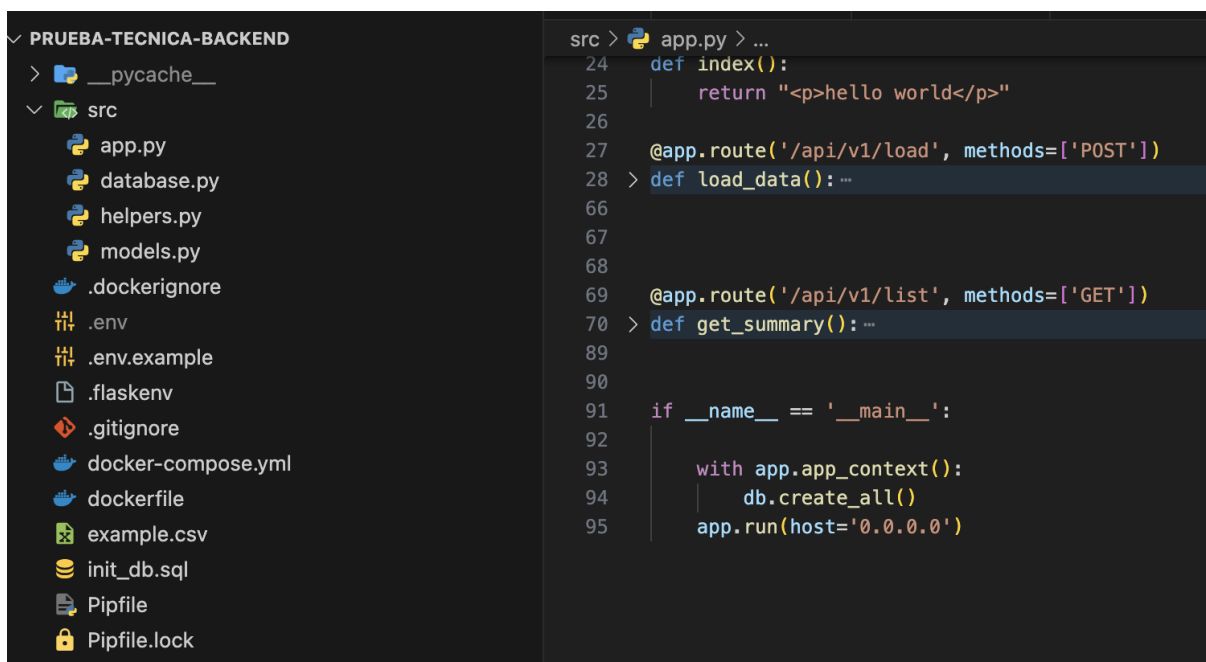


EVIDENCIAS

En este apartado se presentan las evidencias visuales del funcionamiento tanto del frontend como del backend de la aplicación. Las imágenes muestran capturas de pantalla de las distintas vistas y procesos relevantes para el correcto funcionamiento del sistema.

Backend

Creacion de las APIs

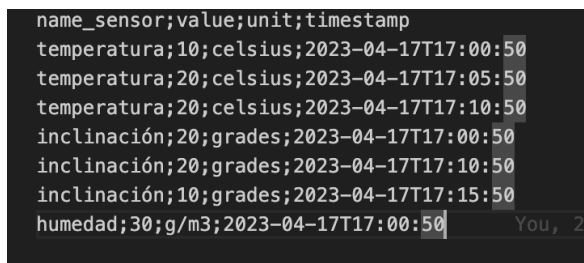


The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure for 'PRUEBA-TECNICA-BACKEND' with files like __pycache__, src, app.py, database.py, helpers.py, models.py, .dockerignore, .env, .env.example, .flaskenv, .gitignore, docker-compose.yml, dockerfile, example.csv, init_db.sql, Pipfile, and Pipfile.lock. The code editor shows the content of app.py, which includes a Flask application with routes for index, load_data, and get_summary, and a main function to run the application.

```
src > app.py > ...
24 def index():
25     return "<p>hello world</p>"
26
27 @app.route('/api/v1/load', methods=['POST'])
28 > def load_data(): ...
66
67
68
69 @app.route('/api/v1/list', methods=['GET'])
70 > def get_summary(): ...
89
90
91 if __name__ == '__main__':
92
93     with app.app_context():
94         db.create_all()
95     app.run(host='0.0.0.0')
```

