- Report IOT labs -

In this short report you will see our work during the Iot labs and our special project called "the connected plant". We are a group of four students Dalil, Caroline, Sasha and Shad from the major Information System and Cloud Engineering. At first you have a short summary for the two Iot labs, then the details from our Iot project and our our distribution of tasks and involvement in this courses.

I/ Lab 1 Getting started with the ESP32

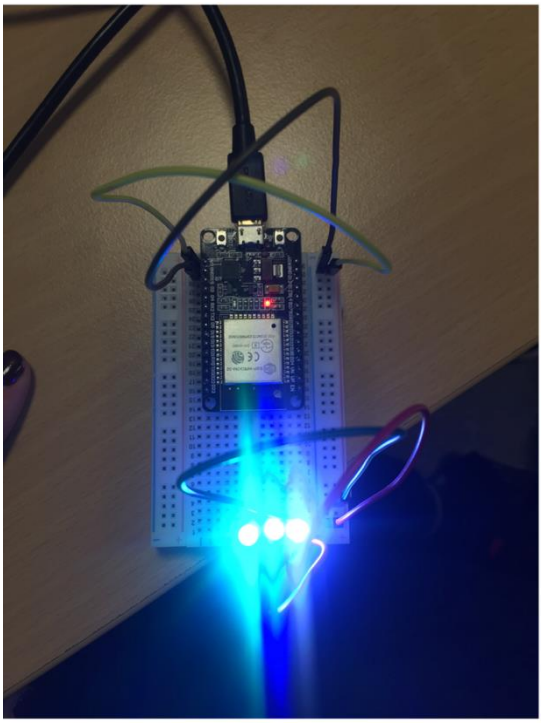
This lab was an introduction to familiarize ourselves with all the aspects related with the ESP32.We initially install the development environment and see how to inject small programs (for example the blink program) that we can flash into the development board in order to run them.
It took us two sessions of TP (8hours) before taking in hand the kit of development because we met small problems with macOS and installation of drivers.
You can see below our terminal, and our struggle to manage to make the ESP32 work on Mac with a lot of errors. Hopefully, we managed to get it working with good progress of the different stages from the lab1 to communicate with the development board (ESP32 card).

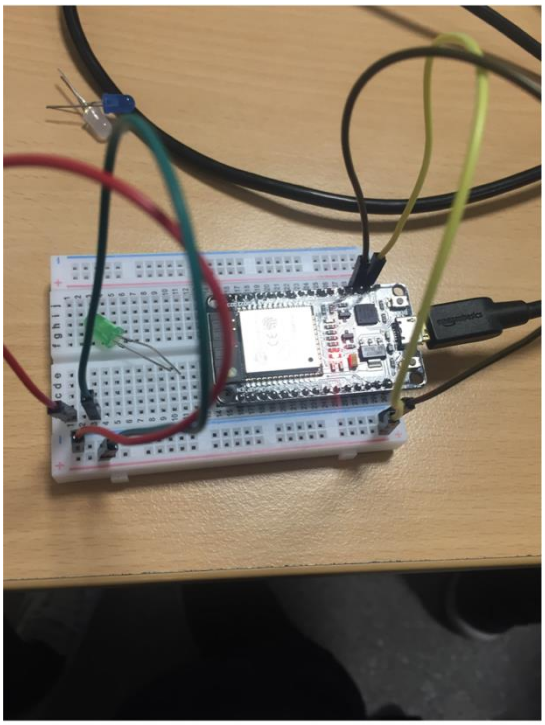
```
Last login: Wed Feb 14 16:49:27 on ttys001
➜  ~  cd ~/esp/myapp
➜  myapp master ✗ export IDF_PATH=~/esp/esp-idf
➜  myapp master ✗ make menuconfig
/bin/sh: xtensa-esp32-elf-gcc: command not found
/bin/sh: xtensa-esp32-elf-gcc: command not found
WARNING: Failed to find Xtensa toolchain, may need to alter PATH or set one in the
figuration menu
MENUCONFIG


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

➜  myapp master ✗ sudo cd /Users/carolinechiale/esp/esp-idf/examples/get-started
Password:
➜  myapp master ✗ make flash
/bin/sh: xtensa-esp32-elf-gcc: command not found
/bin/sh: xtensa-esp32-elf-gcc: command not found
WARNING: Failed to find Xtensa toolchain, may need to alter PATH or set one in the
figuration menu
/bin/sh: xtensa-esp32-elf-gcc: command not found
/bin/sh: xtensa-esp32-elf-gcc: command not found
WARNING: Failed to find Xtensa toolchain, may need to alter PATH or set one in the
figuration menu
CC build/bootloader/bootloader_support/src/bootloader_clock.o
make[2]: xtensa-esp32-elf-gcc: No such file or directory
make[2]: *** [src/bootloader_clock.o] Error 1
make[1]: *** [component-bootloader_support-build] Error 2
make: *** [/Users/carolinechiale/esp/myapp/build/bootloader/bootloader.bin] Error 2
➜  myapp master ✗ make file
/bin/sh: xtensa-esp32-elf-gcc: command not found
/bin/sh: xtensa-esp32-elf-gcc: command not found
WARNING: Failed to find Xtensa toolchain, may need to alter PATH or set one in the
figuration menu
make: *** No rule to make target `file'.  Stop.
➜  myapp master ✗ make menuconfig
/bin/sh: xtensa-esp32-elf-gcc: command not found
/bin/sh: xtensa-esp32-elf-gcc: command not found
```

