

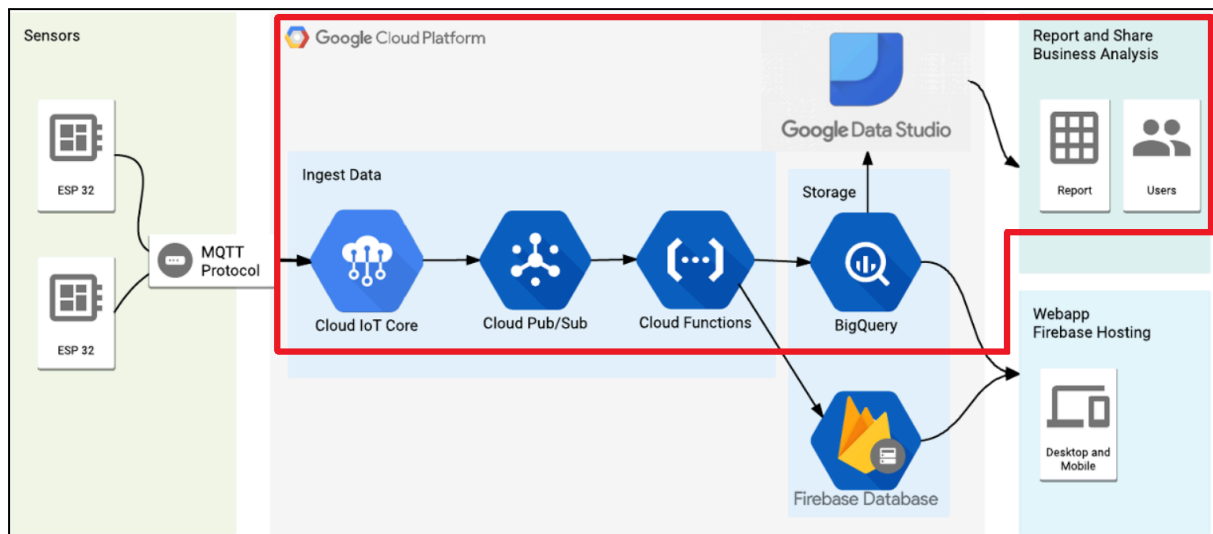
Aplicación IoT utilizando servicios Cloud

Clearblade
+
Google Cloud Platform
+
ESP-IDF 5.1

Autor: Leopoldo Zimperz
Fecha: 7/4/2025

Arquitectura de la aplicación IoT	1
Creación del proyecto en Google Cloud Platform	2
Crear proyecto en GCP	2
Crear base de datos	3
Habilitar Bigquery	3
Crear Dataset	3
Crear objetos en la base	4
Insertar datos descriptivos de sensores	8
Configurar servicio pub/sub en GCP	9
Crear Cloud Function	10
Habilitar servicio Google Cloud Functions	10
Crear la función para el proyecto	10
Crear función y definir activador	11
Cargar el código fuente e implementar la función	13
Seguridad entre dispositivos e IoT Core	17
Generar pares de claves (certificados)	17
Descargar certificado CA Minimo	18
Habilitar IoT hub Clearblade	19
Asociar Clearblade a GCP	19
Crear cuenta en Clearblade	19
Suscribirnos a Clearblade desde GCP	20
Crear proyecto en Clearblade	21
Crear cuenta de servicio en GCP	22
Crear ROL específico en GCP	23
Otorgar permisos con GCP IAM	28
Generar clave JSON para Clearblade desde GCP	28
Dar de alta dispositivos en Clearblade	31
Crear primer "Registry" en Clearblade	31
Dar de alta dispositivos IoT	32
Looker Studio - Visualización de datos básica	33
Creación de conexiones con Bigquery	33
Agregar gráficos al informe	36
Gauge	36
Candlestick	37
Gráfico de línea	38
Puesta en marcha de aplicación en ESP32	39

Arquitectura de la aplicación IoT

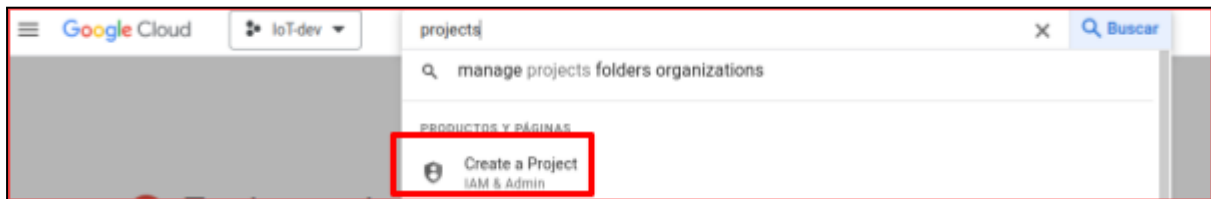


- Se utilizará el esp32 a fin de simular un sensor de temperatura e inyectar datos en el sistema.
- Los datos ingresarán a Clearblade utilizando MQTT sobre TLS.
- Vía Cloud Functions se almacenarán en Bigquery.
- Se visualizarán utilizando Looker Studio

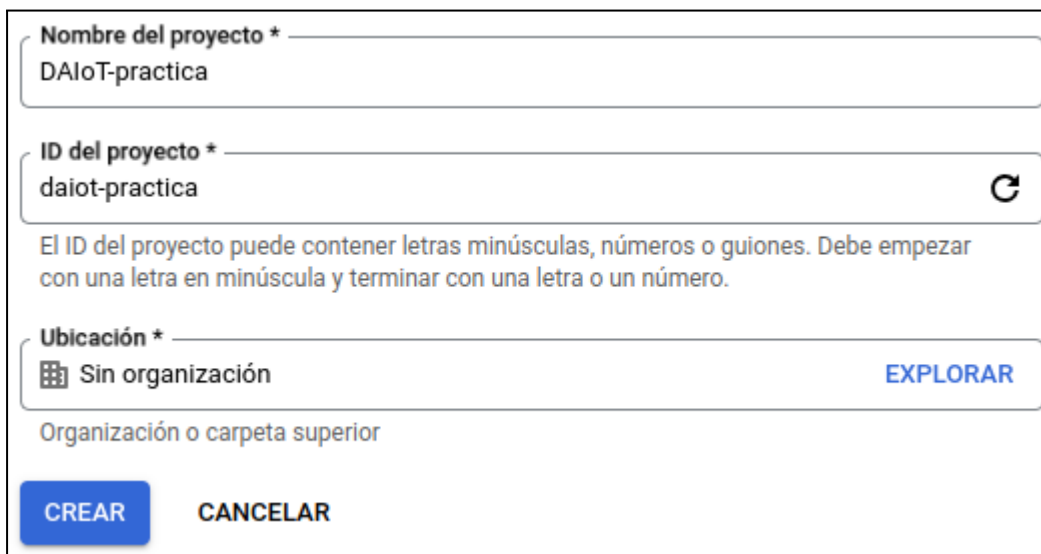
Creación del proyecto en Google Cloud Platform

Crear proyecto en GCP

En Google Cloud Console escribir projects en la barra de búsqueda y seleccionar la opción marcada:



Definir preferentemente un nombre de proyecto corto y claro, dado que luego se va a utilizar en consultas de base de datos y códigos de funciones. Resulta práctico que el ID coincida con el nombre del proyecto (esto no sucede con el primer proyecto por defecto que crea GCP, y el ID no es modificable).

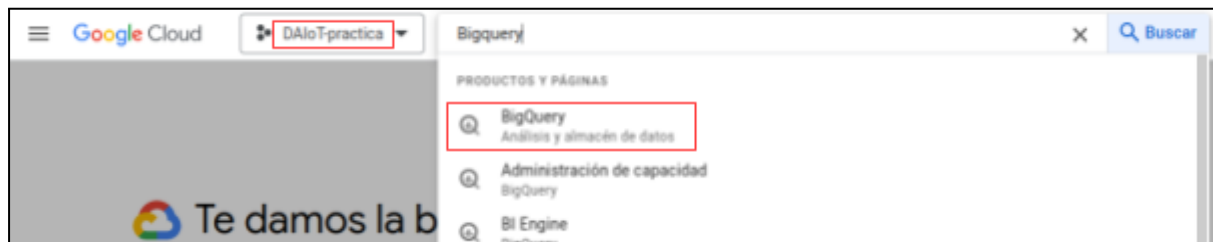
A screenshot of the 'Create Project' form in the Google Cloud Console. The form has three main input fields: 1. 'Nombre del proyecto *' (Project name) with the value 'DAIoT-practica'. 2. 'ID del proyecto *' (Project ID) with the value 'daiot-practica'. To the right of this field is a circular refresh icon. Below this field is a note: 'El ID del proyecto puede contener letras minúsculas, números o guiones. Debe empezar con una letra en minúscula y terminar con una letra o un número.' 3. 'Ubicación *' (Location) with a dropdown menu showing 'Sin organización' (No organization). To the right of this dropdown is a blue button labeled 'EXPLORAR'. Below the location field is the text 'Organización o carpeta superior'. At the bottom of the form are two buttons: a blue 'CREAR' (Create) button and a grey 'CANCELAR' (Cancel) button.

A medida que vayamos avanzando en los distintos pasos, Google solicitará una cuenta de pago e irá pidiendo que habilitemos distintos servicios en nuestra cuenta. El costo que generará el proyecto propuesto es irrelevante, en general no llegaría a U\$S 0,10 en un mes de funcionamiento continuo, pero Google de todos modos nos pide un medio de pago, incluso si tenemos disponible el crédito que nos otorga para realizar pruebas.

Crear base de datos

Habilitar Bigquery

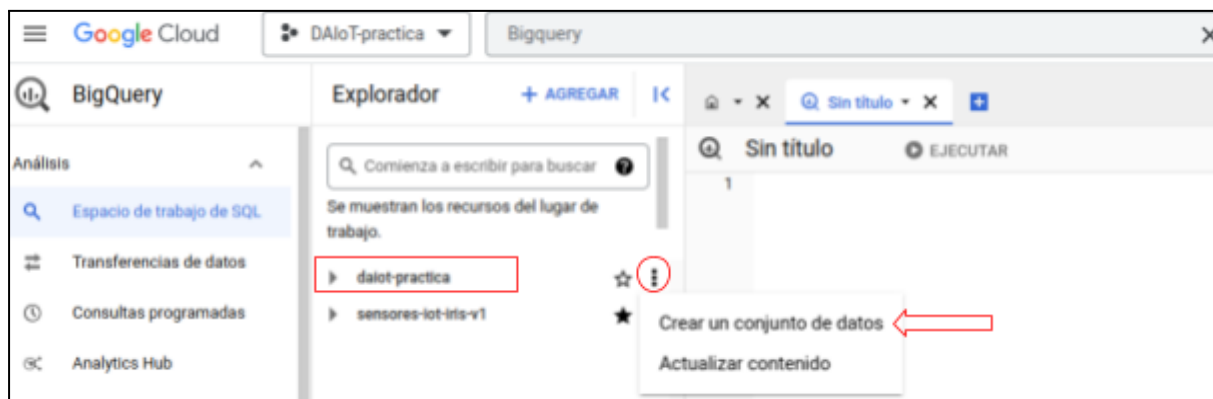
Habiendo seleccionado el proyecto que creamos recientemente en la barra de Google Cloud Console, buscar 'Bigquery'



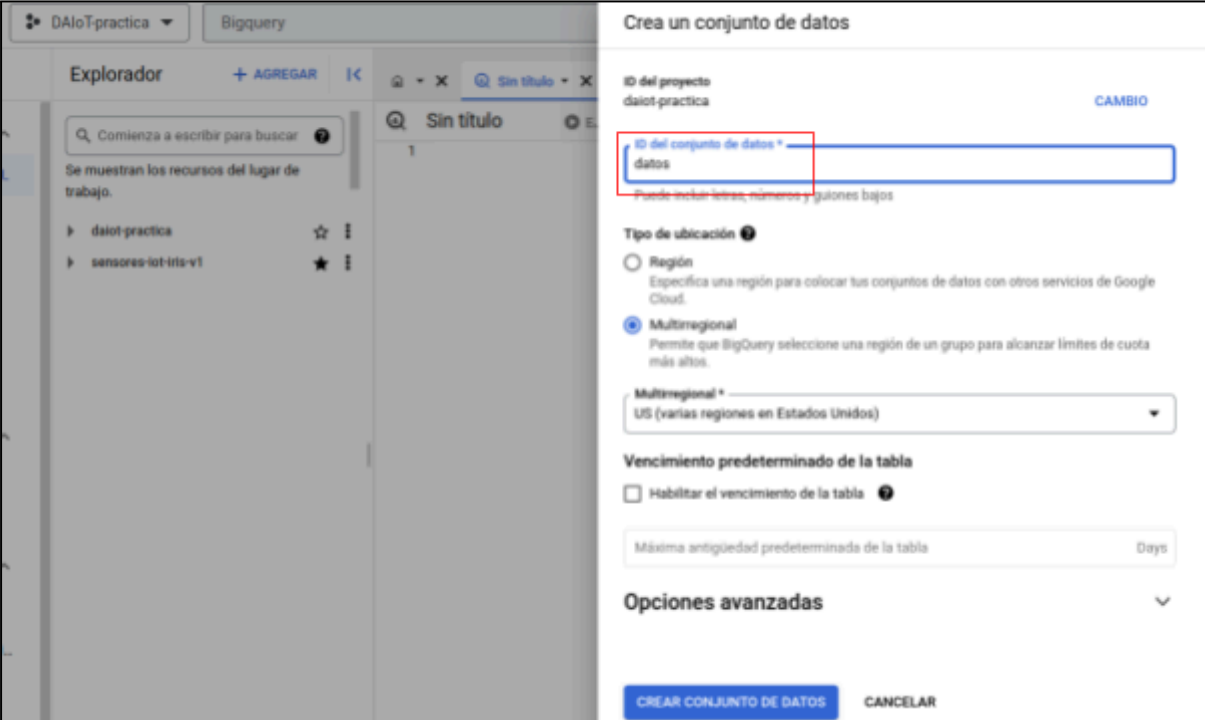
Crear Dataset

Crear el set de datos que contendrá a los objetos de base de datos que necesitamos para nuestra aplicación.

