AgroSense_Leaf Moisture SN-3001

LoRaWAN® Manual

V1.0

Author: Yuki

Time: 2024.12.07

Contents

1 Product Description 1

1.1 Introduction 1

1.2 Feature 2

1.3 Parameter 3

2 Technical route 4

2.1 System Framework 4

2.2 Regional frequency band 4

3 Usage 6

3.1 TTN and ThingSpeak 6

3.1.1Network Server configuration 6

3.1.2 Decoder 9

3.1.3 Application Server configuration 11

3.1.4 Connect the Network Server and Application Server 12

3.1.5 Change Time Interval (5-1440min) 13

3.2 Datacake 15

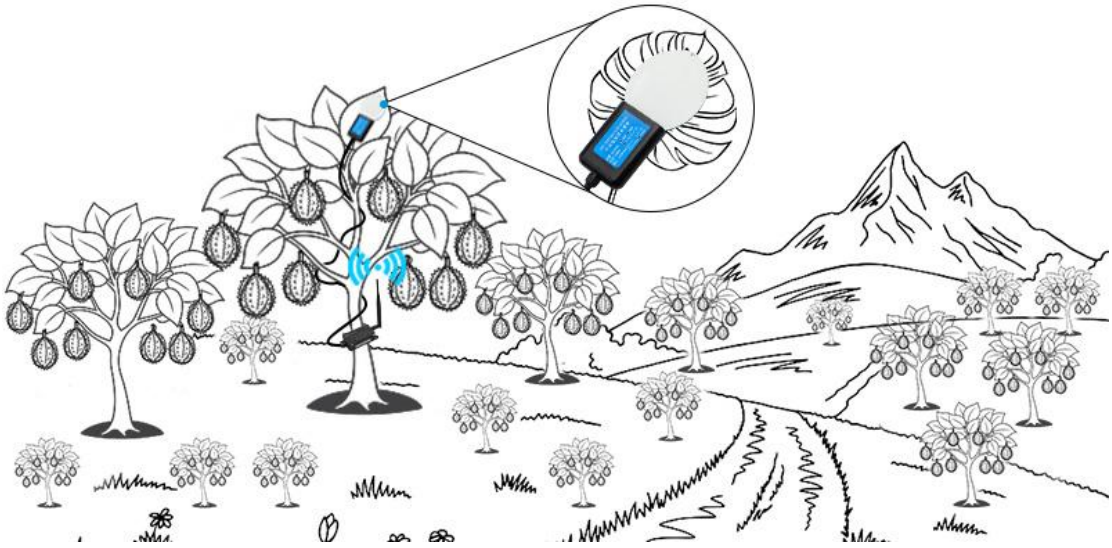
3.2.1 Change Time Interval (5-1440min) 22

1 Product Description

1.1 Introduction

This AgroSense LoRaWAN® Leaf Moisture Sensor uses the SN-3001 sensor, to measures the leaf moisture and temperature, so to send to the platform to analyze the leaf status such as : watering, moisturizing, dew, frozen, at the range of -40℃~60℃and 0%-100% RH, with accuracy ±0.5℃and±3%RH respectively, also with highly waterproof performance tested to IP68, making it widely applicable in agricultural environmental sensing scenarios to support the smart agricultural production.

The SN-3001 sensor adopts SHT4 high-precision chip, detects leaf moisture and temperature. It employs the FDR method, which enables it to sense the dielectric constant caused by liquid over the leaf surface and then cover the value with respect to leaf moisture. The probe has been designed in a leaf shape to best simulate the real leaf characteristics. Suitable for agricultural production scenarios where precise leaf temperature and humidity are required, such as durian, tomato and cucumber production, etc.



The sensor benefits from LoRaWAN, which ensures stability and reliability. It is capable of covering a long transmission range while maintaining low power consumption. Unlike wireline devices, it is battery-powered, reducing the workload and complexity of deployment, design and development for end-users that can work via powering it, and setting the configuration in the cloud server, for LoRaWAN® remote monitoring. It monitors the air temperature and humidity and report them every 1 hour.

1.2 Feature

- Includes a **high precision** sensor.
- Compatible with Worldwide **LoRaWAN® Networks**: Support the universal frequency bands EU868/ US915.
- LoRaWAN version: LoRaWAN Specification **1.0.3**.
- **Long Range**: Up to 2 kilometers in the city, up to 10 kilometers in the wilderness, receive sensitivity -137dBm , transmit power up to 21dBm.
- **Ultra-low power** consumption design, traditional AAA alkaline dry battery can be used for one year.
- **Data encryption**: Provide end-to-end secure communication, including device authentication and network data encryption, to ensure the security of data transmission and prevent data theft and malicious attacks.
- **High stability and reliability:** good stability in noisy environments, able to penetrate buildings and obstacles, so it can maintain good communication quality in urban and suburban environments.
- Suitable for **Harsh Environments**: Can work normally under the temperature of -40℃ ~ 85℃, IP68 waterproof, suitable for outdoor use in harsh conditions, high UV, dusty, heavy rain and other bad weather.
- Monitor data and upload **real-time** data regularly.
- Modify the product parameters through **AT commands**.
- Support **downlink** to modify the time interval, motion status on/off, motion status sensitivity.

1.3 Parameter

1. General Parameters

Product Model	AGLWLM01
Temperature Measurement Range	-40°C ~60°C
Temperature Measurement Accuracy	±0.5°C
Temperature Resolution	0.1°C
Humility Measurement Range	0%-100% RH
Humility Measurement Accuracy	±3%
Humility Resolution	0.1% RH

2.Wireless Parameters

Communication Protocol	Standard LoRaWAN® protocol V1.0.3
Network Access/Operating Mode	OTAA Class A
MAX Transmit Power	21dBm
Receiver Sensitivity	-137dBm/125kHz SF=12
Frequency Band	EU868/US915

3.Physical Parameters

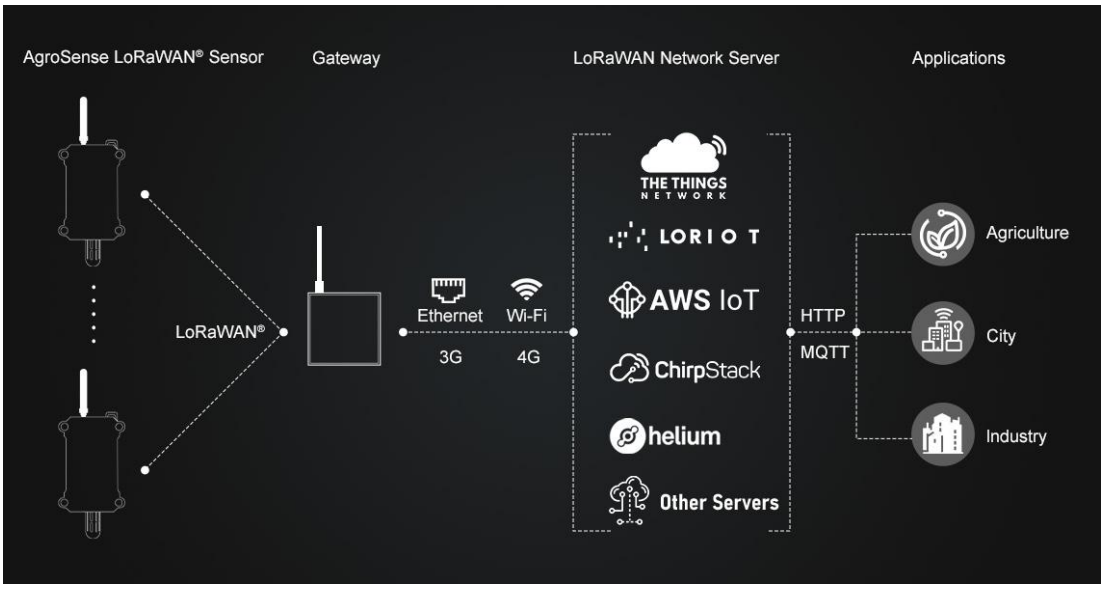
Lead Length	2 meter (custom length available)
Power Supply	1 x 18650 3.7V Lion batteries
Operating Temperature	-40°C ~60°C
Protection Class	IP67
Dimensions	131 × 62.7 × 27.5 mm
Mounting	Wall Mounting

2 Technical route

2.1 System Framework

AgroSense_Leaf Moisture Sensor uses LoRAWAN technology, and it network architecture includes four parts: End Nodes, Concentrator/Gateway, Network Server and Application Server.

