**Sniffer-Mind RESTful Services**

**API Documentations**

25.12.2015

**Revision History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Revision | Date | Team | Author | Description |
| 1 | Aug 20, 2016 | ATOM | VNHUNG | Create the REST API for the application |
| 2 | Aug 25, 2016 | ATOM | Kiet | REST API for SnifferResources for POST method |
| 3 | Aug 26, 2016 | ATOM | Khanh | GET method for Sniffer REST API |
| 4 | Sep 3, 2016 | ATOM | VNHUNG | GET method for Sniffer, POST method for Pollutant Item |
| 5 | Sep 04, 2016 | ATOM | DANGNH | PUT method for Sniffer |
| 6 | Sep 05, 2016 | ATOM | VNHUNG | Update GET method for Sniffer, add Period’s information |
| 7 | Sept 22, 2016 | ATOM | MUONNV | Update the URI of rest api |
| 8 | Oct 15, 2016 | ATOM | Khanh | Update the Bom attribute |
| 9 | Oct 18, 2016 | ATOM | Khanh | Update the URI for rest api |
| 10 | Oct 20, 2016 | ATOM | DANGNH | Update Vocabularies, REST Resources and Request Templates |
| 11 | Oct 21, 2016 | ATOM | VNHUNG | Add more sample about POST PollutantValues Data in json format |
| 12 | Oct 21, 2016 | ATOM | DANGNH | Add sample request data in XML format for POST PollutantValuePacket |
| 13 | Oct 28, 2016 | ATOM | DANGNH | Add error messages, sample request for search sniffer, pollutant and pollutantvalue |
| 14 | Oct 31, 2016 | ATOM | DANGNH | Express query parameters for search sniffer, pollutant and pollutantvalue in more detail |
| 15 | Nov 03, 2016 | ATOM | DANGNH | Change validation in GroupedPollutantValues and macAddress in Sniffer, correct mistakes in Request Templates |
| 16 | Nov 15, 2016 | ATOM | VNHUNG | Add GPSLocation Class |
| 17 | Dec 27, 2017 | ATOM | Kiet | Modify validation constraints for sniffer’s address |
| 18 | Feb 21, 2017 | ATOM | Kiet | Modify API URL to adapt HTTPS protocol |
| 19 | March 27, 2017 | ATOM | Khanh | Re-Correct the request’s body in Json and XML format examples which have the wrong type of gpsLocation. Change the type of latitude and longitude from String to Float |
| 20 | July 31, 2017 | ATOM | HoangNPB | GET method for AQI  Add sample query parameters for search AQI |
| 21 | March 15, 2018 | ATOM | Phong | Add get token API and apply apply token for:  - Sniffer API (Add and Update sniffer info, Change status of sniffer)  - Pollutant API (Add, Update, Delete pollutant info) |

# **Vocabulary**

## **Address**

Contains address information of a person, an organization or a sniffer

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| id | Integer | The technical identify of the address. It’s for internal use only. | Technical id. Client side does not need to care about it |
| objectId | Integer | The identifier of an object that the address is belong to. Currently, it can be address of a person, an organization or a sniffer. | Optional |
| objectKind | ObjectKind | The type of object that the address belongs to. Currently, it can be a person, an organization or a member. | Optional. Possible values: PERSON, ORGANISATION, SNIFFER |
| country | String | The name of country. | Mandatory, Size from 1 to 50 |
| province | String | The name of province. | Mandatory, Size from 1 to 50 |
| district | String | The name of district. | Optional, Size from 1 to 50 |
| ward | String | The name of ward. | Optional, Size from 1 to 50 |
| street | String | The name of street. | Optional, Size from 1 to 50 |
| houseNr | String | The house number | Optional, Size from 1 to 50 |
| gpsLocation | GPSLocation | The longitude and latitude values of address | Mandatory |
| validFrom | Date | The effectively starting date of the sniffer. | Optional |
| validTo | Date | The effectively ending date of the sniffer. | Optional |

*Table 1 Attributes of Address*

## **Sniffer**

Contains data of a sniffer

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| id | Integer | The technical identify of the sniffer. It’s for internal use only. | Technical id. Client side does not need to care about it |
| code | String | The generated or assigned code to the sniffer, must be unique in the system. | Mandatory, Unique. The value is not used in case creating a new sniffer. |
| name | String | The user-defined name for the sniffer. If the name is empty, sniffer's code will be used to display on the GUI. | Optional |
| address | Address | Address information of the sniffer. | Mandatory |
| macAddress | String | Mac address of the sniffer. It's re-defined by sniffer provider and cannot change by the sniffer keeper. | Mandatory, Unique |
| status | SnifferStatus | The current status that the keeper is working on. | Possible values: INACTIVE, VERIFYING, OPERATING, MAINTAINING, STRIKED, HIDDEN |

Table 2 Attributes of Sniffer

## **Pollutant**

Represents a substance that pollutes something, especially the atmosphere (PM2.5, PM10, O3,...) in the sniffer system.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| code | String | The chemical code of pollutant, must be unique in the system (PM2.5, PM10…) | Mandatory, Size from 1 to 10. |
| name | String | The chemical, science name of the pollutant | Mandatory, Size from 1 to 20 |
| description | String | A short description to describe about the pollutant. | Optional |
| defaultUnit | String | The unit will be used by default for values of the pollutant. for example: mg/m3, ppmv… | Mandatory, Size from 1 to 25. |

