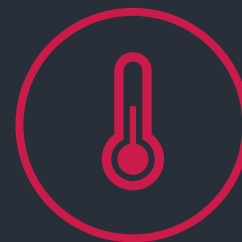


**MICROCONTROLADORES  
ESP8266 & ARDUINO**

# **Sistema Contra**

# **Incendios**



# LLUVIA DE IDEAS



**01**

Estación meteorologica con alarmas de tiempo  
-No soluciona algo importante

**02**

Control remote  
-No tiene programación web

**03**

Sistema de alarma sobre incendios o gases

**04**

Cerradura electronica  
-Muy parecido al Proyecto del led

# LA PROPUESTA GANADORA

Sistema contra incendios y detector de gases



## ¿POR QUÉ?

- Es un Proyecto que puede salvar cosas materiales e incluso puede llegar a salvar vidas.
- Cumple con todos los requisitos del proyecto



## FUNCIONES

- Se monitorea en todo momento tanto la temperatura como los niveles de gases que detecta el sensor MQ2.
- En caso de que algún nivel suba mucho se enciende la alarma y activa las alarmas sonoras, visual y También tiene poder para levantar un Sistema externo por ejemplo un Sistema de riego.

# CASOS DE USO



Detectar incendios para poder ser sofocados a tiempo

Deteccion de fugas de gas licuado en estaciones de gas

Deteccion de altas temperaturas

Poder detector alto nivel de contaminacion en el ambiente

Centro de comandos para un sistema de sofocación de fuego

# LA CONSTRUCCIÓN DE LA IDEA

Antes de programar o comenzar a comprar todo, primero se tiene que hacer un análisis y para ello necesitamos:

