siendo muy fácil de encontrar en la vida diaria es lo que lo hace ideal para usarse y requerirse muy frecuentemente.

YG1006

Es un sensor de flama muy económico ideal para aplicaciones IoT debido a su bajo consumo energético, puede trabajar de manera analógica o digital para detectar la presencia fuentes de luz infrarroja producida por las llamas, y además posee una buena distancia de detección. Pese a su sensibilidad a daño es un elemento muy fácil de reemplazar ya que es parte de la listo de los sensores más económicos que puede existir.

LM2596

Este módulo regulador de voltaje “Step-Down” servirá como regulador de alimentación para el voltaje requerido por el microcontrolador y los sensores, además servirá como elemento de protección por lo que cumplirá con esto dos funciones en el sistema, alimentación y protección. Será

alimentado con 12Vdc por medio de un adaptador de corriente para reducirlos al voltaje requerido.

3.5 Perfil Económico

Tabla 1

Perfil económico en tarjeta y sus elementos

Elementos	Cantidad	Precio	Subtotal
Esp32	1	$9,80	$9,80
Dht11	1	$4,00	$4,00
Ds18b20	1	$6,00	$6,00
Yf-s201	1	$10,00	$10,00
Hc-sr04	1	$3,50	$3,50
Hc-sr501	1	$4,00	$4,00
Yg10006	1	$3,50	$3,50
Modulo step-down	1	$3,50	$3,50
Espadines hembra x40	2	$0,60	$1,20
Espadines macho x40	2	$0,60	$1,20
PCB + Envío	1	$35,00	$35,00
Cables USB	1	$3,00	$3,00
Total			$84,70

Fuente: Elaboración propia.

Dentro del perfil económico se consideran los gastos en principalmente en elementos para el sistema, estos son: Microcontrolador, sensores, módulo de alimentación, espadines, gastos de fabricación de PCB y envío; teniendo como gasto $84,70 en este apartado.

En la segunda tabla del perfil económico tenemos gastos respecto a los materiales donde se colocarán los sensores, tarjeta electrónica, fuente de alimentación; dando un valor de $47.96.

Tabla 2

Perfil económico: Gastos para armado del sistema en gabinete

Elementos	Cantidad	Precio	Subtotal
Fuente Conmutada 12V - 5Amp	1	$9,50	$9,50
Cables para sensores	1	$6,56	$6,56
Tablero plástico	1	$31,40	$31,40
Prensa stop	1	$0,50	$0,50
Total			$47,96

Fuente: Elaboración propia.

Se tiene como total de gastos un valor de $132,66 esto considerando que se desprecia ciertos gastos como el valor de cable para la alimentación

de fuente conmutada debido a que se constaba con este material, por lo que redujo gastos.

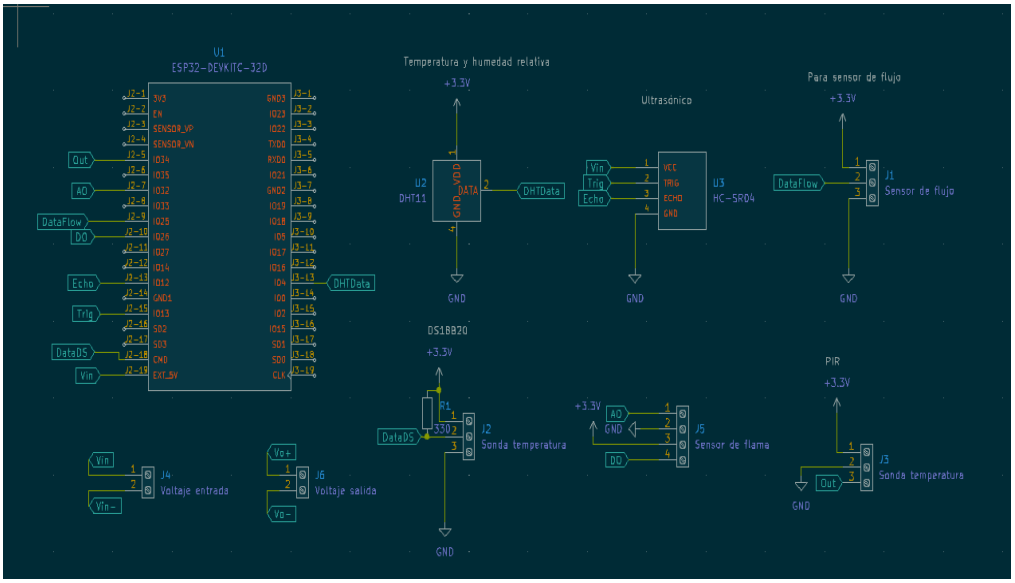
3.6 Modelado electrónico

El circuito fue elaborado en KICAD un software gratuito para el diseño electrónico que se mantiene funcional debido a la comunidad activa y aportadora de donaciones monetarias al proyecto del software.

Se utiliza el software debido a que posee la mayoría de footprints de elementos existentes en el mercado contenidos dentro de su librería que viene por defecto. Esto facilita la elaboración del circuito para su respectiva fabricación. Se utilizaron borneras a falta de los sensores seleccionados. Por otra parte, con los sensores seleccionados se les otorgará un diseño 3D de bornera para fácil integración en la placa. Adicional, la placa constará con los GPIOS no utilizados, libres en forma de borneras para el uso de cualquier posterior aplicación y fácil integración de sensores, relés, diodos o actuadores.

Figura 31

Esquemático electrónico del circuito para PCB



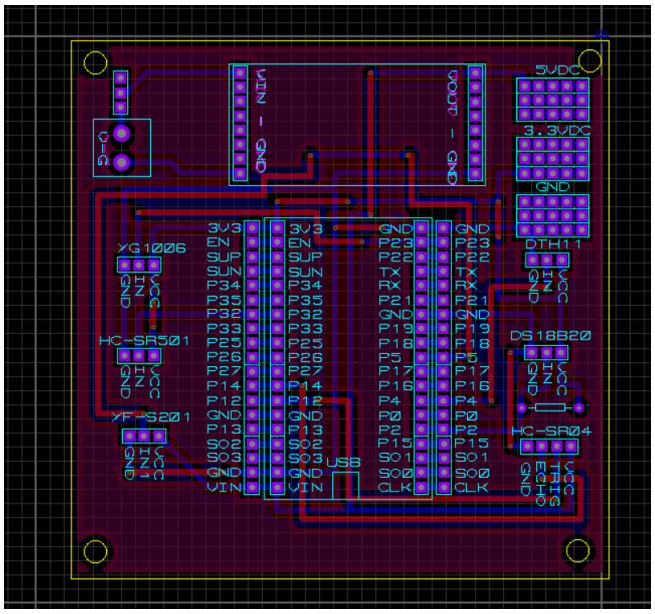
Fuente: Elaboración propia.

El diseño 3d y diseño electrónico de la placa permite ver cómo será el resultado final de la placa luego de su envío a fabricación. Como se puede observar en las imágenes, esta consta con varios pines de alimentación de voltaje y GND para la utilización con los demás GPIOS del microcontrolador

en caso de usos extras deseados. En la misma están reservados los espacios para los sensores seleccionados y para el módulo Step Down de alimentación al circuito entero.

Figura 32

Diseño y trazado de pistas de elementos de la pcb para posterior fabricación.

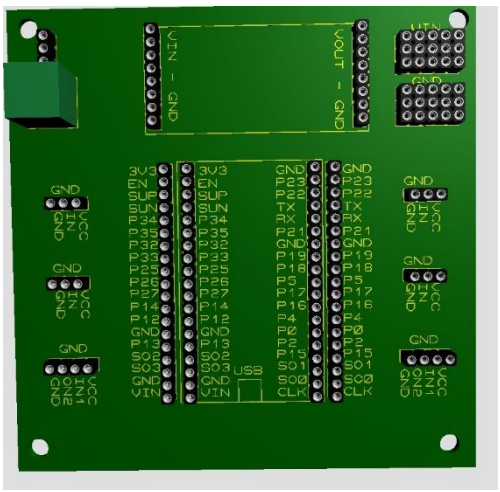


Fuente: Elaboración propia.

Las dimensiones de la placa son 9,5cm*9,5cm por lo que se considera de tamaño reducido.

Figura 33

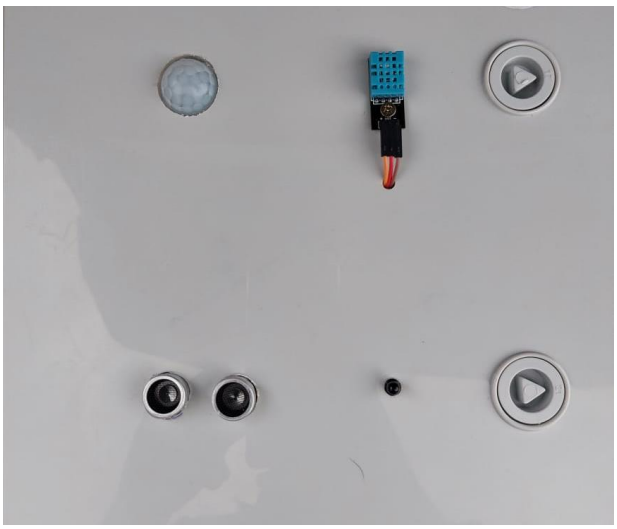
Visibilidad 3D de la placa desarrollada que integrará sensores



Fuente: Elaboración propia.

Figura 34

Ubicación de los sensores colocados en un tablero PVC



Fuente: Elaboración propia.

Figura 35

Segunda ubicación de los sensores colocados en el tablero

Fuente: Elaboración propia.

3.7 Desarrollo de software

Esta parte está dividida en dos, puesto que la primera consta del desarrollo de la página web mientras que la segunda consta de la programación del ESP32 como servidor HTTP incluyendo la página web.

