



Gestion d'une piscine automatisée

Rapport de projet 2021





Table des matières

Analyse du besoin / Etude du cahier des charges.....	3/14
Présentation du système à réaliser(problématique).....	3/5
Schéma synoptique du système.....	4/5
Les diagrammes sysmls.....	5/6
Diagramme de cas d'utilisation.....	5
Diagramme de séquence.....	6
Mise en situation/organisation.....	7/14
Présentation du matériel de notre système	7/10
Ressources logicielles.....	11
Notre organisation.....	11/12
Carnet de bord.....	11/12
Diagramme de Gantt.....	13
Partie individuelle.....	14/28
Présentation de ma partie.....	14
Matériel mis à disposition.....	15
Acquisition des capteurs.....	15/21
Fixation des seuils.....	21/23
Branchemet obtenu	23/26
Transmission à la base de donnée.....	27
Annexes.....	28
.....	



Présentation du système à réaliser :

Comment mettre en place un système automatisé de gestion d'une piscine pour répondre à un cahier des charges et des contraintes ?



La piscine d'Argenteuil souhaite moderniser son système de surveillance de la qualité de l'eau, d'afficher ces informations en temps réel aux usagers. Elle souhaite d'autre part mettre en place un système de surveillance de son local technique.

Dans ce projet notre travail consiste à automatiser une piscine à l'aide de différents capteurs et actionneurs, qui seront gérés par un local technique qui affichera les températures sur une Matrice led pour les baigneurs dont toutes les données sont envoyées à la base de données qui sont ensuite envoyées au site web du local technique.

Nous avons aussi une caméra usb 3.0 qui permet de détecter les intrusions des personnes non autorisées à rentrer au sein de la piscine. Pour cela en cas d'intrusion un mail sera envoyé au responsable de la piscine.

- Le client attend de nous les objectifs suivants :

- Pouvoir se connecter via un login et un mot de passe.
- Pouvoir ajouter ou supprimer des utilisateurs à sa guise.
- Visualiser les mesures des différents capteurs sur matrice led et site du local technique.
- Pouvoir recevoir une alerte en cas d'intrusion ou défaillance de pompe ou autre en envoyant un mail.



Les baigneurs visualiseront les températures des bassins de la piscine.
Les capteurs sont actifs et étalonnés, leur réponse est cohérente,
Les actionneurs sont opérationnels et mis en œuvre en cas de dépassement des seuils autorisés, et peuvent être actionnés via un clavier.
L'afficheur LCD présent dans le local technique donne les informations des différents paramètres.

Synoptique du système :

Dans le système suivant nous avons un capteur de niveau d'eau, permettant de savoir la quantité d'eau. En fonction de cette quantité la pompe rajoute ou reprend de l'eau, l'arduino envoie les données à la raspberry via protocole i2c. La caméra est reliée en usb, les capteurs y compris la caméra seront affichés sur le serveur web de la raspberry puis la température ambiante et la température de l'eau seront affichées sur la matrice led pour les baigneurs.





Présentation du système :

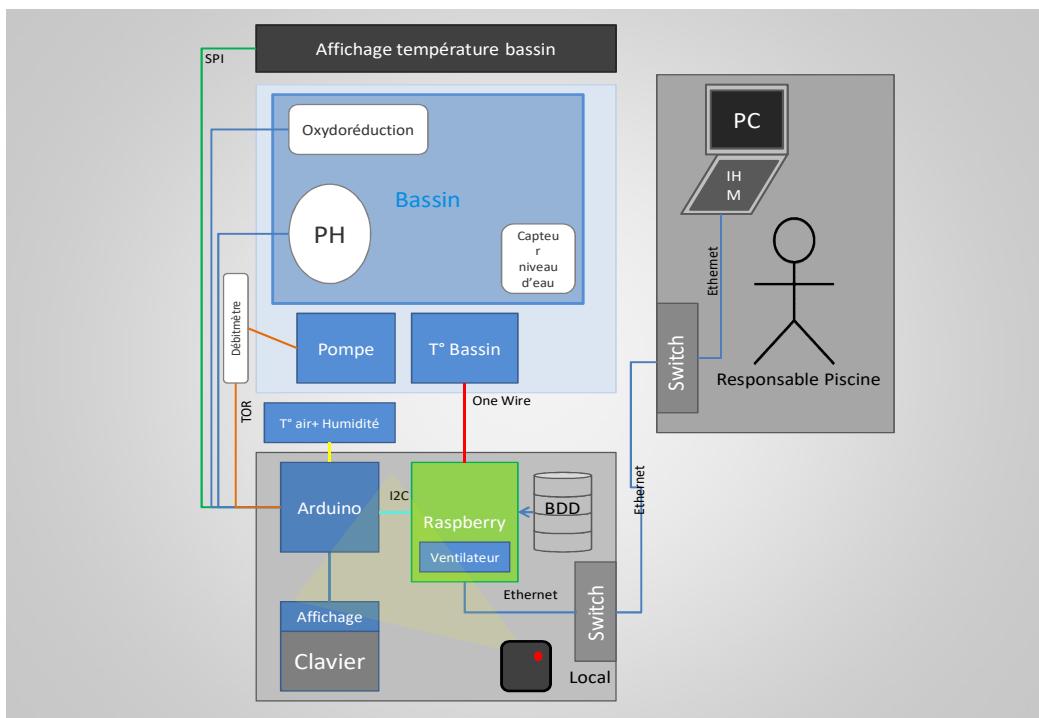


Diagramme de cas d'utilisation :

Dans ce diagramme on peut observer les différentes tâches que nous devons effectuer pour réaliser les besoins du client.

Dans un premier temps le Gestionnaire s'occupe de sélectionner le mode manuel, fixer les consignes, visualiser les données afficher localement, Alerter le responsable en cas d'intrusion.

Le Gestionnaire doit permettre aux baigneurs de voir les températures du bassin sur une Matrice led.

