Indoor Air Quality Detector

USER MANUAL

Shenzhen RAKwireless Technology Co., Ltd. 深圳市瑞科慧联科技有限公司
Room 506, Bldg B, New Compark, Pingshan First Road, Taoyuan Street, Nanshan District,
Shenzhen, Guangdong Provice, The People's Republic of China

www.RAKwireless.com
inquiry@RAKwireless.com
+86-755-86108311

Document Information	
Name	Indoor Air Quality Detector User Manual
Classification	Technical Documentation
Revision Information	
v01	09/18/2024

Table of Contents

1. Overview

Figure 1. Indoor Air Quality Sensor

1.1. Description

The Indoor Air Quality (IAQ) Sensor features a modular design specifically developed to monitor indoor air quality, providing accurate and stable monitoring data for temperature, humidity, CO_2 concentration, TVOC, PM2.5, and PM10. Additionally, it can monitor data for CO, HCHO, O3, and NO2 if needed.

This sensor complies with WELL and RESET standards, making it ideal for use in ventilation systems and real-time indoor air quality monitoring. It can continuously monitor indoor air quality and upload the data, making it highly suitable for applications in schools, residential spaces, offices, hotels, and shopping centers.

1.2. Features

- **24-Hour Real-Time Air Quality Monitoring**: Provides real-time monitoring of PM2.5, PM10, CO_2, TVOC, temperature, humidity.
    - **(Optional)** Other sensor data: CO, HCHO, O_3, and NO_2.
- **Advanced Sensor Technology**: Utilizes proprietary patented technology and integrated environmental temperature and humidity compensation to ensure accurate and stable measurements.
- **Intelligent Data Processing**: Employs big data processing and curve fitting calibration for TVOC measurement to avoid measurement jumps or deviations caused by external factors.

- **WELL v2 Compliance**: Meets the stringent WELL v2 standard for indoor air quality.
- **Easy Installation**: Supports ceiling or wall installation to accommodate different decoration styles.
- **LED Indicator (Optional):** A light ring with a color-changing LED visually displays indoor air quality levels.
- **IoT Connectivity:** Allows remote monitoring and data analysis through integration with IoT systems, facilitating proactive air quality management and maintenance scheduling.

2. Specifications

2.1. Main Specifications

Table 2: Main Specifications

PARAMETER	SPECIFICATION
LoRaWAN® feature	RX Sensitivity: -140 dBm
Transmit Power	22 dBm
Frequency	RU864, IN865, EU868, US915, AU915, KR920, AS923-1/2/3/4
Power Supply	100~240V_{AC}
Ingress Protection	IP30
Enclose material	PC + ABS (flame retardant material)
Operating environment	Temperature: 0~50° C Humidity: 0~90%RH
Installation Method	Ceiling and wall mounting

2.2. Interfaces

2.2.1. LED Indicator and DIP Modes

A light ring in the middle of the housing indicates the concentration range of the measured value. Its color changes according to the concentration.

The indicator light's measured value can be set to an average of one-minute, one-hour, or 24-hour through the communication command.

By default, the light is controlled by the one-minute PM2.5 average. DIP (dual in-line package) switches adjust the indicator light to show AQI (Air Quality Index) changes, keep the green light on continuously, or turn off the light.

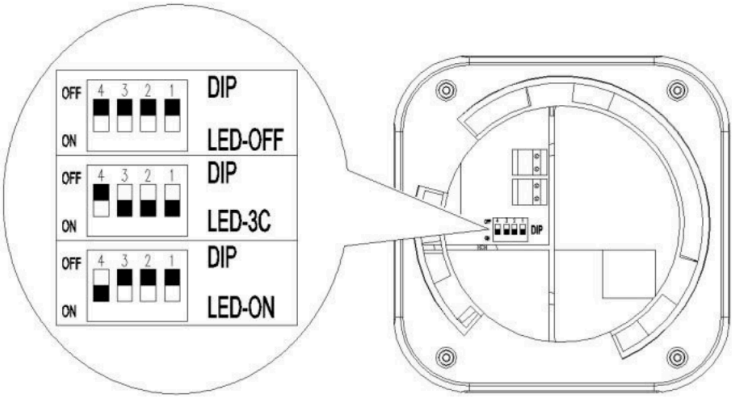

Figure 2. DIP switches

2.2.1.a. DIP Switch Settings

Indicator Lights OFF

- Switch OFF DIP1, DIP2, DIP3, and DIP4.

Three-color indicator lights (default)

- Switch ON DIP1, DIP2, and DIP3, and switch DIP4 OFF.

Green Indicator Lights (Normally ON)

- Switch OFF DIP1, DIP2, and DIP3, and switch DIP4 ON.

2.2.1.b. LED Indicator Status

Below are indicator color changes corresponding to the AQI:

Table 3: LED indicators

LED (ENABLED)	PM2.5	CO_2
🟢 Green LED lights up	< 35 μg/m^3	< 800 ppm
🟡 Yellow LED lights up	35~75 μg/m^3	800~1200 ppm
🔴 Red LED lights up	> 75 μg/m^3	> 1200 ppm