3.7.1 Desarrollo de página web

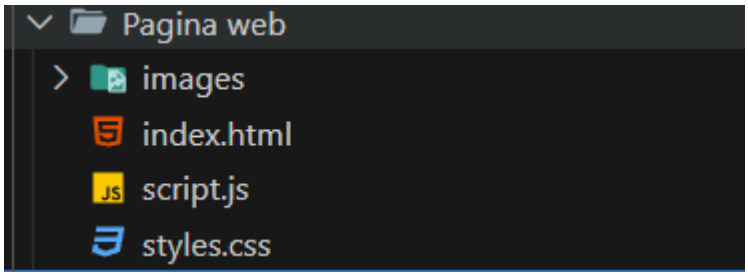
La herramienta utilizada para este desarrollar el documento a subir al servidor es el Visual studio code, el cual es una herramienta o editor de código muy útil a la hora de programar debido al gran repositorio de extensiones de trabajo y configuraciones que posee.

En este programa se utiliza principalmente la extensión “Live Server”, misma sirve para crear un servidor web local de manera rápida sin realizar configuraciones con el fin de poder visualizar contenido de html, css y javascript.

Durante la elaboración de la página web se crea la siguiente estructura de carpeta para un eficiente manejo y desenvolvimiento.

Figura 36

Directorio de trabajo para el desarrollo web



Fuente: Elaboración propia.

Las imágenes utilizadas son contenidas en una carpeta llamada “images” para poder tener categorizados los archivos a utilizar.

Lo primero a realizar en el documento “index” es crear la estructura inicial de una página web en la cual se especifica los metadatos a utilizar, el título de la página web y su icono, el idioma, el uso de la hoja de estilos conjunto con su ubicación mediante dirección relativa o absoluta, archivo o archivos scripts a utilizar con su respectiva dirección de ubicación y de manera opcional las librerías o “frameworks” a usar.

Figura 37

Estructura inicial de página web

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Prueba de Indicadores</title>
    <link rel="stylesheet" href="styles.css">
    <link rel="shortcut icon" href="images/favico.png" type="image/x-icon">
</head>
<body>
    <script src="script.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/raphael"></script>
    <script src="https://cdn.jsdelivr.net/npm/justgage@1.3.5/justgage.min.js"></script>
</body>
</html>
```

Fuente: Elaboración propia.

Los archivos script se dejan para el final debido a que cuando el usuario se conecte al servidor su dispositivo de visualización carga primero la estructura y luego la funcionalidad de la página.

Es importante recalcar que las clases y id’s a usar en las etiquetas son netamente de referenciales y sin ningún tipo de obligación a usar palabras especificar para cada una de estas, y para fines prácticos cada una de estas esta referenciada según su función o característica a resaltar dentro del uso especifico presente.

En el cuerpo HTML se adiciona la etiqueta <header> con el título con etiqueta <h1> y las etiquetas de imágenes a utilizar. Con respecto a las imágenes, se opta por subir las requeridas para la página web a un sitio de alojamiento de imágenes para poder usar la etiqueta <img> e implementar las figuras con url.

Adicional se agrega el <main> que contendrá los elementos de visualización con el uso de elementos

Figura 38

Imágenes y cuerpo de la página

```
<body>
    <header>
        <h1 id="title">Prueba de Indicadores</h1>
        <img id="logo" src="images/UCSG.png" alt="UCSG_logo">

        <img id="logo" src="https://redi.cedia.edu.ec/members/UCSG.png" alt="UCSG_iconlogo">
        <img id="logo2" src="https://www.ucsg.edu.ec/wp-content/uploads/2018/11/fac-tec.jpg" alt="FEDT_logo">
    </header>
    <main>

        <!-- primer contenedor -->
        <div class="gauge-container">

        </div>

        <!-- segundo contenedor -->
        <div class="gauge-container">

        </div>
    </main>
```

Fuente: Elaboración propia.

El desarrollo de un sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT, utilizará el microcontrolador ESP-WROOM-32 integrando diferentes sensores de uso domótico en una tarjeta electrónica de tamaño reducido. En cuanto a su firmware este microcontrolador alojará una página web desarrollada en código HTML, CSS y Javascript; este resultado se aplica en forma de HTML embebido alojado en espacio de memoria del dispositivo, adicional a este el ESP32 captará las mediciones de los sensores para mostrar los resultados en indicadores contenidos en la página web. Con la comunicación wifi-integrada utilizará los protocolos WebServer y WebSocket para el manejo de múltiples clientes locales y comunicación en tiempo real para la visualización de la información obtenida. Debido a las propiedades de CSS la página tiene la particularidad de poseer un diseño responsive por lo que su visualización se ajusta a pantallas más pequeñas como celulares o tablets desde el navegador web. Con la implementación de este sistema de monitorización IoT se espera que sea una herramienta educativa de aplicaciones prácticas para estudiantes de las carreras técnicas de la Facultad de Educación Técnica para el Desarrollo de la UCSG puesto que el módulo electrónico propuesto permite el uso de los GPIOS no utilizados en el presente trabajo de integración.

Posterior, dentro de cada contenedor se añade individualmente div's que contendrán una etiqueta para el objeto indicador y otra etiqueta <label> para nombrar a la variable medida. Esto se realiza tanto para el primer

contenedor como para el segundo por fines visuales tal y como se codifica.

Figura 39

Contenedores de los indicadores y texto de indicadores

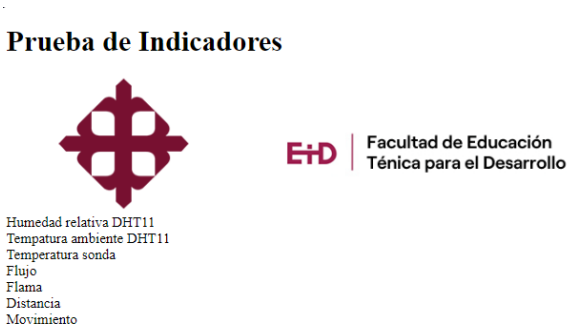
```
<!— primer contenedor —>
        <div class="gauge-container">

            <div class="card">
                <div id="HumidityGauge" class="gauge"></div>
                <label class="sensor-label">Humedad relativa DHT11</label>
            </div>

            <div class="card">
                <div id="TemperatureGauge" class="gauge"></div>
                <label class="sensor-label">Tempatura ambiente DHT11</label>
            </div>

            <div class="card">
                <div id="TemperatureGauge2" class="gauge"></div>
                <label class="sensor-label">Temperatura sonda</label>
            </div>

            <div class="card">
                <div id="FlowGauge" class="gauge"></div>
                <label class="sensor-label">Flujo</label>
            </div>
        </div>

        <!— segundo contenedor —>
        <div class="gauge-container">

            <div class="card">
                <div id="FlameGauge" class="gauge">
                    <i class="fa-solid fa-fire fireStatic fire"></i>
                    <i class="fa-solid fa-fire fireAlert fire fa-beat-fade hidden" style="color: #d88f21;"></i>
                </div>
                <label class="sensor-label">Flama</label>
            </div>

            <div class="card">
                <div id="DistanceGauge" class="gauge"></div>
                <label class="sensor-label">Distancia</label>
            </div>

            <div class="card">
                <div id="MotionGauge" class="gauge">
                    <i class="fa-solid fa-users motionStatic people"></i>
                    <i class="fa-solid fa-users motionAlert people fa-fade hidden" style="color: #9d2121;"></i>
                </div>
                <label class="sensor-label">Movimiento</label>
            </div>

        </div>
```

Fuente: *Elaboración* propia.

El resultado hasta este punto no es nada atractivo por lo que se precisa estilizar con CSS.

Figura 40

Primera visualización de la página web solo con estructura HTML



Fuente: Elaboración propia.

Dentro del CSS se coloca primero las variables en el selector “root”, para este caso se utiliza las variables para almacenar códigos de colores para uso posterior. El selector * se implementa con el objetivo de borrar parámetros que vienen por defecto en una plantilla HTML. Y para el selector “body” se especifica el tipo de fuente, color de fondo, márgenes, relleno, modo de mostrar los elementos, posicionamiento de estos y espacio a utilizar. Para el espacio a utilizar se define como el todo el alto y ancho aprovechable por la pantalla del dispositivo de visualización, es decir, el 100% del alto de la pantalla disponible de un teléfono, de una Tablet o de una pantalla de computadora.

Figura 41

Estilos para los elementos iniciales y variables de css

```css
:root {
--background-color: #F4F6FA;
--primary-color: #4A90E2;
--text-color: #333333;
--card-bg: #FFFFFF;
--shadow-color: rgba(0, 0, 0, 0.1);
}

*{
    box-sizing: border-box;
    margin: 0;
    padding: 0;
}

body {
    font-family: 'Arial', sans-serif;
    background-color: var(--background-color);
    color: var(--text-color);
    margin: 0;
    padding: 0;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    min-height: 100vh;
    width: 100vw;
}

```

Fuente: Elaboración propia.

Se prosigue con el selector del header dándole un margen de separación del título y los elementos de visualización. Se le otorga alineación, color y tamaño a título mediante su selector, así mismo se posiciona las imágenes logo de la universidad y de la facultad FETD con su posición izquierda superior y derecha superior respectivamente, con un tamaño apreciable.

Se añade estilos para el elemento main y a la clase “.gauge-container” indicando su tamaño, forma de mostrar elementos, espacio y alineación.

Figura 42

Estilo para los títulos e imágenes

```css
header {
    margin-bottom: 20px;
}

#title {
    font-size: 2em;
    color: var(--primary-color);
    text-align: center;
}

#logo{
    display: inline;
    position: absolute;
    top:10px;
    left:-15px;
    width: 130px;
}

#logo2{
    display: inline;
    position: absolute;
    top:0px;
    right:0px;
    width: 230px;
    padding: 10px;
    border-radius: 25px;
}
```

Fuente: Elaboración propia.

Se con la siguiente línea de css se añade estilo al contenedor de cada “tarjeta” de indicador, se establece su tamaño, alineación de los elementos hijos, posicionamiento, borde, sombras, color. Asi mismo, con pseudoelementos ::before y :: after se elabora propiedades de animación de líneas trazando una ruta alrededor de cada tarjeta, esto solo fines estéticos. Esta animación surte efecto con la herramienta @keyframe la cual ejecuta la propiedad de animación que se especifique y como se especifique, en este caso una animación de rotación de 0 a 360 grados.

Para finalizar el archivo css, se usa el selector #titles para mostrar los títulos en una sola columna alineada en el centro con respecto al eje vertical.

Figura 43

Estilización en el main y los contenedores

