SensorAct API Documentation

Version 1.0

Pandarasamy Arjunan
Nipun Batra
Dr Amarjeet Singh
Dr Pushpendra Singh
MUC, IIIT Delhi

Table of Contents

Preface

I have to write something about

Who should read this document,

Version information and What is implemented and what is not

Organization of this document

Notations – naming conventions, color/font codes statuscode

Etc…

1. Introduction

1.1 What is SensorAct?

SensorAct is a distributed federated middleware architecture for energy management in buildings. SensorAct provides following features.

1. Efficient data aggregation from diverse sensing sources
2. Secure appliance actuation
3. User, device (sensors and actuators) and key management
4. Privacy preserving sharing mechanism
5. Powerful guard rules to protect sensor data and actuators
6. Light weight tasking framework to program the system

For detail information about these features, please refer <link to paper>.

1.2 Architecture

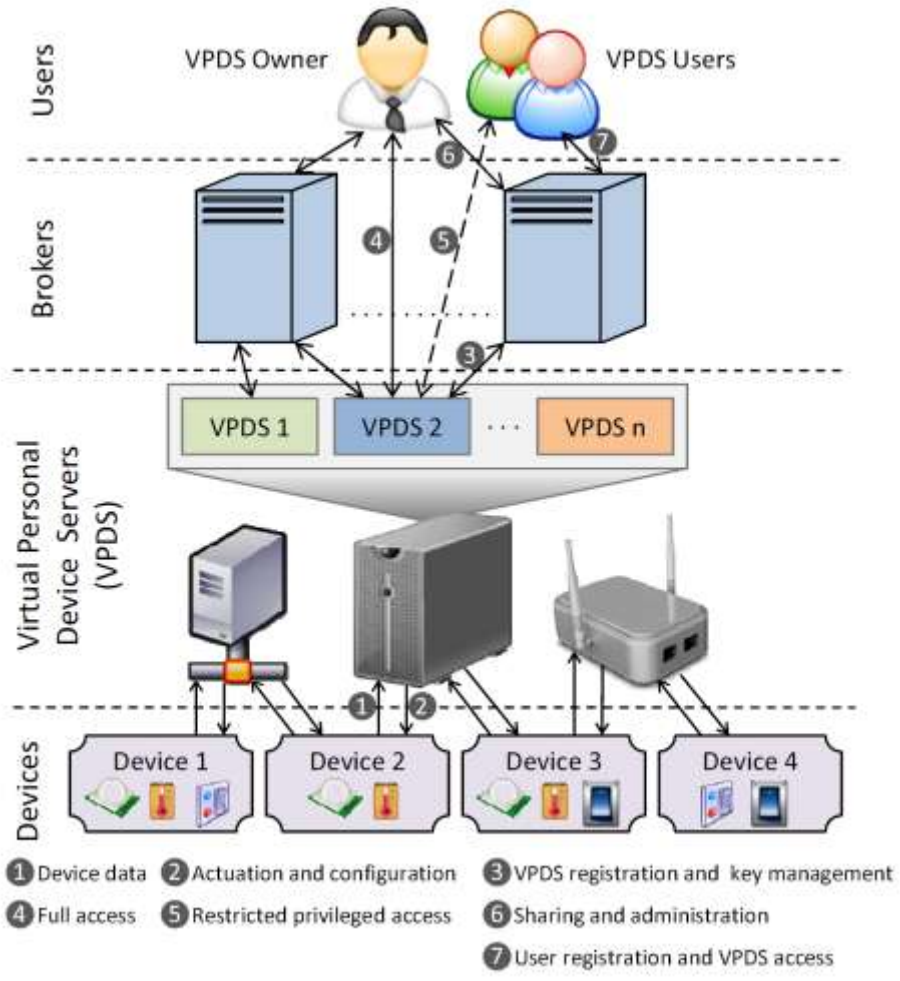

Figure 1 : SensorAct architecture

2. API Documentation

SensorAct exposes rich set of APIs to use most of its functions. Hence users and developers can write custom applications (web, standalone, mobile) using SenorAct APIs depends upon their application requirements such as data visualization, analysis, etc. Moreover, APIs gives a simple, generic and flexible way to use system features. SensorAct as provides following two set of APIs.

1. Broker
   Provides APIs for VPDS and user management.
2. Virtual Personal Device Server (VPDS)
   Provides APIs for devices to upload data, users to configure, manage and access system features and brokers to manage VPDS.

2.1 API format

SensorAct APIs are RESTful HTTP requests that use HTTP verbs (GET and POST) to access system resources. An example API format is

```
POST http://www.sensoract.com/device/add
```

With the body of the request contains the required request data.

2.2 Security and Secretkey

At present, SensorAct does not support secure connection (SSL/HTTPS) but will be supported in near future. Each user profile is mapped to a 32bytes long **`secretkey`** which is generated when the user signs up. Hence all the API requests must contain user's secretkey for authentication purpose. Any API request without valid secretkey will not be serviced. The exact format of the secretkey in the request body is explained in the later sections.

