

# De l'IoT aux systèmes Cyber- Physiques

PROJET :

« Système de Génération de tournées  
de collecte Optimisées »



*Projet Réalisé par :*

Saif Bouhamed  
Florence Noé  
Takroun Momen

*Projet encadré par :*

Mr Gérald Rocher

## **Table des matières :**

1	Description du système .....	3
1.1	Identification des besoins capteurs .....	3
1.2	Caractéristiques des capteurs .....	4
2	Sources d'incertitude.....	8
3	Traitement.....	10
3.1	Fusion des données (FOG) .....	10
3.2	Annotation des données capteurs (Edge) (Données d'exemples doivent être cohérentes avec l'archi data ingestion pipeline) .....	11
3.3	Annotation des anomalies capteurs (FOG) .....	11
3.4	Filtrage (pas de filtre passe-bas finalement) (Edge) .....	12
4	Data Ingestion Pipeline .....	13
4.1	Pipeline des données capteurs.....	13
4.1.1	Technologies utilisées .....	13
4.1.2	Explication du flux des données capteurs .....	15
4.2	Pipeline des reports/ticket d'anomalie des utilisateurs .....	17
4.2.1	Explication de la pipeline des reports/ticket d'anomalie des utilisateurs .....	17
4.3	Pipeline d'ajout des utilisateurs .....	18
4.3.1	Explication du pipeline d'ajout d'utilisateurs .....	18

# 1 Description du système

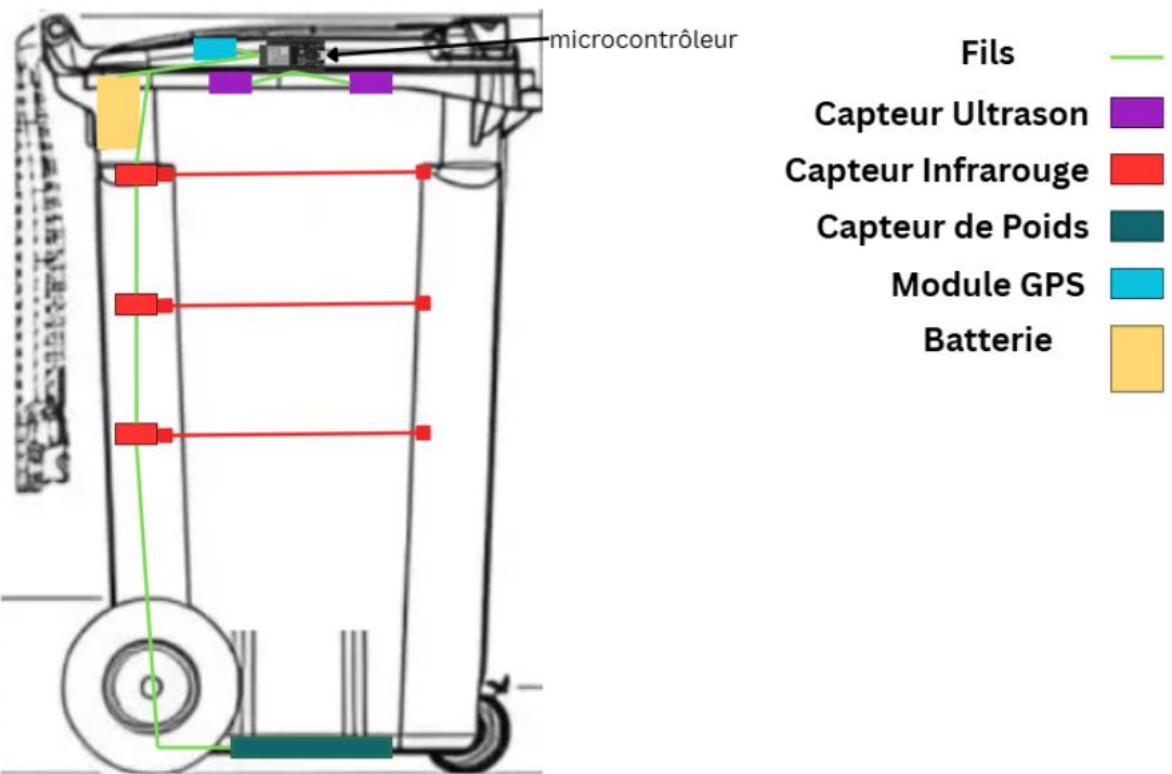


Figure 1 :schéma de la poubelle

## 1.1 Identification des besoins capteurs

Pour déterminer efficacement si la poubelle est pleine, nous avons choisi les capteurs suivants :

- **Mesure du taux de remplissage :**
  - Capteur utilisé : ultrason.
  - Permet de déterminer la distance entre le couvercle et la surface des déchets.
- **Vérification du dépassement d'un seuil par les déchets :**
  - Capteur utilisé : infrarouge.
  - Placé sur les côtés de la poubelle.
  - Sert de validation complémentaire pour l'ultrason.
- **Mesure du poids total du contenu :**
  - Capteur utilisé : cellule de charge.
  - Permet de vérifier la cohérence entre le remplissage et le poids, afin de détecter des anomalies (poids de 0kg alors que poubelle remplie) et de prévoir le trajet en fonction de la masse maximale d'un camion.