End Nodes	It is responsible for collecting sensing data and then transmitting it to Gateway via the LoRaMAC protocol.
Concentrator/Gateway	It is mainly responsible for transmitting node data to the server.
Network Server	Organize the data into JSON packets and decode them.
Application Server	Display the data.



The steps to achieve the detection of leaf moisture is:

1. Collect the leaf moisture data by sensor, and send the data from End Node to Gateway.
2. The Gateway packages node data and transmits it to the Network Server.
3. The Network Server decodes the data and sends it to the Applications.
4. Finally, user can monitor the leaf moisture in the APP.

2.2 Regional frequency band

At the present moment, our product solely accommodates compatibility with the US915 and

EU868.

area	frequency band	center frequency
China	470-510MHz	CN486MHz
America	902-928MHz	US915MHz
Europe	863-870MHz	EU868MHz
Korea	920-923MHz	KR922MHz
Australia	915-928MHz	AU923MHz
New Zealand	921-928MHz	NZ922MHz
Asia	920-923MHz	AS923MHz

3 Usage

We use The Things Network as our Network Server, we need to configuration the country/ area frequency, inputting DEV EUI/ APP EUI/ APP Key, decodes, and connect to ThingSpeak or Datacake.

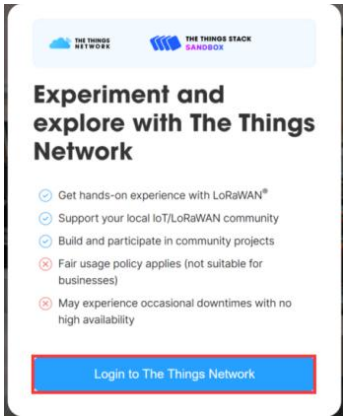
DEV EUI	Unique identification of device, authorized by IEEE
APP EUI	Unique identification of application
APP Key	One of the join network parameters on OTAA mode, calculated by DE EUI

- End Nodes and Gateway: AgroSense_Leaf Moisture Sensor LoRaWAN®. (The AgroSense series is applicable)
- Network Server: The Things Network. (Loriot, AWS IoT, ChirpStack, ect)
- Application Server: ThingSpeak.(Datacake, Blockbax, akenza, ect)

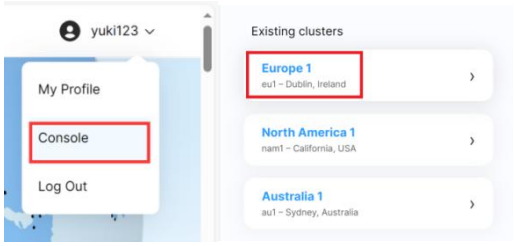
3.1 TTN and ThingSpeak

3.1.1Network Server configuration

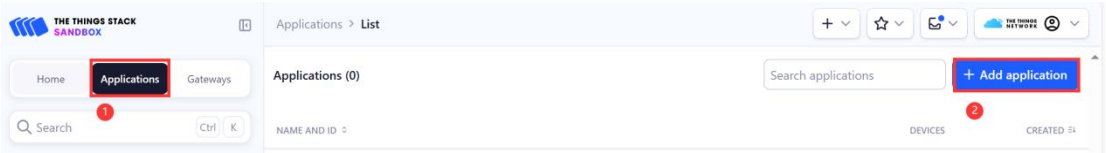
- Open The Things Network in your browser and login it. (Or register an account)



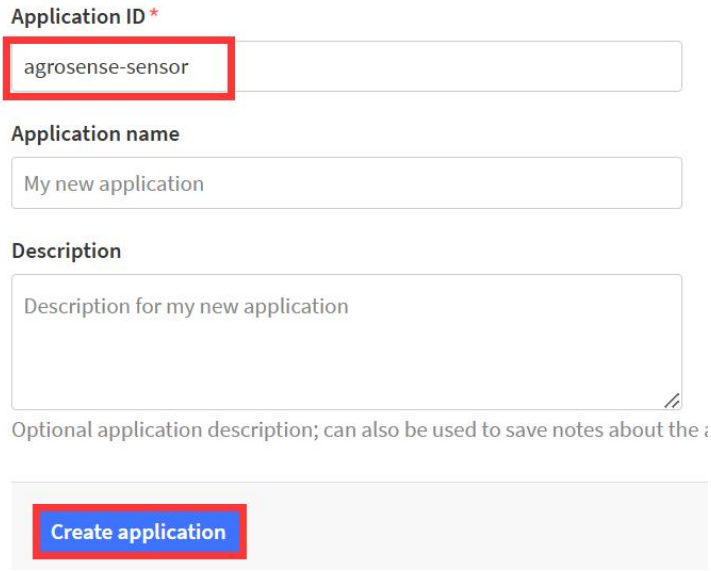
- Click “Console” and select clusters. (we take the European region for example.)



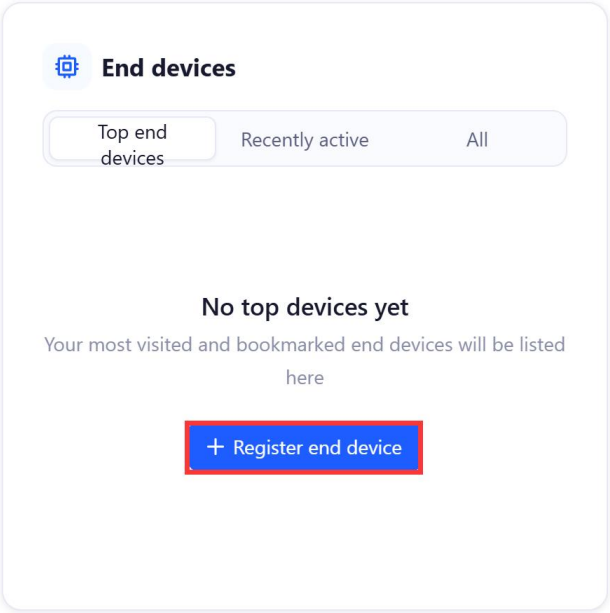
- Click “Go to applications” --> “+ Create application”.



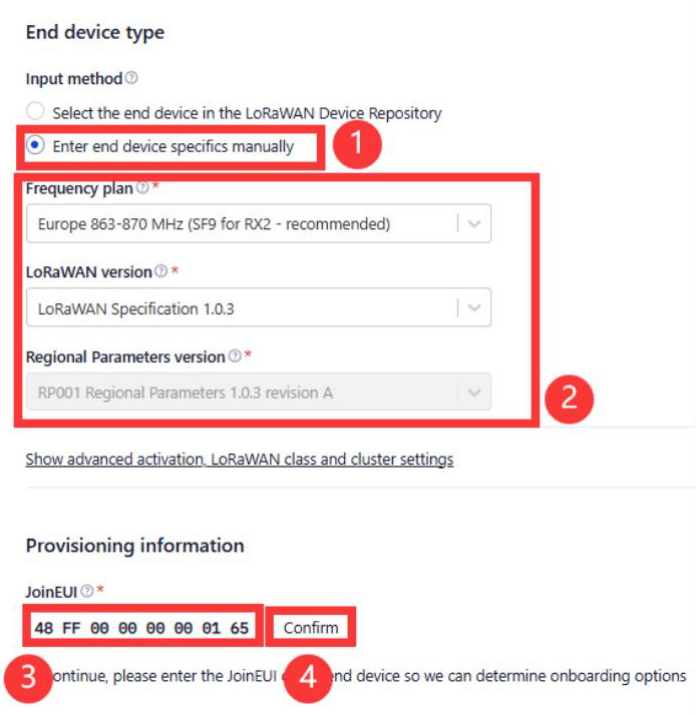
- Write the Application ID and click “Create application”.

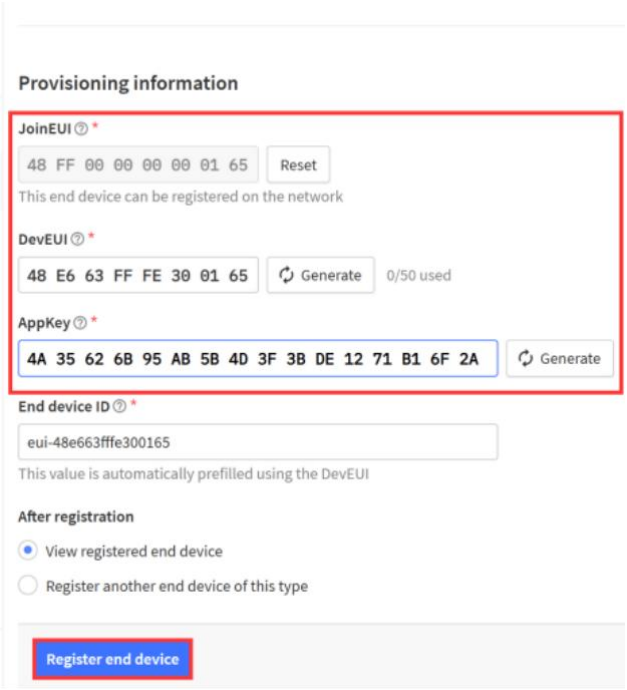


- Click “+ Register and device”.

