Webhook Events Services Specifications

VERSION 1.0

Contents

1 Introduction

1.1 Introduction

This document provides technical information on the events webhooks that can be called by third-parties to send events to ecobook. The events available are:

1. Occupancy sensor events
2. Indoor Air Quality (IAQ) events

1.2 Authentication

1. All requests made to the events webhooks require a bearer token.
2. The bearer token is specific to a particular customer account.
3. When registering for a set of events, the administrator shall generate a bearer token. This is done on the administrative panel.
4. The token is highly confidential and must be stored in a secured location. Compromise of the token can result in third-parties accessing the events webhooks.

1.3 IP Address Restrictions

1. For security purposes, IP address can be provided to whitelist the calling source.

2 WebHooks

2.1 Occupancy Sensor

These webhooks allows devices to notify ecobook when occupancy status changes in the device. This URL must only be called when there is a change in status.

2.1.1 Ecobook setup

1. Each sensor should have a device_name which uniquely identifies the sensor.
2. The device_name must be associated with a specific space (meeting room or desk) in ecobook.
3. The device_name shall be registered in ecobook by the administrator.

2.1.2 Data Format

The following is the data format expected when calling the events webhook. The data will be in Json format. The following are the fields that must be sent:

#	Field Name	Data Type	Remarks
1.	device_name	String	The name of the device. This should match exactly with what was setup by the administrator. Should be in lower case letters preferably without spaces.
2.	occupied	Boolean	Must be either true or false. If null is sent, the system will default to false.
3.	count	Integer	Must be an integer. If null, then zero is the default.

Example:

```
{
    "device_name":"sensor_device_1",
    "occupied":true
}
```

2.1.3 Configuration

#	Parameter	Values
1.	URL	https://events-uat.facilitiesbooking.com/occupancy_sensor/status.aspx
2.	Method Type	POST
3.	BODY	Json formatted data.
4.	Authorization	Bearer: <<token>>
5.	Response	Will be "OK" or "FAIL".
6.	URL to check results	https://events-uat.facilitiesbooking.com/occupancy_sensor/results.aspx

2.2 IAQ Sensor

These webhooks allows devices to notify ecobook when IAQ sensor status changes in the device. This URL must only be called when there is a change in status.

2.2.1 Ecobook setup

4. Each sensor should have a device_name which uniquely identifies the sensor.

5. The device_name may be associated with a specific space (meeting room or desk) in ecobook.
6. The device_name shall be registered in ecobook by the administrator.

2.2.2 Data Format

The following is the data format expected when calling the events webhook. The data will be in Json format. The following are the fields that must be sent:

#	Field Name	Data Type	Remarks
1.	device_name	String	The name of the device. This should match exactly with what was setup by the administrator. Should be in lower case letters preferably without spaces.
2.	virus_index	Numeric(18,2)	Virus index.
3.	temperature	Numeric(18,2)	Temperature.
4.	humidity	Numeric(18,2)	Humidity.
5.	pm1	Numeric(18,2)	Particulate matter with diameter of 1.0 mm.
6.	pm25	Numeric(18,2)	Particulate matter with diameter of 2.5 mm.
7.	pm4	Numeric(18,2)	Particulate matter with diameter of 4.0 mm.
8.	pm10	Numeric(18,2)	Particulate matter with diameter of 10.0 mm.
9.	tvoc	Numeric(18,2)	Total Volatile Organic Compounds.
10.	co2	Numeric(18,2)	Carbon Di-Oxide.
11.	co	Numeric(18,2)	Carbon Monoxide.
12.	pressure	Numeric(18,2)	Pressure.
13.	ozone	Numeric(18,2)	Ozone.
14.	no2	Numeric(18,2)	Nitrogen dioxide.
15.	light	Numeric(18,2)	Light.
16.	sound	Numeric(18,2)	Sound.
17.	h2s	Numeric(18,2)	Hydrogen sulphide.
18.	nh3	Numeric(18,2)	Ammonia.
19.	no	Numeric(18,2)	Nitric oxide.
20.	so2	Numeric(18,2)	Sulphur dioxide.
21.	o2	Numeric(18,2)	Oxygen.
22.	hcho	Numeric(18,2)	Formaldehyde.

