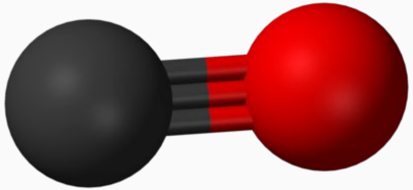
Smoky

Smoke-monitoring web app

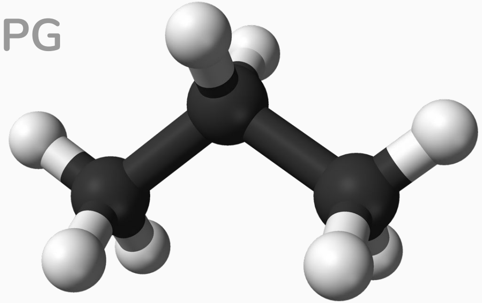
Why Smoky?

CO



- To **avoid Carbon monoxide poisoning**
  (In Italy, deaths due to CO poisoning are estimated between 500 and 600 for every year.
  (source: salute.gov.it)

LPG

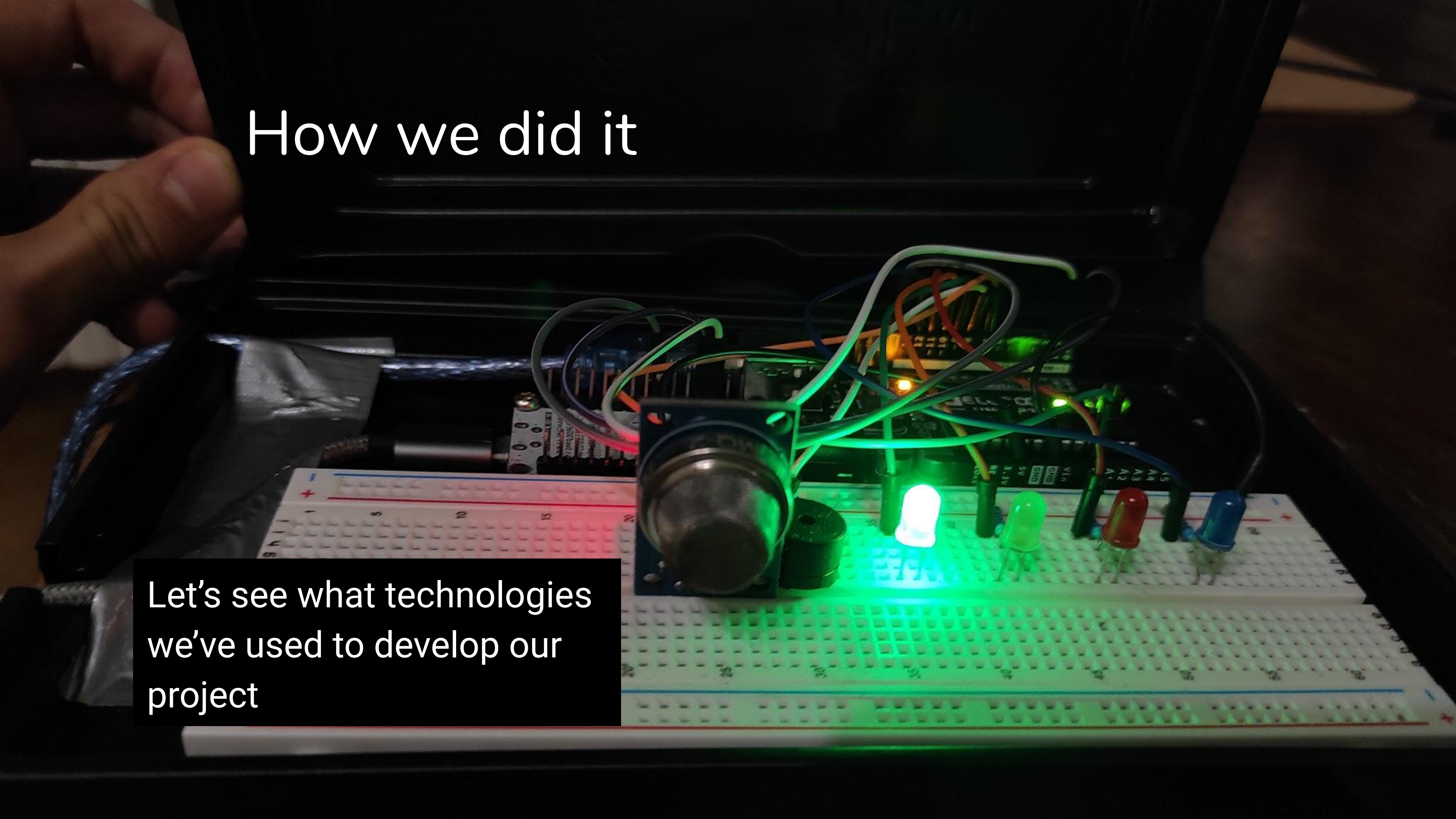


- To **avoid explosions caused by LPG leaks**: even if lpg safety systems have improved a lot, sometimes that's not enough.

Why Smoky?

One of the issues of car-sharing services is that some users **smoke inside the car** without caring about the service rules. The result, most of the times, is a **bad user experience** and sometimes even **upholstery damage**, caused by ash. The only way car-sharing providers can be aware of that misuse is actually just through user reports. We wanted to **realize a straigh-forward automatic solution**.

How we did it



Let’s see what technologies we’ve used to develop our project

