



REDES DIGITALES

SISTEMA DE SEGURIDAD

PROYECTO FINAL

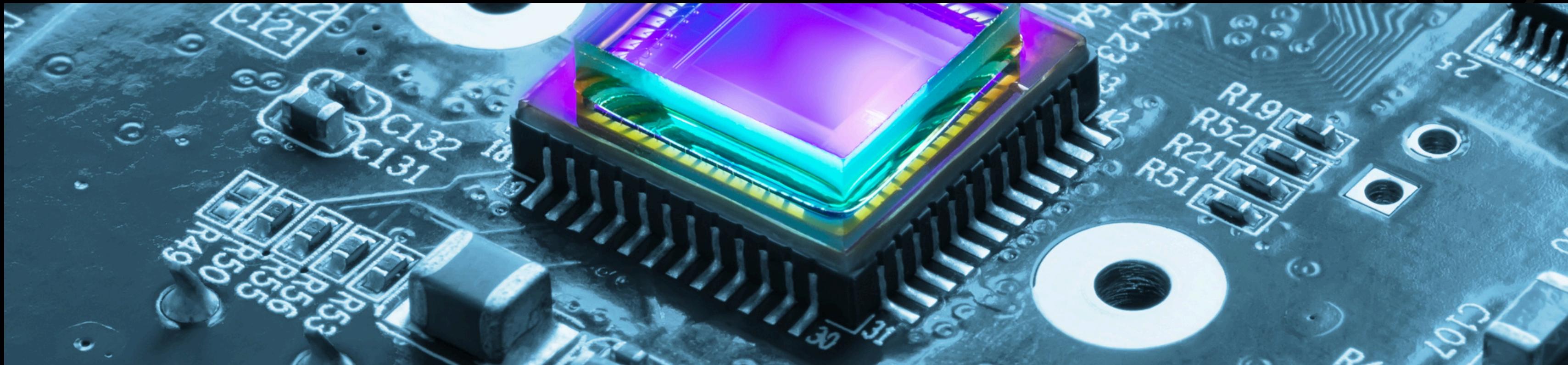


SISTEMA DE SEGURIDAD - HOGAR

El propósito de este proyecto es ofrecer una solución efectiva y económica para un sistema de seguridad residencial, que proporcionará las herramientas necesarias para el usuario del hogar.

Además, el sistema incluirá características avanzadas como cámaras de seguridad, alertas en tiempo real a través de un sitio web y la capacidad de integrarse con otros dispositivos inteligentes del hogar. Se priorizará la facilidad de instalación y uso, asegurando que incluso aquellos sin conocimientos técnicos puedan beneficiarse de la seguridad mejorada que ofrece este sistema.





REQUERIMIENTOS

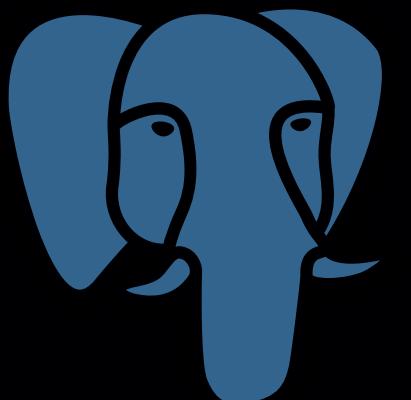
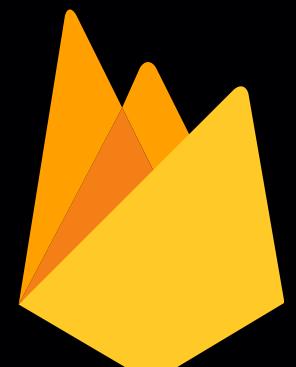
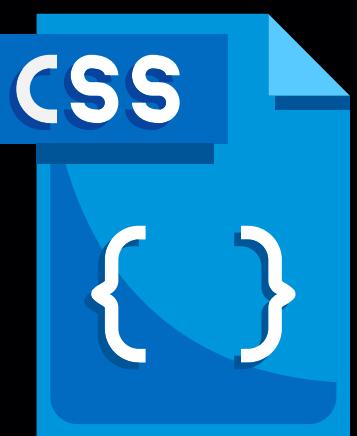
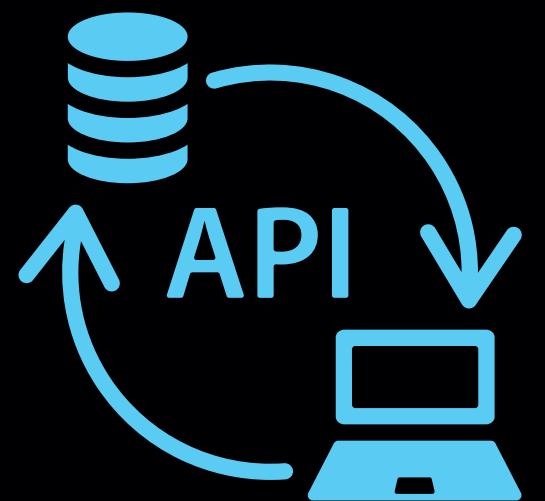
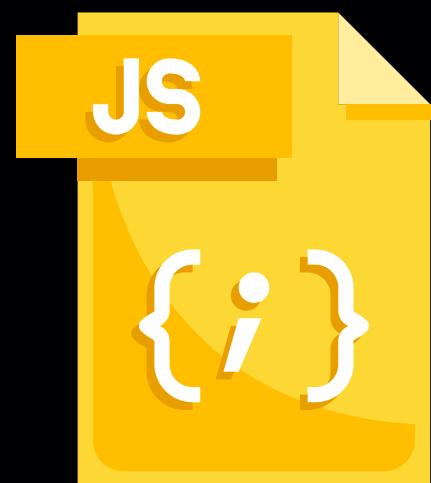
Para el siguiente proyecto, se requiere que se cumpla lo siguiente:

- Uso de WIFI y ESP32.
- Uso de sensores y conectarlos a internet.
- Dashboard para mostrar parametros tomados por sensores.
- Inicio de sesión.
- 6 dispositivos.



TECNOLOGÍAS

Estas fueron las tecnologías usadas en este proyecto:





API



Una **API** (application programming interface) es una manera de enviar y recibir datos mediante el uso de HTTPS como manera principal de comunicación. Esta manera de enviar datos es reconocida por su facilidad y versatilidad. Una API cuenta con los siguientes métodos.

GET: Obtiene la información de determinada query.

POST: Envía información y crea nuevas queries.

PUT: Edita información en una determinada query.

DELETE: Borra información y queries.