As a result of this lab, we managed to present our connections and installation to make a led flicker:





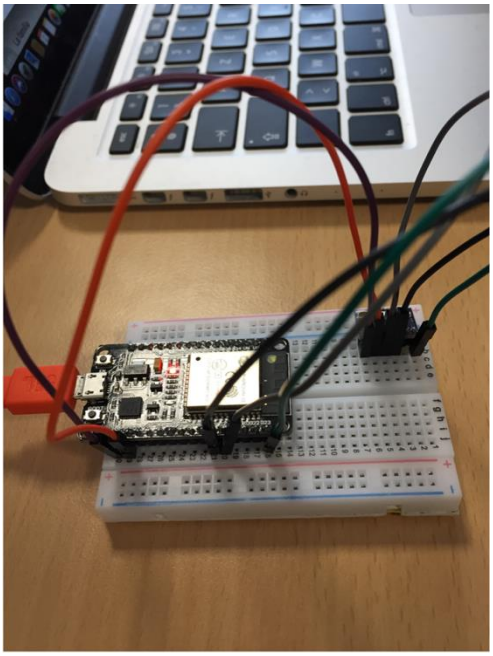
II/ Lab 2 General Purpose Input/Output (GPIO)

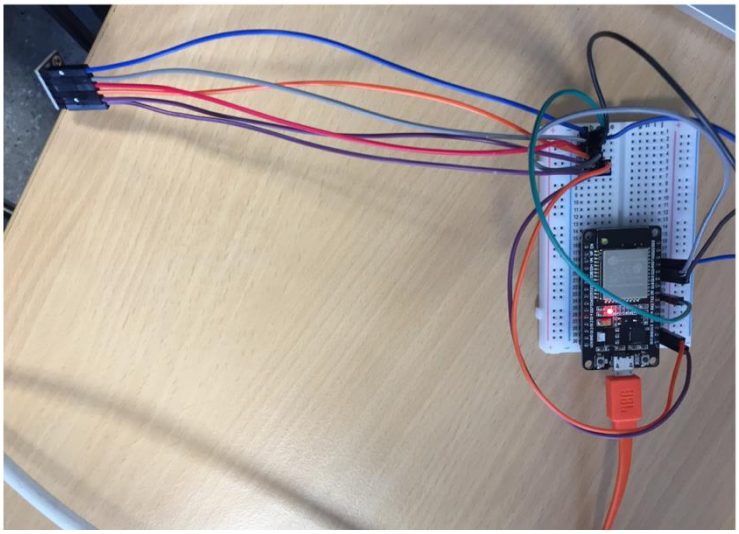
The goal of this second lab was to get you familiarized with the APIs exposed by the ESP-IDF framework.
Firstly, we discover the pinout of the development board and program with the GPIOs API.
After we discover by ourself tard the help of the professor the Serial Peripheral Interface (SPI) protocol used for transmission. Finally the main goal for us was to connect the BME280 sensor to the ESP32 and get measures from it because we dit all the steps until the 2.4 exercices.

