Programme pour capteurs

Ce programme permet la surveillance de votre serre via des capteurs connectés.

Prérequis :

InfluxDB

Grafana

NodeJS

VisualStudioCode

Le dossier « tp-iot-main »

Installation :

1 - Assurez-vous d'avoir téléchargé le dossier « tp-iot-main » et de l'intégrité des fichiers qu'il contient. Créez un dossier « Middleware » dans un dossier avec tp-iot-main.

2 - Ouvrir une console dans visual code dans le fichier middleware et faire *npm init* puis *npm install express –save* et enfin *npm install express*.

3 - Toujours dans Middleware créer un fichier "apps.js" et copier-y le code de calcul suivant:

```
import express from "express";

const app = express()

const port = 3003

'use strict'

  import { InfluxDB, Point } from '@influxdata/influxdb-client'

  / Environment variables /

  const url = "http://localhost:8086/"

  const token =
"SnUR_GGmvagRmg8hZanjskM6mtXaIB-MxgYJRfUbstvKNZhevzQlvXY4KjCHJ6D2wcuzFoWbHFeYUp4KSsTtLA=="
```

```
  const org = "585f96e557294739"
  const bucket = "df00a0325a6f8c15"
  const influxDB = new InfluxDB({ url, token })
  const writeApi = influxDB.getWriteApi(org, bucket)


  writeApi.useDefaultTags({ region: 'west' })
app.post('/api/humidity', (req, res) => {
    let data = "";
    req.on('data', (chunk) => {
        data += chunk;
      });
      req.on('end', () => {
          const frame = JSON.parse(data);
          const code = parseInt("0x" + frame.data.substr(0,2))/10;
          const value = parseInt("0x0" + frame.data.substr(2, 4))/10;
          let alert = "";
          if (frame.data.length > 6) {
              alert = parseInt("0x" + frame.data.substr(6, 4))
          };
          const point1 = new Point('humidity')
            .tag('sensor_id', 'TLM01')
            .floatField('humidity', value)
            .intField('code', code)
            writeApi.writePoint(point1)
            writeApi.flush().then(function () {
                console.log("Write Finished");
             })
      });
})
app.post('/api/temperature', (req, res) => {
    let data = "";
```

```
    req.on('data', (chunk) => {
      data += chunk;
    });
    req.on('end', () => {
      const frame = JSON.parse(data);
      const code = parseInt("0x" + frame.data.substr(0,2))/10;
      const value = parseInt("0x0" + frame.data.substr(2, 4))/10;
      let alert = "";
      if (frame.data.length > 6) {
        alert = parseInt("0x" + frame.data.substr(6, 4))
      };
      const point1 = new Point('temperature')
       .tag('sensor_id', 'TLM01')
       .floatField('temperature', value)
       .intField('code', code)
       writeApi.writePoint(point1)
       writeApi.flush().then(function () {
         console.log("Write Finished");
        })
    });
  })
app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

4 - Ensuite dans le terminal du dossier Middleware télécharger TypeScript à l'aide ces deux commandes:

- *npm i -g typescript && npm i --save-dev @types/node*
- *tsc --init*

