Heu-Combe Nolan TSTI2D2

Revue de Projet :
Le jardin connecté

Introduction :

L'arrosage régulier d'un jardin peut être une tâche fastidieuse et chronophage pour de nombreux propriétaires. Dans le but de simplifier cette tâche et de promouvoir une utilisation durable des ressources en eau, ce projet propose la conception et la mise en place d'un système intelligent d'arrosage automatisé. Ce système utilise une récupération des eaux de pluie à l'aide de deux citernes, une souterraine et une en hauteur, gérées par une application et des cartes Arduino.

Problématique :

La problématique abordée dans ce projet est de concevoir un système d'arrosage automatisé capable de récupérer et d'utiliser efficacement les eaux de pluie tout en offrant une gestion simplifiée pour l'utilisateur. Il vise à optimiser l'arrosage des plantes en utilisant des données, telles que la température, la luminosité, l'humidité et le niveau d'eau.

Objectifs :

1. Gestion des citernes : L'objectif est de mettre en place un système de gestion des citernes qui permettra le transfert de l'eau de la citerne souterraine vers la citerne en hauteur à l'aide d'une pompe. Cela garantira un approvisionnement suffisant en eau pour l'arrosage des plantes.

2. Un affichage visuel des niveaux d'eau : Un affichage visuel est mis en place pour permettre à l'utilisateur de surveiller facilement les niveaux d'eau dans chaque citerne.

3. Communication entre les cartes Arduino : Le système est basé sur des cartes Arduino et des capteurs afin de collecter des données environnementales et la gestion du système. La communication entre les différentes cartes Arduino utilise la fréquence 433 MHz, assurant ainsi une transmission fiable et sans fil des informations.

Objectif du projet :

Ce système vise à réduire le gaspillage des ressources en utilisant efficacement l'eau de pluie disponible. Il assure également le maintien optimal du taux d'humidité dans le sol en s'adaptant aux changements climatiques. L'objectif final est de réduire la consommation d'eau potable précédemment utilisée pour l'arrosage du jardin, favorisant ainsi une production responsable de fruits, légumes et plantes grâce à l'utilisation de l'eau de pluie.

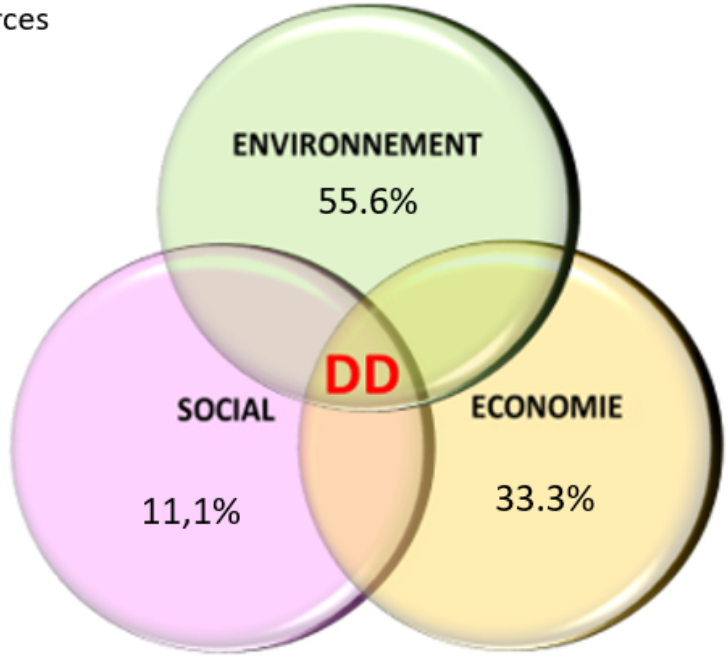
Les 3 piliers du Développement Durable

Environnement:

- Utiliser des énergies renouvelables
- Maintenir ou d'améliorer la qualité du milieu naturel
- Prendre en compte la gestion des ressources naturelles
- Intégrer l'adaptation aux changements climatiques, aux risques majeurs
- Réduire le gaspillage des ressources

Economie:

- Avoir un bien, un service, un produit...accessible à tous les budgets
- Avoir une rentabilité, une efficience à long terme
- Contribuer à la création de richesses économiques individuelles



Social:

- Améliorer la qualité de vie

Conclusion:

Les 3 piliers du Développement Durable sont pris en compte dans le projet. il répond donc à ce concept avec une approche ENVIRONNEMENTALE prédominante.