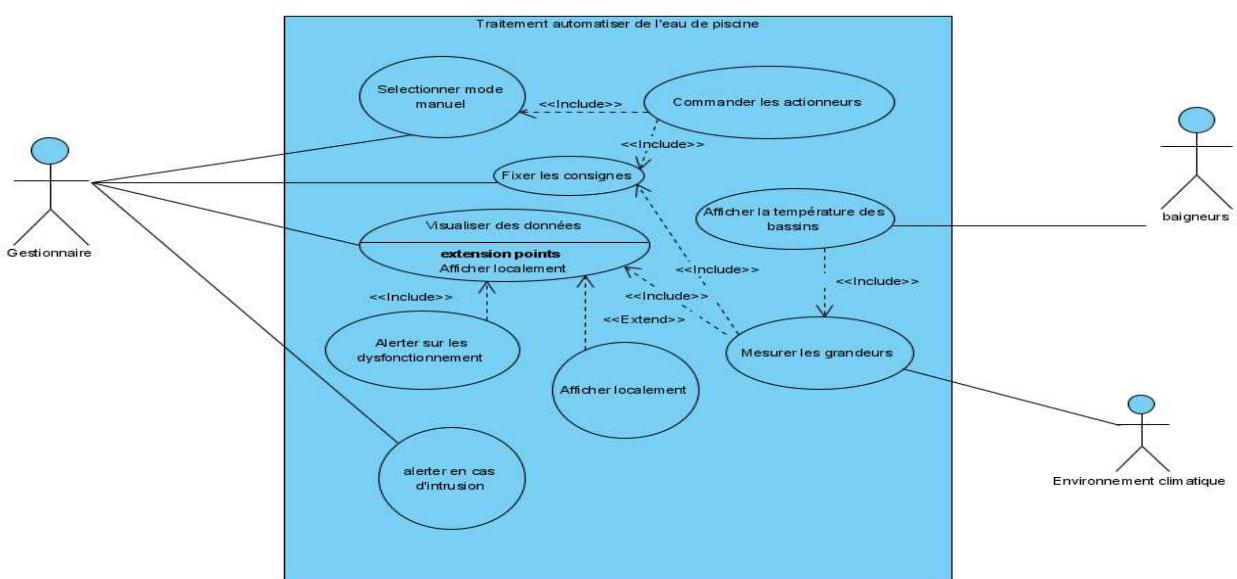


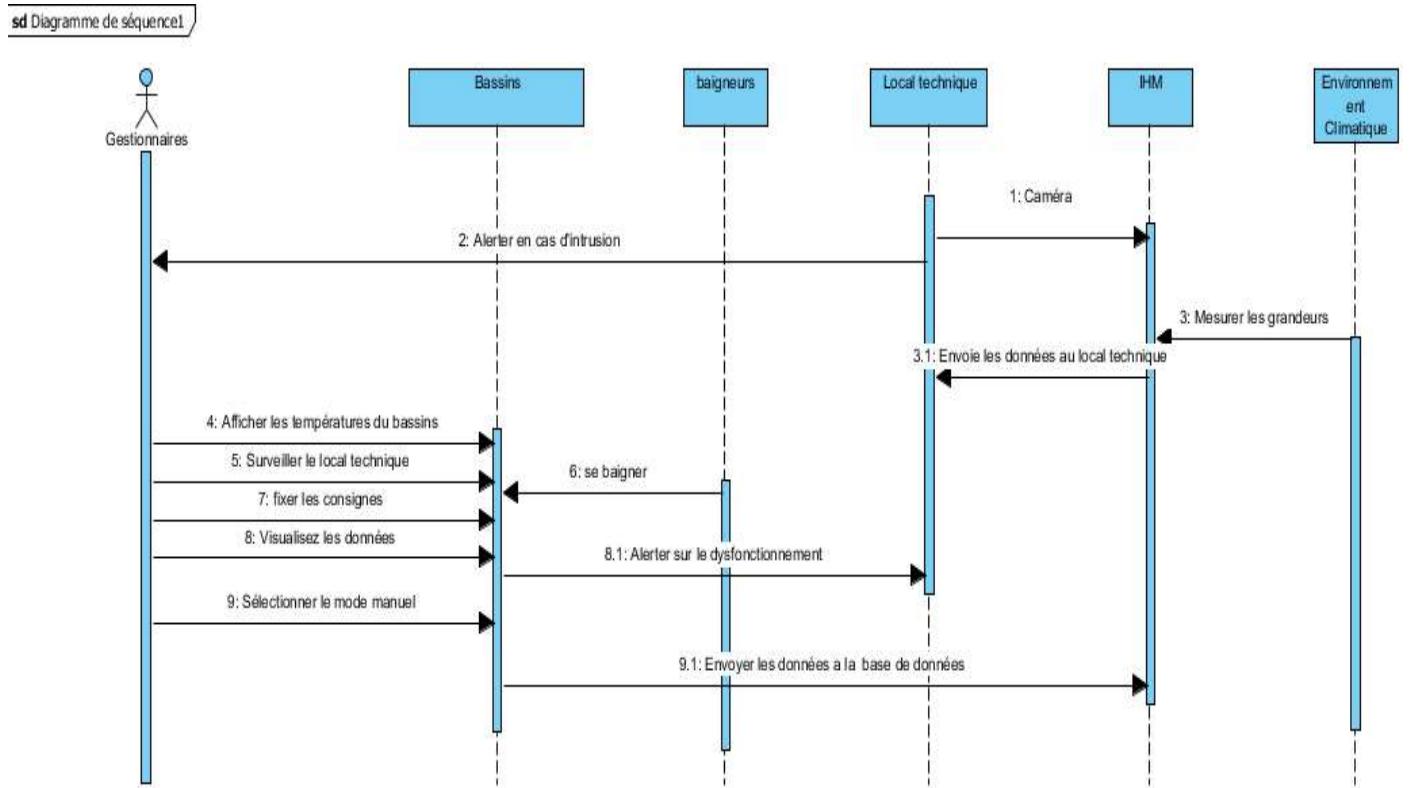


Diagramme de séquence :

Dans ce diagramme, on peut apercevoir différentes tâches que nous devons effectuer pour réaliser les besoins du client.

Dans un premier temps le gestionnaire doit sélectionner le mode manuel, visualiser les données, fixer les consignes, surveiller le local technique, afficher les températures du bassin.

Les différents capteurs envoient les données à la base de données de l'IHM, puis dans le local technique on gère les intrusions et en cas d'intrusions on envoie un mail automatiquement au responsable de la piscine.