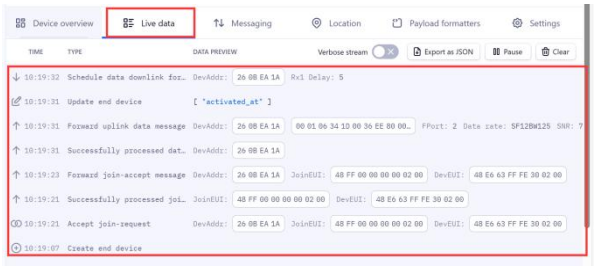


- Fllowing the steps, and input the DEV EUI/ APP EUI/ APP Key (notice: JoinEUI=APP EUI) and subsequently click on "Register end device" to complete the registration process.



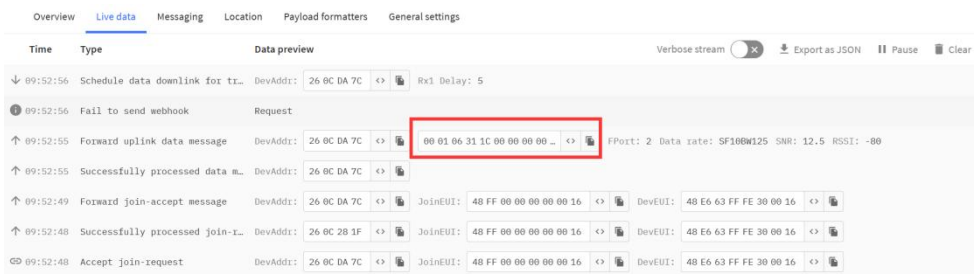


- Plug the battery and press RES button, you can see the device is connected successfully in the TTN.



3.1.2 Decoder

- Now, we need to decoder the data.



Data length	Data description	Value range	Explanation
byte 0	Data packet sequence number high 8 bits	0-0xFFFF	Counting starts from 0 and increments, resetting back to 0 after reaching 65535
byte 1	Data packet sequence number low 8 bits		
byte 2	Battery voltage		The value is obtained by amplifying the data by 10 times, and the actual value needs to be divided by 10 to convert to the actual battery voltage. The purpose of multiplying by 10 is to retain one decimal place of the voltage value. For example, if the value is 0x21 = 33, then the battery voltage is 3.3V.
byte 3	The data validity flag		0 is invalid, 1 is valid.
byte 4	Humility sensor bits 8 to 15		This value is obtained after magnifying the data by 10 times. To obtain the actual relative humidity value, the real value needs to be calculated by dividing

byte 5	Humility sensor bits 0 to 7		it by 10. For example, if the value from the 8th to the 15th bit is 0x02, and the lower 8 bits value is 0x85, then the relative humidity value obtained is 0x00000285= 645. After converting and dividing by 10, the actual relative humidity is 6.45%RH.
byte 6	Temperature sensor bits 8 to 15		This value is obtained after magnifying the data by 10 times. To obtain the actual Temperature value, the real value needs to be calculated by dividing it by 10. For example, if the value from the 8th to the 15th bit is 0x02, and the lower 8 bits value is 0x85, then the relative humidity value obtained is 0x00000285= 645. After converting and dividing by 10, the actual Temperature is 6.45℃.
byte 7	Temperature sensor bits 0 to 7		
byte 8	data transmission interval bits 24 to 31		The time interval for data transmission has been increased by a factor of 100.
byte 9	data transmission interval bits 16 to 23		
byte 10	data transmission interval bits 8 to 15		
byte 11	data transmission interval bits 0 to 7		

Example: 0x00, 0x01, 0x28, 0x01, 0x00, 0x33, 0x01, 0x3F, 0x00, 0x36, 0xEE, 0x80

Data parsing：

Battery voltage is 4V.

Humility value is 5.1%.

Temperature value is 31.9℃.

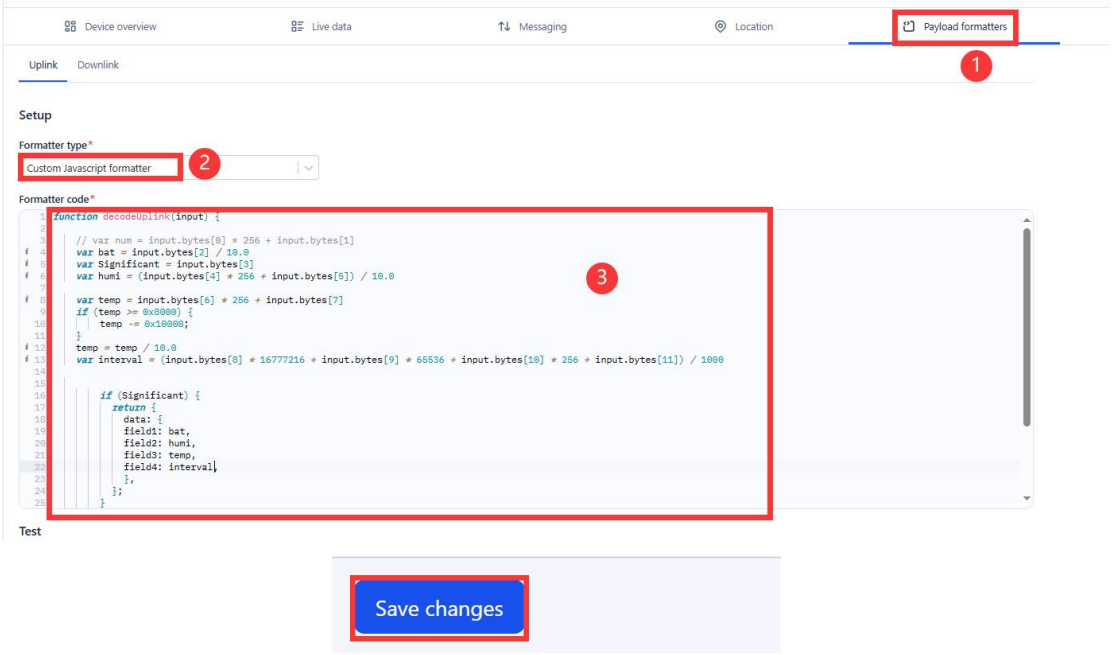
data transmission interval value is 3600.

- Know how to decode it after, we need to write it in code. (You can check it out on Github)

```
function decodeUplink(input) {

    // var num = input.bytes[0] * 256 + input.bytes[1]
    var bat = input.bytes[2] / 10.0
    var Significant = input.bytes[3]
    var humi = (input.bytes[4] * 256 + input.bytes[5]) / 10.0

    var temp = input.bytes[6] * 256 + input.bytes[7]
    if (temp >= 0x8000) {
        temp -= 0x10000;
    }
    temp = temp / 10.0
```

```
    var interval = (input.bytes[8] * 16777216 + input.bytes[9] * 65536 + input.bytes[10] * 256 + input.bytes[11])
/ 1000

        if (Significant) {
          return {
            data: {
            field1: bat,
            field2: humi,
            field3: temp,
            field4: interval,
            },
          };
        }
        else {
          return {
            data: {
            Significant: "data invalid",
            },
          };
        }
}
```

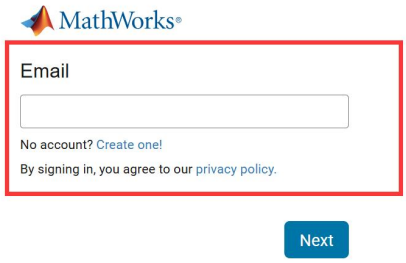
- Select “Payload formatters” and follow the steps.