Registro de la Carga en la Base de Datos con postman, cargamos el example.csv

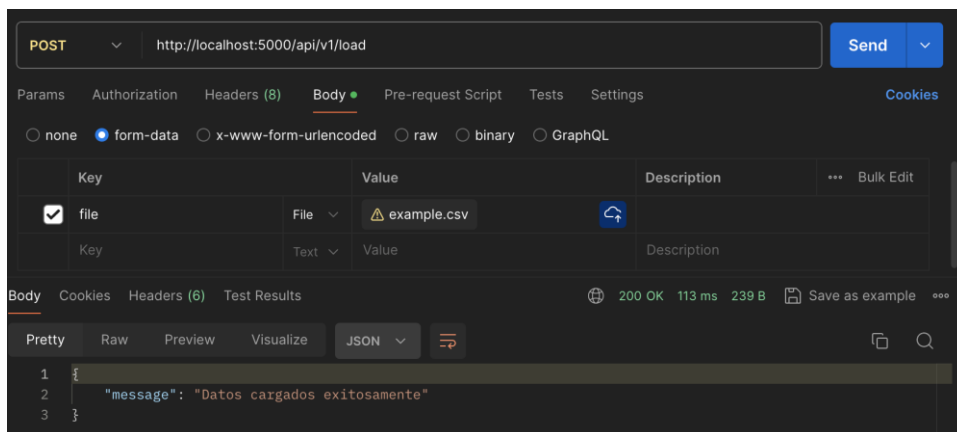
Example.csv



The screenshot shows the content of a CSV file named 'Example.csv'. The file contains a header row and several data rows. The header row is 'name_sensor;value;unit;timestamp'. The data rows are: 'temperatura;10;celsius;2023-04-17T17:00:50', 'temperatura;20;celsius;2023-04-17T17:05:50', 'temperatura;20;celsius;2023-04-17T17:10:50', 'inclinación;20;grades;2023-04-17T17:00:50', 'inclinación;20;grades;2023-04-17T17:10:50', 'inclinación;10;grades;2023-04-17T17:15:50', and 'humedad;30;g/m3;2023-04-17T17:00:50'. The file is open in a text editor, and the cursor is at the end of the last row.

```
name_sensor;value;unit;timestamp
temperatura;10;celsius;2023-04-17T17:00:50
temperatura;20;celsius;2023-04-17T17:05:50
temperatura;20;celsius;2023-04-17T17:10:50
inclinación;20;grades;2023-04-17T17:00:50
inclinación;20;grades;2023-04-17T17:10:50
inclinación;10;grades;2023-04-17T17:15:50
humedad;30;g/m3;2023-04-17T17:00:50
```

- Api/v1/load



Verificamos la tabla `measurement_detail`

```
select *
from measurement_detail ;
```

id	name_sensor	value	unit	timestamp
1	temperatura	10	celsius	2023-04-17 17:00:50.000000
2	temperatura	20	celsius	2023-04-17 17:05:50.000000
3	temperatura	20	celsius	2023-04-17 17:10:50.000000
4	inclinación	20	grades	2023-04-17 17:00:50.000000
5	inclinación	20	grades	2023-04-17 17:10:50.000000
6	inclinación	10	grades	2023-04-17 17:15:50.000000
7	humedad	30	g/m3	2023-04-17 17:00:50.000000

