MultiSensor2 Smart Sensor API Documentation

Table of Contents

1. Overview

1.1 Document Description

This document provides the complete API interface specification for the **MultiSensor2 Smart Sensor** device.

Document Conventions:

- Request and response data formats are both JSON.
- Timestamps are Unix timestamps (seconds).
- String-type boolean values use `"1"` for true and `"0"` for false.

1.2 Basic Information

Item	Description
Base URL	`http://{Device IP Address}:{Port}/api/public/index.php`
Default Port	80 (configurable)
Protocol	HTTP
Request Format	JSON (`application/json`)
Response Format	JSON
Character Encoding	UTF-8

1.3 Interface Specification

Base URL:

```
http://{Device IP Address}:{Port}/api/public/index.php
```

Example Base URL: `http://192.168.0.100/api/public/index.php`

Example - Get Device Info - Full URL:
`http://192.168.0.100/api/public/index.php/device/info`

---

2. General Instructions

2.1 Request Headers

Except for the login interface, all other interfaces require a Token in the request headers:

Header Name	Type	Required	Description
Batoken	string	Yes	The token returned after successful login.

Request Header Example:

```
Batoken: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.m/aqOHlQcvfOBjGYrLD9Ir+w1NKCWgN...
```

2.2 Content-Type

Regular Interfaces:

```
Content-Type: application/json
```

File Upload Interfaces:

```
Content-Type: multipart/form-data
```

2.3 Example

```
POST /device/save HTTP/1.1
Host: 192.168.1.100
Content-Type: application/json
batoken: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.m/aqOHlQcvfOBjGYrLD9Ir+w1NKCWgN...

{
    "data": {
        "position": "Office Room 101",
        "manager": "John"
    }
}
```

2.4 Response Format

All interfaces return a unified JSON format:

```
{
    "code": 1,
    "msg": "",
    "data": {},
    "total": 0
}
```

Field	Type	Description
code	int	Business status code, 1=success, others=failure.
msg	string	Prompt message, usually empty on success.
data	object/array	Response data, varies according to the interface.
total	int	Total number of items for paginated list interfaces.

2.5 Status Code Description

Status Code	Constant Name	Description
1	OK	Request successful.
0	ERROR	Request failed (general error).
301	TOKEN_ERROR	Token invalid or expired, need to re-login.
401	UNAUTHORIZED	Unauthorized, authentication failed.
403	NO_PERMISSION	Insufficient permissions.
404	NO_FOUND	Resource not found.
500	SYSTEM_ERROR	System internal error.
501	CHECK_ERROR	Parameter validation error.

2.6 Pagination Parameters

List-type interfaces support the following generic pagination parameters:

Parameter Name	Type	Required	Default	Description
page	int	No	1	Current page number, starting from 1.
limit	int	No	10	Number of items per page.
order	string	No	-	Sorting field, format: `field,asc` or `field,desc`.
quickSearch	string	No	-	Quick search keyword.
search	object	No	-	Advanced search condition object.
customType	string	No	-	Custom type, `noLimit` means no pagination.

Pagination Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": [
        {"id": 1, "name": "item1"},
        {"id": 2, "name": "item2"}
    ],
    "total": 100
}
```

2.7 Error Handling

Token Expired Response:

```
{
    "code": 301,
    "msg": "please login",
    "data": {
        "type": "need login"
    }
}
```

Parameter Error Response:

```
{
    "code": 0,
    "msg": "param error",
    "data": []
}
```

Insufficient Permissions Response:

```
{
    "code": 403,
    "msg": "Permission denied",
    "data": []
}
```

---

3. Interface Details

3.1 Login Authentication Module (Login)

> The Login Authentication Module provides functions such as user login, logout, and getting user information. The login interface does not require authentication; other interfaces require Token authentication.

3.1.1 User Login

User logs into the system and obtains an access token.

Basic Information:

Item	Description
URL	`/login/login`
Method	`POST`
Requires Authentication	No

Request Parameters:

Parameter Name	Type	Required	Description	Example
username	string	Yes	Username	"admin"
password	string	Yes	Password (plain text)	"123456"

Request Example:

```
{
    "username": "admin",
    "password": "123456"
}
```

Response Parameters:

Parameter Name	Type	Description
userInfo	object	User information object
userInfo.id	int	User ID
userInfo.username	string	Username
userInfo.nickname	string	Nickname
userInfo.avatar	string	Avatar URL
userInfo.last_login_time	string	Last login time
userInfo.token	string	Access token (important)
userInfo.refresh_token	string	Refresh token
devInfo	object	Device information object
devInfo.model	string	Device model
devInfo.temperature_type	string	Temperature type (F/C)

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "userInfo": {
            "id": 1,
            "username": "admin",
            "nickname": "admin",
            "avatar": "",
            "last_login_time": "",
            "token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3MDMzMjIwMDAsInR5cGUiOiJyc190b2
tlbiIsInByb2R1Y3QiOiJNUOOiLCJ1c2VybmFtZSI6ImFkbWluIn0.xxxxx",
            "refresh_token": ""
        },
        "devInfo": {
            "model": "SmartSensor",
            "temperature_type": "F"
        }
    }
}
```


Failure Response Example:


```
{
    "code": 0,
    "msg": "password error",
    "data": []
}
```

3.1.2 Get User Information

Get detailed information and menu permissions for the currently logged-in user.

Basic Information:

Item	Description
URL	`/login/info`
Method	`GET`
Requires Authentication	Yes (batoken)

Request Parameters: None

Response Parameters:

Parameter Name	Type	Description
menus	array	Menu permission list
adminInfo	object	Administrator information
adminInfo.id	int	User ID
adminInfo.username	string	Username
adminInfo.nickname	string	Nickname
adminInfo.super	boolean	Whether it is a super administrator
adminInfo.must_change_pwd	boolean	Whether password change is mandatory

Success Response Example:


```
{
    "code": 1,
    "msg": "",
    "data": {
        "menus": [
            {
                "id": 1,
                "name": "dashboard",
                "title": "Dashboard",
                "icon": "fa fa-dashboard",
                "path": "/dashboard",
                "children": []
            }
        ],
        "adminInfo": {
            "id": 1,
            "username": "admin",
            "nickname": "admin",
            "avatar": "",
            "last_login_time": "",
            "super": true,
            "must_change_pwd": false
        }
    }
}
```

3.1.3 User Logout

Log out of the current login session.