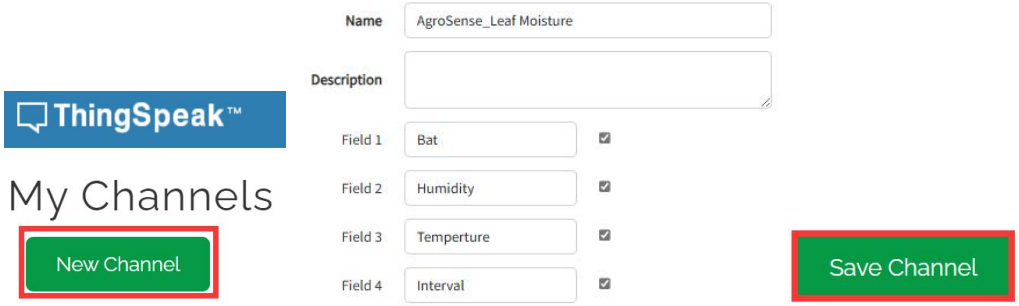
3.1.3 Application Server configuration

In the Application Server configuration, we need to create ThingSpeak channel and get Channel ID and API Key, this is the key to our connection to TTN.

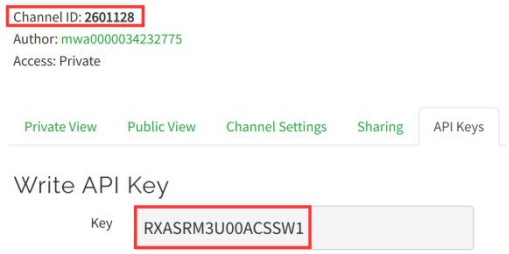
- Login to the ThingSpeak. (Or register an account)



- Click “New Channel”, fill in the Channel name and field names and click “Save Channel”.

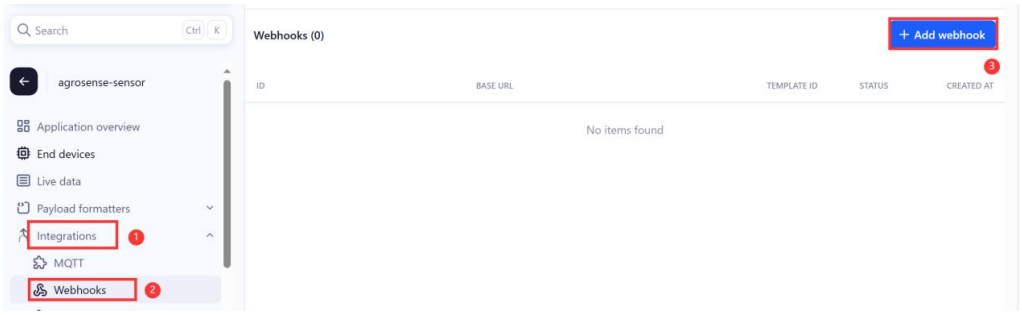


- After successful creation, copy the Channel ID and API Key.

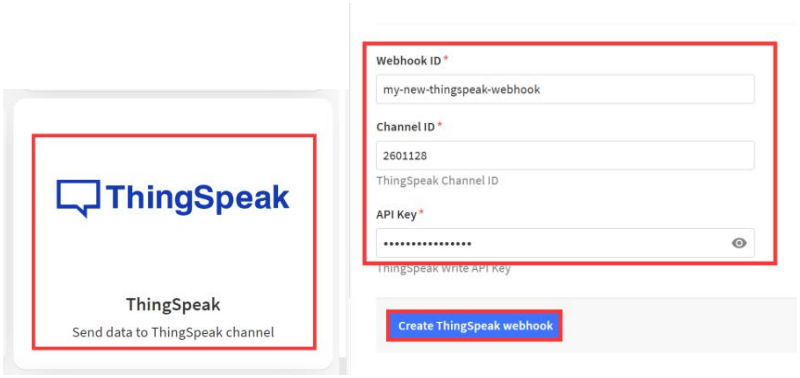


3.1.4 Connect the Network Server and Application Server

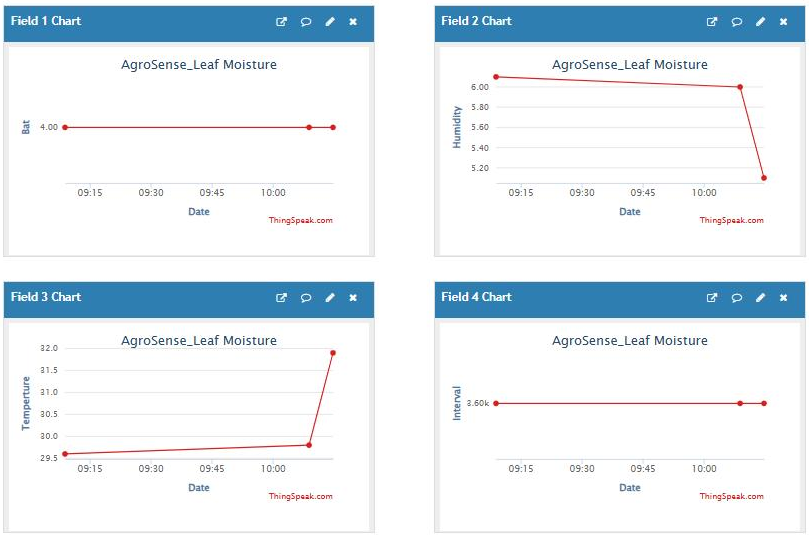
- In the TTN, click “integrations” --> “Webhooks” --> “+ Add webhook”.



- Select “ThingSpeak”, Fill in the Webhook ID and paste the Channel ID and API Key, click “Create ThingSpeak Webhook”.



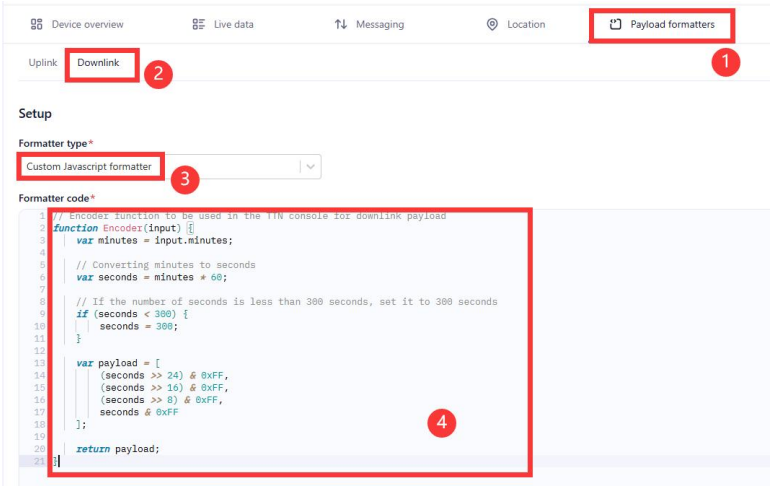
- Press RES button, wait about a minute, you will successfully see the data in ThingSpeak.(You will receive the data every hour.)



3.1.5 Change Time Interval (5-1440min)

1 、 If you need to change time Interval (Default 60 minutes), you can click “Payload formatters-->Downlink” and follow the steps.

Formatter code you can find in Github.



2、Click “Save changes”.

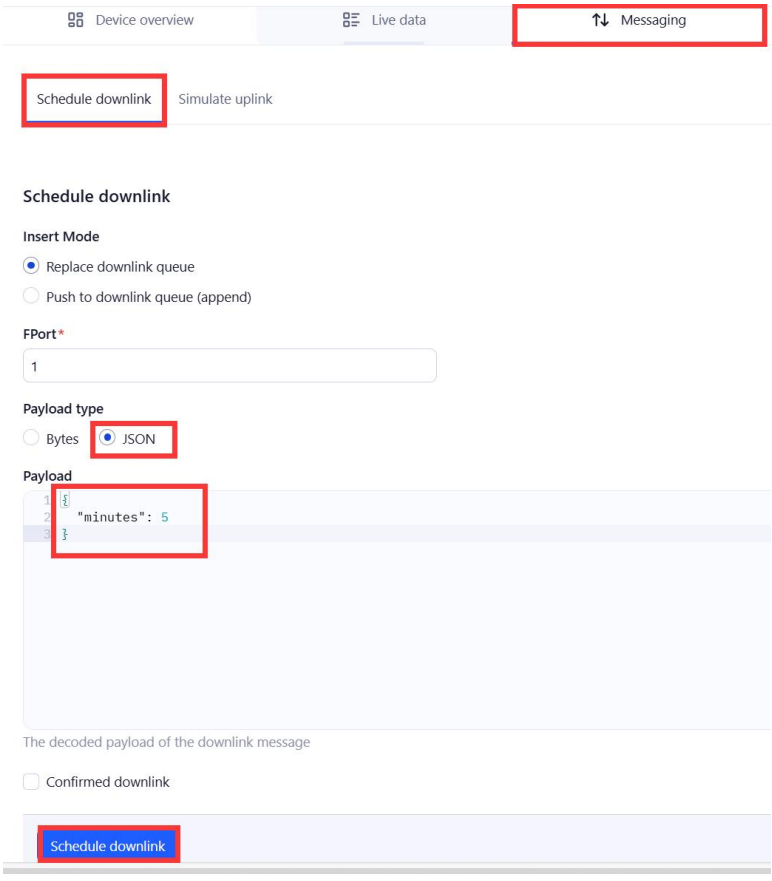
3、Click “Messaging-->Schedule downlink”.

