Table des matières

Table des figures

Liste des tableaux

Introduction générale

Depuis des millénaires, les abeilles ont été essentielles pour la culture humaine et la diversité des plantes. Malheureusement, ces insectes utiles sont de plus en plus menacés dans le monde entier pour diverses raisons. Il est crucial de mieux comprendre et de protéger les abeilles et l'apiculture de manière durable.

L'apiculture, une pratique ancienne, s'est modernisée grâce aux avancées technologiques. Notre projet se concentre sur la conception d'une ruche connectée, alliant tradition et innovation pour répondre aux besoins actuels des apiculteurs. Nous expliquons ici les bases de notre projet, son contexte historique et les défis actuels auxquels font face les apiculteurs.

Au-delà d'une simple tradition, l'apiculture est aujourd'hui essentielle pour préserver les écosystèmes et assurer la production alimentaire. Cependant, les apiculteurs rencontrent des défis croissants comme la diminution des populations d'abeilles et les problèmes environnementaux. Les technologies de l'Internet des objets offrent de nouvelles opportunités pour surveiller et soutenir les abeilles de manière efficace et durable.

Notre projet vise à concevoir une ruche connectée avec des capteurs environnementaux et une interface utilisateur simple. Grâce à ces technologies, les apiculteurs pourront surveiller en temps réel les conditions de la ruche, détecter les problèmes potentiels et agir rapidement pour garantir la santé des abeilles.

Ce rapport est divisé en trois chapitres, à travers lesquels nous décrivons le travail effectué pour la conception de notre système :

Le premier chapitre sera consacré à la présentation du contexte, la problématique, les ruches d'abeilles et leurs types et on fini par la solution proposé et le cahier de charge.

Le deuxième chapitre sera dédié à l'étude du choix de matériels, la conception et la technologie innovante intégrée pour notre solution proposée.

Au troisième chapitre, nous allons présenter d'une part les logiciels utilisés lors de ce projet, et d'autre part la partie programmation et l'utilisation de l'outil Node-RED.

Chapitre 1

Généralités et cadre du projet

Introduction

Ce chapitre se concentre sur l'analyse du contexte et des défis rencontrés dans la gestion des ruches d'abeilles, ainsi que sur la présentation de notre solution innovante. Nous commencerons par examiner le contexte général et la problématique qui sous-tend notre étude. Ensuite, nous détaillerons les phases critiques de la gestion des ruches d'abeilles que nous avons identifiées comme nécessitant une amélioration. Enfin, nous proposerons notre solution en décrivant les exigences spécifiques définies dans notre cahier des charges. Cette analyse nous permettra de poser les bases nécessaires pour aborder les aspects pratiques et techniques de notre projet dans les chapitres suivants.

1.1 Contexte

Le contexte de l'apiculture actuel est confronté à d'importants défis, notamment le déclin alarmant des populations d'abeilles en raison du changement climatique, des pesticides, des maladies et de la perte d'habitats naturels. Les apiculteurs doivent moderniser leurs pratiques de gestion des ruches pour assurer la survie des abeilles et maintenir une production de miel stable. Actuellement, l'inspection manuelle des ruches

est difficile et peu précise. Une surveillance en temps réel est nécessaire pour détecter les anomalies et protéger les colonies d'abeilles. Notre projet de ruche connectée vise à répondre à ces besoins en utilisant des technologies IoT avancées pour offrir aux apiculteurs une solution moderne, automatisée et intelligente, permettant une surveillance constante, une collecte de données précises, des alertes en temps réel et une gestion à distance, afin de préserver les abeilles et d'optimiser la production de miel dans un environnement apicole en évolution rapide.

1.2 Problématique

Les abeilles, jouant un rôle essentiel dans l'équilibre écologique, font actuellement face à des menaces sérieuses, notamment le développement humain, l'utilisation de pesticides, les maladies et le changement climatique, entraînant une diminution alarmante de leurs populations. Face à ces défis pressants, les apiculteurs se trouvent contraints de mettre en place une surveillance régulière et systématique de leurs ruches, non seulement pour suivre la production de miel, mais également pour détecter toute anomalie potentielle.

Cependant, l'inspection manuelle des ruches représente un défi majeur. Les déplacements fréquents dans les champs, nécessaires à un contrôle efficace, entraînent une perte de temps et d'efforts considérable. De plus, la précision des mesures météorologiques, telles que la température et l'humidité, est compromise en raison de divers paramètres perturbateurs.

