

# Artificial Intelligence for Internet of Things and Robotics

-

## Rapport de projet

-

WaterTouch

*ABDRABO Khaled, HERVIER Thibault, BRIGNONE Jean*

## Introduction

Le but de ce projet est d'utiliser un hygromètre pour mesurer le taux d'humidité d'un pot de fleur et d'informer l'utilisateur du besoin en eau de la plante sujet, dont le nom scientifique est trouvé grâce à une API. On a aussi la possibilité de sauvegarder cette valeur sur un serveur pour un potentiel suivi. L'information est transmise à l'utilisateur grâce à un système de LEDs qui sera décrit ultérieurement.

## Répartition des tâches

Nous avons tous commencé par nous occuper du début du montage de l'Arduino. Nous pensions utiliser un tensiomètre, possiblement un haut-parleur ou un écran LCD. Après avoir passé du temps sur la compréhension du montage, nous avons décidé de nous concentrer sur le code et la partie WEB, en n'utilisant que des LEDs simples.

Les tâches ensuite ont été divisées de la sorte :

- Thibault avait pour tâche de développer le code pour l'Arduino et de s'occuper du montage.
- Khaled avait pour objectif de développer le back end de notre site.
- Le rôle de Jean a été de créer le front-end.

Nous avons ensuite cherché une solution pour essayer de lier l'Arduino au serveur par wifi, mais comme expliqué à la soutenance, nous avons fini par opter pour un câble Serial.






## Description du projet

Un scénario d'utilisation est le suivant :

L'utilisateur prend en photo sa plante, et entre la photo sur le site. 3 plantes sont proposées à l'utilisateur pour qu'il puisse l'identifier. Une fois l'identification faite, et le nom scientifique de la plante récupéré, l'utilisateur peut l'entrer dans la deuxième partie du site, ce qui mènera à l'affichage des détails liés aux besoins de la plante. Celui qui nous intéresse est l'humidité de la terre du pot.

L'Arduino vient avec un hygromètre, l'utilisateur le plante dans la terre, et branche également un câble directement à la plante qui servira de capteur. Après calibration de l'hygromètre pendant quelques secondes, l'utilisateur peut pincer la plante pendant 1 seconde pour sauvegarder les données d'humidité sur le serveur pour comparer les valeurs à celles récupérées sur l'API.

Ensuite, l'information du bien-être de la plante (concernant le taux d'humidité de la terre) sont transmises à l'utilisateur grâce au système de LEDs :

- ❖  : la plante est heureuse.
- ❖  : la plante commence à avoir soif.
- ❖  : la plante meurt de soif.
- ❖  : la plante se noie.
- ❖  : le taux d'humidité actuel est sauvegardé sur le serveur.

Pour éteindre la ou les LEDs allumées, il suffit de toucher la plante. Il ne s'agit que d'un interrupteur : si l'utilisateur touche la plante de nouveau pour allumer la LED, il n'est pas nécessaire de refaire la procédure sur le site internet. L'information du bien-être de la plante est visible en temps réel dessus et est mise à jour automatiquement toutes les 5 secondes.

## Fonctionnement

Les besoins en eau de la plante sont récupérés via la connexion à l'API "*OpenPlantBook*" après identification de la plante par photo via l'API "*PlantID*" puis envoyé à l'Arduino.

La communication Arduino-WEB est effectuée par l'intermédiaire d'un script Python (bibliothèque PySerial)

## Extensions (après implémentation de l'hygromètre)

Voici une liste de ce que nous aurions aimé implémenter en plus de ce que nous avons déjà :

- **[FAIT] Ajout d'un capteur capacitif : Transforme la plante en actionneur.**
  - Toucher la plante allume/éteint l'affichage LED.
  - Pincer la plante sauvegarde le taux d'humidité actuel sur le serveur.
- **Implémentation d'un shield Arduino Wifi : Suppression de l'interfaçage Arduino-WEB via Python**
  - Les communications se font directement depuis l'Arduino via des GET/POST
  - *"Cette méthode n'a pas été initialement choisie du fait de son coût (~30€) mais aussi car le temps de commande du composant et de son implémentation ne collait pas à notre deadline..."*
- **Modification du système LED**
  - Ajout d'un écran en lieu et place du système LED.
    - Toucher la plante allume/éteint l'écran
    - Les informations sont transmises par texte défilant ou emojis.