2.3 Request format

SensorAct can receive request data, as body of POST method, only in JSON format, a single JSON object. If the body of the request contains any invalid or multiple JSON objects, then SensorAct will return error message. Following is sample HTTP request with valid JSON object in the body.

```
POST /user/register HTTP/1.1
Host: www.sensroact.com
Content-Length: 63
Content-Type: application/json

{
    "username": "samy1234",
    "password": "mypassword",
    "email": "pandarasamya@iiitd.ac.in"
}
```

Note that `Content-Type` HTTP header field is set with "`application/json`".

2.4 Response format

SensorAct will send response in JSON format. The API response can be either **data** or **status** object. Data objects, for example device profile, will be rendered as JSON string. Status responses will be in the following format for both successful and failure of the API request.

```
{
  "apiname": "apiname",
  "statuscode": <Return code>,
  "message": "<Description of the return code>"
}
```

2.5 Errors

For the successful execution of the request API, **statuscode** attribute in status object will be 0. For failure, statuscode will be a non-zero value and the message attribute will contain the description of the statuscode. Various statuscodes and their corresponding descriptions are explained in detail along with the documentation of each API in the later sections.

3. Broker APIs

3.1 repo/register

URL http://muc.iiitd.com:9001/repo/register

Description Register a new repository profile to the broker.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with following attributes.

Attribute	Required?	Min Length	Max Length	Description
`owner`	Yes	8	20	Owner name (registered user)
`name`	Yes	8	20	Repository name
`URL`	Yes	10	100	URL of the repository

Example JSON:

```
{
  "owner": "samy1234",
  "name": "iiitdrepo",
  "URL": "http://muc.iiitd.com:9001/"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "repo/register",
  "status": "<Success or Failed>",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Success	0	New user profile created: `<username>`
Failed	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	14	Invalid JSON object: <explanation>
	40	Repository already registered: `<name>`

Example JSON Response:

```
{
  "apiname": " repo/register ",
  "status": "Failed",
  "statuscode": 40,
  "description": " Repository already registered: iiitdrepo "
}
```

3.2 repo/list

URL http://muc.iiitd.com:9001/repo/list

Description Retrieves all the repository profiles registered in the broker.

Method GET

Format JSON

Invoked by Application

Parameters No parameter is required.

Returns On success returns all the registered repository profiles as an array of JSON objects. Each Json element contains similar attributes present in the **repo/register** API request format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "repo/list",
  "status": "Failed",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Failed	10	System error
	45	No repository found

Example JSON Response:

```
{
  "apiname": " repo/list ",
  "status": "Failed",
  "statuscode": 45,
  "description": "No repository found"
}
```

3.3 user/register

URL http://muc.iiitd.com:9001/user/register

Description Register a new user profile to the Broker.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with following attributes.

Attribute	Required?	Min Length	Max Length	message
username	Yes	8	20	Username
password	Yes	8	20	Password
email	Yes	8	20	Email address

Example JSON:

```
{
  "username": "samy1234",
  "password": "a1b2@c3d4",
  "email": "pandarasamya@iiitd.ac.in"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "user/register",
  "status": "<Success or Failed>",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Success	0	New user profile created: <username>
	9	Connection refused
Failed	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	14	Invalid JSON object: <explanation>
	20	User profile already exists: <username>

Example JSON Response:

```
{
  "apiname": "registeruser",
  "status": "Failed",
  "statuscode": 20,
  "description": "User profile already exists: samy1234"
}
```

3.4 user/login

URL http://muc.iiitd.com:9001/user/login

Description Login service. It can also be used to get the secretkey of the registered user.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with following attributes.

Attribute	Required?	Min Length	Max Length	message
username	Yes	8	20	Username
password	Yes	8	20	Password

Example JSON:

```
{
  "username": "samy1234",
  "password": "a1b2@c3d4"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "user/login",
  "status": "<Success or Failed>",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Success	0	<secretkey>
Failed	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	14	Invalid JSON object: <explanation>
	15	Login failed: <username>

Example JSON Response:

```
{
  "apiname": "registeruser",
  "status": "Failed",
  "statuscode": 20,
  "description": "User profile already exists: samy1234"
}
```

3.5 device/add

URL http://muc.iiitd.com:9001/device/add

Description Add a new device profile to the repository corresponding to user. A device profile contains one or more sensors and/or actuators. A sensor, in turn, contains one or more channels. Actuator definitions are not yet implemented.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with the following attributes. "–" in front of the attributes corresponds to how many level deeper they are in the JSON hierarchy.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secret key
reponame	Yes	8	20	Repository name
deviceprofile	Yes	NA	NA	Device profile group item
-name	Yes	2	20	Device name
-IP	Yes	NA	NA	Device IP address (IPv4)
-location	Yes	2	50	Location of the device
-longitude	Yes	-180	-180	Longitude in double format
-latitude	Yes	-90	+90	Latitude in double format
-tags	Yes	2	100	Device/Location tags
-sensors	Yes	NA	NA	Array of Sensors (group)
--name	Yes	2	20	Sensor name
--id	Yes	1	1000	Sensor id in Integer format
--channels	Yes	NA	NA	Array of Channels (group)
---name	Yes	2	20	Channel name
---type	Yes	2	20	Channel data type (int, etc)
---unit	Yes	2	20	Channel unit
-actuators	Yes	NA	NA	Array of Actuators (group)
--name	Yes	2	20	Actuator name

Note: At least one sensor or actuator group item must be there in the device profile.

Example device profile JSON

