Objective

Create a HTML page with a sensor data chart for one month, similar to line charts described here. Sensor data is available here as a JSON report. Report format is described later in this document.

Use JavaScript to retrieve, format and display data. You are also allowed to use:

- moment.js time conversion library
- plotly.js charting library
- any other third party JavaScript library if needed

Task breakdown

Research

1. JSON data structures
2. browser Fetch API for retrieving data
3. JavaScript Array manipulation functions: map, filter, find
4. moment.js and plotly.js
5. Single Page Applications (SPA)

Main task

1. create a SPA and setup hosting
2. implement the following procedure in JavaScript as a script block in SPA:
   - fetch JSON report, store as local variable
   - get list of sensors in report, allow user to select a sensor from *Dropdown List*
   - for selected sensor, get hourly data from report and display as *Line Chart* (convert timestamp to human-readable date)

Optional

1. add *Date Pickers* for start and end, allow user to select interval for *Line Chart*
2. allow user to select multiple sensors, and display multiple *Lines* on same *Chart*
3. any other user-experience enhancement

JSON report description

A report is a tabular structure with rows and columns, similar to a *Data Frame* in **R**. Each row contains data for all columns. In JSON it is an object with following properties:

- `"features"` : (*Array* of *String*) sensor capabilities
- `"sensors"` : (*Array* of *Object*) this represents **columns**, each of its n elements is a sensor object with properties:
  - `"id"` : unique sensor ID (*String*)
  - `"type"` sensor type (`T` =temperature, `RH` =relative humidity, etc.) (*String*)
- `"virtual"` : (*Array* of *Object*) similar to above, but for virtual (calculated, not measured) sensors
- `"data"` : (*Array* of *Array*) this represents **rows**, each row is an *Array*, described below:
  - (*Array* = [*Integer*, *Object*]) 2-element *Array*, first element (row index) is hour timestamp and second element (data) is an *Array*, described below:
  - (*Array* of *Object*) n-element *Array*, each of its n elements is a hourly data object for n-th sensor from `"sensors"` *Array*, with following properties:
    - `"value"` : average value for hour beginning on row index timestamp (*Double*)
    - `"value-n"` : total number of measurements in that hour (*Integer*)
    - `"value-max"` : maximum measured value in that hour (*Double*)
    - `"value-min"` : minimum measured value in that hour (*Double*)
    - `"value-unit"` : measurement unit (*String*)
    - `"value-valid"` : average of validated values (*Double*)
    - `"value-valid-n"` : total number of validated values (*Integer*)
    - `"invalidity-code"` : code for cause of invalid measurement, if any measurement was invalid during the hour (*Integer*)

JSON report example

```
{
  "features": [
    "airq"
  ],
  "timezone": "Europe/Vienna",
  "sensors": [
    {
      "id": "5EN4MXnJR8",
      "type": "T"
    },

    ...

  ],
  "virtual": [
    {
      "id": "9yLA6vvNvm",
      "type": "usaqi"
    },

    ...

  ],
  "data": [
    [
      1612105200000,
      [
        {
          "value": 4.446175708232065,
          "value-n": 239,
          "value-max": 4.6420001029968265,
          "value-min": 4.278000164031982,
          "value-unit": "°C",
          "value-valid": 4.446175708232065,
          "value-valid-n": 239,
          "invalidity-code": 0
        },

        ...

      ]
    ]
  ]
}
```