

BALISE DE SURVEILLANCE DE LA QUALITÉ DE L'AIR INTÉRIEUR ET DU CONFORT THERMIQUE

PROJET E4 2019-2020

RÉALISÉ PAR :

BHATTI THAYEEBA
CONSTANT VALENTIN
DALI CYRINE
FENG YUNHAN
LABA RITA
LECONTE LOÏC
NGASSA NELLY

ENCADRÉ PAR:

ALGRE EMMANUELLE
PAGAZANI JULIEN

SOMMAIRE

REMERCIEMENTS	4
INTRODUCTION	5
PRÉSENTATION DE LA BALISE	6
MANAGEMENT DE PROJET	7
CAHIER DES CHARGES	9
ÉTAT DE L'ART	10
<i>Etat de l'art du marché concurrentiel</i>	10
<i>Etat de l'impact de la qualité de l'air sur la santé</i>	12
Les Composés Organiques Volatiles (COV)	13
Les particules fines	13
L'humidité	14
Causes et conséquences d'une augmentation d'humidité dans un logement.	14
Causes et conséquences d'une baisse d'humidité dans un logement	15
La température	15
Causes et conséquences d'une augmentation de température dans un logement	15
Causes et conséquences d'une baisse de température dans un logement	16
Le dioxyde de carbone (CO2)	16
<i>Etat de l'art sur le transfert de données</i>	16
Envoi des données en ligne avec ThingSpeak	16
Transfert des données de la carte Arduino vers ISIS	17
<i>Etat de l'art sur la création d'une base de données</i>	18
a) Qu'est ce qu'une base de données ?	18
b) Quels sont les différents types de base de données ?	18
c) Comment créer une base de données ?	19
Création d'une base de données MYSQL	19
RÉALISATION DU PROJET	21
<i>Sélection de la carte</i>	21

Sélection des capteurs	21
<i>Partie 1 : Assemblage des capteurs de mesure de la qualité de l'air intérieur et du confort thermique.</i>	
Test des capteurs	22
Capteur de Température et d'Humidité Gravity DFR0066	22
Capteur de Gaz HCHO Gravity SEN0231	22
Capteur de CO2 infrarouge Gravity SEN0219	24
Capteur de poussières Gravity SEN0177	25
Le Capteur	25
Le Ventilateur	26
Pièce CATIA créée	27
Capteur de qualité de l'air ADA3709	27
Assemblage des capteurs sans module Wifi	27
Problèmes rencontrés	28
<i>Partie 2 : Récupération des données issues des capteurs sur le module WiFi & transfert et visualisation sur ThingSpeak.</i>	29
Test du module WiFi	29
Transfert des données	30
1) Transfert des données au hasard	31
2) Transfert des données des capteurs	32
3) Montage et broches utilisées pour la récupération des données	34
4) IFTTT	35
Problèmes Rencontrés	38
V. Partie 3 : Création de la balise	39
Création d'un PCB	39
Modélisation des capteurs	42
Création d'un support pour la balise	44
Problème rencontré	47
VI. Partie 4 : Campagne de mesures & Analyse des données	47
VII. Vidéo de Présentation	71
CONCLUSION	72

ANNEXES	73
Annexe 2 : Répartition de l'équipe projet.	74
Annexe 3 : Tableau dynamique de suivi de tâches.	76
Annexe 4 : Code utilisé pour le capteur de température et d'humidité seul.	77
Annexe 5 : Code utilisé pour le capteur de HCHO seul.	77
Annexe 6 : Code utilisé pour le capteur de CO2 seul.	79
Annexe 7 : Code utilisé pour le capteur de poussières seul.	80
Annexe 8 : Code utilisé pour le capteur de qualité de l'air seul.	82
Annexe 9 : Code utilisé pour l'assemblage des capteurs.	83
Annexe 10 : Code utilisé pour le module wifi seul.	86
Annexe 11 : Code final utilisé pour l'ensemble des capteurs	88
Annexe 12 : Code pour enlever les zones d'erreurs	96
Annexe 13 : Code pour calculer la valeur la plus basse / la plus haute	96
Annexe 14 : Code pour calculer les valeurs moyennes	98
BIBLIOGRAPHIE	101

REMERCIEMENTS

Nous tenons à remercier Mme BERLAND et toutes les personnes qui ont contribué de près ou de loin à la recherche des projets pour offrir à chaque étudiant leurs deux premiers voeux.

Ensuite, nous adressons nos remerciements à notre professeur encadrant Mme ALGRE qui nous a aidé tout au long de notre projet. Son écoute et ses conseils nous ont permis d'élaborer une balise permettant de mesurer la qualité de l'air intérieur de manière optimale tout en surveillant le confort de l'utilisateur.

Un merci bien particulier adressé également à M. PAGAZANI, qui nous a aidé lors de la phase de conception et de développement de la balise grâce à ses connaissances en électronique. Son aide fut très précieuse notamment lors des difficultés rencontrées lors de la réalisation de cette balise.

INTRODUCTION

La pollution atmosphérique est devenue un enjeu majeur de santé publique. En effet, au vu des connaissances actuelles, de nombreuses études démontrent que la pollution de l'air a un impact sur la vie d'un individu résidant depuis plus de 30 ans dans une agglomération française : son espérance de vie peut être réduite de 5 à 7 mois. De plus, elle peut ainsi être à l'origine de nombreux problèmes respiratoires (toux, asthme) et d'irritations nasales, des yeux et de la gorge. [1]

De plus en plus de foyers veillent donc à la qualité de leur air intérieur, notamment lorsqu'il y a présence d'enfants ou de personnes ayant des problèmes respiratoires. De plus, cette nécessité de contrôler la qualité de l'air intérieur doit s'accompagner d'un confort thermique en conservant une certaine température et humidité. [1]

C'est donc dans ce contexte que nous a été confiée la tâche de concevoir une balise permettant de mesurer la qualité de l'air intérieur et du confort thermique.

Nous avons réalisé cette balise dans le cadre d'un projet en quatrième année d'école d'ingénieurs à ESIEE PARIS. Ce projet a duré neuf mois, du 3 Octobre 2019 au 3 Juillet 2020.

PRÉSENTATION DE LA BALISE

Pour vous assurer de la qualité de l'air qui vous entoure et vous offrir un environnement confortable, nous avons créé une balise de surveillance de la qualité de l'air et du confort thermique.

Nous avons tout d'abord réfléchi aux indicateurs qui étaient indispensables de mesurer afin de garantir un air de bonne qualité et un confort thermique adéquat. Nous nous sommes également basés sur les valeurs guides de la qualité de l'air intérieur données par l'ANSES Agence Nationale de Sécurité Sanitaire de l'Alimentation, de l'Environnement et du Travail.

À partir de cette base, nous avons donc créé une station connectée qui permet d'analyser l'air ambiant en mesurant les niveaux de température, humidité, dioxyde de carbone, composés organiques volatils ainsi que des particules fines à l'aide de capteurs environnementaux. Les valeurs recueillies sont transmises via WiFi à ThingSpeak, une application open-source dédiée à l'Internet des Objets, qui permet également une visualisation des données et une analyse selon des critères que l'on peut définir. Si l'analyse des données montre une mesure en dehors de l'intervalle choisi, une alerte est envoyée via IFTTT une application disponible via ThingSpeak au propriétaire de la balise qui pourra mettre en place des actions visant à améliorer sa qualité de l'air intérieure et son confort.

MANAGEMENT DE PROJET

Pour mener à bien ce projet, nous avons commencé par utiliser la méthode du QOQCCP : Quoi ? Qui ? Où ? Quand ? Comment ? Combien ? Pourquoi ?

Cela nous a permis à la fois de définir le périmètre du projet, tout en nous permettant de prendre conscience des enjeux et de nous approprier celui-ci. Nous avons donc commencé par le « Quoi ? », ou plus exactement, qu'est ce qu'une balise de surveillance thermique ? Nous nous sommes donc intéressés à ce qui existait déjà sur le marché, aux différentes spécifications techniques et nous avons donc rédigé un état de l'art du marché concurrentiel qui nous servira de base pour établir notre stratégie vis-à-vis de l'existant.

Deuxièmement, nous sommes donc passés à la question « Qui ? » ou plutôt, qui seront les utilisateurs de cette balise ? Nous avons ainsi sélectionné le segment que nous souhaitions viser et cela nous a conforté dans notre première proposition.

Nous sommes donc passées à l'Où ? soit à « Dans quel environnement notre balise évoluera-t-elle ? » Nous avons choisi de nous restreindre à l'étude du confort thermique et de la qualité de l'air intérieur, étant donné que l'Humain reste 85% de son temps entre 4 murs.

Vient ensuite la question du « Quand ? ». Cette question nous a grandement aidé à établir les différents jalons pour réussir ce projet et à nous fixer un planning, avec des tâches et des objectifs.

Concernant le « Combien ? », nous souhaitions commencer par la réalisation d'une seule balise. Et enfin, pour le « Pourquoi ? », car nous n'étions pas satisfaits par ce que le marché proposait.

Une fois la réponse apportée à chacune de ces questions, nous étions désormais en mesure de comprendre l'envergure du sujet et ce que nous souhaitions apporter comme contributions. Par la suite, nous avons donc défini les différentes sous-tâches nécessaires à la réalisation du projet, sous la forme d'un diagramme de Gantt et d'un tableau PERT, élaborés par le chef de projet. Cela nous a permis d'avoir un travail bien organisé et structuré mais aussi de définir une échelle de temps pour chaque sous-tâche afin d'être en mesure de respecter le délai imparti. Afin de faciliter la lecture, les sous-tâches ont été rassemblés en plusieurs « blocs ». Chaque membre du groupe projet est responsable d'un ou plusieurs « blocs » et ce pour faciliter la prise de décisions et de coordination.

A chaque sous-tâche correspond également une barre d'avancement afin d'évaluer la progression de celle-ci. Vous trouverez le diagramme de Gantt en Annexe 1.

Ensuite, nous nous sommes répartis les sous-tâches en fonction du besoin en compétences de celles-ci. Cependant, notre groupe projet était assez mal constitué dans le sens où nous manquons de compétences en informatique, ce qui était assez pénalisant pour la partie « Crédit de base de données ». Nous nous sommes donc répartis à la fois selon nos compétences mais également en fonction de la difficulté des tâches et de nos envies, ce qui nous a conduit à la répartition que vous trouverez en Annexe 2.

Afin de rester informés de notre progression et des différents points bloquants, nous avons donc décidé de tenir un journal de bord que nous mettions à jour chaque séance pour définir nos objectifs mais également pour remonter les problèmes rencontrés lors de chaque séance, ainsi que l'évolution de notre réflexion sur le projet. De même, nous avons mis en place un tableau de tâches dynamiques afin d'avoir un aperçu visuel des tâches terminées, en cours et à faire. Vous retrouverez celui-ci en Annexe 3.

Malheureusement, dû aux conditions sanitaires liées au Covid 19, l'évolution du projet a été très compliqué puisque certaines tâches devenaient dès lors impossible à effectuer et malgré la mise en place du télétravail pour poursuivre au mieux le planning initial, cela ne nous a pas permis de respecter le diagramme de Gantt initial.

Après le déconfinement, certains stages ont été maintenus. De plus, en Mai, nous avons eu le choix entre choisir un autre projet ou continuer celui là jusqu'à la fin du mois de Juillet. Nous avons donc continué à 3 (Cyrine, Rita et Yunhan) ce projet. Il a donc fallu qu'on s'échange le matériel pour que tout le monde puisse réaliser la campagne de mesures ou pour effectuer d'autres tests.

CAHIER DES CHARGES

Le cahier des charges (ou CDC) établi en amont par nos tuteurs nous a permis de comprendre les attentes, les spécificités et les contraintes de ce projet. En d'autres termes, c'est à travers le cahier des charges que nous avons pu apercevoir la charge de travail et la répartition des tâches.

1) Résultats attendus

- Choix de capteurs pertinents pour la surveillance du confort thermique et de la qualité de l'air intérieur
- Réalisation d'une balise connectée intégrant les différents capteurs, l'acquisition et l'enregistrement des données
- Réalisation d'un serveur de stockage et d'affichage web
- Récolte de données issues de différentes conditions environnementales (en intérieur)
- Traitement et comparaison des données obtenues

2) Les livrables

- Etat de l'art des dispositifs existants (3 semaines max)
- Cahier des charges pour la réalisation d'une balise (2 semaines max)
- Caractérisation/Etalonnage des capteurs (4 semaines)
- Intégration sur carte électronique (3 semaines max)
- Balise intégrant les capteurs et le système d'acquisition dans un boîtier (8 semaines)
- Système d'affichage et de stockage (8 semaines)
- Mesures et analyse des données (8 semaines max)

ÉTAT DE L'ART

Afin de réaliser le projet, nous avons dû faire des recherches concernant les balises déjà existantes pour faire un choix judicieux concernant les capteurs à utiliser et pour nous informer sur les balises existantes sur le marché. De plus, pour stocker les valeurs obtenues, nous avons dû créer une base de données et trouver les différents types de transferts de données ainsi que les méthodes utilisées habituellement pour les créer.

1) **Etat de l'art du marché concurrentiel**

Dans un premier temps, nous avons recherché des balises permettant de mesurer la qualité de l'air existantes et vendues dans le commerce. Nous les avons regroupés dans un tableau afin de pouvoir les comparer. Certaines utilisent une application qui affiche les données recueillies sous forme de graphique dans le but d'observer leur évolution dans le temps. D'autres se différencient en ayant une interface qui permet d'observer directement les valeurs brutes.

Voici le tableau récapitulatif des différentes balises existantes et leurs caractéristiques :

Nom Balise :	Prix :	Capteurs :	Caractéristiques :	Sensibilité :
Netatmo [2]	99,99€	- Température - Humidité - Bruit - CO2	Application qui envoie des notifications en temps réel et contrôle plusieurs capteurs pour avoir le suivi permanent de plusieurs pièces.	Non référencé
Eolesens [3]	Non référencé	- Dioxyde de carbone - Humidité - COV - Particules fines	Possède un indicateur lumineux. Les données recueillies sont disponible sur la tablette EOLETTOUCH.	Non référencé
NeMo [4]	Non référencé	- Formaldéhyde - CO2 - COV - Température - Humidité - Pression	Détermine les actions à mener Fonctionne sur batterie, dispose d'une mémoire interne ainsi que d'une compatibilité avec les réseaux IoT. Possède une plateforme pour la gestion des données en temps réel (Nemo cloud). Possède des capteurs nanoporeux. On peut améliorer les capteurs selon nos besoins. → Cependant, la balise a un rôle davantage industriel que domotique.	CO2 : 0 à 5000 ppm Résolution : 1 ppm COV : 30 ppb à 5 ppm Résolution : 1 ppb Température : -55°C à +125°C Résolution : 0,08°C Humidité : 0 à 95 % Résolution : 0,08 % Pression : 260 à 1260 hPa Résolution : +/- 0,02 hPa
HugOne [5]	129,99€ (Darty)	- Température - Humidité	Solution complète intégrant à la fois le suivi du sommeil de toute la famille et de l'environnement intérieur avec restitution des	Non référencé

			données au sein de l'application pour Android et iOS.	
Foobot Home [6]	199,90€ (Darty)	- Température - Humidité - Particules fines (PM2.5) - COV - CO2 - CO	Actions à mener via des notifications Envoi d'alerte lors de donnée anormale. Les capteurs mesurent 24h/24 et 7j/7 la qualité de l'air intérieur. Compatibilité iOS et Android.	Température : 1 à 45°C Résolution : 1°C Humidité : 30 à 80 % Résolution : +/- 5% PM2.5 : 0 à 1 300 µg/m ³ Résolution : +/- 4µg/m ³
Eve Room [7]	99,95€	- Température - Humidité - COV	L'application Eve affiche toutes les mesures sous forme de graphiques détaillés par heure, jour, semaine, mois ou même année. Les actions sont à mener via l'application. Balise sans fil.	Température : 0°C à 50°C Résolution : +/- 0,3°C en condition normale Humidité : 5 à 95% Précision : +/- 3% en condition normale
Airthings wave [8]	199€	- Radon - Température - Humidité	Résultats rapides en 1h. Indicateur lumineux. Très précis car enregistre les niveaux sur de longues périodes. Petit et fonctionnant sur piles avec une durée de vie de 2 ans.	Radon : < 150 bq/ m ³ ≥ 100 et < 150 ≥ 150 Humidité : ≥ 30% et < 60% < 30% et ≥ 25% // < 70% et ≥ 60% < 25% et ≥ 70% Température : < 18 ≥ 18 et ≤ 25 > 25
Airthings Wave Plus [9]	269€	- Radon - CO2 - COV - Humidité - Température - Pression	Les caractéristiques sont identiques à Airthings wave sauf que celle-ci mesure aussi le niveau de COV et de CO2.	Radon : < 150 bq/ m ³ ≥ 100 et < 150 ≥ 150 COV : < 100 ppb ≥ 100 et < 500 ≥ 500 CO2 : < 800 ppm ≥ 800 et < 1000 ≥ 1000 Humidité : ≥ 30% et < 60% < 30% et ≥ 25% // < 70% et ≥ 60% < 25% et ≥ 70% Température : < 18 ≥ 18 et ≤ 25 > 25
IGERESS [10]	249€	- Formaldéhyde - COVT - PM1.0 /PM2.5 /PM10 - Température - Humidité	Multifonctionnel permettant de mesurer à la fois la température, l'humidité mais aussi le taux de formaldéhyde, de COV et de particules fines.	Formaldéhyde : 0 à 1.999 mg/m ³ COVT : 0 à 9,999 mg/m ³ PM1.0 / PM2.5 / PM10 : 0 à 999µg / m ³ Température : 5 à 45°C Humidité : < 90 %
Blueair Aware [11]	309€ (Darty)	- Particules fines - COV - CO2 - Température - Humidité	2 applications mobiles : → recueille des données chaque jour et affiche leurs évolutions par des graphiques → données issues de plus de 2700 stations de surveillance de la	PM2.5 : 1 à 500 µm/m ³ COV : 125 à 1000 ppb CO2 (basé sur la lecture des COV): 4500 à 5000 ppb Température : 0 à 50°C

			qualité de l'air dans 150 pays. Intervalle de 5 minutes entre chaque stockage des données vers le cloud.	Résolution : +/- 1°C Humidité : 25 à 75 % Résolution : +/- 5.5 %
Air Quality Egg 1 & 2 [12]	Intérieur : 130\$ +80\$ +45\$ +40\$ +60\$ Extérieur : 160\$	- Température - Humidité - Pression - Option en plus : CO2 - Option en plus : SO2, NO2 - Option en plus : CO, O3, COV - Option en plus : Particules fines	L'utilisateur peut choisir les données qu'il a envie de mesurer. Il permet aussi de mesurer les concentrations de dioxyde d'azote ce qui n'est pas commun dans les autres balises.	Humidité : 0 à 99% Température : 0 à 40°C

On remarque cependant que les "balises de surveillance" vendues actuellement dans le commerce ne sont pas réellement efficaces et n'indiquent pas à l'utilisateur les actions correctives à mener lorsque la qualité de l'air se détériore. En revanche, certaines d'entre elles proposent un affichage lumineux, la lumière changeante selon la qualité de l'air mesuré afin d'informer l'utilisateur sur l'état de son environnement.

De plus, la plupart des balises existantes se contentent d'intégrer et de mesurer la température ainsi que l'humidité présentent dans la pièce et se vantent ainsi d'être des balises de surveillance de la qualité de l'air. De même, les balises considérées comme performantes ont un prix relativement plus élevés que le prix moyen du marché et propose même à l'utilisateur plusieurs options afin que le produit corresponde à l'utilisation souhaitée.

Par conséquent, il nous semble pertinent d'aller davantage en profondeur par rapport à l'existant, notre balise devra à la fois informer l'utilisateur lorsque les grandeurs mesurées s'écartent des valeurs préconisées par les différentes institutions de la santé, mais devra également lui indiquer les actions à effectuer afin de pouvoir à nouveau vivre dans un environnement sain, à la fois au niveau du confort thermique qu'à son exposition face aux polluants. [13]

2) Etat de l'impact de la qualité de l'air sur la santé

En France, nous passons 80% de notre temps dans des environnements clos dans lesquels nous pouvons être exposés à plusieurs polluants, tels que des :

- polluants chimiques, comme les Composés Organiques Volatiles (COV)
- polluants naturels (moisissures, acariens, animaux domestiques,...)
- polluants domestiques (formaldéhyde, fibre artificielles,...) [14]

Cependant, d'autres facteurs extérieurs viennent également perturber la qualité de l'air intérieur, on peut notamment citer :

- les émissions de polluants provenant des secteurs d'activité (agriculture, industries, transports,...)
- les phénomènes d'origine naturelle (éruptions volcaniques, érosion des sols,...)
- les réactions chimiques s'opérant entre les divers composants chimiques
- les phénomènes d'importation et d'exportation de la qualité de l'air (une partie de la pollution extérieur provient des pays transfrontaliers et une partie de la pollution produite en France s'exporte également vers ses pays voisins) [15]

Une qualité de l'air jugée comme bonne à l'intérieur d'un bâtiment a démontré de nombreux effets positifs : amélioration des capacités de concentration, bien-être au travail, etc... En revanche, une mauvaise qualité de l'air favorise l'apparition de symptômes tels que des maux de tête, des sensations de fatigue, des irritations ainsi que des manifestations allergiques. [16]

Aujourd'hui, les enjeux sanitaires liés à la qualité de l'air intérieur sont importants, c'est pourquoi il devient important de mettre en œuvre des actions pour l'améliorer, que ce soit dans des logements ou dans les lieux publics.[17]

Nous avons donc choisi de nous intéresser de plus près aux différentes sources de dégradation de la qualité de l'air :

a) *Les Composés Organiques Volatiles (COV)*

Les Composés Organiques Volatiles (COV) ont la capacité de s'évaporer à température ambiante, ils se retrouvent donc tout naturellement dans l'air que nous respirons. Ils représentent une part importante des polluants que l'on retrouve dans les bâtiments neufs ou récemment rénovés. Ils proviennent majoritairement des produits ménagers et sont une source de menace pour notre santé en cas d'utilisation dans des endroits clos et peu ventilés. Une exposition prolongée est très néfaste pour notre santé, les COV ayant une capacité à se dégager sur le long terme. Le seuil limite recommandé est de $100 \mu\text{g.m}^{-3}$ au cours de la journée. [18]

b) *Les particules fines*

On appelle particules fines toute particule dont le diamètre est inférieur à $2.5 \mu\text{m}$. Il s'agit d'un des polluants les plus surveillés par les institutions de surveillance sanitaire en France. Ces particules sont très dangereuses pour notre santé : des effets possibles sur la reproduction, risque de naissance prématurée, atteintes au

développement neurologique de l'enfant et démence chez les personnes âgées. L'Agence Nationale de la Santé Publique en France (ANSP) estime que la pollution par les particules fines est responsable d'au moins 48 000 décès prématurés chaque année, soit 9% du taux de mortalité en France. En effet, plus les particules sont fines, plus elles sont capables de pénétrer profondément dans l'arborescence pulmonaire et donc de passer vers d'autres organes via la circulation sanguine. [19]

C'est pourquoi elles ont un fort impact sur la mortalité et la morbidité cardio-respiratoire. On retrouve notamment cette particularité comme étant à l'origine de nombreuses pathologies chroniques tels que des cancers ou des maladies cardiovasculaires et respiratoires après des expositions prolongées à ces particules, même à de faibles niveaux de concentration.[20]

c) *L'humidité*

L'humidité se définit tout simplement comme étant une vapeur d'eau pouvant s'imprégnner dans l'air d'un milieu. On distingue deux types d'humidité : l'humidité absolue et l'humidité relative. Dans le cadre de notre étude, on s'intéressera à l'humidité relative qui est le pourcentage de vapeur d'eau contenue dans l'air pour une température et une pression donnée.

Dans les bâtiments tertiaires et résidentiels, afin d'éviter les sensations d'inconfort, il est nécessaire de maintenir les pièces à une température de 19°C. Ceci correspond à une humidité relative dont la moyenne est de 45% mais qui pourrait changer en fonction de la saison. Il est donc important de garder un œil sur elle car la simple hausse ou baisse de cette humidité peut avoir un impact sur la santé. Cette norme est la plupart du temps respectée, mais malheureusement, on observe souvent une hausse et une baisse d'humidité dans certaines habitations.

1) *Causes et conséquences d'une augmentation d'humidité dans un logement.*

Les causes d'une augmentation d'humidité sont nombreuses. La pluie est un facteur important car l'eau s'infiltre sous l'effet de la pression hydrostatique du sol exercé contre les fondations. Aussi, une mauvaise ventilation pourrait augmenter le taux hygrométrique dans une pièce mal isolée. Les études ont montré que cette hausse d'humidité a un réel impact sur la santé de l'être humain tel que : des allergies à la poussière, des infections respiratoires chroniques ou des rhumatismes et même des problèmes d'articulation chez les vieilles personnes. Sur l'Homme, un taux d'humidité trop important augmente les risques d'allergies et d'asthme.

2) *Causes et conséquences d'une baisse d'humidité dans un logement*

Parallèlement, on peut observer une baisse d'humidité dans un logement due à un air très sec. Ceci a été constaté pendant les hivers très rudes durant lesquels on a tendance à chauffer excessivement les pièces. Dans certains logements très isolés, l'air circule peu et devient très rapidement asséché. Les conséquences de ce phénomène se traduisent par un dessèchement des muqueuses, une déshydratation, la fatigue et également une démangeaison cutanée. Sur l'Homme, une faible humidité peut entraîner des irritations de la gorge, des yeux et de la peau.

d) *La température*

La température quant à elle permet de mesurer le confort thermique dans un logement. Elle influe à la fois sur l'humeur, le niveau d'énergie et la qualité du sommeil d'un individu. Pour obtenir un confort thermique satisfaisant, la température ambiante de confort doit se situer entre 19°C et 20°C. De plus, il faut prendre en compte la température des parois car elle a une grande influence sur la température ressentie. Pour calculer simplement la température ressentie, il faut faire la moyenne entre la température des parois et la température ambiante. Par exemple, pour une température d'ambiance de 20°C :

- Cas n°1 → Température de paroi de 16°C : la température ressentie sera de 18°C.
- Cas n°2 → Température de paroi de 19°C : la température ressentie sera de 19,5°C.

Des solutions sont possibles afin d'augmenter la température des parois : un isolement correct du logement en limitant le plus possible les ponts thermiques (points de jonction où l'isolation n'est pas continue et qui provoquent des pertes de chaleur) et la mise en place de vitrages. [21]

1) *Causes et conséquences d'une augmentation de température dans un logement*

Les températures trop élevées ont tendance à faire perdre de l'énergie. Une perte de concentration et un manque de sommeil sont aussi observés. Cette hausse de température également appelée hyperthermie est notamment liée à un chauffage excessif chez l'habitant.[22] De plus, durant l'été, la température grimpe très vite ce qui peut faciliter un état de fatigue et une déshydratation dont le premier signe est la soif. La température corporelle est supérieure ou égale à 40°C. [23]

2) *Causes et conséquences d'une baisse de température dans un logement*

À l'opposé, les températures trop basses augmentent la pression artérielle et la fréquence cardiaque. Lorsque la température corporelle d'un individu est

inférieure à 35°C, on parle d'hypothermie. Les personnes les plus vulnérables à l'hypothermie sont les personnes âgées, les nourrissons et les jeunes enfants.[24]

e) Le dioxyde de carbone (CO₂)

Le CO₂ est l'un des principaux gaz à effet de serre. Sa concentration dans l'air intérieur est liée à l'occupation humaine et au renouvellement d'air, il est donc un indicateur de confinement de l'air. On surveille donc la concentration en CO₂ dans l'air intérieur non pas à cause de sa toxicité mais parce qu'il est l'un des critères qui fondent la réglementation en matière d'aération des locaux. Les seuils limites pour sa concentration sont entre 1 000 et 1 500 ppm.

Pour résumer, la pollution de l'air intérieure porte une responsabilité avérée dans l'apparition de maladies respiratoires et cardiovasculaires, notamment pour l'asthme. 25 à 30 % de la population française est touchée par cette maladie et une exposition à court et moyen terme peut être très dangereuse pour les personnes asmatisques : crises d'asthme, irritation des voies respiratoires, ...

Ainsi, une exposition prolongée à une mauvaise qualité de l'air conduit aux impacts les plus importants sur la santé et la part des effets sanitaires attribuables aux pics de pollution demeure très faible. Le problème réside davantage dans une exposition tout au long de l'année à des niveaux moyens de pollution. Aussi, en cas d'épisode de pollution de l'air, il est préférable de réduire ou d'éviter l'exposition à ces autres facteurs.[25]

3) Etat de l'art sur le transfert de données

Notre objectif est de transférer les données de la carte Arduino vers un serveur web.

a) Envol des données en ligne avec ThingSpeak

- Création d'un compte sur <https://thingspeak.com> [26]
- Puis création d'un New Channel avec 8 entrées pour les canaux (les fields). Par exemple pour le field 1 nous pouvons écrire Température, pour le field 2 Humidité etc.
- Aller dans le bureau et chercher le dossier "Arduino". Puis, cliquer sur le dossier avec le nom du channel et ouvrir le fichier sur Arduino [27]
- Revenir sur le compte de Thingspeak, dans "API Keys", copier/coller la clé (unique) et mettre dans le fichier Arduino dans `String writeAPIKey = "key=(....)"`
- Il faut mettre un délai de 15 secondes car le serveur va attendre une valeur chaque 15 secondes.

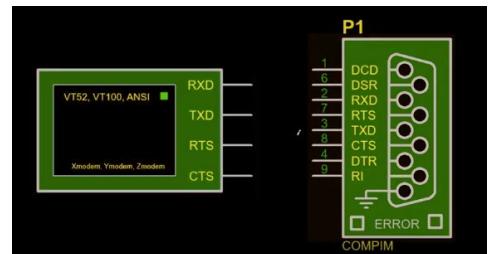
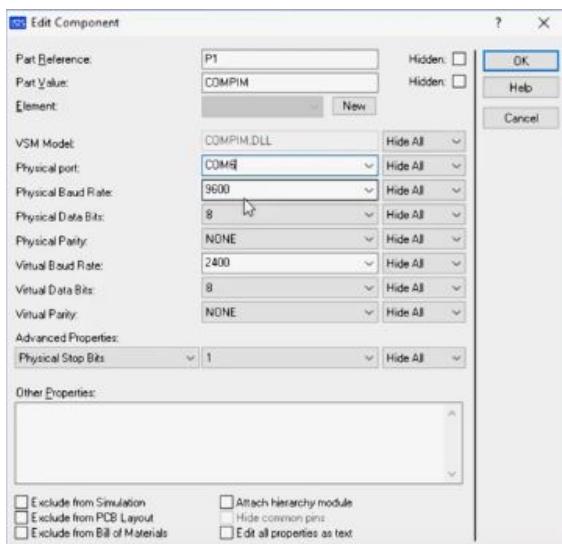
- Une fois le code téléchargé, retourner dans le compte Thingspeak et observer les données s'afficher en temps réel.

b) Transfert des données de la carte Arduino vers ISIS

Il s'agit ici du même procédé que précédemment mais avec une nouvelle interface : ISIS Proteus. [28]

Lorsque l'installation est terminée, il faut ajouter le port de communication virtuelle : Cliquez sur "P" (pour Pick Devices), tapez "Compim" puis le faire glisser à droite de l'interface.

Ensuite, il suffit d'ajouter le terminal de visualisation : "Virtual Terminal" que l'on va aussi glisser à droite de l'interface comme ceci :



L'étape d'après va être de relier les ports ensemble : (RXD - RXD et TXD - TXD)

Une fois la connexion établie, on configure l'interface ci-contre.

Configurons aussi l'interface de visualisation série : mettre "Baud rate" à 9600.

Avant de lancer la simulation, il faut brancher la carte Arduino Uno.

Retournons dans l'interface Arduino pour visualiser les données dans "Moniteur série". En cas d'erreur, débrancher et rebrancher la carte Arduino Uno. Pour observer de façon graphique les données, il faut aller dans "Traceur série".

4) Etat de l'art sur la création d'une base de données

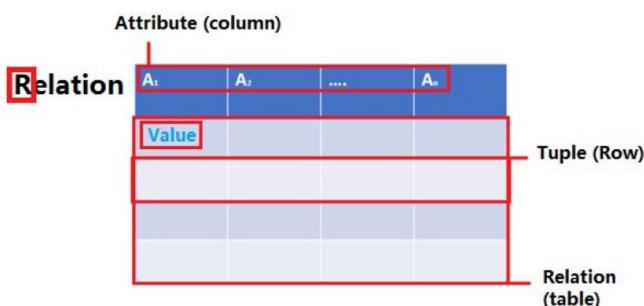
a) Qu'est ce qu'une base de données ?

Une base de données est une entité dans laquelle il est possible de stocker des données de façon structurée et avec le moins de redondances possible. Ces données doivent pouvoir être utilisées par des programmes, et ce par des utilisateurs différents.

b) Quels sont les différents types de base de données ?

On peut différencier les modèles de base de données couramment utilisés selon des critères tels que le développement technique, la transmission électronique des données ou bien encore l'efficacité. Nous allons ici vous présenter les modèles les plus utilisés.

