Belkacem ZEHER
zeherbelkacem@gmail.com
0641915366

Formateur
Corentin HUTEAUX

Rapport du projet Chef d'œuvre

IOT BASED SMART FARMING

Or

Iot based smart irrigation system

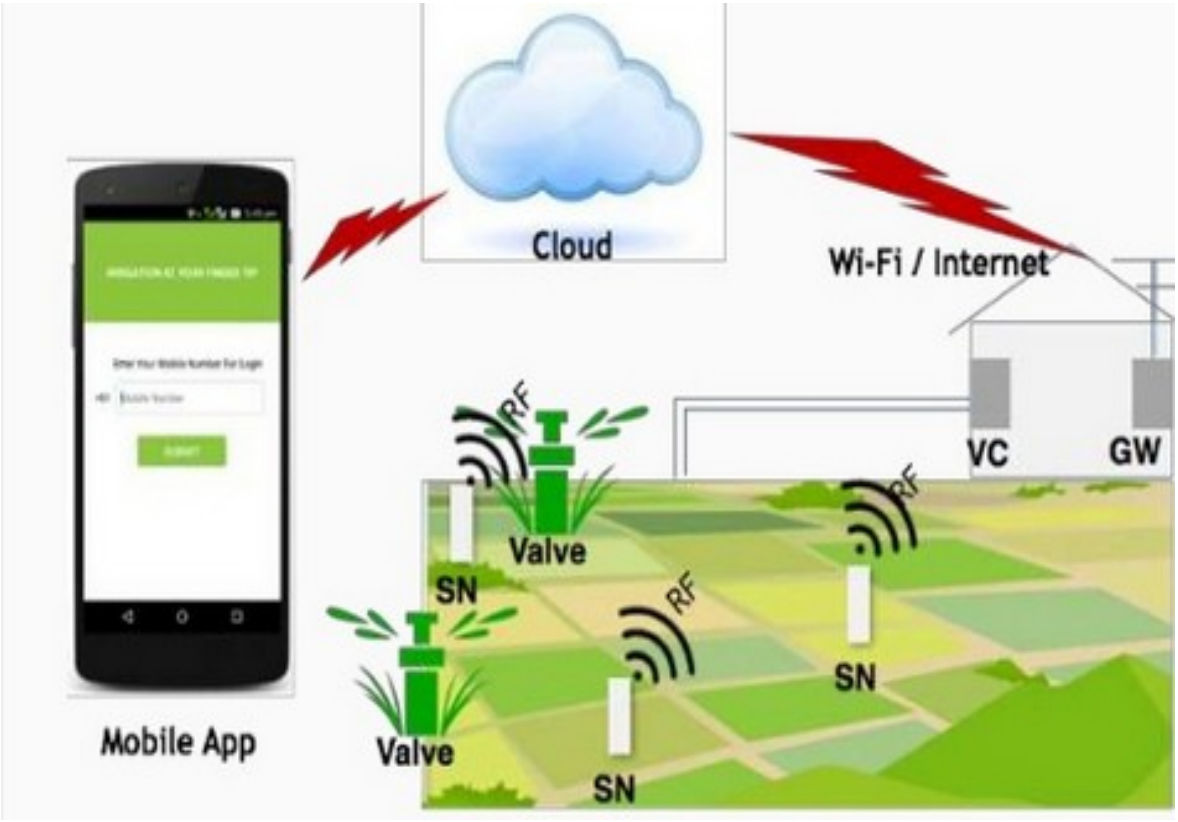


Table de Matières

Introduction

On connaît le plein essor de l'IoT dans les maisons connectées (smart home), ce vent en poupe on le retrouve aussi bien chez le public que chez les entreprises. Le jardin connecté est sur la même voie que la maison. D'après les média, un très grand public suit les émissions de jardinerie à la télévision et que ce marché pèse à lui seul près de 10 milliards d'euro rien qu'en France. Les fermes connectées ne sont pas en reste et ne font pas exception à la règle. Grâce au réseau de capteurs et une vision verticale, celles-ci peuvent accroître largement la qualité et la quantité des récoltes. Ce constat est aussi valable pour l'agriculture, les agriculteurs doivent faire face à des pénuries croissantes d'eau et à la disponibilité limitée des terres ainsi la fluctuation des coûts. La technologie propose des solutions innovantes d'agriculture intelligente pour aider les agriculteurs et les jardiniers à surmonter ces défis. Des solutions IoT pour l'agriculture permettent aux agriculteurs d'utiliser des capteurs/actionneurs, des passerelles intelligentes et des systèmes de surveillance pour collecter et analyser des informations et prendre des décisions plus éclairées.

Cueillir des fruits et des légumes ayant du goût demande de l'attention particulière dans la pratique d'arrosage. Comment et quand arroser est de meilleure façon doit être lié souvent et au préalable à la collecte et d'analyse de données de l'environnement du travail. Pour réussir sa récolte, il faut d'une part que les plants se soient dans un premier temps bien développés (densité du feuillage et hauteurs des tiges). Dans un deuxième temps, la qualité de la récolte dépend de la profusion des fruits et des légumes, de leur maturité et surtout de leur goût.

Des informations météorologiques sont essentielles pour cela et permettent de ne pas se tromper dans le soin apporté aux plantes. Si l'utilisateur ajoute des capteurs, il peut également obtenir la température et surtout l'humidité du sol, l'hygrométrie précise et le taux d'infestation par des insectes nuisibles. Ainsi, il peut adapter son apport en eau, en pesticide et en engrais. Des enjeux importants pour la préservation des sols et un plus grand respect écologique.

L'IoT propose des connexions qui seraient profitables aux jardins et aux fermes connectées. Des connexions internes par exemple, qui, grâce à un systèmes de capteurs connectés chargés de collecter et d'analyser la qualité du sol, la luminosité, la couleur des légumes au cours de leur croissance, la température et les besoins en engrais de chaque plant de manière individuelle, il est possible d'optimiser la croissance des plants avec un minimum d'interaction directe. L'utilisateur peut aussi être alerté en cas de manque de

ressources ou de défaillance en niveau de l'objet connecté lui même, lorsque la récolte est à son terme et quand venu le moment de la cueillir, sans exiger une surveillance manuelle.

Les qualités organoleptique (couleur, saveur…) et nutritives des fruits et légumes peuvent être affectées au niveau de la saveur et la baisse du taux de nutriments, cela est causé par:

- **L'irrigation poussée au profit du volume et du poids des fruits au détriment du goût.**
- La très faible présence ou absence de flore bactérienne dans le substrat faisant office de sol est dommageable pour la saveur des fruits et légumes.

