



如何将 Milesight 的网关和设备集成到 TagoIO 平台



Version Change Log			
Version	Revision Date	Revision Details	Revised By
V1.0	20250518	Initial	Lockon



前言

TagoIO 是一个端到端的物联网 (IoT) 云平台，专为快速部署和管理物联网解决方案而设计，支持设备连接、数据收集、可视化分析和用户权限管理等全流程服务。其平台特点在于“低代码”开发模式，结合拖拽式仪表板与可定制的脚本分析功能，使技术人员和非技术用户均可快速构建物联网应用。TagoIO 提供多语言 SDK (如 Node.js、Python)、完整的 RESTful API、Webhook、MQTT 支持，并预集成超过 480 种主流设备，广泛应用于智慧农业、工业自动化、资产追踪、智慧城市等场景，是中小企业快速实现 IoT 商业落地的高效工具。

本文主要介绍如何使用 UG65 网关对接 TagoIO 平台，并且在 TagoIO 平台上实时查看网关上面 Sensor 的数据。其中 Sensor 本次选择 AM319 设备作为示例。

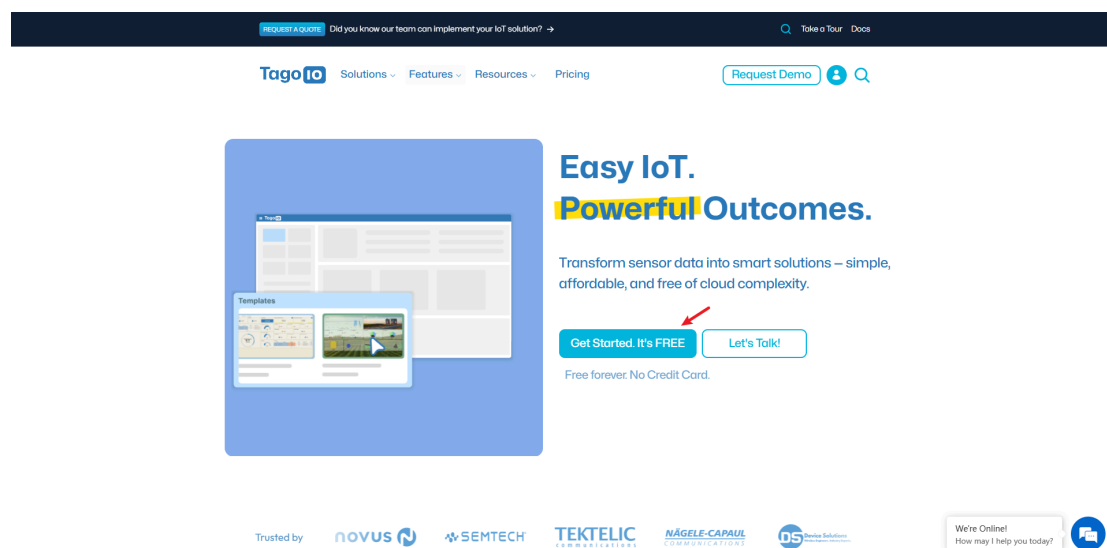
特别说明：本文提到的 AM319 设备仅是演示使用，并不代表不支持其他类型的 Sensor，读者需要根据自己的实际情况参考本文的步骤。

1. 前置条件

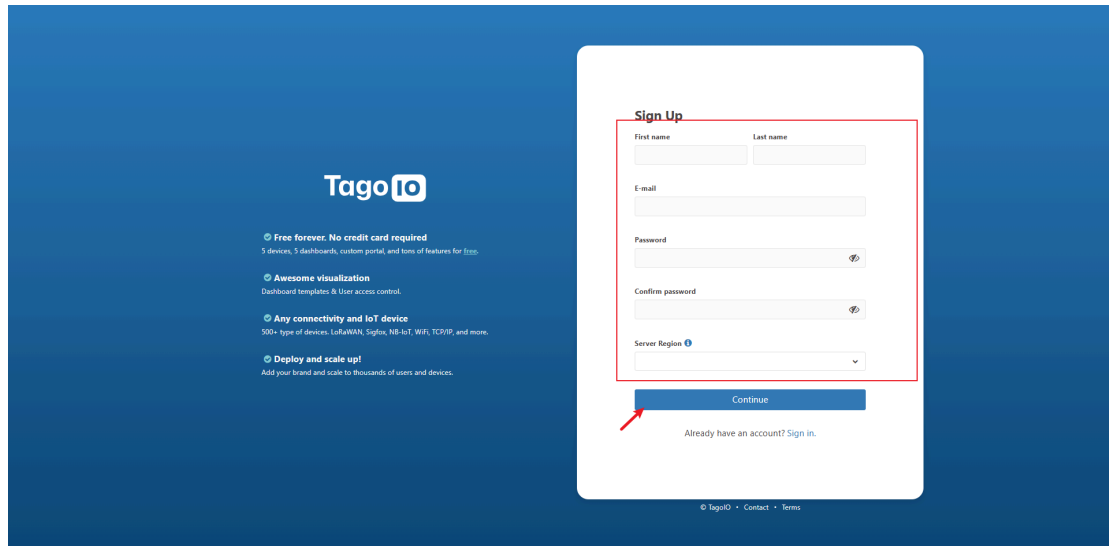
- 网关型号：UG65或者 UG56、UG67、UG63 也可以
- 传感器型号：AM319
- 本文演示用到的频段：US915
- 网关已经接入互联网

2. 注册账号

访问 [IoT Cloud Platform | TagoIO](#) 点击 “Get Started. It's FREE” 按钮：



在弹出的界面中，根据提示填写信息即可：

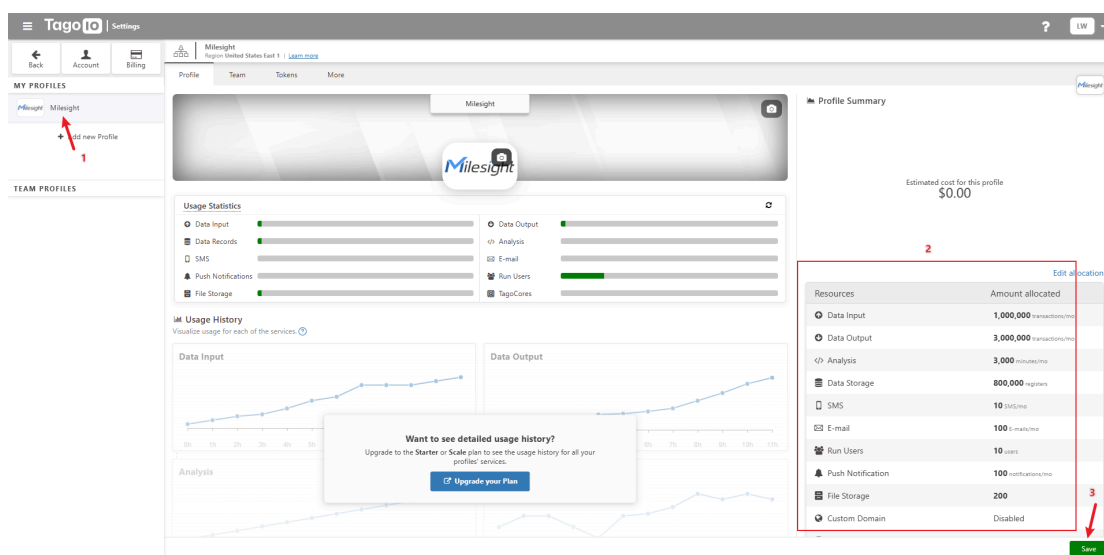
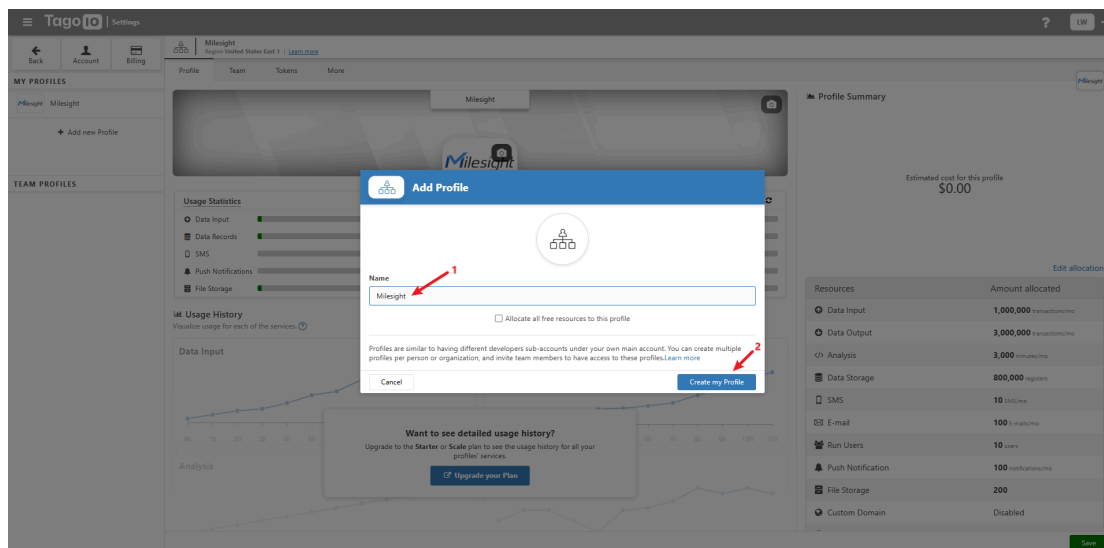
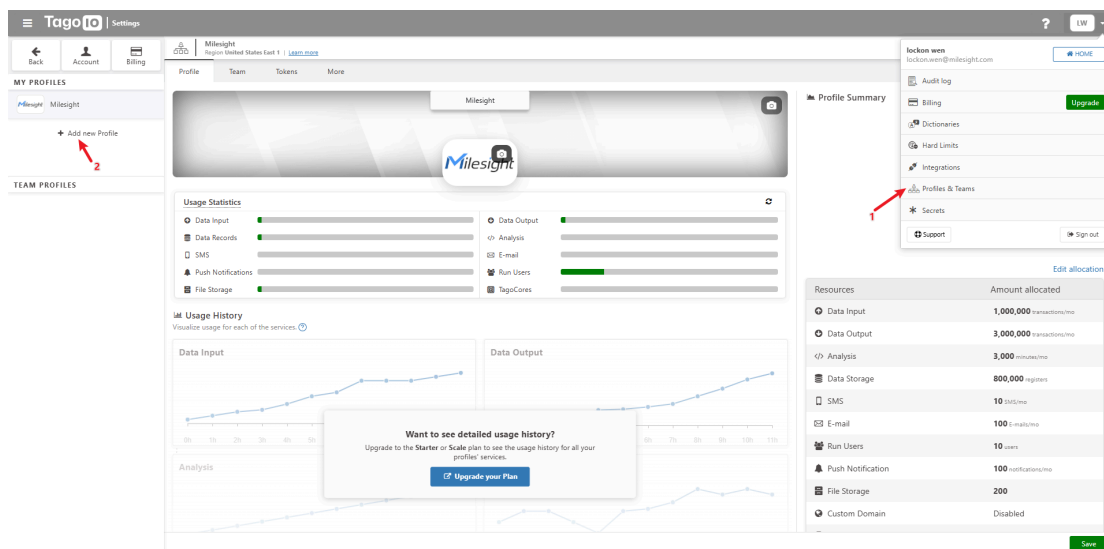