Mise en situation/organisation :

Présentation du matériel :



Capteurs :

Mesurer le niveau d'eau de la piscine :



Capteur de niveau d'eau
Capteur de niveau
De liquide "eTape™"

Mesurer la quantité d'eau distribuée :



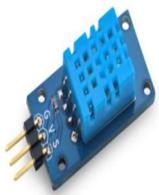
Débitmètre FCH-M

Mesurer Températures :



DS18B20 étanche

Températures des bassins :
sonde DS1820



Température air et Taux
d'humidité : DHT 11

Mesurer qualité eau :



Sonde Ph :Capteur pH
DrDAQ Pico DD011



Sonde Taux oxydoréduction :
RédoxGreisinger GR105-BNC-L01-CO

Surveiller local technique et capturer les images des intrus.

Caméra de surveillance pour intérieur



Caméra : Model Aver 340+USB
3.0



Actionneurs :

Assurer les

Pompes immergées 12V

fonctions pompe eau, chlore, neutralisant



Assurer la

Ventilateur PC

fonction pompe à chaleur



Raspberry Pi Modele 3

Informatique :





Commande de puissance des pompes



PiFace RELAY+ est une carte fille avec 4 sorties (des relais) et empilable HAT offrant de simples commandes marche / arrêt pour des charges jusqu'à 2ADC sur vos programmes Raspberry Pi. Entrées/Sorties analogiques

Carte ArduinoMega 250



La carte ArduinoMega 2560 est basée sur un ATmega2560 cadencé à 16 MHz. Elle dispose de 54 E/S dont 14 PWM, 16 analogiques et 4 UARTs



Carte Arduino Uno



Une carte Arduino est une petite (5,33 x 6,85 cm) carte électronique équipée d'un micro-contrôleur. Le micro-contrôleur permet, à partir d'événements détectés par des capteurs, de programmer et commander des actionneurs ; la carte Arduino est donc une interface programmable.



Interfaces :

Ecran LCD 4 lignes

Affichage en local des paramètres :
Affichage des températures des bassins :

Large AFFICHEUR A MatriceLED Velleman.
Matrice 32x16 leds blanches model : VMA419



Energie :

Alimentation Matrice LED :



Alimenter Matrice à Led
Alimentation RS25-55Vcc 25W
Intensité: 5 A
Précision: \pm 2%.
Régulation: \pm 0,5%.

Logiciels :

Logiciels	
Contrôle du GPIO, SPI, I2C, Liaison Série :	Librairie WiringPi : http://wiringpi.com/
Base de données	MySQL : https://www.mysql.com/fr/
Gestion base de données	PhpMyAdmin https://www.phpmyadmin.net/
Serveur Web	Apache2 : https://httpd.apache.org/
Camera IP/ Vidéosurveillance	Motion : https://raspbian-france.fr/video-surveillance-raspberry-pi-camera/
Arduino/capteurs	LibrarieArduino : https://www.arduino.cc/en/Reference/Libraries

Notre organisation :

Pour réaliser notre projet nous avons choisi une organisation qui consiste à se partager les tâches, chacun travail sur sa partie et vers la fin du projet, suivant la progression générale, nous mettons en commun toutes les parties et toutes les tâches que chacun à effectuer pour construire le projet final collectivement et efficacement :

Voici l'organisation :

Carnet de bord :



Légende

→ Rouge : Pas encore réaliser

→ Bleu : En cours de réalisation

→ Vert : Terminer

Etudiant 1 : Magnant Léo

Liste des fonctions assurées par l'étudiant 1

F1 : Acquérir les grandeurs physiques et chimiques de la piscine

F2 : Acquérir les températures des bassins et de l'air

F3 : Fixer les seuils des différents paramètres

F4 : Transmettre les données à la Raspberry (en collaboration avec l'étudiant 3).

Etudiant 2 : Hervé Baptiste

Liste des fonctions assurées par l'étudiant 2

F5 : Commander les actionneurs de façon automatique (traitement d'hygiène de la piscine)

F6 : Commander les actionneurs de façon manuelle (traitement l'hygiène de la piscine)

F7 :Réalisation de l'Affichage pour le local technique.

Etudiant 3 : Malki Adel

Liste des fonctions assurées par l'étudiant 3

F7 : Développer l'IHM « supervision »

F8 : Déetecter le niveau d'eau de la piscine

F9 : Distribuer ou pomper la quantité d'eau.



