

# Rapport de Travaux Pratiques - 2024.09.29

**Internet des objets - Protocoles LoRaWAN pour la reception en longue distance de données ESP8266 et par modem LoRa E5**

## Table des matières

<b>Introduction.....</b>	<b>2</b>
<b>1.Test du matériel - Tests unitaires.....</b>	<b>3</b>
1.1. Test du capteur DHT22.....	3
Figure 1.1.1 : Diagramme de séquence du relevé des données du capteur.....	3
Figure 1.1.2 : Vu du Serial Monitor validant le test du capteur.....	4
1.2. Test et configuration du module LoRa-E5 pour transmission des données sur TTN...4	
Figure 1.2.1 : Diagramme de séquence de la communication avec le module LoRa-E5.....	5
Figure 1.2.2 : Terminal de communication série montrant le paramétrage du module LoRa-E5.....	6
Figure 1.2.3 : Carte locale des passerelles du 5ème arrondissement de Paris (France), bâtiment Esclangon, du réseau TTN.....	6
Figure 1.2.4 : Diagramme de séquence de la montée de données au réseau LoRaWAN TTN.....	7
Figure 1.2.5 : Visualisation de la réception de données sur le réseau TTN.....	8
<b>2. Envoi des relevés du capteur DHT22 sur Ubidots STEM.....</b>	<b>9</b>
Figure 2.1 : Diagramme de séquence de la transmission de relevés du capteur à Ubidots.....	9
Figure 2.2 : Mise en forme standardisée post-décompression des relevés reçus.....	10
Figure 2.3 : Visualisation de la réception des relevés compactés et de leurs mise en forme sur le réseau TTN.....	11
Figure 2.4 : Graphiques des relevés de température et d'humidité réceptionnés sur Ubidots.....	12
<b>Conclusion.....</b>	<b>13</b>
<b>Annexe.....</b>	<b>15</b>
A. Mise en place de l'environnement de travail.....	15
A.1. Configuration du matériel TTN par étapes.....	15
A.2. Liaison du matériel TTN avec Ubidots.....	17
A.3. Configuration de Arduino IDE pour échanger avec l'ESP.....	20

## Introduction

Dans le cadre de notre parcours universitaire en Électronique et Informatique, nous sommes amenés à interagir avec des objets connectés évoluant dans des environnements peu couverts en WiFi, comme sur le site de Saint-Cyr, ou encore dans des environnements où les ondes radio sont particulièrement pertinentes du fait de leurs basses fréquences et donc meilleurs pénétrations à travers les bâtiments. Ces ondes peuvent se propager sur de longues distances tout en consommant peu d'énergie. Ainsi, nous allons nous familiariser avec le protocole LoRaWAN.

La technologie LoRa est très pertinente dans les cas où l'environnement est ouvert et où il est nécessaire de transmettre des informations sur de longues distances sans devoir installer des antennes à intervalles courts et réguliers. Par exemple, nos copains d'AGRAL pourraient avoir besoin de mettre en réseau des arroseurs automatiques pour des champs intelligents capables d'adapter l'irrigation à partir de capteurs mesurant le taux de CO<sub>2</sub> ou l'humidité du sol, plutôt que de le faire de manière inutilement fréquente. Toujours dans une optique environnementale et économique, d'autant plus importante avec la hausse des prix de l'électricité suite aux événements en Ukraine, un éclairage public ultra-adapté pourrait fonctionner grâce à de multiples capteurs, n'éclairant que quelques lampadaires à l'avance pour les véhicules ou piétons, tout en réduisant la pollution visuelle nocturne pour le voisinage.

Pour résumer, le protocole LoRaWAN est très pertinent pour gérer les échanges entre capteurs et serveurs, notamment pour des relevés de données ou la détection de pannes et fuites énergétiques. Ce protocole permet aux capteurs de communiquer avec une passerelle, puis d'envoyer les données vers un réseau plus large. Il nous permet ainsi d'assurer une grande autonomie des capteurs à distance, tout en optimisant leur consommation d'énergie.

Pour une meilleure gestion, portativité et communication de notre projet, nous avons, sur les conseils de Thibault, mis en place un suivi avec [GitHub](#) et une remise en ligne via [Google Doc](#).

Aide rédactionnelle, au débogage, au code et soutien : Yulin, Daniel, Maxime, Ayman, Sékouba, Victor, Quentin, Nicolas, ChatGPT, HARIAN Elyoth, DOUZE Yann, Benjamin et l'incroyable Monsieur VIATEUR Sylvain.

## 1. Test du matériel - Tests unitaires.

Après avoir configuré l'environnement de travail disponible dans l'annexe [A. Mise en place de l'environnement de travail](#), nous allons vérifier le capteur DHT22 et le module LoRa-E5. Dans un second temps, nous enverrons des messages depuis l'ESP pour pouvoir enfin transmettre des relevés via le capteur DHT22.

### 1.1. Test du capteur DHT22

Nous avons connecté le capteur, à la borne D7 du microcontrôleur, correspondant d'après la documentation technique à la sortie hardware GPIO 13, pour tester le bon fonctionnement du capteur DHT22. En annexe, nous avons la [A.3. Configuration de Arduino IDE pour échanger avec l'ESP](#).

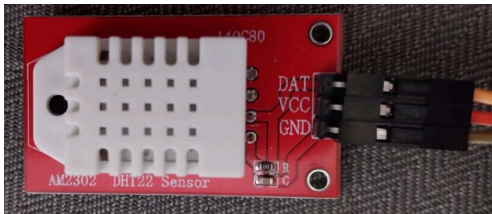


Figure 1.1.1 : Diagramme de séquence du relevé des données du capteur.

