Urban AirQ / Making Sense - Pilot 1
Tech Responsible: Emma Pareschi
Developer: Dave Gonner
Final adaptation: Emma Pareschi

Technical Documentation Sensor Kit

Within the execution of Urban AirQ / Making Sensor project we developed a sensor kit that could answer to the citizen request during the air quality pilot study.
The structure of this document is as following:

Background

Many of the technical choices we made for the development of the kit are based on the ASCL project: http://www.hindawi.com/journals/js/2016/5656245/.
Starting from that experience and the hardware that has been used, we decided to use similar electronic parts but also to improve and integrate more functions.
From ASCL we kept:

- The main board, Arduino Uno. It's open-hardware board for fast-prototyping applications. Nowadays it represents the simplest approach to electronic and programming on the market. We opted for Arduino Uno with the idea that anyone should be capable to reproduce and adapt our kit for his own application.
- The NO2 sensor

Some of the technical aspects that we wanted to improve are:

- 9V battery supply for portable application and 5V USB power for stationary application.
- SD card data logger shield
- transparent acrylic case

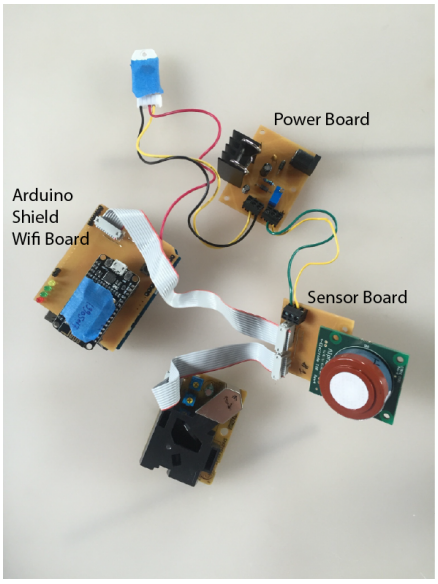
Technical Requests and Specs

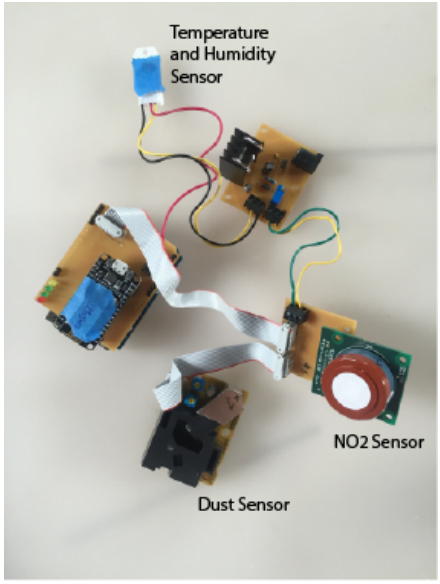
General requests:

- Modular solution: a kit that is assembled by smaller parts (boards). These parts should be easily plugged, swapped and modified.
- Wireless data collection: the goal is to have an independent solution, independent from physical support or memory limitation while the kit is running.
- Power supply: stable and fast power management.

Sensors and Boards

The Kit is so composed by five boards and four sensors.





Sensors

- NO2 → NO2-B42F and NO2-B43F, Alphasense
- Particulate Matter → PPD42NS, Shinyei
- Temperature and Humidity → DHT22, Aosong Electronics Co.

NO2 Sensor – NO2-b42F/ NO2-b43F

To get full sensor performance, low noise interface electronic is necessary. Alphasense offers indeed also a support circuit (ISB) that can be used to optimize the reading of low ppb levels and guarantee low noise environment. For time and resource matter, we decided to buy the sensors already soldered on their ISB (Individual Sensor Board).

Sensor Datasheet: http://www.alphasense.com/WEB1213/wp-content/uploads/2016/07/NO2-B43F.pdf
ISB Datasheet: http://www.alphasense.com/WEB1213/wp-content/uploads/2016/06/ISB.pdf
References: http://www.alphasense.com/index.php/products/nitrogen-dioxide-2/

Note: the pdf documents can also be found in the github repo.

Particulate Matter Sensor - PPD42NS

Sensor Datasheet: http://www.seeedstudio.com/wiki/images/4/4c/Grove_-_Dust_sensor.pdf
References: http://takingspace.org/wp-content/uploads/ShinyeiPPD42NS_Deconstruction_TracyAllen.pdf

Note: the pdf documents can also be found in the github repo.