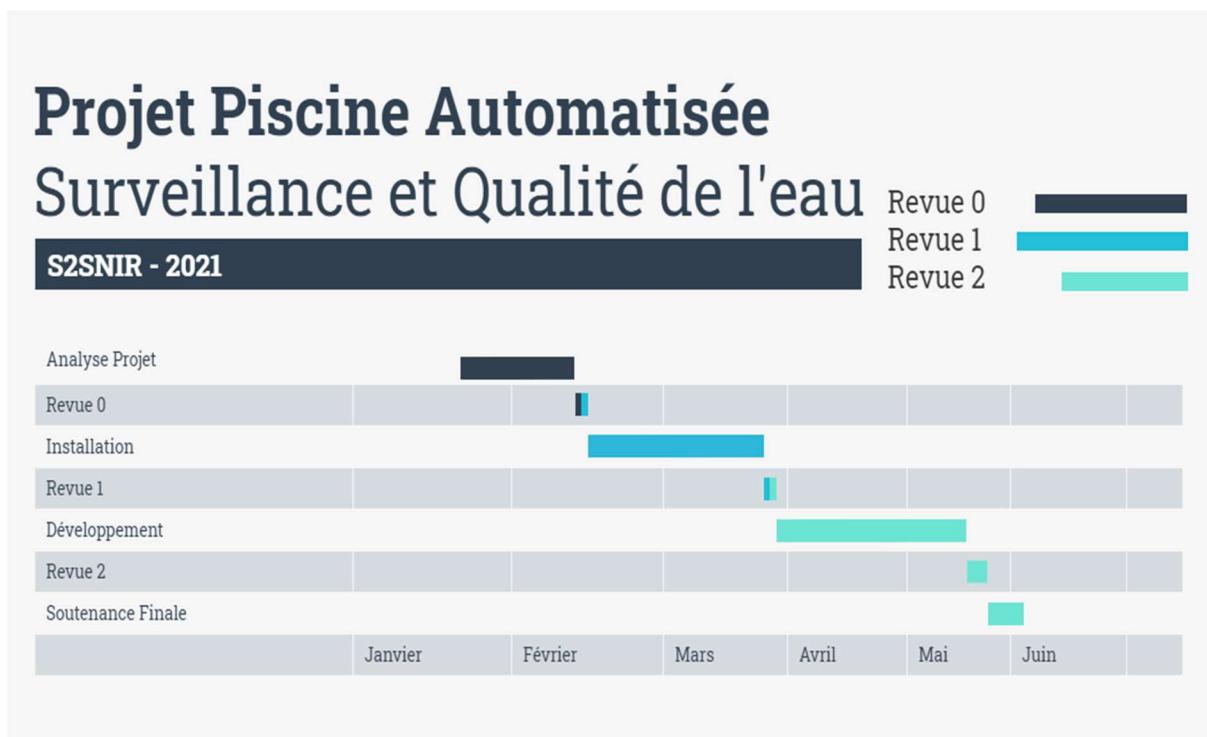
Etudiant 4 : Diallo Souleymane

Liste des fonctions assurées par l'étudiant 4

F8 : Surveiller le local technique

F9 : Afficher les données « Température » de la piscine pour les baigneurs.

Diagramme de Gantt :





Partie individuelle :

Présentation de ma partie :

Installation:

- Installation et configuration de l'ArduinoMega 2050
- Installation des librairies Arduino
- Installation des différents capteurs liés à l'hygiène de la piscine

Mise en œuvre:

- Relier l'ensemble des capteurs liés à la qualité de l'eau et aux températures à l'Arduino
- Relier les capteurs température eau et air à l'Arduino

Configuration:

- Etalonnage et validation des données fournies par les capteurs.
- Configurer les seuils des différents paramètres

Réalisation:

- Coder le module logiciel « captpisc » permettant l'acquisition des paramètres en provenance des capteurs |
- Coder le module logiciel « seuilparametres »
- Coder le module logiciel « transm » permettant de transmettre les données à la Raspberry pour insertion dans la base de donnée



Matériel mis à disposition :

Le matériel mis à disposition pour que je réalise ma partie est :



DS18B20 étanche

Températures des bassins :
sonde DS1820



Température air et Taux
d'humidité : DHT 22



Sonde Ph :Capteur pH
SEN0161



Sonde Taux
oxydoréduction ORP redox:



Carte



La carte ArduinoMega 2560 est basée sur un ATMega2560 cadencé à 16 MHz. Elle dispose de 54 E/S dont 14 PWM, 16 analogiques et 4 UARTs

ArduinoMega 250

Acquisition des capteurs :

Pour l'acquisition des capteurs, j'ai utilisé l'IDE Arduino et observé les résultats dans le moniteur série de celui-ci.

J'ai utilisé différentes librairies que j'ai inclus dans mes codes et commentés.

[Voici le code pour le capteur DHT 22 :](#)



```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#define DHTPIN 2 // broche ou l'on a branché le capteur
#define DHTTYPE DHT22 // DHT 22 (AM2302)
DHT dht(DHTPIN, DHTTYPE); //déclaration du capteur
void setup()
{
    Serial.begin(9600);
    Serial.println("DHT22 test!");
    dht.begin();
}
void loop()
{
    delay(2000);
    // La lecture du capteur prend 250ms
    // Les valeurs lues peuvent être vieilles de jusqu'à 2 secondes (le capteur est lent)
    float h = dht.readHumidity(); //on lit l'hygrométrie
    float t = dht.readTemperature(); //on lit la température en celsius (par défaut)
    // pour lire en fahrenheit, il faut le paramètre (isFahrenheit = true) :
    float f = dht.readTemperature(true);

    //On vérifie si la lecture a échoué, si oui on quitte la boucle pour recommencer
    if (isnan(h) || isnan(t) || isnan(f))
    {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }
    // Calcul de l'indice de température en Fahrenheit
    float hif = dht.computeHeatIndex(f, h);
    // Calcul de l'indice de température en Celsius
    float hic = dht.computeHeatIndex(t, h, false);
    //Affichages :
    Serial.print("Humidité: ");
    Serial.print(h);
    Serial.print(" %\t");
    Serial.print("Température: ");
    Serial.print(t);
    Serial.print(" °C\t");
    Serial.print(f);
    Serial.print(" °F\t");
    Serial.print("Indice de température: ");
    Serial.print(hic);
    Serial.print(" °C\t");
    Serial.print(hif);
    Serial.println(" °F");
}
```

[Ensuite le code pour la sonde DS1820 :](#)



```
#include <OneWire.h> //Librairie OneWire
#include <DallasTemperature.h> //Librairie du capteur

unsigned long previousMillis = 0;//variable delay sans arrêt du programme

const long interval = 1000;// variable delay sans arrêt du programme qui prévoit une lecture de 1 seconde par mesure

OneWire oneWire(2); //PIN digital de la sonde sur la pin 2 de l'arduino
DallasTemperature sensors(&oneWire); //Utilisation du bus OneWire pour les capteurs
DeviceAddress sensorDeviceAddress; //Vérifie la compatibilité des capteurs avec la librairie

void setup(void) {

    Serial.begin(115200); //Définition de la vitesse et de l'ouverture du port série
    sensors.begin(); //Sonde activée
    Serial.print("Etape 1");
    sensors.getAddress(sensorDeviceAddress, 0); //Adresse de la sonde à 0
    Serial.print("Etape 2");
    sensors.setResolution(sensorDeviceAddress, 12); //Résolutions
    Serial.print("Etape 3");

    delay (2000);
}

void loop(void) []

unsigned long currentMillis = millis();// inclure une variable de temps par rapport à millis()

if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;

    sensors.requestTemperatures(); //Demande la température aux capteurs
    Serial.print("Température de la pièce ");
    Serial.print(sensors.getTempCByIndex(0)); //Information récupérée sur l'adresse 0 de la sonde
    Serial.println(" DEGRES ");

}
}
```

