Projet détecteur de densité de présence

Contexte

Motivations

Dans le contexte actuel de crise sanitaire due au virus du covid-19, la qualité de l'air, l'aération des bâtiments et la densité de personnes présentes dans les salles est un problème qu'il faut surveiller. Nous pouvons d'ailleurs étendre la réflexion au fait que même en dehors de cette période de crise sanitaire, il est important de réduire au maximum les attroupements.

Cependant, une des préoccupations importantes que nous avons est la sobriété en matériel et en énergie du système. Cette problématique est commune à tous les systèmes intégrant une partie IOT.

Le développement durable est défini comme l'intersection des problématiques écologique, sociale et économique. Or l'épidémie qui nous touche est un problème dans ces trois domaines. En effet l'impact d'une pandémie sur l'économie et l'aspect social d'une crise sanitaire est évident, et l'utilisation d'un système IOT soulève obligatoirement des préoccupations écologiques. Les enjeux d'un tel système sont donc :

- Limiter les potentiels confinement et travail à distance, qui impactent la productivité dans tous les secteurs ;
- Limiter la propagation d'un virus potentiellement mortel ;
- Limiter l'impact environnemental (fabrication, installation, utilisation et fin de vie) du système.

Périmètres

Périmètre d'action

Ce projet est avant tout une preuve de concept, nous voulons étudier la faisabilité et la viabilité d'une telle infrastructure. Pour ce faire, nous limitons l'étude du système dans le seul bâtiment de l'IM²AG a Grenoble.

Technique

Nous simulons les différents capteurs dont nous aurons besoin par d'autres capteurs ou drivers. Le serveur de données sera lui aussi simulé par un serveur local sur une machine de développement.

Etude de marché

Il existe plusieurs entreprises et solutions qui pourraient être utilisées pour réaliser ce système de surveillance. Notamment la société Quantaflow, qui propose des services précisément de surveillance d'affluence.

https://www.quantaflow.com/fr/accueil/

Chaîne de valeur

Acquisition

L'objectif ici est donc de partir de données issues de la présence humaine nous avons donc pensé à analyser différents niveaux d'élément comme le CO2 ou la chaleur qui peuvent varier en fonction du nombre de personnes présentes. Il existe plusieurs types de capteurs pour acquérir ces données. Notre besoin est assez limité : il s'agit d'une dizaine de capteurs tout au plus. De plus, ces capteurs sont en général peu gourmand en énergie. A l'exception de caméras (infrarouge ou détection de forme/ mouvement), tous les capteurs peuvent être autonomes en énergie pendant un temps conséquent.

Dans un premier temps, et par souci de simplicité, nous opterons pour des capteurs infrarouge (et non des caméras infrarouge). Ils sont petits et autonomes.
Pour le CO2 les capteurs infrarouge utilisent la techno NDIR qui est un des principaux outils pour mesurer le niveau de CO2 grâce à la spectrophotométrie, en fonction de l'absorption de rayons infrarouge par l'air traversée.

Contexte COVID, ces capteurs seront simulés par un script python, envoyant des données à la plateforme Agora régulièrement.

Transport

La question du transport est sujette à débat. D'un côté nous pouvons utiliser à notre avantage le fait que nos capteurs sont installés dans un bâtiment disposant d'électricité et d'un réseau intégré. Dans ce cas, nous pouvons envisager des capteurs filaires, peu gourmands et assez rapides. Cependant le nombre de capteurs et leur positionnement rendent cette installation plus complexe que l'utilisation de capteurs sans fils. Les capteurs que nous utilisons sont de toute façon assez peu énergivores et nous pouvons espérer une autonomie assez grande.

Dans tous les cas, les sensors communiqueront régulièrement par MQTT vers un broker central.

Stockage

Ce broker enverra ensuite ces données sur le serveur applicatif qui les stockera dans une base de données spécialisée (on peut penser à des bases temporelles comme InfluxDB).

Ce type d'architecture et de stockage des données est motivée par la nature même du système de détection de densité de personnes. Ces données ont pour but d'être analysées et visualisées sur une interface la plus facile possible, par des personnes non techniques. Ces données seront donc utilisées par une application client léger (cf. Valorisation) qui s'interface plus simplement avec ce genre de bases.

