



## STAGE PFA

# Optimisation des Pratiques Agricoles par l'Implémentation d'une Plateforme IoT et l'Intégration de l'Intelligence Artificielle dans les Serres Agricoles

Réalisé par :  
Abdallahi DAH

Encadré par :  
Pr. Abderrahim BAJIT

Membre du jury :

Pr Es-sadek Zeriabe  
Pr Ayoub Abdellaoui

Chef de filière, ENSAM Rabat  
Professeur, ENSAM Rabat

Examinateur  
Examinateur

# PLAN

## Introduction

- 01 Problématique - Objective
  - 02 Conception et Architecture
  - 03 Mise en Œuvre et Développement
  - 04 Modélisation et Machine Learning
  - 05 Résultats
  - 06 Recommendation
- Conclusion



# **CONTEXTE DU PROJET**

---

Le projet SCASAIR, fruit d'une collaboration entre le Centre National pour la Recherche Scientifique et Technique (CNRST) et Office chérifien des phosphates (OCP), se positionne à l'intersection de la recherche scientifique et de l'innovation technologique. Axé sur l'optimisation environnementale des serres agricoles, ce projet ambitieux exploite les avancées de l'intelligence artificielle et de l'Internet des Objets (IoT).





# Problématique - Objective





# PROBLÉMATIQUE

---

Actuellement, la surveillance des paramètres cruciaux tels que la température, le pH, le CO<sub>2</sub> et l'humidité dans les serres agricoles souffre :

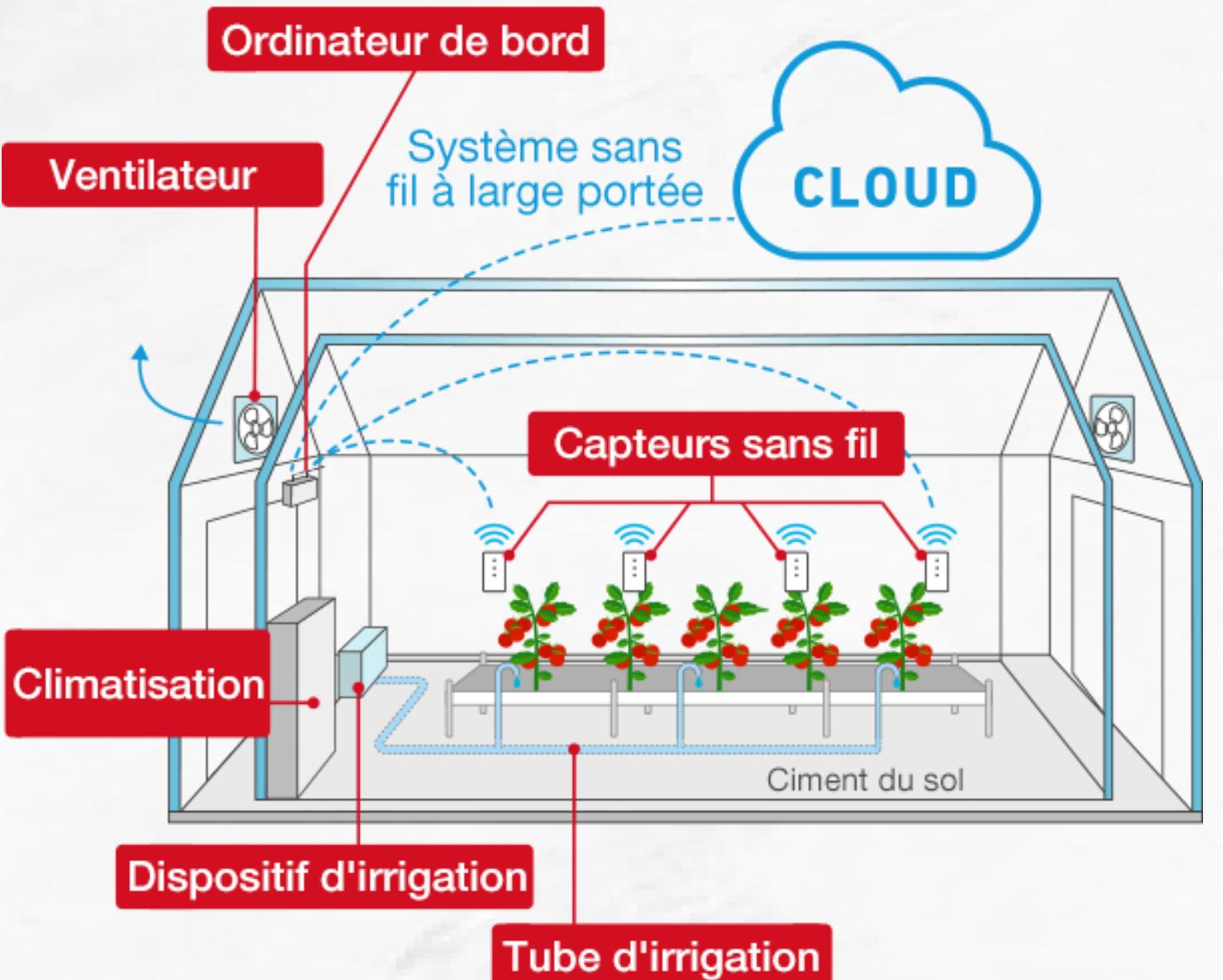
- Manque de réactivité en temps réel,
- Automatisation insuffisante
- Difficultés à maintenir des conditions de croissance optimales .

Ces lacunes peuvent entraîner des pertes de rendement, des incohérences dans les conditions de culture et une utilisation inefficace des ressources.

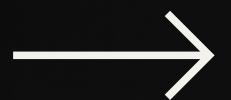
# OBJECTIVE DU PROJET

Le projet réalisé vise l'optimisation des pratiques agricoles en exploitant les technologies émergentes telles que l'Internet des Objets (IoT) et l'Intelligence Artificielle (IA). À travers la mise en place d'une plateforme complète, le projet s'attache à surveiller et à analyser en temps réel les paramètres environnementaux au sein des serres agricoles. L'objectif principal est d'améliorer l'efficacité des processus agricoles en fournissant des données précises sur la température, l'humidité, le pH et le CO<sub>2</sub>.

