**Stations**

* This section includes the station type selection dropdown, a search-bar to search for a specific station or category and a table to display data related to that station such as station name, category and a button to set the limit for that station.
* API for the station category options is: **/stations/options.**
* Key coming as response: **category\_list**.
* The data are stored in the state as **category\_list**.
* API for station list: **/stations/list**. This API gets the list of Stations and its data and is stored in the state in **station\_list**.

1. Category / subcategory selection

* The data will be stored in **category\_selected** and **sub\_category\_selected** in state.
* The default value selected is the category and subcategory in the URL **<Project URL>/settings/stations/?cat=[2]&subcat=[].**
* If no category or subcategory is given in the URL, then the default value selected is Flood.
* The filter is a multi-select type, i.e. multiple categories and subcategories can be selected.
* On clicking a category, that category is selected along with all the subcategories under it.
* If a subcategory is clicked, if it is already unchecked, then it will be checked, and if already checked, it will uncheck.
* Every time the subcategory or category selected is changed, it is reflected in the URL and accordingly the stations in the table are displayed.
* The station\_list is filtered according to the filters applied and stored in **filtered\_stations.**

1. Search

* The result after applying filters can be searched for by category or station name.
* The searched text is stored in the key **table\_search** in the state.
* The list after the search from **filtered\_stations** is stored in **searched\_stations.**
* The searched key is reflected in the URL and the result of the search is displayed in the table.
* URL: **<Project URL>settings/stations/?cat=[2]&subcat=[]&search=demo.**

1. Table

* The table displays the result of the selected categories and sub categories and the searched string.
* The table can be sorted by station name and category in ascending or descending order.
* The sort selection is reflected in the URL with keys **name=asc/ desc** and **category=asc/ desc**.
* URL: **<Project URL>settings/stations/?cat=[2]&subcat=[]&name=asc&category=desc.**
* On clicking on the cog button in the set limit column, the URL is updated with the selected station ID and a popup appears to set the threshold limit for that station.
* API to send the threshold value: **/stations/:id/set\_threshold**
* Keys to be sent in API:
  + id: Station ID
  + threshold\_list: {

param\_key: Key of the parameter for which the threshold object is for

threshold: {Threshold Object}

}

* The URL when the set limit popup is opened is: **<Project URL>settings/stations/:station\_id/limit/?cat=[2]&subcat=[].**
* Show button loader while the API response comes and after getting the success status as response, **:station\_id/limit** is removed from the URL and popup is closed. If the response status is failure, the URL remains same and popup is not closed.