

**Master 1 Informatique CERI**

**Rapport**

**Remote Automatic Watering System**

Responsable

**M. BENSLIMANE**

Réalisé par:

**NDIAYE Elhadj Pathé**

**KANTE Alpha Omar**

**TRAN Bao Thien**

**Année Universitaire**

2018 - 2019

**Remerciement**

Nous tenons à remercier sincèrement Monsieur Benslimane Abderrahim, qui en tant que tuteur s’est toujours montré à l’écoute tout au long de la réalisation de ce travail ainsi pour sa générosité, ses conseils fructueux qu’il nous a prodigués le long de notre projet et la grande patience dont il a su faire preuve, malgré ses charges professionnelles.

Nous remercions également Monsieur Philippe Gozlan, Monsieur Marc Silanus et Monsieur Sahand Khodaparast, qui nous ont accompagné tout au long de notre projet et nous a guidé dans sa réalisation.

# **PLAN**

[**PLAN**](#_erzp0vhrutcc) **3**

[**Liste des figures**](#_1rakg8u1b83z) **5**

[**1. Introduction**](#_as6ejjv4tp1l) **7**

[**2. Présentation du projet**](#_98r0b2974e5b) **8**

[2.1 Problématique](#_pinadr5qqigb) 8

[2.2. Objectifs](#_t2hwewakbw7s) 8

[2.3. Solution proposés](#_23zt55mdobjj) 8

[**3. Mise en place de l’architecture de déploiement**](#_6mwyakcky5r0) **9**

[3.1. Architecture Logique](#_vki1ynwsf83z) 9

[3.2. Architecture Physique](#_o732ssp9vrp5) 11

[3.3. Conception et réalisation du montage](#_88dr96vkoipy) 12

[**4. Implémentation de l’architecture proposée**](#_6q3hs1chr1yd) **13**

[4.1. Programme arduino](#_kex6zra2ytun) 13

[4.2 Application Cloud TTN](#_fhnvqnz09q14) 16

[4.2.1 Création du compte TTN](#_4at4j2eqzuj4) 17

[4.2.2 Création de l'application](#_s205ce4l95u6) 17

[4.2.3 Création de l'equipement](#_roxfkmjimpj4) 19

[4.2.4 Formatage des données](#_uorhdl180dtg) 20

[4.3. Application Node-Red](#_9bpayylrkq5l) 22

[4.3.1 Ajouter des palettes](#_prja6flz6wpu) 23

[4.3.2 Le protocole MQTT et configuration du node](#_pxvuiikj25fb) 24

[4.3.3 Influxdb et configuration du node](#_x0mz89o1so7o) 31

[4.5 Application Grafana](#_jizvyj9y4cb6) 33

[4.5.1 Ajouter des données sources](#_bbls6ldevf1l) 34

[4.5.2 Conception des Graphes](#_svi9a5rb48hk) 35

[4.6. Test de fonctionnement](#_lxhu8elg3uxl) 38

[4.7. Difficultés rencontrés](#_ngacs58r3hux) 38

[**5. CONCLUSION**](#_ssoo87qbj050) **40**

[**ANNEXES**](#_s6p01neca7p5) **41**

[Installation de node-red](#_xag9pb62gj1j) 41

[Installation and configuration](#_3zv1vl8whs3e) 41

[**WEBOGRAPHIE**](#_xnl5n310fslv) **44**

# **Liste des figures**

*Figure 1 : Architecture logique du réseau LORA*

*Figure 2: Architecture physique proposée*

*Figure 3: Schéma du montage*

*Figure 4: Montage réalisé*

*Figure 5 : La fonction SETUP*

*Figure 6 : Code de récupération des données des capteurs*

*Figure 7 : Code d’envoi de données vers TTN*

*Figure 8*: Programme de traitement des données TTN

*Figure 9: Programme de pilotage de l'électrovanne*

*Figure 10: Création compte TTN*

*Figure 11: Création de l’application*

*Figure 12: Rubrique device*

*Figure 13: Création du matériel*

*Figure 14: Data Application*

*Figure 15: Fonction de décodage*

*Figure 16: Résultat décodage*

*Figure 17: Flow uplink de TTN vers la base de donnée*

*Figure 18: Flow downlink de l’interface utilisateur et la carte Arduino MKR*

*Figure 19: Installation des nodes*

*Figure 20: Ajout d’un node*

*Figure 21: Node mqtt connexion*

*Figure 22: TTN Application ID*

*Figure 23: TTN default key devices messages*

*Figure 24: Configuration de la sécurité*

*Figure 25: Node MQTT configuration serveur*

*Figure 26: Node MQTT configuration en uplink*

*Figure 27: Node MQTT configuration en downlink*

*Figure 28: Configuration Node button*

*Figure 29: Résultat configuration button*

*Figure 30: Configuration node influxDB*

*Figure 31: Function formatage des données à enregistrer*

*Figure 32:Configuration data source*

*Figure 33: Configuration data source*

*Figure 34:Configuration InfluxDB*

*Figure 35:Configuration de la graphe de température*

*Figure 36:Configuration de la graphe d'humidité*

*Figure 37: Configuration température actuelle*

*Figure 38:Configuration humidity actuelle*

*Figure 39 : Visualisation de la température et de l’humidité en temps réel sur grafana*

# 

# **1. Introduction**

L’internet des objets (IoT) est une technologie qui permet d'interconnecter des objets électronique à travers d’internet. Cette technologie permet l'extension d’internet aux objets physiques. Ces objets sont en général des systèmes électroniques embarqués équipés de capteur pour faire des mesures telles que la température, l’humidité, position, le débit, la distance et autres.

Ces objets connectés permettent de récolter une masse de données qui favorise une meilleure compréhension de notre environnement mais peuvent être configuré pour contrôler notre environnement et faciliter notre vie quotidienne.

L’internet des objets est utilisé dans divers domaines tel que la santé, la domotique, la robotique alors pourquoi ne pas l’utiliser dans le domaine de l’agriculture ?

L’agriculture étant le secteur utilisant la majeur partie des ressources en eau de la planète, alors un contrôle de cette utilisation de l’eau pourrait permettre une réduction considérable de la consommation en eau.

L’application de l’Internet des Objets, permettra ainsi de recueillir les données nécessaires à la compréhension de l’environnement des plantes et de les approvisionner en eau en fonctions de leurs besoins.

Dans le cadre de ce projet de Master 1 nous allons mettre en place un **Système automatique d’irrigation à distance en utilisant arduino et LoraWan (Remote Automatic Watering System).**

Ce projet vise l’application de l’internet des objets au domaine de l’agriculture pour faire une irrigation contrôlé et distante d’une espace de culture.

Durant le premier semestre, nous avons établit un cahier de charge et proposé une architecture. Dans le deuxième semestre, nous sommes passé à l'implémentation du projet.

# **2. Présentation du projet**

## **2.1 Problématique**

Nous devons mettre en place un réseau LoRaWan qui permet à nos différents objets de communiquer avec le cloud TTN, de collecter des informations sur les capteurs, d’effectuer des actions automatiques pour arroser un espace de culture.

Cependant, pour une étude de l’art l'architecture proposée nous allons vous présenter les différentes technologies utilisées pour la collecte et la communication entre les équipements de bout en bout.

## **2.2. Objectifs**

Nous allons mentionner 3 parties via l'architecture de LoRa:

* Réseau de collecte des données
* Réseau de transport des données
* Applications Node-red, InfluxDB et Grafana

## **2.3. Solution proposés**

Nous allons concevoir et déployer un réseau de collecte de données à savoir la température et l’humidité du sol grâce au protocole LORA et au réseau LORAWAN. Ces données seront ensuite envoyées vers le réseau TTN afin de pouvoir les stocker et de pouvoir piloter les objets connectés. Nous allons enfin démarrer la valve électrique pour arroser la plante en fonction des données recueillies.

# 

# **3. Mise en place de l’architecture de déploiement**

## **3.1. Architecture Logique**

Un réseau LORA est composé de plusieurs éléments qui sont comme suit:

* **Devices ou objets connectés:**

Ce sont des matériaux électroniques généralement des systèmes embarqués qui sont connectés à des capteurs permettant de récolter des données et de les envoyer sur le réseau LoRa mais peuvent aussi être contrôlés pour accomplir des tâches diverses pour améliorer la vie quotidienne comme ouvrir une porte, détecter la présence d’un objet, déclencher un moteur ect.

Ces objets peuvent être des cartes arduino, des cartes raspberry pi et autres.

* **Gateway ou passerelle:**

Ce sont des matériaux électroniques qui permettent d’assurer la communication entre les objets connectés et le réseau TTN des objets connectés. Ils servent de passerelles et sont connectés au réseau LORAWAN et aussi à Internet.

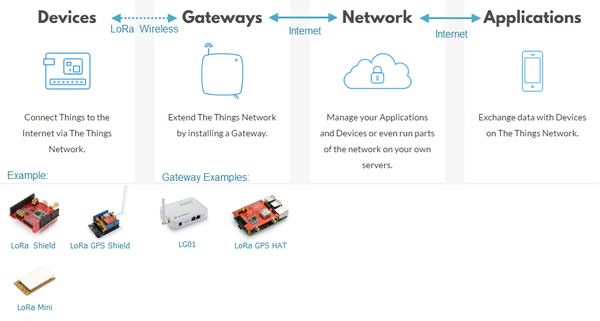
* **Network ou le réseau TTN:**

C’est un réseau des objets connectés qui nous permet de gérer les applications et les objets connectés sur nos propres serveurs.

Cela permet alors de recueillir les données et de les stocker sur nos serveurs personnels et aussi de piloter les objets connectés à distance grâce au réseau TTN.

* **Application:**

Sur le réseau TTN, nous pouvons créer des applications pour gérer nos objets connectés et échanger des données entre les objets connectés et le réseau TTN.