PHP

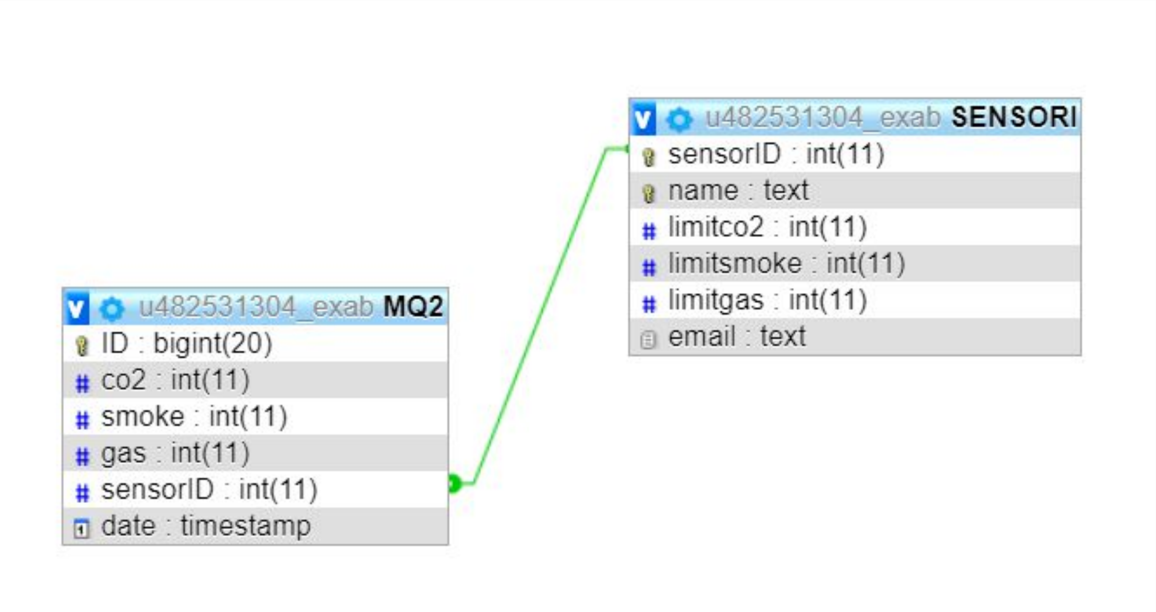
A classic (but powerful) solution. Our aim was to reproduce something we could be able to verify in real world, even when outside the sensors LAN and not just as an hypothetical tool: that’s why we’ve built a real database with the tables we needed both for the data sent by the sensors and for the graph view settings. the data publish is made with GET requests, while POST requests are used to retrieve data from the database.

php

MySQL

To manage data, we've created two different database tables:

- **MQ2**, the table in which the nodes directly publish the real-time data.
- **SENSORI**, the table in use to manage the view configurations in the GUI.



u482531304_exab **MQ2**

ID : bigint(20)
co2 : int(11)
smoke : int(11)
gas : int(11)
sensorID : int(11)
date : timestamp

u482531304_exab **SENSORI**

sensorID : int(11)
name : text
limitco2 : int(11)
limitsmoke : int(11)
limitgas : int(11)
email : text