2.3. Sensor Characteristics

Table 4: Sensor Data Definitions

REGISTER NAME	SENSOR	DATA LENGTH	UNIT	RESOLUTION	RANGE	ACCURACY	DECODED FIELD NAME
Temperature	Digital integrated temperature and humidity sensor	2 bytes	° C	0.01° C	0° C~60° C	±0.5° C (10~40° C)	temperature_3
Humidity		1 byte	% RH	0.01% RH	0~99% RH	±5.0% RH (10%~90% RH)	humidity_2
CO_2	Non-dispersive Infrared Detector (NDIR)	2 bytes	ppm (unsigned)	1 ppm	400~5000 ppm	±50 ppm + 5% @ 400~2000 ppm	co2_35
TVOC	Multi-pixel gas sensor	2 bytes	mg/m^3	0.01° C	0° C~60° C	±0.5° C (10~40° C)	tvoc_16
PM 2.5 value	Laser particle sensor	2 bytes	ug/m^3	1 $\mu g/m^3$	0~1000 $\mu g/m^3$	±5 $\mu g/m^3$ + 20% @ 1~100 $\mu g/m^3$	pm2.5_41
PM 10 value		2 bytes	ug/m^3	1 $\mu g/m^3$	0~1000 $\mu g/m^3$	±5 $\mu g/m^3$ + 20% @ 1~100 $\mu g/m^3$	pm10_42

Table 5: Optional Sensor Data

REGISTER NAME	DATA LENGTH	UNIT	RESOLUTION	RANGE	PRECISION	TYPICAL LIFETIME	REMARKS
CO	2 bytes	ppm	0.1 ppm	0.1~100 ppm	±1 ppm @ 0~10 ppm	> 5 years (typ. application)	Optional
HCHO	2 bytes	ppb	1 ppb	20~1000 ppb	±20 ppb @ 0~100 ppb	> 3 years (typ. application)	Optional
O_3	2 bytes	ppb	1 ppb	10~500 ppb	±10 ppb @ 0~200 ppb		Optional
NO_2	2 bytes	ppb	1 ppb	5~500 ppb	±20 ppb at 0-100 ppb		Optional

2.4. RF Characteristics

Table 6: Wireless Parameters

PARAMETER	SPECIFICATION
Communication Protocol	Standard LoRaWAN® protocol
Supported Frequency Band	RU864, IN865, EU868, US915, AU915, KR920, AS923-1/2/3/4
Transmit Power	22 dBm
Receive Sensitivity	-140 dBm
Network Join/Work Mode	OTAA/ABP; Class A, B, and C

2.5. Mechanical Characteristics

2.5.1. Design and Dimension

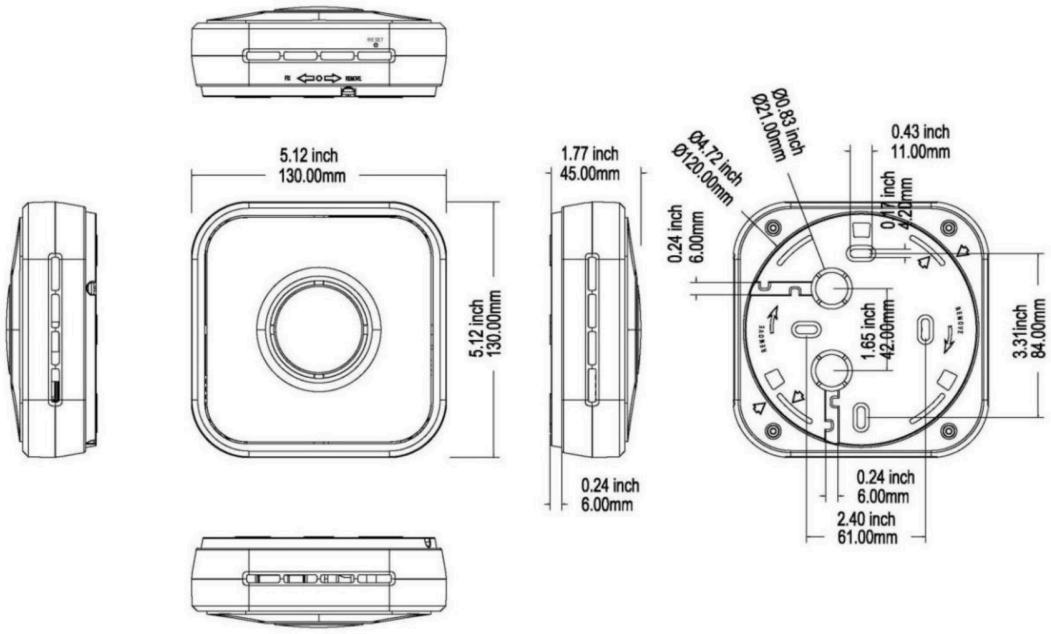

Figure 3. Device Dimensions

2.5.2. Physical Properties

Table 7: Device physical properties

PARAMETER	SPECIFICATION
IP Rating	IP30

Dimension	130 mm × 130 mm × 45 mm (l × w × t)
Power Supply	100 ~ 240V_{AC}
Installation Method	Ceiling and wall mounting

2.6. Environmental Characteristics

Table 8: Operating and Storage Conditions

PARAMETER	SPECIFICATION
Operating Temperature	0° C ~ 50° C
Storage Temperature	10° C ~ 50° C
Storage Humidity	0 < 70 % RH

2.7. Certification

Table 9: Certification

PARAMETER	SPECIFICATION
Certification Standard (CE)	**SAR**: EN 62479&50663 **Health Assessment RF**: ETSI EN 300 328 **LVD (BlueTooth):** EN 61010-1 **EMC:** EN61326-1 **EMC (Wi-Fi):** • ETSI EN 301 489-1 V2.2.3 (2019-11) • ETSI EN 301 489-17 V 3.2.4 (2020-09)

3. Installation

The IAQ sensor is a complete node device, so no assembly is required after unboxing. Refer to the following sections for instructions on how to mount the sensor in the proper location and perform relevant sensor operations.