SCHEMA DE COMMUNICATION

PROJET JARDIN CONNECTE

Application

Ardu block

Envoi commande

Envoi des données

Communication Bluetooth

ARDUINO 1

Communication 433Mhz

433 MHz

ARDUINO 2

Affiche les données

Communication 433Mhz

433 MHz

ARDUINO 3

(Nano)

LED niveau d'eau

Réservoir n°1

Capteur niveau d'eau

Pompe

Capteur niveau d'eau

Réservoir n°2

LED niveau d'eau

Capteur niveau d'eau

Capteur niveau d'eau

Electrovanne

Capteur température, luminosité

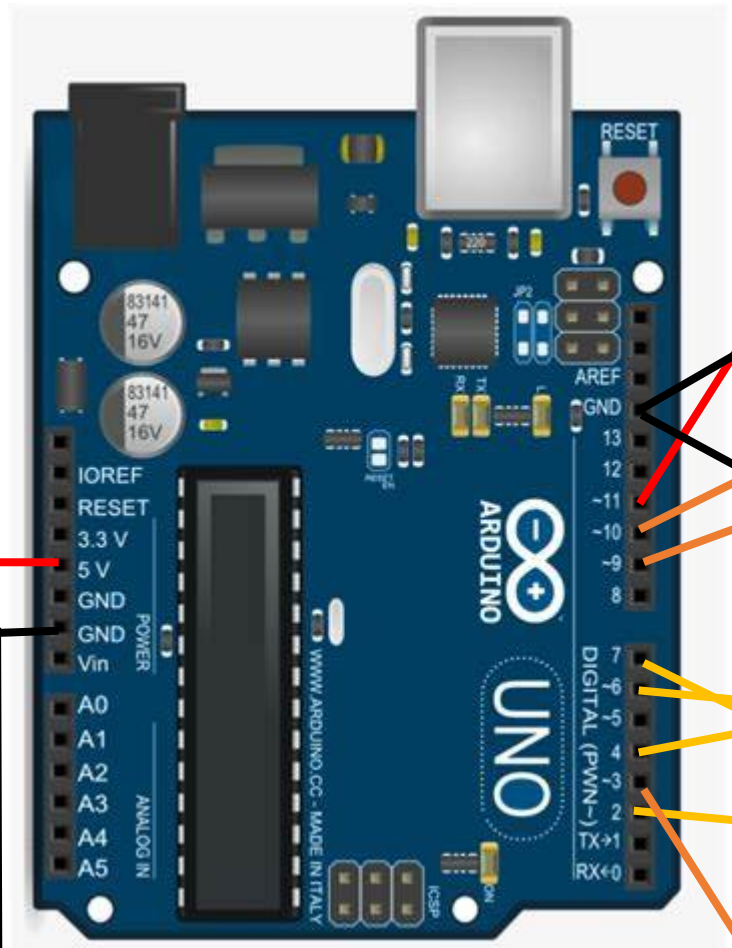
Dans le jardin (sur le sol)