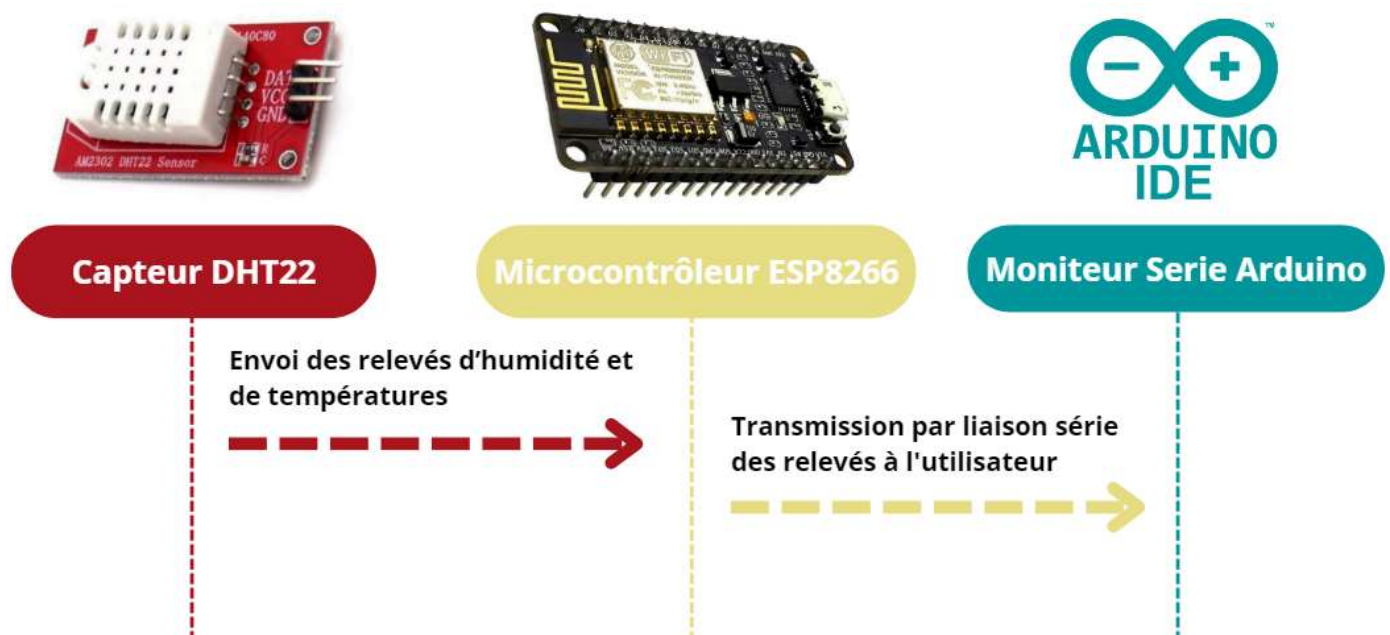


Figure 1.1.2 : Vu du Serial Monitor validant le test du capteur.

```
Output  Serial Monitor  x
Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM3')

Debut des relevees capteur DTH22 avec un intervalle de mesure de 0,5 s
Temperature : 24.10 °C
Humidite : 80.70 %
Temperature : 24.10 °C
Humidite : 80.70 %
Temperature : 24.10 °C
Humidite : 82.60 %
```

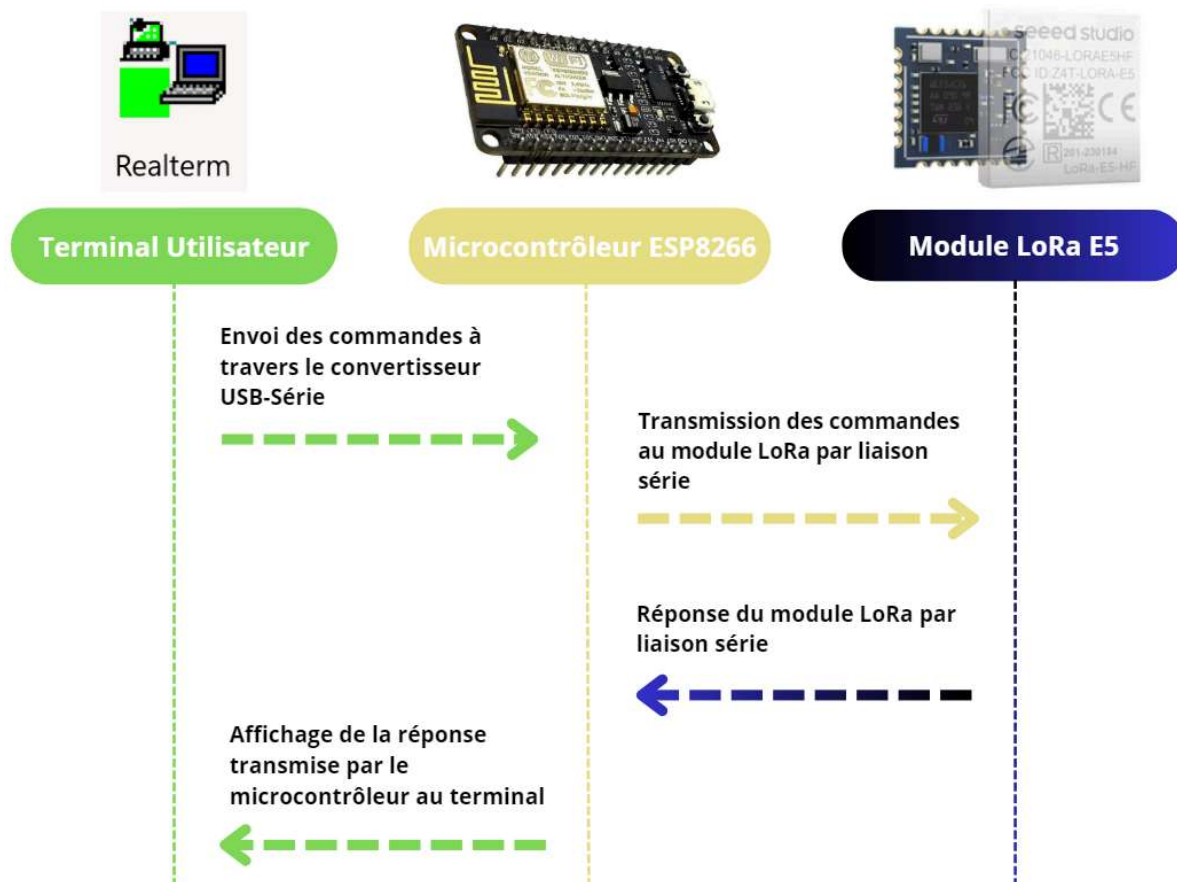
Les données du capteur d'humidité et de température DHT22 fluctuent de manière cohérente avec les interactions expérimentales et s'affichent correctement sur le Serial Monitor, ce qui valide le test.

## 1.2. Test et configuration du module LoRa-E5 pour transmission des données sur TTN

Dans un environnement peu couvert en Wi-Fi, comme sur notre site de Saint-Cyr, les technologies de communication radio à basse fréquence sont souvent plus pertinentes. Le protocole LoRa ( Longue Distance ), faisant partie des technologies LPWAN ( Réseau Étendu à Basse Consommation), se distingue par sa capacité à transmettre des données sur de longues distances tout en consommant peu d'énergie. Ce type de réseau est particulièrement adapté aux objets connectés qui doivent fonctionner sur des batteries pour des durées prolongées, comme vu en introduction avec les capteurs environnementaux ou des systèmes intelligents d'irrigation ou d'éclairage public.

Ainsi, après avoir configuré, le matériel TTN à l'[A.1. Configuration du matériel TTN par étapes](#). Nous allons connecter le Grove LoRa-E5 à un convertisseur sous 3,3V pour la liaison série vers USB afin de pouvoir visualiser et communiquer avec le LoRa-E5 de manière filaire. Nous utiliserons [Realterm](#) comme terminal pour la communication série et les commandes AT du module LoRa-E5 disponible au lien suivant : [AT Commandes](#).

Figure 1.2.1 : Diagramme de séquence de la communication avec le module LoRa-E5.