*Table 3 Attributes of Pollutant*

## **ValueAt**

Contains measured value of pollutant at a specific time

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| value | Float | The value of the pollutant. | Mandatory |
| measuredAt | Date | The time when pollutant is measured. | Optional. Current time will be used in case it’s not given |

*Table 4 Attributes of ValueAt*

## **PollutantValue**

Contains measured pollutant of a sniffer

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| code | String | The chemical code of the pollutant such as PM2.3, PM10, O3... | Mandatory, Size from 1 to 10 |
| value | ValueAt | The value of the pollutant at a specific time | Mandatory |
| sensor | String | The name of sensor | Optional |

*Table 5 Attributes of PollutantValue*

## **Source**

Represent summary information of a sniffer. it's used when sending measured pollutant values from a sniffer.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| sourceId | String | The sniffer's code. | Mandatory. Size from 1 to 10 |
| macAddress | String | The mac address of a sniffer | Mandatory |

Table 6 Attributes of Source

## **PollutantValuePacket**

It encapsulates pollutant’s values (PM2.5, PM10, Ozone,…) transmitted to server from a sniffer.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| source | Source | The source of the packet. it contains information about the sniffer in which values is sent | Mandatory |
| values | List<PollutantValue> | The list of pollutant values measured from the sniffer | Mandatory |

Table 7 Attributes of PollutantValuePacket

## **GroupedPollutantValues**

Group of PollutantValues

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| code | String | The chemical code of pollutants: such as: O3, O2 , PM10,… |  |
| sensor | String | The name of the sensor |  |
| values | List <ValueAt> | A list of pollutant item values in the packet at a specific time |  |

*Table 8 Attributes of GroupedPollutantValues*

## **GPSLocation**

GPSLocation of an Address

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute name | Type | Description | Validation |
| latitude | Float | The latitude value of GPS Coordinate. |  |
| longitude | Float | The longitude value of GPS Coordinate. |  |

