



Table des matières

1) Introduction.....	1
2) Données météorologique locales.....	1
3) Prévision météorologiques.....	3
4) Intégration d'un afficheur LCD.....	7
5) Exploitation des capacités d'actionnement de notre objet connecté.....	13

1) Introduction

Dans le but d'accroître la productivité, de produire des fruits, des légumes et des céréales de bonne qualité, de satisfaire les besoins alimentaires de la population mondiale et de s'adapter aux changements climatiques, de plus en plus d'agriculteurs s'équipent ou envisagent de s'équiper en dispositifs IoT d'aide à l'activité agricole.

Pour plus d'informations, vous pouvez consulter, à titre d'exemples, les sites suivants :

<https://www.objetconnecte.com/agriculteurs-iot-etude-0809/>

<https://www.alphabrown.com/single-post/2018/05/02/IoT-market-for-agriculture-can-reach-4-BL---New-study>

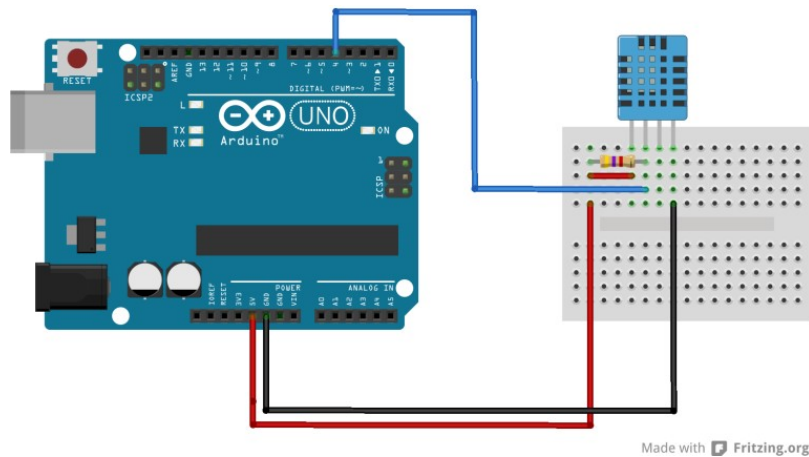
Dans ce TP, nous nous proposons de développer un prototype d'objet connecté permettant d'aider les agriculteurs dans leur activité. Les expérimentations seront réalisées sur une carte de développement utilisant le SoC ESP32.

2) Données météorologique locales

Dans cette première étape, nous nous intéresserons à l'acquisition des données météorologiques actuelles. Nous souhaitons obtenir la température et l'humidité relative (degré d'hygrométrie). Pour cela, nous utiliserons le composant DHT11 fourni dans votre kit.

1) Le DHT11 a 4 broches :

- GND
- VCC
- DATA
- NC



2) Je réalise les interconnexions nécessaires et je fais valider par le prof avant de mettre sous tension

3) Dans arduino je clique sur outils → gérer les bibliothèques et dans la recherche je tape dht.h et je télécharge les librairies d'adafruit pour pouvoir faire un programme qui va lire la température et l'humidité :

```
#include "DHT.h"

#define DHTPIN 4

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println(F("DHTxx test!"));

  dht.begin();
}

void loop() {
  delay(5000);
```

```
float h = dht.readHumidity();
float t = dht.readTemperature();

if (isnan(h) || isnan(t)) {
  Serial.println(F("Failed to read from DHT sensor!"));
  return;
}

float hic = dht.computeHeatIndex(t, h, false);

Serial.print(F("Humidity: "));
Serial.print(h);
Serial.print(F("% Temperature: "));
Serial.print(t);
Serial.println(F("°C "));
}
```

4) Je téléverse mon programme et je test qu'il fonctionne bien en variant la température et l'humidité :

```
Humidity: 64.00% Temperature: 21.20°C
Humidity: 66.00% Temperature: 21.20°C
Humidity: 73.00% Temperature: 21.30°C
Humidity: 78.00% Temperature: 21.50°C
Humidity: 88.00% Temperature: 22.00°C
Humidity: 95.00% Temperature: 22.80°C
Humidity: 95.00% Temperature: 24.10°C
Humidity: 95.00% Temperature: 24.90°C
Humidity: 95.00% Temperature: 25.10°C
```