Note: you must use this format:

{

"minutes": 5

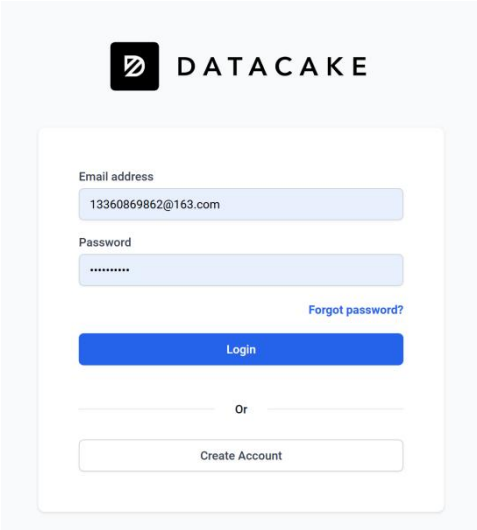
}



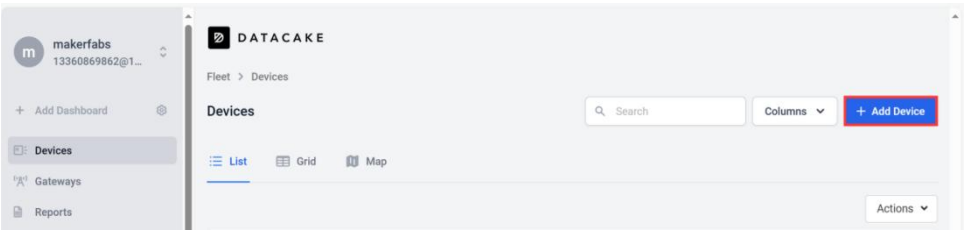
4、The modified interval will be updated after the next data upload.

3.2 Datacake

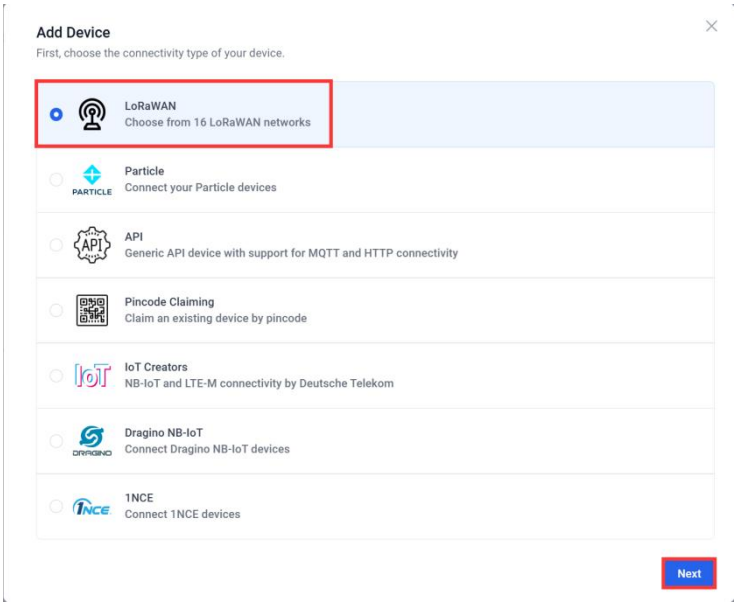
1、Login datacake or Create Account



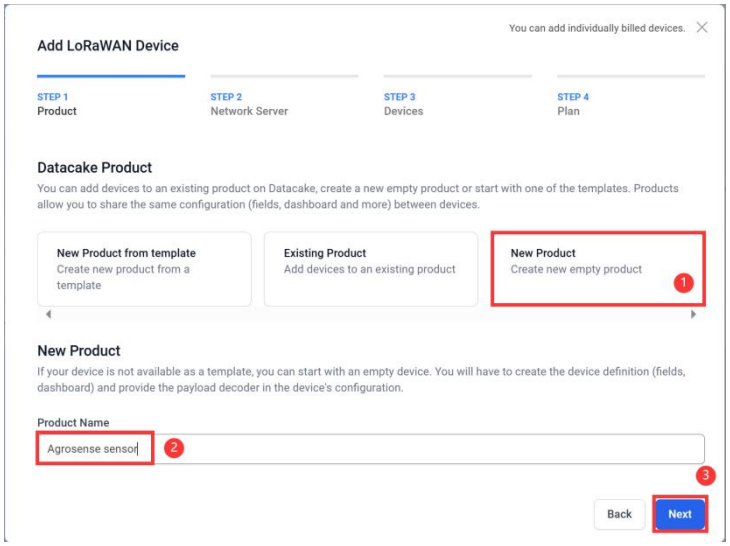
2、Click “Add Device”



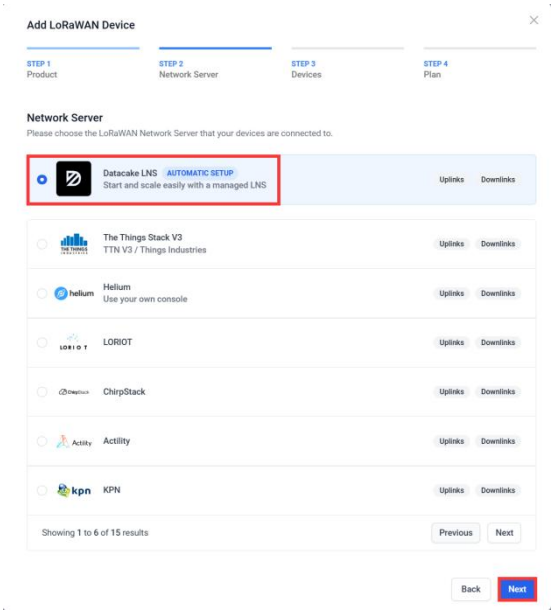
3、Select LoRaWAN and click “Next”



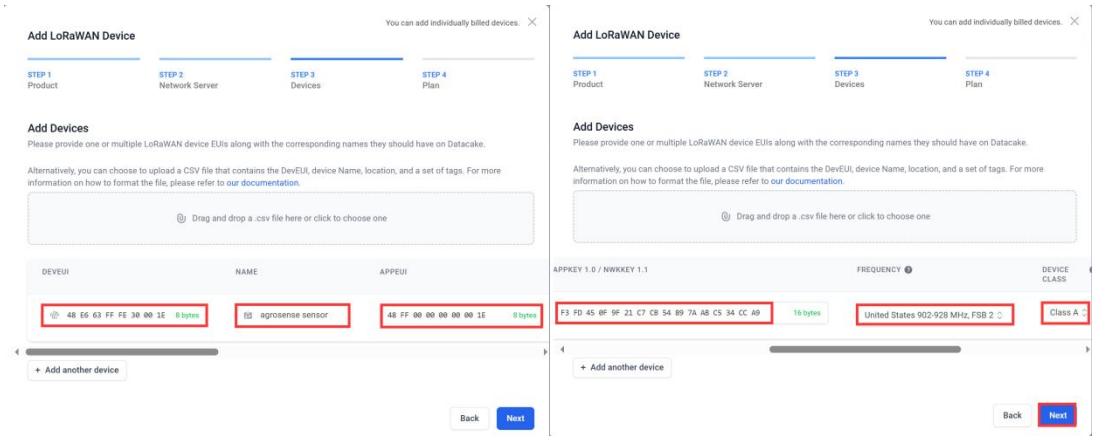
4、Select a Product based on your needs, take "Create new empty product" as an example.