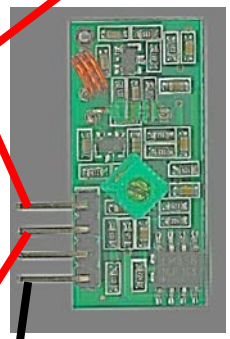
Capteur humidité dans le sol

Partie informatique

Partie Arduino

Schéma de branchement Arduino





11

10

1Kohms

9

4

6

220 Ohm

2

7

3

Citerne n°1

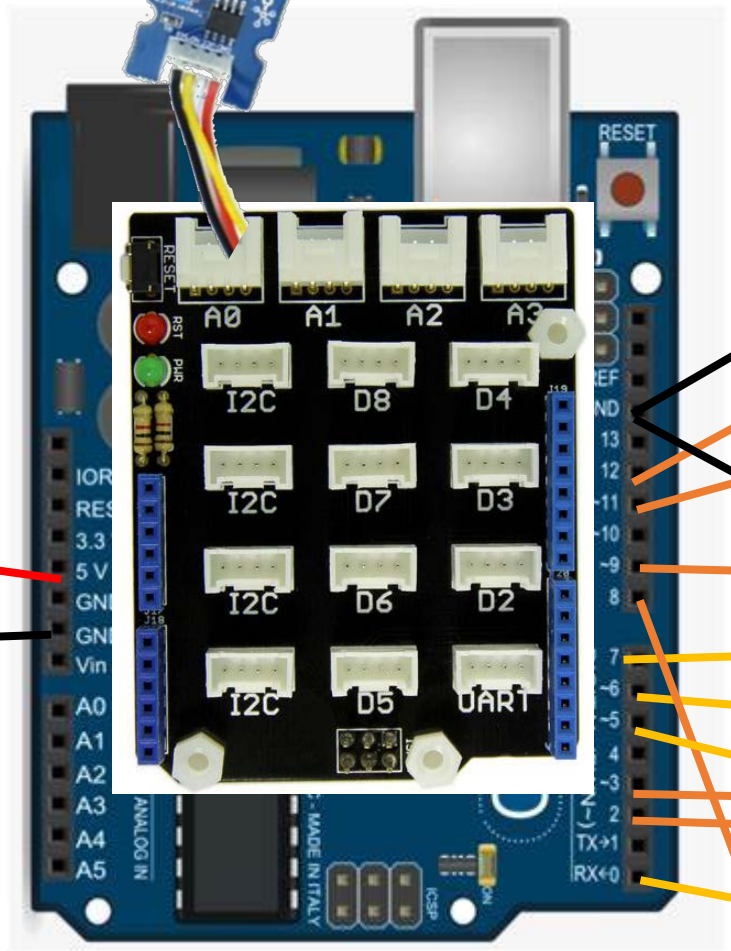
Batterie

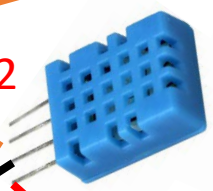
Panneaux Solaire

Régulateur

Pompe

Citerne n°2





3

9

12

1Kohms

11

2

7

6

5 220 Ohm

0

8

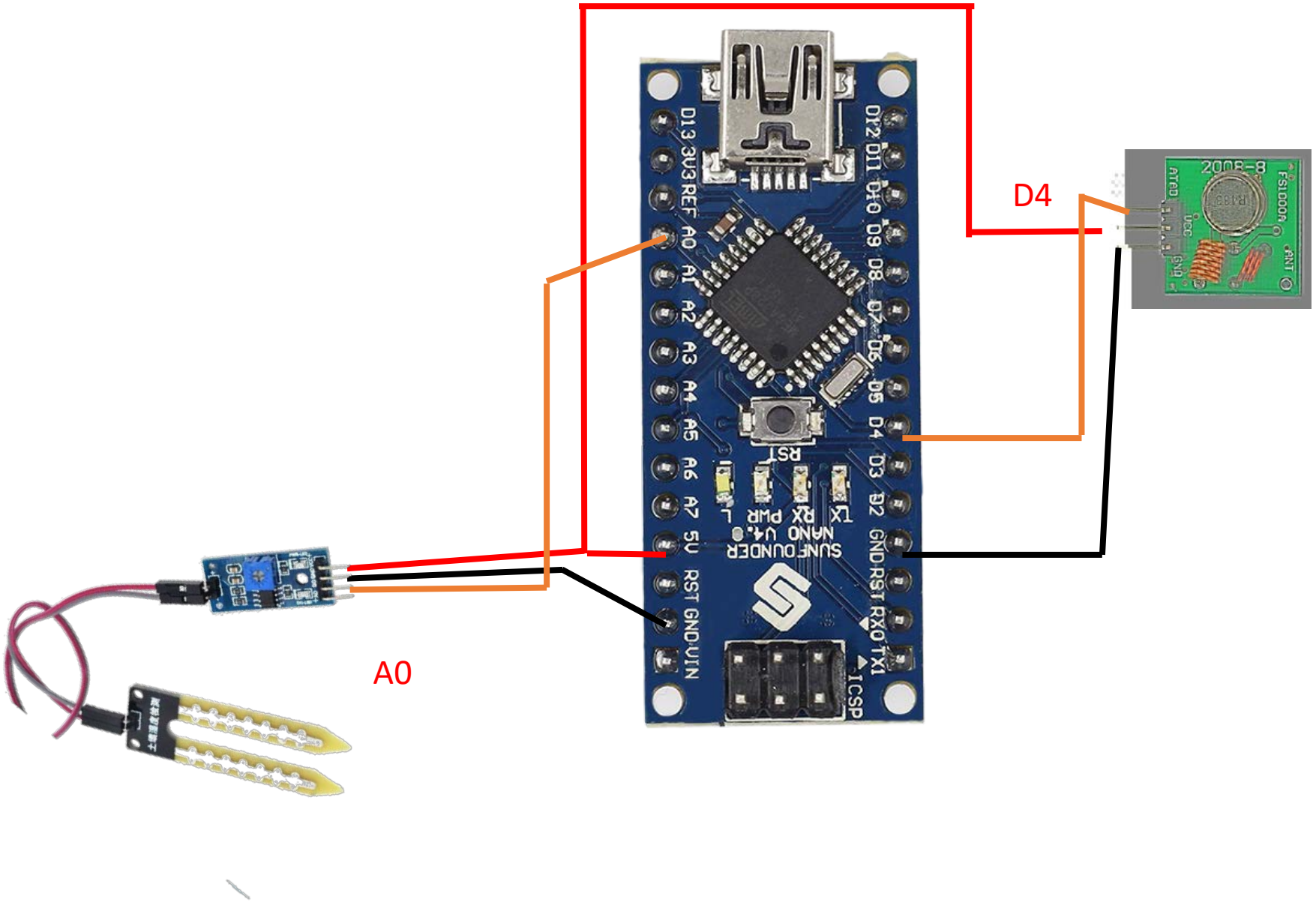
Batterie

Panneaux Solaire

Régulateur

Electrovanne

Arduino n°3


D4
A0
D13 3V3 REF A0 A1 A2 A3 A4 A5 A6 A7 5V RST GND VIN
D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 GND RST RX0 TX1
RST
ICSP
SUNFOUNDER
NANO V4.0
RX PWR
TX
L
FS1000A
ANT
ATED
Vcc
GND

Programme Arduino :

Le programme Arduino utilise les bibliothèques Virtual Wire et Task Scheduler.

- La bibliothèque Virtual Wire permet la communication sans fil entre les différentes cartes Arduino utilisant la fréquence de 433 MHz. Elle assure une transmission fiable et efficace des données entre les composants du système.