### OU

- Ajout d'un haut-parleur en lieu et place du système LED : la plante s'exprime oralement
    - Lorsqu'on la touche.
    - Toutes les heures si elle commence à avoir soif.
    - Tous les quarts d'heure si elle meurt de soif.
    - Immédiatement lorsqu'elle se noie.
- **Ajout d'un capteur de lumière : Comparaison avec les données de l'API**
  - Même principe que pour l'hygromètre dans l'ensemble

## Description des architectures et des configurations

### Logicielle

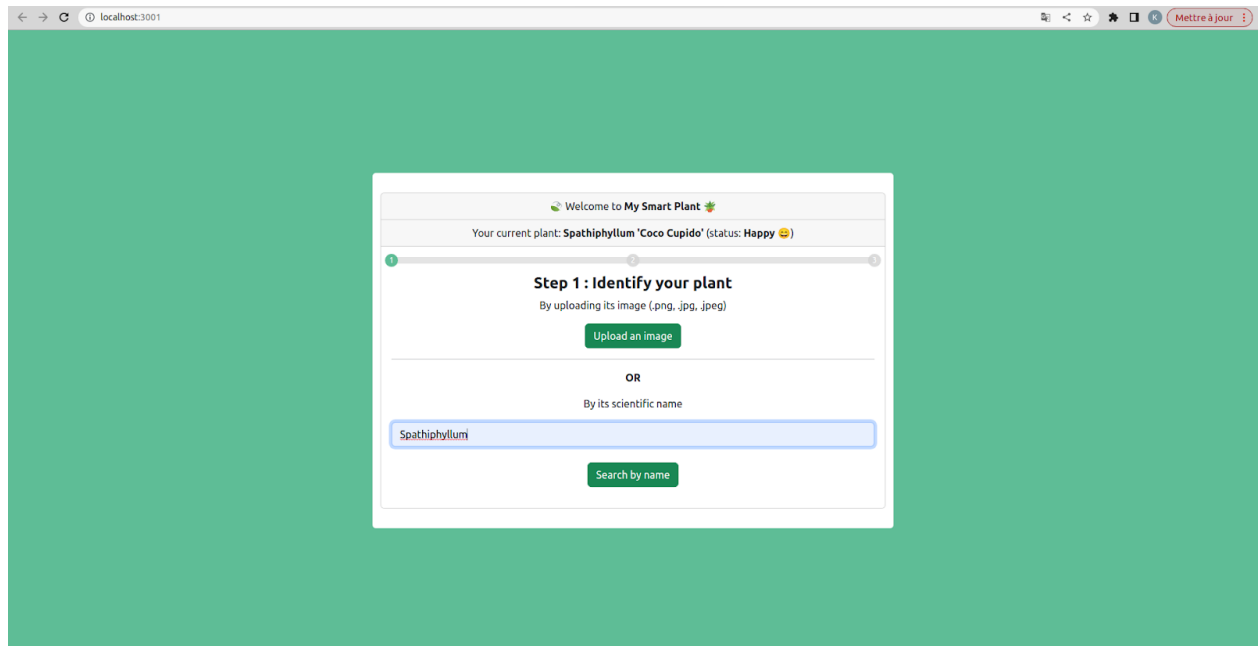
Dans cette section, nous décrirons notre API plus en profondeur, expliquerons brièvement le front-end, et enfin comment les 3 éléments de notre projet s'interconnectent.

Notre API est une simple application Node.js dans laquelle nous avons créé 6 routes principales qui aideront à faciliter notre travail des deux côtés, l'Arduino UNO et front-end.