01

Diagrama de flujo

02

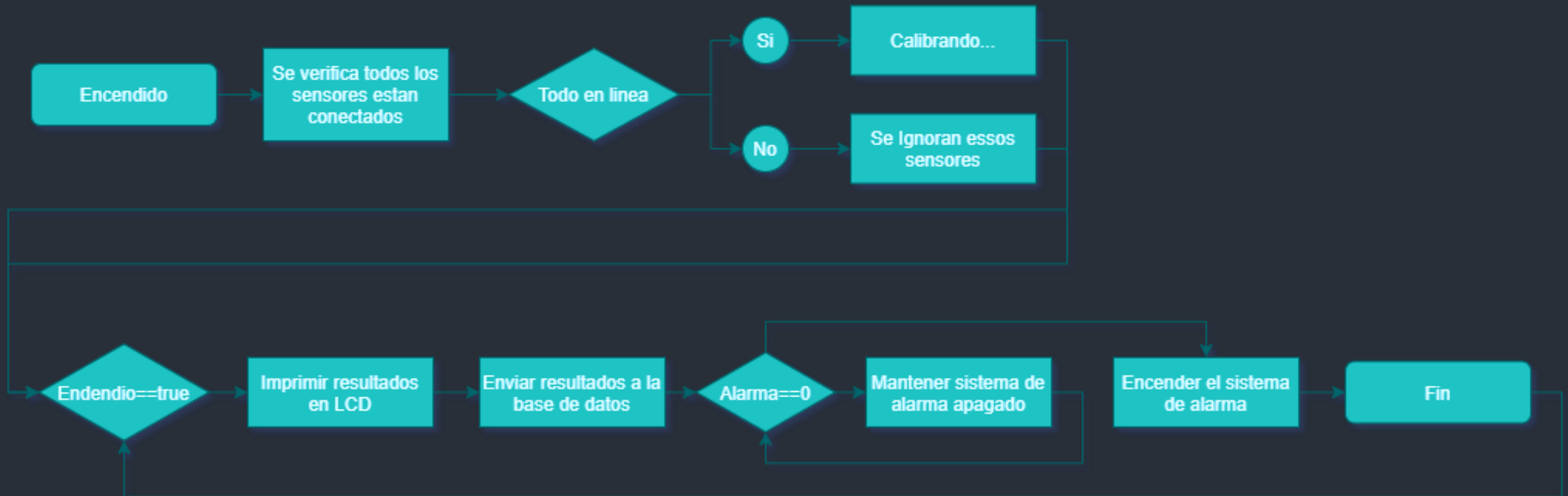
Diagrama de bloques

03

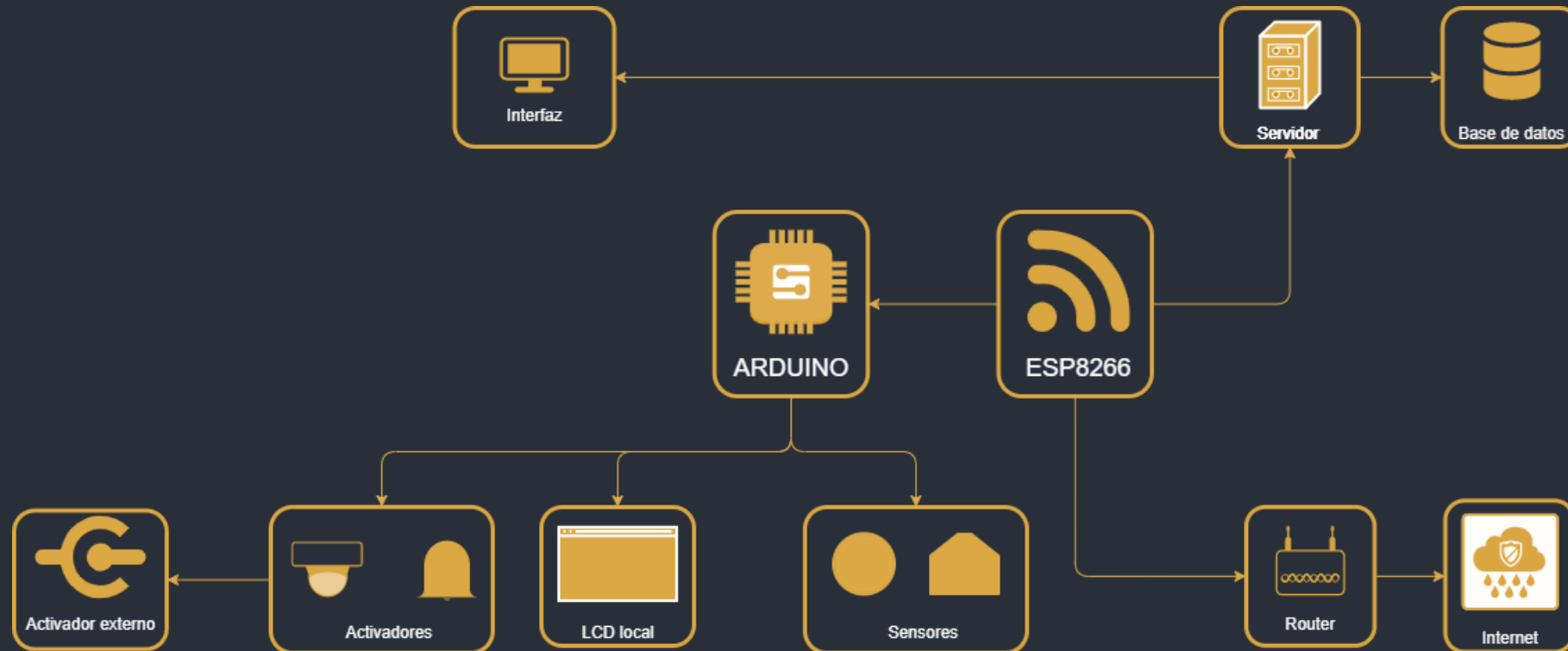
Diagrama electronico



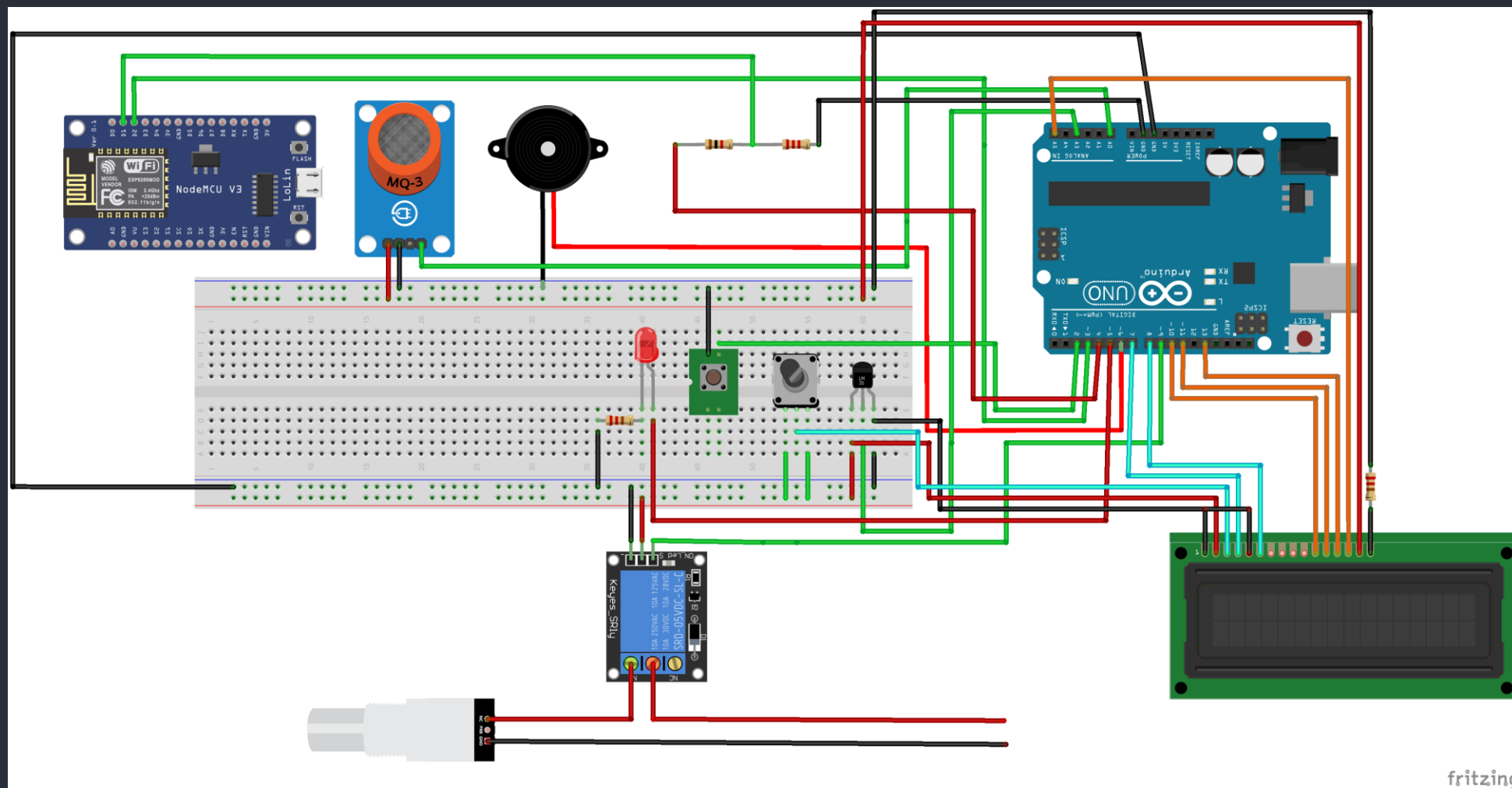
# DIAGRAMA DE FLUJO



# DIAGRAMA DE BLOQUES



# DIAGRAMA ELECTRÓNICO

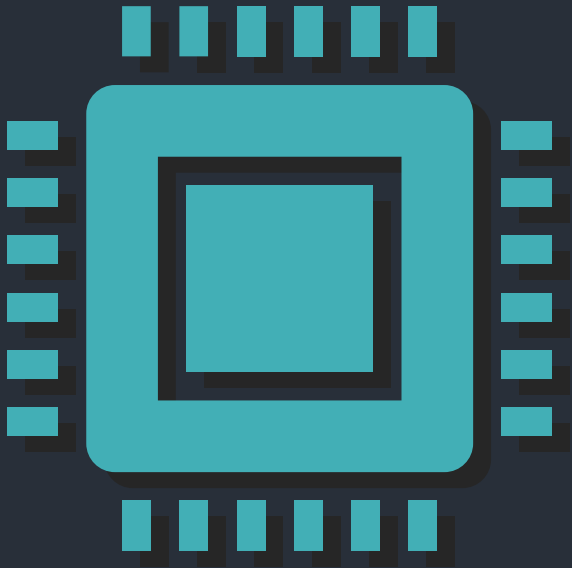




# PROGRAMACION Y CONSTRUCCION



# MATERIALES



- Microcontrolador Arduino UNO
- NodeMCU ESP8266 LUA
- Protoboard de 820 puntos
- Sensor de gases MQ2
- Sensor de temperatura LM35
- Display LCD 1602A
- Buzzer activo
- Relay 5V-110V
- Potenciómetro 10k
- Tira de 40 pines macho recto
- Resistencias de 220 ohms
- Resistencias de 2.2K
- Resistencias de 1K
- Cables jumpers dupont 20 cm
- Caja de 30x15x20 cm
- Enchufe eléctrico
- Contacto eléctrico