Basic Information:

Item	Description
URL	`/login/logout`
Method	`POST`
Requires Authentication	Yes

Request Parameters: None

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": []
}
```

3.1.4 Check System Time

Check and synchronize the system time before login (automatically corrects when device time is abnormal).

Basic Information:

Item	Description
URL	`/login/checkTime`
Method	`POST`
Requires Authentication	No

Request Parameters:

Parameter Name	Type	Required	Description	Example
dateTime	string	Yes	Current time	"2025-12-23 10:30:00"
timezone	string	Yes	Time zone	"GMT-8"

Request Example:

```
{
    "dateTime": "2025-12-23 10:30:00",
    "timezone": "GMT-8"
}
```

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": []
}
```

3.2 Account Management Module (Account)

> The Account Management Module provides CRUD (Create, Read, Update, Delete) functions for user accounts, including changing passwords, adding users, deleting users, etc.

3.2.1 Change Password

Change the password of the currently logged-in user.

Basic Information:

Item	Description
URL	`/account/changePwd`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
old	string	Yes	Old password
pwd	string	Yes	New password
pwd2	string	Yes	Confirm new password (must match `pwd`)

Request Example:

```
{
    "old": "123456",
    "pwd": "newpassword",
    "pwd2": "newpassword"
}
```

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": []
}
```

Failure Response Example:

```
{
    "code": 0,
    "msg": "Confirm Password Error",
    "data": []
}
```

3.2.2 Get Account List

Get a list of all accounts in the system (paginated).

Basic Information:

Item	Description
URL	`/account/getList`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
page	int	No	Page number, default 1
limit	int	No	Items per page, default 10
quickSearch	string	No	Quick search by username
order	string	No	Sort field

Request Example:

```
{
    // Quick search
    "quickSearch": "admin",
    // Current page number
    "page": 1,
    // Items per page
    "limit": 10
}
```

Response Parameters:

Parameter Name	Type	Description
data[].id	int	Account ID
data[].username	string	Username
data[].user_level	string	User level (1=Admin, 2=Normal User)
data[].note	string	Note
total	int	Total record count

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": [
        {
            "id": 1,
            "username": "admin",
            "user_level": "1",
            "note": ""
        },
        {
            "id": 2,
            "username": "viewer",
            "user_level": "2",
            "note": ""
        }
    ],
    "total": 2
}
```

3.2.3 Add Account

Add a new system account.

Basic Information:

Item	Description
URL	`/account/add`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
username	string	Yes	Username (unique)
user_level	string	Yes	User level (1=Admin, 2=Normal User)
pwd	string	No	Password, default is "123456"
pwd2	string	No	Confirm password

Request Example:

```
{
    "username": "user1",
    "user_level": "2",
    "pwd": "mypassword",
    "pwd2": "mypassword"
}
```

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": []
}
```

Failure Response Example:

```
{
    "code": 0,
    "msg": "username already exists",
    "data": []
}
```

3.2.4 Edit Account

Edit an existing account's information.

Basic Information:

Item	Description
URL	`/account/edit`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
id	int	Yes	Account ID
user_level	string	Yes	User level

Request Example:

```
{
    "id": 2,
    "user_level": "1"
}
```

3.2.5 Delete Account

Delete one or more accounts.

Basic Information:

Item	Description
URL	`/account/del`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
ids	array	Yes	Array of account IDs to delete

Request Example:

```
{
    "ids": [2, 3, 4]
}
```

Response Description:
On success, `data` returns an array of deleted usernames.

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": ["user1", "user2", "user3"]
}
```

3.2.6 Get Account Details

Get detailed information for a single account by ID.

Basic Information:

Item	Description
URL	`/account/info`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
id	int	Yes	Account ID

Request Example:

```
{
    "id": 1
}
```

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "id": 1,
        "username": "admin",
        "user_level": "1",
        "note": ""
    }
}
```

3.3 Device Information Module (Device)

> The Device Information Module provides query and setting functions for basic device information.

3.3.1 Get Device Information

Get the device's basic information, including serial number, version, network info, etc.

Basic Information:

Item	Description
URL	`/device/info`
Method	`POST`
Requires Authentication	Yes

Request Parameters: None

Response Parameters:

Parameter Name	Type	Description
sn	string	Device serial number
app_version	string	Web application version
fw_version	string	Firmware version
main_version	string	Main program version
rtsp_version	string	RTSP module version

Parameter Name	Type	Description
mac	string	MAC address
ip	string	IP address
model	string	Device model
position	string	Device location name
manager	string	Administrator
latitude	string	Latitude
longitude	string	Longitude
building	string	Building
floor	string	Floor
room	string	Room
wing	string	Wing
section	string	Section
note	string	Note

Success Response Example:


```
{
    "code": 1,
    "msg": "",
    "data": {
        "sn": "MSM20241223001",
        "app_version": "24.11.01",
        "fw_version": "2.0.0",
        "main_version": "1.5.0",
        "rtsp_version": "1.0.0",
        "mac": "00:11:22:33:44:55",
        "ip": "192.168.1.100",
        "model": "SmartSensor",
        "position": "Office Room 101",
        "manager": "John",
        "latitude": "31.2304",
        "longitude": "121.4737",
        "building": "Building A",
        "floor": "3F",
        "room": "301",
        "wing": "",
        "section": "",
        "note": "Main entrance sensor"
    }
}
```

3.3.2 Save Device Information

Save the device's location and description information.

Basic Information:

Item	Description
URL	`/device/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
data	object	Yes	Device information object
data.position	string	No	Device location name
data.manager	string	No	Administrator
data.latitude	string	No	Latitude
data.longitude	string	No	Longitude
data.building	string	No	Building
data.floor	string	No	Floor
data.room	string	No	Room
data.wing	string	No	Wing
data.section	string	No	Section
data.note	string	No	Note

Request Example:

```
{
    "data": {
        "position": "Conference Room B",
        "manager": "Jane",
        "building": "Building B",
        "floor": "5F",
        "room": "502",
        "note": "Updated location"
    }
}
```

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": []
}
```

3.4 Device Configuration Module (DevCfg)

> The Device Configuration Module provides advanced parameter configuration, restart, factory reset, and other functions for the device.