- Base de donnée relationnelle :

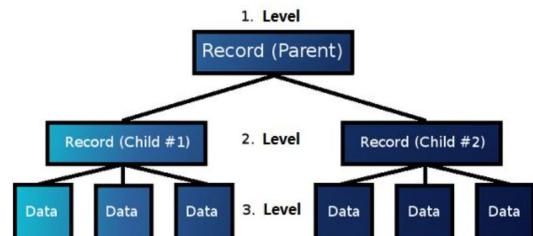


Le modèle de base de données relationnelle fonctionne avec des tables individuelles qui définissent la localisation et les liens entre les informations. Ces informations forment un ensemble de données (dans le diagramme d'une ligne ou d'un "tuple"). Les informations individuelles sont collectées sous forme

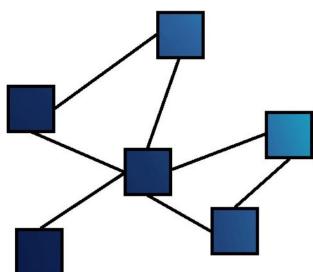
d'attributs (dans les graphiques A₁ à A_n) dans les colonnes. La relation totale ("relation" est souvent utilisée comme synonyme du terme "table") est dérivée d'attributs connexes. La clé primaire, (aussi appelée "ID"), qui est généralement définie comme le premier attribut (A₁) et ne doit jamais changer, est élémentaire pour l'identification unique d'un enregistrement de données. En d'autres termes, cette clé primaire définit la position exacte de l'ensemble de données avec tous ses attributs.

- Base de donnée hiérarchique :

Les bases de données hiérarchiques ont des dépendances très claires. Cela signifie que chaque enregistrement de données a exactement un prédécesseur (relation "parent-enfant"). Bien que chaque "enfant" ne puisse avoir qu'un seul "parent", chaque "parent" peut avoir un nombre quelconque d'"enfants". En raison de l'ordre hiérarchique strict, les couches qui ne sont pas directement adjacentes ne peuvent pas interagir les unes avec les autres. De plus, il n'est pas facile d'établir un lien entre deux arbres différents. Les structures hiérarchiques des bases de données sont donc extrêmement rigides et sont donc des alternatives peu fiables pour suivre l'évolution des systèmes d'information.



- Base de donnée de type réseau :



Contrairement au modèle hiérarchique, les données n'ont pas de relation "parent-enfant" stricte. Chaque enregistrement de données peut avoir plusieurs prédécesseurs, ce qui donne une structure de type réseau. De même, il n'existe pas de chemin d'accès unique à un enregistrement de données. Les ensembles de données peuvent être ajoutés et supprimés de

manière fluide dans le modèle de réseau sans interférer de manière significative avec la structure globale. Il s'agit donc d'une alternative pratique pour gérer l'évolution des systèmes d'informations. [29]

c) Comment créer une base de données ?

Il existe plusieurs méthodes pour créer une base de données :

1) Création d'une base de données MYSQL

- Créer la base de données.[30]

Dans la ligne de commande MySQL, entrez la commande pour créer une database. Nous l'avons appelé "etats_Amerique" : *CREATE DATABASE etats_Amerique;*

- Afficher la liste des bases de données.

Dans la ligne de commande MySQL, entrez la commande qui liste les bases de données disponibles sur le serveur MySQL *SHOW DATABASES;*

- Sélectionner la base de données.

Une fois la base de données créée, il faut la sélectionner afin de commencer son édition. Entrez la commande *USE etats_Amerique;* Un message s'affiche : Database changed, ce qui montre que la base de données active est désormais etats_Amerique.

- Création d'une table.

Au départ, elle est vide. C'est dans la table que nous allons entrer certaines informations. Pour en créer une, il faut commencer par la structurer avec la commande initiale. Pour créer un tableau, entrez la requête suivante : *CREATE TABLE etats (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, etat CHAR(25), population INT(9));* Ainsi une table intitulée "etats" est créée avec trois colonnes : id, etat et population.

- Création d'une entrée dans la table.

La commande pour entrer le premier État : *INSERT INTO etats (id, etat, population) VALUES (NULL, 'Alabama', '4822023');*

- Création d'autres entrées.

Il est possible de créer plusieurs entrées simultanément à l'aide de la commande : *INSERT INTO states (id, state, population) VALUES (NULL, 'Alaska', '731449'), (NULL, 'Arizona', '6553255'), (NULL, 'Arkansas', '2949131');*

- Exécutez une requête sur la base de données.

Pour obtenir toute la base de données, faire : *SELECT * FROM etats_amerique;*

Pour une requête plus spécifique (retourner la table par population croissante où la colonne id n'apparaîtra pas puisqu'on a choisi que les entrées etat et population) :

```
SELECT    etat,    population    FROM    etats_amerique    ORDER    BY  
population;
```

RÉALISATION DU PROJET

I. Sélection de la carte

Dès le début du projet, il a fallu choisir la carte que nous allons utiliser pour assembler tous les capteurs. De ce fait, nous avons beaucoup hésité entre choisir une carte Texas Instruments (TI) ou une carte Arduino Uno. La première idée était

d'opter pour une carte TI car notre école est la seule à disposer d'une bibliothèque 4.0 la "TI-Innovation Gateway", qui permet d'emprunter des composants de la marque TI pour réaliser des projets personnels ou pédagogiques. Cependant, nous n'avons jamais utilisé de carte TI et nous avons constaté un manque cruel d'information sur Internet ce qui nous aurait fortement ralenti dans la réalisation du projet. Nous avons donc préféré l'utilisation d'une carte Arduino Uno puisqu'en plus d'avoir une personne du groupe ayant déjà utilisé cette carte, de nombreux codes existent sur Internet concernant les capteurs que nous voulions utiliser.

II. Sélection des capteurs

Après avoir fait un état de l'art sur la santé afin d'évaluer ce qu'était un air de bonne qualité et après récupération des valeurs guides données par l'ANSES, il a fallu faire un choix concernant les capteurs.

Nous avons fait de longues recherches car des capteurs aux sensibilités que nous souhaitions avoir étaient soit trop difficiles à obtenir soit le montant total était hors budget (150€ maximum de budget alloué pour le projet).

Voici les capteurs que nous avons finalement commandés :

Capteur :	Prix :	Caractéristiques :	Commentaire :
Capteur de température et d'humidité Gravity DFR0066	8.90 €	Plage de mesures : - température: -40 °C à +128 °C - humidité: 0 à 100 % Précision : - température: ±0,3°C (à 25°C) - humidité: ±2,0 % (10 à 90% HR)	Librairie : SoftwareSerial
Capteur de poussières Gravity SEN0177	64.50 €	Plage de mesures : 0 à 500 µg/m³ Particule mini détectable : 0,3 µm Calibres : 0,3 à 1 µm - 1 à 2,5 µm - 2,5 à 10 µm	/
Capteur de gaz HCHO Gravity SEN0231	53.90 €	Plage de mesures : 0 à 5 ppm Résolution : 0,01 ppm	/
Capteur de qualité de l'air ADA3709	22.60 €	Plage de mesures : - eCO2: 0 à 60000 ppm - COVT: 0 à 60000 ppm Précision : 15 %	Interface : I2C
Capteur de CO2 infrarouge Gravity SEN0219	66.95 €	Plage de mesures : 0 à 5000 ppm Précision : ± (50 PPM + 3%)	Sortie analogique : 0,4 à 2 Vcc
Module ESP8266	8.60 €	Compatible WEB, WPA-PSK, WPA2-PSK Compatible protocole TCP/IP WiFi Direct (P2P), soft-AP	Interface : 2 ports GPIO, UART, SPI

→ Tous les capteurs ont été commandés sur le site Gotronic.

III. Partie 1 : Assemblage des capteurs de mesure de la qualité de l'air intérieur et du confort thermique.

Après réception des capteurs, nous les avons donc tous testés un par un dans un premier temps afin de vérifier leur bon fonctionnement. Nous allons donc vous présenter les montages, le codage ainsi que les résultats obtenus.

A. Test des capteurs

1) Capteur de Température et d'Humidité Gravity DFR0066

Il s'agit d'un capteur digital de température et d'humidité basé sur un SHT1x et permettant de mesurer la température ambiante entre -40°C et +128°C et l'humidité via deux sorties indépendantes.

Voici ci-contre le montage utilisé pour ce capteur.

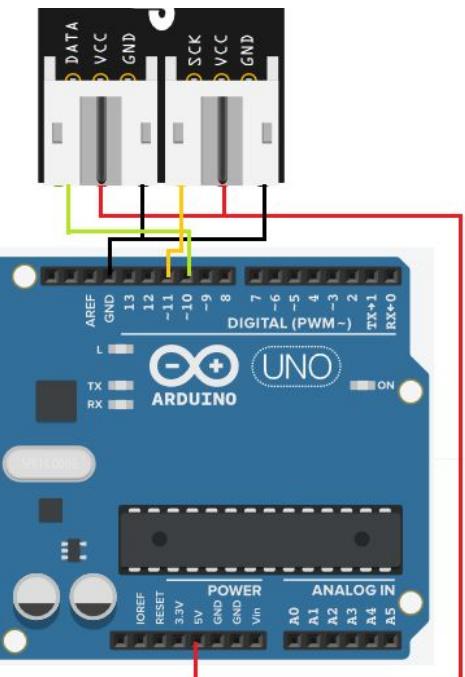
Le code utilisé pour le test du capteur seul est disponible en Annexe 4. [31]

Voici les résultats obtenus :

```
Temperature: 24.9199981689C / 76.9279937744F. Humidity: 47.47%
Temperature: 24.9199981689C / 76.9279937744F. Humidity: 47.47%
```

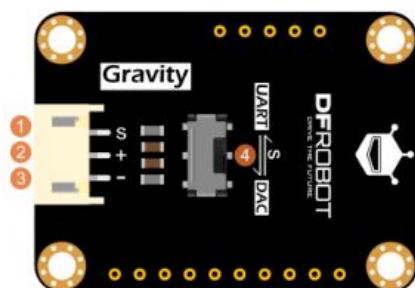
2) Capteur de Gaz HCHO Gravity SEN0231

Le module Gravity permet de détecter les Composés Organiques Volatiles comme le formaldéhyde. Ce module se raccorde sur une entrée série (UART). Grâce à un inverseur, il est possible de basculer d'une sortie analogique à une sortie numérique.



Ce capteur a donc deux modes de fonctionnements :

Voici les broches à utiliser :



Num	Label	Description
1	S	Signal (DAC/UART)

2	+	VCC
3	-	GND
4	SW	Signal (UART or DAC) Output Switch

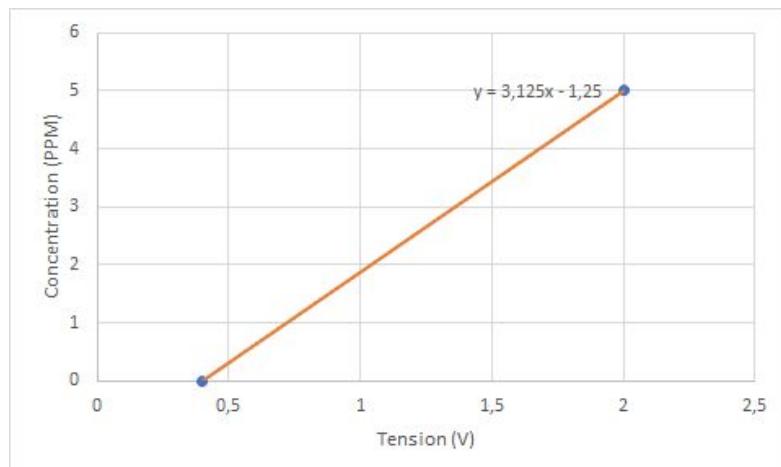
- Fonctionnement en signal analogique :

Pour utiliser un port analogique, il faut placer le commutateur en mode DAC.

Dans le cas d'une carte Arduino UNO, les entrées analogiques peuvent mesurer des tensions comprises entre 0 et 5 volts, avec une précision de 10 bits (soit 1024 points). La fonction analogRead() retourne un nombre entier compris entre 0 et 1023. Ce nombre correspondant à la tension mesurée, 0 = 0 volt et 1023 = tension alimentation = 5 volts.

La relation entre la tension et la concentration est linéaire : 0,4 V correspond à 0 ppm et 2,0 V correspond à 5 ppm.

On obtient donc la droite suivante :



On fait donc :

// On mesure la tension sur la broche A0

```
int valeur = analogRead(A0);
```

// On transforme la mesure (nombre entier) en tension via un produit en croix.

```
float tension = valeur * (5.0 / 1023.0);
```

// On utilise l'équation de droite trouvée précédemment.

```
float ppm = 3.125 * valeur - 1.25;
```

Le code est disponible en Annexe 5.

COM13

0.01ppm
0.01ppm
0.01ppm

Voici un exemple de valeurs reçues sur le moniteur série :

- Fonctionnement en signal numérique :

Avant d'utiliser le port numérique, il est indispensable de placer le commutateur en mode UART. Par défaut, il envoie les données toutes les deux secondes.

Pour ce faire, il utilise la formule et le tableau suivant:

$$\text{Gas concentration value} = \text{High byte of concentration} * 256 + \text{Low byte of concentration}$$

0	1	2	3	4	5	6	7	8
Start Byte	Gas Name	Unit	No decimal byte	Concentration (High Byte)	Concentration (Low Byte)	Full Range (High Byte)	Full Range (Low Byte)	Checksum
0xFF	0X17	0x04	0x00	0x00	0x25	0x13	0x88	0x25

Ces données rendront plus précise la valeur de concentration du formaldéhyde.

Pour ce mode de fonctionnement, la librairie serial software est à télécharger et à inclure dans le code car elle permettra de lire les résultats affichés sur le moniteur série. Le code utilisé est disponible en Annexe 5

Nous avons préféré utiliser le capteur branché sur une sortie analogique afin de laisser un nombre suffisant de broches numériques pour les autres capteurs et éviter l'utilisation assez complexe de la bibliothèque Serial.Software.[32]

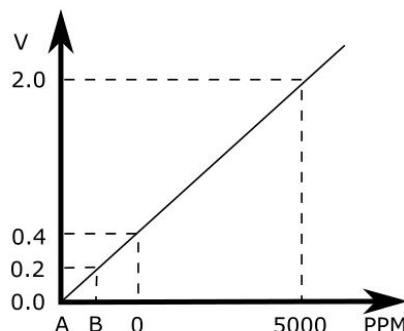
3) Capteur de CO2 infrarouge Gravity SEN0219

Le module est composé d'un capteur de CO2 infrarouge et d'une carte d'interface analogique. Ce module mesure le CO2 avec une grande sensibilité et une excellente linéarité.

Voici la courbe donnée par la datasheet du capteur :

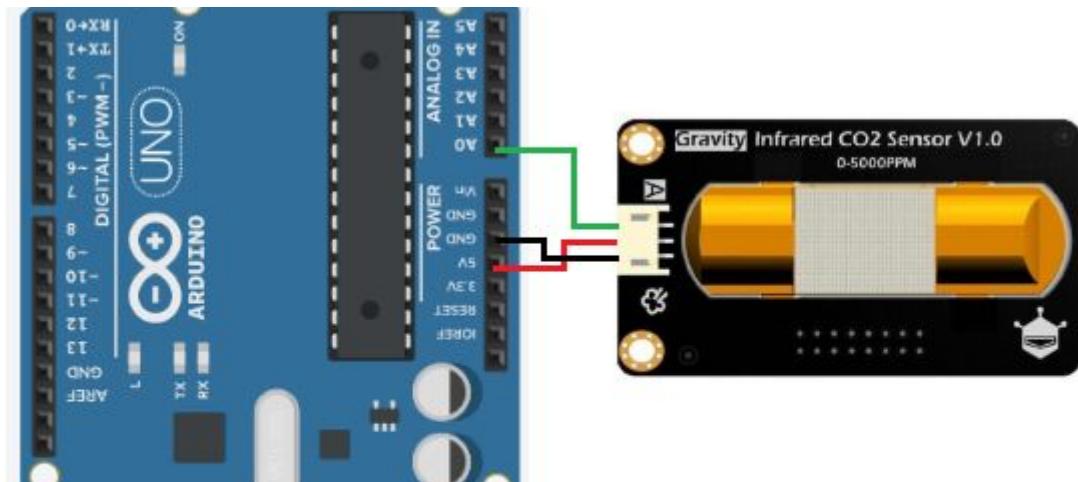
A : Erreur

B : Préchauffage



La méthode utilisée pour le codage est similaire au capteur présenté précédemment. [33]

Voici le montage réalisé :



Le code est disponible en Annexe 6.

4) Capteur de poussières Gravity SEN0177

a) Le Capteur

Ce module permet de mesurer la quantité de particules présentes dans l'air sur 3 calibres (0,3 à 1 µm - 1 à 2,5 µm - 2,5 à 10 µm).

Concernant le principe de fonctionnement, celui-ci repose sur le principe physique suivant : dans un milieu homogène et transparent, la lumière se propage en ligne droite. Lorsque les photons émis par une source lumineuse (généralement une diode laser ou une LED émettant dans le domaine du visible ou du proche infrarouge) rencontrent une particule, une partie du rayon lumineux est diffusée. Un capteur photosensible convertit alors la lumière diffusée en signaux électriques, analysés par un algorithme qui permet de classer les particules par taille, en connaissant l'angle de diffusion, la distance séparant la particule du détecteur, ainsi que l'intensité et la longueur d'onde λ .

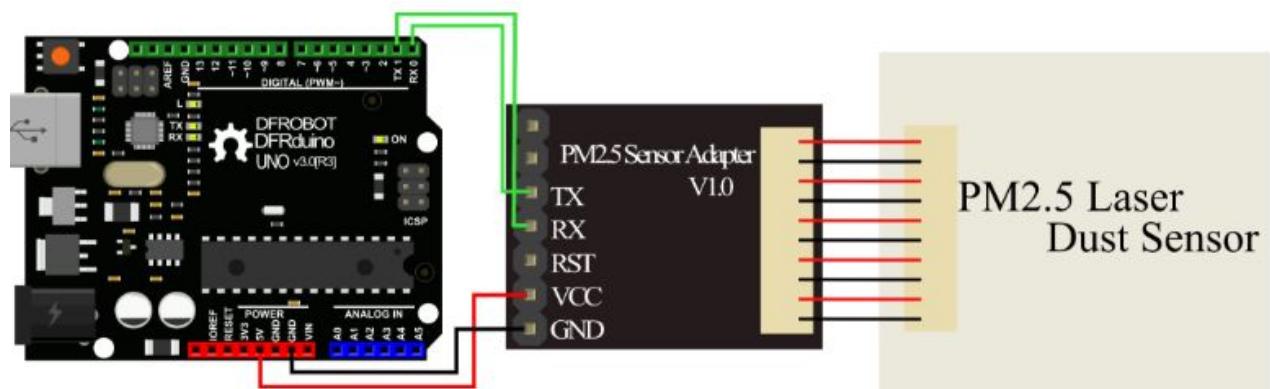
A ce principe physique est généralement associé un principe mécanique : l'air ambiant est aspiré dans le boîtier de manière régulière par un ventilateur ou une pompe. Connaissant le flux d'air et le nombre de particules captées par catégories de taille et en évaluant la densité des particules, il est alors possible d'en déduire la concentration particulaire en $\mu\text{g}/\text{m}^3$ d'air.

Le capteur transmet des paquets fixes de 32 octets dont le contenu est connu. On peut donc sélectionner les parties du paquet qui nous intéressent. [34]

Le capteur est fourni avec un adaptateur que l'on peut voir ci-dessous.



Voici le montage réalisé :



Le code utilisé est disponible en Annexe 7.

b) Le Ventilateur

Une fois le montage ci-dessus réalisé, nous l'avons connecté à Arduino afin de pouvoir lire en sortie une concentration ou un nombre de particule contenu dans la pièce. Malheureusement, en sortie, nous n'avons pu lire aucune donnée. Cela était dû à la taille du ventilateur du capteur. Le débit d'air qui passait à travers celui-ci était insuffisant pour nous permettre de lire une donnée quelconque en sortie.



De ce fait, nous avons rajouté un ventilateur externe un plus grand que celui du capteur qui a été alimenté en 5V afin de pouvoir faire passer un débit d'air plus important.

Nous avons alors refait un test du capteur avec le ventilateur externe et on a pu lire des valeurs en sortie d'arduino donnant le nombre de particules se trouvant dans l'air.

c) Pièce CATIA créée

CATIA (Conception Assistée Tridimensionnelle Interactive Appliquée) est un logiciel de conception assistée par ordinateur: CAO. Son but étant la conception de pièces mécanique. Il nous a été utile dans la réalisation d'une pièce servant pour le capteur de particules fine.

Le ventilateur externe était très grand par rapport à celui du capteur. Ce qui posait un problème lors de l'aspiration de l'air par le capteur car tout le flux d'air traversant le ventilateur externe n'était pas recentré vers celui-ci. Cela devrait avoir un nouvel impact sur les valeurs du nombre de particules une fois le test réalisé.

Afin de remédier à ce problème rencontré, nous avons donc décidé de concevoir via le logiciel CATIA, une pièce respectant une forme souhaitée qui permettrait l'ajustement des deux ventilateurs. Ci-joint la pièce conçue.



5) Capteur de qualité de l'air ADA3709

Le capteur est basé sur un SGP30 permettant la mesure de la qualité de l'air intérieur en mesurant les Composés Organiques Volatiles Totaux (COVT) et le taux de dioxyde de carbone équivalent (eCO₂). Ce module communique avec un microcontrôleur type Arduino ou compatible via une liaison I2C. L'utilisation de ce capteur nécessite la soudure du connecteur inclus en fonction de l'utilisation.[35]

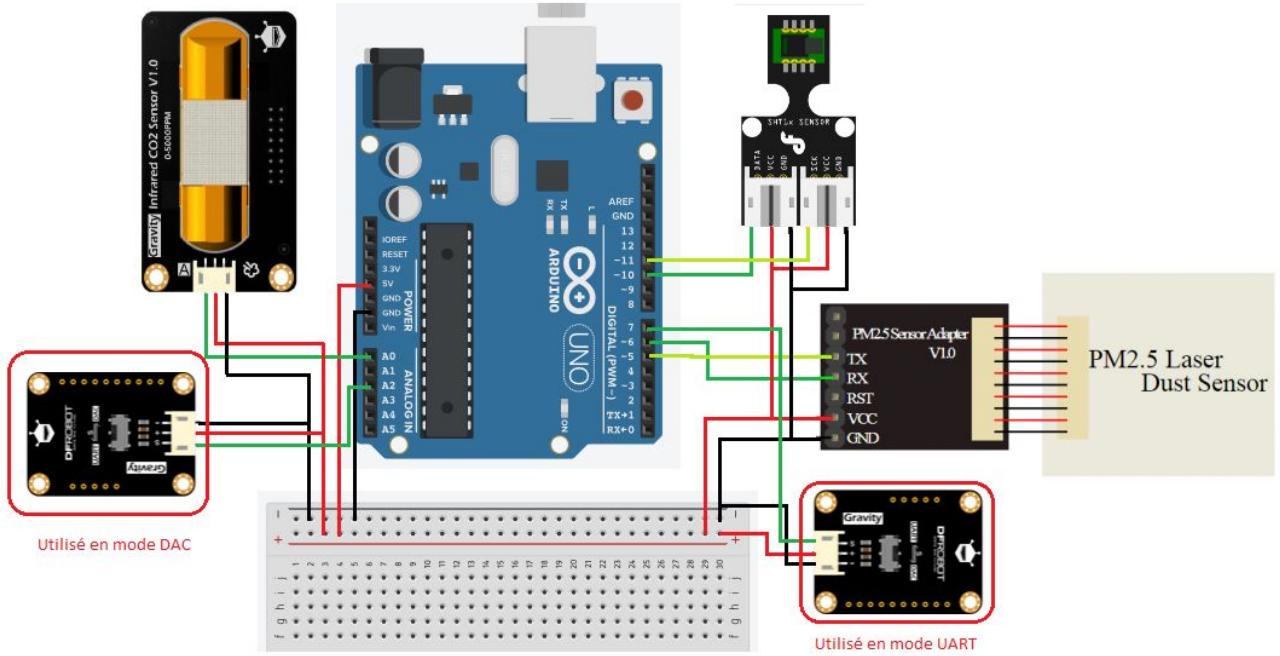
Le code utilisé est disponible en Annexe 8.

B. Assemblage des capteurs sans module Wifi

Après avoir testé tous les capteurs individuellement, nous avons donc commencé à rassembler leurs codes pour les faire fonctionner simultanément et voir s'il y a des difficultés avant la liaison avec le module wifi.

Cependant certains capteurs utilisent les mêmes bibliothèques pour fonctionner ce qui a rajouté une difficulté à la tâche. Ceci sera mieux expliqué dans la partie "Problèmes rencontrés".

Voici le montage sans le module Wifi :



Voici les résultats obtenus sur le moniteur série avec l'utilisation des 4 premiers capteurs :

```

PM1.0: 37 ug/m3
PM2.5: 66 ug/m3
PM1 0: 93 ug/m3

voltage:1469.73mv
3340.62ppm(CO2)
Temperature: 27.5499954223C / 81.6439971923F. Humidity: 52.36%
voltage:1464.84mv
3325.00ppm(CO2)

```

Le code utilisé est disponible en Annexe 9.

C. Problèmes rencontrés

Nous avons perdu du temps lors des essais de fonctionnement du capteur de poussière. En effet son ventilateur interne n'est pas assez puissant et ne traite pas les particules.

Il a fallu faire adapter le capteur de poussière et y ajouter un ventilateur externe ainsi qu'une pièce imprimée en 3D afin de pouvoir récupérer et analyser des valeurs de particules fines, ce qui nous a fait perdre du temps.

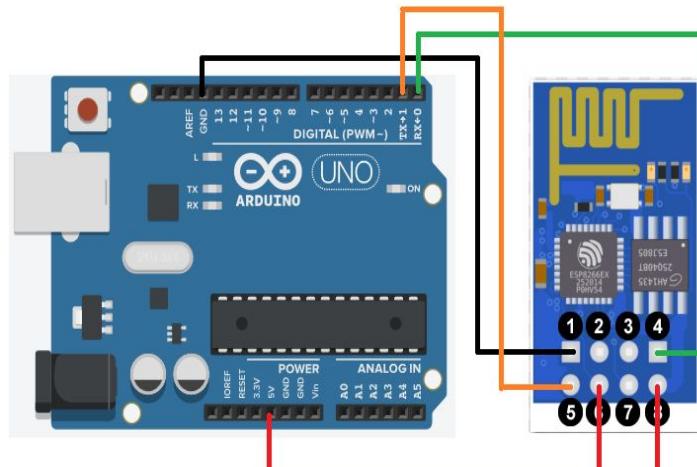
L'association de différents capteurs utilisant la bibliothèque SoftwareSerial a été difficile. Il s'agit d'une librairie bloquante, c'est à dire qu'elle ne permet pas de faire plusieurs tâches en parallèle. Grâce à l'aide des professeurs, nous avons réussi à pallier à ce problème.

IV. Partie 2 : Récupération des données issues des capteurs sur le module Wifi & transfert et visualisation sur ThingSpeak.

A. Test du module WiFi

Le module ESP8266 est un circuit intégré avec un microcontrôleur permettant la connexion en WiFi. [36]

Voici les broches à utiliser pour connecter le microcontrôleur au module wifi lors de l'initialisation :



ESP 8266	Arduino
RX (4)	RX
TX (5)	TX
GRD (1)	GRD
VCC (8)	5V
CH_PD (6)	5V
GPIO 0 (3)	None
GPIO 2 (2)	None

Il existe 3 modes d'utilisation pour le module wifi :

MODE 1 : client

MODE 2 : point d'accès(wifi)

MODE 3 : client + point d'accès

Pendant notre projet, nous avons seulement utilisé le MODE 1. [37]

Nous avons d'abord voulu tester si le module WIFI pouvait se connecter à Internet à l'aide d'un partage des données via l'utilisation d'un portable.

Voici le résultat obtenu sur le moniteur série :

```
/dev/c0
0 => AT+CWMODE=1 OYI
2. at command => AT+CWJAP="Fyh", "Fyh00000" 0. at command => AT OYI
1. at command => AT+CWMODE=1 OYI
2. at command => AT+CWJAP="Fyh", "Fyh00000" OYI
3. at command => AT+CIPMUX=1 OYI
4. at command => AT+CIPSTART=0, "TCP", "perso.esiee.fr/~fengy/toto.txt", 80 Fail
0. at command => AT+CIPSEND=0,36 Fail
1. at command => AT+CIPCLOSE=0 Fail
0. at command => AT+CIFSR OYI
1. at command => AT+CIPMUX=1 OYI
2. at command => AT+CIPSTART=0, "TCP", "perso.esiee.fr/~fengy/toto.txt", 80
```

B. Transfert des données

Pour la partie Récupération des données sur un serveur, nous avons préféré utiliser le site web ThingSpeak qui propose une interface permettant la visualisation de données transmises par la carte arduino via un module Wifi.

Nous avons utilisé cette méthode car aucun des membres de notre groupe n'a de compétences en informatique suffisamment développées ou suivi une unité traitant de cela.

ThingSpeak autorise également la récupération des données au format CSV ainsi que leur traitement via un code Matlab que l'on peut créer. [38]

En effet, ThingSpeak est une application open-source dédiée à l'Internet des Objets (IoT en anglais). Il propose une API (interface de programmation d'application) complètement dédiée au transfert, à la collecte et à l'analyse de données. Développée par MathWorks, entreprise américaine d'édition de logiciels, spécialisée dans les logiciels de calculs mathématiques, cette solution tout-en-un nous a paru approprié pour combler le manque de compétences en informatique de notre groupe.

Couplée à Matlab, cette application open-source pourrait, dans la seconde période du projet, nous permettre une analyse des données plus approfondie.

Nous avons chacun un compte ThingSpeak, car avec notre adresse mail ESIEE, nous disposons d'une licence étudiante pour l'utilisation des logiciels MathWorks dont Thingspeak.

Une fois connecté à ThingSpeak, nous avons créé un channel (chaîne de transmission) pour le premier capteur, Gravity DFR0066, mesurant la température et l'humidité.

Channel Settings

Percentage complete	30%	
Channel ID	958620	
Name	capteur_temperature&humidité	
Description		
Field 1	température	<input checked="" type="checkbox"/>
Field 2	humidité	<input checked="" type="checkbox"/>

Pour ce channel, un API KEY est généré par ThingSpeak. Cet API KEY est inséré dans le code Arduino téléversé sur la carte Arduino. Voici ci-dessous le site de ThingSpeak :

The screenshot shows the ThingSpeak interface with the 'API Keys' tab selected. It displays two sections: 'Write API Key' and 'Read API Keys'.

- Write API Key:** Shows a key field containing 'RGDE9FBTEFBC3QD' and a button to 'Generate New Write API Key'.
- Read API Keys:** Shows a key field containing 'ED26DX60TKPW4KRM' and a note field. Buttons for 'Save Note' and 'Delete API Key' are also present.

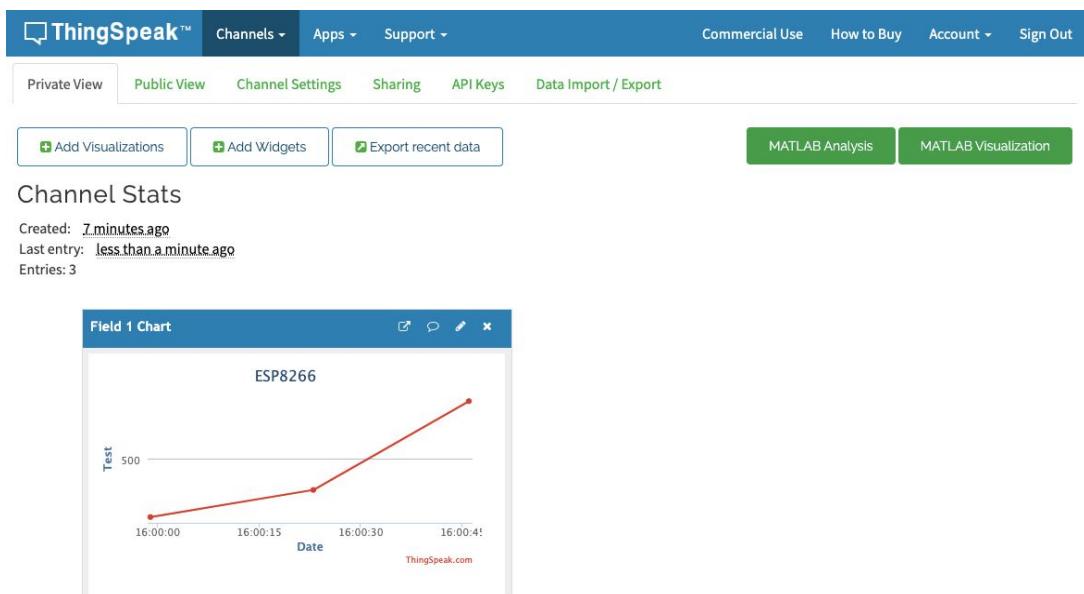
Cette KEY ("clé") va permettre d'envoyer les données spécifiquement à notre channel.

Par défaut, deux graphiques ont été créés affichant deux courbes avec pour abscisse le temps, et pour ordonnée la mesure. En intégrant du code Matlab, nous pouvons améliorer l'affichage des courbes en superposant les courbes. Ce processus sera au centre de l'analyse de données prévue dans la seconde partie du projet.

Le transfert des données doit également être actualisé automatiquement grâce à commande "update" dans la boucle loop() du code Arduino.