[Voici le code de la sonde PH SEN0161 :](#)



```
#define SensorPin A0
// pH metre sortie analogique à Arduino entrée analogique 0
#define Offset 0.00          //deviation compensate
#define LED 13   //on définit le PIN de la LED, utile comme indicateur mais facultatif
                  //pour le fonctionnement de la sonde à PH
#define samplingInterval 20
#define printInterval 800
#define ArrayLenth 40      //Définit taille du tableau
int pHArray[ArrayLenth];    //Stock la valeur moyenne du retour du capteur
int pHArrayIndex=0;
void setup(void)
{
    pinMode(LED,OUTPUT);
    Serial.begin(9600);
    Serial.println("pH meter experiment!");    //Test dans le moniteur série
}
void loop(void)
{
    static unsigned long samplingTime = millis();
    static unsigned long printTime = millis();
    static float pHValue,voltage;
    if(millis()-samplingTime > samplingInterval)
    {
        pHArray[pHArrayIndex++]=analogRead(SensorPin);
        if(pHArrayIndex==ArrayLenth)pHArrayIndex=0;
        voltage = avergearray(pHArray, ArrayLenth)*5.0/1024;
        pHValue = 3.5*voltage+Offset;
        samplingTime=millis();
    }
    if(millis() - printTime > printInterval)    //Toutes les 800 millisecondes, affiche un numérique,
                                                //Converti l'état de l'indicateur de la LED
    {
        Serial.print("Volt:");
        Serial.print(voltage,2);
        Serial.print("    Valeur de PH = ");
        Serial.println(pHValue,2);
        digitalWrite(LED,digitalRead(LED)^1);
        printTime=millis();
    }
}
```



```
double avergearray(int* arr, int number){
    int i;
    int max,min;
    double avg;
    long amount=0;
    if(number<=0){
        Serial.println("Error number for the array to avraging!/n");
        return 0;
    }
    if(number<5){ //Si inférieur à 5, calcule directement statistique
        for(i=0;i<number;i++){
            amount+=arr[i];
        }
        avg = amount/number;
        return avg;
    }else{
        if(arr[0]<arr[1]){
            min = arr[0];max=arr[1];
        }
        else{
            min=arr[1];max=arr[0];
        }
        for(i=2;i<number;i++){
            if(arr[i]<min){
                amount+=min;           //arr<min
                min=arr[i];
            }else {
                if(arr[i]>max){
                    amount+=max;       //arr>max
                    max=arr[i];
                }else{
                    amount+=arr[i]; //min<=arr<=max
                }
            }
        }//if
    }//for
    avg = (double)amount/ (number-2);
}//if
return avg;
}
```



Voici le code pour la sonde ORP redox (taux d'oxydoréduction dans l'eau) :

```
#define VOLTAGE 5.00      //system voltage
#define OFFSET 0           //zero drift voltage
#define LED 13            //operating instructions

double orpValue;

#define ArrayLenth  40     //times of collection
#define orpPin 1          //orp meter output,connect to Arduino controller ADC pin

int orpArray[ArrayLenth];
int orpArrayIndex=0;

double avergearray(int* arr, int number){
    int i;
    int max,min;
    double avg;
    long amount=0;
    if(number<=0){
        printf("Error number for the array to avraging!/n");
        return 0;
    }
    if(number<5){ //less than 5, calculated directly statistics
        for(i=0;i<number;i++){
            amount+=arr[i];
        }
        avg = amount/number;
        return avg;
    }else{
        if(arr[0]<arr[1]){
            min = arr[0];max=arr[1];
        }
        else{
            min=arr[1];max=arr[0];
        }
        for(i=2;i<number;i++){
            if(arr[i]<min){
                amount+=min;           //arr<min
                min=arr[i];
            }
        }
    }
}
```



```
        }else {
            if(arr[i]>max) {
                amount+=max;      //arr>max
                max=arr[i];
            }else{
                amount+=arr[i]; //min<=arr<=max
            }
        }//if
    }//for
    avg = (double)amount/(number-2);
}//if
return avg;
}

void setup(void) {
    Serial.begin(9600);
    pinMode(LED,OUTPUT);
}

void loop(void) {
    static unsigned long orpTimer=millis();    //analog sampling interval
    static unsigned long printTime=millis();
    if(millis() >= orpTimer)
    {
        orpTimer=millis()+20;
        orpArray[orpArrayIndex++]=analogRead(orpPin);    //read an analog value every 20ms
        if (orpArrayIndex==ArrayLenth) {
            orpArrayIndex=0;
        }
        orpValue=((30*(double)VOLTAGE*1000)-(75*avergearray(orpArray, ArrayLenth)*VOLTAGE*1000/1024))/75-OFFSET;

        //convert the analog value to orp according the circuit
    }
    if(millis() >= printTime)    //Every 800 milliseconds, print a numerical, convert the state of the LED indicator
    {
        printTime=millis()+800;
        Serial.print("ORP: ");
        Serial.print((int)orpValue);
        Serial.println("mV");
        digitalWrite(LED,1-digitalRead(LED));
    }
}
```

Fixation des seuils :



Pour fixer les seuils des différents capteurs, je me suis renseigné et j'ai effectué des expériences avec les sondes et capteurs pour déterminer quelles valeurs de ces capteurs vont nous rapprocher le plus d'une piscine publique.

Pour le capteur DHT 22, j'ai fixé la température ambiante à 27 degrés Celsius et une hygrométrie de confort se situe entre 60 % et 70%, c'est-à-dire le taux d'humidité de la pièce.

Pour la sonde à PH, j'ai fixé le PH entre 7.2 et 7.4, car l'eau de la piscine est moins irritante pour les baigneurs (peau, yeux, muqueuses) et ne doit pas être trop basique ou acide.

1
Eau
Acide



→ EAU AGGRESSIVE : corrode les parois en béton, les revêtements en ciment et les objets métalliques, irrite les yeux et les muqueuses, est favorable à l'apparition d'algues, et annule l'action des produits.

pH idéal : 7,2 – 7,4

Eau
Alcaline
14



→ EAU DURE : trouble, corrosive, irritante pour les yeux et les muqueuses, qui provoque des précipitations de sel, entartre les filtres et les parois, milieu favorable au calcaire, au développement d'algues et annule l'action des produits.

Conséquences du taux de pH dans l'eau d'une piscine

Pour la sonde DS1820 qui va mesurer la température des bassins, j'ai fixé la température de l'eau à 28 degrés Celsius qui est un peu plus de la température de l'air pour un bon confort. Il est souvent conseillé de fixer une température équivalente de l'air et de l'eau.

Et enfin, pour la sonde ORP redox qui va mesurer la qualité de l'eau(chlorelibre, pH, stabilisant, etc), je vais la fixer à environ 700 mV (expliquée ci-dessous).