*Table 9 Attributes of GPSLocation*

# **REST Resource**

## **Detailed definitions for resources**

*Note: context root for the API:* ***https://{host:port}/sniffer-mind/api***

|  |  |  |
| --- | --- | --- |
| URL | JEE method | Detail |
| **/sniffers** | save | HTTP verb: **POST**   * Add a new sniffer or update an existing sniffer into the system   Request parameters:   * **sniffer** (**Sniffer)**: contains information of a sniffer (*Table 2 Attributes of Sniffer*)   Data Format (Content-Type):   * application/json * application/xml   Header: (get token from /auth api)   * key - value: token – tokenValue   Response status:   * **201 – Created:** save successfully * **400 – Bad Request:** input data is violated constraint validation, input data is wrong format, code is not existed(in case updating) * **500 – Internal server error**: for internal errors. Please contact admin if you have this error |
| **/sniffers/{code}** | readByCode | HTTP verb: **GET**   * Get a sniffer with a given code   Path parameter:   * **code** (**String)**. The assigned code to the sniffer.   Authorization: Basic Auth  Response:   * **200 – OK** * **404 – Not Found:** No sniffer found with given code * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error   Response data:  **sniffer (Sniffer)**: returned sniffer with given code. |
| **/sniffers/{code}** | updateStatusByCode | HTTP verb: **PUT**   * Update sniffer status with specific code to given status   Path parameter:   * **code** (**String)**: The assigned code to the sniffer.   Query parameters:   * **status (SnifferStatus)**: The status to update   Header: (get token from /auth api)   * key - value: token – tokenValue   Response status:   * **200 – OK:** update successfully * **404 – Not Found**: Sniffer with given code is not existed * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |
| **/sniffers?searchBy=<val>&sortBy=<val>&range=<val>** | search | HTTP verb: **GET**   * Return all existing sniffers from the system that matched with given search criteria. The returned values can be sorted by given sort criteria and split by given range criteria   Query parameters:   * **searchBy** (**SearchByCriteria)**: Contains list of search criteria. Accepted search fields are: ***code, name, status, location***.   Accepted expressions are: ***equal, notequal, gt***(greater than), ***ge***(greater than or equal), ***lt***(less than), ***le***(less than or equal), ***between, notbetween, like***   * **sortBy (SortByCriteria)**: Contains list of criteria for sorting the returned result. Accepted sort fields are: ***name, status, location*** * **range (RangeCriteria)**: Contains two attributes: *from*, *to* which are used to restricted the result   Data Format (Content-Type):   * application/json * application/xml   Authorization: Basic Auth  Response status:   * **200 – Ok** * **400 – Bad Request:** input search criteria or sort criteria is not valid. Search or sort fields are not accepted. * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error   Response data:   * **sniffers (List<Sniffer>)**: all sniffers for the search criteria, return empty if there is no sniffer matched with the criteria |
| **/pollutants** | save | HTTP verb: **POST**   * Add new a pollutant or update an existing pollutant into the system   Request parameters:   * **pollutant (Pollutant)**: contains information of a Pollutant (*Table 3 Attributes of Pollutant*)   Data Format (Content-Type):   * application/json * application/xml   Header: (get token from /auth api)   * key - value: token – tokenValue   Response status:   * **201 – Created:** save successfully * **400 – Bad Request:** input data is violated constraint validation, input wrong format * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |
| **/pollutants/{code}** | readByCode | HTTP verb: **GET**   * Get a pollutant with a given code   Path parameters:   * **code** (**String**): the code of pollutant.   Data Format (Content-Type):   * application/json * application/xml   Authorization: Basic Auth  Response status:   * **200 – OK** * **400 – Bad Request:** pollutant is not existed * **500 – Internal server error**: for other errors. Please contact Administrator if you have this error   Response data:   * **pollutant (Pollutant)**: pollutant item match with given symbol code. |
| **/pollutants/{code}** | removeByCode | HTTP verb: **DELETE**   * Remove an existing pollutant in the system   Path parameter:   * **code** (**String**): the code of pollutant.   Data Format (Content-Type):   * application/json * application/xml   Header: (get token from /auth api)   * key - value: token – tokenValue   Response status:   * **204 – No Content:** delete successfully or pollutant is not existed * **400 – Bad Request**: if the pollutant is already in use. * **500 – Internal server error**: for other errors. Please contact Administrator if you have this error |
| / **pollutants? searchBy=<val>&orderBy=<val>&range=<val>** | search | HTTP verb: **GET**   * Search pollutants from the system based on the given search criteria. The returned result can be sorted with given sort criteria and split by given range criteria.   Query parameters:   * **searchBy** (**SearchByCriteria)**: Contains list of search criteria. Accepted search fields are: ***code, name***.   Accepted expressions are: ***equal, notequal, gt***(greater than), ***ge***(greater than or equal), ***lt***(less than), ***le***(less than or equal), ***between, notbetween, like***   * **sortBy (SortByCriteria)**: Contains list of criteria for sorting the returned result. Accepted sort fields are: **code, *name*** * **range (RangeCriteria)**: Contains two attributes: *from*, *to* which are used to restricted the result   Data Format (Accept / Content-type):   * application/json * application/xml   Authorization: Basic Auth  Response status:   * **200 – Ok** * **400 – Bad Request:** input search criteria or sort criteria is not valid. Search or sort fields are not accepted. * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error   Response data:   * **pollutants** (**List<Pollutant>)**. all pollutants for the search criteria, return empty if there is no pollutant matched with the criteria |
| **/pollutantvalues** | save | HTTP verb: **POST**   * save a packet of pollutant values in to system   Request parameters:   * **values (PollutantValuePacket)**: contains a list of pollutant values that are measured by a sniffer.   Data Format (Content-Type):   * application/json   Response status:   * **201 – Created**: save successfully * **400 – Bad Request:** input data is violated constraint validation, source(sniffer) is not operating, pollutant is not existed, measuredAt is in wrong format * **404 – Bad Request:** source(sniffer) is not existed * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |
| / **pollutantvalues? searchBy=<val>&sortBy=<val>&range=<val>** | search | HTTP verb: **GET**   * Search pollutant values from the system based on the given search criteria. The returned result can be sorted with given sort criteria and split by given range criteria.   Query parameters:   * **searchBy** (**SearchByCriteria)**: Contains list of search criteria.   Accepted search fields are: ***sourceId, code, measuredAt, sensor***.  Accepted expressions for ***sourceId, code, sensor*** are: ***equal, notequal, gt***(greater than), ***ge***(greater than or equal), ***lt***(less than), ***le***(less than or equal), ***between, notbetween, like.***  Accepted expressions for ***measuredAt*** are: ***equal, gt***(greater than), ***ge***(greater than or equal), ***lt***(less than), ***le***(less than or equal)   * **sortBy (SortByCriteria)**: Contains list of criteria for sorting the returned result. Accepted sort fields are: ***code, measuredAt*** * **range (RangeCriteria)**: Contains two attributes: *from*, *to* which are used to restricted the result   Data Format (Accept / Content-type):   * application/json * application/xml   Authorization: Basic Auth  Response status:   * **200 – Ok** * **400 – Bad Request:** input search criteria or sort criteria is not valid. Search or sort fields are not accepted. * **500 – Internal server error**: for other errors. Please contact Administrator if you have this error   Response data:   * **values** (**GroupedPollutantValues):** matched with given criteria. |
| **/caches** | refreshCaches | HTTP verb: **PUT**   * Refresh all caches in system   Response status:   * **200 – OK:** refresh caches successfully * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |
| **/caches/pollutants** | refreshPollutantCache | HTTP verb: **PUT**   * Refresh pollutant cache in system   Response status:   * **200 – OK:** refresh cache successfully * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |
| **/caches/sniffers** | refreshSnifferCache | HTTP verb: **PUT**   * Refresh sniffer cache in system   Response status:   * **200 – OK:** refresh cache successfully * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |
| **/aqi?searchBy=<val>&sortBy=<val>&range=<val>** | search | HTTP verb: **GET**   * Return all the AQI value that are satisfied the given search criteria, sorted by the sort criteria and split by range criteria.   Query Parameters:   * **searchBy : SearchByCriteria** – Contains list of searching criteria, including **sourceId, pollutantCode, aqivalue.**   Accepted comparator statements are: **equal, notequal, gt**(greater than)**, ge(**greater than or equal)**, lt**(less than)**, le**(less than or equal)**, between, notbetween, like.**   * **sortBy : SortByCriteria** – contains list of sorting criteria, including **date, time** and **aqivalue.** * **range : RangeCriteria** – contains two two attributes (from, to) which are used to restricted the result.   Data Format (Content – Type):   * application/json * application/xml   Response status:   * **200 – Ok** * **400 – Bad Request:** input search criteria or sort criteria is not valid. Search or sort fields are not accepted. * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error   Response data:   * **ListAQIResponse(List<AQI>, ResultStatus):** all AQI values that fit for the search criteria, return empty if there is no value matched with the criteria. |
| **/aqi/dailyResponsiblePollutant?source=<val>&time=<val>** | getDailyResponsiblePollutant | HTTP verb: **GET**   * Return the daily AQI value from specific sniffers on a specific day.   Query Parameters:   * **source : String** – contains the **sourceId** of the using sniffer. Should you choose to ignore this query, the result will be considered on all available functional sniffers. * **time : String** – contains the specific date time under the format **yyyy-mm-ddThh:mm:ss** (i.e. 2017-07-31T00:00:00)**.**   Data Format (Content – Type):   * application/json * application/xml   Response status:   * **200 – Ok** * **400 – Bad Request:** input search criteria or sort criteria is not valid. Search or sort fields are not accepted. * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error   Response data:   * **ListAQIResponse(List<AQI>, ResultStatus):** the daily AQI values of that particular day, return empty if there is no value matched with the query parameters. |
| **/aqi/hourlyResponsiblePollutant?source=<val>&time=<val>** | getHourlyResponsiblePollutant | HTTP verb: **GET**   * Return the daily AQI value from specific sniffers on a specific hour.   Query Parameters:   * **source : String** – contains the **sourceId** of the using sniffer. Should you choose to ignore this query, the result will be considered on all available functional sniffers. * **time : String** – contains the specific date time under the format **yyyy-mm-ddThh:mm:ss** (i.e. 2017-07-31T07:03:23)**.**   Data Format (Content – Type):   * application/json * application/xml   Response status:   * **200 – Ok** * **400 – Bad Request:** input search criteria or sort criteria is not valid. Search or sort fields are not accepted. * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error   Response data:   * **ListAQIResponse(List<AQI>, ResultStatus):** the hourly AQI values of that particular, return empty if there is no value matched with the query parameters. |
| **/auth** | getAuthenticationToken | HTTP verb: **POST**   * Get a token   Request parameters:   * **user (User)**: contains information of a user (*user name and password*)   Data Format (Content-Type):   * application/json * application/xml   Response status:   * **200 – OK:** successfully * **400 – Bad Request:** input data is violated constraint validation, input wrong format * **401 – Unauthorized:** input data is invalid * **500 – Internal server error**: for internal errors. Please contact Administrator if you have this error |