L'état du **système racinaire** d'un plant est déterminant pour obtenir des fruits goûteuses. Il dépend fortement des conditions d'arrosage:

- Un arrosage **modéré ou faible** (stress hydrique) pousse la plante à développer ses racines et à donc à augmenter l'absorption en éléments nutritifs. Cette tendance a pour effet de renforcer la saveur des fruits mais peut **assécher** les tomates en cas de sécheresse du sol ou si l'arrosage est insuffisant.
- Trop d'apport en eau amène à l'effet inverse et peut entraîner le **pourrissement** des fruits.

On constate que l'arrosage est donc une affaire d'équilibre subtil d'apport d'eau en quantité modérée mais suffisante pour que la plante et les fruits se développe de manière optimale.

L'essor de l'agriculture bio, du DIY ("Do It Yourself) et des jardins potagers individuels ou partagés nous montre qu'un marché pour des outils d'aide à la décision / monitoring pour le jardinier amateur pourrait voir le jour à courte échéance avec la massification des objets connectés.

Dans ce projet, nous nous proposons de développer une solution pour aider le jardinier amateur à améliorer le suivi du jardin en lui fournissant des données quantitatives pendant les deux phases de la vie d'un plant :

1. Pendant la phase de croissance de la tige et du feuillage de la plante (système foliacé)

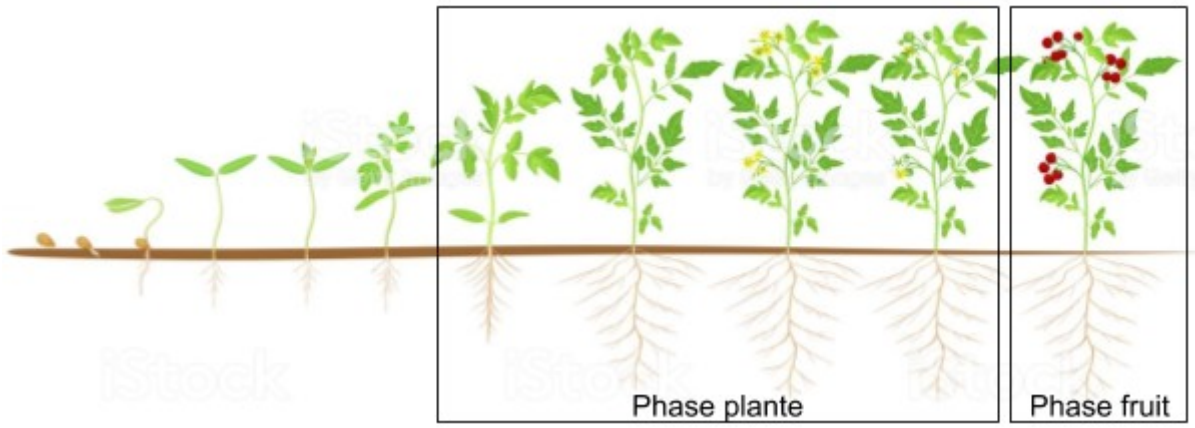
   ↳ Nous appellerons cette phase "**phase plante"**

2. Après la floraison, lors de la croissance et du mûrissement des fruits:
   ↳ Nous appellerons cette phase **"phase fruit"**

Compréhension besoin client

Objectif

Le but de projet est de réaliser un objet connecté à un particulier de manière générale et au jardinier de manière particulière permettant la surveillance et le suivi de ses plants (fruits ou légumes). Des informations pourront être analysées et restituées pour fournir un diagnostic sur l'état de la plante ou le fruit suivi par des décisions. Le processus de surveillance et de recommandations adéquates peut être partagé en deux partie, pendant la période de la croissance des tiges et pendant la période de floraison (fruit).



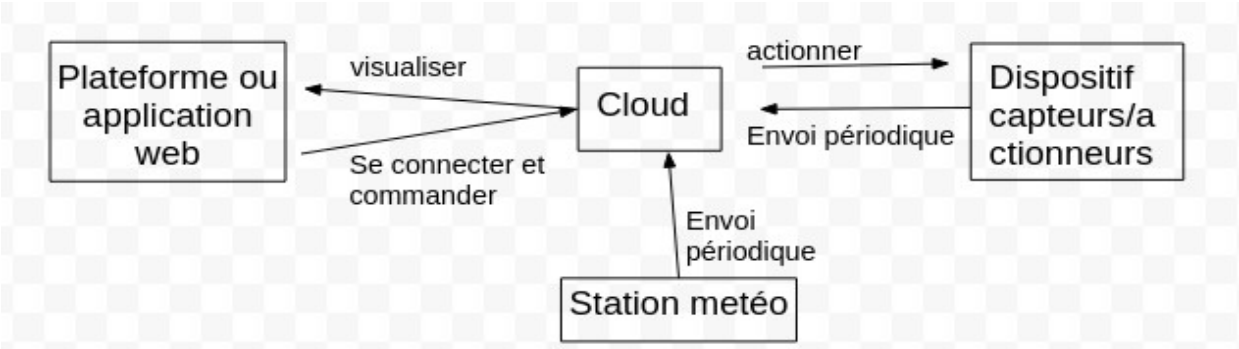
Sur la première phase, une surveillance capteurs fournissant les éléments du climat ou l'humidité du sol sera appliquée *(éventuellement, la couleur des feuilles)* alors que pendant la deuxième phase, des capteurs d'attache seront liés directement à la plante ou au fruit, pour recueillir différents signaux comme son taux de croissance ou son hydratation (dans ce projet, on se contentera d'un capteur couleur pour le suivi de croissance). De manière claire, ces deux parties seront fusionnées pendant la deuxième partie.

Une amélioration et une extension de ce projet est envisageable dans le but d'inclure des parcelles lointaines en utilisant des protocole réseaux de longues distances (Lora par exemple). Généralement, c'est un projet qui pourrait être utile pour les agricultures dans leur travail quotidien.

Techniquement, on doit répondre au cahier de charge d'un jardinier pendant ses longues absences pour qu'il puisse:

- Acquérir les données capteurs (humidité du sol, température, humidité de l'air, la couleur de ses fruits, l'état de batterie autonome…).

