Problem Overview

We at Sheru collect data from hundreds of IoT devices everyday. The type of data collected and its source differ from device to device. An example is a GPS enabled device connected to a battery which reports battery location data, battery percentage and voltages etc. Here is the link of such an IoT device called Xenergy : http://13.233.13.254:2222/xenergyData.json which collects data from the battery every second and it creates a packet along with the GPS data and sends the data through a TCP socket to the server.

You are required to :

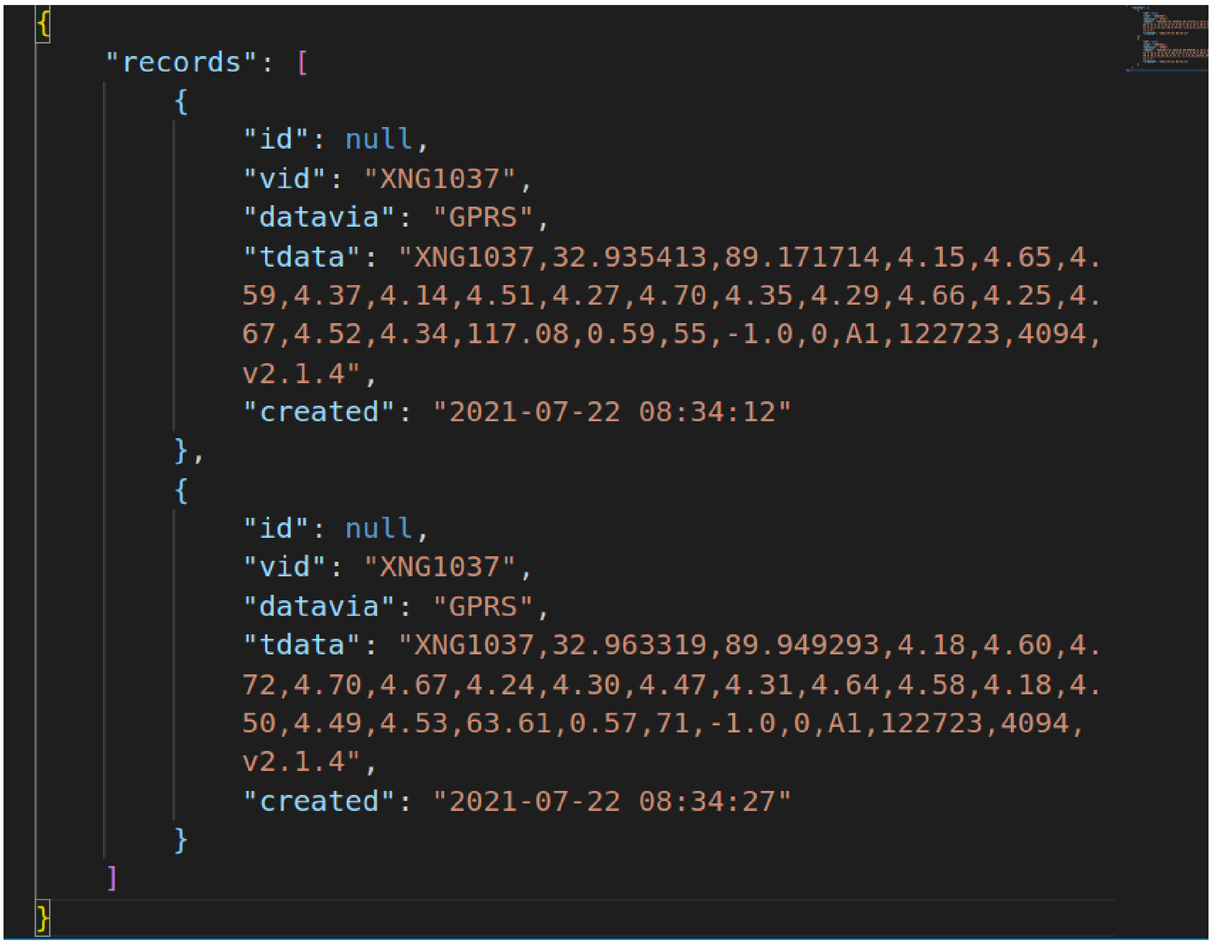
1. Create a server which accepts socket connections from clients and which parses the data.
2. Scrape the data from the given url, check if the data is fresh and different from your previously recorded data and then emulate an IoT device by establishing a socket connection with the server and sending it the data packet and then closing the socket connection. If you find any fresher data on the url, you are required to constantly send the server the data and repeat this process, thus emulating a real time IoT device.
3. Both the server and client processes (emulated IoT device) should be fault tolerant, save any errors to a log and restart incase of any fault.
4. After parsing the data at the server, save this data along with the time when this data was received, to a database best suited for storing data of this type (so that you can also do real time analytics on this gathered data).
5. Create real time alerts for this IoT device and battery in case the battery goes below 20 % or the battery starts discharging (the value of current becomes negative) or the pack voltage shoots across 100 mV. Then, show these alerts and the data received from the device in realtime on a webpage.
6. Create a webpage from where you can issue commands to the IoT device to turn it off remotely.