Valorisation

Traitement

Les données brutes ne sont pas très utiles dans notre cas. Une étape de traitement des données sera certainement nécessaire. Dans le cadre de notre POC nous ne réalisons qu'un traitement superficiel à but démonstratif. Pour un développement réel les traitements réalisées devront être réfléchi en fonction des besoins en visualisation du client (cf. Visualisation)

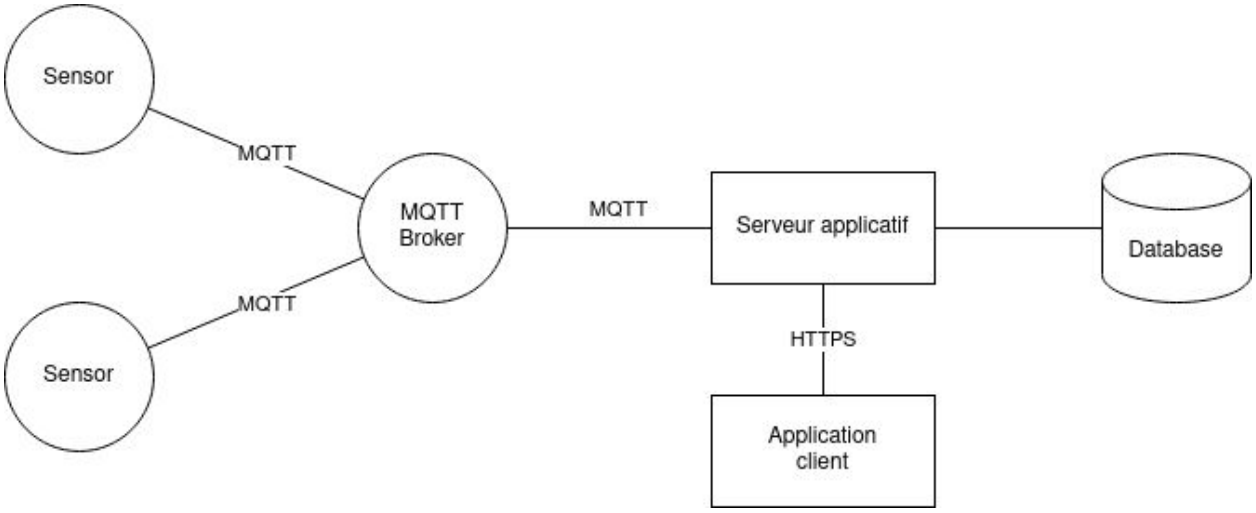
Visualisation

La visualisation des données est similaire au traitement. Nous ne réaliserons ici presque aucune visualisation à part une visualisation brute des données. Dans un contexte de réalisation réel, cette partie sera l'objet d'un dialogue avec les utilisateurs, et serait à la base de la réflexion de tout le projet.

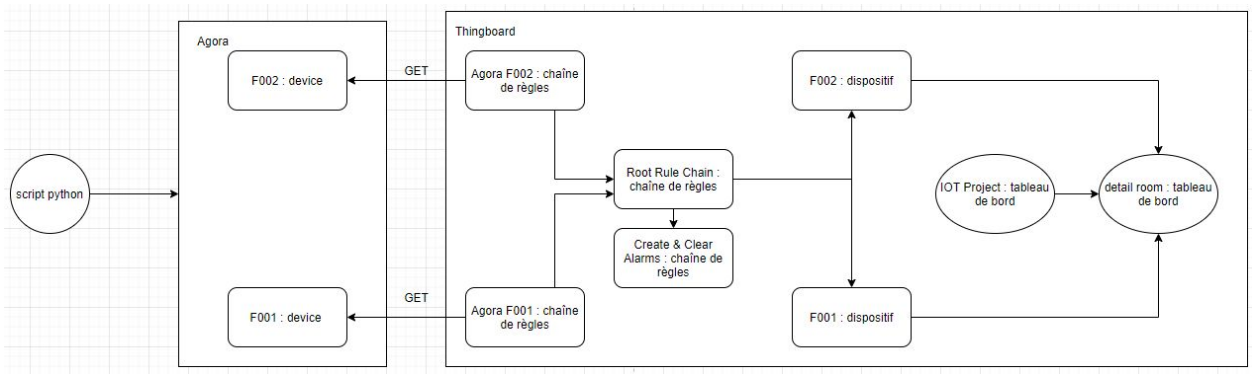
Spécifications technique

Architecture

Nous avons dans un premier temps pensé à une architecture réellement déployable. Dans cette architecture nous avons les capteurs qui communiquent avec le broker mqtt. Les données sont ensuite transmises au serveur applicatif qui les stocke dans la base de données. Ces informations sont ensuite servies par ce même serveur à l'application client qui gère l'affichage sur des terminaux mobiles.