Temperature and Humidity – DHT22

Sensor Datasheet: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf

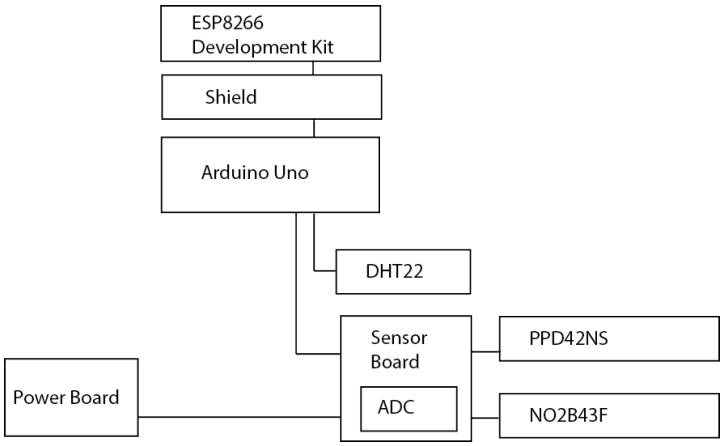
Note: the pdf documents can also be found in the github repo.

Boards

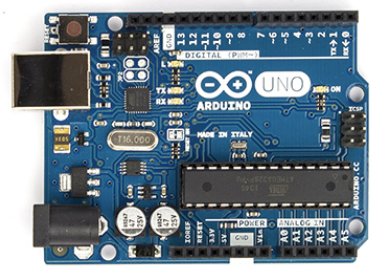
- Main board → Arduino Uno
- Wifi Module → ESP8266 development kit, NodeMCU 1.0
- Shield -> designed and produced in Fablab Amsterdam
- Sensor Board -> designed and produced in Fablab Amsterdam
- Power Board -> designed and produced in Fablab Amsterdam

Note: original design files can be found in the repo.

The image below shows the general block diagram of the kit.

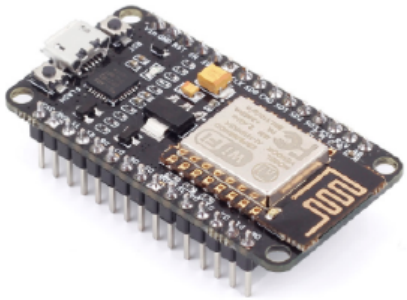


Main board



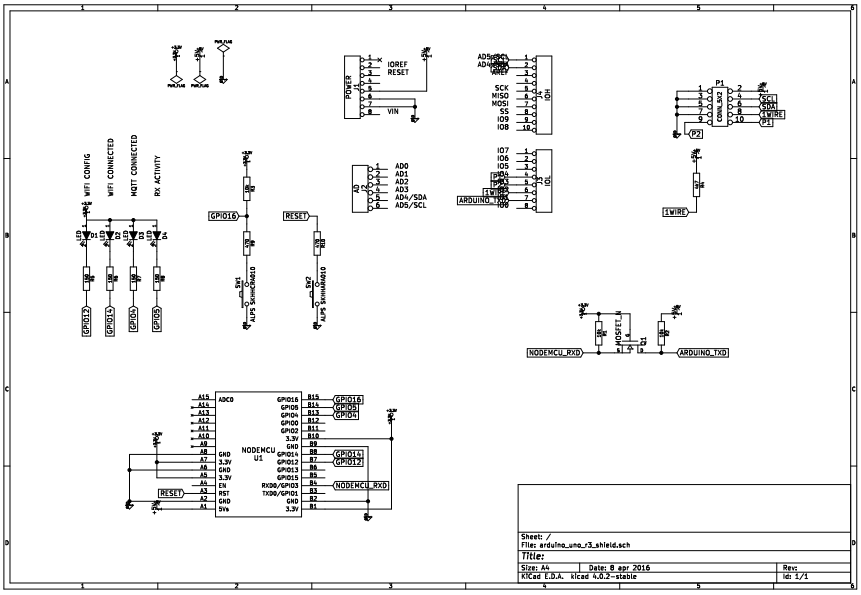
The main board manages the reading of the sensors and collects/sends the data to the Wifiboard.