Make sure you have a plan of action in hand for achieving the given objectives. Even if you are not able to achieve all the objectives in the specified time period, you should have outlined the methodology and prepared relevant reasoning which you plan to undertake for achieving the objectives.

The explanation of the data format, various fields and parsing logic is explained in the subsequent sections.

Data Format

The data hosted at http://13.233.13.254:2222/xenergyData.json will be in a json format and will have a records key having an array of objects:



```
{
    "records": [
        {
            "id": null,
            "vid": "XNG1037",
            "datavia": "GPRS",
            "tdata": "XNG1037,32.935413,89.171714,4.15,4.65,4.
            59,4.37,4.14,4.51,4.27,4.70,4.35,4.29,4.66,4.25,4.
            67,4.52,4.34,117.08,0.59,55,-1.0,0,A1,122723,4094,
            v2.1.4",
            "created": "2021-07-22 08:34:12"
        },
        {
            "id": null,
            "vid": "XNG1037",
            "datavia": "GPRS",
            "tdata": "XNG1037,32.963319,89.949293,4.18,4.60,4.
            72,4.70,4.67,4.24,4.30,4.47,4.31,4.64,4.58,4.18,4.
            50,4.49,4.53,63.61,0.57,71,-1.0,0,A1,122723,4094,
            v2.1.4",
            "created": "2021-07-22 08:34:27"
        }
    ]
}
```

The explanation of the keys and their values in one of the objects is as follows :

1. "vid" - depicts the device IMEI
2. "tdata" - this is the GPRS packet sent by the IoT device accumulating the location and battery data to the server
3. "created" - time when this data was created by the IoT device (you are required to save this time alongwith the time when you received the data at the server in your database)

tdata

tdata is a string containing different values separated by commas. The parsing of "tdata" is as follows :

tdata =
“XNG1037,32.428320,89.015770,4.09,4.14,4.12,4.18,4.11,4.20,4.04,4.16,4.15,4.16,4.07,4.16,4.20,4.08,4.13,57.86,-0.57,93,-1.0,0,A1,122723,4094,v2.1.4";

XNG1037 : device IMEI - string

32.428320 : latitude - float

89.015770 : longitude - float

4.09 : cell 1 voltage (in mV) - float

4.14 : cell 2 voltage (in mV) - float

4.12 : cell 3 voltage (in mV) - float

4.18 : cell 4 voltage (in mV) - float

4.11 : cell 5 voltage (in mV) - float

4.20 : cell 6 voltage (in mV) - float

4.04 : cell 7 voltage (in mV) - float

4.16 : cell 8 voltage (in mV) - float

4.15 : cell 9 voltage (in mV) - float

4.16 : cell 10 voltage (in mV) - float

4.07 : cell 11 voltage (in mV) - float

4.16 : cell 12 voltage (in mV) - float

4.20 : cell 13 voltage (in mV) - float

4.08 : cell 14 voltage (in mV) - float

4.13 : avg cell voltages (in mV) - float

57.86 : pack voltage (in mV) - float

-0.57 : current (in mAmp, negative sign indicates battery is discharging while positive indicates it is charging) -float

93 : SoC or battery percentage (in %) - integer

All the other remaining fields are irrelevant and you are not required to insert those fields into the database.