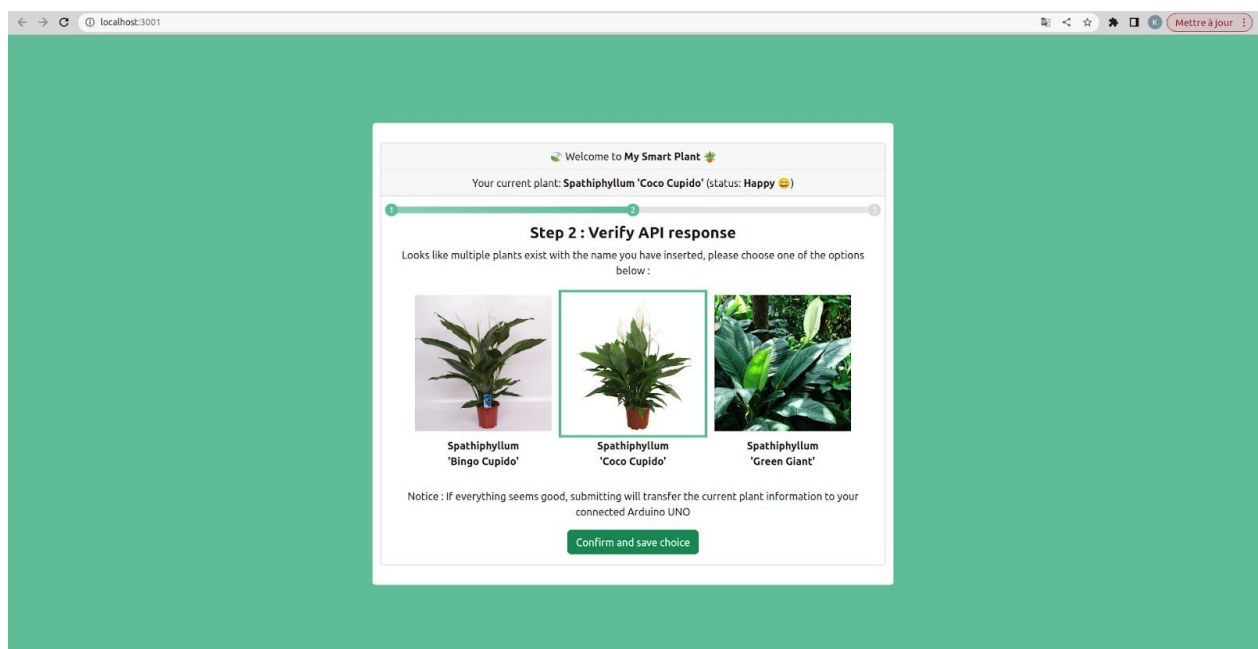
Voici la liste des routes (une collection Postman peut être également trouvée dans notre projet) :

- **POST /api/plant/photo/** : cette route prend en paramètre (dans son body) l'image d'une plante et, en cas de succès, renvoie un objet JSON contenant une liste de plantes possibles, avec leur nom scientifique, les images et, la probabilité que ça soit la vraie plante. Cette route utilise l'API "PlantID" pour obtenir les résultats sur la plante photographiée.
- **GET /api/plant/id** : cette route permet à l'utilisateur de rechercher sa plante en utilisant directement son nom. Cette route prend comme paramètre un nom de plante et renvoie une liste de noms scientifiques et d'images parmi lesquelles l'utilisateur peut choisir. Cette route s'appuie sur "OpenPlantBook" pour obtenir la liste des plantes avec leurs noms scientifiques.
- **GET /api/plant/stats** : cette route prend comme paramètre l'identifiant de la plante (tel que trouvé dans "OpenPlantBook") et, en cas de succès, renvoie toutes sortes de données utiles sur les besoins sur la plante. Cette route enregistre également les données renvoyées dans un fichier local sur notre API.
- **GET /api/plant/myplant** : cette route, lorsqu'elle est appelée, renvoie les détails de la plante enregistrée précédemment recherchés par l'utilisateur.
- **POST /api/plant/update** : cette route prend en paramètre le taux d'humidité actuellement détecté de la plante de l'utilisateur. L'Arduino UNO traite la valeur et renvoie l'une des 4 valeurs possibles, ce qui nous aide à indiquer facilement à l'utilisateur l'état de son installation. Bien évidemment, les données extraites sont enregistrées dans un autre fichier local sur notre API pour une utilisation ultérieure.
- **GET /api/plant/moist\_level** : cette route lit les données sauvegardées de notre fichier local et les renvoie, comme d'habitude, dans un objet JSON.