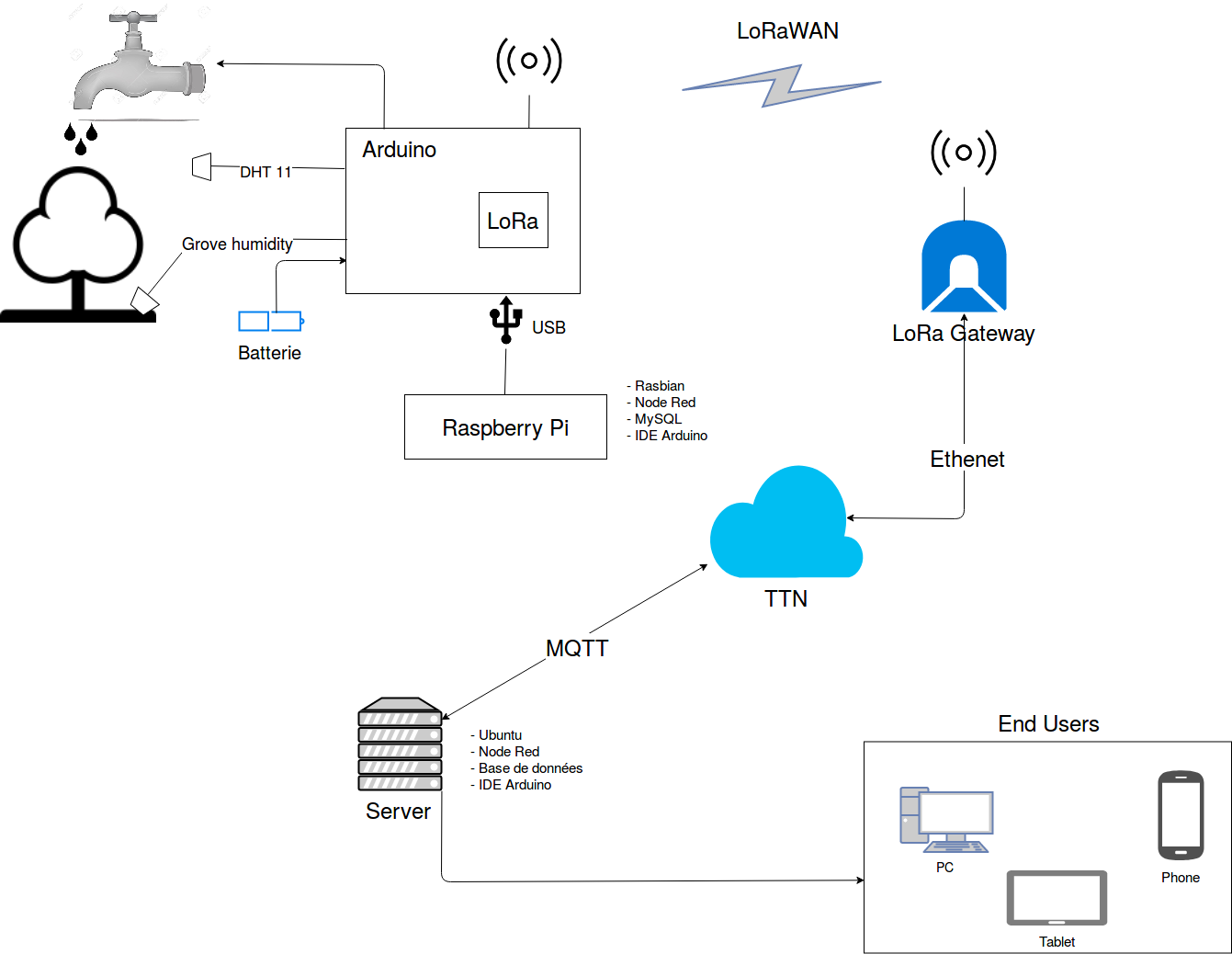
******

*Figure 1 : Architecture logique du réseau LORA*

## 

## **3.2. Architecture Physique**

Pour implémenter notre solution l’architecture finale proposée se présente comme suit:

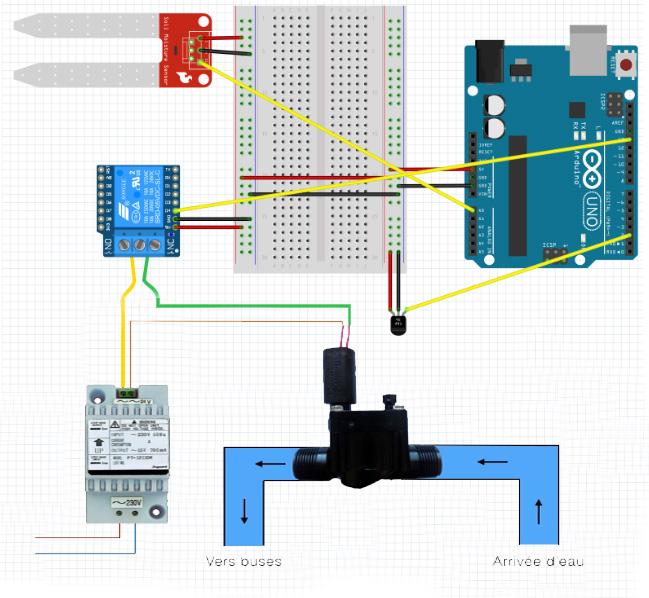
*Figure 2: Architecture physique proposée*

Cette architecture est composée des éléments suivants:

* Une carte arduino MKRWAN 1300 connectée à des capteurs d'humidité et de température et à une valve électrique qui peut être pilotée pour arroser une culture,
* Une Passerelle Lora TTN qui permet de transférer les données recueillies par la carte arduino vers le réseau TTN,
* Un serveur de base de donnée influxDB pour stocker les données des capteurs,
* Un **serveur Node-Red** pour traiter les données reçues et les formater, pour implémenter les procédures de stockage des données et de pilotage des objets connectés,
* Des Utilisateurs finaux au travers de PC, Tablette ou Téléphones qui peuvent visualiser les données et piloter la valve électrique à distance.

## **3.3. Conception et réalisation du montage**

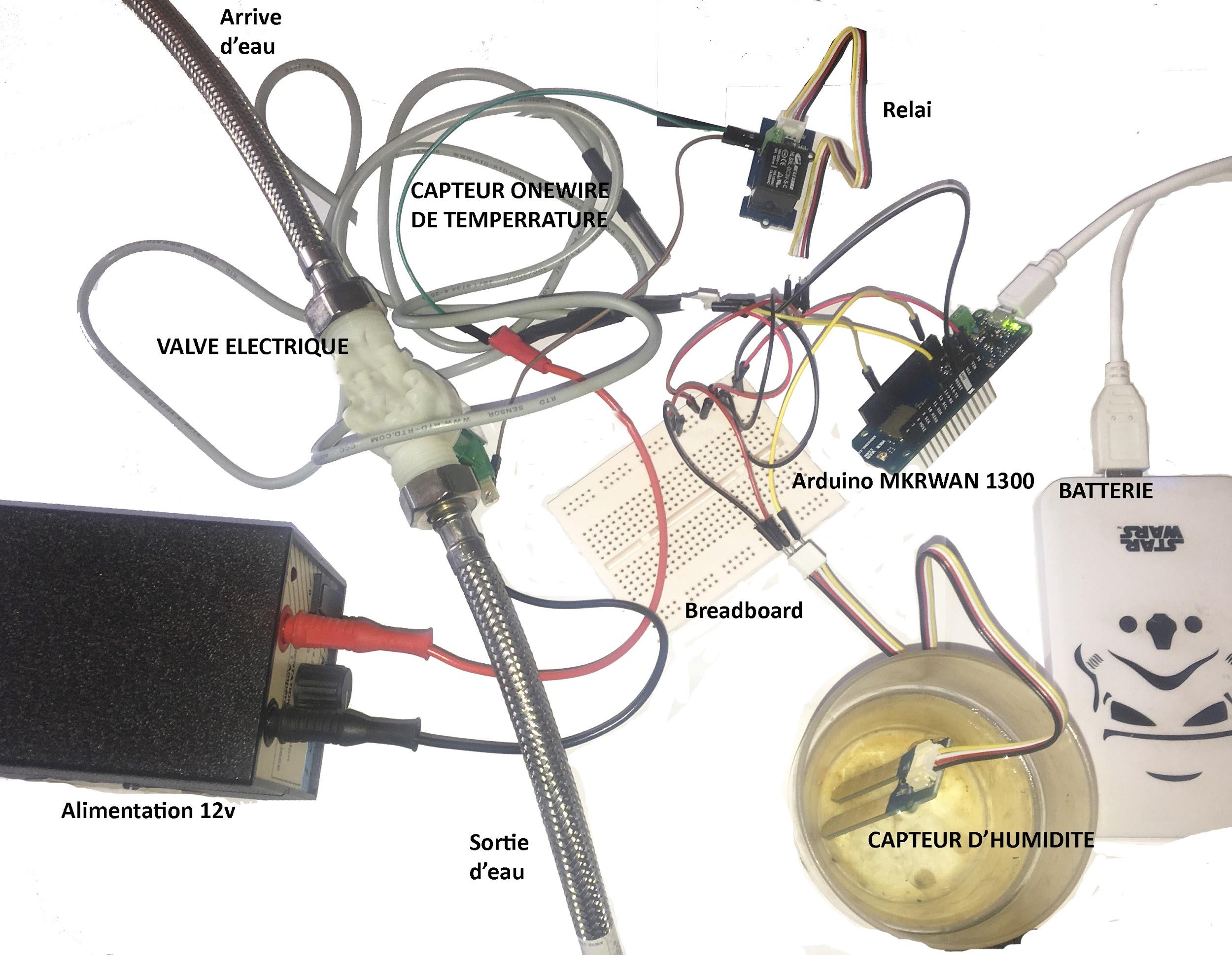
un montage de l’architecture de collecte des données se présente comme suit:



*Figure 3: Schéma du montage*

Il est composé d’une carte arduino, De capteur de température et d’humidité, d’une valve électrique pour déclencher l’arrosage et d’un relai.

