

Implementación de un Sistema de Monitoreo de Variables Ambientales utilizando el Sensor BME280 y un Microcontrolador ESP32

Gutiérrez Landero Ian (202240288)
Ortiz Medina Miguel Ángel (202235022)
Cruz Vaquerizo Rodolfo Alejandro (202221827)
Figueroa Salgado Jessica (202238778)
Bojorgez Ordaz José Gael (202236751)
Martínez Mora Juan Carlos (202263488).
Benemérita Universidad Autónoma de Puebla
Facultad de ciencias de la electrónica.
Puebla, México.

La introducción y justificación esta bien presentada y citada

Resumen—Este proyecto presenta la implementación y diseño de un sistema de monitoreo de variables ambientales (temperatura, presión y humedad) utilizando un microcontrolador ESP32 y un sensor **BME280**. Los datos recolectados son mostrados en un display LCD 20x4 con un controlador I2C, además, los datos recolectados son almacenados localmente en la memoria del microcontrolador en un archivo de texto y también se exploraron otras alternativas relacionadas con las IoT para el almacenamiento de estos datos. Este sistema tiene una gran variedad de aplicaciones, como el monitoreo de las condiciones meteorológicas en tiempo real, con enfoque de la adquisición de datos por un bajo costo. Los resultados obtenidos muestran la eficiencia del sistema para medir estas variables ambientales con cierto grado de precisión de manera continua.

Index Terms—ESP32, BME280, Sensores ambientales, I2C, Almacenamiento de datos, Microcontrolador.

I. OBJETIVOS

1. Objetivo General

- Diseñar e implementar un sistema de recolección de datos ambientales por un periodo de 24 hrs utilizando un microcontrolador ESP32 y el sensor BME280, capaz de medir temperatura, presión y humedad, mostrando los datos en un display LCD y almacenándolos en un archivo de texto para un análisis posterior.

2. Objetivo Específicos

- Configurar el microcontrolador ESP32 para establecer comunicación con el sensor BME280 y poder recolectar los datos medidos.
- Añadir un display LCD 20x4 para poder visualizar los datos recolectados del sensor.
- Implementar un sistema de almacenamiento simple y seguro en el microcontrolador para el almacenamiento de los datos recolectados.
- Estudiar las variaciones de estas variables en un entorno específico para caracterizar las condiciones ambientales.

II. INTRODUCCIÓN.

En el presente proyecto, nos propusimos diseñar e implementar un sistema de adquisición de datos capaz de medir y registrar la presión atmosférica, temperatura y humedad relativa utilizando el sensor BME280. Estos tres parámetros son fundamentales en la caracterización de las condiciones ambientales, ya que influyen directamente en fenómenos como la calidad del aire, el comportamiento del clima, e incluso en la toma de decisiones en diversas aplicaciones de Internet de las Cosas (IoT).

El sensor BME280 es ampliamente utilizado en proyectos de IoT por su precisión, bajo consumo energético y su capacidad de comunicación mediante las interfaces I2C y SPI. En nuestro caso, utilizamos la interfaz I2C para conectar el sensor a un microcontrolador ESP32, lo que permitió desarrollar un sistema compacto y eficiente para la recolección y visualización de datos.

El objetivo principal de este proyecto es medir y monitorear las tres variables mencionadas a lo largo de un periodo de 24 horas de manera continua, mostrando los valores en tiempo real en un display LCD y almacenándolos para su posterior análisis. Este tipo de sistema tiene aplicaciones en diferentes entornos, como la monitorización ambiental en espacios interiores y exteriores, y el control climático de invernaderos.

También, se exploran herramientas de inteligencia artificial (IA) para el análisis de los datos recogidos, con el fin de detectar patrones que mejoren la interpretación de las condiciones ambientales. Con este proyecto, buscamos entender mejor el desarrollo de sistemas de adquisición de datos y su aplicabilidad en la resolución de problemas reales dentro del ámbito de la ingeniería y la tecnología.

III. JUSTIFICACIÓN

El monitoreo de variables ambientales como la presión atmosférica, la temperatura y la humedad relativa es fundamental para entender cómo varían las condiciones del entorno en diferentes escenarios. Estos parámetros no solo influyen en el

comportamiento del clima, sino que también afectan múltiples áreas como la meteorología, la construcción y el bienestar humano.

El desarrollo de un sistema de adquisición de datos basado en el sensor BME280 nos permite medir estas variables de forma continua y precisa, lo que facilita la detección de fluctuaciones y patrones que pueden ser de gran utilidad para el análisis ambiental. Además, la capacidad de almacenar y visualizar los datos en tiempo real es un paso importante en la implementación de soluciones prácticas para aplicaciones futuras.

A través de este proyecto, buscamos fortalecer nuestras competencias en la instrumentación y programación, además de **aplicar conceptos de mecánica de fluidos en un contexto real**. Al trabajar con herramientas de medición y análisis de datos, estamos desarrollando habilidades que serán esenciales en el campo de la ingeniería y que nos permitirán enfrentar problemáticas relacionadas con el monitoreo ambiental y el control de condiciones en diferentes escenarios.

IV. MARCO TEÓRICO

Se abordarán tres variables fundamentales para la caracterización ambiental: a presión atmosférica, la temperatura y la humedad relativa. A continuación, se describen algunos conceptos clave relacionados con estas variables y los dispositivos utilizados para la recolección de estos datos.

Presión

La presión en fluidos es la fuerza que un fluido, ya sea un líquido o gas, ejerce de manera perpendicular sobre una superficie por unidad de área. Este concepto, fundamental en la mecánica de fluidos, se origina por el movimiento y la interacción de las partículas del fluido.[1]

La presión en un fluido tiene aplicaciones en fenómenos como la flotación y el flujo de fluidos a través de tuberías, además de ser esencial en el diseño de estructuras hidráulicas. Para medirla, se utilizan dispositivos como manómetros y barómetros.[1]

Presión atmosférica

La presión atmosférica es la fuerza que ejerce la columna de aire de la atmósfera sobre la superficie terrestre en un punto determinado. Esta fuerza varía en función de la altitud: cuanto mayor es la altitud, menor es la presión atmosférica, y viceversa. La mayor presión se registra a nivel del mar, que sirve como referencia para la presión atmosférica estándar.[2]

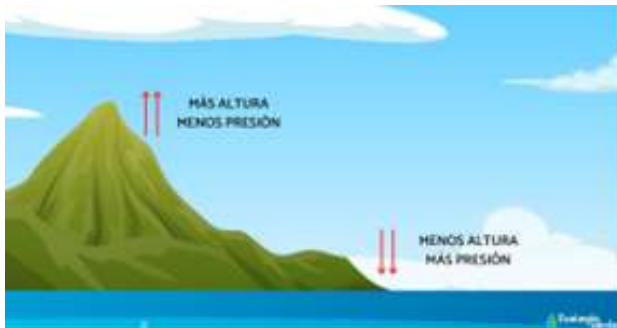


Figura 1. Variación de la presión atmosférica dependiendo la altitud.

Unidades de presión atmosférica

La presión atmosférica se mide en varias unidades, siendo la más utilizada en el Sistema Internacional el Pascal (Pa) o el hectopascal (hPa). También se emplean unidades como el bar, el milibar (mb), la atmósfera (atm), el milímetro de mercurio (mm Hg) y Torricellis (Torr).[2]

Fórmula de la presión atmosférica

Calcular la presión atmosférica o barométrica se rige por los principios de la ecuación fundamental hidrostática. Veamos a continuación. [2]

$$P = \rho \cdot g \cdot h$$

En esta fórmula,

- P es la presión ejercida en un punto del fluido.
- ρ es la densidad del fluido.
- g es la aceleración de la gravedad.
- h es la altura o profundidad.