# CODIFICACIÓN DE ARDUINO



Se unieron todos los componentes de acuerdo al diagrama mostrado en la diapositiva 7 pero a modo de prueba sin aun montar todo para poder hacer cambios en caso de necesitarlos

```
Code_Arduino 1.8.15 Hourly Build 2021/05/31 10:33
Archivo Editar Programa Herramientas Ayuda

Code_Arduino

#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

const byte espRx = 3;
const byte espTx = 4;

// alarma
const byte ledAlarm = 5;
const byte buzzerAlarm = 6;
const byte relay = 9;

// led
const byte rs = 7;
const byte e = 8;
const byte d4 = 10;
const byte d5 = 11;
const byte d6 = 13;
const byte d7 = 19; //A5

// sensores
const byte mq = A0;
const byte lm35 = A3;

// boton
const byte ack = 2; // pin capaz de interrumpir
/**** Pin References *****/

// variables globales
long co = 0; // (if saturated > 65535)
long lpg = 0;
long smoke = 0;
float temperature = 0;
bool generalAlarm = false; // si se dispara alguna alarma será cierto
volatile bool ackPressed = false;

// set de alarma
const int lpgSet = 2100;
const int coSet = 200;
const int smokeSet = 1150;
const float temperatureSet = 45.0; // °C

// Instancias
SoftwareSerial SerialEsp(espRx, espTx); // RX, TX
LiquidCrystal lcd(rs, e, d4, d5, d6, d7); // Crea un objeto LCD

// Tiempos
long startingLoopTime = 0;
long currentLoopTime = 0;
const int loopTimeSet = 1000; // 1.5s

long startingSerialTime = 0; // para enviar datos a esp
long currentSerialTime = 0;
const int serialTimeSet = 10000; // 30s

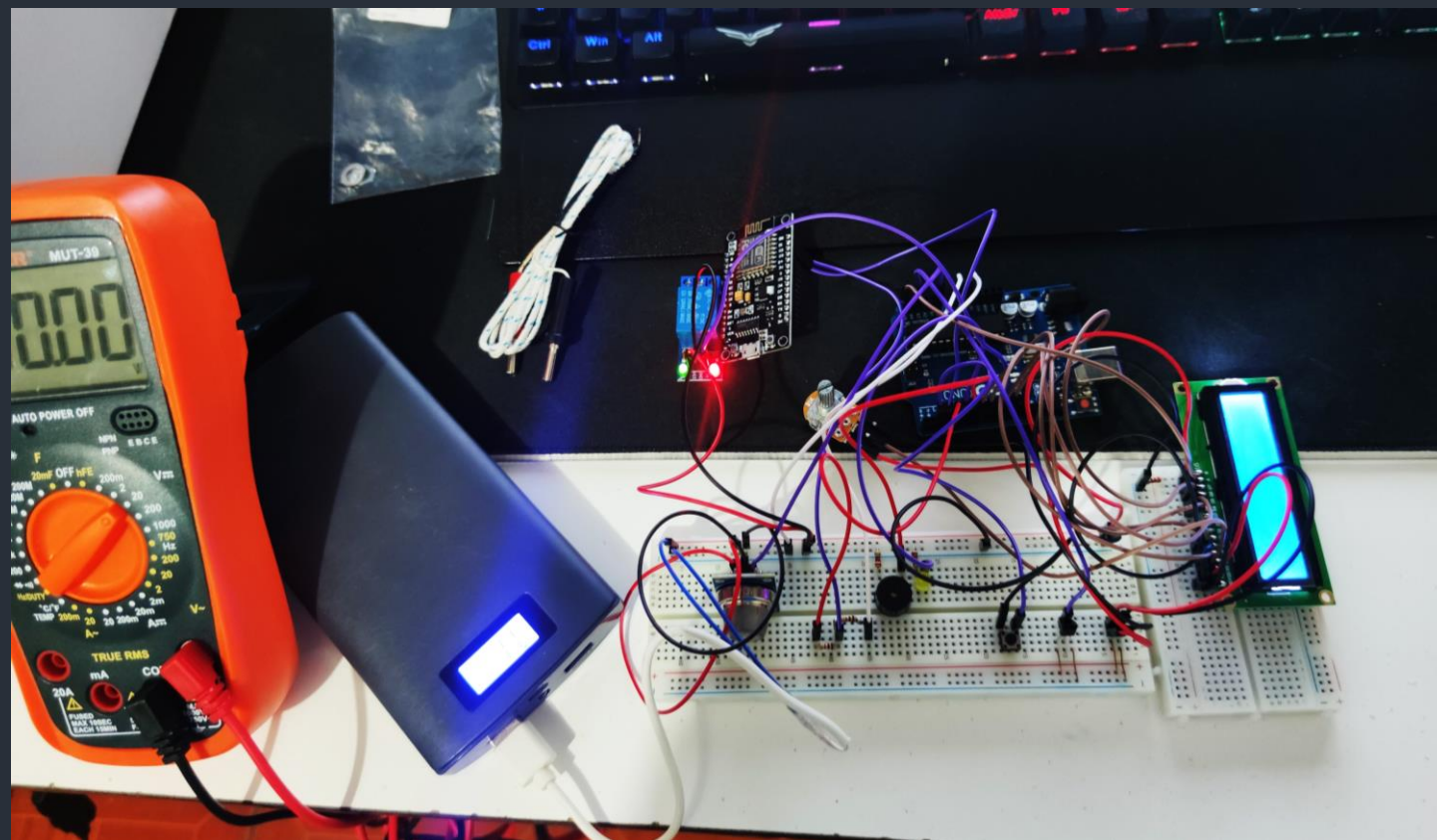
// Caracter personalizado de campana
byte bell[8] = {
  0b0100,
  0b0110,
  0b0110,
  0b0110,
  0b1111,
  0b1111,
  0b1111,
  0b0000
}; // eof bell

/***** MQ 1 GASRES *****/
/*****Macros relacionadas con el hardware*****/
// ledAlarm = 4; //cuando comience la calibración, el LED pin 13 se iluminará, se apagará cuando
//mq = A0; //define qué canal de entrada analógica va a utilizar
int RL_VALUE = 1; //definir la resistencia de carga en la placa, en kilo ohms
4
```

# UNIÓN



Se carga el sketch en el Arduino y se unieron todos los componentes de acuerdo al diagrama mostrado en la diapositiva 7 pero a modo de prueba sin aun montar todo para poder hacer cambios en caso de necesitarlos



# CODIFICACIÓN DE ESP8266



Una vez que comprobamos que el programa de Arduino funciona y que el circuito esta trabajando de forma correcta procedemos a programar el código para el modulo NodeMCU ESP8266 para que este envíe los datos al servidor y los guarde en la base de datos