Le montage realisé est comme suit:



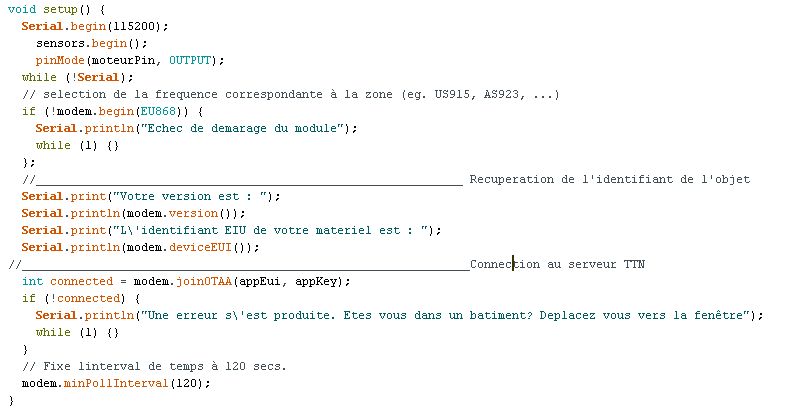
*Figure 4: Montage réalisé*

# **4. Implémentation de l’architecture proposée**

## **4.1. Programme arduino**

Pour programmer la carte arduino MKRWAN 1300, nous avons écrit un programme arduino en C++ qui est composé d'instruction à savoir la déclaration des variable et des librairies, une fonction SETUP et une fonction LOOP.

La Fonction SETUP permet de faire la configuration initiale de la carte arduino et des différents PINs en Input ou output, la récupération de l'identifiant de l’objet et la connexion au serveur TTN.



*Figure 5 : La fonction SETUP*

La fonction LOOP est exécutée de façon itérative et est constituée d’un ensemble d’instruction permettant de faire des tâches diverses:

* **Collecte des données de température et d’humidité:**

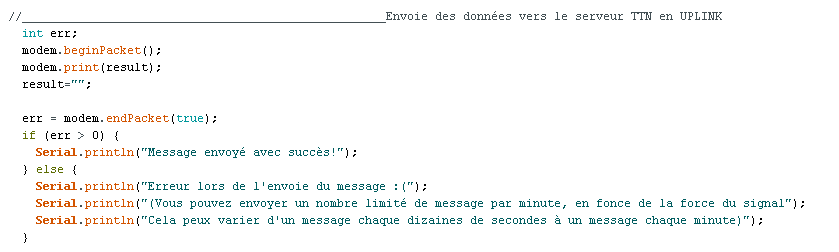
Ce code permet de récupérer les données d’humidité et de température recueillies grâce aux capteurs onewire de température et le capteur d’humidité. Elles sont ensuite concaténées dans la variable *result* qui sera ensuite envoyé vers le serveur TTN.



*Figure 6 : Code de récupération des données des capteurs*

* **Envoi des données des capteurs vers le Serveur TTN en UPLINK**

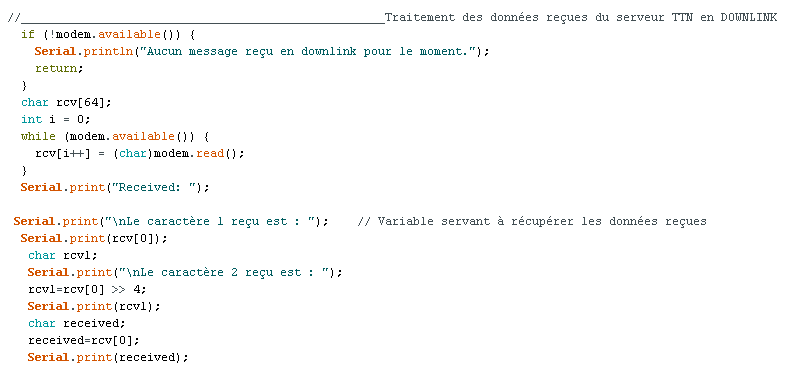
Les données sont envoyées au serveur TTN grâce à l’objet *modem* de la librairie du *MKRWAN*.



*Figure 7 : Code d’envoi de données vers TTN*

* **Traitement des données reçus depuis TTN:**

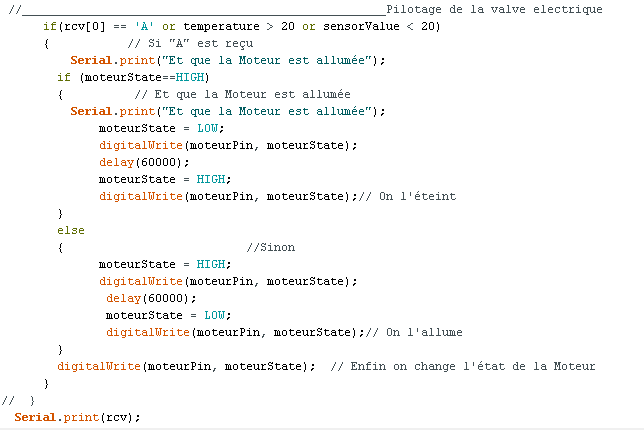
Ce code permet de lire la données reçues sur la carte arduino à partir du réseau TTN. Cette donnée est envoyée depuis le serveur Node-Red par l’utilisateur.



*Figure 8*: Programme de traitement des données TTN

* **Pilotage de la valve électrique**

On déclenche la valve pendant une minute lorsque la donnée reçue en Downlink est ‘A’ ou que la température > 20 ° C ou que l’humidité < 20 %.



*Figure 9*: Programme de pilotage de l'électrovanne

## **4.2 Application Cloud TTN**

The Things Network (TTN) est un ensemble d'outils ouverts et un réseau mondial pour construire une application IoT à faible coût en utilisant la technologie LoRa, avec une sécurité maximale et une capacité d'évolution.

C'est une application cloud gratuite et Open Source. En revanche, le fait d’utiliser TTN impose à ceux qui enregistrent des Gateways LoRa de les mettre à disposition de tous les autres utilisateurs de TTN. L’objectif de la mise en place de cette application est de réaliser un réseau global ouvert.

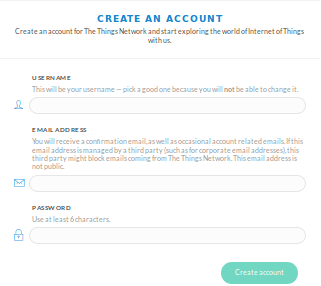
Pour pouvoir utiliser l'application TTN il faut d'abord créer un compte sur le site web <https://account.thethingsnetwork.org/register>. Le compte va permettre d'identifier l'utilisateur sur l'application cloud mais aussi d'identifier les données de l'utilisateur dans le réseau LoRa qui est ouvert et gratuit.

Ensuite enregistrer le matériel Iot qui va échanger des données avec l'application cloud via le réseau LoRa. L'enregistrement est précédé de la création de l'equipement sur l'application qui sera créée sur le cloud.

Cependant il est important de savoir que les données reçues de la carte *arduino MKR 130*0 est sous format hexadécimal donc il est nécessaire de mettre en place une fonction de décodage pour transformer les données en ascii.

### **4.2.1 Création du compte TTN**

La création du compte est une étape très simple, il faut se rendre sur le site web officiel puis cliquer sur **Sign In** (lien <https://account.thethingsnetwork.org/register>), renseigner les informations demandées (login, email et mot de passe) et enfin valider la création du compte avec l'email reçu.

*Figure 10*: Création compte TTN

Le compte créé va permettre de créer des applications qui peuvent être des projets IoT différentes et chaque application contient une ou plusieurs équipements qui sont des matériels IoT connectés au réseau LoRaWAN.

### 4.2.2 **Création de l'application**

La création de l'application nécessite de se connecter à votre compte TTN puis se rendre sur le console qui va se présenter comme suit :

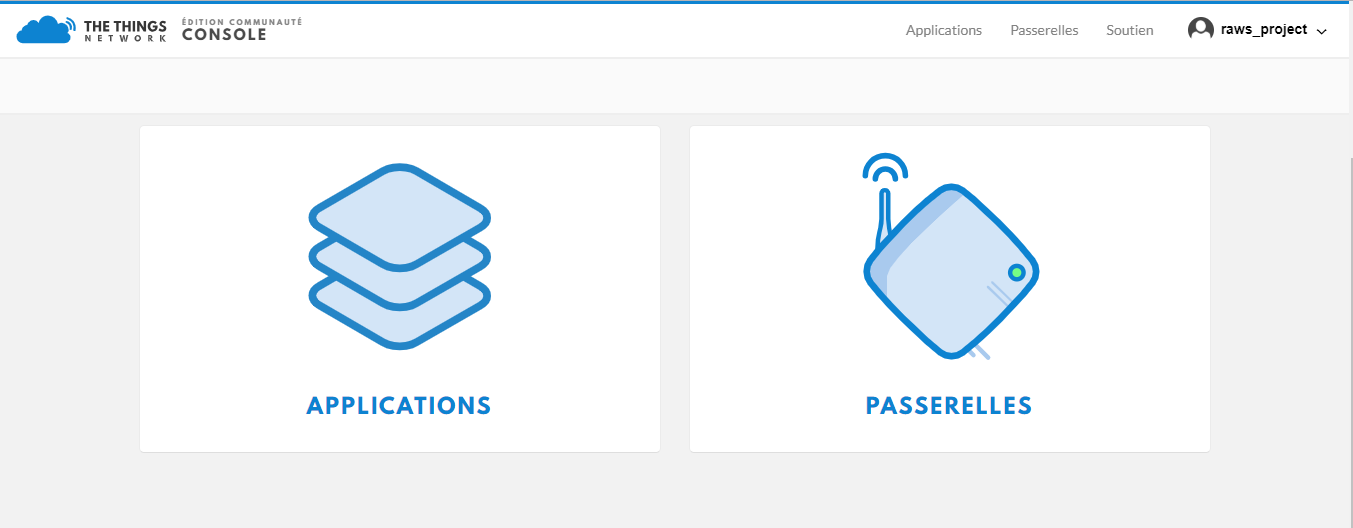


Figure:

Pour ajouter une application cliquer sur **APPLICATIONS**, puis renseigner les informations demandées à savoir l'identifiant de l'application qui ne doit pas contenir d'espace et la description de l'application (les autres informations ne sont pas nécessaire à renseigner) et enfin enregistrer.