```
main{
    display: flex;
    flex-direction: column;
    justify-content: space-evenly;
    height: 80vh;
}

.gauge-container {
    display: flex;
    flex-wrap: wrap;
    gap: 20px;
    justify-content: space-evenly;
    align-items: center;
    width: 70vw;
    padding-bottom:20px ;
}
```

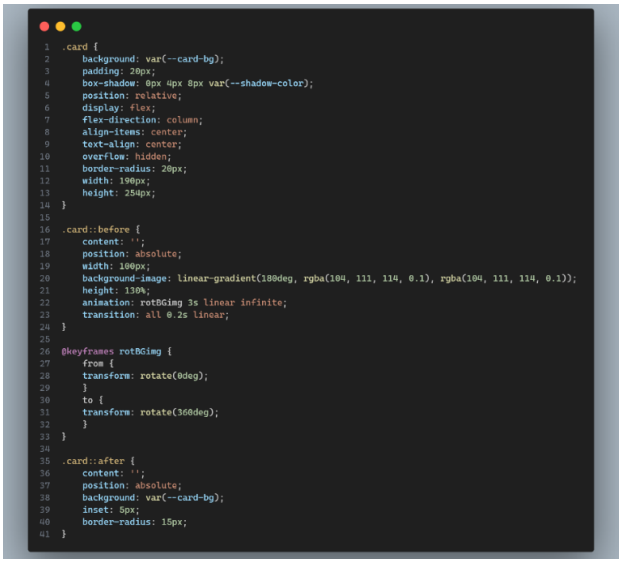
Fuente: Elaboración propia.

Con el selector “.fire” y “.people” se colocan los elementos de los indicadores de sensores de flama y movimiento respectivamente y se establece su tamaño y posición. Se crea una clase (.hidden) para uso posterior para ocultar elementos y hacer una multiplexación entre los elementos de estos sensores mencionados y mostrar la animación respectiva de cuando el sensor envía la señal de la variable medida y ocultar la representación estática, asi mismo en caso contrario. Por último, se usa la característica “@media” para establecer una configuración de adaptabilidad cuando la pantalla tenga menos de 768px, el cual es un tamaño de pantallas de dispositivos móviles, para hacer de esta página web responsiva al dispositivo en el que se esté visualizando los datos.

La visualización hasta este punto da fruto a la siguiente página web, quedando como parte final de esta etapa la programación con javascript, para implementación de funcionalidad.

Figura 44

Animación para cartas y sus estilos

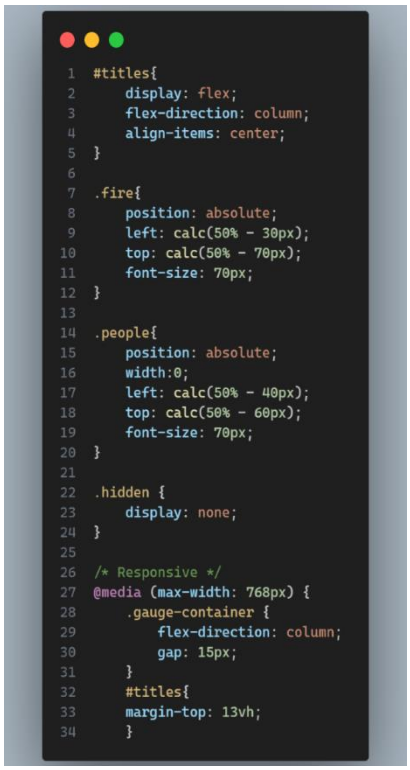


```css
.card {
    background: var(--card-bg);
    padding: 20px;
    box-shadow: 0px 4px 8px var(--shadow-color);
    position: relative;
    display: flex;
    flex-direction: column;
    align-items: center;
    text-align: center;
    overflow: hidden;
    border-radius: 20px;
    width: 190px;
    height: 254px;
}

.card::before {
    content: '';
    position: absolute;
    width: 100px;
    background-image: linear-gradient(180deg, rgba(104, 111, 114, 0.1), rgba(104, 111, 114, 0.1));
    height: 130%;
    animation: rotBGimg 3s linear infinite;
    transition: all 0.2s linear;
}

@keyframes rotBGimg {
    from {
    transform: rotate(0deg);
    }
    to {
    transform: rotate(360deg);
    }
}

.card::after {
    content: '';
    position: absolute;
    background: var(--card-bg);
    inset: 5px;
    border-radius: 15px;
}
```

Fuente: Elaboración propia.

Figura 45

Estilo para los iconos para los indicadores de señal digital.



```css
#titles{
    display: flex;
    flex-direction: column;
    align-items: center;
}

.fire{
    position: absolute;
    left: calc(50% - 30px);
    top: calc(50% - 70px);
    font-size: 70px;
}

.people{
    position: absolute;
    width:0;
    left: calc(50% - 40px);
    top: calc(50% - 60px);
    font-size: 70px;
}

.hidden {
    display: none;
}

/* Responsive */
@media (max-width: 768px) {
    .gauge-container {
        flex-direction: column;
        gap: 15px;
    }
    #titles{
    margin-top: 13vh;
    }
```

Fuente: Elaboración propia

Para el código de JS se empieza con escribiendo en el archivo “script.js”, la primera línea establece la conexión WebSocket declarándola en una

constante “ws” permitiendo que el sistema asigne una IP utilizando el puerto 81 el cual se indica dentro de la Clase WebSocket.

Figura 46

Creación de objeto con la clase WebSocket para su conexión con HTML y uso en el servidor

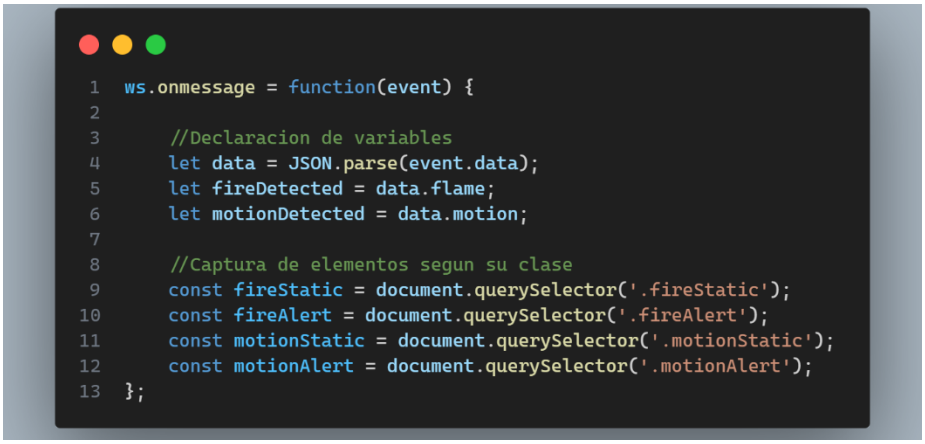
```
1  //Conexion WebSocket
2          const ws = new WebSocket('ws://' + window.location.hostname + ':81/');
3
```

Fuente: Elaboración propia

Luego se usa el método “onmessage” del objeto para poder escuchar eventos y obtener la información enviada del ESP32 a la página web en formato JSON parseando la información para obtener el resultado deseado. Para esto se utilizan variables locales en la función con el fin de evitar errores de código y que sea más legible. Adicionalmente se recorre al DOM (Document Objet Model) poder trabajar con los nodos del html, sus hijos y propiedades, en este caso de lo utiliza para poder cambiar propiedades de los elementos que se desean manipular para ocultarlos y mostrarlos con la detección de los sensores digitales.

Figura 47

Obtención de la información mediante la obtención del evento y selección de elementos por medio del DOM



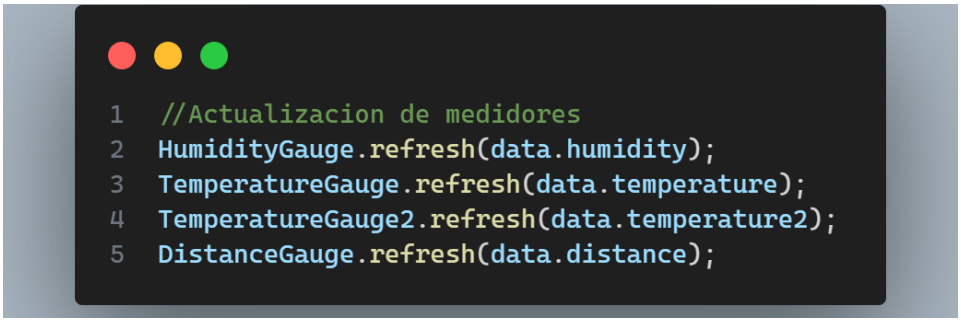
```
1  ws.onmessage = function(event) {
2
3      //Declaracion de variables
4      let data = JSON.parse(event.data);
5      let fireDetected = data.flame;
6      let motionDetected = data.motion;
7
8      //Captura de elementos segun su clase
9      const fireStatic = document.querySelector('.fireStatic');
10     const fireAlert = document.querySelector('.fireAlert');
11     const motionStatic = document.querySelector('.motionStatic');
12     const motionAlert = document.querySelector('.motionAlert');
13 };
```

Fuente: Elaboración propia

Posteriormente, continuando en la misma función, se declara los métodos de los indicadores de tipo objeto con la información a utilizar para su visualización bajo comunicación WS. Estos nombres deben ser los mismo para los objetos de los indicadores para poder usar sus métodos siendo ".refresh" el utilizado para el envío de los valores para los indicadores.

Figura 48

Actualización de los indicadores con el método "refresh" utilizando la información del evento como parámetro de la función



```
1  //Actualizacion de medidores
2  HumidityGauge.refresh(data.humidity);
3  TemperatureGauge.refresh(data.temperature);
4  TemperatureGauge2.refresh(data.temperature2);
5  DistanceGauge.refresh(data.distance);
```

Fuente: Elaboración propia

Figura 49

Control de flujo para la asignación de clase a los iconos que contienen y no contienen animación