- **Localiser la poubelle sur la carte :**
  - Capteur utilisé : module GPS dans le microcontrôleur
  - Afin de pouvoir localiser la poubelle de manière efficace sur la carte et pouvoir générer un trajet cohérent
- **Communiquer avec l'IoT Gateway :**
  - Module WIFI
  - Permet de se connecter au wifi du particulier et communiquer avec l'IoT Gateway. On a fait ce choix car on envoie des données de temps en temps, on n'a donc pas à faire attention à la forte consommation du wifi. Nous aurions pu prendre LoRa mais faire avec le wifi du particulier réduit les coûts économiques.

Les avantages de la combinaison des 3 premiers capteurs sont :

- La validation croisée pour obtenir un taux de remplissage fiable
- Détection des anomalies ou incohérences dans les mesures

## 1.2 Caractéristiques des capteurs

Pour choisir ses caractéristiques, nous avons regardé nos besoins (nous n'avons pas besoin d'une précision extrême dans notre cas) mais aussi ce qui se fait sur le marché, par exemple HC-SR04 pour l'ultrason ou HX711 pour le poids ou NEO-6M pour le gps.

### 1. Capteur ultrason

- **Type :** digital
  - Fournit directement la distance et est facile à lire par le microcontrôleur.
- **Sensibilité :** 1 cm (pourrait être plus grand mais la plupart des capteurs du marché possèdent cette sensibilité)
  - Largement suffisante pour détecter les changements de niveau de déchets dans la poubelle.
- **Plage de mesure (Range) :** 5 à 150 cm
  - En dessous de 5 cm, c'est plus difficile de mesurer avec l'ultrason. La limite de 140 cm couvre toutes les poubelles individuelles, même les plus grandes.
- **Résolution :** 1 cm
  - Convient pour distinguer les différents niveaux de remplissage.
- **Accuracy :** 3 cm
  - Assure que la distance mesurée est proche de la distance réelle, sans forcément être très exacte.
- **Precision :** 3 cm

- Assure que les mesures répétées soient très proches sans forcément être trop précis non plus.
- **Calibration** : pré-calibré (usage fiable sans intervention de notre part)
- **Protection et robustesse** :
  - IP65, résistant aux projections et poussières
  - Doit fonctionner entre -10 °C et +60 °C
  - Correctement fixé à l'intérieur avec un boîtier pour éviter les chocs (vidage de la poubelle ou déplacement par le particulier)
- **Sortie** : distance en cm

## 2. Capteur infrarouge

- **Type** : digital
  - La détection binaire présence/absence suffit, vu que l'on mesure à quel seuil se trouvent les déchets.
- **Sensibilité** : 2cm
  - C'est assez vu que l'on est sur une range assez grande, pour détecter la présence d'un objet.
- **Plage de mesure (Range)** : 5 à 30 cm
- **Résolution** : non pertinente car la sortie est binaire
- **Accuracy** : non pertinente, nous n'avons qu'une valeur binaire
- **Precision** : non pertinente car la sortie est binaire
- **Calibration** : pré-calibré (usage fiable sans intervention de notre part)
- **Protection et robustesse** :
  - IP65, résistant aux projections et poussières
  - Doit fonctionner entre -10 °C et +60 °C
  - Correctement fixé à l'intérieur avec un boîtier pour éviter les chocs (vidage de la poubelle ou déplacement par le particulier), mais aussi pour le protéger de la lumière ambiante
- **Sortie** : 0 ou 1 (absence ou présence)

## 3. Capteur de poids

- **Type** : analogique (tension convertie via ADC)

- Le fonctionnement du capteur rend l'analogique obligatoire (pont de Wheatstone) et même s'il existe des capteurs digitaux, ils sont beaucoup plus cher et ont le même fonctionnement, il y a juste l'ADC qui est intégré au capteur.
- **Sensibilité :** 0.5 kg
  - Une variation de 0.5 kg est assez significative. (C'est cette sensibilité qui est souvent sur le marché)
- **Plage de mesure (Range) :** 0 à 50 kg
  - Adaptée aux poubelles individuelles, nous avons peu de chance de dépasser ce poids.
- **Résolution :** 0.5 kg
  - Suffisante pour identifier des variations de charge.
- **Accuracy :** 1 kg
  - Assure que la mesure est proche du poids réel.
- **Precision :** 1 kg
  - Assure que les mesures répétées soient très proches sans forcément être trop précis non plus.
- **Calibration :** pré-calibré (usage fiable sans intervention de notre part)
- **Protection et robustesse :**
  - IP65, résistant à l'humidité, poussière et projections
  - Doit fonctionner entre -20 °C et +85 °C
  - Correctement fixé à l'intérieur avec un boîtier pour éviter les chocs (vidage de la poubelle ou déplacement par le particulier)
- **Sortie :** poids en kg