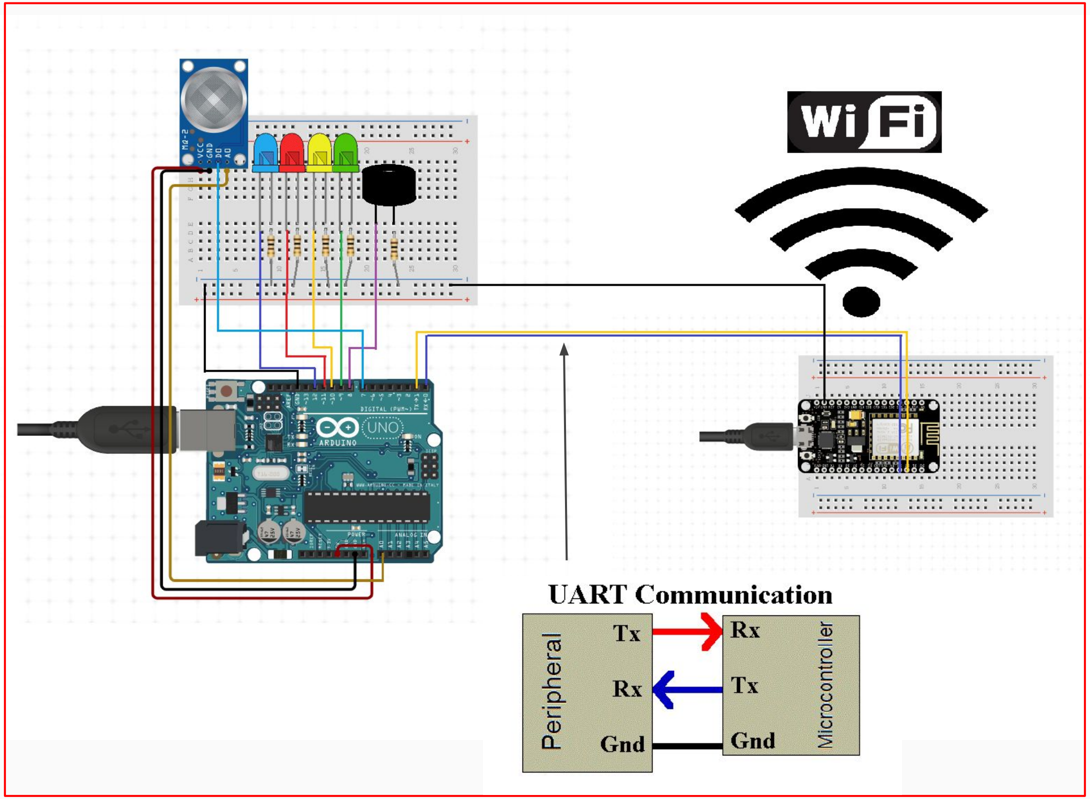
MySQL™
Developement Services

Sensor nodes: Arduino

Each node is composed by:

- 1 MQ2 sensor
- 1 Arduino UNO
- 1 NodeMCU

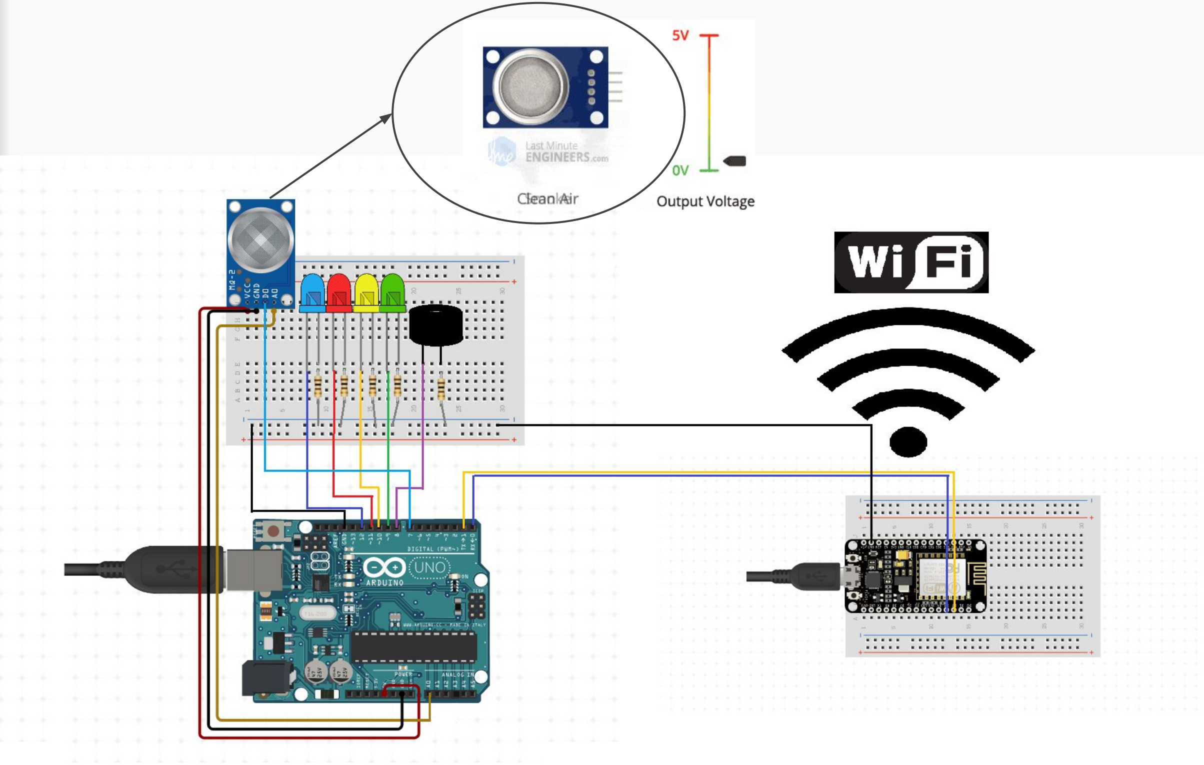
We've specified a unique identifier for every node to better recognize it in later steps (it fakes a factory number).



ID = 1

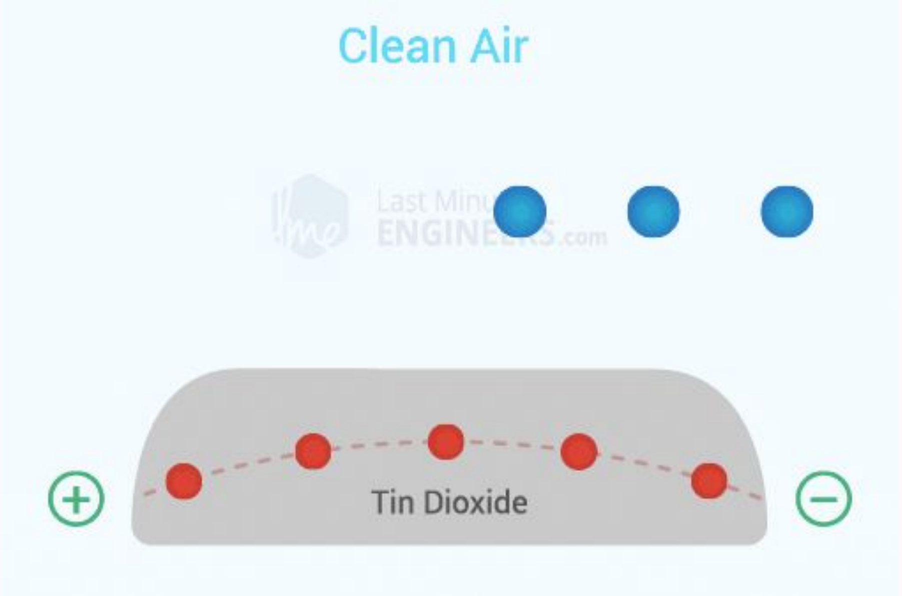
Sensor nodes: Arduino

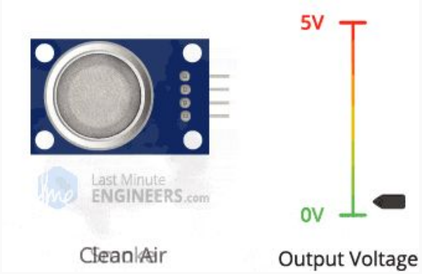
MQ2 communicates values in a 5V line, so we couldn’t attach it directly to the NodeMCU input Analog pin (and by the way, that is the only pin available on the Node)



MQ-2 sensor in-depth

The behavior of the sensor depends on the quantity of Oxygen on its surface.

