

## LICENCE 3 INFORMATIQUE

---

Année : 2019-2020

### RAPPORT DE PROJET EN INTERNET DES OBJETS

# PlantCare

## POT DE FLEUR CONNECTÉ

BOUZIANE Hajar  
DECHAUMET Léo  
ZHANG Bingqin

Enseignant : M. HAMIDI  
M. OSMANI

Team : Les Licornes

20 janvier 2020

## SOMMAIRE

### I. INTRODUCTION

A. PRÉSENTATION GÉNÉRALE	Page 3
--------------------------	--------

### II. ÉTAT DE L'ART

A. CONTEXTE	Page 4
B. SOLUTIONS EXISTANTES	Page 4
C. OBJECTIFS	Page 5

### III. RÉALISATION

A. LES MATÉRIAUX	Page 6
i. LES COMPOSANTS	Page 6
ii. OBJETS A CONSTRUIRE	Page 7
iii. LE PRINCIPE	Page 8
B. LANGAGES ET LOGICIELS	Page 8
C. IMPLÉMENTATION	Page 8
a. LE POT DE FLEUR	Page 8
b. L'APPLICATION	Page 10
c. BASE DE DONNÉES	Page 11
d. FIREBASE	Page 12
e. FONCTIONNEMENT	Page 12
D. DIAGRAMME UML	Page 14
i. SOFTWARE	Page 14
ii. HARDWARE	Page 15
iii. ARDUINO : Diagramme de séquence	Page 16
iv. APPLICATION : Diagramme de séquence	Page 17

### IV. LES CONTRAINTES

A. NORMES ET RÉGLEMENTATIONS	Page 18
B. DIFFICULTÉS	Page 18
C. ORGANISATION	page 19

### V. CONCLUSION

A. CE QU'ON RETIENT	Page 20
---------------------	---------

IV. ANNEXE	Page 21
------------	---------

## **I. INTRODUCTION**

### **A. PRÉSENTATION GÉNÉRALE**

Dans le cadre de l'UE (Unité d'Enseignement) Internet des Objets, nous allons devoir réaliser un projet de groupe. L'objectif de ce travail est de découvrir Arduino, une carte composée de plusieurs composants électroniques dont un microcontrôleur (cœur de la carte Arduino). Carte que nous programmerons afin de créer notre tout premier objet connecté.

L'internet des Objets est une technologie qui existent depuis de très nombreuses années. Le principe de ces objets est qu'ils peuvent se connecter à internet et communiquer soit directement, soit indirectement entre eux.

Cette technologie a débuté en 2003 avec la commercialisation, par la firme Violet, du premier objet connecté : la lampe DAL connecté par Wi-Fi. Par la suite, les nombreux progrès dans les systèmes embarqués, la télécommunication ainsi que le traitement de données ont permis d'améliorer la performance des objets connectés. Ainsi, très présent dans notre environnement quotidien, l'Internet des Objets touche aussi de très nombreuses domaines d'activités comme les transports avec les voitures autonomes, la santé avec les tensiomètres ou encore la vente grâce aux caisses automatisés. Sans cesse en développement, le cabinet Gartner estime ainsi qu'en 2020 il existera plus de 30 milliards d'objets connectés dans le monde.

Pour ce projet de groupe, nous avons décidé de créer un pot de fleur couplé à une application Android. Pour mieux comprendre les démarches prises pour la réalisation de notre projet, il nous a été demandé de rédiger un rapport.

De ce fait, nous débuterons par une présentation du problème. Nous énumérerons, par la suite, nos besoins matériels et logiciels. Nous présenterons ensuite notre projet final. Et enfin, nous terminerons avec les contraintes et les difficultés que nous avons relevée.

## II. ÉTAT DE L'ART

### A. CONTEXTE

Ces dernières années, la tendance est aux objets connectés. Toujours plus variés et innovants, ils sont utiles, pratiques et ergonomiques. À l'heure où notre quotidien se remplit d'objets connectés et où on les associe le plus souvent à la télévision, aux ordinateurs, aux portables, quand est-il du jardin ?

D'après les études publiées par *Mon Eden*,  $\frac{3}{4}$  des Français ont un espace extérieur pour jardiner mais seulement 30% s'en occupe quotidiennement. En effet, certain type de plantes requièrent plus d'attention, de temps et de connaissances que d'autre car toutes n'ont pas les mêmes besoins. Cependant, nous ne possédons pas tous suffisamment de temps pour nous en occuper alors quelle serait la solution pour y remédier.

Notre objectif est donc de créer un accessoire de jardinage connecté qui permettrait à tout type de personnes (insérés dans la vie active ou s'absentant régulièrement) de profiter d'un espace vert sans en avoir à s'en occuper.

### B. LES SOLUTIONS EXISTANTES

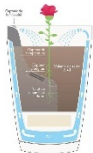
Aujourd'hui, on peut trouver sur le marché des accessoires de jardinage répondant aux besoins des clients. Parmi ces accessoires, on y trouve :

➤ Le Flower Power de l'entreprise Parrot.



- Il s'agit d'un accessoire en forme de lance pierre que l'on plante dans un pot. Il collecte des données grâce à ses capteurs de température, d'humidité, de luminosité et d'engrais pour ensuite les envoyer à une application qui avertira le client des besoins de sa plante.
- Inconvénient : Si le client est en déplacement pendant plusieurs jours, la plante ne pourra pas recevoir ce dont elle a besoin.

➤ Le Parrot Pot de l'entreprise Parrot.



- Il s'adapte à plus de 8 000 plantes différentes, en plus d'être connecté via un smartphone grâce à une application.
- Il est aussi doté d'une intelligence artificielle lui permettant de s'adapter pour concocter des programmes adaptés.
- Il est composé d'un pot au design simple alimenté par 4 piles.
- Il peut contenir jusqu'à 2,4 Litres de terre, et son réservoir d'eau jusqu'à 2,2 Litres.
- Ses 4 capteurs permettent d'analyser l'environnement d'une plante, permettant au pot d'agir en conséquence. Il peut ainsi arroser la plante grâce à une pompe.
- Inconvénients : Fixé à 149€,
- Le pot propose uniquement une connexion Bluetooth ce qui ne permet pas à l'utilisateur d'analyser l'état de sa plante que dans un périmètre restreint.
- Inconvénient pour remplir son réservoir, il faut démonter le pot et le nettoyer. Et enfin, ce pot de fleur connecté n'est plus en vente à cause de ses nombreux défauts de fabrication.