API

Su funcionamiento es apartir de un **servidor web**, mismo que aloja esta API. En este caso se utilizó **Render** y **python**, esto para lograr hacer funcionar este servidor web. Después se creo una base de datos **postgres**, misma que también fue creada en render para alojar los datos.

The screenshot shows the Render web service dashboard for the project 'redes-practice'. The top navigation bar indicates it's a Python 3 application running on a free instance. The main area displays deployment logs, with two entries visible: a successful deployment at 8:56 PM on July 1, 2024, and a first deployment at 8:52 PM on the same date. A sidebar on the left lists various management options like Events, Logs, Disks, Environment, Shell, Previews, Jobs, Metrics, Scaling, and Settings.

Haciendo un **GET** al link: <https://redes-practice.onrender.com/api/state>

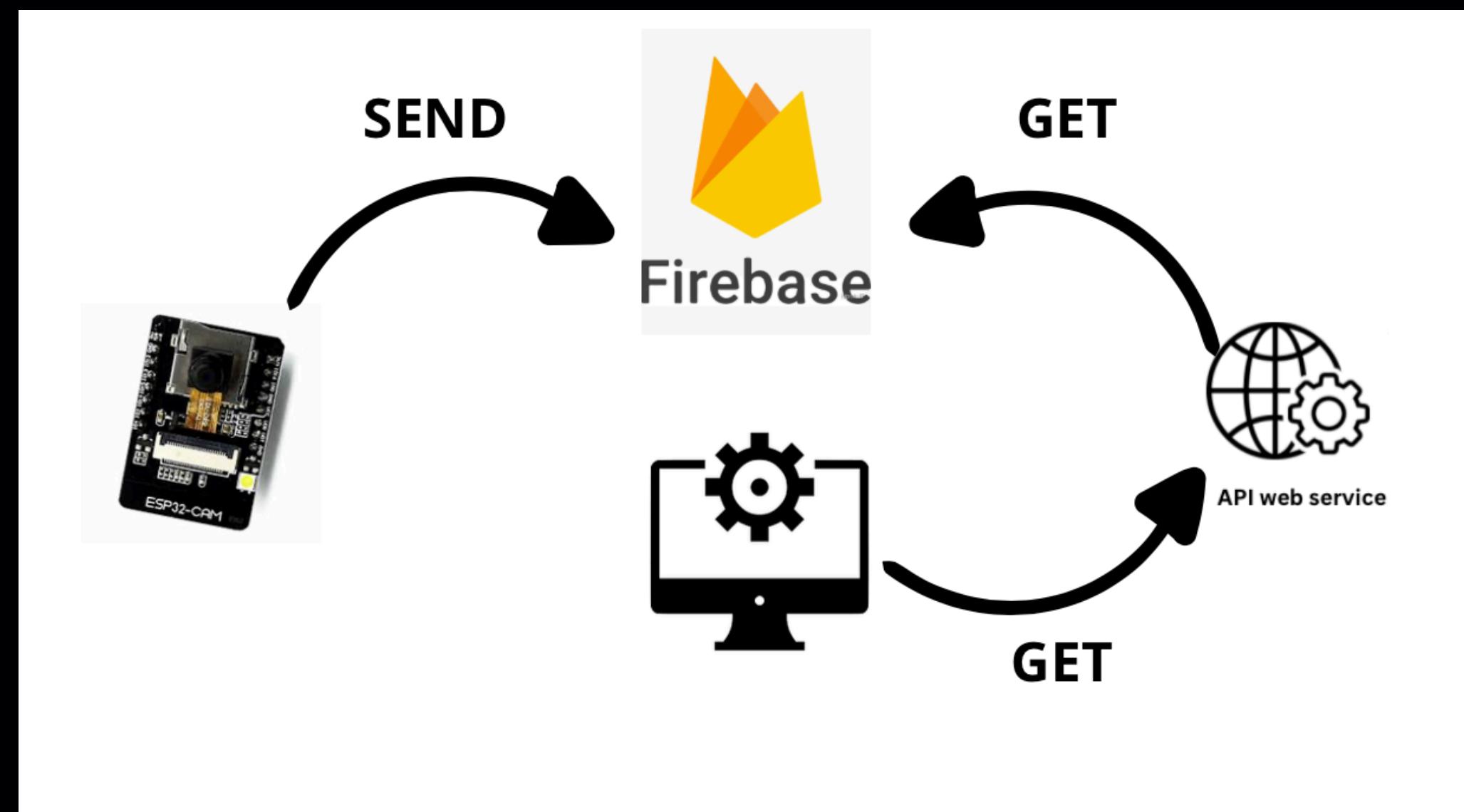
The screenshot shows a terminal window displaying a JSON object. The object contains several fields: 'doorbell' (false), 'doorstatus' (Closed), 'id' (4), 'laser' (Seguro), 'temperature' (40.6), and 'windowstatus' (Open). The JSON is displayed in a monospaced font against a dark background.

```
"doorbell": "false",
"doorstatus": "Closed",
"id": 4,
"laser": "Seguro",
"temperature": 40.6,
>windowstatus": "Open"
```