3.4.1 Get Device Configuration

Get the device's advanced configuration parameters.

Basic Information:

Item	Description
URL	`/devCfg/info`
Method	`POST`
Requires Authentication	Yes

Request Parameters: None

Response Parameters:

Parameter Name	Type	Description
data_post_switch	string	Data forwarding switch (0/1)
data_post_url	string	Data forwarding URL
vape_algorithm	string	Vape algorithm switch (0/1)
temperature_type	string	Temperature type (F=Fahrenheit/C=Celsius)
ehi_calc	string	EHI calculation algorithm (1: Maximum Value, 2: Average Value)
http_port	int	HTTP port
onvif_port	int	ONVIF port
rtsp_port	int	RTSP port

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "data_post_switch": "0",
        "data_post_url": "",
        "ehi_calc": "1",
        "vape_algorithm": "0",
        "http_port": 80,
        "onvif_port": 80,
        "rtsp_port": 554
    }
}
```

3.4.2 Save Device Configuration

Save the device's advanced configuration parameters.

Basic Information:

Item	Description
URL	`/devCfg/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
data	object	Yes	Configuration object, fields same as response parameters

Request Example:


```
{
    "data": {
        "data_post_switch": "0",
        "data_post_url": "",
        "ehi_calc": "1",
        "vape_algorithm": "0",
        "http_port": 80,
        "onvif_port": 80,
        "rtsp_port": 554
    }
}
```

3.4.3 Device Reboot

Reboot the device.

Basic Information:

Item	Description
URL	`/devCfg/reboot`
Method	`POST`
Requires Authentication	Yes

Request Parameters: None

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": []
}
```

> **Note**: After calling this interface, the device will reboot and the connection will be lost.

3.4.4 Factory Reset

Restore the device to factory default settings.

Basic Information:

Item	Description
URL	`/devCfg/factory`
Method	`POST`
Requires Authentication	Yes

Request Parameters: None

> **Warning**: This operation will clear all configuration data, and the device will reboot.

3.5 Network Configuration Module (Network)

> The Network Configuration Module provides configuration functions for wired and WiFi networks.

3.5.1 Get Wired Network Configuration

Get Ethernet (eth0) configuration information.

Basic Information:

Item	Description
URL	`/network/getIp`
Method	`POST`
Requires Authentication	Yes

Request Parameters: None

Response Parameters:

Parameter Name	Type	Description
dhcp_switch	string	DHCP switch (1=On, 0=Off)
ipv4_addr	string	IPv4 address
ipv4_mask	string	Subnet mask
ipv4_gateway	string	Default gateway
mac	string	MAC address
dns1	string	Primary DNS server
dns2	string	Secondary DNS server

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "dhcp_switch": "1",
        "ipv4_addr": "192.168.1.100",
        "ipv4_mask": "255.255.255.0",
        "ipv4_gateway": "192.168.1.1",
        "mac": "00:11:22:33:44:55",
        "dns1": "8.8.8.8",
        "dns2": "8.8.4.4"
    }
}
```

3.5.2 Set Wired Network Configuration

Set the Ethernet (eth0) configuration.

Basic Information:

Item	Description
URL	`/network/setIp`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
dhcp_switch	string	Yes	DHCP switch (1=On, 0=Off)
ipv4_addr	string	Conditional	IP address (required when DHCP is off)
ipv4_mask	string	Conditional	Subnet mask (required when DHCP is off)
ipv4_gateway	string	Conditional	Gateway (required when DHCP is off)
dns1	string	No	Primary DNS server
dns2	string	No	Secondary DNS server

Request Example (Static IP):

```
{
    "dhcp_switch": "0",
    "ipv4_addr": "192.168.1.100",
    "ipv4_mask": "255.255.255.0",
    "ipv4_gateway": "192.168.1.1",
    "dns1": "8.8.8.8",
    "dns2": "8.8.4.4"
}
```

Request Example (DHCP):

```
{
    "dhcp_switch": "1"
}
```

3.5.3 Get WiFi Configuration

Get WiFi network configuration information.

Basic Information:

Item	Description
URL	`/network/getWifi`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
wifi_switch	string	WiFi switch (1=On, 0=Off)
conn_status	string	Connection status (COMPLETED=Connected)
ssid	string	WiFi name (SSID)
password	string	WiFi password
bssid	string	Access point MAC
key_mgmt	string	Encryption method (WPA-PSK/NONE)
ip_address	string	IP address
mac_address	string	WiFi NIC MAC address
gateway	string	Gateway

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "wifi_switch": "1",
        "conn_status": "COMPLETED",
        "ssid": "MyWiFi",
        "password": "mypassword",
        "bssid": "",
        "key_mgmt": "WPA-PSK",
        "ip_address": "192.168.1.150",
        "mac_address": "AA:BB:CC:DD:EE:FF",
        "gateway": "192.168.1.1"
    }
}
```

3.5.4 Set WiFi Configuration

Set WiFi network configuration.

Basic Information:

Item	Description
URL	`/network/setWifi`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
wifi_switch	string	Yes	WiFi switch (1=On, 0=Off)
ssid	string	Conditional	WiFi name (SSID) (required when WiFi is on)
password	string	No	WiFi password (empty for open networks)

Request Example:

```
{
    "wifi_switch": "1",
    "ssid": "OfficeWiFi",
    "password": "office123"
}
```

3.5.5 Get WiFi Connection Status

Get the current WiFi connection status.