- Stocker ces données sur un serveur distant (le cloud) pour une possibilité d'analyse.
- Pouvoir se connecter sur une plateforme web ou application mobile pour la visualisation des données.
- Avoir la possibilité d'actionner l'arrosage de ces plants et autre depuis le web et l'application dédiée.
- Se connecter sur un service en ligne afin de récupérer des données météorologiques

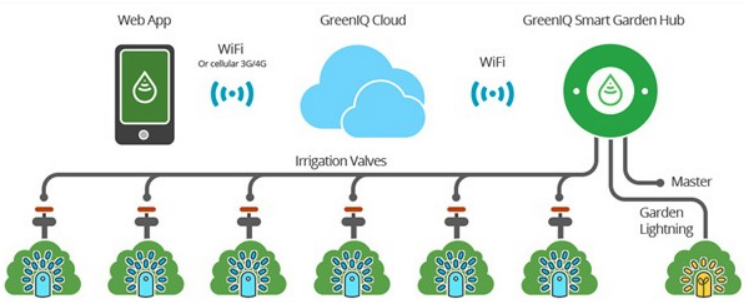


État de l'art

IoT dans le domaine jardinier et dans l'agriculture en général, a pris une avancée considérable dans le traitement ou la surveillance des récoltes. Il est autant développé par les amateurs de l'informatique et de la jardinerie que par les professionnels du métier. Quelques alternatives à ce projets vous sont exposées comme suit:

Capteur pour plantes et arbres fruitiers (Pro) Ce capteur d'attache directement à la plante ou au fruit, pour recueillir différents signaux comme son taux de croissance ou son hydratation. Combinées avec d'autres capteurs fournissant les éléments du climat ou l'humidité du sol, ces informations sont analysées et restituées pour fournir un diagnostic sur l'état de la plante et des recommandations. Proposé par: Phytech

Système d'irrigation intelligent (Pro) Agissant sur des solénoides et valves apposées sur le système d'irrigation, ce dispositif connecté à Internet ajuste les plannings d'irrigation du terrain en fonction des données de prévisions météo locales (pluie, température). Une base de 48000 stations météo est exploitable. Proposé par Hydrawise

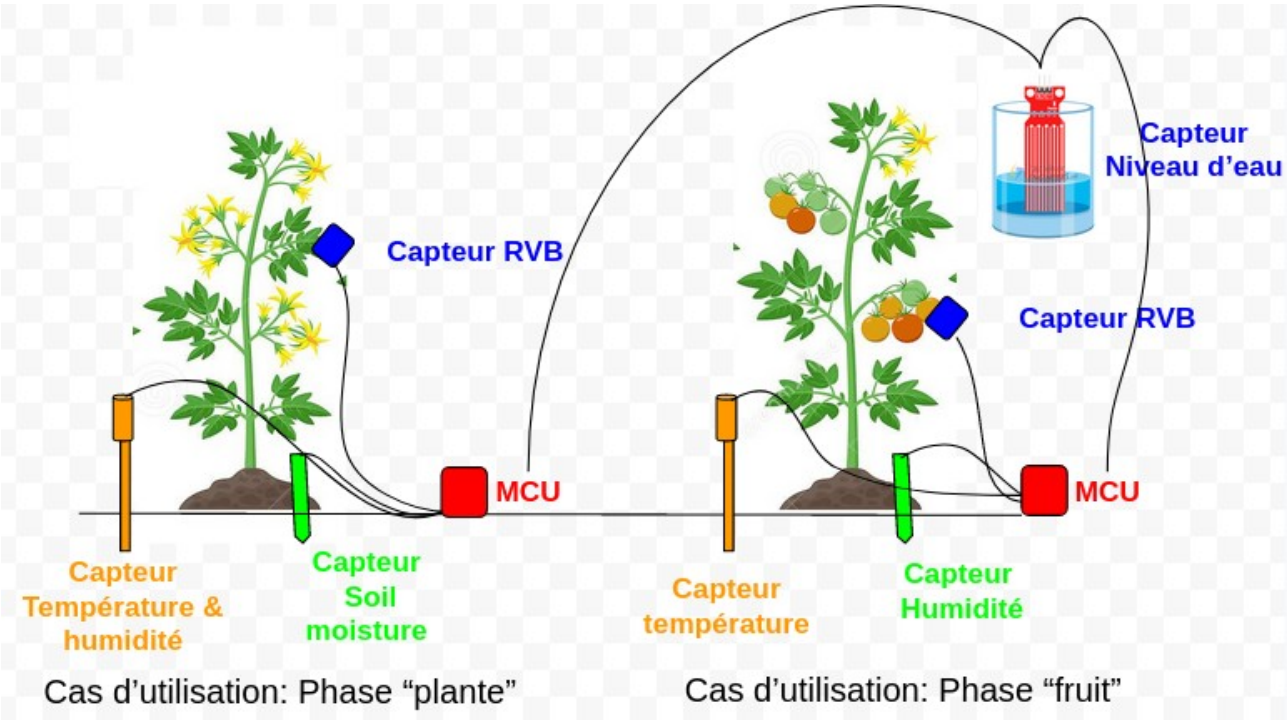


Système de contrôle et de programmation d'arrosage intelligent de Green IQ (Pro) Le système de programmation et de contrôle d'arrosage intelligent Smart Garden Hub de Green IQ est connecté au Web et peut être contrôlé à distance avec un ordinateur, un téléphone intelligent ou une tablette. Originalement conçu pour réduire les frais d'arrosage dans les territoires où l'eau est facturée à la consommation, ce dispositif comporte de nombreux avantages en plus d'être une solution écologique qui réduit le gaspillage.

Traduction technique

L'acquisition des données consiste à relier **4 capteurs (**Cf. schémas ci-dessus selon le cas d'utilisation**)** au dispositif électronique (microcontrôleur - MCU) comme suit:

- Un capteur d'**humidité du sol** planté en pleine terre nous renseigne sur trois valeur
  - sol humide
  - sol sec
  - ou quand le capteur est hors sol ou déconnecté
- Un capteur de **couleur RGB** fixé:
  - sur une feuille de la plante pendant la "phase plante"
    - ➔ Tendance pour le vert: **CROISSANCE EN COURS**
    - ➔ Tendance pour le jaune/brun/marron: **STRESS HYDRIQUE OU MALADIE**
  - sur un fruit ou un légume de la plante pour conclure sur sa maturité et son stress hydrique. Un seuil (code couleur) doit être défini, ex:
    - ➔ Tendance pour le vert: **PAS MÛRE**
    - ➔ Tendance pour le noir (ou points noirs détectés): **TROP D'EAU**
    - ➔ Tendance pour un début de rouge: **DÉBUT DE MATURITÉ**
    - ➔ Tendance pour un rouge intense: **MATURITÉ ATTEINTE**
- Un capteur de **température et humidité de l'air** fixé sur un support pour prévenir de la sécheresse du sol et de la température du lieu.
- Un capteur du niveau de l'eau fixé le bassin dédié pour l'arrosage