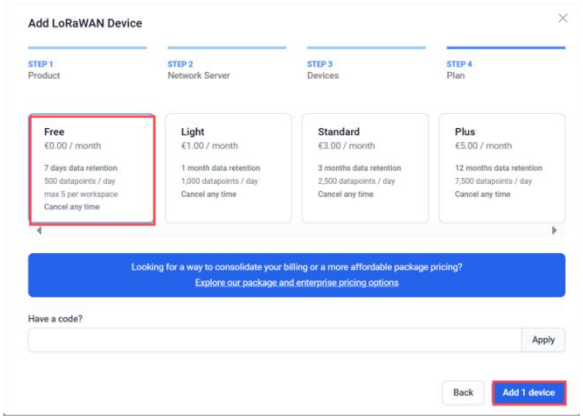
5、Select “Datacake LNS”



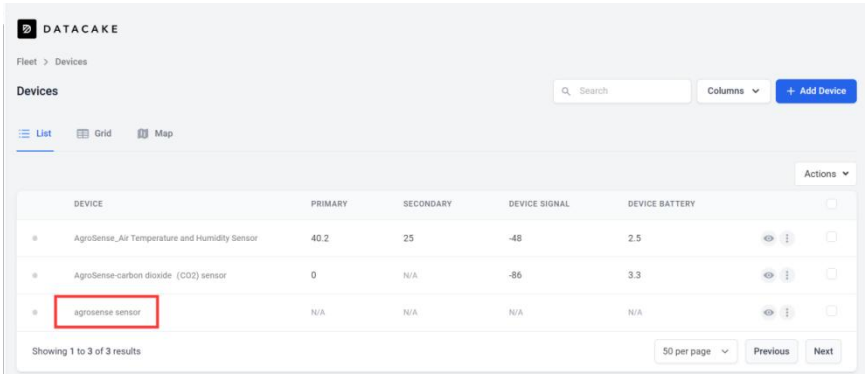
6、Enter DEVEUI、APPEUI、APPKEY、FREQUENCY(take 915 for example) and DEVICE CLASS.



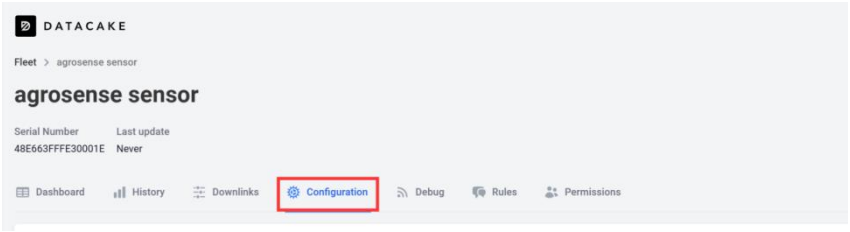
7、Choose the type according to your needs, and click “Add 1 device”.

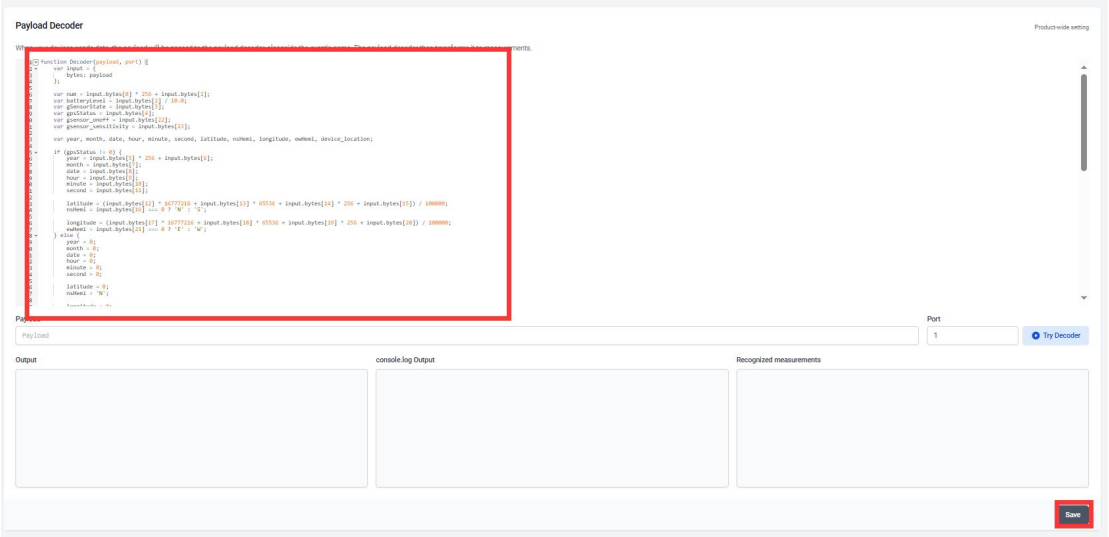


8、Click to go to the device you just added.



9、Click “Configuration”, enter Decoder and click “Save”.(You can check it out on Guihub)