3.1. Wall Mounting

1. To separate the backboard and the detector, rotate the backboard clockwise according to the direction of the arrow.

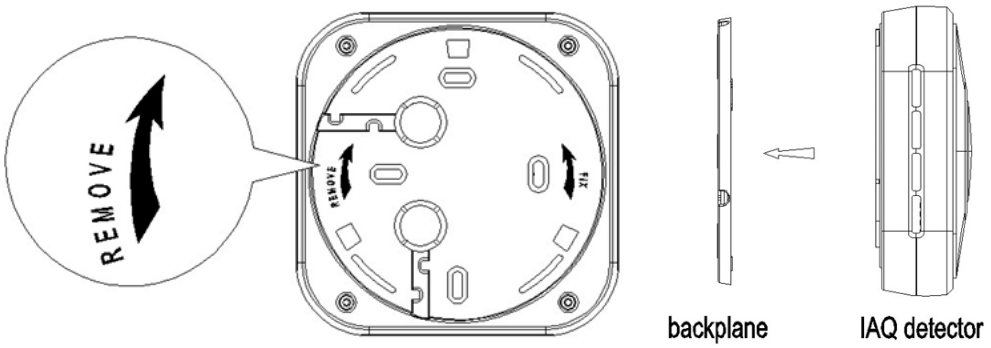

Figure 4. Separate the backboard from the detector

2. Use a screwdriver to punch the threading hole on the backboard and remove the cover of the threading hole.

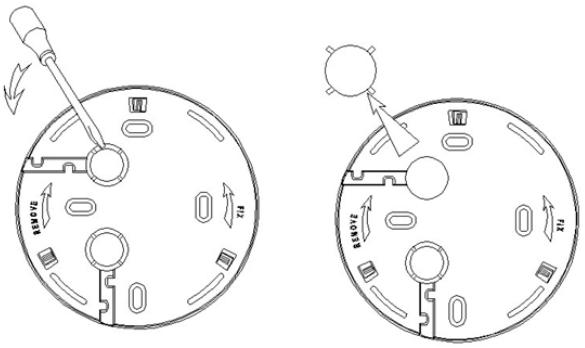

Figure 5. Remove the threading hole cover

3. Pull the cables on the wall through the threading hole.

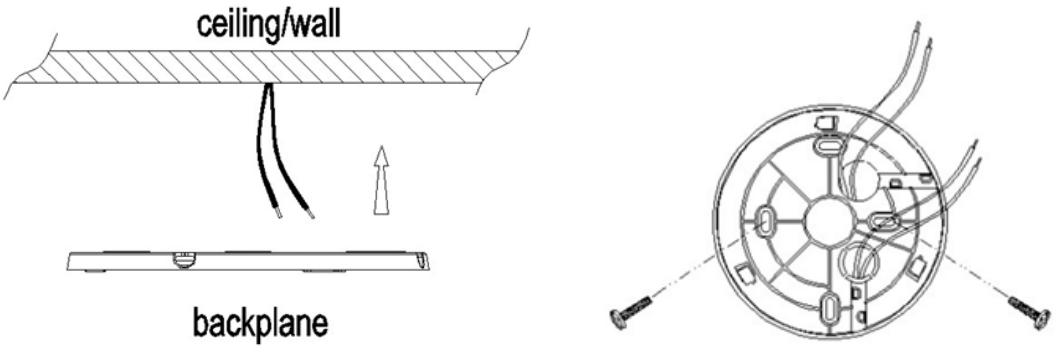

Figure 6. Pull the cables

4. Unplug the terminal block from the contact pin.

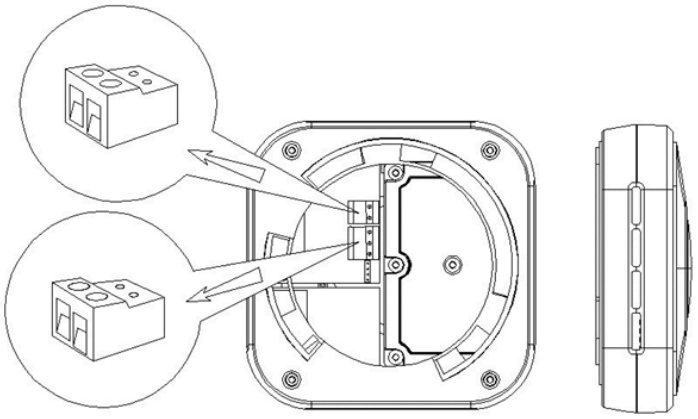

Figure 7. Unplug the terminal block

5. Connect the cable to the terminal block, then tightly lock the mounting screw.

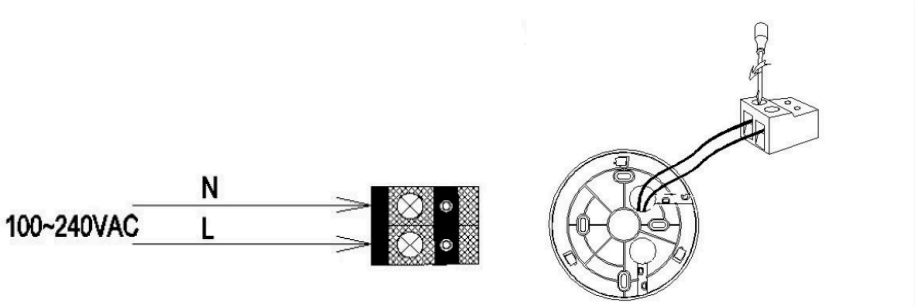

Figure 8. Connect to the terminal block

6. Plug the contacted terminal block back into the contact pin.

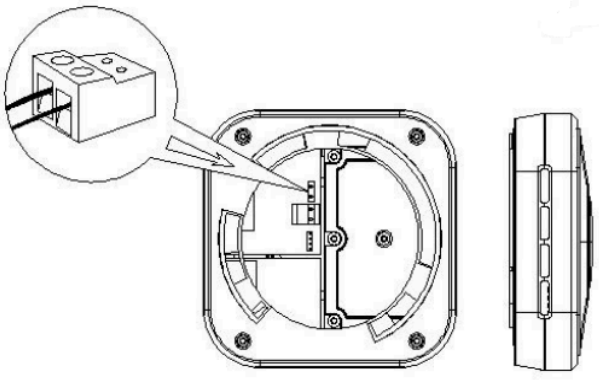

Figure 9. Plug the terminal block to the contact pin

7. Aim the dot located in the middle of two arrows on the side of the detector with the vertical lines on the backboard. Then rotate the detector following the **FIX** direction until it's tight.

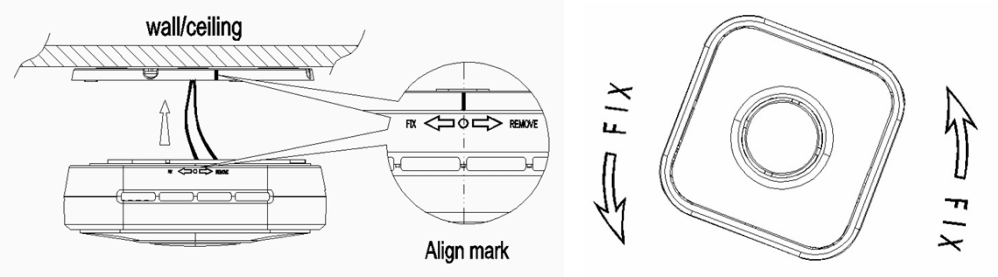

Figure 10. Attach the sensor to the backplane

⚠ WARNING

Installation Guidelines:

- Avoid installing near kitchens, heating units, air conditioners, direct sunlight, or sources of heat and pollutants. This sensor is suitable for ceiling and wall installation only.
- Keep it away from high-power or electrostatic equipment to maintain accuracy. Ensure the location allows for regular maintenance.
- Install after construction or renovation is completed, and the area is cleaned. If renovations are needed, remove the monitor first and reinstall it afterward, or wrap it to protect it from paint and dust.

Usage Instructions:

Prevent damage from drops, impacts, or exposure to high concentrations of volatile organic compounds (VOCs), which can impair sensor accuracy.

Allow the monitor to adapt to temperature changes before powering on. For example, let it sit for 8 hours in a warm environment after receiving it in cold weather, or 2 hours if moving from air-conditioned to non-air-conditioned areas.

Maintenance Tips:

- Avoid painting the sensor's casing to avoid clogging the inlet and outlet.

- Do not use cigarettes for PM2.5 measurement testing, as it may cause inaccuracies. Cigarette particles range from 0.1 to 0.3 microns, resulting in a significant PM2.5 measurement deviation.
- For accurate readings, power the monitor continuously for at least 48 hours after initial use or long periods of inactivity.
- The built-in CO2 sensor can self-calibrate. Initial readings may fluctuate but should stabilize after 2-7 days of continuous operation.

4. Device Configuration

Before configuring your IAQ sensor, make sure you have the recommended OS and browser. The list below only includes the minimum requirements. Any versions lower than the minimum requirement are not recommended for use.

Minimum requirement for Operating Systems (OS):

- Microsoft Windows 10
- Apple macOS 12
- Debian Linux 11
- Ubuntu Linux 22
- Apple iOS/iPadOS 12
- Android 12

Recommended browsers:

- Microsoft Edge 110
- Google Chrome 110
- Firefox 110

4.1. Wi-Fi Configuration

1. Create a hotspot by powering on the monitor. If it does not connect after 90 seconds, the hotspot will be turned off automatically and you need to restart it to reconnect.
2. Open your PC or mobile device's WLAN/Wi-Fi settings and find the network signal that matches the client ID. Connect to the network and

enter the default password.

- Default password: **a1B2c3D4**

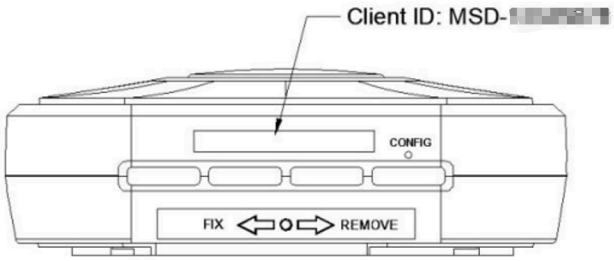

Figure 11. Client ID

NOTE

The Client ID is on the label on the device shell.