# **Messages**

## **Error**

**Format of error messages**:

* **SFER0\*\*:** Data sent from request violated business logic.

|  |  |  |
| --- | --- | --- |
| **Error code** | **Message in English** | **Message in Vietnamese** |
| SFER001 | Source is not registered! Please register it before sending a packet. | Source chưa được đăng kí! Xin hãy đăng kí trước khi gửi một packet. |
| SFER002 | This source {0} had already sent a packet with the same creation time! | Source với mã {0} đã từng gửi một packet với cùng creation time! |
| SFER003 | Pollutant item with symbol code {0} has already existed in database! | Pollutant Item với mã {0} đã tồn tại trong database! |
| SFER004 | MacAddress is not matched! | MacAddress không trùng khớp. |
| SFER005 | Address doesn't belong to sniffer! | Địa chỉ của sniffer không chính xác. |
| SFER006 | The pollutant {0} is already in use. | Thành phần {0} đang được sử dụng. |

* **SFER1\*\*:** Entity not found.

|  |  |  |
| --- | --- | --- |
| **Error code** | **Message in English** | **Message in Vietnamese** |
| SFER101 | Pollutant item with symbol code {0} does not exist or inactivated! | Pollutant với mã {0} không tồn tại hoặc đã bị vô hiệu hoá! |
| SFER102 | Packet with id {0} does not exist! | Packet với id {0} không tồn tại! |
| SFER103 | Sniffer is not operating! | Sniffer không hoạt động! |
| SFER104 | Sniffer is not existed! | Sniffer không tồn tại! |

* **SFER2\*\*:** Data sent from request in wrong format or there’re errors in request header.

|  |  |  |
| --- | --- | --- |
| **Error code** | **Message in English** | **Message in Vietnamese** |
| SFER201 | {0} has wrong datatype! Please check API documents for more information! | {0} bị sai kiểu dữ liệu! Xin hãy xem API documentations để biết thêm thông tin! |
| SFER202 | MIME type is not supported! | MIME type trong yêu cầu không được hỗ trợ! |
| SFER203 | Content type is not supported! | Content type trong yêu cầu không được hỗ trợ! |
| SFER204 | There're syntax errors in your request body! Please contact admin to fix this! | Đã có lỗi cú pháp trong yêu cầu! Xin hãy liên hệ admin để khắc phục vấn đề! |
| SFER205 | Server can not recognize your query parameter(s)! Please check again! | Máy chủ không thể nhận diện giá trị của tham số được truyền vào trong URL! Xin hãy kiểm tra lại! |
| SFER206 | The operation ({0}) is not supported! | Phương thức {0} không được hỗ trợ! |
| SFER207 | The field {0} is not supported for ordering | Trường dữ liệu {0} không được hỗ trợ phương thức sắp xếp |

* **SFER3\*\*:** Internal server errors.

|  |  |  |
| --- | --- | --- |
| **Error code** | **Message in English** | **Message in Vietnamese** |
| SFER301 | Can't find message for errorCode code {0}! Please contact admin to fix this! | Không tìm thấy thông báo cho errorCode code {0}! Xin hãy liên hệ với admin để khắc phục vấn đề ! |

* **SFER4\*\*:** Data sent from request violated constraint validation.

