Documentación Sprint 0 | API Rest, Endpoints y Lógica de Negocio

En este documento vamos a explicar el funcionamiento de la API Rest y Endpoints con ejemplos de datos y explicado paso a paso. También explicaremos la lógica de negocio con su constructor y sus métodos.

NOTA IMPORTANTE: El objetivo de este documento es definir la estructura base de la API Rest, los endpoints principales y las lógica de negocio que gestionará las mediciones. El contenido de este documento corresponde a la primera versión funcional, centrado en la comunicación entre la aplicación Android y el servidor mediante peticiones GET y POST.

Autor: Ferran Sansaloni Prats

Fecha: 07/10/2025

Contenido

1.- Introducción

Esta API permite gestionar las mediciones de sensores de CO2.

Incluye los siguientes endpoints:

- GET /api_get.php → Recupera las mediciones de la base de datos.
- POST /api_post.php → Inserta nuevas mediciones a la base de datos.

La Lógica de Negocio se gestiona mediante la clase LogicaMediciones.php.

2.- Endpoints

2.1 GET /api_get.php

Descripción: Devuelve todas las mediciones almacenadas en la base de datos.

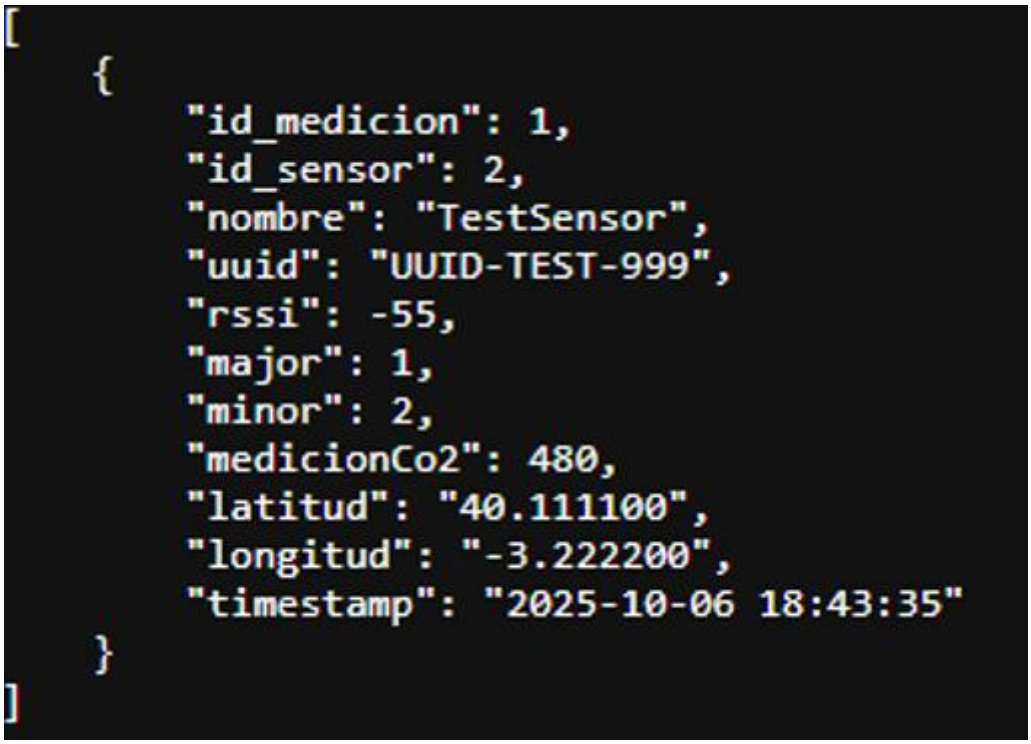
Parámetros GET:

- Id_sensor (int): Filtra mediciones por sensor.

Ejemplo de solicitud:

Le pasamos al método un id, Ej: “id_sensor = 1” y nos devuelve todas las mediciones que el id_sensor = 1 ha realizado.

Respuesta JSON:



```
[
    {
        "id_medicion": 1,
        "id_sensor": 2,
        "nombre": "TestSensor",
        "uuid": "UUID-TEST-999",
        "rssi": -55,
        "major": 1,
        "minor": 2,
        "medicionCo2": 480,
        "latitud": "40.111100",
        "longitud": "-3.222200",
        "timestamp": "2025-10-06 18:43:35"
    }
]
```


2.2 GET /api_post.php

Descripción: Recibe mediciones desde Android Studio y las almacena en la base de datos.

Cabeceras HTTP:

- Content-Type: application/json
- Access-Control-Allow-Origin: *
- Access-Control-Allow-Methods: POST

Cuerpo del JSON esperado:


```
{
 "nombre": "Sensor1",
 "uuid": "fistro",
 "rssi": -50,
 "major": 0,
 "minor": 0,
 "latitud": 0.0,
 "longitud": 0.0,
 "medicionCo2":  1234,
 "id_sensor": 1
}
```


Validaciones:

UUID, RSSI y medicionCo2 son obligatorios. Si el JSON recibido no tiene estos campos no los guardará en la base de datos.

Ejemplo de medida guardada:


```
Medida enviada correctamente: {"success":true,"mensaje":"Medida guardada correctamente","id":1}
```

3.- Lógica de negocio: LogicaMediciones.php

Clase LogicaMediciones: Gestiona las operaciones sobre las mediciones.

Constructor:

public function _construc($con) → Inicializa la conexión a la base de datos.

Métodos:

1.- guardarMediciones(nombre, rssi, major, minor, latitud, longitud, co2, id_sensor = 1)

- Añade una medición a la base de datos y devuelve el id de la medición.

2.- recogerMediciones(id_sensor = null)

- Recupera todas las mediciones. Devuelve un array asociativo con las mediciones de ese sensor.