```
1  // Control de Flama
2  if (fireDetected === "ON") {
3      fireStatic.classList.remove('hidden');
4      fireAlert.classList.add('hidden');
5  } else {
6      fireStatic.classList.add('hidden');
7      fireAlert.classList.remove('hidden');
8  }
9
10 //Control de presencia
11 if (motionDetected === "DETECTADO") {
12     motionStatic.classList.add('hidden');
13     motionAlert.classList.remove('hidden');
14 } else {
15     motionStatic.classList.remove('hidden');
16     motionAlert.classList.add('hidden');
17 }
```

Fuente: Elaboración propia

Para finalizar la función se implementa un control de flujo en el que compara si el valor fuego detectado (“fireDetected”) es igual a “ON” procederá asignar la clase “.hidden” al elemento sin animación y a borrar esta misma clase al elemento con animación para que sea visible este efecto en la página web mostrar la animación respectiva y en caso contrario asignará la clase para ocultar el elemento animado y asignará la clase al elemento sin animación. Esto este mismo principio es aplicado para el sensor de movimiento si la variable de que indique que hay movimiento detectado (“motionDetected”) posee un valor de “DETECTADO” y su inverso.

Figura 50

Instanciación de los objetos indicadores analógicos

```
let HumidityGauge = new JustGage({
    id: "HumidityGauge",
    gaugeWidthScale: .4,
    decimals:2,
    value: 0,
    min: 0,
    max: 100,
    label: "%",
    labelFontColor: "#000000",
    levelColors: ["#00ff00", "#ffdd00", "#ff0000"],
    labelMinFontSize:20,
    valueMinFontSize:20,
});
let TemperatureGauge = new JustGage({
    id: "TemperatureGauge",
    gaugeWidthScale: .4,
    decimals:2,
    value: 0,
    min: 0,
    max: 80,
    label: "°C",
    labelFontColor: "#000000",
    levelColors: ["#00ff00", "#ffdd00", "#ff0000"],
    labelMinFontSize:20,
    valueMinFontSize:20,
});
let TemperatureGauge2 = new JustGage({
    id: "TemperatureGauge2",
    gaugeWidthScale: .4,
    decimals:2,
    value: 0,
    min: -10,
    max: 80,
    label: "°C",
    labelFontColor: "#000000",
    levelColors: ["#00ff00", "#ffdd00", "#ff0000"],
    labelMinFontSize:20,
    valueMinFontSize:20,
});
let FlowGauge = new JustGage({
    id: "FlowGauge",
    gaugeWidthScale: .4,
    decimals:2,
    value: 0,
    min: 0,
    max: 40,
    label: "L/min",
    labelFontColor: "#000000",
    levelColors: ["#00ff00", "#ffdd00", "#ff0000"],
    labelMinFontSize:20,
    valueMinFontSize:20,
});
let DistanceGauge = new JustGage({
    id: "DistanceGauge",
    gaugeWidthScale: .4,
    decimals:2,
    value: 0,
    min: 0,
    max: 400,
    label: "cm",
    labelFontColor: "#000000",
    levelColors: ["#00ff00", "#ffdd00", "#ff0000"],
    labelMinFontSize:20,
    valueMinFontSize:20,
});
```

Fuente: Elaboración propia

Como punto final se instancia los sensores utilizando los nombres de variables utilizados en los métodos para actualizar valores con el fin de que se cree correctamente la herencia de atributos y métodos. Cabe mencionar que para instanciar estos objetos de la librería utilizada se requiere brindar el cómo parámetro del atributo “id” la clase que se asignó para el contenedor de los sensores según corresponda. Además, durante l instancia se definen los valores máximos y mínimos para el indicador, los colores, fuentes, etiquetas, decimales y tamaño.

El resultado visual del desarrollo del código de la página es el siguiente:

Figura 51

Página final con las tecnologías de desarrollo web

Fuente: Elaboración propia.

3.7.2 Desarrollo de programa para ESP32

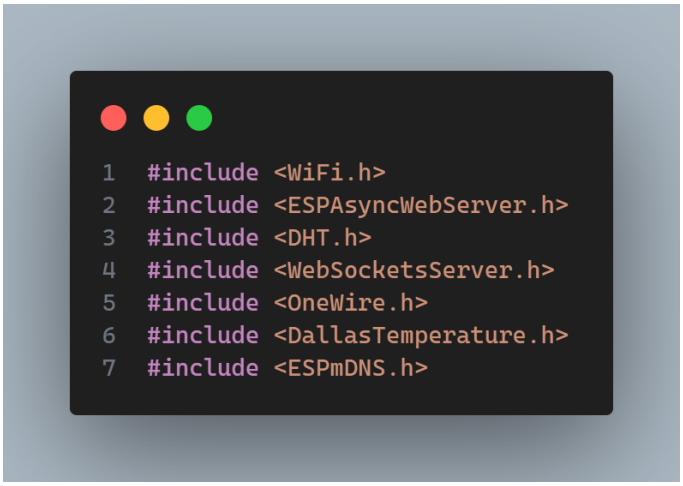
En este punto, la programación es realizada en el IDE de Arduino para utilizar las diversas herramientas que ofrece como por ejemplo el gestor de librerías para poder utilizarlas sin ningún inconveniente y el gestor de placas para poder cargar la programación al microcontrolador ESP32.

Para este caso se necesita la librería WiFi.h para poder establecer conexión a internet por conectividad WiFi, obtener la dirección IP del microcontrolador y verificar su estado de red. La librería ESPAsyncWebServer.h permite el manejo de múltiples clientes y procesar las peticiones. DHT.h corresponde al sensor DHT11 para su correcto uso. WebSocketsServer permite la conexión en tiempo real y el envío de

información sin la necesidad de recargar la página web. OneWire.h establece la comunicación para el uso del sensor DS18B20 y otros dispositivos onewire ,DallasTemperature.h permite leer las temperaturas leídas por el mismo sensor y finalmente ESPmDNS para otorgar un link de acceso local a la página web.

Figura 52

Librerías utilizadas en entorno de desarrollo de Arduino

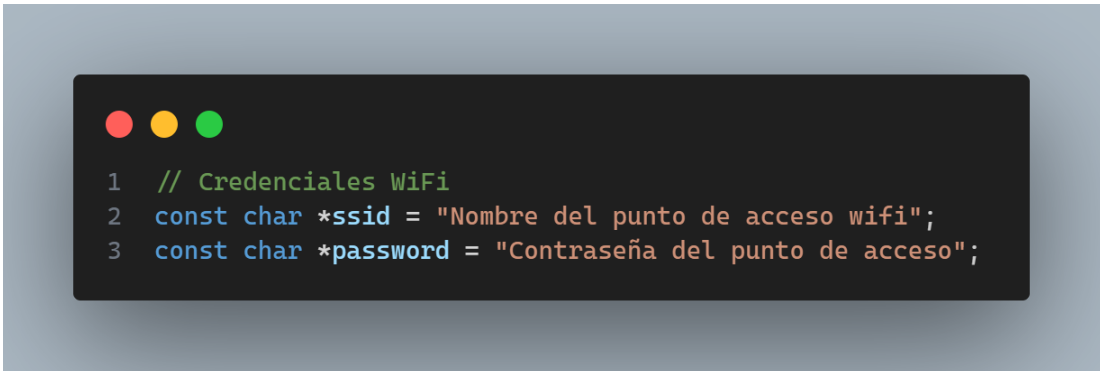


Fuente: Elaboración propia.

Lo siguiente es definir como constantes las credenciales wifi del punto de acceso de red para la conexión del servidor ESP32.

Figura 53

Constantes para guardar las credenciales de acceso a internet



Fuente: Elaboración propia.

Posterior a esto se definen las variables a usar durante la ejecución del programa y guardar en estas los datos obtenidos de los sensores, así como los pines a utilizar para cada sensor.

Figura 54

Inicialización de sensores y variables

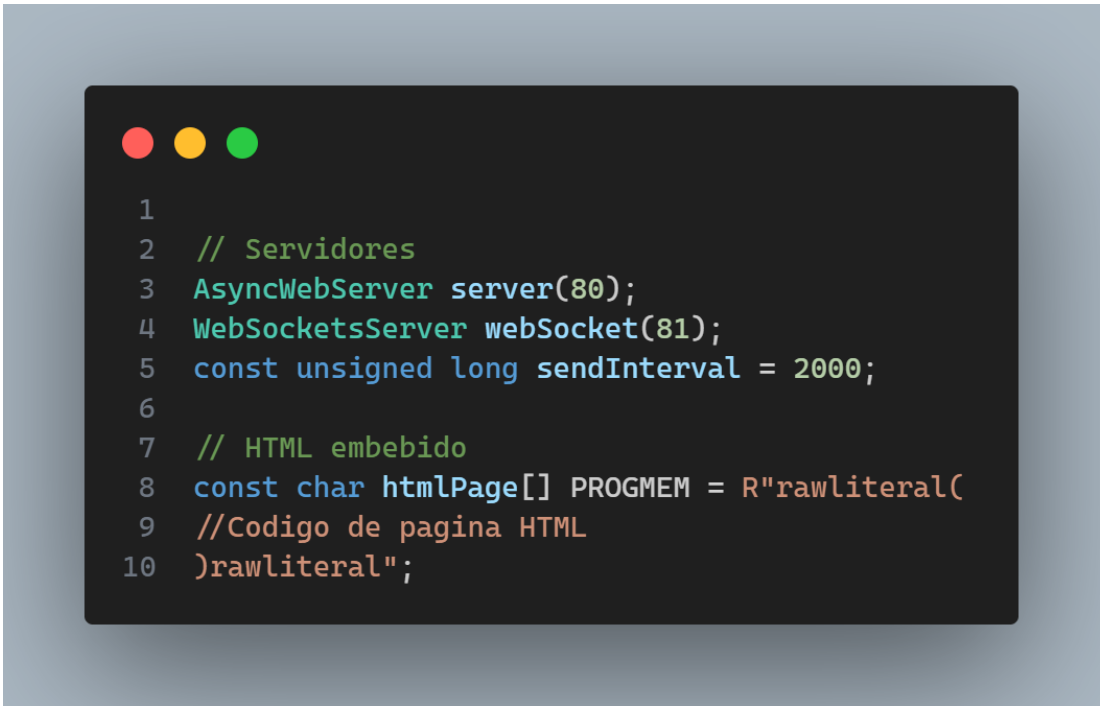
```
// Sensor de temperatura y humedad DHT11
const uint8_t dhtPin = 5;
DHT dht(dhtPin, DHT11);
float temperature = 0.0;
float humidity = 0.0;