- **Client ID: MSD-XXXXXXXX**

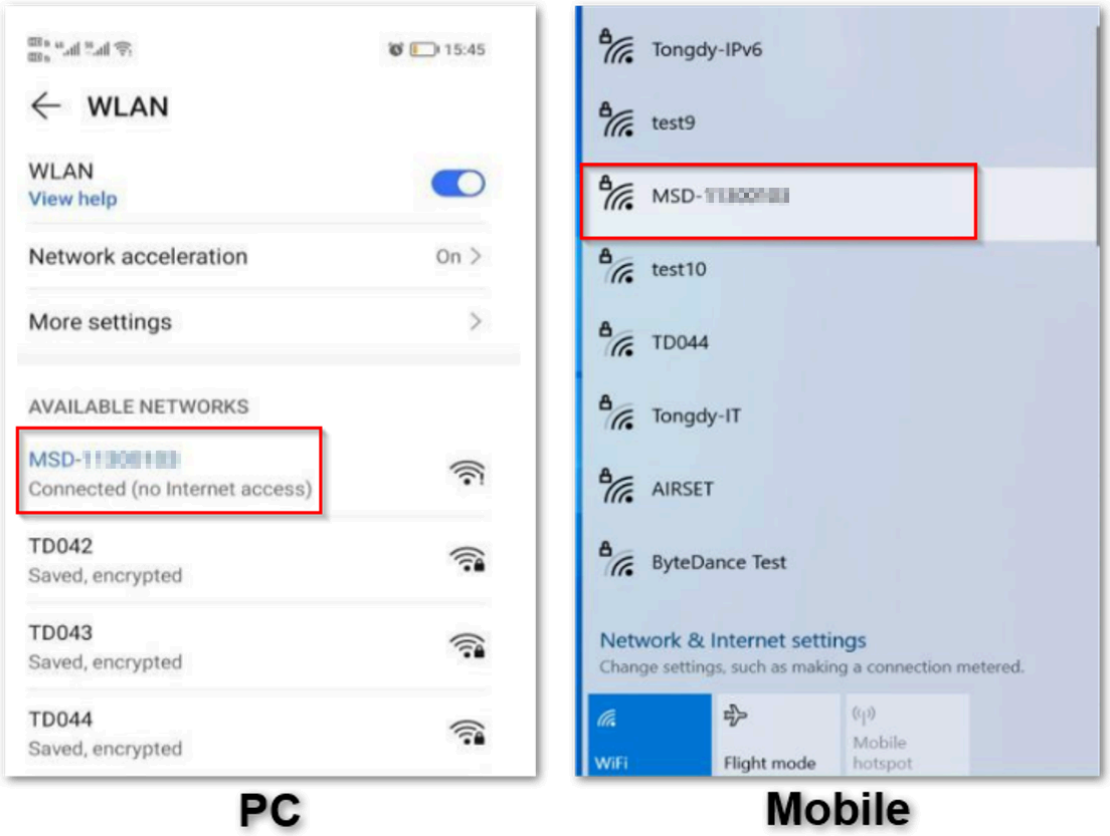

Figure 12. WLAN/Wi-Fi Settings

3. Open a browser and go to *192.168.9.1*. If the browser fails to load the page, check for the following:

- Whether the system has successfully connected to the Wi-Fi network of the device. If the system failed to connect, restart the device and try again.
- Whether it is the recommended system and browser. If yes, try to switch the browser to incognito mode, re-power on the device and try again.