Basic Information:

Item	Description
URL	`/network/getWifiConnStatus`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
status	string	Connection status

Status Value Description:

- `0`: Connected
- `1`: Scanning/Connecting
- `9`: Connection Failed

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "status": "0"
    }
}
```

3.6 Time Configuration Module (Ntp)

> The Time Configuration Module provides NTP synchronization and manual time setting functions.

3.6.1 Get NTP Configuration

Get the current NTP and system time configuration.

Basic Information:

Item	Description
URL	`/ntp/info`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
ntp_switch	string	NTP synchronization switch (1=On, 0=Off)
ntp_address	string	NTP server address
time_zone	string	Time zone
dst_switch	string	Daylight Saving Time switch
system_time	string	Current system time

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "ntp_switch": "1",
        "ntp_address": "pool.ntp.org",
        "time_zone": "GMT-8",
        "dst_switch": "0",
        "system_time": "2025-12-23 10:30:00"
    }
}
```

3.6.2 Save NTP Configuration

Save NTP and time configuration.

Basic Information:

Item	Description
URL	`/ntp/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
ntp_switch	string	Yes	NTP synchronization switch
ntp_address	string	Conditional	NTP server address (required when NTP is on)
time_zone	string	Yes	Time zone
dst_switch	string	No	Daylight Saving Time switch
system_time	string	No	Manually set time (available when NTP is off)

Request Example (NTP Sync):


```
{
    "ntp_switch": "1",
    "ntp_address": "pool.ntp.org",
    "time_zone": "America/New_York",
    "dst_switch": "1"
}
```


Request Example (Manual Time):

```
{
    "ntp_switch": "0",
    "time_zone": "GMT-8",
    "system_time": "2025-12-23 15:30:00"
}
```

3.7 Event Configuration Module (Event)

> The Event Configuration Module is used to manage sensor event trigger rules, including thresholds, intervals, etc.

3.7.1 Get Event List

Get a list of all event configurations.

Basic Information:

Item	Description
URL	`/event/getList`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
data[].id	int	Event ID
data[].identifier	string	Event identifier (unique)
data[].data_source	string	Data source type
data[].threshold	string	Trigger threshold
data[].pause	string	Release threshold
data[].trigger_time	int	Last trigger timestamp
data[].interval	string	Trigger interval (seconds)
data[].advanced_json	string	Advanced configuration JSON
data[].sort_num	int	Sort number
total	int	Total record count

Success Response Example:


```
{
    "code": 1,
    "msg": "",
```

```
    "data": [
        {
            "id": 1,
            "identifier": "vape_detected",
            "data_source": "vape",
            "threshold": "1",
            "pause": "0",
            "trigger_time": 0,
            "interval": "60",
            "advanced_json": "{}",
            "sort_num": 1
        },
        {
            "id": 2,
            "identifier": "high_temperature",
            "data_source": "temperature",
            "threshold": "35",
            "pause": "0",
            "trigger_time": 1703318400,
            "interval": "120",
            "advanced_json": "{}",
            "sort_num": 2
        }
    ],
    "total": 10
}
```

3.7.2 Save Event Configuration

Save a single event configuration.

Basic Information:

Item	Description
URL	`/event/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
id	int	No	Event ID
identifier	string	Yes	Event identifier (must be unique)
data_source	string	Yes	Data source type
threshold	string	No	Trigger threshold
pause	string	No	Release threshold
interval	string	No	Trigger interval (seconds)

Request Example:

```
{
    "id": 3,
    "identifier": "co2_high",
    "data_source": "co2",
    "threshold": "1200",
    "pause": "0",
    "interval": "300"
}
```

3.7.3 Save Advanced Configuration

Save advanced configuration for an event (JSON format).

Basic Information:

Item	Description
URL	`/event/saveAdvanced`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
id	int	Yes	Event ID
data	object	Yes	Advanced configuration object

Request Example:

```
{
    "id": 3,
    "data": {
        "alert_weeks": ["Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"],
        "alert_times": {
            "from_hour": "06:30:00",
            "to_hour": "19:00:00"
        }
    }
}
```

3.7.5 Save All Events

Save all event configurations in batch (deletes existing then inserts new).

Basic Information:

Item	Description
URL	`/event/saveAll`
Method	`POST`
Requires Authentication	Yes

Request Parameters: Array of event configurations

Request Example:

```
[
    {
        "id": 1,
        "identifier": "vape_detected",
        "data_source": "vape",
        "threshold": "1",
        "pause": "0",
        "interval": "60"
    },
    {
        "id": 2,
        "identifier": "high_temp",
        "data_source": "temperature",
        "threshold": "35",
        "pause": "0",
        "interval": "120"
    }
]
```

3.7.6 Restore Default Events

Restore event configurations to factory defaults.

Basic Information:

Item	Description
URL	`/event/setDefault`
Method	`POST`
Requires Authentication	Yes

3.8 Action Configuration Module (Action)

> The Action Configuration Module is used to set response actions after event triggers, such as sending emails, controlling LEDs, playing audio, etc.

3.8.1 Get Action List

Get all action configurations associated with events.

Basic Information:

Item	Description
URL	`/action/getList`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
data[].event_id	int	Associated event ID
data[].identifier	string	Event identifier
data[].data_source	string	Data source
data[].id	int	Action configuration ID
data[].smtp_set	string	Send email on trigger (1=Yes)
data[].smtp_reset	string	Send email on release (1=Yes)
data[].delay	string	Delay time (seconds)
data[].led_color	string	LED color
data[].led_pattern	string	LED pattern
data[].led_priority	string	LED priority
data[].audio_id	string	Audio file path

Parameter Name	Type	Description
data[].http_set	string	Send HTTP on trigger
data[].tcp_set	string	Send TCP on trigger
data[].mqtt_set	string	Send MQTT on trigger
data[].http_reset	string	Send HTTP on release
data[].tcp_reset	string	Send TCP on release
data[].mqtt_reset	string	Send MQTT on release

3.8.2 Get Action Resources

Get available LED colors, patterns, and audio resource lists.