|  |  |  |
| --- | --- | --- |
| **Error code** | **Message in English** | **Message in Vietnamese** |
| SFER400 | This is the default message for Bean validation being violated! Please contact admin to fix this! | Đây là thông báo mặc định khi một trong số dữ liệu từ yêu cầu vi phạm điều kiện của máy chủ! Xin hãy liên hệ với admin để sửa chữa vấn đề này! |
| SFER401 | Size of {0} must between 1 and 30 | Độ dài của {0} phải nằm giữa 1 và 30 |
| SFER402 | {0} must is missing or invalid | Độ dài của {0} phải nằm giữa 1 và 30 |
| SFER403 | {0} must be a valid MacAddress | {0} không phải là MacAddress |
| SFER404 | Sniffer code is missing or invalid | Sniffer code là bắt buộc. |

# **Request Templates**

## **Sniffer**

* 1. **Add a new sniffer:**

Example request’s body in JSON format:

{

"name": "axon\_sniffer",

"address":

{

"country": "vietnam",

"province": "Ho Chi Minh City",

"district": "Tan Binh Dist.",

"ward": "ward 4",

"street": "Truong Son St.",

"houseNr": "123",

"gpsLocation": {

"latitude": 10.805596,

"longitude":106.662175

}

},

"macAddress": "90-A2-DA-00-2F-E4",

"status": "INACTIVE"

}

Example request’s body in XML format:

<sniffer>

<name>axon\_sniffer</name>

<address>

<country>vietnam</country>

<province>Ho Chi Minh City</province>

<district>Tan Binh Dist.</district>

<ward>ward 4</ward>

<street>Truong Son St.</street>

<houseNr>123</houseNr>

<gpsLocation>

<latitude>10.805596</latitude>

<longitude>106.662175</longitude>

</gpsLocation>

</address>

<macAddress>90-A2-DA-00-2F-E4</macAddress>

<status>INACTIVE</status>

</sniffer>

* 1. **Update an existing sniffer:**

Example request’s body in JSON format:

{

"code": "143253",

"name": "axon\_sniffer",

"address":

{

"country": "vietnam",

"province": "Ho Chi Minh City",

"district": "Tan Binh Dist.",

"ward": "ward 4",

"street": "Truong Son St.",

"houseNr": "123",

"gpsLocation": {

"latitude": 10.805596,

"longitude":106.662175

}

},

"macAddress": "90-A2-DA-00-2F-E4",

"status": "OPERATING"

}

Example request’s body in XML format:

<sniffer>

<code>143253</code>

<name>axon\_sniffer</name>

<address>

<country>vietnam</country>

<province>Ho Chi Minh City</province>

<district>Tan Binh Dist.</district>

<ward>ward 4</ward>

<street>Truong Son St.</street>

<houseNr>123</houseNr>

<gpsLocation>

<latitude>10.805596</latitude>

<longitude>106.662175</longitude>

</gpsLocation>

</address>

<macAddress>90-A2-DA-00-2F-E4</macAddress>

<status>OPERATING</status>

</sniffer>

* 1. **Search sniffer:**

Example for **searchBy** query parameters:

This query parameters is used for search sniffers which have **code equal to “143253”** or **name contain “sniffer”.**

* Query parameters in JSON format:

{

"criteria": [

{

"field":"CODE",

"expression": "equal",

"value":"143253"

}

],

"or":[

{

"criteria": [

{

"field":"NAME",

"expression": "like",

"value":"sniffer"

}

]

}

]

}

* Query parameters in JSON format without blank spaces:

{"criteria":[{"field":"CODE","expression":"equal","value":"143253"}],"or":[{"criteria":[{"field":"NAME","expression":"like","value":"sniffer"}]}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22criteria%22%3A%5B%7B%22field%22%3A%22CODE%22%2C%22expression%22%3A%22equal%22%2C%22value%22%3A%22143253%22%7D%5D%2C%22or%22%3A%5B%7B%22criteria%22%3A%5B%7B%22field%22%3A%22NAME%22%2C%22expression%22%3A%22like%22%2C%22value%22%3A%22sniffer%22%7D%5D%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/sniffers?searchBy=%7B%22criteria%22%3A%5B%7B%22field%22%3A%22CODE%22%2C%22expression%22%3A%22equal%22%2C%22value%22%3A%22143253%22%7D%5D%2C%22or%22%3A%5B%7B%22criteria%22%3A%5B%7B%22field%22%3A%22NAME%22%2C%22expression%22%3A%22like%22%2C%22value%22%3A%22sniffer%22%7D%5D%7D%5D%7D

Example for **sortBy** query parameters:

This query parameters is used for **sort sniffers by name in ascending order**

* Query parameters in JSON format:

{

"order": [{

"field": "name",

"descending": false

}]

}

* Query parameters in JSON format without blank spaces:

{"order":[{"field":"name","descending":false}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22order%22%3A%5B%7B%22field%22%3A%22name%22%2C%22descending%22%3Afalse%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/sniffers?sortBy=%7B%22order%22%3A%5B%7B%22field%22%3A%22name%22%2C%22descending%22%3Afalse%7D%5D%7D

Example for **range** query parameters:

This query parameters is used for **get the results from 1 to 2**:

* Query parameters in JSON format:

{

"from": 1,

"to": 2

}

* Query parameters in JSON format without blank spaces:

{"from": 1,"to": 2}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22from%22%3A+1%2C%22to%22%3A+2%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/sniffers?range**=**%7B%22from%22%3A+1%2C%22to%22%3A+2%7D