#### 4. Capteur GPS

- **Type :** Digital
- **Sensibilité :** -160 dBm
  - Permet la réception des signaux satellites même dans des environnements urbains ou légèrement couverts.
- **Plage de mesure (Range) :** régionale
  - On contrôle les tournées sur des régions, pas besoin d'avoir une couverture nationale.
- **Résolution :** 0.00001° (environ 1 m)
- **Accuracy :** 3 m
- **Precision :** 3 m

- **Calibration** : pré-calibré
- **Protection et robustesse :**
  - IP65, résistant à la poussière et aux projections d'eau.
  - Fonctionne entre -20 °C et +70 °C.
  - Doit être correctement fixé et protégé dans un boîtier étanche.
- **Sortie** : NMEA (latitude, longitude, altitude, qualité du signal (HDOP), nombre de satellites).
  - En plus des coordonnées on a la qualité du signal et le nombre de satellites qui vont servir à mesurer la fiabilité des données envoyées par le capteur

## 5. Module Wi-Fi

- **Type** : Digital
- **Sensibilité** : -90 dBm
  - Permet une connexion stable même avec un signal faible (distance modérée du point d'accès).
- **Plage de mesure (Range)** : jusqu'à 100 m
  - Suffisant pour pouvoir se connecter au réseau wifi du particulier
- **Résolution** : non pertinente
- **Accuracy** : non pertinente
- **Precision** : non pertinente
- **Calibration** : non pertinente
- **Protection et robustesse :**
  - IP65, résistant à la poussière et aux projections d'eau.
  - Fonctionne entre -20 °C et +70 °C.
- **Sortie** : Non pertinente

## 2 Sources d'incertitude

### a) Incertitudes liées à l'humain :

- Actions humaines imprévisibles perturbant le bon fonctionnement du système (le particulier laisse sa poubelle non fermée).

### b) Incertitudes liées à l'environnement :

- Humidité, poussière et condensation pouvant perturber les capteurs.
- Vibrations ou chocs lors du déplacement de la poubelle.
- Déchets pouvant abîmer les capteurs.
- Ciel obstrué (bâtiment, nuages denses, ...) pour le GPS.
- Températures extrêmes affectant les capteurs.
- Absorption du son par certains déchets (mous).
- Lumière influençant le capteur infrarouge (arrive peu souvent vu que la poubelle sera fermée).
- Signal WIFI affaibli par obstacles (murs).
- Saturation de bande passante (réseau partagé).
- Wifi du particulier indisponible.

### c) Incertitudes liées aux capteurs :

- Panne des capteurs.
- Retards ou pertes de communication entre capteurs et microcontrôleur (si les fils les reliant sont abîmés par exemple).
- Données envoyées par les capteurs bruitées
- Perte de signal ou nombre de satellites insuffisant (GPS).
- Détérioration de l'antenne (GPS).

### d) Moyens de réduction des incertitudes :

- Donner aux particuliers des consignes à respecter pour assurer le bon fonctionnement des capteurs.
- Redondance de capteurs.
- Moyenne des mesures sur plusieurs acquisitions pour lisser les données.

- Vérifications croisées des données entre capteurs.
- Boîtiers et capteurs étanches afin de limiter les perturbations par l'environnement.
- Application de filtres pour réduire le bruit, et la fusion de Dempster-Shafer pour une meilleure certitude au niveau des données.
- Utilisation d'une IA afin de prédire le taux de remplissage d'une poubelle en cas de poubelles avec données anormales, lors de la génération d'un trajet optimisé.
- Pour le GPS, ne considérer les données que si HDOP  $\leq 5$  et nombre de satellites  $\geq 5$ . HDOP est un indicateur permettant de déterminer si la position calculée par le capteur est précise. Autour de 5, nous avons une plutôt bonne précision.
- Stocker les données dans une queue dans le microcontrôleur, puis les renvoyer dès que le réseau est rétabli.

## 3 Traitement

### 3.1 Fusion des données (FOG)

Dans le Fog, nous utiliserons le filtre de fusion de Dempster-Shafer pour fusionner les valeurs des capteurs de même type (par exemple, les valeurs des capteurs infrarouge) puis pour fusionner les valeurs des capteurs de différents types (poids, infrarouge et ultrason). Le résultat nous indiquera le niveau de remplissage de la poubelle parmi plusieurs états.

Exemple :

La première étape sera de lire la valeur brute d'un capteur est lui associer un état

Pour simplifier le cadre de Discernement  $\Theta$  sera {Vide, Moyen, Plein} (ou {V, M, P}) (il y aura plus d'états dans notre application).

Exemple : si un capteur ultrason lit 10cm, on lui associera "Plein", si un autre lit 30cm, on lui associera "moyen".

Ensuite il faudra traduire les lectures brutes en "poids de preuve" pour définir une fonction de masse pour chaque capteur.