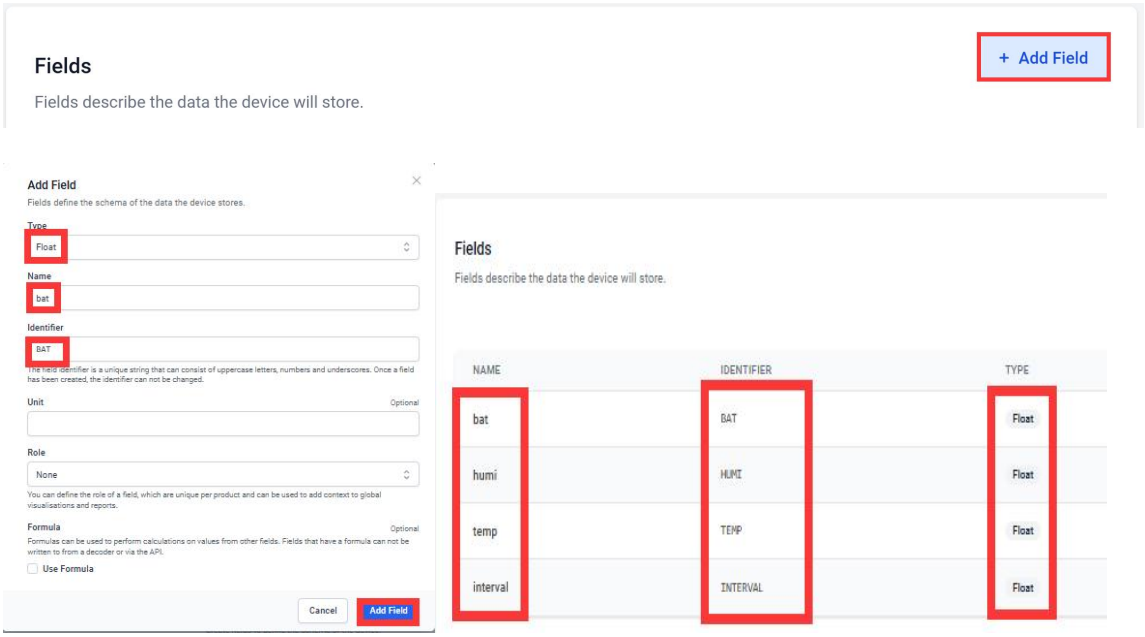




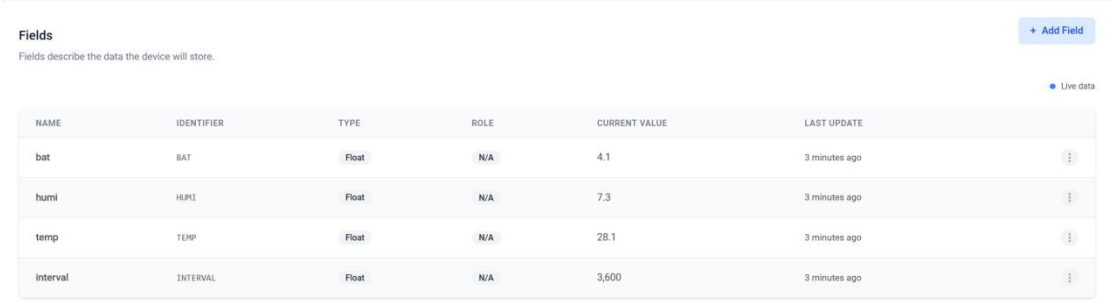
```
function Decoder(payload, port) {
    var input = {
        bytes: payload
    };

    // var num = input.bytes[0] * 256 + input.bytes[1];
    var bat = input.bytes[2] / 10.0
    var Significant = input.bytes[3]
    var humi = (input.bytes[4] * 256 + input.bytes[5]) / 10.0

    var temp = input.bytes[6] * 256 + input.bytes[7]
    if (temp >= 0x8000) {
        temp -= 0x10000;
    }
    temp = temp / 10.0
    var interval = (input.bytes[8] * 16777216 + input.bytes[9] * 65536 + input.bytes[10] * 256 + input.bytes[11])
/ 1000

    var decoded = {
        bat: bat,
        Significant: Significant,
        humi: humi,
        temp: temp,
        interval: interval,
    };

    // Test for LoRa properties in normalizedPayload
    try {
        console.log('normalizedPayload:', normalizedPayload);    // Log to check normalizedPayload structure

        decoded.lora_rssi =
            (normalizedPayload.gateways && Array.isArray(normalizedPayload.gateways) &&
normalizedPayload.gateways.length > 0 && normalizedPayload.gateways[0].rssi) || 0;
        decoded.lora_snr =
            (normalizedPayload.gateways && Array.isArray(normalizedPayload.gateways) &&
normalizedPayload.gateways.length > 0 && normalizedPayload.gateways[0].snr) || 0;
        decoded.lora_datarate = normalizedPayload.data_rate || 'not retrievable';
    } catch (error) {
        console.log('Error occurred while decoding LoRa properties: ' + error);
    }

    if (Significant) {
        return [
            { field: "bat", value: decoded.bat },
```

```
                { field: "humi", value: decoded.humi },
                { field: "temp", value: decoded.temp },
                { field: "interval", value: decoded.interval },
                { field: "lora_rssi", value: decoded.lora_rssi },
                { field: "lora_snr", value: decoded.lora_snr },
                { field: "lora_datarate", value: decoded.lora_datarate }
            ];
    }
    else {
            return [
                { field: "Significant", value: "data invalid" },
            ];
    }
}
```

10、Follow the steps to add a field. (Every fields is the same way)



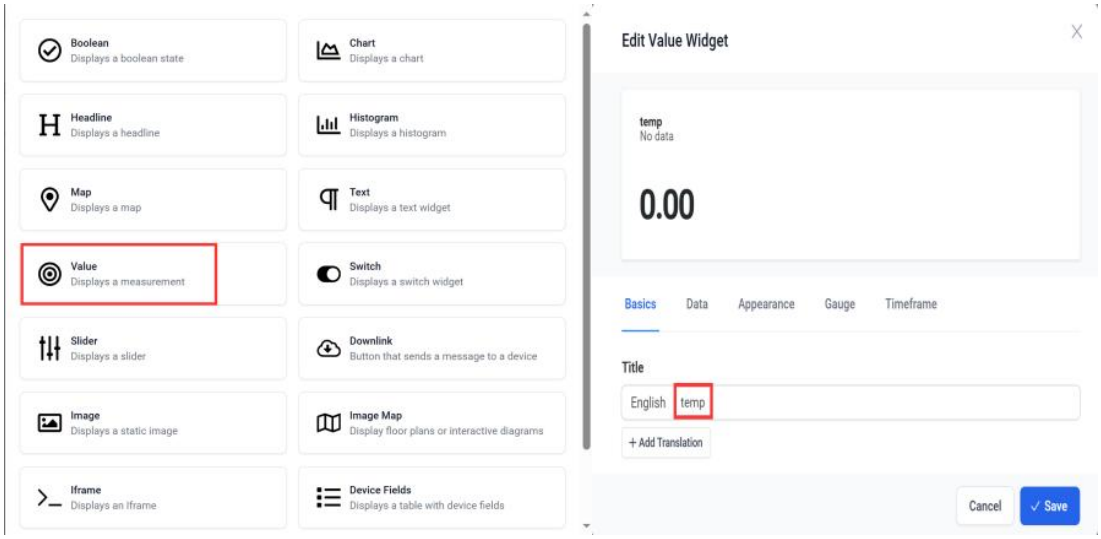
11、Press RST button, wait until the sensor connects to the gateway successfully, you will see the data the sensor is currently reading.

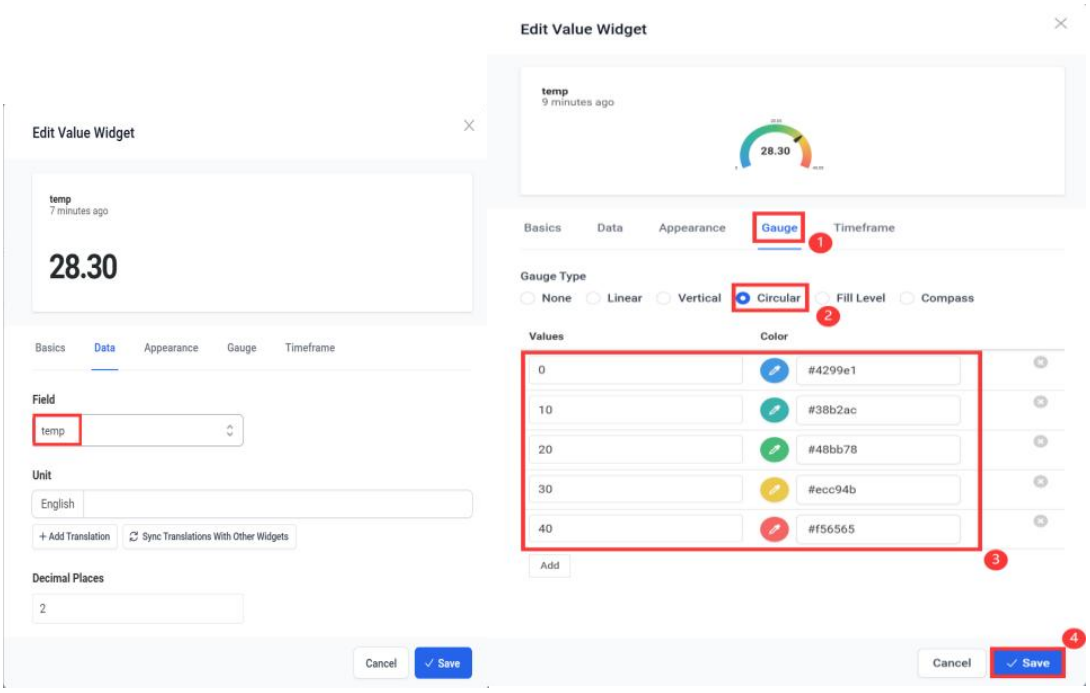


12、To get a better look at the data, we can add widget.

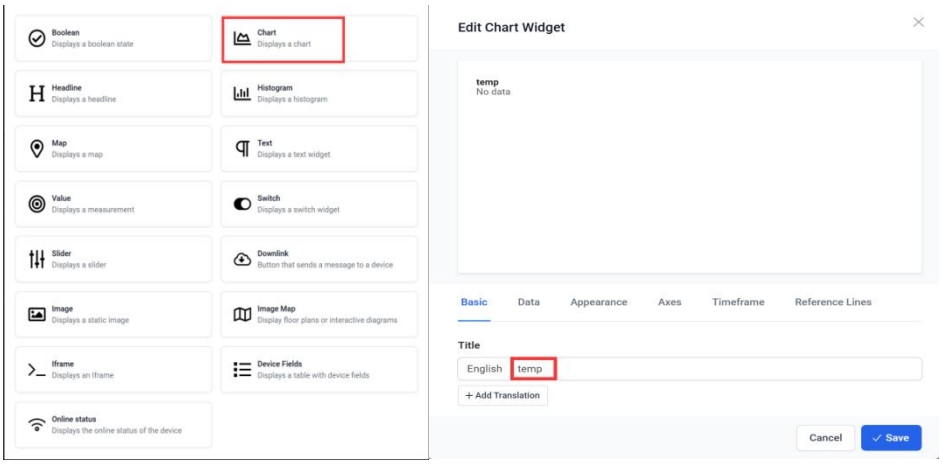
Click “Dashboard-->switch-->+ Add Widget”.