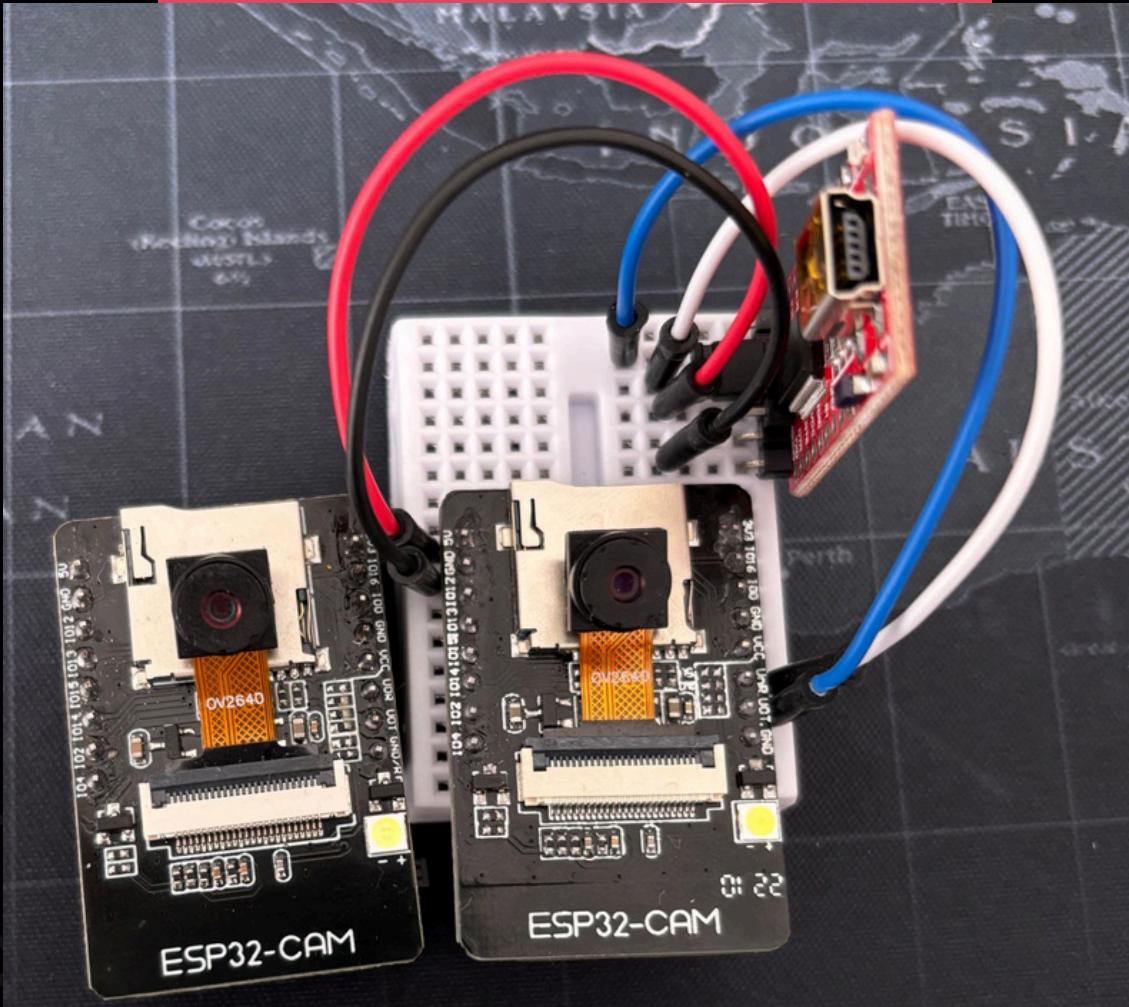
FUNCIONAMIENTO DE CAMARAS

Este es un esquema de como funcionan las cámaras:





CAMARAS DE VIGILANCIA



Cámara 1: interior Miguel Carrera

La Cámara 1 está diseñada para la vigilancia en interiores, ideal para monitorear espacios dentro de una casa. Captura imágenes a intervalos de 15 segundos y las guarda temporalmente en la memoria interna de la ESP32-CAM. Luego, las imágenes son subidas a Firebase Storage, se recuperan con una solicitud GET para guardarlas en una API, y finalmente, otra solicitud GET las envía a una página web.

Cámara 2: exterior Erick Guevara

La Cámara 2 se utiliza para vigilancia en exteriores, monitoreando áreas como patios o entradas de la casa. Captura imágenes periódicamente y las sube a Firebase Storage. Luego, las imágenes se recuperan con una solicitud GET para guardarlas en una API y, mediante otra solicitud GET, se envían a la página web para una vigilancia continua y accesible.



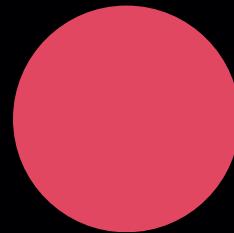
Intervalo de 15 segundos: la elección ideal para captura y subida de imágenes

El intervalo de **15 segundos** entre la captura de imágenes en la ESP32-CAM ha sido elegido tras evaluar diferentes períodos y sus impactos en el rendimiento del sistema. Este tiempo permite realizar todo el ciclo de captura, procesamiento y subida de manera eficiente y confiable.

Evaluación de Intervalos:

- 5 segundos: Insuficiente, provocando pérdidas de imágenes y fallos en la subida.
- 10 segundos: Todavía ajustado, con riesgos de fallos en condiciones de red variables.
- **15 segundos**: Óptimo, proporcionando el tiempo necesario para todas las operaciones sin saturar los recursos.
- 20 y 25 segundos: Intervalos más largos de lo necesario, reduciendo la frecuencia de imágenes y, por tanto, la eficacia del monitoreo.





Configuración de firebase

Firebase esp32 ▾ ☰

Descripción gene... | ☰

IA generativa

Build with Gemini NUEVO

Accesos directos a proyectos

Storage Subir archivo ☰

Authentication

Categorías de producto

Compilación

Ejecución

Analytics

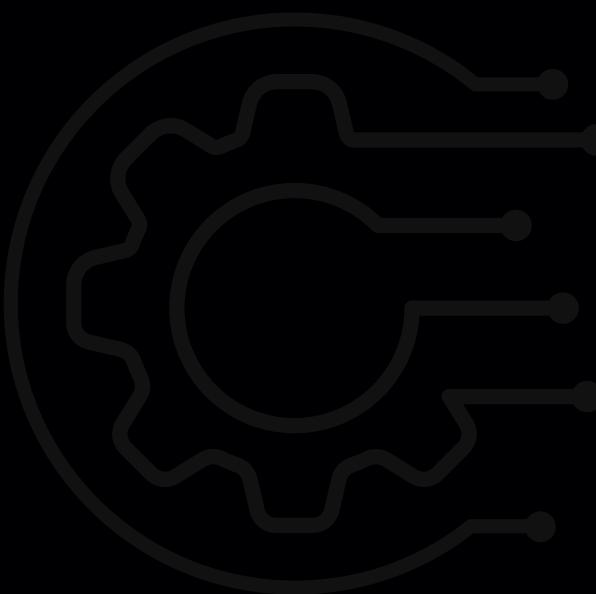
Todos los productos

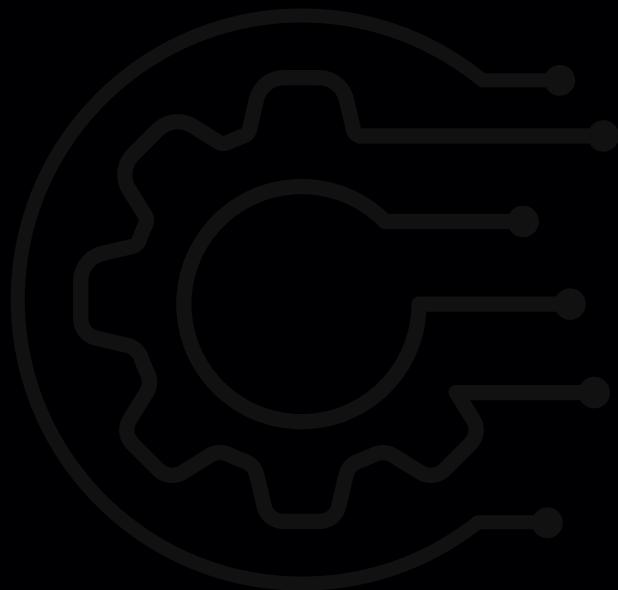
Storage

Archivos Reglas Uso Extensions

Protege tus recursos de Storage contra los abusos, como fraudes de facturación o phishing. Configurar la Verificación de aplicaciones X