El marco teórico presenta de forma puntual las variables que se van a medir con el sistema de adquisición. De esta manera se relaciona de forma adecuada la U3 de Mec de fluidos con la aplicación de monitoreo de variables atmosféricas

Temperatura

La temperatura nos permite conocer el nivel de energía térmica con que cuenta un cuerpo. Las partículas que poseen los cuerpos se mueven a una determinada velocidad, por lo que cada una cuenta con una determinada cinética. El valor medio de dicha energía cinética E_C está directamente relacionado con la temperatura del cuerpo. Así, a mayor energía cinética media de las partículas, mayor temperatura y a menor energía cinética media, menor temperatura.[3]

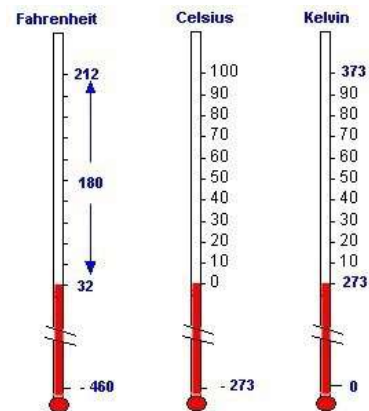


Figura 2. Distintas unidades de medida de la temperatura.

Humedad

La humedad refiere a la cantidad de agua integrada a un determinado cuerpo o sistema, ya sea en estado líquido o gaseoso. En meteorología, hablamos de humedad del aire para referirnos al vapor de agua que este contiene, pero también puede utilizarse el concepto para indicar, por ejemplo, qué cantidad de agua por unidad de volumen contiene el suelo (humedad del suelo).[4]

El agua está presente en cada organismo vivo (animal o vegetal) y hace posible todos los procesos biológicos del planeta, por lo que se considera una sustancia fundamental para la vida tal

y como la conocemos. También es de gran importancia para muchas actividades de los seres humanos, como la ganadería, la agricultura o la industria. Y, por si fuera poco, su presencia es un factor regulador del clima.[4]

La humedad del aire puede interpretarse de diferentes maneras. Con la misma terminología podemos referirnos a la concentración de vapor de agua en el aire, a la cantidad de vapor presente en una determinada masa de aire seco, o a la relación entre el vapor de agua que contiene el aire y el máximo que este puede contener. Por tal motivo, se diferencian distintos tipos de humedad:

- **Humedad absoluta:** Es la cantidad de vapor de agua presente en un determinado volumen de aire. En otras palabras, es la concentración real de vapor de agua en el aire. Se expresa en $\frac{g}{m^3}$.
- **Humedad específica:** Es la masa total de vapor de agua contenida en 1 kg de aire seco. Se expresa en $\frac{g}{kg}$.
- **Humedad relativa:** Es la relación entre la cantidad de vapor de agua presente en una parcela de aire, y la cantidad necesaria para que ésta se sature sin variar la temperatura. Se expresa con un número porcentual del 0 al 100 (mientras más alto, mayor humedad). Es el tipo de humedad que más se utiliza en meteorología, y depende tanto del vapor de agua presente como de la temperatura del aire, dado que esta condiciona su capacidad para almacenar vapor. Para esta práctica estaremos utilizando la Humedad relativa.[4]

Sensor BME280

Para la caracterización de datos utilizamos el sensor BME280, el cual integra en un solo dispositivo sensores de presión atmosférica, temperatura y humedad relativa, con gran precisión, bajo consumo energético y un formato ultra compacto, lo que lo convierte en una excelente opción para aplicaciones de Internet de las Cosas (IoT).[5]

Basado en tecnología BOSCH piezo-resistiva con gran robustez EMC, alta precisión y linealidad, así como con estabilidad a largo plazo. Se conecta directamente a un microcontrolador a través de I2C o SPI. [5]



Figura 3. Sensor BME280.

La etapa de sensado de presión es similar al sensor BMP280, y por lo tanto es superior al BMP180. En cuando al sensor de hu-

medad relativa presenta un desempeño sobresaliente comparado a los sensores DHT22 o DHT21. [5]

Este tipo de sensores pueden ser utilizados para calcular la altitud con gran precisión (barómetro), por lo que es un sensor muy utilizado en sistemas de Autopiloto para Drones (UAVs) entregando medidas de altitud con una precisión de hasta 1m. Otras aplicaciones son: **Monitoreo de clima, Internet de las Cosas, Monitor de salud/fitness, Automatización del hogar o Domótica y Aire acondicionado.** [5]

Microcontrolador ESP32



Es importante especificar el modelo de la ESP32 y la versión de MicroPython que utilizaron.

Nota: Es importante para que su trabajo pueda ser replicado en otra parte del mundo

Figura 4. Estructura del microcontrolador ESP32.

ESP32 es la denominación de una familia de chips SoC (System on a Chip) de bajo costo y consumo de energía, contando con tecnología WiFi y Bluetooth. Es un microcontrolador de bajo costo y bajo consumo que incluye conectividad WiFi y Bluetooth, lo que lo hace ideal para proyectos de IoT. Este microcontrolador puede manejar una amplia gama de sensores y módulos, lo que permite su uso en aplicaciones que van desde la transmisión de datos hasta el control de dispositivos. Con una corriente de reposo baja y una capacidad de procesamiento eficiente, el ESP32 es una solución robusta para sistemas embebidos.

MicroPython

MicroPython es una implementación ágil y eficiente del lenguaje de programación Python 3, que incluye un pequeño subconjunto de la biblioteca estándar de Python y esta optimizado para ejecutarse en microcontroladores y entornos restringidos.

MicroPython es un compilador y entorno de ejecución Python completo que se ejecuta en hardware. Obtiene un mensaje interactivo (REPL) para ejecutar comandos de inmediato, junto con la capacidad de ejecutar e importar scripts desde el sistema de archivos integrado. El REPL tiene historial, autocompletado con tabuladores, sangría automática y modo de pegar para una excelente experiencia de usuario. Este lenguaje emplea muchas técnicas de codificación avanzadas y muchos trucos para mantener un tamaño compacto y al mismo tiempo tener un conjunto completo de funciones, algunos de los elementos mas destacados son: Altamente configurable, Soporte para muchas arquitecturas, amplio conjunto de pruebas, cobertura de código, tiempo de inicio rápido, etc. [7].

Internet de las cosas (IoT)

el Internet de las Cosas (IoT) es un término que hace referencia a la interconexión de objetos físicos por medio del internet, permitiendo la recopilación, y procesamiento de datos en tiempo real. Un sistema IoT, todos los dispositivos que lo conforman están conectados a una red para compartir información entre sí y lograr una automatización del proceso al que van dirigidos, esto se implementa a distintos campos de la industria e ingeniería.

Una de las piezas fundamentales de las IoT es su capacidad para recolectar datos del entorno que los rodea, esto con ayuda de los sensores. Estos datos, desde luego, son compartidos por protocolos de comunicación Wifi o Bluetooth a un servidor o nube, donde son almacenados para su posterior análisis.



Figura 5. Internet de las Cosas (IoT).

En este proyecto, el sensor BME280 y el microcontrolador ESP32 son el pilar fundamental de todo el sistema IoT, y bien es cierto que esta parte no fue requerida por el docente, al equipo se le hizo muy interesante lograr agregar esta característica que por supuesto, agrega un plus a todo el trabajo en si. El sensor BME280 recolecta datos de temperatura, presión atmosférica y humedad relativa, tres variables críticas para la caracterización de las condiciones ambientales. El ESP32, con su capacidad de conectividad Wi-Fi, permite la transmisión de estos datos en tiempo real para el almacenamiento de los mismos para un análisis posterior.