Example:

```
{
  "device_name": "iaq_sensor_1",
  "virus_index": 1.25,
```

```
  "temperature": 2.37,
  "humidity": 3.45,
  "pm1": 4.32,
  "pm25": 5.67,
  "pm4": 6.54,
  "pm10": 7.67,
  "tvoc": 8.88,
  "co2": 9.0,
  "co": 10.0,
  "pressure": 11.0,
  "ozone": 12.0,
  "no2": 13.0,
  "light": 14.0,
  "sound": 15.0,
  "h2s": 16.0,
  "nh3": 17.0,
  "no": 18.0,
  "so2": 19.0,
  "o2": 20.0,
  "hcho": 21.0
}
```


2.2.3 Configuration

#	Parameter	Values
1.	URL	https://events-uat.facilitiesbooking.com/iaq_sensor/status.aspx
2.	Method Type	POST
3.	BODY	Json formatted data.
4.	Authorization	Bearer: <<token>>
5.	Response	Will be “OK” or “FAIL”.
6.	URL to check results	https://events-uat.facilitiesbooking.com/iaq_sensor/results.aspx

3 Web API

The web API is used to get the data from the system for data processing, analytics, and business rules processing.

3.1 URL Scheme

Base Url (UAT): https://api-uat.facilitiesbooking.com/api/

3.2 Response Format

1) All responses shall have the standard format:

{

“type”:””,

“title”:””,

“status”:””,

"message": "",

“traceId”:””,

"**data**": [

]

},

“rows_per_page”:0,

“total_rows”:0

}

2) The response JSON contains the details of the response. This is shown below:

#	Parameter	Details
1.	Type	A reference to the RFC document indicating the definition of the status.
2.	Title	A general error messages. Empty if no error. List of messages are: - `One or more validation errors have occured.` - `One or more permission errors have occured.` - `The resource was not found.` - `A server error has occured.`
3.	Status	The status code of the response. Corresponds to the HTTP Status Codes indicated below.
4.	Message	The request specific message.

5.	traceId	A unique string corresponding to this request. If supplied by the caller as part of the query string, it is sent back. If not supplied by the caller, a unique GUID will be generated. More details provided below.
6.	Data	The response data for the query. If errors are present, then this parameter will not be sent or will be empty.
7.	Rows_per_page	Indicates the total number of rows returned from the page.
8.	Total_rows	Indicates the total rows available in the query.

3.3 HTTP Status Codes

#	Response Code	Details
1.	200	API processed successfully. *This is sent when the API has successfully processed the request.* https://www.rfc-editor.org/rfc/rfc7231#section-6.3.1
2.	400	Bad Request. *This is sent when the request is invalid.* https://www.rfc-editor.org/rfc/rfc7231#section-6.5.1
3.	401	Unauthorized. *This is sent when the token has expired.* https://www.rfc-editor.org/rfc/rfc7235#section-3.1
4.	403	Forbidden. *This is sent when a transaction is not allowed.* https://www.rfc-editor.org/rfc/rfc7231#section-6.5.3
5.	404	Not Found.

		This is sent when the requested record is not found. `https://www.rfc-editor.org/rfc/rfc7231#section-6.5.4`
6.	500	Internal Error. *This is sent when the API has an internal error.* `https://www.rfc-editor.org/rfc/rfc7231#section-6.6.1`
7.	429	Too Many Requests. *This is sent when the rate limit has been exceeded.* `https://www.rfc-editor.org/rfc/rfc6585#section-4`

3.4 Actions

All APIs will have a common set of method calls. These are indicated below:

#	HTTP Methods	Details
1.	GET	Gets one or more records.
2.	POST	Creates a record.
3.	PUT	Updates a record.
4.	DELETE	Deletes a record.

3.5 Trace

All queries can have a query string with a "traceId" parameter.

Example:

Base_url/categories?traceId= 96bf2762-bf9b-4446-a389-b50872365981

The TraceId can be in any string format. The traceId is reflected back in the response body to allow applications to map the request and response.

