



Unidad Académica  
Multidisciplinaria  
Mante



## **Universidad Autónoma De Tamaulipas**

**Materia:** Diseño Electrónico Basado En Sistemas

Embebidos

**Trabajo:** Proyecto Integrador (Documentación)

### **Integrantes:**

Daniel Turrubiates Cervantes

Leonardo Ramos Espinoza

Sebastián Rodela Castillo

Jorge Alejandro Quiroga Hoy

Axel Aram Verlage Aceves

Enríquez Hernández Galdino

Marco Antonio Rojas Olvera

**Docente:** López Piña Daniel

**Grado Y Grupo:** 8 E, J, F

**Fecha:** 11/03/2025



**Unidad Académica  
Multidisciplinaria  
Mante**





## **1. Introducción**

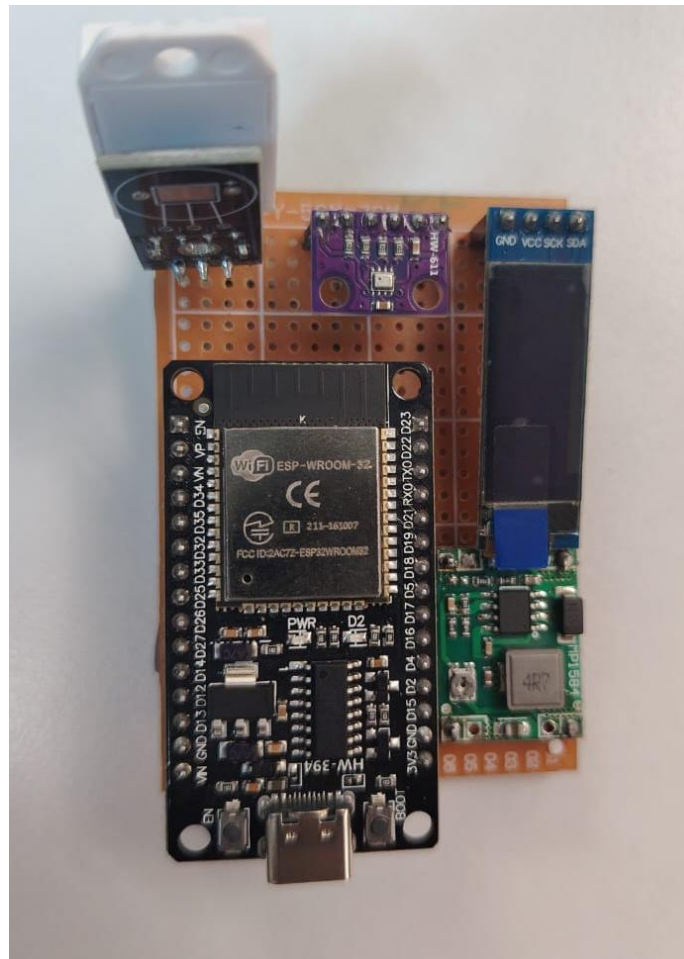
El monitoreo ambiental es fundamental en diversas áreas como la agricultura, la climatología y el control de calidad del aire. Este proyecto tiene como objetivo desarrollar un sistema meteorológico basado en ESP32 para la medición de temperatura, humedad y presión atmosférica, utilizando los sensores DHT22 y BMP280. Los datos obtenidos se visualizarán en una pantalla OLED y serán enviados en tiempo real a la plataforma ThingSpeak para su análisis y almacenamiento. Este sistema proporciona una solución eficiente y económica para el monitoreo de condiciones ambientales.



## Proceso

### 2.1 Componentes utilizados

- ESP32
- Sensor de temperatura y humedad DHT22
- Sensor de presión BMP280
- Pantalla OLED
- Plataforma ThingSpeak
- Placa de circuito impreso
- Fuente de alimentación
- Regulador MP1584





## 2.2 Diagrama de conexión

Para el ensamblaje del sistema, se realizan las siguientes conexiones:

- El ESP32 se conecta a la pantalla OLED mediante el protocolo I2C.
- El sensor DHT22 se conecta al ESP32 mediante una de sus entradas digitales.
- El sensor BMP280 se conecta también mediante I2C.
- La fuente de alimentación provee energía estable al sistema.