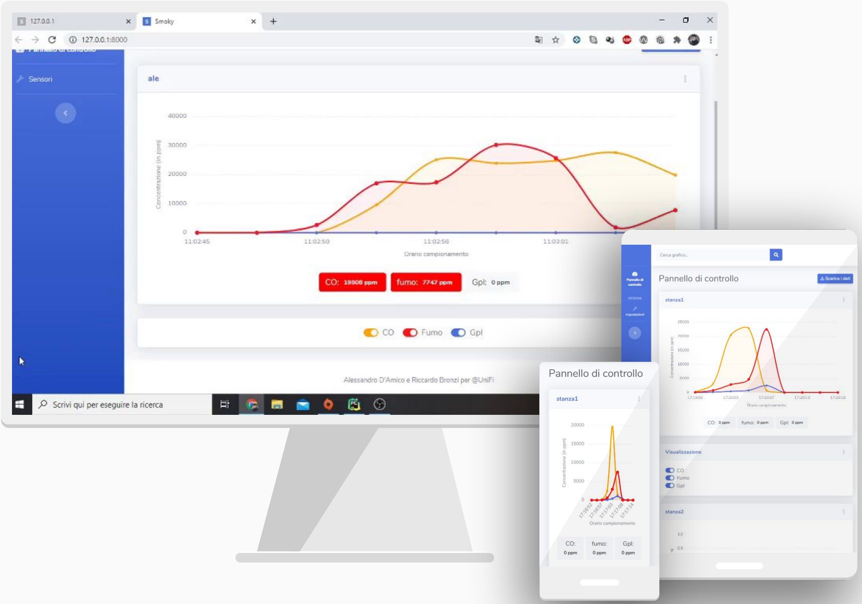




Django

Django is a Python Framework with many features: we’ve actually used it mainly in addition of Bootstrap and coupled with apache server (with WSGI NGIX) to test the GUI inside our LAN.

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components, to achieve the best user experience.