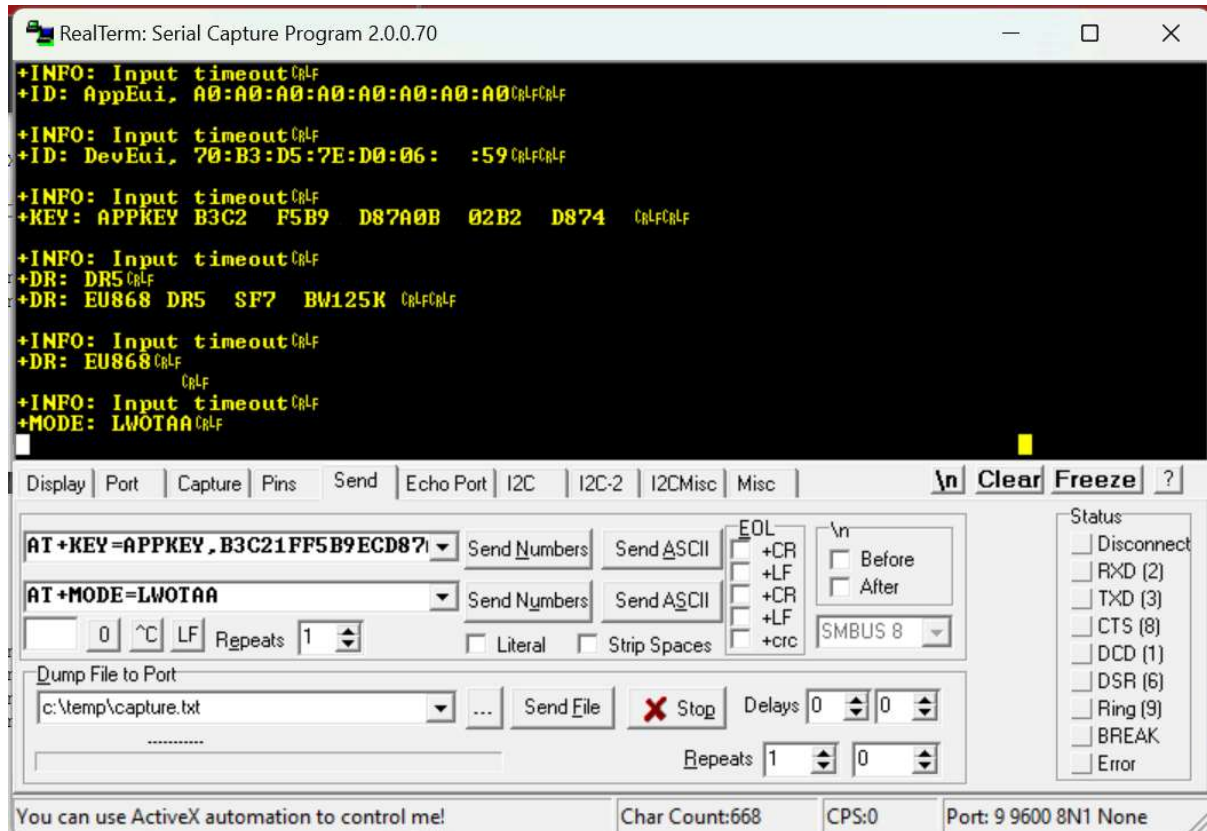


Pour cette partie, on cherche à communiquer avec le module LoRa E5 via des commandes AT simples de configuration depuis un terminal ( Realterm ).

Pour tester la liaison et le bon paramétrage du terminal de communication, nous avons tout d'abord utilisé la commande "AT", qui renvoie "OK". Ainsi, nous pouvons conclure à la bonne connexion du convertisseur et du module LoRa avec le PC via la liaison série. On note par ailleurs les caractères CR LF ( Carriage Return Line Feed ) sur le terminal, qui indiquent simplement une fin et un retour à la ligne.



Figure 1.2.2 : Terminal de communication série montrant le paramétrage du module LoRa-E5.



Pour pouvoir se connecter au réseau, nous avons relié les données du réseau TTN et du module LoRa. On note que DevAddr est l'adresse locale assignée automatiquement par le réseau via OTAA.

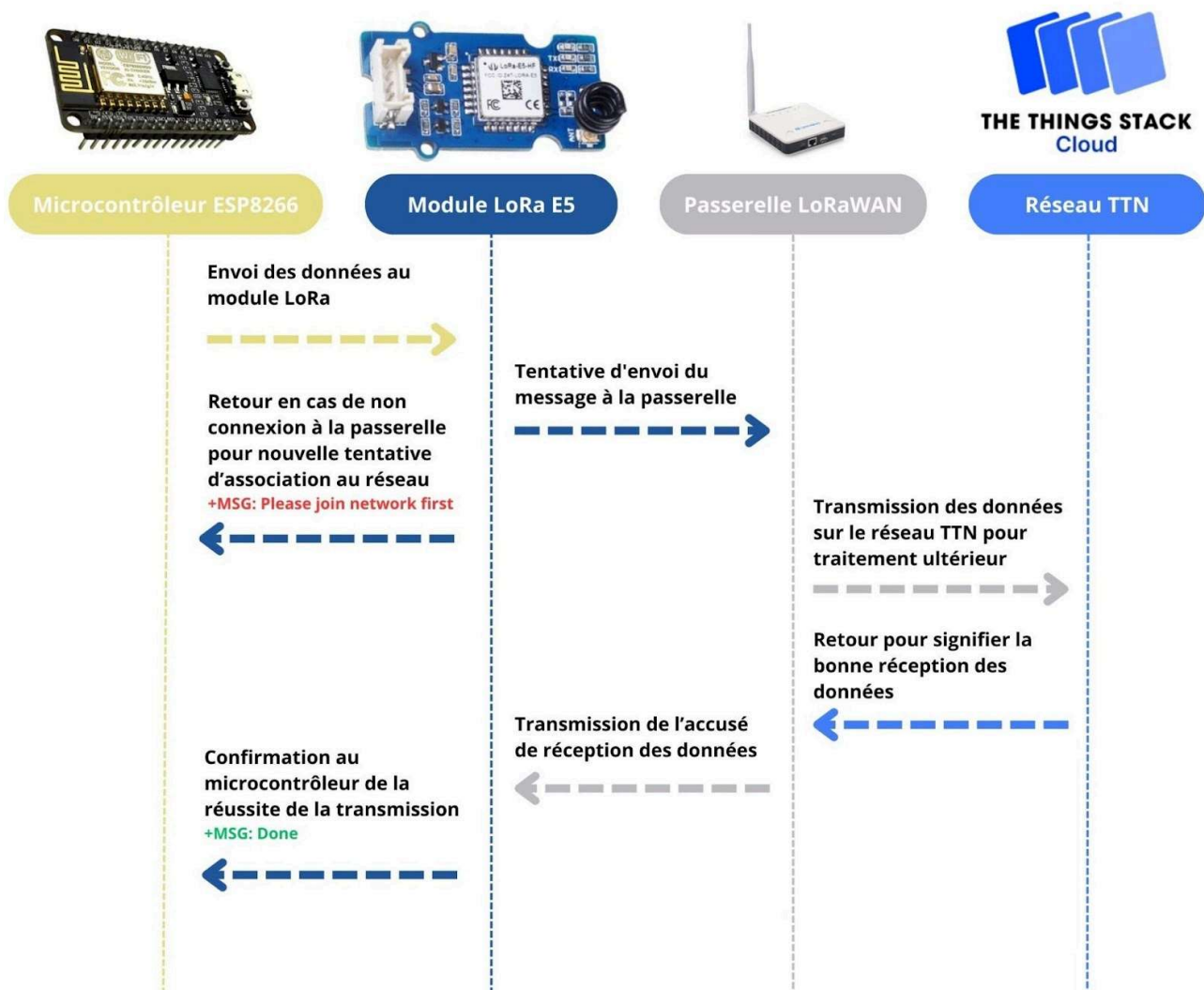
Enfin, nous allons chercher une passerelle pour nous connecter au réseau. Les passerelles TTN sont répertoriées à l'adresse internet : [www.thethingsnetwork.org/map](http://www.thethingsnetwork.org/map)

Figure 1.2.3 : Carte locale des passerelles du 5ème arrondissement de Paris (France), bâtiment Esclangon, du réseau TTN.