Efectivamente los datos se ingresaron correctamente

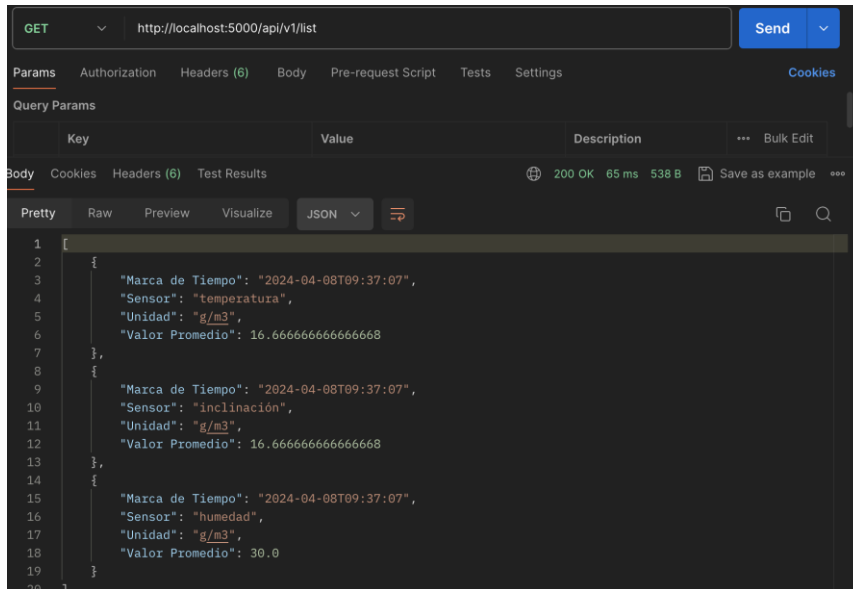
Ahora verificamos la tabla `summary`

```
select *
from summary ;
```

id	name_sensor	average_value	unit	timestamp
10	temperatura	16.666666666666668	g/m3	2024-04-08 09:37:07.352676
11	inclinación	16.666666666666668	g/m3	2024-04-08 09:37:07.352921
12	humedad	30	g/m3	2024-04-08 09:37:07.352977

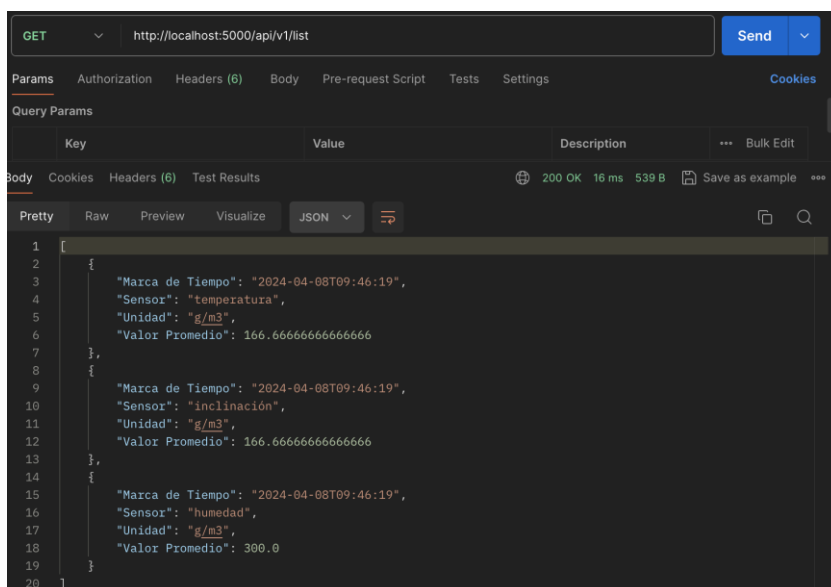
- Api/v1/list

Nuevamente hacemos una consulta con postman



OJO: los promedios solo se calculan de la última carga, si se desea calcular de toda la tabla cree una función llamada “update_summary_full()” ubicada en helper.py

Ahora editare el archivo example.csv y que los promedios sean diferentes. Agregando un 0 a todos los valores

