# Tables

## Families

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Notes** |
| **tag** | Genetate by an hash function from “name”  PRIMARY KEY, varchar[12] | Control collision |
| **name** | Varchar[50], family name |  |

## Users

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Notes** |
| **id** | User id, primary key |  |
| **family\_tag** | Foreign key, relationship whit Families table |  |
| **user\_name** | Varchar[50] | This is NOT a UNIQUE KEY.  Control users with the same name only if they belong to the same family |
| **password** | Varchar[50] |  |
| **User\_type** | Foreign key, relationship whit User\_definition table | Default is “1” |

## Users\_definition

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Notes** |
| **id** | primary key |  |
| **name** | Varchar[50] |  |
| **description** | Varchar[150] |  |
| **Access\_level** | int | The lower the number, the deeper is the access permission |

## THINGS

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Notes** |
| **tag** | primary key, varchar[12] | This code identifies the device which send metrics to server |
| **name** | Varchar[50] |  |
| **family\_tag** | Foreign key, relationship whit Families table |  |

# PHP components

## Database()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | void | Initialize private variable $conn |
| **closeConnection()** | void | Close connection |
| **query()** | $querystring | Perform query. Return object array of results or false if query returned an error. No result will return an empty array.  Query with no result (only true or false, such as INSERT) will return true if are successfully performed |

## User()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $username | Initialize user with username, password, email, user type, family tag.  If user does not exist or an error occur during query operation, $this->user = false; |
| **exists ()** | void | If fetching operations from db have been performed successfully, it returns true. Otherwise it returns false. |
| **getUser()** | Void | Return $this->user |
| **controlValidation()** | $password | Check if $password is equal to  $this->user->password (case sensitive) return true if they are equal. |
| **createUser()** | $username, $password, $email, $family | Inserts into table users a new user. Returns false if an error occur. Else it returns true |

## Family()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $tag | Initialize user with username, password, email, user type, family tag.  If user does not exist or an error occur during query operation, $this->user = false; |
| **exists ()** | void | If fetching operations from db have been performed successfully, it returns true. Otherwise it returns false. |
| **getThingList()** | void | Return array of Thing type object that contains all things connected to the family |
| **getUserList()** | Void | Return array of User type object that contains all users connected to the family |
|  |  |  |

## Thing()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $tag |  |
| **exists ()** | void | If fetching operations from db have been performed successfully, it returns true. Otherwise it returns false. |
| **getThing()** | Void | Return $this->thing, which contains:  -tag  -family tag  -name  -user\_type (id)  -access\_level |
| **updateThing()** | $name | Return true if updated |
| **getMetrics()** | Void | Return array with all metrics **only data. Those are not Metric() instance.** |

## EmailSender()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $address | Initialize object with email address |
| **sendFamilyTagEmail()** | $object, $accessKey | Send email with login family tag  **Warning: file\_get\_contents with url that work only on local webserver** |
|  |  |  |

## FileReader()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $path | Initialize object with file path |
| **exists()** | void | Return true if the file exists |
| **Read()** |  | Return a string with the content of file |

## PageBuilder()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $user | Initialize object with user who have opened the webpage |
| **buildMenu()** | $page | Generate Menu items HTML objects according to $this->user access level  $page is used to add class selected-menu .  $page values:   * “USERS\_SETUP\_PAGE” * “THING\_SETUP\_PAGE” |
|  |  |  |

## Metric()

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| **\_\_constructor()** | $id | Init metric object |
| **exists ()** | void | If fetching operations from db have been performed successfully, it returns true. Otherwise it returns false. |
| **Getters()** | void | getId, getName, getThingTag, getUnit.  Return false if metric does not exist. |
| **updateMetrics()** | $name, $unit | Update metric. Return false if metric does not exist or if an error occurs. |
| **createMetric()** | $thing\_tag, $name, $unit | Create metric. Bind a thing to the metric using $thing\_tag |

# Api

## Login

**Request method:** POST;

**Data:** data{user, pass};

### Error

* **User does not exist:** die and return error to index.html
* **Wrong password:** die and return error to index.html

### Success

* **Auth performed:** create session variable $\_SESSION[“user”];

## Signup

**Request method:** POST;

**Data:** data{user, pass, email, family};

### Error

* **Family does not exist:** die and return error to signup.html {status:”error”, error:”error message”, showFamily:true}
* **User already exists**
* **Email is already used**

### Success

* **signup performed:** create session variable $\_SESSION[“user”];

## Signup

**Request method:** POST;

**Data:** data{user, pass, email, family};

### Error

* **User already exists**
* **Email is already used**

### Success

* **signup performed:** create session variable $\_SESSION[“user”], sent email with family tag;

## GetUserThingList

**Request method:** GET;

**Data:** data{ };

### Error

* **Wrong GET param**
* **$\_SESSION[‘user’] does not exist**

### Success

* Echo user’s thing list

If isset $\_SESSION[thingList] and it’s user is equal to $\_SESSION[‘user’]->username return to client the cached thing list instead of reload the whole list from DB

## CacheThingList

**Request method:** POST;

**Data:** data{ list};

Save in $\_SESSION[thingList] current user’s username and his thing list.