Dentro de la interfaz de Bigquery veremos el nombre del proyecto que hemos creado. A su derecha incluye un menú de opciones que permite crear el dataset.



En la ventana emergente luego de seleccionar la opción de creación de conjunto de datos, sólomente debemos definir el ID



Crear objetos en la base

En el editor SQL de la interfaz de Bigquery debemos ejecutar las instrucciones DDL detalladas más adelante. DDL es el “Lenguaje de Definición de Datos” de SQL.

Tener en cuenta que en cada instrucción se define el nombre del objeto a crear o aquel con el que deseamos trabajar, y se compone de tres partes: **nombre-proyecto.set-de-datos.nombre-objeto**. Tendremos que sustituir en cada instrucción el nombre de proyecto y el correspondiente al set de datos, para que concuerde con lo que hemos creado previamente.

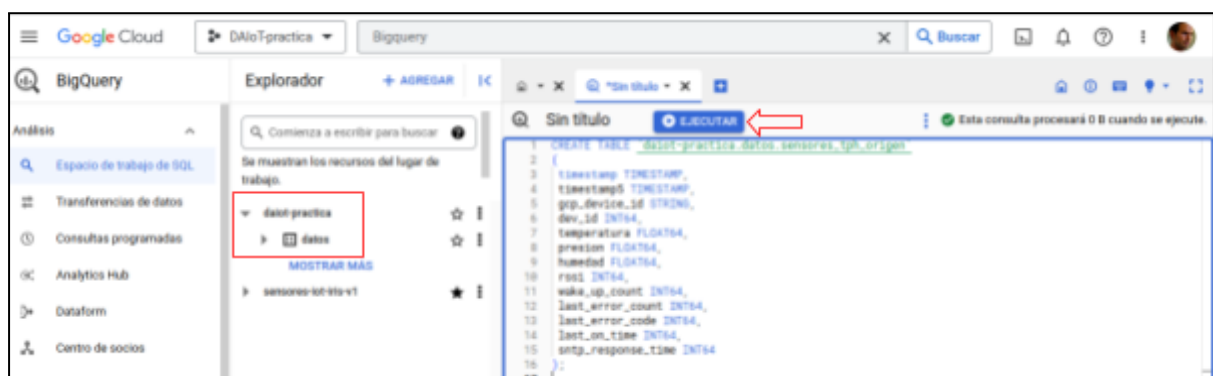
Con las instrucciones detalladas, crearemos dos tablas y dos vistas.

- Tabla principal que almacena la información relevada por los sensores.
- Tabla secundaria que incluirá información descriptiva sobre cada sensor.
- Vista que permite consultar los datos, organizados en intervalos de 5 minutos.
- Vista que permite consultar los últimos datos recibidos desde los sensores.

Instrucción para generar tabla a):

```
CREATE TABLE `daiot-practica.datos.sensores_tph_origen`  
(  
    timestamp TIMESTAMP,  
    timestamp5 TIMESTAMP,  
    gcp_device_id STRING,  
    dev_id INT64,  
    temperatura FLOAT64,  
    presion FLOAT64,  
    humedad FLOAT64,  
    rssi INT64,  
    wake_up_count INT64,  
    last_error_count INT64,  
    last_error_code INT64,  
    last_on_time INT64,  
    sntp_response_time INT64  
);
```

En este caso, se vería así:



Instrucción para generar tabla b):

```
CREATE TABLE `daiot-practica.datos.sensores_tph_desc_ubicacion`  
(  
    dev_id INT64,  
    ubicacion STRING,  
    propietario STRING,  
    descripcion STRING,  
    temp_offset FLOAT64  
);
```

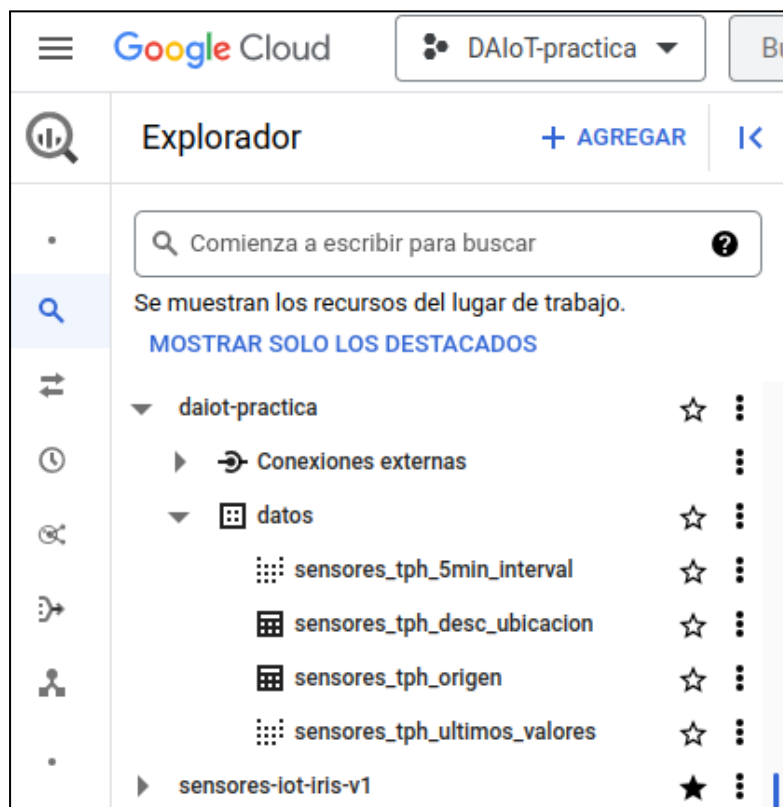
Instrucción para generar vista c):

```
CREATE VIEW `daiot-practica.datos.sensores_tph_5min_interval`  
AS  
SELECT  
    T1.dev_id, propietario,  
    timestamp5 tiempo_registro,  
    ROUND(AVG(temperatura),1) AS temperatura,  
    ROUND(AVG(humedad),1) AS humedad,  
    ROUND(AVG(presion),1) AS presion,  
    ROUND(AVG(rssi),0) AS rssi  
FROM `daiot-practica.datos.sensores_tph_origen` AS T1  
LEFT JOIN `daiot-practica.datos.sensores_tph_desc_ubicacion` AS  
T2  
ON T1.dev_id = T2.dev_id  
GROUP BY timestamp5, dev_id, propietario  
ORDER BY tiempo_registro DESC;
```


Instrucción para generar vista d):

```
CREATE VIEW `daiot-practica.datos.sensores_tph_ultimos_valores`  
AS  
  SELECT  
    T2.*  
  FROM (  
    SELECT dev_id, MAX(timestamp5) tiempo_registro  
    FROM `daiot-practica.datos.sensores_tph_origen`  
    GROUP BY dev_id ) AS T1  
  LEFT JOIN `  
    daiot-practica.datos.sensores_tph_5min_interval` AS T2  
  ON  
    T1.dev_id = T2.dev_id and  
    T1.tiempo_registro = T2.tiempo_registro;
```

Luego de ejecutar las cuatro consultas, los objetos se verán así en el explorador de Bigquery:



Insertar datos descriptivos de sensores

En la tabla secundaria debemos insertar un registro por cada sensor que deseemos conectar al sistema. Para ello podemos utilizar la siguiente instrucción DML (Lenguaje de modificación de datos):

```
INSERT INTO `daiot-practica.datos.sensores_tph_desc_ubicacion`  
    (dev_id, propietario, temp_offset)  
VALUES  
    (101, 'Leopoldo Zimperz', 0.0);
```

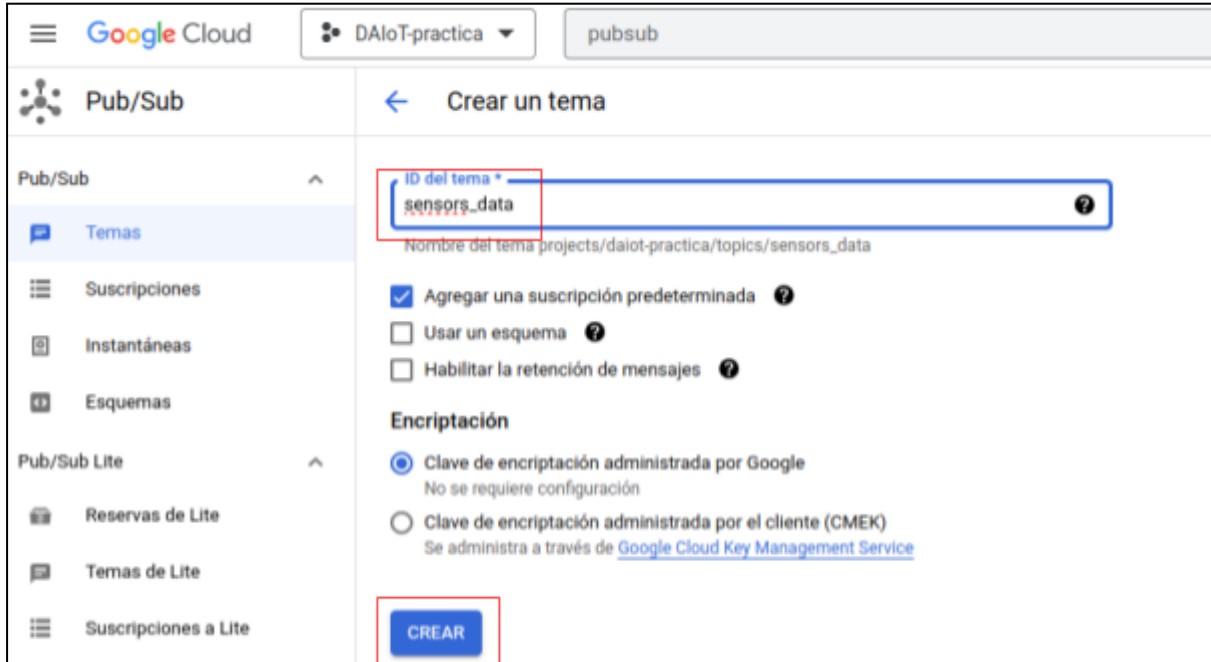
Configurar servicio pub/sub en GCP

Debemos habilitar el servicio pub/sub y crear uno o más topics MQTT a los que asociaremos Clearblade. Estos topics que creemos también nos servirán como trigger para disparar la función cloud a fin de que cargue los datos recibidos en la base de datos Bigquery.

Para habilitar pub/sub en GCP buscamos el servicio en la barra:



Una vez habilitado pub/sub podemos crear uno o más topics:

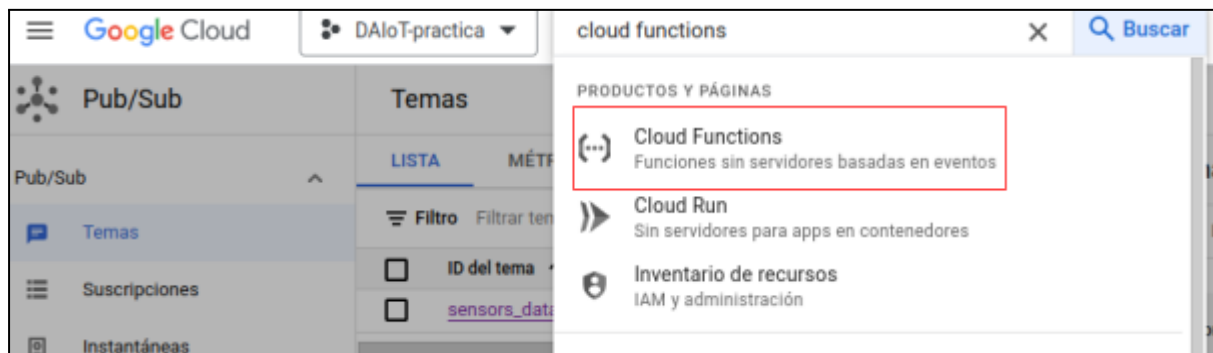


Crear Cloud Function

Ahora necesitamos crear una función (implementación serverless) que inserte en la tabla principal de la base de datos, la información proveniente de los sensores.