I) Transfert des données au hasard

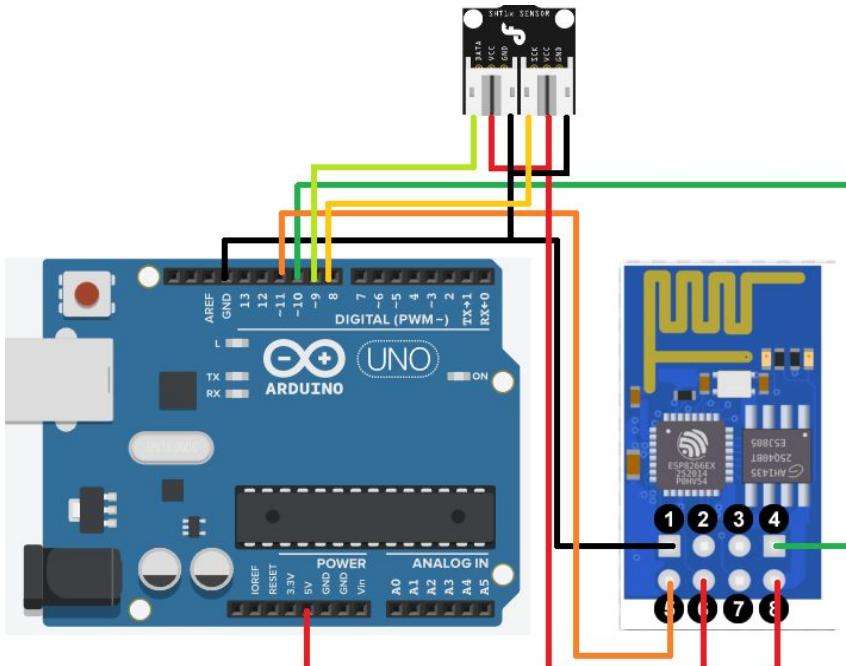
Avant le test sur un capteur, nous avons testé le transfert d'une valeur au hasard :



Nous avons réussi à transférer les données à l'aide du site ThingSpeak : nous pouvons ainsi tester les données instantanées du capteur.

2) Transfert des données des capteurs

Nous avons branché le capteur de température et d'humidité avec le module WIFI ESP8266. Le montage réalisé est le suivant :



ESP 8266	Arduino
RX (4)	11
TX (5)	10
GND (1)	GND
VCC (8)	5V
CH_PD (6)	5V
GPIO 0 (3)	None
GPIO 2 (2)	None
DFR 0066	Arduino
Data_température	8
Data_humidité	9

Sur le moniteur série :

```

Arduino

5. at command => AT+CIPMUX=1 OYI
6. at command => AT+CIPSTART=0,"TCP","api.thingspeak.com",80 OYI
7. at command => AT+CIPSEND=0,66 OYI
9. at command => AT+CIPCLOSE=0 OYI
Temperature: 26.5800018310 Humidity: 39.79%
10. at command => AT+CIPMUX=1 OYI
11. at command => AT+CIPSTART=0,"TCP","api.thingspeak.com",80 OYI
12. at command => AT+CIPSEND=0,66 Fail
1. at command => AT+CIPCLOSE=0 Fail
Temperature: 26.5599975585 Humidity: 39.93%
0. at command => AT+CIPMUX=1 OYI
1. at command => AT+CIPSTART=0,"TCP","api.thingspeak.com",80 OYI
2. at command => AT+CIPSEND=0,66 Fail
1. at command => AT+CIPCLOSE=0 Fail
Temperature: 26.5599975585 Humidity: 39.76%
0. at command => AT+CIPMUX=1 OYI
1. at command => AT+CIPSTART=0,"TCP","api.thingspeak.com",80 OYI
2. at command => AT+CIPSEND=0,66 OYI
4. at command => AT+CIPCLOSE=0 OYI
Temperature: 26.5899963378 Humidity: 39.86%
5. at command => AT+CIPMUX=1 OYI
6. at command => AT+CIPSTART=0,"TCP","api.thingspeak.com",80 OYI
7. at command => AT+CIPSEND=0,66 Fail
1. at command => AT+CIPCLOSE=0 Fail
Temperature: 26.569996948 Humidity: 40.13%
0. at command => AT+CIPMUX=1 OYI
1. at command => AT+CIPSTART=0,"TCP","api.thingspeak.com",80 OYI
2. at command => AT+CIPSEND=0,66 Fail
1. at command => AT+CIPCLOSE=0 Fail
Temperature: 26.5000000000 Humidity: 39.99%

```

Nous pouvons observer que si la commande "AT+CIPSEND" est validée, alors le résultat du capteur à l'instant t est bien téléversé sur ThingSpeak. Cependant, nous avons rencontré plusieurs problèmes lors de l'envoi. En effet, le site ThingSpeak n'arrive pas toujours à recevoir tous les messages envoyés car la durée d'envoi entre 2 données est très courte. Nous avons donc dû paramétrier un délai de 15 secondes entre chaque envoi pour que les messages reçus par ThingSpeak soient plus réguliers.

Sur ThinkSpeak :

Nous avons activé 2 fields pour température et humidité, ces 2 fields doivent aussi préciser dans le code Arduino.[40]

Channel Settings

Percentage complete 30%

Channel ID 958620

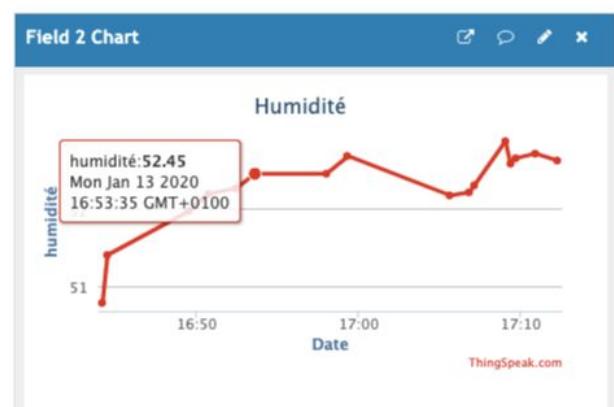
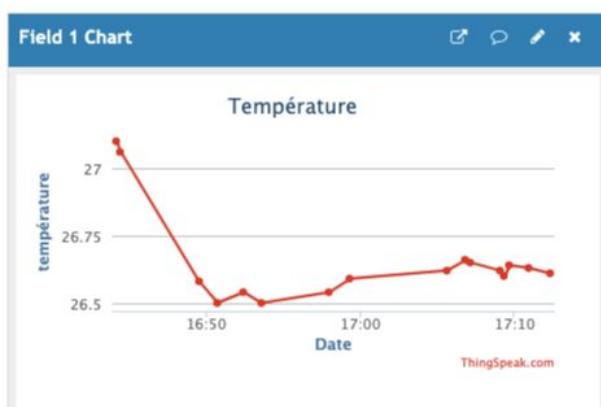
Name capteur_temperature&humidité

Description

Field 1 température

Field 2 humidité

Nous pouvons voir ci dessous, la visualisation des données récupérées pour le capteur de température et d'humidité.



Après avoir fait cela, nous avons continué en ajoutant un par un les différents capteurs. À chaque ajout, nous avons effectué des tests pour voir si l'ensemble fonctionnait bien ensemble.

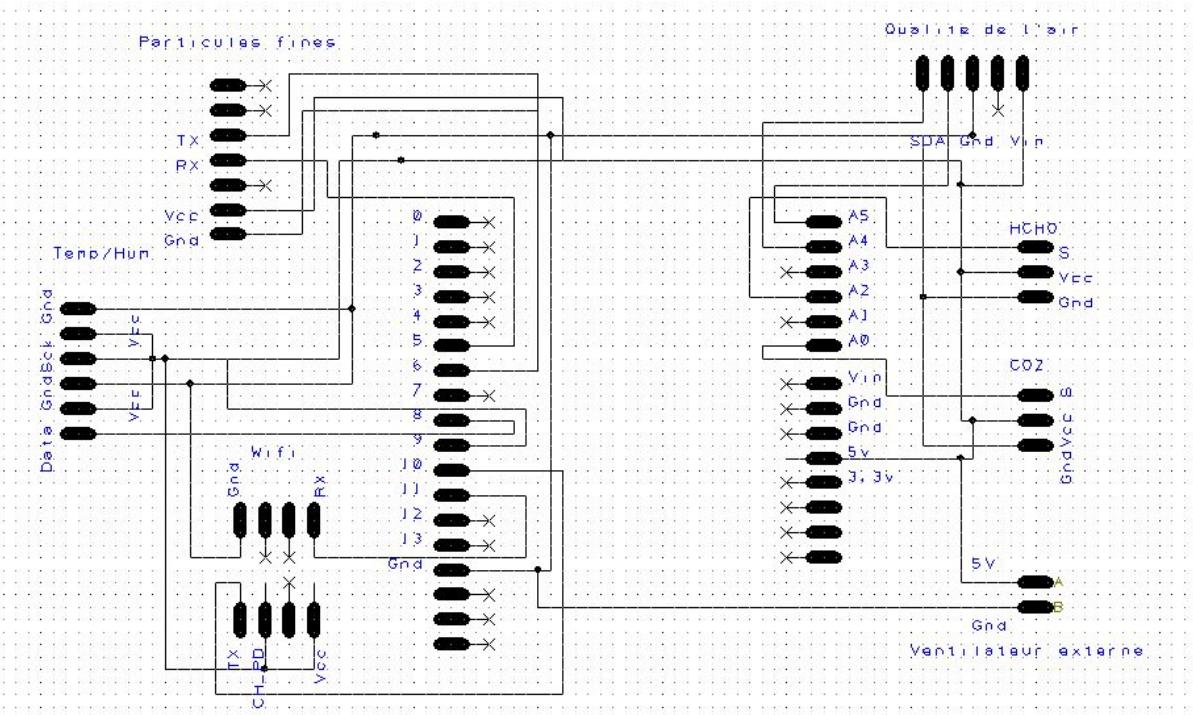
La plupart du temps, un grand nombre de valeurs n'étaient pas transmises par le module Wifi (échec de l'envoi), nous avons donc décidé d'essayer la connection à un réseau domestique. En effet, nous avons pu constater qu'un nombre plus important de valeurs ont été transmises.

3) Montage et broches utilisées pour la récupération des données

Voici un tableau récapitulatif du branchement à effectuer lorsqu'on utilise tous les capteurs avec le module WiFi :

Capteurs :	Broches :
Capteur de température & d'humidité	dataPin: 8, clockPin: 9
Capteur de HCHO	A2
Capteur de CO2	A0
Capteur de poussières	RX: 5, TX: 6
module WiFi	RX: 10, TX: 11
Capteur de qualité de l'air	SDA: A4, SCL: A5

Voici un schéma du montage final :



4) IFTTT

Il existe également une application, IFTTT , qui pourrait nous permettre à plus long terme d'envoyer des notifications à l'utilisateur.

Si l'analyse des données montre une mesure en dehors de l'intervalle de sécurité, une alerte doit être envoyée au propriétaire de la balise. ThingSpeak dispose de plusieurs outils, dont un possible ajout du système IFTTT.

IFTTT est un “service web gratuit permettant à ses utilisateurs de créer des chaînes d'instruction simples appelées *applets*”.

Ce service web gratuit permet d'accomplir une série d'instructions si une condition préétablie est remplie.

Pour cela, nous avons créé un évènement de réaction test dans ThingSpeak. Si la valeur du capteur de température était supérieur à 15 °C, l'évènement de réaction test_capteur était produit. Ceci est un évènement interne à ThingSpeak.

Apps / React / test_capteur

Edit React

Name: test_capteur

Condition Type: Numeric

Test Frequency: On data insertion

Last Ran: 2020-01-13 16:12

Channel: capteur_temperature&humidité

Condition: Field 1 (température) is greater than 15

ThingHTTP: test_temp

Run: Each time the condition is met

Created: 2020-01-13 3:13 pm

Ensuite nous avons crée un deuxième évènement cette fois ci dans le ThingHTTP. La particularité de cet évènement est qu'il est détectable par un service extérieur à ThingSpeak, dont IFTTT. Il envoie un requête WEB, comme le nom l'indique (ThingHTTP).

Name:	test_temp
API Key:	IJIA3D2BN12GD8B5
Regenerate API Key	
URL:	https://maker.ifttt.com/trigger/test_capteur/with/key/c_cJNEe_LuBJThKu1cbJZl4hr1UxHxWQMEsOPRkG4RS

Si l'évènement test_capteur se produit, alors l'évènement test_temps se produit. La détection de cet évènement est communiquée à Webhook par le biais d'un key, la clé de la requête. Cette clé permet de créer le dernier événement de la chaîne qui est l'envoi d'une notification au smartphone du propriétaire de la balise. Ce dernier évènement est modulable, en effet, cela peut être aussi l'envoi d'un email, d'un message Whatsapp.

Nous avons créé un évènement sur Webhook, ci-dessous la page de création de la réception de la requête web par Webhook.

Your key is:
c_cJNEe_LuBJThKu1cbJZl4hr1UxHxWQMEsOPRkG4RS

◀ Back to service

To trigger an Event

Make a POST or GET web request to:

```
https://maker.ifttt.com/trigger/{event}/with/key/c_cJNEe_LuBJThKu1cbJZl4hr1UxHxWQMEsOPRkG4RS
```

With an optional JSON body of:

```
{ "value1" : "_____", "value2" : "_____", "value3" : "_____"}
```

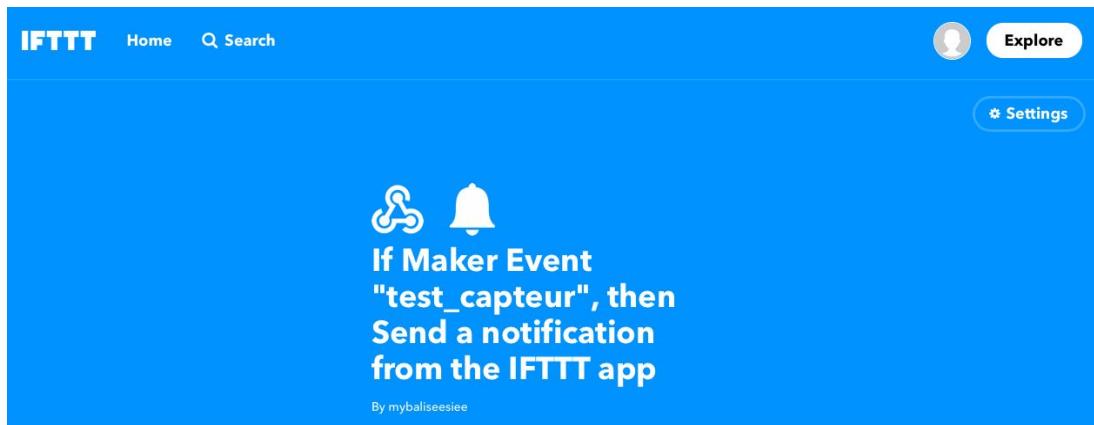
The data is completely optional, and you can also pass value1, value2, and value3 as query parameters or form variables. This content will be passed on to the Action in your Recipe.

You can also try it with curl from a command line.

```
curl -X POST https://maker.ifttt.com/trigger/{event}/with/key/c_cJNEe_LuBJThKu1cbJZl4hr1UxHxWQMEsOPRkG4RS
```

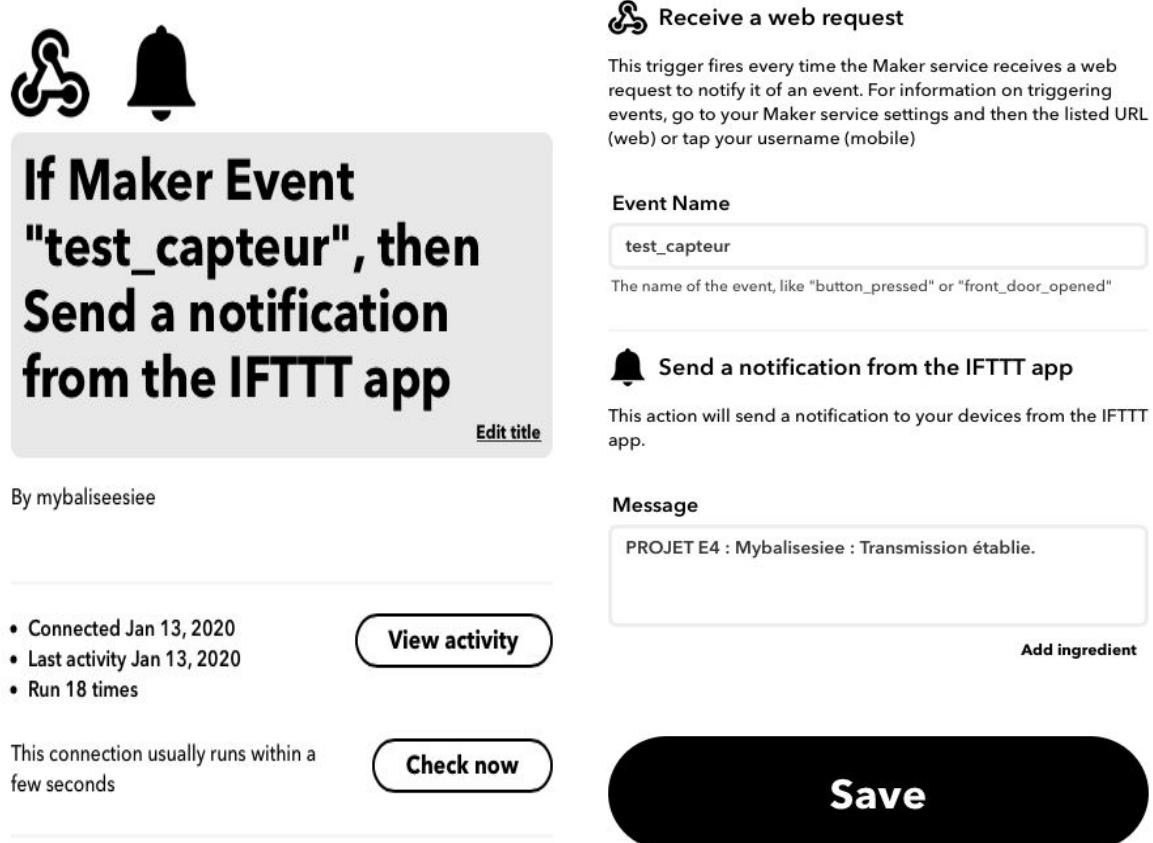
[Test It](#)

Ensute nous avons connecté Webhook avec le système d'alertes : Notifications.



(Vue Ordinateur)

Paramètres



The configuration screen for the "If Maker Event "test_capteur", then Send a notification from the IFTTT app" recipe. It shows the trigger (Receive a web request) and action (Send a notification from the IFTTT app). The message field contains "PROJET E4 : Mybalisesee : Transmission établie." A "Save" button is at the bottom right.

By mybaliseesee

- Connected Jan 13, 2020
- Last activity Jan 13, 2020
- Run 18 times

This connection usually runs within a few seconds

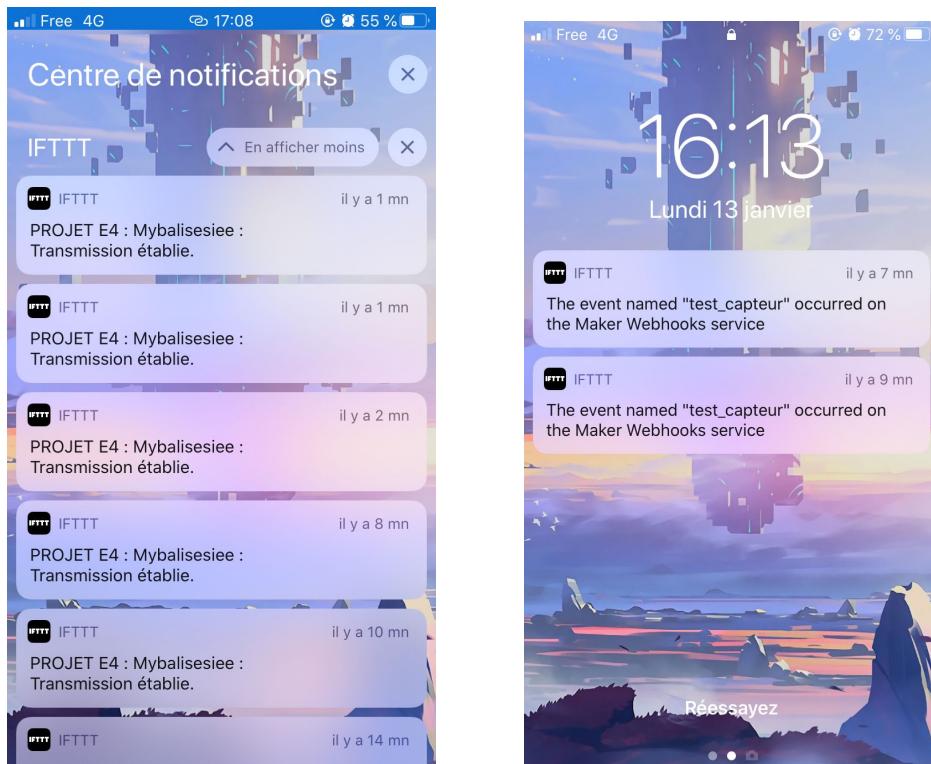
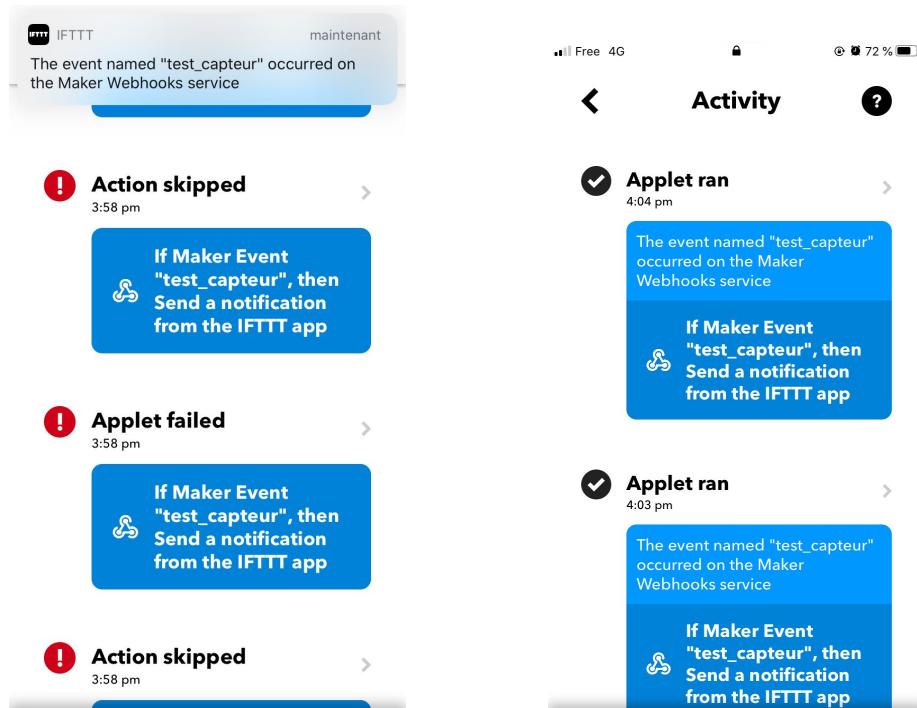
View activity

Check now

Save

(Vue Smartphone)

Sur le smartphone, où l'application IFTTT a été installée, nous pouvons voir que les notifications se sont bien affichées lorsque la condition de l'affichage est vraie.[39]



C. Problèmes Rencontrés

Concernant la puissance, d'après la datasheet et les différentes références que nous avons vu, il ne fallait pas utiliser plus de 3.5V pour alimenter le module. Cependant, lors des tests, le module ne démarrait pas correctement avec 3.5V, ce

qui rendait le débogage très difficile au début. Nous sommes finalement passés à 5V et tout s'est bien passé.

La création du code pour le microcontrôleur a posé quelques problèmes pour certains capteurs notamment ceux utilisant la même bibliothèque "SoftwareSerial". Il a fallu adapter le code en conséquence et faire de nombreuses modifications.

V. Partie 3 : Création de la balise

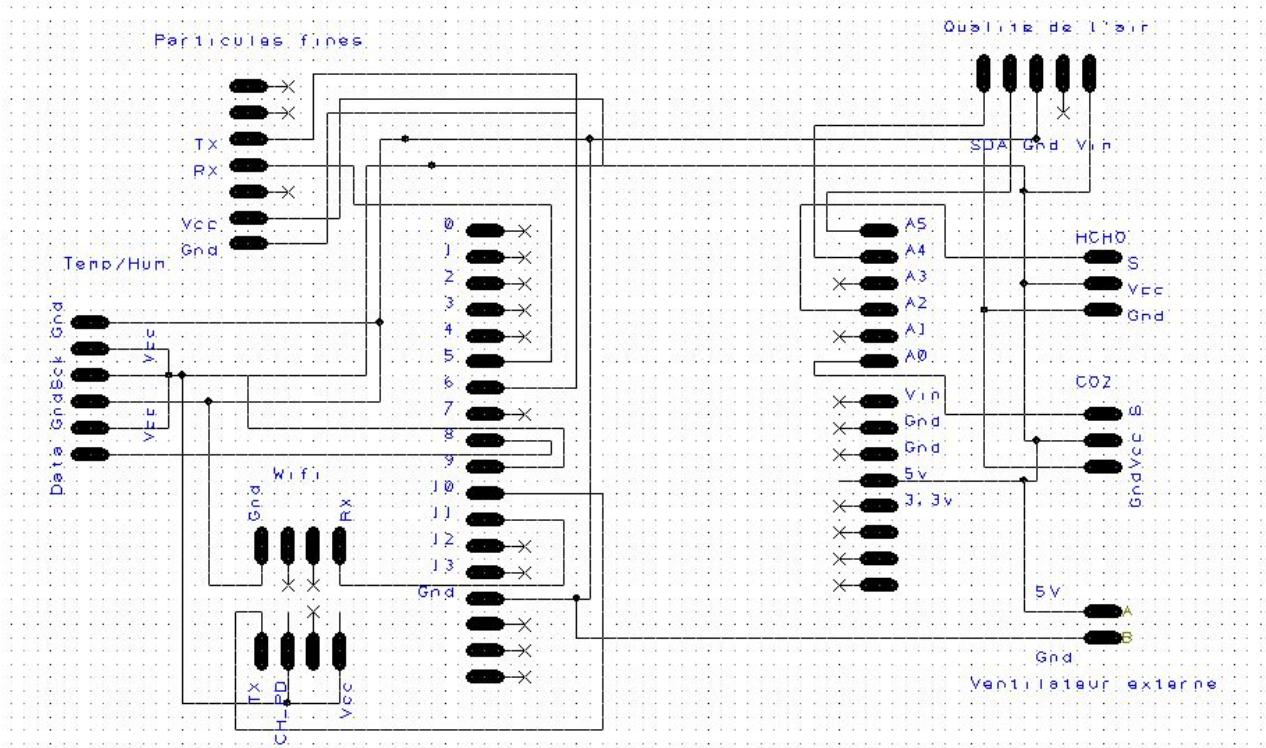
Afin d'avoir une balise fonctionnelle et transportable tout en ayant un résultat esthétique, nous avons décidé de faire un circuit imprimé ainsi qu'un support renfermant tous les capteurs et le montage effectué.

1) Crédation d'un PCB

a) Modélisation du PCB

Nous avons souhaité réaliser un circuit imprimé et pour cela, nous avons utilisé le logiciel DesignSpark.

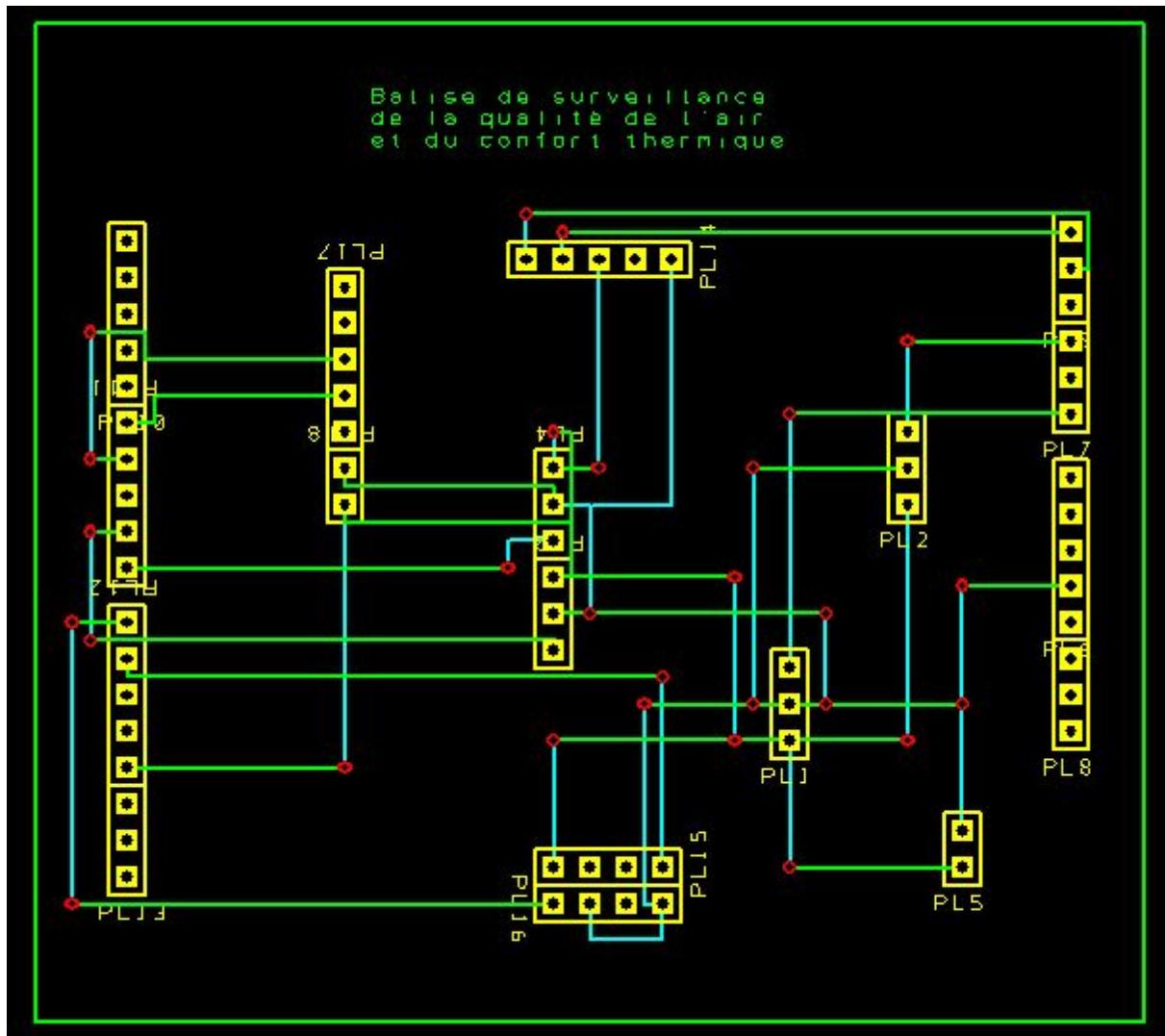
Tout d'abord, il fallait réaliser un schéma du circuit avec les connections entre les différents capteurs :



Par la suite, on peut passer du schéma au PCB. Nous avons choisi de faire un circuit imprimé à deux couches.

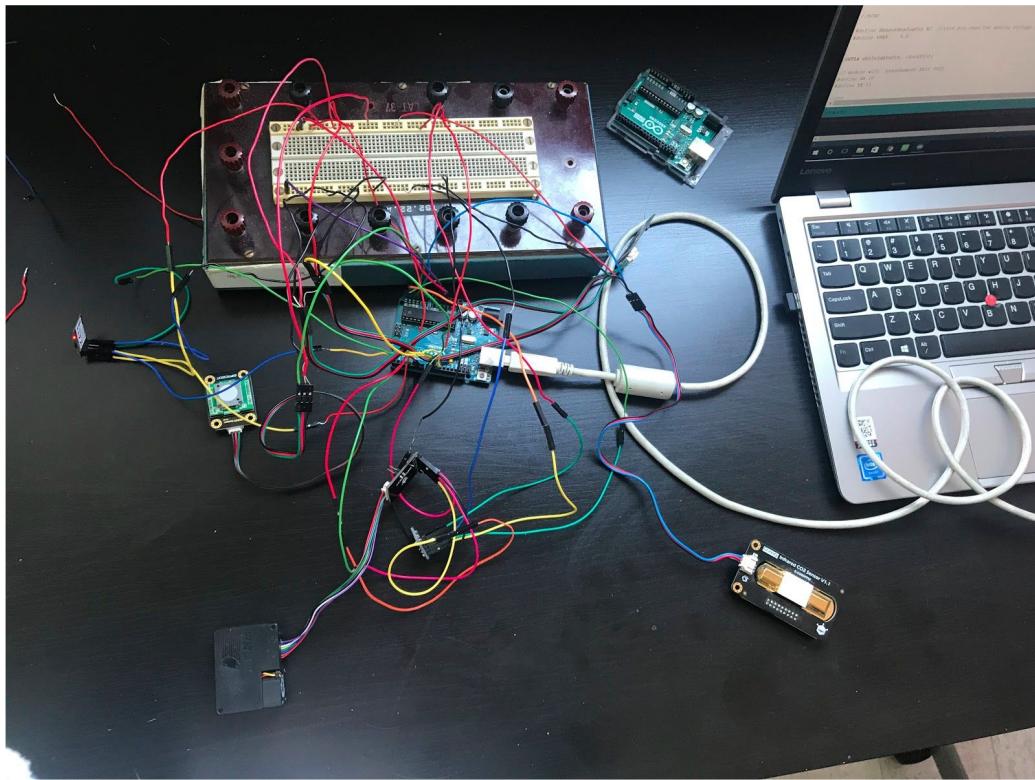
L'utilisation de l'option " Automatically Position" ne satisfaisant pas les distances que l'on devait respecter, nous avons préféré placer les composants par nous même et également créer les liaisons.