Nombre	Tamaño	Tipo	Modificación más reciente
Exterior/	—	Carpeta	—
Interior/	—	Carpeta	—





Configuración de firebase

```
1 rules_version = '2';
2 service firebase.storage {
3     match /b/{bucket}/o {
4         match /{allPaths=**} {
5             allow read, write: if request.auth !=null;
6         }
7     }
8 }
```



https://flask-api-esp32.onrender.com/exterior

JSON Datos sin procesar Encabezados

Guardar Copiar Contraer todo Expandir todo Filtro JSON

```
0:
  created: "2024-07-08 23:58:51"
  name: "Exterior/image_0001.jpg"
  url: "https://storage.googleapis.com/esp32-fb6da.appspot.com/Exterior/image_0001.jpg?Expires=1721088398&GoogleAccessId.firebaseio-adminsdk-ko1ku%40esp32-fb6da.iam.gserviceaccount.com&Signature=k%2FasLcUSRbSDrvT%2BjedGix8m4qPexfYyfwehXs2DVAAg4oqZwL8qF2vvZVscNs1ueU%2FzafdzIc7eixLX%2FNPV7hRiqPUFeIbMYDA%2BmYCHThQiUKs6%2F13naFsHzqatzP950VmJVnhJTW2FTJRaWRRmd8X%2F6i72qvsvX8zVD%2FNk67YK66umHACmdkhIYb%2FHeBDYZZLE4xkC3erk%2FjiL3ph3nh3WKGfXoJsrmkhT%2Ft0xHXDeN15j5Z2C29cqalD4eUA40CVKnqr%2FnthzuP%2F8edEE7HD0pVbHiI2farq5bOMq4DJeK1%2BtBLz8A0zpaPkGh9C2kOn6TuNW2VgK3bEm0%3D%3D"

1:
  created: "2024-07-08 23:59:11"
  name: "Exterior/image_0002.jpg"
  url: "https://storage.googleapis.com/esp32-fb6da.appspot.com/Exterior/image_0002.jpg?Expires=1721088398&GoogleAccessId.firebaseio-adminsdk-ko1ku%40esp32-fb6da.iam.gserviceaccount.com&Signature=wIk0pSIJLT%2FE%2FneNo1Uaq55UEmIQTwLx9mZa37NRfm%2BL0tt2ptc7iGcRLicI0bVBLtzMFFEsGa3rqv8rLdLVwiDX0vIJccuK1lrSrqdRr0kb06I0f0qKFJqLUk2bGK03EIn8ZXPQaLt2rtHsMywElmiN4xRoXFVD2jiByfvLLVMWz2no8Z6nSDUdNy0c2iXALLmt%2Fy%2BAP0Yw5p05%2F0LOffi%2BTIXmgna82Zqwmf67jDaf1tofJeMqmuSFsxjwEm1AyehqbNKYjtIVR43BwVMygarzH19tUeZcdA3GbPcTHCF5uQyA5XUAxuZA8dNN4Tqaf3UB2cqohMoVod2Ea%3D%3D"
```

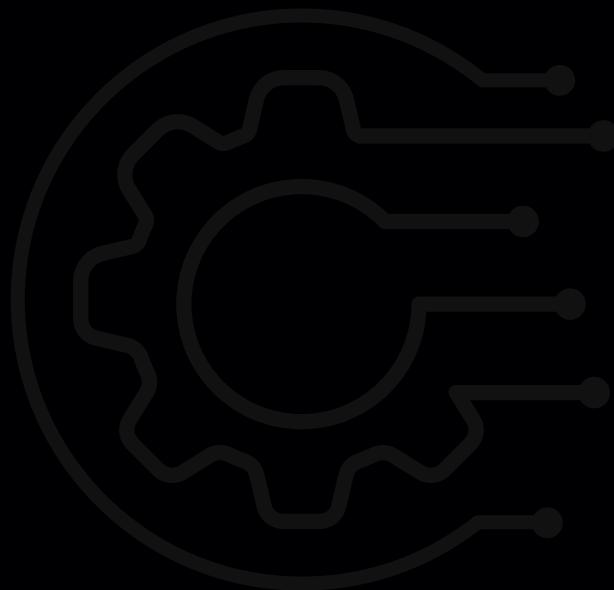
https://flask-api-esp32.onrender.com/interior