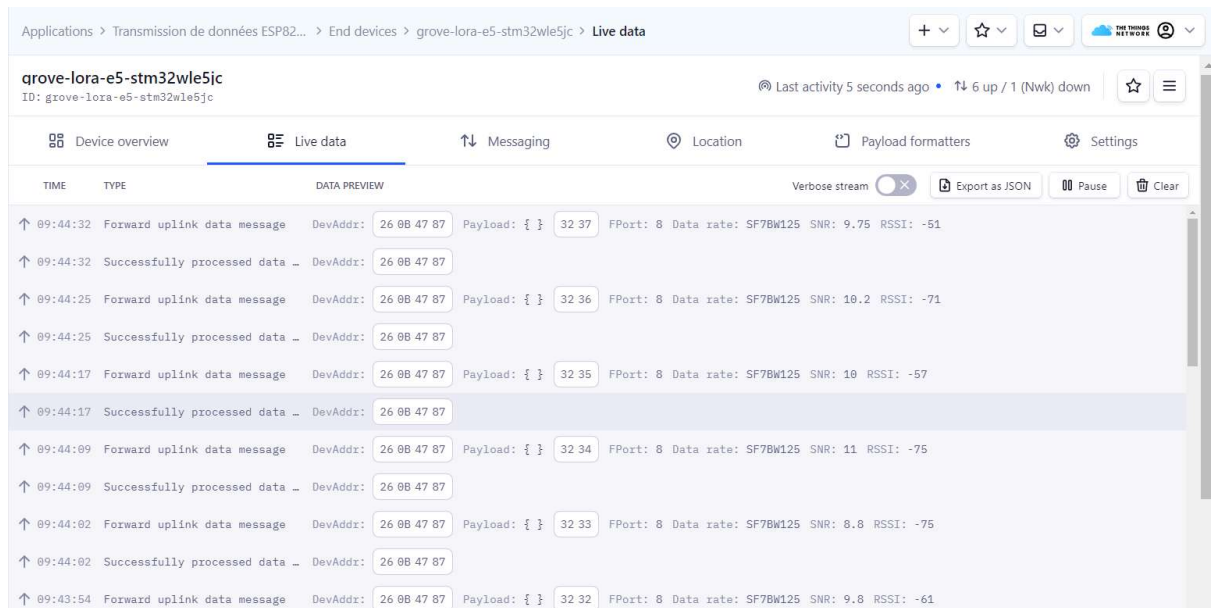
Maintenant que nous avons vérifié le bon fonctionnement de notre assemblage, nous allons enfin procéder à un envoi automatisé de messages directement depuis l'ESP.

Figure 1.2.4 : Diagramme de séquence de la montée de données au réseau LoRaWAN TTN.



Sur ce schéma, nous pouvons observer jusqu'à la passerelle l'utilisation de la technologie LoRa, qui correspond à la modulation du signal radio longue portée transportant les données. Ensuite, incluant le réseau TTN, le protocole de communication LoRaWAN, prend le relais pour contrôler et régir la manière dont ces données sont envoyées et reçues sur le réseau, garantissant ainsi une transmission efficace et sécurisée.

Figure 1.2.5 : Visualisation de la réception de données sur le réseau TTN.



Dans notre test, nous avons simulé l'envoi de messages successifs représentant une série de relevés de température. Chaque envoi incrémente la valeur transmise, ce qui nous permet d'observer la réception des données sous forme de codes ASCII, correspondant à des températures comprises entre 21°C et 29°C.

Nous avons également mis à jour l'affichage des logs de communication série entre l'ESP et le PC pour afficher clairement les réponses aux commandes envoyées au modem LoRa-E5.

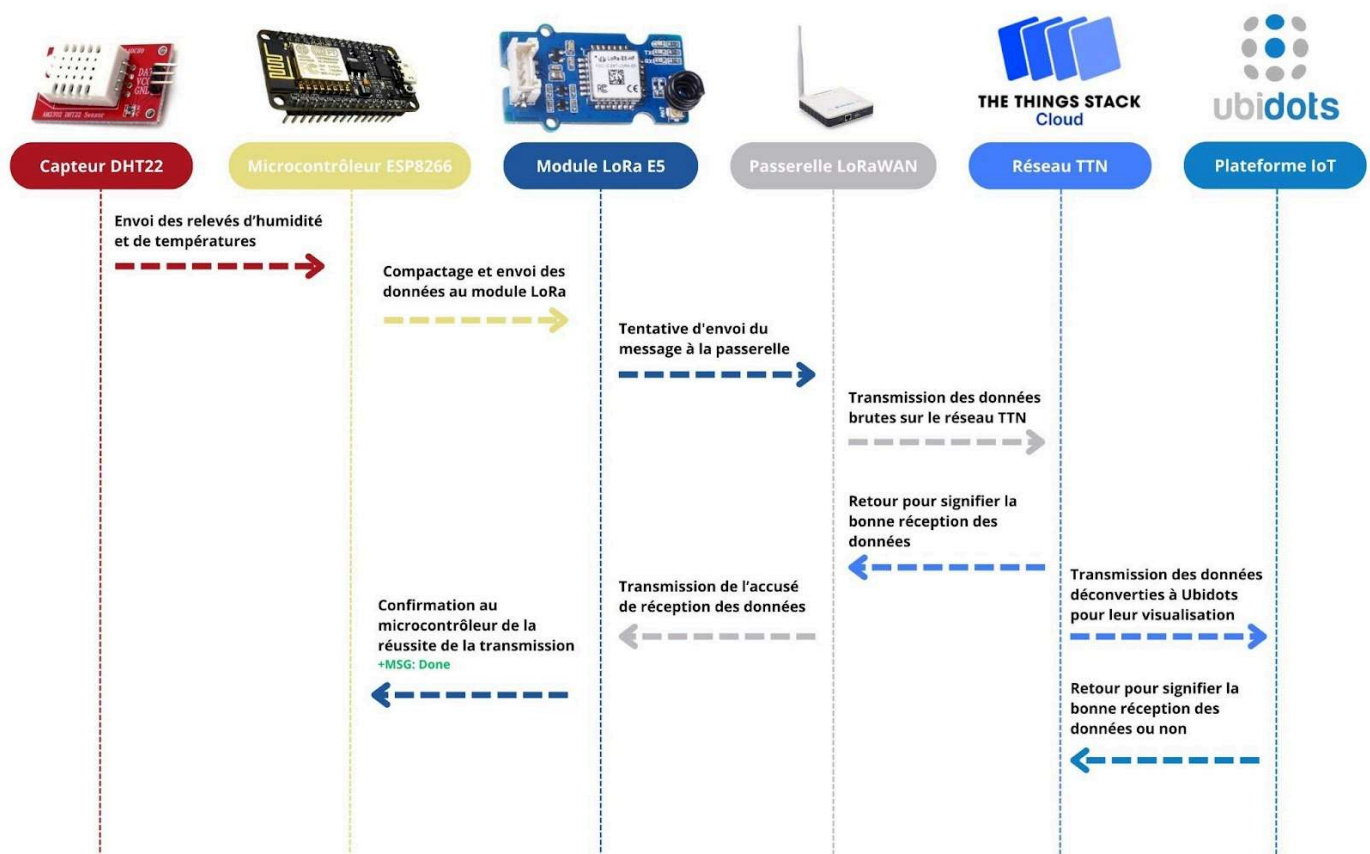
Cette première phase de tests a permis de vérifier le bon fonctionnement du capteur DHT22 ainsi que la configuration correcte du module LoRa-E5 pour la transmission de données vers le réseau TTN. Nous avons établi une communication fluide entre l'ESP et le LoRa-E5 via nos commandes AT, tout en assurant pour la prochaine étape de transmission des relevés de température une bonne réception sur le réseau de tel donnée. Les logs obtenus nous confirment que l'intégration entre un futur capteur, le microcontrôleur, et le réseau LoRaWAN fonctionne correctement. Nous sommes maintenant opérationnel à automatiser les données de températures et d'humidité envoyées pour exploiter pleinement les capacités du réseau LoRaWAN et leurs visualisation sur la plateforme Ubidots STEM.