(Mas sin embargo no se sube el programa hasta que tengamos el servidor simulado y la base de datos creada)

```
Code_ESP_Arduino 1.8.15 Hourly Build 2021/05/31 10:33
Archivo Editor Programa Herramientas Ayuda

Code_ESP
#include <ESP8266WiFi.h>
#include <SoftwareSerial.h>

#define SSID "Totalplay-459D"
#define PASS "459D3G09AXhw2eN"

const char* ssid = SSID;
const char* password = PASS;

const char* host = "192.168.100.11";
const uint16_t port = 80;

const byte espRx = 5;
const byte espTx = 4;
SoftwareSerial SerialEsp(espRx, espTx); //se utilizan los puentes RX y TX

// RX msg
bool received = false;
String receiveMsg;

// variables to control
String s_alarm;
String s_co;
String s_smoke;
String s_lpg;
String s_temp;
const byte numVars = 5;
//
String controlledVars[] = {s_alarm, s_co, s_lpg, s_smoke, s_temp};

// variables float y int
int alarm = 0;
int co = 0;
int lpg = 0;
int smoke = 0;
float temp = 0.0;

void setup()
{
  Serial.begin(115200);
  SerialEsp.begin(9600);

  // Empezamos por conectarnos a una red WiFi

  Serial.println();
  Serial.println();
  Serial.print("Conectando a ");
  Serial.println(ssid);

  /*Configure explícitamente el ESP8266 para que sea un cliente WiFi, de lo contrario,
  * de forma predeterminada, intentaría actuar como cliente y punto de acceso y podría
  * causar problemas de red con sus otros dispositivos WiFi en su red WiFi. */

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi conectado");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void loop()
{
  checkSerialCom();
  if (received){
    ProcessMsg();
    receiveMsg = "";
    received = false;
  }
}
```

# CREACIÓN DE LA BASE DE DATOS

13



Se crea la base de datos en el servidor que hayamos creado o comprado, en este caso la instalación se hará en un localhost mediante XAMPP y el administrador de base de datos que se usará será phpMyAdmin

phpMyAdmin

Reciente Favoritas

Nueva

- esp8266datos
- information\_schema
- mysql
- performance\_schema
- phpmyadmin
- test

Servidor: 127.0.0.1 » Base de datos: esp8266datos » Tabla: Estadísticas

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento Disparadores

Nombre de la tabla: Estadísticas Agregar 1 columna(s) Continuar

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	Comentarios	Virtualidad	Mover columna
id	INT		Ninguno				PRIMARY			
Selec...							PRIMARY			
alarma	BOOLEAN		Ninguno							
Selec...										
co	INT		Ninguno							
Selec...										
lpg	INT		Ninguno							
Selec...										
humo	INT		Ninguno							
Selec...										
temp	FLOAT		Ninguno							
Selec...										
fecha	TIMESTAMP		CURRENT_TIME							
Selec...										

Estructura

Comentarios de la tabla: Cotejamiento: Motor de almacenamiento: InnoDB

definición de la PARTICIÓN:

Dividido por: ( Expresión o lista de columna )

Particiones:

Consola

# CONEXIÓN CON LA BASE DE DATOS



Se crea la conexión con la base de datos para poder utilizarla con el modulo esp8266 y también con la pagina web

```
1 <?php
2 // db variables
3 $host = "localhost";
4 $dbname = "esp8266datos";
5 $user = "root";
6 $password = "";
7
8
9
10 try{
11     $conn = new PDO("mysql:host=$host;dbname=$dbname", $user, $pa
12     echo "<br>Conexión exitosa <br><br>";
13 }
14 catch(Exception $e){ //PDO Exception obj
15     die('Error: ' . $e->getMessage()); // kill the process
16 }
17
18 // $conn = null; // at the end we free memory and resources
19 ?>
```

# PRUEBA DE LA CONEXIÓN CON LA BASE DE DATOS



Entramos a la dirección `localhost/esp8266/connection.php` y comprobamos que el acceso a la base de datos sea correcto