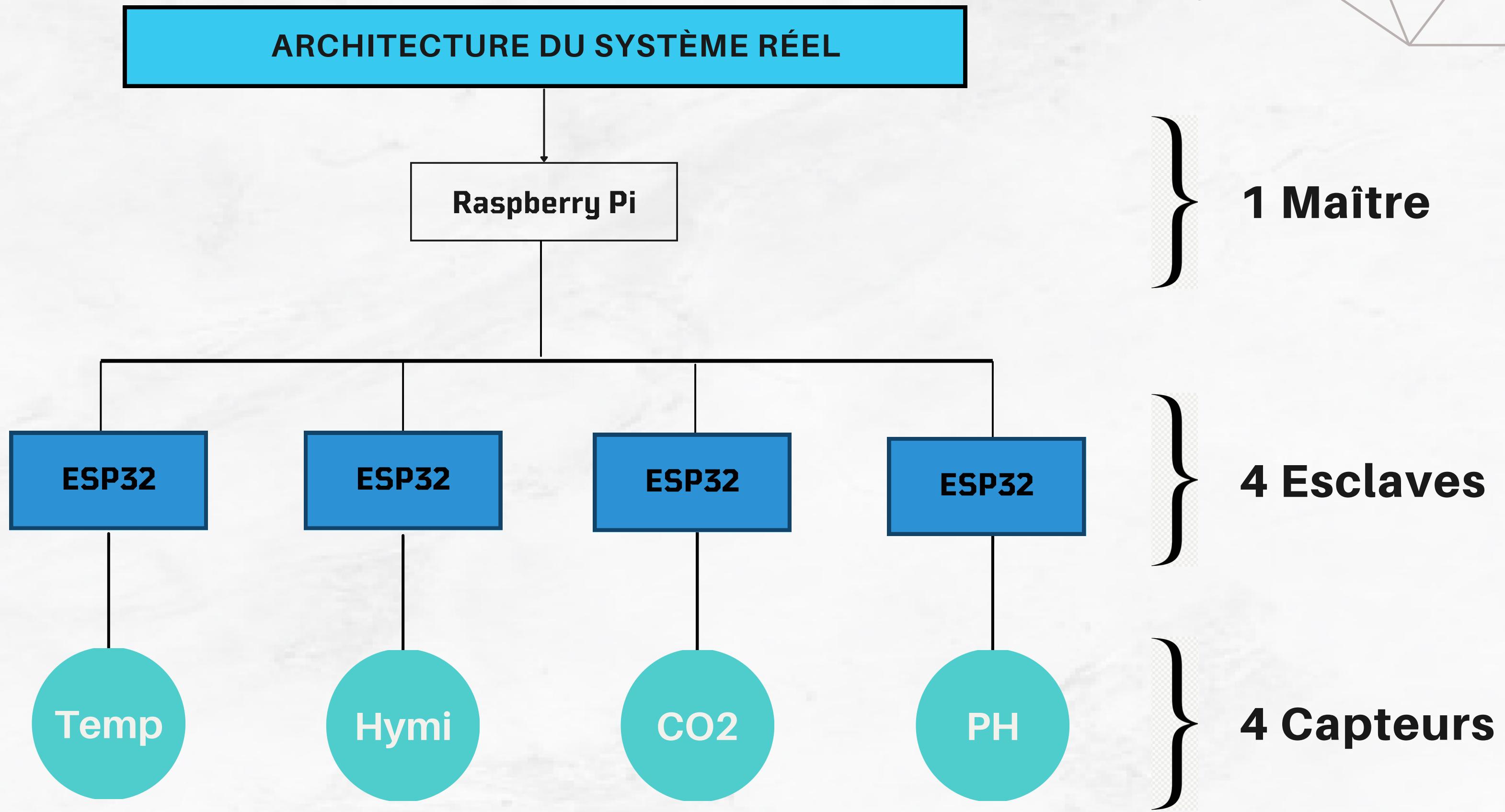
# Serre intelligente



# Conception et Architecture

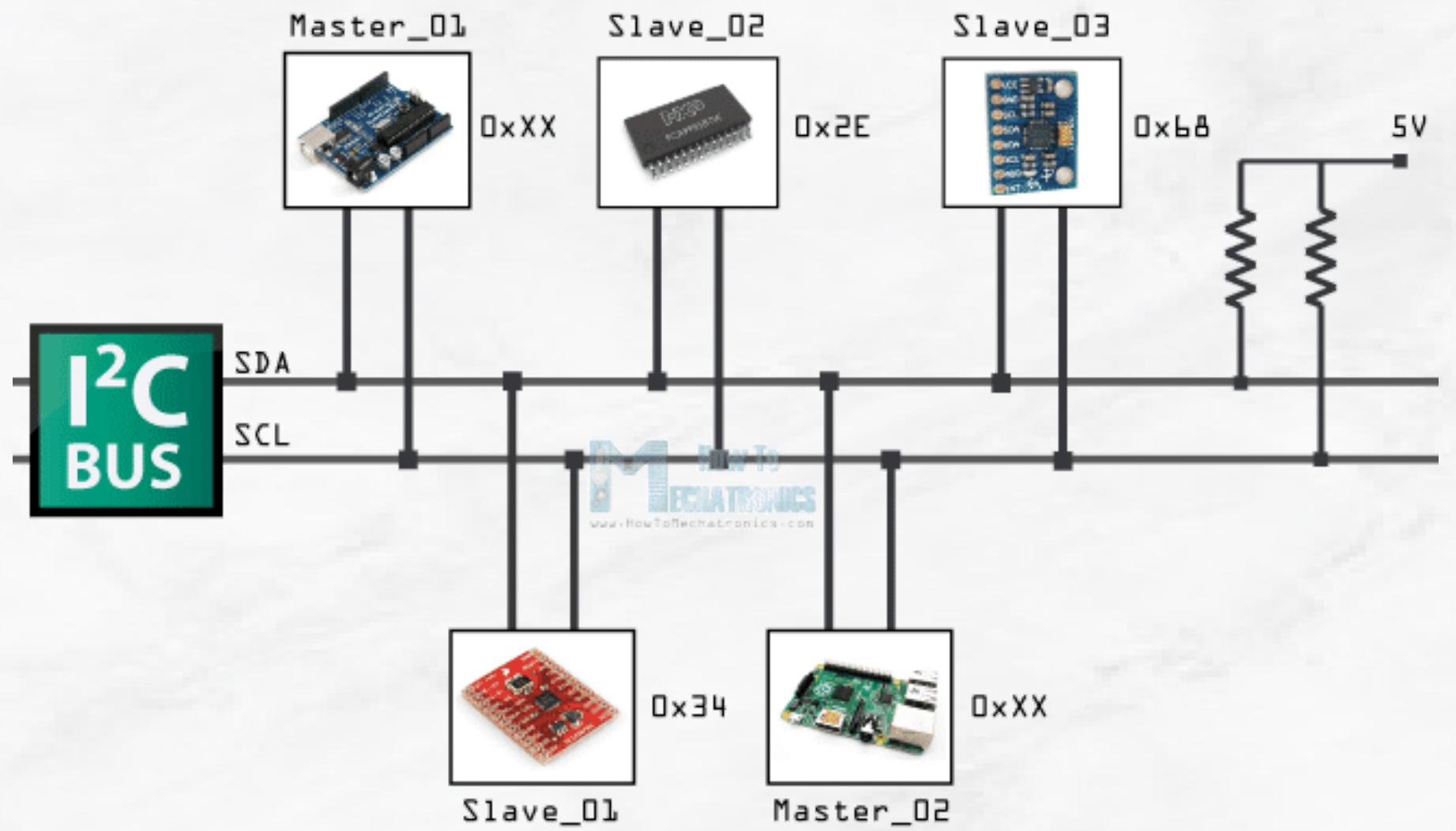


# Conception et Architecture





## Description de la Communication InterNœuds Réels



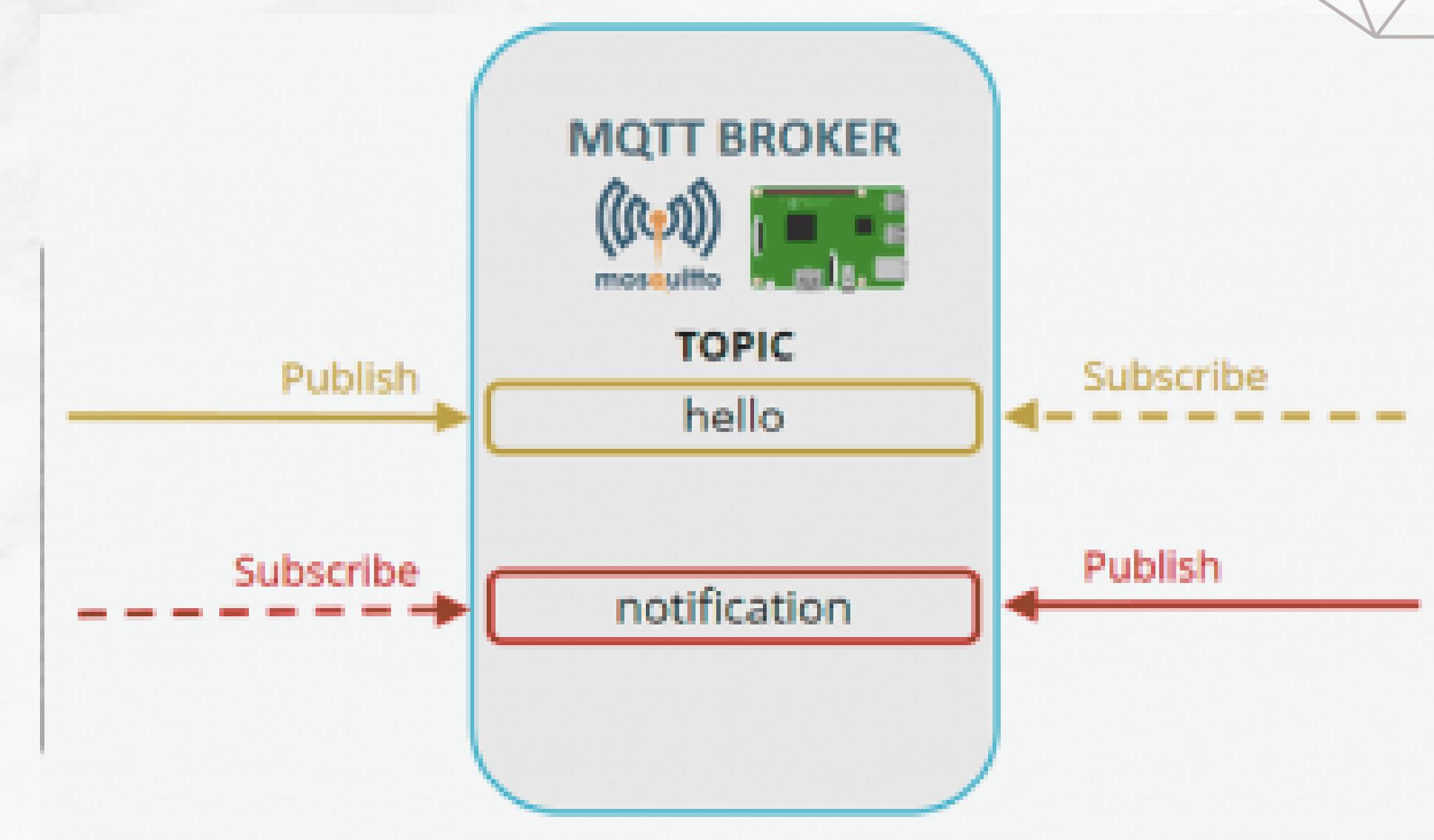
I<sup>2</sup>C (Inter-Integrated Circuit)

- Communication I<sup>2</sup>C : Les nœuds réels utilisent le protocole de communication I<sup>2</sup>C (Inter-Integrated Circuit) pour échanger des données entre eux et avec la Raspberry Pi.
- Demandes Actives : Dans cette architecture, les nœuds réels restent inactifs jusqu'à ce qu'une demande de données soit initiée par la Raspberry Pi. Lorsqu'une demande est reçue, les nœuds réels activent leurs capteurs pour collecter des données fraîches.
- Agrégation des Données : La Raspberry Pi établit une communication I<sup>2</sup>C avec chaque nœud réel pour demander des données, les collecte et les stocke temporairement avant de les envoyer vers le cloud.

## Description de la Communication InterNœuds Réels

La communication MQTT est utilisée pour transmettre les données collectées depuis le maître de la plateforme vers le cloud, Node-RED et la page web.

La communication MQTT permet de distribuer efficacement les données collectées à travers le système, permettant ainsi une analyse et une visualisation en temps réel.

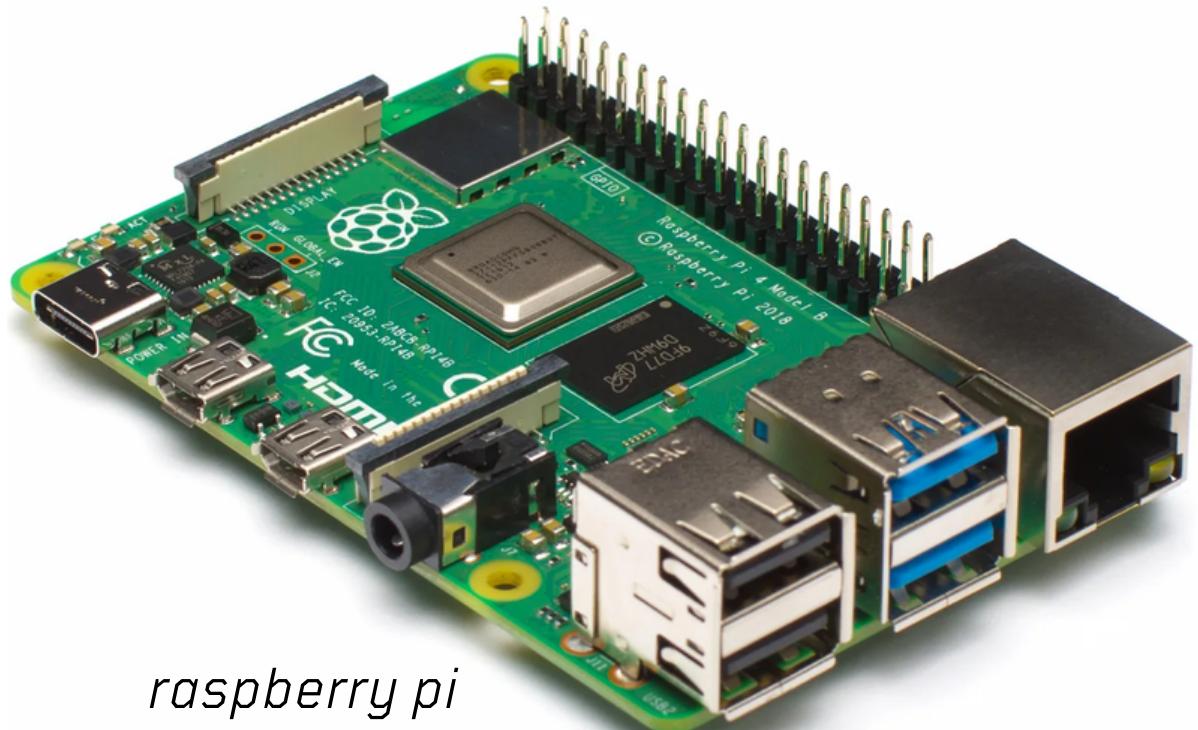
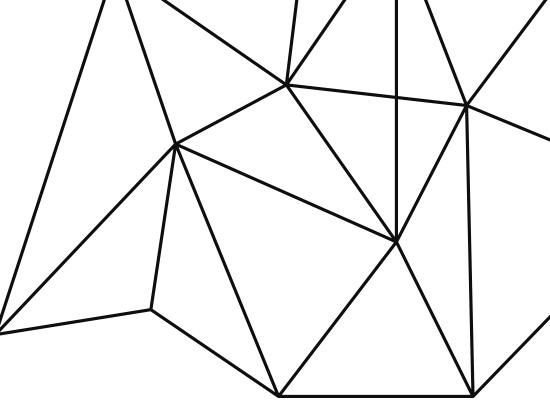