// Sensor de temperatura DS18B20
const uint8_t oneWireBus = 4; // Pin de datos DS18B20
OneWire oneWire(oneWireBus);
DallasTemperature sensors(&oneWire);
float temperature2 = 0.0;

// Sensor de Flama YG1006 (Salida digital)
const uint8_t flamePin = 2; // Pin GPIO para el sensor de flama
String flameStatus = "OFF";

// Sensor de Movimiento HC-SR501
const uint8_t motionPin = 14; // Pin GPIO para el sensor de movimiento
String motionStatus = "NO DETECTADO";

// Sensor de Distancia HC-SR04
const uint8_t trigPin = 12; // Pin TRIG
const uint8_t echoPin = 13; // Pin ECHO
float distance = 0.0;

// Sensor de flujo YF-S201
const int flowPin = 25;
volatile int flowCount = 0; // Contador de pulsos
float flowRate = 0.0;
unsigned long lastTime = 0;
float totalFlow = 0.0;
```

Fuente: Elaboración propia.

Prosigue la configuración del servidor en el esp32 en lo que se asigna el puesto 80 para WebServer y el puerto 81 para WebSockets y se determina un intervalo de 2000 milisegundos que será usado como tiempo de esperar para captación de las temperaturas ya que estos sensores suelen funcionar mejor con un tiempo de censado determinado. También se define constante la cual contendrá el código HTML de la página web dando resultado a un HTML embebido. En la imagen de comenta “código de página HTML” para no repetir el código de la página anterior y en esta ocasión esta parte de la programación debe ser considerada como una sola parte y no estar entre tres documentos (HTML, CSS y JS) o variables.

Figura 55

Configuración de WebSockets y WebServer



```
1
2  // Servidores
3  AsyncWebServer server(80);
4  WebSocketsServer webSocket(81);
5  const unsigned long sendInterval = 2000;
6
7  // HTML embebido
8  const char htmlPage[] PROGMEM = R"rawliteral(
9  //Codigo de pagina HTML
10 )rawliteral";
```

Fuente: Elaboración propia.

Luego tenemos las funciones a consumir las cuales se realizar con el fin de reescribir código en el void loop. Primero tenemos a la función readDistance() de tipo float la cual sirve para poner en funcionamiento el sensor HC-SR04 y poder medir la distancia, además se establece el rango de medición y los valores resultantes a retornar por la función, en este caso retorna “-1” si esta fuera de rango de medición y retorna la variable “distanceCM” con el nuevo valor resultante. Después está la función pulseCounter la cual incrementa la variable “flowCount” cada vez que se genera un pulso por parte del sensor de flujo. La función leerFlujo() de tipo float nos actualiza dos variables, una para el flujo por minuto y otra para el total de flujo medido, esto considerando los interrupciones a realizar. Por ultimo de las funciones tenemos sendData(), la cual nos sirve para convertir las variables del programa, con los valores censados por los sensores, convertirlas en un string que es enviado como un JSON a por medio de webSocket al cliente o los clientes.

Figura 56

Declaración de funciones para leer sensores y enviar información en formato JSON

```
/ Función para leer distancia del HC-SR04
float readDistance() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    float duration = pulseIn(echoPin, HIGH);
    float distanceCm = duration * 0.0343 / 2; // Conversión a cm

    if (distanceCm ≥ 400 || distanceCm ≤ 2) {
        return -1; // Valor fuera de rango
    }
    return distanceCm;
}

// Interrupción para contar los pulsos del sensor
void IRAM_ATTR pulseCounter()
{
  flowCount++;
}


// Función para leer el flujo
float leerFlujo() {
  // Calcular el flujo cada segundo
  unsigned long currentTime = millis();

  if (currentTime - lastTime ≥ 1000) {
    // Deshabilitar interrupciones para leer el contador de pulsos
    noInterrupts();
    int pulses = flowCount;
    flowCount = 0;
    interrupts();

    // Calcular la tasa de flujo (en litros por minuto)
    flowRate = pulses / 7.5; // 7.5 pulsos por litro según el sensor YF-S201
    totalFlow += flowRate / 60.0; // Acumulamos el total de litros

    lastTime = currentTime;
  }

  return flowRate;
}

// Enviar datos al cliente WebSocket
void sendData() {
    String data = "{\"temperature\":" + String(temperature) +
                  ",\"humidity\":" + String(humidity) +
                  ",\"temperature2\":" + String(temperature2) +
                  ",\"flame\":\"" + flameStatus +
                  "\",\"motion\":\"" + motionStatus +
                  "\",\"distance\":" + String(distance) +
                  ",\"flowRate\":" + String(flowRate) +
                  ",\"totalFlow\":" + String(totalFlow) + "}";
    webSocket.broadcastTXT(data);
}

```

Fuente: Elaboración propia.

Con respecto al void setup tenemos que inicializamos la comunicación serial a 115200 baudios de depuración con los pines de los sensores de flama, movimiento y distancia definidos como entrada o salida respectivamente. luego se configura una interrupción para contar pulsos del sensor de flujo cuando detecta flancos descendentes. Se inicia la conexión a wifi con las credenciales utilizando las variables correspondientes utilizando un bucle while que no pare de intentar establecer conexión al punto de acceso si no hay red. Por monitor serial de imprime la dirección IP asignada al ESP32 como servidor. Se inicializa el sensor de tipo DHT. Con la función server.on() se busca en raíz y se solicita la página almacenada como HTML embebido devolviendo un status de 200 para el resultado encontrado. Con server.onNotFound devolvemos un status de 404 tipico de error 404 para sitio no encontrado. Server.begin() y Websocket.begin() inicializan el servidor y con Websocket.onEvent se captura la dirección IP del dispositivo cliente conectado y se notifica cuando se desconecta, ambas cosas por monitor serial.

Figura 57

Void Setup. Configuración inicial de comunicación, pines, wifi y servidor

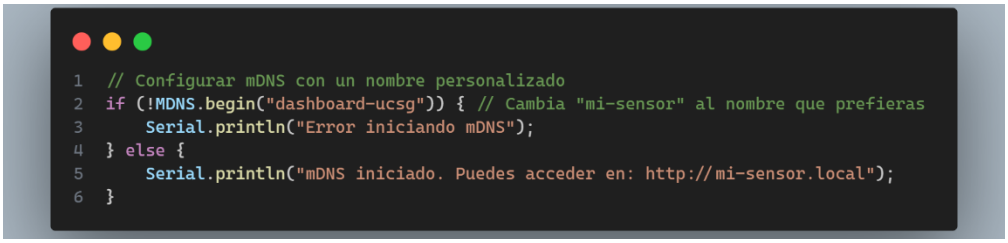
```
void setup() {
    Serial.begin(115200);
      // Configurar el pin del sensor de flama como entrada
    pinMode(flamePin, INPUT);
      // Configurar sensor de movimiento como entrada
    pinMode(motionPin, INPUT);
      // Configurar sensor de distancia ultrasonico
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);

      // Configurar el pin del sensor de flujo
    pinMode(flowPin, INPUT_PULLUP);

      // Configurar interrupción para contar los pulsos
    attachInterrupt(digitalPinToInterrupt(flowPin), pulseCounter, FALLING);

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
    }
    Serial.println("Conectado a la red, direccion IP:");
    Serial.println(WiFi.localIP());

    dht.begin();

    // Servidor Web
    server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
        request->send_P(200, "text/html", htmlPage);
    });

    server.onNotFound([](AsyncWebServerRequest *request) {
        request->send(404, "text/plain", "¡Página no encontrada!");
    });

    server.begin();

    // WebSocket
    webSocket.begin();
    webSocket.onEvent([](uint8_t num, WStype_t type, uint8_t *payload, size_t length) {
        switch (type) {
            case WStype_DISCONNECTED:
                Serial.printf("[%u] Desconectado!\n", num);
                break;
            case WStype_CONNECTED: {
                IPAddress ip = webSocket.remoteIP(num);
                Serial.printf("[%u] Conectado desde %d.%d.%d.%d\n", num, ip[0], ip[1], ip[2], ip[3]);
                break;
            }
        }
    });
}
```

Fuente: Elaboración propia.

Adicional, para otorgar un fácil acceso se con el uso de la librería mDNs un link de ingreso local medaiante una validación if. Con esto se logra un acceso rapido al apagina mediante el url “http://dashboard-ucsg.local”. Cabe mencionar que esta via de ingreso es permitido solo con navegadores con capacidad de trabajar con DNS que, generalmente lo tienes los navegadores de escritores pero no de dispositivos moviles.

Figura 58

Iniciación de mDNS para creación de ruta de fácil acceso



```
// Configurar mDNS con un nombre personalizado
if (!MDNS.begin("dashboard-ucsg")) { // Cambia "mi-sensor" al nombre que prefieras
    Serial.println("Error iniciando mDNS");
} else {
    Serial.println("mDNS iniciado. Puedes acceder en: http://mi-sensor.local");
}
```

Fuente: Elaboración propia.

Por último, void loop(), la sección de código que se ejecuta repetitivamente, se empieza con la función webSocket.loop() para la gestión de conexión y mensajes en tiempo real. Con la función de temporizador Millis y la variable prevMillis se compara si ha pasado el intervalo establecido en la variable sendInterval para poder leer todos los sensores y enviar la información en formato JSON usando la función sendData (), luego de cada envío el temporizados y prevMillis se reinician. Cabe resaltar que, en la lectura del sensor de flama se usa el operar ternario aplicando lógica inversa ya que este sensor devuelve 1 si no se detecta flama y 0 si se detecta flama, en booleano esto sería 0 para presencia de flama y 1 para flama no detectada. Esta lógica inversa adicional hace que se trabaje los valores de este sensor con lógica ordinaria, es decir 1 para flama detectada y 0 para flama no detectada.

Figura 59

Void Loop. Lectura de sensores según el intervalo de tiempo establecido y envío de información a la página con formato JSON