## 2. Envoi des relevés du capteur DHT22 sur Ubidots STEM

Après avoir configuré, le matériel TTN à l'[A.1. Configuration du matériel TTN par étapes](#) et effectué la liaison avec Ubidots à l'[A.2. Liaison du matériel TTN avec Ubidots](#). Nous allons maintenant procéder à l'envoi de données depuis un capteur sur Ubidots à travers le réseau TTN.

Figure 2.1 : Diagramme de séquence de la transmission de relevés du capteur à Ubidots.



Le schéma ci-dessous illustre le processus complet de transmission des relevés du capteur DHT22, de la collecte initiale à la visualisation sur la plateforme Ubidots STEM. Le capteur DHT22, connecté à un microcontrôleur ESP8266, capture des données de température et d'humidité. Ces données sont ensuite compactées et transmises via le module LoRa E5 vers une passerelle LoRaWAN, qui relaye les informations au réseau TTN.

Une fois sur le réseau TTN, les données brutes sont décompactées et structurées avant d'être envoyées à la plateforme Ubidots pour une visualisation plus synthétique dans le temps. Le retour de chaque étape de transmission permet de vérifier la bonne réception des données et d'assurer une continuité fiable dans le flux d'information. Bien que quelques relevés puissent être manqués, cela n'a pas d'incidence notable sur l'exploitation des données, car les prélèvements restent proches dans le temps par rapport à la variation des facteurs environnementaux relativement lent.

*Figure 2.2 : Mise en forme standardisée post-décompression des relevés reçus.*



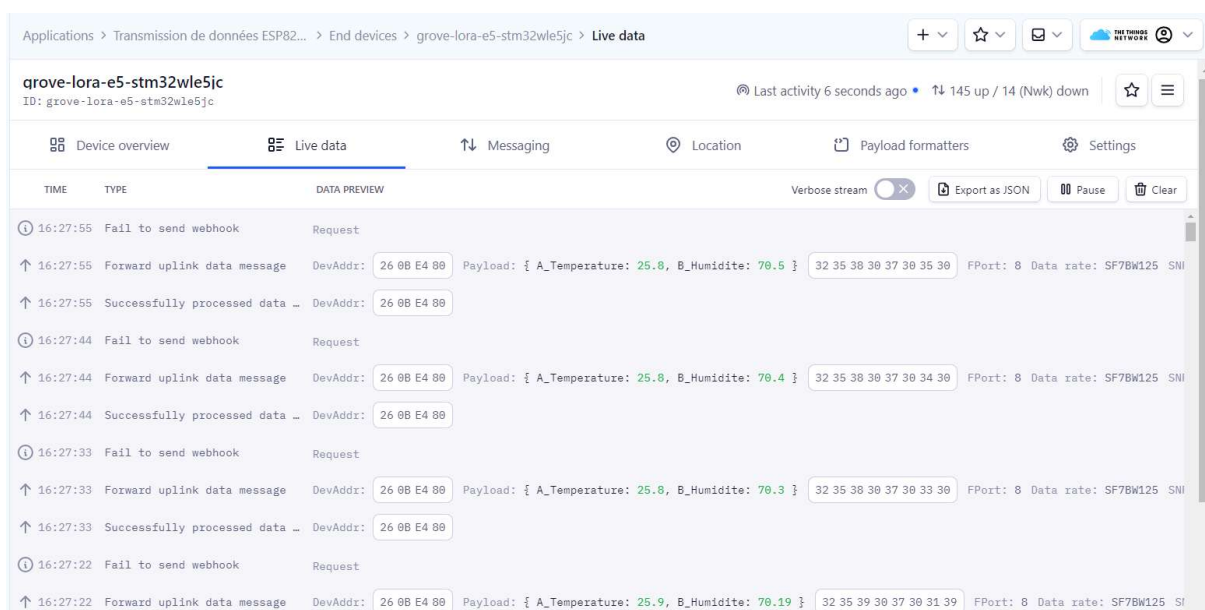
Les données reçues par le réseau TTN arrivent sous un format compressé afin de minimiser la taille des messages envoyés via le réseau LoRa. Ce processus de compression consiste à formater, au niveau du microcontrôleur, les relevés bruts de température et d'humidité capturés par le capteur DHT22.

Dans notre cas, la compression se fait en combinant une température de 25.70 °C et une humidité de 71.30 % en une chaîne de caractères "25707130", envoyée au format ASCII pour optimiser l'envoi. Bien que ce format soit compact, il est important de noter que les relevés sont transmis sous forme de chaîne de caractères plutôt qu'en hexadécimal, ce qui présente plusieurs avantages. En effet, le format ASCII, même s'il utilise 8 bits par caractère contre seulement 4 pour l'hexadécimal, est plus facile à manipuler et à lire pour les humains. Cela facilite le débogage et le suivi des données.