The other goal of the lab was to understand the context of reading documentation from different sources in order to get devices communicate with each other but we didn't see how to communicate in details and didn't do the 2.5 exercices.

We encountered several problems with our machines at the beginning of the labs, but later with the help of the responsible teacher, we gradually got there at our own pace to perform the exercises. We have learned a lot with the lab 2 but it was not easy for us to understand all the steps and it was hard to hang.

Below you can see our connections with the BME280 sensor:





- IoT Project: the connected plant -

Introduction:

For this project, we created a system to take care of different kind of plants via the transmission of humidity data coming from the ground.

Thanks to the sensor, we can display vital data for the plant and act if the humidity modification is too high or too low.

We had thought about weather predictions but because of the lack of time and resources we could not follow up on this aspect of the project

• Hardware & Software:

The vital parameters for the plant management are: humidity in the ground, humidity in the air, ambient temperature, and the quality of lightning. We decided to work on the 3 first parameters.

These are the hardware components we decided to use to complete our objective:

- DHT11 sensor: This sensor will retrieve the humidity from the ground and from the air. Sadly, this sensor will not be present in the demo video since it broke during the project realization.

- Ground humidity sensor: This sensor is in U-shape that goes underground et that is capable of calculating the ground humidity depending on the resistance that is present between its branches. Indeed, the more the ground is conductive, the more the humidity level is high.

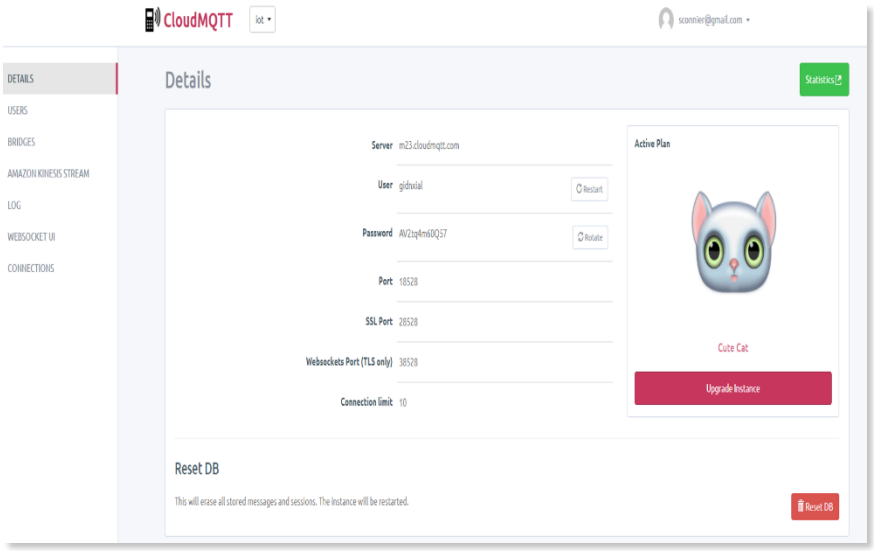
- Arduino Uno: Responsible of the sensors and this software has a good margin of evolution