- La bibliothèque Task Scheduler permet de gérer et d'ordonner les différentes tâches et actions à effectuer par le système. Elle permet de programmer des tâches ce qui facilite la gestion des différentes opérations du système d'arrosage automatisé.

En utilisant ces bibliothèques, le programme Arduino peut interagir avec les capteurs de température, d'humidité et de niveau d'eau, et exécuter les actions appropriées en fonction des données collectées. Cela permet une automatisation efficace du système d'arrosage en fonction des besoins en eau des plantes et des conditions environnementales.

- Le programme Arduino est également composé de trois sous-programmes (« Pompe », « Affichage visuel de la citerne », « Réception 433Mhz »)

Début du programme :

```
#include <TaskScheduler.h>
#include <VirtualWire.h>
#include <VirtualWire_Config.h>

// Déclaration des broches
const int relaisPin = 3;  // Broche du relais
  float humiditeAir;
  float temperature;
  int ID = 2; // ID de l'émetteur
  int waterLevelPin = 3; // Pin pour la lecture du niveau d'eau
  float luminosity = 0.0;
  const int led1 = 2;
  const int led2 = 6;
  const int led3 = 4;
  const int led4 = 7;
  const int capteur1 = 9;
  const int capteur2 = 10;
  const int receive_pin = 11;
```