```
void loop() {
    webSocket.loop();
    static uint32_t prevMillis = 0;
    if (millis() - prevMillis ≥ sendInterval) {
        prevMillis = millis();

        //lectura DHT11
        temperature = dht.readTemperature();
        humidity = dht.readHumidity();

        // Lectura DS18B20
        sensors.requestTemperatures();
        temperature2 = sensors.getTempCByIndex(0);

        // Leer el estado del sensor de flama
        flameStatus = digitalRead(flamePin) == LOW ? "ON" : "OFF";

        // Leer el estado del sensor de movimiento
        motionStatus = digitalRead(motionPin) == HIGH ? "DETECTADO" : "NO DETECTADO";

        // Leer distancia HC-SR04
        distance = readDistance();

        // lectura de sensor de flujo
        leerFlujo();

        // Enviar JSON
        sendData();
    }
}
```

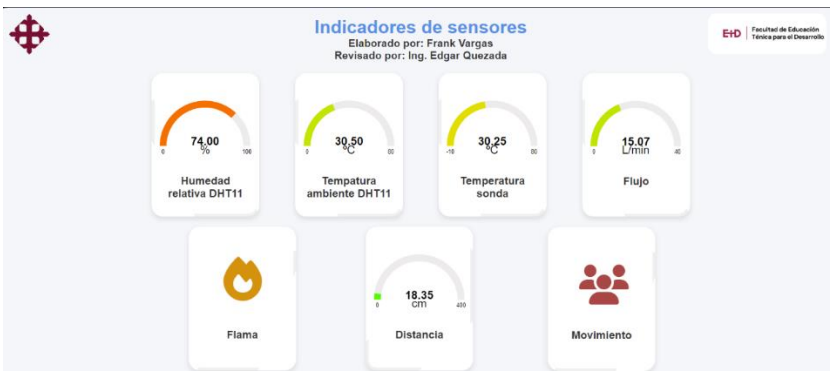
Fuente: Elaboración propia.

3.8 Resultados Obtenidos

Hasta el momento la programación ha permitido llevar lecturas físicas al mundo digital por medio de protocolos de comunicación hasta el internet donde se pueden visualizar los resultados obtenidos en la página web desarrollada.

Figura 60

Visualización de datos con todos los indicadores mostrando los valores obtenidos por los diferentes sensores



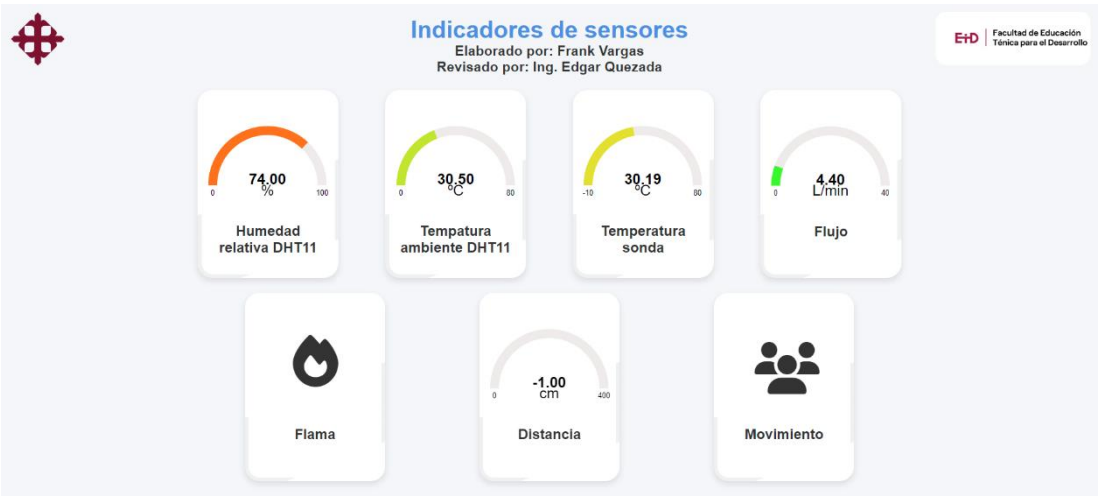
Fuente: Elaboración propia.

Para la prueba de este sistema de procedió a hacer contacto con el sensor DTH11 y el sensor DS18B20 para poder elevar la temperatura y humedad respectivamente. Para el sensor yg1006 se encendió un fosforo dentro del rango de medida para poder observar la animación en la web mientras que con el sensor HC-SR501 de procedió a realizar movimientos con las manos dentro del rango captura. Con el sensor HC-SR04 se procedió de manera similar que con el anterior puesto que se colocó la mano a varias distancias del sensor para verificar su correcto funcionamiento. Y por último con el sensor YF-S201 se sopló la entrada del sensor simulando el movimiento de un líquido a través de este para poder generar una señal medible.

Para los sensores digitales (flama y movimiento) se muestra una animación con color cuando se mide lo deseado, pero cuando no es así se muestra el mismo icono estático en color negro. Respecto al sensor de distancia. Si el objeto está fuera del rango de medición (4 metros), el sensor retornará el valor “-1”, y el sensor de flujo retornará “0” si no hay flujo.

Figura 61

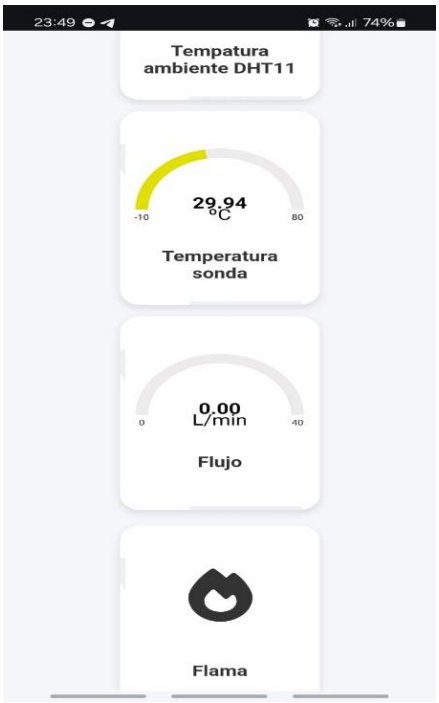
Visualización de página web con falta de detección de algunos sensores



Fuente: Elaboración propia.

Figura 62

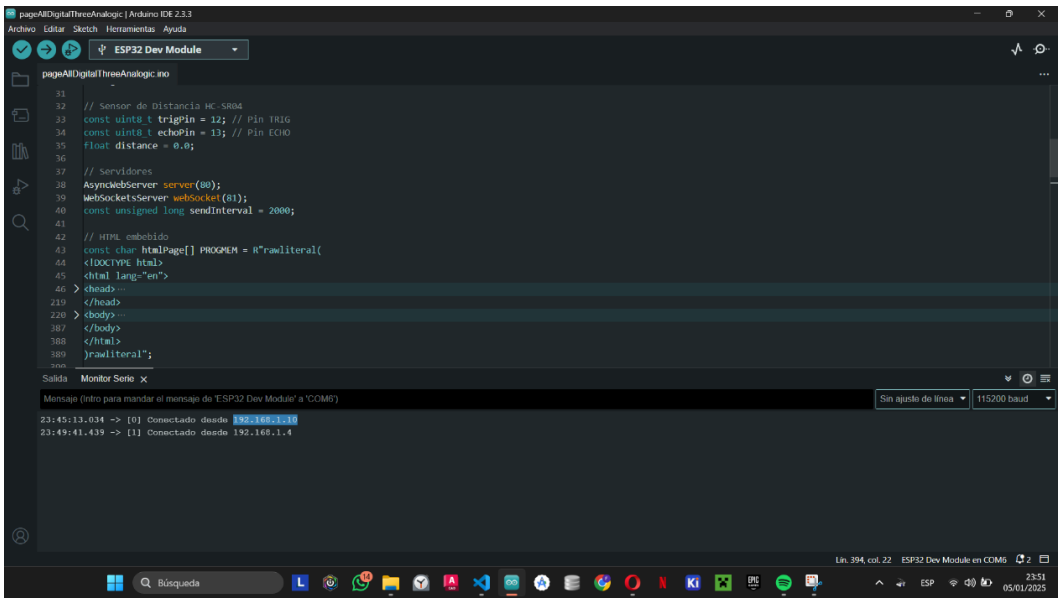
Visualización de página web desde dispositivo móvil



Fuente: Elaboración propia

Figura 63

Reconocimiento de dispositivo conectado y desconectado al servidor por medio de monitor serial



Fuente: Elaboración propia.

Capítulo 4: Conclusiones y Recomendaciones

4.1 Conclusiones

La Implementación del sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT concluye los siguiente:

- En referencia a rendimiento, el sistema funciona bien acorde a la cantidad de sensores y código HTML embebido aplicado para su correcto funcionamiento considerando su cantidad de memoria Flash de 4MB.
- En cuanto a conectividad, soporta muy bien la conexión de hasta 4 usuarios, esto debido a la carga con la que debe trabajar el ESP32.
- Debido a la falta de un servicio de host no es posible la visualización de la página web alojada en el internet global para el acceso desde cualquier lugar del mundo.
- Debido a sus propiedades, el microcontrolador puede funcionar como un nodo sensor inalámbrico para la adquisición de datos y enviarla a algún otro sitio además del cliente conectado al servidor, sin embargo, dependiendo del destino adicional de envío de información el microcontrolador puede incrementar la carga y ver un retraso en la visualización de la página web.

4.3 Recomendaciones

- Se recomienda adquirir un servicio de hosting económico para poder alojar el sitio web y visualizar las variables medidas desde cualquier parte del mundo, esto considerando realizar una API en la cual enviar los datos capturados y poder usarlos para la página. Además, un control de acceso de usuario o “log In” para seguridad. Además, el servicio de hosting puede brindar un dominio o url personalizado de acceso a la página.
- Si se desea utilizar como un nodo sensor poniendo más sensores en la tarjeta se recomienda utilizar microcontroladores adicionales (cuanto sean necesarios para la aplicación deseada) y de ser posible de mejores prestaciones para a fin de que uno sirva como nodo de adquisición de datos y procesamiento y otro que pueda servir como servidor HTTP para la visualización de datos en página web local.
- En el caso de adicionar un microcontrolador con mejores prestaciones. Se recomienda utilizar técnicas de minificación, para el uso de más sensores, puesto que esto reducirá el peso de archivos y programa a subir al ESP32 lo que se traduce en memoria con mejor aprovechamiento de uso.
- Se recomienda adquirir más microcontroladores para practicas del alumnado y familiarización con sistemas embebidos.

Bibliografía

Advances in Arithmetic Logic Units: A Comprehensive Review. (2024, septiembre 21). ResearchGate. *https://www.researchgate.net/publication/384113903_Advances_in_Arithmetic_Logic_Units_A_Comprehensive_Review*

Aguirre, T., & Alexander, J. (2024). Implementación de prototipos de monitoreo remoto basados en ESP32 y página web: Implementación de un prototipo de un sistema de monitoreo ambiental de temperatura, humedad, presión y calidad del aire, basado en ESP32 y página web. https://bibdigital.epn.edu.ec/handle/15000/25599

AIX 7.2. (2024, agosto 27). https://www.ibm.com/docs/es/aix/7.2?topic=protocol-tcpip-protocols

Armas, F. I. G., Peñarreta, K. S. J., & Jaramillo, J. J. B. (2023). *SISTEMA DE ALARMA Y APAGADO AUTOMÁTICO DE UN MOTOR COMO RESPUESTA A LA DETECCIÓN DE MONÓXIDO DE CARBONO EN NIVELES NOCIVOS PARA LA SALUD HUMANA EN EL HABITÁCULO DE UN VEHÍCULO*.

Arquitectura de computadoras: Modelos esenciales. (2023, agosto 3). Tiffin University. https://global.tiffin.edu/blog/que-es-arquitectura-de-computadoras

Arriola Salazar, A. (2024). Integración de WebSockets y WebRTC para la monitorización remota de terminales de autopago en una empresa de desarrollo de software.

B, G. (2019, enero 24). ¿Qué es CSS? *Tutoriales Hostinger*. https://www.hostinger.es/tutoriales/que-es-css

Behnke, I., & Austad, H. (2024). Real-Time Performance of Industrial IoT Communication Technologies: A Review. *IEEE Internet of Things Journal*, *11*(5), 7399-7410. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2023.3332507

Beijing Yaohuadechang Electronic Co., Ltd. (s.f.). *HSTS016I Datasheet*. Recuperado de https://www.datasheetbank.com/es/HSTS016I-doc-

YHDC

Betania V. (2022, febrero 9). ¿Qué es un servidor web y cómo funciona? *Tutoriales Hostinger*. https://www.hostinger.es/tutoriales/que-es-un-servidor-web

Bodnar, D. (2021). *¿Qué es TCP/IP?* ¿Qué Es TCP/IP? https://www.avg.com/es/signal/what-is-tcp-ip

CAPAS DEL MODELO TCPIP | Universidad del Azuay. (s. f.). Recuperado 25 de diciembre de 2024, de https://www.uazuay.edu.ec/sistemas/teleprocesos/modelo_tcpip/capas