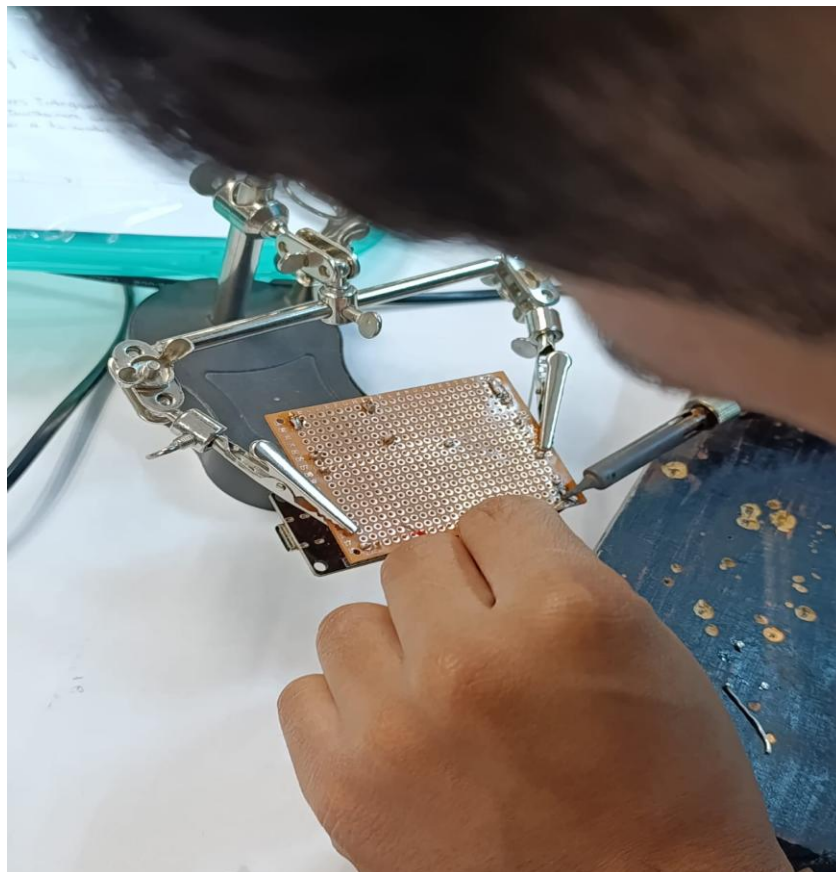
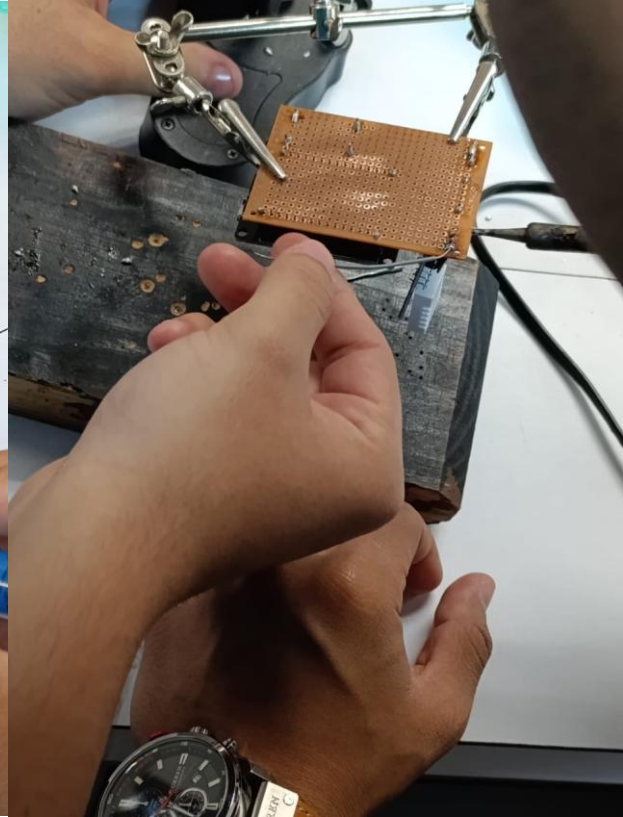
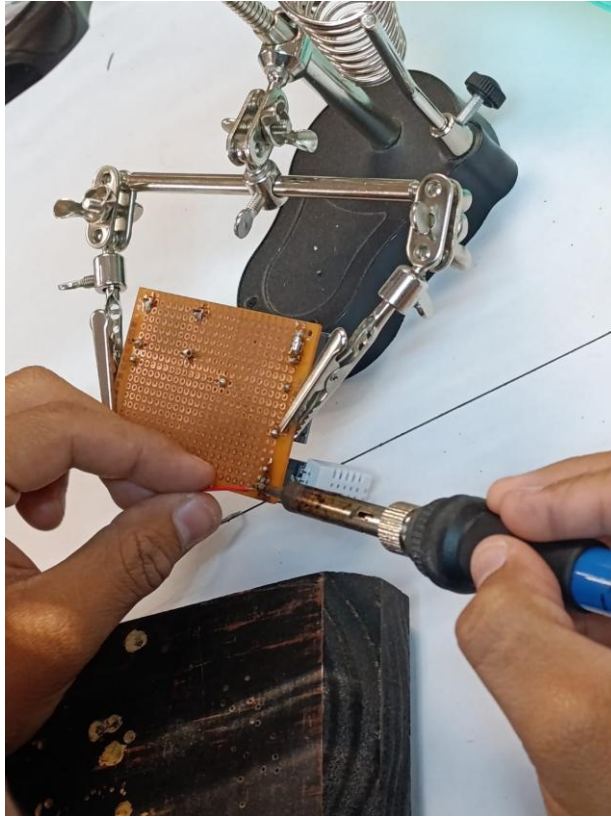