```
{
  "secretkey": "367f95de6d29411b86a933e04ffcd9a7",
  "reponame": "iiitdrepo",
  "deviceprofile": {
    "name": "device1",
    "IP": "192.168.0.7",
    "location": "PhD Lab",
    "tags": "3rd floor; IIIT Delhi",
    "latitude": 0,
    "longitude": 0,
    "sensors": [
      {
```

```
      "name": "temperature",
      "id": 1,
      "channels": [
        {
          "name": "channel1",
          "type": "Double",
          "unit": "Celsius"
        }
      ]
    },
    {
      "name": "Accelerometer",
      "channels": [
        {
          "name": "X",
          "type": "Int",
          "unit": "None"
        },
        {
          "name": "Y",
          "type": "Int",
          "unit": "None"
        },
        {
          "name": "Z",
          "type": "Int",
          "unit": "None"
        }
      ]
    }
  ],
  "actuators": [
    {
      "name": "actuator2"
    }
  ]
 }
}
```


Returns On success or failure, returns a JSON object with following attributes.


```
{
  "apiname": "device/add",
  "status": "<Success or Failed>",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```


Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Success	0	New device profile created: <deviceprofile.name>
Failed	9	Connection refused
	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	12	Unregistered secretkey: <secretkey>
	14	Invalid JSON object: <explanation>
	25	Device profile already exists: <deviceprofile.name>

Note: A device is uniquely identified using only its device name for a particular user.

Example JSON Response:

```
{
  "apiname": "adddevice",
  "status": "Failed",
  "statuscode": 14,
  "description": " Invalid JSON object: … Unterminated object near or … "
}
```

3.6 device/delete

URL http://muc.iiitd.com:9001/device/delete

Description Deletes a previously added device profile from the repository for a particular user.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secretkey
reponame	Yes	8	20	Repository name
devicename	Yes	2	20	Device name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "reponame": "iiitdrepo",
  "devicename": "device1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/delete",
  "status": "<Success or Failed>",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Success	0	Device successfully deleted: <deviceprofile.name>
Failed	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	12	Unregistered secretkey: <secretkey>
	14	Invalid JSON object: <explanation>
	30	Device not found: <deviceprofile.name>

Example JSON response

```
{
  "apiname": "deletedevice",
  "status": "Success",
  "statuscode": 0,
  "description": "Device successfully deleted: device1"
}
```

3.7 device/get

URL http://muc.iiitd.com:9001/device/get

Description Retrieves a previously added device profile from the repository for a particular user.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secretkey
reponame	Yes	8	20	Repository name
devicename	Yes	2	20	Device name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "reponame": "iiitdrepo",
  "devicename": "device1"
}
```

Returns On success returns the requested device profile in JSON format similar to the request format for **adddevice** API. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/get",
  "status": "Failed",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
Failed	9	Connection refused
	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	12	Unregistered secretkey: <secretkey>
	14	Invalid JSON object: <explanation>
	30	Device not found: <deviceprofile.name>

Example JSON response on failure

```
{
  "apiname": "device/get",
  "status": "Failed",
  "statuscode": 30,
  "description": " Device not found: device2"
}
```

3.8 device/list

URL http://muc.iiitd.com:9001/device/list

Description Retrieves (all) the device profiles from the repository for a particular user.

Method POST

Format JSON

Invoked by Application

Parameters Body of the request should contain a single JSON object with the following attributes. If, optional, reponame is given, retrieves device profiles only from the given repository name.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`reponame`	No	8	20	Repository name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the previously added device profiles in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "deletedevice",
  "status": "Failed",
  "statuscode": <Return code>,
  "description": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

Status	Code	message
	9	Connection refused
Failed	10	System error
	11	Validation failed: <attribute names and reasons separated by ;>
	12	Unregistered secretkey: <`secretkey`>
	14	Invalid JSON object: <explanation>
	35	No device found

Example JSON response on success

```
{  "devices": [
     {
       "deviceprofile": {< device profile similar to request format for adddevice API >}
     },
     {
       "deviceprofile": {< device profile similar to request format for adddevice API >}
     }
  ]
}
```

4. Repository APIs

4.1 user/register

URL http://muc.iiitd.com:9000/user/register

Description Register a new user profile to the repository.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with following attributes.

Attribute	Required?	Min Length	Max Length	Description
`username`	Yes	8	20	User name
`password`	Yes	8	20	User password
`email`	Yes	8	20	Email address

Example JSON:

```
{
  "username": "samy1234",
  "password": "mypassword",
  "email": "pandarasamya@iiitd.ac.in",
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "user/register",
  "statuscode": <Return code>,
  "message": "<Description of the return code>"
}
```

Various status codes and their corresponding messages are as follows.

statuscode	message
0	New Userprofile registered: <username>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
20	Userprofile already exists: <username>

Example JSON Response:

```
{
  "apiname": "user/register",
  "statuscode": 20,
  "message": "Userprofile already exists: samy1234"
}
```

4.2 user/login

URL http://muc.iiitd.com:9000/user/login

Description Authenticate the user and returns the sercretkey

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with following attributes.

Attribute	Required?	Min Length	Max Length	Description
username	Yes	8	20	User name
password	Yes	8	20	User password

Example JSON:

```
{
  "username": "samy1234",
  "password": "mypassword",
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "user/login",
  "statuscode": <Return code>,
  "message": "<Description of the return code>"
}
```

Various status codes and their corresponding messages are as follows.

statuscode	message
0	<secretkey>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
21	Login failed: <username>

Example JSON Response:

```
{
  "apiname": "user/login",
  "statuscode": 21,
  "message": "sLogin failed: samy1234"
}
```

4.3 user/list

URL http://muc.iiitd.com:9000/user/list We need think again, whether this API is required or not. Because, returning all registered users in the system is NOT a good idea concerning security issues.

Description Lists all the registered users in the repository

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey

Example JSON:

```
{
  "secretkey": "aaaaabbbbbccccddddeeeeeffffgg",
}
```

Returns On failure, returns a JSON object with following attributes.

```
{
  "apiname": "user/list",
  "statuscode": <Return code>,
  "message": "<Description of the return code>"
}
```

Various status codes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>

Example JSON Response:

```
{
  "usernames":  [“user1”, “user2”]
}
```

4.4 device/add

URL http://muc.iiitd.com:9000/device/add

Description Add a new device profile to the repository corresponding to user. A device profile contains one or more sensors and/or actuators. A sensor, in turn, contains one or more channels. Actuator definitions are not yet implemented.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes. "–" in front of the attributes corresponds to how many level deeper they are in the JSON hierarchy.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secret key
deviceprofile	Yes	NA	NA	Device profile group item
-name	Yes	2	20	Device name
-IP	Yes	NA	NA	Device IP address (IPv4)
-location	Yes	2	50	Location of the device
-longitude	Yes	-180	-180	Longitude in double format
-latitude	Yes	-90	+90	Latitude in double format
-tags	Yes	2	100	Device/Location tags
-sensors	Yes	NA	NA	Array of Sensors (group)
--name	Yes	2	20	Sensor name
--id	Yes	1	1000	Sensor id in Integer format
--channels	Yes	NA	NA	Array of Channels (group)
---name	Yes	2	20	Channel name
---type	Yes	2	20	Channel data type (int, etc)
---unit	Yes	2	20	Channel unit
-actuators	Yes	NA	NA	Array of Actuators (group)
--name	Yes	2	20	Actuator name

Note: At least one sensor or actuator group item must be there in a device profile

Example device profile JSON


```
{
  "secretkey": "367f95de6d29411b86a933e04ffcd9a7",
  "deviceprofile": {
    "name": "device1",
    "IP": "192.168.0.7",
    "location": "PhD Lab",
    "tags": "3rd floor; IIIT Delhi",
    "latitude": 0,
    "longitude": 0,
    "sensors": [
      {
        "name": "temperature",
        "id": 1,
        "channels": [
          {
```

```
            "name": "channel1",
            "type": "Double",
            "unit": "Celsius"
          }
        ]
      },
      {
        "name": "Accelerometer",
        "channels": [
          {
            "name": "X",
            "type": "Int",
            "unit": "None"
          },
          {
            "name": "Y",
            "type": "Int",
            "unit": "None"
          },
          {
            "name": "Z",
            "type": "Int",
            "unit": "None"
          }
        ]
      }
    ],
    "actuators": [
      {
        "name": "actuator2"
      }
    ]
  }
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/add",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	New device added: <deviceprofile.name>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
30	Device profile already exists: <deviceprofile.name>

Note: A device is uniquely identified using only its device name for a particular user.