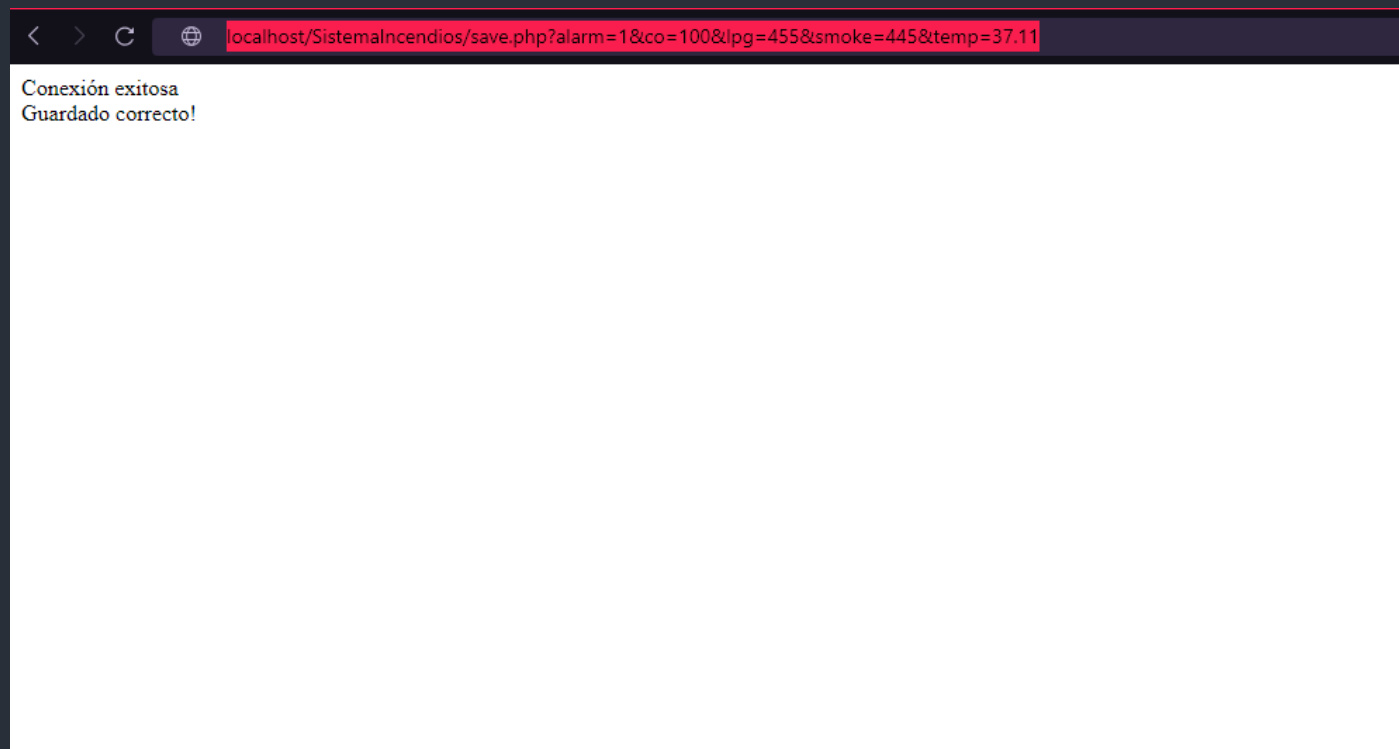
Conexión exitosa



# JQUERY



Usamos la función jQuery y mandamos información como parámetros para que se haga el guardado y comprobamos



# CREACIÓN DE LA PAGINA WEB



Para la parte de la pagina web se dividió en 2 archivos, el primero es un archivo .HTML el cual preestablece la pagina, así como su canvas y algunos otros parámetros y el segundo que es .PHP se dice mas la lógica y el diseño completo de la pagina web, así como la solicitud a la base de datos

```
1 <doctype html>
2 <html lang="en">
3
4 <head>
5   <!-- Required meta tags -->
6   <meta charset="utf-8">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8
9   <!-- Bootstrap CSS -->
10  <link rel="stylesheet" href="node_modules/bootstrap/dist/css/bootstrap.min.css">
11
12  <title>Sistema Incendios</title>
13
14  <style>
15    body {
16      background-color: #FFFFFF;
17    }
18
19    .myTitle {
20      color: #000000;
21      font-size: 200%;
22      text-align: center;
23      font-weight: bold;
24      font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Sans', 'Arial', sans-serif;
25      margin: 10px 0;
26    }
27
28    .rule {
29      height: 3px;
30      width: 100%;
31      background: #000000;
32      margin-top: 0;
33    }
34  </style>
35 </head>
36
37 <body class="container-fluid">
38   <div class="myTitle">MONITOREO DE VARIABLES</div>
39   <div class="rule"></div>
40   <div class="row">
41     <div class="col-12 col-lg-4">
42       <canvas id="myChart" width="400" height="400"></canvas>
43     </div>
44     <div class="col-12 col-lg-4">
45       2 of 2
46     </div>
47     <div class="col-12 col-lg-4">
48       3 of 3
49     </div>
50   </div>
51   <div class="row">
52     <div class="col-12 col-lg-4">
53       1 of 3
54     </div>
55     <div class="col-12 col-lg-4">
56       2 of 3
57     </div>
58     <div class="col-12 col-lg-4">
59       3 of 3
60     </div>
61   </div>
62
63   <!-- Optional JavaScript -->
64   <!-- jQuery first, then Popper.js, then Bootstrap JS -->
65   <script src="node_modules/jquery/dist/jquery.slim.js"></script>
66   <script src="node_modules/popper.js/dist/popper.js"></script>
67   <script src="node_modules/chart.js/dist/chart.min.js"></script>
68   <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
69
70   <script>
71     var samples = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
72     var ctx = document.getElementById('myChart').getContext('2d');
73     var chart = new Chart(ctx, {
74       // The type of chart we want to create
75       type: 'line',
76     });
77   </script>
```