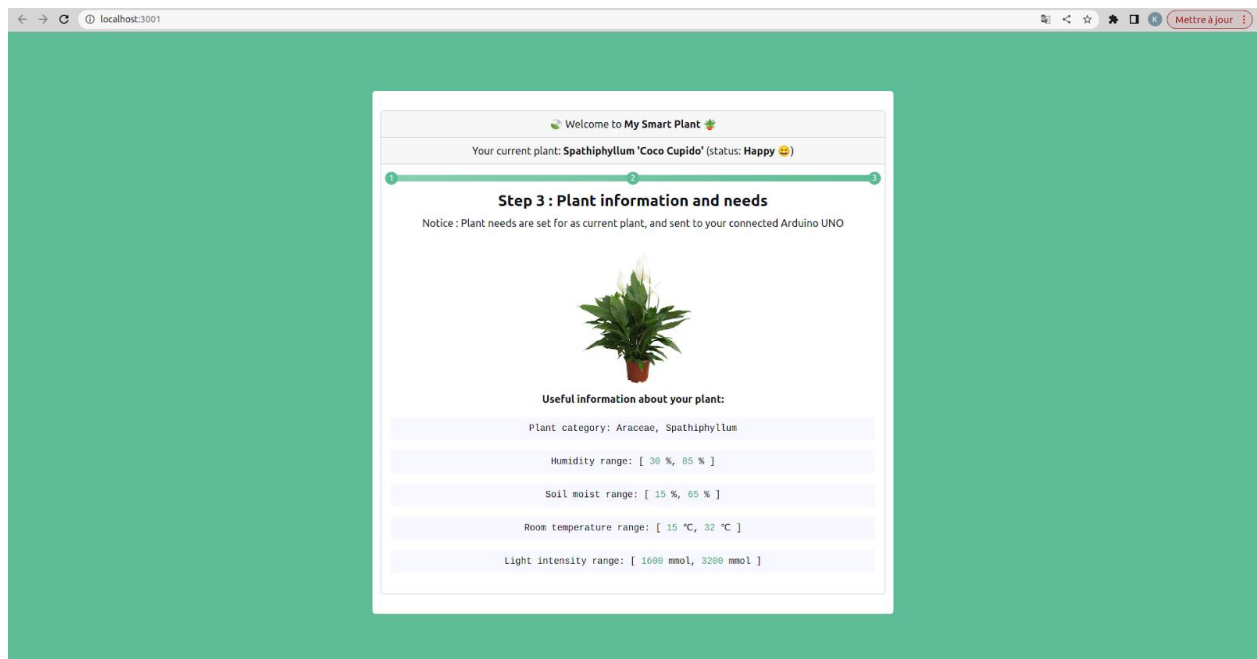
Notre front-end est une application web React.js. Il s'agit d'un formulaire en 2 étapes et qui s'appuie sur les routes préalablement détaillées pour guider l'utilisateur tout au long du processus de recherche et d'identification de sa plante et de suivi de son état.



La première étape consiste à rechercher la plante par son nom ou en téléchargeant une image. Une fois que l'utilisateur télécharge une image ou tape un nom de plante, il sera guidé vers la deuxième étape du formulaire.