Exemple : pour le capteur ultrason qui voit "Plein" on aurait :

- $Mus_1(\{\text{plein}\}) = 0.8$ (forte croyance)
- $Mus_1(\{\text{plein, moyen}\}) = 0.1$
- $Mus_1(\Theta)=0.1$ (marge d'erreur/ignorance)

Et pour le capteur ultrason qui voit "moyen" avec que 30cm on aurait :

- $mus_2(\{\text{moyen}\}) = 0.6$ (croyance moyenne)
- $mus_2(\{\text{plein, moyen}\}) = 0.3$
- $mus_2(\Theta)=0.1$ (marge d'erreur/ignorance)

Après ça nous fusionnons les valeurs des capteurs ultrason pour obtenir un consensus et calculerons les conflits.

Nous utilisons un "tableau de fusion" pour croiser toutes les hypothèses. La cellule contient l'intersection et le produit des masses.

Exemple simplifié avec nos 2 capteurs ultrason (mus1 et mus2) :

mus1\ mus2	{Plein} 0.8	{(Plein, moyen)} 0.1	$\Theta$ 0.1
{Moyen} 0.6	(Ø ,0.48) [Conflit]	{(moyen}, 0.06)	{(moyen}, 0.06)
{Plein, moyen} 0.3	{(plein}, 0.24)	{(plein, moyen}, 0.03)	{(plein, moyen}, 0.03)
$\Theta$ 0.1	{(plein}, 0.08)	{(plein, moyen}, 0.01)	(Θ, 0.01)

Avant la normalisation on obtient ces valeurs de masse : {Moyen} : 0.12, {plein, moyen} :0.07, {plein} :0.32,  $\Theta$  :0.01, Conflit K : 0.48

Et après la normalisation on obtient dans cet exemple :{plein} :0,62, {Moyen} :0,23, {plein, moyen} :0,13, {Θ} :0,02

Conclusion le consensus de ces 2 capteurs ultrason penche fortement pour plein (62%) avec une petite marge de conflit

On pourra appliquer la même méthode avec le résultat des autres capteurs pour obtenir un niveau de remplissage.

### 3.2 Annotation des données capteurs (Edge) (Données d'exemples doivent être cohérentes avec l'archi data ingestion pipeline)

Dans notre architecture, cette étape se produit au niveau "Edge", c'est-à-dire directement sur le microcontrôleur de la poubelle connectée. L'annotation consiste à ajouter les informations de la poubelle avec les données des différents capteurs de la poubelle en une seule structure de donnée.

Nous avons fait ce choix pour n'envoyer qu'une seule structure de donnée organisée dans le Fog qui contient toutes les données de tous les capteurs de la poubelle.

**Exemple de message JSON annoté envoyé par l'Edge :**

```
{"bin_id": "PBL-SOPH-12A7", "timestamp": "2025-11-01T14:30:10Z",
  "Bin_type": "Verre",
  "GPS_value": {"lat": 43.6158, "lon": 7.0722},
  "sensors": [{"id": "US-01", "type": "ultrasonic", "value": 15, "unit": "cm"}, {"id": "US-02", "type": "ultrasonic", "value": 16, "unit": "cm"}, {"id": "IR-01", "type": "infrared", "value": 1, "unit": "boolean"}, {"id": "IR-02", "type": "infrared", "value": 1, "unit": "boolean"}, {"id": "IR-03", "type": "infrared", "value": 0, "unit": "boolean"}, {"type": "load_cell", "value": 21.5, "unit": "kg"}],
  "status": {"battery_level": 82}}
```

### 3.3 Annotation des anomalies capteurs (FOG)

Dans le Fog, une seconde vérification des données est effectuée pour détecter les éventuelles anomalies des données des capteurs. Cette vérification consistera à vérifier que les données des capteurs soient cohérentes, qu'elles existent, etc. Ensuite, nous annoterons la donnée de la poubelle en rajoutant un statut en fonction du résultat de la détection d'anomalie

Cette annotation nous servira pour savoir si on a besoin ou non d'utiliser le service de prédiction de niveau(ia) du cloud pour la génération de trajet optimisé.

### **3.4 Filtrage (pas de filtre passe-bas finalement) (Edge)**

On va appliquer un filtre dont le principe est de supprimer les valeurs aberrantes, car on va prendre plusieurs mesures à certains moments de la journée, il peut donc y avoir certaines valeurs incohérentes.

Par exemple, si un capteur mesure les valeurs suivantes : 10 cm ; 10.2 cm ; 15cm ; 10.3 cm.

Ici, le filtre va éliminer la valeur “15cm” étant aberrante.

Ce filtrage permet donc d'obtenir des données plus cohérentes et moins affectées par les mesures erronées.