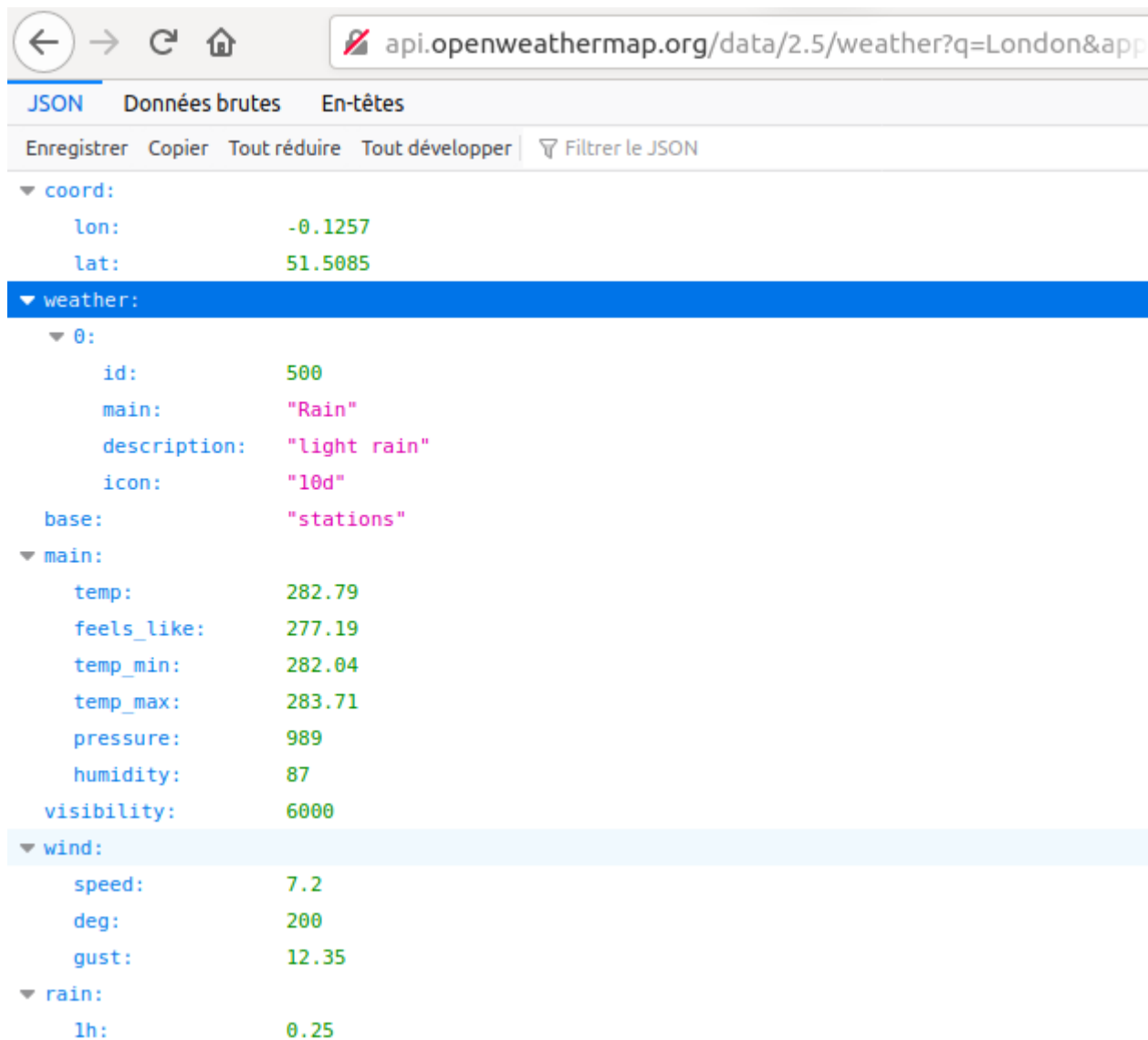
3) Prévision météorologiques

5) L'API se sert de données météorologiques récupérées via des satellites, des radars, de grands réseaux de stations météo... Dans l'API on a un exemple d'utilisation qui est le suivant :

```
api.openweathermap.org/data/2.5/weather?q=London&appid={API key}
```

Il suffit simplement de copier ce lien dans une page web et de remplacer le "API key" par la clé API qui nous a été donnée par mail

6) Je me crée un compte et je teste l'utilisation de l'API :



```
{
  "coord": {
    "lon": -0.1257,
    "lat": 51.5085
  },
  "weather": [
    {
      "id": 500,
      "main": "Rain",
      "description": "light rain",
      "icon": "10d",
      "base": "stations"
    }
  ],
  "main": {
    "temp": 282.79,
    "feels_like": 277.19,
    "temp_min": 282.04,
    "temp_max": 283.71,
    "pressure": 989,
    "humidity": 87,
    "visibility": 6000
  },
  "wind": {
    "speed": 7.2,
    "deg": 200,
    "gust": 12.35
  },
  "rain": {
    "1h": 0.25
  }
}
```

Quand on met le lien dans une page web il nous sort un fichier .json qui nous donne des infos sur la météo.