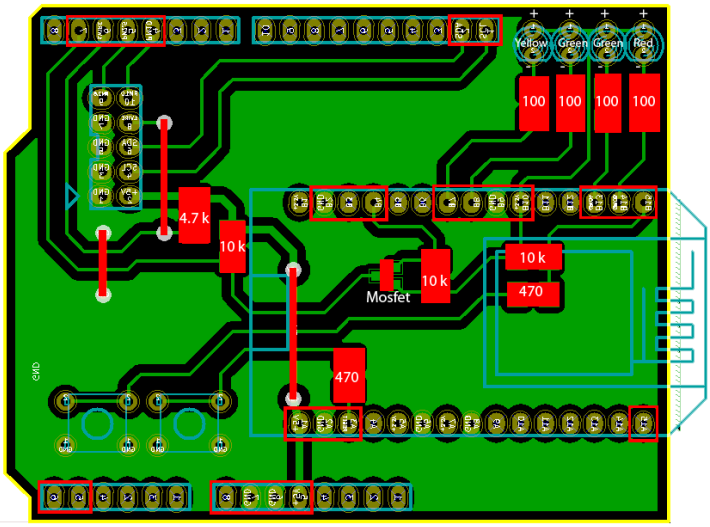
Wifi module

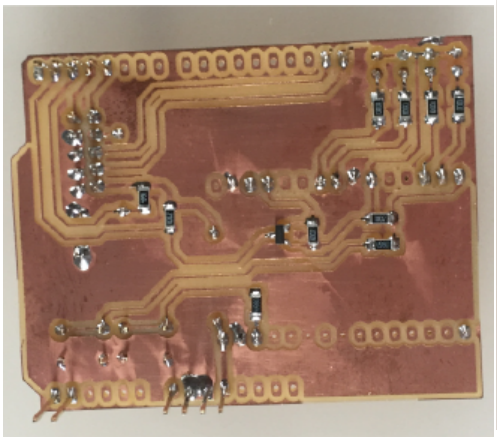


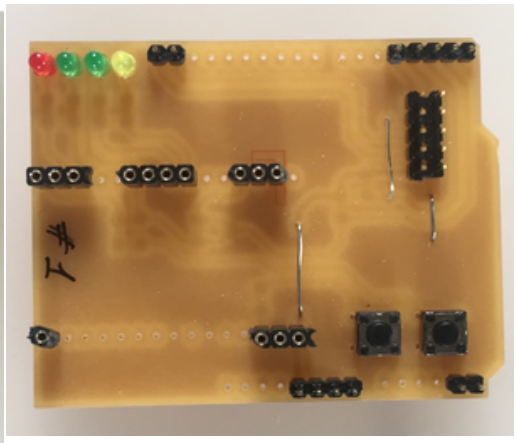
The wifi module is a NodeMCU development kit 1.0. The advantage to use the whole development kit instead of the single wifi module, is the possibility to program the ESP8266 chip without the need of extra hardware, as an FTDI cable or chip. To flash this module you only need a micro usb cable and of course the right setup in the software. In our application the module has the function to receive the data from the main board, prepare the payload for the server and send it every minute.

Shield


WIFI CONFIG
WIFI CONNECTED
MQTT CONNECTED
RX ACTIVITY
IOREF
RESET
VIN
NODEMCU
MOSFET
1WIRE
Sheet: /
File: arduino_uno_r3_shield.sch
Title:
Size: A4
Date: 8 apr 2016
KiCad E.D.A. kicad 4.0.2-stable
Rev:
Id: 1/1


Yellow
Green
Green
Red
100
100
100
100
4.7 k
10 k
10 k
Mosfet
10 k
470
470
GND





The shield has several functions:

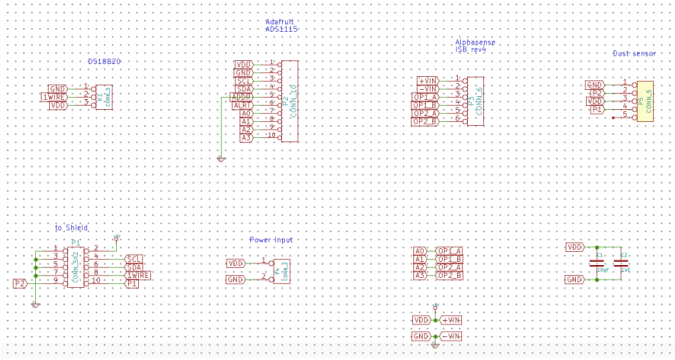
- It is the interface between Arduino UNO and the ESP8266 module. On the shield there is indeed the level shifter that allows the communication between the 5V and 3.3V domains, respectively of the main board and the esp8266.

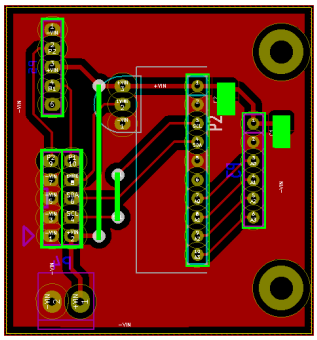
- It helps the functional debugging. There are indeed four led that helps to understand what is the operational mode and if there are some issues with the communication.
  - LED 1: yellow led. It indicates the configuration mode.
  - LED 2: green led. It indicates if the kit is connected to the wifi.
  - LED 3: green light. It indicate if the kit can reach the MQTT server.
  - LED 4: red light. It blinks when there a communication signal from the main board to the ESP8266.