V. MATERIALES Y EQUIPO

1. Sensor BME280.
2. Microcontrolador ESP32.
3. Tablilla de experimentos Protoboard .
4. LCD 20x4 con módulo I2C .
5. Jumpers.
6. Computadora.

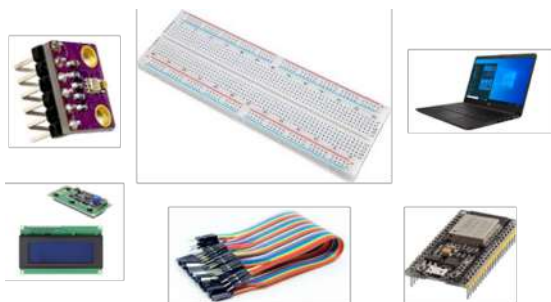


Figura 6. Materiales y equipo.

VI. PROCEDIMIENTO EXPERIMENTAL

Explicación del Código Principal

Tenemos que comprender que el código descargado en el ESP32 realiza las siguientes tareas:

1. Lee los datos del sensor BME280.
2. Muestra esos datos en un display LCD 20x4 con controlador I2C.
3. Imprime los datos en el monitor serial.
4. Almacena los datos en un archivo .txt en la memoria flash del ESP32.
5. Controla el led del ESP32 para indicar que está funcionando el sistema

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <FS.h>
#include <SPIFFS.h>
```

Figura 7. Librerías utilizadas

En la imagen 7 se puede ver las librerías utilizadas para todo el código.

```
LiquidCrystal_I2C lcd(0x27, 20, 4); // Dirección del LCD
Adafruit_BME280 bme;

// Control tiempos
unsigned long previousMillisLED = 0;
unsigned long previousMillisLCD = 0;
unsigned long previousMillisSerial = 0;

// Intervalos de tiempo
const long intervalLED = 3000; // LED cada 3 segundos
const long intervalLCD = 5000; // LCD cada 5 segundos
const long intervalSerial = 900000;
```

Figura 8. Variables que son utilizadas en el archivo

En la **Figura 8** se puede ver la parte del código que se encarga de declarar las variables de tiempo que se utilizarán más adelante, define la dirección y el tamaño del LCD y crea un objeto para el sensor.

```

void setup() {
  Serial.begin(115200);

  // Inicialización del LED
  pinMode(LED_BUILTIN, OUTPUT);

  // Inicialización del LCD
  lcd.init();
  lcd.backlight();

  // Test de funcionalidad del sensor
  Serial.println(F("BME280 test"));

  // Inicialización del sensor BME280
  if (bme.begin(0x76)) {
    Serial.println("No se encuentra el sensor BME280, revisa las conexiones.");
    while (1);
  }

  // Inicialización del SPIFFS
  if (!SPIFFS.begin(true)) {
    Serial.println("Error al montar SPIFFS.");
    return;
  }

  // Crear o abrir el archivo si ya existe
  File file = SPIFFS.open("/datos.txt", FILE_APPEND);
  if (!file) {
    Serial.println("Error al abrir archivo para escribir.");
  } else {
    Serial.println("Archivo abierto correctamente.");
    file.close(); // Cerrar el archivo después de abrirlo
  }
}

```

Figura 9. Función Setup.

En la **imagen 9** se puede ver la parte del código que se encarga de la inicialización de todos los componentes del sistema.

```

void loop() {
  // Obtener el tiempo actual
  unsigned long currentMillis = millis();

  // Leer los datos del sensor
  float temperatura = bme.readTemperature();
  float presion = bme.readPressure() / 100.0F;
  float humedad = bme.readHumidity();

  // Control del LED
  if (currentMillis - previousMillisLED >= intervalLED) {
    previousMillisLED = currentMillis;
    // Alternar el estado del LED
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
  }

  // Actualizar el LCD
  if (currentMillis - previousMillisLCD >= intervalLCD) {
    previousMillisLCD = currentMillis;
    updateLCD(temperatura, humedad, presion); // Pasar los datos a la función LCD
  }

  // Imprimir los valores en el monitor serie y escribir en el archivo
  if (currentMillis - previousMillisSerial >= intervalSerial) {
    previousMillisSerial = currentMillis;
    printValues(temperatura, presion, humedad);
    saveDataToFile(temperatura, presion, humedad);
  }
}

```

Figura 10. Función Loop.

```

// Función para mostrar los valores en el LCD
void updateLCD(float temperatura, float humedad, float presion) {
  lcd.clear();
  lcd.setCursor(4, 0);
  lcd.print("Datos Sensor");

  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(temperatura);
  lcd.print(" C");

  lcd.setCursor(0, 2);
  lcd.print("Hum: ");
  lcd.print(humedad);
  lcd.print(" %");

  lcd.setCursor(0, 3);
  lcd.print("Pres: ");
  lcd.print(presion);
  lcd.print(" hPa");
}

```

Figura 11. Función de actualización del LCD.

En la **imagen 11** se puede ver la parte del código que se encarga de actualizar la pantalla cada cierto tiempo para mostrar los nuevos valores.

```

// Función para imprimir los valores en el monitor serie
void printValues(float temperatura, float presion, float humedad) {
  Serial.print("Temperatura: ");
  Serial.print(temperatura);
  Serial.print(" C, Presión: ");
  Serial.print(presion);
  Serial.print(" hPa, Humedad: ");
  Serial.print(humedad);
  Serial.println(" %");
}

```

Figura 12. Función de impresión de datos en el monitor serial.

En la **imagen 12** se puede ver la parte del código que se encarga de imprimir los datos en el monitor serial del arduino IDE.

```

// Función para guardar los datos en el txt
void saveDataToFile(float temperatura, float presion, float humedad) {
  File file = SPIFFS.open("/datos.txt", FILE_APPEND);
  if (!file) {
    Serial.println("Error");
  } else {
    file.printf("Temperatura: %.2f C, Presión: %.2f hPa, Humedad: %.2f%%\n",
      temperatura, presion, humedad);
    Serial.println("Datos guardados");
    file.close();
  }
}

```

Figura 13. Función de almacenamiento de datos en archivo txt.

En la **imagen 13** se puede ver la parte del código que se encarga de almacenar los datos en el archivo txt. Una vez que creó por primera vez el archivo solo lo abre para modificar y agregar datos en cierto intervalo de tiempo y procede a cerrarlo, de lo contrario, envía un mensaje de error.

Explicación del Código Secundario (Extracción de Datos)

```

#include <FS.h>
#include <SPIFFS.h>

```

Figura 14. Librerías utilizadas.

En la **imagen 10** se puede ver la parte del código que se encarga de la lectura y registro de los datos y del control de tiempo de los mismos.

En la **imagen 14** se puede ver las librerías utilizadas para todo el código.

```

void setup() {
  Serial.begin(115200);

  // Inicializar SPIFFS
  if (!SPIFFS.begin(true)) {
    Serial.println("Error al montar SPIFFS.");
    return;
  }

  // Función para leer el archivo
  readFile(SPIFFS, "/datos.txt");
}

```

Figura 15. Función Setup.

En la **imagen 15** se puede ver la parte del código que se encarga de la inicialización de SPIFFS del ESP32.