Voici le PCB obtenu :

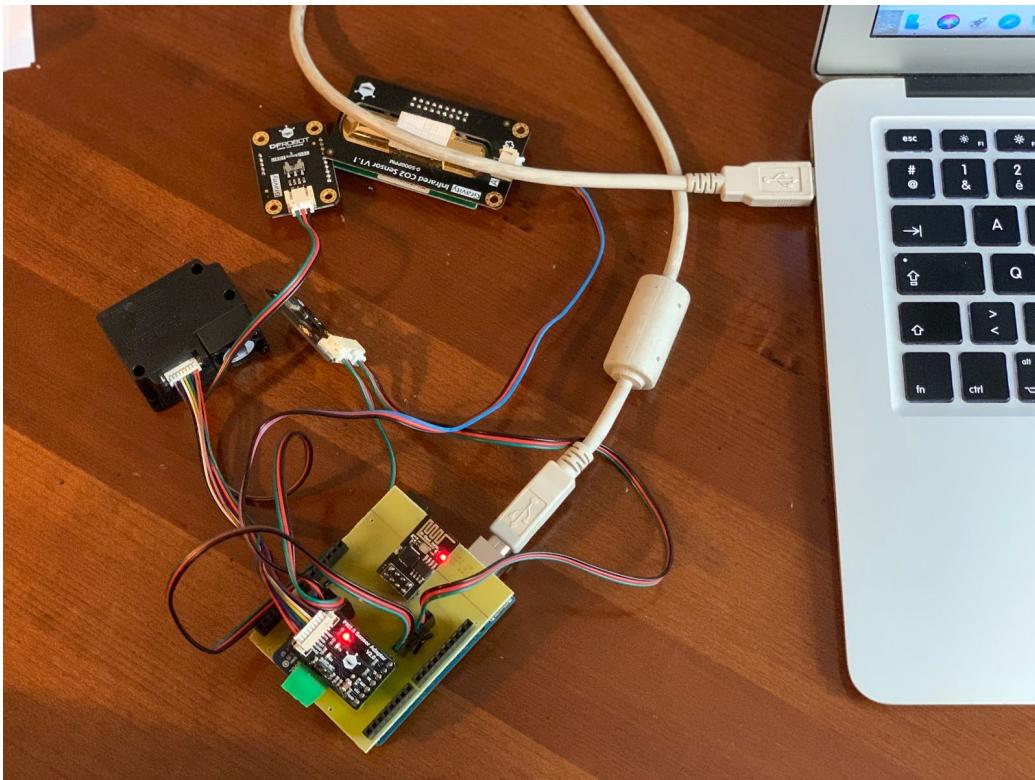


b) Impression du PCB

Malgré le déconfinement, nous n'avons pas pu retourner à l'école pour continuer le projet. Cependant, grâce à l'aide de M. Julien Pagazani qui s'est chargé d'imprimer le PCB et de le souder avec les capteurs, nous avons pu poursuivre la campagne de mesures.



Branchements sans le PCB



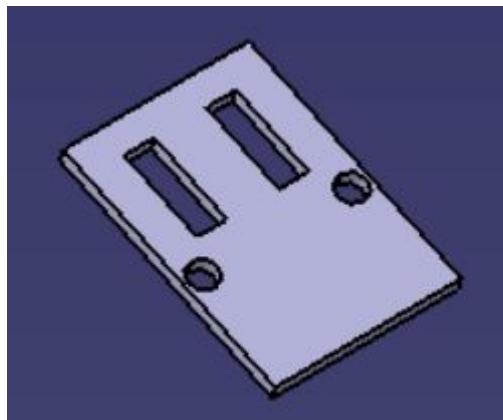
Branchements avec le PCB

Le branchement avec le PCB permet à la fois un rendu plus esthétique et moins contraignant car pas de fil de par et d'autres.

2) Modélisation des capteurs

La modélisation des capteurs nous a permis de visualiser au mieux le rendu de la balise finale. Les dimensions de chaque capteur ont été trouvées sur le site Gotronic ou mesurées avec une règle.

Ci-dessous le capteur de Température et d'Humidité :

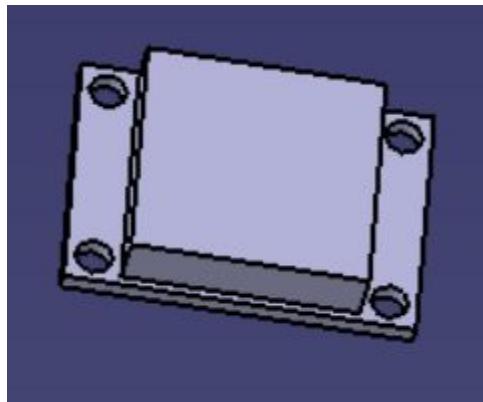


Modélisation sur Catia



Capteur réel

Ci-dessous le capteur de Formaldéhyde :

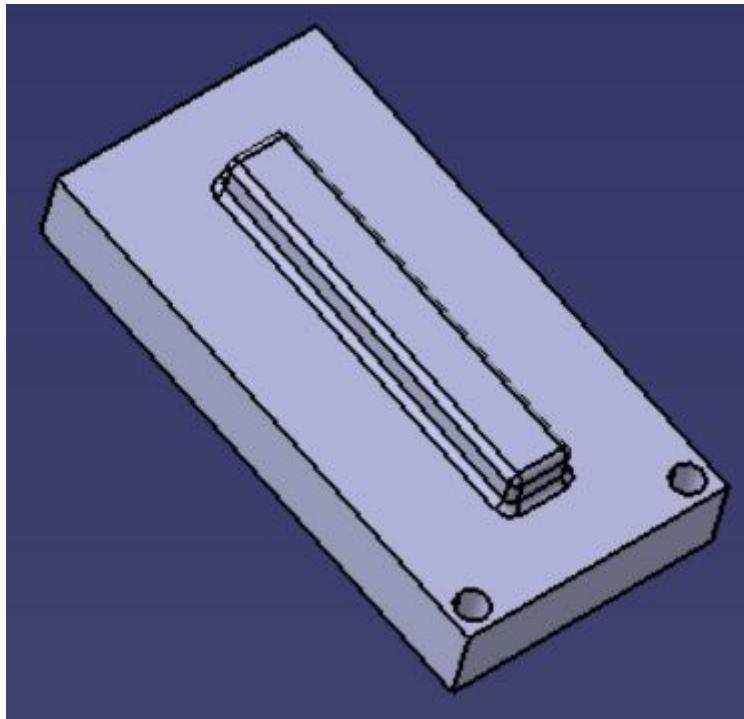


Modélisation sur Catia



Capteur réel

Ci-dessous le capteur de CO₂ :

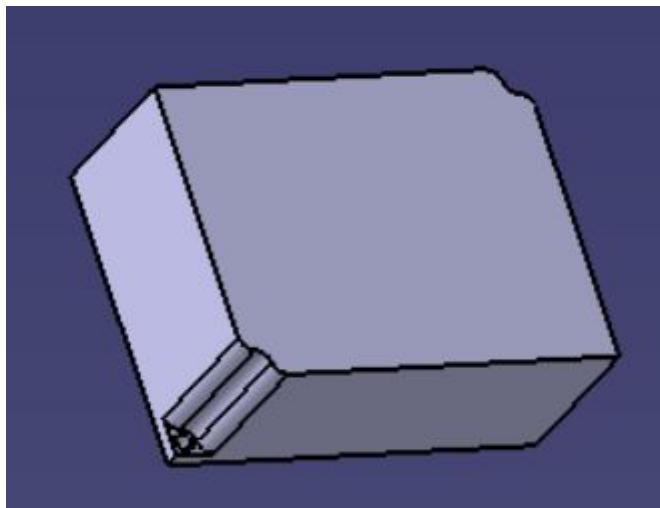


Modélisation sur Catia



Capteur réel

Ci-dessous le capteur de Particules fines :



Modélisation sur Catia



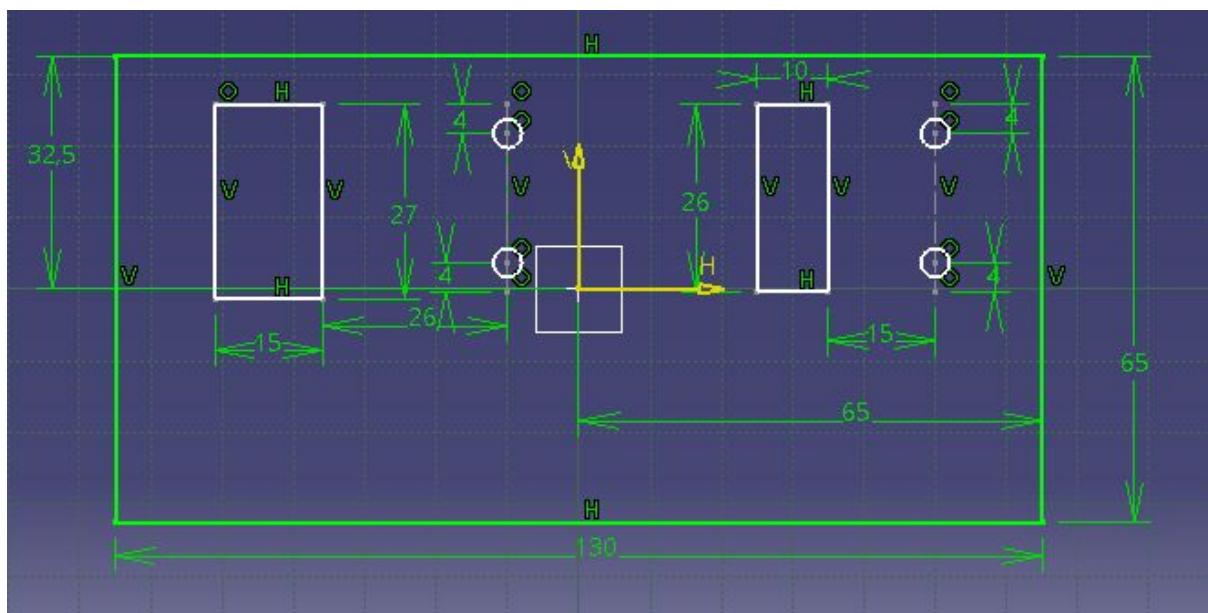
Capteur réel

3) Création d'un support pour la balise

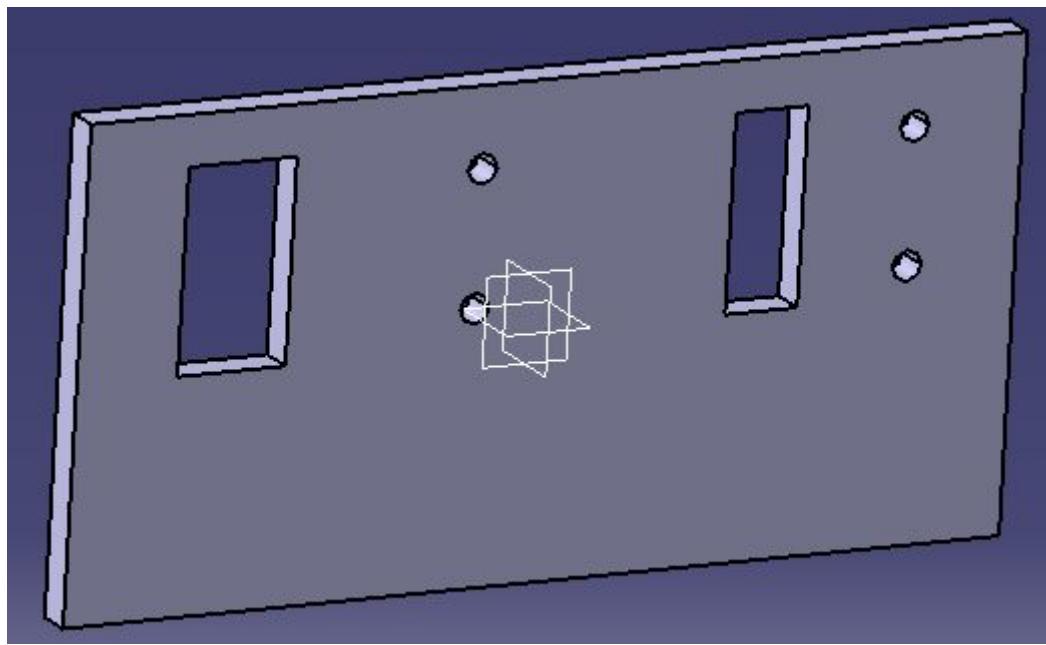
Nous avons décidé de modéliser un support pour les composants sur CATIA V5. Il a fallu également laisser des espaces suffisants pour que les capteurs soient en contact avec l'air ambiant. De plus, certains capteurs chauffent (module wifi), il a donc fallu choisir avec soin leur emplacement. Il devait par exemple être loin du capteur de température afin de ne pas perturber ses valeurs.

Pour ne pas se perdre, il est plus simple de faire chaque côté indépendamment sous forme de "Part" puis d'assembler les parties en un produit final "Product". Si on avait fait la structure en une seule pièce, la moindre erreur aurait impacté la structure entière, il aurait fallu tout modifier alors que dans notre cas on peut se permettre de ne modifier que la surface qui pose problème.

Voici l'esquisse de l'un des côtés de la balise. On peut voir à droite un emplacement pour le capteur de température et d'humidité avec les emplacements des trous pour fixer le capteur à l'aide de vis. Le rectangle quant à lui est une ouverture pour faire passer les fils à l'intérieur de la balise. De même à gauche, il s'agit d'un emplacement pour y mettre le capteur de formaldéhyde.



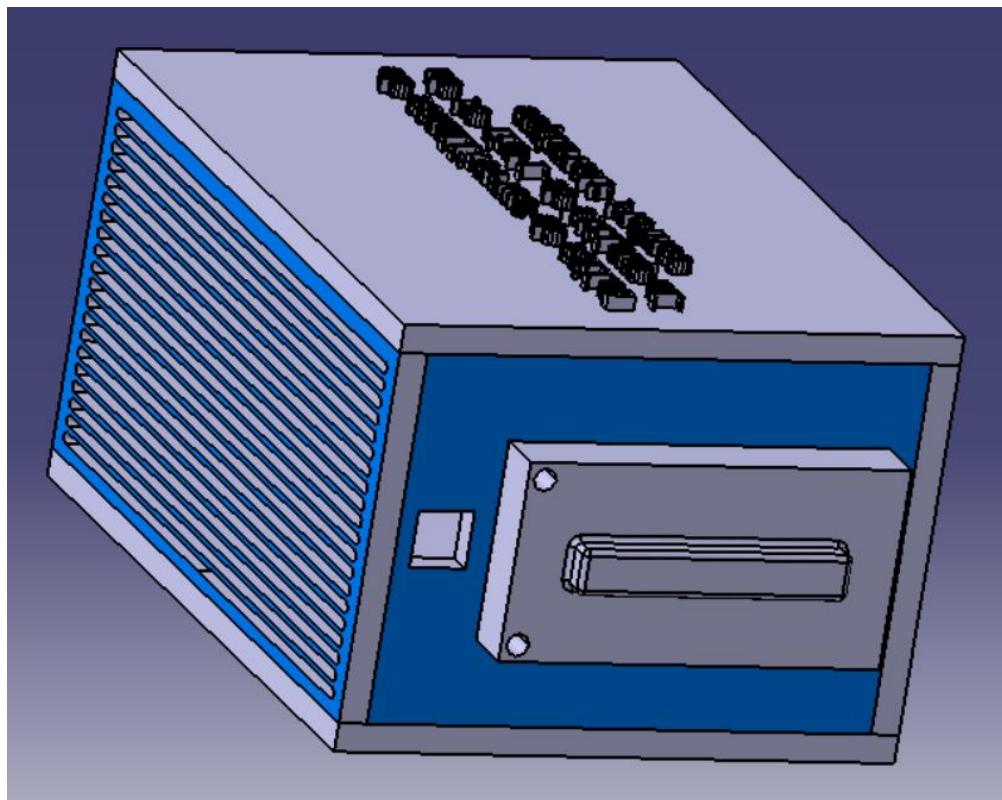
On ajoute ensuite une épaisseur à la surface et on obtient :



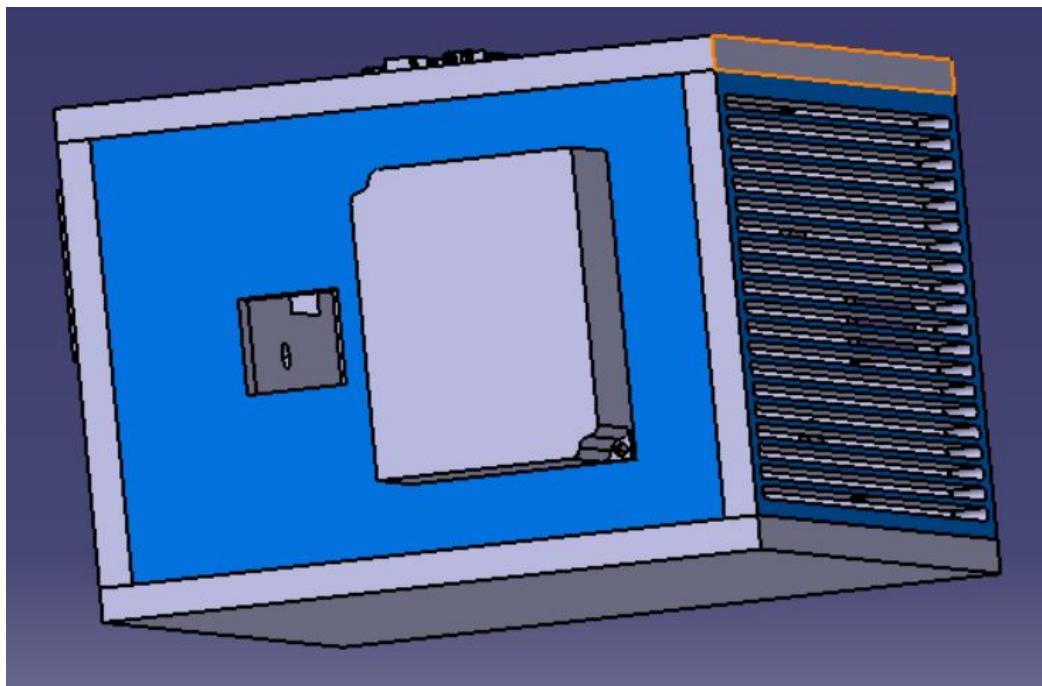
Le résultat final est le suivant :



Vue de face avec le capteur de Formaldéhyde à gauche et le capteur de Température & Humidité à droite



Vue de gauche avec le capteur de CO₂



Vue de droite avec le capteur de Particules fines

Nous avons décidé de réaliser une grille d'aération pour permettre une meilleure circulation de l'air.

4) Problème rencontré

Par manque de temps et au vu de la fermeture de l'école, nous n'avons pas pu imprimer le support de la balise.

VI. Partie 4 : Campagne de mesures & Analyse des données

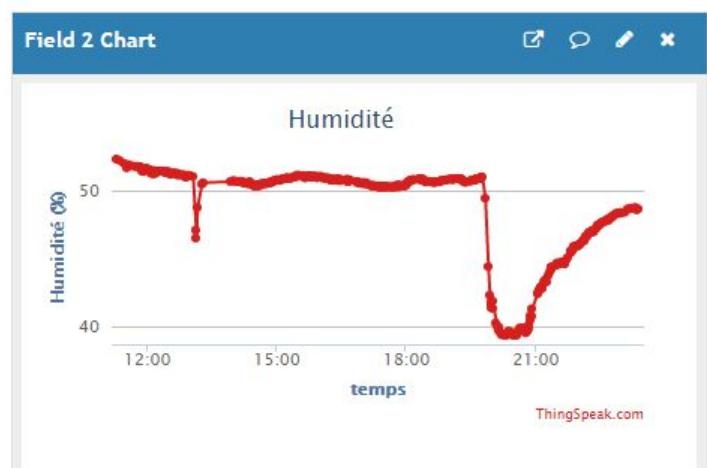
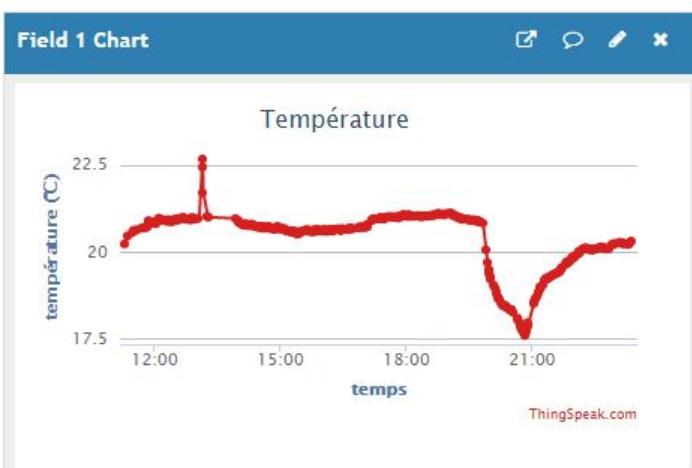
1) Campagne de mesures

Concernant la récupération des données, ThingSpeak permet la récupération des données au format CSV. Nous avons pu récupérer des données sur plusieurs jours, pour des durées variables et dans différentes villes.

Voici le type de résultat récupérés :

created_at	entry_id	field1	field2	field3	field4	field5	field6	latitude	longitude	elevation	status
2020-03-30 11:17:46 CEST	1	20.23	52.38	5.00	350.00	0.00	0	,,,	,,,	,,,	,,,
2020-03-30 11:22:08 CEST	2	20.46	52.29	4.69	2837.50	32.00	0	,,,	,,,	,,,	,,,
2020-03-30 11:27:55 CEST	3	20.55	52.08	5.00	2746.88	0.00	0	,,,	,,,	,,,	,,,
2020-03-30 11:31:36 CEST	4	20.62	51.74	3.83	487.50	0.00	46	,,,	,,,	,,,	,,,
2020-03-30 11:32:02 CEST	5	20.62	51.78	4.49	3037.50	0.00	128	,,,	,,,	,,,	,,,
2020-03-30 11:35:04 CEST	6	20.63	51.97	4.50	2806.25	0.00	209	,,,	,,,	,,,	,,,
2020-03-30 11:41:26 CEST	7	20.68	51.88	4.44	2943.75	0.00	201	,,,	,,,	,,,	,,,
2020-03-30 11:44:30 CEST	8	20.69	51.82	4.50	3112.50	0.00	188	,,,	,,,	,,,	,,,
2020-03-30 11:47:07 CEST	9	20.70	51.85	4.46	2837.50	0.00	194	,,,	,,,	,,,	,,,
2020-03-30 11:49:08 CEST	10	20.72	51.79	3.45	2884.38	0.00	155	,,,	,,,	,,,	,,,
2020-03-30 11:49:47 CEST	11	20.72	51.76	5.00	2715.63	0.00	0	,,,	,,,	,,,	,,,
2020-03-30 11:50:23 CEST	12	20.76	51.82	4.55	2868.75	0.00	17	,,,	,,,	,,,	,,,
2020-03-30 11:51:35 CEST	13	20.80	51.80	4.20	2731.25	0.00	17	,,,	,,,	,,,	,,,
2020-03-30 11:52:01 CEST	14	20.88	51.75	4.85	2609.38	0.00	0	,,,	,,,	,,,	,,,
2020-03-30 11:52:32 CEST	15	20.87	51.62	4.52	2684.38	0.00	0	,,,	,,,	,,,	,,,

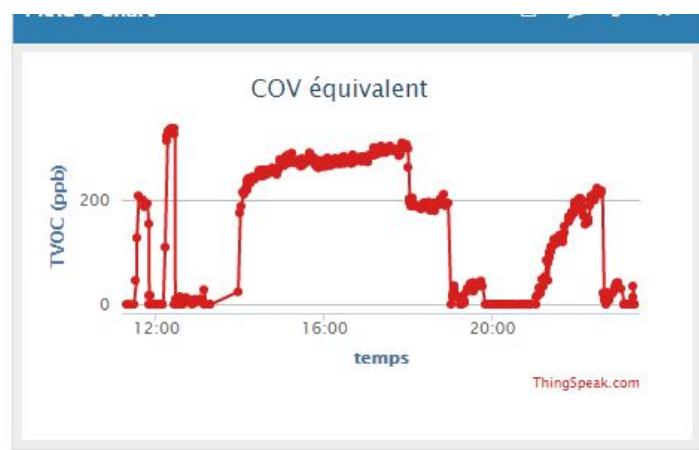
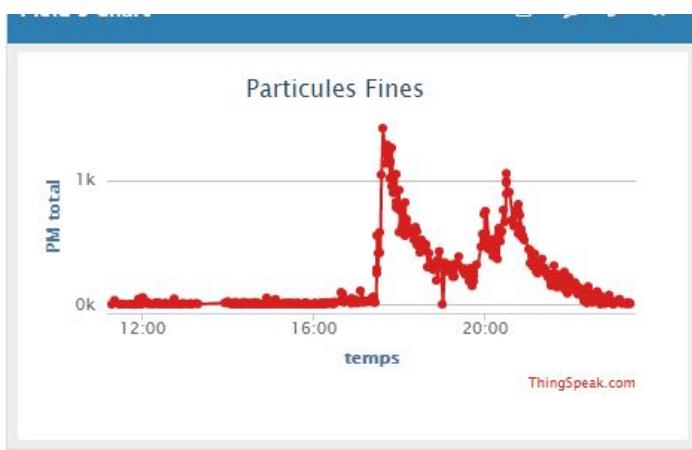
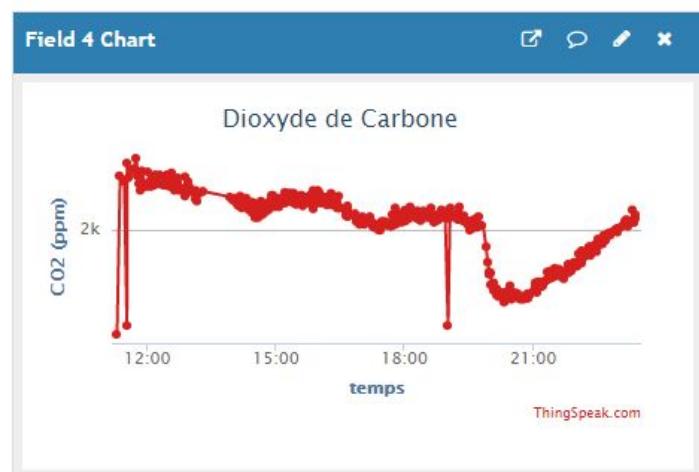
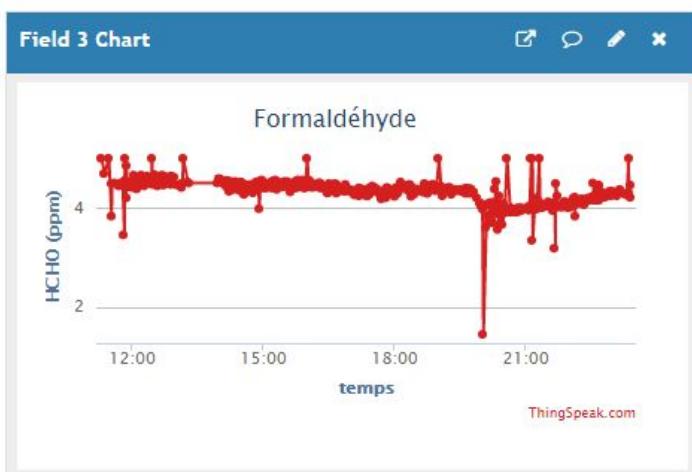
Nous pouvons également visualiser des graphiques sur ThingSpeak. Voici des valeurs recueillies sur 12h :



Comme nous pouvons le voir, le rendu est assez "grossier", il y a énormément de valeurs, ce qui empêche de les analyser.

Pour pouvoir mieux interpréter des valeurs, nous avons donc changé les conditions de la pièce dans laquelle a été placé la balise, par exemple en ouvrant une fenêtre. Cela nous permet d'observer une évolution des indicateurs tout en confirmant que les capteurs fonctionnent bien et qu'ils détectent un changement.

Concernant les valeurs recueillies sur 12h, nous avons ouvert les fenêtres de la pièce entre 19h50 et 20h50.



Après traitement, on obtient le tableau suivant :

created_at	entry_id	field1	field2	field3	field4	field5	field6
2020-03-30 11:17:46 CEST	1	20.23	52.38	5.00	350.00	0.00	0
2020-03-30 11:22:08 CEST	2	20.46	52.29	4.69	2837.50	32.00	0
2020-03-30 11:27:55 CEST	3	20.55	52.08	5.00	2746.88	0.00	0
2020-03-30 11:31:36 CEST	4	20.62	51.74	3.83	487.50	0.00	46
2020-03-30 11:32:02 CEST	5	20.62	51.78	4.49	3037.50	0.00	128

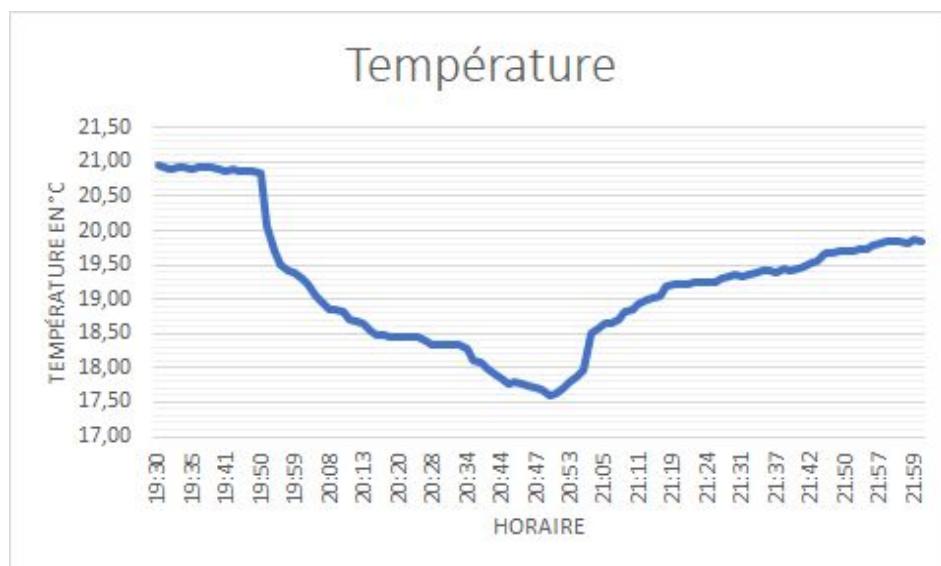
Chaque "field" correspond à un type de données recueillies par les capteurs. Par exemple la colonne field 1 correspond à la Température, comme défini précédemment sur ThingSpeak. Le field 2 correspond à l'Humidité. Le field 3 est le Formaldéhyde. Le field 4 est le CO2. Le field 5 représente les particules fines. Et le field 6 est le COV équivalent.

Afin de voir l'impact du changement dû à l'ouverture d'une fenêtre, nous allons analyser les valeurs recueillies entre 19h30 et 22h00.

Voici les graphiques obtenus concernant les mesures de tous les capteurs :

a) Mesures recueillies à Drancy (93) dans une chambre au premier étage

- ❖ La température

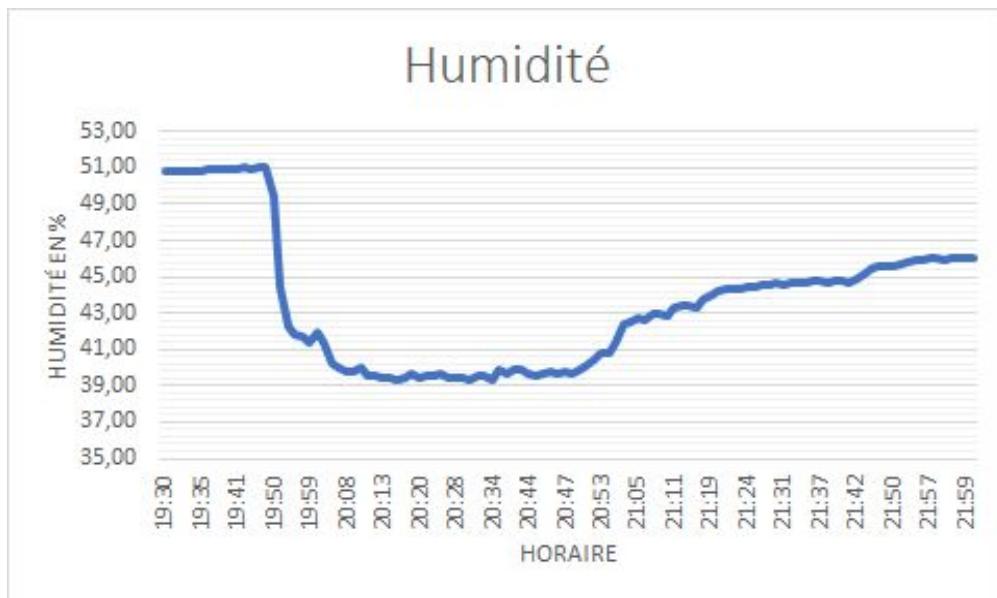


On peut voir en effet une température assez stable avant l'ouverture de la fenêtre. A partir de son ouverture à 19h50, on constate une baisse brutale de la température intérieure et à partir de 20h50 une remontée progressive de la température. Ce phénomène est tout à fait normal car l'air extérieur est beaucoup plus frais. La baisse de température est donc dû à l'entrée de l'air extérieur dans la pièce qui a provoqué une confrontation avec l'air intérieur plus chaud et donc une homogénéisation progressive de la température.

Ici la température intérieure initiale n'est pas alarmante car elle respecte l'intervalle de la température de confort de la RT2012 (réglementation thermique).

En revanche, quand elle est élevée cela signifie que l'air chaud se dilate et devient plus léger. La dispersion des polluants se fait alors plus facilement.