4. Enter the default username and password.
   - Username: **admin**
   - Password: **public**

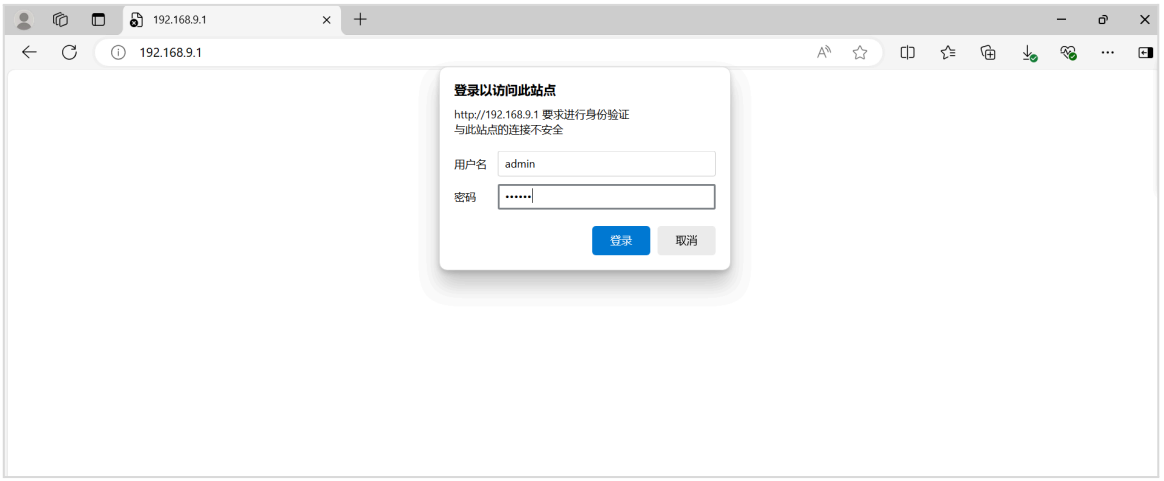

Figure 13. Log in with the default credentials

5. Go to the Configuration page.

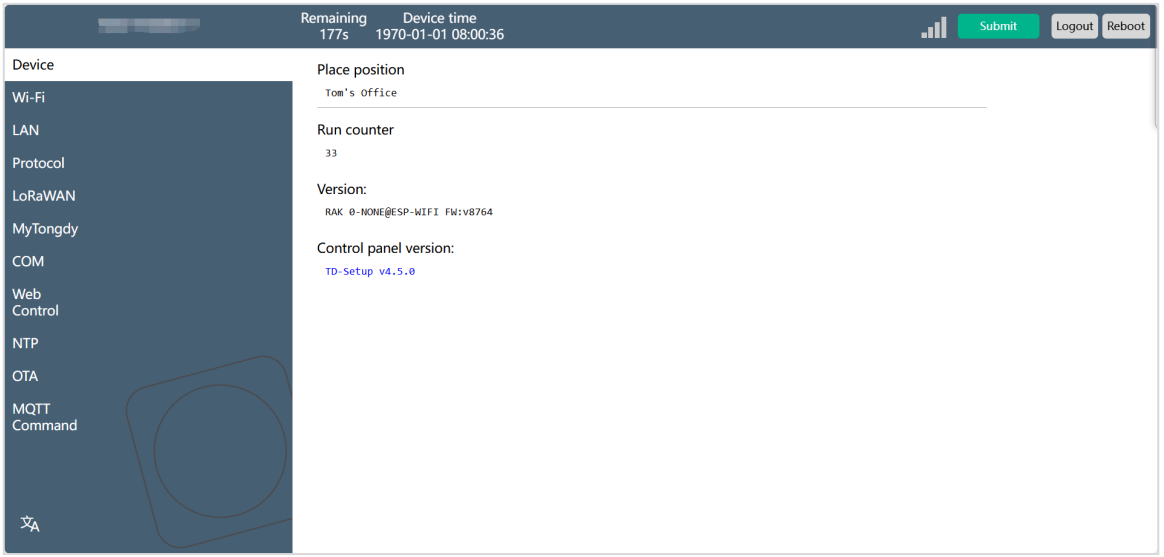

Figure 14. Configuration Page

6. Navigate to the **LoRaWAN** interface, configure and check LoRaWAN connection information, and click the Submit button in the upper right corner to save the settings of the current page.

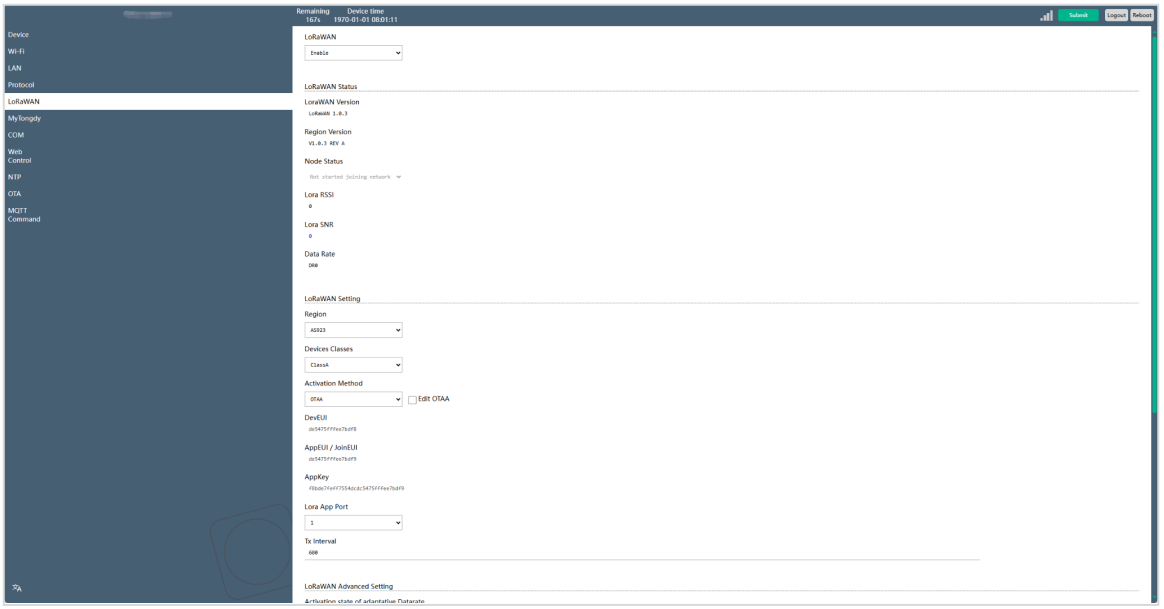

Figure 15. LoRaWAN interface

NOTE

Do not change the DevEUI, AppEUI, JoinEUI, AppKey and MAC address. They are generated by the communication module.