Communication MQTT

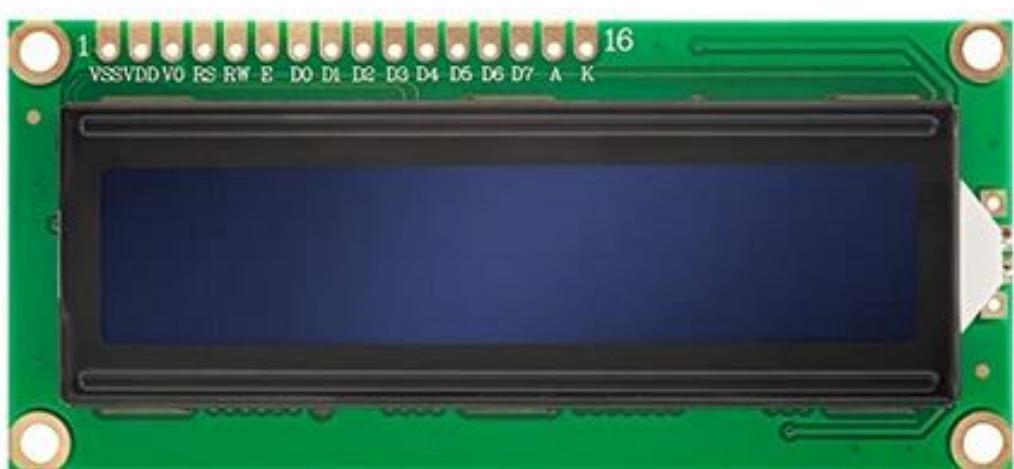
# Mise en Œuvre et Développement



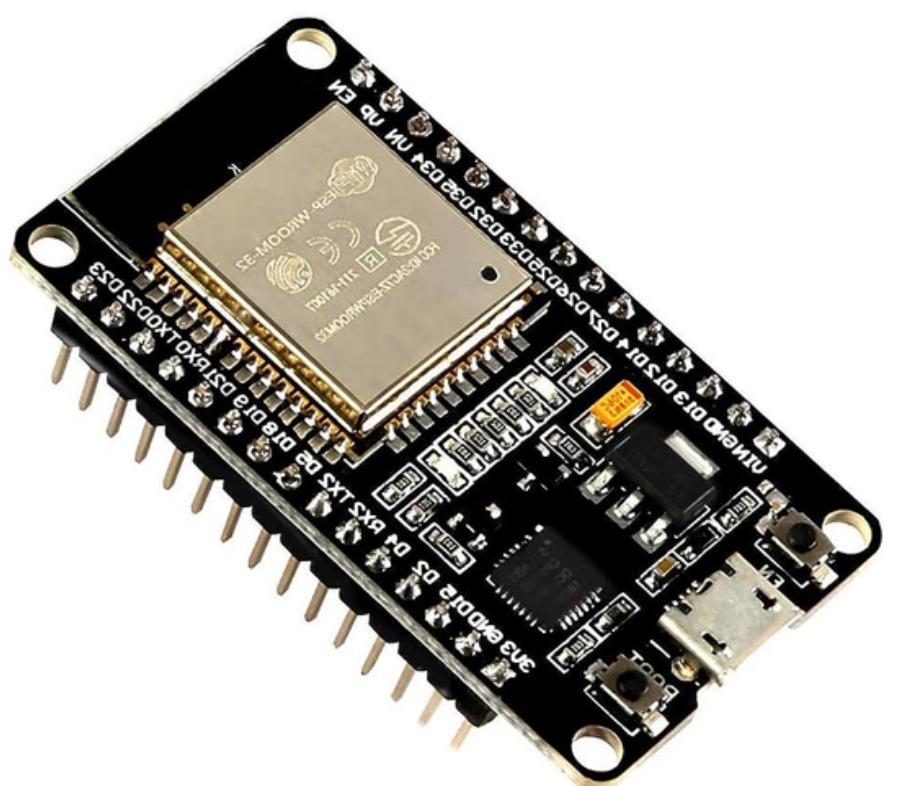
# LES MATIÈRES



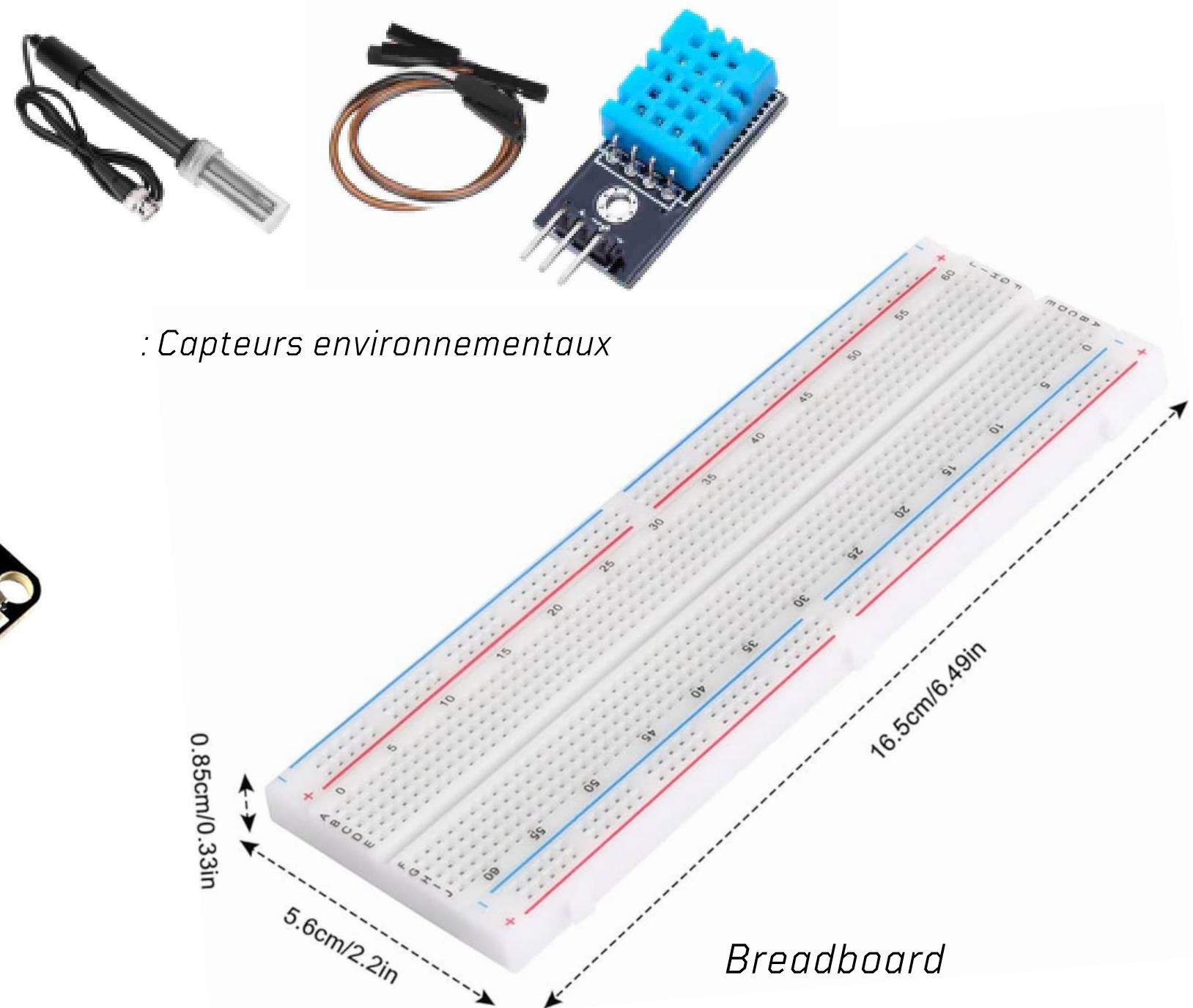
*raspberry pi*



*Écran LCD 16 × 2*



*: ESP 32/82*



*Breadboard*

# LES TECHNOLOGIES

---



Chart.js

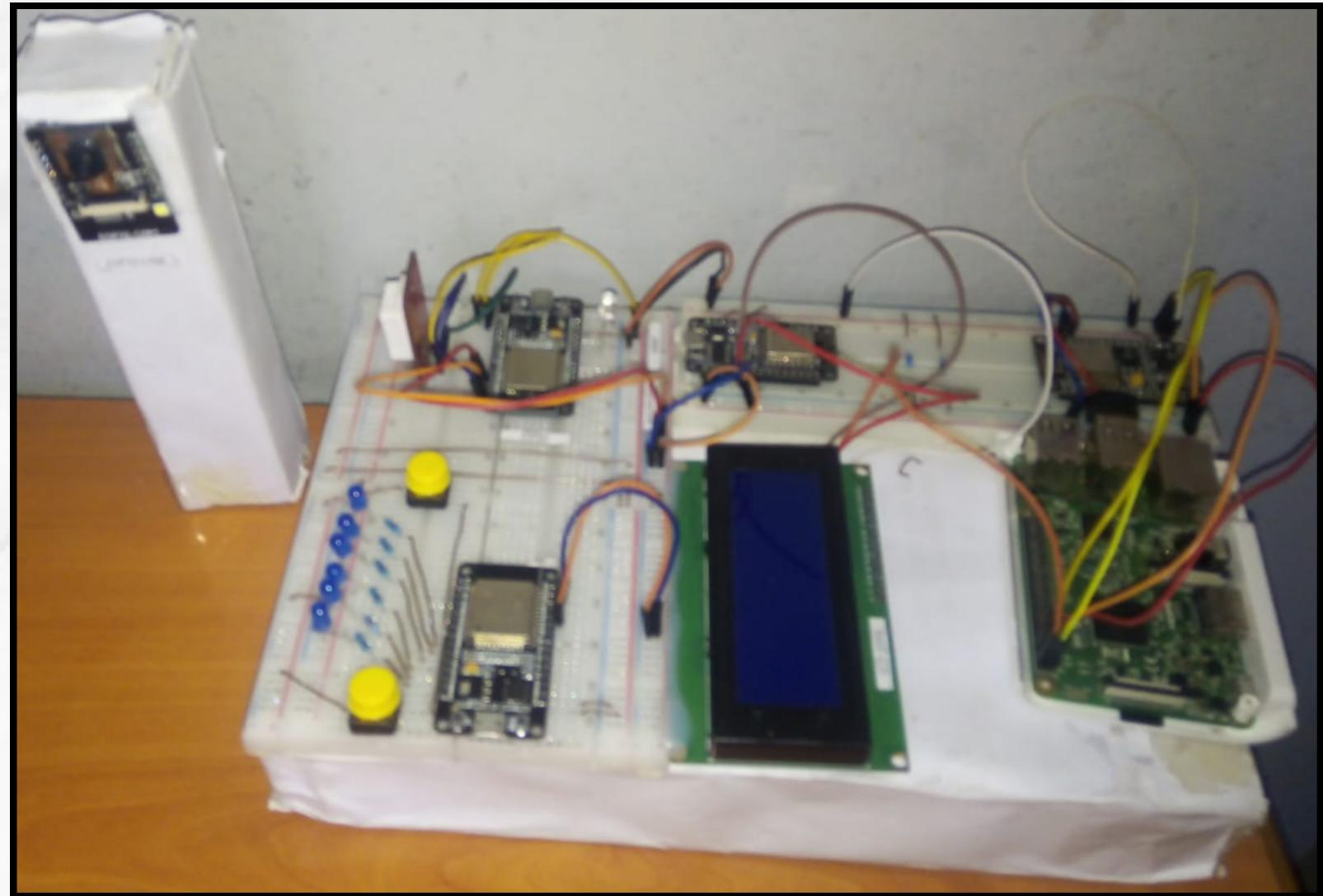


# PLATEFORME RÉELLE

---

## NŒUDS RÉELS

Chaque nœud réel est composé d'une Raspberry Pi agissant en tant que maître et de 4 microcontrôleurs ESP32 agissant en tant qu'esclaves. Chaque ESP32 est connecté à un capteur spécifique (température, pH, humidité et CO<sub>2</sub>).

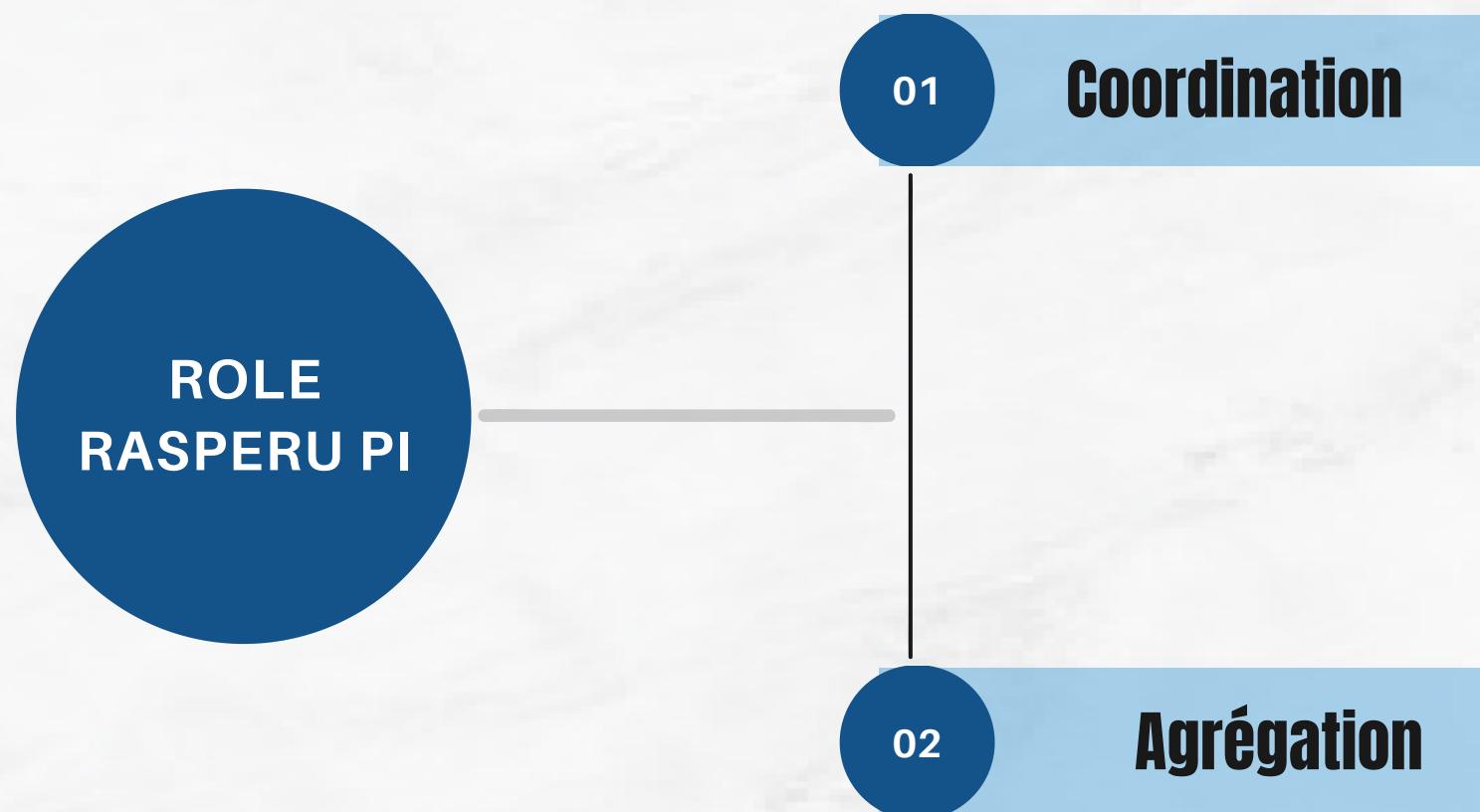


*Nœuds Réels*

# PLATEFORME RÉELLE (NŒUDS RÉELS)

## RÔLE DU RASPBERRY PI MAÎTRE

Le Raspberry Pi maître est le cerveau du système réel, jouant un rôle essentiel dans la coordination des nœuds réels et la gestion des données

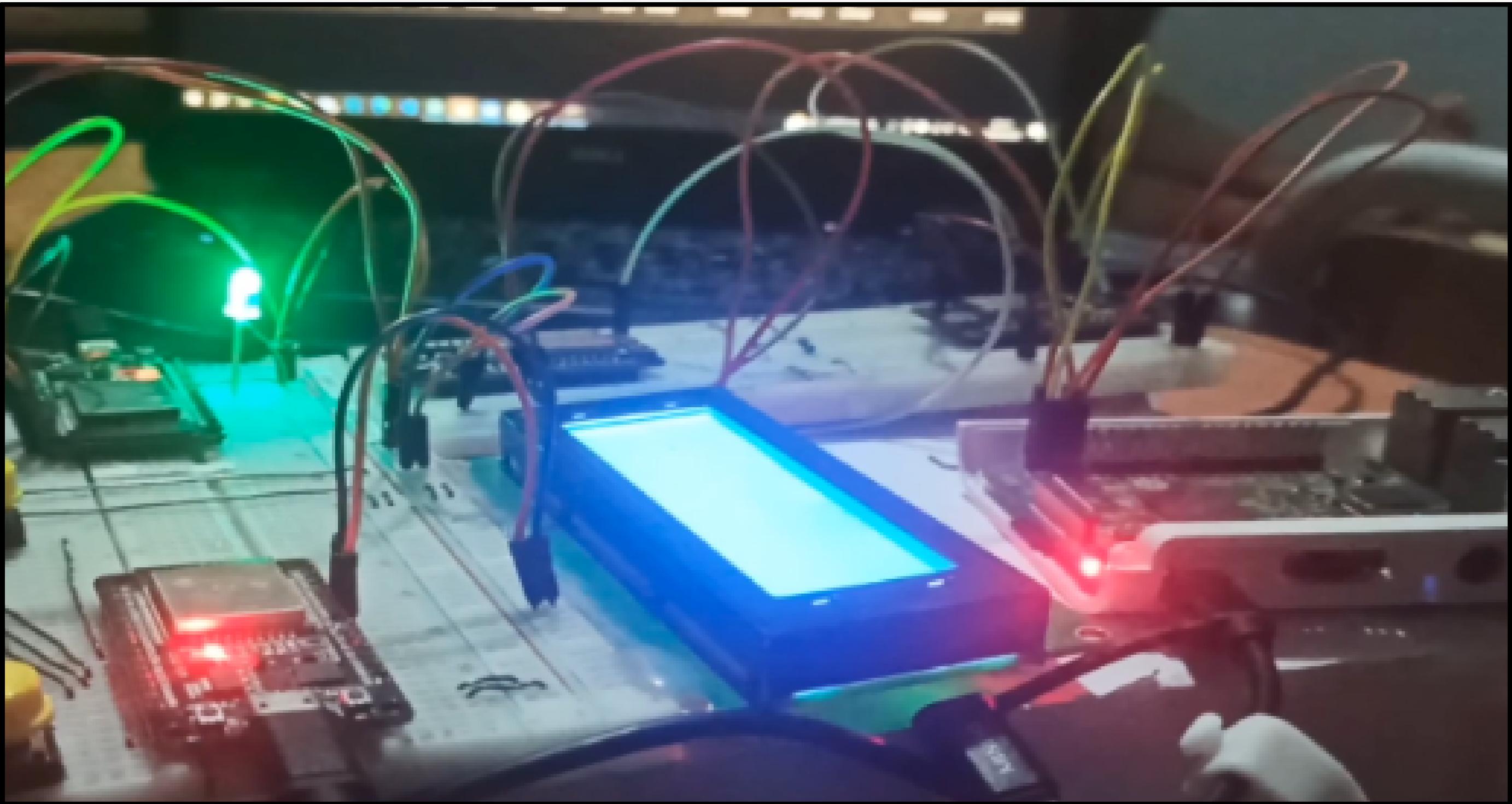


Le maître assure la coordination des activités des nœuds réels, y compris la collecte de données. Il émet des demandes de collecte de données aux nœuds réels en réponse aux demandes du cloud ou de Node-RED.