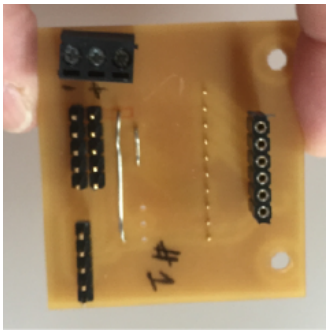
- It has two switches:
  - SW 1: it's used to go in configuration mode.
  - Sw 2: reset button

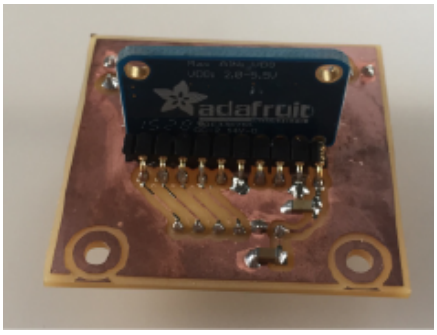
- It's the physical support of the power lines for the main board and the esp8622

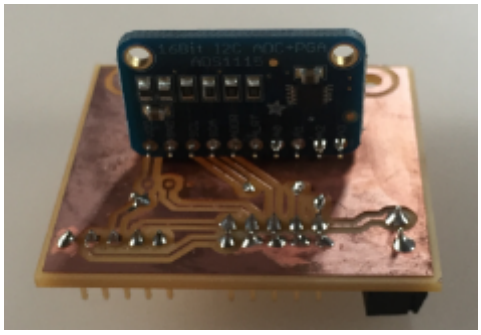
Sensor Board and ADC







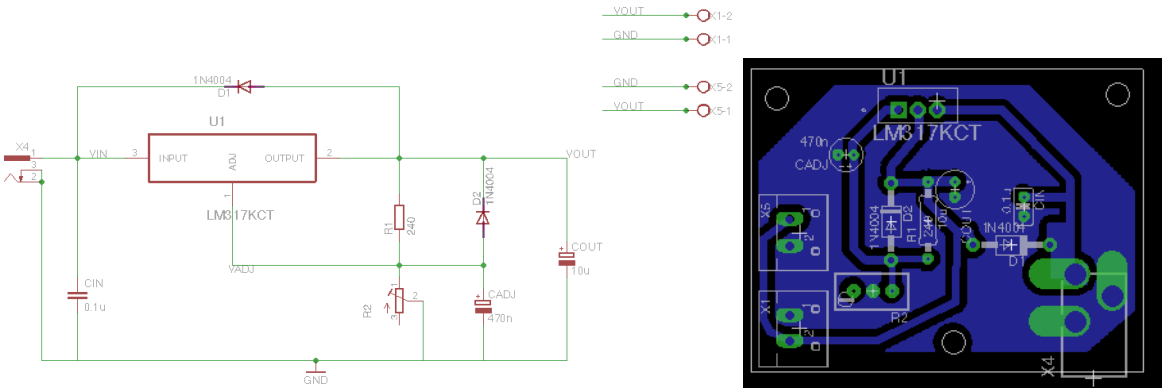


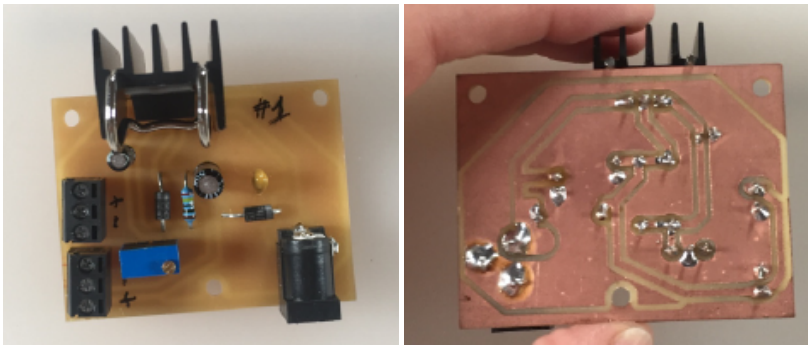


The sensor board is used to:

- to bring the power (5V) from the regulator to the main board.
- To host the ADC (Analog to Digital Converter) that is necessary to read the output of the NO2 sensor. See Code-Sensor section. The ADC is an ADS1115 4 channel (Adafruit Breakout board).
- To connect the NO2 sensor to the ADC.
- To connect the Dust Sensor to the main board.