这里的 **Server Region** 我们选择 “**United States East 1**” 即可。
注册完毕后，点击你注册使用的邮箱里面的激活链接点击激活，账号就可以使用了。

3. 首次登陆

访问 [Admin](#) 输入账号密码后，如图操作创建一个 Profile 即可：



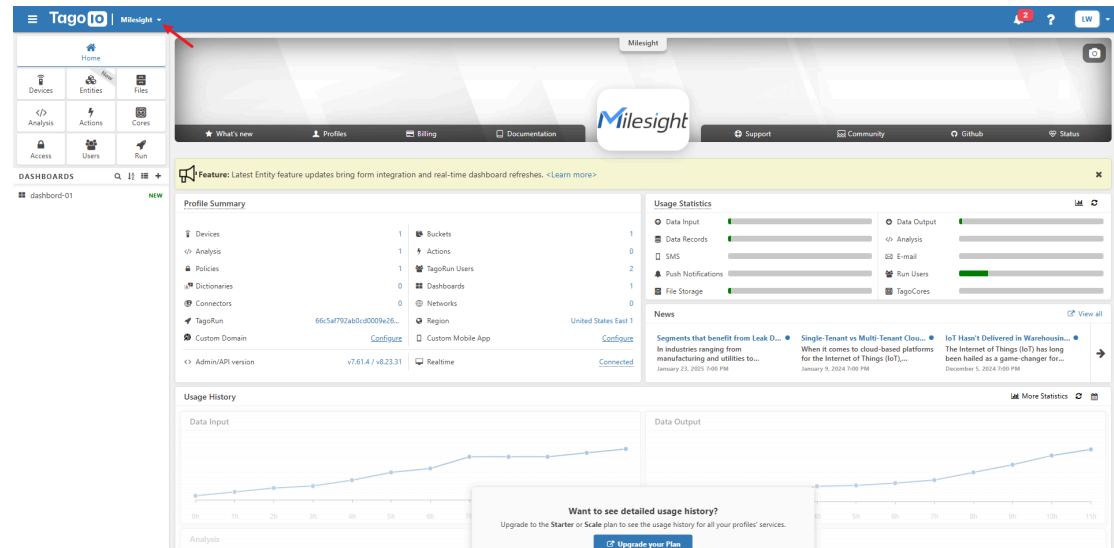


至此，属于我们自己的 Profile 就创建完了，名称是 “Milesight”，Resources 里面显示了您当前 Account Plan 的资源限制，读者可以根据自己的实际需求升级自己的 Plan 以满