```

// Función para leer el archivo y mostrarlo en el Monitor Serie
void readFile(fs::FS &fs, const char * path) {
  Serial.printf("Leyendo archivo: %s\n", path);

  File file = fs.open(path, FILE_READ);
  if (!file || file.isDirectory()) {
    Serial.println("Error.");
    return;
  }

  // Leer el archivo línea por línea y mostrarlo en el Monitor Serie
  while (file.available()) {
    String line = file.readStringUntil('\n');
    Serial.println(line);
  }
  file.close();
}

```

Figura 16. Función de extracción de datos e impresión en el monitor serial.

En la **imagen 16** se puede ver la parte del código que se encarga de leer línea por línea el archivo de texto creado en la recolección de datos para imprimirlo en el monitor serial.

Bonus: Explicación del código IoT

```

#include <Wire.h>
#include <ThingSpeak.h>
#include <LiquidCrystal_I2C.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <WiFi.h>

```

Figura 17. Librerías utilizadas

En la imagen 17 se puede ver las librerías utilizadas para todo el código.

```

LiquidCrystal_I2C lcd(0x27, 20, 4); // Dirección del LCD
Adafruit_BME280 bme;

WiFiClient cliente;

//Conexion a internet
const char* ssid = "TP-Link_31C6";
const char* password = "98287956";

unsigned long channelID = 2706778;
const char* WriteAPIKey = "L9UWQ8UKLHIBHTZB";

// Control tiempos
unsigned long previousMillisLED = 0;
unsigned long previousMillisLCD = 0;
unsigned long previousMillisSerial = 0;

// Intervalos de tiempo
const long intervalLED = 3000; // LED cada 3 segundos
const long intervalLCD = 5000; // LCD cada 5 segundos
const long intervalSerial = 5000;

```

Ojo: Tengan cuidado con las llaves de su API . Busquen una forma de mantenerlas en secreto y no colocarla de forma textual en su código

Figura 18. Variables que son utilizadas en el archivo

En la **imagen 18** se puede ver la parte del código que se encarga de declarar las variables de tiempo que se utilizarán más adelante, define la dirección y el tamaño del LCD, crea un objeto para el sensor y define las variables para la conexión a internet y a la página donde se van monitorear los datos.

```

void setup() {
  Serial.begin(115200);

  // Inicialización del LED
  pinMode(LED_BUILTIN, OUTPUT);

  // Inicialización del LCD
  lcd.init();
  lcd.backlight();

  Serial.println(F("BME280 test"));

  // Inicialización del sensor BME280
  if (!bme.begin(0x76)) {
    Serial.println("No se encuentra el sensor BME280, revisa las conexiones.");
    while (1);
  }
  WiFi.begin(ssid,password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Wifi conectado!");

  ThingSpeak.begin(cliente);
}

```

Figura 19. Función Setup.

En la **imagen 19** se puede ver la parte del código que se encarga de la inicialización de todos los componentes del sistema.

```

void loop() {
  // Obtener el tiempo actual
  unsigned long currentMillis = millis();

  // Leer los datos del sensor
  float temperatura = bme.readTemperature();
  float presion = bme.readPressure() / 100.0F;
  float humedad = bme.readHumidity();

  // Control del LED
  if (currentMillis - previousMillisLED >= intervalLED) {
    previousMillisLED = currentMillis;
    // Alternar el estado del LED
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
  }

  // Actualizar el LCD
  if (currentMillis - previousMillisLCD >= intervalLCD) {
    previousMillisLCD = currentMillis;
    updateLCD(temperatura, humedad, presion); // Pasar los datos a la función LCD
  }

  // Imprimir los valores en el monitor serie
  if (currentMillis - previousMillisSerial >= intervalSerial) {
    previousMillisSerial = currentMillis;
    printValues(temperatura, presion, humedad);
  }

  ThingSpeak.writeFields(channelID, WriteAPIKey);
  Serial.println("Datos enviados a ThingSpeak!");
  delay(5000);
}

```

Figura 20. Función Loop.

En la [imagen 20](#) se puede ver la parte del código que se encarga de la lectura y registro de los datos y del control de tiempo de los mismos y de la comunicación con la página de monitoreo.

```

// Función para mostrar los valores del BME280 en el LCD
void updateLCD(float temperatura, float humedad, float presion) {
  lcd.clear();
  lcd.setCursor(4, 0);
  lcd.print("Datos Sensor");

  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(temperatura);
  lcd.print(" C");

  lcd.setCursor(0, 2);
  lcd.print("Hum: ");
  lcd.print(humedad);
  lcd.print(" %");

  lcd.setCursor(0, 3);
  lcd.print("Pres: ");
  lcd.print(presion);
  lcd.print(" hPa");
  ThingSpeak.setField(1, temperatura);
  ThingSpeak.setField(2, humedad);
  ThingSpeak.setField(3, presion);
}

```

Figura 21. Función de actualización del LCD y comunicación con la página web de monitoreo.

En la [imagen 21](#) se puede ver la parte del código que se encarga de actualizar la pantalla cada cierto tiempo para mostrar los nuevos valores y donde hacia donde se van a mandar a la página de monitoreo.

Implementación física:

1. El primer paso en la implementación física de nuestro proyecto fue detectar los pines bajo los que se iba a trabajar en nuestra placa Esp32.



Figura 22. Pines del Esp32.

Como podemos ver en la figura 22 tenemos la parte superior de la placa en donde está la asignación de pines que trae por defecto nuestro Esp32.

2. Una vez que los pines ya se habían anteriormente seleccionado, se procedió a realizar la instalación de los mismos mediante jumpers a nuestra placa protoboard.

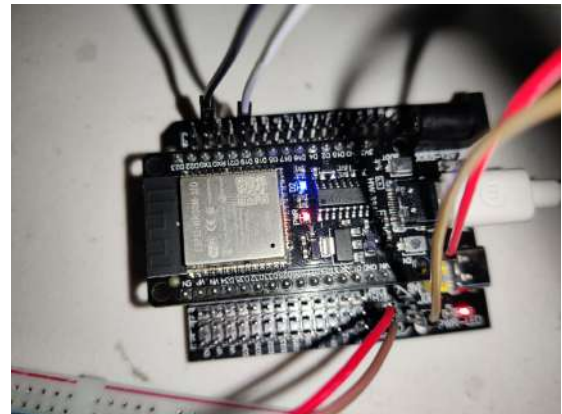


Figura 23. Pines del Esp32.

Como podemos ver en la figura 23 hasta ese momento la primera decisión tomada fue la implementación de jumpers en el protoboard y la extensión del esp32.

3. La tercera decisión a tomar fue la implementación de nuestro modulo sensor medidor de presión, humedad, temperatura y altitud, el BME280, el cual como punto extra, puede medir como se mencionó anteriormente, la altitud del sistema.

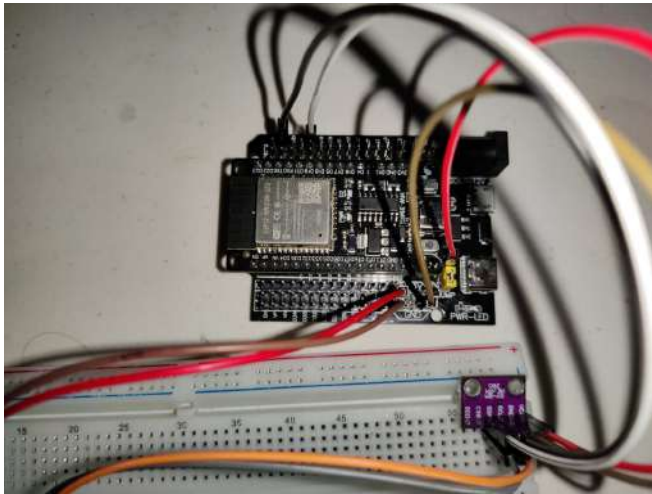


Figura 24. Pines del Esp32.

Como podemos ver en la figura 24 se continuaron las decisiones hasta llegar a la implementación del sensor de variables termodinámicas, sensor que medirá y leerá los datos tomados y nos ayudará a monitorear.

4. Hasta el punto anterior, como se podrá ver en la sección del código, ya existían variables de lectura en respuesta en nuestro sistema de Arduino IDE, ahora venía la siguiente mitad del problema, conectar un panel LCD

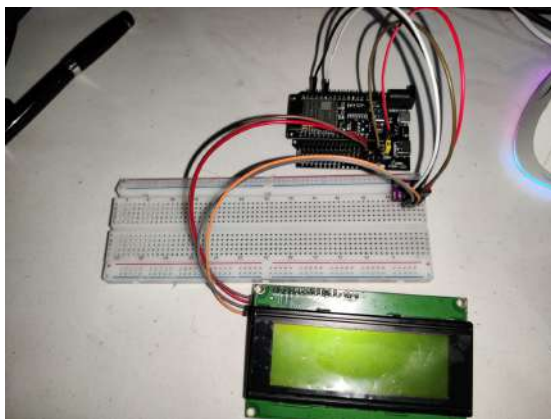


Figura 25. Panel LCD.

Como podemos ver en la figura 25 tenemos la conexión del panel LCD conectado hasta el que era en su momento nuestro sistema de toma de datos, aunque era necesario encontrar una manera de conectar variables y mostrarlas al mundo mediante un lenguaje sencillo de comprender.

5. El último paso fue el monitoreo de datos, es decir, la compilación del código para su ejecución y verificación de funcionamiento antes de comenzar a tomar los datos durante 24 horas y continuar con la siguiente sección del reporte.

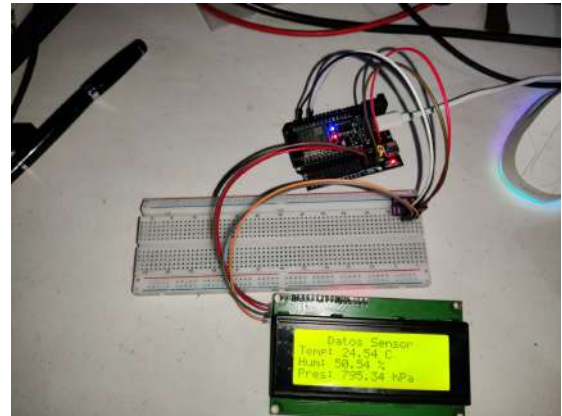


Figura 26. Funcionamiento correcto.

Como podemos ver en la figura 26 existe una compilación de código y correcto funcionamiento de la pantalla LCD así como de las conexiones, ahora vienen los siguientes pasos, el almacenamiento de datos.