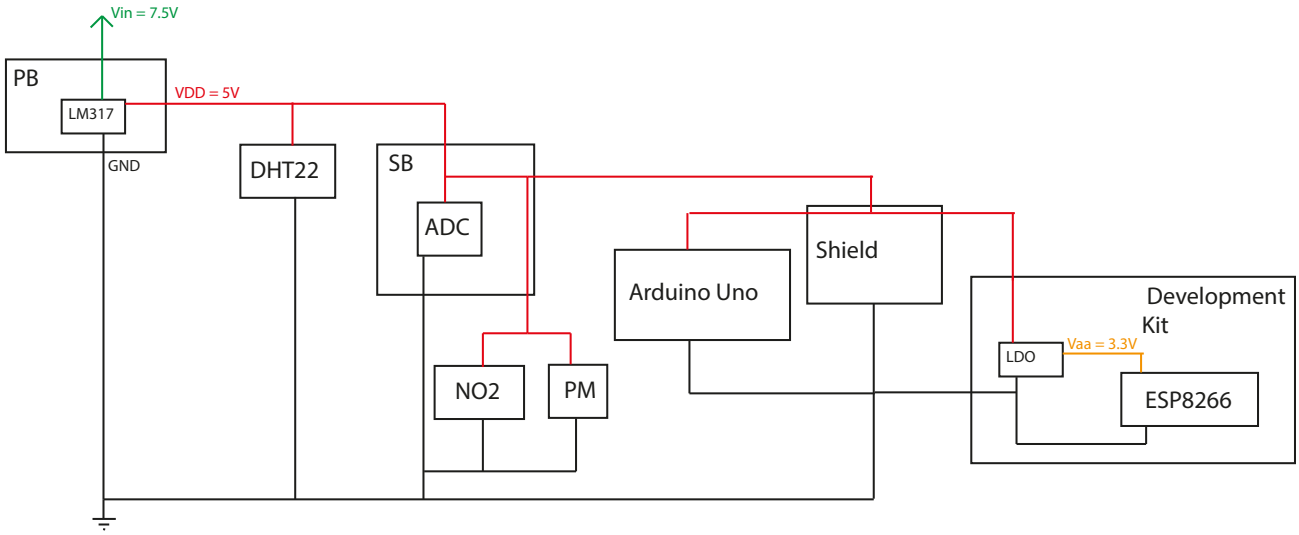
Power Board and Power Management





The Sensor Kit works at 5V that is generated on the power board by the LM317, adjustable three-terminal positive-voltage regulator. The output voltage is adjusted by the use of the potentiometer.

Power Management block diagram



The kit is powered at 7.5V with a universal wall adapter. The power board generates and regulates the 5V that supplies the kit through the sensor board. The dht22 is directly connected to the power board because the original design didn't include the temperature sensor. It has been a late request and the time didn't allow the its embedded implementation.

Communication Protocols

The image below shows the communication connections and protocols used between the different boards and devices.

IMAGES of protocols communication.

Starting from the sensors, the NO2 sensor is read by the ADC (ADS1115) that communicates with the Arduino Uno through I2C protocol. The Dust sensor is simply connected to two digital pins of the main board and it doesn't require standards to be used. The DHT22 includes both the temperature and humidity sensors and it can be used through single-bus support.
The Arduino Uno transmits the data to the ESP8266 using the integrated Serial Ports of the Arduino microcontroller. In this application the communication is one direction so only one wire is used (TX on the main board and TX on the development kit). Eventually the data are sent on air based on MQTT protocol.

Note: for library and code, see the Code section.