Déclaration des sous programmes :

```
// Sous-programme 1 : Pompe
void pompe() {
  int waterLevel = 0;
  bool buttonState1 = digitalRead(9); //bas
  bool buttonState2 = digitalRead(10);//haut
  if ( waterLevel == 0 || waterLevel == 25 && ( buttonState1 == LOW  && buttonState2 == LOW ) || ( buttonState1 == HIGH   && buttonState2 == LOW  ))
  {
  digitalWrite(relaisPin, HIGH);
  digitalWrite(7,HIGH );
  }
 else if (( buttonState1 == LOW && buttonState2 == HIGH ) ||  waterLevel == 90)
{
  digitalWrite(relaisPin, LOW);
}
}
```

```
// Sous-programme 2 : Affichage visuel de la citerne
void affichage() {
 bool buttonState1 = digitalRead(9); //bas
  bool buttonState2 = digitalRead(10);//haut
  if ( buttonState1 == LOW  && buttonState2 == LOW   )
  {
  digitalWrite(6,HIGH );
  digitalWrite(2,HIGH );
  digitalWrite(4,LOW );
  Serial.println("Niveau d'eau réservoir 1: 25%");
  delay(1500);
  }
 else if ( buttonState1 == HIGH   && buttonState2 == LOW  )
  {
   digitalWrite(6, HIGH);
   digitalWrite(4,HIGH );
   digitalWrite(2,HIGH);
   Serial.println("Niveau d'eau réservoir 1: 90%");
   delay(1500);
  }

  else if ( buttonState1 ==LOW && buttonState2 == HIGH )
  {
  digitalWrite(6, LOW);
  digitalWrite(4,LOW) ;
  digitalWrite(2,LOW) ;
  Serial.println("Niveau d'eau réservoir 1: 0%");
  delay(1500);
  }
}
```

```
// Sous-programme 3 : Réception 433Mhz
void reception() {
  uint8_t buflen = VW_MAX_MESSAGE_LEN; // Taille maximale du message reçu
  uint8_t buf[buflen]; // Tampon pour stocker le message reçu

  if (vw_get_message(buf, &buflen)) { // Si un message a été reçu
    char waterLevelString[buflen];
    strncpy(waterLevelString, (char *)buf, buflen);
    waterLevelString[buflen - 1] = '\0'; // Ajouter le caractère nul de fin de chaîne
    int id, waterLevel;
    sscanf(waterLevelString, "%d,%d", &id, &waterLevel);
    if (id == ID) {
      Serial.print("ID = ");
      Serial.println(id);
      Serial.print("Niveau d'eau réservoir 2: ");
      Serial.print(waterLevel);
      Serial.println("%");

}
```

Configuration et planification des tâches qui seront exécutées périodiquement par le programme :

```
// Objets TaskScheduler
Scheduler scheduler;
Task pompeTask(0, TASK_FOREVER, &pompe); // Tâche de la pompe toutes les 0 seconde
Task affichageTask(100, TASK_FOREVER, &affichage); // Tâche de l'affichage toutes les 0.1 secondes
Task receptionTask(100, TASK_FOREVER, &reception); // Tâche de la réception toutes les 0,1 seconde
```

Initialisation et configuration du programme Arduino au démarrage.

```
void setup() {
  // Ajout des tâches au planificateur
 scheduler.addTask(pompeTask);
 scheduler.addTask(affichageTask);
 scheduler.addTask(receptionTask);

  // Démarre les tâches
  pompeTask.enable();
  affichageTask.enable();
  receptionTask.enable();
  pinMode(2, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(9,INPUT_PULLUP);
  pinMode(10,INPUT_PULLUP);
  pinMode(7, OUTPUT);
  pinMode(relaisPin, OUTPUT);       // Configure la broche du relais en sortie
  pinMode(waterLevelPin, INPUT); // Configuration de la broche de lecture du niveau d'eau en entrée
  // Éteint la pompe au démarrage
  digitalWrite(relaisPin, LOW);
  vw_set_rx_pin(11); // Configuration de la broche de réception sans fil
  Serial.begin(9600);
  Serial.println("Setup Receiver");
  vw_setup(2000);  // Bits per sec
  vw_rx_start();

}
```

Boucle principale du programme Arduino

```
void loop(){
 scheduler.execute(); // Exécute les tâches planifiées

}
```

Fin du programme

Fonctionnement du projet :

Le fonctionnement du projet est automatisé dès que toutes les cartes Arduino sont alimentées. Le programme s'exécute de manière autonome sans nécessiter d'interventions supplémentaires.

La pompe de relevage s’active quand le réservoir n°2 est vide et quand le réservoir n°1 est plein ou rempli à moitié. La pompe s’éteint quand le réservoir n°2 est rempli ou si le réservoir n°1 est vide.

On peut retrouver toutes les informations dans le moniteur de série en temps réel.