Example JSON Response:

```
{
  "apiname": "device/add",
  "statuscode": 14,
  "message": " Invalid JSON object: … Unterminated object near or … "
}
```

4.5 device/delete

URL http://muc.iiitd.com:9000/device/delete

Description Deletes a previously added device profile from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`devicename`	Yes	2	20	Device name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "devicename": "device1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/delete",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Device successfully deleted: <`deviceprofile.name`>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
31	Device not found: <`deviceprofile.name`>

Example JSON response

```
{
  "apiname": "device/delete",
  "statuscode": 0,
  "message": "Device successfully deleted: device1"
}
```

4.6 device/get

URL http://muc.iiitd.com:9000/device/get

Description Retrieves a previously added device profile from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`devicename`	Yes	2	20	Device name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "devicename": "device1"
}
```

Returns On success returns the requested device profile in JSON format similar to the request format for **`device/add`** API. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/get",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
31	Device not found: <`deviceprofile.name`>

Example JSON response on failure

```
{
  "apiname": "device/get",
  "statuscode": 31,
  "message": " Device not found: device2"
}
```

4.7 device/list

URL http://muc.iiitd.com:9000/device/list

Description Retrieves all the previously added device profiles from repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secretkey

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the previously added device profiles in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/list",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
32	No device found

Example JSON response on success

```
{
  "devices": [
    {
     "deviceprofile": {< device profile similar to request format for device/add API
>}
    },
    {
     "deviceprofile": {< device profile similar to request format for device/add API
>}
    }
  ]
}
```

4.8 device/search

URL http://muc.iiitd.com:9000/device/search

Description Searches devices – **yet to be implemented**

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
Secretkey	Yes	32	32	Secretkey
?				

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the matching device profiles in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/search",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
32	No device found

Example JSON response on success

```
{
  "devices": [
    {
     "deviceprofile": {< device profile similar to request format for device/add API
>}
    },
  ]
}
```

4.9 device/share

URL http://muc.iiitd.com:9000/device/share

Description Share devices with other registered users – **yet to be implemented**

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
Secretkey	Yes	32	32	Secretkey
devicename				Device
??				

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/share",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Device shared successfully: <deviceprofile.name>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
32	No device found

Example JSON response on success

```
{
  "apiname": "device/share",
  "statuscode": 32,
  "message": " Device not found: device2"
}
```

4.10 device/template/add

URL http://muc.iiitd.com:9000/device/template/add

Description Add a new device template to the repository corresponding to user. A device template contains one or more sensors and/or actuators. A sensor, in turn, contains one or more channels. Actuator definitions are not yet implemented.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes. "–" in front of the attributes corresponds to how many level deeper they are in the JSON hierarchy.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secret key
devicetemplate	Yes	NA	NA	Devicetemplate group item
-isglobal	No	-	-	Is global template?
-name	Yes	2	20	Device name
-IP	Yes	NA	NA	Device IP address (IPv4)
-location	Yes	2	50	Location of the device
-longitude	Yes	-180	-180	Longitude in double format
-latitude	Yes	-90	+90	Latitude in double format
-tags	Yes	2	100	Device/Location tags
-sensors	Yes	NA	NA	Array of Sensors (group)
--name	Yes	2	20	Sensor name
--id	Yes	1	1000	Sensor id in Integer format
--channels	Yes	NA	NA	Array of Channels (group)
---name	Yes	2	20	Channel name
---type	Yes	2	20	Channel data type (int, etc)
---unit	Yes	2	20	Channel unit
-actuators	Yes	NA	NA	Array of Actuators (group)
--name	Yes	2	20	Actuator name

Note: At least one sensor or actuator group item must be there in a device profile

Example device profile JSON


```
{
  "secretkey": "367f95de6d29411b86a933e04ffcd9a7",
  "devicetemplate": {
    "name": "device1",
    "IP": "192.168.0.7",
    "location": "PhD Lab",
    "tags": "3rd floor; IIIT Delhi",
    "latitude": 0,
    "longitude": 0,
    "sensors": [
      {
        "name": "temperature",
```

```
      "id": 1,
      "channels": [
        {
          "name": "channel1",
          "type": "Double",
          "unit": "Celsius"
        }
      ]
    },
    {
      "name": "Accelerometer",
      "channels": [
        {
          "name": "X",
          "type": "Int",
          "unit": "None"
        },
        {
          "name": "Y",
          "type": "Int",
          "unit": "None"
        },
        {
          "name": "Z",
          "type": "Int",
          "unit": "None"
        }
      ]
    }
  ],
  "actuators": [
    {
      "name": "actuator2"
    }
  ]
 }
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/template/add",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	New device template added: <devicetemplate.name>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
35	Device template already exists: <devicetemplate.name>

Note: A template profile is uniquely identified using only its device name for a particular user.

Example JSON Response:


```
{
  "apiname": "device/template/add",
  "statuscode": 11,
  "message": " Invalid JSON object: … Unterminated object near or … "
}
```

4.11 device/template/delete

URL http://muc.iiitd.com:9000/device/template/delete

Description Deletes a previously added device template from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`templatename`	Yes	2	20	Device template name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "templatename": "templatename1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/template/delete",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Device template successfully deleted: <devicetemplate.name>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
36	Device template not found: <devicetemplate.name>

Example JSON response

```
{
  "apiname": "device/template/delete",
  "statuscode": 0,
  "message": "Device template successfully deleted: template1"
}
```