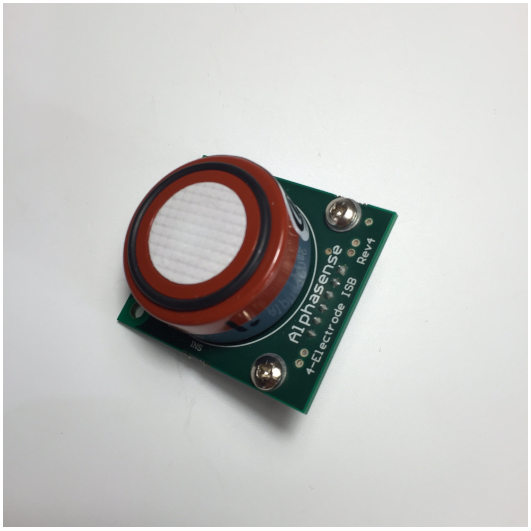
Code

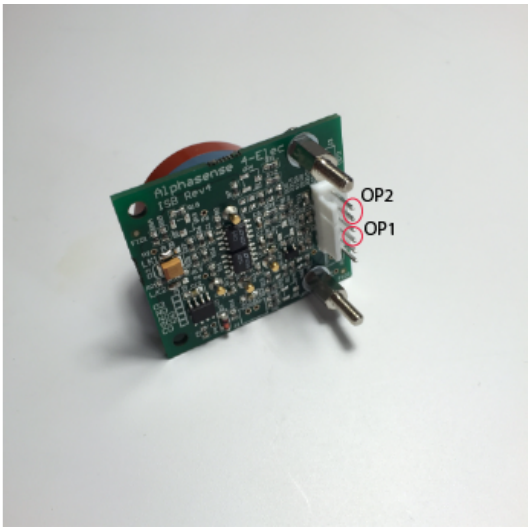
It's necessary to program both the boards, Arduino UNO and ESP8266.

Arduino UNO code

The code is written in Arduino I.D.E. 1.6.9.
In the sketch of the main board there are the commands to read the sensors and prepare the package of data for the ESP8266 module. The name of the sketch is datalogger.ino and it can be found in the github repo.

In the NO2 sensor a small amount of current is generated by a chemical reaction. The outputs of the NO2-B43F device are two and differential, it means that it's necessary to measure the difference between four pins in total to get the OP1 and OP2 data to then extrapolate the NO2 concentration. In the sketch it's not included the analog translation of the ADC output and the calculation of the gas concentration, those two steps are operated on the server side. Base on this, the main board executes two differential reading of the ADC.





To have a more reliable measure of the NO2, the Arduino UNO collects the data continuously for 30 seconds and it calculates the average over that time of both data, OP1 and OP2.

The code for the Dust sensor has been taken by a different open source project: Dustuino from PublicLab, https://publiclab.org/notes/Schroyer/11-23-2013/measure-coarse-and-fine-air-particulates-with-a-dustduino and the measurement is based on the following consideration: https://i.publiclab.org/system/images/photos/000/010/160/original/Size_Discrimination(PPD42NJ).pdf.
For the Bora Kit the cycle to measure the dust is 30 second long and it happens before the NO2 one.

The temperature and the Humidity are measured using the specific library for arduino environment.

The process is the following:
- ➔ 30 second: dust measurement
- ➔ 30 second: NO2 measurement and temperature/humidity

and every minute the main board send the data package to the wifi module.

The format of the payload is the following:
—ESP:15.16:65.30:1258:1226:124.95:466.30#

that in terms of meaning, the data are sent in the following order:
—ESP:temperature:humidity:op1:op2:pm10:pm2.5#

nodeMCU code

The code for the nodeMCU manages the Configuration Mode (see Configuration Mode section) and the transmission of the data to the server through WiFi.

The configuration mode is based on the “WiFi Manager” library for Arduino IDE and it handle the connection of your device to the network. Yes, you don’t need to save the credentials of you network in the code and upload it. Quoting the developer of the open source library, the WiFiManager‘s Features are:

- user friendly setup of esp8266 devices
- no need to hardcode credentials anymore
- simple and self contained, get started with a couple of copy/pastes
- captive portal so you get the configuration page as soon as you connect to the WiFi from mobile devices
- configuration portal is mobile visible/usable.

Source: https://tzapu.com/esp8266-wifi-connection-manager-library-arduino-ide/
To know how it works and how to connect your device to your network, see Configuration Set-Up section.

The transmission works based on Json protocol and the uses the PubSubClient.h library for the ESP to subscribe and publish on an MQTT server.
In this application the data is sent to a MQTT server with Mosquito as broker.
The payload has the following structure:

```
{"i":53788,"r":-79,"t":"15.60","a":"1258","b":"1226","p10":"124.95","p2.5":"466.30","h":"65.30"}
```

The single parts are the described in the below list.

i: chip id (unique identifier of the esp chip)
r: rssi
t: temperature
o1: op1 from no2
o2: op2 from no2
p10: pm10
p25: pm5
h: humidity

There is not a RTC (real time clock) device on the kit so the related time of every transmission is taken by the server.