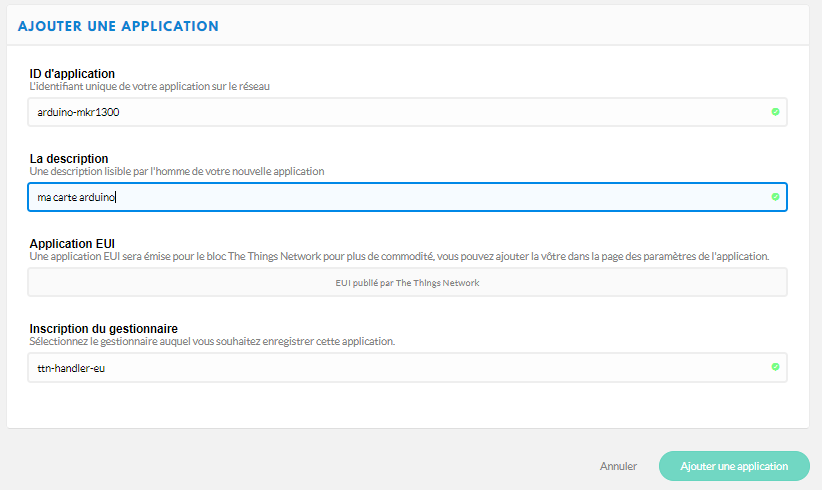


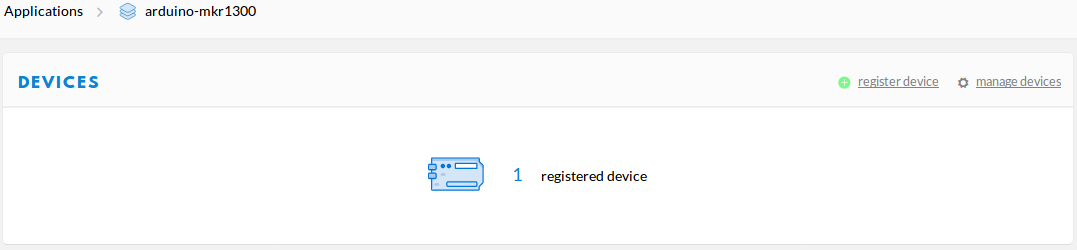
Figure 11: Création de l’application

L'application ainsi créée contiendra le ou les différent(s) équipement(s) utilisé(s). Dans notre cas elle contiendra comme matériel la carte arduino MKR qui va collecter et envoyer les données à notre serveurs via le réseau LoRa.

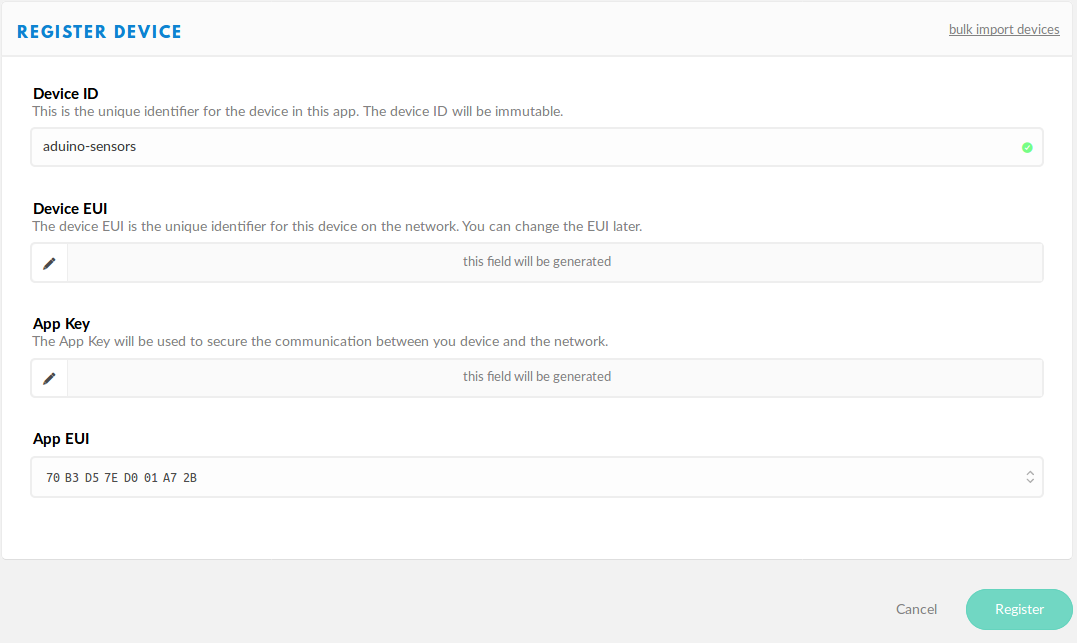
### **4.2.3 Création de l’équipement**

L'équipement qui sera créé est le matériel connecté à l'autre bout qui permet d'envoyer des données vers l'application TTN ou de recevoir des informations de TTN. Dans ce projet l'équipement dont il s'agit est la carte arduino MKR 1300.

L'équipement est créé en allant dans l'application créer sur le rubrique device cliquez sur *registered device.*

Figure 12: Rubrique device

Il reste plus qu’à renseigner le nom du device et enregistre.



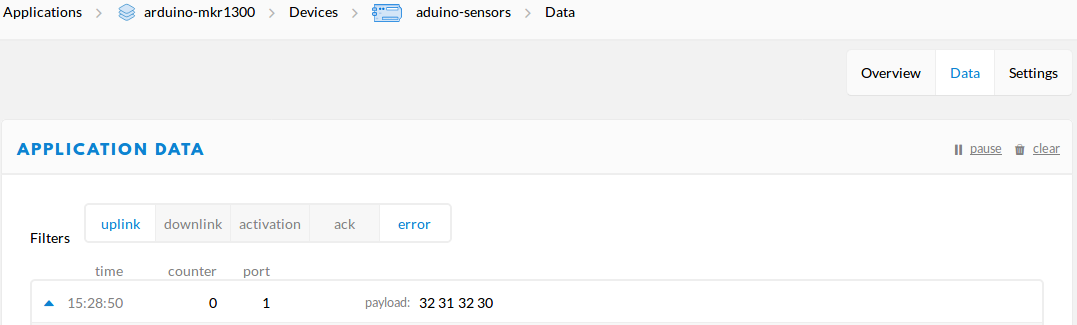
*Figure 13: Création du matériel*

Maintenant que l'application et l'équipement sont créés, l'équipement physique c'est à dire la carte *Arduino MKR 1300* pourra envoyées les données collectées à l'équipement TTN.

### **4.2.4 Formatage des données**

Il faut savoir que par défaut les informations reçues en **uplink** par TTN sont sous le format **hexadecimal** d'ou l'importance de formater ces informations pour ressortir les données temperature et humidite envoyées par la carte arduino.

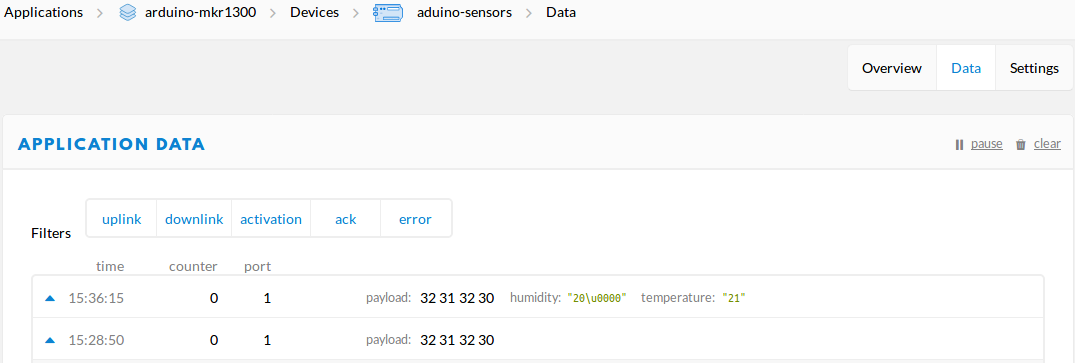
Les données reçues sans le formatage seront présentés sous la forme suivant:

Figure 14: Data Application

Au niveau de l'application créée sur TTN, on trouve une rubrique ***Payload Formats*** qui contient une autre rubrique ***decoder*** qui permettra d'écrire un programme en javascript pour **decoder** les données reçues en **uplink**  c'est à dire de la carte arduino vers TTN.

Figure 15: Fonction de décodage

Et après l'ajout de la fonction decoder les informations vont se présenter sous la forme suivante:

Figure 16: Résultat décodage 

## **4.3. Application Node-Red**

Node-red est une application codée en **javascript** qui permet de visualiser et de traiter des données.Cependant il existe d’autres moyens de traitement et de visualisation en Iot beaucoup plus sophistiqué qui sont utilisés par les industrielles qui traite du **BIG DATA** comme *Cayenne*, *IBM Watson IoT platform, AT&T M2X* … Puisque nous avons un projet qui ne traite pas méga donnée *node-red* répond à nos exigences. Elle est très simple à utiliser et intègre des modules à installer pour un traitement spécifique.