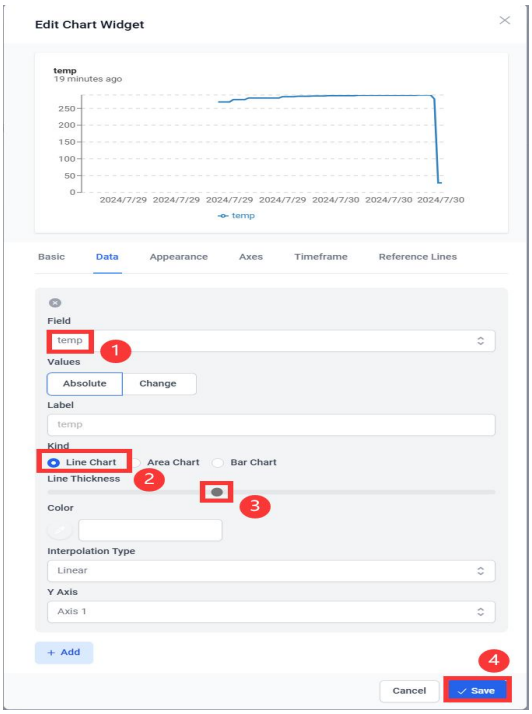
13、Select “Value” and set Title, Field and presentation form as well as the interval color.



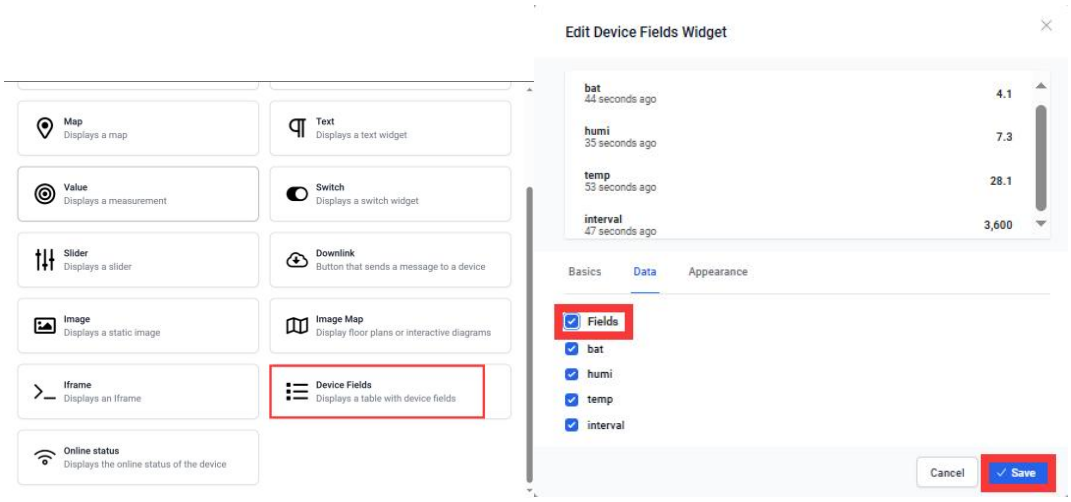


14、Select Chart and set Title, Field, Kind, Line Thickness and click “save”.



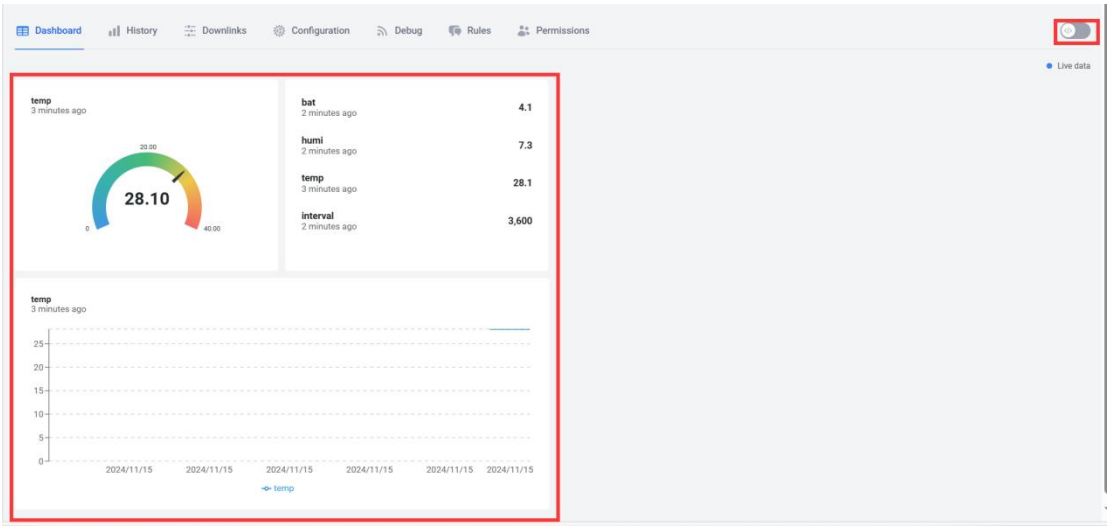


15、Select Device Fields, check “Fields” and click “Save”.



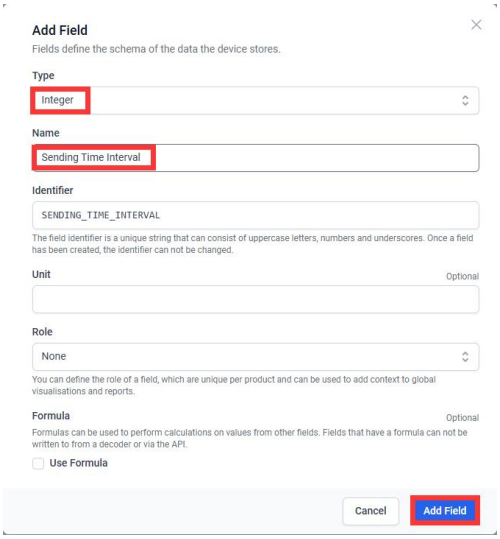
16、Click the switch to save, and you can see the data visually.

17、The steps for humidity are the same as above, and you can add your own.

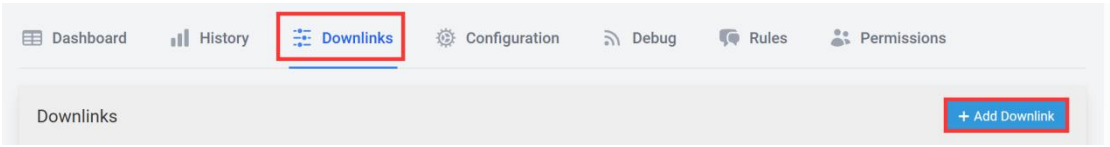


3.2.1 Change Time Interval (5-1440min)

1 、 If you need to change time Interval (Default 60 minutes), you can click “Configuration-->Fields-->+Add Field”



2、Click ”Downlink-->Add Downlink”.



Enter name、description、fields used and payload encoder respectively.

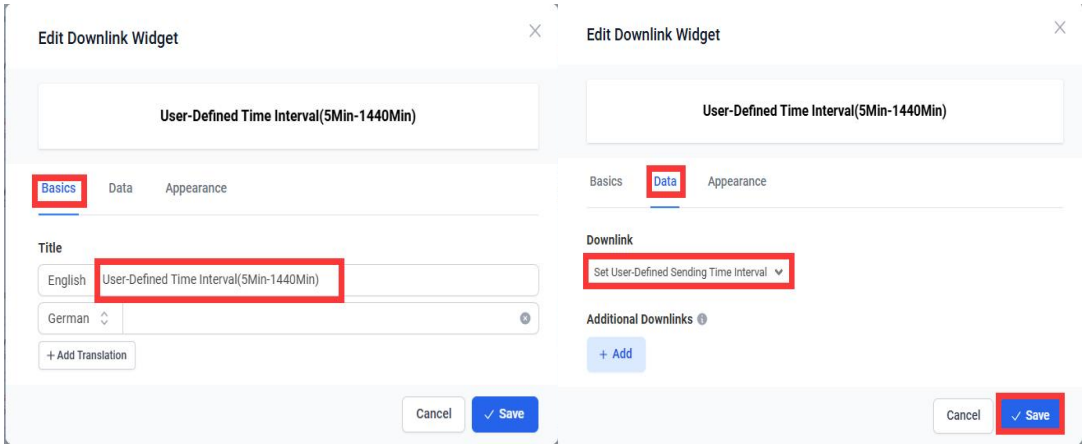
Name: Set User-Defined Sending Time Interval

Description: Set the user-defined report transmission interval and store it in the configuration variable.(5Min-1440Min)

Payload Encoder: copy in Github.

3、Click “Dashboard-->switch-->+ Add Widget”.

Select “Downlink” and setting as follow image.



4、Click the switch to save, and you can click to change your time Interval.