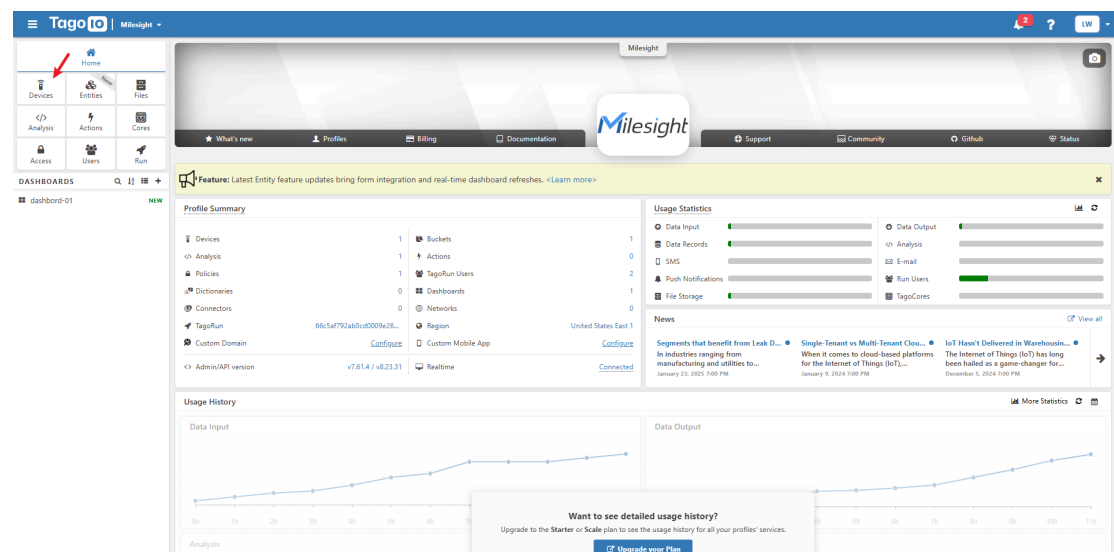
足自己的项目需要。

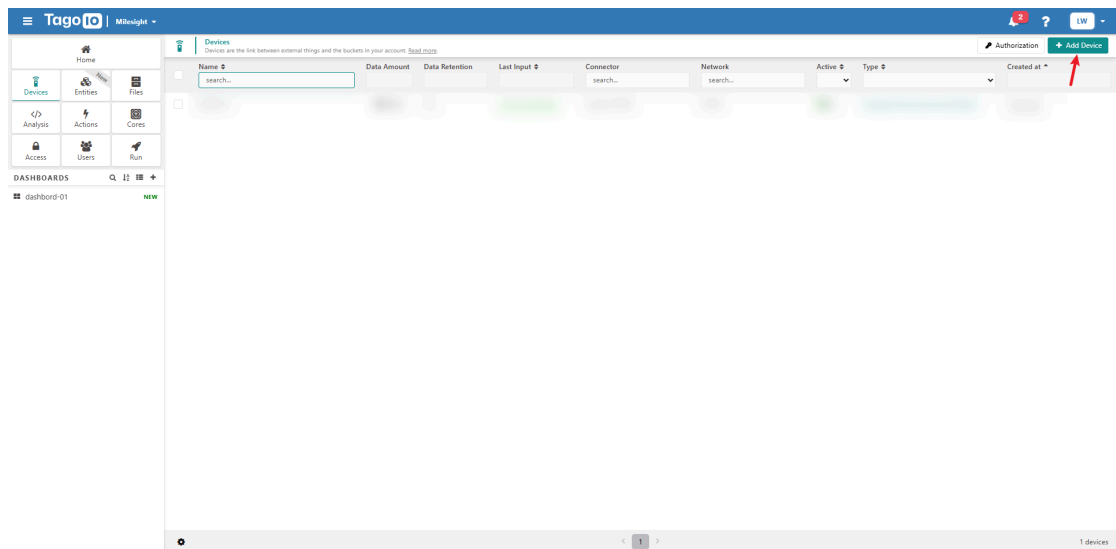
4. 创建 Device

访问 [Admin](#) ，可以看到我们在自己的 “Milesight” 区域中：

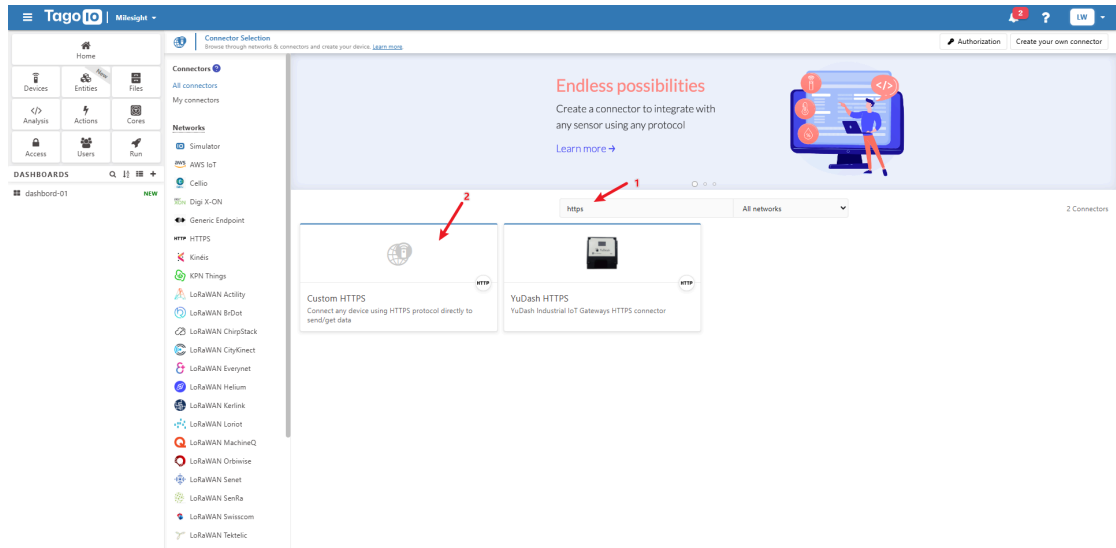


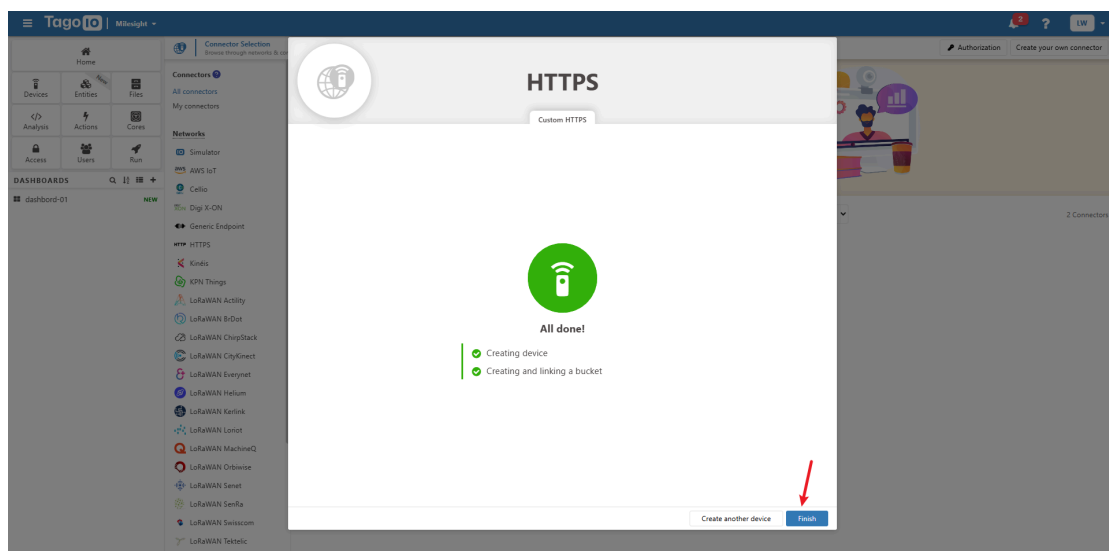
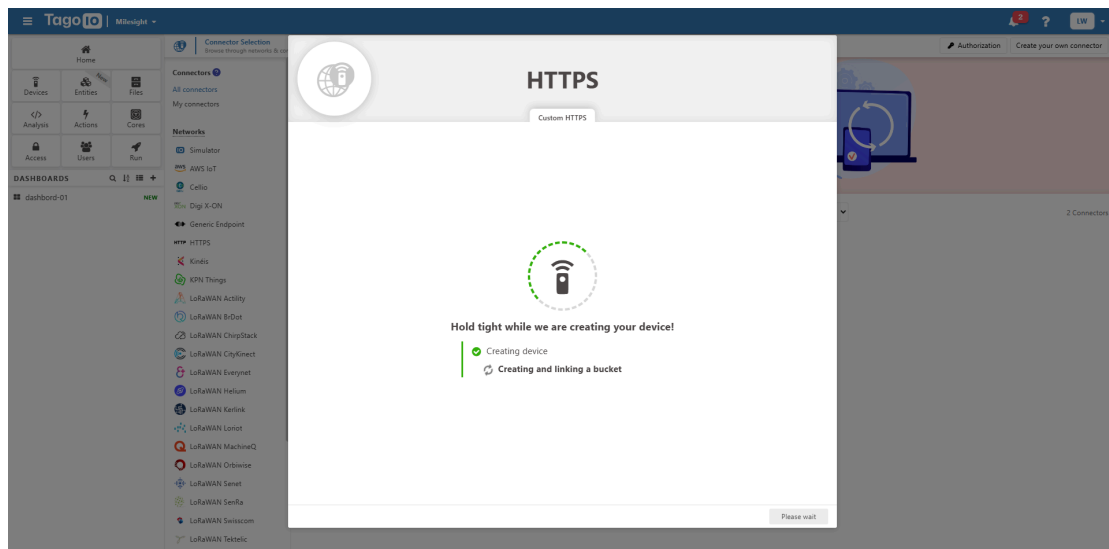
如图操作即可：



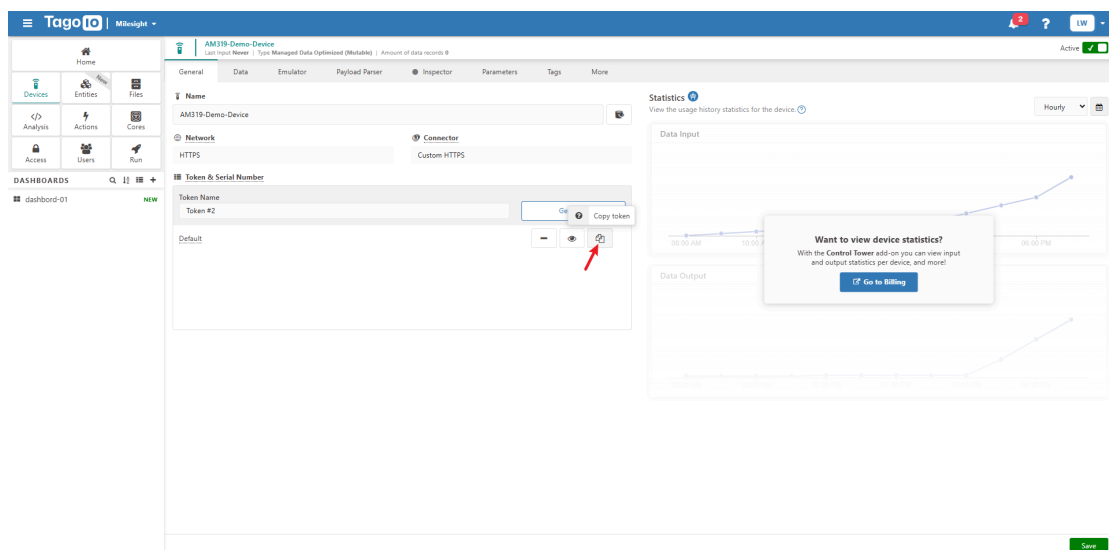


在搜索框输入 “https”:





然后如图操作，把这个 Device 的 token 拷贝出来，后面会用到：



这里的 token 的明文一般是下面的格式，本次演示所使用的是：

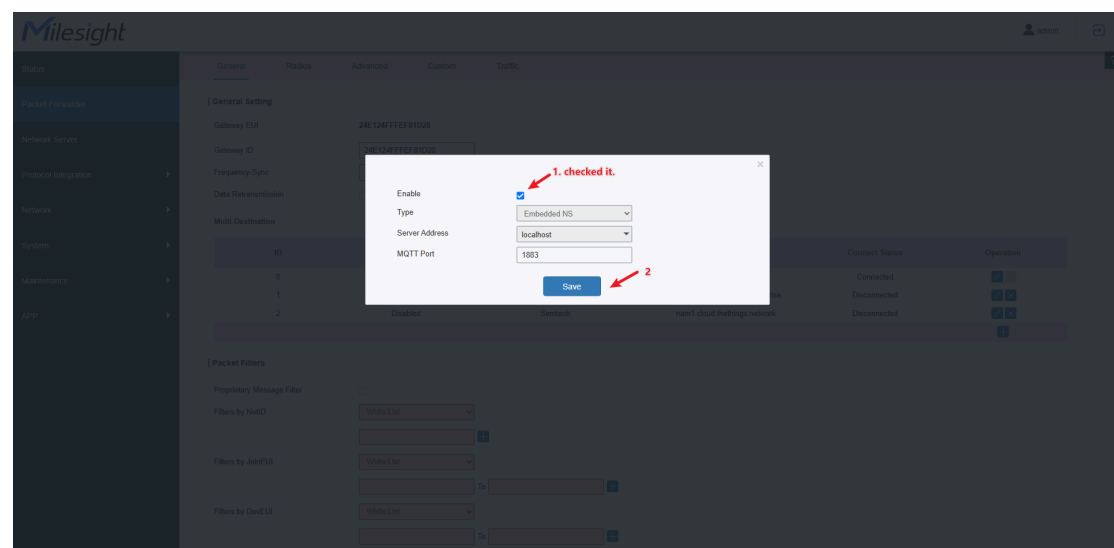
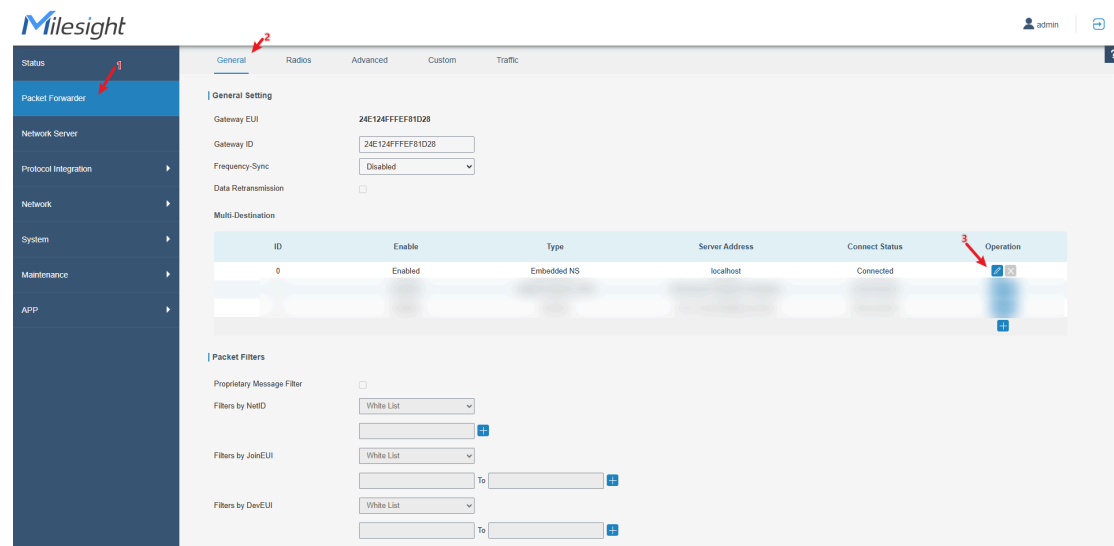
2040aa18-b7a9-4179-94cf-a12ca6c6bed4

至此，我们的 Device 创建就结束了，接下来开始配置我们的网关和 Sensor。

5. 网关配置

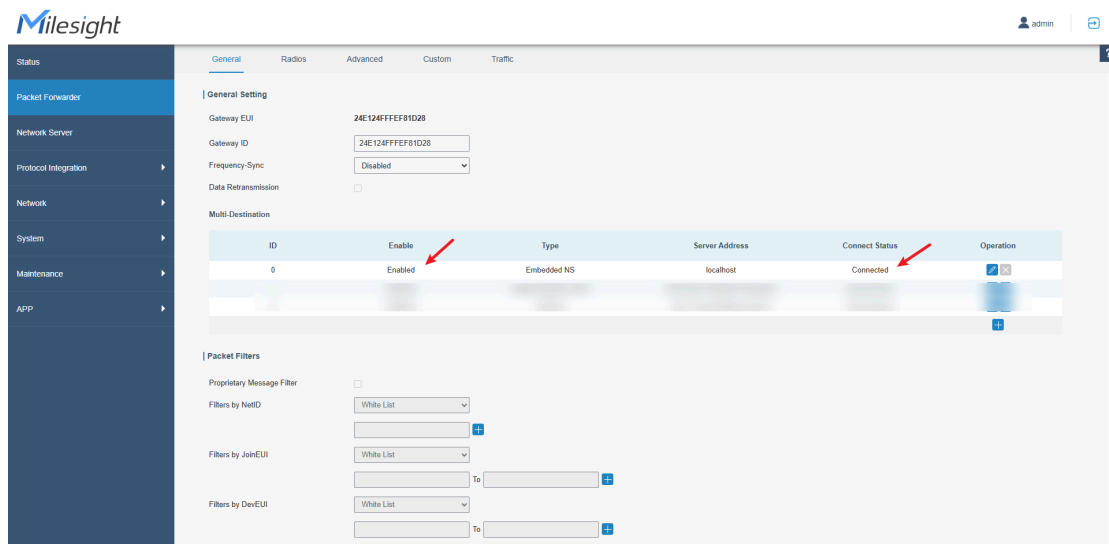
5.1. 开启内置 NS：

首先登录我们的网关管理界面（参考 <[How to Login Web GUI of Milesight Gateway](#)>）
然后参考下面的截图操作即可（如果已经开启了，这步骤可以跳过）：



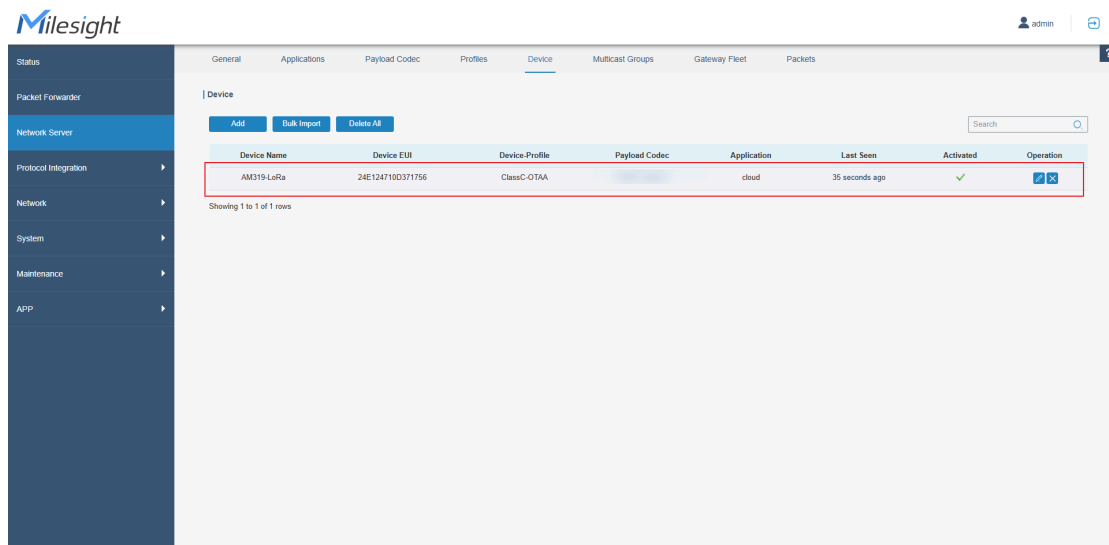
如下图，显示 Embedded NS 已经 Enabled 并且 Connected，则说明开启成功了：





5.2. 添加 Sensor

参考 <[How to Connect LoRaWAN Nodes to Milesight Gateway](#)> 操作即可：
完成后的截图如下：



5.3. 创建 Decode 代码

由于网关默认的 Decode 解析出来的数据，TagoIO 平台不识别，所以我们需要在原来的基础上修改 Decode 代码，具体操作参考下面的截图即可（本章节主要参考 <[How to Use Payload Codec on Milesight Gateway](#)>）：

Milesight

Status

Packet Forwarder

Network Server

Protocol Integration

Network

System

Maintenance

APP

General Applications **Payload Codec** Profiles Device Multicast Groups Gateway Fleet Packets

Note: Ensure that the Internet access is available.

