# Module Smart-house-REST-API

## Sub-modules

* Smart-house-REST-API.AutoML
* Smart-house-REST-API.Clustering
* Smart-house-REST-API.app
* Smart-house-REST-API.controller
* Smart-house-REST-API.dashboard
* Smart-house-REST-API.db Module Smart-house-REST-API.app ===============================

## Functions

delete\_table(name, token)

Flask app DELETE request that delets whole database table given by name Endpoint data/

Args: name (string): name of the database table that will be dropped token (string): API token

Returns: string: message

get\_data(token)

Flask app GET request that returns all data stored in the database Endpoint data/

Args: token (string): API token

Returns: string: jsonified database content in the form of string

get\_model(name, token)

Flask app GET request that returns parameters of a ml model given by its name Endpoint rules/ Args: name (string): name of a ml model token (string): API token

Returns: string: jsonified ml model in the form of string

get\_models\_names(token)

Flask app GET request that returns all ml model names Endpoint rules/ Args: token (string): API token

Returns: string: jsonified names of ml models in the form of string

get\_tails(size, token)

Flask app GET request that returns last data stored in the database. Number of records is specified by parameter size Endpoint data/

Args: size (string): size of returned data/numer of last records token (string): API token

Returns: string: jsonified database content in the form of string

train\_all\_models()

Scheduled function that trains ml models for all available devices.

train\_clustering()

Scheduled function that fits clustering algorithm

use\_all\_models()

Scheduled function that uses all available ml models on the previous records from database Module Smart-house-REST-API.controller ======================================

## Functions

f1\_score(y\_true, y\_pred, threshold=0.5) : Module Smart-house-REST-API.dashboard ===================================== Module Smart-house-REST-API.db ==============================

## Functions

create\_table(name, columns, database\_name='sensors\_data.db')

Function that table given by name, and columns.

Args: name (string): name of the table that will be added to the database. columns (list): names of columns that will be added, must contain column “data”. Returns: (boolean, string): [description].

drop\_table(name, database\_name='sensors\_data.db')

Function that drops the table given by ‘name’

Args: name (string): name of the table database\_name (str, optional): Name of the database file. Defaults to ‘sensors\_data.db’.

Returns: (tuple): first element of the tuple is a bool or a list of data, second is a message.

get\_column\_names(name, database\_name='sensors\_data.db')

get\_list\_of\_table\_names(database\_name='sensors\_data.db')

Function returning list of table names.

Args: database\_name (str, optional): name of the databse file. Defaults to ‘sensors\_data.db’.

Returns: tuple: (tuple): first element of the tuple is a bool or a list of names, second is a message.

insert\_record\_into\_table(name, data, database\_name='sensors\_data.db')

Function inserting record into table given by name.

Args: name (string): name of the table. data (list): data record. database\_name (str, optional): name of the databse file. Defaults to ‘sensors\_data.db’.

Returns: (tuple): first element of the tuple is a bool, second is a message.

query\_db(query, args=(), one=False, database\_name='sensors\_data.db')

New function that soon will replace all of the other functions in db.py. Performs any SQL query that user want.

Args: query (string): Sql query to be executed args (tuple, optional): Arguments of the query. Defaults to (). one (bool, optional): If True, only the first parameter will be returned, else whole query result. Defaults to False. database\_name (str, optional): Name of the database file. Defaults to ‘sensors\_data.db’.

Returns: list: query result in the form of list

select\_all\_from\_table(name, database\_name='sensors\_data.db')

Function returning all records from table given by name.

Args: name (string): name of the table.

Returns: (tuple): first element of the tuple is either a bool (False), or a list.

select\_tail(name, n\_rows=10, database\_name='sensors\_data.db')

Function returning last ‘n\_rows’ records from the table given by ‘name’

Args: name (string): name of the table n\_rows (int, optional): Size of the tail. Defaults to 10. database\_name (str, optional): Name of the database file. Defaults to ‘sensors\_data.db’.

Returns: (tuple): first element of the tuple is a bool or a list of data, second is a message.