Cette étape consiste à confirmer les résultats renvoyés soit par "OpenPlantBook" en cas d'image téléchargé, soit par "PlantID" en cas de recherche par nom. L'utilisateur a le choix entre 3 plantes. En faisant un choix, la plante choisie est enregistrée localement dans notre API locale, ses besoins sont affichés et son nom est mis à jour dans l'en-tête du formulaire.



L'Arduino UNO utilise également les données enregistrées de la plante pour comparer le niveau d'humidité actuel du sol avec ses besoins, qui peuvent être trouvés sur le fichier JSON local sur notre API.

Tout d'abord, l'Arduino envoie une requête GET (sur la route myplant) pour récupérer l'intervalle de niveau d'humidité du sol de la plante. Il la compare ensuite avec la valeur obtenue en temps réel par le capteur de taux d'humidité. Ensuite, il envoie une requête POST (sur la route update) avec une valeur entière qui représente : 0 si la plante est contente, 1 si elle a soif et a besoin de plus d'eau, 2 si elle est desséchée et 3 si la plante se noie.

Enfin, côté front-end, le composant d'en-tête envoie une requête GET (sur la route moist\_level) et se rafraîchit toutes les 5 secondes pour mettre à jour l'état actuel de la plante à l'utilisateur.

## Matérielle

### Fonctionnement du code de l'Arduino :

#### - Entrées:

- Le capacitive sensor (plante) sur le pin 4
- L'hygrometre sur le pin A0

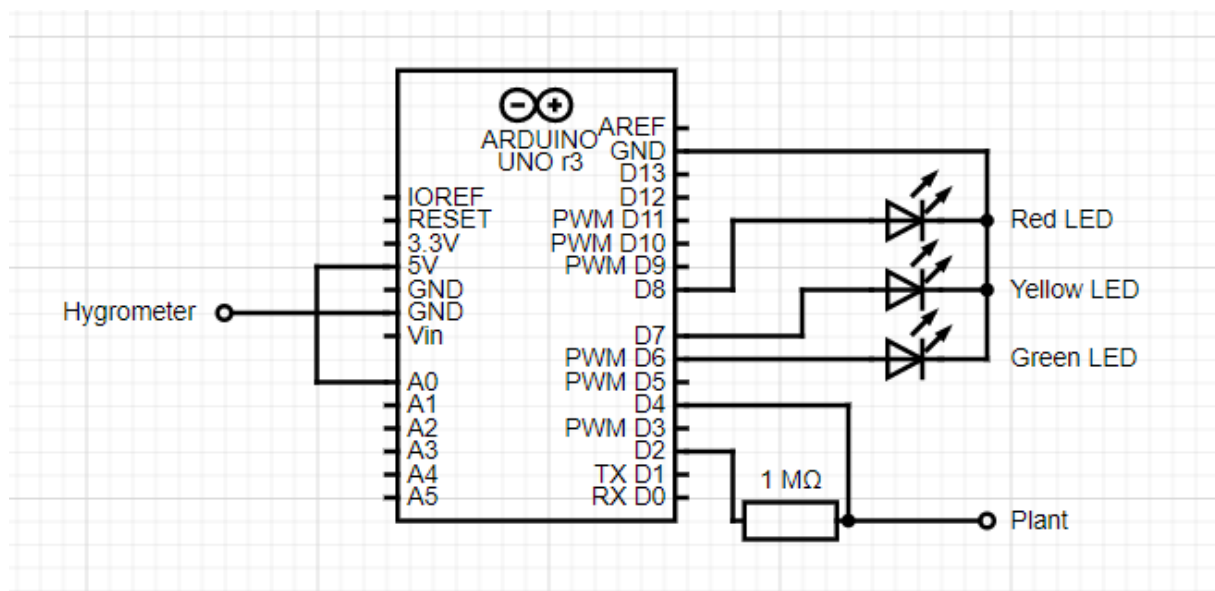
#### - Sorties:

- Les leds sur les pin 6, 7 et 8

- L'Algorithme :