➤ Planty.



- Prix fixé à 99€.
- Il possède les mêmes fonctionnalités que le précédent
- Il peut notamment apprendre tout en surveillant son environnement avec ses capteurs afin de donner des conseils personnalisés pour l'entretien de la plante.

➤ Leo :



- Prix fixé à 99€.
- Le pot Leo, par rapport à ses congénères, possède des capteurs solaires, alimentant le pot si le temps est favorable.

### C. OBJECTIFS

Comme vu précédemment, les solutions sont nombreuses mais restent relativement excessives et, dans certains cas, insuffisantes. Ainsi, un accessoire à moindre coût, écologique ainsi que pratique aurait sa place dans le marché moyen et sera notre objectif.








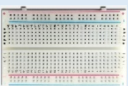
Nous avons donc pensé à fabriquer un petit boîtier contenant un ensemble de capteurs. Le boîtier sera à côté d'un pot de fleur et d'un réservoir d'eau qui lui sera relié à une pompe à eau.


L'application sera munie d'une base de données regroupant de nombreuses espèces de plantes. L'Arduino surveillera entre autres le niveau d'eau dans le réservoir, l'humidité dans le sol et la température.

### III. RÉALISATION

#### A. LES MATÉRIAUX

##### i. COMPOSANTS

Composant	Illustration	Description	Utilité	Prix
<b>Thermistor (Capteur de température)</b>		Renvoie la température de l'environnement où se trouve la plante (intérieur ou extérieur).	Si la température est élevée, il faudra placer la plante dans un endroit plus frais et inversement.	<b>~ 10€ Fournie</b>
<b>Capteur d'humidité</b>		Mesure le taux d'humidité dans la terre. L'eau est conductrice, donc plus le taux d'humidité est élevé, plus le courant électrique passera facilement entre les deux électrodes.	Plus le taux d'humidité est bas, plus il faudra arroser la plante.	<b>~ 9€</b>
<b>Capteur de niveau d'eau</b>		Indique le niveau d'eau d'un réservoir	Détermine si le réservoir est vide	<b>~6€</b>
<b>Moteur à courant continue</b>		Moteur qui transforme l'énergie électrique en énergie mécanique.	Servira au fonctionnement de la pompe à eau.	<b>Fournie</b>
<b>ESP32</b>		<del>Fonctionne comme la carte Arduino</del>	<del>Il va nous permettre de connecter notre objet connecté au Wifi.</del>	<b><u>Fournie</u></b>
<b>Arduino</b>		Carte composée d'un microcontrôleur.	Programmer le microcontrôleur pour créer l'objet connecté.	<b>Fournie</b>
<b>Files</b>		File permettant la transmission de l'énergie.	Files qui serviront à connecter les composants à l'Arduino.	<b>Fournie</b>
<b>BreadBoard</b>		Planche permettant de faire circuler le courant électrique (à l'intérieur circulation verticale et aux extrémités circulation horizontale)	Elle va nous permettre de brancher plusieurs composants en série.	<b>Fournie</b>

<b>Relais</b>		Comme l'Arduino ne peut délivrer que 5V, alors on utilise un relais pour contrôler des appareils qui demande plus de puissance.	Permet de faire tourner notre moteur en utilisant des piles sans abîmer la carte Arduino.	<b>Fournie</b>
---------------	---	---	---	----------------

Nous avons voulu utiliser l'ESP32 pour connecter notre pot de fleur au wifi afin de pouvoir interagir avec notre application Android. Cependant, aucun de nous trois n'a réussi à se connecter à l'ESP32. Nous obtenons toujours la même erreur suivante « A fatal error occurred : failed to connect to esp32 ».

## ii. Objets à construire

### Pour la pompe à eau

Composant	Illustration	Utilité	
<b>Moteur à courant continu</b>		Il sert à faire tourner l'hélice pour faire circuler l'eau du réservoir vers la plante.	<b>Fournie</b>
<b>Piles</b>		Les piles vont alimenter le moteur.	<b>Fournie</b>
<b>Relais</b>		Permet de faire tourner notre moteur en utilisant des piles sans abîmer la carte Arduino.	<b>Fournie</b>
<b>Bouchon de bouteille</b>		Former l'hélice qui va servir à pomper l'eau.	<b>Recyclé</b>
<b>Tuyau</b>		Fera circuler l'eau du réservoir à la plante.	<b>~6</b>
<b>2x boîtes de ferrero rocher</b>		1 pour le réservoir 1 contenant le circuit électrique	<b>Fournie</b>
<b>Pistolet à colle</b>		Permettra de fixer tous les matériaux qui formeront la pompe à eau	<b>Fournie</b>

### iii. LE PRINCIPE

Grâce aux composants électroniques mentionnés plus tôt, notre objet connecté collectera et analysera les données récupérées par les capteurs et ainsi prendre des décisions appropriées. Les résolutions prises seront comparées aux données stockées dans une base de données qui regroupe un ensemble de plante et leurs besoins. Entre autres, si le taux d'humidité dans la terre est trop faible, alors la pompe pourra être s'activée pour arroser la plante.

~~Notre pot connecté sera muni d'une carte ESP32 qui, une fois programmée, permettra au pot de se connecter au wifi. Par conséquent, il pourra ainsi envoyer des notifications à l'utilisateur, via une application, par exemple :~~

- ~~• Alerte, si la température ambiante est trop élevée.~~
- ~~• Demande, si le niveau d'eau dans le réservoir est trop faible.~~

## B. LANGAGES ET LOGICIELS

Pour programmer notre pot de fleur connecté, nous avons utilisé le logiciel Arduino IDE et nous avons utilisé le langage Arduino pour pourvoir manipuler les données envoyées par les capteurs.