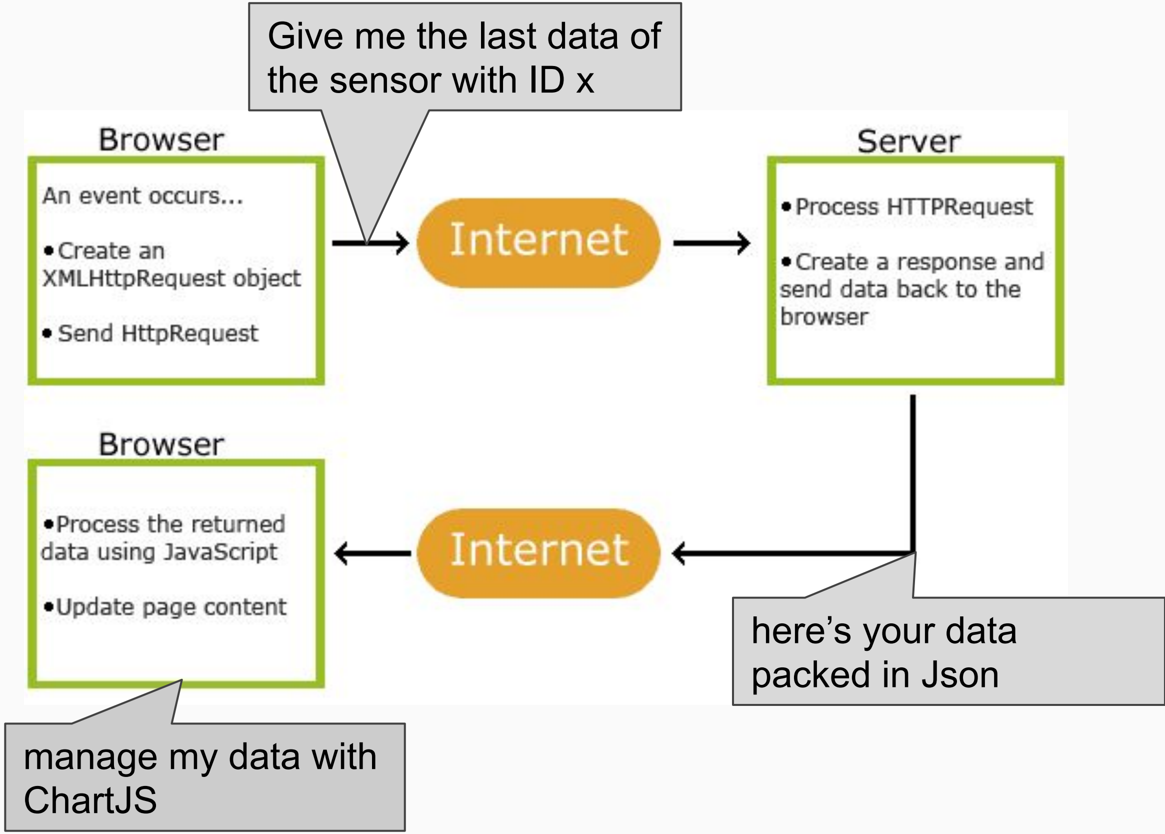


django

Bootstrap

Javascript with Ajax

Our code perform asynchronous fetches through the Ajax functions, using GET and POST requests to the smokysmokysmoky.com server and obtain a JSON response.


Browser
An event occurs...
•Create an XMLHttpRequest object
•Send HttpRequest
Internet
Server
•Process HTTPRequest
•Create a response and send data back to the browser
Internet
Browser
•Process the returned data using JavaScript
•Update page content

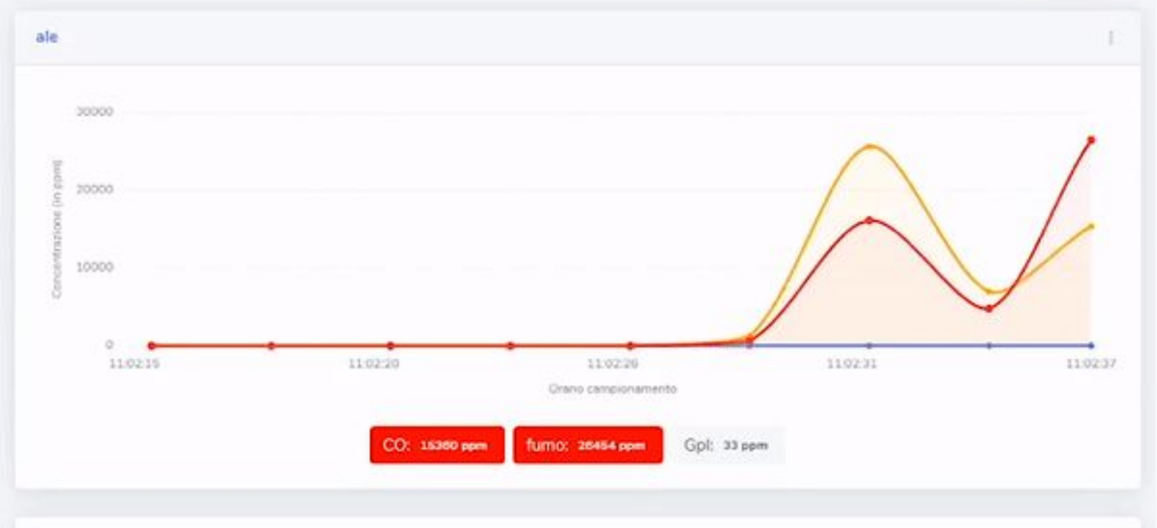
Give me the last data of the sensor with ID x

here’s your data packed in Json

manage my data with ChartJS

Live Graphs

Our graphs are realized with Chart.js, a useful library to create graphs. every data chart is represented as an object with its own datasets. Datasets are stored as arrays.