Une fois que les données sont collectées à partir des nœuds réels, le Raspberry Pi maître les agrège pour créer un ensemble complet de données environnementales. Les données agrégées sont prêtes à être transmises vers le cloud, Node-RED et la page web pour une visualisation et une analyse ultérieures.

# PLATEFORME RÉELLE (NŒUDS RÉELS)

Assemblage des Nœuds Réels :

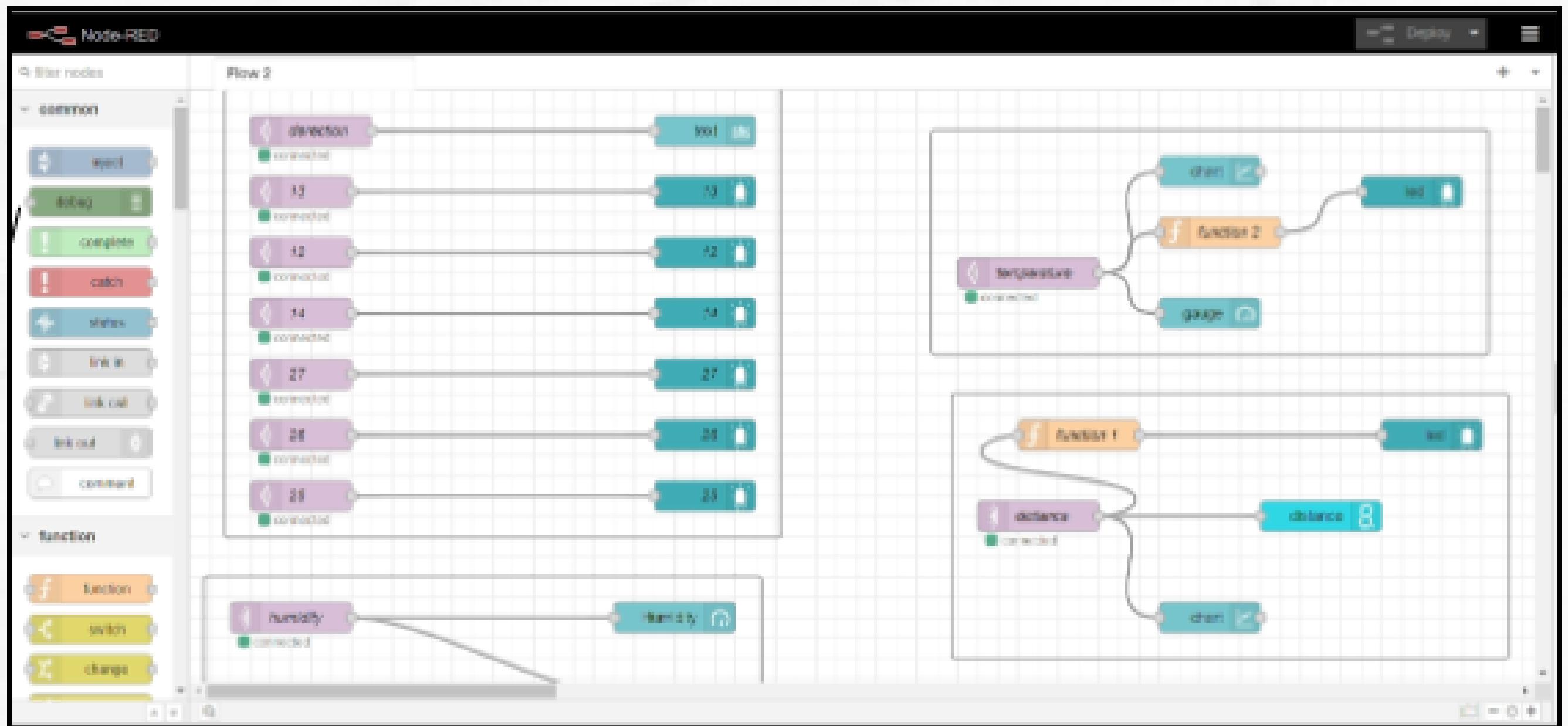


# PLATEFORME RÉELLE (NŒUDS RÉELS)



## Node-RED

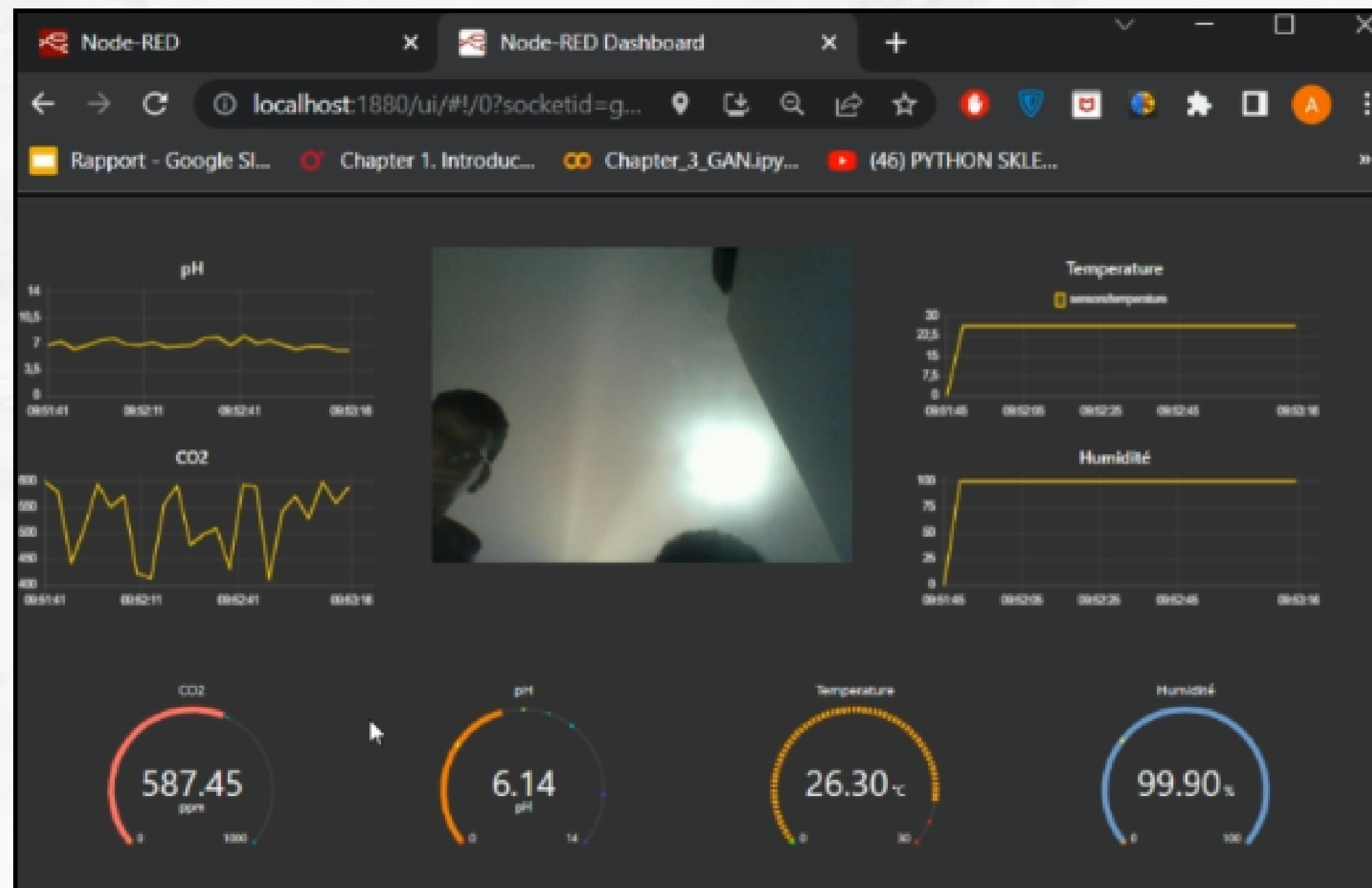
Les données agrégées sont transmises de manière continue à Node-RED via MQTT. Node-RED est capable de souscrire à ces données et de les afficher graphiquement. Il permet également de configurer des règles et des automatisations en fonction des données collectées.



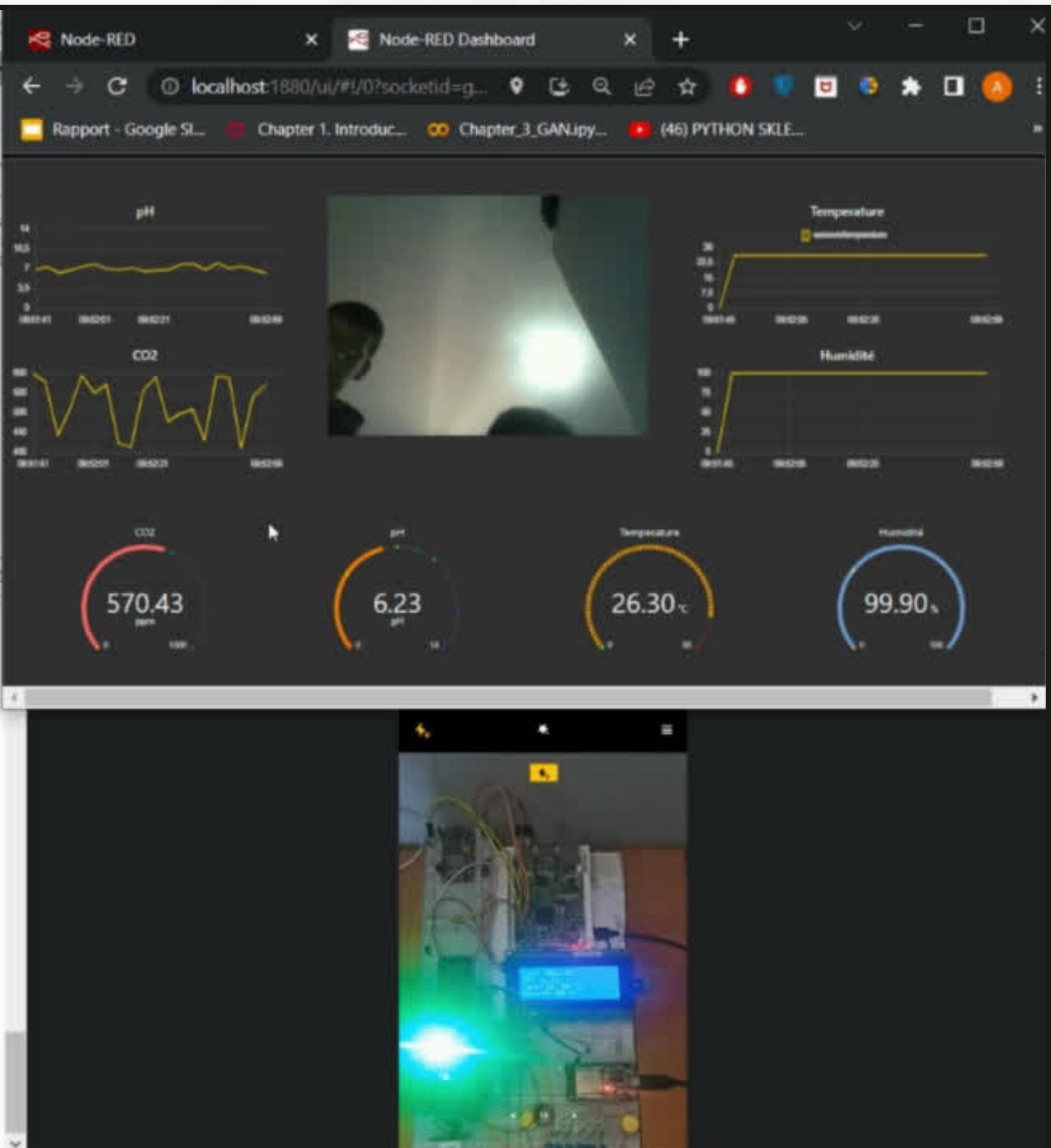
# PLATEFORME RÉELLE

La combinaison de Node-RED et de MQTT permet une visualisation aisée et une gestion avancée des données environnementales collectées.

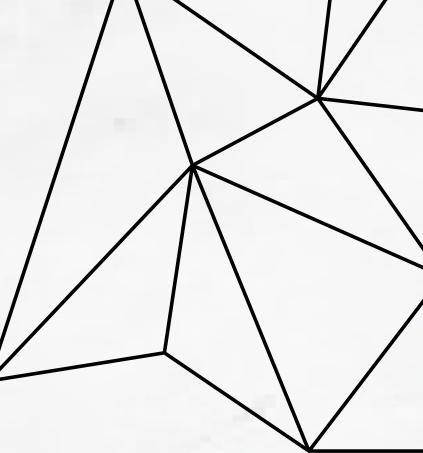
## Node-RED



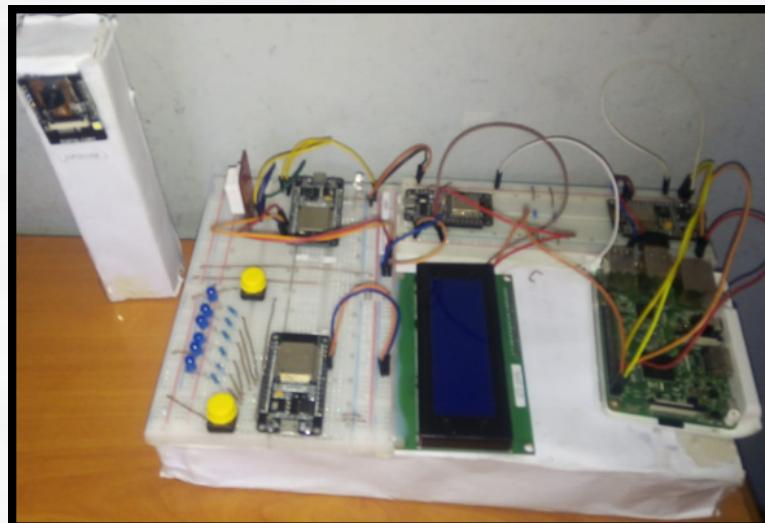
# **Presentation de resultas**



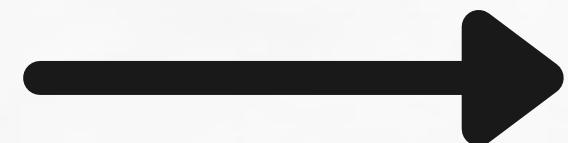
# Plateforme Virtuelle

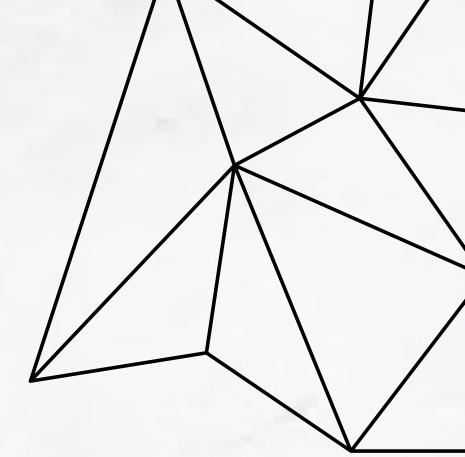


La plateforme virtuelle consiste en une simulation informatique de la plateforme réelle. Dans cette simulation, chaque composant réel est représenté par un fichier Python distinct. Chaque composant est simulé par un script Python qui imite son comportement réel. Cette approche permet de tester, développer et valider le système sans avoir besoin de matériel physique.