4.12 device/template/get

URL http://muc.iiitd.com:9000/device/template/get

Description Retrieves a previously added device template from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`templatename`	Yes	2	20	Device template name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "templatename": "template1"
}
```

Returns On success returns the requested device template in JSON format similar to the request format for **`device/template/add`** API. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/template/get",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
36	Device template not found: <`devicetemplate.name`>

Example JSON response on failure

```
{
  "apiname": "device/template/get",
  "statuscode": 36,
  "message": " Device template not found: template2"
}
```

4.13 device/template/list

URL http://muc.iiitd.com:9000/device/template/list

Description Retrieves all the previously added device templates from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the previously added device templates in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/template/list",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
37	No device template found

Example JSON response on success

```
{
  "templates": [
    {
     "devicetemplate": {< device template similar to request format for
device/template/add API >}
    },
   {
     "devicetemplate": {< device template similar to request format for
device/template/add API >}
    }
  ]
}
```

4.14 device/template/global/list

URL http://muc.iiitd.com:9000/device/template/global/list

Description Retrieves all global device templates from the repository.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the previously added global device templates in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "device/template/global/list",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
37	No device template found

Example JSON response on success

```
{
  "templates": [
    {
     "devicetemplate": {< device template similar to request format for
/device/template/add API >}
    },
   {
     "devicetemplate": {< device template similar to request format for
/device/template/add API >}
    }
  ]
}
```

4.15 guardrule/add

URL http://muc.iiitd.com:9000/guardrule/add

Description Adds new guard rule to the repository

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secret key
rule	Yes	NA	NA	Guard rule group item
-name	No	?	?	Guard rule name
-description	Yes	?	?	Description
-target Operation	Yes	?	?	Target operation (Read or Write)
-priority	Yes	?	?	Priority of the guard rule
-condition	Yes	?	?	Conditions
-action	Yes	?	?	Action – permit or deny

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
  "rule": {
    "name": "add noise rule",
    "description": "Adds Gaussian noise",
    "target_operation": "READ",
    "priority": 1,
    "condition": "USER.email == *@xyz.edu && TIME == WORKTIME",
    "action": "AddGaussionNoise(70,15)",
  }
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "guardrule/add",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Guard rule added successfully : <rule.name>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
??	Guard rule already exists : <rule.name>

Example JSON Response:

```
{
   "apiname": "guardrule/add",
   "statuscode": 0,
   "message": "Guard rule added successfully: <rule.name>"
}
```

4.16 guardrule/delete

URL http://muc.iiitd.com:9000/guardrule/delete

Description Deletes a previously added guard rule from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`name`	Yes	2	20	Guard rule name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "name": "rule1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "guardrule/delete",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Guard rule successfully deleted: <`guardrule name`>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
??	Guard rule not found: <`guardrule name`>

Example JSON response

```
{
  "apiname": "guardrule/delete",
  "statuscode": 0,
  "message": " Guard rule successfully deleted: rule1"
}
```

4.17 guardrule/get

URL http://muc.iiitd.com:9000/guardrule/get

Description Retrieves a previously added guard rule from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`name`	Yes	2	20	Guardrule name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "name": "rule1"
}
```

Returns On success returns the requested guard rule in JSON format similar to the request format for **`guardrule/add`** API. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "guardrule/get",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
??	Guardrule not found: <`guardrule name`>

Example JSON response on failure

```
{
  "apiname": " guardrule/get",
  "statuscode": ??,
  "message": "Guard rule not found: rule2"
}
```

4.18 guardrule/list

URL http://muc.iiitd.com:9000/guardrule/list

Description Retrieves all guard rules from the repository for a particular user

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the previously added guard rules in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "guardrule/list",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
??	No guard rule found

Example JSON response on success

```
{
  "guardrulelist": [
    {
      <guard rule similar to request format for guardrule/add API >
    },
    {
      <guard rule similar to request format for guardrule/add API >
    }
  ]
}
```

4.19 tasklet/add

URL http://muc.iiitd.com:9000/tasklet/add

Description Adds a tasklet to the repository for a particular user

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secret key
taskletname	Yes	?	?	Tasklet name
desc	Yes	2	100	Description
param	No	?	?	Parameter list (set of key:pair values)
input	No	?	?	Input list (set of key:pair values)
when	No	2	50	Condition clause
execute	Yes	2	1000	Lua script

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
  "taskletname": "monitor ac"
  "desc": "Monitor airconditioner"
  "param": {
    "D1": "user:device:sensor:id",
  },
  "input": {
    "D1": "user:device:sensor:id",
  }
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/add",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Tasklet added successfully : <taskletname>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
??	Tasklet already exists : <taskletname>

Example JSON Response:


```
{
   "apiname": "tasklet/add",
   "statuscode": 0,
   "message": "Tasklet added successfully: <taskletname>"
}
```

4.20 tasklet/delete

URL http://muc.iiitd.com:9000/tasklet/delete

Description Deletes a previously added tasklet from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`taskletname`	Yes	2	20	Tasklet name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "taskletname": "tasklet1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/delete",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Tasklet successfully deleted: <`taskletname`>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
50	Tasklet not found: <`taskletname`>