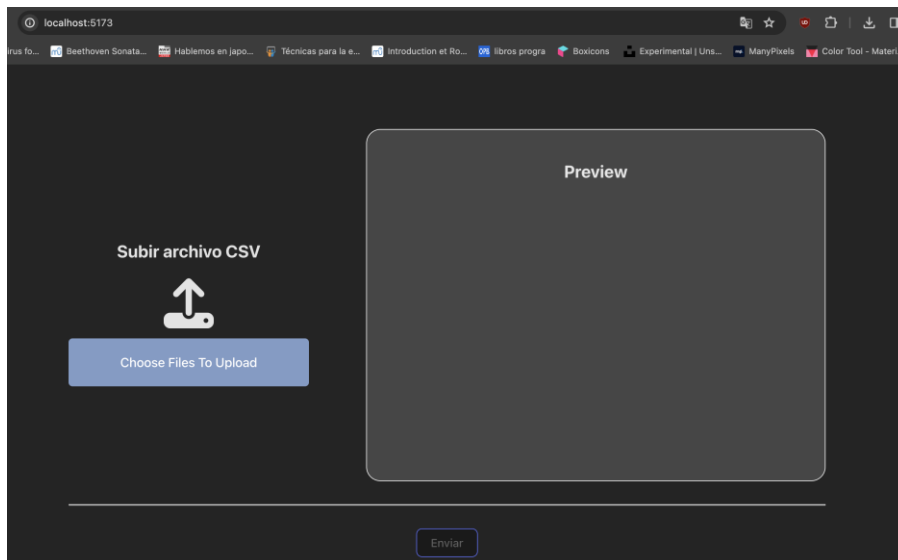


Como puede observar los promedios se multiplicaron por 10 .