```
BME280 test
Archivo abierto correctamente.
Temperatura: 24.91 C, Presión: 795.05 hPa, Humedad: 49.06 %
Datos guardados
Temperatura: 24.90 C, Presión: 795.03 hPa, Humedad: 48.96 %
Datos guardados
Temperatura: 24.90 C, Presión: 795.05 hPa, Humedad: 49.20 %
Datos guardados
Temperatura: 24.90 C, Presión: 795.04 hPa, Humedad: 49.19 %
Datos guardados
Temperatura: 24.90 C, Presión: 795.03 hPa, Humedad: 49.34 %
Datos guardados
Temperatura: 24.90 C, Presión: 795.03 hPa, Humedad: 49.29 %
Datos guardados
Temperatura: 24.89 C, Presión: 795.03 hPa, Humedad: 49.48 %
Datos guardados
Temperatura: 24.89 C, Presión: 795.05 hPa, Humedad: 49.57 %
Datos guardados
Temperatura: 24.89 C, Presión: 795.05 hPa, Humedad: 49.54 %
Datos guardados
Temperatura: 24.89 C, Presión: 795.03 hPa, Humedad: 49.19 %
Datos guardados
```

Figura 27. Almacenamiento de datos en la memoria Flash del ESP32.

En el monitor serial (imagen 27) se puede observar que se van almacenando los datos avisándonos cada vez que lo hace.

Temperatura: 24.89 C, Presión: 795.03 hPa, Humedad: 49.19%
 Temperatura: 24.88 C, Presión: 795.05 hPa, Humedad: 49.09%
 Temperatura: 24.88 C, Presión: 795.05 hPa, Humedad: 49.13%
 Temperatura: 24.87 C, Presión: 795.02 hPa, Humedad: 49.10%
 Temperatura: 24.86 C, Presión: 795.04 hPa, Humedad: 49.06%
 Temperatura: 24.86 C, Presión: 795.01 hPa, Humedad: 49.42%
 Temperatura: 24.86 C, Presión: 795.05 hPa, Humedad: 49.31%
 Temperatura: 24.86 C, Presión: 795.03 hPa, Humedad: 49.30%
 Temperatura: 24.86 C, Presión: 795.00 hPa, Humedad: 49.19%
 Temperatura: 24.85 C, Presión: 795.06 hPa, Humedad: 49.35%
 Temperatura: 24.85 C, Presión: 795.03 hPa, Humedad: 49.11%
 Temperatura: 24.86 C, Presión: 795.05 hPa, Humedad: 49.22%
 Temperatura: 24.86 C, Presión: 795.02 hPa, Humedad: 49.38%
 Temperatura: 24.86 C, Presión: 795.01 hPa, Humedad: 49.34%
 Temperatura: 24.86 C, Presión: 795.02 hPa, Humedad: 49.28%
 Temperatura: 24.86 C, Presión: 795.05 hPa, Humedad: 49.92%
 Temperatura: 24.86 C, Presión: 795.04 hPa, Humedad: 49.45%
 Temperatura: 24.86 C, Presión: 795.01 hPa, Humedad: 49.15%
 Temperatura: 24.85 C, Presión: 795.01 hPa, Humedad: 49.05%
 Temperatura: 24.85 C, Presión: 795.03 hPa, Humedad: 49.06%
 Temperatura: 24.85 C, Presión: 795.03 hPa, Humedad: 49.21%
 Temperatura: 24.85 C, Presión: 795.05 hPa, Humedad: 49.54%
 Temperatura: 24.84 C, Presión: 795.05 hPa, Humedad: 49.09%
 Temperatura: 24.84 C, Presión: 795.07 hPa, Humedad: 49.15%
 Temperatura: 24.84 C, Presión: 795.03 hPa, Humedad: 49.13%
 Temperatura: 24.83 C, Presión: 795.03 hPa, Humedad: 49.10%
 Temperatura: 24.83 C, Presión: 795.04 hPa, Humedad: 49.25%
 Temperatura: 24.83 C, Presión: 795.02 hPa, Humedad: 49.98%
 Temperatura: 24.82 C, Presión: 795.06 hPa, Humedad: 50.18%
 Temperatura: 24.82 C, Presión: 795.04 hPa, Humedad: 49.86%
 Temperatura: 24.82 C, Presión: 795.07 hPa, Humedad: 50.06%
 Temperatura: 24.83 C, Presión: 795.04 hPa, Humedad: 49.71%
 Temperatura: 24.82 C, Presión: 795.04 hPa, Humedad: 49.64%

Figura 28. Almacenamiento de datos en la memoria Flash del ESP32.

Para extraer los datos, simplemente cargamos el segundo código para que lo muestre en el monitor serial (imagen28).

Monitoreo de datos en tiempo real:

1. El primer paso después de:

- La conexión y el circuito
- La programación de los sensores
- Configuración del servidor

Fue **enviar los datos a nuestra página ThingSpeak en donde guardaremos todos los datos**

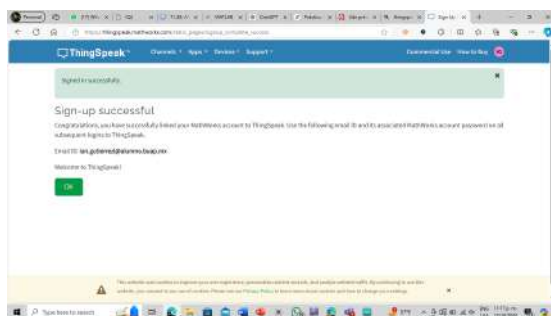


Figura 29. Página de recopilación de datos.

2. El siguiente paso es **craar** una nueva base de datos y rellenar los campos de las variables que utilizaremos

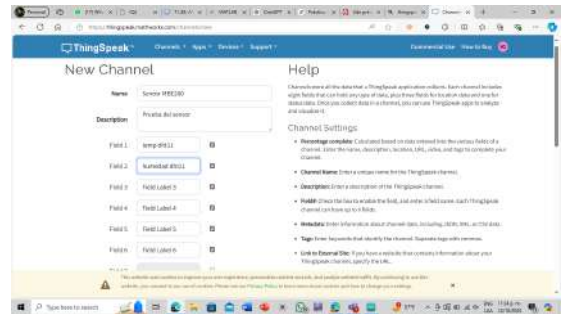


Figura 30. Página de recopilación de datos, primeros pasos.

3. Finalmente guardamos el canal y cargamos los datos.

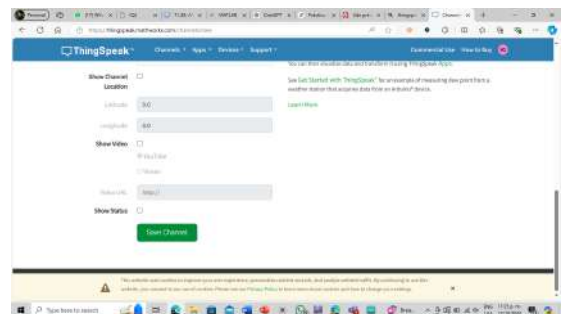


Figura 31. Página de recopilación de datos, pasos canal.

4. Finalmente en esta sección aparecerán las gráficas

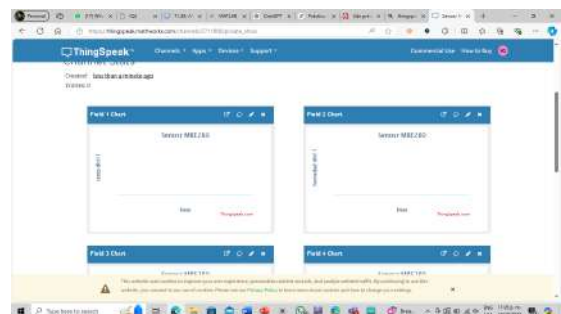


Figura 32. Página de recopilación de datos, pasos canal.

5. Como paso importante solamente recordar que el uso de las librerías y la conexión arduino viene explicada en la sección del código.

VII. ANÁLISIS DE LOS DATOS EXPERIMENTALES:

Posterior al armado del circuito, procedimos a ubicar nuestro circuito armado, ajustando el código para que registrara los datos de **presión, humedad y temperatura cada 15 minutos**. Una vez transcurridas las 24 horas de análisis, arrojó los siguientes datos, los cuales fueron tabulados y graficados para observar de una mejor manera dichos resultados.

Tiempo Horas	Temperatura °C	Presión HPa	Humedad %
08:00	13.01	793.44	56.52
08:15	13.49	793.16	60.32
08:30	13.52	793.47	54.18
08:45	13.78	793.97	56.65
09:00	13.89	794.32	55.35
09:15	14.02	792.92	57.01
09:30	14.26	793.12	57.8
09:45	14.4	793.32	53.94
10:00	14.56	793.49	58.64
10:15	14.64	793.88	53.07
10:30	14.73	792.97	56.02
10:45	14.99	794.05	58.83
11:00	15.1	793.41	58.68
11:15	15.1	793.99	57.51
11:30	15.15	794.35	56.16
11:45	15.34	793.93	57.44