4.2. Data Description

4.2.1. Decoding the Payload

For detailed decoding scripts of this product on TTN and ChirpStack v2, refer to this decoder:

```
function lppDecode(bytes) {
      var sensor_types = {
            0: { 'size': 1, 'name': 'digital_in', 'signed': false,
'divisor': 1 },
            1: { 'size': 1, 'name': 'digital_out', 'signed':
false, 'divisor': 1 },
            2: { 'size': 2, 'name': 'analog_in', 'signed': true,
'divisor': 100 },
            3: { 'size': 2, 'name': 'analog_out', 'signed': true,
'divisor': 100 },
            100: { 'size': 4, 'name': 'generic', 'signed': false,
'divisor': 1 },
```

```
            101: { 'size': 2, 'name': 'illuminance', 'signed':
false, 'divisor': 1 },  //unit:Lux
            102: { 'size': 1, 'name': 'presence', 'signed': false,
'divisor': 1 },
            103: { 'size': 2, 'name': 'temperature', 'signed':
true, 'divisor': 10 },  //unit:℃
            104: { 'size': 1, 'name': 'humidity', 'signed': false,
'divisor': 2 },
            112: { 'size': 2, 'name': 'humidity_prec', 'signed':
true, 'divisor': 10 },  //unit:%RH
            113: { 'size': 6, 'name': 'accelerometer', 'signed':
true, 'divisor': 1000 },
            115: { 'size': 2, 'name': 'barometer', 'signed':
false, 'divisor': 10 },  //unit:hPa
            116: { 'size': 2, 'name': 'voltage', 'signed': false,
'divisor': 100 },
            117: { 'size': 2, 'name': 'current', 'signed': false,
'divisor': 1000 },
            118: { 'size': 4, 'name': 'frequency', 'signed':
false, 'divisor': 1 },
            120: { 'size': 1, 'name': 'percentage', 'signed':
false, 'divisor': 1 },
            121: { 'size': 2, 'name': 'altitude', 'signed': true,
'divisor': 1 },
            125: { 'size': 2, 'name': 'concentration', 'signed':
false, 'divisor': 1 },
            128: { 'size': 2, 'name': 'power', 'signed': false,
'divisor': 1 },
            130: { 'size': 4, 'name': 'distance', 'signed': false,
'divisor': 1000 },
            131: { 'size': 4, 'name': 'energy', 'signed': false,
'divisor': 1000 },
            132: { 'size': 2, 'name': 'direction', 'signed':
false, 'divisor': 1 },
            133: { 'size': 4, 'name': 'time', 'signed': false,
'divisor': 1 },
            134: { 'size': 6, 'name': 'gyrometer', 'signed': true,
'divisor': 100 },
            135: { 'size': 3, 'name': 'colour', 'signed': false,
'divisor': 1 },
            136: { 'size': 9, 'name': 'gps', 'signed': true,
'divisor': [10000, 10000, 100] },
            137: { 'size': 11, 'name': 'gps', 'signed': true,
'divisor': [1000000, 1000000, 100] },
            138: { 'size': 2, 'name': 'voc', 'signed': false,
'divisor': 1 },
            142: { 'size': 1, 'name': 'switch', 'signed': false,
'divisor': 1 },
            188: { 'size': 2, 'name': 'soil_moist', 'signed':
false, 'divisor': 10 },
            190: { 'size': 2, 'name': 'wind_speed', 'signed':
false, 'divisor': 100 },  //unit:m/s
            191: { 'size': 2, 'name': 'wind_direction', 'signed':
false, 'divisor': 1 },  //unit:°
            192: { 'size': 2, 'name': 'soil_ec', 'signed': false,
```

```
'divisor': 1000 },
            193: { 'size': 2, 'name': 'soil_ph_h', 'signed':
false, 'divisor': 100 },
            194: { 'size': 2, 'name': 'soil_ph_l', 'signed':
false, 'divisor': 10 },
            195: { 'size': 2, 'name': 'pyranometer', 'signed':
false, 'divisor': 1 },
            203: { 'size': 1, 'name': 'light', 'signed': false,
'divisor': 1 },

            //Tongdy 20230805
            211: { 'size': 2, 'name': 'co2', 'signed': true,
'divisor': 1 },  //unit:ppm
            212: { 'size': 2, 'name': 'tvoc', 'signed': true,
'divisor': 1000 },  //unit:mg/m3
            213: { 'size': 2, 'name': 'pm0.3', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            214: { 'size': 2, 'name': 'pm0.5', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            215: { 'size': 2, 'name': 'pm1', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            216: { 'size': 2, 'name': 'pm2.5', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            217: { 'size': 2, 'name': 'pm4', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            218: { 'size': 2, 'name': 'pm10', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            219: { 'size': 2, 'name': 'pm100', 'signed': true,
'divisor': 1 },  //unit:ug/m3
            220: { 'size': 2, 'name': 'co', 'signed': true,
'divisor': 10 },  //unit:ppm
            221: { 'size': 2, 'name': 'o3', 'signed': true,
'divisor': 1 },  //unit:ppb
            222: { 'size': 2, 'name': 'so2', 'signed': true,
'divisor': 1 },  //unit:pbb
            223: { 'size': 2, 'name': 'no2', 'signed': true,
'divisor': 1 },  // unit:pbb
            224: { 'size': 2, 'name': 'hcho', 'signed': true,
'divisor': 1000 },  //unit:pbb
            225: { 'size': 2, 'name': 'noise', 'signed': true,
'divisor': 10 },  //unit:dB(A)
      };

      function arrayToDecimal(stream, is_signed, divisor) {

            var value = 0;
            for (var i = 0; i < stream.length; i++) {
                  if (stream[i] > 0xFF)
                        throw 'Byte value overflow!';
                  value = (value << 8) | stream[i];
            }