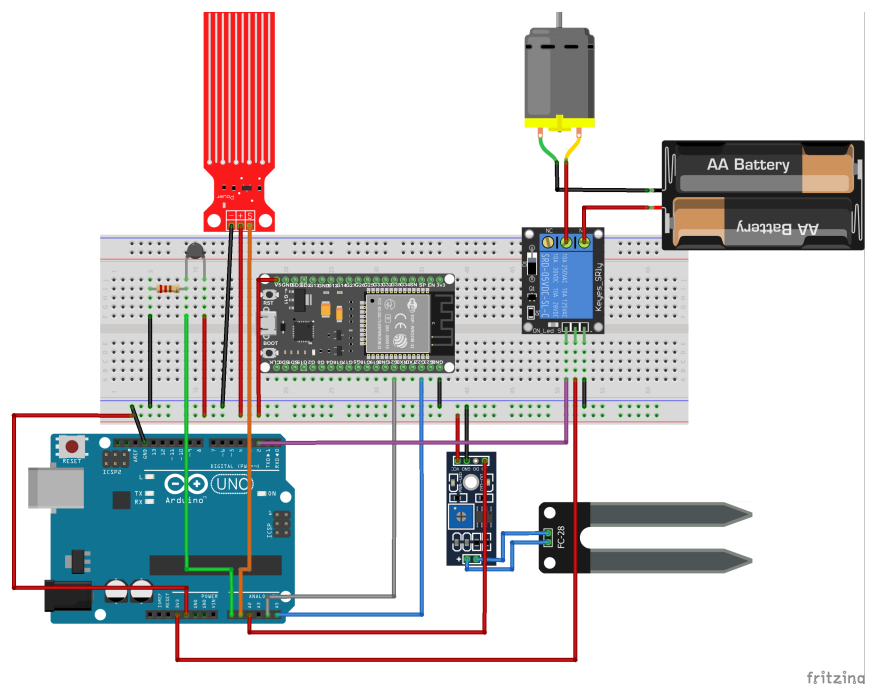
Notre application a été programmée en Java et XML sur le logiciel Android Studio et possède une base de données répertoriant des plantes.

Et enfin, comme nous n'avons pas pu utiliser l'ESP32, nous avons implémenté un code en Python, pour envoyer des données depuis Arduino à Base de données et Base de données à Arduino.

## C. IMPLÉMENTATION

### a. LE POT DE FLEUR

La première étape de la création de notre pot de fleur a été le branchement des capteurs et du moteurs. Pour ce faire nous nous sommes aidés du document « Starter Kit for Arduino » pour le réaliser et nous avons opté pour le branchement suivant :



Nous avons mis sur ce sketch l'ESP32 pour montrer le branchement qu'on aurait effectué si on avait pu l'utiliser.



La seconde étape consiste à récupérer les valeurs renvoyées par les capteurs pour ensuite les manipuler. De manière général, les capteurs renvois des valeurs qui sont comprises entre 0 et 1024. Il nous a donc fallut, dans certains cas, convertir ces valeurs.

➤ Concernant le thermistor :

Il nous a fallu utiliser la formule de **Steinhart-Hart** pour pouvoir convertir la valeur renvoyée par le thermistor en degré Celsius. La formule est la suivant :

$$T = 1 / (A + B(\ln R) + C(\ln R)^3)$$

Avec T la température en kelvin, R la résistance électrique (en ohms) du thermistor et A, B, C les coefficients de Steinhart-Hart qui ont pour valeurs (avec notre thermistor) :

**A = 0.0011129148;**  
**B = 0.000234125;**  
**C = 0.0000000876741;**

Ainsi, pour obtenir la température, on a obtenu le code suivant :

```
66 //TEMPERATURE
67 int bit_temperature = analogRead(TEMP);
68 double R = log(1000.0*(1024/bit_temperature-1));
69 double temperature = 1/(a+b*R+c*R*R*R);
70 temperature = temperature - 273.15;
71 Serial.print("temperature ");
72 Serial.println(temperature);
73
```

Ce bout de code consiste à lire la valeur récupérer par le pin TEMP = A0. Ont convertie la valeur en ohm puis on calcul la température en kelvin que l'on convertie en degré.

➤ Concernant le capteur niveau d'eau :

```
74 //NIVEAU EAU
75 double niveau = analogRead(LEVEL);
76 double niveau1 = (niveau*100)/analogMaxNiv;
77 Serial.print("niveau ");
78 Serial.println(niveau1);
79
```

Comme précédemment, on lit stocke dans une variable la valeur récupérer par le pin LEVEL = A1. Nous avons convertie la valeur en pourcentage pour pouvoir envoyé un valeur qui soit compréhensible vers la base de donnée de l'application.

➤ Concernant le capteur d'humidité :

```

80 //HUMIDITE
81 double humidite = analogRead(HUM);
82 humidite = 100-((humidite*100)/analogMaxHum);
83 Serial.print("humidite ");
84 Serial.println(humidite);
85

```

Ici aussi, on stocke dans une variable la valeur récupérée par le pin HUM = A2. Puis on la convertit en pourcentage.

Une fois l'ensemble des valeurs récupérées, nous devons savoir quand activer notre pompe à eau. Pour cela nous avons décidé d'activer cette dernière quand (si le réservoir n'était pas vide) :

- La température ambiante était supérieure à celle que pouvait supporter la plante et que la terre était sèche.
- La température ambiante était correcte mais que la terre était sèche.
- La température était trop faible (mais pas négative au risque que l'eau ne gèle) et que la terre était sèche.

Si une des conditions est vérifiée, alors on envoie du courant au moteur pour qu'il puisse activer la pompe à eau. On obtient ainsi le code suivant :

```

130 void lancement_moteur(int temp, int humi, int niv){
131     if(niv >= analogMinNiv){
132         if((temp >= Tm) && (humidite <= Hm)){ //Si il reste suffisamment d'eau pour un arrosage
133             //s'il fait trop chaud et que la terre est sèche, on arrose
134             digitalWrite(MOT, HIGH);
135             delay(3000);
136             digitalWrite(MOT, LOW);
137         }
138     } else if(((temp >= Tm) && (temp <= Tm)) && (humidite <= Hm)){ //Si la temperature est normal mais que la terre est sèche, on arrose
139         digitalWrite(MOT, HIGH);
140         delay(3000);
141         digitalWrite(MOT, LOW);
142     } else if(((temp <= Tm) && (temp >= 0)) && (humidite <= Hm)){ //Si la temperature est très basse mais pas négative (risque de geler) et que
143         digitalWrite(MOT, HIGH);
144         delay(3000);
145         digitalWrite(MOT, LOW);
146     }
147     else
148         //Serial.println("Pas besoin d'arroser !"); //Sinon on arrose pas
149         return;
150 }
151 else
152     //On ne fait rien rien pour ne pas habiller la pome tant que le réservoir n'est pas plein
153     //Serial.println("Niveau d'eau trop faible pour activer la pompe!");
154     return;
155 }

```

## b. L'APPLICATION

Afin de créer une interaction entre l'utilisateur et l'Arduino, nous avons choisi de faire une interface graphique (sous la forme d'une application mobile). Cette application fournira les informations récoltées par l'Arduino (valeurs des capteurs), ainsi elle servira de dashboard.