Choix technique

On peut envisager plusieurs solutions en partant d'une simple carte WEMOS D1 MINI PRO (ESP8266) à une carte STM32. Compte tenu des contraintes matérielles imposées et le nombre d'entrées sorties à utiliser, on a choisi une solution basée sur une carte NUCLEO-G071RB de la famille STM32 associée au module WIFI ESP8266 pour le transfert de données vers le cloud :

- ➔ Partie embarquée : Nucleo-G071rv (stm32)
- ➔ partie gateway : ESP8266 (WEMOS D1mini Pro)
- ➔ Cloud : AWS (amozon web services)

Liste du matériel utilisé

Matériel	Datasheet
Carte NUCLEO de type (µC) **STM32G071RBTx**	https://www.st.com/en/evaluation-tools/nucleo-g071rb.html
Un dispositif wifi en communication UART avec le µC: **ESP8266**	https://ardustore.dk/error/Manuel%20-%20D1%20Mini.pdf
Capteur de température et d'humidité **DHT11**	https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

Capteur d'humidité de type **LM393** (controler) + Sonde	https://www.onsemi.com/pub/Collateral/LM393-D.PDF
Capteur de couleur RGB avec filtre IR et LED blanche -**TCS34725**	https://cdn-shop.adafruit.com/datasheets/TCS34725.pdf
Capteur de niveau d'eau **Water Level Sensor** (choix disponible)	https://www.emartee.com/product/42285/High%20Sensitivit%E2%80%8By%20Water%20Sensor%20%20Red%20Version
Des fils de câblage	/market
Batterie autonome (2) (**duracell 9V et 12V** pour la pompe à eau)	/market
Une LED RGB (pour indiquer la permutation mode plante/fruit) **KY-009**	https://datasheetspdf.com/datasheet/KY-009.html
Modem wifi domestique	/market
Pompe à eau 9V (**GROTEHN**)	https://www.aliexpress.com/i/32600769362.html
Relais **KY-019**	https://arduinomodules.info/ky-019-5v-relay-module/

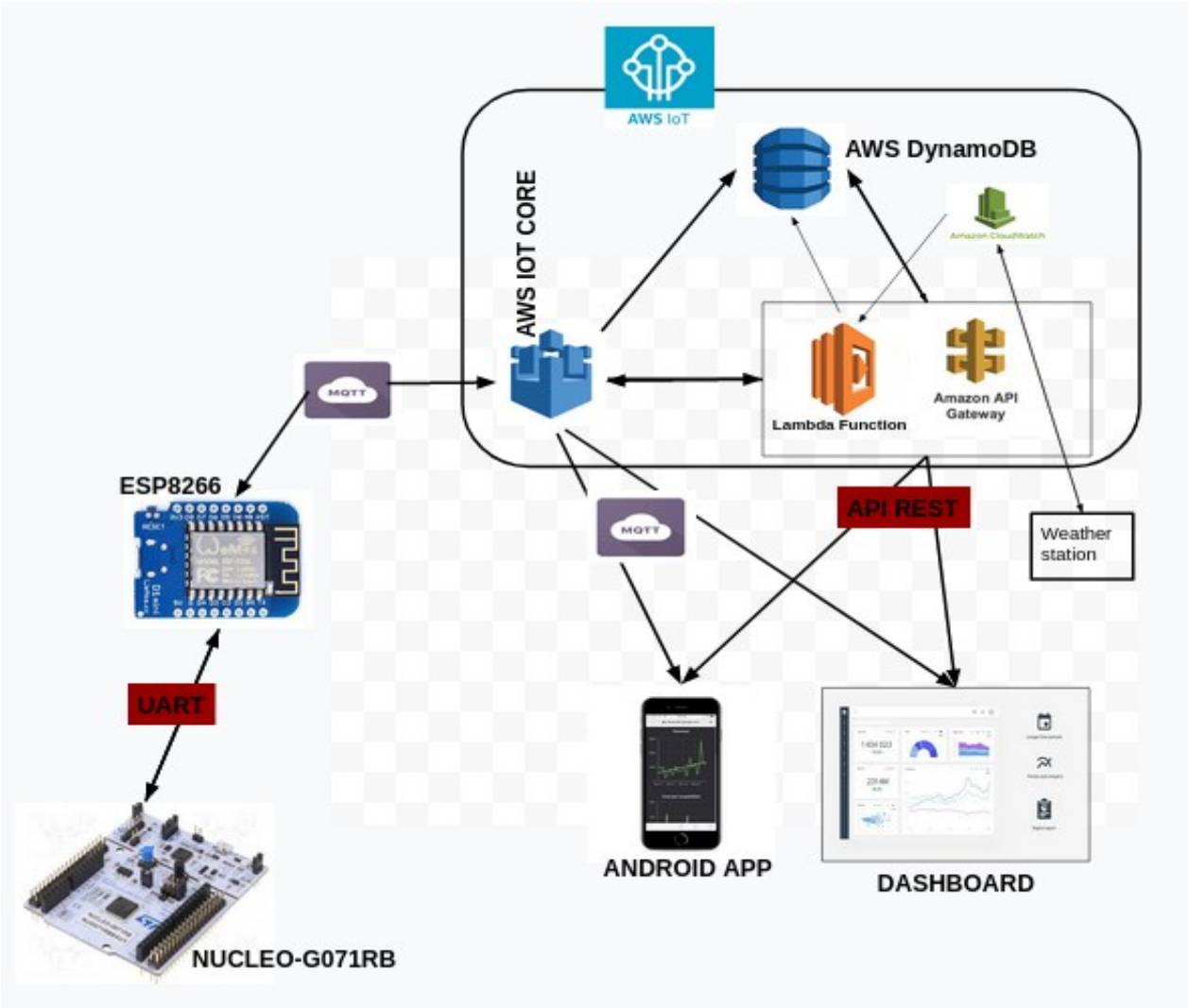
Conception

Fonctionnalités

N° Exigence	Description	
EXF-001	Mesurer l'humidité du sol	La solution doit mesurer l'humidité sur sonde capteur plantée dans
EXF-002	Mesurer le niveau d'eau	La solution doit mesurer le niveau d'eau avec un capteur fixé dans bac un eau
EXF-003	Mesurer la température et l'humidité de l'air	La solution doit mesurer l'humidité et la température avec capteur libre
EXF-004	Mesurer la couleur du fruit ou du légume	La solution doit récupérer la couler avec un capteur attaché au fruit
EXF-005	Mesurer le niveau de batterie	La solution doit mesurer le niveau en utilisant une solution propre au µC
EXF-006	Envoyer les donner sur le cloud	La solution envoie les mesures relevées sur service cloud dédié