Example query parameters for search sniffers which have **code equal to “143253”** or **name contain “sniffer”**, then **sort by name in ascending order**, finally **get the results from 1 to 2**:

http://iot.axonactive.vn/sniffer-mind/api/sniffers?**searchBy=**%7B%22criteria%22%3A%5B%7B%22field%22%3A%22CODE%22%2C%22expression%22%3A%22equal%22%2C%22value%22%3A%22143253%22%7D%5D%2C%22or%22%3A%5B%7B%22criteria%22%3A%5B%7B%22field%22%3A%22NAME%22%2C%22expression%22%3A%22like%22%2C%22value%22%3A%22sniffer%22%7D%5D%7D%5D%7D**&sortBy=**%7B%22order%22%3A%5B%7B%22field%22%3A%22name%22%2C%22descending%22%3Afalse%7D%5D%7D**&range=**%7B%22from%22%3A+1%2C%22to%22%3A+2%7D

## **Pollutant**

* 1. **Save a pollutant:**

Example request’s body in JSON format:

{

"code": "CO2",

"name": "Carbon dioxide",

"description": "Carbon dioxide (chemical formula CO2) is a colorless and odorless gas vital to life on Earth. This naturally occurring chemical compound is composed of a carbon atom covalently double bonded to two oxygen atoms",

"defaultUnit": "ppb"

}

Example request’s body in XML format:

<pollutant>

<code>CO2</code>

<name>Carbon dioxide</name>

<description>

Carbon dioxide (chemical formula CO2) is a colorless and odorless gas vital to life on Earth. This naturally occurring chemical compound is composed of a carbon atom covalently double bonded to two oxygen atoms

</description>

<defaultUnit>ppb</defaultUnit>

</pollutant>

* 1. **Search pollutant:**

Example for **searchBy** query parameters:

This query parameters is used for search pollutants which have **code contain “O”** or **name equal to “Temperature”**

* Query parameters in JSON format:

{

"criteria": [

{

"field":"CODE",

"expression": "like",

"value":"O"

}

],

"or":[

{

"criteria": [

{

"field":"NAME",

"expression": "equal",

"value":"Temperature"

}

]

}

]

}

* Query parameters in JSON format without blank spaces:

{"criteria":[{"field":"CODE","expression":"like","value":"O"}],"or":[{"criteria":[{"field":"NAME","expression":"equal","value":"Temperature"}]}]}}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22criteria%22%3A%5B%7B%22field%22%3A%22CODE%22%2C%22expression%22%3A%22like%22%2C%22value%22%3A%22O%22%7D%5D%2C%22or%22%3A%5B%7B%22criteria%22%3A%5B%7B%22field%22%3A%22NAME%22%2C%22expression%22%3A%22equal%22%2C%22value%22%3A%22Temperature%22%7D%5D%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/pollutants?searchBy= %7B%22criteria%22%3A%5B%7B%22field%22%3A%22CODE%22%2C%22expression%22%3A%22like%22%2C%22value%22%3A%22O%22%7D%5D%2C%22or%22%3A%5B%7B%22criteria%22%3A%5B%7B%22field%22%3A%22NAME%22%2C%22expression%22%3A%22equal%22%2C%22value%22%3A%22Temperature%22%7D%5D%7D%5D%7D

Example for **orderBy** query parameters:

This query parameters is used for **sort pollutants by code in descending order**

* Query parameters in JSON format:

{

"order": [{

"field": "code",

"descending": true

}]

}

* Query parameters in JSON format without blank spaces:

{"order":[{"field":"code","descending":true}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22order%22%3A%5B%7B%22field%22%3A%22code%22%2C%22descending%22%3Atrue%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/pollutants?orderBy= %7B%22order%22%3A%5B%7B%22field%22%3A%22code%22%2C%22descending%22%3Atrue%7D%5D%7D

Example for **range** query parameters:

This query parameters is used for **get the results from 0 to 2**:

* Query parameters in JSON format:

{

"from": 0,

"to": 2

}

* Query parameters in JSON format without blank spaces:

{"from": 0,"to": 2}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22from%22%3A+0%2C%22to%22%3A+2%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/pollutants?range**=** %7B%22from%22%3A+0%2C%22to%22%3A+2%7D

Example query parameters for search pollutants which have **code contain “O”** or **name equal to “Temperature”**, then **sort by code in descending order**, finally **get results from 0 to 3**:

http://iot.axonactive.vn/sniffer-mind/api/pollutants?**searchBy=**%7B%22criteria%22%3A%5B%7B%22field%22%3A%22CODE%22%2C%22expression%22%3A%22like%22%2C%22value%22%3A%22O%22%7D%5D%2C%22or%22%3A%5B%7B%22criteria%22%3A%5B%7B%22field%22%3A%22NAME%22%2C%22expression%22%3A%22equal%22%2C%22value%22%3A%22Temperature%22%7D%5D%7D%5D%7D&**orderBy=**%7B%22order%22%3A%5B%7B%22field%22%3A%22code%22%2C%22descending%22%3Atrue%7D%5D%7D&**range=**%7B%22from%22%3A+0%2C%22to%22%3A+3%7D

## **PollutantValue**

* 1. **Save PollutantValues without measuredAt:**

Example request’s body in JSON format:

{

"source": {

"sourceId": "143253",

"macAddress": "90-A2-DA-00-2F-E4",

"gpsLocation":

{

"latitude": 10.805596,

"longitude":106.662175

}

},

"values":[{

"code": "O3",

"sensor":"Plan Tower",

"value":