Basic Information:

Item	Description
URL	`/action/getResource`
Method	`POST`
Requires Authentication	Yes

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "ledColor": [
            {"name": "Red", "val": "red"},
            {"name": "Green", "val": "green"},
            {"name": "Blue", "val": "blue"},
            {"name": "Yellow", "val": "yellow"}
        ],
        "ledPattern": [
            {"name": "Solid", "val": "solid"},
            {"name": "Flash", "val": "flash"},
            {"name": "Pulse", "val": "pulse"}
        ],
        "audio": [
            {"name": "Vape_Detected.wav", "val":
"/path/audio/Vape_Detected.wav"},
            {"name": "Emergency.wav", "val": "/path/audio/Emergency.wav"},
            {"name": "custom_alert.wav", "val":
"/path/audio_custom/custom_alert.wav"}
        ]
    }
}
```

3.8.3 Save Action Configuration

Save a single action configuration.

Basic Information:

Item	Description
URL	`/action/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
event_id	int	Yes	Associated event ID
id	int	No	Action ID (required for edit)
smtp_set	string	No	Send email on trigger
smtp_reset	string	No	Send email on release
delay	string	No	Delay time (seconds)
led_color	string	No	LED color
led_pattern	string	No	LED pattern
led_priority	string	No	LED priority

Parameter Name	Type	Required	Description
audio_id	string	No	Audio file path
http_set	string	No	Send HTTP on trigger
tcp_set	string	No	Send TCP on trigger
mqtt_set	string	No	Send MQTT on trigger
http_reset	string	No	Send HTTP on release
tcp_reset	string	No	Send TCP on release
mqtt_reset	string	No	Send MQTT on release

3.8.4 Test Action

Test the effect of an action configuration (trigger or release).

Basic Information:

Item	Description
URL	`/action/test`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
type	string	Yes	Test type (set=trigger, reset=release)
data	object	Yes	Action configuration data
data.event_id	int	Yes	Event ID
data.identifier	string	Yes	Event identifier

Request Example:

```
{
    "type": "set",
    "data": {
        "event_id": 1,
        "identifier": "vape_detected",
        "http_set": "1",
        "smtp_set": "1",
        "led_color": "red",
        "led_pattern": "flash"
    }
}
```

3.8.5 Save All Actions

Save all action configurations in batch.

Basic Information:

Item	Description
URL	`/action/saveAll`
Method	`POST`
Requires Authentication	Yes

3.8.6 Restore Default Actions

Restore action configurations to defaults.

Basic Information:

Item	Description
URL	`/action/setDefault`
Method	`POST`
Requires Authentication	Yes

3.9 Platform Integration Configuration Module (Setup)

> The Platform Integration Configuration Module is used to configure third-party platform integration parameters such as HTTP, TCP, MQTT, SMTP, etc.

3.9.1 Get Platform Configuration

Get the corresponding platform configuration based on type.

Basic Information:

Item	Description
URL	`/setup/info`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
type	string	Yes	Configuration type (see table below)

Configuration Type Description:

Type Value	Description
tcpAddress	TCP server configuration
httpAddress	HTTP server configuration
mqttAddress	MQTT server configuration
rtspCfg	RTSP video stream configuration
heartbeat	Heartbeat configuration
smtpCfg	SMTP mail server configuration

TCP Configuration Response:


```
{
    "code": 1,
    "msg": "",
    "data": {
        "host": "192.168.1.200",
        "port": "8080",
        "set_content": "{\"event\":\"${identifier}\",\"value\":\"${value}\"}",
        "reset_content": "{\"event\":\"${identifier}\",\"status\":\"clear\"}",
        "set_is_on": "1",
        "reset_is_on": "0",
        "status": "1",
        "set_lang": "json",
        "reset_lang": "json"
    }
}
```


HTTP Configuration Response:


```
{
    "code": 1,
    "msg": "",
    "data": {
        "server": "https://api.example.com/webhook",
        "method": "POST",
        "content_type": "application/json",
        "set_content": "{\"type\":\"alert\",\"event\":\"%EID@TRIGGER%\"}",
        "reset_content": "{\"type\":\"clear\",\"event\":\"%EID@TRIGGER%\"}",
        "set_is_on": "1",
        "reset_is_on": "1",
        "status": "1",
        "set_lang": "json",
        "reset_lang": "json",
        "auth_method": "private",
        "auth_key": "aaa",
        "auth_value": "bbb",
        "auth_token_url": ""
```

```
    }
}
```

MQTT Configuration Response:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "server": "mqtt.example.com",
        "port": "1883",
        "client_id": "sensor_001",
        "user": "mqttuser",
        "password": "mqttpass",
        "topic": "sensors/alerts",
        "qos": "1",
        "set_content": "{\"event\":\"${identifier}\"}",
        "reset_content": "{\"event\":\"${identifier}\",\"clear\":true}",
        "set_is_on": "1",
        "reset_is_on": "1",
        "status": "1",
        "set_lang": "json",
        "reset_lang": "json"
    }
}
```

SMTP Configuration Response:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "host": "smtp.gmail.com",
        "security": "tls",
        "user": "sender@gmail.com",
        "password": "app_password",
        "port": "587",
        "sender": "sender@gmail.com",
        "recipients": "admin@example.com,security@example.com",
        "status": "1",
        "provider": "gmail",
        "client_id": "",
        "client_secret": "",
        "tenant_id": ""
    }
}
```

RTSP Configuration Response:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "show_dev_name": "1",
        "show_datetime": "1",
        "show_logo": "0",
        "logo": "",
        "framerate": "15",
        "screen_width": "1920",
        "screen_height": "1080"
    }
}
```


Heartbeat Configuration Response:


```
{
    "type": "heartbeat",
    "data": {
        "status": "1",
        "protocol": "http",
        "interval": "60",
        "content": "{\"sn\": \"%DEVICE%SN%\"}",
        "lang": "json",
        "http_conf": {
            "server": "http://192.168.0.150:7000",
            "method": "post",
            "content_type": "application/json",
            "user": "",
            "auth_key": "",
            "auth_method": "",
            "auth_value": "",
            "password": ""
        },
        "mqtt_conf": {
            "host": "",
            "port": "",
            "user": "",
            "password": "",
            "qos": "1",
            "client_id": "",
            "topic": ""
        },
        "tcp_conf": {
            "host": "",
            "port": ""
        }
    }
}
```

3.9.2 Save Platform Configuration

Save platform integration configuration.