La gestion d'un grand nombre de ruches entraîne également une perte significative de temps lors de la collecte des mesures. De plus, l'organisation manuelle des informations provenant des ruches rend les études visant à identifier les zones propices à une production optimale de miel difficiles et imprécises.

Notre ambitieux projet aspire à concevoir et déployer une ruche connectée intelligente, représentant une solution novatrice destinée à révolutionner la gestion des colonies d'abeilles. Confrontés aux défis complexes actuels de l'apiculture, notre ruche

intelligente s'engage à intégrer des technologies de pointe afin d'optimiser la surveillance des ruches, alléger les contraintes opérationnelles pesant sur les apiculteurs, et jouer un rôle essentiel dans la préservation des abeilles, ces précieux pollinisateurs de notre écosystème.

1.3 Les ruches d'abeilles

Avant de rédiger notre cahier des charges et de proposer notre solution, nous avons d'abord pris le temps de définir ce qu'est une ruche d'abeilles ainsi que les différents types de ruches existants. Cette étape préliminaire nous a permis d'acquérir une compréhension approfondie de la structure et du fonctionnement des ruches, ce qui est essentiel pour concevoir une solution adaptée et efficace pour notre projet.

La ruche est une structure presque fermée abritant une colonie d'abeilles. L'intérieur de la ruche est composé de rayons formés par des cellules hexagonales de cire d'abeille. Les abeilles utilisent ces cellules pour le stockage de la nourriture (miel et pollen), et pour le renouvellement de la population (œufs, larves et nymphes). L'enruchage ou l'enruchement est l'action de peupler une ruche d'abeilles. Un groupe de ruches est un rucher.

1.4 Solution proposée

Pour développer une ruche connectée innovante, nous avons conçu une solution complète et intelligente qui intègre des technologies de pointe. Voici les caractéristiques principales de notre solution :

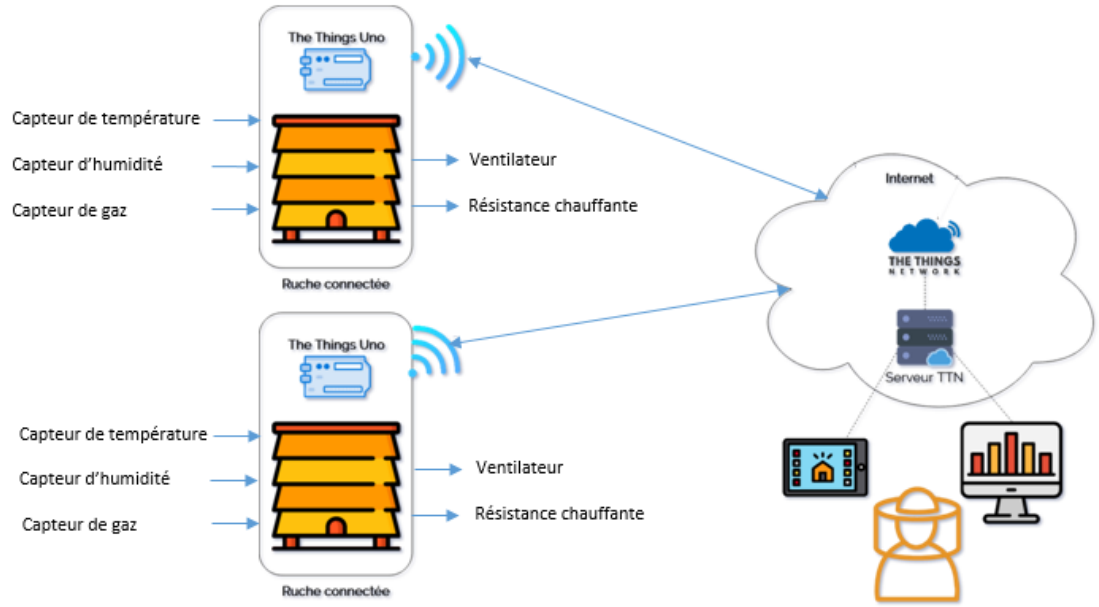


FIGURE 1.1 – Synoptique de la solution proposée

- **Capteurs Intelligents :**

  Capteur de Température et d'Humidité : Placé stratégiquement dans la ruche pour surveiller l'environnement interne.

  Capteur de Gaz : Détection précise du monoxyde de carbone pour assurer la sécurité des abeilles.