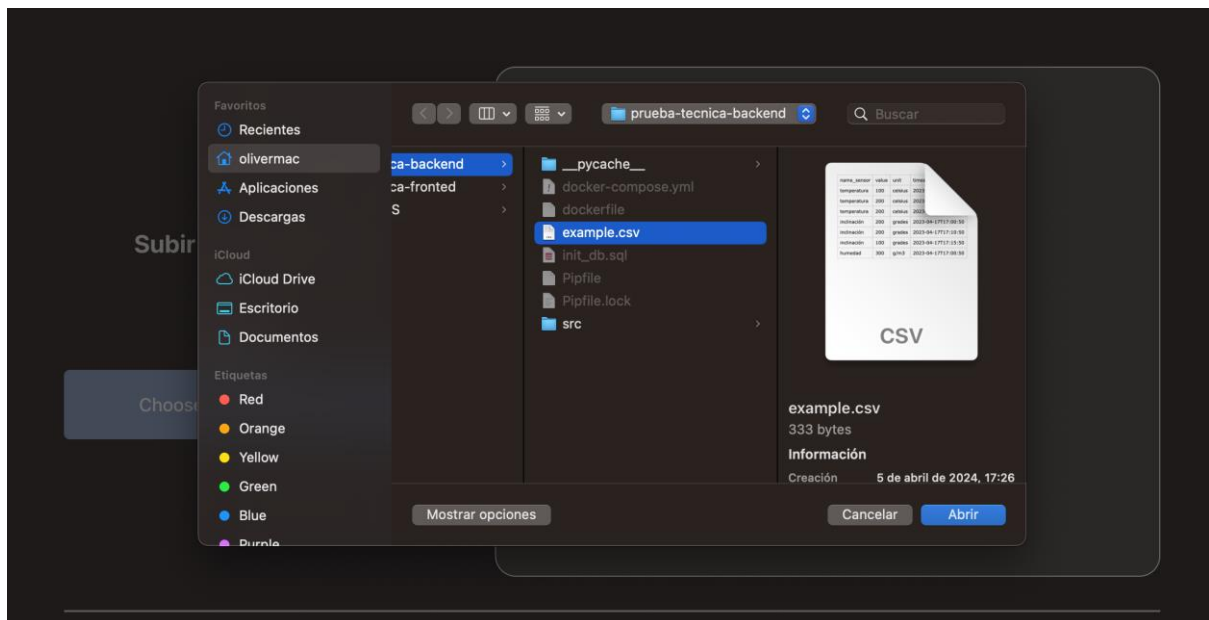
Frontend

Vista de Carga de Archivos CSV

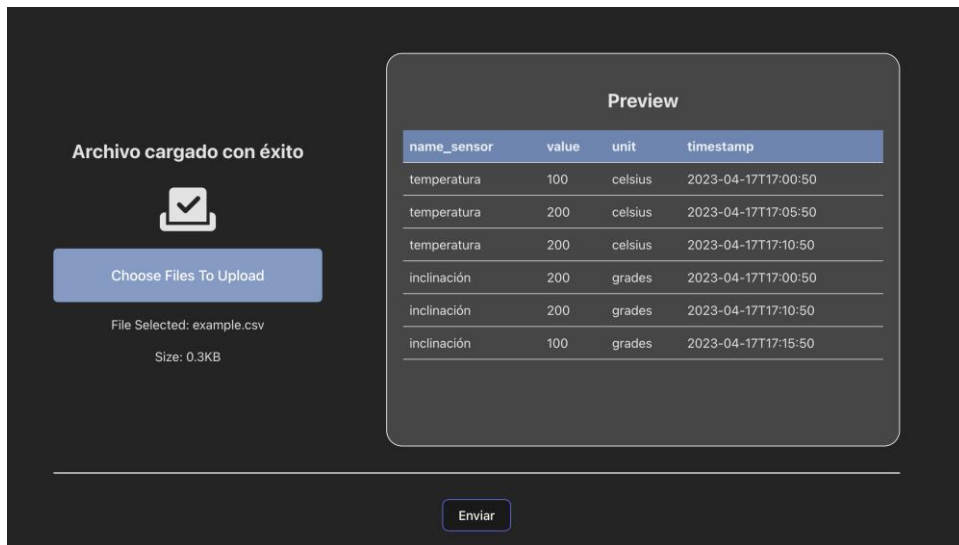
Entramos a la vista



Cargamos el archivo



Se muestra el preview para verificar los datos



Le damos click a enviar, el cual nos llevara a ver los resúmenes de la carga actual