L'application disposera entre autres des fonctionnalités suivantes :

1. Choix d'une plante
  - L'utilisateur doit choisir une espèce de plante déjà présente dans la base de données (cf base de données), à l'aide d'une barre de recherche. Afin de faciliter ce choix, l'application doit

proposer une liste de plantes « candidates » grâce aux premières lettres tapées par l'utilisateur, comme sur un moteur de recherche.

2. Envoie d'informations à la carte Arduino

-Une fois le choix de la plante validé, les informations sur la plante nécessaires pour la carte Arduino (cf plateforme IOT) sont envoyés par l'application à firebase.

3. Réception et affichage des relevés

L'application possédera une vue pour afficher les relevés réalisées sur la plante. Sur le même écran on aura donc les informations suivantes la température, le taux d'humidité de la terre qui seront affichées, avec des alertes si le taux d'humidité ou la température ne sont pas adéquates pour la plante. Et une alerte est affichée si le réservoir d'eau est presque vide.

Pour concevoir cette application, il a fallu programmer en java et en XML en exploitant le logiciel Android Studio.

### c. BASE DE DONNÉES

Le SGBD est de type sqllite, et la base de données est installé en local sur le téléphone dès le téléchargement de l'application. La base de données sql comportera les informations sur les besoins de chaque espèce de plante. Elle sera composée d'une table unique.

Plante
<u>Nom</u>
TempExtremeMin
TempExtremeMax
HumiditeMin
HumiditeMax

On aura ainsi pour chaque plante les informations suivantes :

colonne	type	contrainte	description
<u>Nom</u>	Chaîne de caractère	Clé primaire	Le nom de l'espèce
TempExtremeMin	réel	- non NULL	Intervalle de température que l'espèce peut supporter
TempExtremeMax	réel	- non NULL	
HumiditeMin	réel	- Pourcentage (compris	L'intervalle du degrés d'humidité de la

		entre 0 et 100) - non NULL	terre. Le taux d'humidité de la terre doit toujours être dans cet intervalle.
HumiditeMax	réel	- Pourcentage (compris entre 0 et 100) - non NULL	

#### d. FIREBASE

Pour la connexion entre l'Arduino et l'application, nous avons utilisé **firebase realtime database**, qui est un outil permettant de stocker temporairement des données (variable temporaire). L'URL de notre base de données est <https://careplant-bad6a.firebaseio.com/.json>. Les données sont stockées en JSON dont le schéma est donné ci dessous :

```
{
  "capteurs": {
    "humidite": 0.1,
    "niveau": 2,
    "temperature": 26.45,
  },
  "info": {
    "HumiditeMax": 40,
    "HumiditeMin": 20,
    "temperatureMax": 10,
    "temperatureMin": 20,
  }
}
```

#### e. FONCTIONNEMENT

D'un côté, nous utilisons l'API de ce service depuis l'application. L'application écrit dans le champ « info », en remplissant les caractéristiques de la plante sélectionnée. L'application lit dans le champ capteurs pour récupérer les données de l'Arduino.

De l'autre côté, l'Arduino écrit dans le champ « capteurs » les valeurs récupérées par les capteurs, et lit dans le champ « info » les caractéristiques de la plante, afin d'arroser selon le résultat de la comparaison entre les informations écrites et lues.

Pour raison de simplicité et comme les opérations réalisées par l'Arduino sur firebase sont élémentaires, nous n'utilisons pas l'API de firebase depuis l'Arduino, mais directement avec les requêtes de type PATCH, sur l'URL <https://careplant-bad6a.firebaseio.com/capteurs.json>. La carte Arduino ne possédant pas de moyen de se connecter à internet, nous devons trouver un moyen de connecter le système Arduino à firebase. Comme dit auparavant, de nombreux problèmes rencontrés avec notre solution initiale, l'ESP32, nous ont poussé à trouver une solution alternative pour tester le code Arduino. Cette solution est décrite ci dessous.

#### La connexion physique Arduino-Firebase (proxy)

Nous avons décidé, pour nos tests, de garder l'Arduino connecté à l'ordinateur, et de faire passer les informations à envoyer sur le port serial. Nous avons créé un proxy en python (python 3), sur l'ordinateur, qui fait office d'intermédiaire entre l'Arduino et firebase. Le proxy communique simultanément avec l'Arduino grâce au port serial (module pyserial de python3) et avec firebase grâce à l'accès internet fournis par l'ordinateur (module requests de python3).

#### **Le protocole de communication Arduino → proxy**

L'Arduino envoie sur le port serial une chaîne de caractère de la forme "ChampAModifier valeur" (sans oublier l'espace entre les deux paramètre).

-ChampAModifier est une chaîne parmi "humidite", "temperature", "niveau".

-Valeur est une valeur numérique.

Le proxy peut ensuite modifier le champ correspondant (dans firebase) avec la valeur correspondante.

Exemple: "humidite 50"

#### **Protocole de communication proxy → Arduino**

Le proxy en python envoie sur le port serial une chaîne de 4 caractères de la forme "initialeInfo valeur" (sans espace entre les deux paramètre).