- ESP32: The ESP32 is used as a WIFI spot since Arduino Uno doesn't have one

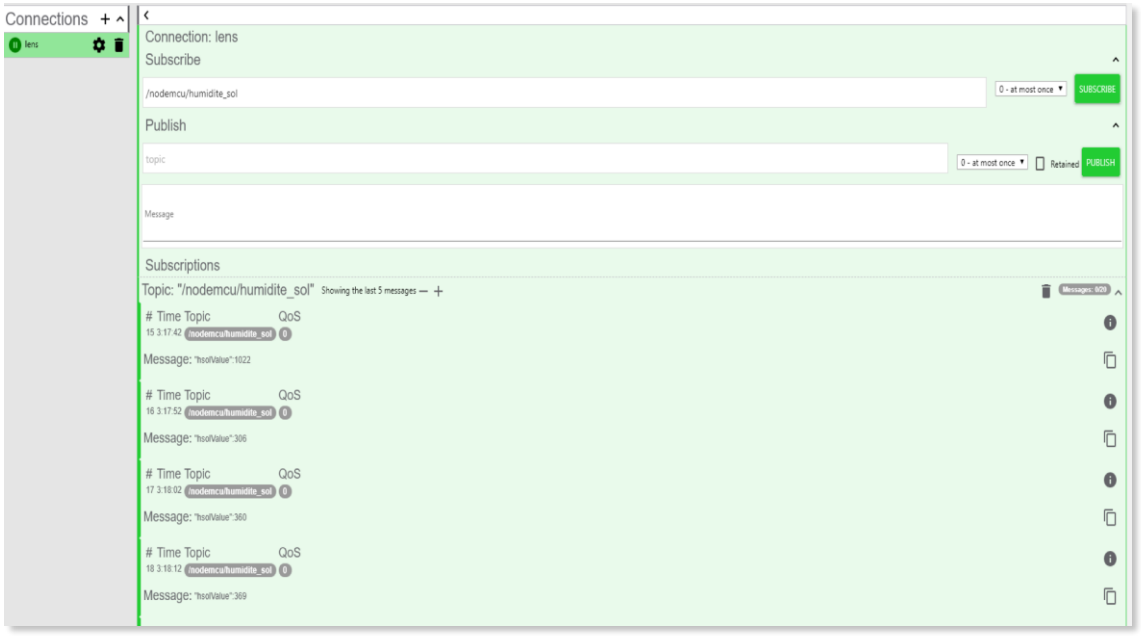
- Level shifter: The level Shifter will allow the transition of the information between Arduino and the ESP32 since one speaks in 5V and the other speaks in 3,3V.

- How does it work?

When the sensors retrieved the information, it is transiting to the ESP32 which will send it to the MQTT server.



As shown above, we used for this project the CloudMQTT service which offers a server based on Mosquitto that allows the easy creation of topics. We added this technology to a topic reader named MQTT Lens (on Windows) but it is interesting to note that this technology is available on other platforms such as Linux (MQTT-Spy) or Android (MQTT Dashboard). By doing so, we are sure that we can retrieve the information from different platforms and from anywhere.



Cards coding:

1) Arduino Uno:

```
//Librairie qui permet à la arduino de communiqué en série sur les broches 1 et 2
#include <SoftwareSerial.h>
//Définie notre broche de communication, ici la broche 0 noté D3 sur la arduino
int PinAnalogiqueHumidite=0;

//notre variables d'humidité sol
int hsol;

//défini nos broches de communication RX et TX sur les broches D2 et D3
SoftwareSerial sw(2, 3);

void setup(){
    Serial.begin(9600);
    pinMode(PinAnalogiqueHumidite, INPUT);
    Serial.begin(115200);
    sw.begin(115200);


}

void loop() {
    //lit la valeur du capteur
    hsol = analogRead(PinAnalogiqueHumidite);

    //envoi la valeur du capteur
    sw.print("\"hsolValue\":");
    sw.print(hsol);//offset
    sw.println();

    //défini la frequence d'envoi en miliseconde, soit ici 10sec
    delay(10000);

}
```

2) ESP32 code:

We will detail here only the main functions that will be used by the ESP32:

Server and WIFI login definition:

```
//on définie les informations de connection wifi et serveur
const char* ssid = "wifiPTR";
const char* password = "mdpwifi.";
const char* mqtt_server = "m23.cloudmqtt.com";
#define mqtt_port 18528
#define MQTT_USER "nodemcu"
#define MQTT_PASSWORD "test"
```

Initialization of the WIFI connection:

```
void setup_wifi() {

    delay(10);
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
    //Permet d'initié la connection wifi
    WiFi.begin(ssid, password);
    //Attend que la connexion s'éffectue
    while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
    }
}
   sw.begin(115200);
```