How to upload the code
To save the codes in your boards you need to take three steps:

Step 1
Upload sketch datalogger.ino on the Arduino Uno

Step 2 - 1
In the code making-sense-esp8266.ino, uncomment the “//Format FS, reset Wifi settings, for testing” part, it should look like:

```
//-------------------------------------------
  //Format FS, reset Wifi settings, for testing
  Serial.print("Formatting FS...");
  SPIFFS.format();
  Serial.println("Done.");
  Serial.print("Reset WiFi settings...");
   wifiManager.resetSettings();  //****
  Serial.println("Done.");
  while(1) {
    delay(1000);
    Serial.println("loop...");
  }
  //-------------------------------------------
```

Step 2 - 2
Upload the modified making-sense-esp8266.ino sketch on you nodeMCU, in the ARduino IDE you need to select the board: NodeMCU (ESP-12E).

Step 3 - 1
In the code making-sense-esp8266.ino, comment the “//Format FS, reset Wifi settings, for testing” part, it should look like:

```
//-------------------------------------------
  //Format FS, reset Wifi settings, for testing
//  Serial.print("Formatting FS...");
//  SPIFFS.format();
//  Serial.println("Done.");
//  Serial.print("Reset WiFi settings...");
//   wifiManager.resetSettings();  //****
//  Serial.println("Done.");
//  while(1) {
//    delay(1000);
//    Serial.println("loop...");
//  }
  //-------------------------------------------
```

Step 3 - 2
Upload the code from Arduino IDE on your nodeMCU board.

Configuration Set-Up

The process to connect the sensor kit to the network is described in the following steps.

Step 1

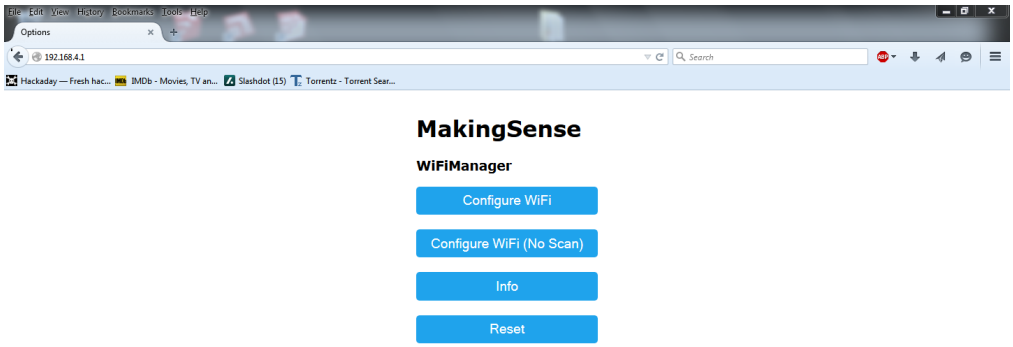
Plug the kit with the wall adapter. If the kit doesn't recognize any network it will go automatically in Configuration Mode and the yellow light (L1) will turn on.

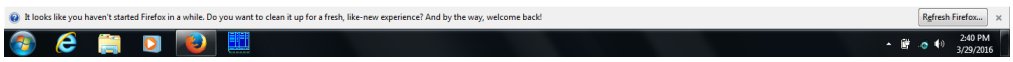
Step 2

Open your computer and look in the available wifi list, you will find a network called Making Sense. Select this network and connect to it.

Step 3

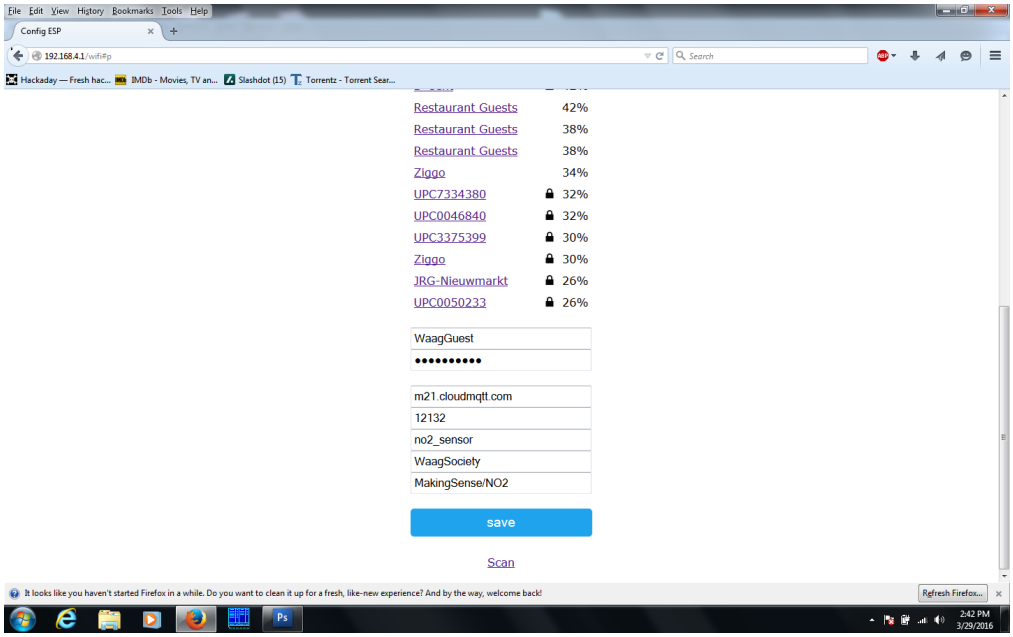
Once you are connected to Making Sense, open a browser and go to the address: 192.168.4.1, the WiFi Manager page will show up.