Vista de Resumen de Valores Promedio



DOCKERIZACIÓN

Para el front end corremos

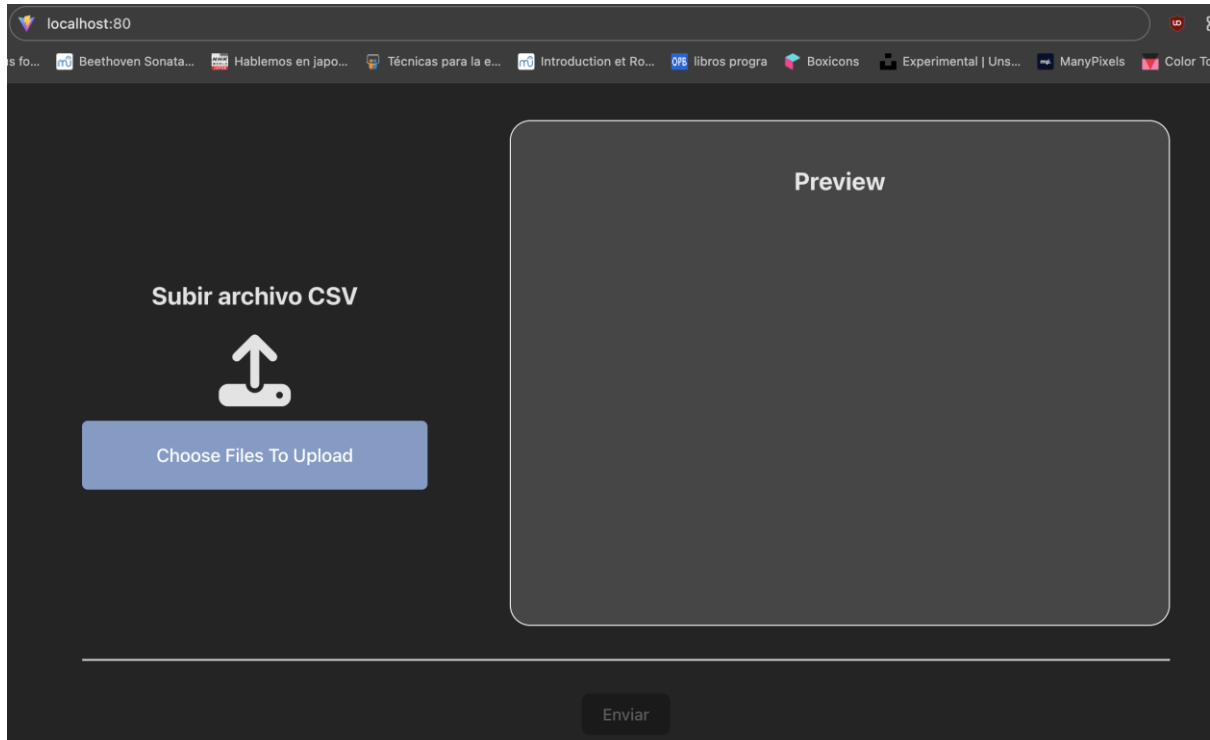
"Docker build --no-cache -t prueba-tecnica-frontend ."

"docker run --name sensor-frontend -d -it -p 80:80/tcp prueba-tecnica-frontend"

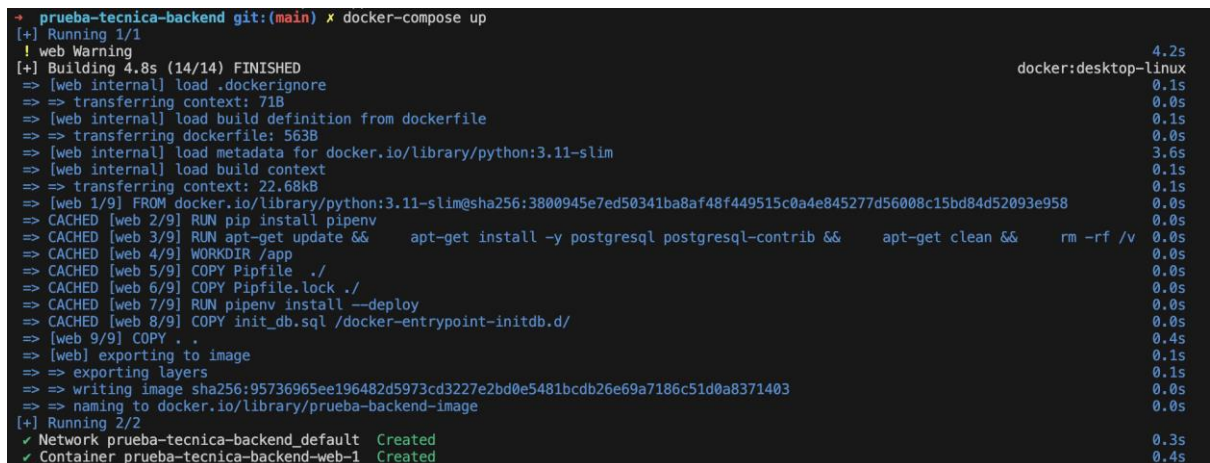
Verificamos si el contenedor funciona



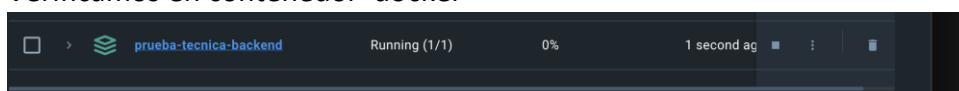
Nos dirigimos a <http://localhost:80>



Ahora para el backend corremos "docker-compose up"