5 - Une fois ces deux commandes exécuter, installer des dépendances pour java script à l'aide de ces deux commandes:

- *npm install --save @influxdata/influxdb-client*
- *npm install --save @influxdata/influxdb-client-apis*

6 - Une fois que vous avez exécuté influxdb en vous connectant sur ce lien : http://localhost:8086, vous pouvez vous inscrire. (faites attention à ne pas oublier tout ce que vous avez mis lors de votre inscription).

7 - Puis allez dans load data et Buckets et créer un buckets, ensuite allez dans API tokens et créez un token qui aura seulement le droit d'écrire sur le bucket que vous venez de créer et enfin créez un autre API token avec les droits d'administrateurs. (Prenez garde à bien sauvegarder vos tokens).

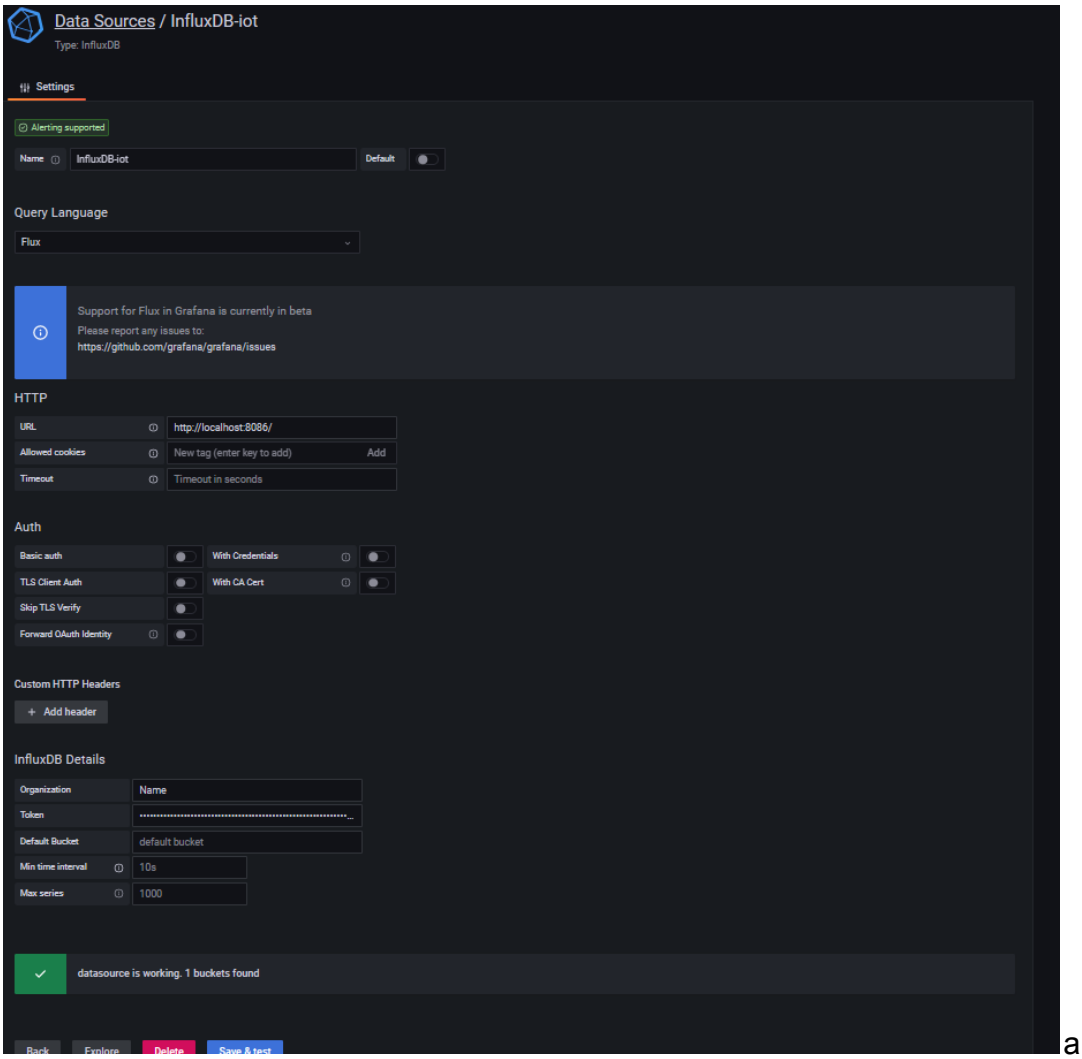
8 - Une fois cela fait, il faut aller dans apps.js et mettre dans *const token* = le token admin généré précédemment et dans *const bucket* = l'ID du bucket que vous avez créé. et enfin mettre dans *const org* = ce qui après /orgs/ dans l'adresse du site exemple: http://localhost:8086/orgs/585f96e557294739/load-data/tokens donc j'écris 585f96e557294739.

9 - Ensuite allez sur Graphana en écrivant localhost:3000 sur votre navigateur. Ceci fait créez vous un compte.

10 - Sur Graphana faite add your first data source et sélectionner InfluxDB.

11 - Écrivez un nom que vous retenez dans query Language sélectionner Flux et comme URL écrivez l'url de votre influxdb, donc normalement http://localhost:8086/ et tout en bas dans Organisation écrivez le nom de l'organisation que vous avez écrit lors de l'inscription à votre influxdb. Écrivez le token que vous avez fait qui donne accès seulement au bucket en écriture.

Cela devrait ressembler à peu près à cela:



a.

12 - dans Humidity.js et dans temperature.js replacer *const ENDPOINT = 'http://localhost:8000/api/humidity'* par *const ENDPOINT = 'http://localhost:3003/api/humidity'* .

13 - Dans package.json de Middleware après *"main": "index.js",* il faut mettre *"type": "module",*

14 - Ouvrir un terminal situé dans middleware et dans tp-iot-main et faire *npm i* dans les deux.

15 - Dans la console de middleware, mettre *npm run apps.js* cela lui permettra d'écouter sur le port.

16 - et dans la console de tp-iot-main mettre *npm run sensors* ce qui permettra de le faire envoyer des paquets depuis les capteurs simulées.

17 - Enfin allez dans graphana et faite new dashboard et add new panel et sélectionner la data source que vous avez configuré plus tôt, et copiez ce code :

```
(bucket: "TP IOT")
  |> range(start: -1h)
  |> filter(fn: (r) =>
    r._measurement == "humidity" and
    r._field == "humidity"
  )
```

N'oubliez pas de remplacer TP IOT par le nom de votre bucket, celui-ci est pour humidity mais si vous voulez les graphiques de température remplacer juste “humidity” par “temperature”.
Vous avez maintenant les graphs, ceux ci devrait ressembler à ça:

N’oubliez pas que vous pouvez régler les alertes et seuils montrés sur le graphique selon vos besoins.

Merci de l’avoir lu jusqu’aux bout.