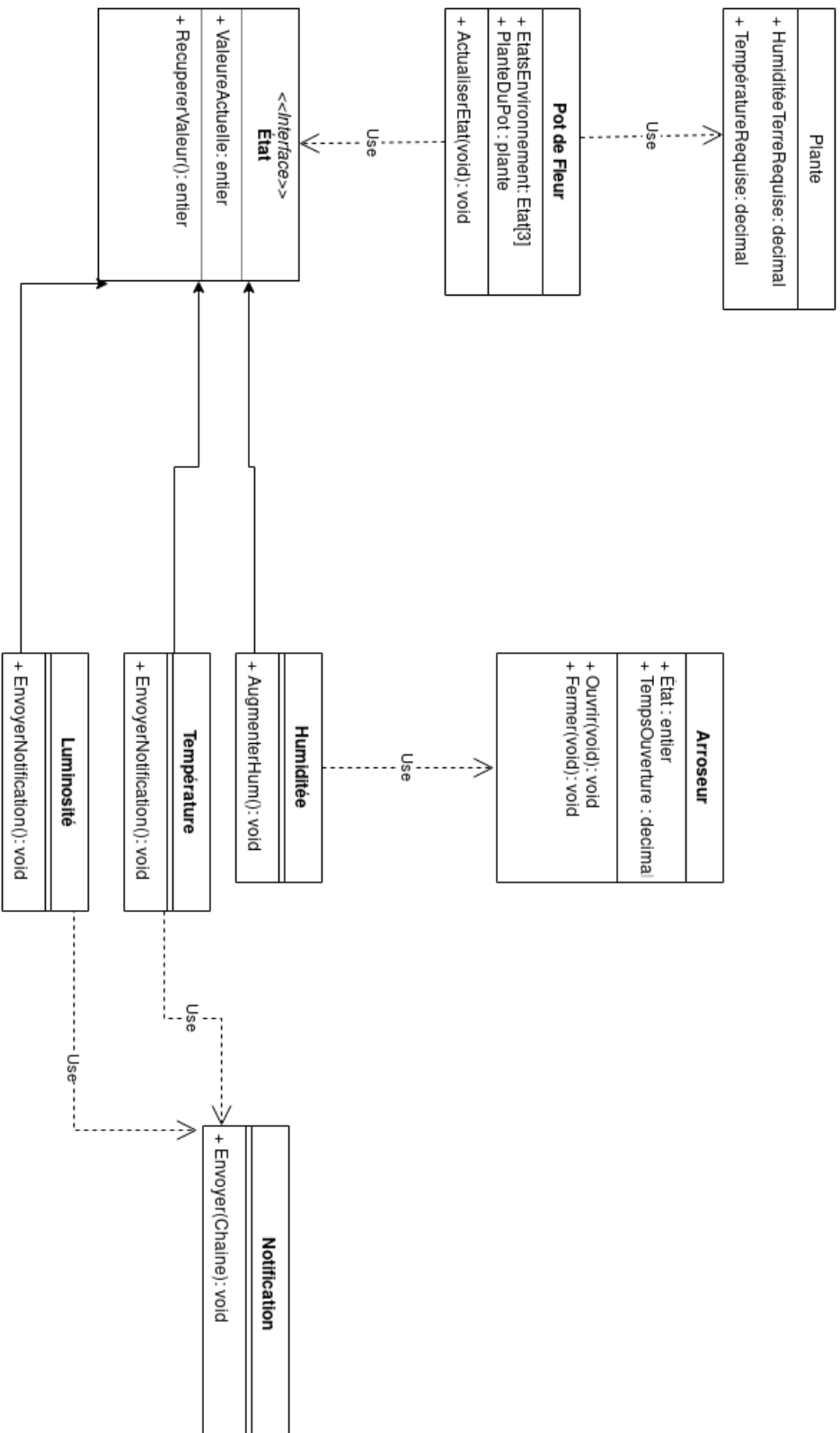
-InitialeInfo est une valeur parmi "HM", "HP", "TM", "TP" (respectivement humiditeMoins, humiditePlus, temperatureMoins, temperaturePlus).

-Valeur est une valeur numérique, de la taille de deux caractères.

Exemple : "HM05"

"TP40"