Habilitar servicio Google Cloud Functions

Buscar en la barra de GCP “Cloud Functions”, ingresar al servicio y habilitarlo si nunca se utilizó:



Crear la función para el proyecto

La primera vez que hagamos clic en ‘Crear función’ GCP nos pedirá que aceptemos habilitar el servicio y luego podremos continuar.

Crear función y definir activador

Para lograr implementar la función necesitamos realizar varios pasos, en principio asignarle un nombre y agregar un “activador” relacionado con el servicio pub/sub:

Google Cloud DAIoT-practica functions

Cloud Functions ← Crear función

1 Configuración — 2 Código

Aspectos básicos

Entorno
2ª gen.

Nombre de la función *
MQTT_to_Bigquery

Región *
us-central1 (Iowa)

Activador

🔒 HTTPS ?

Autenticación ?

☐ Permitir invocaciones sin autenticar
Marca esta opción cuando crees una API pública o un sitio web.

☒ Activador de Pub/Sub con Cloud IAM.

☐ Activador de Cloud Storage

☐ Activador de Firestore ifunctions.net/MQTT_to_Bigquery

☐ Otro activador

+ AGREGAR ACTIVADOR

Al agregar un activador pub/sub, debemos seleccionar el tipo de eventos que deseamos que dispare a la función y el topic que hayamos creado anteriormente:

Activador de Eventarc

Tipo de activador
Fuentes de Google

Proveedor del evento *
Cloud Pub/Sub

Evento *
google.cloud.pubsub.topic.v1.messagePublished

MOSTRAR DETALLES

Selecciona un tema de Cloud Pub/Sub *
projects/daiot-practica/topics/sensors_data

Región *
us-central1 (Iowa)

Cloud Pub/Sub necesita que la cuenta de servicio `service-269537163027@gcp-sa-pubsub.iam.gserviceaccount.com` tenga la función `roles/iam.serviceAccountTokenCreator` en este proyecto a fin de crear tokens de identidad. Puedes cambiar esto más adelante.

OTORGAR

MÁS INFORMACIÓN

Cuenta de servicio
App Engine default service account

Identidad que usará el activador de Eventarc creado.

GUARDAR ACTIVADOR

CANCELAR

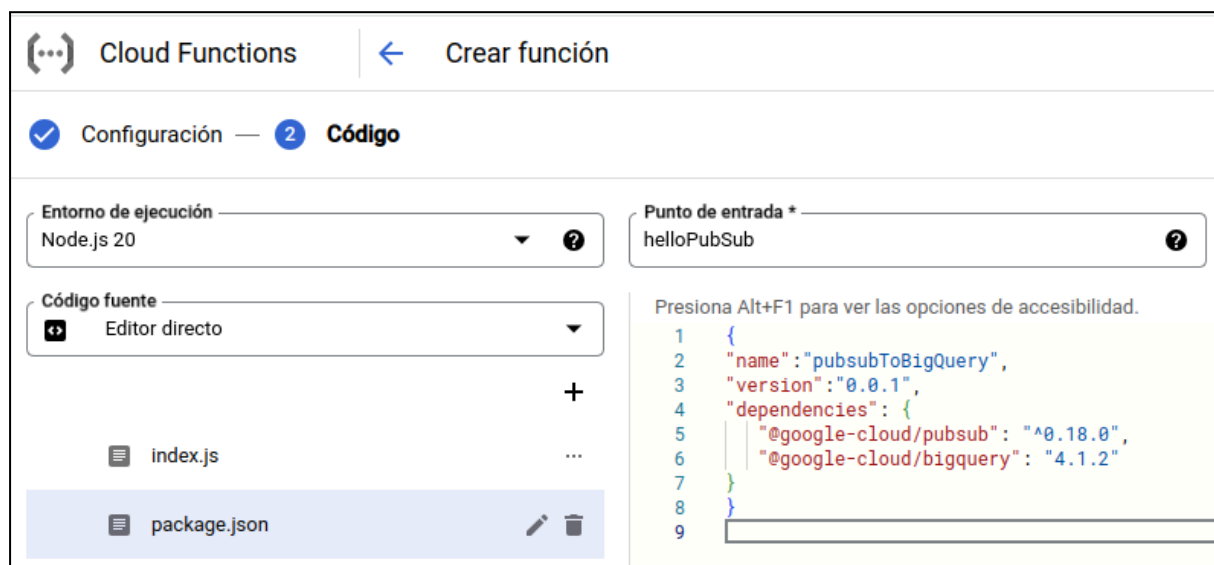
Cargar el código fuente e implementar la función

En este ejemplo utilizamos Noje.js y debemos cargar el archivo de dependencias y el código fuente para darle vida a la Cloud Function:

En el archivo de dependencias, necesitamos cargar el siguiente código:

```
{  
  "name": "pubsubToBigQuery",  
  "version": "0.0.1",  
  "dependencies": {  
    "@google-cloud/pubsub": "^0.18.0",  
    "@google-cloud/bigquery": "4.1.2"  
  }  
}
```

En el editor se verá así:



En el archivo de código fuente, necesitamos cargar el siguiente código y modificar el “Punto de entrada” para que se pueda ejecutar:

```
exports.pubsubToBQ = (event, callback) => {
  const pubsubMessage = event.data;
  const pubsubGCPdeviceId = event.attributes.deviceId;
  const { BigQuery } = require("@google-cloud/bigquery");
  const bigquery = new BigQuery();

  console.log(event.data.toString());

  var datos_sensor = JSON.parse(
    Buffer.from(pubsubMessage, "base64").toString()
  );
  var datos_insert = JSON.parse("{}");

  var timestamp_registro = new Date();
  var precisionTimestamp5min = 5 * 60 * 1000;
  var timestamp5 =
    Math.round(timestamp_registro / precisionTimestamp5min) *
    precisionTimestamp5min;

  // Generamos el timestamp que se cargará en la base, con tiempo
  exacto de ocurrencia
  timestamp_registro.setHours(timestamp_registro.getHours() - 3);

  // Generamos otro timestamp, redondeado y ubicado precisamente en
  intervalos de 5 minutos
  var timestamp5date = new Date(timestamp5);
  timestamp5date.setHours(timestamp5date.getHours() - 3);

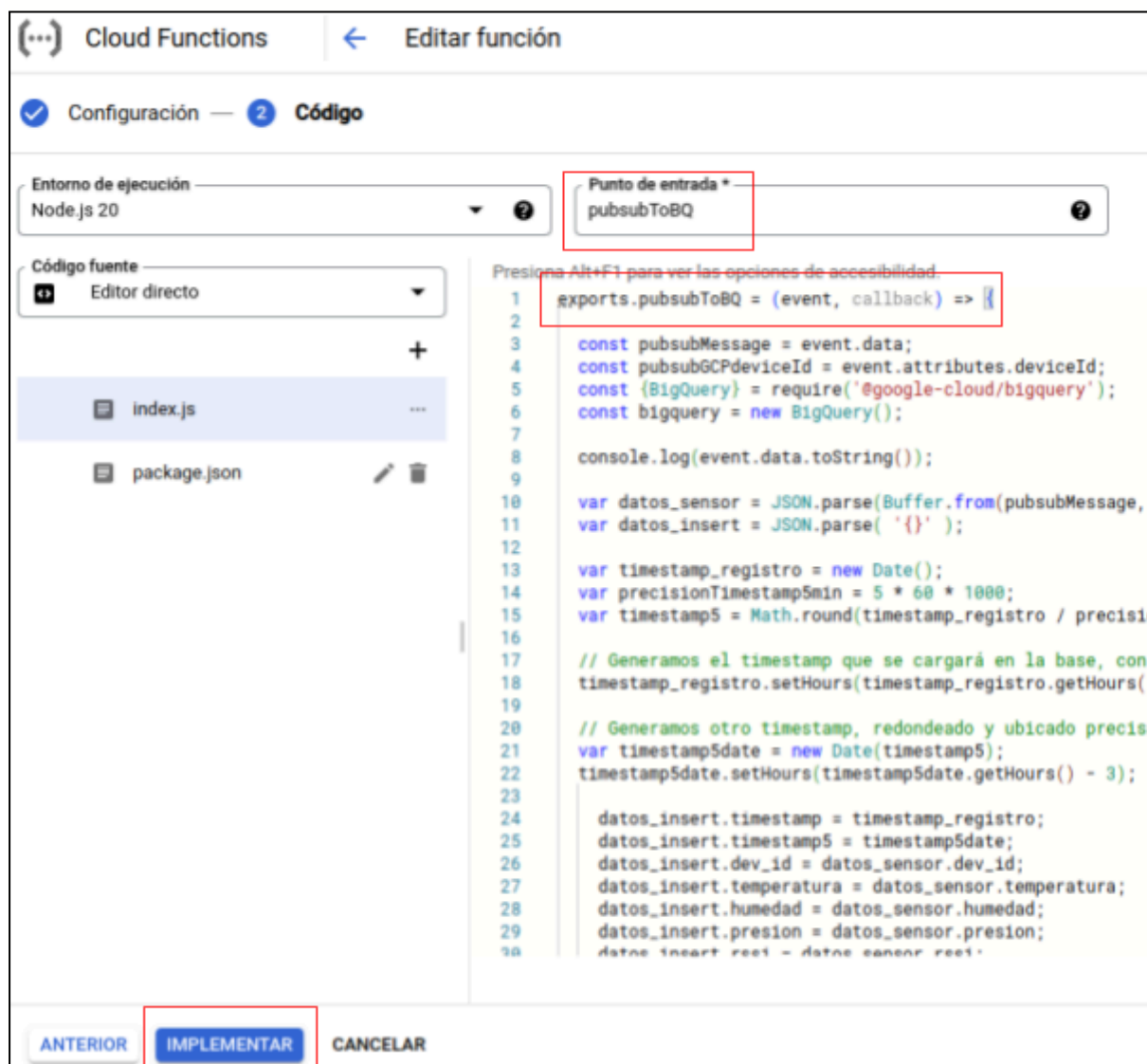
  datos_insert.timestamp = timestamp_registro;
  datos_insert.timestamp5 = timestamp5date;
  datos_insert.dev_id = datos_sensor.dev_id;
  datos_insert.temperatura = datos_sensor.temperatura;
  datos_insert.humedad = datos_sensor.humedad;
  datos_insert.presion = datos_sensor.presion;
  datos_insert.rssi = datos_sensor.rssi;
  datos_insert.gcp_device_id = pubsubGCPdeviceId;
  datos_insert.wake_up_count = datos_sensor.wake_up_count;
  datos_insert.last_error_count = datos_sensor.last_error_count;
  datos_insert.last_error_code = datos_sensor.last_error_code;
```



```
datos_insert.temperatura = datos_sensor.temperatura;
datos_insert.humedad = datos_sensor.humedad;
datos_insert.presion = datos_sensor.presion;
datos_insert.rssi = datos_sensor.rssi;
datos_insert.gcp_device_id = pubsubGCPdeviceId;
datos_insert.wake_up_count = datos_sensor.wake_up_count;
datos_insert.last_error_count = datos_sensor.last_error_count;
datos_insert.last_error_code = datos_sensor.last_error_code;
datos_insert.last_on_time = datos_sensor.last_on_time;
datos_insert.snmp_response_time = datos_sensor.snmp_response_time;

bigquery
  .dataset("datos")
  .table("sensores_tph_origen")
  .insert(datos_insert, { ignoreUnknownValues: true, raw: false })
  .catch((err) => {
    if (err && err.name === "PartialFailureError") {
      if (err.errors && err.errors.length > 0) {
        console.log("Insert errors:");
        err.errors.forEach((err) => console.error(err));
      }
    } else {
      console.error("ERROR BigQuery:", err);
    }
  });
};
```

Se vería algo así y estaría listo para implementar:



El proceso de implementación puede durar algunos minutos y si se implementa de manera correcta, lo veremos de este modo en la consola GCP:

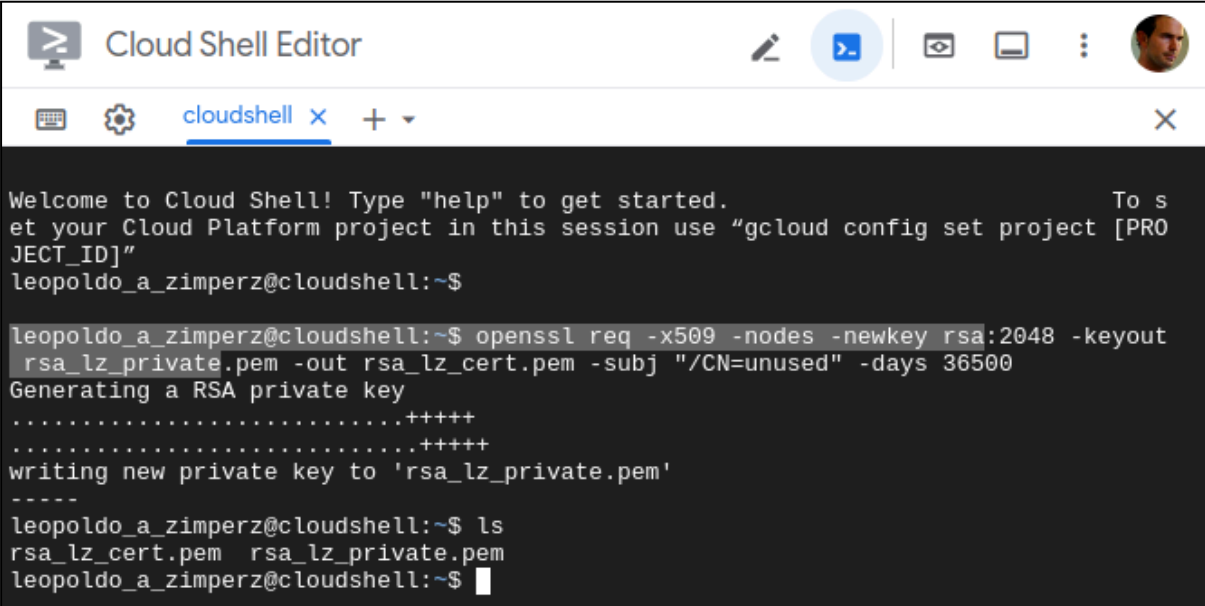
Cloud Functions		Funciones		+ CREAR FUNCIÓN	ACTUALIZAR
Filtro	Filtrar funciones				
<input type="checkbox"/>	Entorno	Nombre ↑	Última implementación	Región	
<input checked="" type="checkbox"/>	2nd gen	MQTT_to_Bigquery	20 sept 2023 16:53:29	us-central1	

Seguridad entre dispositivos e IoT Core

Generar pares de claves (certificados)

Ingresa al SHELL de GCP y ejecutar la siguiente instrucción utilizando la herramienta OPENSSL:

```
openssl req -x509 -nodes -newkey rsa:2048 -keyout device.key -out rsa_lz_cert.pem  
-subj "/CN=unused" -days 36500
```



The screenshot shows the Cloud Shell Editor interface. At the top, there's a header with the 'Cloud Shell Editor' title and various icons. Below the header, a terminal window is open with the following text:

```
Welcome to Cloud Shell! Type "help" to get started. To s  
et your Cloud Platform project in this session use "gcloud config set project [PRO  
JECT_ID]"  
leopoldo_a_zimperz@cloudshell:~$  
  
leopoldo_a_zimperz@cloudshell:~$ openssl req -x509 -nodes -newkey rsa:2048 -keyout  
rsa_lz_private.pem -out rsa_lz_cert.pem -subj "/CN=unused" -days 36500  
Generating a RSA private key  
.....+++++  
.....+++++  
writing new private key to 'rsa_lz_private.pem'  
-----  
leopoldo_a_zimperz@cloudshell:~$ ls  
rsa_lz_cert.pem  rsa_lz_private.pem  
leopoldo_a_zimperz@cloudshell:~$
```

Desde el menú de opciones del Shell de GCP podremos descargar fácilmente los archivos.

La clave con nombre “device.key” se copiará dentro del proyecto del esp32, en la carpeta /components/clearblade_connector.

La clave con nombre “rsa_lz_cert.pem” se incluirá al dar de alta cada dispositivo en Clearblade, copiando manualmente el contenido en la sección de autenticación y seleccionando certificados tipo RS256_X509.

[Clearblade - Managing credenciales - Generating key pairs](#)

Descargar certificado CA Minimo

En la propia ayuda de Clearblade indica desde donde descargar el certificado CA Mínimo de largo término (LTS) con vigencia hasta 2031 y 1,3 KB de peso.

The screenshot shows the ClearBlade IoT Core documentation page. On the left is a sidebar with a menu for 'ClearBlade IoT Core' containing items like 'Migration project plan templates', 'Creating registries and devices', 'Add developers to a project', 'Add service accounts to a project', 'Retargeting devices', 'Using gateways', 'Publishing over MQTT' (highlighted), 'Publishing over HTTP', 'Configuring devices and getting st...', 'Sending commands to devices', and 'Managing credentials'. The main content area has a red-bordered box titled 'Downloading MQTT server certificates' with the text: 'ClearBlade uses DigiCert-based certificates (long-term support, 2031, and 1.3K in size). The minimal root ca file can be found [here](#).' Below this is a section titled 'Configuring MQTT clients' with the text: 'MQTT clients authenticate devices by connecting to the MQTT bridge. To configure an MQTT client to authenticate a device:'. It then lists five steps: 1. Set the MQTT client ID to the full device path: (with a code block showing '1 projects/PROJECT_ID/locations/REGION/registries/REGISTRY_ID/devices/DEVICE_ID'), 2. Associate the MQTT client with MQTT server certificates, 3. Choose the appropriate MQTT host name, 4. Specify a username. The MQTT bridge ignores the username field, but some MQTT client libraries will not send the password field unless the username field is specified. For best results, supply an arbitrary username like `unused` or `ignored`, 5. Set the password. The password field must contain the JWT.

[Clearblade lot-Core - Publishing over MQTT](#)

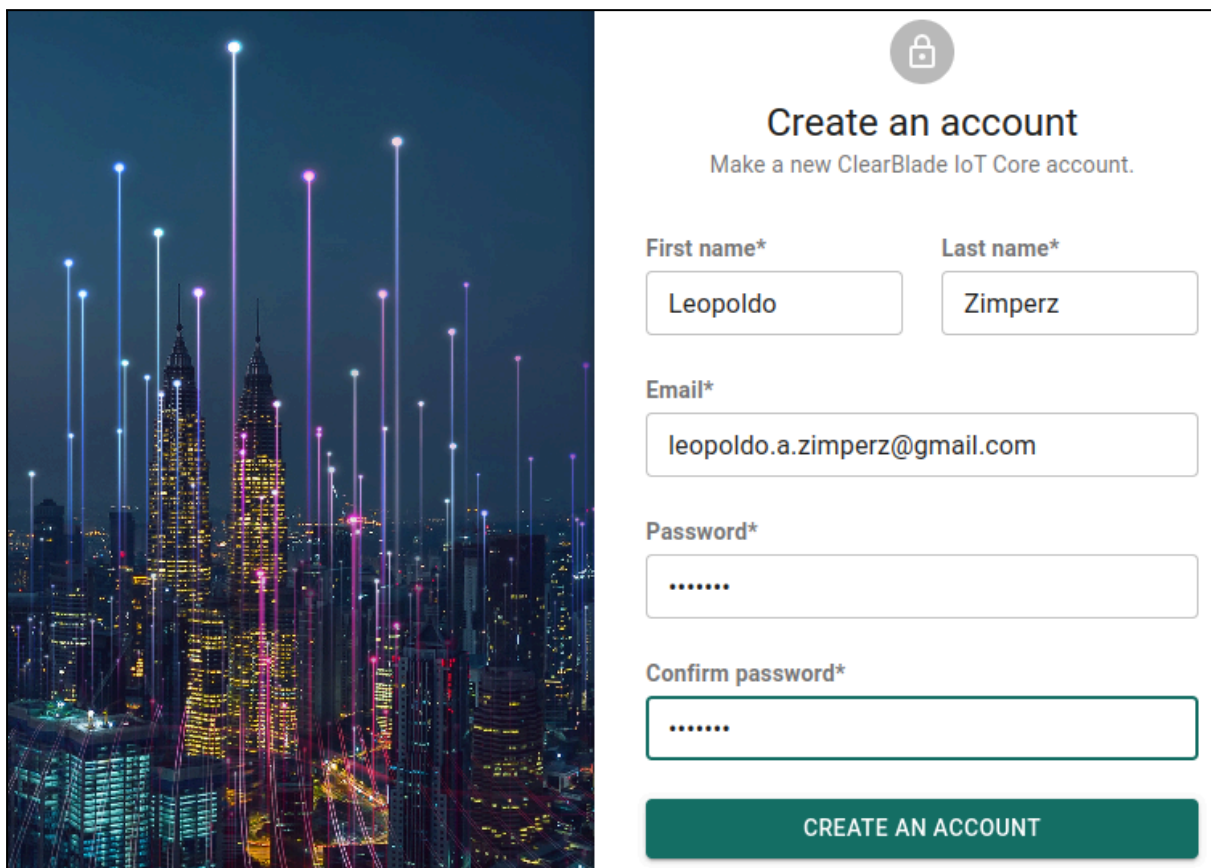
Habilitar IoT hub Clearblade


Asociar Clearblade a GCP

Crear cuenta en Clearblade

Registrarnos en Clearblade, preferentemente utilizando el correo electrónico de la cuenta de Google en la cual creamos el proyecto en GCP:

<https://iot.clearblade.com/iot-core/auth/register>





Create an account

Make a new ClearBlade IoT Core account.

First name*

Leopoldo

Last name*

Zimperz

Email*

leopoldo.a.zimperz@gmail.com

Password*

.....

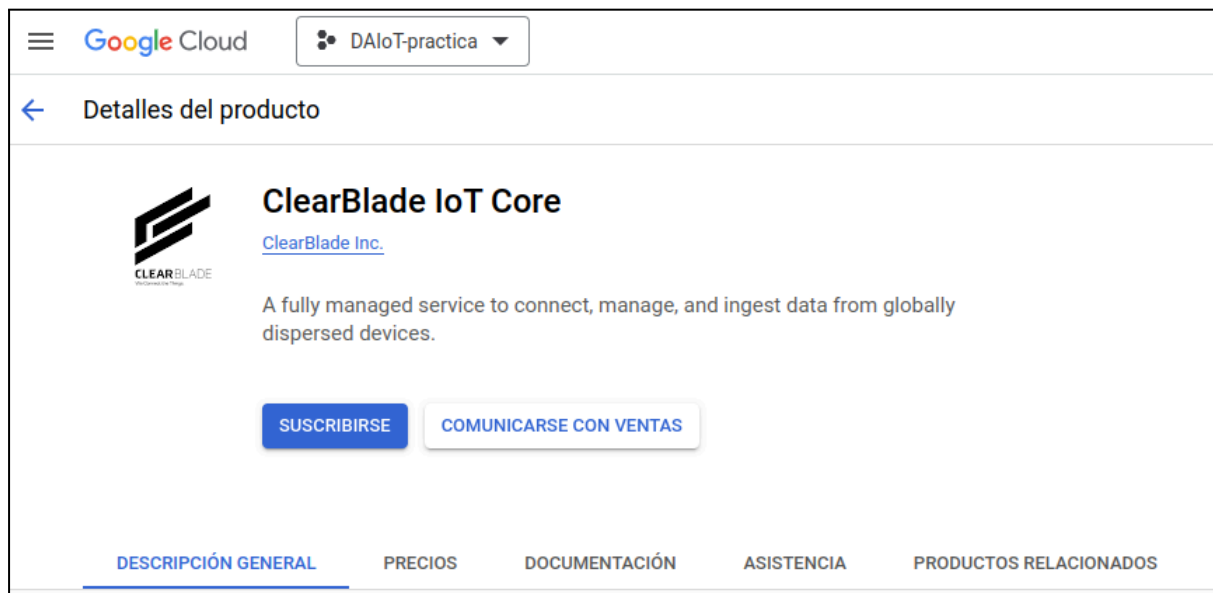
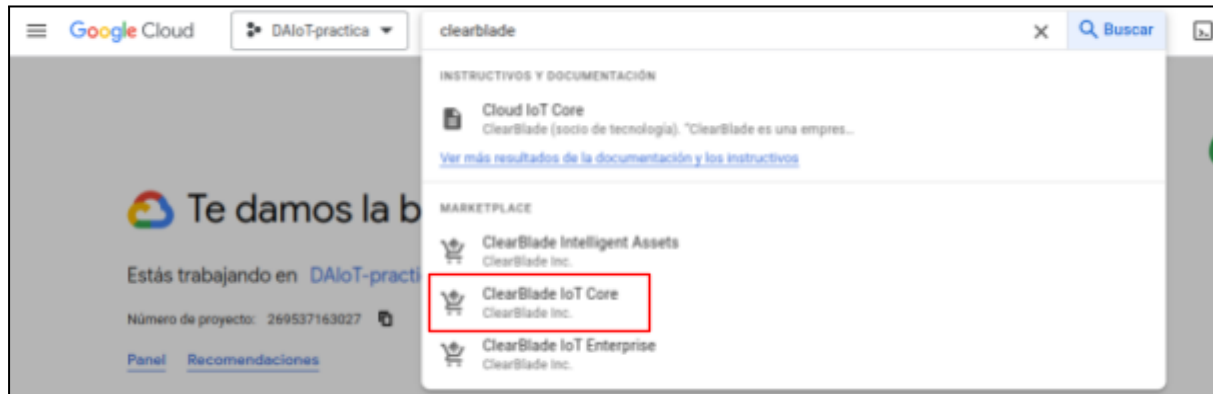
Confirm password*

.....