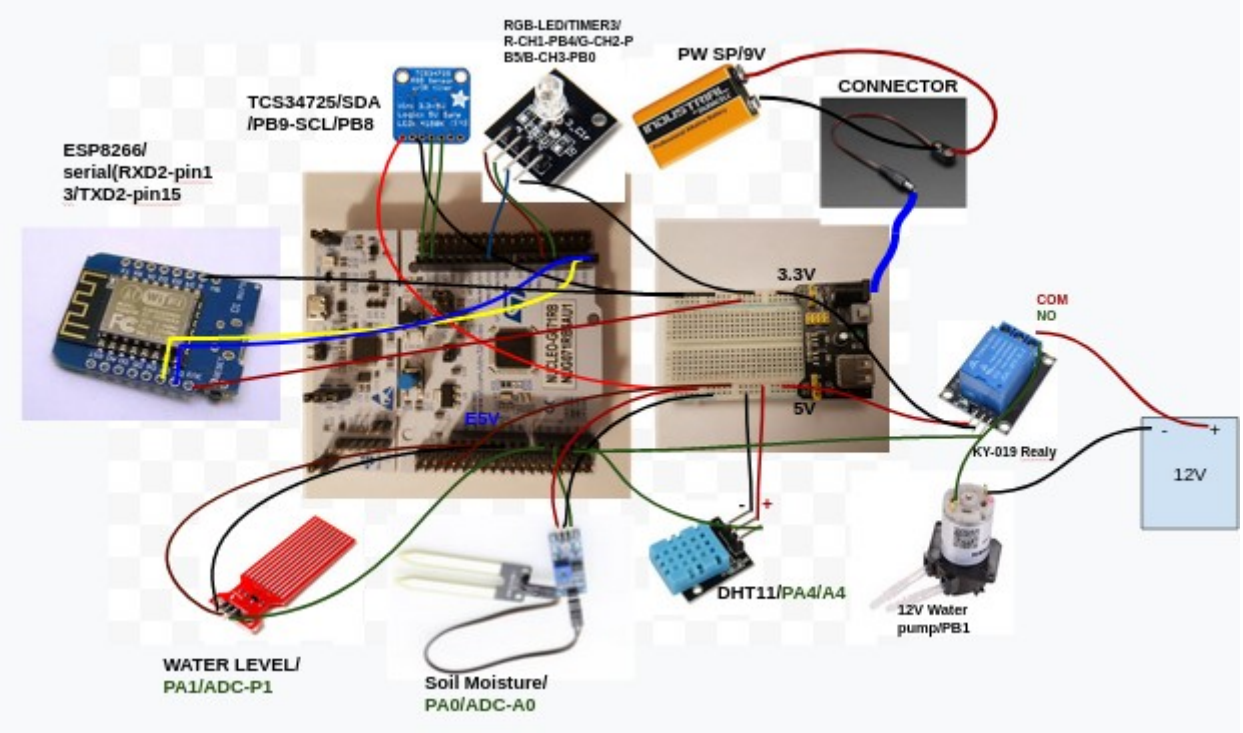
EXF-007	Visualiser les donner sur un dashboard	La solution doit permettre de visualiser les données envoyées, ces données concernent l'humidité du sol, la température, le niveau d'eau et la couleur
EXF-008	Actionner l'arrosage à distance	La solution doit permettre la commande depuis un bouton sur le dashboard ou application web

Schéma d'ensemble



Mise en œuvre

Branchement technique



Gestion de projet

Partie firmware

1- Environnement du développement:

OS: Linux (ubuntu 18.4)

IDE: stm32CubeIDE (page de téléchargement: https://www.st.com/en/development-tools/stm32cubeide.html)

Développement: langage C

Outils de gestion de versions: GIT/Gitlab

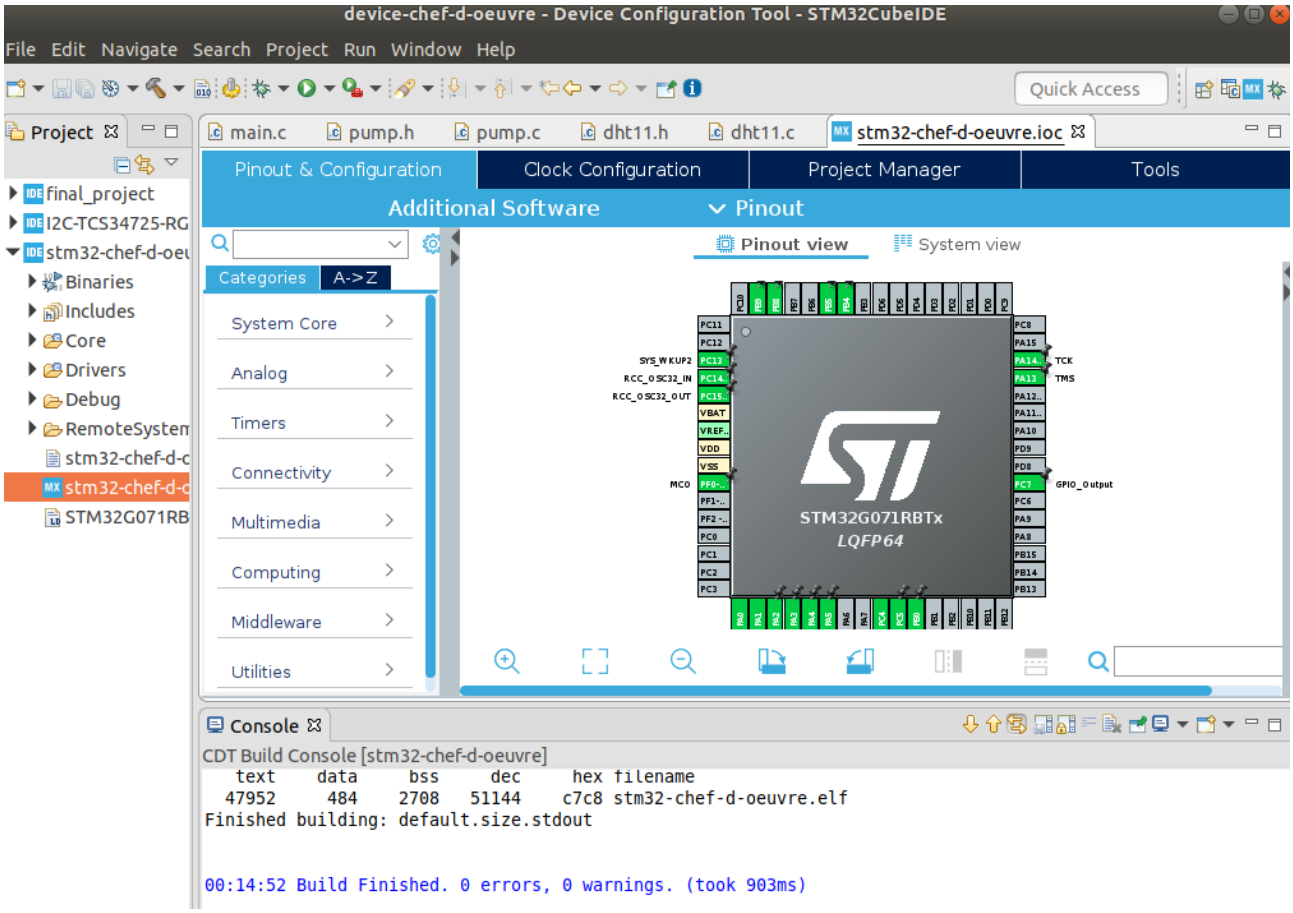
Carte STM32: Nucleo-G071RB (https://www.st.com/en/evaluation-tools/nucleo-g071rb.html)