Name	Payload Decoder Function	Payload Encoder Function	Object Mapping Function	Details
AM102	✓	✓	✓	?
AM102L	✓	✓	✓	?
AM103	✓	✓	✓	?
AM103L	✓	✓	✓	?
AM104	✓	✓	✓	?
AM107	✓	✓	✓	?
AM307	✓	✓	✓	?
AM307L	✓	✓	✓	?
AM308	✓	✓	✓	?
AM308L	✓	✓	✓	?

Showing 1 to 10 of 106 rows 10 rows per page

Custom Payload Codec

Name	Description	Payload Decoder Function	Payload Encoder Function	Object Mapping Function	Operation

Showing 1 to 2 of 2 rows

Milesight

Status

Packet Forwarder

Network Server

Protocol Integration

Network

System

Maintenance

APP

General Applications **Payload Codec** Profiles Device Multicast Groups Gateway Fleet Packets

Custom Payload Codec

Name AM319-TagIO

Description

Template AM319-HCHO-IR

Payload Decoder

Payload Decoder Function

```
1 /*  
2  * Payload Decoder  
3  *  
4  * Copyright 2025 Milesight IoT  
5  *  
6  * @product AM319-HCHO  
7  */  
8 = function Decode(payload, bytes) {  
9  
10     var output = [];  
11     var input = milesight(bytes);  
12  
13     for (var key in input) {  
14         if (input.hasOwnProperty(key)) {  
15             output.push({  
16                 variable: key,  
17                 value: input[key]  
18             });  
19  
20     }  
21 }  
22
```

Payload Encoder

Payload Encoder Function

```
1 /*  
2  * Payload Encoder  
3  *  
4  * Copyright 2025 Milesight IoT  
5  *  
6  * @product AM319-HCHO  
7  */  
8 = function Encode(payload, obj) {  
9     var encoded = [];  
10  
11     return encoded;  
12 }
```

然后把函数 Decode() 的代码替换成下面的代码：



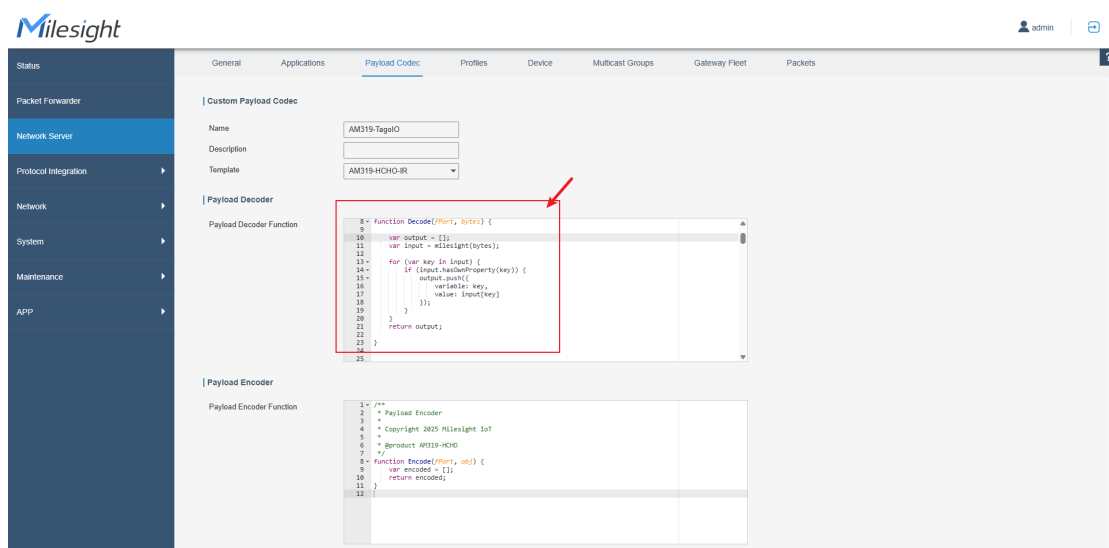
```
function Decode(fPort, bytes) {

    var output = [];
    var input = milesight(bytes);

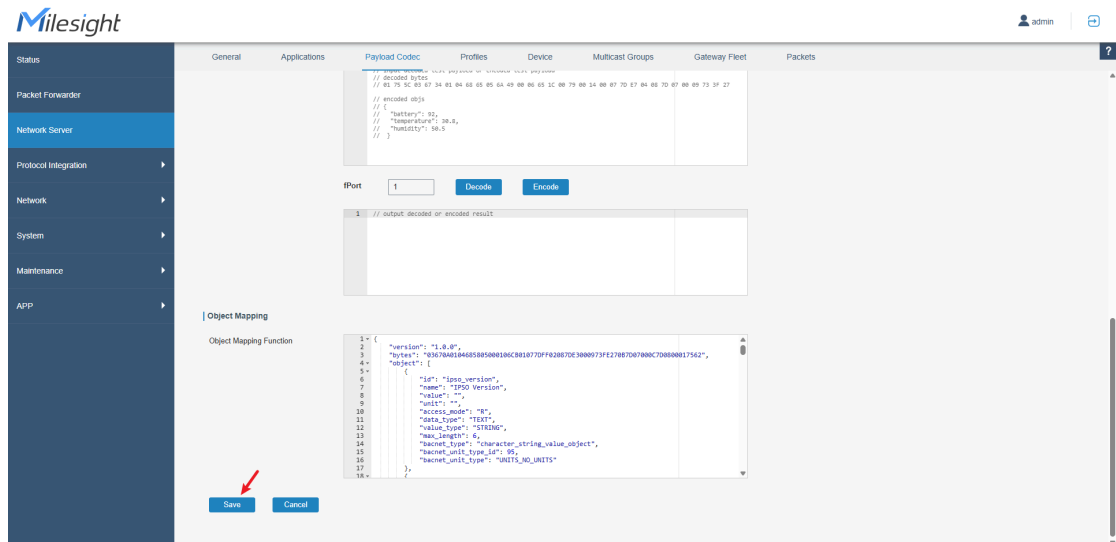
    for (var key in input) {
        if (input.hasOwnProperty(key)) {
            output.push({
                variable: key,
                value: input[key]
            });
        }
    }

    return output;
}
```