CREATE AN ACCOUNT

Suscribimos a Clearblade desde GCP

Para habilitar Clearblade debemos buscarlo en el Marketplace de GCP

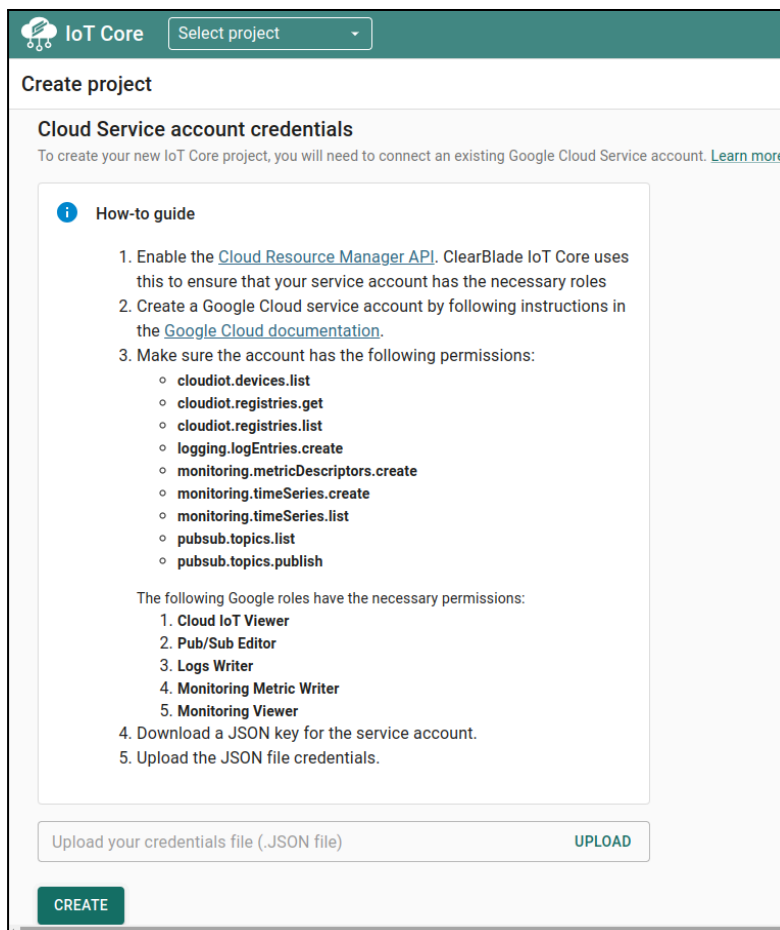


Una vez que nos suscribimos, tendremos que esperar la aprobación por parte de Clearblade, en general demoran algunas horas, pero también se les puede solicitar al soporte, indicándoles el número de orden que podemos obtener desde el Marketplace de Google:



Crear proyecto en Clearblade

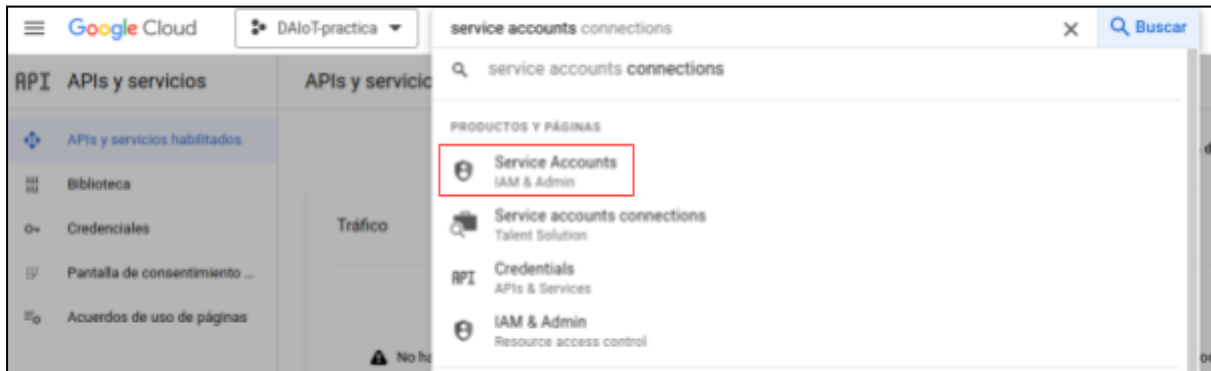
Una vez que tenemos una cuenta funcional en Clearblade debemos generar nuestro proyecto y asociarlo a Google. Clearblade nos solicitará que habilitemos ciertos permisos y roles desde GCP y que luego carguemos el archivo con las correspondientes credenciales:



Crear cuenta de servicio en GCP

En base al detalle que nos brinda Clearblade [aquí](#), debemos crear una cuenta de servicios en GCP, con los permisos necesarios.

Ingresar en la sección de GCP para crear cuenta de servicio:



Definir un nombre para la nueva cuenta de servicio:

←

Crear cuenta de servicio

1

Detalles de la cuenta de servicio

Nombre de la cuenta de servicio

Cuenta-servicio-Clearblade

Mostrar nombre de esta cuenta de servicio

ID de la cuenta de servicio *

cuenta-servicio-clearblade

X ↺

Dirección de correo electrónico: cuenta-servicio-clearblade@daiot-

practica.iam.gserviceaccount.com

Descripción de la cuenta de servicio

Describe lo que hará esta cuenta de servicio

CREAR Y CONTINUAR

2

Otorga a esta cuenta de servicio acceso al proyecto

(opcional)

3

Otorga a usuarios acceso a esta cuenta de servicio

(opcional)

LISTO

CANCELAR

Crear ROL específico en GCP

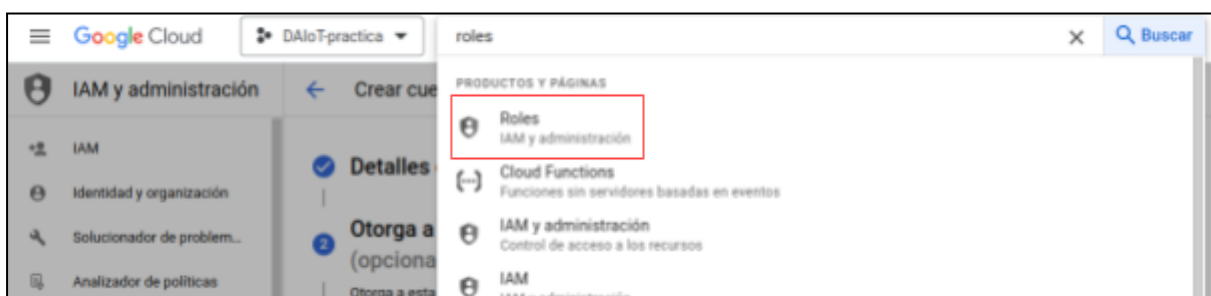
Crear el rol específico y necesario con los permisos detallados en el punto 3 por Clearblade:

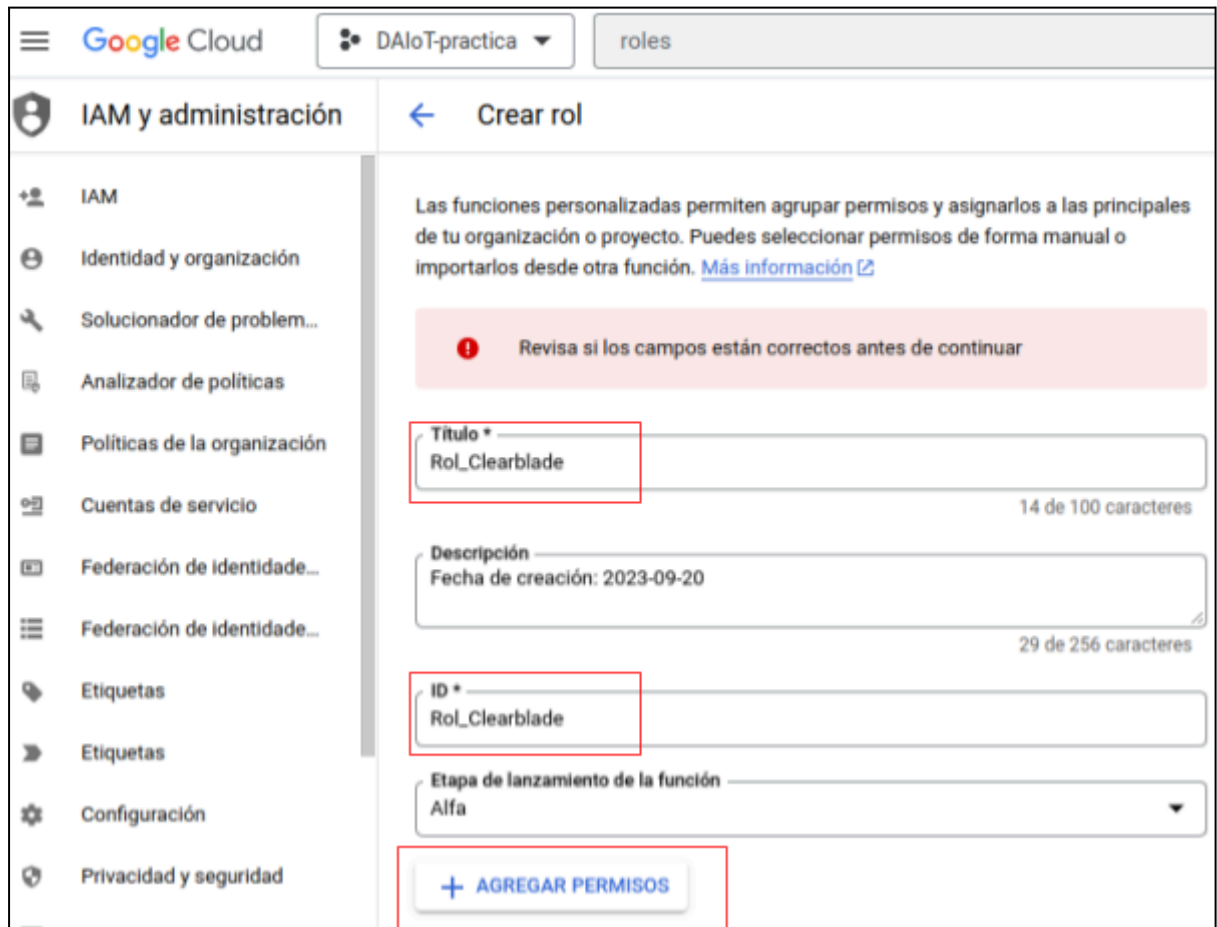
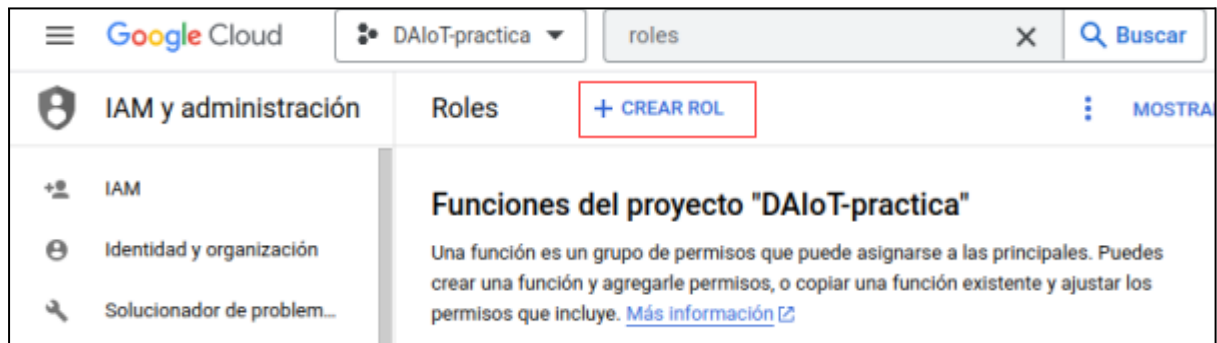
- [cloudiot.devices.list](#) (para migración, actualmente no disponible)
- [cloudiot.registries.get](#) (para migración, actualmente no disponible)
- [cloudiot.registries.list](#) (para migración, actualmente no disponible)
- logging.logEntries.create
- monitoring.metricDescriptors.create
- monitoring.timeSeries.create
- monitoring.timeSeries.list
- pubsub.topics.list
- pubsub.topics.publish

Se debe tener en cuenta que estos permisos apuntaban a una migración desde Google IoT Core. Como el Google IoT Core fue deprecado hace tiempo, los tres primeros permisos no existen más en la lista de roles de GCP y por lo tanto no son necesarios.