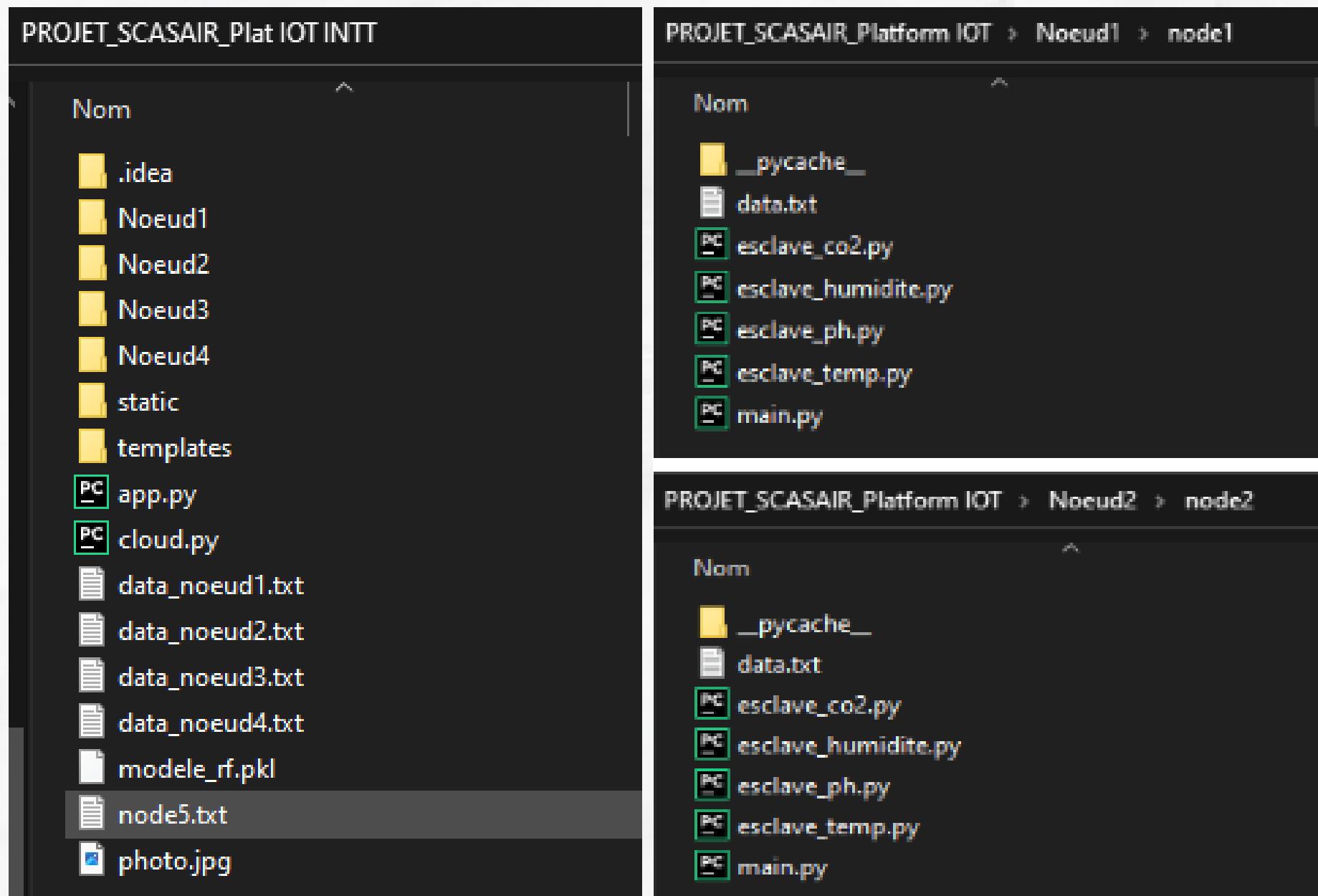


simulation





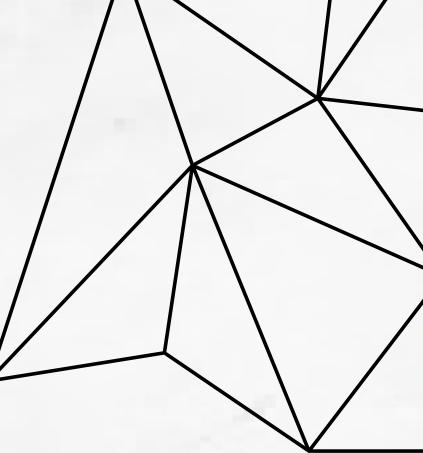
# Plateforme Virtuelle



## Fonctionnement de la Plateforme Virtuelle :

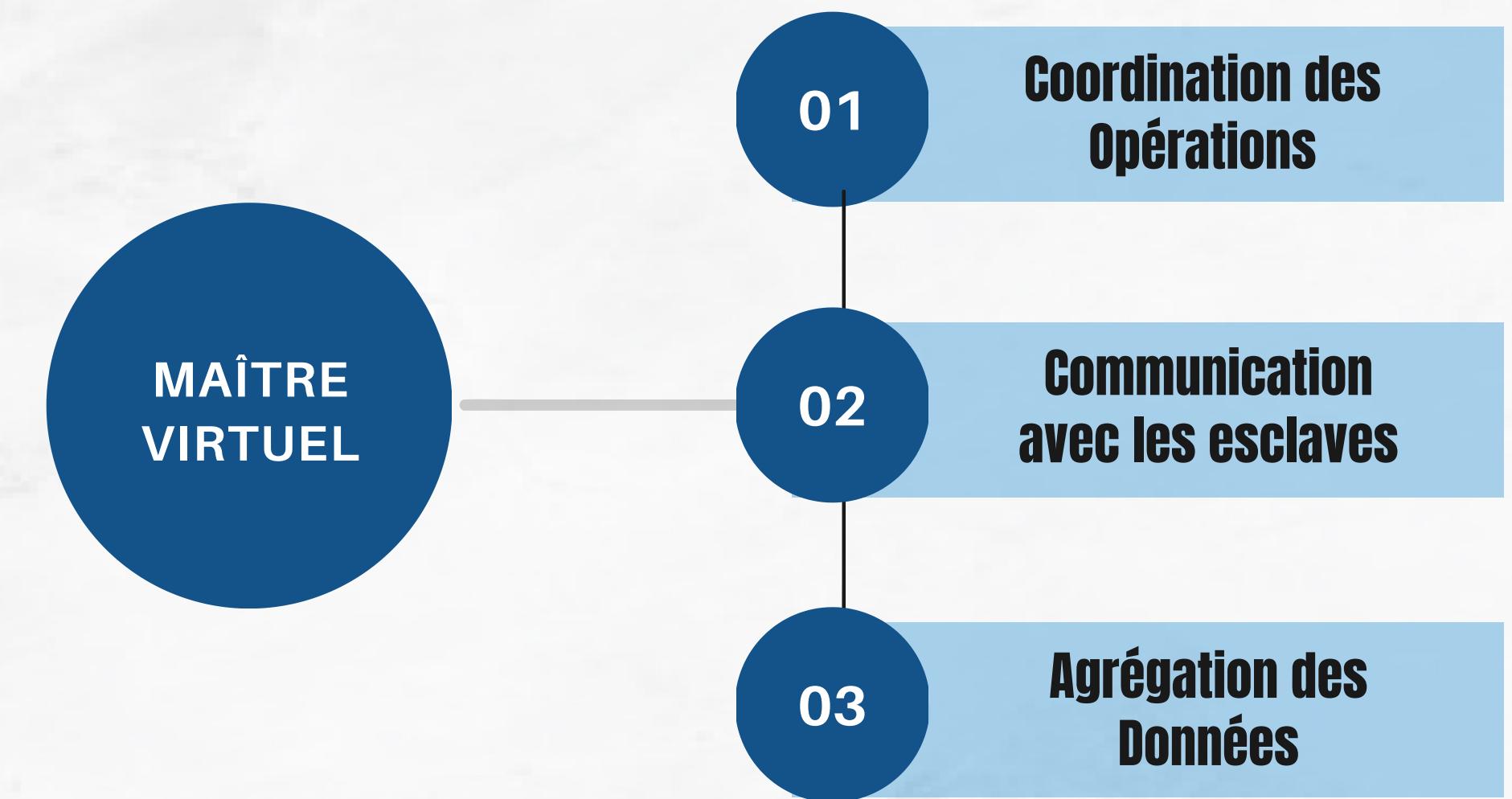
- La plateforme virtuelle fonctionne comme une réplique virtuelle de la plateforme réelle. Lorsque le cloud virtuel demande des données, il envoie des requêtes aux
- Les maîtres virtuels, représentant le Raspberry Pi maître dans la plateforme réelle, envoient des demandes aux esclaves virtuels pour collecter des données.
- Les données collectées par les esclaves virtuels sont renvoyées aux maîtres virtuels,
- Le cloud virtuel récupère toutes les données des nœuds virtuels, les agrège et les stocke dans des bases de données distinctes.

# Plateforme Virtuelle



## Maître Virtuel

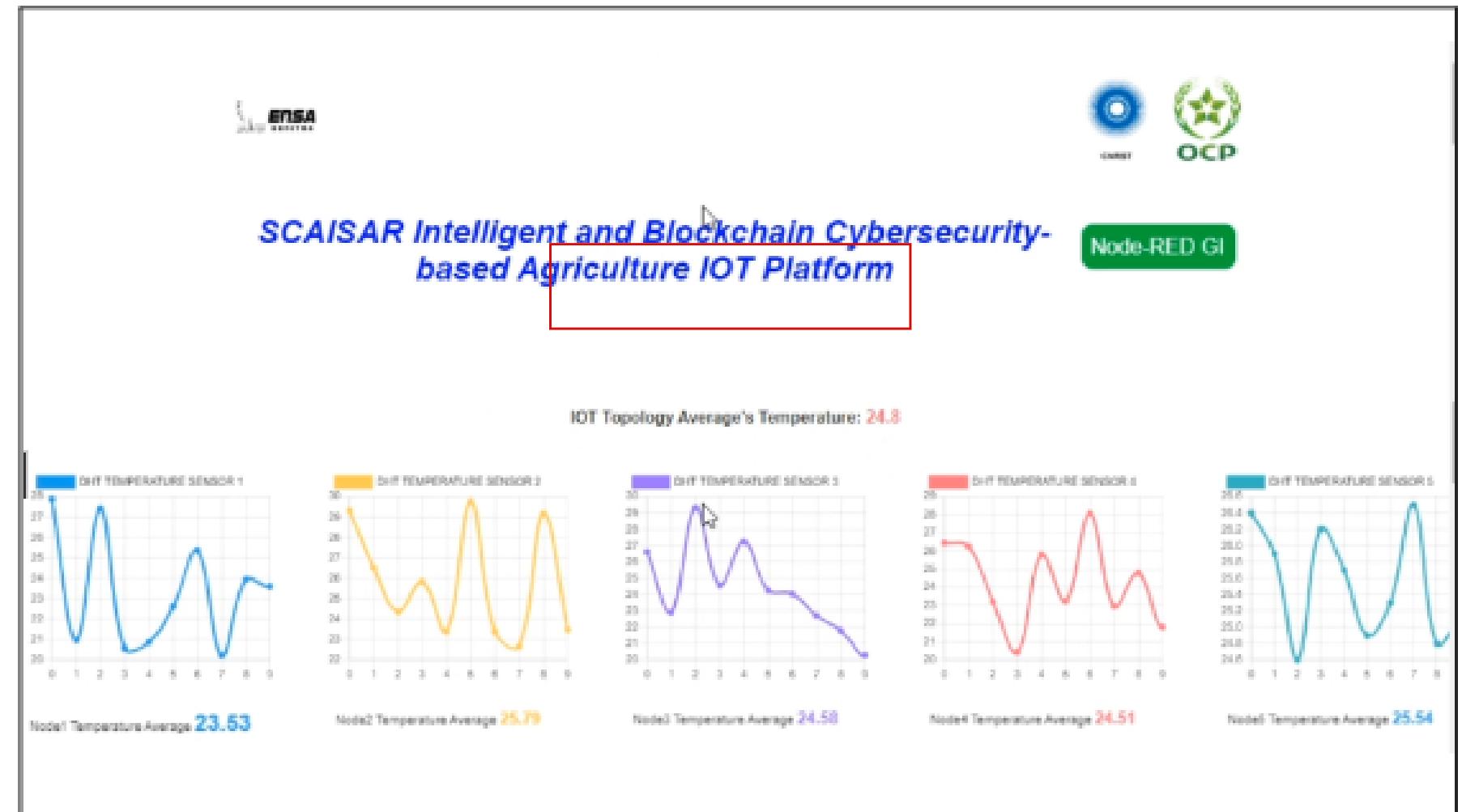
Chaque nœud virtuel est dirigé par un fichier Python appelé "maître.py". Ce script simule le rôle du Raspberry Pi maître dans la plateforme réelle. Le maître virtuel est essentiellement le cerveau de chaque nœud virtuel.