Verificamos en contenedor docker

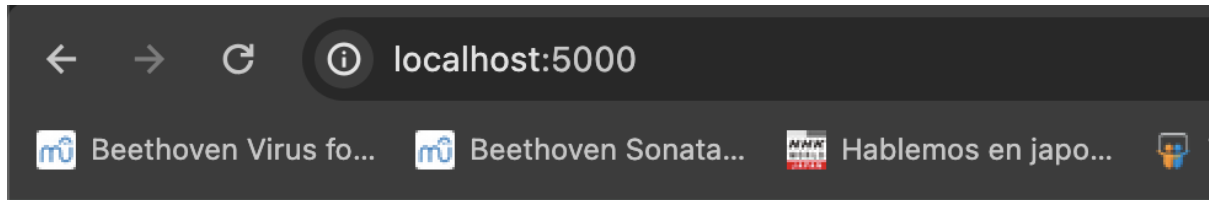


Nota: no olvidar añadir los contenedores a la misma network o red con

docker network create sensorNetwork

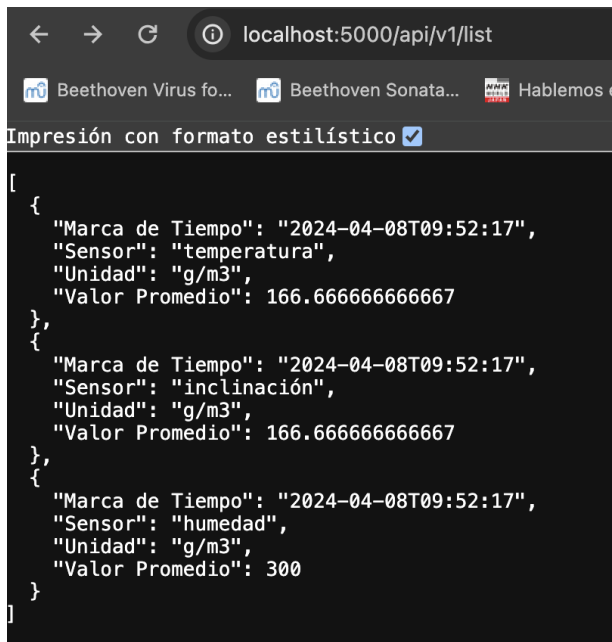
docker network connect sensorNetwork <nombre_del_contenedor>

Veamos el puerto 5000 que es el expuesto



hello world

También veamos la API de /api/v1/list




Finalmente la última verificación de carga y comunicación de los contenedores en el puerto 80

localhost:80

us fo...Beethoven Sonata...Hablemos en japo...Técnicas para la e...Introduction et Ro...libros prograBoxiconsExperimental | Uns...ManyPixelsColor

Archivo cargado con éxito



Choose Files To Upload

File Selected: example.csv

Size: 0.3KB

Preview

name_sensor	value	unit	timestamp
temperatura	100	celsius	2023-04-17T17:00:50
temperatura	200	celsius	2023-04-17T17:05:50
temperatura	200	celsius	2023-04-17T17:10:50
inclinación	200	grades	2023-04-17T17:00:50
inclinación	200	grades	2023-04-17T17:10:50
inclinación	100	grades	2023-04-17T17:15:50

Enviar

le damos a enviar

localhost:80/resumen

s fo...Beethoven Sonata...Hablemos en japo...Técnicas para la e...Introduction et Ro...libros prograBoxiconsExperimental | Uns...ManyPi

Resumen

Sensor	Valor Promedio	Unidad	Marca de Tiempo
temperatura	166.66666666666666	g/m3	2024-04-08T09:52:17
inclinación	166.66666666666666	g/m3	2024-04-08T09:52:17
humedad	300	g/m3	2024-04-08T09:52:17

Cargar nuevo archivo