1. On récupère la valeur du CS
2. On récupère la valeur de l'hygro
3. Si on ne connaît pas la fourchette d'humidité de notre plante, on la requête à l'API
  - a. On requête la valeur min
  - b. On récupère la valeur min
  - c. On requête la valeur max
  - d. On récupère la valeur max
4. On vérifie s'il y a interaction avec la plante
  - a. Selon le seuil du CS on renvoie: 0:pincée 1:touchée 2:rien
  - b. Et on modifie le comportement en fonction:
    - i. 0 : on requête la fourchette d'humidité à l'API et on fait clignoter les LEDS
    - ii. 1 : on change l'état des LEDS (on/off)
5. On vérifie le taux d'humidité du pot
  - a. On converti la valeur de l'hygromètre en pourcentage
  - b. On la compare à notre fourchette et on renvoie: 0:contente 1:assoiffée 2:déséchée 3:noyée
  - c. Si l'état de la plante a changé, on envoie l'information à l'API
  - d. On change l'affichage LED en fonction
6. On recommence

Ci-dessous un schéma représentant le montage de notre Arduino :



## Interfaçage Arduino-WEB via pySerial en Python

### - Les fonctions :

#### 1. api\_get(var)

- Prend le nom de la variable à requêter en entrée
- Fait la requête à l'API
- Renvoi la valeur de cette variable en sortie

#### 2. api\_post(val)

- Prend la valeur d'état de la plante en entrée (0:contente 1:assoifée 2:déséchée 3:noyée)
- Envoie la valeur à l'API
- Renvoie l'URL de la requête

### - L'Algorithme:

1. On récupère la liste des ports arduino
2. S'il y a des ports actifs, on les affiche
3. S'il y en a plusieurs on demande à l'utilisateur de choisir, sinon on prend l'unique port disponible
4. On se connecte à l'arduino via ce port
5. On boucle en continu :
  - a. On lit les données affichées sur le port Serial
  - b. S'il y a des choses à lire :
    - i. S'il s'agit des deux premiers échanges, on les considère comme phase de "test" car pour une raison étrange ils n'ont pas l'air d'être pris en compte
    - ii. Si on lit "min", on appelle api\_get() pour la valeur de min\_soil\_moist, on y concatène un '\$' comme stoppeur de lecture, on l'encode et on l'envoie à l'arduino
    - iii. Si on lit "max", on fait de même pour la valeur de max\_soil\_moist
    - iv. Sinon c'est que la valeur lue correspond à l'état de la plante, on vérifie donc s'il s'agit de '0', '1', '2' ou '3' et on l'envoie au serveur via api\_post()

## Matériel utilisé

- [Capteur] : 1 [Hygromètre](#) *Initialement nous avons essayé d'utiliser le graphite de crayons à papier, mais l'efficacité était moindre, nous avons donc commandé un véritable hygromètre.*
- [Capteur] : 1 Plante
- [Actionneur] : 3 LED (rouge, jaune, verte)
- [Capteur] : 1 Luxmètre (si extension)
- [Actionneur] : 1 Ecran LCD (si extension)
- [Actionneur] : 1 Haut-parleur + shield audio (si extension)



Ainsi que :

- 1 Arduino UNO
- 1 Bread-board
- 1 Résistance 1M (*pour le capteur capacitif*)
- 8 Câbles (*dont 1 pour le capteur capacitif, 4 pour le système LED et 3 pour l'hygromètre*)

## Intelligence

Les noms scientifiques des plantes sont récupérés sur une API, et leurs informations (besoin en humidité, lumière, etc..) sur une autre. La plante sujet sera identifiée grâce à une photo, et à un algorithme de Machine Learning qui tourne sur la 1ère API. La quantité d'eau nécessaire au bien-être de la plante sujet sera utilisée comme fourchette pour notre Arduino qui informera l'utilisateur que la quantité d'eau dans la terre sort, ou non, de cette fourchette.

API utilisées :

- Identification de la plante en en prenant une photo, envoyée sur l'[API](#) de détection
- Récupération des détails des besoins de la plante identifiée sur la deuxième [API](#)