3.6 Occupancy Methods

3.6.1 Object Schema

Parameter	Value
occupancy_status_id	Numeric integer greater than zero. Auto-generated. Read only.
created_on	The date and time when the record was created in UTC time. Format is in yyyy-MM-dd HH:mm:ss. System generated. Read only.
occupied	Indicates status of the sensor. True – occupied. False – Unoccupied.
count	The number of people detected by the sensor.
device_id	The internal ID of the device by ecobook.
asset_id	The internal ID of the meeting room/desk by ecobook.
Device_name	The name of the device.

3.6.2 Get Data By Device Name

The following API call allows the calling party to retrieve the data based on the device_name for a set of date range period:

Action	GET

URL	**baseurl**/ Occupancy/device_name/[device_name]?from=[from]&to=[to]&row_count=[row_count]&page_no=[page_no]
Input Parameters	[device_name] – the name of the device. [from] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [to] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [row_count] – the number of rows returned per page. Maximum of 50. [page_no] – the page number to return.
Input Body	None
Validations	• If [device_name] is empty - 'id' is invalid. (Bad Request: 400) • If [device_name] does not exist - 'id' not found. (Not Found: 404) • If [to] is empty or [from] is empty – 'to' is invalid. 'from' is invalid. (Bad Request: 400) • If [to] earlier than [from] – 'to' earlier than 'from' (Bad Request: 400)

Response


```
{
 "errors": "",
 "title": "",
 "traceId": "e0ed33c0-29ba-4f2e-8d8d-ea199189d9c2",
 "type": "https://www.rfc-editor.org/rfc/rfc7231#section-6.3.1",
 "status": 200,
 "data": [
  {
   "occupancy_status_id": 1,
   "device_name": "sensor_device_1",
   "occupied": true,
   "count": 0,
   "device_id": 114,
   "asset_id": 1,
   "created_on": "2025-04-16T12:54:31"
  }
 ],
 "rows_per_page": 50,
```

```
  "total_rows": 1
}
```


3.6.3 Get Data By Asset ID

The following API call allows the calling party to retrieve the data based on the asset_id for a set of date range period:

Action	**GET**
URL	**baseurl**/ Occupancy/asset/[asset_id]?from=[from]&to=[to]&row_count=[row_count]&page_no=[page_no]
Input Parameters	[asset_id] – the Id of the space (meeting room or desk). [from] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [to] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [row_count] – the number of rows returned per page. Maximum of 50. [page_no] – the page number to return.
Input Body	None
Validations	• If [asset_id] is empty - 'id' is invalid. (Bad Request: 400) • If [asset_id] does not exist - 'id' not found. (Not Found: 404) • If [to] is empty or [from] is empty – 'to' is invalid. 'from' is invalid. (Bad Request: 400) • If [to] earlier than [from] – 'to' earlier than 'from' (Bad Request: 400)
Response	


```
{
  "errors": "",
  "title": "",
  "traceId": "e0ed33c0-29ba-4f2e-8d8d-ea199189d9c2",
  "type": "https://www.rfc-editor.org/rfc/rfc7231#section-6.3.1",
  "status": 200,
  "data": [
```

```
  {
   "occupancy_status_id": 1,
   "device_name": "sensor_device_1",
   "occupied": true,
   "count": 0,
   "device_id": 114,
   "asset_id": 1,
   "created_on": "2025-04-16T12:54:31"
  }
 ],
 "rows_per_page": 50,
 "total_rows": 1
}
```


3.7 IAQ Methods

3.7.1 Object Schema

Parameter	Value
iaq_status_id	Numeric integer greater than zero. Auto-generated. Read only.
created_on	The date and time when the record was created in UTC time. Format is in yyyy-MM-dd HH:mm:ss. System generated. Read only.
device_id	The internal ID of the device by ecobook.
asset_id	The internal ID of the meeting room/desk by ecobook.
Device_name	The name of the device.
virus_index	Virus index.
temperature	Temperature.

humidity	Humidity.
pm1	Particulate matter with diameter of 1.0 mm.
pm25	Particulate matter with diameter of 2.5 mm.
pm4	Particulate matter with diameter of 4.0 mm.
pm10	Particulate matter with diameter of 10.0 mm.
tvoc	Total Volatile Organic Compounds.
co2	Carbon Di-Oxide.
co	Carbon Monoxide.
pressure	Pressure.
ozone	Ozone.
no2	Nitrogen dioxide.
light	Light.
sound	Sound.
h2s	Hydrogen sulphide.
nh3	Ammonia.
no	Nitric oxide.
so2	Sulphur dioxide.
o2	Oxygen.

hcho	Formaldehyde.