# 4 Data Ingestion Pipeline

## 4.1 Pipeline des données capteurs

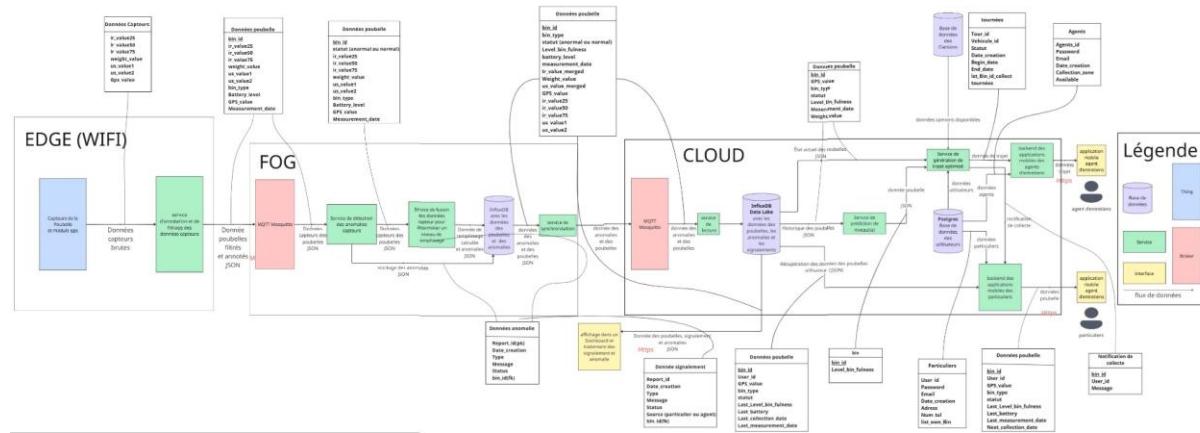


Figure 2 : Pipeline des données capteurs

### 4.1.1 Technologies utilisées

#### Broker du Fog: Mosquitto (MQTT)

Nous avons choisi le protocole MQTT, souvent utilisé pour récupérer des petites données en grande quantité (notamment des données capteurs). Ce qui est notre cas, nous avons plusieurs sources, mais la taille des données est très légère. L'utilisation d'autres protocoles comme HTTP ou AMQP serait surdimensionnée.

On va opter pour mosquitto, qui est open-source et qui nous permet d'utiliser le QoS de niveau 1. Vu que l'on communique très peu de fois entre les poubelles et le FOG (une à quelques transmissions par jour), il faut qu'on s'assure un minimum que le message arrive à destination. En cas de doublon, on va les supprimer dans le FOG. Utiliser QoS 2 serait un peu plus compliqué car on aura une multiplication des échanges et sachant que l'on peut avoir beaucoup de poubelles.

Plusieurs mécanismes de MQTT peuvent nous être utiles :

- **Sessions persistantes** : stocker temporairement les messages non reçus si une poubelle se déconnecte, garantissant qu'aucune donnée n'est perdue lors de la reconnexion.
- **Topics hiérarchiques** : organiser les messages par type de poubelle, zone géographique ou type de capteur pour faciliter le filtrage et l'abonnement sélectif.

#### Broker du Cloud : Mosquitto (MQTT)

On utilise la même chose, vu que les données, même réunies par le fog prennent peu d'espace. Mosquitto suffit donc largement. On passe ici en QoS 2 car on n'a pas beaucoup de FoG (environ une dizaine) publant les données sur le broker et ça permet d'éviter de gérer les doublons.

### **Base des données des capteurs et anomalies du Fog et du Cloud (le Datalake) : InfluxDB**

Nous avons choisi cette base de données parce que les données que nous y enregistrons (données poubelles et anomalies) sont des données temporelles. Elle est également open-source, très performante pour l'ingestion et constitue une solution tout-en-un, ce qui la rend parfaite pour notre Fog (léger, facile à déployer). De plus influx est également très performant pour l'ingestion (surtout avec InfluxDB v3) ce qui convient parfaitement avec notre flux de donnée en batch et du gros volume de donnée qu'il aura besoin de stocker(surtout dans le cloud).

Nous trouvons aussi que ce choix est plus proportionné par rapport aux autres bases de données temporelles concurrentes comme kdb+ (plus adapté pour la finance et payant de surcroît) et plus simple à installer/déployer par rapport à des solutions comme KairosDB, qui nécessite un cluster Cassandra. TimescaleDB aurait été une solution et aurait simplifié l'architecture en ayant qu'une seule base de données (basée sur PostgreSQL), mais aurait représenté une lourdeur dans le Fog et aurait été moins performante pour l'ingestion pure qu'InfluxDB.

### **Base des données des utilisateurs du Fog et du cloud : PostgreSQL**

Pour la base de données des utilisateurs, nous avons choisi PostgreSQL, une base relationnelle. Les utilisateurs ont des données structurées et stables, ce qui rend inutile la flexibilité d'une base NoSQL. De plus, la scalabilité horizontale du NoSQL est moins pertinente. Il répond à un besoin critique : l'intégrité. En effet, les données des utilisateurs sont critiques, notamment pour garantir que les notifications de collecte soient envoyées correctement lors de la génération d'un trajet optimisé. PostgreSQL assure cette intégrité, car ses transactions respectent pleinement les propriétés ACID. Cela garantit qu'il est impossible de générer une notification pour une poubelle sans propriétaire valide et que chaque notification correspond bien à l'utilisateur associé. Enfin, PostgreSQL est un choix fiable, open-source et largement utilisé. Il offre également des fonctionnalités avancées pertinentes pour notre projet, comme la possibilité de créer des fonctions et de supporter des types de données comme JSON.