Basic Information:

Item	Description
URL	`/setup/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
type	string	Yes	Configuration type
data	object	Yes	Configuration data object

Request Examples:

```
{
    "type": "httpAddress",
    "data": {
        "server": "https://api.example.com/webhook",
        "method": "POST",
        "content_type": "application/json",
        "set_content": "{\"type\":\"alert\",\"event\":\"%EID@TRIGGER%\"}",
        "reset_content": "{\"type\":\"clear\",\"event\":\"%EID@TRIGGER%\"}",
        "set_is_on": "1",
        "reset_is_on": "1",
        "status": "1",
        "set_lang": "json",
        "reset_lang": "json",
        "auth_method": "private",
        "auth_key": "aaa",
        "auth_value": "bbb",
        "auth_token_url": ""
    }
}
```

```
{
    "type": "tcpAddress",
    "data": {
        "host": "192.168.1.200",
        "port": "8080",
        "set_content": "{\"event\":\"${identifier}\",\"value\":\"${value}\"}",
        "reset_content": "{\"event\":\"${identifier}\",\"status\":\"clear\"}",
        "set_is_on": "1",
        "reset_is_on": "0",
        "status": "1",
```

```
        "set_lang": "json",
        "reset_lang": "json"
    }
}
```

```
{
    "type": "mqttAddress",
    "data": {
        "server": "mqtt.example.com",
        "port": "1883",
        "client_id": "sensor_001",
        "user": "mqttuser",
        "password": "mqttpass",
        "topic": "sensors/alerts",
        "qos": "1",
        "set_content": "{\"event\":\"${identifier}\"}",
        "reset_content": "{\"event\":\"${identifier}\",\"clear\":true}",
        "set_is_on": "1",
        "reset_is_on": "1",
        "status": "1",
        "set_lang": "json",
        "reset_lang": "json"
    }
}
```

```
{
    "type": "smtpCfg",
    "data": {
        "host": "smtp.gmail.com",
        "security": "tls",
        "user": "sender@gmail.com",
        "password": "app_password",
        "port": "587",
        "sender": "sender@gmail.com",
        "recipients": "admin@example.com,security@example.com",
        "status": "1",
        "provider": "gmail",
        "client_id": "",
        "client_secret": "",
        "tenant_id": ""
    }
}
```

```
{
    "type": "rtspCfg",
    "data": {
        "show_dev_name": "1",
        "show_datetime": "1",
        "show_logo": "0",
        "logo": "",
        "framerate": "15",
        "screen_width": "1920",
        "screen_height": "1080"
    }
}
```



```
{
    "type": "heartbeat",
    "data": {
        "status": "1",
        "protocol": "http",
        "interval": "60",
        "content": "{\"sn\": \"%DEVICE%SN%\"}",
        "lang": "json",
        "http_conf": {
            "server": "http://192.168.0.150:7000",
            "method": "post",
            "content_type": "application/json",
            "user": "",
            "auth_key": "",
            "auth_method": "",
            "auth_value": "",
            "password": ""
        },
        "mqtt_conf": {
            "host": "",
            "port": "",
            "user": "",
            "password": "",
            "qos": "1",
            "client_id": "",
            "topic": ""
        },
        "tcp_conf": {
            "host": "",
            "port": ""
        }
    }
}
```

3.10 Platform Connection Test Module (Platform)

> The Platform Connection Test Module is used to test connections for various platform integrations.

3.10.1 HTTP Connection Test

Test HTTP server connection.

Basic Information:

Item	Description
URL	`/platform/httpTest`
Method	`POST`
Requires Authentication	Yes

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "status": "success",
        "response_code": 200
    }
}
```

3.10.2 SMTP Connection Test

Test SMTP mail server connection and send a test email.

Basic Information:

Item	Description
URL	`/platform/smtpTest`
Method	`POST`
Requires Authentication	Yes

3.10.3 TCP Connection Test

Test TCP server connection.

Basic Information:

Item	Description
URL	`/platform/tcpTest`
Method	`POST`
Requires Authentication	Yes

3.10.4 MQTT Connection Test

Test MQTT server connection.

Basic Information:

Item	Description
URL	`/platform/mqttTest`
Method	`POST`
Requires Authentication	Yes

3.10.5 Heartbeat Connection Test

Test heartbeat service connection.

Basic Information:

Item	Description
URL	`/platform/heartTest`
Method	`POST`
Requires Authentication	Yes

3.11 Firmware Upgrade Module (Upgrade)

> The Firmware Upgrade Module provides firmware upload and upgrade functionality.

3.11.1 Get Upgrade Status

Check if the device can currently be upgraded.

Basic Information:

Item	Description
URL	`/upgrade/getStatus`
Method	`POST`
Requires Authentication	Yes

3.11.2 Upload Firmware

Upload a firmware file to the device.

Basic Information:

Item	Description
URL	`/upgrade/upload`
Method	`POST`
Requires Authentication	Yes
Content-Type	multipart/form-data

Request Parameters:

Parameter Name	Type	Required	Description
file	File	Yes	Firmware file (.bin format, max 20MB)

3.11.3 Get Firmware List

Get information about uploaded firmware files.

Basic Information:

Item	Description
URL	`/upgrade/getList`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
data[].id	int	Record ID
data[].version	string	Firmware version number
data[].md5	string	File MD5 value
data[].username	string	Upload user
data[].status	string	Upgrade status
data[].uptime	string	Upload time
data[].note	string	Note (original filename)

Status Description:

- `-1`: Not upgraded
- `0`: Upgrade successful
- `1`: Upgrading
- `9`: Upgrade failed

3.11.4 Execute Upgrade

Start firmware upgrade execution.

Basic Information:

Item	Description
URL	`/upgrade/upgrade`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
md5	string	Yes	MD5 value of the firmware to upgrade

3.11.5 Get Upgrade Progress

Poll for current upgrade progress.

Basic Information:

Item	Description
URL	`/upgrade/getStep`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
md5	string	Yes	Firmware MD5 value

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "status": "1",
        "progress": 75
    }
}
```

3.12 Audio Resource Module (Audio)

> The Audio Resource Module provides functions for uploading, deleting, and playing test audio files.