❖ L'humidité



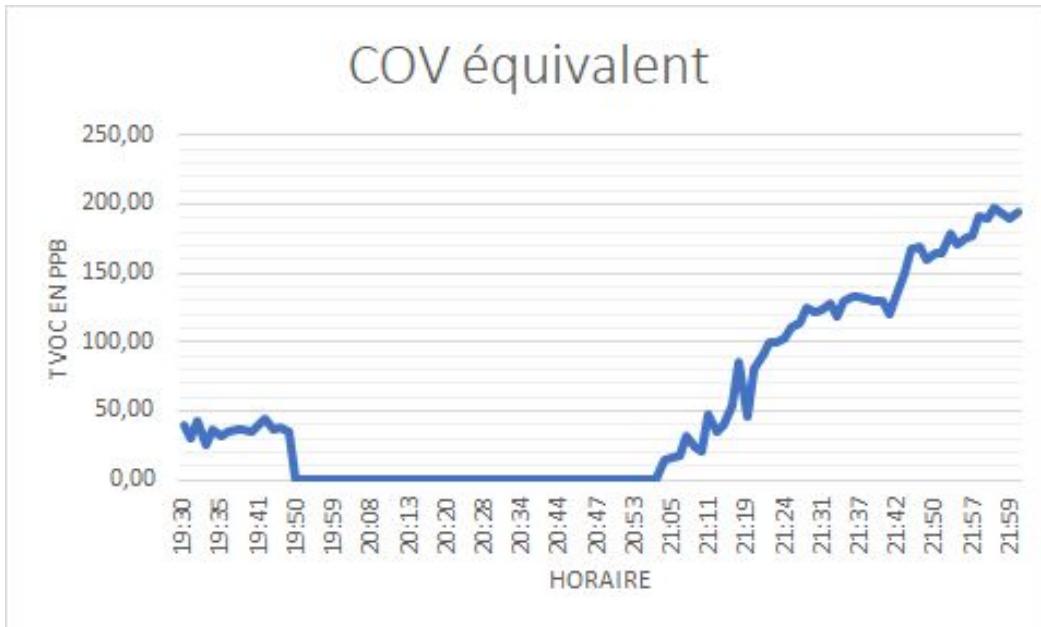
Après ouverture de la fenêtre jusqu'à sa fermeture, on constate une baisse importante de l'humidité qui passe d'approximativement de 51% à une moyenne de 39%. Ceci est justifié par l'air extérieur entrant dans la pièce.

En effet, avant l'ouverture de la fenêtre, l'humidité était assez grande dû à plusieurs phénomènes :

- Une occupation importante car le fait de respirer, de transpirer et d'avoir une activité physique dans la pièce fait dégager une quantité de vapeur d'eau d'environ 0,2 litre par heure et par occupant. Alors si la pièce est occupée par plusieurs personnes à la fois durant toute la journée, une quantité énorme de vapeur d'eau est dégagée dans l'air et est l'une des raisons pour laquelle on observe une augmentation d'humidité.
- La porte de la pièce a été laissée par moment entrouverte, alors la seconde raison de l'augmentation de l'humidité vient de ce fait. Il suffit que la cuisine soit proche de cette pièce ou même la salle de bain pour provoquer une évolution de l'humidité car ces pièces restent des lieux très humides.

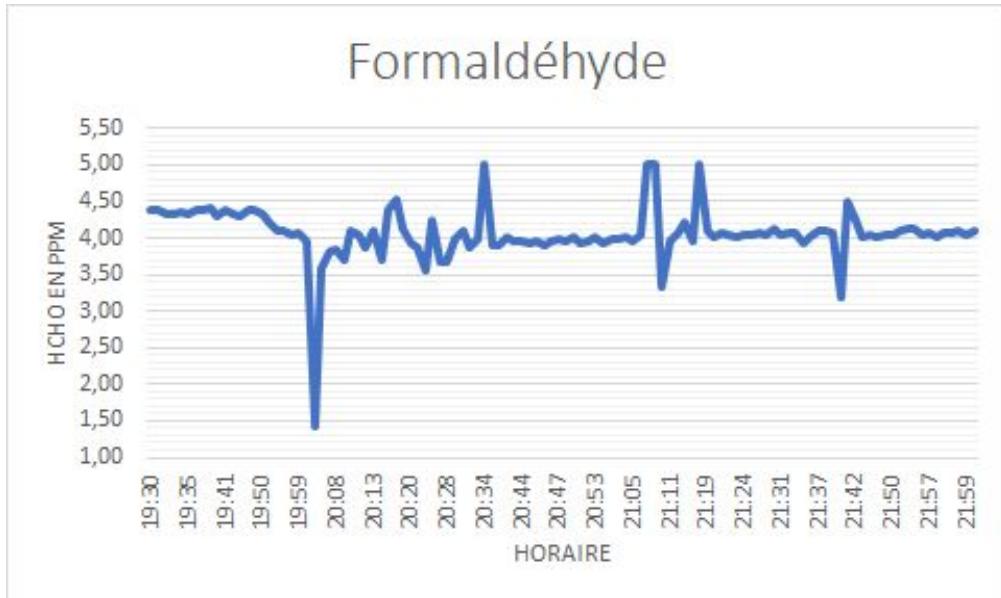
Pour résoudre ce problème d'augmentation de l'humidité, il est nécessaire d'aérer la pièce car l'air contenu dans celle-ci est vicié. Si on revient à notre graphique, l'ouverture de la fenêtre correspondant à l'aération de la pièce justifie la baisse de l'humidité observée.

❖ Le Composé organique volatil



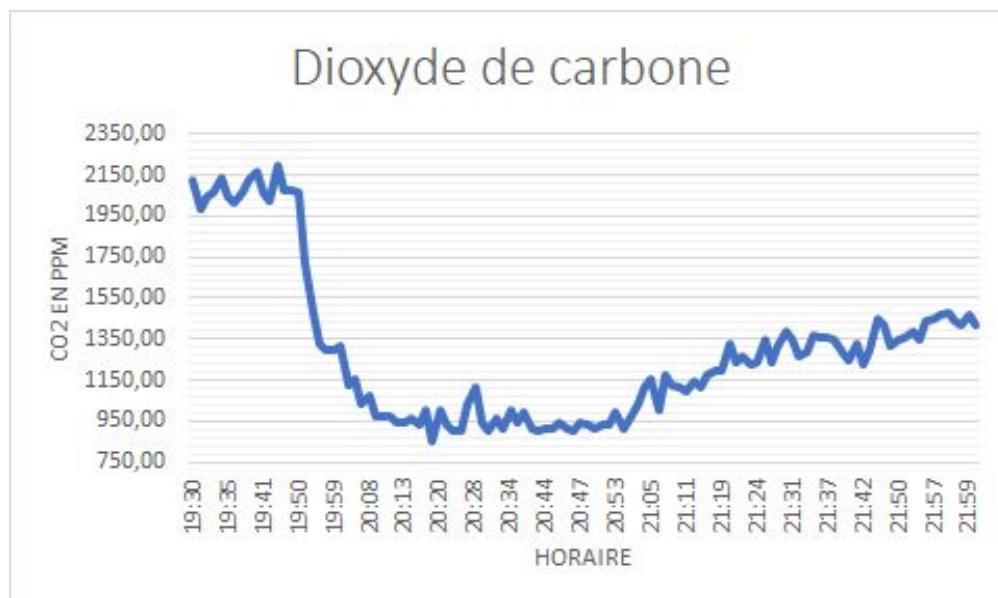
Avant 19h50, on observe une certaine quantité de COV mais une fois la fenêtre ouverte, cette quantité chute au plus bas et devient nulle. Ceci est tout simplement causé par l'aération de la chambre car il y a eu renouvellement d'air neuf. Les composés organiques volatils augmentent avec la température et inversement. Du coup, on constate une baisse de température et de COV durant cette période où la fenêtre reste ouverte. En règle générale, les COV peuvent être dégagés par de nombreux matériaux tels que la colle, la peinture, les produits de nettoyage, les parfums chimiques et bien d'autres. C'est la raison pour laquelle il y a augmentation de ce composé à partir de 20h53. De ce fait, les COV sont des polluants de la vie quotidienne qui doivent être mieux surveillés.

- ❖ Le formaldéhyde



Le formaldéhyde fait parti de la famille des composés organiques volatils. Alors son analyse est la même que celui des COV et on remarque bien une baisse entre 19h59 et 20h08. En revanche, cette baisse ne s'applique pas durant toute la période d'ouverture de la fenêtre. On constate plutôt une fluctuation de ce dernier, ce qui est tout à fait incohérent. Alors cela peut venir d'un faux contact à cet instant là ou après le câblage du capteur.

❖ Le dioxyde de carbone

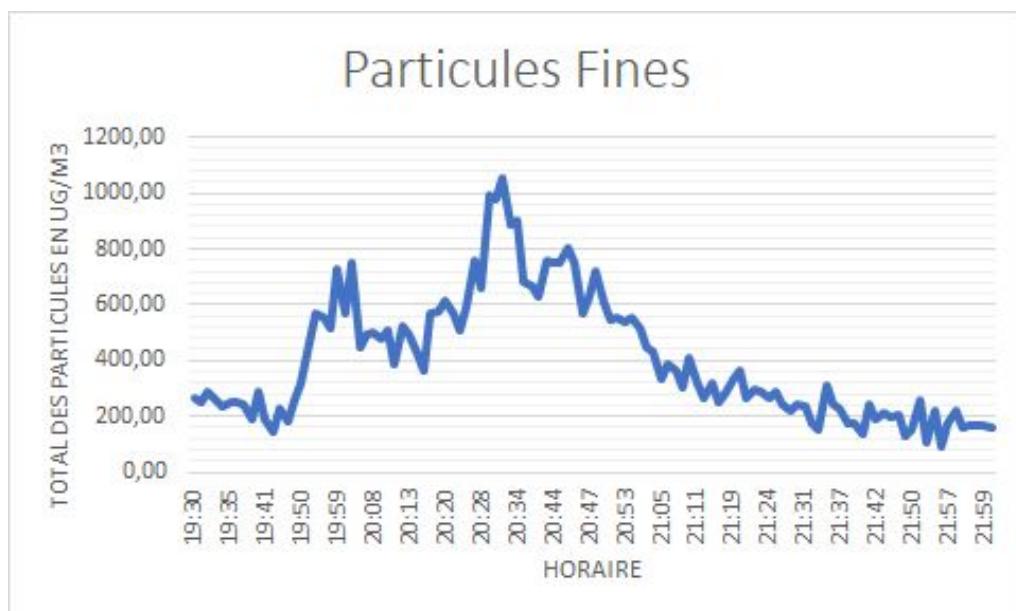


Ici est représenté le CO₂ en fonction du temps. Sa concentration est très importante avant l'ouverture de la fenêtre. Le CO₂ est une molécule produite et rejetée par l'organisme humain au cours de la respiration et donc plus on occupe

une pièce sans aérer, plus celle-ci est polluée par simple rejet répétitif de CO₂ et son taux augmente. C'est le cas sur notre figure.

Pour éviter ce type de problématique, il est primordial d'avoir un système d'aération performant afin de renouveler l'air. On peut très bien constater que lors de l'aération de la chambre, le taux de CO₂ décroît très rapidement et à la fermeture de la fenêtre sa concentration remonte progressivement.

❖ Les particules fines

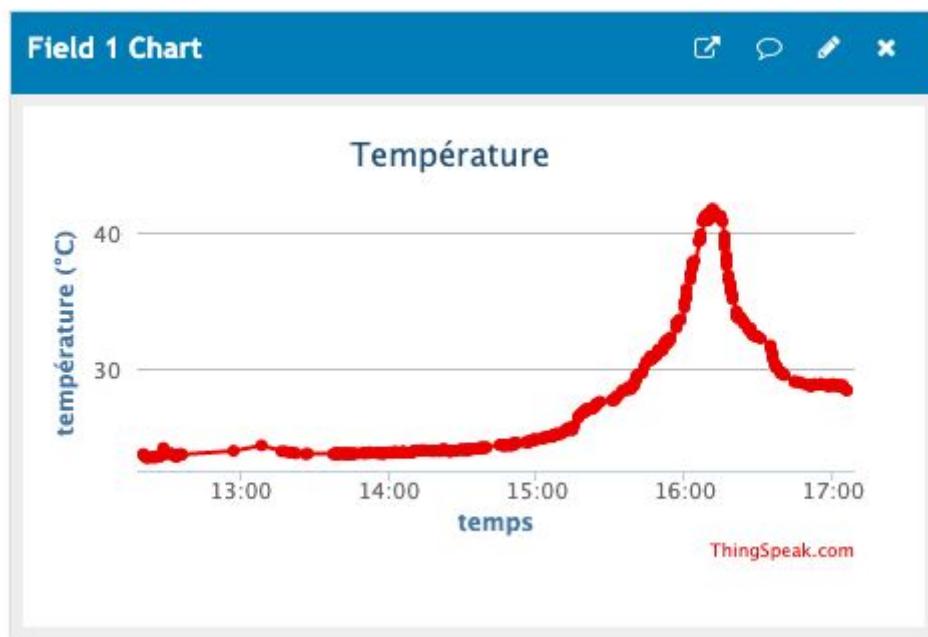


Sur cette figure représentative de la concentration des particules fines, on remarque que lorsque la fenêtre est ouverte il y a une hausse brutale des particules. Ce qui est très évident car on retrouve dans l'air extérieur beaucoup de particules par rapport à l'air intérieur. En réalité, les particules fines représentent la poussière et donc il suffit que la chambre soit bien nettoyée pour être moins poussiéreuse. Alors, les résultats obtenus sur la figure sont bien concluant.

b) Mesures recueillies à Fontenay-sous-Bois (94) dans une chambre

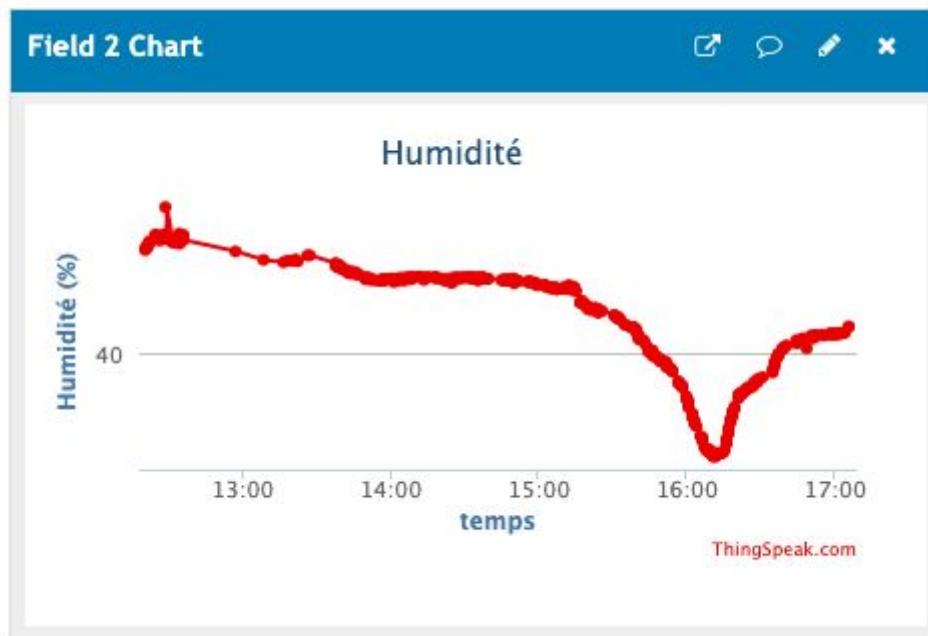
Contexte : Le 26 mai de 12h à 17h, fermeture de la fenêtre, personne se trouve dans la chambre. La chambre est exposée au soleil toute l'après midi et est proche d'un chantier.

❖ La température



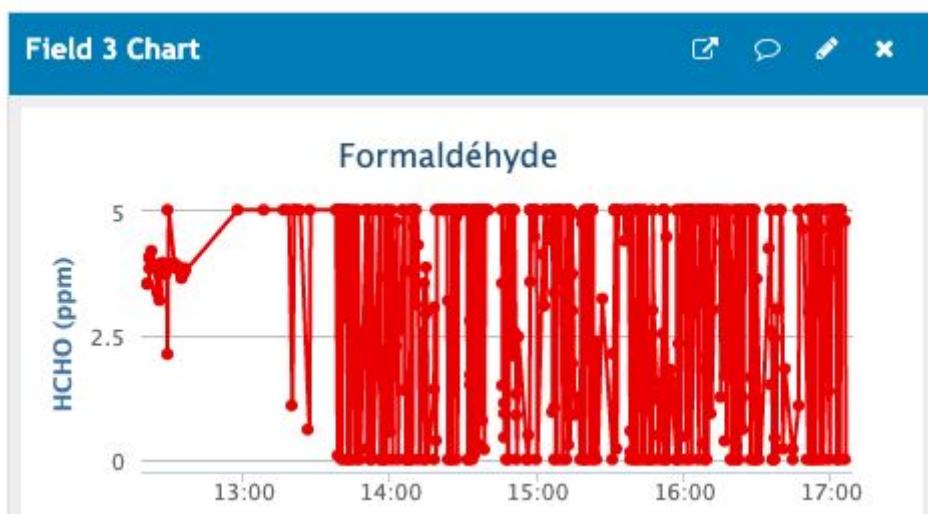
La chambre est exposée au soleil ce qui explique ces hautes températures.

- ❖ L'humidité



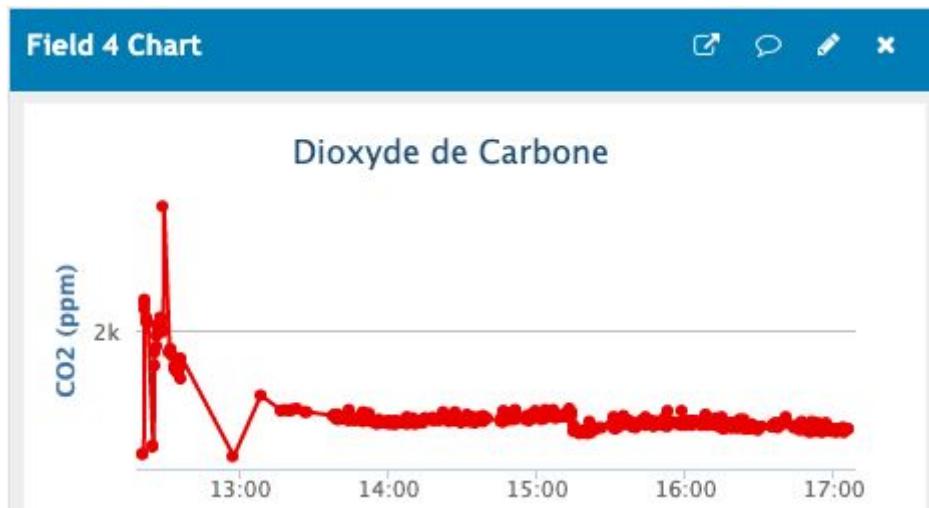
Nous pouvons voir que vers 16h, la température s'élève à 40 degrés à cause de la chaleur. La température augmente constamment et le pourcentage d'humidité diminue. Nous pouvons donc observer une corrélation entre température et humidité.

- ❖ Le formaldéhyde



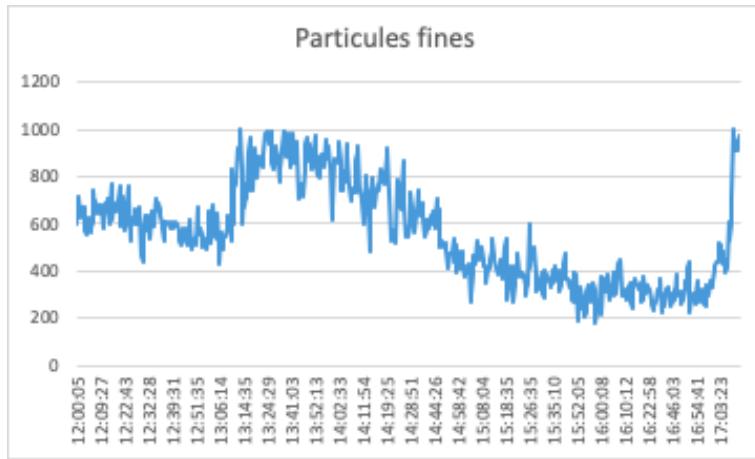
Pour le formaldéhyde, les valeurs sont en-dessous de 5 ppm, la masse molaire du formaldéhyde est de 30 g/mol, nous pouvons donc convertir l'unité en $\mu\text{g}/\text{m}^3$ ce qui donne $6.14 \mu\text{g}/\text{m}^3$. Suite à un barème de "valeurs repères" à appliquer au formaldéhyde, la valeur de formaldéhyde doit être en-dessous de $10 \mu\text{g}/\text{m}^3$, ce qui est convenable ici.

- ❖ Le dioxyde de carbone



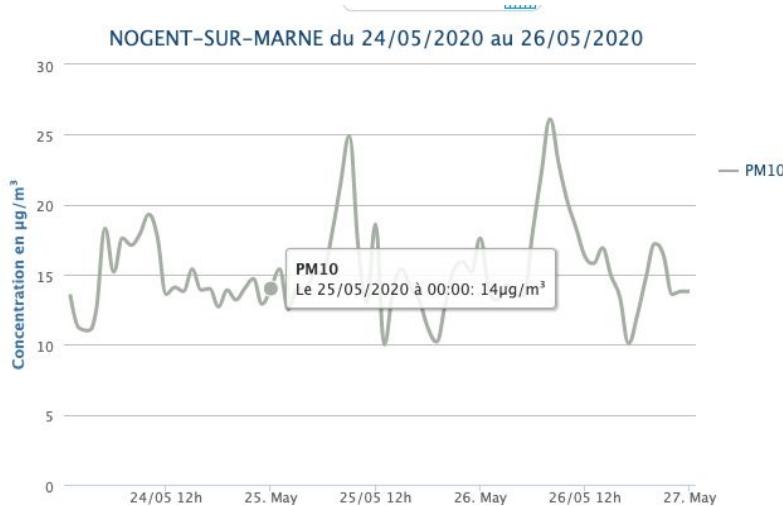
La valeur moyenne de CO₂ est environ $800 \mu\text{g}/\text{m}^3$. À l'intérieur, une concentration de CO₂ de 600 à 800 ppm est dite « correcte ». Les personnes sensibles peuvent avoir des effets secondaires si la concentration de CO₂ est au-delà de 1 000 ppm.

- ❖ Les particules fines

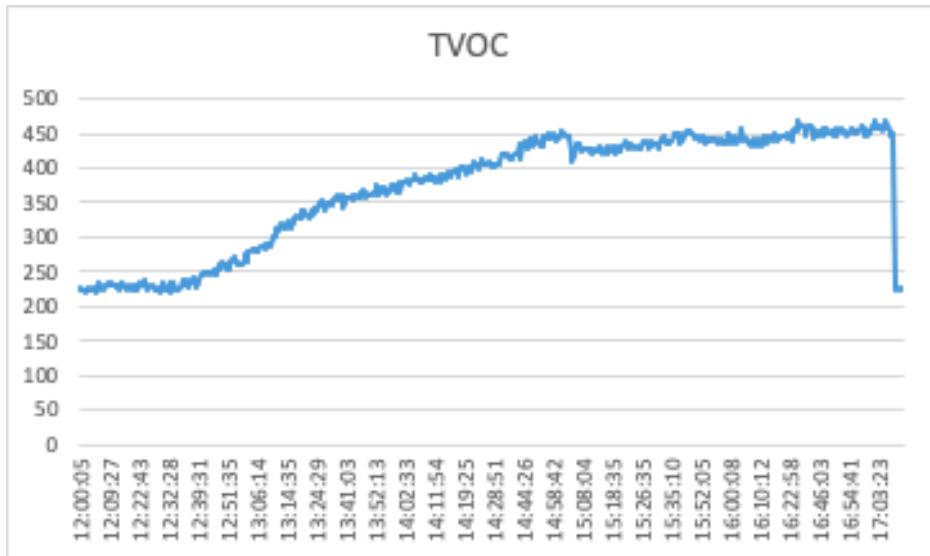


Les Particules fines mesurées par la balise sont le total de particules fines 0.1, 0.2, 0.5, 1, 2, 2.5, 5 et 10. La limitation de ThingSpeak fait que nous n'avons pas pu mesurer les valeurs séparément.

Pour comparer avec le site Airparif situé à Nogent-sur-Marne (environ 3 km), ce site mesure seulement le PM 10, et sa valeur est un peu près de $15 \mu\text{g}/\text{m}^3$, et dans la chambre, il y a presque pas de PM10 selon l'affichage du moniteur d'arduino. Cependant à cause de la proximité du chantier, les valeurs sont plus supérieures que la normale :



❖ Le Composé organique volatil



Les Composés Organiques Volatils (COV) sont des polluants issus des hydrocarbures, solvants, pots d'échappement.

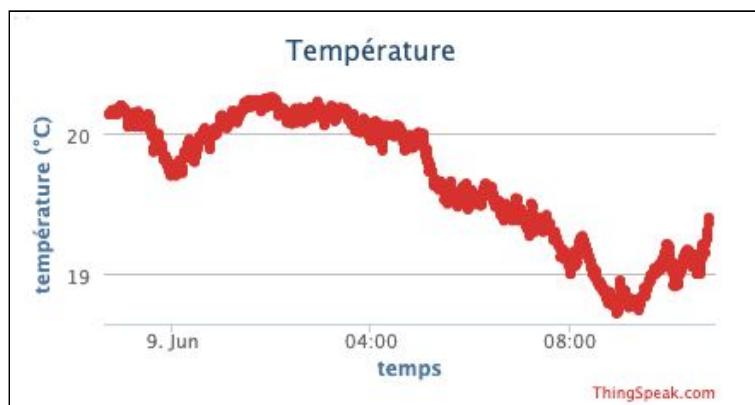
La valeur moyenne des mesures est d'environ 300 ppb.

c) Mesures recueillies à Verrière le Buisson (91) dans une chambre

Contexte : Les mesures ont été prises dans une chambre, dont les fenêtres ne permettent pas une isolation totale, dans la nuit du 8 au 9 juin de 23h à 11h. Le temps était frais avec une température extérieure annoncée à 16°C et une pluie légère. L'ouverture des volets s'est faite dans les alentours de 9h et de la fenêtre à 10h. L'utilisation de la douche dans la salle de bain collé à la chambre a eu lieu 10h30 .

Lors de ce test le capteur de COV n'était pas opérationnel

- ❖ La température

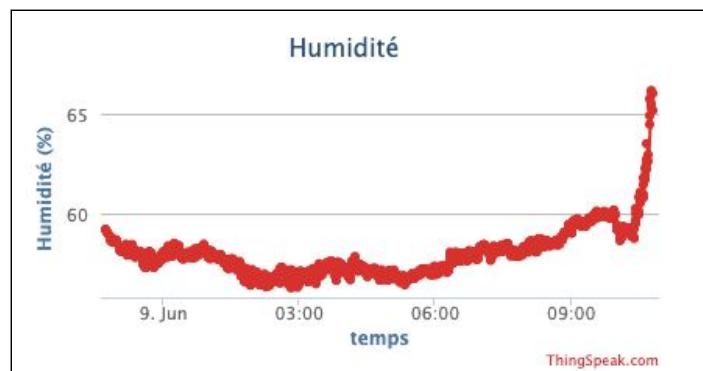


La température de la chambre pendant la nuit à tourner entre 21°C et 18°C. Ces valeurs, et la chute de température sont cohérentes avec les températures extérieures qui sont restées fraîches (16°C) et ont elles aussi chuté pendant la nuit.

Ces valeurs remontent à l'ouverture des fenêtres (et volets) qui permettent l'arrivée d'un air nouveau chauffé par le lever du soleil. L'augmentation continue avec l'utilisation de la salle de bain puis du sèche-cheveux.

Ces valeurs entrent dans les normes indiquées.

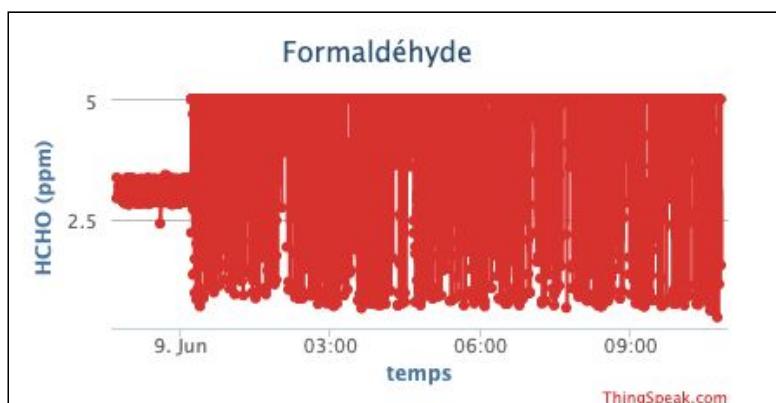
❖ L'humidité



Le taux d'humidité est resté relativement stable pendant la nuit. Il tend à augmenter légèrement à cause de l'occupation de la pièce close, en effet une personne dégage environ 0.2 litres par heure. Cela peut aussi être dû à la mauvaise isolation et au temps humide de la nuit. Ces valeurs sont acceptables mais sont dans la moyenne haute.

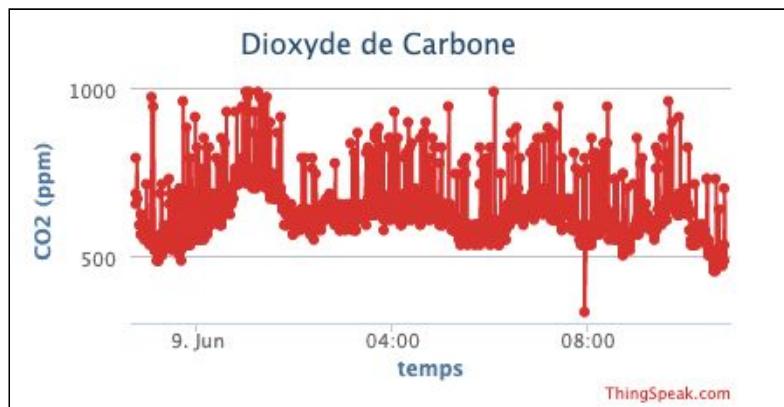
La baisse de l'humidité est dû à l'ouverture de la fenêtre et l'arrivée d'un air nouveau dans la pièce. L'augmentation brusque est directement reliée à l'utilisation de la salle de bain et de la douche qui dégagent beaucoup de vapeur d'eau (ici pièce mitoyenne à la chambre).

❖ Le formaldéhyde



Le tot de particule fini est constant et n'a pas varié avec l'ouverture de la fenêtre ou l'utilisation de la salle de bain. Ces valeurs sont à discuter.

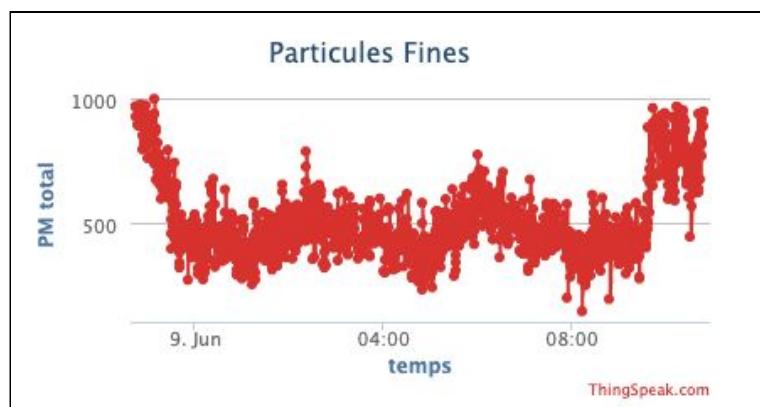
❖ Le dioxyde de carbone



La production de CO₂ est faite par l'Homme et est donc directement liée à l'occupation de la pièce. Celui-ci est resté relativement constant pendant la nuit (probablement dû à une mauvaise isolation des fenêtres qui a permis un renouvellement partiel de l'air).

A l'ouverture de la fenêtre à 10h le CO₂ chute.

❖ Les particules fines

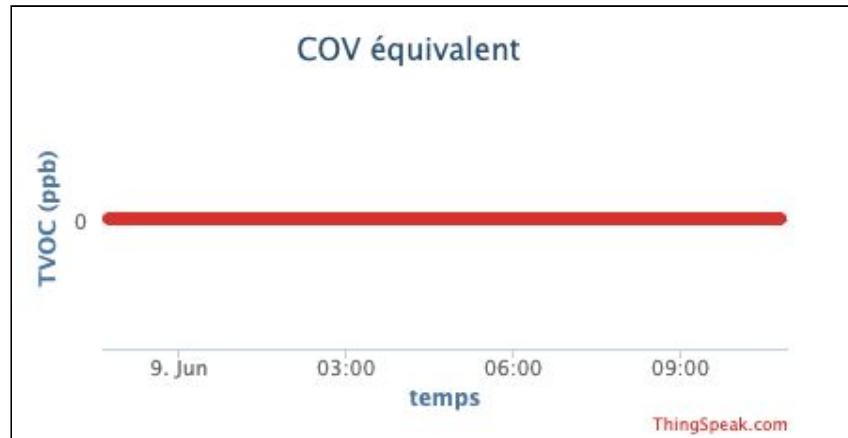


Le taux de particule fine reste constant durant la nuit, il augmente à l'ouverture de la fenêtre, ce qui reflète un air extérieur pollué.

Ces valeurs sont particulièrement hautes. Quand on prend en compte la localisation de l'appartement (zone verte) et la période (printemps), on peut émettre l'hypothèse que cela est lié au pollen.

De plus, à l'utilisation du sèche-cheveux les valeurs semblent augmenter (brassage de l'air et de la poussière).

