API Reference

RIoT - OpenAPI definition

API Version: v1

INDEX

API

1. ALERT-CONTROLLER

1.1 GET /alerts/{alertId}

Get single alert

The request return the alert with corresponding id as alertId if visible by the current user.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*alertId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  threshold  number
  type       enum     ALLOWED:GREATER, LOWER, EQUAL
  deleted    boolean
  entity     integer
  sensor     integer
  lastSent   string
  alertId    integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

1.2 PUT /alerts/{alertId}

Edit alert

The request return an object corresponding to the alert edited if successful

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*alertId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  threshold  number
  type       enum     ALLOWED:GREATER, LOWER, EQUAL
  deleted    boolean
  entity     integer
  sensor     integer
  lastSent   string
  alertId    integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

1.3 POST /alerts/{alertId}

Enable/disable alert for user

The request used to enable/disable user alert by alert id

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*alertId	int32	

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
*userId	int32	
*enable	boolean	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

1.4 DELETE /alerts/{alertId}

Delete alert by id

The request used to delete an alert by its id

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*alertId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

1.5 GET /alerts

Get alerts

The request return an object with enabled and disabled list of the alerts visible by the current user. If admin all alerts will be in enabled

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
entityId	int32	
sensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

1.6 POST /alerts

Create alert

The request return an object corresponding to the alert created if successful

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  threshold  number
  type       enum     ALLOWED:GREATER, LOWER, EQUAL
  deleted    boolean
  entity     integer
  sensor     integer
  lastSent   string
  alertId    integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

1.7 DELETE /alerts

Delete alerts by sensorId

The request used to delete an alert by sensorId

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
*sensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - **application/json**

STATUS CODE - **401:** The authentication failed

RESPONSE MODEL - **application/json**

STATUS CODE - **403:** Not authorized. Only admins can do it

RESPONSE MODEL - **application/json**

STATUS CODE - **500:** Server error

RESPONSE MODEL - **application/json**

---

2. AUTH-CONTROLLER

2.1 POST /auth

Normal authentication

The request for getting the authentication token used for other requests

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: The authentication is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

2.2 POST /auth/telegram

Telegram authentication

The request for getting the authentication token used for other requests for telegram

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: The request is successful. It returns as code: "0" - the telegramName is not associated with anybody, "1" - the telegramName is associated with somebody and the telegramChat will be setted, "2" - the telegramName is associated with somebody and the telegramChat is present and the same as the request

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

2.3 POST /auth/tfa

Tfa authentication

The request for getting the authentication token used for other requests using the tfa code and the previous token for the authorization

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The authentication is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

---

3. DATA-CONTROLLER

3.1 GET /data

Get sensors values

The request return a map containing couples "key-list of values" where the key is a sensor id and the list of values is made of the records of values related to the sensor with that id

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
sensors	array of integer	
limit	int32	
entityId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

3.2 GET /data/{sensorId}

Get last sensor value

The request return the last value record related to the sensor that is identified by the given id.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*sensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4. DEVICE-CONTROLLER

4.1 GET /devices

See a list of all the devices you have access to

This request allows you to see all the devices you have access to. You can also filter this research by either giving in input the entityId and/or the gatewayId, or by the cmdEnabled parameter. This last filter is available for administrators only.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
entityId	int32	
gatewayId	int32	
cmdEnabled	boolean	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
   name          string
   frequency     integer
   realDeviceId  integer
   gateway       integer
   deviceId      integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.2 POST /devices

Inserting a device in the database

This request is available for administrators only. It allows you to create a new device

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name          string
  frequency     integer
  realDeviceId  integer
  gateway       integer
  deviceId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.3 GET /devices/{deviceId}

See the details of a single device

This request allows you to see the details of a single device

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name          string
  frequency     integer
  realDeviceId  integer
  gateway       integer
  deviceId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.4 PUT /devices/{deviceId}

Editing a device

This request is available for administrators only. It allows you to edit a device already saved in the database.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.5 DELETE /devices/{deviceId}

Deleting a device

This request is available for administrators only. It allows you to delete a device from the database.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.6 GET /devices/{deviceId}/sensors

Get access to the sensors of a single device

This request returns a list of all the sensors of a device

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
cmdEnabled	boolean	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
   type          string
   realSensorId  integer
   cmdEnabled    boolean
   device        integer
   sensorId      integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.7 POST /devices/{deviceId}/sensors

Inserting a sensor in the database

This request is available for administrators only. It allows you to create a new sensor ant connect it to a device