Le potentiel rédox (ou potentiel d'oxydo-réduction) est une mesure qui indique le degré auquel une substance peut oxyder ou réduire une autre substance. Le Redox indique donc le pouvoir oxydant ou réducteur d'une substance par rapport à une autre.

Plus simplement, le potentiel redox va nous permettre de juger de l'état de l'eau de notre piscine (plus ou moins oxydante en fonction de la concentration de désinfectant).

Le rédox est aussi appelé ORP (Oxydo Reduction Potential en anglais).



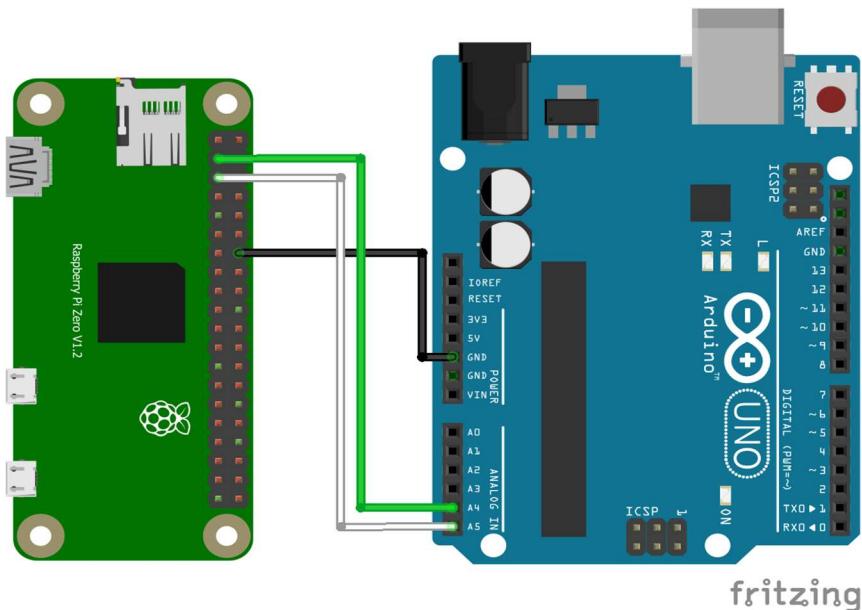
Le potentiel redox dépend quant à lui du taux de chlore libre, du type de chlore utilisé, de la valeur du pH et du stabilisant qui doit avoir un taux compris entre 20 et 40 ppm. La mesure redox piscine devrait être comprise entre 650 et 750 mV (millivolts) pour avoir une eau désinfectée et désinfectante.

Le potentiel rédox donne donc une indication sur l'état de l'eau de la piscine. L'eau est plus ou moins oxydante en fonction de la concentration de chlore.

Une mesure positive du rédox signifie que l'eau est oxydante. Une eau à 0 mV signifierait que l'eau est neutre et pure.

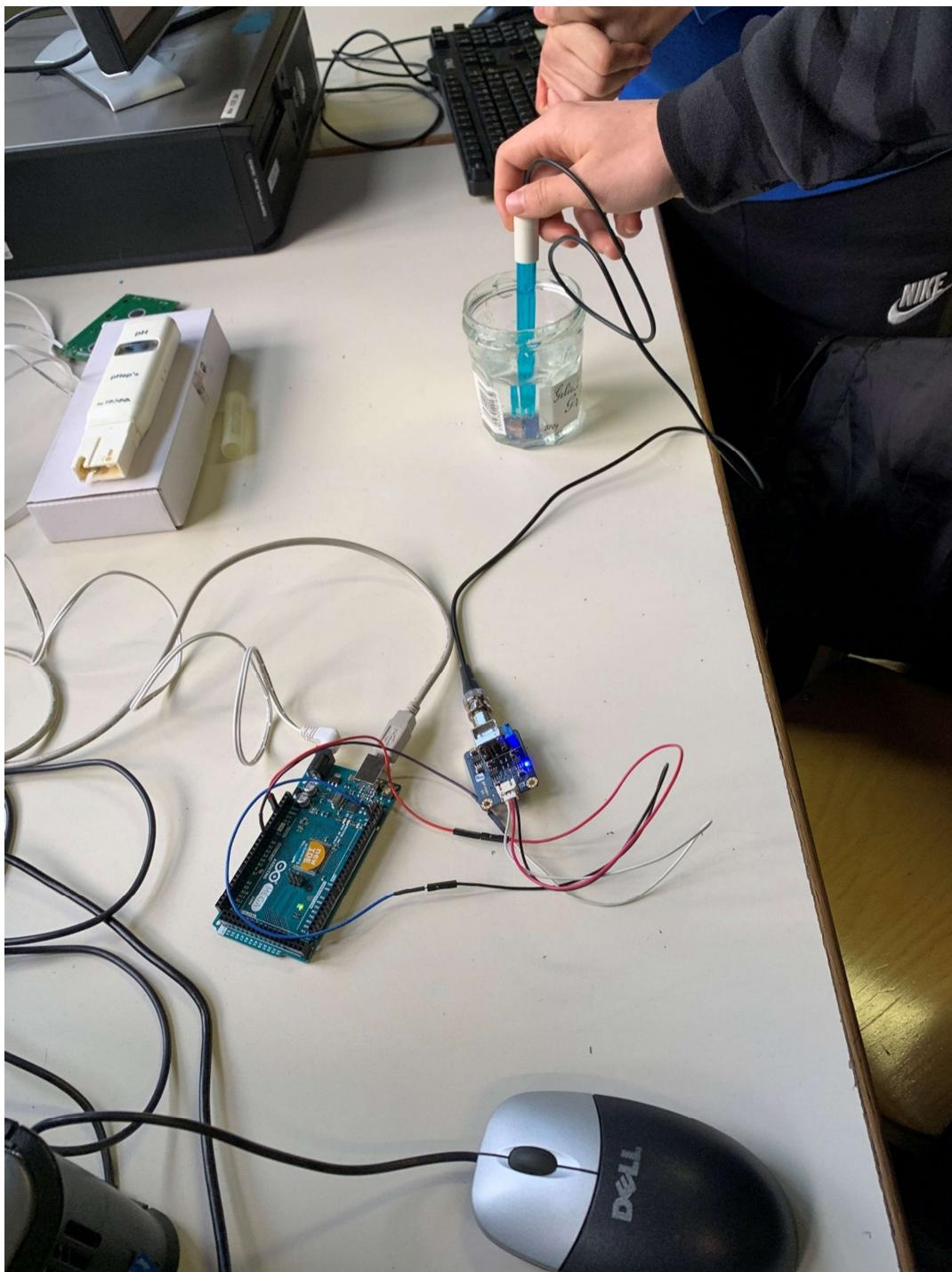
Branchements obtenus :