Casellas, B. (2020, octubre 22). *Sistema de adquisición de datos con ESP32*. https://upcommons.upc.edu/handle/2117/344400

Ccora, A., & Jarlin, M. (2020). Implementación de un sistema electrónico para conocer la correlación entre el Costo de alquiler y el consumo de agua en una habitación. *Repositorio Institucional - UTP*. https://repositorio.utp.edu.pe/handle/20.500.12867/4204

Cimpleo. (2020, abril 23). Arduino pH-meter using PH-4502C[EN]. https://cimpleo.com/blog/arduino-ph-meter-using-ph-4502c/

Crockford, D. (2020). JavaScript: The Good Parts. O'Reilly Media.

Dallas Semiconductor. *DS18B20: Programmable resolution 1-wire digital thermometer*. Dallas Semiconductor. Recuperado el 10 de diciembre de 2024, de https://www.analog.com/media/en/technical-documentation/data-sheets/DS18B20.pdf

Danish, K. (2023). Design of Multi-sensor Walking Aid for Blind People.

Dispositivos IoT en el entorno empresarial | Empresas | INCIBE. (2021). https://www.incibe.es/empresas/blog/dispositivos-iot-el-entorno-empresarial

Eras, A. P. P., Bueno, J. I. R., Gavilanes, M. D. G., & Núñez, E. F. H. (2024). Manual de Iniciación en el Uso y Aplicaciones Básicas de la Tarjeta ESP32.: Introduction Manual for the Use and Basic Applications of the ESP32 Card. *Revista Científica Multidisciplinar G-nerando*, *5*(2), Article

2. https://doi.org/10.60100/rcmg.v5i2.254

Flanagan, D. (2020). JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language. O'Reilly Media.

Flores, X., & Guadalupe, J. (2024). Desarrollo de una red inalámbrica ad hoc para realizar tareas de monitoreo de temperatura y humedad. *REPOSITORIO NACIONAL CONACYT*.

Fonseca Yupa, J. D., & Soria Badillo, D. A. (2020). *Diseño e implementación de control domótico (con sistemas embebidos) para conectarse con aplicaciones adaptivas basados en IOT* [bachelorThesis]. https://dspace.ups.edu.ec/handle/123456789/18986

García, V., & Alberto, J. (2022). *Desarrollo de un sistema de supervisión IoT para un banco de pruebas utilizando autómata programable PLC S7-1200 y LabVIEW.* http://repositorio.ucsg.edu.ec/handle/3317/19182?mode=full

Generalidades del protocolo HTTP - HTTP | MDN. (2024, julio 8). https://developer.mozilla.org/es/docs/Web/HTTP/Overview#arquitectura_de_los_sistemas_basados_en_http

Getting Started with Arduino IDE 2 | Arduino Documentation. (s. f.). Recuperado 3 de noviembre de 2024, de https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/

Gómez, C., & Daniel, A. (2024). *Implementación de un prototipo de sistema de alarma residencial basado en ESP32 y página web.* https://bibdigital.epn.edu.ec/handle/15000/25836

Gordillo, R., & Michel, A. (2024). *Diseño de un módulo de monitoreo para un laboratorio remoto mediante dispositivos IoT*. http://www.riaa.uaem.mx/xmlui/handle/20.500.12055/4620

Herrera Chávez, D. (2020). Diseño e implementación de un prototipo de seguridad para control domótico basado en IoT bajo ambientes de dispositivos móviles con Android [bachelorThesis, Quito, 2020.]. https://bibdigital.epn.edu.ec/handle/15000/20857

HTML Semántico: Qué Es y Cómo Usarlo Correctamente. (2023). Semrush

Blog. https://es.semrush.com/blog/html-semantico/null

HTTP | MDN. (2023, julio 24). https://developer.mozilla.org/es/docs/Web/HTTP

IBM Maximo Monitor 8.8 to 9.0 continuous delivery. (2024, junio 25). https://www.ibm.com/docs/es/masv-and-l/maximo-monitor/continuous-delivery?topic=security-mqtt-messaging

Ikiss, J. (2020). *Sistema de adquisición de datos con ESP32* (Bachelor's thesis, Universitat Politècnica de Catalunya).

Industries, A. (s. f.). *DHT11 basic temperature-humidity sensor + extras*. Recuperado 10 de diciembre de 2024, de https://www.adafruit.com/product/386

International Roadmap for Devices and Systems (IRDS™) 2020 Edition—IEEE IRDS™. (s. f.). Recuperado 17 de noviembre de 2024, de https://irds.ieee.org/editions/2020

IoT Industrial y sus principales aplicaciones. (2022, agosto 2). UNIR. https://www.unir.net/revista/ingenieria/iot-industrial/

IoT Projects. (2024, enero 1). *Arquitecturas IoT - ¿Cuáles son las más comunes?* https://iotprojects.io/arquitecturas-iot/

JESÚS, P. P. (2020). Internet de las cosas (IOT) con ESP. Manual práctico. Ediciones Paraninfo, S.A.

Lima, D., & Edmundo, R. (2024). Diseño e implementación de un prototipo de un sistema IOT para la identificación de actos delictivos y ejecución de acciones disuasorias mediante el uso de la tarjeta ESP32-CAM, servidor web y algoritmos de aprendizaje automático. https://repositorio.ucsm.edu.pe/items/d6e7332d-c73c-470f-afde-143d515efdbd

Mahedero Biot, F. (2020). *Desarrollo de una aplicación IoT para el envío de imágenes mediante el protocolo MQTT* [Proyecto/Trabajo fin de carrera/grado, Universitat Politècnica de València]. https://riunet.upv.es/handle/10251/152408

MDN contributors. (2024, diciembre 21). *CSS básico—Aprende desarrollo web | MDN*. https://developer.mozilla.org/es/docs/Learn_web_development/Getting_started/Your_first_website/Styling_the_content

MDN Web Docs. (2021). *HTML: HyperText Markup Language*. https://developer.mozilla.org/docs/Web/HTML

Microsoft. (2024). *Introduction to Visual Studio Code*. Microsoft Learn. https://learn.microsoft.com/en-us/training/modules/introduction-to-visual-studio-code

Micucci, M. (2024, mayo 10). *Anatomía de los Paquetes TCP/IP desde la Perspectiva de la Ciberseguridad*. https://www.welivesecurity.com/es/recursos-herramientas/tcp-ip-protocol-perspectiva-ciberseguridad/

Milla-Herrero, Z. (2023). *Desarrollo de prácticas con entornos virtuales en el ámbito de las telecomunicaciones*. https://crea.ujaen.es/handle/10953.1/21103

Modelo TCP/IP vs. Modelo OSI | Similitudes y Diferencias. (2022). Fortinet. https://www.fortinet.com/lat/resources/cyberglossary/tcp-ip-model-vs-osi-model.html?utm_source=chatgpt.com

Módulo Sensor de luz ultravioleta (UV) GUVA-S12SD. (2023). Naylamp Mechatronics - Perú. https://naylampmechatronics.com/sensores-luz-y-sonido/1221-modulo-sensor-de-luz-ultravioleta-uv-guva-s12sd.html

MÓDULO SENSOR MQ-4 GAS METANO – Grupo Electrostore. (2023, junio 30). https://grupoelectrostore.com/shop/sensores/gas/modulo-sensor-mq-4-gas-metano/

MQTT - The Standard for IoT Messaging. (2024). https://mqtt.org/

Naylamp Mechatronics. (2024). *Sensor de presión BMP280*. Naylamp Mechatronics - Perú. https://naylampmechatronics.com/sensores-posicion-inerciales-gps/358-sensor-de-presion-bmp280.html

Novatronic. (2020). *Sensor de temperatura Digital DS18B20 de acero inoxidable sumergible – Novatronic*.

https://novatronicec.com/index.php/product/sensor-de-temperatura-digital-ds18b20-de-acero-inoxidable-sumergible/

O'Neill, L. (2020). Modern CSS: Master the Art of Web Styling with Advanced CSS Techniques. Addison-Wesley.

Ochoa, L., & Jeanella, D. (2024). Revisión de literatura sobre el uso del internet de las cosas enfocada a la domótica.

Perera, A. (2020, septiembre 6). Microcontroladores, la base de Arduino. *AutomatismosMundo - Electricidad, Automatización y Domótica para todos*. https://automatismosmundo.com/microcontroladores-la-base-de-arduino/