Unidad Académica  
Multidisciplinaria  
Mante

**UAT**  
Universidad Autónoma de  
**TAMAULIPAS**









yo

~~shoes~~ Grabar

~~LED~~ Ensamblaje

~~Apogea~~ Documentación

~~Maico~~ prueba

~~Maico~~ S código

Pic 4 leds, 4 resistencias 230

Fuente de 5V / baterías

cables y proto

BM280

VCC 3V ✓



✓

SD

D22 ✓

SDA

D21 ✓

Oled

VCC 3V



✓

SCK

D22

SDA

D21

MP1584

OUT + 3V ESP

OUT -



ESP

IN +

Fuente alime  
ntación

IN -



Fuente

DHT22

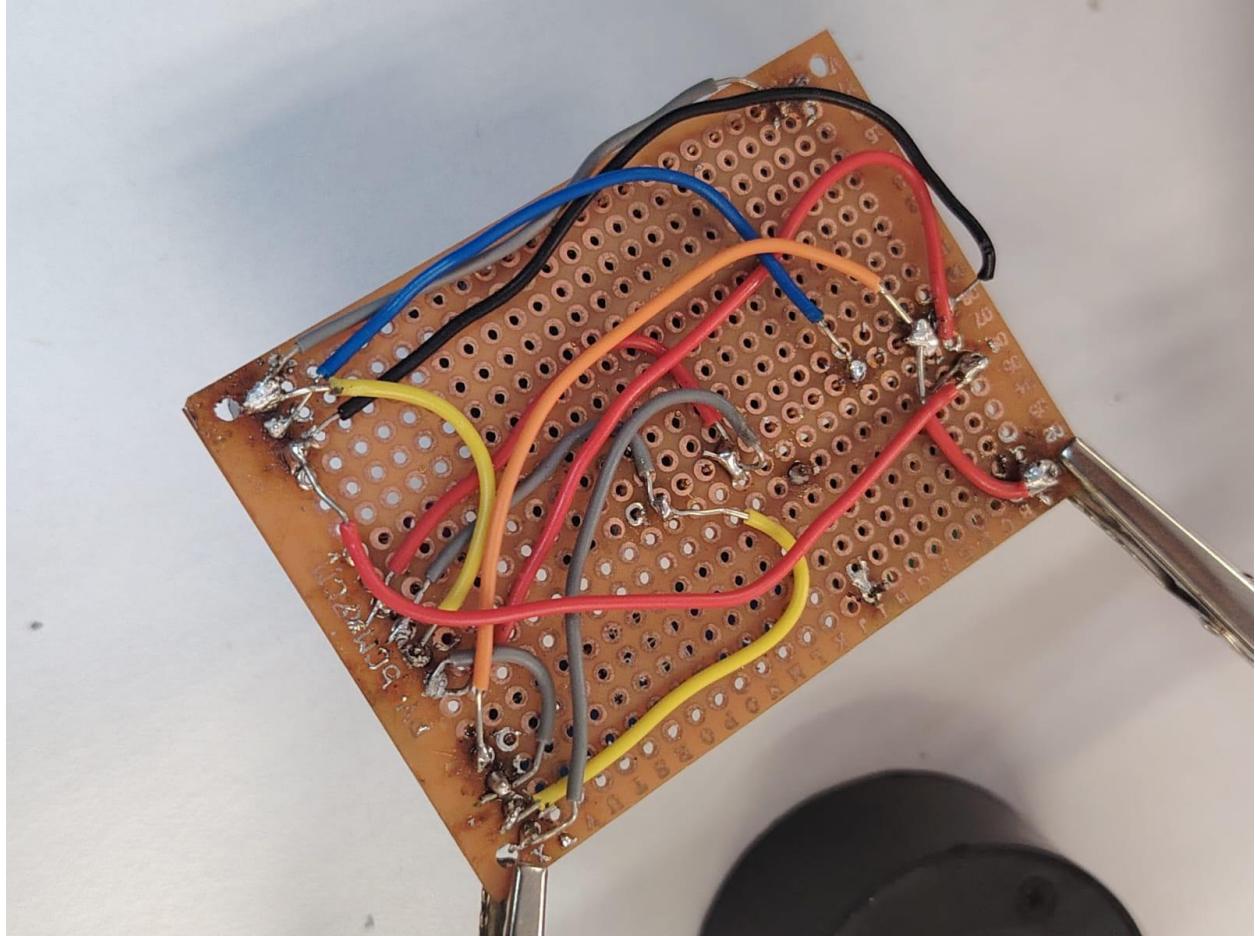
+ 3V ✓

OUT D4 ✓



✓







## 2.3 Desarrollo del código

1. **Lectura de sensores:** Se configura el ESP32 para leer los valores de temperatura y humedad del DHT22, y la presión atmosférica del BMP280.
2. **Visualización en OLED:** Se despliegan los datos en la pantalla OLED para su monitoreo local.
3. **Envío a ThingSpeak:** Se establece la conexión WiFi y se envían los datos a la plataforma en intervalos definidos.

```
sketch_mar11a | Arduino IDE 2.3.4
File Edit Sketch Tools Help
Arduino BT

sketch_mar11a.ino
1 #include <WiFi.h>
2 #include <HTTPClient.h>
3 #include <Adafruit_Sensor.h>
4 #include <DHT.h>
5 #include <Adafruit_BMP280.h>
6 #include <Adafruit_GFX.h>
7 #include <Adafruit_SSD1306.h>
8 #include <Wire.h>
9 #include "secrets.h" // Aseguramos que las credenciales sean importadas desde secrets.h
10
11 // Configuración de sensores y pantalla OLED
12 #define DHTPIN 4
13 #define DHTTYPE DHT22
14 #define SCREEN_WIDTH 128
15 #define SCREEN_HEIGHT 64
16 #define BMP_ADDRESS 0x76
17 #define OLED_RESET -1
18 #define THINGSPEAK_SERVER "http://api.thingspeak.com/update"
19
20 char ssid[] = SECRET_SSID; // Nombre de la red WiFi
21 char pass[] = SECRET_PASS; // Contraseña WiFi
22 String apiKey = SECRET_WRITE_APIKEY; // Clave API de ThingSpeak
23 WiFiClient client;
24
25 DHT dht(DHTPIN, DHTTYPE);
26 Adafruit_BMP280 bmp;
27 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
28
29 void setup() {
30   Serial.begin(115200);
31
32   // Conectar a WiFi
33   connectWiFi();
34
35   // Inicializar sensores
36   dht.begin();
37   if (!bmp.begin(BMP_ADDRESS)) {
38     Serial.println("Error: No se encontró el BMP280.");
39     while (1);
40   }
41
42   // Inicializar pantalla OLED
43   if (!display.begin(SSD1306_SWITCHCAPWCC, 0x3C)) {
44     Serial.println("Error: fallo al inicializar OLED.");
45     while (1);
46   }
47
48   display.clearDisplay();
49   display.setCursor(0,0);
50 }
```



```
sketch_mar11a | Arduino IDE 2.3.4
File Edit Sketch Tools Help
Arduino BT

sketch_mar11a.ino
50 display.setTextColor(WHITE);
51 display.display();
52 }
53
54 void loop() {
55   float temperature = dht.readTemperature();
56   float humidity = dht.readHumidity();
57   float pressure = bmp.readPressure() / 100.0F; // Convertir a hPa
58
59   if (isnan(temperature) || isnan(humidity)) {
60     Serial.println("Error al leer DHT22.");
61     return;
62   }
63
64   // Mostrar datos en OLED
65   mostrarEnPantalla(temperature, humidity, pressure);
66
67   // Enviar datos a ThingSpeak
68   enviarThingSpeak(temperature, humidity, pressure);
69
70   delay(10000); // Espera 10 segundos
71 }
72
73 // Función para conectar a WiFi
74 void conectarWiFi() {
75   Serial.print("Conectando a WiFi...");
76   WiFi.begin(ssid, pass); // Corregimos el error aquí
77
78   int intentos = 0;
79   while (WiFi.status() != WL_CONNECTED && intentos < 20) {
80     delay(500);
81     Serial.print(".");
82     intentos++;
83   }
84
85   if (WiFi.status() == WL_CONNECTED) {
86     Serial.println("WiFi conectado.");
87   } else {
88     Serial.println("\nError: No se pudo conectar a WiFi.");
89   }
90 }
91
92 // Función para mostrar datos en pantalla OLED
93 void mostrarEnPantalla(float temp, float hum, float pres) {
94   display.clearDisplay();
95   display.setCursor(0, 0);
96   display.print(temp);
97   display.print(hum);
98   display.print(pres);
99 }
100
101 // Función para enviar datos a ThingSpeak
102 void enviarThingSpeak(float temp, float hum, float pres) {
103   if (WiFi.status() != WL_CONNECTED) {
104     Serial.println("Error: WiFi desconectado. Intentando reconectar...");
105     conectarWiFi();
106   }
107
108   if (WiFi.status() == WL_CONNECTED) {
109     HTTPClient http;
110     String url = String(THINGSPEAK_SERVER) + "?api_key=" + apiKey +
111                 "&field1=" + String(temp) +
112                 "&field2=" + String(hum) +
113                 "&field3=" + String(pres);
114
115     Serial.println("Enviando datos a ThingSpeak...");
116     http.begin(url);
117     int httpCode = http.GET();
118
119     if (httpCode > 0) {
120       Serial.print("Respuesta HTTP: ");
121       Serial.println(httpCode);
122       if (httpCode == 200) {
123         Serial.println("Datos enviados correctamente.");
124       } else {
125         Serial.println("Error en ThingSpeak.");
126       }
127     } else {
128       Serial.println("Error al conectar con ThingSpeak.");
129     }
130
131     http.end();
132   }
133 }
```

```
sketch_mar11a | Arduino IDE 2.3.4
File Edit Sketch Tools Help
Arduino BT

sketch_mar11a.ino
50 display.setCursor(0, 0);
51 display.print("Temp: ");
52 display.print(temp);
53 display.print("C");
54 display.setCursor(0, 10);
55 display.print("Hum: ");
56 display.print(hum);
57 display.print("%");
58 display.setCursor(0, 20);
59 display.print("Pres: ");
60 display.print(pres);
61 display.print("hPa");
62 display.display();
63 }
64
65 // Función para enviar datos a ThingSpeak
66 void enviarThingSpeak(float temp, float hum, float pres) {
67   if (WiFi.status() != WL_CONNECTED) {
68     Serial.println("Error: WiFi desconectado. Intentando reconectar...");
69     conectarWiFi();
70   }
71
72   if (WiFi.status() == WL_CONNECTED) {
73     HTTPClient http;
74     String url = String(THINGSPEAK_SERVER) + "?api_key=" + apiKey +
75                 "&field1=" + String(temp) +
76                 "&field2=" + String(hum) +
77                 "&field3=" + String(pres);
78
79     Serial.println("Enviando datos a ThingSpeak...");
80     http.begin(url);
81     int httpCode = http.GET();
82
83     if (httpCode > 0) {
84       Serial.print("Respuesta HTTP: ");
85       Serial.println(httpCode);
86       if (httpCode == 200) {
87         Serial.println("Datos enviados correctamente.");
88       } else {
89         Serial.println("Error en ThingSpeak.");
90       }
91     } else {
92       Serial.println("Error al conectar con ThingSpeak.");
93     }
94
95     http.end();
96   }
97 }
```