- ❖ Le composé organique volatil



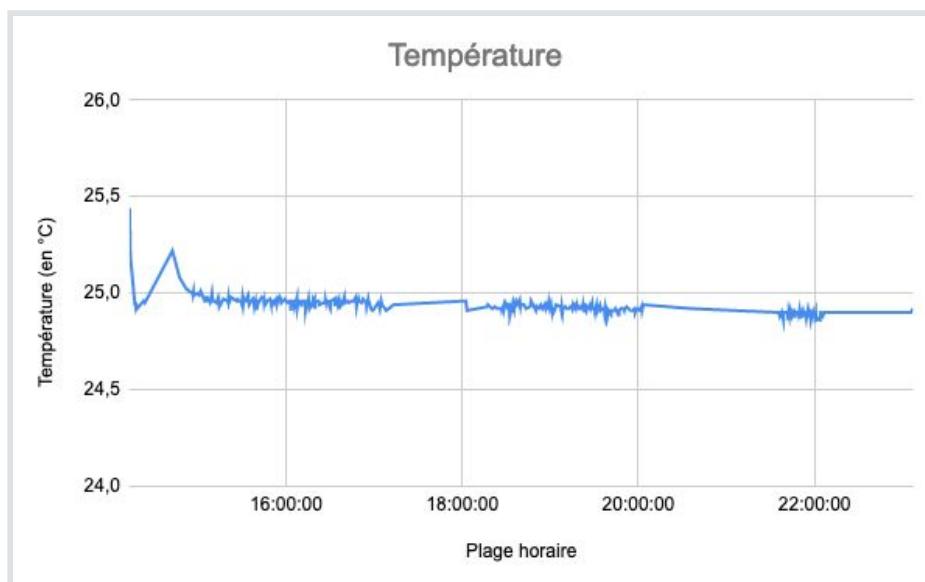
Lors de ce relevé un faux contact sur le capteur nous a empêché d'avoir un résultat.

d) Mesures recueillies à Paris (75) dans une salle de bain

Contexte : le 28 Juin à 16h dans la salle de bain jusqu'au 29 Juin à 13h, fenêtre légèrement ouverte (elle n'est pas complètement fermée). La connexion wifi n'était pas très bonne, il y a eu notamment des mesures qui ne se sont pas envoyées/ n'ont pas été recueillies. L'hypothèse la plus plausible est que la box est "trop" sécurisée. La clé de sécurité (WPA) a été modifiée mais aucun changement.

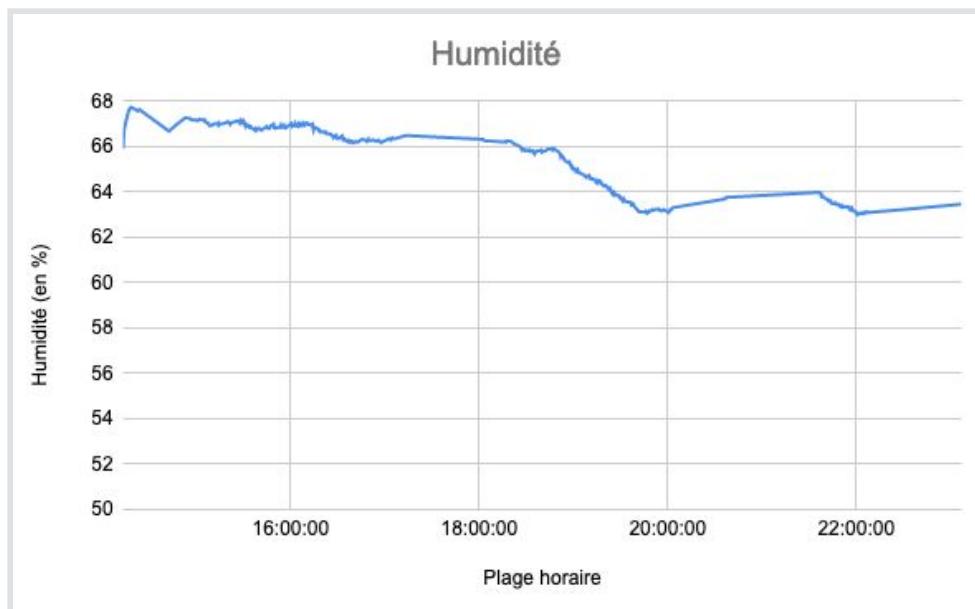
Nous avons voulu faire des mesures dans une salle de bain car nous le faisions souvent dans une chambre.

- ❖ La température



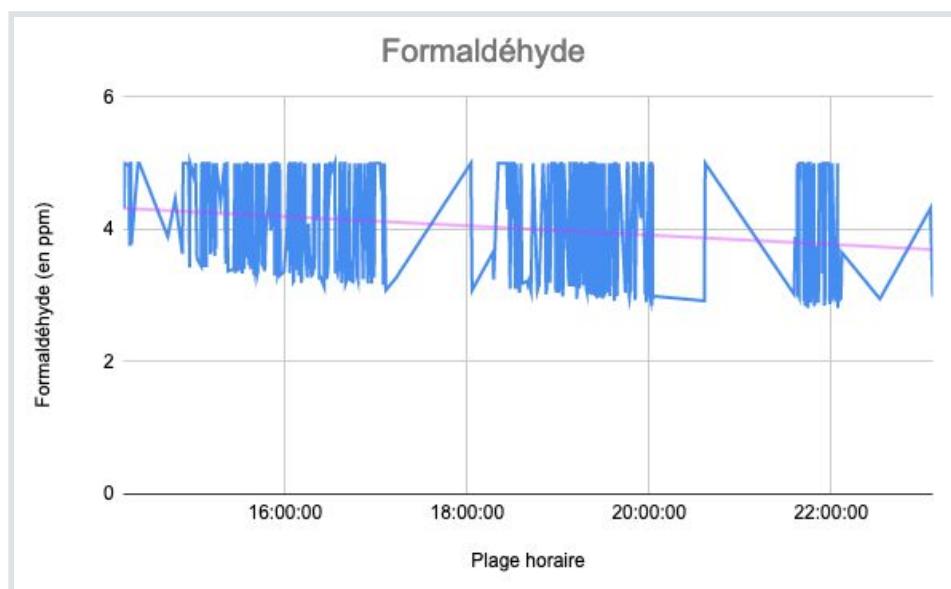
Dans une salle de bain la température idéale tourne autour de 17°C voire 22°C en cas d'utilisation. La température reste stable et se maintient à 25°C approximativement.

❖ L'humidité



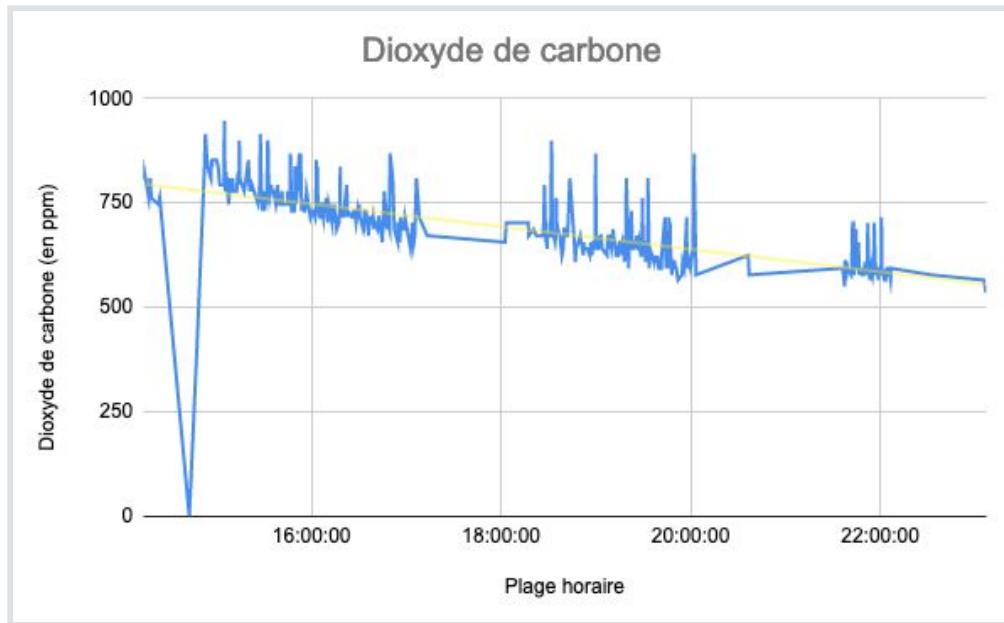
La salle de bain est naturellement une des pièces les plus humides. A chaque douche, de la vapeur d'eau est produite. Lorsque cette vapeur d'eau chaude entre en contact avec les surfaces plus froides (exemple les murs), elle se condense pour se retransformer en eau. Le phénomène de condensation est tout à fait normal dans une salle de bain. Le taux d'humidité avoisinant les 66% est donc normal. Il faut quand même faire attention car cela peut provoquer des dégâts importants.

❖ Le formaldéhyde



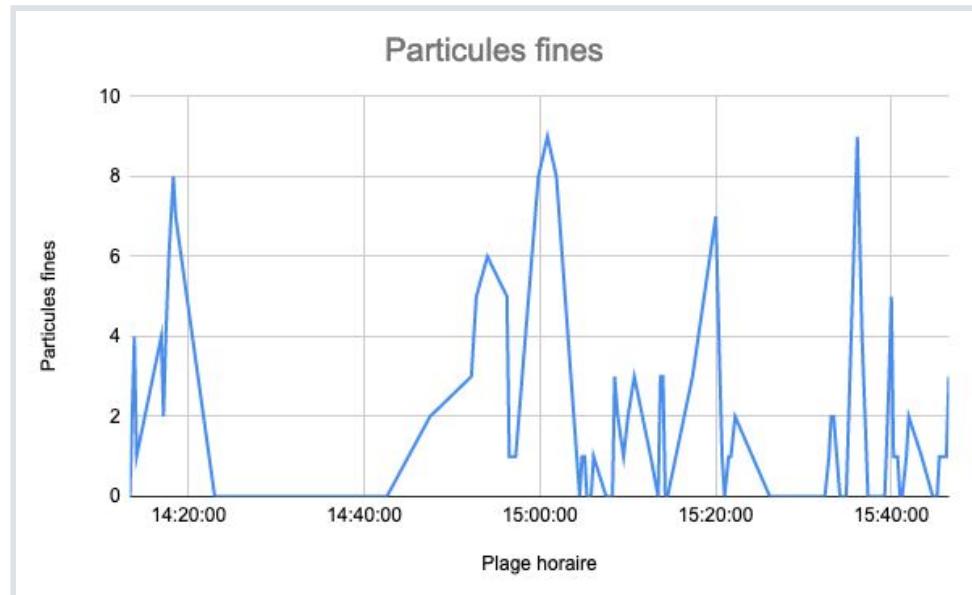
On peut constater comme pour la température que la concentration de formaldéhyde, qui fait parti de la famille des composés organiques volatils, tourne autour de 3 à 5 ppm. La courbe tendance (en rose) montre qu'en moyenne, elle est de 4 ppm.

❖ Le dioxyde de carbone



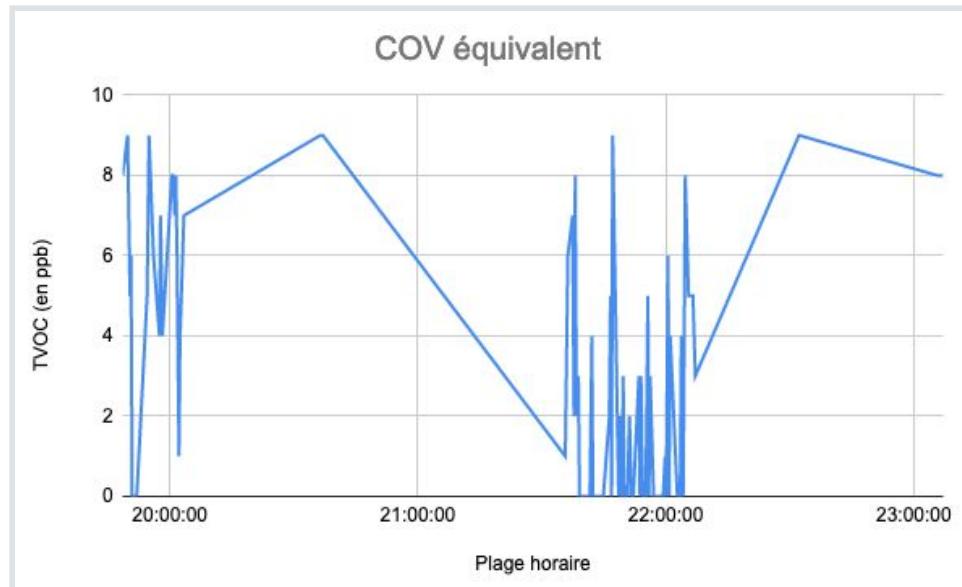
Ici est représenté le CO₂ en fonction du temps. Elle n'est pas si importante que ça puisque personne est entré dans la salle de bain, à part pour se nettoyer les mains. Etant donné que le CO₂ rejetée par l'organisme humain au cours de la respiration, ce n'est donc pas étonnant que la concentration soit faible. La valeur guide est de 1000 ppm.

❖ Les particules fines



Au vu de toutes les mesures prises depuis le début du projet, on observe que les valeurs recueillies sont anormales. Normalement, on devrait retrouver une quantité de particules fines moindres car celles-ci se retrouvent plus dans l'air extérieur que dans l'air intérieur.

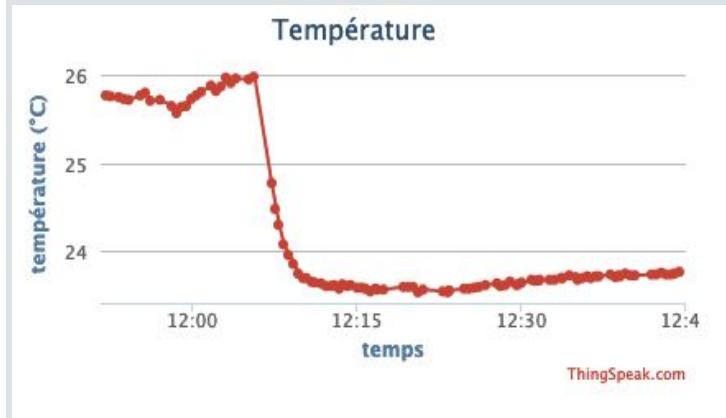
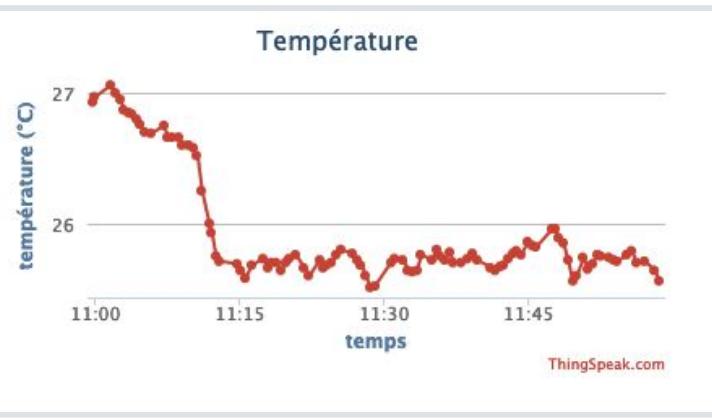
❖ Le composé organique volatil



De même, les valeurs recueillies ne sont pas cohérentes avec ce qu'on devrait observer. Comme dit précédemment, cela peut venir d'un faux contact ou d'un problème de wifi.

Contexte : Le 2 Juillet de 9h30 à 14h20, une personne se trouve dans la chambre au premier étage. Cette chambre est assez mal isolée. Il ne fait pas très beau. Les mesures ont été faites dans une plage horaire courte car nous avons voulu comparer les mesures une fois qu'on change de pièce. C'est à 12h07 que les mesures ont été prise dans la salle à manger en bas.

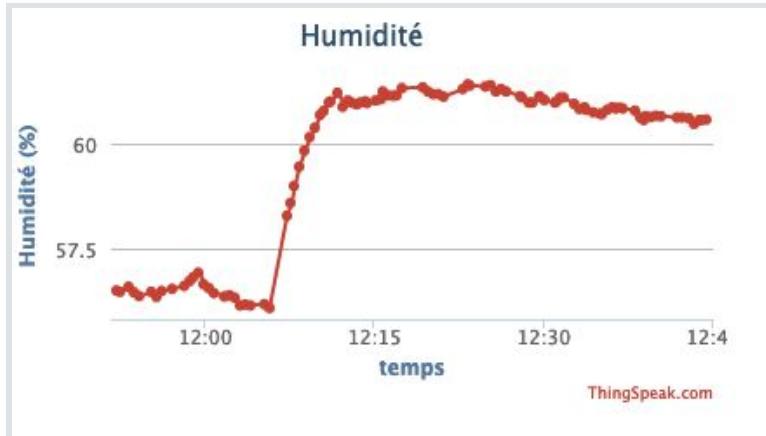
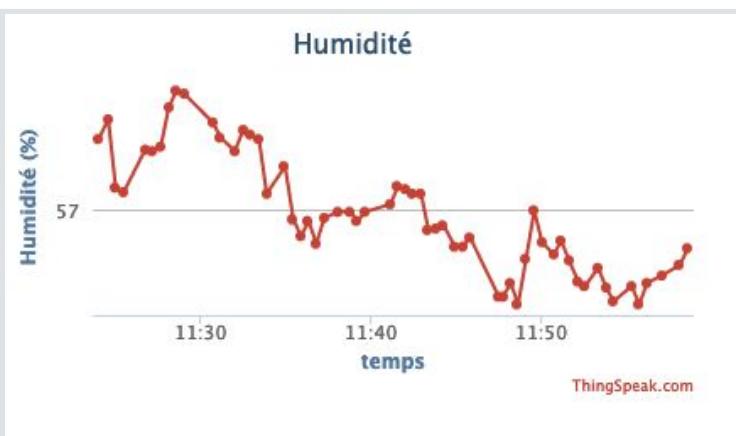
❖ La température



La température a fortement baissé dû à la météo. En effet, ce matin là était nuageux et puisque la chambre est mal isolée, l'air extérieur a pu rentrer. Après 11h, la température devient assez stable, et tourne autour de 25°C.

Le pic s'explique par le changement de pièce : on passe d'une chambre au premier étage à la salle à manger qui se trouve en bas. De plus, l'air dans la salle à manger est assez frais et le soleil n'était pas au rendez-vous. La température a chuté passant de 26°C à 23,6°C.

❖ L'humidité



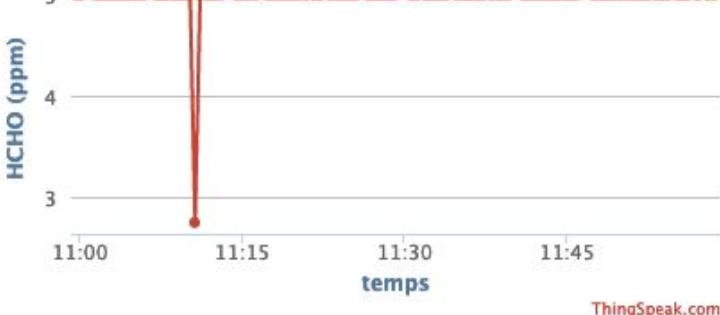
Le taux d'humidité dans la pièce est correcte car celui-ci se trouve entre 40 et 60% soit le seul recommandé. L'humidité reste quand même forte puisque le matin très tôt, deux personnes étaient dans la pièce en même temps. Il y a donc eu une quantité énorme de vapeur d'eau (respirer, transpirer, etc). De plus, la porte de la chambre a été plusieurs fois ouverte dû à des va-et-vient. Il faudrait donc aérer la pièce en ouvrant les fenêtres.

Le changement de pièce est la cause principale de la hausse de l'humidité. Tout d'abord parce que la teneur en vapeur d'eau de l'air diminue rapidement avec l'altitude. Donc il fera moins humide dans une pièce à l'étage qu'au rez-de-chaussée. Le taux d'humidité est supérieur à 60% mais s'explique aussi parce

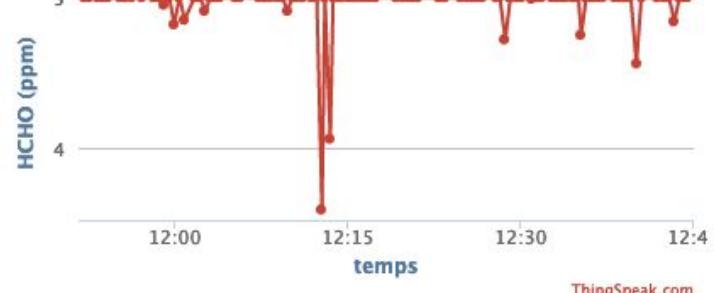
que la cuisine, qui ne dispose pas d'un système d'aération, se trouve à côté. Il se peut que la préparation du repas de la veille ou du petit-déjeuner ait un lien.

❖ Le formaldéhyde

Formaldéhyde



Formaldéhyde

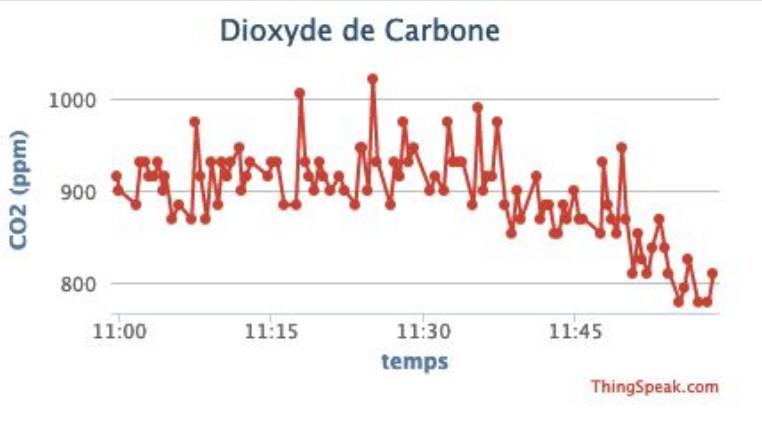


Le formaldéhyde est resté constant tout au long des mesures. Cependant, il y a eu un pic au moment où le capteur a été (un peu) bougé et touché. Il s'agit donc d'un faux contact à cet instant là.

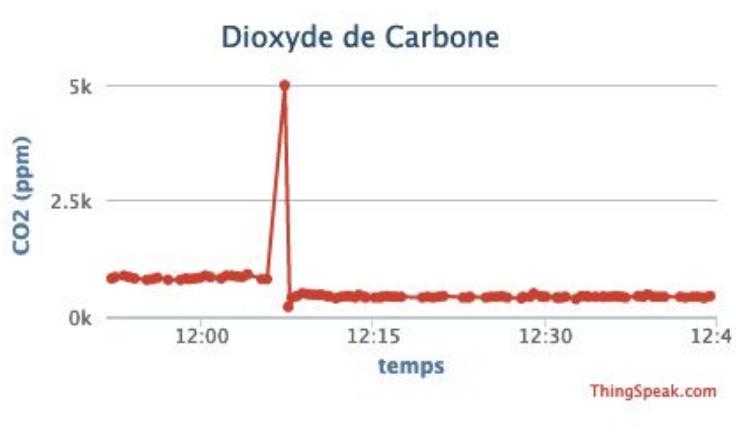
Dans la salle à manger, le taux de formaldéhyde reste correcte. On a 6,15 mg/m³ (car 1 ppm=1,23 mg/m³)

❖ Le dioxyde de carbone

Dioxyde de Carbone



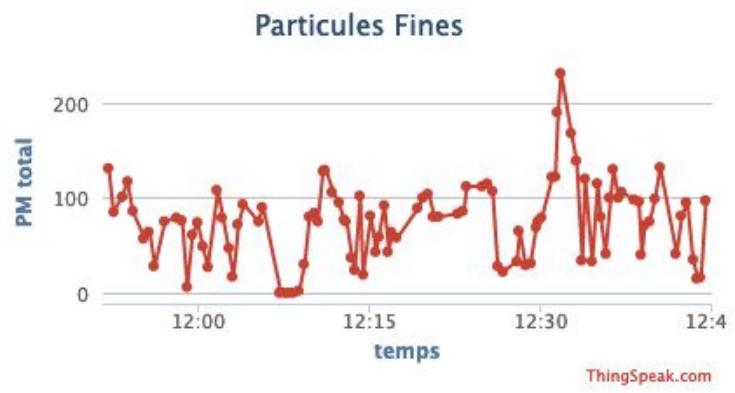
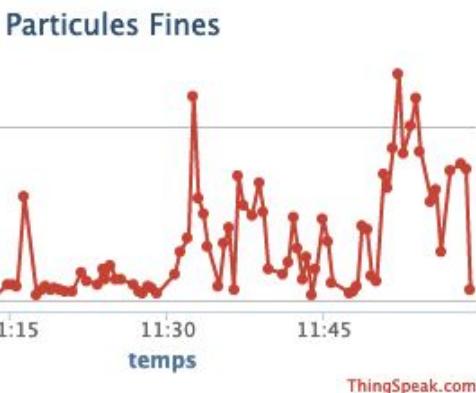
Dioxyde de Carbone



Le CO₂ est une molécule présente par le biais de la respiration. La concentration du CO₂ est très importante (celle-ci ne doit pas dépasser 1000 ppm). Comme dit plus haut, cette pièce n'a pas été aérée ce qui explique ce fort taux. Il faudrait aérer la pièce plus souvent au cours de la journée.

Le pic est incohérent, il s'agit de l'heure où le changement de pièce a été fait.

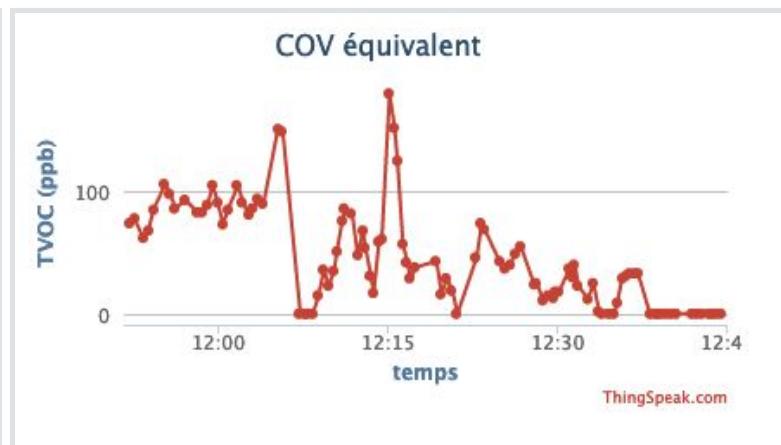
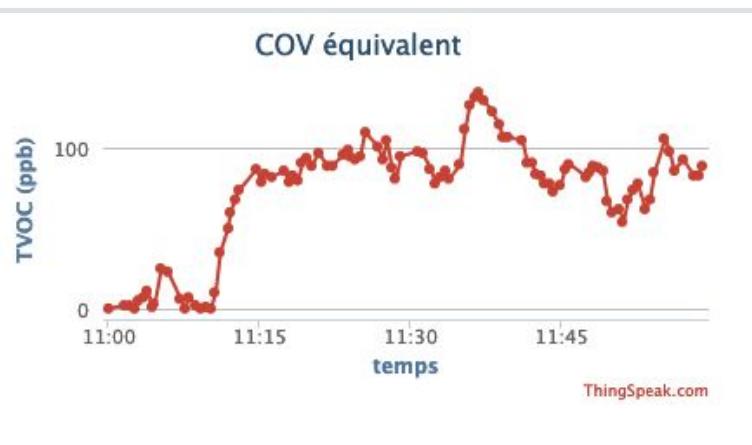
❖ Les particules fines



Cette figure représente de la concentration des particules fines, on remarque une faible présence de particules fines. La chambre a été nettoyée il y a 2-3 jours avec un aspirateur. Or, les particules fines représentent la poussière ce qui semble cohérent avec les résultats obtenus.

Les particules fines sont légèrement plus présentes dans la salle à manger mais leurs concentrations sont relativement convenables.

❖ Le composé organique volatil



On observe une certaine quantité de COV qui est assez basse puisqu'en règle générale, les COV peuvent être dégagés par de nombreux matériaux tels que la colle, la peinture, les produits de nettoyage, les parfums chimiques et bien d'autres. C'est pourquoi il y a une faible concentration dans la chambre.

Dans la salle à manger, il y a moins de concentration de COV car la température est plus faible. Ils doivent quand même être mieux surveillés.

Pour conclure, nous pouvons observer que les valeurs recueillies par les différents capteurs semblent donner un résultat cohérent par rapport à l'environnement et son évolution au cours du temps.

2) Analyse des données sur ThingSpeak

Après la campagne de mesures, il faut analyser les données récupérées par la balise. Le site Thingspeak permet cette analyse de données en utilisant le code Matlab et permet l'envoi des données traitées à un autre canal sélectionné.

a) Elimination des erreurs : valeurs nulles ou non correspondantes

Certaines valeurs sont nulles ou anormales dû à un mauvais contact. Pour éliminer ces zones d'erreurs, nous pouvons utiliser le code en Annexe 12.

Cela permet d'éliminer les valeurs anormales, mais sur le nouveau canal, on observe que les valeurs traitées de field1 (donc la température).

Nous avons pensé que c'est la fonction *thingSpeakWriter* qui ne peut pas envoyer les données trop fréquemment, et la fonction *pause* n'est pas supportable pour thingSpeak, nous n'avons pu régler ce bug.

b) Calcul de la valeur la plus élevée et la plus basse recueillies dans une journée

Nous avons écrit un code permettant de calculer la valeur la plus élevée et la plus basse au cours d'une journée.

Nous avons choisi d'afficher la température, la concentration de CO₂ et les particules fines :

Le temperature plus eleve en 24h est =

28.2700

C est à =

datetime

02-Jul-2020 08:34:04

Le temperature plus froid en 24h est =

23.3600

C est à =

datetime

02-Jul-2020 11:47:00

Le CO2 plus eleve en 24h est ppm =

4975

C est à =

datetime

02-Jul-2020 10:07:15

Le CO2 plus bas en 24h est ppm =

196.8800

C est à =

datetime

02-Jul-2020 10:07:41

```
Le concentration des particules fines plus eleve en 24h est =
```

```
231
```

```
timeMaxPM =
```

```
datetime
```

```
02-Jul-2020 10:31:47
```

```
Le concentration des particules fines plusbas en 24h est ppm =
```

```
0
```

```
timeMinPM =
```

```
datetime
```

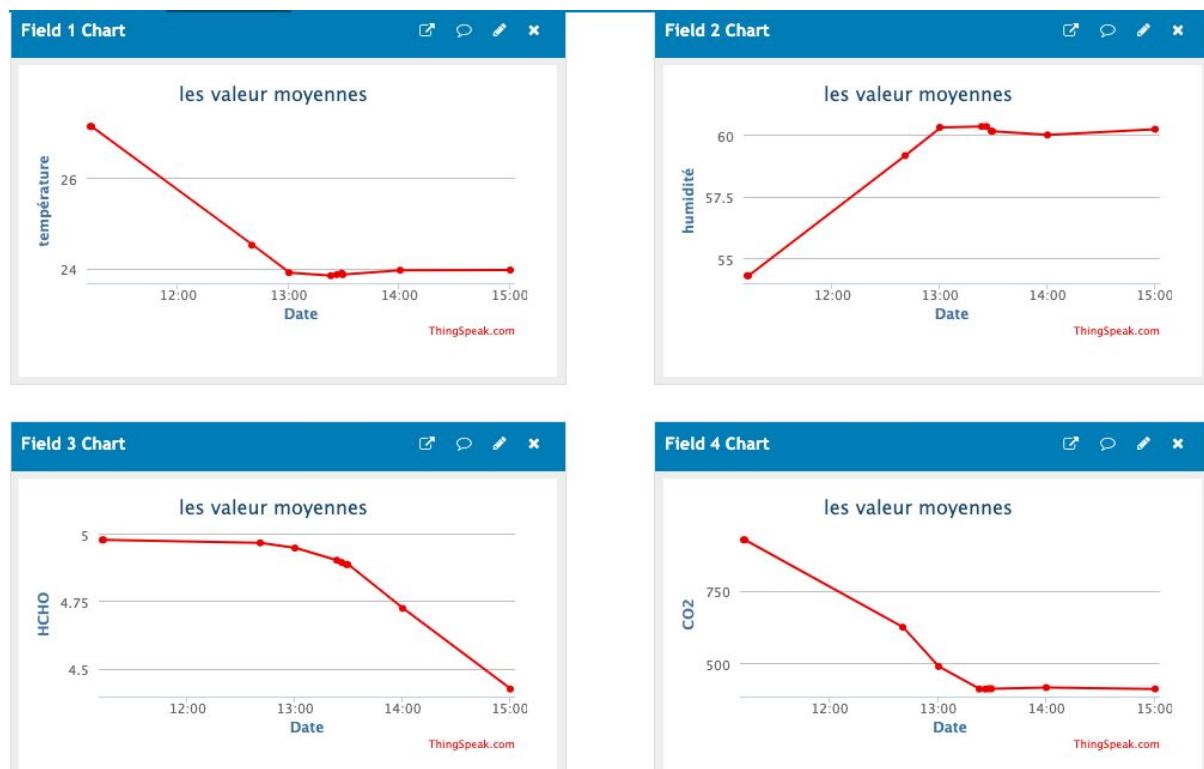
```
02-Jul-2020 08:14:29
```

Ce code se trouve dans l'Annexe 13.

c) Calcul des valeurs moyennes de chaque donnée

Le site ThingSpeak peut planifier une durée récurrence pour lancer régulièrement ce programme, nous avons choisi de lancer le code qui calcule les valeur moyennes chaque une heure quand la balise est active et ces valeurs moyennes sont envoyées à un autre canal qui permet d'enregistrer ces données.