12:00	15.55	793.57	57.34
12:15	15.55	793.46	59.81
12:30	15.81	793.92	57.48
12:45	15.92	793.16	57.75

Figura 33. Tabla 1. Registro de datos de temperatura, presión y humedad a las primeras horas.

13:00	16.63	794.42	58.04
13:15	17.03	793.45	59.42
13:30	17.09	794.23	54.74
13:45	17.18	793.95	58.75
14:00	17.35	793.2	56.36
14:15	17.73	793.84	54.14
14:30	18.07	793.19	58.18
14:45	18.12	794.2	56.27
15:00	18.59	794.16	56.65
15:15	18.61	793.56	53.9
15:30	18.77	793.09	54.42
15:45	18.94	794.51	58.82
16:00	19.45	793.2	57.37
16:15	19.6	793.11	55.27
16:30	19.62	793.22	58.65
16:45	19.67	793.92	60.27
17:00	19.81	793.77	55.74
17:15	20.29	793.05	57.83
17:30	20.5	793.48	53.91
17:45	20.54	794.4	54
18:00	20.74	793.04	55.4
18:15	21.26	793.68	58.5
18:30	21.85	794.34	60.13
18:45	21.93	793.89	59.65
19:00	22.22	793.1	58.39
19:15	22.68	793.22	60.06
19:30	22.84	793.9	56.44
19:45	22.91	794.31	57.96
20:00	22.97	793.19	53
20:15	22.94	793.19	57.99
20:30	22.8	793.53	55.04
20:45	22.24	794.18	54.49
21:00	22.05	793.04	57.7
21:15	21.92	793.04	55.7
21:30	21.74	794.21	54.15
21:45	21.46	793.32	54.86
22:00	21.33	792.96	54.77
22:15	20.54	794.03	57.41
22:30	20.13	793.02	56.19
22:45	20.08	794.52	56.71
23:00	19.93	792.99	60.25
23:15	19.76	793.88	57.94
23:30	19.72	792.99	58.55
23:45	19.45	793.36	54.6
00:00	19.43	793.1	57.87

Figura 34. Tabla 2. Registro de datos de temperatura, presión y humedad a las segundas horas.

00:15	18.93	794.43	57.47
00:30	18.66	793.43	57.18
00:45	18.51	794.31	56.74
01:00	18.05	793.86	53.82
01:15	17.91	793.06	58.96
01:30	17.61	794.4	56.17
01:45	17.3	794.38	58.96
02:00	16.95	794.02	54.17
02:15	16.78	794.48	60.02
02:30	16.64	794.47	53.57
02:45	16.53	792.99	55.75
03:00	16.52	792.99	56.54
03:15	15.95	793.43	60
03:30	15.86	794.48	60.09
03:45	15.46	793.73	54.41
04:00	15.39	793.88	56.59
04:15	14.98	794.14	54.77
04:30	14.96	794.03	58.2
04:45	14.9	793.04	53.8
05:00	14.59	794.27	58.62
05:15	14.59	794.32	58.47
05:30	14.45	794.04	56.35
05:45	14.21	794.48	57.94
06:00	14.07	793.11	60.3
06:15	13.74	793.25	55.81
06:30	13.72	792.95	54.58
06:45	13.52	793.63	53.2
07:00	13.49	793.42	53.91
07:15	13.41	793.4	54.42
07:30	13.32	793.39	60.23
07:45	13.07	793.51	58.99
08:00	13.37	793.21	59.34

Figura 35. Tabla 3. Registro de datos de temperatura, presión y humedad a las últimas horas.

En las tablas de las figuras 33, 34 y 35 podemos ver los datos que se tomaron y las variables que tuvimos, teniendo al rededor de 96 mediciones de cada variable detectada.

A continuación, generamos las gráficas a partir de los resultados mostrados anteriormente y, con esto, analizamos el comportamiento de estas tres variables en el entorno de estudio.

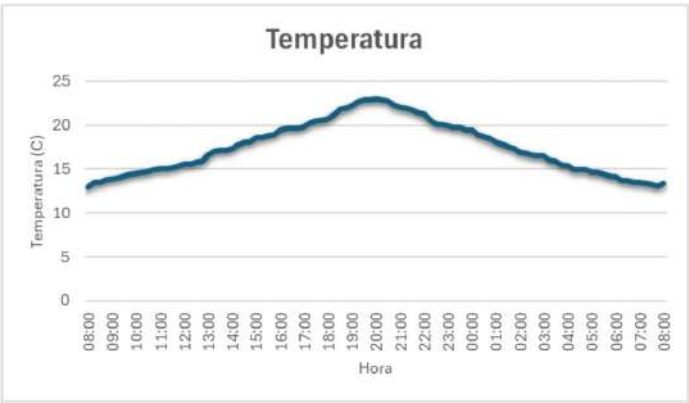


Figura 36. Gráfica 1, Temperatura Recopilada en el lapso de 24 horas

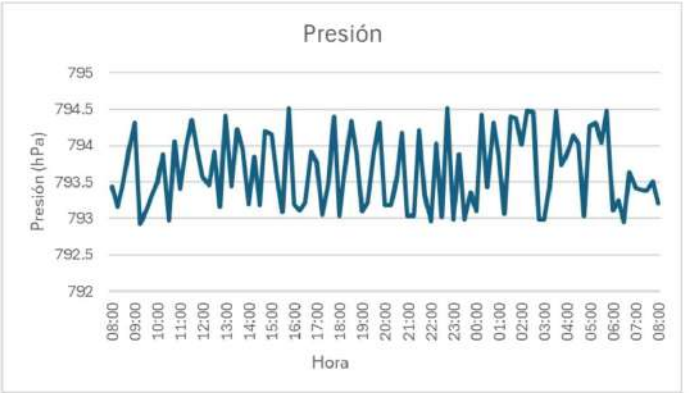


Figura 37. Gráfica 2. Presión Atmosférica recopilada en el lapso de 24 horas

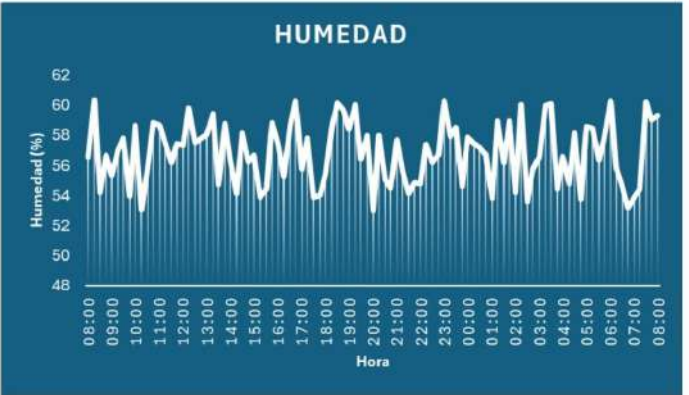


Figura 38. Gráfica 2. Presión Atmosférica recopilada en el lapso de 24 horas

A través de este sistema, logramos recopilar y analizar datos en tiempo real, lo cual es de gran utilidad si se quisiera aplicar en áreas que requieran control ambiental como lo es la agricultura o en general en la industria. A lo largo del periodo de 24 horas, las tres variables medidas (**temperatura, presión atmosférica y humedad relativa**) mostraron un comportamiento esperado, aunque hubo algunas fluctuaciones interesantes que vale la pena comentar. En cuanto a la temperatura, la 36 Gráfica 1 muestra una tendencia clara de aumento durante las horas del día, alcanzando su máximo cercano a los 25°C hacia la mitad

del lapso de 24 hrs, para luego descender de forma progresiva hasta los 10°C en la madrugada. Esto es lo que esperábamos, ya que el lugar donde se realizó la medición tiene un clima típico donde la temperatura sube durante el día y baja en la noche.

Puede que este cambio se deba también a la temperatura de la computadora debido al lapso continuo en el que estuvo trabajando. La presión atmosférica en la imagen 37 Gráfica 2 fue bastante estable, pero con algunas oscilaciones. La presión varió entre 793 hPa y 795 hPa, lo cual es normal dentro de un periodo corto de tiempo como 24 horas. Las fluctuaciones más marcadas pueden estar relacionadas con ligeros cambios en las condiciones meteorológicas, como el paso de alguna masa de aire o la influencia de alguna corriente atmosférica. Sin embargo, el rango en general no cambió mucho, lo que indica que no hubo fenómenos meteorológicos significativos durante la medición.

Por otro lado, la humedad relativa, mostrada en la imagen 38 Gráfica 3, tuvo picos más pronunciados. Durante el día, se mantuvo dentro de un rango de 40 por ciento de unidad a 60 por ciento de unidad, lo que es bastante razonable. Los picos observados durante la madrugada y las primeras horas de la mañana pueden deberse al aumento de la humedad cuando la temperatura baja, lo que genera más condensación en el ambiente. De acuerdo con lo que investigamos, al disminuir la temperatura, el aire puede retener menos vapor de agua.