## 2.4 Configuración de ThingSpeak

1. Crear una cuenta en ThingSpeak.
2. Configurar un canal para almacenar los datos.
3. Obtener las claves API para la transmisión de datos.
4. Configurar ThingSpeak para graficar la información.

Channel ID 2865002

Name	Monitoreo de Temperatura y Humedad	
Description	Olvera Padrón Carlos Michel Verlange Aceves Axel Aram	
Field 1	Humedad	<input checked="" type="checkbox"/>
Field 2	Temperatura	<input checked="" type="checkbox"/>
Field 3	Presión	<input checked="" type="checkbox"/>

ThingSpeak™

Channels ▾ Apps ▾ Devices ▾ Support ▾

Com

## Monitoreo de Temperatura y Humedad

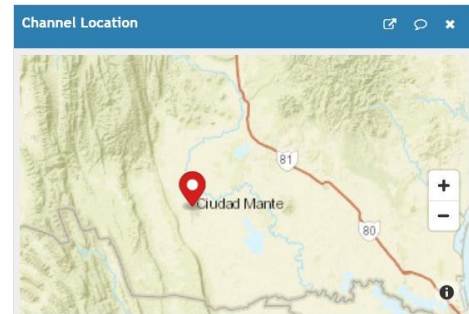
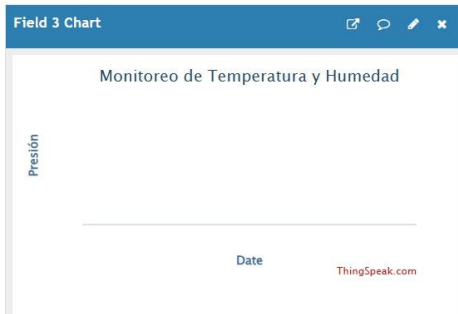
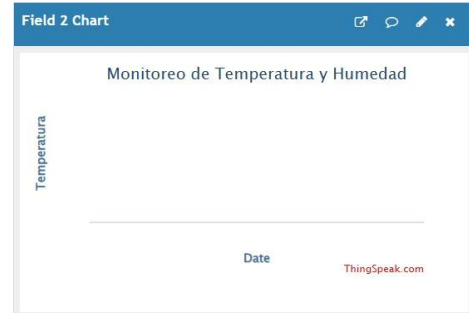
Channel ID: 2865002

Author: mwa0000037092173

Access: Public

Olvera Padrón Carlos Michel Verlange Aceves Axel Aram Ramos Espinoza  
Leonardo Rodela Castillo Sebastian Quiroga Hoy Jorge Alejandro Enríquez  
Hernández Galdino Rojas Olvera Marco Antonio Soria Ortiz Marco Antonio  
Turrubiates Cervantes Daniel  
🔗 [humedad, dht22, bmp280](#)





Tags

humedad, dht22, bmp280

(Tags are comma separated)