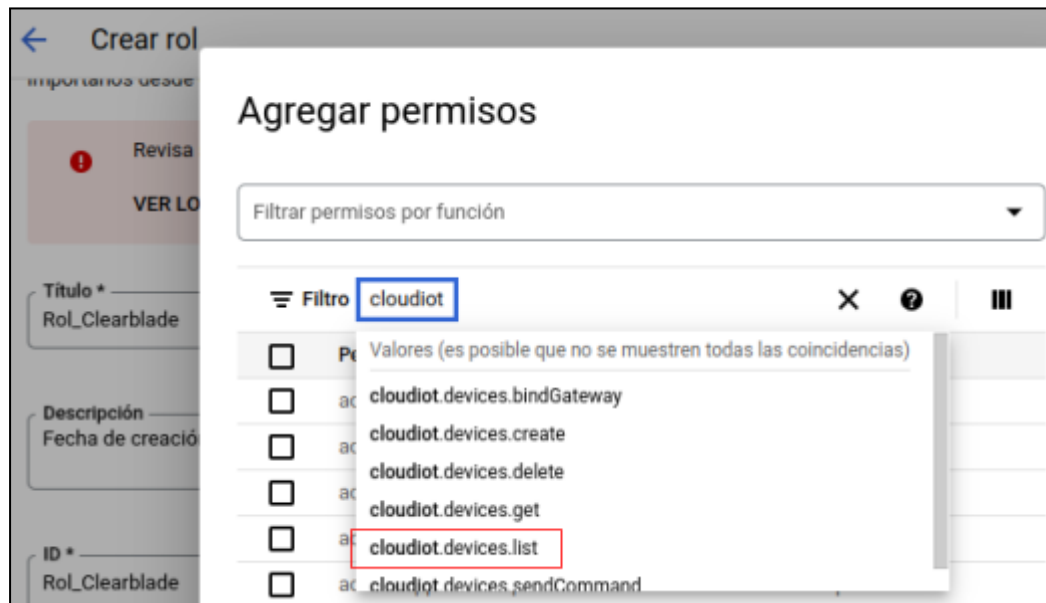
[Aquí](#) se puede ver la explicación de cómo Clearblade utiliza cada rol.

Ingresamos a la sección para administrar roles





Agregaremos uno a uno los permisos detallados por Clearblade, ayudándonos con el filtro y seleccionando cada uno. Aquí va el ejemplo relacionado al primero:

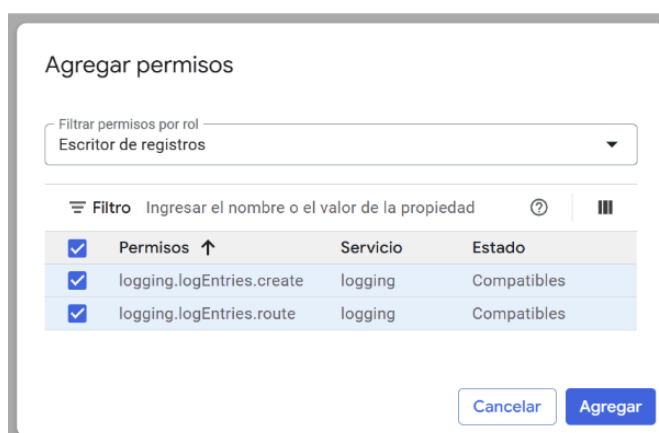


Al realizar este paso, es importante tener presente que si estamos utilizando GCP en español, tendremos que buscar en el filtro las traducciones de los roles que contengan las funciones que estamos necesitando:

Buscar:

1. Escritor de métricas de supervisión
2. Visualizador de Monitoring
3. Editor de Pub/Sub
4. Escritor de Registros

En cada uno de los roles tenemos que tildar las funciones específicas que necesitamos agregar, como se muestra en las siguientes imágenes:



Agregar permisos

Filtrar permisos por rol
Editor de Pub/Sub

Filtro Ingresar el nombre o el valor de la propiedad ?

Permisos ↑	Estado
<input type="checkbox"/> pubsub.subscriptions.update	Compatibles
<input type="checkbox"/> pubsub.topics.attachSubscription	Compatibles
<input type="checkbox"/> pubsub.topics.create	Compatibles
<input type="checkbox"/> pubsub.topics.delete	Compatibles
<input type="checkbox"/> pubsub.topics.detachSubscription	Compatibles
<input type="checkbox"/> pubsub.topics.get	Compatibles
<input checked="" type="checkbox"/> pubsub.topics.list	Compatibles
<input checked="" type="checkbox"/> pubsub.topics.publish	Compatibles
<input type="checkbox"/> pubsub.topics.update	Compatibles
<input type="checkbox"/> pubsub.topics.updateTag	Compatibles

21 - 30 de 34 < >

Cancelar
Agregar

Agregar permisos

Filtrar permisos por rol
Escritor de métricas de supervisión

Filtro Ingresar el nombre o el valor de la propiedad ?

Permisos ↑	Estado
<input checked="" type="checkbox"/> monitoring.metricDescriptors.create	Compatibles
<input type="checkbox"/> monitoring.metricDescriptors.get	Compatibles
<input type="checkbox"/> monitoring.monitoredResourceDescriptors.get	Compatibles
<input type="checkbox"/> monitoring.monitoredResourceDescriptors.list	Compatibles

Cancelar
Agregar

Agregar permisos

Filtrar permisos por rol
Visualizador de Monitoring

Filtro

Ingresar el nombre o el valor de la propiedad

?

|||

Permisos	Estado
<input type="checkbox"/> monitoring.slos.list	Compatibles
<input type="checkbox"/> monitoring.snoozes.get	Compatibles
<input type="checkbox"/> monitoring.snoozes.list	Compatibles
<input checked="" type="checkbox"/> monitoring.timeSeries.list	Compatibles
<input type="checkbox"/> monitoring.uptimeCheckConfigs.get	Compatibles
<input type="checkbox"/> monitoring.uptimeCheckConfigs.list	Compatibles
<input type="checkbox"/> opsconfigmonitoring.resourceMetadata.list	Compatibles
<input type="checkbox"/> resourcemanager.projects.get	Compatibles
<input type="checkbox"/> resourcemanager.projects.list	No aplicables ⚠
<input type="checkbox"/> stackdriver.projects.get	Obsoletos ⓘ

21 – 30 de 31 < >

Cancelar

Agregar

Otorgar permisos con GCP IAM

Luego de generar el rol específico para Clearblade y la cuenta de servicio, ingresamos a la sección IAM y le otorgamos permisos en nuestro proyecto:

Otorgar acceso a "DAIoT-practica"

Otorga a las principales acceso a este recurso y agrega roles para especificar qué acciones pueden realizar. De manera opcional, puedes agregar condiciones para otorgar acceso a las principales solo cuando se cumplan criterios específicos. [Más información sobre las condiciones de IAM](#)

Recurso

DAIoT-practica

Agregar principales

Las principales son usuarios, grupos, dominios o cuentas de servicio. [Más información sobre las principales de IAM](#)

Principales nuevas *

cuenta-servicio-clearblade@daiot-practica.iam.gserviceaccount.com

Asignar funciones

Los roles se componen de conjuntos de permisos y determinan lo que la principal puede hacer con este recurso. [Más información](#)

Rol *

Rol_Clearblade

Condición de IAM (opcional) ?

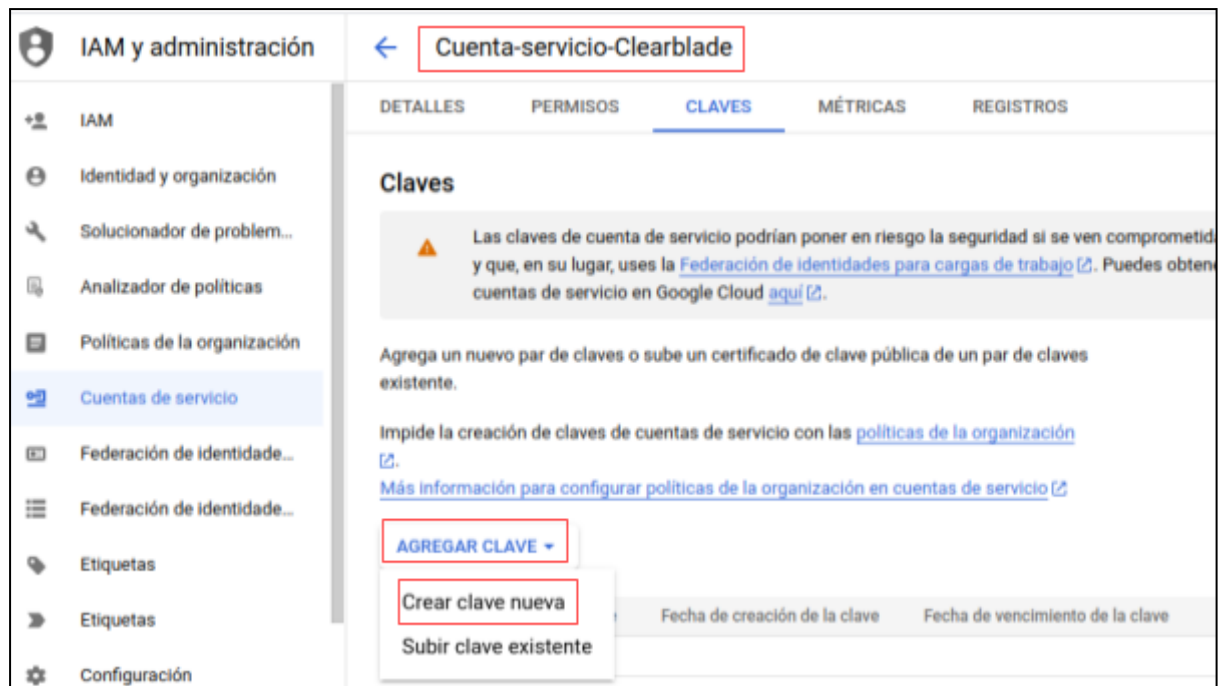
+ AGREGAR CONDICIÓN DE IAM

+ AGREGAR OTRO ROL

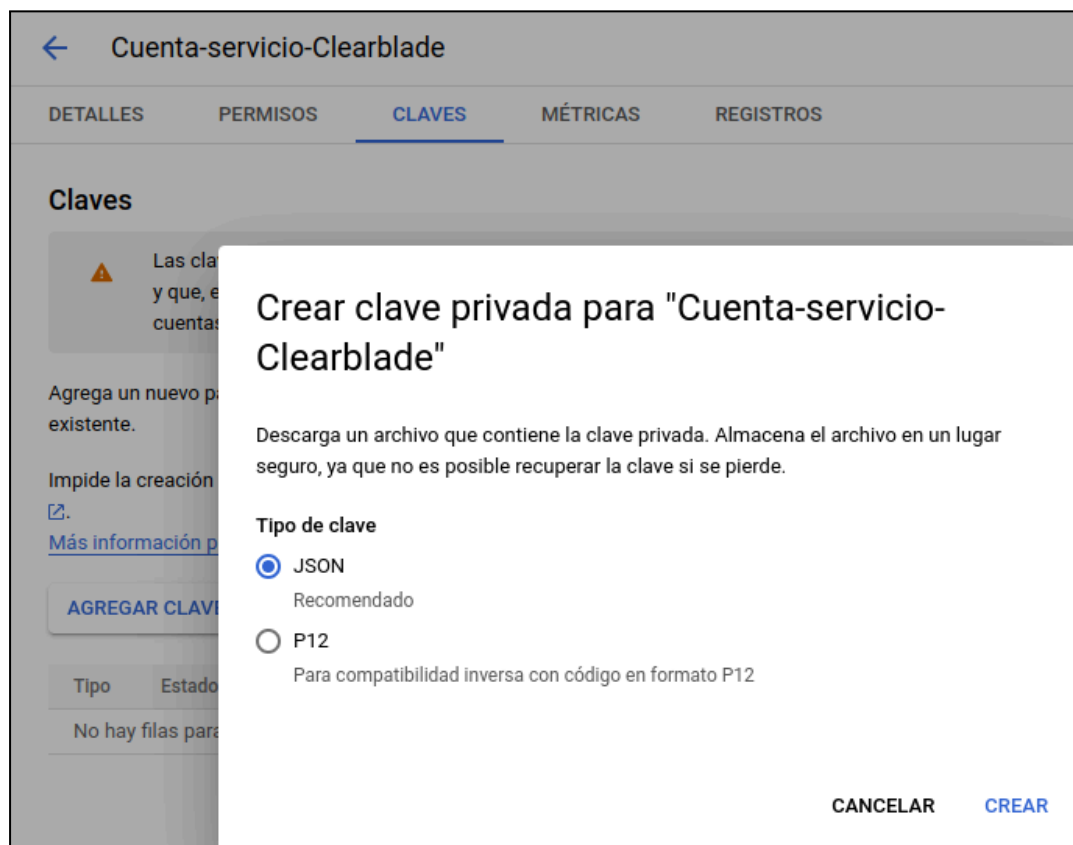
GUARDAR CANCELAR

Generar clave JSON para Clearblade desde GCP

Una vez que tenemos todo configurado, generamos la clave para subir a Clearblade, desde la sección de cuentas de servicio:



Seleccionamos formato JSON, tal como solicita Clearblade:



Una vez creada y descargada la nueva clave, la cargamos en Clearblade y ya podremos ver nuestro proyecto, agregar los registros necesarios y crear nuestros dispositivos.