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  type          string
  realSensorId  integer
  cmdEnabled    boolean
  device        integer
  sensorId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.8 GET /devices/{deviceId}/sensors/{realSensorId}

Get access to a single sensor of the given device

This request allows you to see the details of a single sensor connected to a device

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	
*realSensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  type          string
  realSensorId  integer
  cmdEnabled    boolean
  device        integer
  sensorId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.9 PUT /devices/{deviceId}/sensors/{realSensorId}

Editing a sensor

This request is available for administrators only. It allows you to edit a sensor already saved in the database.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	
*realSensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

4.10 DELETE /devices/{deviceId}/sensors/{realSensorId}

Deleting a sensor

This request is available for administrators only. It allows you to delete a sensor from the database.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*deviceId	int32	
*realSensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

---

5. ENTITY-CONTROLLER

5.1 GET /entities

Get entities

The request return a list of entities objects

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
sensorId	int32	
userId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
  name      string
  location  string
  deleted   boolean
  entityId  integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

5.2 POST /entities

Create entity

The request return the entity that is been created if successful

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name      string
  location  string
  deleted   boolean
  entityId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

5.3 GET /entities/{entityId}

Get entity

The request return an entity by entity id if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*entityId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name      string
```

```
  location  string
  deleted   boolean
  entityId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

5.4 PUT /entities/{entityId}

Edit entities

The request return the entity that is been edited if successful

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*entityId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name      string
  location  string
  deleted   boolean
  entityId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

5.5 DELETE /entities/{entityId}

Delete entity

The request deletes the specified entity

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*entityId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 409: Database error during the delete

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

---

6. GATEWAY-CONTROLLER

6.1 GET /gateways

Get gateways

The request returns a list of gateways objects

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
deviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
  name       string
  lastSent   string
  gatewayId  integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.2 POST /gateways

Create gateway

The request returns the gateway that has been created, if this operation was successful

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name       string
  lastSent   string
  gatewayId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.3 GET /gateways/{gatewayId}

Get gateway

The request return a gateway by the gateway id if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name       string
  lastSent   string
  gatewayId  integer
```

}

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.4 PUT /gateways/{gatewayId}

Edit a gateway or send a configuration to a gateway

The request returns the gateway that has been edited, if this operation was successful. If the request body contains : "reconfig:true",then the new configuration will be sent to the specified gateway

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.5 DELETE /gateways/{gatewayId}

Delete gateway

The request deletes the specified gateway

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 409: Database error during the delete

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.6 GET /gateways/{gatewayId}/devices

Get devices

The request return a list of devices connected to the gateway thathas the given gateway id

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
   name          string
   frequency     integer
   realDeviceId  integer
   gateway       integer
   deviceId      integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.7 GET /gateways/{gatewayId}/devices/{realDeviceId}

Get device

The request returns the device with the specified realDeviceId if it is connected to the gateway with the specified gatewayId and if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	
*realDeviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
   name          string
   frequency     integer
   realDeviceId  integer
```

```
    gateway       integer
    deviceId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.8 GET /gateways/{gatewayId}/devices/{realDeviceId}/sensors

Get sensors' list

The request returns the sensors connected to the device with the given realDeviceId, only if it is connected to the gateway with the given gatewayId.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	
*realDeviceId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
```
Array of object:
```
    type          string
    realSensorId  integer
    cmdEnabled    boolean
    device        integer
    sensorId      integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

6.9 GET /gateways/{gatewayId}/devices/{realDeviceId}/sensors/{realSensorId}

Get sensor

The request returns the sensor with the specified realSensorId, which isconnected to the device with the specified realDeviceId, which is connected tothe gateway with the specified gatewayId, and only if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*gatewayId	int32	
*realDeviceId	int32	
*realSensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name          string
  frequency     integer
  realDeviceId  integer
  gateway       integer
  deviceId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

---

7. LOG-CONTROLLER

7.1 GET /logs

Get logs

The request return a list of logs objects

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
entityId	int32	
limit	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
```
Array of object:
```
  time       string
  userId     integer
  ipAddr     string
  operation  string
  data       string
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

8. SENSOR-CONTROLLER

8.1 GET /sensors/{sensorId}

Get sensor

The request return a sensor by sensor id if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*sensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  type          string
  realSensorId  integer
  cmdEnabled    boolean
  device        integer
  sensorId      integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

8.2 PUT /sensors/{sensorId}

Send command to a sensor

The request return a string that correspond to the command sent

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*sensorId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
string
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

8.3 GET /sensors

Get sensors

The request return a list of sensors

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
entityId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
```