Moniteur série :

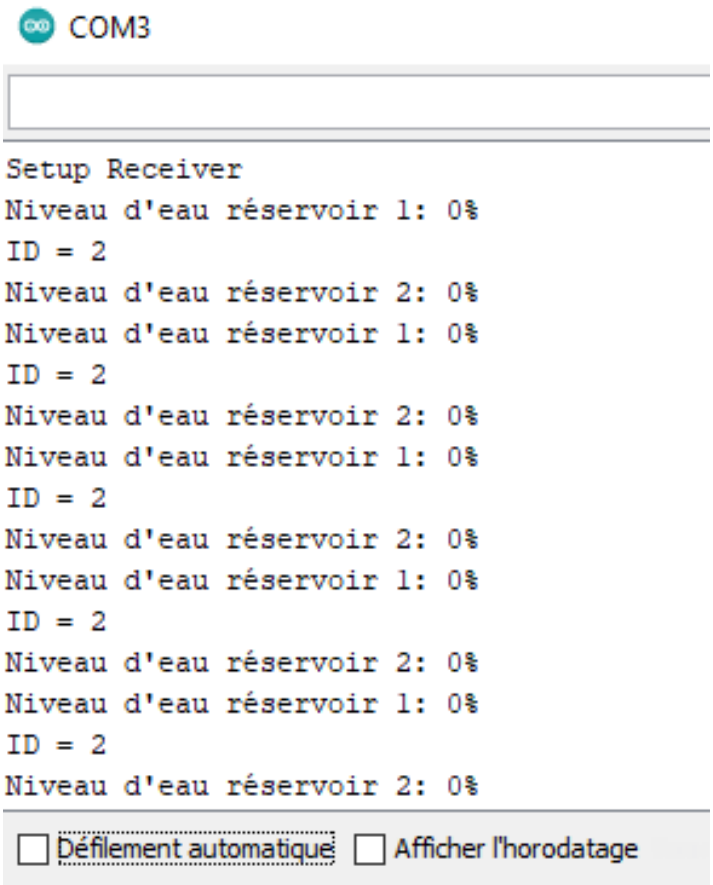