JSON Datos sin procesar Encabezados

Guardar Copiar Contraer todo Expandir todo Filtro JSON

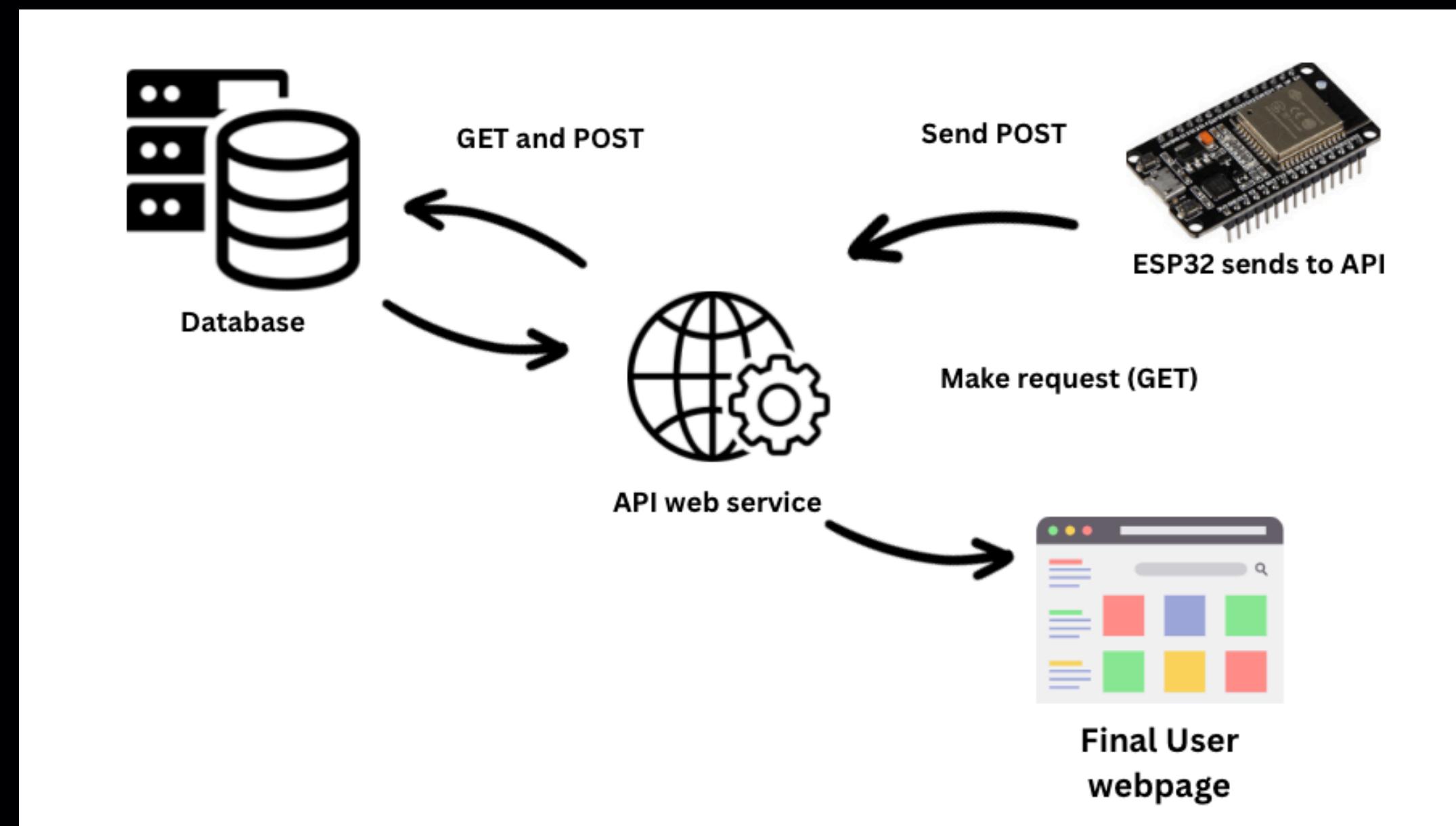
```
0:
  created: "2024-07-08 23:57:38"
  name: "Interior/image_0001.jpg"
  url: "https://storage.googleapis.com/esp32-fb6da.appspot.com/Interior/image_0001.jpg?Expires=17210884578&GoogleAccessId.firebaseio-adminsdk-ko1ku%40esp32-fb6da.iam.gserviceaccount.com&Signature=Pi83ozU0VrXfTB5eH0junI%2FRuhHrnNftZ17JcBj0FrKo%2B5gNWPYrQwMF3%2ByWZQ0erbMifpLSJttNrv3qIK41piax0J0Hcv2y2DwV2va5aaJAq0d7vkvzVaiPhmprPGzAntcoRH3rD2Miwh9aaDKLxNFxSNqqIDPdTNiicEEPNwfG02VMMmf9ZDMwr12hfxcCNHLG%2B0w81x9%2B8MF7%2B9zGwiwwva6s2exmaqkwntYVGMAfnXk7wuvv1Lm9LN7vnWIf5HLoewm6LZPmi7VqE5av8nAaw1G5nBEcTUz5DCoLPATQkPt2ji0b6CPiHvu2M%2B8HuayyxLeya5rrYz8z0r4LA%3D%3D"

1:
  created: "2024-07-08 23:57:55"
  name: "Interior/image_0002.jpg"
  url: "https://storage.googleapis.com/esp32-fb6da.appspot.com/Interior/image_0002.jpg?Expires=17210884578&GoogleAccessId.firebaseio-adminsdk-ko1ku%40esp32-fb6da.iam.gserviceaccount.com&Signature=ax6dyPqs9p4R0abtkobzbCdMxj4WpW7PrJfWYLpZN3D0fd56xn2OUCKZGaIn9V7bMk0mpv%2F6PFRZLp4%2FKsAMA8cLwPwpuLNzi66oW6KA6WH10Nj6sUTf4uUFLt0Ja505iiik3sy30k00pH%2FMcbBKLo9HmUiitDcafr7kEerUbqWbt0JXa2rFsc4JaLMJGsimcy3aRvwz%2BTdCIzKvr9q5Z0fy%2FNxaVLa0OML5Y2Cj5usB3zTD87fmfw5204i98iyQC%2BEsD0%2FETigkYY6SujpJwEpPW5EqMLX2nF4tEkFs7o0j%2Bq8A7B7vLEEorvyc2xZaqWqwgZ2%2F0el02tpBsPBw%3D%3D"
```