De plus, si l'on souhaite, à l'avenir, envoyer des chaînes de caractères représentant des états ou des commandes, le format texte permet de le faire de manière plus fluide. Par exemple, dans le cas d'un capteur de niveau d'eau qui doit envoyer des informations telles que "DEBORDEMENT", "NORMALE" ou encore "ANOMALIE", il serait plus logique d'envoyer ces messages sous forme de chaîne de caractères plutôt qu'en hexadécimal, ce qui compliquerait leur interprétation.

Une fois les données reçues sur la plateforme Ubidots, un petit script de formatage en JavaScript les décompose et les restructure dans un format JSON lisible : { A\_Temperature : 25.70, B\_Humidite : 71.30 }. Il convient de noter que les caractères "A\_" et "B\_" ont été ajoutés pour forcer l'ordre d'affichage, car, dans un même payload, l'ordre des données n'a pas d'importance et celles-ci sont triées de manière aléatoire, ici par ordre alphabétique. Ce processus garantit un transfert efficace des données tout en permettant une interprétation claire et précise sur la plateforme IoT pour l'analyse et la visualisation.

*Figure 2.3 : Visualisation de la réception des relevés compactés et de leurs mise en forme sur le réseau TTN.*

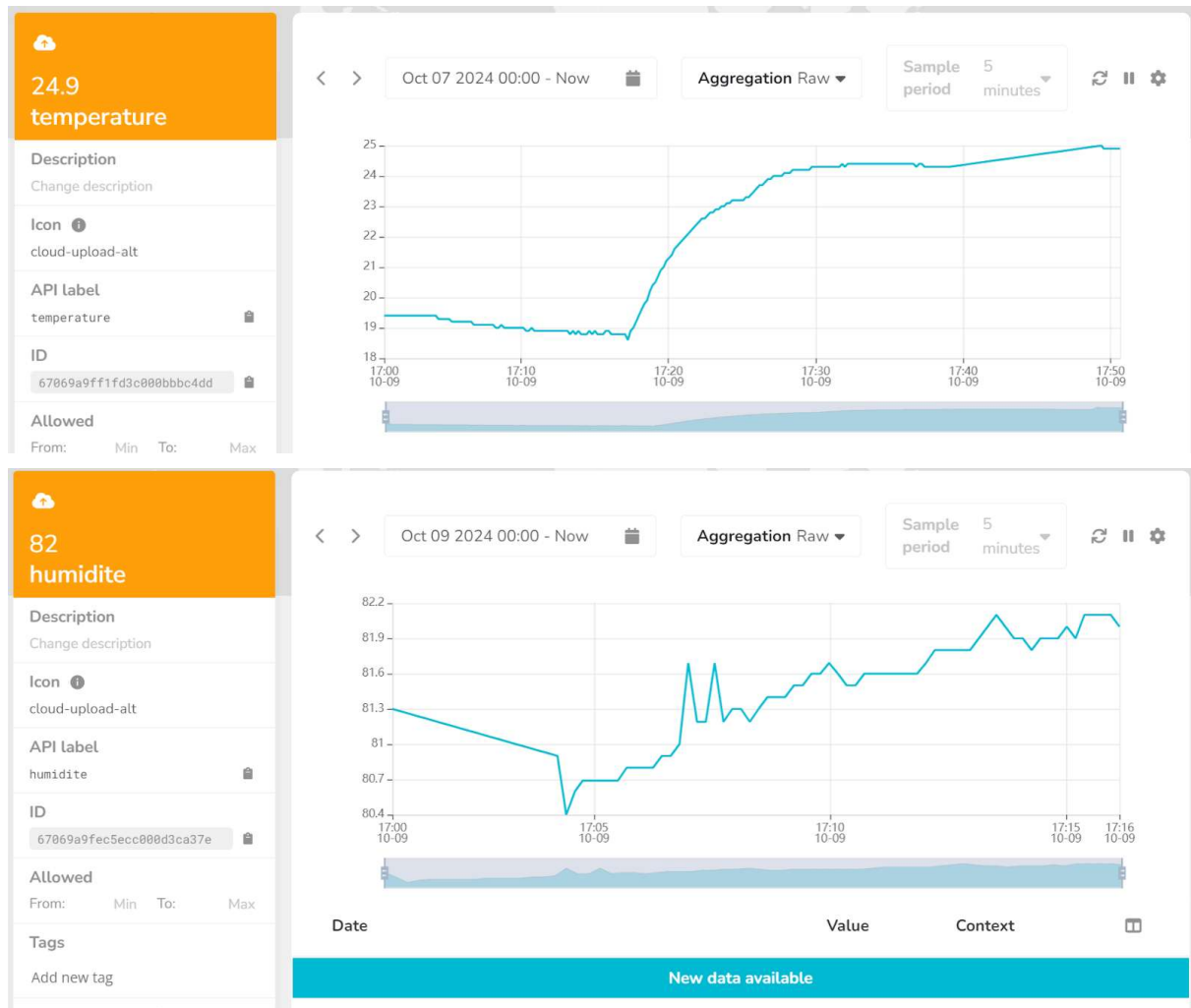


Enfin, les relevés formatés sont transmis à la plateforme Ubidots, permettant ainsi la lecture et la visualisation à distance des variations de température et d'humidité. Ces graphiques offrent une vue continue de l'évolution des paramètres environnementaux, facilitant les interventions si nécessaire. Par exemple, cela est particulièrement utile pour contrôler les conditions climatiques dans une serre ou un laboratoire.

Dans le graphique de température ci-après, nous avons simulé une forte variation de température due au passage de l'extérieur à l'intérieur du bâtiment, ce qui pourrait signaler un incident facilement repérable par une simple visualisation des relevés. Imaginons que nous souhaitons maintenir une température confortable de 19 °C, mais qu'une paroi du système se soit fissurée, laissant entrer de l'air très chaud. Cela pourrait nuire aux performances du système, tant pour le personnel, qui serait moins efficace sous une forte chaleur, que pour des plantations, qui préfèrent une température plus douce de 19 °C à 25 °C.

Dans ce cas, nous aurions pu détecter rapidement et efficacement ce brusque changement de température et faire intervenir un technicien sur place, et ce, même en ayant repéré l'anomalie à l'autre bout du territoire.

*Figure 2.4 : Graphiques des relevés de température et d'humidité réceptionnés sur Ubidots.*



Dans cette section, nous avons montré comment les relevés de température et d'humidité, capturés par le capteur DHT22, sont efficacement transmis à la plateforme IoT Ubidots STEM via le réseau LoRaWAN et TTN. Ce processus, qui va de la collecte initiale des données à leur visualisation, assure non seulement une transmission fiable, mais aussi un accès à distance et en temps réel aux données. Cela permettra, dans une exploitation future, une analyse approfondie.



Les performances du réseau LoRa, combinées à Ubidots, offrent une solution efficace pour la surveillance environnementale à distance. Ces données pourront être utilisées pour optimiser des systèmes automatisés, prévoir des tendances ou générer des alertes en cas d'événements critiques. De plus, l'intégration future de capteurs supplémentaires et de solutions automatisées pourrait encore renforcer cette surveillance.