Link to External Site

http://

Link to GitHub

https://github.com/

Elevation

Show Channel  
Location



Latitude

22.7469426852

Longitude

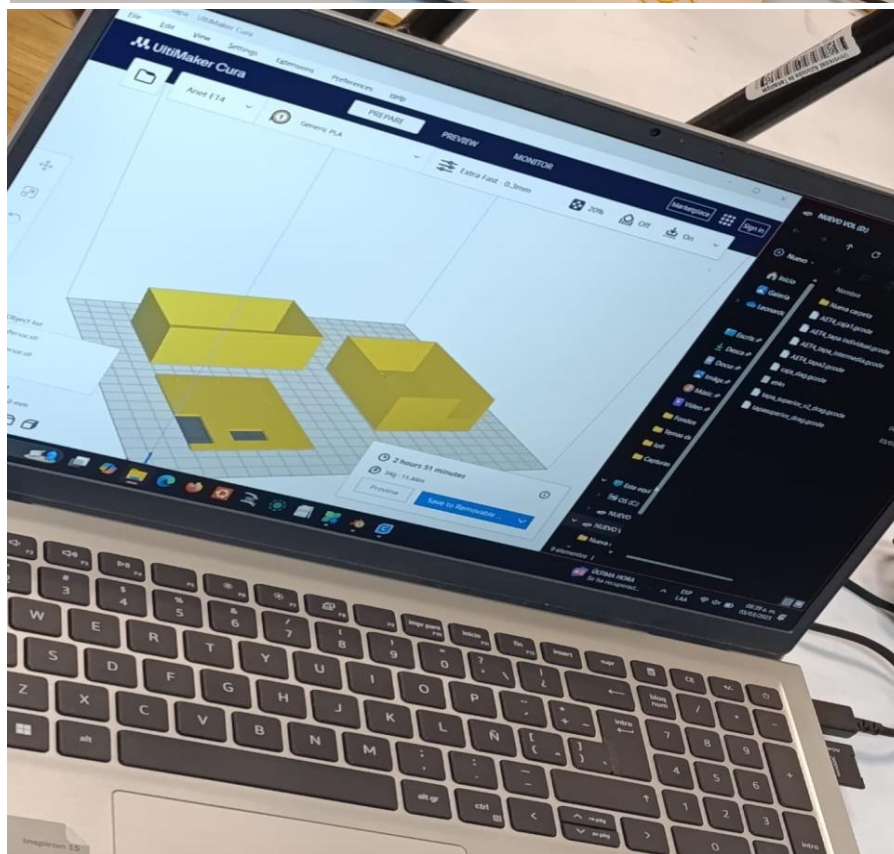
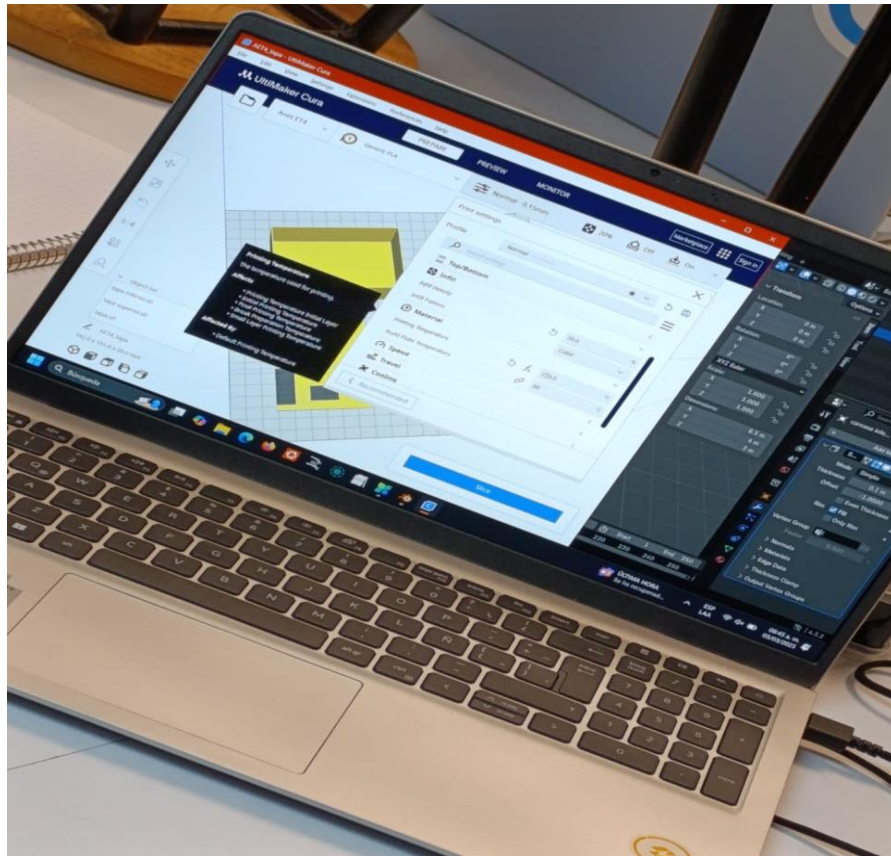
-98.983488471



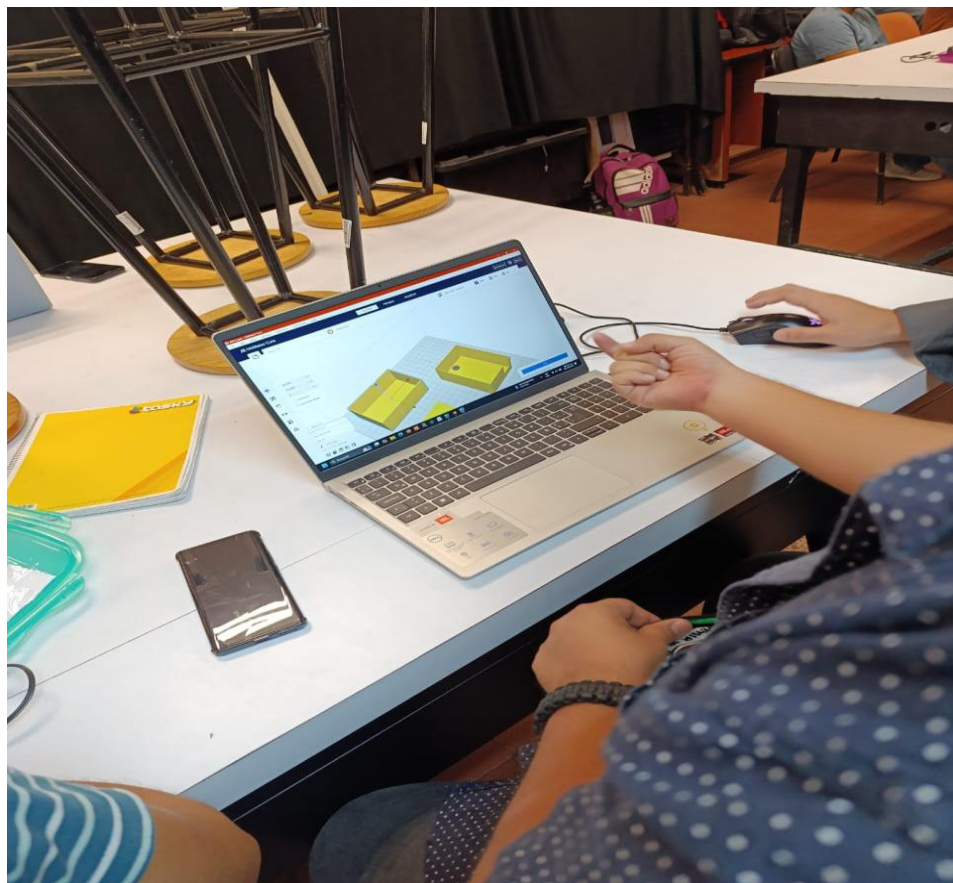
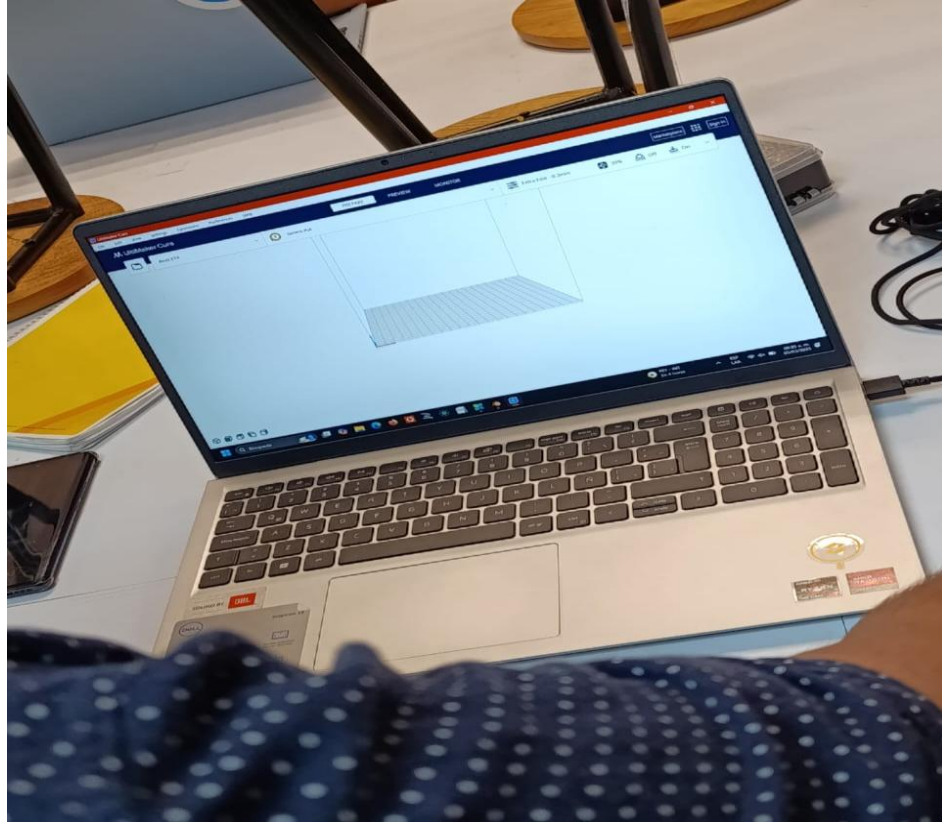
## **2.5 Fabricación de la carcasa impresa en 3D**