Branchements entre la carte Arduino et la raspberry Pi :



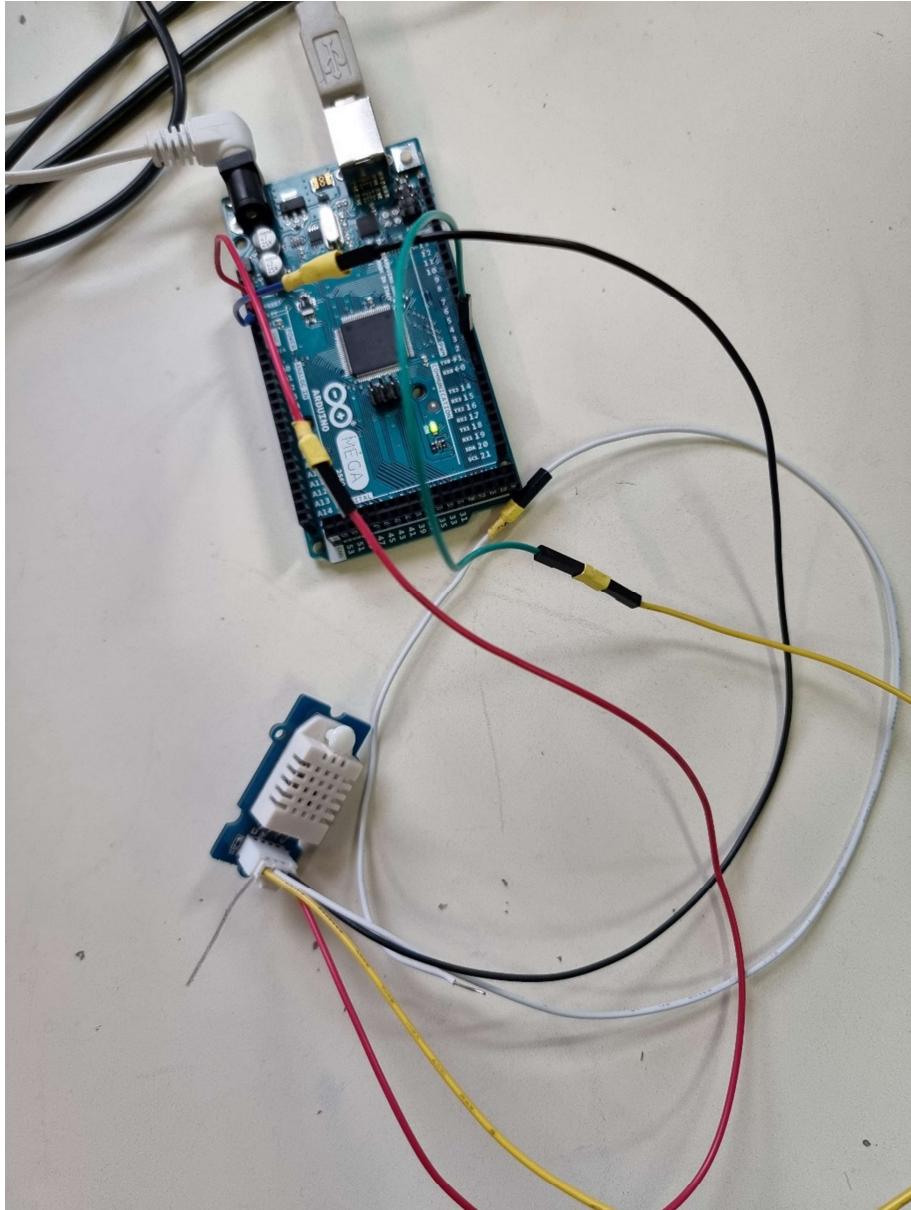


Branchements entre la sonde à pH SEN0161 et la carte arduino :