VIII. CONCLUSIÓN

El diseño de este circuito, basado en la integración de sensores y microcontroladores que procesan los datos de forma eficiente, permite llevar un control, no solo de las variables que se analizaron durante el proyecto, sino de cualquier variable ambiental de la que se necesite estudiar el comportamiento.

Además, nos ha servido como una herramienta de estudio práctico de la unidad, pues es importante reflexionar los conocimientos teóricos aplicados a la cotidianidad.

En general, ha sido una excelente oportunidad para desarrollar habilidades técnicas y de análisis, y que ofrece una base sólida para futuros proyectos más complejos en áreas relacionadas con la electrónica y la instrumentación ambiental.

IX. USO DE LA INTELIGENCIA ARTIFICIAL

El uso de la inteligencia artificial fue una herramienta imprescindible para este proyecto, el principal uso que tomó, fue apoyarnos en la elaboración del código para hacerlo funcionar.

El equipo constaba de integrantes que al unir todos sus conocimientos, tenían una idea por sobre como programar, usar la placa de desarrollo, usar los sensores, las pantallas, todos tenían una parte de idea en el trabajo, es por ello que al buscar ser un equipo eficiente, se trabajó en conjunto para unir todo y hacerlo funcionar, por desgracia, aún existían algunas brechas de conocimientos, existían huecos y eso era un impedimento para poder cumplir nuestro objetivo, así que la inteligencia artificial fue usada de la siguiente manera:

- Unir nuestro conocimiento.

Propósito: Integrar lo que sabemos de diferentes áreas, como mecánica de fluidos, electrónica (Arduino), y sensores, para resolver un problema o mejorar un proyecto.

Pasos concretos:

1. Identificar qué sabía cada miembro del equipo y en qué áreas tenía cada uno más experiencia.
 2. Definir cómo el conocimiento de cada uno puede contribuir a resolver un aspecto del proyecto.
- Identificar lagunas y puntos de apoyo

Propósito: Encontrar áreas donde hay falta de conocimiento (lagunas) y otras donde ya se tiene dominio (puntos de apoyo) para mejorar en conjunto.

Pasos concretos:

1. Hacer una evaluación de los temas clave del proyecto (sensores, programación, integración de hardware, etc.).
 2. Identificar las áreas donde hay mayor dificultad o desconocimiento.
 3. Utilizar los puntos de apoyo (áreas bien dominadas) para facilitar el trabajo.
- Examinar el código elaborado en cada fase y mejorar librerías para Arduino.

Propósito: Revisar el código del proyecto en Arduino y optimizar el uso de librerías para asegurar un funcionamiento eficiente y sin errores.

Pasos concretos:

1. Revisar el código en busca de posibles errores o redundancias.
 2. Verificar que las librerías utilizadas son las más actuales y están bien documentadas.
 3. Probar el código con los sensores y la pantalla para asegurarse de que las librerías están funcionando correctamente.
- Guiarnos en la conexión de la pantalla de 3 líneas y resolver problemas técnicos del ensamblado

Propósito: Asegurarse de que la pantalla esté correctamente conectada al Arduino y resolver cualquier problema técnico relacionado con el hardware.

Pasos concretos:

1. Revisar el esquema de conexiones de la pantalla de 3 líneas, asegurándose de que los pines correctos están conectados al Arduino.
 2. Verificar la compatibilidad de la pantalla con las librerías y el código que se está utilizando.
 3. Solucionar problemas de ensamblado como mala conexión de cables, falta de alimentación, o incompatibilidad de componentes.
 4. Guiarnos en la conexión de la pantalla de 3 líneas y resolver problemas técnicos del ensamblado
- Buscar ideas sobre el uso actual de sensores de presión y temperatura

Propósito: Analizar cómo se utilizan actualmente estos sensores en diferentes campos, para obtener inspiración sobre cómo mejorar el proyecto o explorar nuevas aplicaciones.

Pasos concretos:

1. Investigar aplicaciones actuales de sensores de presión y temperatura en industrias como automotriz, médica, o energías renovables.
2. Estudiar los avances tecnológicos que están siendo implementados como sensores más precisos, inalámbricos o de menor consumo energético
3. Evaluar cómo se podrían mejorar o adaptar las tecnologías existentes para futuras aplicaciones, fuera del contexto del proyecto de la materia, como en proyectos de investigación o desarrollo de productos.

X. REFERENCIAS

1. Y. A. Cengel, *Mecánica de Fluidos*. McGraw-Hill Interamericana, 2018.
2. E. De, “Presión Atmosférica: qué es, fórmula, valor y unidades,” Enciclopedia Significados, 20-feb-2020. [En línea]. Disponible en: <https://www.significados.com/presion-atmosferica/>. [Consultado: 21-oct-2024].
3. “Humedad,” El tiempo.es, 14-jun-2023. [En línea]. Disponible en: <https://www.clima.com/meteopedia/humedad>. [Consultado: 21-oct-2024].
4. J. L. Fernández, “Temperatura,” Fisicalab.com. [En línea]. Disponible en: <https://www.fisicalab.com/apartado/temperatura>. [Consultado: 21-oct-2024].