3.7.2 Get Data By Device Name

The following API call allows the calling party to retrieve the data based on the device_name for a set of date range period:

Action	**GET**
URL	**baseurl**/ iaq/device_name/[device_name]?from=[from]&to=[to]&row_count=[row_count]&page_no=[page_no]
Input Parameters	[device_name] – the name of the device. [from] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [to] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [row_count] – the number of rows returned per page. Maximum of 50. [page_no] – the page number to return.
Input Body	None
Validations	• If [device_name] is empty - 'id' is invalid. (Bad Request: 400) • If [device_name] does not exist - 'id' not found. (Not Found: 404) • If [to] is empty or [from] is empty – 'to' is invalid. 'from' is invalid. (Bad Request: 400) • If [to] earlier than [from] – 'to' earlier than 'from' (Bad Request: 400)
Response	
{ "errors": "", "title": "", "traceId": "063b33db-5dc9-4a67-ae61-c6e5156ab428", "type": "https://www.rfc-editor.org/rfc/rfc7231#section-6.3.1", "status": 200, "data": [	

```
  {
    "iaq_status_id": 5858,
    "device_name": "iaq_sensor_1",
    "device_id": 115,
    "asset_id": 1,
    "created_on": "2025-04-16T14:08:16",
    "virus_index": 1.25,
    "temperature": 2.37,
    "humidity": 3.45,
    "pm1": 4.32,
    "pm25": 5.67,
    "pm4": 6.54,
    "pm10": 7.67,
    "tvoc": 8.88,
    "co2": 9,
    "co": 10,
    "pressure": 11,
    "ozone": 12,
    "no2": 13,
    "light": 14,
    "sound": 15,
    "h2s": 16,
    "nh3": 17,
    "no": 18,
    "so2": 19,
    "o2": 20,
    "hcho": 21
  }
 ],
 "rows_per_page": 50,
 "total_rows": 1
}
```


3.7.3 Get Data By Asset ID

The following API call allows the calling party to retrieve the data based on the asset_id for a set of date range period:

Action	**GET**

URL	**baseurl**/ iaq /asset/[asset_id]?from=[from]&to=[to]&row_count=[row_count]&page_no=[page_no]
Input Parameters	[asset_id] – the Id of the space (meeting room or desk). [from] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [to] – the timestamp in UTC in yyyy-MM-dd HH:mm:ss format. [row_count] – the number of rows returned per page. Maximum of 50. [page_no] – the page number to return.
Input Body	None
Validations	• If [asset_id] is empty - 'id' is invalid. (Bad Request: 400) • If [asset_id] does not exist - 'id' not found. (Not Found: 404) • If [to] is empty or [from] is empty – 'to' is invalid. 'from' is invalid. (Bad Request: 400) • If [to] earlier than [from] – 'to' earlier than 'from' (Bad Request: 400)

Response


```
{
 "errors": "",
 "title": "",
 "traceId": "063b33db-5dc9-4a67-ae61-c6e5156ab428",
 "type": "https://www.rfc-editor.org/rfc/rfc7231#section-6.3.1",
 "status": 200,
 "data": [
  {
   "iaq_status_id": 5858,
   "device_name": "iaq_sensor_1",
   "device_id": 115,
   "asset_id": 1,
   "created_on": "2025-04-16T14:08:16",
   "virus_index": 1.25,
   "temperature": 2.37,
   "humidity": 3.45,
   "pm1": 4.32,
   "pm25": 5.67,
```

```
    "pm4": 6.54,
    "pm10": 7.67,
    "tvoc": 8.88,
    "co2": 9,
    "co": 10,
    "pressure": 11,
    "ozone": 12,
    "no2": 13,
    "light": 14,
    "sound": 15,
    "h2s": 16,
    "nh3": 17,
    "no": 18,
    "so2": 19,
    "o2": 20,
    "hcho": 21
   }
  ],
  "rows_per_page": 50,
  "total_rows": 1
}
```


4 Version

Version	Date	Details
1.0	17-Apr-2025	Initial release of the specifications.