# Plateforme Virtuelle

## PAGE WEB

La page web joue un rôle crucial en fournissant une vue en temps réel des données collectées à partir des nœuds virtuels, en les organisant de manière claire et en permettant aux utilisateurs de suivre et d'analyser les tendances.

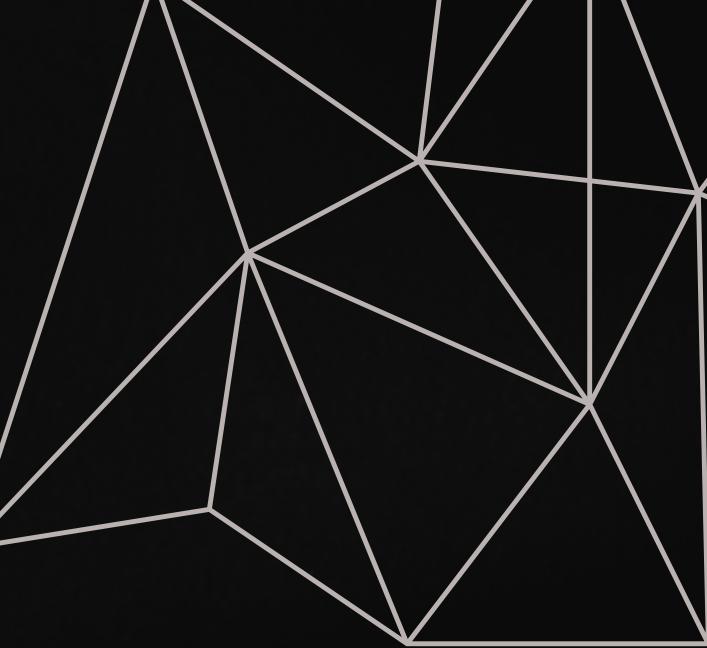


# Plateforme Virtuelle

## PAGE WEB

The screenshot shows a dark-themed IDE interface with multiple windows and panes:

- Terminal:** Shows the command `python .\cloud.py` being run, outputting "GET DATA1" and "publier".
- Explorer:** Displays a file tree under the folder  `noeudes > python .\cloud.py`. It includes subfolders `NOEUDS`, `static`, and `templates`, along with files like `main.py`, `app.py`, `cloud.py`, and several `data_noeudX.txt` files.
- Code Editor:** The `app.py` file is open, showing Python code for reading data from files and calculating temperatures for five nodes.
- Output:** Shows the command `PS C:\Users\DELL\Downloads\Compressed\noeudes> python .\app.py`.
- Terminal:** Shows four separate terminal instances running the command `python .\main.py` in different directory paths: `C:\Users\DELL\Downloads\Compressed\noeudes\Noeud1\node1`, `\Noeud2\node2`, `\Noeud3\node3`, and `\Noeud4\node4`. Each instance outputs "Les données sont publiées".
- PowerShell:** A sidebar on the right lists multiple PowerShell sessions.



# Modélisation et Machine Learning



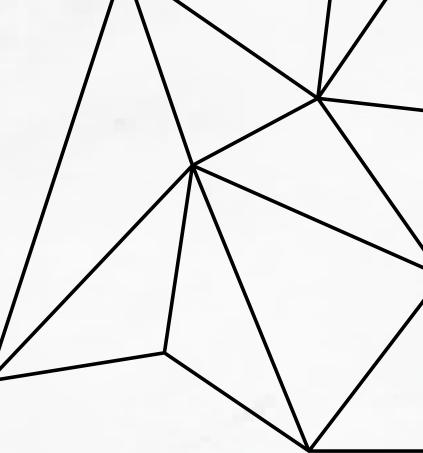


# Base de données :

Les données ont été générées à partir d'une série de trois réseaux de capteurs identiques, construits sur mesure et basés sur une maquette. Chaque baie était connectée aux appareils Raspberry Pi. Chacun des trois appareils IoT a été placé dans un emplacement physique présentant des conditions environnementales variées.

device	environmental conditions
00:0f:00:70:91:0a	stable conditions, cooler and more humid
1c:bf:ce:15:ec:4d	highly variable temperature and humidity
b8:27:eb:bf:9d:51	stable conditions, warmer and dryer

*Les trois devices dans la Dataset*



# Base de données :

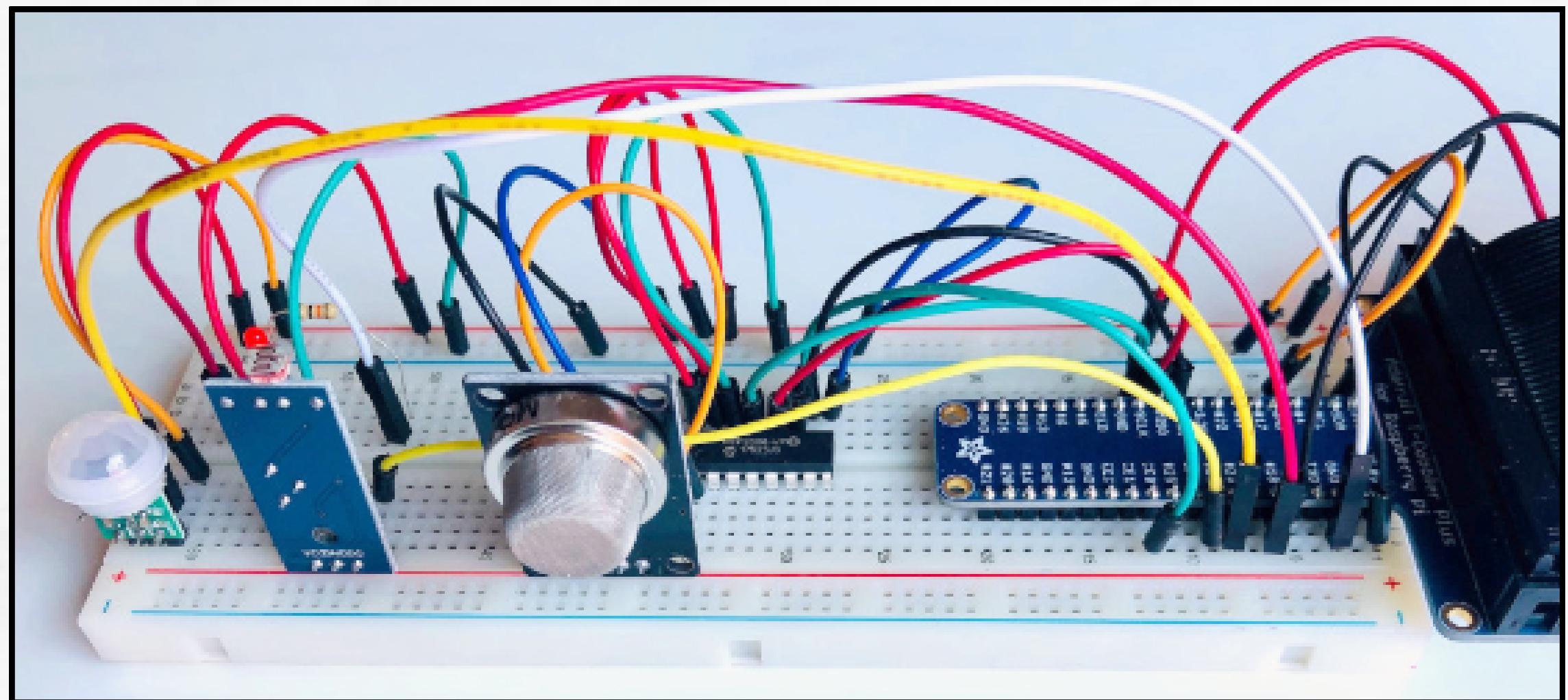
column	description	units
ts	timestamp of event	epoch
device	unique device name	string
co	carbon monoxide	ppm (%)
humidity	humidity	percentage
light	light detected?	boolean
lpg	liquid petroleum gas	ppm (%)
motion	motion detected?	boolean
smoke	smoke	ppm (%)
temp	temperature	Fahrenheit

```
{  
  "data": {  
    "co": 0.006104480269226063,  
    "humidity": 55.099998474121094,  
    "light": true,  
    "lpg": 0.000895956948783413,  
    "motion": false,  
    "smoke": 0.023978358312270912,  
    "temp": 31.799999237060547  
  },  
  "device_id": "6e:81:c9:d4:9e:58",  
  "ts": 1594419195.292461  
}
```

Les lectures du capteur, ainsi qu'un identifiant unique de périphérique et un horodatage, ont été publiées sous la forme d'un message unique, à l'aide du protocole réseau MQTT standard ISO. Vous trouverez ci-dessous un exemple de charge utile de message MQTT.

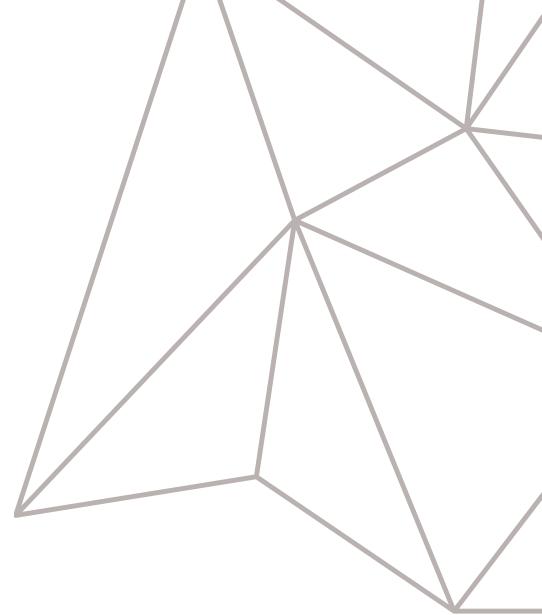
# Base de données :

Chaque appareil IoT a collecté un total de sept lectures différentes provenant des quatre capteurs à intervalle régulier. Les lectures du capteur incluent la température, l'humidité, le monoxyde de carbone (CO), le gaz de pétrole liquéfié (GPL), la fumée, la lumière et le mouvement. Les données couvrent la période du 12/07/2020 00:00:00 UTC au 19/07/2020 23:59:59 UTC. Il y a un total de 405 184 lignes de données.