Perez Campoy, G. (2023). Estudio de la viabilidad de dispositivos IMU lowcost para usos de navegación y rastreo con Arduino/ESP32, y construcción de su prototipo [Bachelor thesis, Universitat Politècnica de Catalunya]. https://upcommons.upc.edu/handle/2117/396584

Pico-series Microcontrollers—Raspberry Pi Documentation. (s. f.). Recuperado 24 de diciembre de 2024, de https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html

(PDF) Cloud Computing and Internet of Things: Recent Trends and Directions. (s. f.). En *ResearchGate*. https://doi.org/10.1007/978-3-031-05528-7_1

¿Qué es el Internet de las cosas y cómo funciona? (s. f.). Recuperado 24 de diciembre de 2024, de https://www.redhat.com/es/topics/internet-of-things/what-is-iot

¿Qué es el MQTT? - Explicación del protocolo MQTT - AWS. (s. f.). Amazon Web Services, Inc. Recuperado 25 de diciembre de 2024, de https://aws.amazon.com/es/what-is/mqtt/

¿Qué es un microcontrolador? | *IBM*. (2024, junio 4). https://www.ibm.com/mx-es/think/topics/microcontroller

¿Qué es un servidor WEB? - Aprende desarrollo web | *MDN*. (2024, diciembre 21). https://developer.mozilla.org/es/docs/Learn_web_development/Howto/

Web_mechanics/What_is_a_web_server

¿Qué es una CPU? Explicación de la unidad central de procesamiento: AWS. (2024). Amazon Web Services, Inc. *https://aws.amazon.com/es/what-is/cpu/*

Registers of Embedded Microcontrollers- Explained. (s. f.). Recuperado 17 de noviembre de 2024, de https://www.learningaboutelectronics.com/Articles/Registers-embedded-microcontrollers.php

Román, A., Román Herrera, J., Sandoval, S., Andrade, M., & Ramos, E. (2023). *Internet de las cosas: teoría y práctica*. Universidad de Colima.

Sampedro, I. M., González, R. Á., Ramírez, R. L. M., & Gálvez, A. M. S. Implementación del módulo ESP32 como herramienta para el desarrollo de prácticas enfocadas al IoT.

Sanz Martín, D. (2022). Sistema de adquisición de datos para un monitor de neutrones basado en ESP32 y tecnología IoT.

Sensor MQ-7 gas CO Monóxido de carbono. (2024). Naylamp Mechatronics - Perú. https://naylampmechatronics.com/sensores-gas/74-sensor-mq-7-gas-monoxido-de-carbono-co.html

Sigma Elektronik. (2024). *Sensor de corriente no invasivo 300A HSTS016I*. Sigma Elektronik. https://www.sigmagye.net/product-page/sensor-de-corriente-no-invasivo-300a-hsts016i

Sigma Elektronik. (2024). *Sensor infrarrojo de movimiento PIR*. Sigma Elektronik. https://www.sigmagye.net/product-page/sensor-infrarrojo-de-movimiento-pir

Silicon—Raspberry Pi Documentation. (2024). https://www.raspberrypi.com/documentation/microcontrollers/silicon.html

Tapia, T., & Enrique, J. (2024). Repotenciación de un módulo interactivo a través de Arduino con comunicación de Radioenlace a IOT y Node Red para integración a Django en la visualización y almacenamiento de datos (SQL SERVER) [bachelorThesis].

https://dspace.ups.edu.ec/handle/123456789/28131

Universitat Politècnica De València, E. (2014). Universitat Politècnica de València. *Ingeniería del agua*, *18*(1), ix. https://doi.org/10.4995/ia.2014.3293.

Valenzuela-Ramírez, S. G., Contreras-Basurto, A., & Rivera-Landeros, E. A. (2024). Práctica sensor ultrasónico y buzzer con Arduino. *Ingenio y Conciencia Boletín Científico de la Escuela Superior Ciudad Sahagún*, *11*(21), Article 21. https://doi.org/10.29057/escs.v11i21.11728

W3C. (2020). *HTML Review Draft — Published 29 January 2020 is a W3C Recommendation*. https://www.w3.org/news/2021/html-review-draft-published-29-january-2020-is-a-w3c-recommendation/

Xu, X., & Peng, X. (2023). Memory-Centric Microcontroller (MCU) Architecture Design and Implementation. *2023 9th Annual International Conference on Network and Information Systems for Computers (ICNISC)*, 107-110. https://doi.org/10.1109/ICNISC60562.2023.00080

Anexos

Figura 64

Código de programación del proyecto y manual

Fuente: Elaboración propia

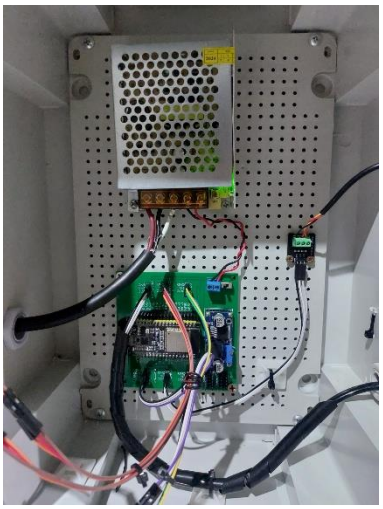
Figura 65

Sistema armado en tablero y energizado para la visualización de resultados en página web

Fuente: Elaboración propia

Figura 66

Energizado del sistema y correcto funcionamiento para pruebas de visualización de datos



Fuente: Elaboración propia

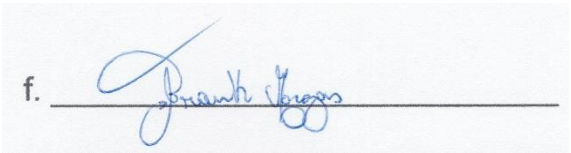
DECLARACIÓN Y AUTORIZACIÓN

Yo, **Vargas Castro Frank Allan,** con C.C: **0953594546** autor del trabajo de titulación: **Implementación de sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT,** previo a la obtención del título de **ingeniero en electrónica y automatización** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 20 de febrero de 2025



Nombre: **Vargas Castro, Frank Allan**

C.C: **0953594546**

Presidencia de la República del Ecuador

Plan Nacional de Ciencia, Tecnología, Innovación y Saberes

SENESCYT
Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TEMA Y SUBTEMA:	Implementación de sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT.		
AUTOR(ES)	Vargas Castro, Frank Allan.		
REVISOR(ES)/TUTOR(ES)	Ing. Quezada Calle, Edgar Raúl M. Sc.		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad De Educación Técnica Para El Desarrollo		
CARRERA:	Ingeniería en electrónica y automatización		
TITULO OBTENIDO:	Ingeniero en electrónica y automatización		
FECHA DE PUBLICACIÓN:	**20 de febrero de 2025**	**No. DE PÁGINAS:**	82
ÁREAS TEMÁTICAS:	**Internet de las cosas, Diseño electrónico, Sistemas embebidos, Desarrollo Web.**		
PALABRAS CLAVES/ KEYWORDS:	**ESP32, IoT, Embebidos, HTML, CSS, Javascript, WebServer, WebSockets.**		

RESUMEN/ABSTRACT (150-250 palabras):
El desarrollo de un sistema de lectura de variables con sensores utilizando sistemas embebidos en IoT, utilizará el microcontrolador ESP-WROOM-32 integrando diferentes sensores de uso domótico en una tarjeta electrónica de tamaño reducido. En cuanto a su firmware este microcontrolador alojará una página web desarrollada en código HTML, CSS y Javascript; este resultado se aplica en forma de HTML embebido alojado en espacio de memoria del dispositivo, adicional a este el ESP32 captará las mediciones de los sensores para mostrar los resultados en indicadores contenidos en la página web. Con la comunicación wifi-integrada utilizará los protocolos WebServer y WebSocket para el manejo de múltiples clientes locales y comunicación en tiempo real para la visualización de la información obtenida. Debido a las propiedades de CSS la página tiene la particularidad de poseer un diseño responsive por lo que su visualización se ajusta a pantallas más pequeñas como celulares o tablets desde el navegador web. Con la implementación de este sistema de monitorización IoT se espera que sea una herramienta educativa de aplicaciones prácticas para estudiantes de las carreras técnicas de la Facultad de Educación Técnica para el Desarrollo de la UCSG puesto que el módulo electrónico propuesto permite el uso de los GPIOS no utilizados en el presente trabajo de integración.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593-96 138 1717	E-mail: frank_vargas2000@outlook.es Frank.vargas02@cu.ucsg.edu.ec
CONTACTO CON LA INSTITUCIÓN (C00RDINADOR DEL PROCESO UTE)::	**Nombre:** Inge. Ricardo Xavier Ubilla González M. Sc.	
	Teléfono: +593-99 952 8515	
	E-mail: ricardo.ubilla@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA

Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	