            if (is_signed) {
                  var edge = 1 << (stream.length) * 8;  //
0x1000..
```

```
                var max = (edge - 1) >> 1;              //
0x0FFF.. >> 1
                value = (value > max) ? value - edge : value;
           }
           value /= divisor;
           return value;
     }
     var sensors = [];
     var i = 0;
     while (i < bytes.length) {

           var s_no = bytes[i++];
           var s_type = bytes[i++];
           if (typeof sensor_types[s_type] == 'undefined') {
                throw 'Sensor type error!: ' + s_type;
           }
           var s_value = 0;
           var type = sensor_types[s_type];
           switch (s_type) {

                case 113:   // Accelerometer
                case 134:   // Gyrometer
                     s_value = {
                          'x': arrayToDecimal(bytes.slice(i + 0,
i + 2), type.signed, type.divisor),
                          'y': arrayToDecimal(bytes.slice(i + 2,
i + 4), type.signed, type.divisor),
                          'z': arrayToDecimal(bytes.slice(i + 4,
i + 6), type.signed, type.divisor)
                     };
                     break;
                case 136:   // GPS Location
                     s_value = {
                          'latitude':
arrayToDecimal(bytes.slice(i + 0, i + 3), type.signed,
type.divisor[0]),
                          'longitude':
arrayToDecimal(bytes.slice(i + 3, i + 6), type.signed,
type.divisor[1]),
                          'altitude':
arrayToDecimal(bytes.slice(i + 6, i + 9), type.signed,
type.divisor[2])
                     };
                     break;
                case 137:   // Precise GPS Location
                     s_value = {
                          'latitude':
arrayToDecimal(bytes.slice(i + 0, i + 4), type.signed,
type.divisor[0]),
                          'longitude':
arrayToDecimal(bytes.slice(i + 4, i + 8), type.signed,
type.divisor[1]),
                          'altitude':
arrayToDecimal(bytes.slice(i + 8, i + 11), type.signed,
type.divisor[2])
```

```
                        };
                        sensors.push({
                              'channel': s_no,
                              'type': s_type,
                              'name': 'location',
                              'value': "(" + s_value.latitude + ","
+ s_value.longitude + ")"
                        });
                        sensors.push({
                              'channel': s_no,
                              'type': s_type,
                              'name': 'altitude',
                              'value': s_value.altitude
                        });
                        break;
                  case 135:   // Colour
                        s_value = {
                              'r': arrayToDecimal(bytes.slice(i + 0,
i + 1), type.signed, type.divisor),
                              'g': arrayToDecimal(bytes.slice(i + 1,
i + 2), type.signed, type.divisor),
                              'b': arrayToDecimal(bytes.slice(i + 2,
i + 3), type.signed, type.divisor)
                        };
                        break;

                  default:    // All the rest
                        s_value = arrayToDecimal(bytes.slice(i, i +
type.size), type.signed, type.divisor);
                        break;
            }
            sensors.push({
                  'channel': s_no,
                  'type': s_type,
                  'name': type.name,
                  'value': s_value
            });
            i += type.size;
      }
      return sensors;
}

// For TTN, Helium and Datacake
function Decoder(bytes, fport) {

      // flat output (like original decoder):
      var response = {};
      lppDecode(bytes, 1).forEach(function (field) {
            response[field['name'] + '_' + field['channel']] =
field['value'];
      });

      return response;
}
```

```
// For Chirpstack V3
// function Decode(fPort, bytes, variables) {

      // flat output (like original decoder):
      // var response = {};
      // lppDecode(bytes, 1).forEach(function (field) {
            // response[field['name'] + '_' + field['channel']] =
field['value'];
      // });
      // return response;
// }

// Chirpstack v3 to v4 compatibility wrapper
// function decodeUplink(input) {
      // return {
            // data: Decode(input.fPort, input.bytes,
input.variables)
      // };
// }


function encodeDownlink(input) {
      return {
            bytes: stringToBytes(JSON.stringify(input.data)),
            fPort: input.fPort,
      }
}


function stringToHex(str) {
      var hex = '';
      for (var i = 0; i < str.length; i++) {
            hex += '' + str.charCodeAt(i).toString(16);
      }
      return hex;
}

function hexToBytes(hex) {
      for (var bytes = [], c = 0; c < hex.length; c += 2)
            bytes.push(parseInt(hex.substr(c, 2), 16));
      return bytes;
}

function stringToBytes(str) {
      return hexToBytes(stringToHex(str));
}

function decodeDownlink(input) {
      return {
            data: {
                  field: "value"
            },
            warnings: ["warning 1", "warning 2"],
            errors: ["error 1", "error 2"]
      }
```

```
}
```