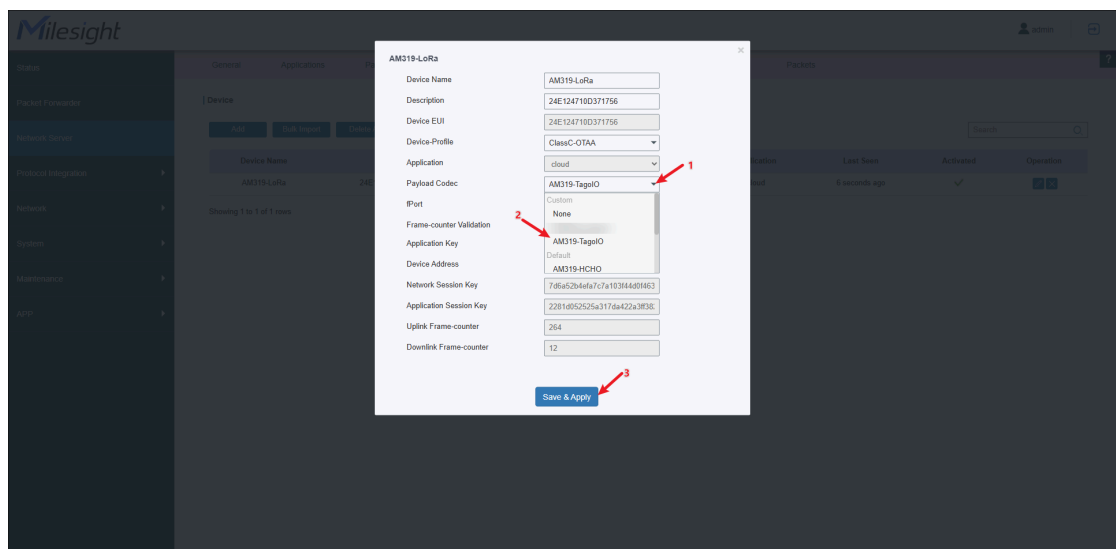
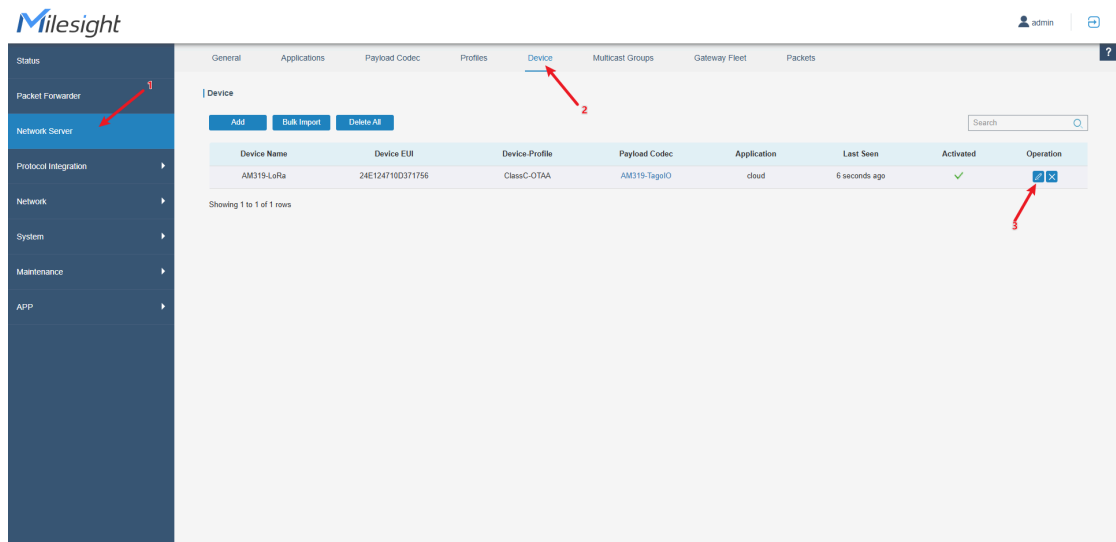
完成后的结果如下：