## Conclusion

La technologie LoRaWAN est réputée pour son efficacité dans la transmission de données sur de longues distances tout en consommant très peu d'énergie. C'est une solution idéale pour des scénarios où l'autonomie des appareils est cruciale, notamment pour des capteurs dans des environnements isolés ou peu accessibles. Cependant, lors de nos tests en milieu urbain très dense, les résultats ont été loin des attentes théoriques.

Les spécifications techniques et les ressources en ligne promettaient des distances de transmission allant jusqu'à 10 ou 15 kilomètres. De plus, certains exploits ont renforcé nos attentes, comme le record en mer de 1336 kilomètres mentionné par Monsieur DOUZE, ou encore le précédent record du monde de 832 kilomètres atteint avec une puissance de 25 mW par un ballon en altitude, relayé par l'incroyablissime Monsieur ViATEUR.

Ces performances exceptionnelles laissaient présager des capacités impressionnantes de la technologie. Cependant, la réalité sur le terrain s'est révélée différente : depuis les antennes dispersées dans la capitale, les tests n'ont permis d'échanger avec des passerelles TTN qu'à peine jusqu'à un demi-kilomètre. Dans notre bâtiment, les résultats étaient encore plus décevants, avec une portée de seulement quelques dizaines de mètres.

Cette performance décevante s'explique en grande partie par des contraintes techniques. En effet, les bornes n'ont pas pu être installées en extérieur, ce qui aurait certainement permis de meilleures conditions de réception. Dans un environnement aussi dense, les signaux LoRaWAN semblent avoir du mal à pénétrer les obstacles tels que les murs épais et les bâtiments.

Malgré cette déception concernant les distances atteintes en milieu urbain, il est important de souligner que la technologie LoRa reste extrêmement prometteuse, notamment dans des environnements ouverts comme les zones rurales ou les champs, où les obstacles sont beaucoup moins nombreux. En outre, LoRa est avant tout une technologie "Low Power", ce qui signifie qu'elle peut contribuer à un futur plus durable en offrant une solution écoénergétique pour l'Internet des objets.

Ainsi, bien que les résultats dans un contexte urbain n'aient pas été à la hauteur des espérances, LoRa conserve son potentiel dans de nombreux autres domaines d'application, en particulier là où la consommation d'énergie est un facteur clé.

## Annexe

### A. Mise en place de l'environnement de travail

#### A.1. Configuration du matériel TTN par étapes

Pour garantir une communication correcte entre nos relevés de capteurs et notre serveur, il est essentiel de configurer le réseau TTN (The Things Network).

1) Création d'une application depuis le réseau communautaire The Things Network :

<https://eu1.cloud.thethings.network/console/applications>

2) Ajout des modules utilisés : **End devices -> Top end devices -> Register end device**

<https://eu1.cloud.thethings.network/console/applications/echange-lorae5-esp8266/devices/add>

3) Selection du **type du module** tel que :

End device type

Input method ⓘ

- ☒ Select the end device in the LoRaWAN Device Repository  
☐ Enter end device specifics manually

End device brand ⓘ *	Model ⓘ *	Hardware Ver. ⓘ *	Firmware Ver. ⓘ *	Profile (Region) *
<input type="text" value="Seeed Technolog..."/>	<input type="text" value="LoRaWAN Dev Kit"/>	<input type="text" value="1.0"/>	<input type="text" value="1.0"/>	<input type="text" value="EU_863_870"/>

4) Choix de la **bande de fréquence** utilisée ici standard européen :

Frequency plan ⓘ \*

5) Generation d'identifiants réseau aléatoire pour pouvoir se connecter à l'application :

### Provisioning information

JoinEUI ? \*

A0 A0 A0 A0 A0 A0 A0 A0

Reset

This end device can be registered on the network

DevEUI ? \*

70 B3 D5 7E D0 06 59

Generate

1/50 used

AppKey ? \*

B3 C2 F5 B9 D8 7A 0B 02 B2 D8 74

Generate

End device ID ? \*

grove-lora-e5-stm32wle5jc

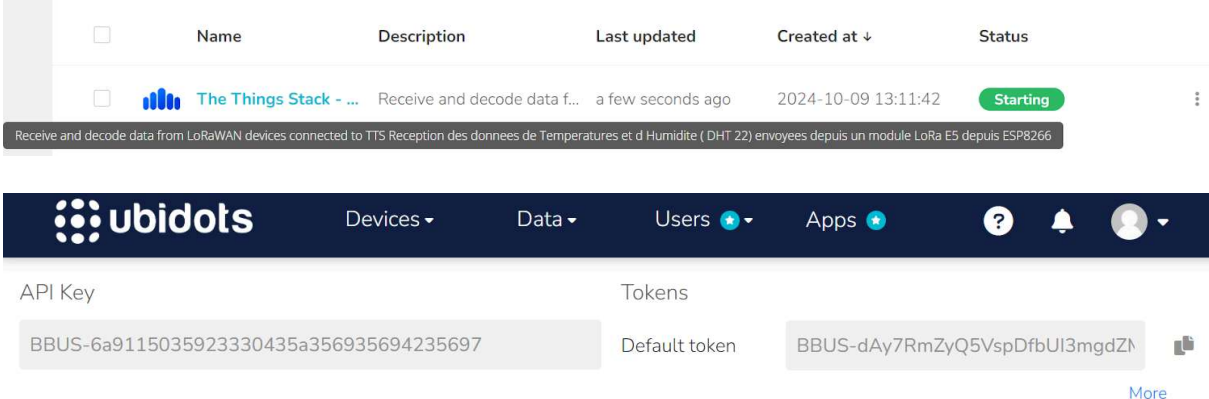


## A.2. Liaison du matériel TTN avec Ubidots

Ubidots est une plateforme IoT qui permet de stocker, visualiser et exploiter les données issues de dispositifs connectés. Bien qu'il soit payant après un certain seuil d'utilisation, Ubidots présente des avantages non négligeables pour la gestion à distance des données capturées par les capteurs. En effet, Ubidots permet aux utilisateurs de consulter les relevés des capteurs sans nécessiter une connexion locale directe avec les dispositifs, contrairement à une approche traditionnelle où il serait nécessaire d'être dans la même zone de couverture réseau.

Pour la configuration nous avons en partie suivi les instructions au lien suivant : <https://help.ubidots.com/en/articles/5096476-plugins-connect-the-things-stack-to-ubidots>

1) Création d'une nouvelle application pour recevoir et gérer les données des capteurs sur [Ubidots](https://ubidots.com).



The screenshot shows the Ubidots web interface. At the top, there's a table of applications. One application is listed: 'The Things Stack - ...' with a description 'Receive and decode data f...', last updated 'a few seconds ago', created at '2024-10-09 13:11:42', and status 'Starting'. Below this, a detailed description of the application is visible: 'Receive and decode data from LoRaWAN devices connected to TTS Reception des donnees de Temperatures et d Humidite ( DHT 22) envoyees depuis un module LoRa E5 depuis ESP8266'. Below the application list, the 'API Key' and 'Tokens' section is shown. The API Key is 'BBUS-6a9115035923330435a356935694235697'. The Default token is 'BBUS-dAy7RmZyQ5VspDfbUI3mgdZM'. There is a 'More' link next to the token.