Création de projet: Ouverture du stm32cudeIde /choisir un espace du travail/ choisir la carte STM32 (nucleo-g071rb) /nommer son projet /choisir C comme langage.

Voir get started with STM32G0: https://www.youtube.com/watch?v=Xh3ORJ_-5Gs

Utilisation des connectivité UART: dans ce projet deux UART on été utilisées:

- USART2 pour les print console série (sur cubemx ou indépendamment CuteCom)
- USART3 pour l'envoi/réception (RXTX) des données avec la Gateway



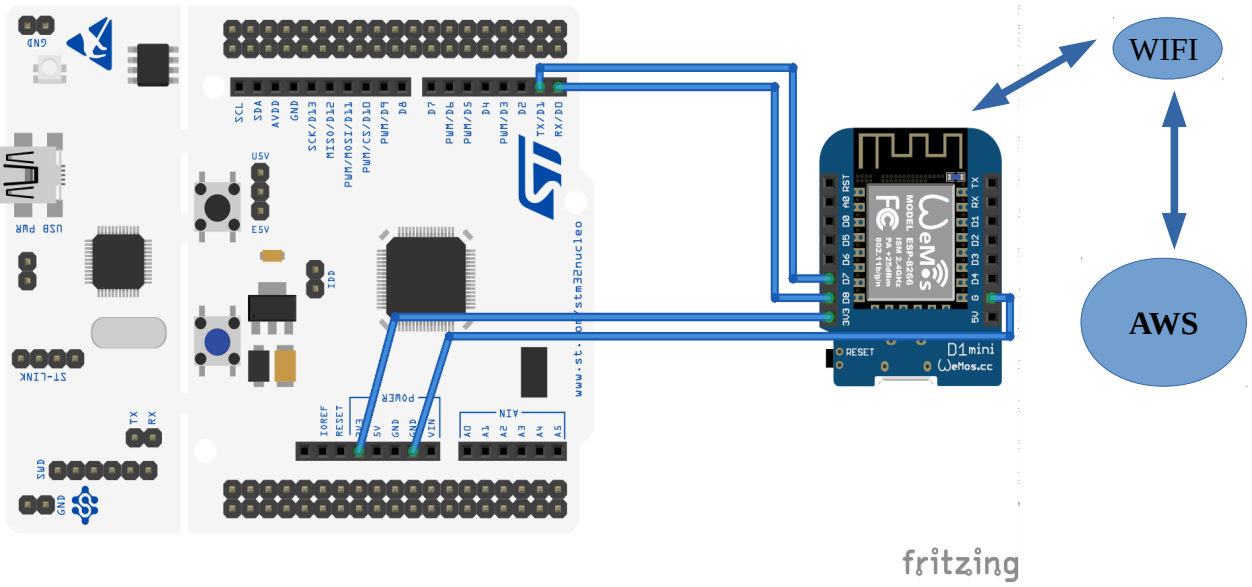
2- Mise en œuvre des capteur:

Capteur	Interface avec Nucleo-G071	Observation
Couleur RGB TCS34725	I2C (SDA/SCL)	**On lit 4 valeurs: couleur R, G et B et la clarté C.** Il faut tout un chapitre pour parler de ce capteur associé au port I2C t : https://cdn-

		shop.adafruit.com/datasheets/TCS34725.pdf
Capteur humidité LM393 **Capteur du niveau d'eau**	ADC	Sur la stm32, on a qu'un seul ADC. Pour pouvoir lire plus d'un capteur en ADC, une lecture en multi-channel s'impose, on peut utiliser la méthode **PollforConversion** (retenue pour ce projet) ou DMA. https://controllerstech.com/how-to-read-multichannel-adc-in-stm32/
température/ humidité DHT11	Une pin GPIO en INPUT/OUTPUT	- Créer un timer en microseconde - Initialiser le capteur en mode input/output - Lire la réponse du capteur - Lire les données capteur
Relais KY-019	Une pin GPIO en OUTPUT	Nucleo+Relais+Pompe: voir câblage A chaque fois le pin GPIO se met à 1 (5V), la pompe est mise en marche pour une certaine période, et 0 (0V) la pompe est mise à l'arrêt

Partie Gateway

Pour faire l'interface entre le device (partie embarqué) et le cloud (aws), on utilise carte WEMOS D1 mini Pro (ESP8266). Elle lit et écrit sur la Nucleo en utilisant le protocole UART (RxTx) et à l'aide d'une connexion WIFI elle fait la même chose en utilisant le protocole **MQTT**[1] et des clés de sécurité préalablement fournies en inscrivant son objet (device) sur AWS IOT.



1: (Message Queuing Telemetry Transport) est un protocole de messagerie publish-subscribe basé sur le protocole TCP/IP. De très nombreuses bibliothèques sont disponibles pour programmer des clients MQTT, pour la plupart des langages (C, C++, Java, JavaScript, PHP, Python…) et sur la plupart des plates-formes (GNU/Linux, Windows, iOS, Android, Arduino…).