# CREACIÓN DE LA PAGINA WEB



.PHP

```
index.php - esp8266 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.php index.html connection.php index.php x save.php
EXPLORER
ESP8266
node_modules
connection.php
index.html
index.php
package-lock.json
save.php
index.php
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <meta http-equiv="X-UA-Compatible" content="ie=edge">
7 <title>Sistema Incendios</title>
8
9 <!-- Bootstrap CSS -->
10 <link rel="stylesheet" href="node_modules/bootstrap/dist/css/bootstrap.min.css">
11 <style>
12     body {
13         background-color: #FFFFFF;
14         color: black;
15     }
16
17     .div.col-12 {
18         /* background-color: gray; */
19     }
20
21     .myTitle {
22         color: #00151a;
23         font-size: 200%;
24         text-align: center;
25         font-weight: bold;
26         font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'sans-serif';
27         margin: 10px 0;
28     }
29
30     .rule {
31         height: 3px;
32         width: 100%;
33         background-color: #006972;
34         margin-top: 0;
35     }
36
37     #tableModal {
38         color: #000000;
39     }
40 </style>
41 </head>
42 <body>
43     <!-- form action="" method="POST" -->
44     <input type="text" name="dateRequest">
45     <input type="submit" name="submit" value="Search">
46 </form> -->
47
48 <!-- canvas parts graphics test for now -->
49 <div class="myTitle">SISTEMA CONTRA INCENDIOS</div>
50 <hr class="rule">
51 <div class="row">
52     <div class="col-12 col-lg-4">
53         <canvas id="coChart" width="" height="200"></canvas>
54     </div>
55     <div class="col-12 col-lg-4">
56         <canvas id="lpgChart" width="" height="200"></canvas>
57     </div>
58     <div class="col-12 col-lg-4">
59         <canvas id="smokeChart" width="" height="200"></canvas>
60     </div>
61 </div>
62 <div class="row">
63     <div class="col-12 col-lg-4">
64         <canvas id="tempChart" width="" height="200"></canvas>
65     </div>
66     <div class="col-12 col-lg-4">
67         <canvas id="alarmChart" width="" height="200"></canvas>
68     </div>
69 </div>
70 <div class="col-12 col-lg-4">
71     <div class="row col-12 d-flex justify-content-center">
72     </div>
73     <div class="row col-12 d-flex justify-content-center">
74         <!-- Modal button -->
75     </div>
76 </div>
77 </body>
</html>
```

# PRUEBA DE LA PAGINA WEB



Entramos a la dirección donde hemos alojado la pagina web y comprobamos que este funcionando tanto la pagina web como la obtención de datos de la BD