Nous avons ensuite ajouté un appareil (ici, un module LoRa-E5 connecté à un capteur DHT22), en récupérant son identifiant DevEUI et en configurant les paramètres OTAA pour établir la connexion au réseau LoRaWAN. Cela nous permet de recevoir des données de température et d'humidité envoyées par l'ESP8266 via LoRa.

2) Création d'un webhook Ubidots sur [TTN](#) :

<https://eu1.cloud.thethings.network/console/applications/echange-lorae5-esp8266/integrations/webhooks/dht22-to-esp-to-lorae5-to-ttn-ubidot>

Applications > Transmission de données ESP82... > Webhooks > Add > Ubidots

**Transmission de données ESP8266 to Grove Lora E5**  
ID: echange-lorae5-esp8266

 **Setup webhook for Ubidots**  
Integrate with Ubidots using Plugins  
[About Ubidots](#) |

**Webhook ID\***

**Plugin ID\***  
  
Unique identifier found in your Plugin's HTTPS URL

**Ubidots token\***  
  
Token used to authenticate Ubidots API requests

[Create Ubidots webhook](#)

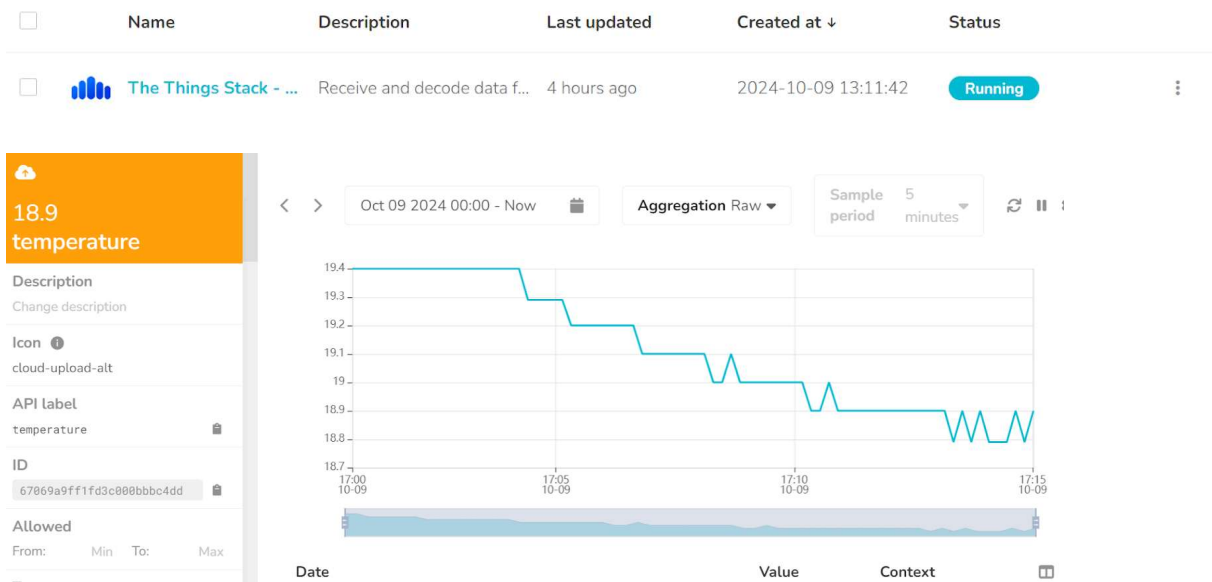
Le webhook permet à chaque message uplink reçu par TTN d'être redirigé automatiquement vers Ubidots, où il peut être visualisé et traité.

### 3) Visualisation des données sur [Ubidots App Devices](#) :

```

1 function decodeUplink(input)
2 {
3   //Fonction de conversion generee par ChatGPT : Ascii to Str
4   let asciiString = String.fromCharCode.apply(null, input.bytes);
5
6   let Caractere_1_to_2 = asciiString.substring(0, 2);
7   let Caractere_3_to_4 = asciiString.substring(2, 4);
8   let temperature = parseFloat(Caractere_1_to_2 + "." + Caractere_3_to_4);
9
10  let Caractere_5_to_6 = asciiString.substring(4, 6);
11  let Caractere_7_to_8 = asciiString.substring(6, 8);
12  let humidite = parseFloat(Caractere_5_to_6 + "." + Caractere_7_to_8);
13
14  return {
15    data: {
16      A_Temperature : temperature,
17      B_Humidite : humidite
18    },
19    warnings: [],
20    errors: []
21  }
22 }
  
```

Pour visualiser les données reçues depuis le réseau LoRa avec un format compacté, nous allons effectuer une conversion des données brutes. Nous allons effectuer depuis Payload formatters une déconversion rendant les données exploitables et facilement interprétables pour l'utilisateur final.



Sur la plateforme Ubidots, les données transmises depuis TTN apparaissent sous la forme de séries temporelles, permettant ainsi de suivre l'évolution des relevés de température et d'humidité en temps réel.

### A.3. Configuration de Arduino IDE pour échanger avec l'ESP

1) Installation de Arduino IDE depuis :

<https://downloads.arduino.cc/arduino-ide/arduino-ide-windows.exe>

2) Dans Arduino IDE, indiquer le module à utiliser depuis **File->Preferences...->** et indiquer dans « **Additional boards manager URLs :** » l'url depuis recherche internet « esp8266 url for arduino ide » :

[https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json) / [https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json),  
[http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)

sur le site

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/> ou celui de l'ESP8266

3) Installation de la carte ESP depuis **Tools->Board->Boards Manager...** :  
« **Arduino Nano ESP32** » et dans notre cas **ESP8266**

4) Après avoir relié le capteur avec le microcontrôleur, **Tools->Board :** **Node MCU 1.0 (ESP-12E Module) -> esp8266 -> Node MCU 1.0 (ESP-12E Module)**

5) Installation librairie DTH :

a )

<https://www.gotronic.fr/art-capteur-d-humidite-et-de-t-grove-101020011-18963.htm>

b ) fiche technique :

[http://wiki.seeedstudio.com/Grove-TemperatureAndHumidity\\_Sensor/](http://wiki.seeedstudio.com/Grove-TemperatureAndHumidity_Sensor/)

c ) « **Software**

- **Step 1.** Download the [Seeed DHT library](#) » -> code -> download zip

d ) Inclusion de la librairie téléchargée : **Sketch -> Include Library -> Add .ZIP Library... -> Grove\_Temperature\_And\_Humidity\_Sensor-master.zip**

5 ) Installation du driver ( si le port n'est pas détecté ) depuis

<https://randomnerdtutorials.com/getting-started-with-esp8266-wifi-transceiver-review/> ->

<https://randomnerdtutorials.com/install-esp32-esp8266-usb-drivers-cp210x-windows/> ->

<https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads> ->

[https://www.silabs.com/documents/public/software/CP210x\\_Windows\\_Drivers.zip](https://www.silabs.com/documents/public/software/CP210x_Windows_Drivers.zip)

6) Connecter le Port : **Tools -> Port...-> COM3**