Example JSON response

```
{
  "apiname": "tasklet/delete",
  "statuscode": 0,
  "message": "Tasklet successfully deleted: tasklet1"
}
```

4.21 tasklet/get

URL http://muc.iiitd.com:9000/tasklet/get

Description Retrieves a previously added tasklet from the repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`taskletname`	Yes	2	20	Tasklet name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "taskletname": "tasklet1"
}
```

Returns On success returns the requested tasklet in JSON format similar to the request format for **`tasklet/add`** API. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/get",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
50	Tasklet not found: <`taskletname`>

Example JSON response on failure

```
{
  "apiname": "tasklet/get",
  "statuscode": 50,
  "message": "Tasklet not found: tasklet2"
}
```

4.22 tasklet/list

URL http://muc.iiitd.com:9000/tasklet/list

Description Retrieves all previously added tasklets from the repository for a particular user

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503"
}
```

Returns On success returns all the previously added taskletss in JSON array format. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/list",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
??	No tasklet found

Example JSON response on success

```
{
  "taskletlist": [
    {
      < tasklet similar to request format for tasklet/add API >
    },
    {
      < tasklet similar to request format for tasklet/add API >
    }
  ]
}
```

4.23 tasklet/execute

URL http://muc.iiitd.com:9000/tasklet/execute

Description Schedule a tasklet for execution in repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`taskletname`	Yes	2	20	Tasklet name

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "taskletname": "tasklet1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/execute",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Tasklet scheduled successfully: `<taskletid>`
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: `<secretkey>`
50	Tasklet not found: `<taskletname>`
??	Unable to schedule tasklet: `<taskletname>`

Example JSON response on failure

```
{
  "apiname": "tasklet/execute",
  "statuscode": 50,
  "message": "Tasklet not found: tasklet2"
}
```

4.24 tasklet/cancel

URL http://muc.iiitd.com:9000/tasklet/cancel

Description Cancel the execution of a previously scheduled tasklet in repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secretkey
taskletid	Yes	2	20	Tasklet Id

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "taskletid": "user_tasklet1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/cancel",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Tasklet cancelled successfully: <taskletid>
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>
51	Tasklet not scheduled: <taskletid>
??	Unable to cancel tasklet: <taskletid>

Example JSON response on failure

```
{
  "apiname": "tasklet/cancel",
  "statuscode": 51,
  "message": "Tasklet not scheduled: user_tasklet1"
}
```

4.25 tasklet/status

URL http://muc.iiitd.com:9000/tasklet/status

Description Retrieves the status of a previously scheduled tasklet in repository for a particular user.

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
`secretkey`	Yes	32	32	Secretkey
`taskletid`	Yes	2	20	Tasklet Id

Example

```
{
  "secretkey": "a3060dd5db9e4b1186c4c425962c6503",
  "taskletid": " user_tasklet1"
}
```

Returns On success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "tasklet/status",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <`secretkey`>
51	Tasklet not scheduled: <`taskletid`>

Example JSON response on failure

```
{
  "apiname": "tasklet/status",
  "statuscode": 51,
  "message": "Tasklet not scheduled: user_tasklet1"
}
```

4.26 data/query

URL http://muc.iiitd.com:9000/data/query

Description Retrieves required readings of a sensor from the repository of a particular user. **Not yet fully implemented. This API undergoes lot of changes.**

Method POST

Format JSON

Invoked by Broker, Application

Parameters Body of the request should contain a single JSON object with the following attributes.

Attribute	Required?	Min Length	Max Length	Description
username	Yes	8	20	Secretkey
devicename	Yes	2	20	Device name
sensorname	Yes	2	20	Sensor name
Sensorid	Yes	1	1000	Sensor Id in Integer format
channel	No	2	20	Channel name
conditions	Yes	NA	NA	Group items
-fromtime	Yes	-	-	From time in Epoch format
-totime	Yes	-	-	To time in Epoch format

Example

```
{
  "username": "samy1235",
  "devicename": "device1",
  "sensorname": "temperature",
  "sensorid": 1,
  "channel": temp,
  "conditions": {
    "fromtime": 1335895438,
    "totime": 1335897438
  }
}
```

Returns On success returns all the requested readings as JSON array of wavesegents. On Failure, returns a JSON object with following attributes.

```
{
  "apiname": "data/query ",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
14	Unregistered username: <username>

4.27 data/upload/wavesegment

URL http://muc.iiitd.com:9000/data/upload/wavesegment

Description Stores a wave segment to the repository sent by a device

Method POST

Format JSON

Invoked by Device

Parameters Body of the request should contain a single JSON object with the following attributes. "–" in front of the attributes corresponds to how many level deeper they are in the JSON hierarchy.
Validation of all these attributes is not yet done due performance issues.

Attribute	Required?	Min Length	Max Length	Description
secretkey	Yes	32	32	Secretkey
data	Yes	NA	NA	Data (group item)
-loc	No	2	50	Device location
-dname	Yes	2	20	Device name
-sname	Yes	2	20	Sensor name
-sid	Yes	1	1000	Sensor Id in Integer format
-timestamp	Yes	NA	NA	In Epoc time format
-sinterval	Yes	NA	NA	Sampling interval
-channels	Yes	NA	NA	Array of Channels (group)
--cname	Yes	2	20	Channel name
--unit	Yes	2	20	Channel unit
--readings	Yes	NA	NA	Array of channel readings

Example