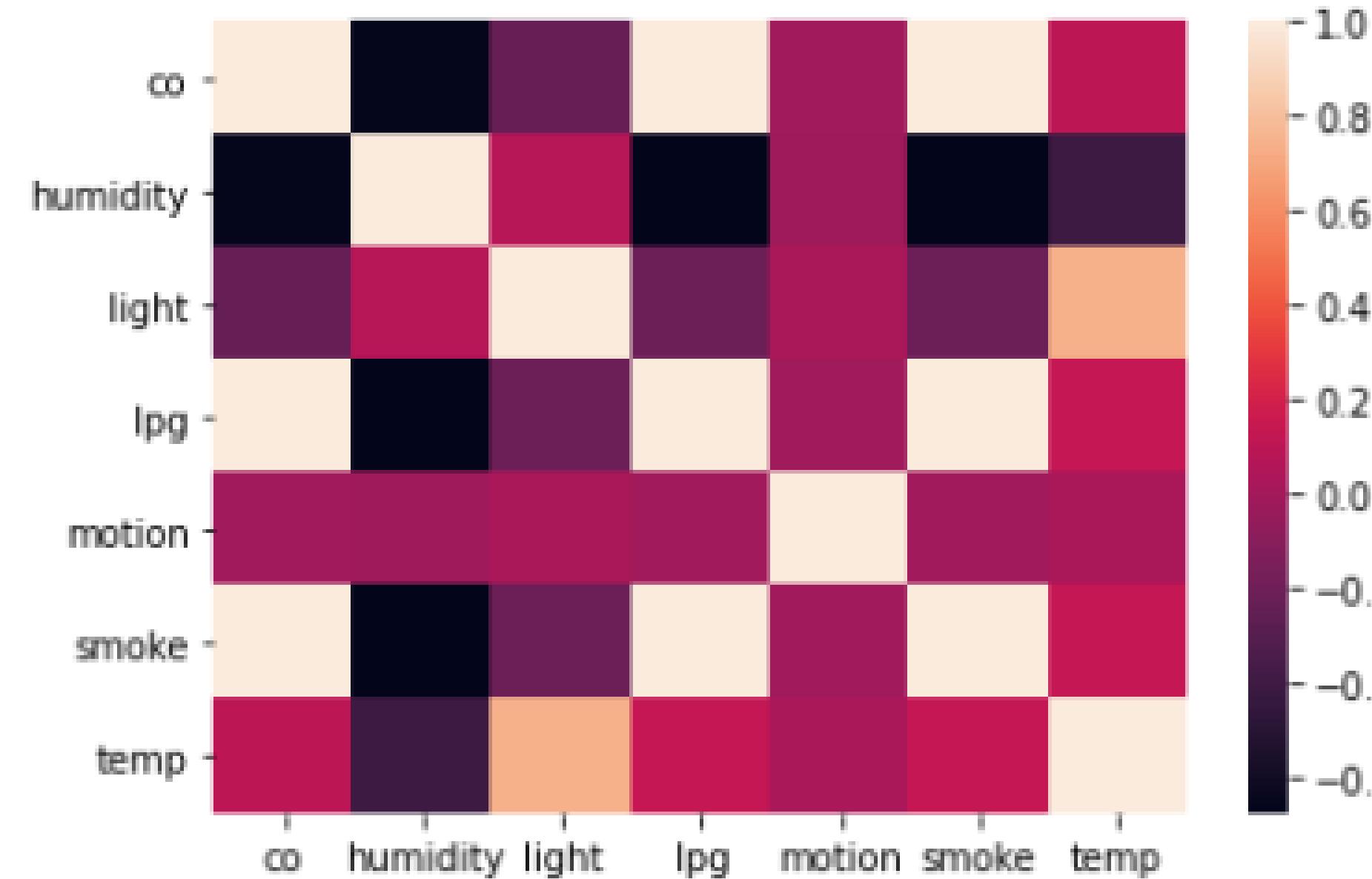


*Les appareils IoT*

# Prétraitement des données

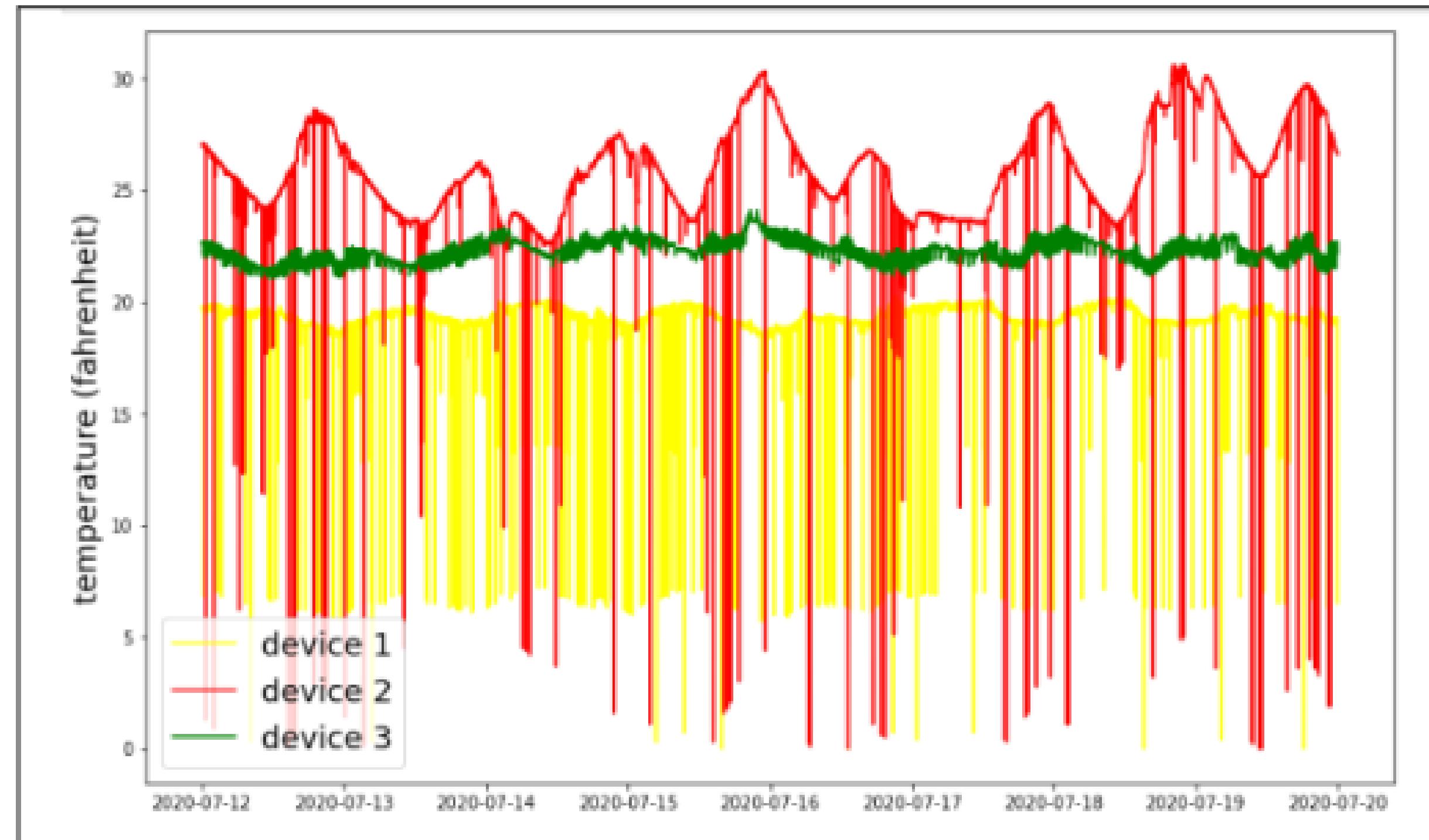


<matplotlib.axes.\_subplots.AxesSubplot at 0x7c306e549a10>



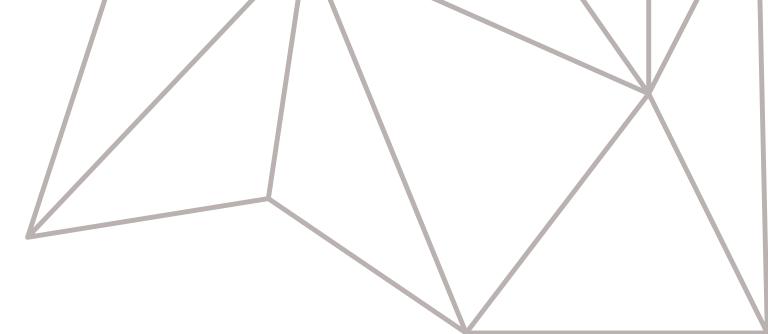
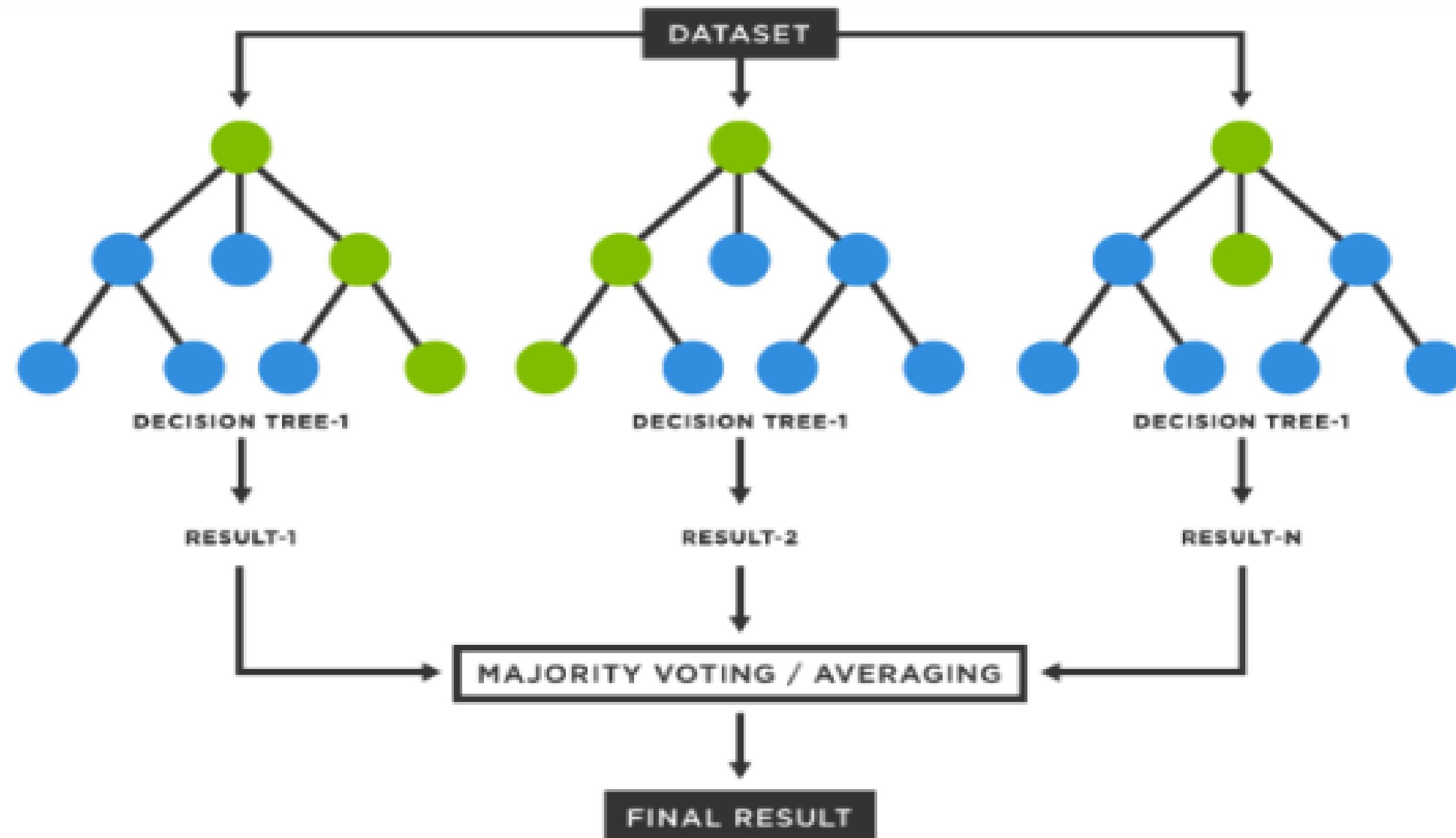
*Matrice de corrélation*

# Prétraitement des données



# Model Selection

## Modèle Random Forest



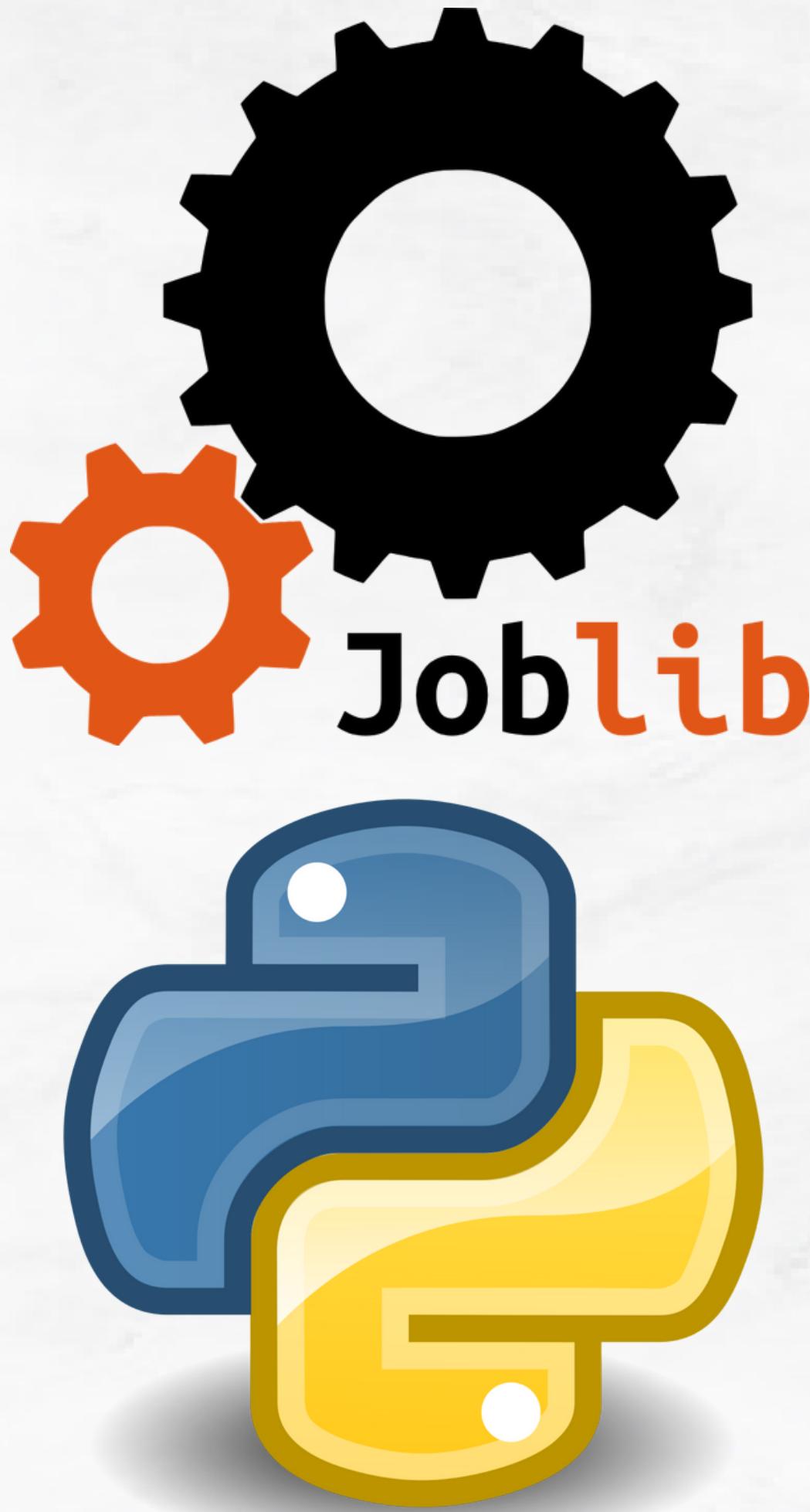
# Formation et Évaluation du Modèle

---

Le modèle a été formé en utilisant ces données historiques pour apprendre les relations et les tendances entre les différentes variables et la température. Une partie des données a été réservée pour l'évaluation, afin de mesurer la précision du modèle.



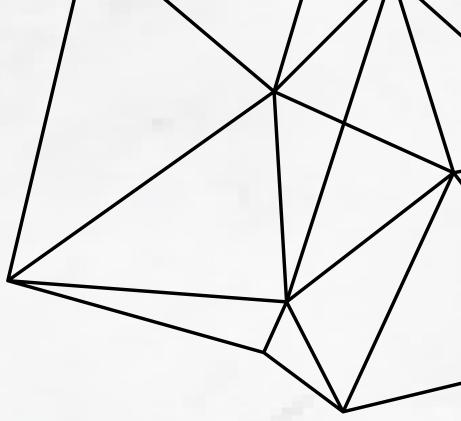
Et pour la métriques d'évaluation du Modèle nous avons utilisé l'erreur quadratique moyenne (RMSE) et la validation croisée pour déterminer la qualité de la prédiction



## Déploiement du Modèle dans la Plateforme Virtuelle

Le modèle est déployé dans l'environnement Python à l'aide de la bibliothèque Joblib, ce qui permet une intégration transparente dans le projet. Il est en mesure de générer des prédictions en temps réel pour garantir la continuité des opérations, même en cas de défaillance d'un nœud.

Lorsqu'un nœud réel tombe en panne, le Cloud peut faire appel au modèle pour prédire la température en utilisant les autres données disponibles. Cela permet de maintenir la surveillance et le contrôle de l'environnement du Greenhouse même en cas de panne d'un nœud.