Pour le projet on va ajouter le module **MQTT** pour les informations en ***uplink*** de TTN vers la base de données influxDB et en ***downlink*** de l'interface utilisateur vers la carte ***arduino MKR***. Tous les nodes seront dans un même un flow. Un flow est la plage qui intègre tous les nodes à utiliser pour la réalisation. Le *flow* se présente comme suit:

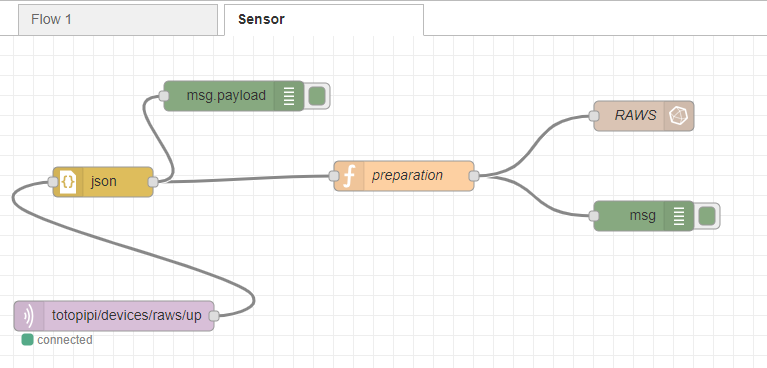


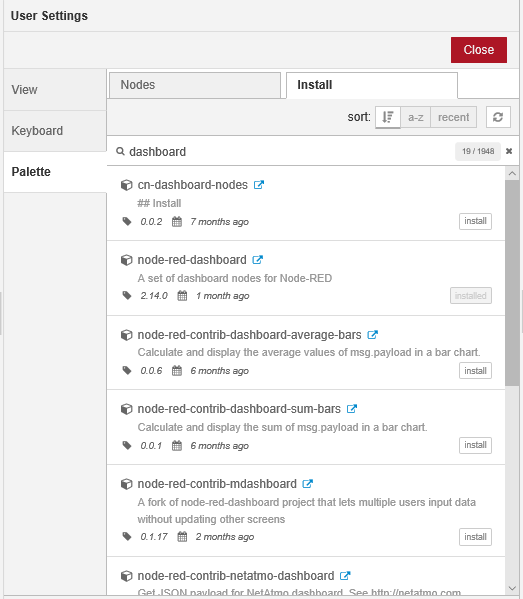
Figure 17: Flow uplink de TTN vers la base de donnée

Figure 18: Flow downlink de l’interface utilisateur et la carte Arduino MKR

### **4.3.1 Ajouter des palettes**

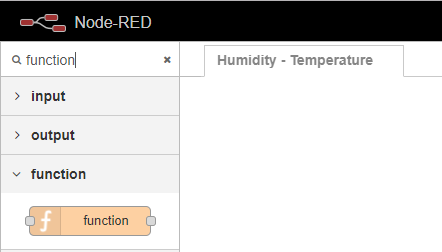
L’ajout des palettes se fait soit en ligne de commande soit sur l’interface de l’application. La deuxième méthode sera utilisé car elle est la plus simple.

Avant tout il faut démarrer l’application en ligne de commande avec la commande suivante **node-red start** puis accéder à l’application au niveau du navigateur et renseigner le lien <http://localhost:1880>. Enfin installer les palettes en allant sur le **menu de l’application**  puis cliquer sur **manager palettes** et installer les modules suivantes: **node-red-node-serialport,** **node-red-dashboard** et **node-red-node-influxDB**. Le node serialport permet de faire les tests en local, le node dashboard permet de visualiser les données reçues sous forme de graphes, gauges, etc et le node influxDB permet se connecter à la base de données influxDB et egalement de creer les tables de la base et d’enregistrer les données.

****

*Figure 19*: Installation des nodes

Après l’ajout des modules, on peut observer la présence des nouveaux nodes à gauche de l’interface de node-red.



*Figure 20: Ajout d’un node*

### **4.3.2 Le protocole MQTT et configuration du node**

C’est un protocole qui est utilisé dans les réseaux Iot et est utilisé avec différents technologie comme le raspberry pi.

Dans ce projet le protocole MQTT (Mosquitto) intervient entre le serveur TTN et l’application node-red. Il permet de transmettre des informations en uplink comme en downlink entre l’application TTN et la base de donnée influxDB ou de l’interface utilisateur Grafana et l’application TTN.

Node-red intègre par défaut les nodes MQTT en uplink et en downlink qui vont permettre de remplir les fonctions d’envoie et de réception d’informations.

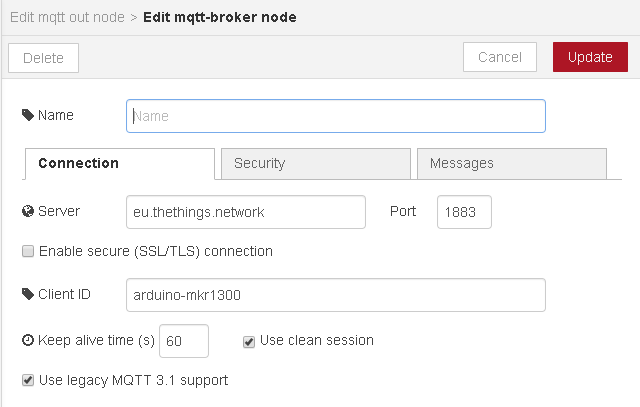
En uplink les données seront sous forme de chaîne de caractère, de ce faits, il faut une fonction JSON pour mettre les données dans un objet json et en downlink un bouton sera mis en place pour l‘envoi de l’information ‘A’ qui va permet à la carte d’actionner l’ouverture de l'électrovanne.

* **Configuration de MQTT en uplink**:

Après l’ajout du node MQTT au flow, il faut renseigner les informations nécessaire pour le faire connecter à TTN. Les informations de connexion de TTN seront retrouvées dans l’API référence du site de officiel de TTN <https://www.thethingsnetwork.org/docs/applications/mqtt/api.html>. Il faut d’abord configurer le serveur avec les informations suivantes:

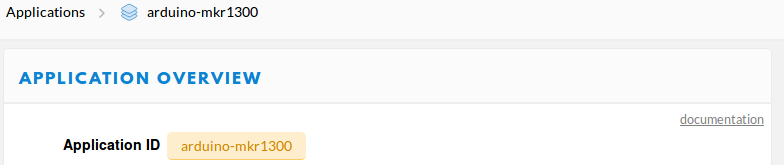
* **Hôte**: eu.thethings.network où eu est la <Region> qui est la dernière partie du gestionnaire que vous avez enregistré votre application.
* **Port**: 1883 ou 8883 pour TLS

Il faut renseigner les informations de connexion à savoir le serveur, le client et le port trouvés dans l’API.



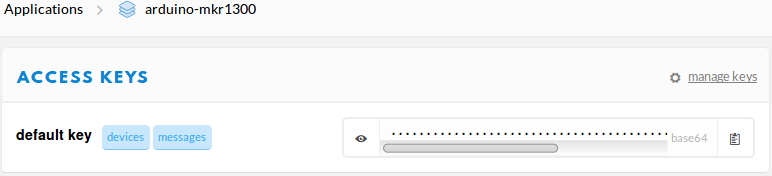
*Figure 21: Node mqtt connexion*

* **Username**: l’identifiant de l’application TTN est le username.



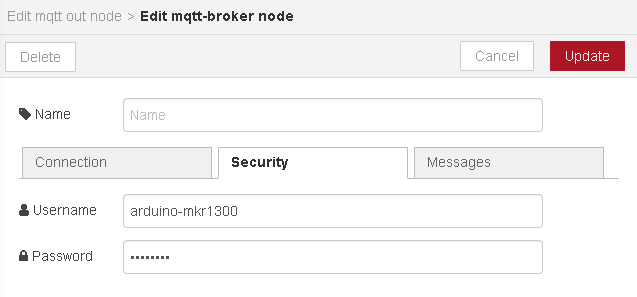
*Figure 22: TTN Application ID*

* **Password**: La clef d'accès de l’application TTN doit être renseigné.



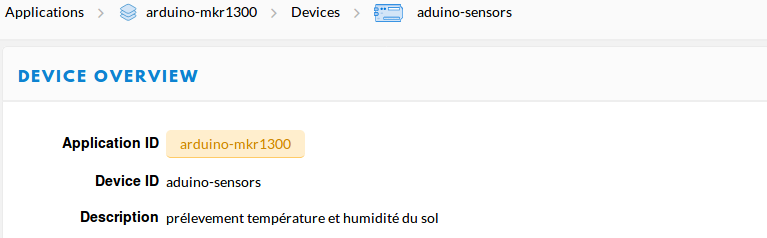
*Figure 23: TTN default key devices messages*

À ce niveau il faut renseigner le username et le mot de passe pour avoir accéder aux données reçues par TTN.



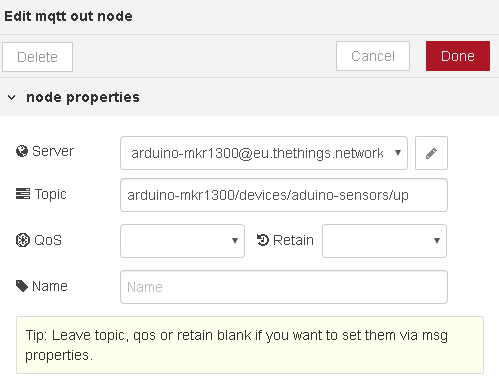
*Figure 24: Configuration de la sécurité*

Ces informations permettent d'identifier les messages ou paquets qui sont destinés à l’utilisateur.



*Figure 25: Node MQTT configuration serveur*

Ensuite il faut configurer le **topic** avec les informations en les renseignant comme indiqué ci-dessous.



*Figure 26: Node MQTT configuration en uplink*

Cette configuration permet de spécifier les informations en uplink.

* **Configuration de MQTT en downlink**:

La seule différence entre la configuration du uplink et du downlink est la configuration du topic. La configuration est en uplink pour le mettre en downlink il faut juste remplacer ***up*** par ***down***.

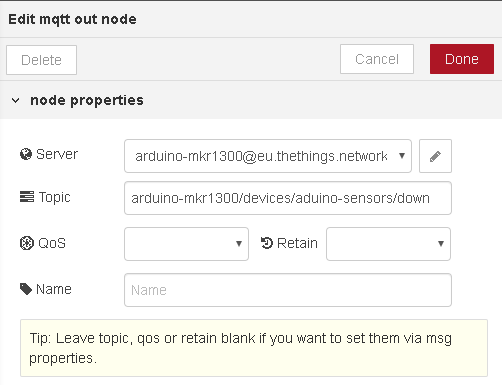


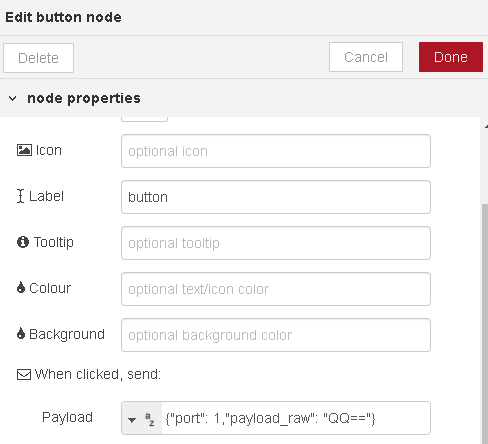
Figure 27: Node MQTT configuration en downlink

* **Configuration du bouton**:

Le bouton permet de déclencher la descente de l’information ‘A’ qui sera envoyé à la carte MKR pour actionner l'électrovanne pour arroser la plante. Il faut noter que la node MQTT prendre une valeur spécifique en entrée qui est un objet json qui contient deux variables:

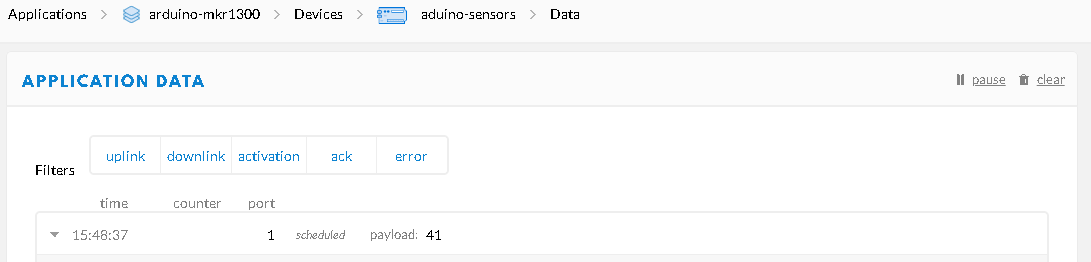
* la variable port: pour spécifier le ***port*** d'accès
* et la variable ***payload\_raw*** qui contient l’information à envoyer en base 64. ***‘A’*** en base64 vaut ***“QQ==”***

La configuration du node se fait comme suit:



*Figure 28: Configuration Node button*

La configuration des nodes est terminé maintenant, on peut déployer le flow. En actionnant le button sur l’interface <http://localhost:1880/ui>, on voit sur le device de TTN la donnée reçue en hexadécimal:



*Figure 29: Résultat configuration button*

### **4.3.3 Influxdb et configuration du node**

InfluxDB a été conçue pour conserver, analyser et injecter en temps réel des données de séries temporelles (métriques et événements) avec de fortes exigences en termes de disponibilité et de performances.

Comparer aux autres technologies, elle est la plus adaptée au réseau Iot raison pour laquelle il est utilisé pour réalisation de ce projet.

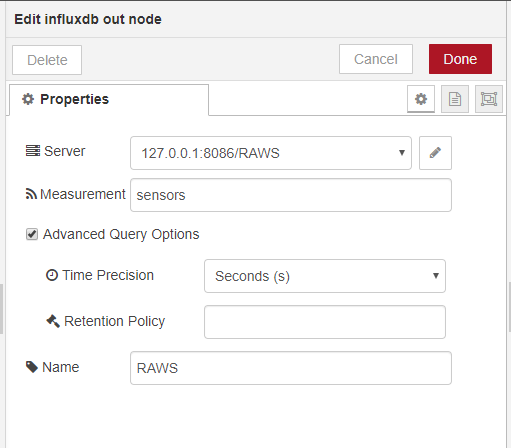
La configuration de la base est une chose très facile, il suffit juste de créer une base de donnée au niveau de influxDB et le node influxDB va se charger de la création des tables et de l’insertion des données.

* **Création de la base de données**:

**InfluxDB** est accessible en ligne de commande de ce faits la création de la base de données se fait en console et la commande de création de la base est identique à celle de **SQL *create database*** *nom\_base*. Le nom de la base de données est ***RAWS****.*

* **Configuration de la node**:

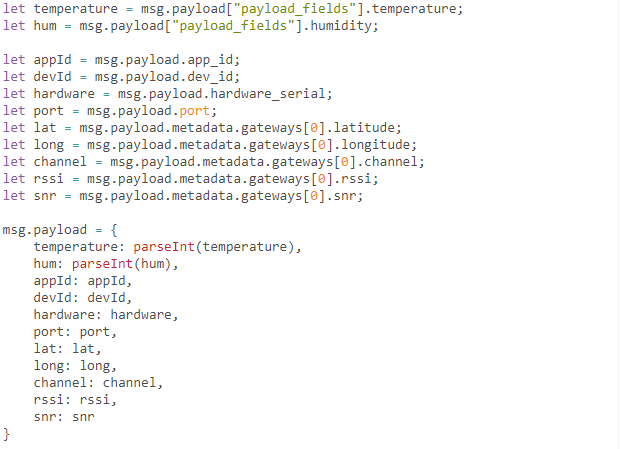
Après ajout du node de la base de donnée influxDB dans le ***flow***du projet, il faut juste le configurer en renseignant l’*adresse ip du serveur* ***localhost***, le *port* utilisé ***8086***, le nom la base de donnée ***RAWS*** et le nom de la table ***sensors***.



*Figure 30: Configuration node influxDB*

Il faut noter que la table ***sensors*** sera créée automatiquement lors du déploiement du flow et les attributs de la table seront créés en fonction de la valeur d'entrée du node. C’est pourquoi il est nécessaire de formater la valeur d'entrée.

Le formatage des données est faits grâce au node *function* qui reçoit les données reçues de MQTT sous format JSON.



*Figure 31:* Function formatage des données à enregistrer

Les données reçues contiennent beaucoup d’informations de ce faits on extrait juste les informations nécessaires: la température, l'humidité, l’identifiant de l’application, l’identifiant du matériel, le matériel du gateway, le port utilisé, la latitude et la longitude du gateway, le canal, la puissance du signal et le bruit.

Ces informations sont insérées dans un objet *msg.payload* qui va être retournée au node influxDB et les informations seront automatiquement enregistrées.

## **4.5 Application Grafana**

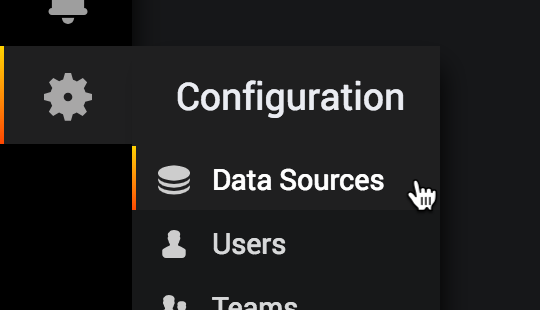
La surveillance des serveurs se fait souvent à l'aide de graphiques. Grafana propose une interface moderne, personnalisable et est une application open source.

Il est utilisé dans le projet pour visualiser toutes les informations intéressantes reçues du gateway LoRaWAN.

### **4.5.1 Ajouter des données sources**

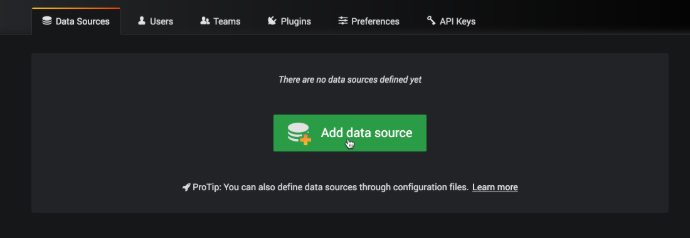
Avant de créer votre premier tableau de bord, vous devez ajouter votre source de données.

Déplacez d'abord votre curseur sur le rouage situé sur le menu latéral qui vous montrera le menu de configuration. Si le menu latéral n'est pas visible, cliquez sur l'icône Grafana dans le coin supérieur gauche. Le premier élément du menu de configuration concerne les sources de données. Cliquez dessus pour accéder à la page des sources de données où vous pouvez ajouter et modifier des sources de données. Vous pouvez également simplement cliquer sur le rouage.



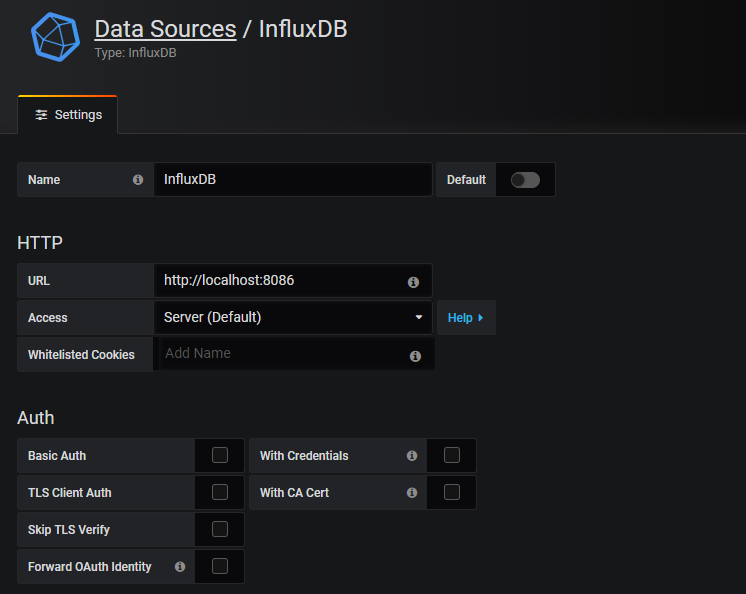
*Figure 32:Configuration data source*

Cliquez sur Ajouter une source de données et vous arriverez à la page des paramètres de votre nouvelle source de données.



*Figure 33: Configuration data source*

Commencez par attribuer un nom à la source de données, puis sélectionnez le type de source de données que vous souhaitez créer



*Figure 34:Configuration InfluxDB*

Après avoir configuré votre source de données, vous êtes prêt à enregistrer et à tester.

### **4.5.2 Conception des Graphes**

Grafana offre plusieurs types de graphes.