```
{
  "secretkey": " a3060dd5db9e4b1186c4c425962c6503",
  "data": {
    "loc": "00:1E:C0:03:EF:6F:",
    "dname": "RAJESH HEATER",
    "sname": "jPlug",
    "sid": "1",
    "timestamp": 1335873900.55,
    "sinterval": 1,
    "channels": [
      {
        "cname": "RMSV",
        "unit": "Voltage",
        "readings": [
          225.463,
          225.719,
          225.762,
          225.746,
          225.814
        ]
      },
```

```
        {
          "cname": "Energy",
          "unit": "Voltage",
          "readings": [
            20.872,
            20.893,
            20.915,
            20.937,
            20.958
          ]
        }
      ]
    }
}
```

Returns Since the device publishes its readings, sending response to the caller is optional and that can be enabled or disabled. If enabled, on success or failure, returns a JSON object with following attributes.

```
{
  "apiname": "data/upload/wavesegment",
  "statuscode": <Return code>,
  "message": "<Description of the Return code>"
}
```

Various statuscodes and their corresponding messages are as follows.

statuscode	message
0	Wave segment stored successfully
10	System error
11	Invalid JSON object: <explanation>
12	Validation failed: <attribute names and reasons separated by ;>
13	Unregistered secretkey: <secretkey>

Appendix A: Glossary of Terms

Broker The broker is the module in the system which facilitates user profile management, for which it can act as a single point of contact. Depending upon repository settings a broker may also assume the role of device configuration for a particular contributor.

Computational Sensor *A computational sensor is a virtual sensor in which computed aggregated readings from one or more physical sensors are maintained. Ex. Aggregated temperature information in a room over a period of time*

Contributor *A contributor is a special user of our system who uploads his data to a repository.*

Device *A device is a collection of sensors and actuators. In special cases a sensor is a device with no actuator and similarly an actuator may be a device without any sensor. An apt example of a device would be the node installation in the faculty offices which consist of 3 sensors (PIR, Door and temperature) and an actuator (turn on/off an electrical load). A device may contain one or more other devices. For instance a Node consisting of 2 sensors (temperature and presence) and an actuator(fan) is a device which contains 3 sub devices-temperature and presence sensors and fan.*

Device profile *Every device has the following attributes attached to it's profile:*
1. Sensors and its various attributes (name, #channels, data type, etc)
2. Actuators and its various attributes - actuation URL

Guard Rules *Rules defined by the contributor to protect/share the devices.*
Guard rules are the set of rules which are used to guard against certain kinds of actuation. For instance an AC might have guard rules not to cool beyond a certain point. Certain virtual actuators may not allow the individual actuators to be configured beyond a certain point if done individually.

Place tag *A place tag adds context information to a Location (GPS coordinates). Example place tags for PhD lab can be 4th floor, IIIT Delhi, Library building, NSIT campus, etc.*

Repository *A repository is the physical location (server/machine) where a contributor stores his data and his privacy settings.*

Script *Set of commands and/or requests to be issued to data repository. Scripts are associated with Tasks which are schedulable.*

Sensor Owner *Sensor owner is the person/organization who has installed the device and has complete control over the same. For eg. IIIT Delhi is the owner of the sensor node/device which has been deployed in faculty offices. A better term may be to used device owner instead of sensor owner and actuator owner.*

Task *A Task is either an actuation command or a data query. A task is of the form :<time,script> where the time is mentioned in a cron style format and the script may be used to actuate/query the DB*

Time tag *A time tag adds context information to a particular time period. Ex. Time period 9AM to 5PM can be tagged as day time, office hours, etc.*

User/Consumer *A person who wants the use the services of our system. By default a user implies a consumer.*

Username *It is the unique string taken by each user of the system. We may impose certain restriction on this string such as it's maximum length*

Vector actuator *A vector actuator is an actuator wherein all the sub actuating actuators are controlled in a synchronous fashion. For eg control of automobile wheels*

Vector Sensor *A vector sensor is a sensor that has multiple synchronously sampled channels. For eg a 3 axis accelerometer where we synchronously sample the 3 channels(X,Y and Z)*

Virtual Actuator *A virtual actuator is a bundle of physical actuators. It is a concept introduced for easy facilitation of groups of actuators which may have dependency within themselves. For instance an AC may turn on only if the Door is closed.*

Wave segment *A wavesegment is an aggregated set of readings of a device over a period of time.*