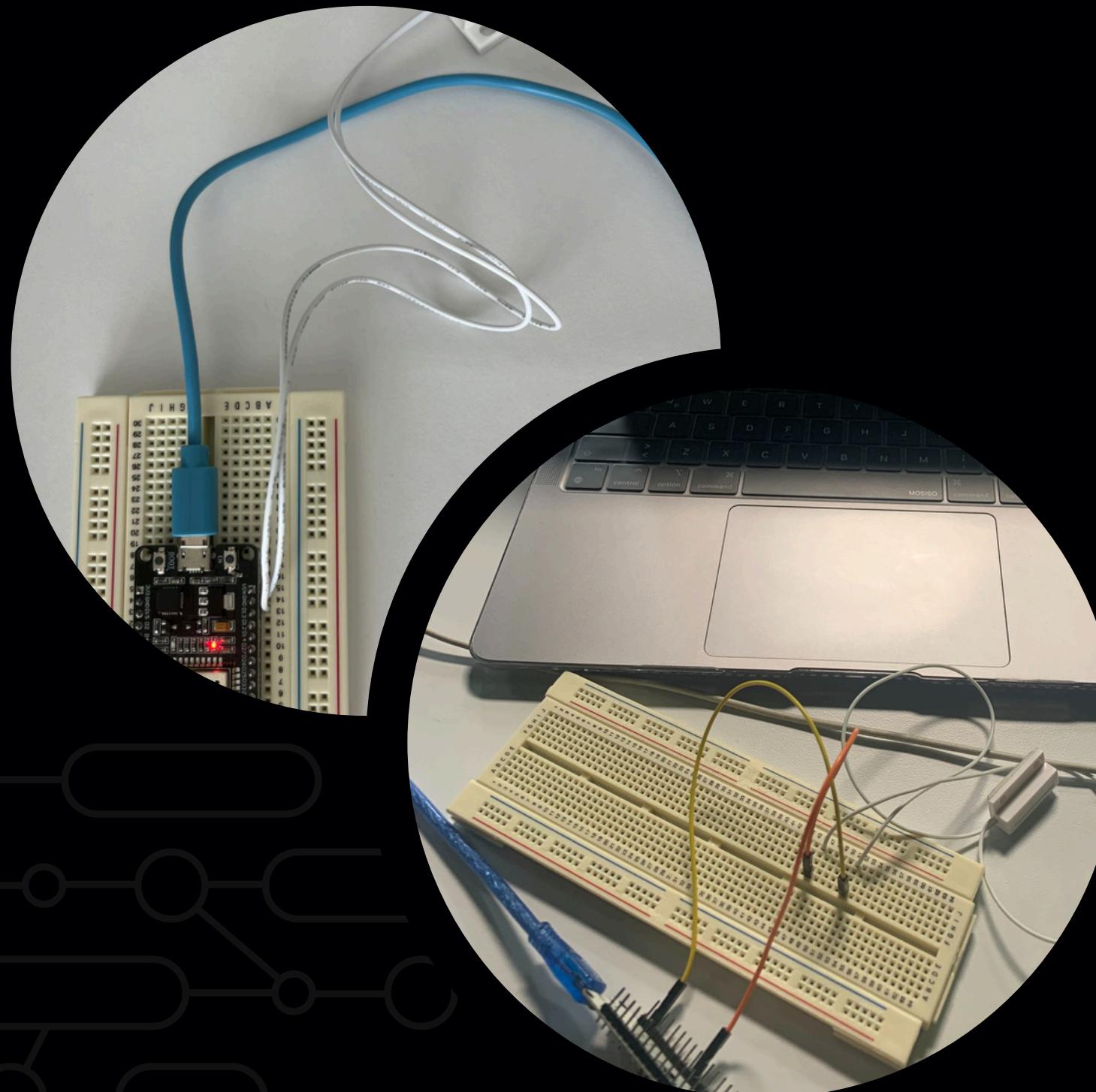
FUNCIONAMIENTO SENSORES

Este es un esquema de como funcionan los sensores:





SENsoRES DE APERTURA



Sensor de ventana: Antía Cores

Este dispositivo detecta cualquier intento de apertura de ventanas y envía una alerta inmediata del estado de la ventana al usuario a través del sistema de monitoreo en tiempo real.

Sensor de puerta: Sebastian De los Santos

Este dispositivo tiene como objetivo principal detectar si una puerta se encuentra abierta o cerrada, enviando así los datos a la interfaz creada.



SENsoRES DE APERTURA: VENTANA

```
#include <WiFi.h>
#include <HTTPClient.h>

const int WINDOW_SENSOR_PIN = 19;
int windowState;

const char* ssid = "iPhone de Diego";
const char* pass = "12345678";
void setup() {
  Serial.begin(9600);

  WiFi.begin(ssid, pass);
  Serial.print("Conectando a ");
  Serial.print(ssid);

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