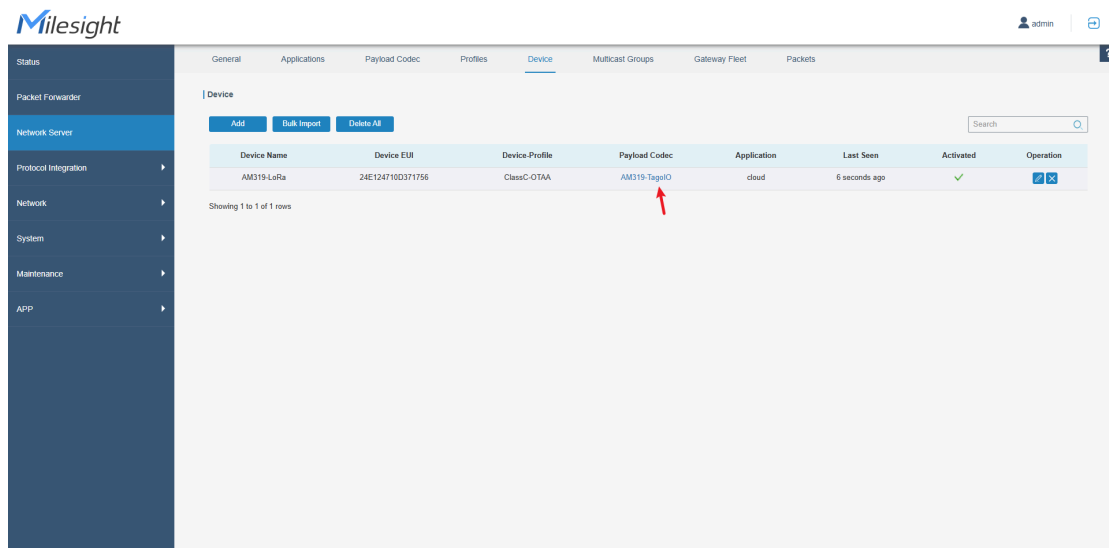
最后点击 “Save” 保存：



接下来配置 Device 和 Decode 的关联关系，如图操作：

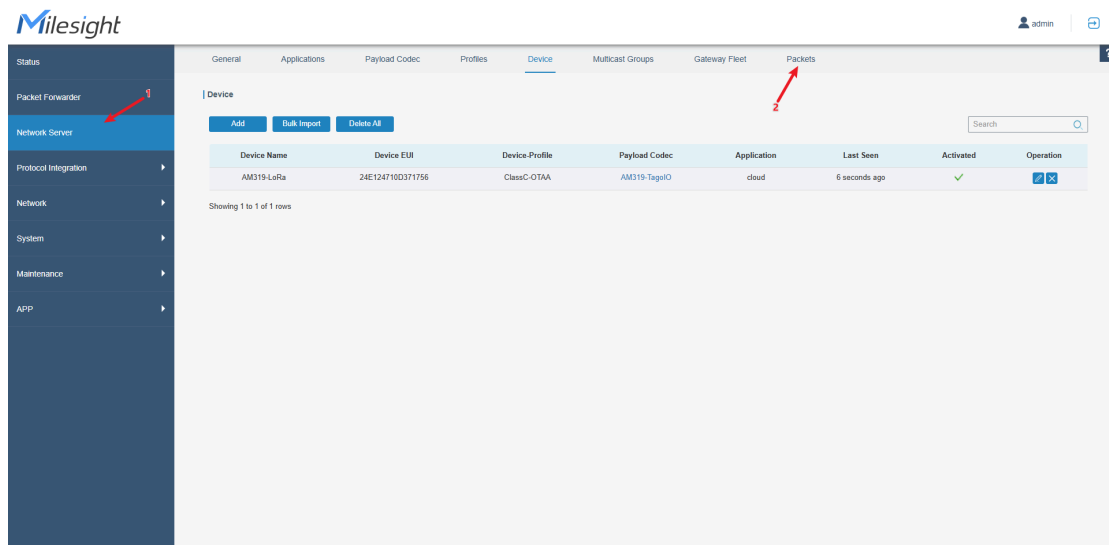


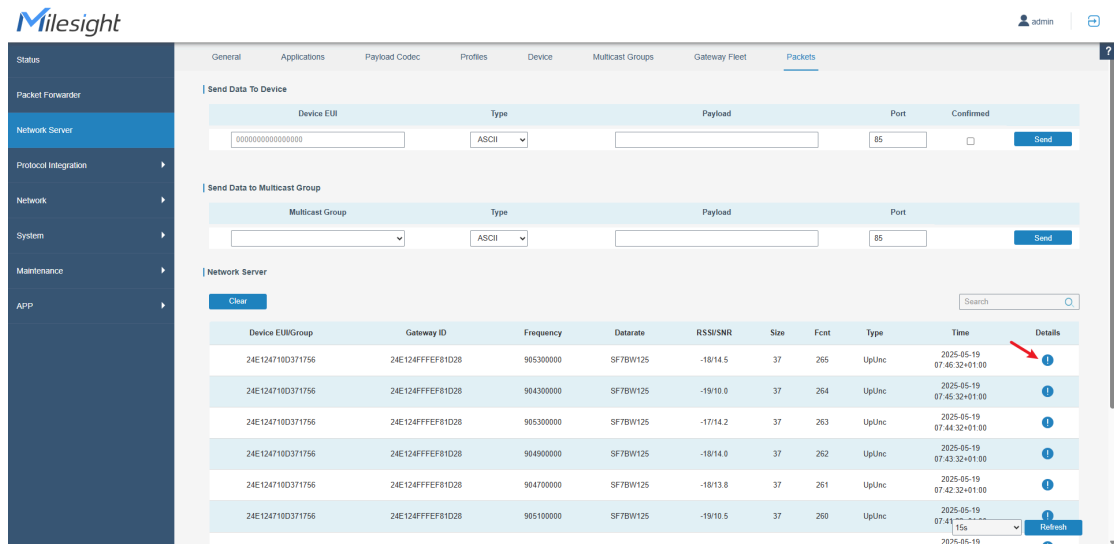
配置完成后的结果如下：



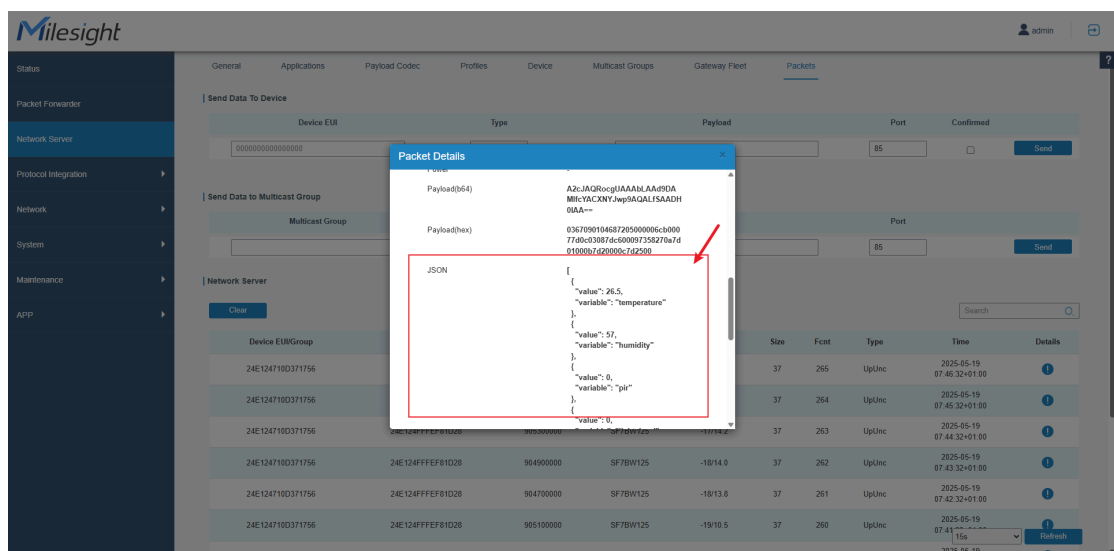
5.4. 查看数据情况

接下来检查一下我们的配置之后的 Sensor 的解析出来的数据是否符合我们的预期，如图操作：



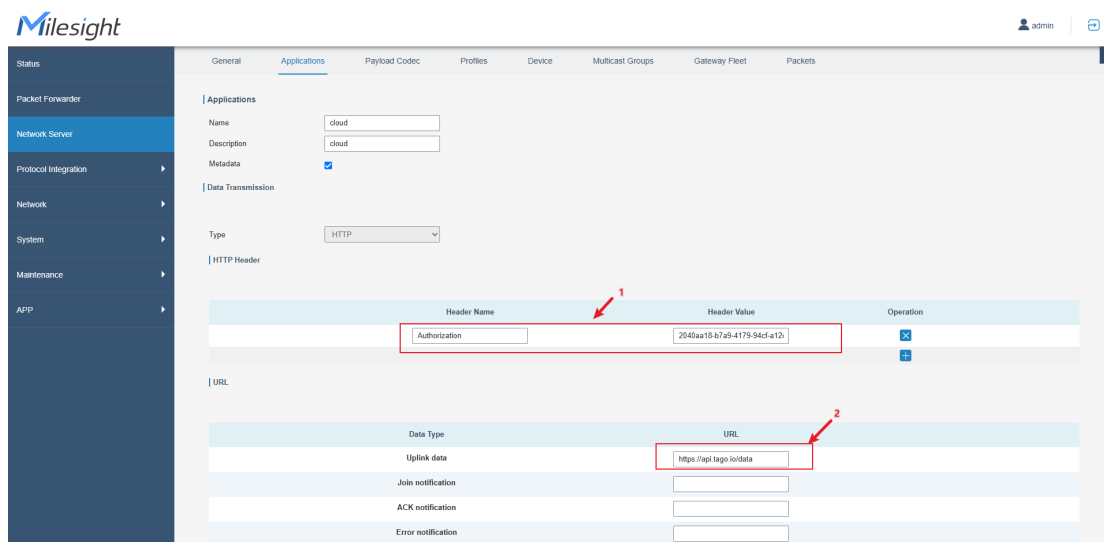


观察 JSON 的显示的数据是否和下图一样，如果为空或者别的格式，说明前面的操作配错了，需要重新检查：



5.5. 配置 HTTP 参数

参考 <[How to Connect Milesight LoRaWAN Gateway to HTTP\(s\) Server?](#)>操作即可，配置完成后的截图如下：



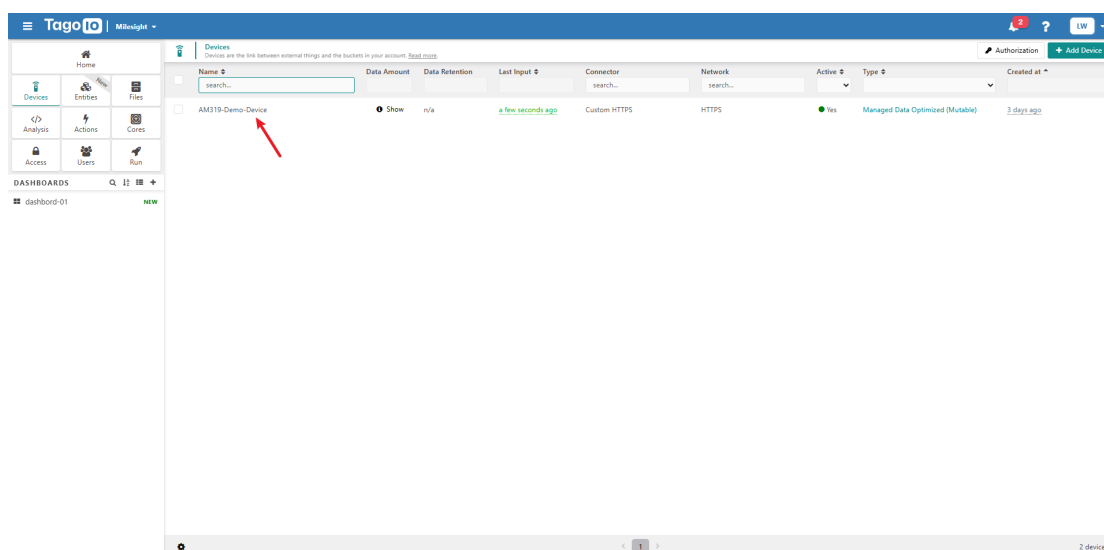
注意：

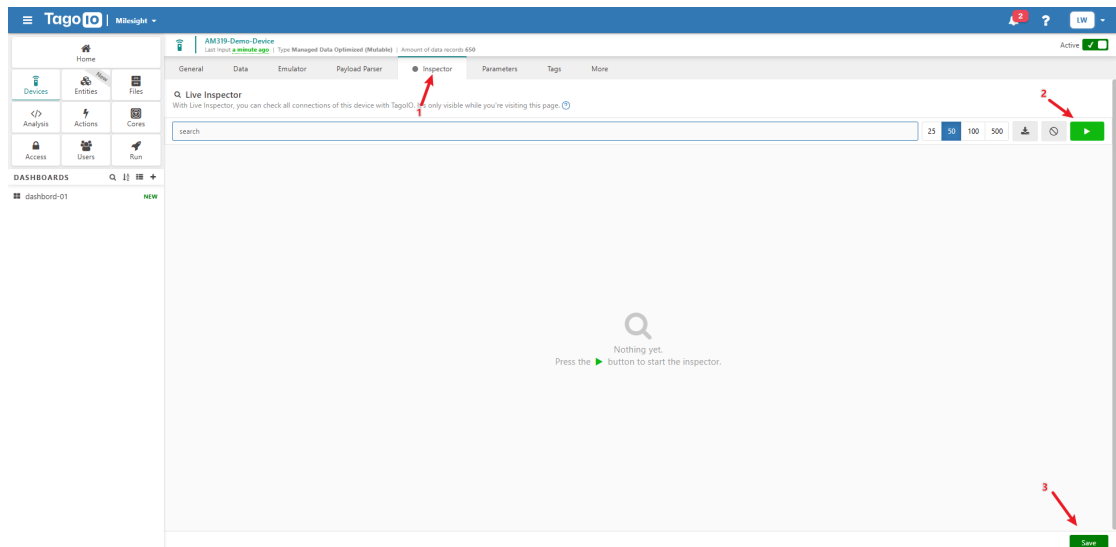
- **Uplink data** 的地址是 <https://api.tago.io/data> 不要填错。
- **Authorization** 就是我们在 第 4 步获取的参数