Cependant l'épidémie qui nous touche tous actuellement nous a contraint à abstraire beaucoup d'aspect de cette architecture. Le schéma physique que nous avons choisi de réaliser est le suivant :



La plateforme Agora et le script home made simulent les capteurs, alors que la plateforme Thingboard simule le serveur applicatif ainsi que l'application client.

Spécification de développement

Plateformes

Hors contexte de télétravail nous pensions réaliser un petite application Java/Angular et donc naturellement avec les capteurs physique également. Nous avons donc bifurqué sur les technos Agora et Thingboard.

1) Agora

Agora nous sert à récupérer des données via deux capteurs simulés par des devices, techniquement on a profité de cela pour simuler des capteurs qui récupèrent tous les éléments pour une seule salle à la fois.

Exemple, le device F001 d'Agora reçoit les données de CO2, température, … en même temps.

Pour avoir un minimum de réalité nous avons réalisé un script bash envoyant des nouvelles entrées aux devices Agora de façon régulière (5sec d'intervalles).

2) Thingboard

Thingboard lui est composé de trois chaînes de règles de notre conception, deux d'entres elle nous permettent de récupérer les données des devices Agora de façon individuelle, ne parvenant pas dans le script "transform" à orienter les données vers les dispositifs qui leur sont liés nous avons dû faire deux règles. La dernière chaîne de règle vise à créer une alarme lorsque le niveau d'occupation dépasse 50%. (Annexe 1)

On retrouve nos deux salles au travers des deux dispositifs que nous avons créé, ces derniers sont donc appelés dans le tableau de bord (Annexe 2)

Securité

L'aspect sécurité est comme toujours un point particulièrement important. Le niveau de risque ainsi que la sensibilité sont des valeurs à revoir. Sans utilisation de caméra, l'aspect vie privée des personnes n'est pas menacé. Cependant, certaines attaques peuvent être anticipées comme le déclenchement d'alertes alors qu'il n'y en a pas eu, pour évacuer une partie du bâtiment, ou interrompre un examen par exemple. Le niveau de sécurité attendu n'est cependant pas extrêmement élevé. En effet le système ne concerne pas de zones sensibles, et les capteurs ne sont pas directement reliés à internet donc ont peu d'intérêt pour les botnets.

Il faudra donc apporter une attention particulière à la sécurisation du système. Notamment du broker qui centralise toutes les données. Les capteurs utilisant des liaisons sans fil, il y a plusieurs attaques possibles via ce bias. De plus, le système entier est connecté à internet. Là encore le broker est la porte d'entrée sur internet. Il est donc important de le sécuriser.

La liaison entre le broker et le serveur applicatif est du même type qu'entre les sensors et le broker, nous pouvons mettre en place la même sécurité.

La sécurité de la base, du client et du serveur applicatif est quant à elle assez courante. L'utilisation d'un API restful par exemple, l'utilisation d'un framework, d'HTTPS et isoler la base de données sont des pratiques qu'il faudra mettre en place.