IoT Core Select project ▾

Create project

How-to guide

1. Enable the [Cloud Resource Manager API](#). ClearBlade IoT Core uses this to ensure that your service account has the necessary roles
2. Create a Google Cloud service account by following instructions in the [Google Cloud documentation](#).
3. Make sure the account has the following permissions:
 - `cloudiot.devices.list`
 - `cloudiot.registries.get`
 - `cloudiot.registries.list`
 - `logging.logEntries.create`
 - `monitoring.metricDescriptors.create`
 - `monitoring.timeSeries.create`
 - `monitoring.timeSeries.list`
 - `pubsub.topics.list`
 - `pubsub.topics.publish`

The following Google roles have the necessary permissions:

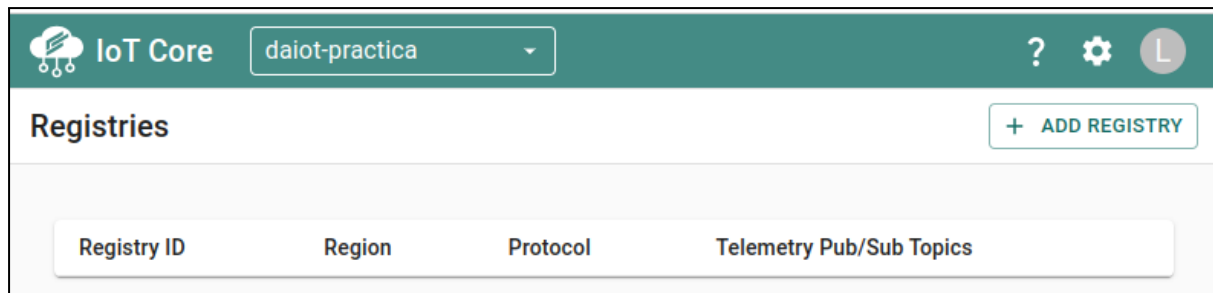
1. Cloud IoT Viewer
2. Pub/Sub Editor
3. Logs Writer
4. Monitoring Metric Writer
5. Monitoring Viewer

4. Download a JSON key for the service account.
5. Upload the JSON file credentials.

UPLOAD

CREATE

Una vez cargadas las credenciales a través del archivo JSON en Clearblade, veremos nuestro proyecto en el combo de selección:



Dar de alta dispositivos en Clearblade

Crear primer “Registry” en Clearblade

Debemos crear un nuevo registro teniendo en cuenta los siguientes detalles:

- Registry ID: registry_1
- Region: us-central1
- Cloud Pub/Sub topics: projects/daiot-practica/topics/sensors_data

The screenshot shows the 'Create a registry' form. Under 'Registry properties', the 'Registry ID' is 'registry_1' and the 'Region' is 'us-central1'. Under 'Cloud Pub/Sub topics', the selected topic is 'projects/daiot-practica/topics/sensors_data'. At the bottom, there's an 'Additional Topics' section with an '+ ADD' button, a 'SHOW ADVANCED OPTIONS' link, and 'CREATE' and 'CANCEL' buttons.

Dar de alta dispositivos IoT

Ingresando al registro creado, ya se pueden dar de alta dispositivos IoT en el hub de Clearblade. Se hará teniendo en cuenta los siguientes detalles:

- Crearemos la cantidad que deseemos.
- El formato del nombre será el siguiente “device-101”.
- Todos los dispositivos comenzarán con “device-” y luego continuará un número de tres dígitos.
- En la sección “Authentication” se seleccionará el tipo de certificado RS256_X509
- Se cargará de forma manual el contenido de las claves generadas previamente, específicamente la incluida en el archivo “rsa_lz_cert.pem”.

IoT Core daiot-practica

Registries ← **Create a device**

Registry Details

Devices

Gateways

Monitoring

Authentication (optional)

Specify the public key that will be used to authenticate this device. You can leave the key empty, but devices will not be able to connect to ClearBlade without a key.

Input method

☒ Enter manually

☐ Upload

Public key format

RS256_X509

Public key value

```
AT2hwqAixOiAyvvUexlv1q3XMyLsa2EHckc6JmW6GomXgezxi
Xx+pE1MwkFJtiG
t++V2ldOzITJ8dc/dEfCuq/3StweQ/xk4vr7XmdHOuZnll9vvikyWE
ypaNyiQWaA
a7KZLAq51/dP11/IHGIMd+3eUAUUQn2LUgkm+oWDSm50leKf
ULnzellfkUWyQ4UY
Q7r572Wra3kD
-----END CERTIFICATE-----
```

Public key expiration date (optional)

☐ Expires on:

MM/DD/YYYY hh:mm aa

↑ COMMUNICATION, LOGGING, AUTHENTICATION

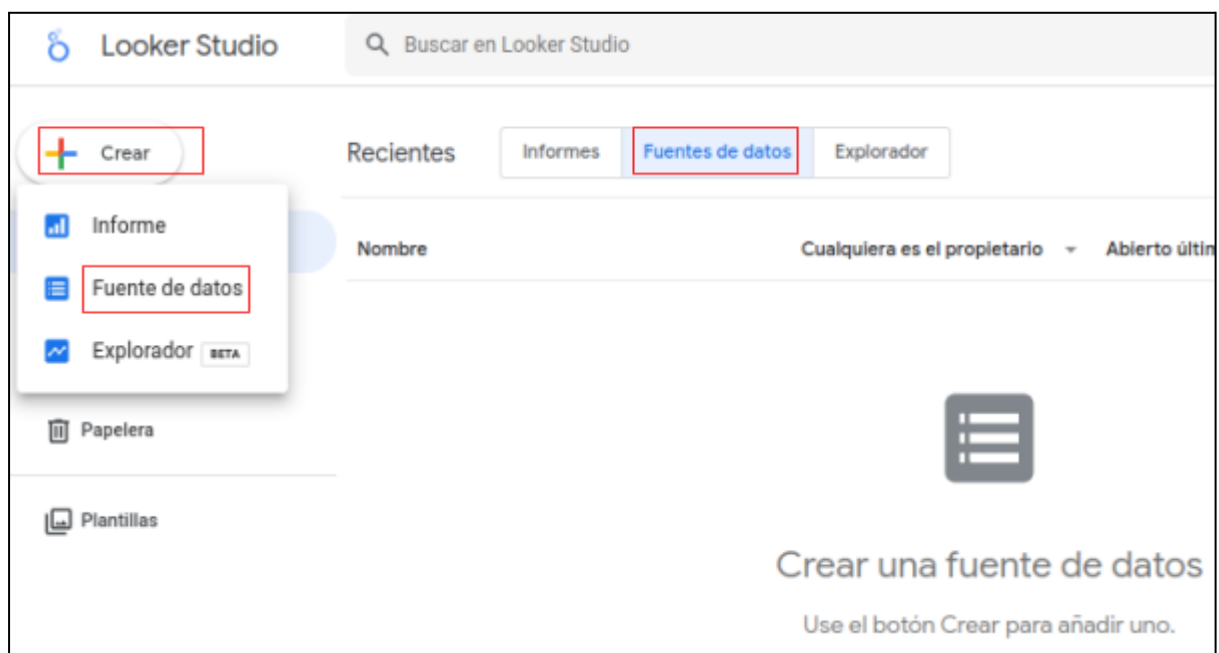
SUBMIT

Looker Studio - Visualización de datos básica

Looker studio es una aplicación de Google cuyo objetivo no tiene relación alguna al mundo IoT. En este caso se utilizará simplemente para visualizar algunos datos de la base de datos, sin implementar un backend y frontend, pero se aclara que de ningún modo es una aplicación apta o destinada para generar dashboards IoT ni nada similar.

Creación de conexiones con Bigquery

El primer paso para generar gráficos o mostrar datos dentro de Looker Studio, consiste en crear las conexiones necesarias con otros servicios o fuentes de datos. En este caso se utilizará el conector para Bigquery.



Se generarán dos conexiones, una para cada vista generada dentro de Bigquery.

La conexión a la primera vista se utiliza para alimentar gráficos que simulan series temporales y aquellos que incluyen máximos, mínimos y promedios.

Fuente de datos sin título

SELECCIONAR CONECTOR

Haga que sus informes de BigQuery se carguen más rápido con BigQuery BI Engine. [Más información](#)

BigQuery
 De Google
 BigQuery es un almacén de datos de análisis de bajo coste administrado íntegramente por Google y en el que se utiliza una escala de petabytes. Este servicio se cobra por consulta o procesamiento de datos, y el importe de esas consultas se carga en la tarjeta de crédito del proyecto de facturación.

[MÁS INFORMACIÓN](#)
[NOTIFICAR UN PROBLEMA](#)

PROYECTOS RECIENTES	Proyecto	Conjunto de datos	Tabla
MIS PROYECTOS	Introduzca el ID del proyecto manualmente	datos	sensores_tph_5min_interval
PROYECTOS COMPARTIDOS	DAlot-practica		sensores_tph_desc_ubicacion
CONSULTA PERSONALIZADA	IoT-dev		sensores_tph_origen
CONJUNTOS DE DATOS PÚBLICOS	test12345		sensores_tph_ultimos_valores
	DAlot-8va		
	Sensores IOT iris v1		
	Autorizacion-IoT-App		

sensores_tph_5min_interval

Compartir

poldo Zimperz | Actualización de datos: 12 horas | Acceso a visualizaciones comunitarias **Activado** | Edición de campos de informes: **Activada**






EDITAR CONEXIÓN | FILTRAR POR CORREO ELECTRÓNICO

AÑADIR UN CAMPO

AÑADIR UN PARÁMETRO


Campo	Tipo	Agregación predeterminada	Descripción
DIMENSIONES (7)			
dev_id	Fecha	Ninguna	
humedad	123 Número	Ninguna	
presion	123 Número	Ninguna	
propietario	RBC Texto	Ninguna	
rsi	123 Número	Ninguna	
temperatura	123 Número	Ninguna	
tiempo_registro	Fecha y hora	Ninguna	
MÉTRICAS (1)			
Record Count	123 Número	Automática	


La conexión a la segunda vista se utiliza para alimentar gráficos que muestren últimos valores medidos de uno o más dispositivos.

 sensores_tph_ultimos_valores   Compartir  

[VOLVER A CONECTAR](#)

← SELECCIONAR CONECTOR CAMPOS →






 Haga que sus informes de BigQuery se carguen más rápido con BigQuery BI Engine. [Más información](#)


 **BigQuery**
De Google



BigQuery es un almacén de datos de análisis de bajo coste administrado íntegramente por Google y en el que se utiliza una escala de petabytes. Este servicio se cobra por consulta o procesamiento de datos, y el importe de esas consultas se carga en la tarjeta de crédito del proyecto de facturación.


[MÁS INFORMACIÓN](#) [NOTIFICAR UN PROBLEMA](#)

PROYECTOS RECIENTES	Proyecto	Conjunto de datos	Tabla
MIS PROYECTOS	Introduzca el ID del proyecto manualmente	datos	sensores_tph_ultimos_valores
PROYECTOS COMPARTIDOS	DAIoT-practica		sensores_tph_5min_interval
CONSULTA PERSONALIZADA	IoT-dev		sensores_tph_desc_ubicacion
CONJUNTOS DE DATOS PÚBLICOS	test12345		sensores_tph_origen
	DAIoT-8va		
	Sensores IOT iris v1		
	Autorizacion-IoT-App		

 sensores_tph_ultimos_valores   Compartir  

opoldo Zimperz | Actualización de datos: 12 horas | Acceso a visualizaciones comunitarias **Activado** | Edición de campos de informes: **Activada**  [CREAR INFORME](#) [EXPLORAR](#)