Para proteger y organizar los componentes del sistema meteorológico, se diseñó y fabricó una carcasa impresa en 3D. Esta estructura permite:

- Asegurar los sensores y la pantalla OLED en una posición estable.
- Proteger los componentes de condiciones externas como polvo y humedad.
- Facilitar el mantenimiento y acceso a los componentes internos.
- Mejorar la estética y portabilidad del sistema.





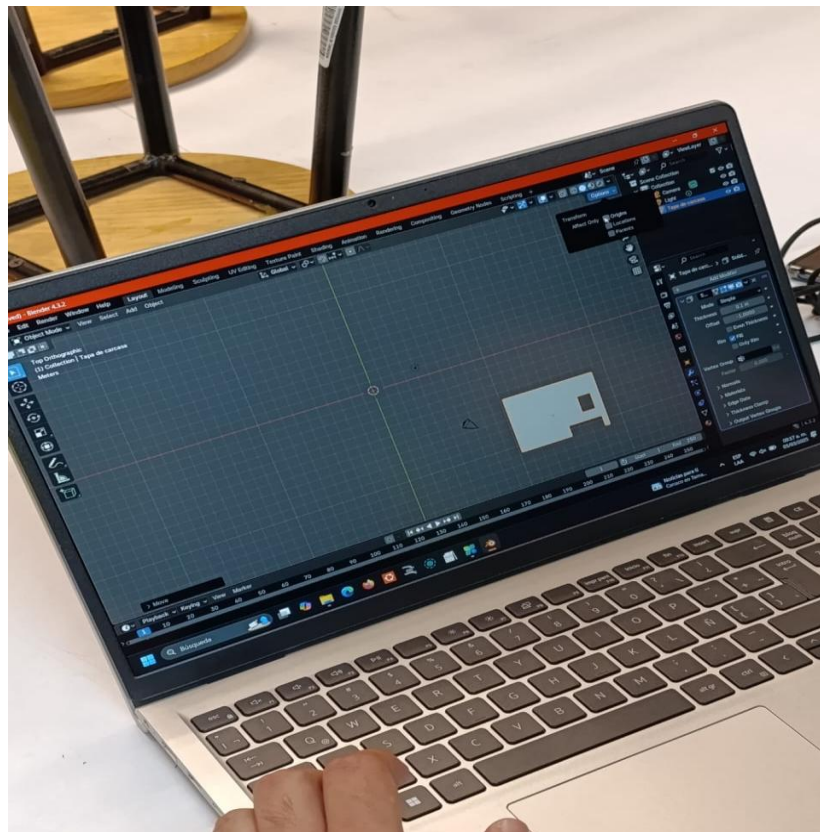
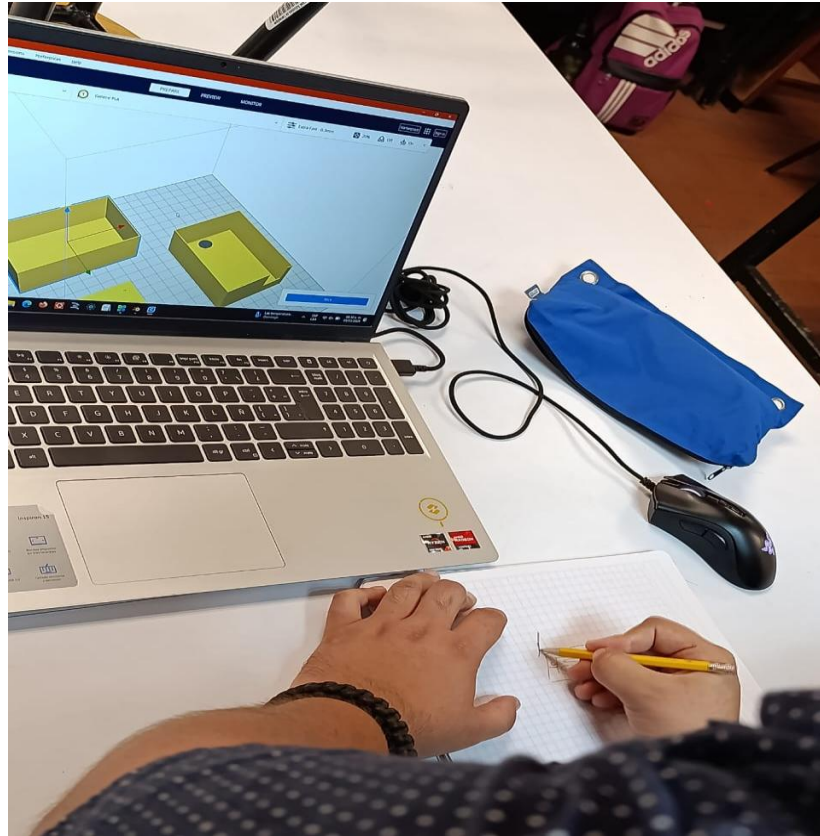