Serial.println("");
Serial.println("WiFi conectado");
Serial.print("Dirección IP: ");
Serial.println(WiFi.localIP());

pinMode(WINDOW_SENSOR_PIN,
INPUT_PULLUP);

}

void loop() {
  windowState =
  digitalRead(WINDOW_SENSOR_PIN);
  String windowStatus = (windowState
  == HIGH) ? "Open" : "Closed";

  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    String url = "https://redes-
    practice.onrender.com/api/state/4";
    http.begin(url);
    http.addHeader("Content-Type",
    "application/json");

    String jsonData = "{\"windowstatus\":\""
    + windowStatus + "\"}";

    int httpResponseCode = http.PUT(jsonData);

    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println(response);
    } else {
      Serial.print("Error en la petición POST: ");
    }

    http.end();
  }

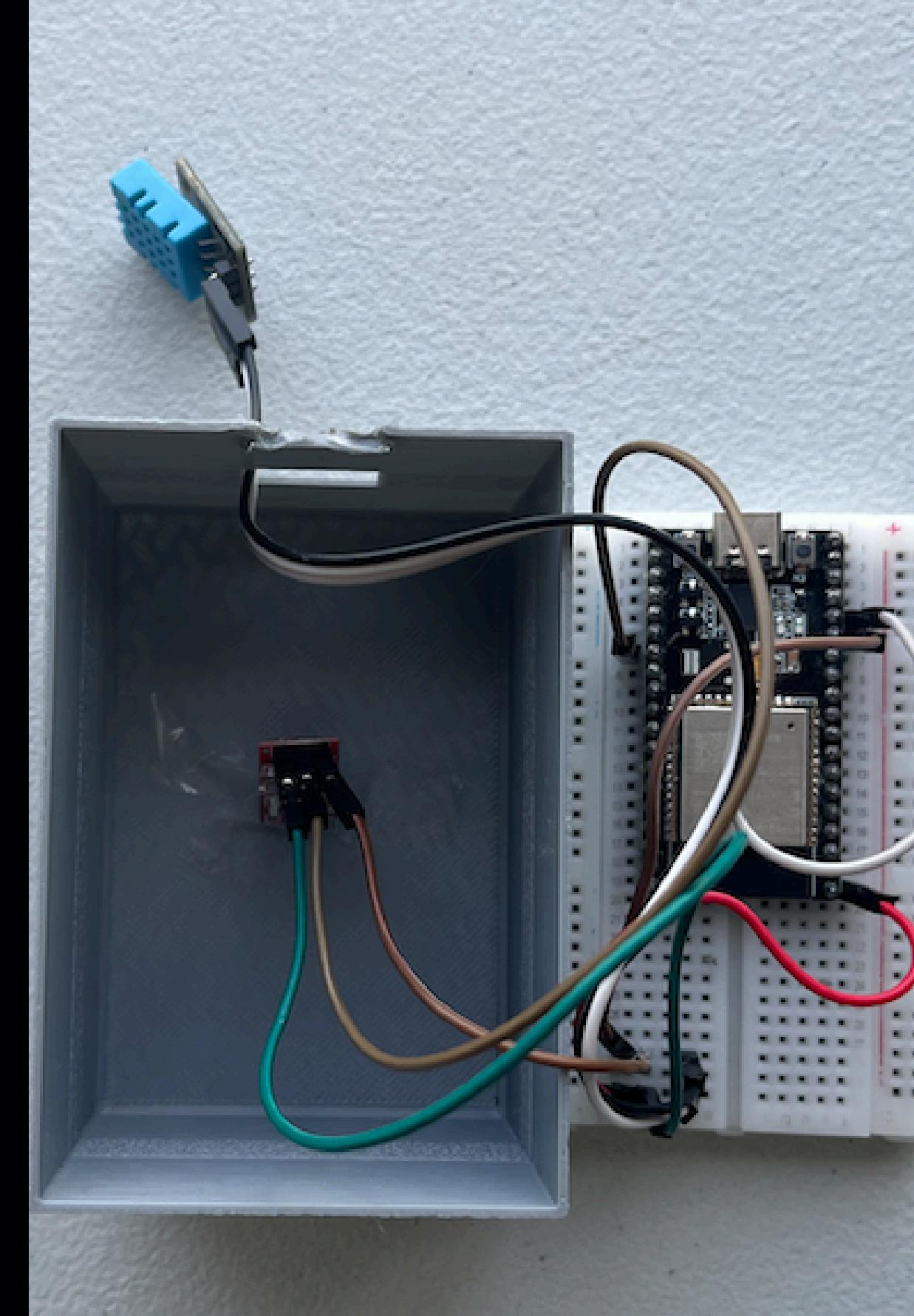
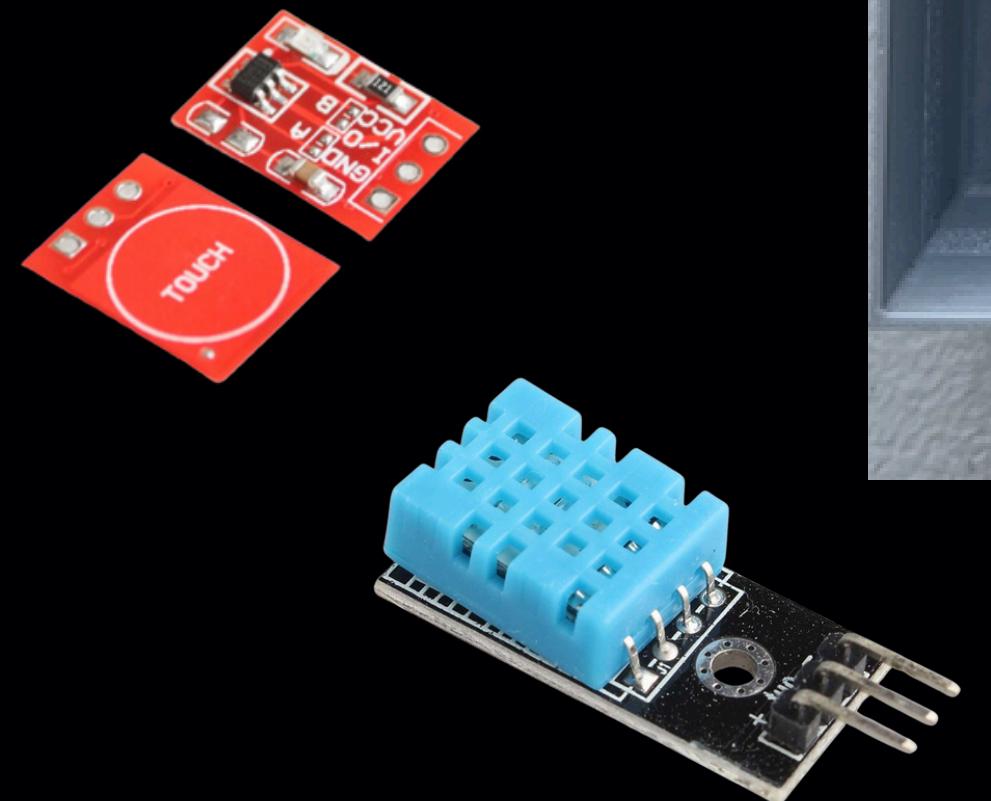
  delay(10000);
}
```