← EDITAR CONEXIÓN | FILTRAR POR CORREO ELECTRÓNICO  AÑADIR UN CAMPO  AÑADIR UN PARÁMETRO

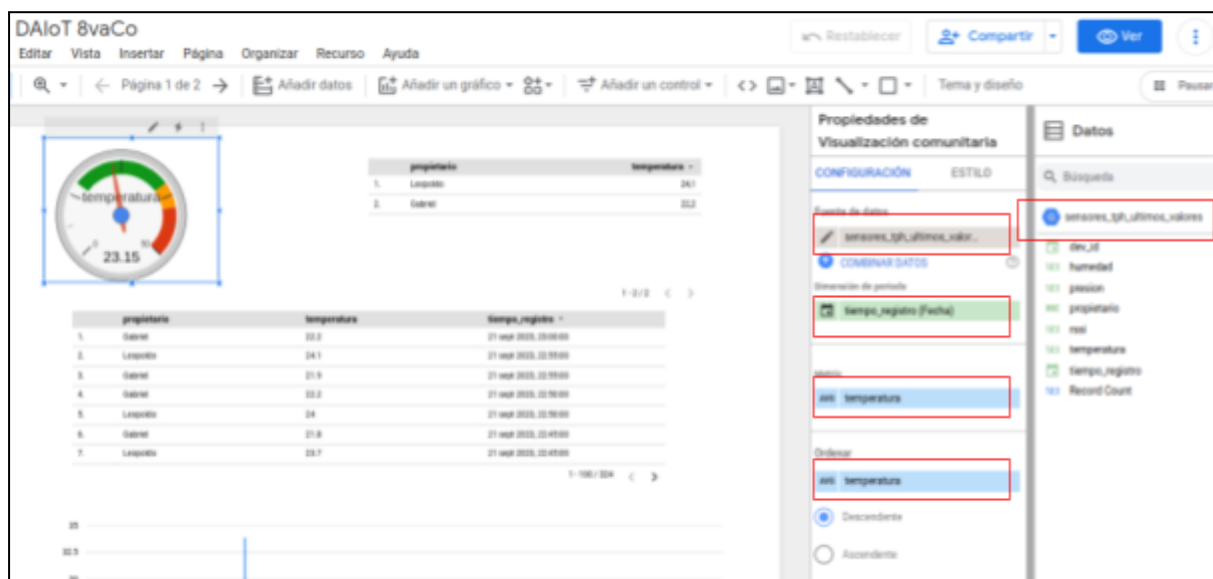
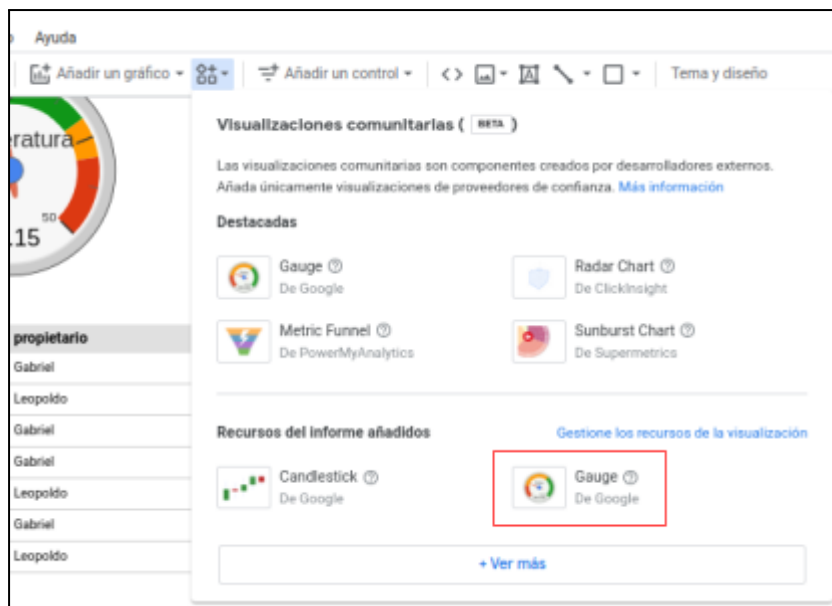
Agregación predeterminada Descripción ↓  Buscar campos

Campo ↓	Tipo ↓	Descripción ↓
DIMENSIONES (7)		
dev_id	Fecha	Ninguna
humedad	123 Número	Ninguna
presion	123 Número	Ninguna
propietario	RBC Texto	Ninguna
rsi	123 Número	Ninguna
temperatura	123 Número	Ninguna
tiempo_registro	Fecha y hora	Ninguna
MÉTRICAS (1)		
Record Count	123 Número	Automática

Agregar gráficos al informe

Se pueden agregar distintos tipos de gráficos. Se muestra la configuración de alguno de ellos a continuación:

Gauge



Candlestick

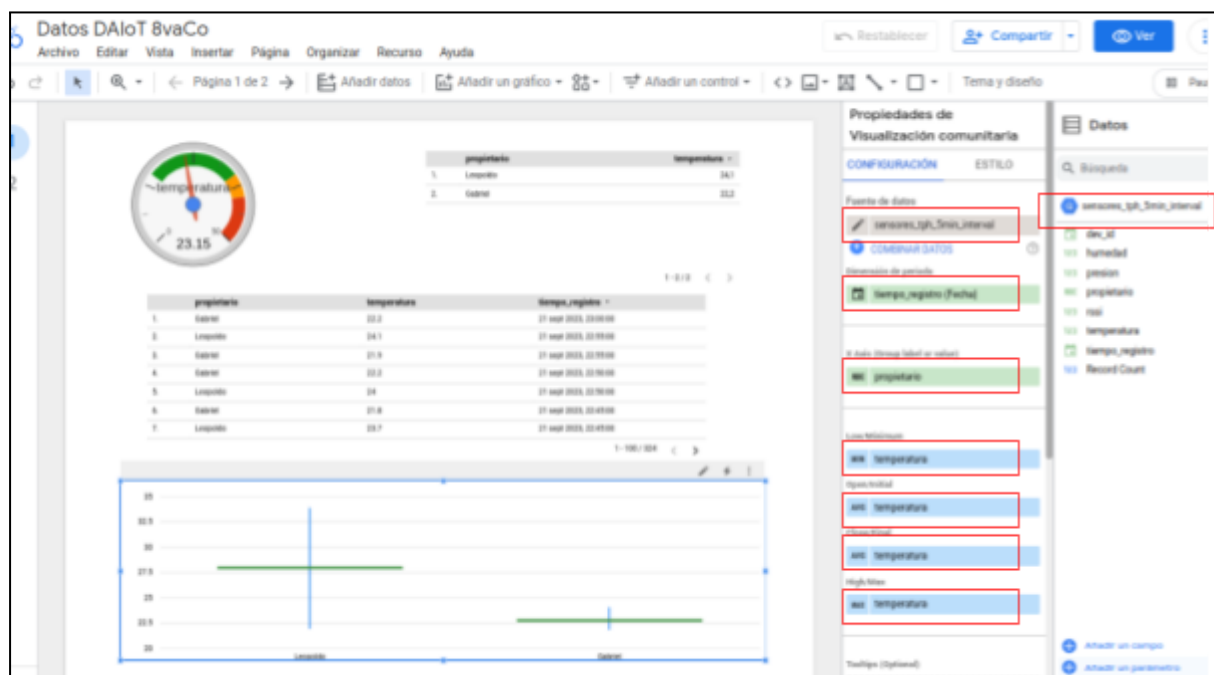
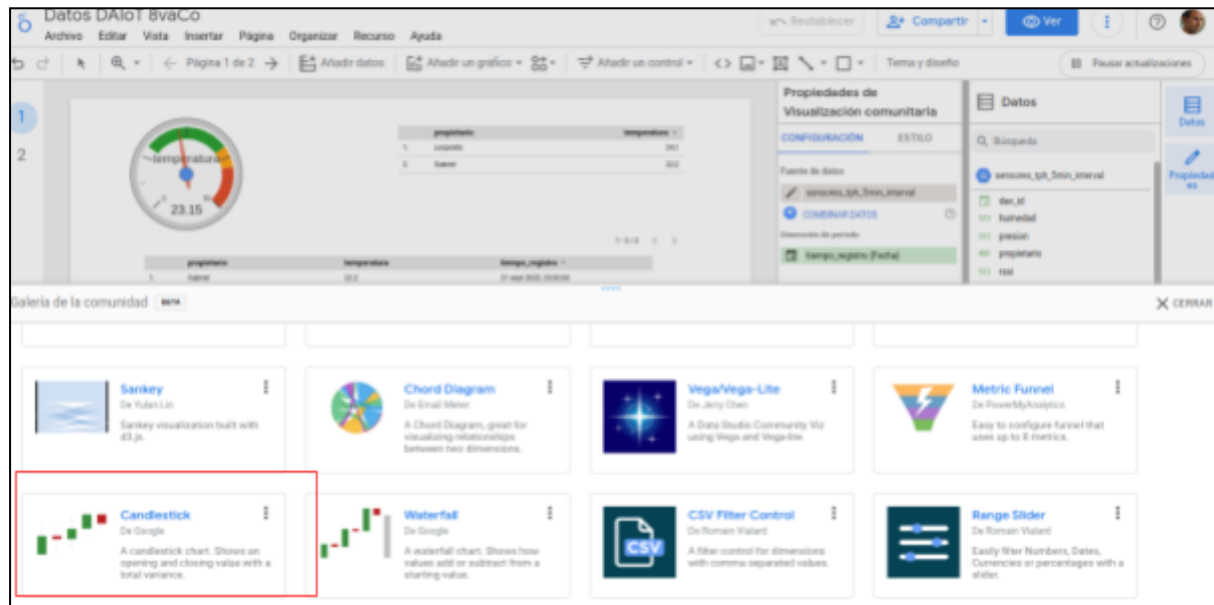
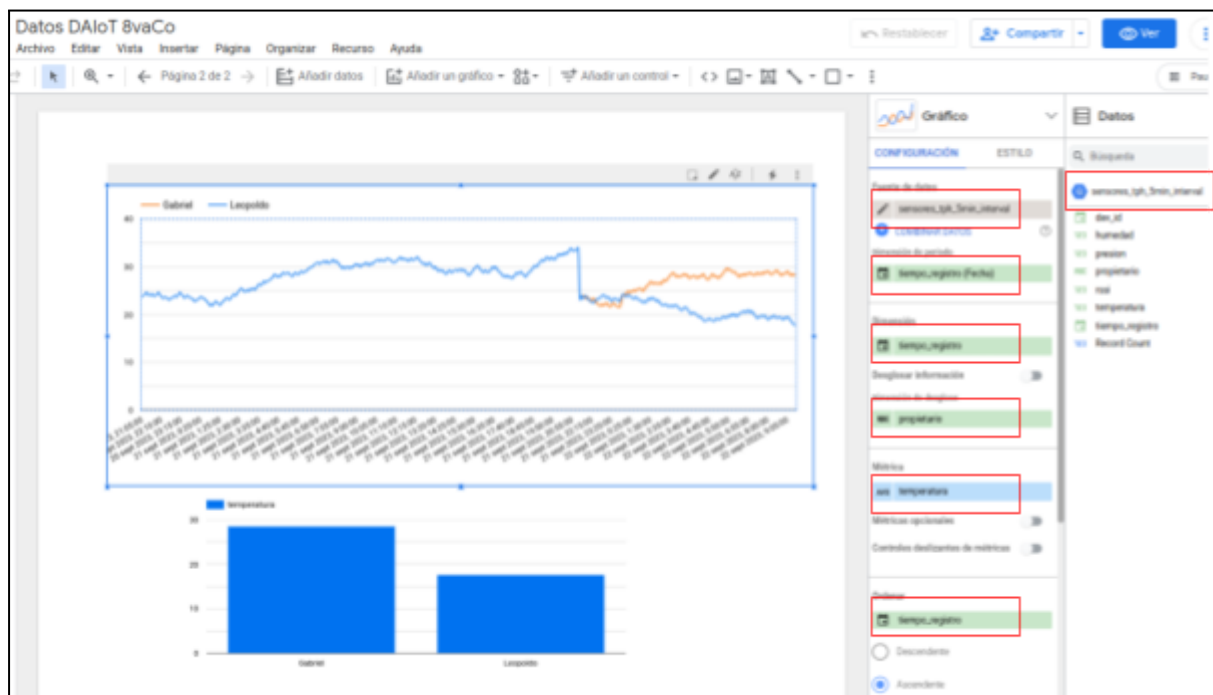
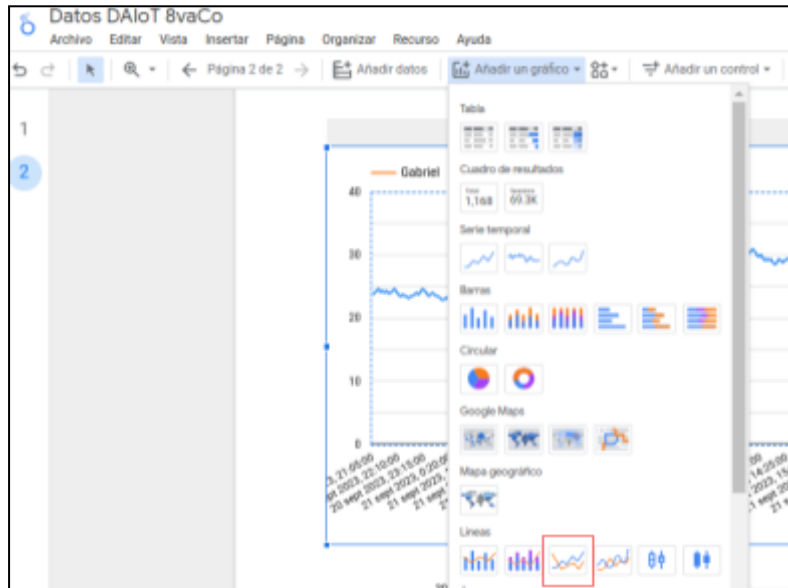


Gráfico de línea

(No utilizar serie temporal)



Puesta en marcha de aplicación en ESP32

Clonar o forkear [el proyecto desde Github](#).

Configurar lo siguiente en “main.c”:

- WIFI_SSID
- WIFI_PASS
- DEVICE_ID

Compilar utilizando ESP-IDF versión 5 o 5.1

Flashear en esp32. Si todo el proyecto está completo se conectará y comenzará a cargar datos en la base de datos, que podrán visualizarse desde Looker Studio.