Ressources

Lien du git : https://gitlab.com/mendesj/iot2020.git

Annexes

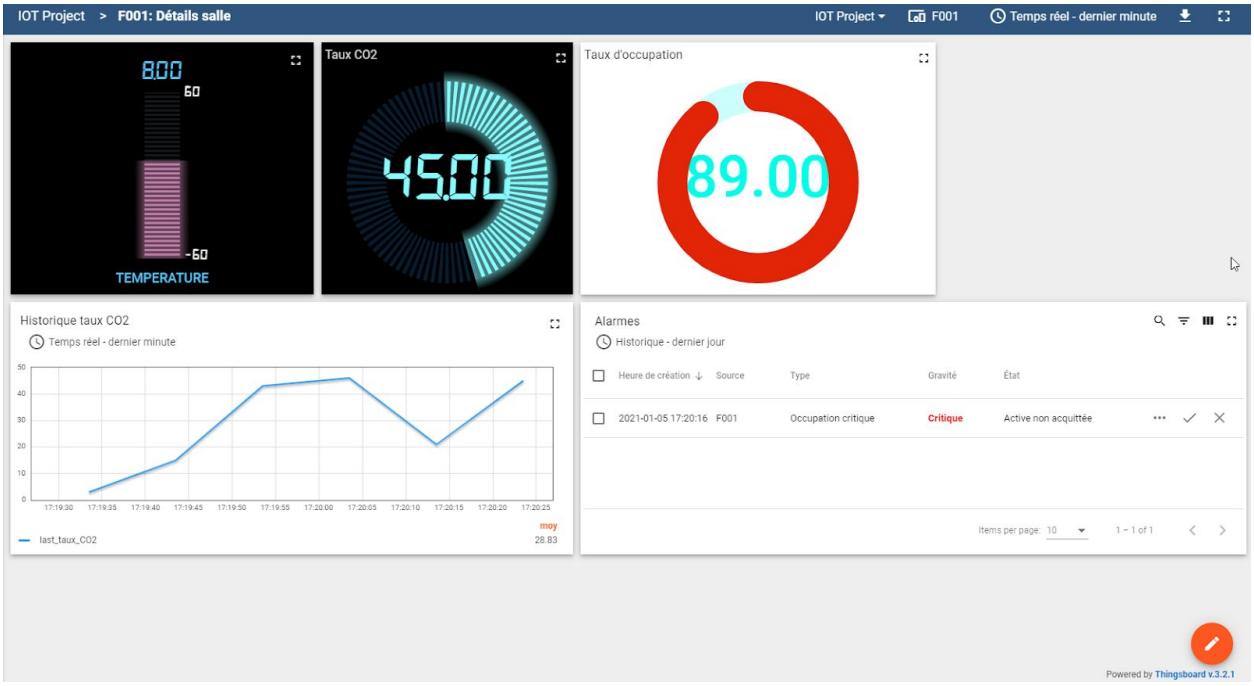
Annexe 1

Script des balises transform des chaînes de règle Thingboard :

```
// Device name
capteurCO2DeviceName = "F001@Bruno_Vernay.etud21";

// Device property name
capteurCO2DevicePropertyName = "taux_CO2";
capteurTemperatureDevicePropertyName = "temperature";
capteurOccupationDevicePropertyName = "taux_occupation";

// Get Device data
capteurCO2DeviceDataList = msg.result.filter((res) => res
    .device == capteurCO2DeviceName);

// Get capteur CO2 last data
last = capteurCO2DeviceDataList.length - 1;
metadataret = {};
capteurCO2LastValue = capteurCO2DeviceDataList[last].data;

// Get all capteur CO2 data
capteurCO2AllValues = capteurCO2DeviceDataList.map((res) =>
    res.data);

// Map result
var mapCapteurCO2LastValue;
var mapCapteurTemperatureLastValue;
var mapCapteurOccupationLastValue;
for (const [key, value] of Object.entries(
        capteurCO2LastValue)) {
    if (key === capteurCO2DevicePropertyName) {
        mapCapteurCO2LastValue = value;
    }
    if (key === capteurTemperatureDevicePropertyName) {
        mapCapteurTemperatureLastValue = value;
    }
    if (key === capteurOccupationDevicePropertyName) {
        mapCapteurOccupationLastValue = value;
    }
}
```

```
// Property in thingboard
msgret = {
    "last_taux_CO2": mapCapteurCO2LastValue,
    "temperature": mapCapteurTemperatureLastValue,
    "taux_occupation": mapCapteurOccupationLastValue
};

return {
    msg: msgret,
    metadataret: metadataret,
    msgType: msgType
};
```

Annexe 2



Capture d'écran du tableau de bord

Annexe 3

Script d'envois des données simulées des capteurs :

```
import requests
import json
import random
```

```
import time



headers = {
   "carriots.apikey" :
"6b22d32723c4c09f631d2aae72e1efddccd3eab7fba28c661e7504fdfeaa8a77",
   "Content-Type" : "application/json"
}

for x in range(10):
   payloadF001 = {
      "protocol": "v2",
      "checksum": "",
      "device": "F001@Bruno_Vernay.etud21",
      "at": "now",
      "data": {
         "taux_CO2": random.randint(0, 100),
         "temperature": random.randint(-10, 40),
         "taux_occupation": random.randint(0, 100)
      }
   }

   payloadF002 = {
      "protocol": "v2",
      "checksum": "",
      "device": "F002@Bruno_Vernay.etud21",
      "at": "now",
      "data": {
         "taux_CO2": random.randint(0, 100),
         "temperature": random.randint(-10, 40),
         "taux_occupation": random.randint(0, 100)
      }
   }

   url = "http://api.iotagora.net/streams"
   r = requests.post(url, data=json.dumps(payloadF001), headers=headers)

   r = requests.post(url, data=json.dumps(payloadF002), headers=headers)

   time.sleep(5);
```