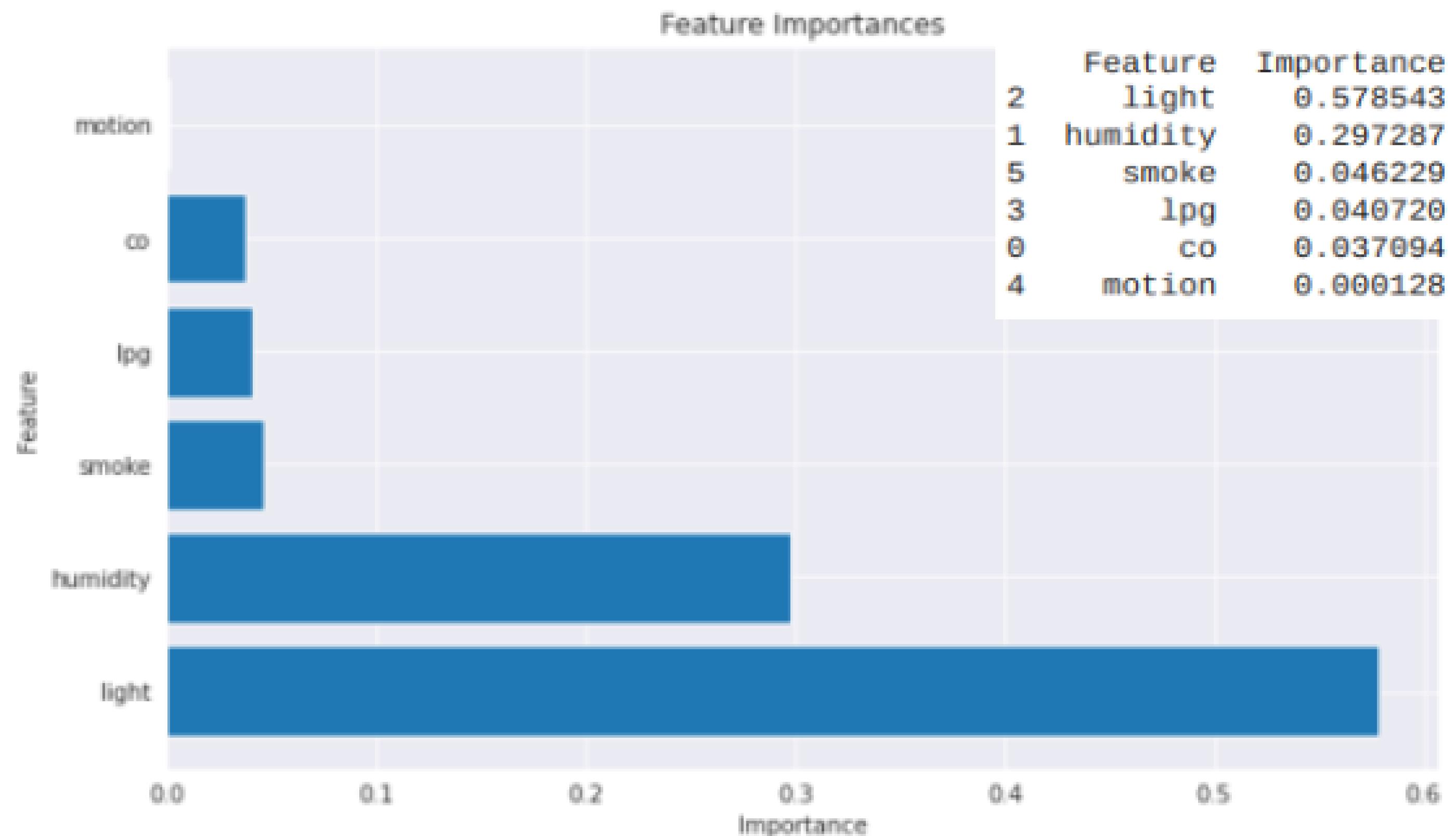


# Résultats

---

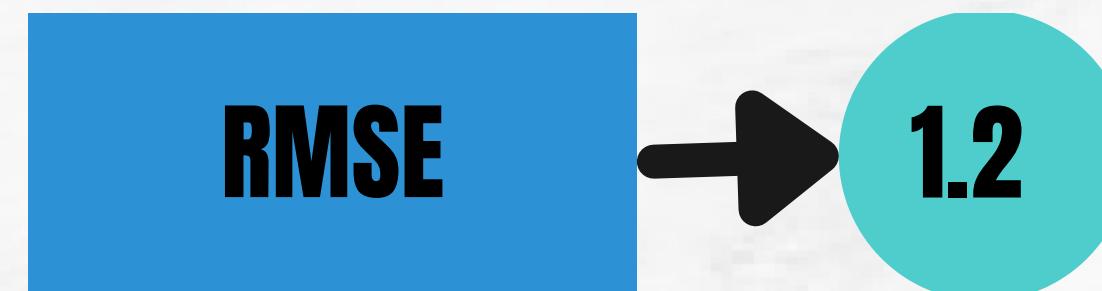


# Contribution des Variables à la Prédiction de Température

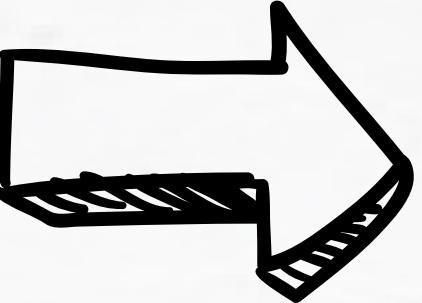


# Performances du Modèle

Le modèle Random Forest entraîné pour la prédiction de la température a atteint des performances remarquables. Il a obtenu une précision de 90 %, ce qui signifie qu'il prédit correctement la température dans la plupart des cas. De plus, la racine carrée de l'erreur quadratique moyenne (RMSE) s'est établie à 1,2. Le RMSE est une mesure de l'écart entre les valeurs prédites par le modèle et les valeurs réelles, et une valeur de 1,2 indique que le modèle est en mesure de fournir des prédictions précises et fiables.



# Utilisation du Modèle pour la Prédiction en temps réels



```
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud1\node1>python .\main.py
Les données sont publier
```

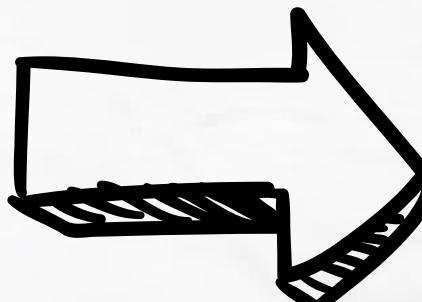
```
C:\Users\DELL\Downloads\Compressed\noeudes>python .\cloud.py
les données de noeud1 sont sauvegarder.
les données de noeud2 sont sauvegarder.
les données de noeud3 sont sauvegarder.
les données de noeud4 sont sauvegarder.
```

```
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud2\node2>python .\main.py
Les données sont publier
```

```
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud3\node3>python .\main.py
Les données sont publier
```

```
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud4\node4>python .\main.py
Les données sont publier
```

# Utilisation du Modèle pour la Prédiction en temps réels



```
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud1\node1>python .\main.py
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud1\node1>python .\main.py
Une Exception arrête la publication des données
Une Exception arrête la publication des données

C:\Users\DELL\Downloads\Compressed\noeudes>python .\cloud.py
les données de noeud2 sont sauvegarder.
erreur.....
C:\Users\DELL\Downloads\Compressed\noeudes>python .\cloud.py
les données de noeud1 sont sauvegarderavec la prédiction
les données de noeud2 sont sauvegarder.
les données de noeud3 sont sauvegarder.
les données de noeud4 sont sauvegarder.

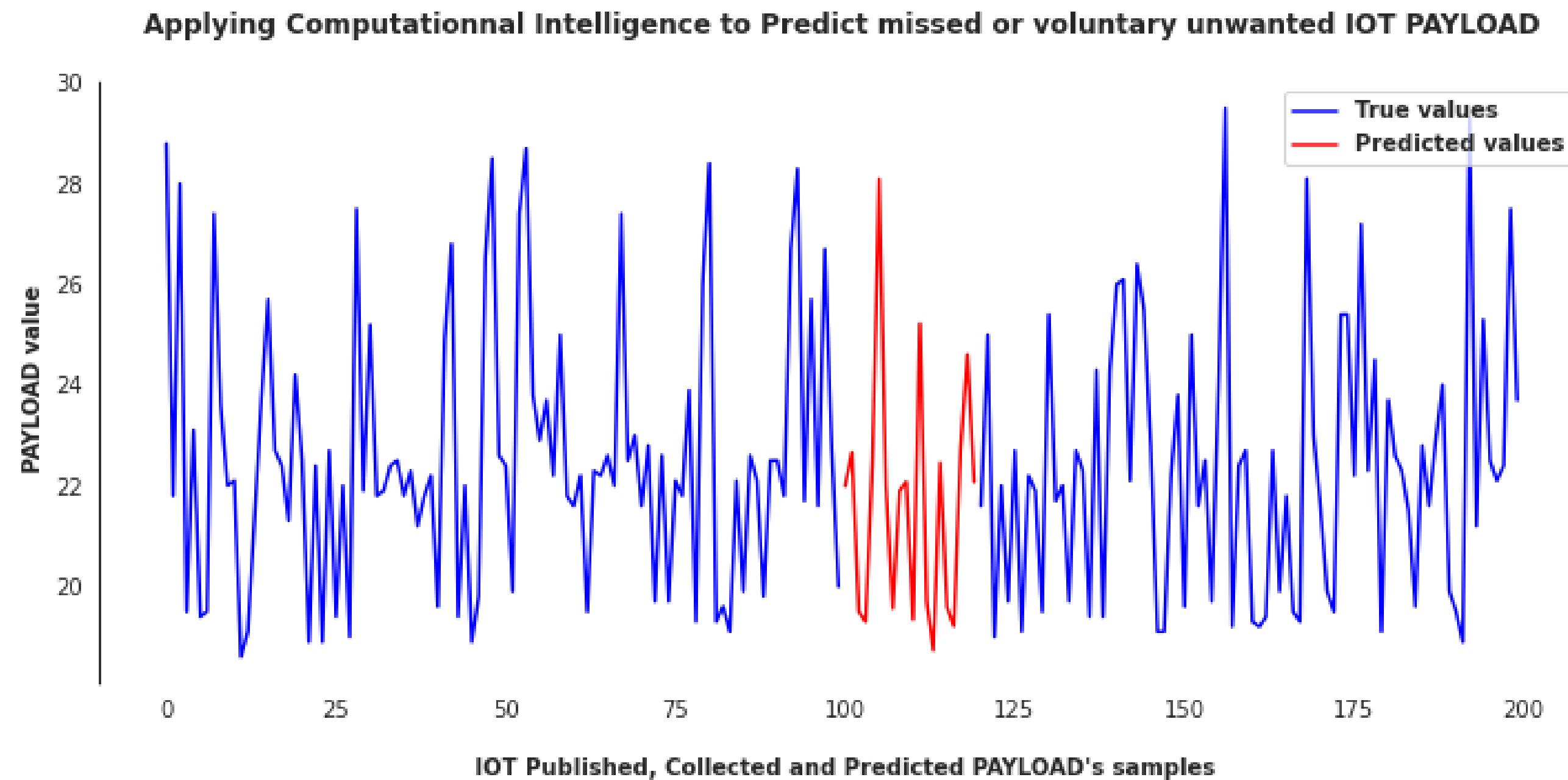
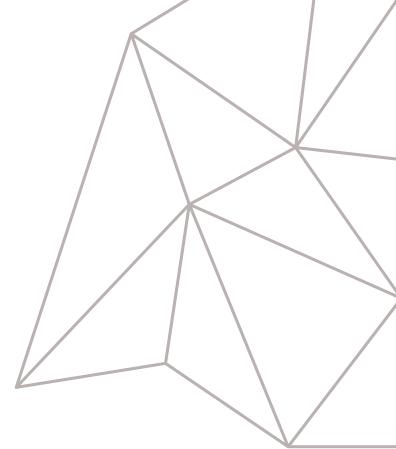
les données de noeudi1 sont sauvegarderavec la prédiction
les données de noeud2 sont sauvegarder.
les données de noeud3 sont sauvegarder.

C:\Users\DELL\Downloads\Compressed\noeudes\Noeud2\node2>python .\main.py
une Exeption arrête la publication des données...
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud2\node2>python .\main.py
Les donneés sont publier
Les donneés sont publier
Les donneés sont publier

C:\Users\DELL\Downloads\Compressed\noeudes\Noeud3\node3>python .\main.py
Les donneés sont publier
Les donneés sont publier
Les donneés sont publier
une Exeption arrête la publication des données...
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud3\node3>python .\main.py
Les donneés sont publier
Les donneés sont publier

C:\Users\DELL\Downloads\Compressed\noeudes\Noeud4\node4>python .\main.py
Les donneés sont publier
une Exeption arrête la publication des données...
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud4\node4>python .\main.py
Les donneés sont publier
```

# Visualisation d'une simulation



# Simulation

```
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud1\node1>python .\main.py
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud2\node2>python .\main.py
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud3\node3>python .\main.py
C:\Users\DELL\Downloads\Compressed\noeudes\Noeud4\node4>
```

# Recommandations

---

1. Exploiter des bases de données plus exhaustives.
2. Améliorer l'efficacité des modèles de machine learning.
3. Intégrer des capteurs offrant une précision en temps réel supérieure.
4. Augmenter la flexibilité en temps réel en élargissant le nombre de nœuds.

# CONCLUSION

---

En conclusion, le projet Greenhouse IoT a été une aventure captivante qui a permis de relever des défis majeurs de l'agriculture grâce à l'intégration de technologies avancées. Nous avons réussi à créer une plateforme robuste et innovante pour la surveillance en temps réel des paramètres environnementaux dans les serres agricoles. Les résultats obtenus, que ce soit dans la collecte de données en temps réel, l'utilisation de modèles de machine learning pour la prédition, ou encore la visualisation web des données, démontrent le potentiel considérable de cette approche. Ce projet ouvre des perspectives prometteuses pour l'avenir de l'agriculture intelligente et durable.

Merci pour votre attention !