Les données météorologiques qui nous intéressent le plus ici sont :

- La température minimale et maximale
- La pression dans l'air
- L'humidité

7) il faut qu'avec l'esp32 on se connecte à notre réseau wifi puis qu'ensuite on se connecte sur le site de weathermap et qu'on récupère les infos d'une ville puis qu'on l'affiche dans le moniteur série

8) Je fais donc le programme qui se connecte à mon wifi puis ensuite au site et qui envoi toutes les données dans mon moniteur série :

```
#include <WiFi.h>
#include <HTTPClient.h>
const char* ssid = "Nom_Box";
const char* password = "Mot_De_Passe";

const String endpoint =
"http://api.openweathermap.org/data/2.5/weather?
q=La_Ville,pt&APPID=";
const String key = "Clé_Site";
HTTPClient http;

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }
  Serial.println("Connected to the WiFi network");
}

void loop() {
  if ((WiFi.status() == WL_CONNECTED)) { //Check the current
connection status
    http.begin(endpoint + key);
    int httpCode = http.GET();
    if (httpCode > 0) { //Check for the returning code

      String payload = http.getString();
      Serial.println(httpCode);
      Serial.println(payload);

    } else {
      Serial.println("Error on HTTP request");
    }
    http.end();
  }
  delay(2000);
}
```

9) Le programme suivant va se connecter sur l'API et afficher la température et l'humidité récupérées sur Agde et afficher la température et l'humidité de mon capteur :

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
const char* ssid = "Livebox-FF14";
const char* password = "xykzPsGC7ikubneYy6";

const String endpoint =
"http://api.openweathermap.org/data/2.5/weather?
q=Agde,fr&units=metric&APPID=";
const String key = "0fd4436d05d93c9338e70a832028a57b";
HTTPClient http;

void setup() {
  Serial.begin(115200);
  dht.begin();
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }
  Serial.println("Connected to the WiFi network");
}

void loop() {
  delay(2000);
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Local Humidity: "));
  Serial.print(h);
  Serial.print(F("% Local Temperature: "));
  Serial.print(t);
  Serial.println(F("°C "));
}
```

```
    if ((WiFi.status() == WL_CONNECTED)) { //Check the current
connection status
    http.begin(endpoint + key);
    http.useHTTP10(true);
    int httpCode = http.GET();
    if (httpCode > 0) { //Check for the returning code
        DynamicJsonDocument doc(2048);
        deserializeJson(doc, http.getStream());

        Serial.println("Résultats Température et Humidité API :");
        Serial.print(doc["main"]["temp"].as<long>());
        Serial.println("°C");
        Serial.print(doc["main"]["humidity"].as<long>());
        Serial.println("%");
    }
}
}
```

```
Connecting to WiFi..
Connected to the WiFi network
Local Humidity: 67.00% Local Temperature: 22.60°C
Résultats Température et Humidité API :
12°C
76%
```

Je peux voir ma température et humidité locales et aussi celles de l'API et donc je peux comparer entre chez moi et dehors.

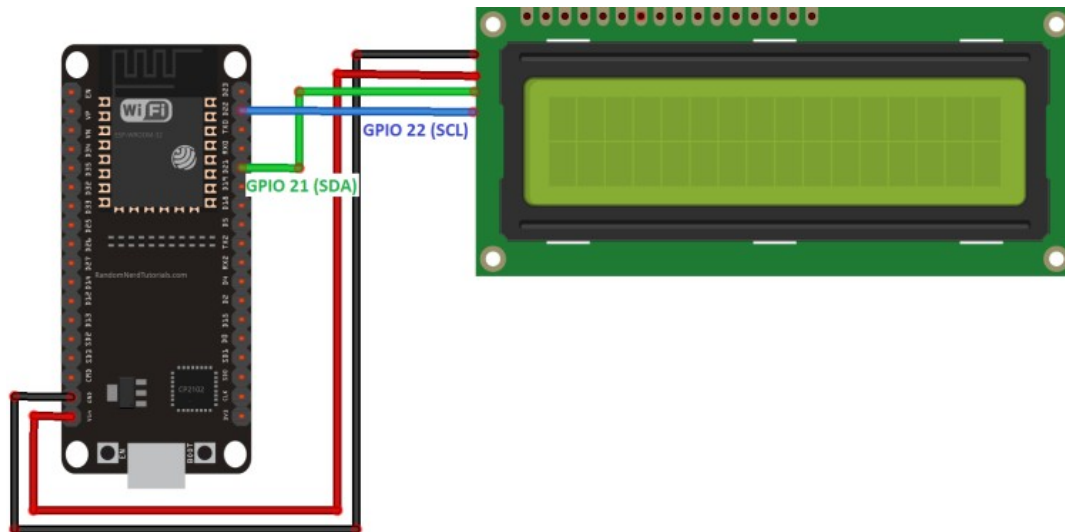
4) Intégration d'un afficheur LCD

10) Notre écran LCD communique via le bus i2c (4 broches) :

- VCC (5V)
- GND
- SCL (Serial Clock Line)
- SDA (Serial Data Line)

On a juste à connecter ces 4 broches sur les bons ports de notre ESP et ensuite tout se passe dans la partie programme.

11) Je réalise les interconnexions et je fais vérifier par le prof :



12) J'installe la librairie donnée dans le sujet du TP (c'est un github).

Pour l'installer je la télécharge puis dans le dossier "libraries" d'arduino je crée un dossier nommé "LiquidCrystal_I2C" et dedans je place tous les fichiers téléchargés

13) Je prend un programme exemple nommé "HelloWorld" situé dans LiquidCrystal_I2C et je le téléverse dans mon ESP et je vérifie que "Hello, World" s'affiche bien sur mon LCD :

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup()
{
    // initialize the LCD
    lcd.begin();

    // Turn on the backlight and print a message.
    lcd.backlight();
    lcd.print("Hello, world!");
}

void loop()
{
    // Do nothing here...
}
```