#### 4.1.2 Explication du flux des données capteurs

Le flux des données commence par la collecte des données des différents capteurs de la poubelle connectée (Edge). Ces données sont filtrées une première fois en utilisant un filtre de bruit pour trier les données et enlever celles qui sont erronées. Ces données seront ensuite annotées et rassemblées en une structure de données qui contient toutes les données des capteurs de la poubelle.

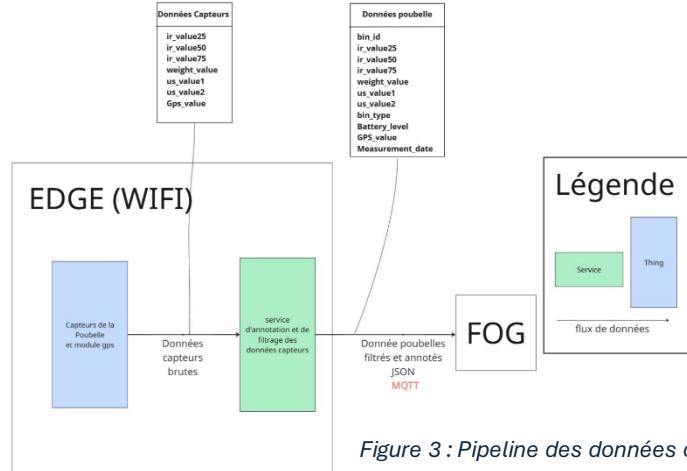


Figure 3 : Pipeline des données capteurs au niveau Edge

Cette structure de données est ensuite envoyée au Fog par Wi-Fi en utilisant Mosquito/MQTT, où un second filtrage des données des capteurs est effectué. Ce filtrage ajoute la donnée de la poubelle en fonction de la présence (ou non) d'une ou plusieurs anomalies. Les anomalies identifiées dans ce service seront enregistrées dans la base InfluxDB du Fog. Après cette étape, nous utiliserons la méthode de Dempster-Shafer pour fusionner les données des différents capteurs afin de déterminer un état de remplissage de la poubelle qui sera ajouté à la structure des données. Ces données seront ensuite stockées dans une base de données temporelle InfluxDB dans le Fog (Ces données pourront être utilisées en mode dégradé en cas de panne du Cloud.).

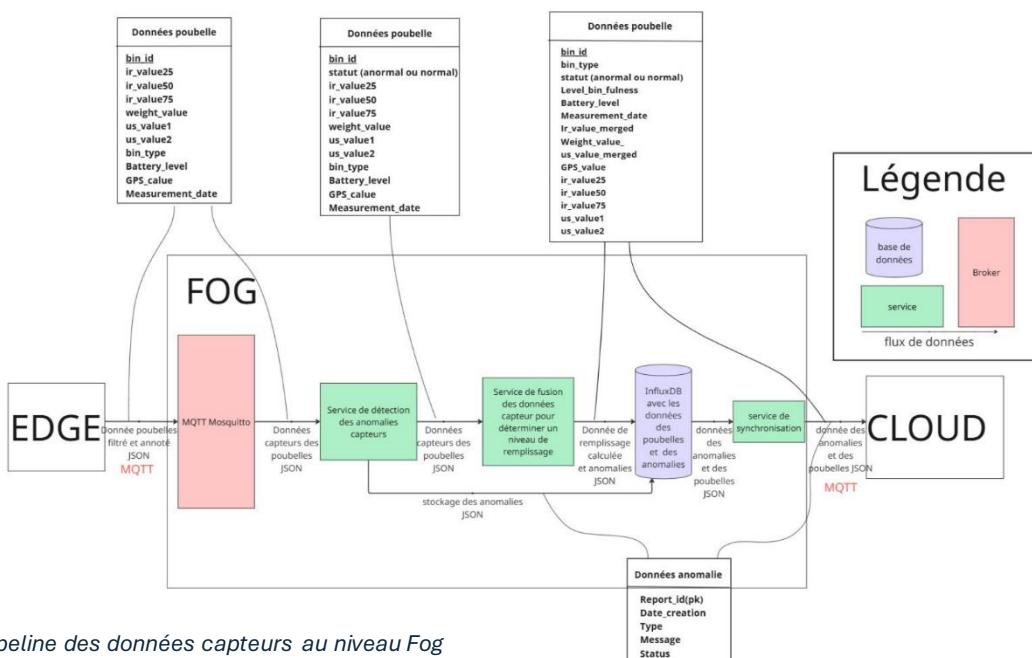


Figure 4 : Pipeline des données capteurs au niveau Fog

Les informations des poubelles et des anomalies des différents Fog sont ensuite transmises vers le Cloud par Mosquito/MQTT, où elles seront lues et stockées dans un datalake InfluxDB.