We then connect to the MQTT Server:

```
void reconnect() {

  while (!client.connected()) {
    //défini notre nom sur le serveur
    String clientId = "ESP32";
    //permet d'inité la connectio au serveur MQTT
    if (client.connect(clientId.c_str(),MQTT_USER,MQTT_PASSWORD)) {
      Serial.println("connected");
      //Si on arrive à se connecté on envoi un helloword sur le topic test
      client.publish("/nodemcu/test", "hello world");
    } else {
      //On retry de ce connecté toutes les 5secondes
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

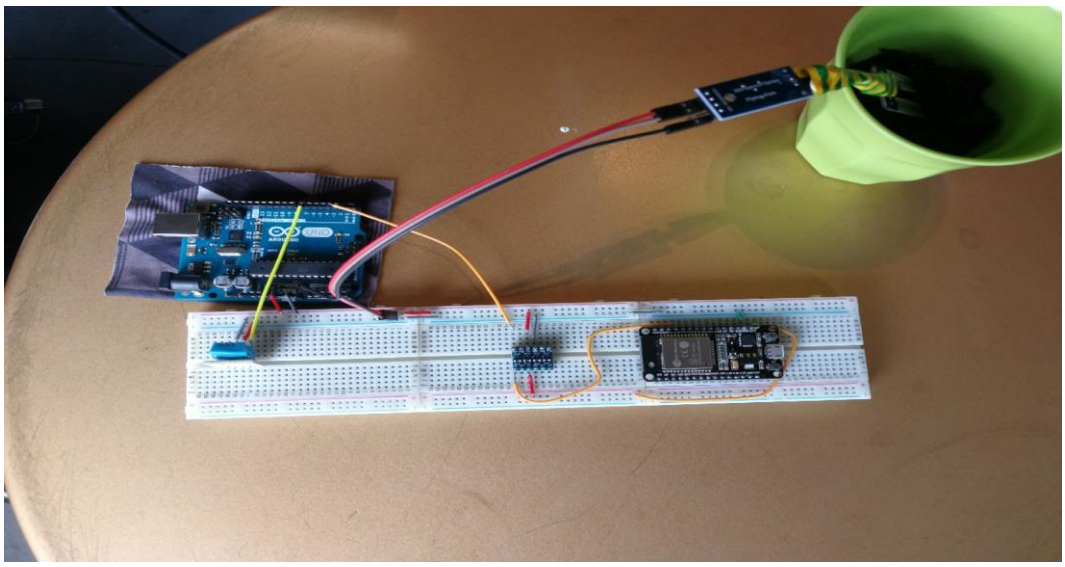
Then we publish the information to the topic:

```
void publishSerialData(char *serialData){//cette fonction publie les données capteur sur le topic humidite_sol
  if (!client.connected()) {
    reconnect();
  }
  client.publish("/nodemcu/humidite_sol", serialData);
}
```

Conclusion:

We managed to reach our objective that was stated before but there are a lot more upgrades that we could do in the future such as the automatization of the water spray.

In this project, we learned about the principle of MQTT servers, that are the nerve center of the IOT, as well as the programming of microcontrollers.



Feelings about the labs and the project:

Caroline: I was present at all labs and it took me a long time to understand what we were doing and why not use Arduino, which seemed more attractive to me for small projects like ours. I understood the usefulness of these labs, but I think that the organizations of these were poorly made. In these labs, I discovered a new environment and with the project now I am very interested about the IOT objects.

Dalil: I first didn't understand the point of this course to be honest. The first lab wasn't that hard, introducing the ESP32 and Arduino but afterwards, on the second lab it became way too difficult for someone who has no knowledge in this field. By coming at mostly all TPs (because I had an appointment at one of them on the day it was displayed), I realized that the environment of microcontrollers is very large. Participating into the server setup and the cable management made me increase my abilities into this field. That's why our team wanted to be innovative with the project of connected plants, and I think I got out with some values on the IoT fields.

Shad: I think it was an interesting subject, even if the course was a little difficult to apprehend if you never did something like this before. It was the case for many people in our class and mine too. It was difficult to comprehend at first, but the more you tried, the easier it was.I understand the need to have a subject like this, even if it is possible I will never have to do something like that again, because to understand how are gathered the information we are going to use is primordial to better work with them.

Sasha: In this project, I found again the joys of the microcontroller, the pleasure of cable management and I also was able to push further in this domain thanks to the setup of the MQTT server and the discover of the protocols related to it. Moreover, this is one of the rare opportunities we have to learn how to centralize data in context that is different from what we learned previously. This course can open us to new environments that we didn't even think about.