至此，我们的网关和 Sensor 的配置就全部结束了，接下来回到 TagoIO 平台观察 Device 的数据上报情况。

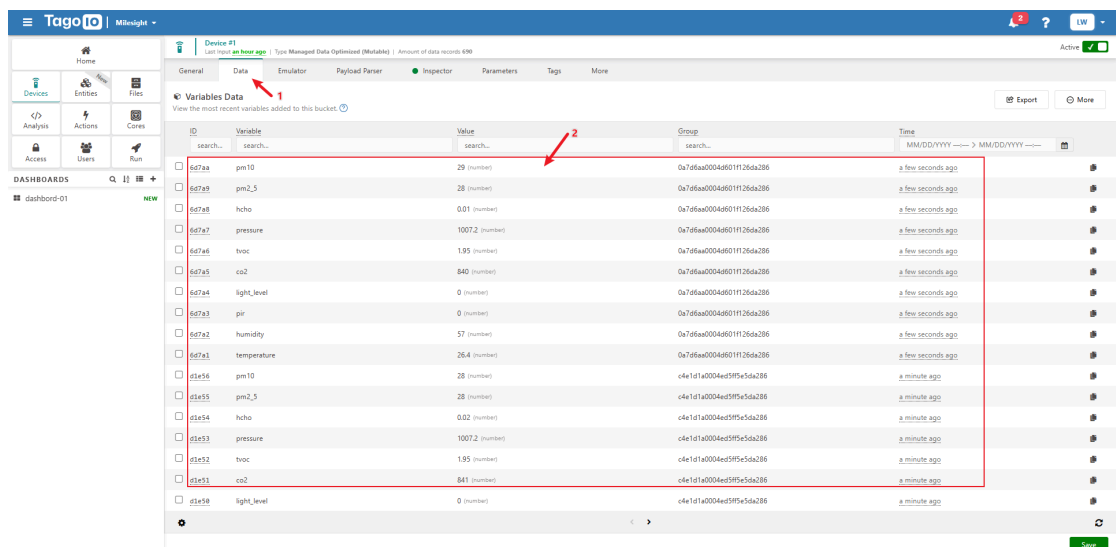
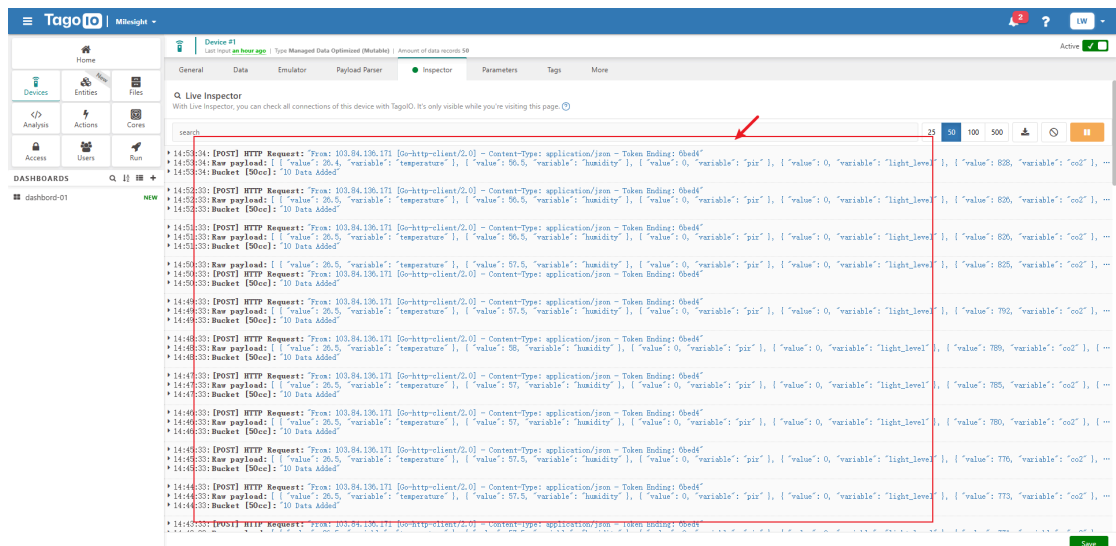
6. 观察 Device 数据

如图进行操作即可：



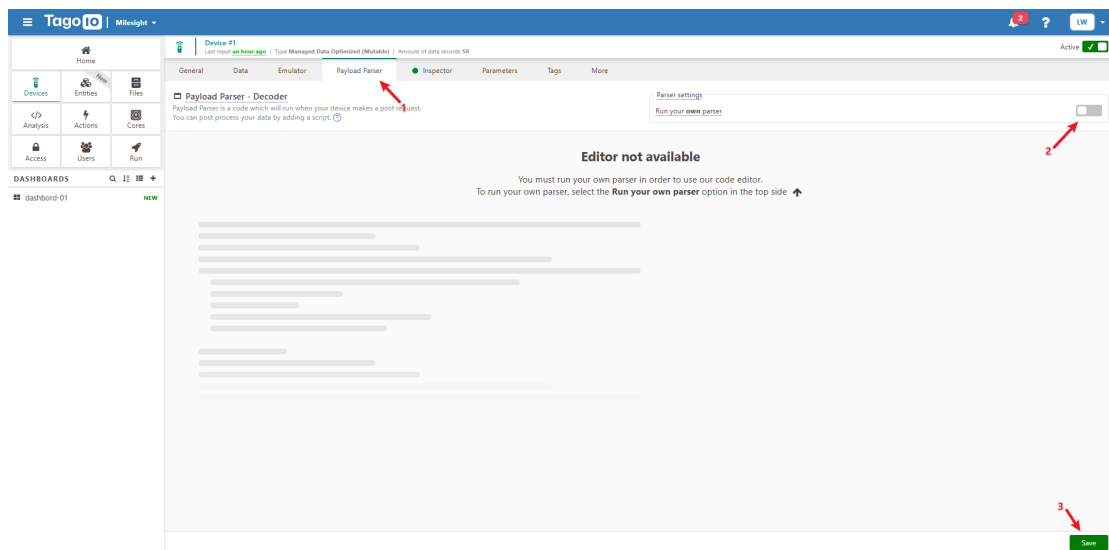


稍等片刻，会看到如下所示的信息：

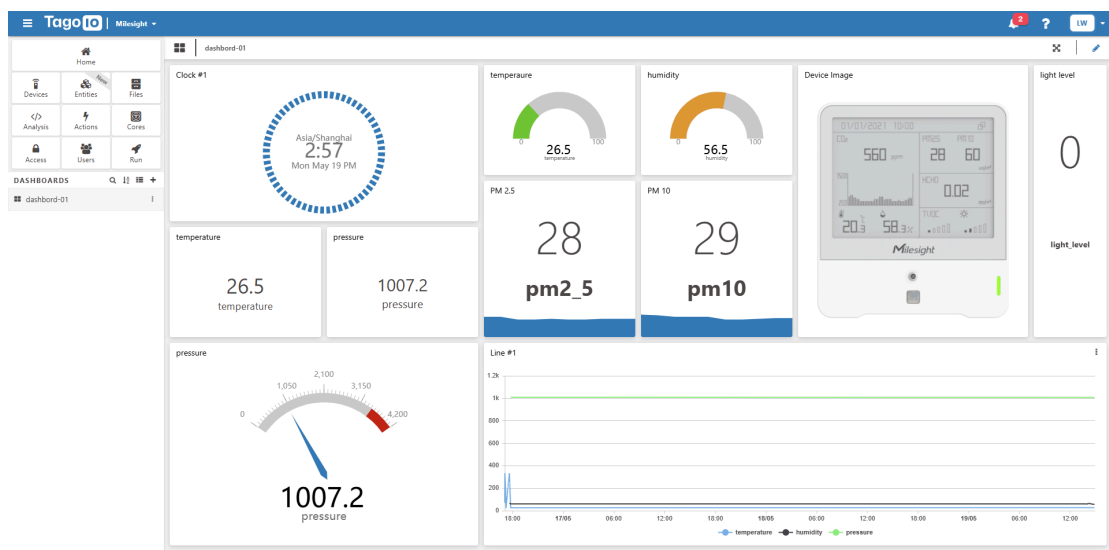


说明 TagoIO 平台已经收到了数据。

另外，需要注意的是，由于我们的 Sensor 是在 网关上面解析的，所以 TagoIO 自带的 Payload Parser 需要禁用，配置成如图所示即可：



7. 创建 Dashboard 示例



-END-