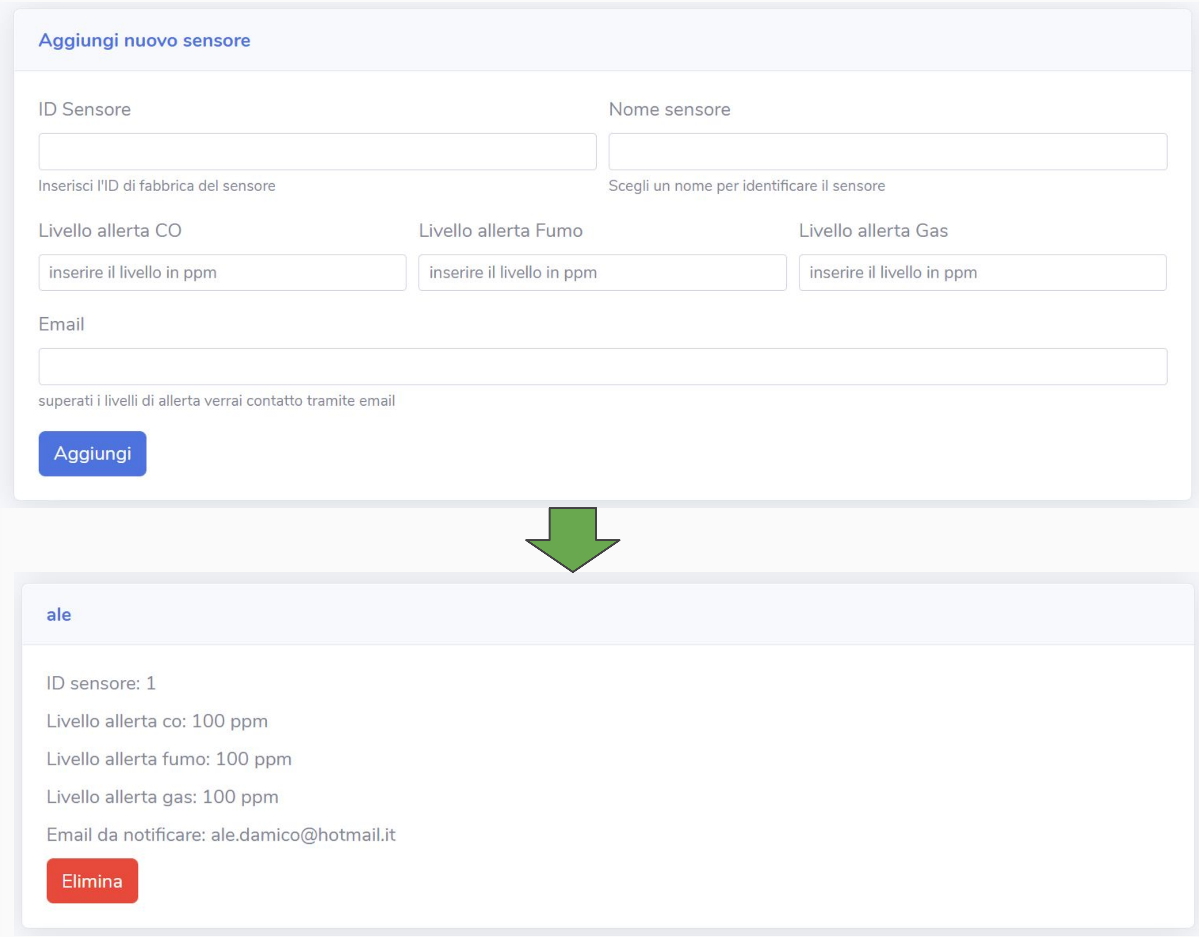


LIVE

Chart.js

User defined configs

The user just has to create a simple configuration, guided through a simple form in which is needed a sensor pseudonym, the chosen alert levels and the e-mail to be notified.