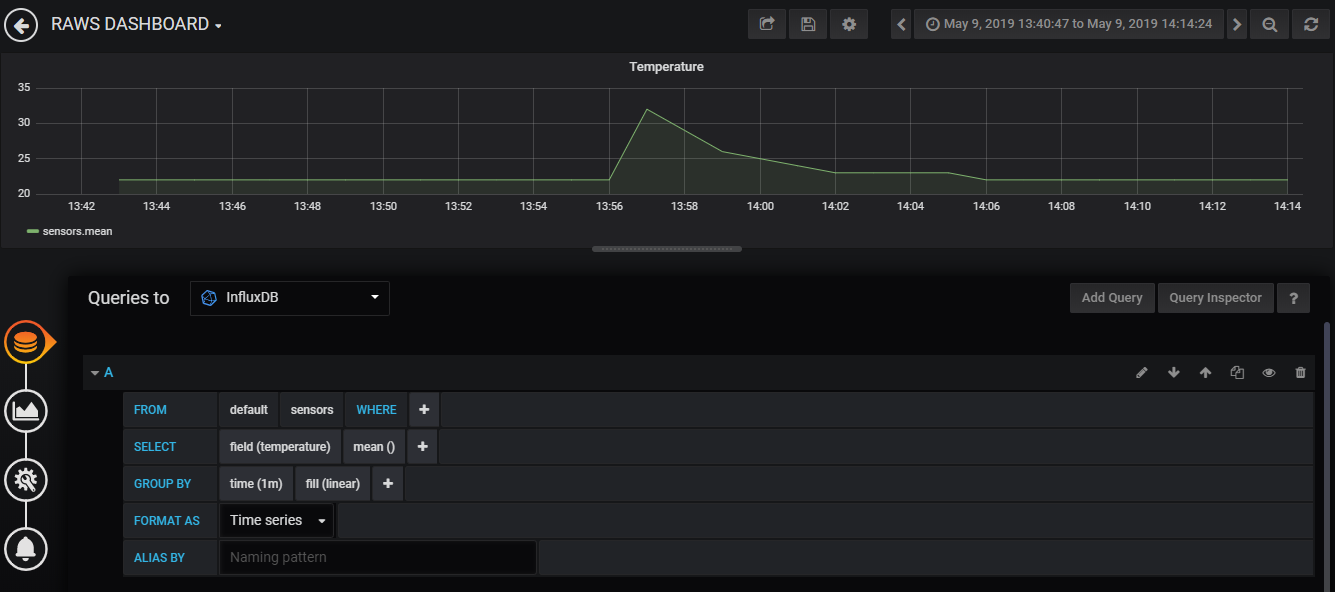
* Vous ajoutez des panneaux en cliquant sur l'icône Ajouter un panneau dans le menu supérieur.
* Pour modifier le graphique, cliquez sur son titre pour ouvrir le menu du panneau, puis sur Modifier.
* Cela devrait vous mener à l'onglet Métriques. Dans cet onglet, vous devriez voir l'éditeur de votre source de données par défaut.

Lorsque vous cliquez sur l'onglet Métriques, un éditeur de requête spécifique à la source de données du panneau vous est présenté. Utilisez l'éditeur de requêtes pour créer vos requêtes et Grafana les visualiser en temps réel.

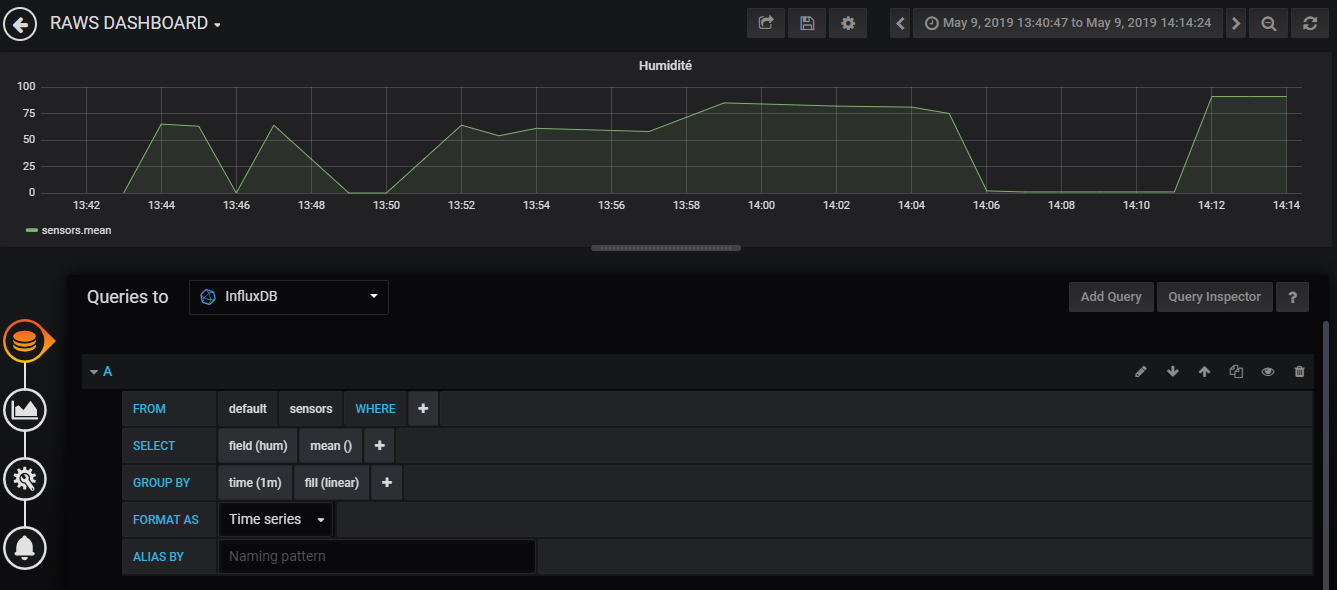
* Pour temperature/humidity graph, on doit choisir le query comme ca:

SELECT **temperature** FROM **sensors**

SELECT **humidity** FROM **sensors**



*Figure 35:Configuration de la graphe de température*

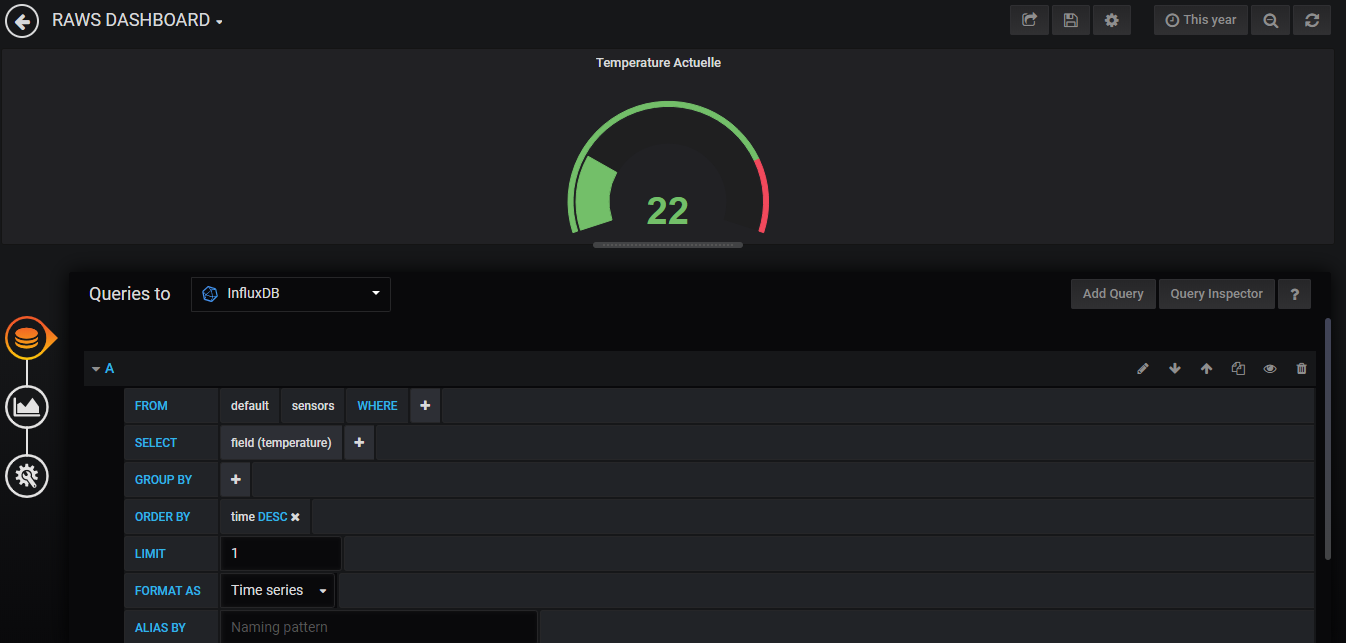


*Figure 36:Configuration de la graphe d'humidité*

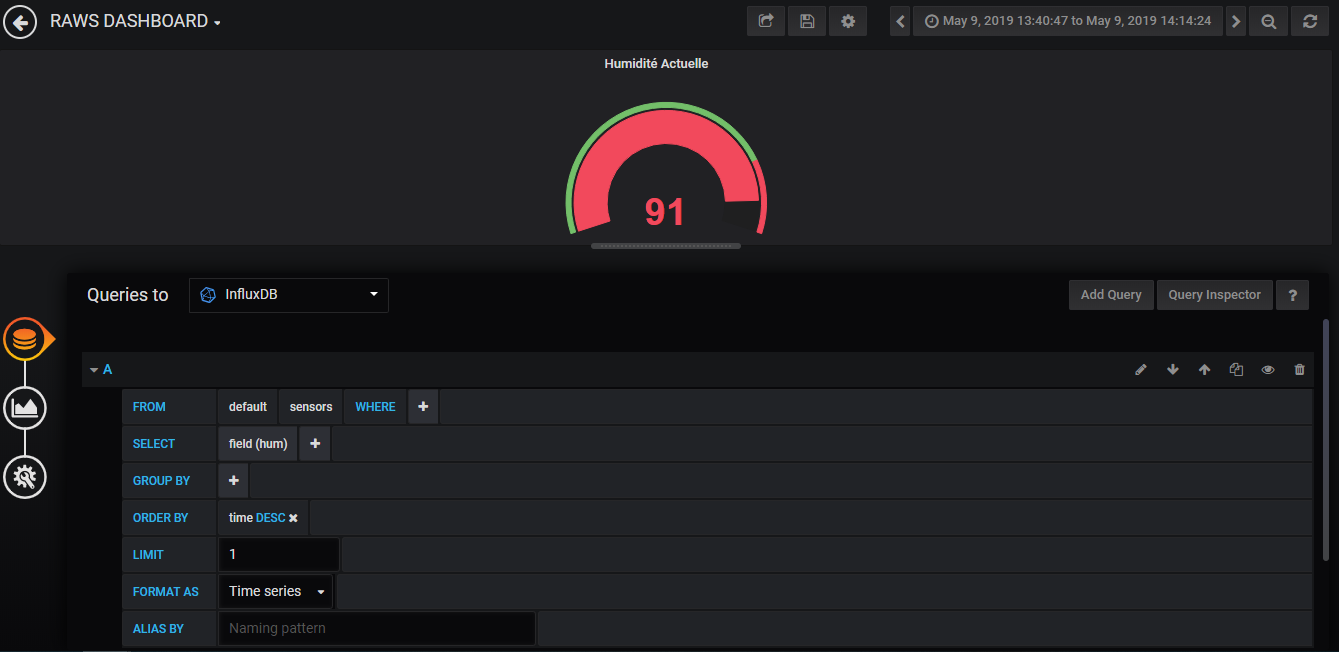
* Pour temperature/humidity actuelle graph, on doit ajouter le condition pour le query comme ca:

*ORDER BY* ***DESC***

*LIMIT* ***1***



*Figure 37: Configuration température actuelle*



*Figure 38:Configuration humidity actuelle*

## **4.6. Test de fonctionnement**

Après avoir implémenté notre solution, nous avons implémenter des tests qui nous ont permis de pouvoir allumer la valve électrique soit lorsque la température > 20°C ou que l'humidité du sol est < 20% ou en cliquant un bouton au travers de l’interface graphique de Node-Red.