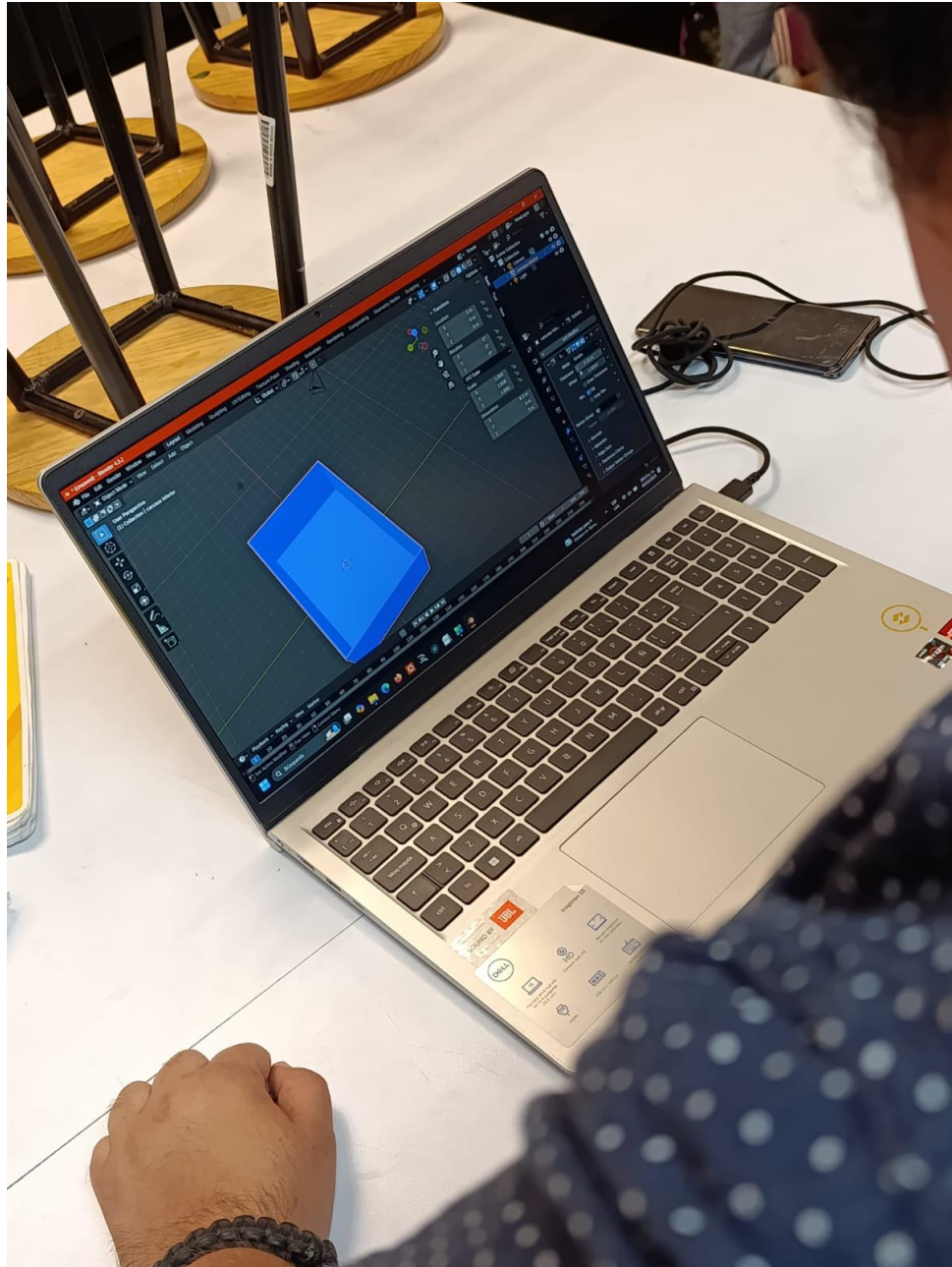




Unidad Académica  
Multidisciplinaria  
Mante

**UAT**  
Universidad Autónoma de  
**TAMAULIPAS**

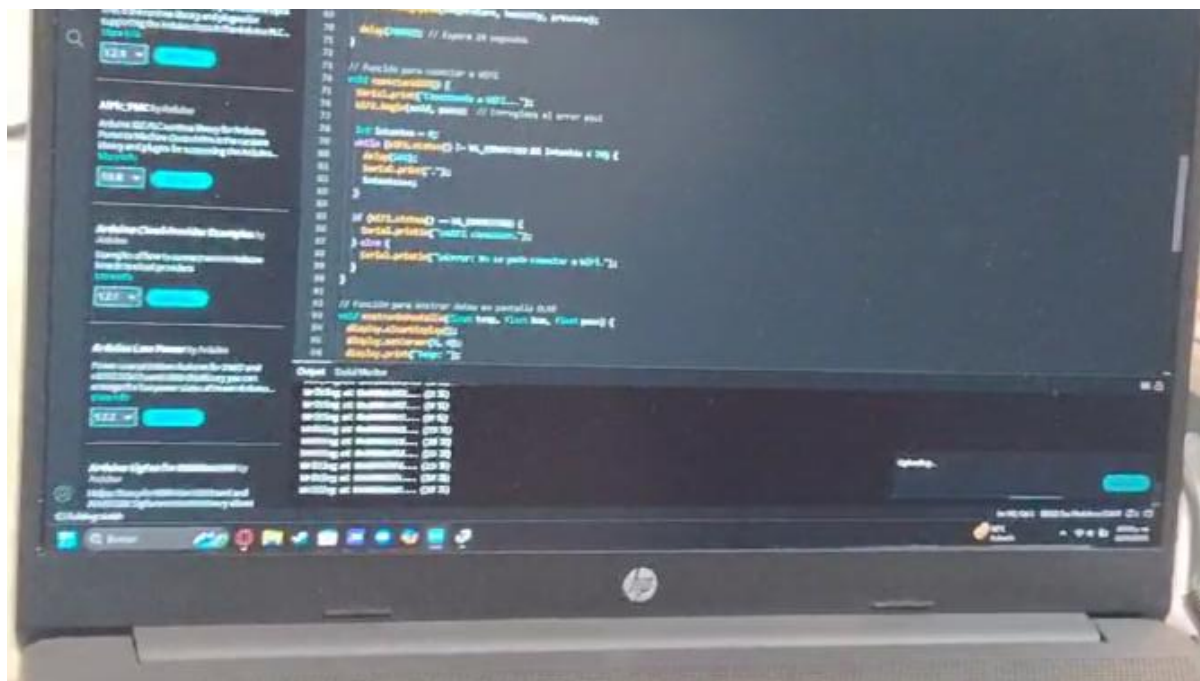
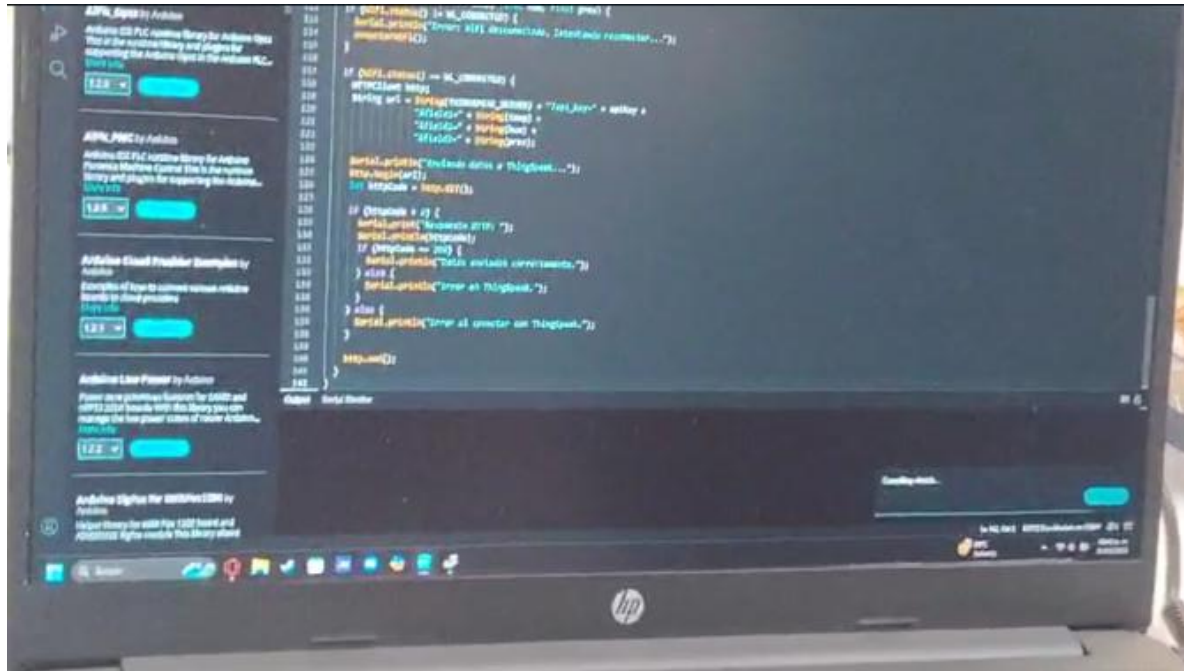


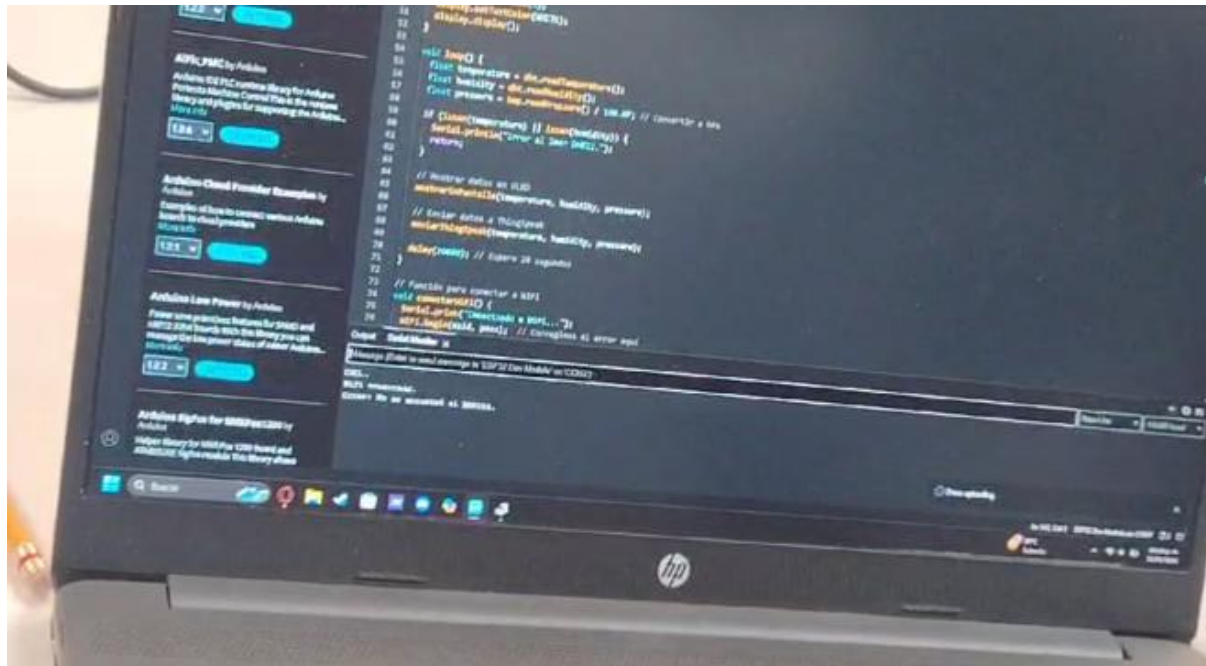




## 2.6 Pruebas y resultados

- Capturas de pantalla de los datos en la OLED y ThingSpeak.
- Comparación de valores con otros dispositivos de referencia para verificar la precisión.









## **Conclusión**

Aun que tuvimos problemas con el DHT22 ya que encontramos un error y no lo solucionamos a tiempo El desarrollo de este sistema meteorológico basado en ESP32 demostró ser una solución eficiente para el monitoreo en tiempo real de temperatura, humedad y presión atmosférica. La integración con ThingSpeak permitió la visualización remota de los datos, facilitando su análisis y almacenamiento. En futuras versiones, se podrían agregar más sensores para obtener información más detallada, o mejorar la interfaz gráfica de la pantalla OLED para una mejor experiencia de usuario.



#### 4. Bibliografía

- Datasheet del sensor DHT22:  
<https://www.adafruit.com/datasheets/DHT22.pdf>
- Datasheet del sensor BMP280: <https://www.bosch-sensortec.com/products/environmental-sensors/pressure-sensors/bmp280/>
- Documentación de ESP32: <https://docs.espressif.com/projects/espressif/en/latest/>
- Documentación de ThingSpeak:  
<https://www.mathworks.com/help/thingspeak/>



**Unidad Académica  
Multidisciplinaria  
Mante**