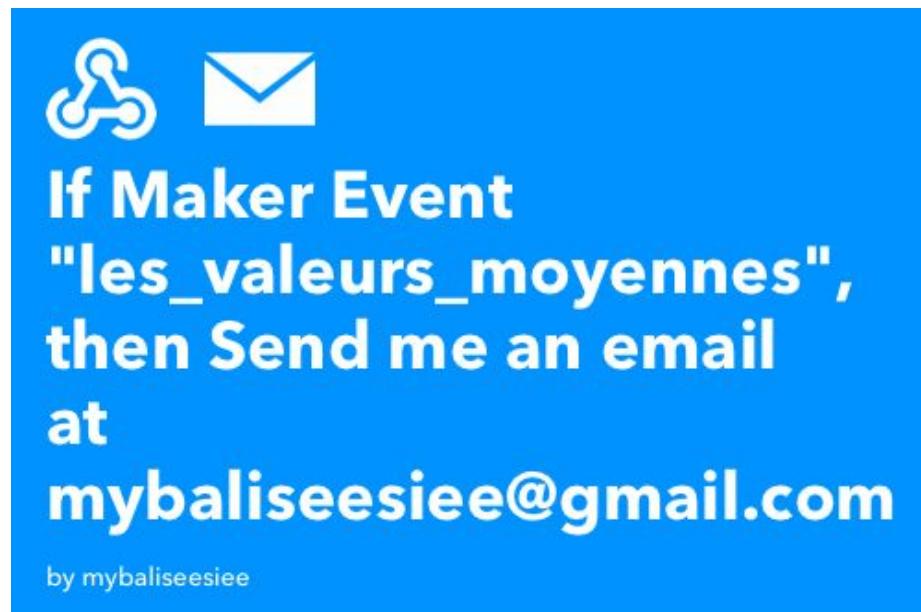
Name:	React valeur moyenne
Condition Type:	Numeric
Test Frequency:	Every 60 minutes
Last Ran:	2020-07-02 13:00
Channel:	Surveillance de la qualité de l'air intérieur
Condition:	Field 1 (température) is greater than 0
MATLAB Analysis:	les valeurs moyennes
Run:	Each time the condition is met
Created:	2020-06-29 9:27 am





Il permet de visualiser plus clairement le changement de valeurs en regardant les valeurs moyennes.

En outre, ces valeurs sont envoyées au site IFTTT au moment du lancement du programme, un notification et un mail est envoyé quand le site IFTTT reçoit les valeurs.



Le mail :

"les_valeurs_moyennes" pour cette heure est disponible ➔ Boîte de réception x

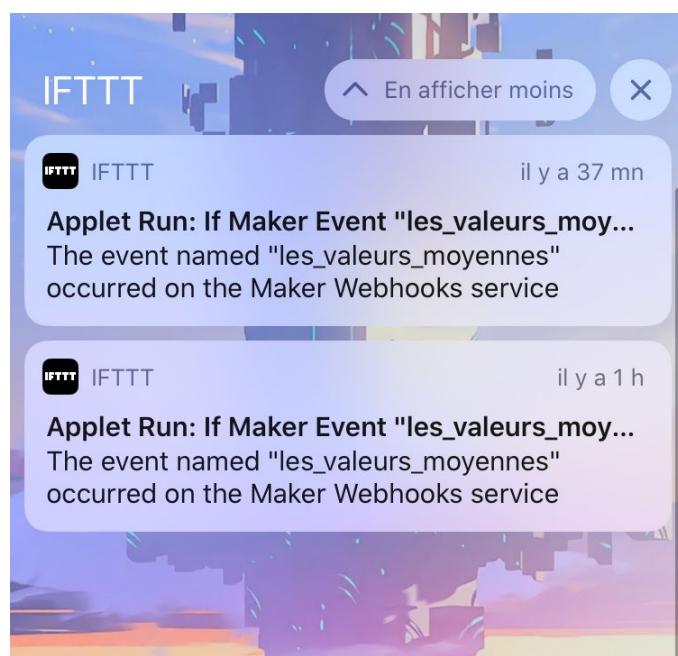
IFTTT Webhooks via IFTTT <action@ifttt.com> [Se désabonner](#)
À moi ▾

13:29 (il y a 0 minute) ★ ⏪ ⏴

les_valeurs_moyennes est disponible
à July 2, 2020 at 01:29PM
temperature:23.86632075471698 °C
humidité 60.14805555555555 %
HCHO:4.885963302752293 ppm

Le site IFTTT limite 3 valeurs maximum à envoyer dans le mail.

Notification dans le portable :



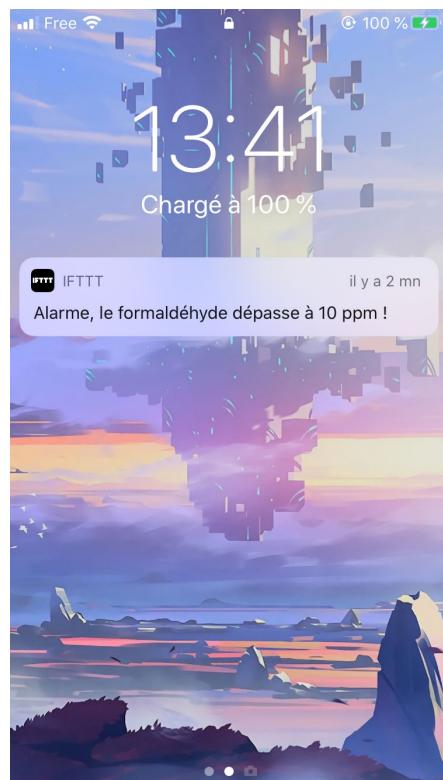
Le code est à retrouver dans l'Annexe 14.

d) Alarme

Si les valeurs dépassent celles des valeurs guides, le site IFTTT peut envoyer une alerte via une notification :



Sur le portable :



VII. Vidéo de Présentation

Nous avons en parallèle du projet, également réalisé une vidéo de présentation du projet. La vidéo a été réalisée sur Animaker et a une durée de 2 minutes maximum. La vidéo fait tout d'abord une présentation succincte de ce qu'est la qualité de l'air intérieur et également le confort thermique. Nous présentons ensuite la balise, les indicateurs sélectionnés et mesurés ainsi que les fonctionnalités annexes offertes.

Voici le lien de la vidéo :

https://www.youtube.com/watch?v=neZnZPeS85c&feature=emb_logo



CONCLUSION

Le travail présenté dans ce rapport, réalisé dans le cadre du projet de quatrième année au sein de ESIEE PARIS, a été le fruit d'un accomplissement de neuf mois de recherche.

Il a fallu répondre à la problématique induite, celle de développer une balise permettant de mesurer la qualité de l'air intérieur et du confort thermique. La tâche n'a pas été facile étant donné que de nombreuses balises existent sur le marché. Le problème était donc de concevoir notre balise à nous, avec des capteurs spécifiques qu'on estimait les plus performants pour évaluer la qualité de l'air intérieur.

Nous avons pu mener à bien la campagne de mesures puisque nous avons recueilli des données dans cinq villes différentes : Paris (75), Sartrouville (78), Verrières le buisson (91), Fontenay-sous-Bois (94) et Drancy (93). L'impression du PCB a pu se faire grâce à M. Julien Pagazani. De plus, en attendant d'avoir le matériel, nous avons écrit un code permettant de calculer les valeurs les plus hautes et les plus basses dans une journée, les valeurs moyenne et d'éliminer les zones d'erreurs existantes sur ThingSpeak.

Une amélioration de la balise sur le plan visuel aurait été un des points à améliorer; une forme plus originale et plus poussée pour se *démarquer* de ce qui existe actuellement. Concernant le support de la balise, nous n'avons pas pu réaliser son impression puisque l'école a dû rester fermé pour la sécurité de tous.

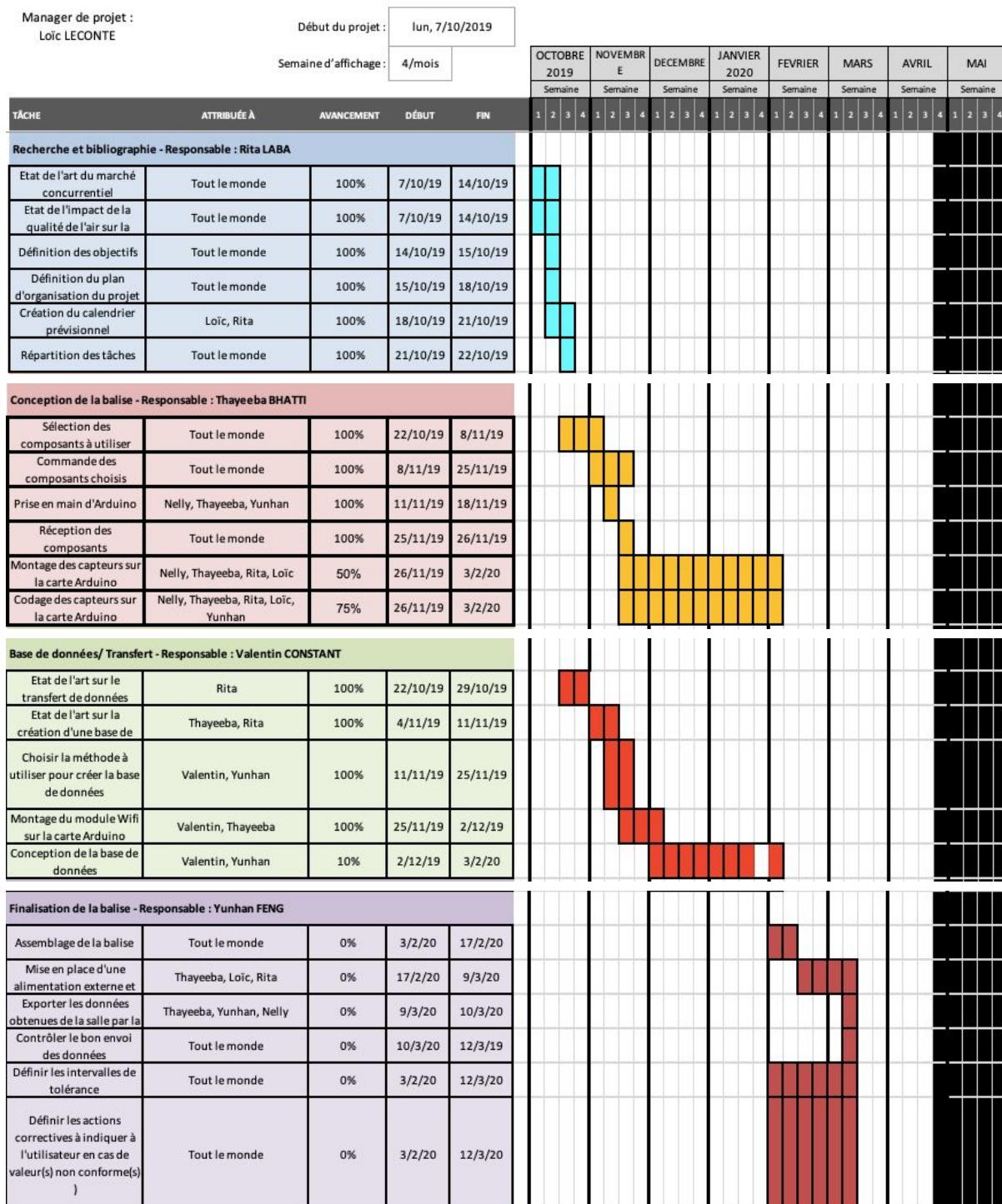
Sur le plan humain, nous avons été très heureux de travailler ensemble sur ce projet. Cela nous a permis de développer notre sens de l'organisation et de coopération. Nous avons chacun pu soumettre nos points de vue et en débattre. Le fait que chaque membre du groupe projet ait été responsable d'une partie du projet nous a également permis de prendre en maturité et en assurance et de comprendre les différents conflits qu'un manager de projet rencontre fréquemment.

ANNEXES

Annexe 1 : Diagramme de Gantt.

Balise de surveillance de la qualité de l'air et du confort thermique

DIAGRAMME DE GANTT



Campagne de mesures des données - Responsable : Nelly NGASSA									
Définir la campagne de mesures (type)	Nelly, Rita	0%	3/2/20	12/3/20					
Mener la campagne de mesures	Tout le monde	0%	12/3/20	30/3/20					
Analyse de la campagne de mesures - Responsable : Valentin CONSTANT									
Exporter les données obtenues lors de la campagne de mesures vers la base de données	Nelly, Yunhan, Thayeeba	0%	12/3/20	30/3/20					
Analyse des données de la campagne de mesures et confirmation des intervalles choisis	Tout le monde	0%	30/3/20	13/4/20					
Livrables - Responsable : Loïc LECONTE									
Rapport intermédiaire	Tout le monde	25%	7/10/19	21/1/20					
Soutenance intermédiaire	Tout le monde	0%	13/01/20	21/1/20					
Rapport final	Tout le monde	13%	7/10/19	18/4/20					
Soutenance finale	Tout le monde	0%	11/4/19	18/4/20					
Présentation vidéo du projet	Adéterminer	0%	26/3/20	18/4/20					
Pitch	Tout le monde	0%	26/3/20	18/4/20					

Annexe 2 : Répartition de l'équipe projet.

Tâche	Nelly	Thayeeba	Rita	Valentin	Yunhan	Loïc
A		X	X	X	X	X
B	X	X	X	X	X	X
C	X	X	X	X	X	X
D	X	X	X	X	X	X
E						X
F	X	X	X	X	X	X
G	X	X	X	X	X	X
H	X	X	X	X	X	X
I	X	X			X	
J	X	X	X	X	X	X
K	X	X	X			X
L	X	X	X		X	X
M			X			
N		X	X			

O				X	X	
P		X		X		
Q				X	X	
R	X	X	X	X	X	X
S		X	X			X
T	X	X			X	
U	X	X	X	X	X	X
V	X	X	X	X	X	X
W	X	X	X	X	X	X
X	X		X			
Y	X	X	X	X	X	X
Z	X	X			X	
AA	X	X	X	X	X	X
L1	X	X	X	X	X	X
L2	X	X	X	X	X	X
L3	X	X	X	X	X	X
L4	X	X	X	X	X	X
L5						
L6	X	X	X	X	X	X
Nombre de tâches affectées	24	26	25	22	25	23

Annexe 3 : Tableau dynamique de suivi de tâches.

Suivi des tâches dynamiques																																														
Loïc LECONTE January 19, 2020																																														
Recherche et bibliographie	Conception de la balise	Base de données/Transfert	Finalisation de la balise	Campagne de mesure de données	Analyse de la campagne de mesure	Livrables																																								
 Responsable : Rita LABA	 Responsable : Thayeeba BHATTI	 Responsable : Valentin CONSTANT	 Responsable : Yunhan FENG	 Responsable : Nelly NGASSA	 Responsable : Valentin CONSTANT	 Responsable : Loïc LECONTE																																								
<table border="1"> <thead> <tr> <th>À faire</th> <th>En cours</th> <th>Terminé</th> </tr> </thead> <tbody> <tr> <td>Phase: Finalisation de la balise Assemblage de la balise Échéance: 17/02/20</td><td>Phase: Finalisation de la balise Mise en place d'une alimentation externe et du PCB (si besoin) Échéance: 09/03/20</td><td>Phase: Recherche et bibliographie Etat de l'art du marché concurrentiel Échéance: 14/10/19</td></tr> <tr> <td>Phase: Finalisation de la balise Exporter les données obtenues de la salle par la balise vers la base de données Échéance: 10/03/20</td><td>Phase: Finalisation de la balise Contrôler le bon envoi des données Échéance: 12/03/20</td><td>Phase: Recherche et bibliographie Définition des objectifs Échéance: 15/10/19</td></tr> <tr> <td>Phase: Finalisation de la balise Définir les intervalles de tolérance Échéance: 12/03/20</td><td>Phase: Finalisation de la balise Définir les actions correctives à indiquer à l'utilisateur en cas de valeur(s) non conforme(s) Échéance: 12/03/20</td><td>Phase: Recherche et bibliographie Création du calendrier prévisionnel Échéance: 21/10/19</td></tr> <tr> <td>Phase: Campagne de mesures de données Mener la campagne de mesures Échéance: 30/03/20</td><td>Phase: Analyse de la campagne de mesures Exporter les données obtenues par la balise vers la base de données Échéance: 30/03/20</td><td>Phase: Recherche et bibliographie Répartition des tâches Échéance: 22/10/19</td></tr> <tr> <td>Phase: Analyse de la campagne de mesures Analyse des données de la campagne de mesures et confirmation des intervalles choisis Échéance: 13/04/20</td><td>Phase: Livrable Soutenance intermédiaire Échéance: 23/01/20</td><td>Phase: Conception de la balise Sélection des composants à utiliser Échéance: 08/11/19</td></tr> <tr> <td>Phase: Livrable Rapport final Échéance: 18/04/20</td><td>Phase: Livrable Soutenance finale Échéance: 18/04/20</td><td>Phase: Conception de la balise Commande des composants choisis Échéance: 25/11/19</td></tr> <tr> <td>Phase: Livrable Présentation vidéo du projet Échéance: 18/04/20</td><td>Phase: Livrable Pitch Échéance: 18/04/20</td><td>Phase: Conception de la balise Prise en main d'Arduino Échéance: 18/11/19</td></tr> <tr> <td></td><td></td><td>Phase: Conception de la balise Montage des capteurs sur la carte Arduino Échéance: 03/02/20</td></tr> <tr> <td></td><td></td><td>Phase: Conception de la balise Réception des composants Échéance: 26/11/19</td></tr> <tr> <td></td><td></td><td>Phase: Base de données/Transfert Choisir la méthode à utiliser pour créer la base de données Échéance: 25/11/19</td></tr> <tr> <td></td><td></td><td>Phase: Base de données/Transfert Etat de l'art sur le transfert de données Échéance: 29/10/19</td></tr> <tr> <td></td><td></td><td>Phase: Base de données/Transfert Etat de l'art sur la création d'une base de données Échéance: 11/11/19</td></tr> </tbody> </table>	À faire	En cours	Terminé	Phase: Finalisation de la balise Assemblage de la balise Échéance: 17/02/20	Phase: Finalisation de la balise Mise en place d'une alimentation externe et du PCB (si besoin) Échéance: 09/03/20	Phase: Recherche et bibliographie Etat de l'art du marché concurrentiel Échéance: 14/10/19	Phase: Finalisation de la balise Exporter les données obtenues de la salle par la balise vers la base de données Échéance: 10/03/20	Phase: Finalisation de la balise Contrôler le bon envoi des données Échéance: 12/03/20	Phase: Recherche et bibliographie Définition des objectifs Échéance: 15/10/19	Phase: Finalisation de la balise Définir les intervalles de tolérance Échéance: 12/03/20	Phase: Finalisation de la balise Définir les actions correctives à indiquer à l'utilisateur en cas de valeur(s) non conforme(s) Échéance: 12/03/20	Phase: Recherche et bibliographie Création du calendrier prévisionnel Échéance: 21/10/19	Phase: Campagne de mesures de données Mener la campagne de mesures Échéance: 30/03/20	Phase: Analyse de la campagne de mesures Exporter les données obtenues par la balise vers la base de données Échéance: 30/03/20	Phase: Recherche et bibliographie Répartition des tâches Échéance: 22/10/19	Phase: Analyse de la campagne de mesures Analyse des données de la campagne de mesures et confirmation des intervalles choisis Échéance: 13/04/20	Phase: Livrable Soutenance intermédiaire Échéance: 23/01/20	Phase: Conception de la balise Sélection des composants à utiliser Échéance: 08/11/19	Phase: Livrable Rapport final Échéance: 18/04/20	Phase: Livrable Soutenance finale Échéance: 18/04/20	Phase: Conception de la balise Commande des composants choisis Échéance: 25/11/19	Phase: Livrable Présentation vidéo du projet Échéance: 18/04/20	Phase: Livrable Pitch Échéance: 18/04/20	Phase: Conception de la balise Prise en main d'Arduino Échéance: 18/11/19			Phase: Conception de la balise Montage des capteurs sur la carte Arduino Échéance: 03/02/20			Phase: Conception de la balise Réception des composants Échéance: 26/11/19			Phase: Base de données/Transfert Choisir la méthode à utiliser pour créer la base de données Échéance: 25/11/19			Phase: Base de données/Transfert Etat de l'art sur le transfert de données Échéance: 29/10/19			Phase: Base de données/Transfert Etat de l'art sur la création d'une base de données Échéance: 11/11/19							
À faire	En cours	Terminé																																												
Phase: Finalisation de la balise Assemblage de la balise Échéance: 17/02/20	Phase: Finalisation de la balise Mise en place d'une alimentation externe et du PCB (si besoin) Échéance: 09/03/20	Phase: Recherche et bibliographie Etat de l'art du marché concurrentiel Échéance: 14/10/19																																												
Phase: Finalisation de la balise Exporter les données obtenues de la salle par la balise vers la base de données Échéance: 10/03/20	Phase: Finalisation de la balise Contrôler le bon envoi des données Échéance: 12/03/20	Phase: Recherche et bibliographie Définition des objectifs Échéance: 15/10/19																																												
Phase: Finalisation de la balise Définir les intervalles de tolérance Échéance: 12/03/20	Phase: Finalisation de la balise Définir les actions correctives à indiquer à l'utilisateur en cas de valeur(s) non conforme(s) Échéance: 12/03/20	Phase: Recherche et bibliographie Création du calendrier prévisionnel Échéance: 21/10/19																																												
Phase: Campagne de mesures de données Mener la campagne de mesures Échéance: 30/03/20	Phase: Analyse de la campagne de mesures Exporter les données obtenues par la balise vers la base de données Échéance: 30/03/20	Phase: Recherche et bibliographie Répartition des tâches Échéance: 22/10/19																																												
Phase: Analyse de la campagne de mesures Analyse des données de la campagne de mesures et confirmation des intervalles choisis Échéance: 13/04/20	Phase: Livrable Soutenance intermédiaire Échéance: 23/01/20	Phase: Conception de la balise Sélection des composants à utiliser Échéance: 08/11/19																																												
Phase: Livrable Rapport final Échéance: 18/04/20	Phase: Livrable Soutenance finale Échéance: 18/04/20	Phase: Conception de la balise Commande des composants choisis Échéance: 25/11/19																																												
Phase: Livrable Présentation vidéo du projet Échéance: 18/04/20	Phase: Livrable Pitch Échéance: 18/04/20	Phase: Conception de la balise Prise en main d'Arduino Échéance: 18/11/19																																												
		Phase: Conception de la balise Montage des capteurs sur la carte Arduino Échéance: 03/02/20																																												
		Phase: Conception de la balise Réception des composants Échéance: 26/11/19																																												
		Phase: Base de données/Transfert Choisir la méthode à utiliser pour créer la base de données Échéance: 25/11/19																																												
		Phase: Base de données/Transfert Etat de l'art sur le transfert de données Échéance: 29/10/19																																												
		Phase: Base de données/Transfert Etat de l'art sur la création d'une base de données Échéance: 11/11/19																																												

Annexe 4 : Code utilisé pour le capteur de température et d'humidité seul.

```
//Bibliothèque utilisée pour le capteur
#include <SHT1x.h>

// Définitions des pins utilisées
#define dataPin 10
#define clockPin 11
SHT1x sht1x(dataPin, clockPin);

void setup()
{
    Serial.begin(9600);
    Serial.println("Starting up");
}

void loop()
{
    float temp_c;
    float humidity;

    // Lis les valeurs du capteur
    temp_c = sht1x.readTemperatureC();
    humidity = sht1x.readHumidity();

    // Affiche les valeurs sur le port série
    Serial.print("Temperature: ");
    Serial.print(temp_c, DEC);
    Serial.print("C / ");
    Serial.print(humidity);
    Serial.println("%");

    delay(500);
}
```

Annexe 5 : Code utilisé pour le capteur de HCHO seul.

→ Fonction analogique

```

#define SensorAnalogPin A2 //this pin read the analog voltage from the HCHO sensor
#define VREF 5.0 //voltage on AREF pin

void setup()
{
    Serial.begin(9600);
}

void loop()
{
    Serial.print(analogReadPPM());
    Serial.println("ppm");
    delay(1000);
}

float analogReadPPM()
{
    float analogVoltage = analogRead(SensorAnalogPin) / 1024.0 * VREF;
    float ppm = 3.125 * analogVoltage - 1.25; //linear relationship(0.4V for 0 ppm and 2V for 5ppm)
    if(ppm<0) ppm=0;
    else if(ppm>5) ppm = 5;
    return ppm;
}

```

→ Fonction numérique

```

#include <DFRobotHCHOSensor.h>
#include <SoftwareSerial.h>

#define SensorSerialPin 10 //this pin read the uart signal from the HCHO sensor

SoftwareSerial sensorSerial(SensorSerialPin,SensorSerialPin);

DFRobotHCHOSensor hchoSensor(&sensorSerial);

void setup()
{
    sensorSerial.begin(9600); //the baudrate of HCHO is 9600
    sensorSerial.listen();
    Serial.begin(9600);
}

void loop()
{
    if(hchoSensor.available()>0)
    {
        Serial.print(hchoSensor.uartReadPPM());|
        Serial.println("ppm");
    }
}

```

Annexe 6 : Code utilisé pour le capteur de CO2 seul.

```
int sensorIn = A1;

void setup(){
    Serial.begin(9600);
    // Set the default voltage of the reference voltage
    analogReference(DEFAULT);
}

void loop(){
    //Read voltage
    int sensorValue = analogRead(sensorIn);

    // The analog signal is converted to a voltage
    float voltage = sensorValue*(5000/1024.0);
    //Serial.println(voltage);
    if(voltage == 0)
    {
        Serial.println("Fault");
    }
    else if(voltage < 400)
    {
        Serial.println("preheating");
    }
    else
    {
        int voltage_diference=voltage-400;
        float concentration=voltage_diference*50.0/16.0;
        // Print Voltage
        //Serial.print("voltage:");
        Serial.print(voltage);
        Serial.println("mv");
        //Print CO2 concentration
        Serial.print(concentration);
        Serial.println("ppm");
    }
    delay(500);
}
```

Annexe 7 : Code utilisé pour le capteur de poussières seul.

```
#include "Arduino.h"
#include "serialReadPMValue.h"

uint16_t PM01Value=0;           //define PM1.0 value of the air detector module
uint16_t PM2_5Value=0;          //define PM2.5 value of the air detector module
uint16_t PM10Value=0;           //define PM10 value of the air detector module

#define receiveDatIndex 32

uint8_t receiveDat[receiveDatIndex]; //receive data from the air detector module

void setup() {

    // put your setup code here, to run once:

    Serial.begin(9600); //set the serial's Baudrate of the air detector module
}

void loop()
{

    int length = serialRead(Serial,receiveDat,receiveDatIndex,5); //change the serial port:Serial1,Serial2..
    int checkSum=checkValue(receiveDat,receiveDatIndex);

    if(length==checkSum)
    {

        PM01Value=transmitPM01(receiveDat); //count PM1.0 value of the air detector module
        PM2_5Value=transmitPM2_5(receiveDat); //count PM2.5 value of the air detector module
        PM10Value=transmitPM10(receiveDat); //count PM10 value of the air detector module

    }
    static unsigned long OledTimer=millis();           //every 0.5s update the temperature and humidity from DHT11 sensor
    if (millis() - OledTimer >=1000)
    {
        OledTimer=millis();

        Serial.print("PM1.0: "); //send PM1.0 data to bluetooth
        Serial.print(PM01Value);
        Serial.println(" ug/m3");

        Serial.print("PM2.5: "); //send PM1.0 data to bluetooth
        Serial.print(PM2_5Value);
        Serial.println(" ug/m3");

        Serial.print("PM10: "); //send PM1.0 data to bluetooth
        Serial.print(PM10Value);
        Serial.println(" ug/m3");
    }
}
```

```

}

char checkValue(unsigned char *thebuf, char leng)
{
    char receiveflag=0;
    int receiveSum=0;

    for(int i=0; i<(leng-2); i++){
        receiveSum=receiveSum+thebuf[i];
    }
    receiveSum=receiveSum + 0x42;

    if(receiveSum == ((thebuf[leng-2]<<8)+thebuf[leng-1])) //check the serial data
    {
        receiveSum = 0;
        receiveflag = 1;
    }
    return receiveflag;
}

int transmitPM01(unsigned char *thebuf)
{
    int PM01Val;
    PM01Val=((thebuf[3]<<8) + thebuf[4]); //count PM1.0 value of the air detector module
    return PM01Val;
}

//transmit PM Value to PC
int transmitPM2_5(unsigned char *thebuf)
{
    int PM2_5Val;
    PM2_5Val=((thebuf[5]<<8) + thebuf[6]); //count PM2.5 value of the air detector module
    return PM2_5Val;
}

//transmit PM Value to PC
int transmitPM10(unsigned char *thebuf)
{
    int PM10Val;
    PM10Val=((thebuf[7]<<8) + thebuf[8]); //count PM10 value of the air detector module
    return PM10Val;
}

```

Annexe 8 : Code utilisé pour le capteur de qualité de l'air seul.

```
#include <Wire.h>
#include "Adafruit_SGP30.h"

Adafruit_SGP30 sgp;

/* return absolute humidity [mg/m^3] with approximation formula
 * @param temperature [°C]
 * @param humidity [%RH]
 */
uint32_t getAbsoluteHumidity(float temperature, float humidity) {
    // approximation formula from Sensirion SGP30 Driver Integration chapter 3.15
    const float absoluteHumidity = 216.7f * (humidity / 100.0f) * 6.112f * exp((17.62f * temperature) / (243.12f + temperature)) / (273.15f + temperature); // [g/m^3]
    const uint32_t absoluteHumidityScaled = static_cast<uint32_t>(1000.0f * absoluteHumidity); // [mg/m^3]
    return absoluteHumidityScaled;
}

void setup() {
    Serial.begin(9600);
    Serial.println("SGP30 test");

    if (! sgp.begin()){
        Serial.println("Sensor not found :(");
        while (1);
    }
    Serial.print("Found SGP30 serial #:");
    Serial.print(sgp.serialnumber[0], HEX);
    Serial.print(sgp.serialnumber[1], HEX);
    Serial.print(sgp.serialnumber[2], HEX);

    // If you have a baseline measurement from before you can assign it to start, to 'self-calibrate'
    //sgp.setIAQBaseline(0x8E68, 0x8F41); // Will vary for each sensor!
}

int counter = 0;
void loop() {
    // If you have a temperature / humidity sensor, you can set the absolute humidity to enable the humidity compensation for the air quality signals
    //float temperature = 22.1; // [°C]
    //float humidity = 45.2; // [%RH]
    //sgp.setHumidity(getAbsoluteHumidity(temperature, humidity));

    if (! sgp.IAQmeasure()) {
        Serial.println("Measurement failed");
        return;
    }
    Serial.print("TVOC "); Serial.print(sgp.TVOC); Serial.print(" ppb\t");
    Serial.print("eCO2 "); Serial.print(sgp.eco2); Serial.println(" ppm");

    if (! sgp.IAQmeasureRaw()) {
        Serial.println("Raw Measurement failed");
        return;
    }
    Serial.print("Raw H2 "); Serial.print(sgp.rawH2); Serial.print(" \t");
    Serial.print("Raw Ethanol "); Serial.print(sgp.rawEthanol); Serial.println("");

    delay(1000);

    counter++;
    if (counter == 30) {
        counter = 0;

        uint16_t TVOC_base, eCO2_base;
        if (! sgp.getIAQBaseline(&eCO2_base, &TVOC_base)) {
            Serial.println("Failed to get baseline readings");
            return;
        }
        Serial.print("****Baseline values: eCO2: 0x"); Serial.print(eCO2_base, HEX);
        Serial.print(" & TVOC: 0x"); Serial.println(TVOC_base, HEX);
    }
}
```

Annexe 9 : Code utilisé pour l'assemblage des capteurs.

```
#include <SHTlx.h>
#include <DFRobotHCHOsensor.h>
#include <Arduino.h>
#include <SoftwareSerial.h>

#define LENG 31 //0x42 + 31 bytes equal to 32 bytes
unsigned char buf[LENG];
#define SensorSerialPin 10 //this pin read the uart signal from the HCHO sensor

//temperature et humidite
#define dataPin 9
#define clockPin 11
SHTlx shtlx(dataPin, clockPin);

static unsigned long OledTimer=millis();

int PM01Value=0; //define PM1.0 value of the air detector module
int PM2_5Value=0; //define PM2.5 value of the air detector module
int PM10Value=0; //define PM10 value of the air detector module
int mesure_particule=0; //
int sensorIn = A0; //CO2

SoftwareSerial PMSerial(5, 6); // RX, TX pour le capteur de Particules fines
SoftwareSerial sensorSerial(SensorSerialPin,SensorSerialPin);

DFRobotHCHOsensor hchoSensor(&sensorSerial);

void setup()
{
    PMSerial.begin(9600);
    sensorSerial.begin(9600); //the baudrate of HCHO is 9600
    PMSerial.setTimeout(1500);
    sensorSerial.listen();
    Serial.begin(9600);
    analogReference(DEFAULT);
    Serial.println("Starting up");
}

void loop()
{
    if(PMSerial.find(0x42)){
        PMSerial.readBytes(buf,LENG);

        if(buf[0] == 0x4d){
            if(checkValue(buf,LENG)){
                PM01Value=transmitPM01(buf); //count PM1.0 value of the air detector module
                PM2_5Value=transmitPM2_5(buf); //count PM2.5 value of the air detector module
                PM10Value=transmitPM10(buf); //count PM10 value of the air detector module
            }
        }
    }
}
```

```

if(measure_particule==0 && hchoSensor.available()>0)
{
    Serial.print(hchoSensor.uartReadPPM());
    Serial.println("ppm");
    sensorSerial.end();
    PMSerial.begin(9600);
    PMSerial.setTimeout(1500);
    OledTimer=millis();
    measure_particule=1;
}

if (measure_particule && millis() - OledTimer >=1000)
{
    Serial.print("PM1.0: ");
    Serial.print(PM01Value);
    Serial.println(" ug/m3");

    Serial.print("PM2.5: ");
    Serial.print(PM2_5Value);
    Serial.println(" ug/m3");

    Serial.print("PM1 0: ");
    Serial.print(PM10Value);
    Serial.println(" ug/m3");
    Serial.println();
    measure_particule=0;
    sensorSerial.begin(9600);
    sensorSerial.listen();
}

//Read voltage
int sensorValue = analogRead(sensorIn);

// The analog signal is converted to a voltage
float voltage = sensorValue*(5000/1024.0);
if(voltage == 0)
{
    Serial.println("Fault");
}
else if(voltage < 400)
{
    Serial.println("preheating");
}
else
{
    int voltage_difference=voltage-400;
    float concentration=voltage_difference*50.0/16.0;
    // Print Voltage
    Serial.print("voltage:");
    Serial.print(voltage);
}