3.12.1 Play Test Audio

Play test audio.

Basic Information:

Item	Description
URL	`/audio/play`
Method	`POST`
Requires Authentication	Yes

3.12.2 Stop Audio Playback

Stop currently playing audio.

Basic Information:

Item	Description
URL	`/audio/stop`
Method	`POST`
Requires Authentication	Yes

3.12.3 GPIO High Level Test

Set GPIO to high level (test relay, etc.).

Basic Information:

Item	Description
URL	`/audio/upGpio`
Method	`POST`
Requires Authentication	Yes

3.12.4 GPIO Low Level Test

Set GPIO to low level.

Basic Information:

Item	Description
URL	`/audio/downGpio`
Method	`POST`
Requires Authentication	Yes

3.12.5 Upload Audio File

Upload custom audio file.

Basic Information:

Item	Description
URL	`/audio/upload`
Method	`POST`

Item	Description
Requires Authentication	Yes
Content-Type	multipart/form-data

Request Parameters:

Parameter Name	Type	Required	Description
file	File	Yes	Audio file

File Restrictions:

- Format: `.wav`, `.mp3`
- Size: Max 10MB
- Filename: Only numbers, letters, underscores allowed

3.12.6 Get Audio List

Get a list of audio files.

Basic Information:

Item	Description
URL	`/audio/getList`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
type	string	Yes	Type (default=system audio, custom=custom audio)
page	int	No	Page number
size	int	No	Items per page

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": [
        {
            "filename": "custom_alert.wav",
            "uptime": "2025-12-23 10:00:00"
        }
    ],
    "total": 5
}
```

3.12.7 Delete Audio File

Delete custom audio file(s).

Basic Information:

Item	Description
URL	`/audio/del`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
ids	array	Yes	Array of filenames

Request Example:

```
{
    "ids": ["custom1.wav", "custom2.wav"]
}
```

3.13 LED Resource Module (Led)

> The LED Resource Module provides management functions for LED colors and patterns.

3.13.1 Get LED Status

Get current LED status.

Basic Information:

Item	Description
URL	`/led/get`
Method	`POST`
Requires Authentication	Yes

3.13.2 Set LED

Manually set LED color and pattern.

Basic Information:

Item	Description
URL	`/led/set`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
color	string	Yes	LED color
pattern	string	Yes	LED pattern

3.13.3 Get LED Resource List

Get LED color/pattern/priority resource lists.

Basic Information:

Item	Description
URL	`/led/getList`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
type	string	Yes	Type (color/pattern/priority)
page	int	No	Page number
limit	int	No	Items per page

3.13.4 Set LED Resource Status

Enable or disable specific LED colors/patterns.

Basic Information:

Item	Description
URL	`/led/setStatus`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
type	string	Yes	Resource type
ids	array	Yes	Array of resource IDs
status	string	Yes	Status (1=Enable, 0=Disable)

3.14 RTSP Resource Module (Rtsp)

> The RTSP Resource Module is used to manage RTSP video stream logos and background images.

3.14.1 Get RTSP Resource List

Get uploaded RTSP Logo/background resources.

Basic Information:

Item	Description
URL	`/rtsp/info`
Method	`POST`
Requires Authentication	Yes

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": [
        {
            "uid": "logo_001",
            "type": "logo",
            "filename": "logo.png",
            "full_filename": "/path/to/logo.png",
            "base64": "..."
        }
    ]
}
```

3.14.2 Save RTSP Resource

Upload RTSP Logo or background image.

Basic Information:

Item	Description
URL	`/rtsp/save`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
uid	string	Yes	Unique identifier
type	string	Yes	Type (logo/bg)
base64	string	Yes	Base64 encoded image data (includes data header)
filename	string	No	Filename

Supported Image Formats: jpg, jpeg, png, gif, bmp, webp

3.14.3 Delete RTSP Resource

Delete an RTSP resource.

Basic Information:

Item	Description
URL	`/rtsp/del`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
uid	string	Yes	Resource unique identifier

3.15 Data Preview Module (Preview)

> The Data Preview Module provides query functions for sensor real-time data and historical chart data.

3.15.1 Get Real-time Data

Get the latest sensor real-time data.

Basic Information:

Item	Description
URL	`/preview/getData`
Method	`POST`
Requires Authentication	Yes

Response Parameters:

Parameter Name	Type	Description
pm_1	string	PM1.0 (μg/m³)
pm_25	string	PM2.5 (μg/m³)
pm_10	string	PM10 (μg/m³)
tvoc	string	TVOC (ppb)
co2	string	CO2 (ppm)
co	string	CO (ppm)
no2	string	NO2 (ppb)
nh3	string	NH3 (ppm)
hcho	string	HCHO (ppb)

Parameter Name	Type	Description
hcho	string	HCHO (ppb)
temperature	string	Temperature (°C)
temperature_f	string	Temperature (°F)
humidity	string	Humidity (%)
aqi	string	Air Quality Index
person_detection	string	1=Someone, 0=No one
motion	string	Motion detection

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "pm_1": "25.5",
        "pm_25": "35.2",
        "pm_10": "48.7",
        "tvoc": "120",
        "co2": "856",
        "co": "2.1",
        "no2": "18",
        "nh3": "5",
        "hcho": "45",
        "temperature": "24.6",
        "temperature_f": "76.3",
        "humidity": "55.2",
        "aqi": "68",
        "person_detection": "3",
        "motion": "1"
    }
}
```

3.15.2 Get Chart Data

Get sensor historical chart data (last 30 minutes).

Basic Information:

Item	Description
URL	`/preview/getChart`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description

Parameter Name	Type	Required	Description
type	array	Yes	Array of data types

Request Example:

```
{
    "type": ["pm_25", "tvoc", "co2"]
}
```

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "xData": ["10:00", "10:01", "10:02", "10:03", "..."],
        "yData": {
            "pm_25": [25, 30, 28, 32, "..."],
            "tvoc": [100, 95, 110, 105, "..."],
            "co2": [800, 850, 830, 820, "..."]
        }
    }
}
```

3.15.3 Get Raw Chart Data

Get chart data in raw format.