TIMBRE Y SENSOR DE TEMPERATURA

Funcionamiento

Conectando un sensor **ttp223** (**sensor touch**), permite detectar cuando alguien presiona el mismo, esto manda una señal a la API y 20 segundos después, esta regresa a False. De igual manera un **sensor de temperatura**, mismo que cada 20 segundos, envía la temperatura.

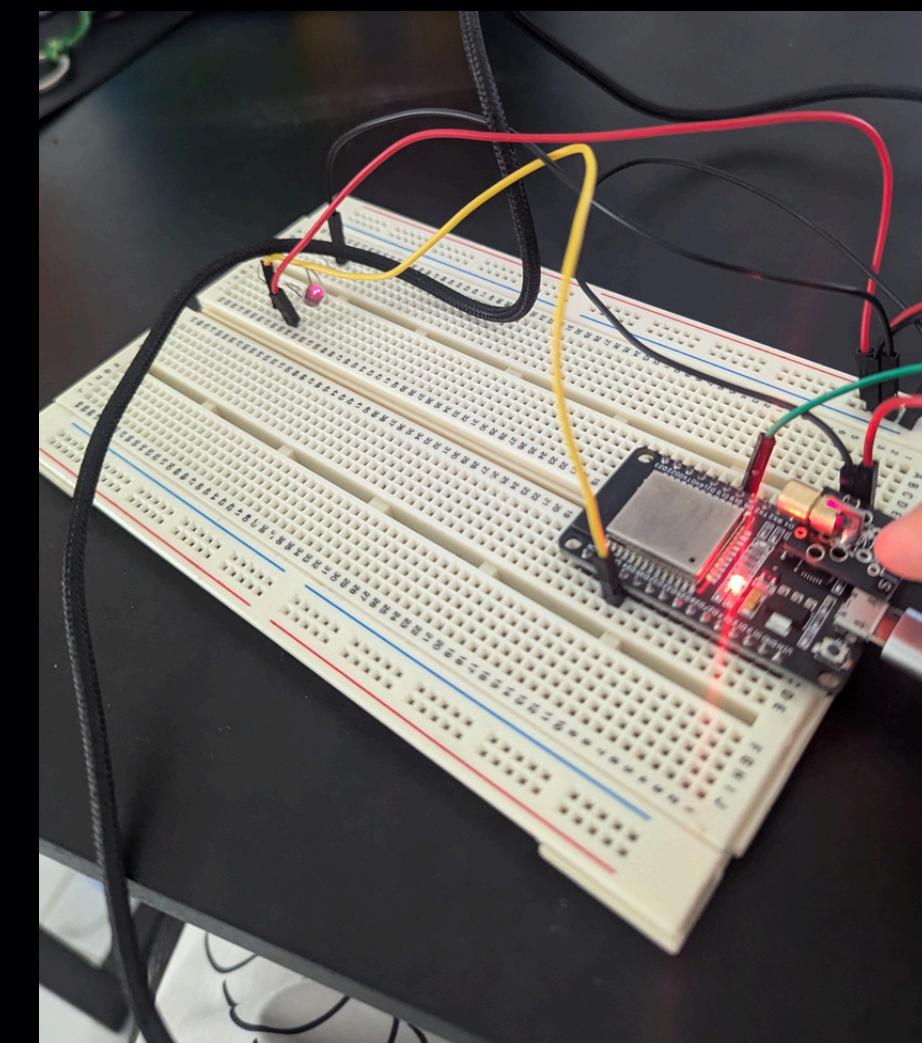
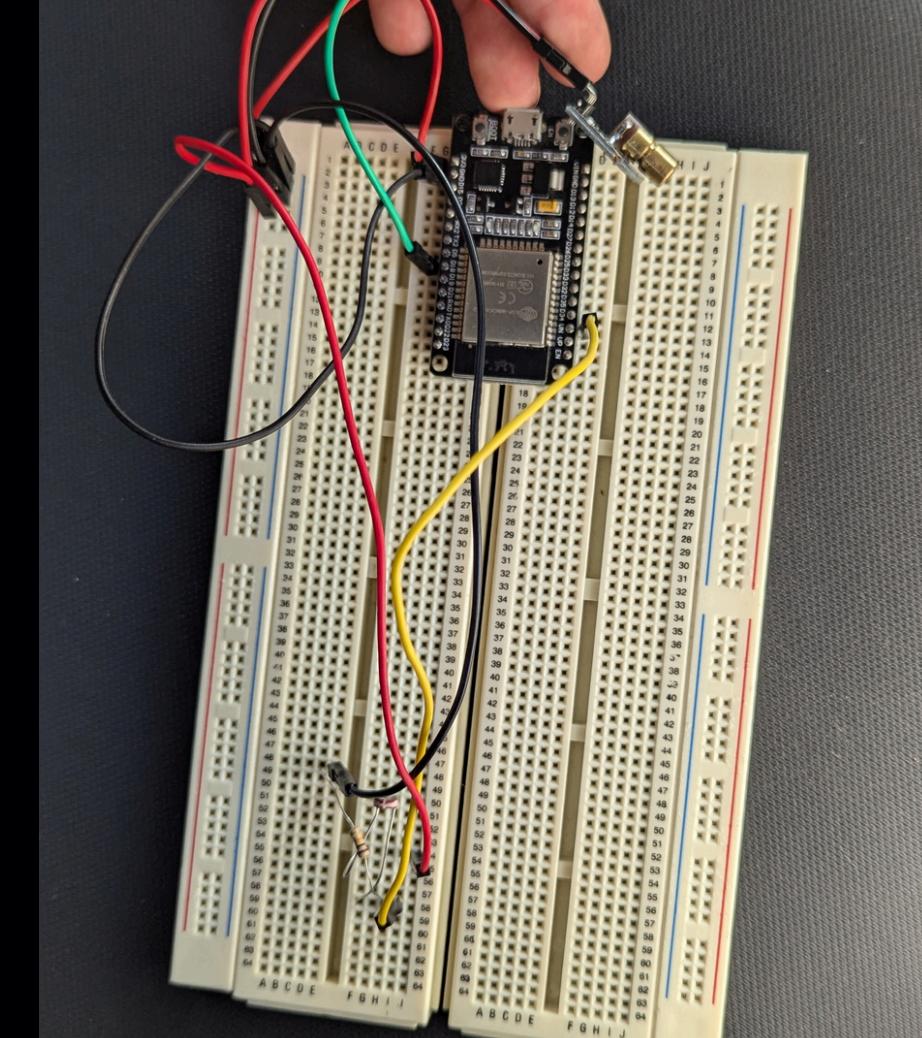




SENSOR DE ZONAS SEGURAS - LASER

Funcionamiento

Este sensor detecta cuando la línea del láser ha sido rota, mismo que permite visualizar si alguien ha pasado alguna "zona segura" donde el acceso no era permitido.

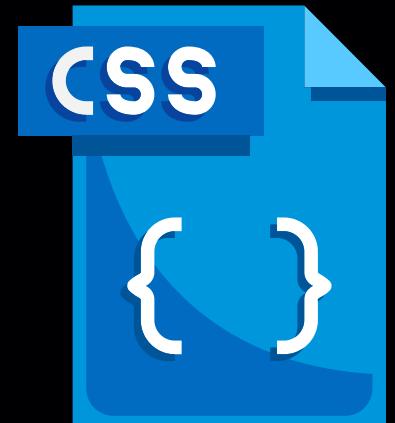
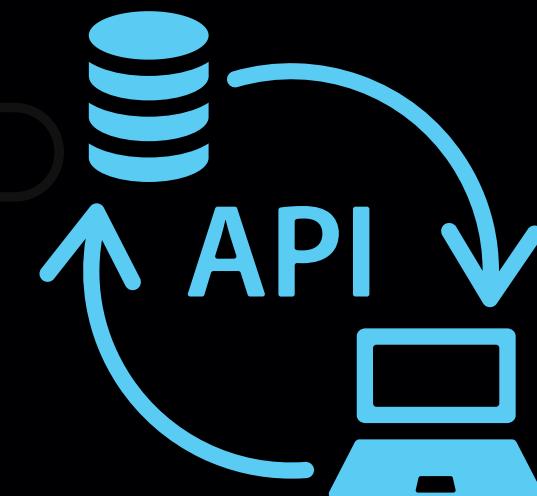
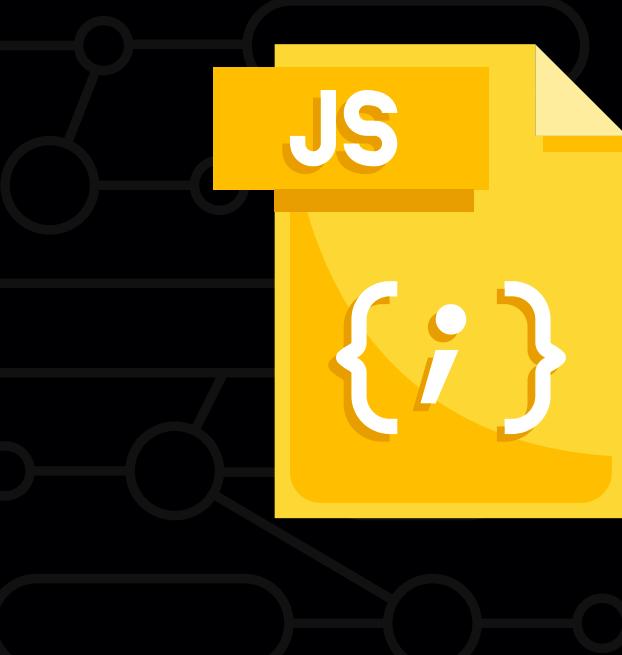




DASHBOARD – PAGINA WEB

En esta pagina web se encuentra la visualización de los datos

<https://e7237c45-87b0-414a-9b90-4d95c7a74854-00-28lgwb376agtx.kirk.replit.dev/>





REDES DIGITALES

GRACIAS