```

```

    Serial.println("mv");
    //Print CO2 concentration
    Serial.print(concentration);
    Serial.println("ppm(CO2)");
}
delay(100);

float temp_c;
float temp_f;
float humidity;

// Read values from the sensor
temp_c = shtlx.readTemperatureC();
temp_f = shtlx.readTemperatureF();
humidity = shtlx.readHumidity();

// Print the values to the serial port
Serial.print("Temperature: ");
Serial.print(temp_c, DEC);
Serial.print("C / ");
Serial.print(temp_f, DEC);
Serial.print("F. Humidity: ");
Serial.print(humidity);
Serial.println("%");

delay(500);

}

char checkValue(unsigned char *thebuf, char leng)
{
    char receiveflag=0;
    int receiveSum=0;

    for(int i=0; i<(leng-2); i++){
        receiveSum=receiveSum+thebuf[i];
    }
    receiveSum=receiveSum + 0x42;

    if(receiveSum == ((thebuf[leng-2]<<8)+thebuf[leng-1])) //check the serial data
    {
        receiveSum = 0;
        receiveflag = 1;
    }
    return receiveflag;
}

int transmitPM01(unsigned char *thebuf)
{
    int PM01Val;
    PM01Val=((thebuf[3]<<8) + thebuf[4]); //count PM1.0 value of the air detector module
    return PM01Val;
}

```

```

//transmit PM Value to PC
int transmitPM2_5(unsigned char *thebuf)
{
    int PM2_5Val;
    PM2_5Val=((thebuf[5]<<8) + thebuf[6]); //count PM2.5 value of the air detector module
    return PM2_5Val;
}

//transmit PM Value to PC
int transmitPM10(unsigned char *thebuf)
{
    int PM10Val;
    PM10Val=((thebuf[7]<<8) + thebuf[8]); //count PM10 value of the air detector module
    return PM10Val;
}

```

Annexe 10 : Code utilisé pour le module wifi seul.

```

#include <SoftwareSerial.h>
#define RX 10
#define TX 11
String AP = "Fyh";           // CHANGE ME
String PASS = "Fyh00000"; // CHANGE ME
String API = "0";   // CHANGE ME
String HOST = "perso.esiee.fr/~fengy/toto.txt";
String PORT = "80";
String field = "field1";
int countTrueCommand;
int countTimeCommand;
boolean found = false;
int valSensor = 1;
SoftwareSerial esp8266(RX,TX);

void setup() {
    Serial.begin(9600);
    esp8266.begin(115200);
    sendCommand("AT",5,"OK");
    sendCommand("AT+CNMODE=1",5,"OK");
    sendCommand("AT+CNJAP=\\" + AP + "\",\"" + PASS + "\",20,"OK");
}
void loop() {
    valSensor = getSensorData();
    // a modifier "GET" + @ de PHP puis ajoute les variables(donc les données des capteur instantané)
    String getData = "GET /update?api_key=" + API + "&" + field + "=" + String(valSensor);
    sendCommand("AT+CIPMUX=1",5,"OK");
    sendCommand("AT+CIPSTART=0,\"TCP\",\"" + HOST + "\",\"" + PORT,15,"OK");
    sendCommand("AT+CIPSEND=0," + String(getData.length() + 4),4,>"); 
    esp8266.println(getData); delay(1500); countTrueCommand++;
    sendCommand("AT+CIPCLOSE=0",5,"OK");
    sendCommand("AT+CIFSR",5,"OK");
}

int getSensorData(){
    return random(1000); // Replace with
}

```

```
void sendCommand(String command, int maxTime, char readReplay[])
{
    Serial.print(countTrueCommand);
    Serial.print(". at command => ");
    Serial.print(command);
    Serial.print(" ");
    while(countTimeCommand < (maxTime*1))
    {
        esp8266.println(command);//at+cipsend
        if(esp8266.find(readReplay))//ok
        {
            found = true;
            break;
        }

        countTimeCommand++;
    }

    if(found == true)
    {
        Serial.println("OYI");
        countTrueCommand++;
        countTimeCommand = 0;
    }

    if(found == false)
    {
        Serial.println("Fail");
        countTrueCommand = 0;
        countTimeCommand = 0;
    }

    found = false;
}
```

Annexe 11 : Code final utilisé pour l'ensemble des capteurs

```
#include <Arduino.h>
#include <SoftwareSerial.h>
#include <SHT1x.h>

#define dataPin 8 //branche capteur d'humidité et température
#define clockPin 9

//HCHO

#define SensorAnalogPin A2 //this pin read the analog voltage from the HCHO sensor
#define VREF 5.0

SHT1x sht1x(dataPin, clockPin);

// module wifi branchement RX11 TX10
#define RX 10
#define TX 11

//CO2
int sensorIn = A0; //CO2

//PM2.5
#define LENG 31 //0x42 + 31 bytes equal to 32 bytes
unsigned char buf[LENG];
int PM01Value=0; //define PM1.0 value of the air detector module
int PM2_5Value=0; //define PM2.5 value of the air detector module
int PM10Value=0; //define PM10 value of the air detector module
int PART0_3Value=0; //define particule 0.3 value of the air detector module
int PART0_5Value=0; //define particule 0.5 value of the air detector module
int PART1Value=0; //define particule 1 value of the air detector module
int PART2_5Value=0; //define particule 2.5 value of the air detector module
int PART5Value=0; //define particule 5 value of the air detector module
int PART10Value=0; //define particule 10 value of the air detector module
int mesure_particule=1; //
int total=0;

SoftwareSerial PMSerial(6,5); //TX:6 RX:5

//////////////////TVOC/////////////////
#include <Wire.h>
#include "Adafruit_SGP30.h"
Adafruit_SGP30 sgp;

uint32_t getAbsoluteHumidity(float temperature, float humidity) {
    // approximation formula from Sensirion SGP30 Driver Integration chapter 3.15
    const float absoluteHumidity = 216.7f * ((humidity / 100.0f) * 6.112f * exp((17.62f * temperature)
    / (243.12f + temperature)) / (273.15f + temperature)); // [g/m^3]
    const uint32_t absoluteHumidityScaled = static_cast<uint32_t>(1000.0f * absoluteHumidity); // [mg/m^3]
    return absoluteHumidityScaled;
}
```

```

int valTVOC;

String AP = "Ebox-Bhatti"; // le nom du wifi connecté
String PASS = "taslima2001"; // le mot de passe de wifi
//String API = "JP4P21BFOURZQ1Y1"; // une clé Write_API
//String API = "ADHOKXMM2WUQCCHJL"; pour 3 capteurs //thingspeak yunhan.feng@edu.esiee.fr mdp: Fyh19960114
String API = "F881VKL04YQHGNUD";
String HOST = "api.thingspeak.com"; // site
String PORT = "80"; // port Internet
String field1 = "field1"; // récupération des données de température
String field2 = "field2"; // récupération des données d'humidité
String field3 = "field3"; //HCHO
String field4 = "field4"; //CO2
String field5 = "field5"; //PM2.5
String field6 = "field6"; //TVOC
int countTrueCommand;
int countTimeCommand;
boolean found = false;
float valSensorT = 1;
float valSensorH = 1;
float valHCHO = 1;
float temp_c; //température en degree
float humidity; // humudité en %
float valCO2; //co2 en ppm
float valtotal;//ensemble de particule fine

SoftwareSerial esp8266(RX, TX);

void setup() {
    Serial.begin(9600);
    esp8266.begin(115200);
    // PMSerial.setTimeout(1500);
    esp8266.listen();
    analogReference(DEFAULT);
    Serial.println("SGP30 test");

    if (! sgp.begin()){
        Serial.println("Sensor not found :(");
        while (1);
    }
    Serial.print("Found SGP30 serial #");
    Serial.print(sgp.serialnumber[0], HEX);
    Serial.print(sgp.serialnumber[1], HEX);
    Serial.println(sgp.serialnumber[2], HEX);
}

//////////////////////////////TEMPERATURE et HUMIDITE///////////
//float temp_f;

temp_c = shlx.readTemperatureC();

//temp_f = shlx.readTemperatureF();
humidity = shlx.readHumidity();

// Print the values to the serial port
Serial.print("Temperature: ");
Serial.print(temp_c, DEC);
Serial.print(" Humidity: ");
Serial.print(humidity);
Serial.println("%");

//delay(200);
valSensorT = getSensorDataT();
valSensorH = getSensorDataH();

```

```

//////////HCHO//////////HCHO//////////HCHO//////////HCHO//////////HCHO//////////HCHO
delay(200);
Serial.print(analogReadPPM());
Serial.println("ppm");
valHCHO = analogReadPPM();

//////////CO2//////////CO2//////////CO2//////////CO2//////////CO2//////////CO2
valCO2 = CO2();
//////////PM//////////PM//////////PM//////////PM//////////PM//////////PM
valtotal = PM();
//////////TVOC//////////TVOC//////////TVOC//////////TVOC//////////TVOC
valTVOC = TVOC(); //SCL: A5 SDA:A4

sendCommand("AT",5,"OK");
sendCommand("AT+CWMODE=1",5,"OK"); // on selecte le module wifi en mode "Station"
sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK"); // connexion avec le wifi selectionné

String getData = "GET /update?api_key="+ API +"&" + field1 +"="+ String(valSensorT)+"&" +field2
+"="+String(valSensorH) + "&" + field3 + "=" +String(valHCHO) + "&" + field4 + "=" +String(valCO2)+ 
"&" + field5 + "=" +String(valtotal)+"&" + field6 + "=" +String(valTVOC);
sendCommand("AT+CIPMUX=1",5,"OK");
sendCommand("AT+CIPSTART=0,\"TCP\",\"" + HOST +"\","+ PORT,15,"OK");

sendCommand("AT+CIPSEND=0," +String(getData.length())+4,4,>"); 
esp8266.println(getData);delay(1500);countTrueCommand++;
sendCommand("AT+CIPCLOSE=0",5,"OK");

}

void sendCommand(String command, int maxTime, char readReplay[]) {
    Serial.print(countTrueCommand);
    Serial.print(". at command => ");
    Serial.print(command);
    Serial.print(" ");
    while(countTimeCommand < (maxTime*1))
    {
        esp8266.println(command);//at+cipsend
        if(esp8266.find(readReplay)) //ok
        {
            found = true;
            break;
        }
    }
    countTimeCommand++;
}

float analogReadPPM()
{
    float analogVoltage = analogRead(SensorAnalogPin) / 1024.0 * VREF;
    float ppm = 3.125 * analogVoltage - 1.25; //linear relationship(0.4V for 0 ppm and 2V for 5ppm)
    if(ppm<0) ppm=0;
    else if(ppm>5) ppm = 5;
    return ppm;
}

float getSensorDataT()
{
    return sh1lx.readTemperatureC(); // Replace with
}

float getSensorDataH()
{
    return sh1lx.readHumidity(); // Replace with
}

```

```

void sendCommand(String command, int maxTime, char readReplay[]) {
    Serial.print(countTrueCommand);
    Serial.print(". at command => ");
    Serial.print(command);
    Serial.print(" ");
    while(countTimeCommand < (maxTime*1))
    {
        esp8266.println(command);//at+cipsend
        if(esp8266.find(readReplay))//ok
        {
            found = true;
            break;
        }

        countTimeCommand++;
    }

    if(found == true)
    {
        Serial.println("OYI");
        countTrueCommand++;
        countTimeCommand = 0;
    }
    if(found == false)
    {
        Serial.println("Fail");
        countTrueCommand = 0;
        countTimeCommand = 0;
    }

    found = false;
}

float CO2()
{
    //Read voltage
    int sensorValue = analogRead(sensorIn);

    // The analog signal is converted to a voltage
    float voltage = sensorValue*(5000/1024.0);
    if(voltage == 0)
    {
        Serial.println("Fault");
    }
    else if(voltage < 400)
    {
        Serial.println("preheating");
    }
    else
    {
        int voltage_diference=voltage-400;
        float concentration=voltage_diference*50.0/16.0;
        // Print Voltage
        Serial.print("voltage:");
        Serial.print(voltage);
        Serial.println("mv");
        //Print CO2 concentration
        Serial.print(concentration);
        Serial.println("ppm");
        return concentration;
    }
}

```

```

    delay(100);
}

//////////////////////////////PM2.5////////////////////////

int PM()
{
    esp8266.end();
    PMSerial.begin(9600);
    PMSerial.setTimeout(1500);
    delay(2000);

    if(PMSerial.find(0x42)){
        PMSerial.readBytes(buf,LENG);
        Serial.println("PM: lecture");
        if(buf[0] == 0x4d){
            if(checkValue(buf,LENG)){
                PM01Value=transmitPM01(buf); //count PM1.0 value of the air detector module
                PM2_5Value=transmitPM2_5(buf); //count PM2.5 value of the air detector module
                PM10Value=transmitPM10(buf); //count PM10 value of the air detector module
                PART0_3Value=transmitPart0_3(buf);
                PART0_5Value=transmitPart0_5(buf);
                PART1Value=transmitPart1(buf);
                PART2_5Value=transmitPart2_5(buf);
                PART5Value=transmitPart5(buf);
                PART10Value=transmitPart10(buf);
            }
        }
    }

    static unsigned long OledTimer=millis();
    // if (mesure_particule && millis() - OledTimer >=1000)
    if (millis() - OledTimer >=1000)
    {
        OledTimer=millis();

        //Serial.print("PM1.0: ");
        //Serial.print(PM01Value);
        // Serial.println(" ug/m3");

        // Serial.print("PM2.5: ");
        // Serial.print(PM2_5Value);
        //Serial.println(" ug/m3");

        //Serial.print("PM1.0: ");
        //Serial.print(PM10Value);
        //Serial.println(" ug/m3");
        //Serial.println();

        Serial.print("Particule 0.3 :");
        Serial.print(PART0_3Value);
        Serial.println();

        Serial.print("Particule 0.5 :");
        Serial.print(PART0_5Value);
        Serial.println();
    }
}

```

```

Serial.print("Particule 1 : ");
Serial.print(PART1Value);
Serial.println();

Serial.print("Particule 2.5 : ");
Serial.print(PART2_5Value);
Serial.println();

Serial.print("Particule 5 : ");
Serial.print(PART5Value);
Serial.println();

Serial.print("Particule 10 : ");
Serial.print(PART10Value);
Serial.println();

Serial.print("Total Particule : ");
total = PART10Value + PART5Value + PART2_5Value + PART1Value + PART0_5Value + PART0_3Value;
Serial.print(total);
Serial.println();
Serial.println("_____");

}

PMSerial.end();
esp8266.begin(115200);
esp8266.listen();
if (total >=0)
{
    return total;
}

}

char checkValue(unsigned char *thebuf, char leng)
{
    char receiveflag=0;
    int receiveSum=0;

    for(int i=0; i<(leng-2); i++){
        receiveSum=receiveSum+thebuf[i];
    }
    receiveSum=receiveSum + 0x42;

    if(receiveSum == ((thebuf[leng-2]<<8)+thebuf[leng-1])) //check the serial data
    {
        receiveSum = 0;
        receiveflag = 1;
    }
    return receiveflag;
}

```

```

int transmitPM01(unsigned char *thebuf)
{
    int PM01Val;
    PM01Val=((thebuf[3]<<8) + thebuf[4]); //count PM1.0 value of the air detector module
    return PM01Val;
}

//transmit PM Value to PC
int transmitPM2_5(unsigned char *thebuf)
{
    int PM2_5Val;
    PM2_5Val=((thebuf[5]<<8) + thebuf[6]); //count PM2.5 value of the air detector module
    return PM2_5Val;
}
//transmit PM Value to PC
int transmitPM10(unsigned char *thebuf)
{
    int PM10Val;
    PM10Val=((thebuf[7]<<8) + thebuf[8]); //count PM10 value of the air detector module
    return PM10Val;
}

//transmit Part Value 0.3
int transmitPart0_3(unsigned char *thebuf)
{
    int PARTVal;
    PARTVal=((thebuf[15]<<8) + thebuf[16]); //count value of the air detector module
    return PARTVal;
}
//transmit Part Value 0.5
int transmitPart0_5(unsigned char *thebuf)
{
    int PARTVal;
    PARTVal=((thebuf[17]<<8) + thebuf[18]); //count value of the air detector module
    return PARTVal;
}
//transmit Part Value 1
int transmitPart1(unsigned char *thebuf)
{
    int PARTVal;
    PARTVal=((thebuf[19]<<8) + thebuf[20]); //count value of the air detector module
    return PARTVal;
}
//transmit Part Value 2.5
int transmitPart2_5(unsigned char *thebuf)
{
    int PARTVal;
    PARTVal=((thebuf[21]<<8) + thebuf[22]); //count value of the air detector module
    return PARTVal;
}

```

```

//transmit Part Value 5
int transmitPart5(unsigned char *thebuf)
{
    int PARTVal;
    PARTVal=((thebuf[23]<<8) + thebuf[24]); //count value of the air detector module
    return PARTVal;
}
//transmit Part Value 10
int transmitPart10(unsigned char *thebuf)
{
    int PARTVal;
    PARTVal=((thebuf[25]<<8) + thebuf[26]); //count value of the air detector module
    return PARTVal;
}

///////////////////////////////qualité de l'air////////////////////

int counter = 0;
int TVOC() {
    // If you have a temperature / humidity sensor, you can set the absolute humidity to
    // enable the humidity compensation for the air quality signals
    float temperature = 22.1; // [°C]
    float humidity = 45.2; // [%RH]
    sgp.setHumidity(getAbsoluteHumidity(temperature, humidity));

    if (! sgp.IAQmeasure()) {
        Serial.println("Measurement failed");
        return;
    }
    Serial.print("TVOC "); Serial.print(sgp.TVOC); Serial.print(" ppb\t");
    Serial.print("eCO2 "); Serial.print(sgp.eCO2); Serial.println(" ppm");

    if (! sgp.IAQmeasureRaw()) {
        Serial.println("Raw Measurement failed");
        return;
    }
    Serial.print("Raw H2 "); Serial.print(sgp.rawH2); Serial.print(" \t");
    Serial.print("Raw Ethanol "); Serial.print(sgp.rawEthanol); Serial.println("");

    delay(1000);

    counter++;
    if (counter == 30) {
        counter = 0;

        uint16_t TVOC_base, eCO2_base;
        if (! sgp.getIAQBaseline(&eCO2_base, &TVOC_base)) {
            Serial.println("Failed to get baseline readings");
            return;
        }
        Serial.print("****Baseline values: eCO2: 0x"); Serial.print(eCO2_base, HEX);
        Serial.print(" & TVOC: 0x"); Serial.println(TVOC_base, HEX);
    }
    int a = sgp.TVOC;
    if(a>=0){
        return a;
    }
}

```

Annexe 12 : Code pour enlever les zones d'erreurs

```
% TODO - Replace the [] with channel ID to write data to:  
writeChannelID = 958620;  
% TODO - Enter the Write API Key between the '' below:  
writeAPIKey = 'RGDE9FBFTEFBC3QD';  
  
temperatureField = 1;  
humiditeField = 2;  
HCHOField = 3;  
CO2Field = 4;  
PMField = 5;  
TVOCField = 6;  
%% Read Data %%  
[%data, time] = thingSpeakRead(readChannelID,'Fields',[1:6],'Numminutes',60);  
[data, time] = thingSpeakRead(readChannelID,'Fields',[1:6], 'NumPoints',100);  
  
% 'analyzedData' variable.  
analyzedData = data;  
  
temperature = analyzedData(:,1);  
humidite = analyzedData(:,2);  
HCHO = analyzedData(:,3);  
CO2 = analyzedData(:,4);  
PM =analyzedData(:,5);  
TVOC = analyzedData(:,6);  
  
for i = 1:6  
    datatrait = analyzedData(:,i);  
    moyennes = mean(datatrait);  
    n = length(datatrait);  
    m=n;  
    time0 = time;  
    for j = 1:n  
        if j>m  
            break;  
        end  
        if datatrait(j)== 0  
            datatrait(j) = [];  
            time0(j)=[];  
            j = j-1;  
            m=m-1;  
        end  
        if datatrait(j)> 10*moyennes  
            datatrait(j) = [];  
            time0(j)=[];  
            j = j-1;  
            m=m-1;  
        end  
    end  
    thingSpeakWrite(writeChannelID,'Fields',[i], 'Values',datatrait, 'WriteKey', writeAPIKey, 'TimeStamp',time0  
    %pause(10)  
end
```

Annexe 13 : Code pour calculer la valeur la plus basse / la plus haute

```
% Read temperature data from a ThingSpeak channel over the past 24  
hours  
% to calculate the high and low temperatures and write to another  
channel.
```

```

% Channel 12397 contains data from the MathWorks Weather Station,
located
% in Natick, Massachusetts. The data is collected once every minute.
Field
% 4 contains temperature data.

% Channel ID to read data from
readChannelID = 980277;
% Temperature Field ID
TemperatureFieldID = 1;
humidityFieldID = 2;
HCHOFieldID = 3;
CO2FieldID = 4;
PMFieldID = 5;
TVOCFieldID = 6;
% Channel Read API Key
% If your channel is private, then enter the read API Key between the
' ' below:
readAPIKey = ' ';

% Read temperature data for the last 24 hours from the MathWorks
weather
% station channel. Learn more about the thingSpeakRead function by
going to
% the Documentation tab on the right side pane of this page.

[temperature,timeStamp] =
thingSpeakRead(readChannelID,'Fields',TemperatureFieldID,
'numDays',1,'ReadKey',readAPIKey);
[CO2,timeStampCO2] =
thingSpeakRead(readChannelID,'Fields',CO2FieldID,
'numDays',1,'ReadKey',readAPIKey);
[PM,timeStampPM] = thingSpeakRead(readChannelID,'Fields',PMFieldID,
'numDays',1,'ReadKey',readAPIKey);

%[temperature,timeStamp] =
thingSpeakRead(readChannelID,'Fields',TemperatureFieldID,
'numDays',1,'ReadKey',readAPIKey);
% Calculate the maximum and minimum temperatures
[maxtemperature,maxTempIndex] = max(temperature);
[mintemperature,minTempIndex] = min(temperature);

[maxCO2,maxCO2Index] = max(CO2);
[minCO2,minCO2Index] = min(CO2);

[maxPM,maxPMIndex] = max(PM);
[minPM,minPMIndex] = min(PM);

% Selection du temps
timeMaxTemp = timeStamp(maxTempIndex);

```

```

timeMinTemp = timeStamp(minTempIndex);

timeMaxCO2= timeStampCO2(maxCO2Index);
timeMinCO2 = timeStampCO2(minCO2Index);

timeMaxPM= timeStampPM(maxPMIndex);
timeMinPM = timeStampPM(minPMIndex);

display(maxtemperature,'Le temperature plus eleve en 24h est');
display(timeMaxTemp,'C est à')
display(mintemperature,'Le temperature plus froid en 24h est');
display(timeMinTemp,'C est à')

display(maxCO2,'Le CO2 plus eleve en 24h est ppm');
display(timeMaxCO2,'C est à')
display(minCO2,'Le CO2 plus bas en 24h est ppm');
display(timeMinCO2,'C est à')

display(maxPM,'Le concentration des particules fines plus eleve en
24h est');
display(timeMaxPM,'C est à')
display(minPM,'Le concentration des particules fines plusbas en 24h
est ppm');
display(timeMinPM,'C est à')

writeChannelID = [];

writeAPIKey = '';

%thingSpeakWrite(writeChannelID,maxtemperature,'timestamp',timeMaxTem
p,'WriteKey',writeAPIKey);
%thingSpeakWrite(writeChannelID,maxCO2,'timestamp',timeMaxCO2,'WriteK
ey',writeAPIKey);
%thingSpeakWrite(writeChannelID,maxPM,'timestamp',timeMaxPM,'WriteKey
',writeAPIKey);

```

Annexe 14 : Code pour calculer les valeurs moyennes

```

% Channel ID to read data from
readChannelID = 980277;
% Fields ID
temperatureFieldID = 1;
humidityFieldID = 2;
HCHOFieldID = 3;
CO2FieldID = 4;
PMFieldID = 5;
TVOCFieldID = 6;

```

```

readAPIKey = '';

temperature =
thingSpeakRead(readChannelID, 'Fields', temperatureFieldID, 'Numminutes'
, 60, 'ReadKey', readAPIKey);

humidity =
thingSpeakRead(readChannelID, 'Fields', humidityFieldID, 'Numminutes', 60
, 'ReadKey', readAPIKey);

HCHO =
thingSpeakRead(readChannelID, 'Fields', HCHOFieldID, 'Numminutes', 60, 'Re
adKey', readAPIKey);

CO2 =
thingSpeakRead(readChannelID, 'Fields', CO2FieldID, 'Numminutes', 60, 'Rea
dKey', readAPIKey);

PM =
thingSpeakRead(readChannelID, 'Fields', PMFieldID, 'Numminutes', 60, 'Rea
d Key', readAPIKey);

TVOC =
thingSpeakRead(readChannelID, 'Fields', TVOCFieldID, 'Numminutes', 60, 'Re
adKey', readAPIKey);

% Calculer les valeurs moyennes
moyTemperature = mean(temperature);
display(moyTemperature, 'Temperature moyenne');

moyHumidity = mean(humidity);
display(moyHumidity, 'Humidite moyenne');

moyHCHO = mean(HCHO);
display(moyHCHO, 'HCHO moyenne');

moyCO2 = mean(CO2);
display(moyCO2, 'CO2 moyenne');

moyPM = mean(PM);
display(moyPM, 'Particules fines moyenne');

```

```
moyTVOC = mean(TVOC);
display(moyTVOC, 'TVOC moyenne');

% les infos de channel enregistré
writeChannelID = 980313;
writeAPIKey = 'JP4P21BFOURZQ1Y1';

% Learn more about the THINGSPEAKWRITE function by going to the
Documentation tab on
% the right side pane of this page.

thingSpeakWrite(writeChannelID, {moyTemperature, moyHumidity, moyHCHO, moyCO2, moyPM, moyTVOC}, 'WriteKey', writeAPIKey);
```

BIBLIOGRAPHIE

- [1] Françoise NOARS, (2015). La pollution de l'air, c'est quoi ? Tiré de :
<https://www.netatmo.com/fr-fr/aircare/homecoach>
- [2] Frédéric Potter, Balise netatmo, Vivez dans une maison plus saine. Tiré de :
<https://www.netatmo.com/fr-fr/aircare/homecoach>
- [3] Eoletec, (2016). UNE STATION DE DIAGNOSTIC QUI MESURE LA QUALITÉ DE L'AIR DE VOTRE HABITAT. Tiré de :<http://www.eoletec.fr/produits/eolesens/>
- [4] ETHERA SA. Analyseur de la qualité de l'air intérieur NEMo. Tiré de :
<http://www.etheralabs.com/analyseur-qualite-air-interieur-nemo/>
- [5] Boulanger. Activateur et analyseur de sommeil Sevenhugs HugOne. Tiré de :
<https://www.boulanger.com/ref/1056307>
- [6] Foobot Airboxlab. SA. (2010). Better Air, Better Life. Tiré de :<https://foobot.io/>
- [7] Eve home. Prenez soin de votre air. Tiré de :<https://www.evehome.com/fr/eve-room>
- [8] Airthings. Présentation de Wave. Tiré de :<https://www.airthings.com/fr/wave>
- [9] Airthings. Wave Plus. Tiré de :<https://www.airthings.com/fr/wave-plus>
- [10] IGERESS. Moniteur de qualité de l'air IGERESS détecteur de pollution de l'air intérieur pour formaldéhyde, VOC, PM2.5, PM1.0, PM10, température, d'humidité, testeur d'air en temps réel avec écran LCD coloré. Tiré de :
<https://www.amazon.fr/surveillance-IGERESS-Formald%C3%A9hyde-temp%C3%A9rature-Eregistrement/dp/B07B8LNGFT>
- [11] Blueair. (2020). Qu'est-ce que de l'air pur ? Blueair purifie votre air intérieur. Tiré de :
<https://www2.blueair.com/fr/clean-air>
- [12] WICKED DEVICE LLC.(2018).The Egg. Tiré de :<https://airqualityegg.com/egg>
- [13] Roger Genet, Valeurs Guides de qualité d'Air Intérieur (VGAI). Tiré de:
<https://www.anses.fr/fr/content/valeurs-guides-de-qualit%C3%A9-d%E2%80%99air-int%C3%A9rieur-vgai>
- [14] Marie YANOWITZ-DURAND (2015), Qualité de l'air : Sources de pollution et effets sur la santé. Tiré de :

<https://solidarites-sante.gouv.fr/sante-et-environnement/air-exterieur/qualite-de-l-air-exterieur-10984/article/qualite-de-l-air-sources-de-pollution-et-effets-sur-la-sante>

[15] Roger Genet. Qualité de l'air intérieur. Tiré de :
<https://www.ecologique-solidaire.gouv.fr/qualite-lair-interieur>.

[16] Rodolphe Vincent.(2018). Qualité de l'air intérieur. Tiré de:
<https://www.anses.fr/fr/content/qualit%C3%A9-de-l%20air-int%C3%A9rieur>

[17] ADEME. (2019). L'air intérieur du logement . Tiré de :
<https://www.ademe.fr/particuliers-eco-citoyens/habitation/bien-gerer-habitat/lair-interieur-logement>

[18] ADEME. (2019). Impact des produits d'entretien sur la qualité de l'air intérieur. Tiré de :
<https://www.ademe.fr/impact-produits-dentretien-qualite-lair-interieur>

[19] ADEME.(2015).Les chiffres en air intérieur. Tiré de :
<https://www.ademe.fr/expertises/air-bruit/chiffres-cles-observations/chiffres-air-interieur>

[20] Murprotec.(2019). 3 Astuces pour améliorer la qualité de l'air dans une maison. Tiré de :
<https://news.murprotec.be/fr/ameliorer-la-qualite-de-l-air-dans-une-maison>

[21] Conseils Thermiques, (2016). Le confort thermique. Tiré de :
<https://conseils-thermiques.org/contenu/confort-thermique.php>

[22] Delphine Waquier., (2017). L'hypothermie, les causes, les symptômes. Tiré de :
https://www.passeportsante.net/fr/Maux/Problemes/Fiche.aspx?doc=hypothermie_pm

[23] Alexandra Besson, (2018). Quels sont les risques liés à des températures élevées ? Tiré de
<https://www.santemagazine.fr/actualites/quels-sont-les-risques-lies-a-des-temperatures-elevees-302398>

[24] https://www.passeportsante.net/fr/Maux/Problemes/Fiche.aspx?doc=hypothermie_pm

[25] (2016). Dioxyde de carbone. Tiré de :
<https://www.respire-asso.org/dioxyde-de-carbone-co2/>

[26] Openhardware Algerie, (2016). Internet of Things avec Arduino Partie 3 : envoyer des données en ligne avec Thingspeak. Tiré de :
<https://www.youtube.com/watch?v=5pdSJkjW8Dw>

[27] (2016). Tiré de :
https://github.com/OpenHDZ/thingspeak_arduino_yun/blob/master/ts_test_dht11/ts_test_dht11.ino

[28] Mohamed AIT MANSOUR. ARDUINO ISIS : COMMENT TRANSFÉRER LES DONNÉES DE LA CARTE ARDUINO VERS ISIS ? Tiré de :
<https://www.electronique-mixte.fr/proteus-isis/arduino-isis-comment-transferer-les-donnees-de-la-carte-arduino-vers-isis/>

[29] IONOS SARL, (2019). Bases de données : pourquoi en avez-vous besoin et quels sont leurs types ? Tiré de :

<https://www.ionos.fr/digitalguide/hebergement/aspects-techniques/base-de-donnees/>

[30] L'équipe de wikiHow. Comment créer une base de données ? Tiré de :

<https://fr.wikihow.com/cr%C3%A9er-une-base-de-donn%C3%A9es-MySQL>

[31] SHT1x Humidity and Temperature Sensor. Tiré de :

https://wiki.dfrobot.com/SHT1x_Humidity_and_Temperature_Sensor_SKU_DFR0066

[32] Gravity HCHO Sensor. Tiré de :

https://wiki.dfrobot.com/Gravity__HCHO_Sensor_SKU__SEN0231

[33] Gravity Analog Infrared CO2 Sensor. Tiré de :

https://wiki.dfrobot.com/Gravity__Analog_Infrared_CO2_Sensor_For_Arduino_SKU_SEN0219

[34] PM2.5 laser dust sensor. Tiré de :

https://wiki.dfrobot.com/PM2.5_laser_dust_sensor_SKU_SEN0177

[35] Adafruit. Arduino Test. Tiré de :

<https://learn.adafruit.com/adafruit-sgp30-gas-tvoc-eco2-mox-sensor/arduino-code>

[36] ESP8266, module WIFI. Tiré de:

<https://medium.com/@cgrant/using-the-esp8266-wifi-module-with-arduino-uno-publishing-to-thingspeak-99fc77122e82>

[37] Alexandre Pailhoux. Arduino Ep.16 - Installation du module Wifi ESP8266. Tiré de :

<https://www.les-electroniciens.com/videos/arduino-ep16-installation-du-module-wifi-esp8266>

[38] Openhardware Algerie, (2016). Internet of Things avec Arduino Partie 3 : envoyer des données en ligne avec Thingspeak. Tiré de :

<https://www.youtube.com/watch?v=5pdSJkjW8Dw>

[39] IFTTT et ThingSpeak: Affichage sur portable. Tiré de :

<https://fr.mathworks.com/help/thingspeak/use-ifttt-to-send-text-message-notification.html>

[40] The MathWorks, Inc.(2020). ThingSpeak for IoT Projects. Tiré de :

<https://thingspeak.com/>