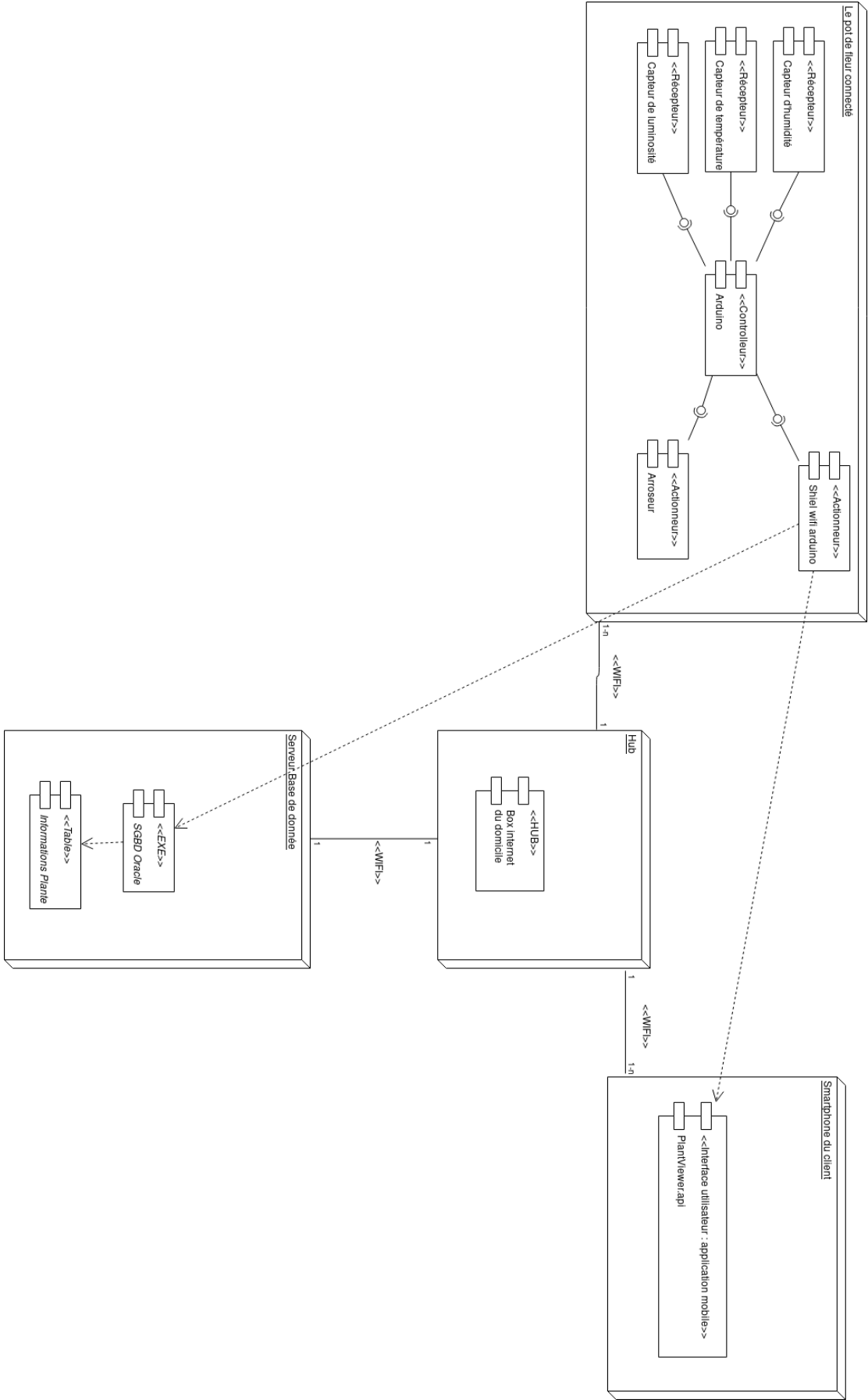
## Diagramme de classe



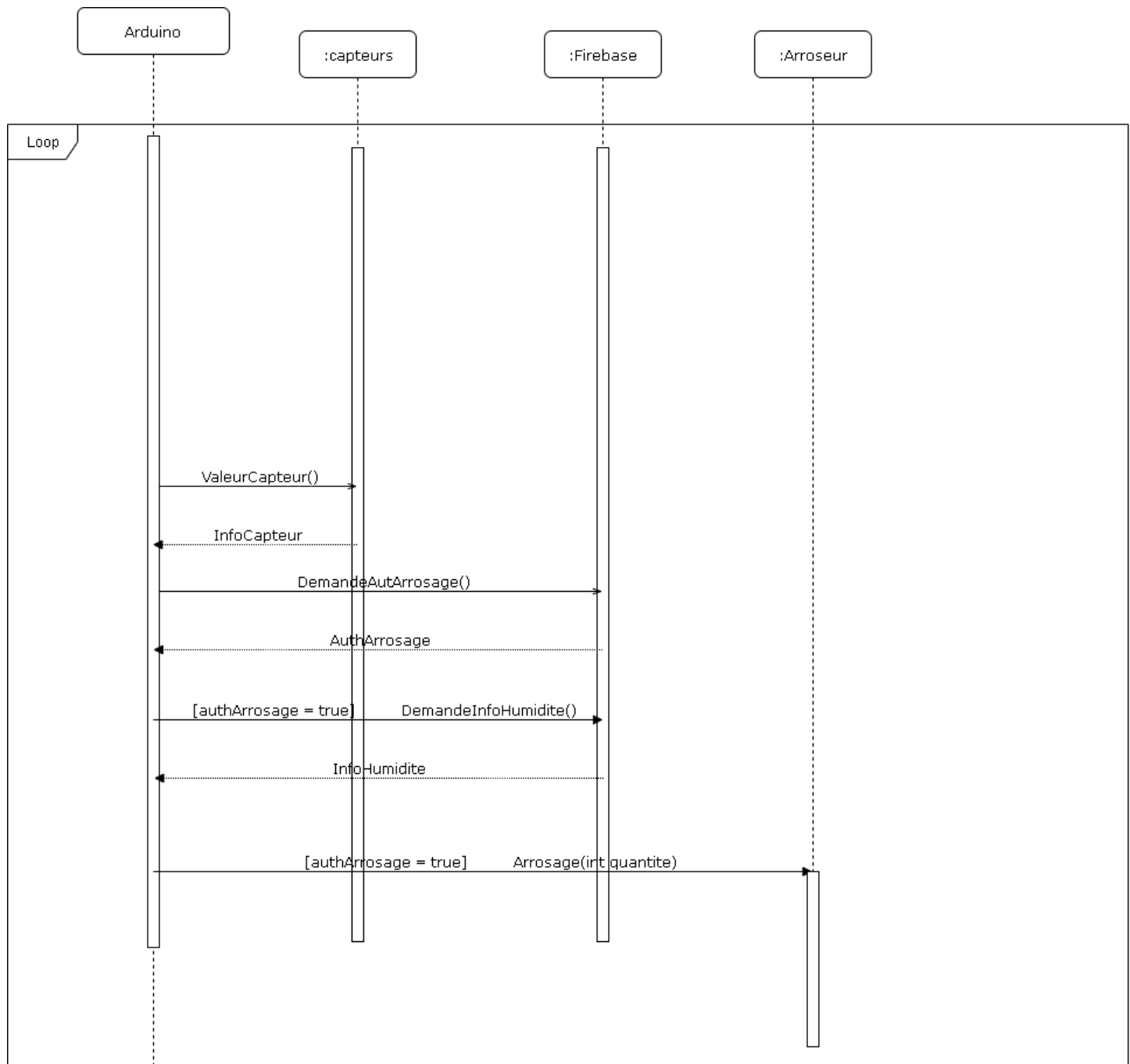
### D. DIAGRAMME UML i. SOFTWARE

ii. HARDWARE

Vue de déploiement

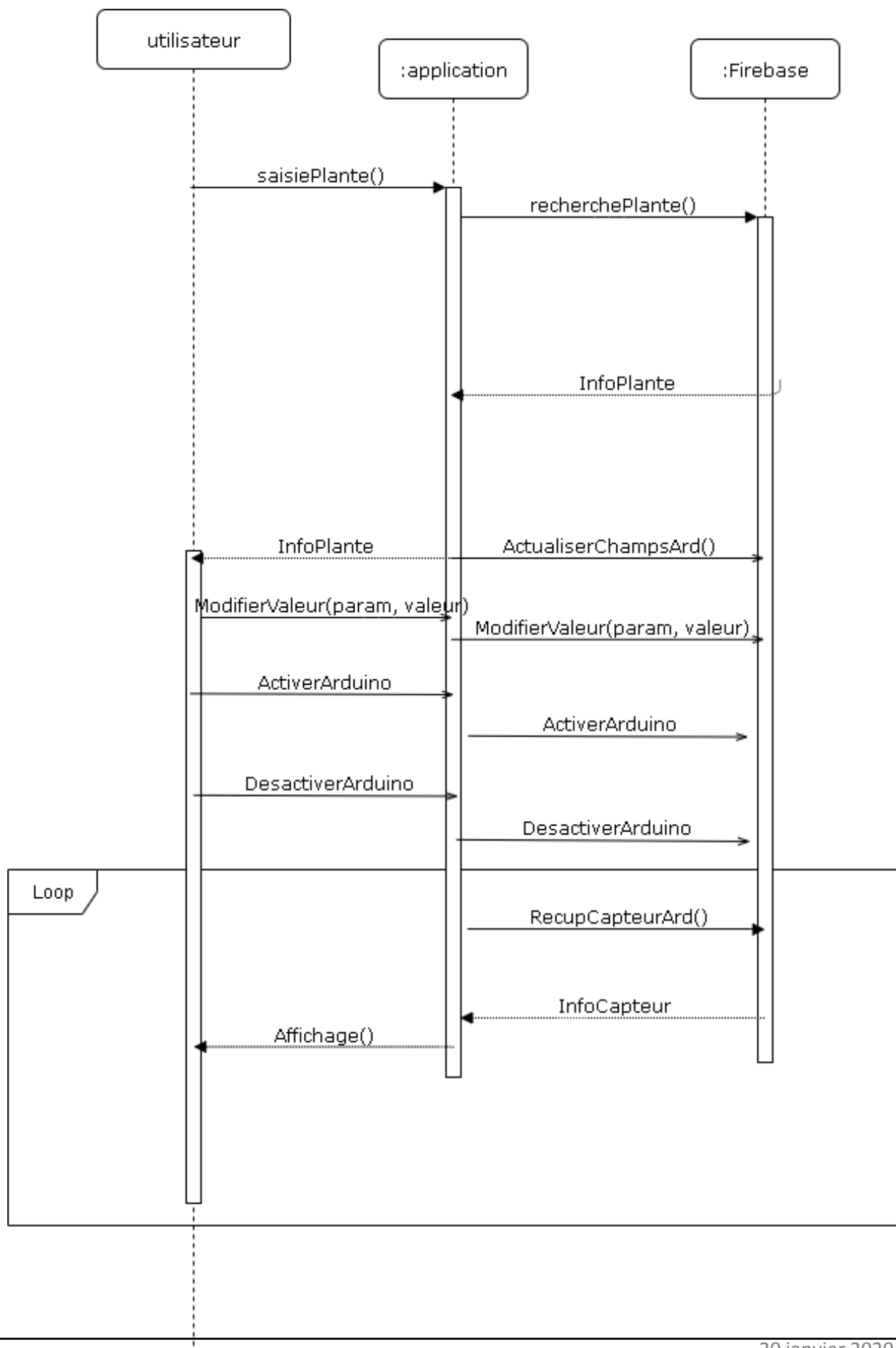


### iii. ARDUINO : Diagramme de séquence





iv. APPLICATION : Diagramme de séquence



## IV. CONTRAINTES

### A. NORMES ET RÉGLEMENTATIONS

Toute conception d'objet connecté doit respecter des normes et des réglementations afin que son utilisation soit rendue légale. Voici une liste non exhaustive des conditions à respecter dans le cas de notre projet :

- L'accrochage de jardinières aux fenêtres doit respecter les dispositions de l'article 96-2 du règlement sanitaire départemental type qui prévoit que « les objets et plantes ainsi que les fenêtres ne doivent pas créer d'insalubrité pour les passants et les occupants des immeubles riverains ».
- Un pot de fleur ne doit pas dépasser une taille trop grande si l'utilisation se fait sur un balcon car ce dernier ne peut supporter un poids supérieur à 350 kg/m<sup>2</sup>. Le calcul du poids prend en compte celui du pot et de la terre en sachant qu'une fois humidifiée, la terre devient beaucoup plus lourde.
- D'après l'article 226-1 du code pénal (atteinte à la vie privée), il est interdit de collecter des données à l'insu du consommateur et des les utiliser à d'autres fins que celles annoncées. De plus, les données requièrent une protection contre le piratage.