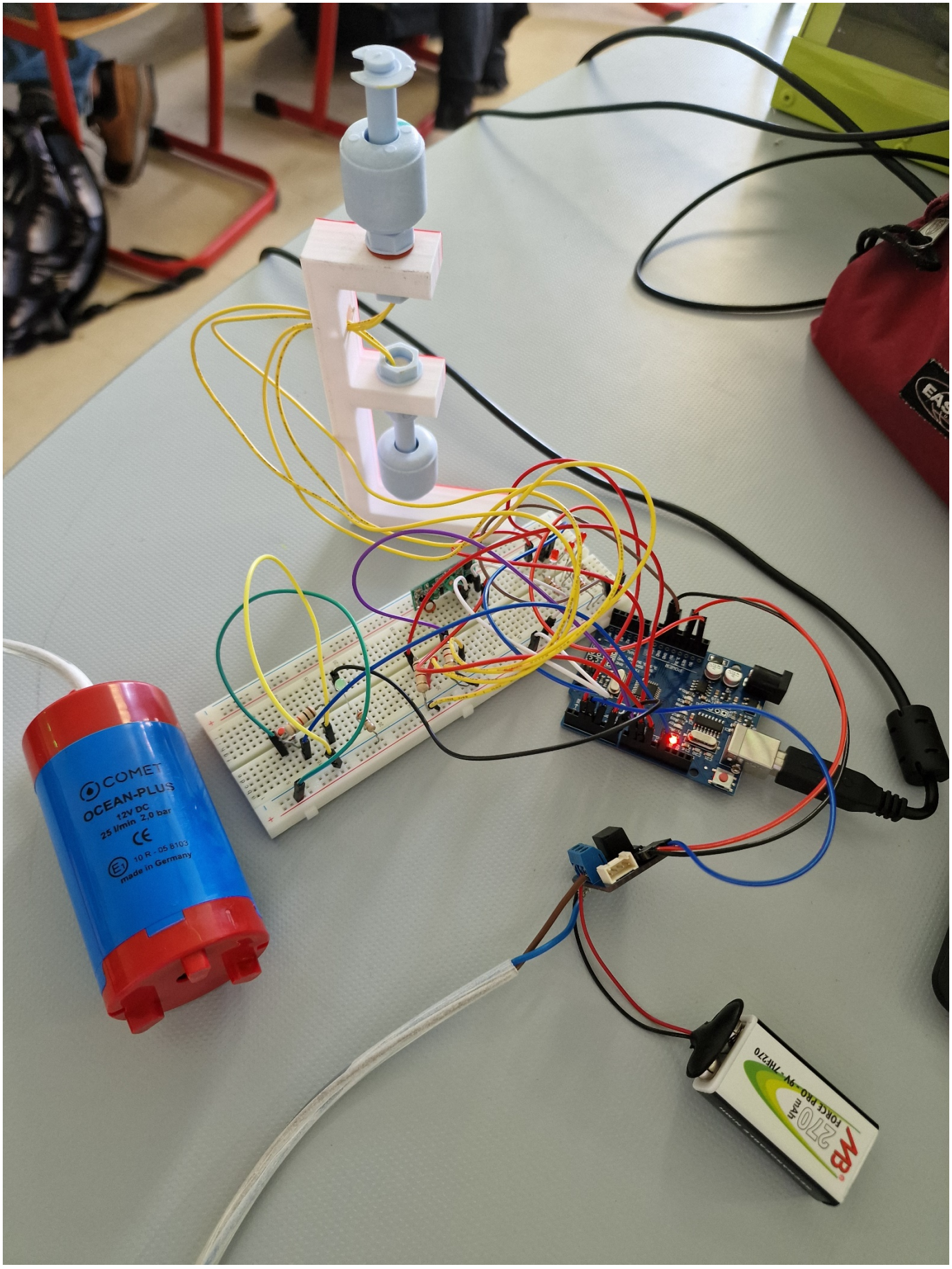
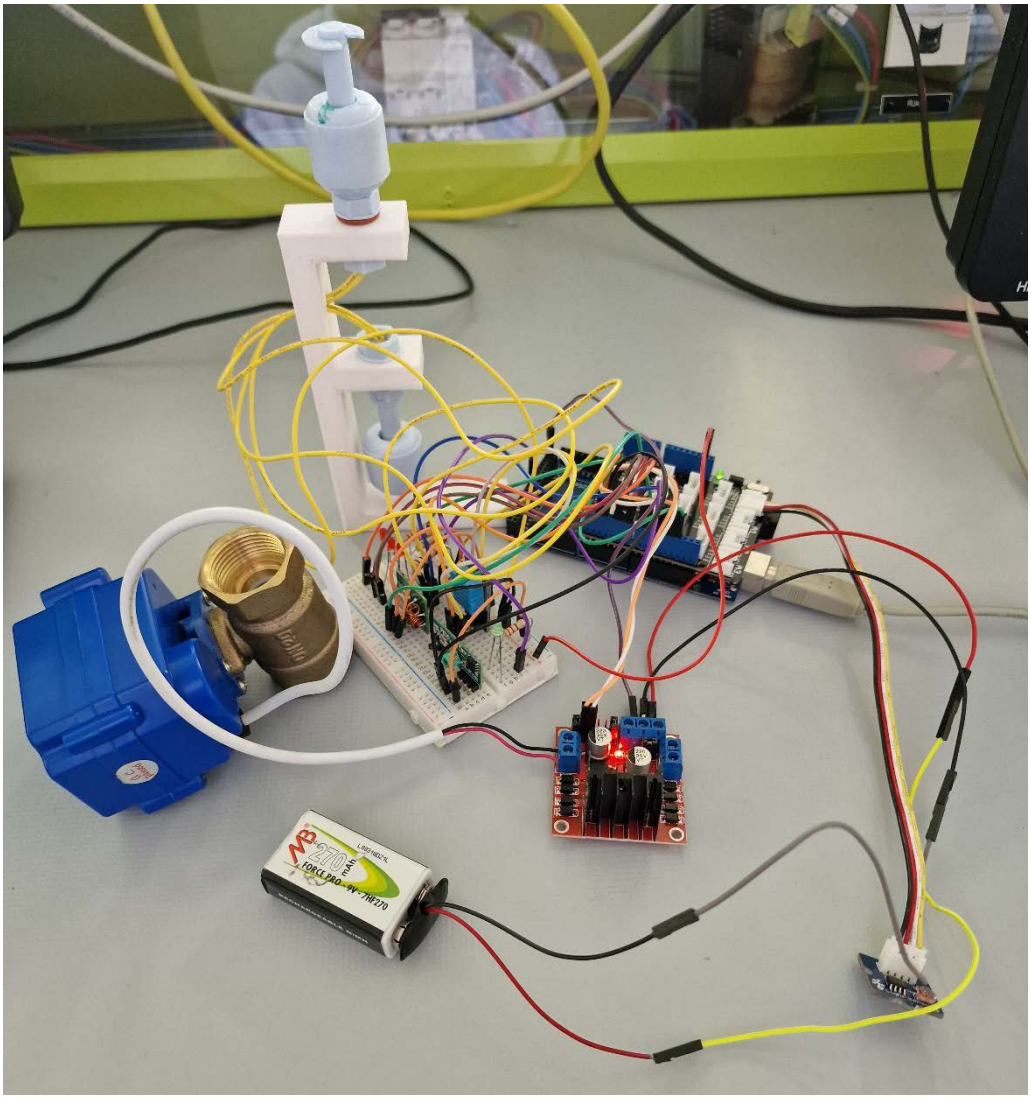