14) J'adapte maintenant le programme exemple pour afficher sur l'écran LCD les valeurs d'humidité et de température récupérées par mon capteur dht11 :

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

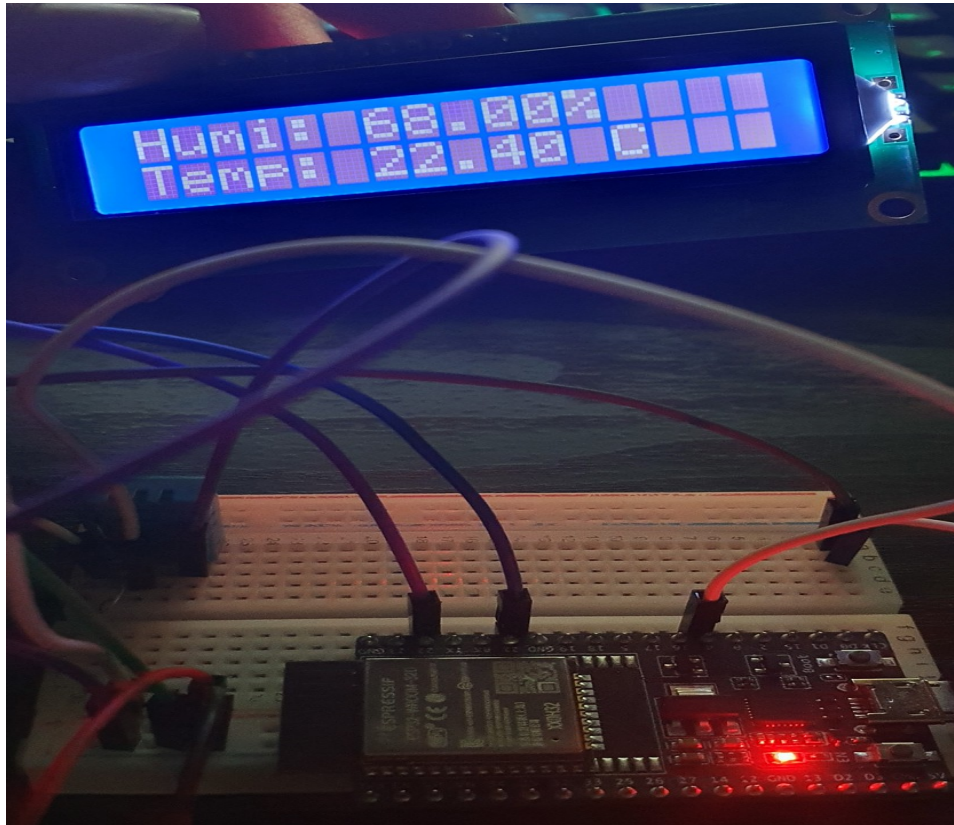
// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup()
{
    dht.begin();
    lcd.begin();
    lcd.backlight();
}

void loop()
{
    float h = dht.readHumidity();
    float t = dht.readTemperature();

    if (isnan(h) || isnan(t)) {
        lcd.println(F("Failed to read from DHT sensor!"));
        return;
    }