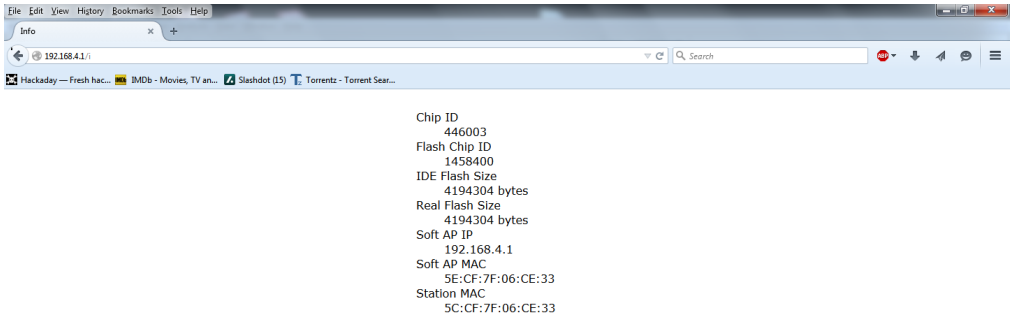


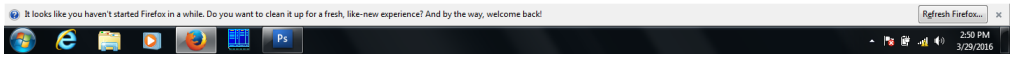
Step 4

Click on Configure WiFi and you will then see the list of available WiFi. Select the network you want to connect to and in the dedicated window digit the password. If everything was correct the ESP will now be connected to the network and the two green LED on the kit will turn off.



Some more information can be found from the WiFi Manager page.
Clicking on Info, you will be able to see some characteristic and ID of the chip.





It's also possible to force the device to enter in Configuration Mode.
How to enter the Configuration mode:

- Hold down switch 1
- Press once switch 2
- Hold down switch 1 till the yellow light turn off and the two green leds turn on.

Troubleshooting

If during the Configuration Mode:

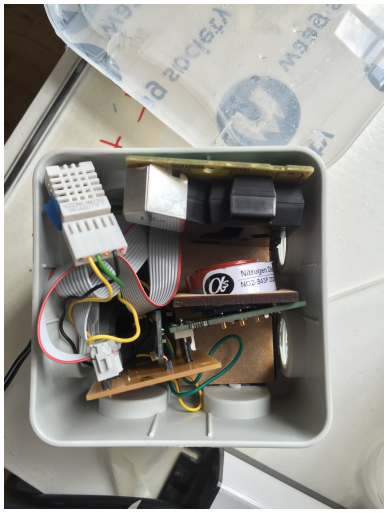
- you don't see Making Sense network: re-flash you nodeMCU
- you can't connect to Making Sense network, reset the device and try again.
  To reset the device you just need to press once the switch 2 on the board.

If after your device was properly working and suddenly it stops:

- turn off/on the device
- force the configuration mode, see Configuration Set-up section and repeat the connection process.
- check the green lights on the shield and:
  - if the LED 2 is off: repeat the previous step
  - if the LED 3 is off, contact the MQTT server provider

Case

The boards need to be enclosed in a Dutch weather proof case, it means that need to survive under stressful conditions of strong wind and rain.
The adopted solution was the use of universal stackable cable box from a Dutch-Hardware store chain.



The kit is developed to operate in outdoor conditions, during the pilot study we fixes the kit outside windows, balconies and gardens. Most of the time it was fixed with duck tape…the case of the kit is in the improvement list for the next version (if there will be a second version).
The kit need a good internet connection, in some case it was necessary to use a repeater to increase the quality of the wifi near by the sensor kit.