1- Environnement du développement:

IDE: Visual Studion Code (vscode) (page de téléchargement: https://code.visualstudio.com/download)

Développement: langage C++

environnement de développement ESP8266: PlatformIO à installer sur vscode

Connexion WIFI: connexion domestique

2- Librairies utilisées:

ESP8266-Nucleo-G071RB	**SoftwareSerial**: connexion UART **Arduino**: console série
ESP8266-AWS	**ESP8266WIFI**: connexion WIFI **PubSubClent**: on s'abonne à un « topic » (boite au lettre) pour publier ou récupérer des données **ArduinoJson**: construction des données en format json **libb64/decode**: décodage des certificats AWS **ctime**: Unix time dans le format json

Partie Amazon Web Servies (AWS)

Le service AWS IoT Core vous permet de connecter facilement un nombre indéfini d'appareils au cloud et à d'autres appareils. AWS IoT Core prend en charge les protocoles HTTP, WebSockets et MQTT. Ce dernier est un protocole de communication léger spécialement conçu pour gérer les connexions intermittentes, limiter la longueur du code sur les appareils et réduire les exigences en termes de bande passante réseau. AWS IoT Core est également compatible avec d'autres protocoles standard ou personnalisés, et les appareils peuvent communiquer entre eux même s'ils utilisent des protocoles différents. (source: https://aws.amazon.com/fr/iot-core/?nc=sn&loc=2&dn=3)

1- Enregistrer un objet dans AWS Iot:

- S'inscrire à AWS IOT CORE pour avoir un compte
- Sectionner **Services/IoT Core**
- Sur la colonne de gauche : **Gérer/Créer/Créer Objet** unique/ nommer l'objet puis **suivant**
- **Créer un certificat,** dans un dossier en local, télécharger les trois certification plus la CA (**rootCA**) avant d'activer puis appuyer sur **Attacher une stratégie** et sur la page suivante **Enregistrer l'objet**
- dans le menu de gauche, choisir **Sécurité/Stratégie**, **Créer**/nommer la stratégie/**action**(iot :*), **Ressource** (*)/cocher **autoriser** puis fermer
- **Sécurité/Certificats,** attacher à la certificat déjà générée la stratégie et l'objet crées précédemment
- Dans le menu de gauche: **Gérer**/ l'objet crée/ **Interagir** / sous **https** enregistrer l'adresse brocker (*xxxxxxxxxxxxxx-ats.iot.eu-west-2.amazonaws.com*) de l'objet pour la suite (communication MQTT)

Exemple d'enregistrement d'objet:
https://docs.aws.amazon.com/fr_fr/iot/latest/developerguide/create-aws-thing.html

2- Créer une base de donnée DynamoDB:

Dans ce projet, deux table sont nécessaires, une pour stocker les données envoyer par l'objet firmware et une deuxième pour stoker les données météorologiques depuis une station météo.

- Dans **Services** choisir **Dynamodb**
- **Tables/Créer une table/** donner un nom, une clé primaire (**Clé de partition**, **Clé de tri** (facultatif)) puis **Créer.**
- Dans **Présentation** de la table, il faut chercher l'**ARN** _de la table le conserver pour la suite

Exemple de création de tables:
https://aws.amazon.com/fr/getting-started/hands-on/create-nosql-table/

3- Créer aws rule (règle) pour insérer des données dans DynamoDB:

L'action DynamoDB permet de prendre des informations d'un message MQTT entrant et de les écrire dans une table DynamoDB. A chaque message MQTT

(données json), la règle est actionnée pour insérer ce données dans la table correspondante.

- **Services/IOT Core,** dans le menu de gauche **Agir/Créer**
- Donner un **nom** à la règle et choisir un **topic** pour le MQTT: SELECT * FROM '**le nom du topic**'
- Appuyer sur **Ajouter Action/** choisir **Insérer un message dans une table DynamoDB/** chercher le nom de la table et remplir les case pour l'**insertion**
- Appuyer sur **Ajouter action** puis **Créer** la règle

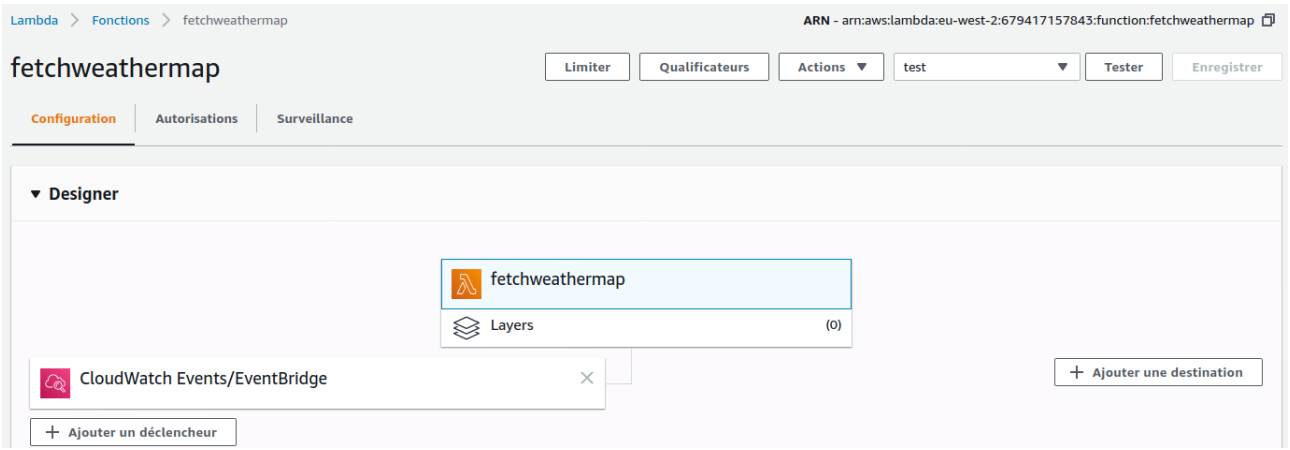
Exemple d'enregistrement d'objet:
https://docs.aws.amazon.com/fr_fr/iot/latest/developerguide/iot-ddb-rule.html

2- Créer une fonction aws lambda pour les données météorologiques:

Open Weather Map va nous servir de station météo nous permettant d'avoir données météorologique d'une région donnée (ici Toulouse). Il faut s'inscrire et sur la plateforme et récupérer une **clé API** et un **endpoint** pour les requettes API.

Dans Aws Services, on développe une fonction lambda (en python) qui va interroger l'API openwaethermap.com à l'aide de la clé API. Elle récupère un tas de données et elle sélectionne la température, l'humidité et la pression à envoyer sur la base de données dédiée (nom de la table: weatthermap).

Afin de définir la périodicité des appels, on fait appel au service CloudWatch Events d'aws en lui configurant une règle de périodicité (ex. :1 jour). Ce service joue le rôle de déclencheur pour l'exécution de la fonction lambda.



Partie Dashboard

Afin de visualiser différentes données lues depuis l'objet ou depuis la station météorologique, on construit une page web statique qu'on va héberger dans un compartiment Aws S3 fournissant un URL.

Outils de développent web: HTML/CSS/JavaScript/jQuery, la librairie D3.js pour les graphes

Outils AWS: aws S3 pour l'hébergement, API Gateway pour déclencher les fonctions lambda, **lambda function** pour publier dans les topics, dynamoDB pour la lecture des données, Cognito pour s'identifier au compte AWS, IAM pour les rôles et les stratégies.