Photo projet :

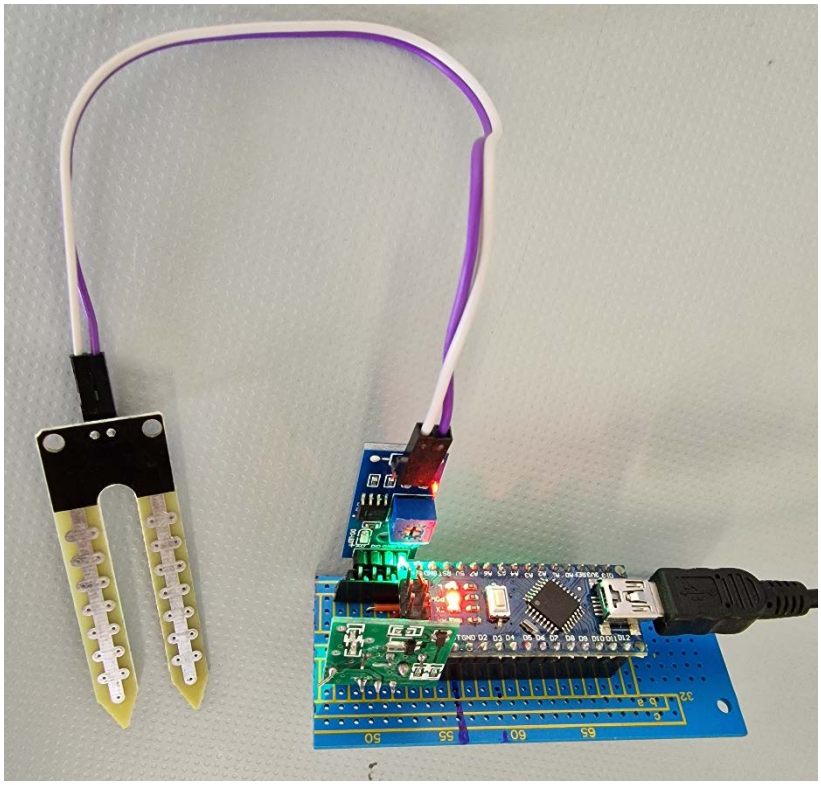
Réservoir n°1 :



Réservoir n°2 :



Capteur humidité du sol :



Liste du matériel :

- Carte Arduino Méga
- 2xCarte Arduino Uno
- 2x câble USB A vers USB B
- 2x électrovanne 12V
- Pompe 12V
- 2x Capteur humidité
- 4x Capteur Niveau d'eau
- Capteur de température
- Capteur de lumière
- 8x LEDS
- 3x Plaque Labdec
- 6 résistances 220 ohm
- Câbles male-male
- Câbles male- femelle
- 2xPile 9v
- 4xSupport de pile 9v
- Bouton
- Grove base shield
- Module HC-05
- 3x Module émetteur 433MHz
- 2xModule récepteur 433Mhz
- 2xArduino nano
- Câble pour arduino nano
- 2xModule relais
- 5x Résitances 1Kohm
- Pont H