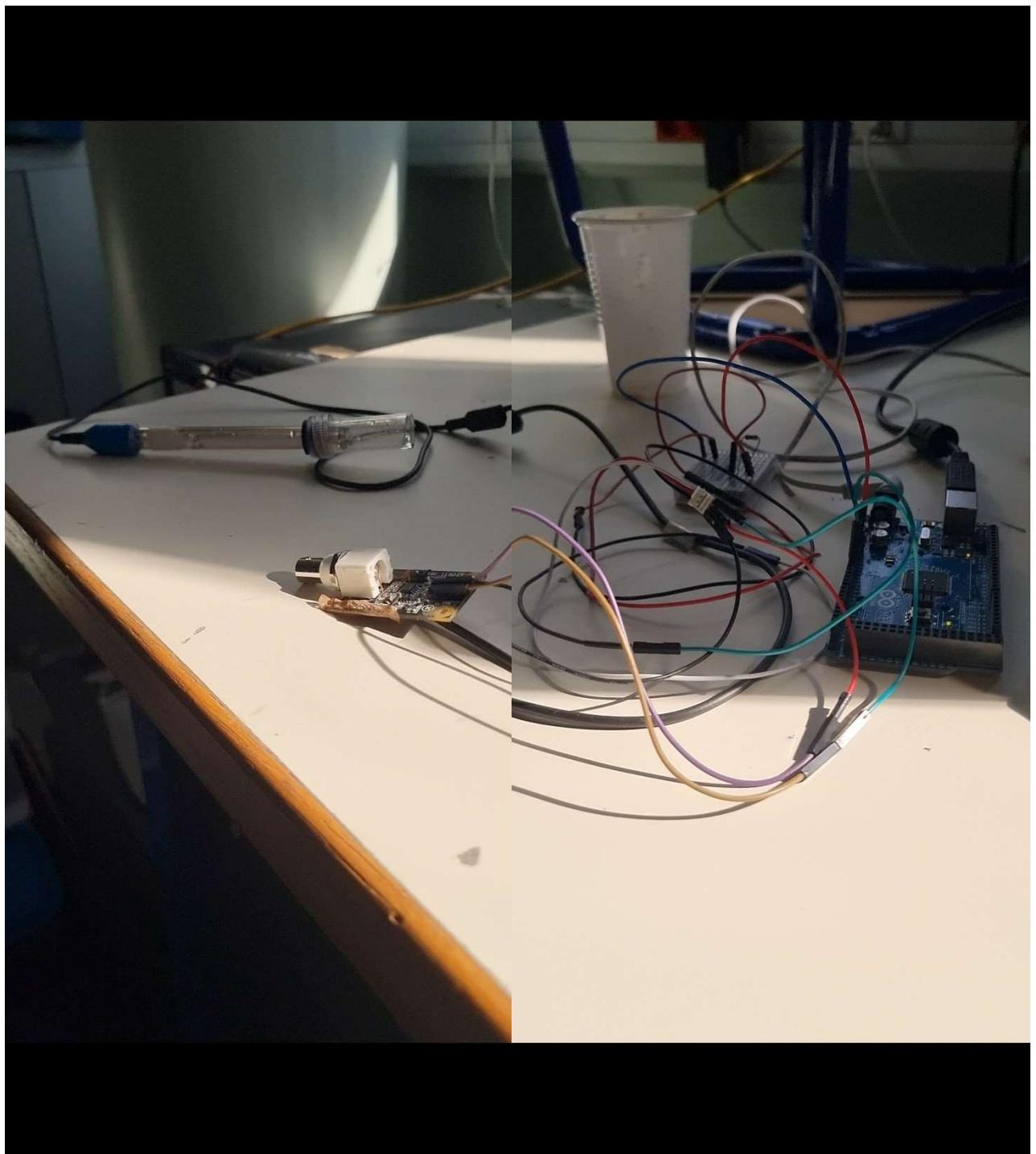


Branchements entre le capteur DHT 22 et la carte arduino :





Branchements entre la sonde oxydoréduction ORP redox et la carte arduino :





Transmission à la base de donnée :

Pour transmettre les informations des capteurs dont j'ai fait l'acquisition, je vais utiliser la transmission I2C entre la carte Arduino et la Raspberry Pi. Pour cela, il faut :

Matériel

- Ordinateur
- Arduino UNO x1
- Raspberry Pi 3B+
- Jumper cables x3

Schéma de câblage

Pour établir la communication I2C entre Raspberry Pi et Arduino, il nous faut relier physiquement le bus qui utilise 3 broches. Une communication I2C est défini par un bus de deux fils (parfois appelé TWI, Two Wire Interface) et une adresse. Les broches utilisées par la communication I2C sont généralement fixé pour chaque appareil. L'une sur laquelle sont envoyées les données (SDA Serial Data Line) et sur l'autre l'horloge de synchronisation (SLC Serial Clock Line). Les masses des deux cartes doivent être reliées pour établir une référence commune de potentiel.

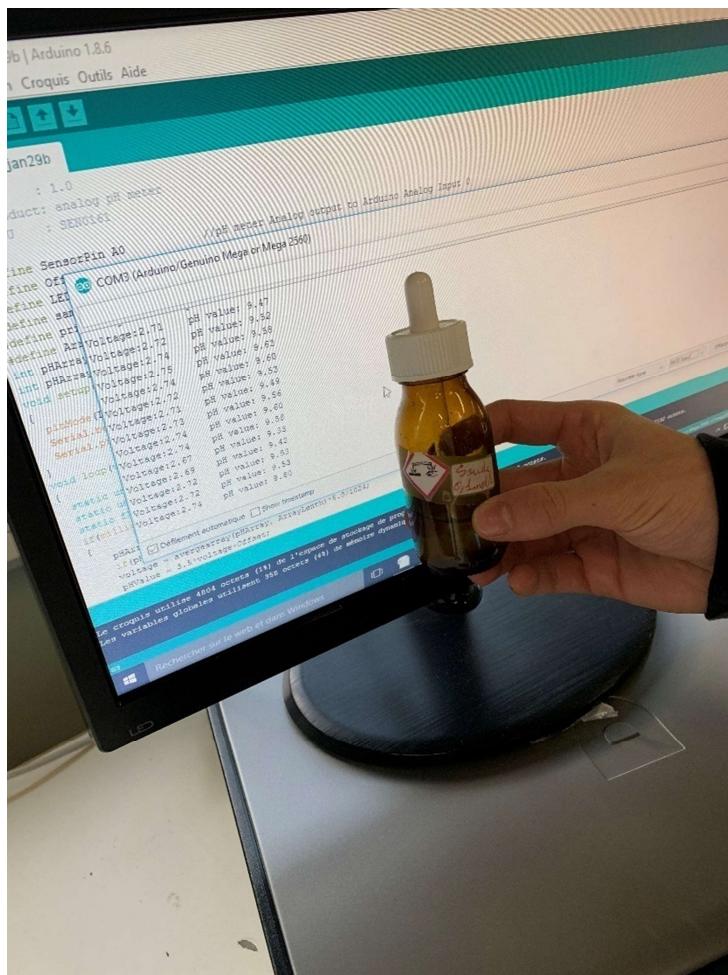
- SDA BCM2(RPI) <-> SDA A4(Arduino)
- SCL BCM3(RPI) <-> SCL A5(Arduino)
- GND (RPI) <-> GND(Arduino)

[Le cablage a été effectué dans la partie précédente ----> Branchement obtenu]

Le reste de la procédure a été faite par l'étudiant 3 (Adel Malki) car nous étions en collaboration pour cette tâche.



Annexes :



```
#include <Arduino.h>
//pH meter Analog output to Arduino Analog Input
void setup() {
    Serial.begin(9600);
}
void loop() {
    float pHValue = analogRead(A0);
    pHValue = map(pHValue, 0, 1023, 9.0, 14.0);
    Serial.println("pH value: " + String(pHValue));
}
float pHArray[10];
int ArrayLenth = 0;
if(ArrayLenth < 10) {
    pHArray[ArrayLenth] = pHValue;
    ArrayLenth++;
}
else {
    if(pHArray[0] > 0) {
        pHValue = averagearray(pHArray, ArrayLenth);
        Serial.println("pH value: " + String(pHValue));
    }
}
```

Le croquis utilise 4804 octets (1%) de l'espace de stockage de programmes. Le maximum est de 50650 octets. Les variables globales utilisent 358 octets (4%) de mémoire dynamique, ce qui laisse 47422 octets disponibles.