{

"value": 11

}

},

{

"code": "HUM",

"sensor":"Plan Tower",

"value":

{

"value": 11

}

}]

}

Example request’s body in XML format:

<pollutantValuePacket>

<source>

<sourceId>143253</sourceId>

<macAddress>90-A2-DA-00-2F-E4</macAddress>

<gpsLocation>

<latitude>10.805596</latitude>

<longitude>106.662175</longitude>

</gpsLocation>

</source>

<values>

<code>O3</code>

<sensor>Plan Tower</sensor>

<value>

<value>11</value>

</value>

</values>

<values>

<code>CO</code>

<sensor>Plan Tower</sensor>

<value>

<value>11</value>

</value>

</values>

</pollutantValuePacket>

* 1. **Save PollutantValues with measuredAt (default TimeZone and GTM+7 TimeZone):**

Example request’s body in JSON format:

{

"source": {

"sourceId": "143253",

"macAddress": "90-A2-DA-00-2F-E4",

"gpsLocation": "sampleGPS"

},

"values":[{

"code": "HUM",

"sensor":"sampleSensor1",

"value":

{

"value": 25.3,

"measuredAt": "2016-10-21T07:45:00.000Z"

}

},

{

"code": "TEMP",

"sensor":"sampleSensor2",

"value":

{

"value": 28.3,

"measuredAt": " 2016-10-21T07:45:00.000+07"

}

}]

}

Example request’s body in XML format:

<pollutantValuePacket>

<source>

<sourceId>143253</sourceId>

<macAddress>90-A2-DA-00-2F-E4</macAddress>

<gpsLocation>

<latitude>10.805596</latitude>

<longitude>106.662175</longitude>

</gpsLocation>

</source>

<values>

<code>O3</code>

<sensor>Plan Tower</sensor>

<value>

<value>11</value>

<measuredAt>2016-10-21T07:45:00.000Z</measuredAt>

</value>

</values>

<values>

<code>CO</code>

<sensor>Plan Tower</sensor>

<value>

<value>11</value>

<measuredAt>2016-10-21T07:45:00.000+07</measuredAt>

</value>

</values>

</pollutantValuePacket>

* 1. **Search pollutantvalue:**

Example for **searchBy** query parameters:

This query parameters is used for search pollutants which have **measuredAt greater than “2016-10-06T11:54:55.000+07” and less than “2016-10-10T11:52:25.000+07”**

* Query parameters in JSON format:

{

"criteria": [

{

"field":"measuredAt",

"expression": "gt",

"value":"2016-10-06T11:54:55.000+07"

},

{

"field":"measuredAt",

"expression": "lt",

"value":"2016-10-10T11:52:25.000+07"

}

]

}

* Query parameters in JSON format without blank spaces:

{"criteria":[{"field":"measuredAt","expression":"gt","value":"2016-10-06T11:54:55.000+07"},{"field":"measuredAt","expression":"lt","value":"2016-10-10T11:52:25.000+07"}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22criteria%22%3A%5B%7B%22field%22%3A%22measuredAt%22%2C%22expression%22%3A%22gt%22%2C%22value%22%3A%222016-10-06T11%3A55%3A36.000%2B07%22%7D%2C%7B%22field%22%3A%22measuredAt%22%2C%22expression%22%3A%22lt%22%2C%22value%22%3A%222016-10-10T11%3A51%3A58.000%2B07%22%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/pollutantvalues?searchBy= %7B%22criteria%22%3A%5B%7B%22field%22%3A%22measuredAt%22%2C%22expression%22%3A%22gt%22%2C%22value%22%3A%222016-10-06T11%3A55%3A36.000%2B07%22%7D%2C%7B%22field%22%3A%22measuredAt%22%2C%22expression%22%3A%22lt%22%2C%22value%22%3A%222016-10-10T11%3A51%3A58.000%2B07%22%7D%5D%7D

Example for **sortBy** query parameters:

This query parameters is used for **sort pollutantvalues by code in ascending order**

* Query parameters in JSON format:

{

"order": [{

"field": "code",

"descending": false

}]

}

* Query parameters in JSON format without blank spaces:

{"order":[{"field":"code","descending":false}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22order%22%3A%5B%7B%22field%22%3A%22code%22%2C%22descending%22%3Afalse%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/pollutantvalues?orderBy=%7B%22order%22%3A%5B%7B%22field%22%3A%22code%22%2C%22descending%22%3Afalse%7D%5D%7D

Example for **range** query parameters:

This query parameters is used for **get the results from 0 to 3**:

* Query parameters in JSON format:

{

"from": 0,

"to": 3

}

* Query parameters in JSON format without blank spaces:

{"from": 0,"to": 3}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22from%22%3A+0%2C%22to%22%3A+3%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/pollutantvalues?range**=** %7B%22from%22%3A+0%2C%22to%22%3A+3%7D

Example query parameters for search pollutantvalues which have **measuredAt greater than “2016-10-06T11:54:55.000+07” and less than “2016-10-10T11:52:25.000+07”**, then **sort by code in ascending order**, finally **get results from 0 to 3**:

<http://iot.axonactive.vn/sniffer-mind/api/pollutantvalues?searchBy=%7B%22criteria%22%3A%5B%7B%22field%22%3A%22measuredAt%22%2C%22expression%22%3A%22gt%22%2C%22value%22%3A%222016-10-06T11%3A54%3A55.000%2B07%22%7D%2C%7B%22field%22%3A%22measuredAt%22%2C%22expression%22%3A%22lt%22%2C%22value%22%3A%222016-10-10T11%3A52%3A25.000%2B07%22%7D%5D%7D&sortBy=%7B%22order%22%3A%5B%7B%22field%22%3A%22code%22%2C%22descending%22%3Afalse%7D%5D%7D&range=%7B%22from%22%3A+0%2C%22to%22%3A+3%7D>

## **AQI (Air Quality Index)**

* 1. **Search AQI Value:**

Example for **searchBy** query parameters:

This query parameters is used for search SO2 AQI Value which have was not collected from the sniffer with the source Id 143253.