À partir de là, les données pourront servir à alimenter les différents services présents dans le Cloud :

- Elles seront utilisées par le service de génération des trajets de collecte optimisés (notre *core domain*) qui déterminera un trajet de collecte en fonction du niveau de remplissage des différentes poubelles et des agents et camions de collecte disponibles. Si le niveau de remplissage d'une poubelle n'est pas disponible (par exemple parce qu'une anomalie est présente), un service de prédiction de niveau sera appelé pour déterminer le niveau de remplissage potentiel en fonction de l'historique de remplissage de cette dernière. Le ou les trajets de collecte optimisés générés seront ensuite envoyés aux applications mobiles des agents pour la collecte, et les propriétaires des poubelles seront avertis de la collecte par une notification sur leur application mobile pour les inviter à sortir leur poubelle.
- Le niveau de remplissage d'une poubelle sera envoyé à l'application mobile du propriétaire de cette dernière. Lorsque ce dernier, demande à voir les données relatives à sa poubelle
- Les anomalies seront envoyées à une équipe de maintenance IoT où elles seront affichées dans des dashboards et où elles seront traitées.

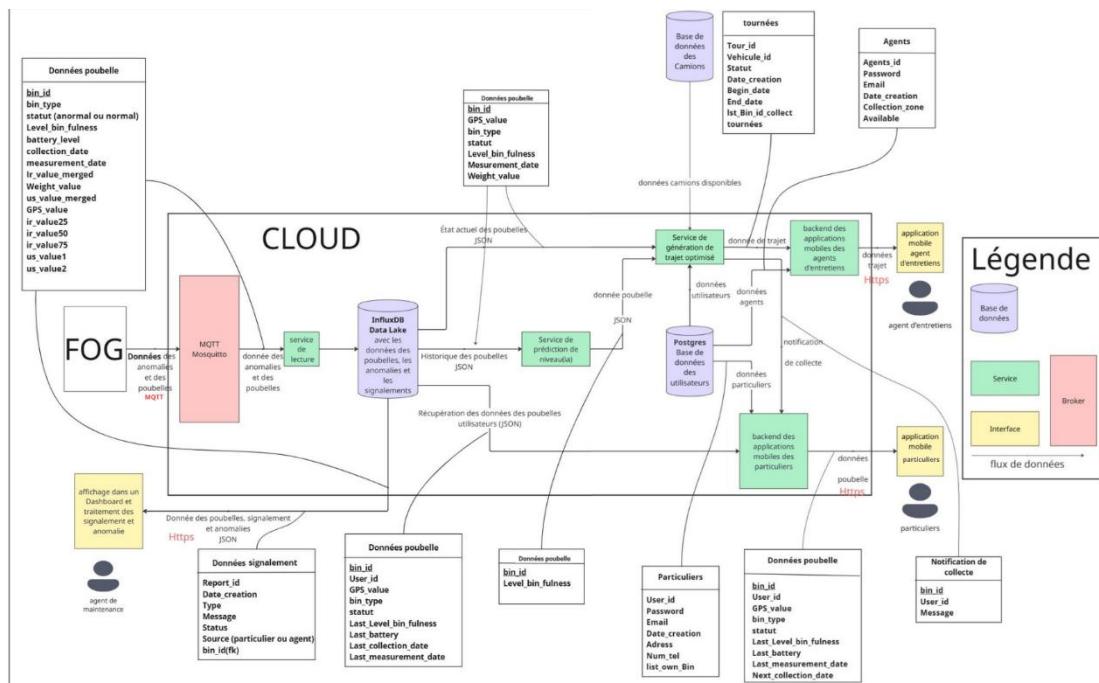


Figure 5 : Pipeline des données capteurs au niveau Cloud

## 4.2 Pipeline des reports/ticket d'anomalie des utilisateurs

### 4.2.1 Explication de la pipeline des reports/ticket d'anomalie des utilisateurs

Les signalements sont émis par les utilisateurs (agents et particuliers) depuis leur application mobile. Ils seront ensuite stockés dans notre datalake pour ensuite être envoyés aux équipes de maintenance pour y être traité dans une interface graphique (dashboard, application web).