5. “Sensor BME280 datasheet,” Naylampmechatronics.com. [En línea]. Disponible en: <https://naylampmechatronics.com/sensores-posicion-inerciales-gps/357-sensor-bme280-presion-temperatura-y-humedad.html>. [Consultado: 21-oct-2024].
6. Todomaker, “Conociendo al ESP32,” Todomaker. [En línea]. Disponible en: <https://todomaker.com/blog/conociendo-al-esp32/>. [Consultado: 21-oct-2024].
7. MicroPython, “The MicroPython project,” MicroPython. [En línea]. Disponible en: <https://micropython.org/>. [Consultado: 21-oct-2024].
8. E. (iq2) I., “Sensor de temperatura, presión y humedad. Módulo (GY-BMP280/BM280) + ESP32.” [En línea]. Disponible en: <https://youtube.com/watch?v=O6O-Z135598feature=shared>. [Consultado: 22-oct-2024].
9. Ingeniero en Casa. “Estación meteorológica wifi con ESP32 y ThingSpeak.” 31 de agosto de 2019. [Video en línea]. Disponible: <https://www.youtube.com/watch?v=LMTtIC2jKUg>. [Consultado: 24-oct-2024].
10. TodoMaker. “SPIFFS en ESP32 y Arduino IDE — Sube archivos a la memoria del ESP32.” 15 de febrero de 2022. [Video en línea]. Disponible: <https://www.youtube.com/watch?v=sjti3etJXoo>. [Consultado: 24-oct-2024].

11. Computadoras y Sensores. “ESP32 IoT con ThingSpeak — Estación meteorológica en la nube WiFi con ESP32 y BME280 — Micropython.” 4 de marzo de 2022. [Video en línea]. Disponible: <https://www.youtube.com/watch?v=P1D3kkeNuMU>. [Consultado: 24-oct-2024].
12. Robotica, Impresión 3D y algo más. “Tarjeta de expansión para NodeMcu 8266.” 26 de mayo de 2022. [Video en línea]. Disponible: <https://www.youtube.com/watch?v=z3NfjH6liOA>. [Consultado: 24-oct-2024].
13. Tecnoyfoto. “Sensor BME280 en ESPHome - ¿Qué es, cómo funciona y qué utilidades tiene?” 8 de noviembre de 2022. [Video en línea]. Disponible: <https://www.youtube.com/watch?v=O2QVE0yXF4A>. [Consultado: 24-oct-2024].

XI. ANEXOS

Uso de la inteligencia artificial La inteligencia artificial fue de gran ayuda en la elaboración del experimento y en el reporte de práctica de la siguiente manera

1. Investigación del experimento a realizar

En un principio se utilizó la inteligencia artificial para conocer los tipos de experimentos que se podían hacer relacionados al tema, desgraciadamente, ninguno fue del agrado para los integrantes del equipo por lo que se optó por investigar en la plataforma de Youtube, se encontró un experimento de calentar los fluidos con un globo, pero, observando las implicaciones que tendría a la hora de la presentación de la exposición, se le preguntó a la inteligencia artificial como se podría mejorar el mismo experimento y proporcionó ideas muy creativas y lógicas que se tomaron en cuenta, como por ejemplo, sustituir el globo por una cazuela de metal debido a la eficacia del material para la transferencia del calor.

2. Variables de medición

Otro uso de la inteligencia artificial fue preguntarle que variables se podrían obtener a partir de este experimento tan simple, y con una gran sorpresa, se descubrieron muchas varias variables que se podían obtener a partir de las mediciones de Temperatura, tiempo y calor específico del fluido, pues se podrían obtener variables secundarias derivadas de las primarias mencionadas anteriormente, estas son el calor suministrado al fluido, la tasa de calentamiento del mismo y la variación de temperatura.

3. Ayuda con el marco teórico.

Se utilizó la inteligencia artificial para conocer que temas están relacionados con el calor específico y la energía con la finalidad de enriquecer el reporte y abarcar la mayoría de conceptos posibles para poder entender por completo la complejidad del proceso, que aunque pueda parecer bastante simple, tiene mucho conocimiento oculto que es importante de mencionar y abarcar para su comprensión completa.

4. Establecimiento de objetivos específicos

Si bien ya se conocía el objetivo general del experimento, también era importante resaltar algunos objetivos específicos que se cumplen en la elaboración de mismo,

como la obtención de los datos experimentales mediante la medición precisa de la temperatura y el tiempo de calentamiento para después poder graficar esos valores con la finalidad de tener un apoyo visual que ayude a comprender de mejor manera el comportamiento de los fluidos en su calentamiento en función del tiempo. otro objetivo importante es comprender y poder comparar los resultados obtenidos para poder lograr tener una mejor visión del experimento.

El trabajo es muy bueno. Ya cuentan con un prototipo funcional de monitoreo de variables atmosféricas. Si quieren dar el siguiente paso, pueden continuar integrando el sistema en un estuche robusto que les permita realizar mediciones en el exterior.

Recuerden que el proyecto les ha ayudado a explorar conocimientos básicos de Mecánica de Fluidos y a aplicar sus habilidades en electrónica. Sin embargo, para capitalizar este conocimiento, deben encontrar afuera (en la industria) la forma de resolver problemas que generen un beneficio económico.