# PAGINA WEB RESPONSIVA



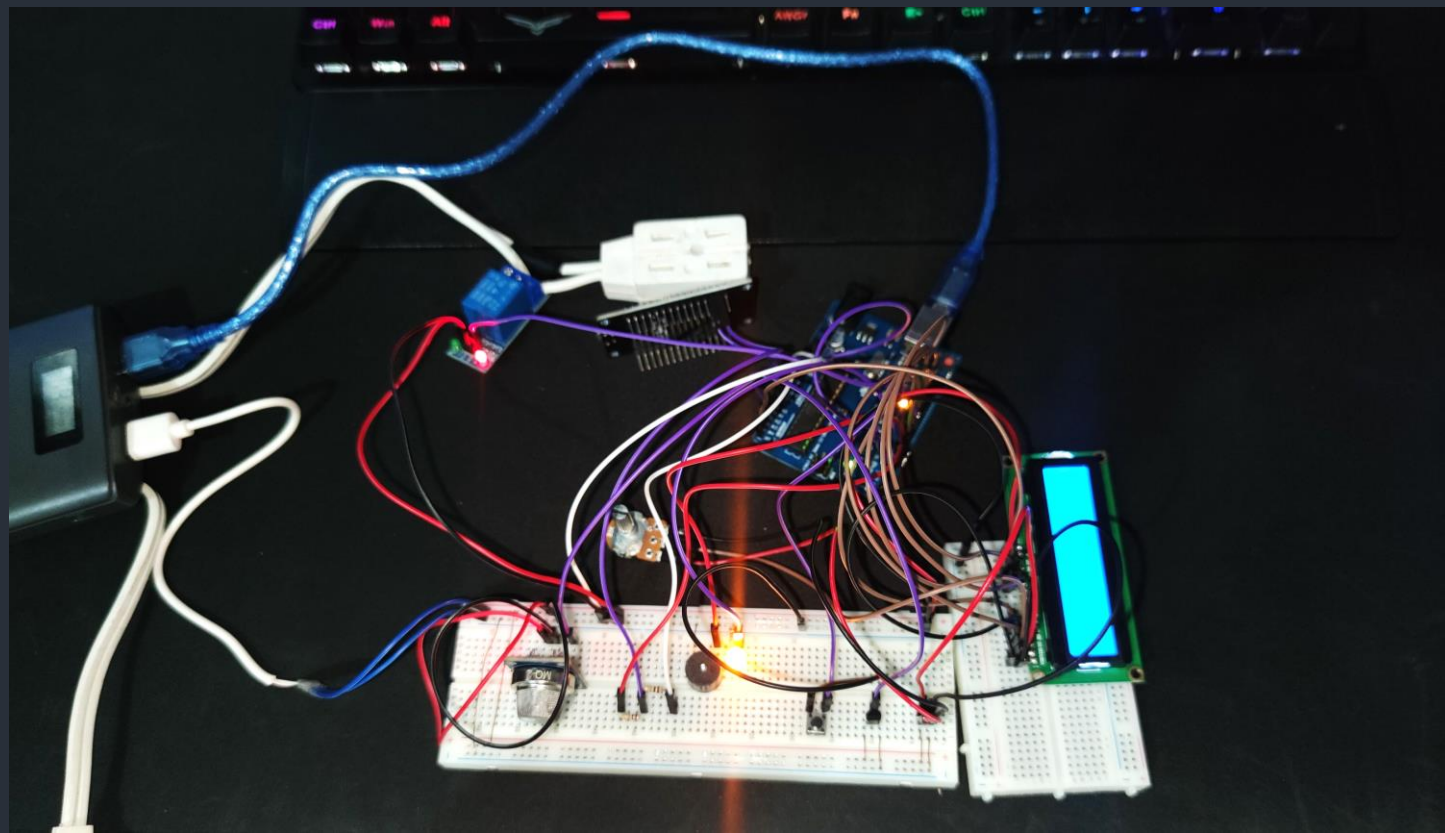
# PROCESO DE CONTRUCCIÓN



# PRUEBAS PREFINALES



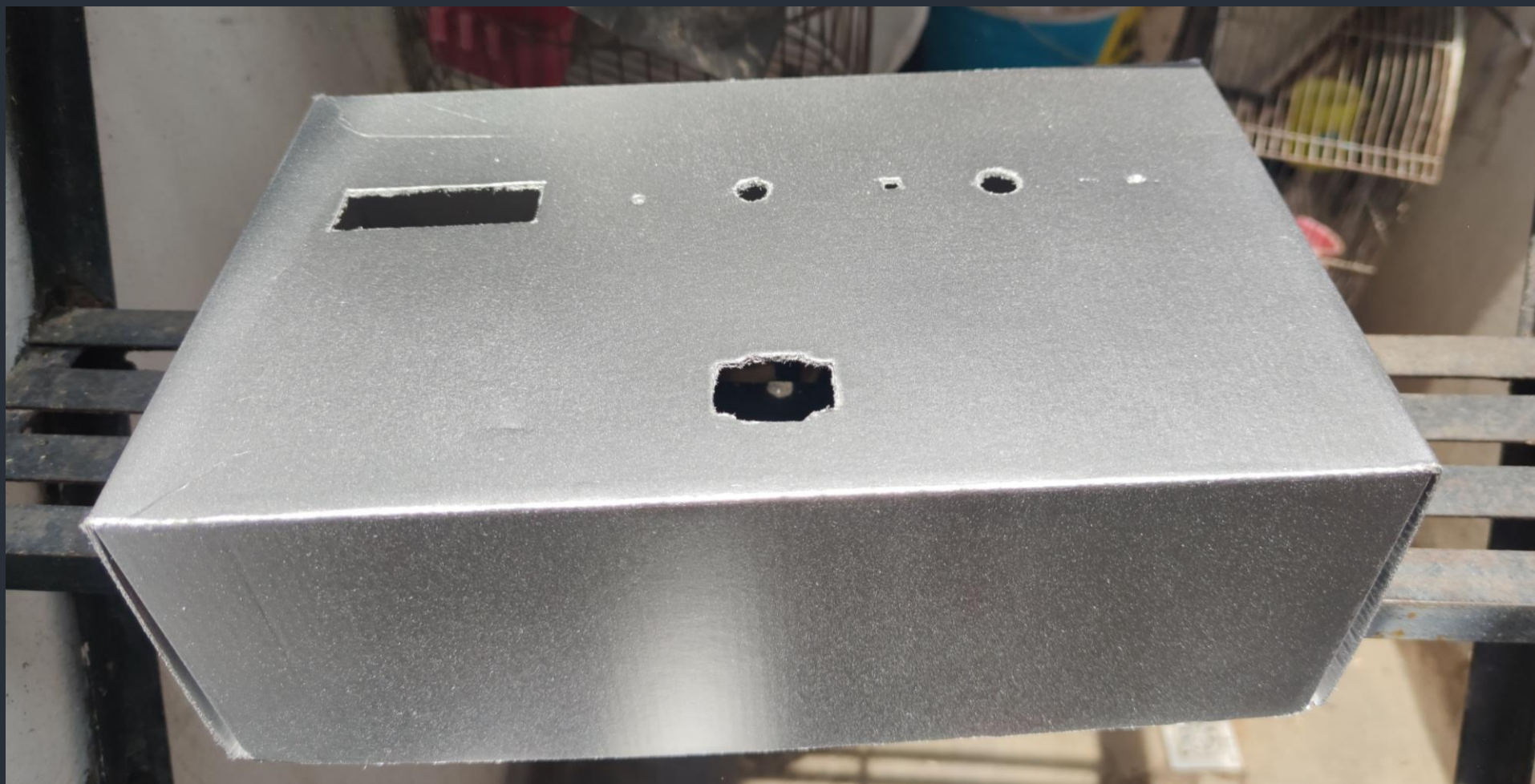
Se hace pruebas con todo el setup ya conectado entre si





# GABINETE DE MONTAJE

23





# PROCESO DE MONTAJE

24



# PRUEBAS FINALES





# MONTAJE FINAL



# REQUISITOS



- ✓ **Aplicar todo lo visto en el taller**
- ✓ **Resolver una problematica**
- ✓ **Debe tener front-end**
- **Debe tener back-end**
- ✓ **Debe ser responsivo**
- ✓ **Debe tener IOT**

# LENGUAJES USADOS



# EXTENSIONES USADAS



# CÓDIGO EN GITHUB

30



# PRESENTADOR Y ALUMNO



**Gustavo de Jesus Quintero Cerpa**  
219294307