Basic Information:

Item	Description
URL	`/preview/getRawChart`
Method	`POST`
Requires Authentication	Yes

Success Response Example

```
{
    "code": 1,
    "msg": "",
    "data": [
        ["type", "value", "time"],
        ["pm_1", 40, "10:14"],
        ["pm_25", 66, "10:14"],
        ["pm_10", 80, "10:14"],
        ["tvoc", 13, "10:14"],
        ["co2", 572, "10:14"],
        ["no2", 0, "10:14"],
        ["co", 0, "10:14"],
        ["hcho", 13, "10:14"],
```

```
        ["temperature", 26.3, "10:14"],
        ["humidity", 62.9, "10:14"],
        ["noise_ratio", 64, "10:14"],
        ["person_detection", 0, "10:14"],
        ["aqi", 29, "10:14"],
        ["nh3", 4, "10:14"],
        ["temperature_f", 79.3, "10:14"]
        .......
    ]
}
```

3.15.4 Get Weekly Average Data

Get daily average data for the past week.

Basic Information:

Item	Description
URL	`/preview/getWeekAvg`
Method	`POST`
Requires Authentication	Yes

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": {
        "yList": ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"],
        "xList": {
            "pm_25": [32.5, 28.3, 30.1, 35.2, 29.8, 25.6, 27.4],
            "tvoc": [98, 105, 92, 88, 110, 85, 90],
            "temperature": [24.5, 25.1, 24.8, 25.3, 24.2, 23.8, 24.0],
            "humidity": [55, 58, 52, 60, 57, 54, 56]
        }
    }
}
```

3.15.5 Get Event Trigger Status

Get a list of currently triggered event data sources.

Basic Information:

Item	Description
URL	`/preview/getEventTrigger`
Method	`POST`
Requires Authentication	Yes

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": ["vape", "motion", "co2"]
}
```

3.16 Data List Module (DataList)

3.16.1 Get Sensor Data List

Get historical sensor data list (paginated).

Basic Information:

Item	Description
URL	`/dataList/getList`
Method	`POST`
Requires Authentication	Yes

Request Parameters: Supports generic pagination parameters

Success Response Example:

```
{
    "code": 1,
    "msg": "",
    "data": [
        {
            "id": 1,
            "pm_1": "25",
            "pm_25": "35",
            "pm_10": "50",
            "tvoc": "100",
            "co2": "800",
            "temperature": "25.5",
            "temperature_f": "77.9",
            "humidity": "60",
            "date": "2025-12-23",
            "time": "10:30:00"
        }
    ],
    "total": 1000
}
```

3.17 Event Log Module (EventLog)

3.17.1 Get Event Log List

Get event trigger history logs (paginated).

Basic Information:

Item	Description
URL	`/eventLog/getList`
Method	`POST`
Requires Authentication	Yes

Request Parameters: Supports generic pagination parameters

Response Parameters:

Parameter Name	Type	Description
data[].ctime	int	Trigger timestamp
data[].identifier	string	Event identifier
data[].type	string	Type (trigger=trigger, release=release)
data[].val	string	Value at trigger time
data[].thr	string	Threshold
data[].note	string	Note
data[].date	string	Date
data[].time	string	Time

Success Response Example:

```
```

```
{
    "code": 1,
    "msg": "",
    "data": [
        {
            "ctime": 1703318400,
            "identifier": "vape_detected",
            "type": "trigger",
            "val": "1",
            "thr": "1",
            "note": "",
            "date": "2025-12-23",
            "time": "10:30:00"
        },
        {
            "ctime": 1703318460,
            "identifier": "vape_detected",
            "type": "release",
            "val": "0",
            "thr": "1",
            "note": "",
            "date": "2025-12-23",
            "time": "10:31:00"
        }
    ],
    "total": 150
}
```


3.18 Device Test Module (Test)

3.18.1 Fan Switch Test

Test fan switch functionality.

Basic Information:

Item	Description
URL	`/test/setFan`
Method	`POST`
Requires Authentication	Yes

Request Parameters:

Parameter Name	Type	Required	Description
value	string	Yes	1=On, 0=Off

Request Example:

```
{
    "value": "1"
}
```

4. Appendices

4.1 Data Source Type Description

Data Source	Description	Unit
vape	Vape detection	-
motion	Motion detection	-
aggression	Aggression/fight detection	-
pm_1	PM1.0	μg/m³
pm_25	PM2.5	μg/m³
pm_10	PM10	μg/m³
tvoc	Total Volatile Organic Compounds	ppb
co2	Carbon Dioxide	ppm
co	Carbon Monoxide	ppm
no2	Nitrogen Dioxide	ppb
nh3	Ammonia	ppm
hcho	Formaldehyde	ppb
etoh	Ethanol	ppm
temperature	Temperature (Celsius)	°C
temperature_f	Temperature (Fahrenheit)	°F
humidity	Relative Humidity	%
aqi	Air Quality Index	-
noise_ratio	Noise ratio	-

4.2 LED Color and Mode Options

LED Colors:

Color Value	Description
red	Red
green	Green
blue	Blue
yellow	Yellow
purple	Purple
cyan	Cyan
orange	Orange
white	White

4.3 Time Zone List

The following two time zone formats are supported:

UTC Offset Format:

- `UTC-12` to `UTC+12`

Geographic Time Zone Format (supports DST):

- `US`
- `Canada`

4.4 Version History

Version	Date	Author	Description
1.0.0	2025-12-27	-	Initial version, includes complete API documentation

---

End of Document

If you have any questions, please contact technical support.