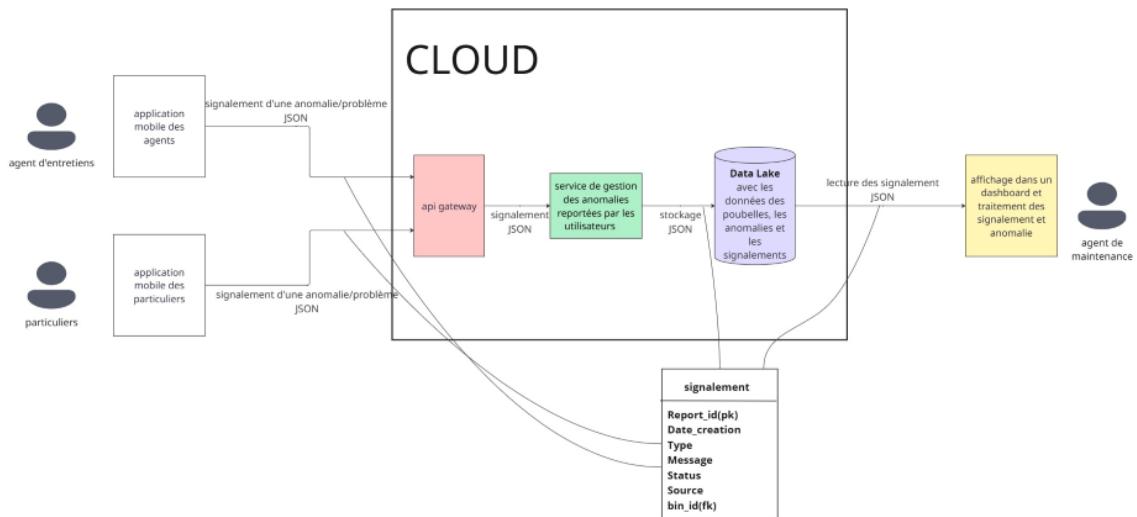


Figure 6 : Pipeline des données des signalements

## 4.3 Pipeline d'ajout des utilisateurs

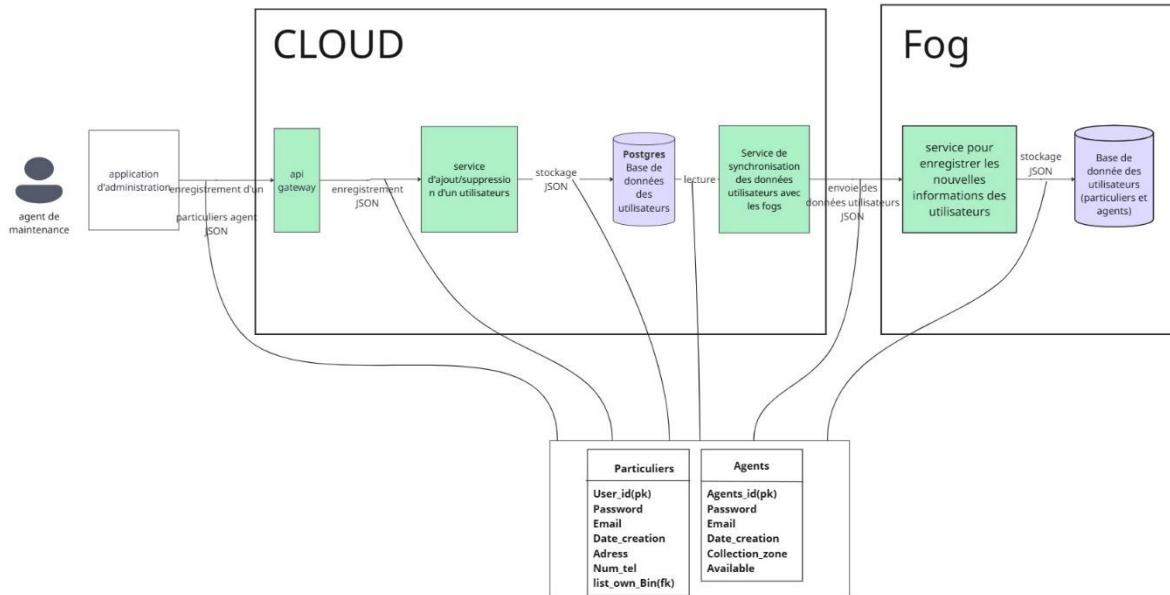


Figure 7 : Pipeline des données d'ajouts utilisateurs

### 4.3.1 Explication du pipeline d'ajout d'utilisateurs

L'agent de maintenance via l'application administratif peut enregistrer un nouvel utilisateur qui sera remonté au cloud avec le gateway ou avec le service d'ajout d'un utilisateur. Le nouvel utilisateur va être stocké dans la base de données centrale Postgres et immédiatement, le Service de Synchronisation du Cloud enverra une copie de cet utilisateur au FOG.

Le FOG enregistre cette copie dans sa propre base de données locale, garantissant ainsi que l'authentification reste fonctionnelle en Mode Dégradé.

**Table des figures :**

Figure 1 :schéma de la poubelle .....	3
Figure 2 : Pipeline des données capteurs .....	13
Figure 3 : Pipeline des données capteurs au niveau Edge .....	15
Figure 4 : Pipeline des données capteurs au niveau Fog .....	15
Figure 5 : Pipeline des données capteurs au niveau Cloud .....	16
Figure 6 : Pipeline des données des signalements .....	17
Figure 7 : Pipeline des données d'ajouts utilisateurs .....	18

**Rédaction avec l'aide de ChatGPT/Gemini**