    float hic = dht.computeHeatIndex(t, h, false);
    lcd.setCursor(0,0);
    lcd.print(F("Humi: "));
    lcd.print(h);
    lcd.print(F("%"));
    lcd.setCursor(0,1);
    lcd.print(F("Temp: "));
    lcd.print(t);
    lcd.print(F(" C"));
    delay(1000);
    lcd.clear();
}
```

Mon écran LCD m'affiche bien mes valeurs :

Le programme suivant me permet d'afficher la température et l'humidité en locale (avec capteur dht11) sur le LCD pendant 5s puis ensuite afficher la température et l'humidité récupéré via le site api.openweathermap.org pendant 5s aussi :

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
const char* ssid = "Livebox-FF14";
const char* password = "xykzPsGC7ikubneYy6";

const String endpoint =
"http://api.openweathermap.org/data/2.5/weather?
q=Agde,fr&units=metric&APPID=";
const String key = "0fd4436d05d93c9338e70a832028a57b";
HTTPClient http;
```

```
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  dht.begin();
  lcd.begin();
  lcd.backlight();
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {

    delay(1000);
    Serial.println("Connecting to WiFi..");

  }
  Serial.println("Connected to the WiFi network");
}

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();

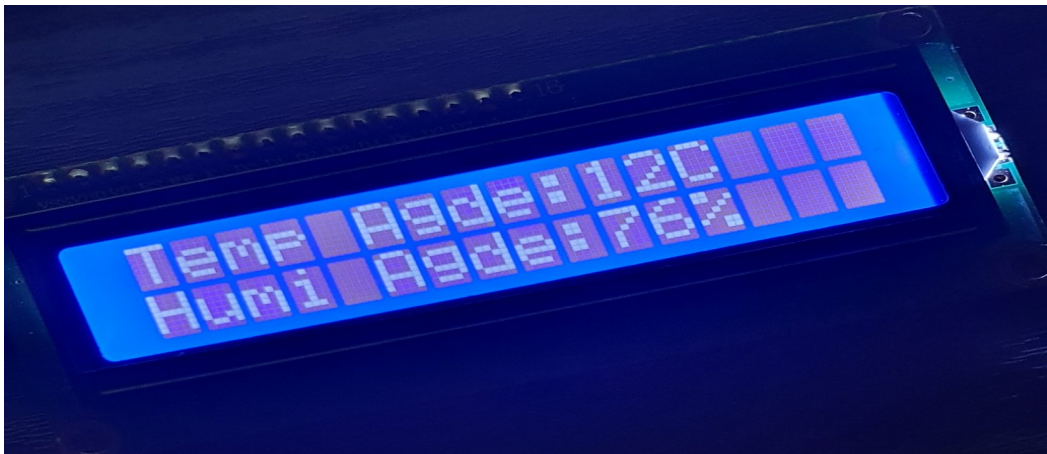
  if (isnan(h) || isnan(t)) {
    lcd.println(F("Failed to read from DHT sensor!"));
    return;
  }

  float hic = dht.computeHeatIndex(t, h, false);
  lcd.setCursor(0,0);
  lcd.print(F("Humi Loc:"));
  lcd.print(h);
  lcd.print(F("%"));
  lcd.setCursor(0,1);
  lcd.print(F("Temp Loc:"));
  lcd.print(t);
  lcd.print(F("C"));
  delay(5000);
  lcd.clear();

  if ((WiFi.status() == WL_CONNECTED)) { //Check the current
connection status
    http.begin(endpoint + key);
    http.useHTTP10(true);
    int httpCode = http.GET();
    if (httpCode > 0) { //Check for the returning code
      DynamicJsonDocument doc(2048);
      deserializeJson(doc, http.getStream());
```

```
    lcd.setCursor(0,0);  
    lcd.print("Temp Agde:");  
    lcd.print(doc["main"]["temp"].as<long>());  
    lcd.print("C");  
    lcd.setCursor(0,1);  
    lcd.print("Humi Agde:");  
    lcd.print(doc["main"]["humidity"].as<long>());  
    lcd.print("%");  
    delay(5000);  
    lcd.clear();  
  }  
}  
}
```

Résultat en photo :



5) Exploitation des capacités d'actionnement de notre objet connecté

15) On peut utiliser l'objet pour mesurer le taux d'humidité et la température extérieure et ainsi si on a une humidité ou une température trop basse on peut activer un système d'arrosage automatique et à l'inverse qui ne s'activera pas si il y a de fortes chaleurs

16) On pourrait se servir de ce genre d'objets connectés pour la traite des vaches par exemple. On pourrait vérifier le taux de lait récupéré...