### B. DIFFICULTÉS

Nous avons rencontré de nombreuses difficultés pour réaliser ce projet. Dans un premier temps, les valeurs que renvoie les capteurs varient d'un test à un autre et sont parfois incohérentes. Par exemple, concernant le capteur d'humidité,

- quand il est planté dans de la terre sèche, les valeurs varient entre 923 et 985 (1023 à d'autres tests),
- quand le capteur est plongé dans l'eau, les valeurs varient entre 537 et 586 (autour de 300 à d'autres tests) (alors qu'on devrait être proche de 0),
- quand le capteur est dans le sol humide, on a des valeurs comprises entre 430 et 577 (ce qui est encore plus bas que quand le capteur est dans l'eau).

Concernant le capteur de niveau d'eau :

- quand le réservoir est vide, le capteur renvoie des valeurs <25 (que 0 à certains tests),
- quand on ajoute 2mm d'eau le capteur renvoie des valeurs comprises entre 410 et 420 (valeurs autour de 500 à d'autres tests),
- quand on est à la moitié du réservoir le capteur renvoie des valeurs entre 488 et 532 et
- quand il est plein, on a des valeurs qui ne dépassent pas les 600 (ou 620 à d'autres tests) (alors qu'on est censé aller jusqu'à 1023) on remarque que ces valeurs sont très proches de celles obtenues quand le réservoir est à moitié plein).