1- Aws Cognito et les groupes d'identité

Pour pouvoir travailler sur les données qu'on a dans Aws Iot, on doit s'identifier à l'aide du service **Aws Cognito** (configurer un groupe d'identité pour JavaScript)**.** Dans le fichier JavaScript (ici refresh.js), il faut rajouter l'identification comme suit :

```
// Initialiser le fournisseur d'informations d'identification Amazon Cognito
AWS.config.region = 'eu-west-2'; // Région
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'eu-west-2:d7f65-ef9b-4e6c-a233-9c821dcdaae6',
});
```

En configurant le groupe d'identité, un rôle aws est automatiquement généré à qui il faudrait attacher des autorisations (stratégies) de lecture sur DynamoDB.

2- Fonctionnalités du dashboard:

RGB Color:

On lui associe une fonction de lecture (query) sur la base de données ou la couleur du fruit est stocké, on récupère sa dernière valeur acquise et on met à jour la couleur sur le graphe.

Temperature & Humidity:

On se limite à afficher les 15 dernières données acquises. Une fonction jQuery qui lit quotidiennement la base données dédiée à la température et l'humidité et mit à jour le graphe avec les 15 valeurs le plus récentes.

Power & water level:

C'est le même principe qu'avec la couleur, quotidiennement on récupère les dernières valeurs acquises depuis depuis DynamoDB dédiée et on met à jour la bar de progression. Une valeur globale (les niveaux) est associée pour dernière valeur acquise et la valeur correspondant à la progression de la bar dans la fonction jQuery.

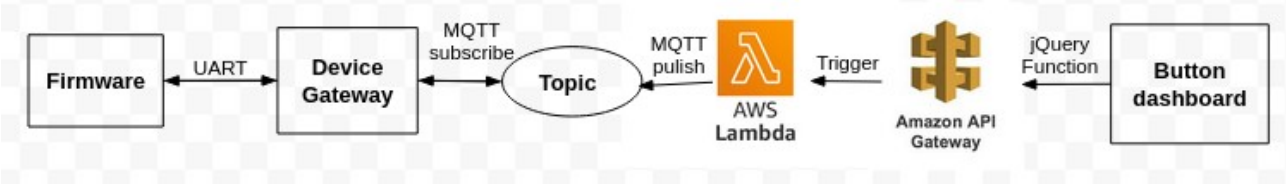
Les boutons d'action:

Ici, c'est toute autre chose, il s'agit de descendre les informations (commandes de mise à jours) depuis le dashboard en passant par Aws IoT vers le firmware. Pour cette descente d'information, on s'appuie sur certains services aws:

- **Aws API Gateway:** en appuyant sur le bouton du dashboard (ex: *Irrigate*) une fonction jQuery dédiée fera appel à l'URL de l'API Gateway déjà déployée

- API Gateway déclenche une **Aws lambda function** qui a le rôle publier dans un topic (ex. de message en format json : {'command' : 'Irrigate'} défini avec un ARN

- Et comme l'objet (**firmware**) par l'intermédiaire de la Gateway (ESP8266) est abonné au topic précédent, en boucle, il le consulte et à chaque message il le traite et actionne la commande demandée.



Retour d'expérience sur les outils

Dans ce projet, un grand nombre d'outils ont été utilisées, quels soient matériels ou software, ils m'ont permit de développer des compétences nouvelles dans le domaine de l'IoT.

Les IDE's: j'étais plus à l'aise en travaillant avec stm32CubeIde pendant la partie embarquée et moins à l'aise VSCode pour la partie Gateway. Sur le dernier associé avec PlatformIO, je constaté une consommation de temps considérable pendant la compilation et le téléversement, ce qui me rendait un nerveux.

Les capteurs: à part le TCS34725 qui demande beaucoup de temps pour comprendre son fonctionnement, les autres capteurs m'ont pas causé beaucoup de problèmes pour mettre en œuvre leurs drivers.

Aws Iot: c'est très intéressant, un océan d'information et de techniques à prendre en compte, ça demande beaucoup de temps pour bien maîtriser cet outils et la mise en place un objet connecté sur la plateforme. Personnellement, j'ai aimé travailler dessus et j'en fais même un domaine de spécialisation à l'avenir.

Les cartes (Boards): plus simple de travailler avec la WEMOS qu'avec la STM32. C'est normal, c'est beaucoup de fonctionnalités et de techniques, il faut passer plus de temps à développer des projets sur cette carte pour qu'au fur et à mesure on se sentira à l'aise.

Git/Gitlab: une nouveauté pour moi, je suis content de travailler avec cet outils et de le rajouter sur mon CV.

Retour d'expérience sur les techniques

Tout au long de la formation, j'ai pu découvrir différentes techniques et méthodes de travail. J'ai pu voir et expérimenter la méthode agile **Scrum**

qu'est une méthode hautement exigée pour la recherche d'emploi. J'ai découvert le protocole **MQTT** qu'est un puissant protocole de transfert de données « machine to machine ».

Globalement, j'ai touché à pas mal de langages de programmation dont certains sont nouveaux (python, JavaScript...) et m'ont permis de d'enrichir mon CV et j'ai monté en compétence concernant ceux que je connaissais déjà (C/C++).

Enfin, je dois noter mes difficultés en travaillant sur les **timers** et le **freertos** de stm32 en général. Là déçus je dois encore travailler et m'améliorer

Amélioration

- Refaire la partie embarquée en utilisant le FREERTOS de la Nucleo-G071
- Prévoir un protocole (ex : HDLC) d'échange de données en la STM32 et la WEMOS, c'est plus sécurisant et plus professionnel
- Remplacer la Gateway actuelle (WEMOS) avec un module Lora afin de transporter l'objet sur des fermes lointaines
- Finir la partie Android pour la visualisation et les recommandations

Bilan

En se référant au but initial qui est de réaliser une chaîne complète de remonté de données pour un objet connecté, je dirai que c'est réussi. J'ai su exploité les compétences acquises pour réaliser les quatre parties (firmware, gateway, cloud et dashbord) malgré le manque de documentation sur certaines techniques. Pour l'avenir, il me reste exploiter cet objet connecté dans des conditions réelles et de l'améliorer et sans oublier de finir la partie Android.

Remerciements : je tiens à remercier l'équipe pédagogie de Simpson à leur tête Nicolas et Corentin pour leur disponibilité tout au long de la formation. Je tiens aussi à saluer l'esprit d'équipe de la promo, c'est aussi grâce à l'ambiance et entraide de l'équipe qu'on a pu réaliser ce projet.