Les données collectés ont été enregistrés dans une bas de données influxdb. Pour les visualiser on peut utiliser l’application grafana.

Ce qui donne le résultat suivant:



Figure 39 : Visualisation de la température et de l’humidité en temps réel sur grafana

## **4.7. Difficultés rencontrés**

Les problèmes que nous avons rencontrés sont :

* **Le retard dans la réception du matériel**

En effet le matériel n’étant pas disponible, il a dû être commandé par le CERI et cela a pris du temps pour le livrer et pour qu’on le reçoive. Ce qui nous a retardés dans la réalisation du projet.

* **Le Gateway**:

Pour réaliser le projet nous avons forcément besoin de nous connecter au Gateway. Celui-ci se trouvant au CERI, nous devons nous déplacer sur place pour pouvoir nous y connecter car il n y a pas de Gateway proche de notre lieu d’habitation.

* **Le nombre limité de message en uplink**

Suite à la réglementation du serveur TTN, nous ne pouvons envoyer qu’un nombre limité de message vers le serveur. Ainsi nous avons limité le nombre de message à 1 par minute.

Mais il peut arriver d’épuiser notre quota de message pour la journée.

# 

# **5. CONCLUSION**

En fonction des exigences du cahier de charges, nous avons pu réaliser un système automatique d’irrigation à distance avec le protocole LORA, la carte arduino et les données des capteurs, et le Réseaux TTN.

L'implémentation de ce projet nous a permis de prendre en compte les besoins de la plante et de s'approvisionner en eau tout en réduisant le le gaspillage.

Ce qui démontre la bonne application l’internet des objets au domaine de l’agriculture.

Ainsi l’internet des objets peut représenter une solution au gaspillage de l’eau sur la planète.

Ce projet va permettre au agriculteurs de mieux comprendre les besoins de la plante et de l’arroser au moment où cela sera nécessaire automatiquement ou à distance.

Ce qui va accroître la production et palier à la surconsommation en eau.

# **ANNEXES**

## **Installation de node-red**

Pour installer node-red sous une machine linux il faut taper en ligne de commande les instructions suivantes:

1. sudo npm install -g --unsafe-perm node-red
2. Node-red start
3. Open<http://localhost:1880>

La documentation de node-red est accessible via ce lien <https://nodered.org/docs/>.

## **Installation and configuration**

* Pour Windows
* Installation pour Windows

Télécharger le zip fichier dans ce source

<https://grafana.com/grafana/download?platform=windows>

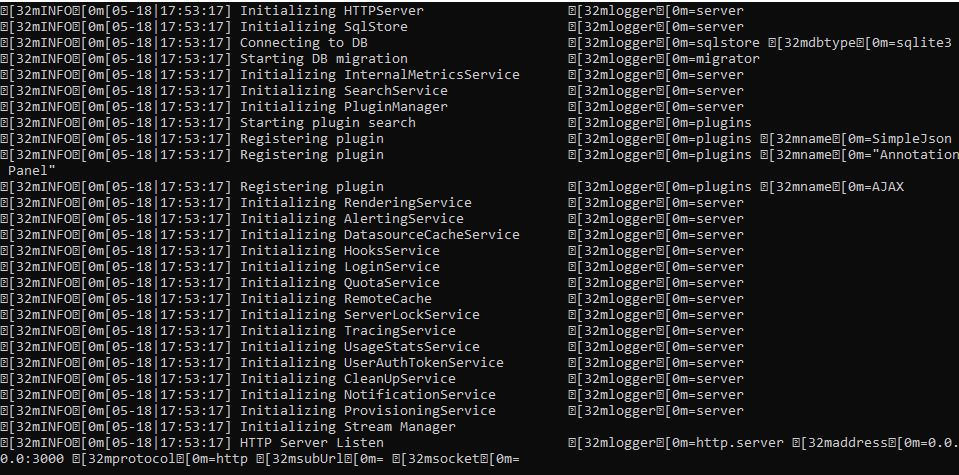
Le fichier zip contient un dossier avec la version actuelle de Grafana. Extrayez ce dossier là où vous voulez que Grafana s'exécute:

* Configuration

Le port Grafana par défaut est **3000**. Ce port nécessite des autorisations supplémentaires sur Windows. Modifiez custom.ini et décommentez l'option de configuration http\_port (; est le caractère de commentaire dans les fichiers ini) et remplacez-la par quelque chose comme **8080** ou similaire. Ce port ne devrait pas nécessiter de privilèges Windows supplémentaires.

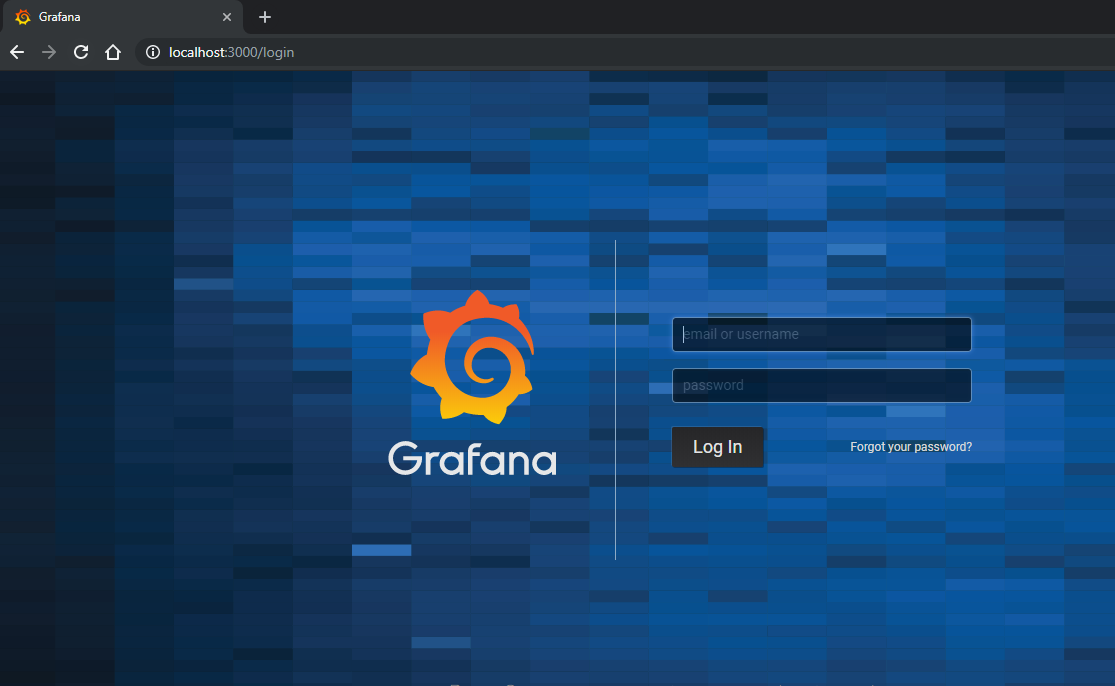
Identifiant et mot de passe par défaut est **admin** / **admin.**

Démarrez Grafana en exécutant **grafana-server.exe**, situé dans le répertoire bin, de préférence à partir de la ligne de commande. Si vous voulez exécuter Grafana en tant que service Windows, téléchargez NSSM. Il est très facile d'ajouter Grafana en tant que service Windows à l'aide de cet outil.



*Figure: Démarrez Grafana*

Pour exécuter Grafana, ouvrez votre navigateur et accédez au port que vous avez configuré ci-dessus, par exemple [**http://localhost:3000/**](http://localhost:3000/)**.**



*Figure: Page d'accueil Grafana*

* Pour Linux/Ubuntu
* Installation pour Linux/Ubuntu

**$ sudo yum install** [**https://dl.grafana.com/oss/release/grafana-5.4.2-1.x86\_64.rpm**](https://dl.grafana.com/oss/release/grafana-5.4.2-1.x86_64.rpm)

Vous pouvez démarrer Grafana en lançant:

**$ sudo service grafana-server start**

Cela démarrera le processus grafana-server en tant qu'utilisateur grafana, créé lors de l'installation du paquet. Le port HTTP par défaut est **3000** et l'utilisateur et le groupe par défaut sont admin.

Identifiant et mot de passe par défaut **admin / admin**

* Configuration

Le fichier de configuration est situé à **/etc/grafana/grafana.ini**

# WEBOGRAPHIE

[www.silanus.fr](http://www.silanus.fr)

[www.thethingsnetwork.org](http://www.thethingsnetwork.org)

[www.grafana.com](http://www.grafana.com)

[www.wikipedia.org](http://www.wikipedia.org)

[www.seeedstudio.com](http://www.seeedstudio.com)

<https://lora-alliance.org/about-lorawan>

<https://www.digitalmatter.com/Solutions/Whitepapers/LoRaWAN-Introduction>

<https://nodered.org/docs/getting-started/installation>

<https://www.cooking-hacks.com/forum/viewtopic.php?f=37&t=8856&sid=d19572860b6c1cdb2e248a6926b16837&start=10>

<https://www.cooking-hacks.com/forum/viewtopic.php?f=37&t=8803>