Aggiungi nuovo sensore

ID Sensore

Inserisci l'ID di fabbrica del sensore

Nome sensore

Scegli un nome per identificare il sensore

Livello allerta CO: inserire il livello in ppm

Livello allerta Fumo: inserire il livello in ppm

Livello allerta Gas: inserire il livello in ppm

Email

superati i livelli di allerta verrai contatto tramite email

ale

ID sensore: 1

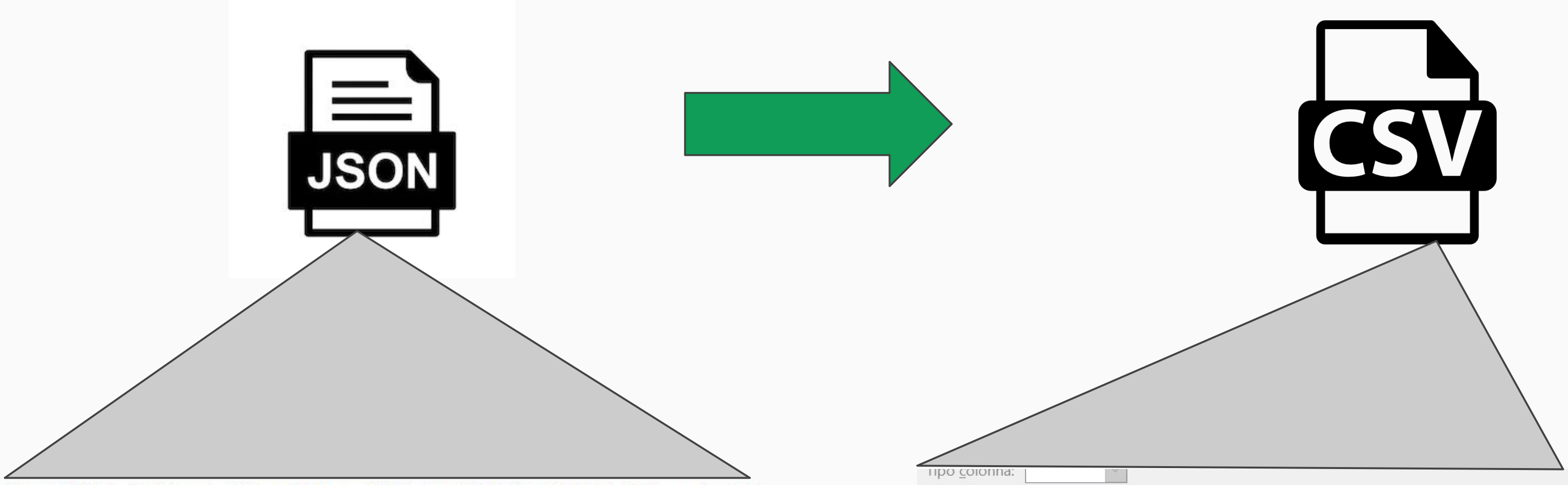
Livello allerta co: 100 ppm

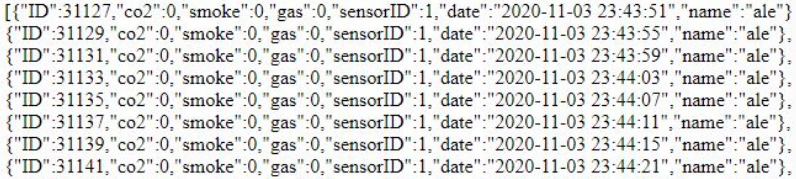
Livello allerta fumo: 100 ppm

Livello allerta gas: 100 ppm

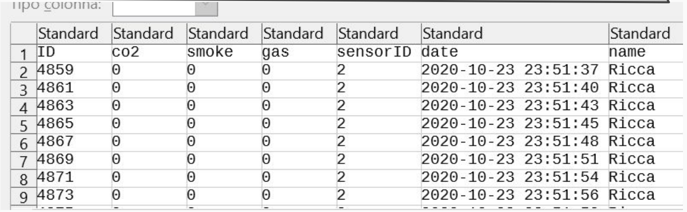
Email da notificare: ale.damico@hotmail.it

Data download (CSV)





```
[{"ID":31127,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:43:51","name":"ale"}
{"ID":31129,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:43:55","name":"ale"},
{"ID":31131,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:43:59","name":"ale"},
{"ID":31133,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:44:03","name":"ale"},
{"ID":31135,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:44:07","name":"ale"},
{"ID":31137,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:44:11","name":"ale"},
{"ID":31139,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:44:15","name":"ale"},
{"ID":31141,"co2":0,"smoke":0,"gas":0,"sensorID":1,"date":"2020-11-03 23:44:21","name":"ale"},
```



Tipo colonna:

	Standard	Standard	Standard	Standard	Standard	Standard	Standard
1	ID	co2	smoke	gas	sensorID	date	name
2	4859	0	0	0	2	2020-10-23 23:51:37	Ricca
3	4861	0	0	0	2	2020-10-23 23:51:40	Ricca
4	4863	0	0	0	2	2020-10-23 23:51:43	Ricca
5	4865	0	0	0	2	2020-10-23 23:51:45	Ricca
6	4867	0	0	0	2	2020-10-23 23:51:48	Ricca
7	4869	0	0	0	2	2020-10-23 23:51:51	Ricca
8	4871	0	0	0	2	2020-10-23 23:51:54	Ricca
9	4873	0	0	0	2	2020-10-23 23:51:56	Ricca

Smoky

Smoke sensor monitoring web app

Thank you for watching!

Stay safe!

Silicon Valley, Santa Clara

```
var still_alive = true;
while (still_alive) {
    WearMask();
    Stay6ftApart();
    WashHands();
    GetTested();
}
```