4.2.2. Data Format

The packet data follows the Cayenne LPP packet format, but it contains additional channel IDs that are not included in the default Cayenne LPP specification. This is required because some sensor data does not fit into any of the existing channel IDs.

Table 10: Data Format

SENSOR DATA UNIT	ID (CHANNEL)	TYPE	DATA
Temperature	1 Byte	1 Byte	2 Bytes
Humidity	1 Byte	1 Byte	1 Byte
CO2	1 Byte	1 Byte	2 Bytes
TVOC	1 Byte	1 Byte	2 Bytes
PM 2.5 value	1 Byte	1 Byte	2 Bytes
PM 10 value	1 Byte	1 Byte	2 Bytes

Data Sample 1:

Payload (hex) received data:

036700FA02686B23D301CD10D4010E29D8001A2ADA001F

Table 11. Sensor Data Sample 1

SENSOR DATA UNIT	ID (CHANNEL)	TYPE	DATA
Temperature	03	67	00FA
Humidity	02	68	6B
CO2	23	D3	01CD
TVOC	10	D4	010E
PM 2.5 value	29	D8	001A
PM 10 value	2A	DA	001F

Convert the sensor data from hexadecimal to decimal:

0367 (Temperature) - Data 00FA

$00FA_{16} = 250_{10}$
250 x 0.1 (conversion factor) = **25° C**

0268 (Humidity) - Data 6B

$6B_{16} = 107_{10}$
250 x 0.7 (conversion factor) = **53.5 % RH**

23D3 (CO2) - Data 01CD

$01CD_{16}$ $= 461_{10}$
= **461 ppm**

10D4 (TVOC) - Data 010E

$010E_{16} = 270_{10}$
270 x 0.001 (conversion factor) = **0.27 mg/m^3**

29D8 (PM2.5) - Data 001A

$001A_{16} = 26_{10}$
= **26 ug/m^3**

2ADA (PM10) - Data 001F

$001F_{16} = 31_{10}$
= **31 ug/m^3**