Il était donc difficile de pouvoir manipuler ces données mais après de nombreux tests qui ont duré plus longtemps, nous avons pu obtenir des valeurs plus exploitables et nous avons donc pu déterminer des constantes pour pouvoir convertir des valeurs en pourcentage et indiquer si le réservoir était vide ou non :

```

10  const int analogMaxTemp = 1024;
11  const int analogMinTemp = 0;
12  const int analogMaxHum = 1024; //sec
13  const int analogMinHum = 120;  //dans l'eau
14  const int analogMaxNiv = 1000; //plein
15  const int analogMinNiv = 100;  //vide
16

```

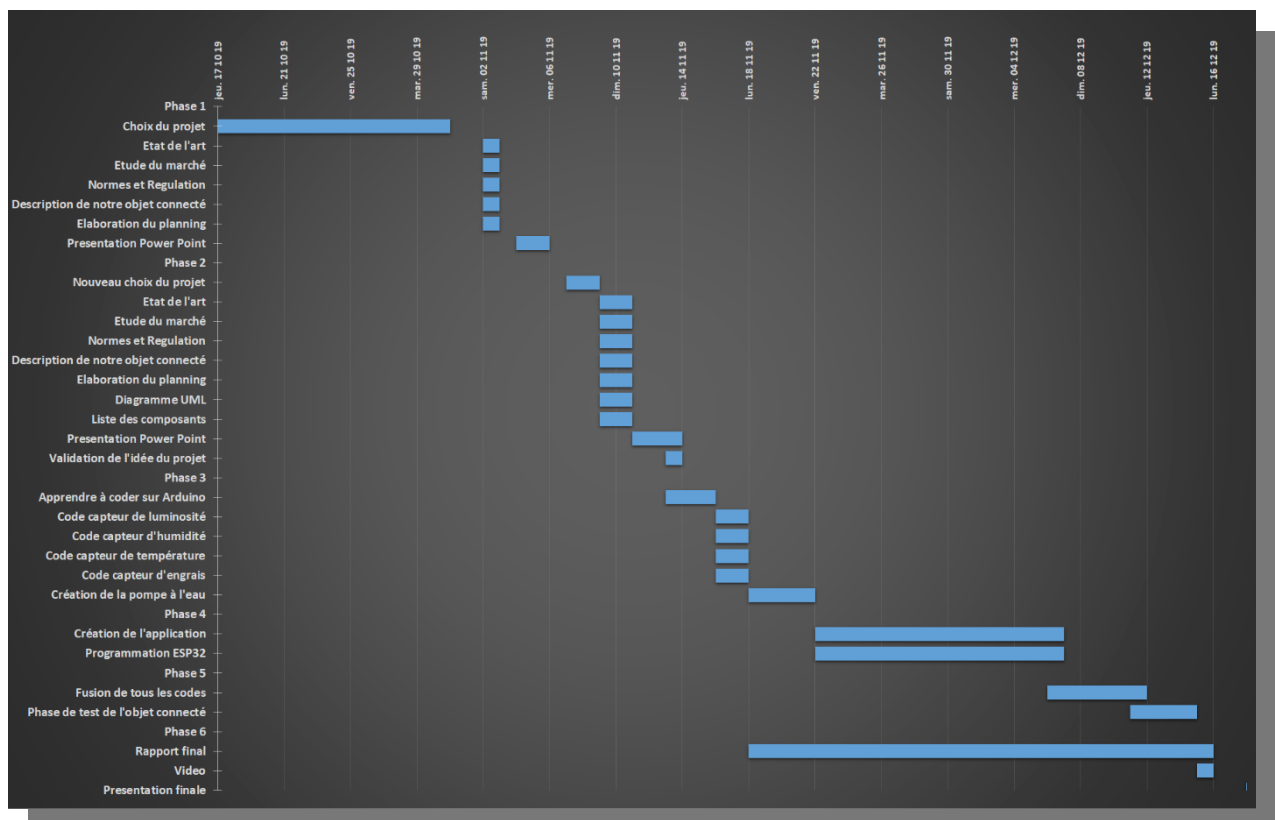
Enfin, nous avons rencontrés beaucoup de difficultés concernant la fabrication de la pompe à eau.

- la soudure des fils au moteur ne tenais pas. Il nous a fallu souder 4 fois les fils au moteur
- Il a aussi fallu changer de moteur car l'une des "ailes" est tombée.

## C. ORGANISATION

Une des difficultés auxquelles nous seront confrontée sera le temps. En effet, il nous en faudra pour maîtriser le langage Arduino, pour nous familiariser avec les différents composants pour pouvoir agir en cas d'imprévu nous amenant à travailler sur un composant plutôt qu'un autre, et pour apprendre à créer une application.

Pour cela nous avons créé un emploi du temps sous forme d'un digramme de Gantt avec toute les tâches à réaliser et ainsi gérer au mieux notre temps.



## **V. CONCLUSION**

### **A. CE QU'ON RETIENT**

Ce projet va nous permettre de découvrir les étapes de conception d'un objet connecté, d'enrichir nos connaissances dans les techniques de programmation et d'acquérir de nouvelles compétences à travers la création de notre pot de fleur connecté.

Le pot de fleur connecté est une solution optimale pour ceux qui n'ont pas la main verte et qui souhaiteraient avoir un peu de nature chez soi. Il s'agit d'une invention qui permet de garder longtemps une plante en vie sans réelle intervention humaine. C'est pour cette raison que nous avons décidé de travailler sur ce projet.

Réaliser un projet en groupe sera très enrichissant puisque chacun pourra exprimer son point de vue et proposer des solutions ou des améliorations qui rendront ce travail très intéressant.

## ANNEXE

<https://hacksmile.com/android-sqlite-search-searching-sqlite-database-in-android/>

Pour la fonction de recherche de plante dans la base de données.

[Sending Float values from Python to Arduino via serial communication](#)

Exemple de code qui nous a aidé à recevoir des données sur le serial port depuis l'Arduino.

[pySerial — pySerial 3.4 documentation](#)

La doc de pyserial, permettant d'envoyer et de recevoir des données de l'Arduino via le serial port depuis un script python3.

<https://www.youtube.com/watch?v=8CR40Dp1srw&list=PLMS9Cy4Enq5JnwAxe6Ao74qSTxxXjiw7N>

Créer une application sur Android Studio.

<https://www.youtube.com/watch?v=un3t9pKnpw>

Inspiration pour la pompe à eau.