* Query parameters in JSON format:

{

"criteria":

[

{

"field":"sourceId",

"expression": "notequal",

"value":"143253"

},

{

"field":"pollutantCode",

"expression": "equal",

"value":"SO2"

}

]

}

* Query parameters in JSON format without blank spaces:

{"criteria": [{"field":"sourceId", "expression": "notequal","value":"143253"},{"field":"pollutantCode","expression": "equal","value":"SO2"}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22criteria%22%3A+%5B%7B%22field%22%3A%22sourceId%22%2C+%22expression%22%3A+%22notequal%22%2C%22value%22%3A%22143253%22%7D%2C%7B%22field%22%3A%22pollutantCode%22%2C%22expression%22%3A+%22equal%22%2C%22value%22%3A%22SO2%22%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/aqi?searchBy= %7B%22criteria%22%3A+%5B%7B%22field%22%3A%22sourceId%22%2C+%22expression%22%3A+%22notequal%22%2C%22value%22%3A%22143253%22%7D%2C%7B%22field%22%3A%22pollutantCode%22%2C%22expression%22%3A+%22equal%22%2C%22value%22%3A%22SO2%22%7D%5D%7D

Example for **sortBy** query parameters:

This query parameters is used for sort AQI Value based on date and time in ascending order.

* Query parameters in JSON format:

{

"order":

[

{

"field": "date",

"descending": false

},

{

"field": "time",

"descending": false

}

]

}

* Query parameters in JSON format without blank spaces:

{"order": [{"field": "date","descending": false},{"field": "time","descending": false}]}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22order%22%3A+%5B%7B%22field%22%3A+%22date%22%2C%22descending%22%3A+false%7D%2C%7B%22field%22%3A+%22time%22%2C%22descending%22%3A+false%7D%5D%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/aqi?sortBy=%7B%22order%22%3A+%5B%7B%22field%22%3A+%22date%22%2C%22descending%22%3A+false%7D%2C%7B%22field%22%3A+%22time%22%2C%22descending%22%3A+false%7D%5D%7D

Example for **range** query parameters:

This query parameters is used for **get the results from 0 to 3**:

* Query parameters in JSON format:

{

"from": 0,

"to": 3

}

* Query parameters in JSON format without blank spaces:

{"from": 0,"to": 3}

* Query parameters after encoded (use online tool at <http://www.url-encode-decode.com/>):

%7B%22from%22%3A+0%2C%22to%22%3A+3%7D

* Final URL:

http://iot.axonactive.vn/sniffer-mind/api/aqi?range**=** %7B%22from%22%3A+0%2C%22to%22%3A+3%7D

Example query parameters for search SO2 AQI Value which have was not collected from the sniffer with the source Id 143253, sorting by date and time ascendingly and display the first 3 result is:

http://iot.axonactive.vn/sniffer-mind/api/aqi?searchBy= %7B%22criteria%22%3A+%5B%7B%22field%22%3A%22sourceId%22%2C+%22expression%22%3A+%22notequal%22%2C%22value%22%3A%22143253%22%7D%2C%7B%22field%22%3A%22pollutantCode%22%2C%22expression%22%3A+%22equal%22%2C%22value%22%3A%22SO2%22%7D%5D%7D&sortBy=%7B%22order%22%3A+%5B%7B%22field%22%3A+%22date%22%2C%22descending%22%3A+false%7D%2C%7B%22field%22%3A+%22time%22%2C%22descending%22%3A+false%7D%5D%7D&range=%7B%22from%22%3A+0%2C%22to%22%3A+3%7D

* 1. **Get Daily AQI Value for a specific day:**

Example for query parameters:

Example query parameters for search Daily AQI Value from sniffer sourceId = 143253 and on the 21 Jun, 2017: http://iot.axonactive.vn/sniffer-mind/api/ dailyResponsiblePollutant?source=143253&time=2017-06-21T00:00:00

Example query parameters for search Daily AQI Value globally (consider on all sniffers) and on the 21 Jun, 2017: http://iot.axonactive.vn/sniffer-mind/api/ dailyResponsiblePollutant?time=2017-06-21T00:00:00

* 1. **Get Daily AQI Value for a specific hour:**

Example for query parameters:

Example query parameters for search Daily AQI Value from sniffer sourceId = 143253 and on the 21 Jun, 2016 at 7:00: http://iot.axonactive.vn/sniffer-mind/api/aqi/hourlyResponsiblePollutant?time=2016-06-21T07:00:00

Example query parameters for search Daily AQI Value globally (consider on all sniffers) and on the 21 Jun, 2017 at 7:00: http://iot.axonactive.vn/sniffer-mind/api/ hourlyResponsiblePollutant?time=2016-06-21T07:00:00

## **Authentication**

Example request’s body in JSON format:

{

"username":"username",

"password":"password"

}

Example request’s body in XML format:

<user>

<username>username</username>

<password>password</password>

</sniffer>