Array of object:

```
  type          string
```

```
    realSensorId  integer
    cmdEnabled    boolean
    device        integer
    sensorId      integer
  }]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

9. STATS-CONTROLLER

9.1 GET /stats

Get stats values

The request returns a map containing couples "key-value" where the key is a stat name and the value is an integer. Members and moderators can see more stats than an admin related to the entity to which they belong

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

10. USER-CONTROLLER

10.1 GET /users/{userid}

Get access to a single user

This request allows you to see the details of the user who is identified with the given id

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*userid	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name          string
  surname       string
  email         string
  password      string
  type          enum     ALLOWED:USER, MOD, ADMIN
  telegramName  string
  telegramChat  string
  tfa           boolean
  deleted       boolean
  entity        integer
  userId        integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

10.2 PUT /users/{userid}

Editing a user

It allows you to edit a user already saved in the database and returns the user edited with it's new edited values.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*userid	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

10.3 DELETE /users/{userid}

Deleting a user

It allows you to logically delete a user from the database.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*userid	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: Request successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

10.4 GET /users

Get access to the users

This request returns a list of all the users to which you have access, and you can use different parameters to filter the search result set.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
entityId	int32	
disabledAlert	int32	
viewId	int32	
telegramName	string	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

[{
Array of object:

```
name          string
surname       string
email         string
password      string
type          enum     ALLOWED:USER, MOD, ADMIN
telegramName  string
telegramChat  string
```

```
  tfa           boolean
  deleted       boolean
  entity        integer
  userId        integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized. Only admins can do it

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

10.5 POST /users

Create user

The request returns the user that has been created, if this operation was successful

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name          string
  surname       string
  email         string
  password      string
  type          enum     ALLOWED:USER, MOD, ADMIN
  telegramName  string
  telegramChat  string
  tfa           boolean
  deleted       boolean
  entity        integer
  userId        integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 409: Conflict. Database error

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

---

11. VIEW-CONTROLLER

11.1 GET /views

Get views

The request return a list of views

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
   name    string
   user    integer
   viewId  integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

11.2 POST /views

Create view

The request return the view that is been created if successful

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name    string
  user    integer
  viewId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

11.3 GET /views/{viewId}

Get view

The request return a view by view id if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*viewId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  name    string
  user    integer
  viewId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

11.4 DELETE /views/{viewId}

Delete view

The request is successful if the view is been deleted

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*viewId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*Authorization	string	

RESPONSE

STATUS CODE - 200: The delete is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

---

12. VIEW-GRAPH-CONTROLLER

12.1 GET /viewGraphs

Get viewGraphs

The request return a list of viewGraphs

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
userId	int32	
viewId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
[{
Array of object:
   correlation  enum     ALLOWED:NULL, COVARIANCE, PEARSON, SPEARMAN
   view         integer
   sensor1      integer
   sensor2      integer
   viewGraphId  integer
}]
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

12.2 POST /viewGraphs

Create viewGraph

The request return the viewGraph that is been created if successful

REQUEST

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  correlation  enum     ALLOWED:NULL, COVARIANCE, PEARSON, SPEARMAN
  view         integer
  sensor1      integer
  sensor2      integer
  viewGraphId  integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

12.3 GET /viewGraphs/{viewGraphId}

Get viewGraphs

The request return a viewGraph by id if it is visible for the current user

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*viewGraphId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  correlation enum    ALLOWED:NULL, COVARIANCE, PEARSON, SPEARMAN
  view        integer
  sensor1     integer
  sensor2     integer
  viewGraphId integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

12.4 PUT /viewGraphs/{viewGraphId}

Edit viewGraph

The request return the viewGraph that is been edited if successful

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*viewGraphId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The request is successful

RESPONSE MODEL - application/json

```
{
  correlation enum    ALLOWED:NULL, COVARIANCE, PEARSON, SPEARMAN
  view        integer
  sensor1     integer
  sensor2     integer
  viewGraphId integer
}
```

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json

12.5 DELETE /viewGraphs/{viewGraphId}

Delete viewGraph

The request is successful if the viewGraph is been deleted

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*viewGraphId	int32	

HEADER PARAMETERS

NAME	TYPE	DESCRIPTION
*authorization	string	

RESPONSE

STATUS CODE - 200: The delete is successful

RESPONSE MODEL - application/json

STATUS CODE - 400: There is an error in the request

RESPONSE MODEL - application/json

STATUS CODE - 401: The authentication failed

RESPONSE MODEL - application/json

STATUS CODE - 403: Not authorized

RESPONSE MODEL - application/json

STATUS CODE - 500: Server error

RESPONSE MODEL - application/json