- **Actionneurs :**

  Ventilateur : Contrôle automatique de la ventilation pour réguler la température et l'humidité à l'intérieur de la ruche, assurant un environnement confortable pour les abeilles.

  Résistance Chauffante : Activation automatique pour maintenir une température optimale dans la ruche par temps froid, protégeant ainsi les abeilles contre les températures extrêmes.

- **Système de Communication IoT :**

  Communication sans Fil : Transfert des données vers une plateforme cloud centralisée.

- **Plateforme Cloud et Application Mobile :**

  Stockage Centralisé : Conservation sécurisée des données pour une analyse

ultérieure.

Interface Utilisateur Conviviale : Application mobile permettant aux apiculteurs de visualiser les données, de recevoir des alertes et de prendre des décisions informées.

- **Coût Abordable :**

  Choix de Composants Économiques : Sélection de composants offrant un excellent rapport qualité-prix.

- **Installation Facile dans les Ruches :**

  Conception Ergonomique : Intégration discrète dans la ruche avec un minimum de perturbation pour les abeilles.

  En intégrant ces éléments, notre solution de ruche connectée ambitionne de révolutionner la gestion apicole en offrant une surveillance précise, des alertes en temps réel, une facilité d'utilisation et une autonomie énergétique. Elle vise à préserver la santé des abeilles et à optimiser la production de miel dans le contexte apicole en constante évolution.

1.5 Cahier de charge

Pour que le système soit efficace et bien adapté aux besoins de nos apiculteurs, il est nécessaire qu'il obéi à un certain nombre de critères, d'exigences et de contraintes. Ces différents points doivent être le résultat d'une réflexion profonde qui prennent en compte :

- Les besoins spécifiques de l'apiculteur
- Le système doit être concurrentiel
- La nature du lieu où le système doit être implanté
- Les effets nuisibles du système sur le ruché
- Les bénéfices escomptés du système
- L'efficacité du système
- L'estimation du prix du système

Pour la réalisation de notre système nous avons fixé un certain nombre d'exigence et de contraintes qui peuvent être énuméré comme suit :

- Le système doit fonctionner de manière autonome
- Le système doit être reconfigurable
- Le système doit être facilement intégrables dans les ruches
- Faible consommation d'énergie
- Un coût réduit pour la réalisation du système

Le système est censé mesurer les paramètres suivants :

- La température dans la ruche
- L'humidité sous la couverture de la ruche
- La concentration de monoxyde de carbone
- En plus, il doit fournir des données de diagnostic.

Conclusion

Tout au long de ce chapitre, nous avons exposé le contexte ainsi que la problématique à partir de laquelle nous avons détaillé la phase des ruches d'abeilles. Ensuite, nous avons présenté notre solution en décrivant le cahier des charges.

Dans le chapitre suivant, nous aborderons le choix du matériel, la conception électrique et la technologie utilisée.

Chapitre 2

Choix du matériel et stratégie IoT pour la ruche connectée

Introduction

Dans ce chapitre, nous détaillons le choix du matériel pour notre projet de ruche connectée, sa conception électrique, ainsi que la mise en place de la communication IoT. Ces éléments sont indispensables pour concrétiser notre solution innovante, en fournissant les bases solides nécessaires à son développement et à son fonctionnement efficace.

2.1 Choix du matériel

Dans notre projet, nous avons utilisé divers matériels cruciaux pour concevoir et déployer notre solution IoT pour la ruche connectée.

2.1.1 Carte de dévloppement : ESP32

Nous avons sélectionné la carte de développement ESP32 WROOM OLED en raison de sa grande popularité dans les projets IoT. Cette carte est dotée du microcontrôleur ESP32 à double cœur intégrant le Wifi et le Bluetooth, ainsi qu'un écran OLED de 0,96

pouce pour afficher en temps réel les données. Sa taille compacte en fait un choix idéal pour les projets nécessitant une empreinte réduite, et sa faible consommation d'énergie est adaptée aux applications sur batterie. La carte ESP32 WROOM OLED est largement utilisée dans des domaines tels que la domotique, la surveillance environnementale et les dispositifs portables.

Et leur caractéristique illustrée par le tableau ci-dessous :

TABLE 2.1 – Caractéristiques de la carte : ESP32

Caractéristique	**Description**
Microcontrôleur	ESP32 (dual-core)
Connectivité sans fil	WiFi (802.11 b/g/n) et Bluetooth intégrés
Affichage	Écran OLED 0.96 pouce
Résolution de l'écran	Variable (généralement 128x64 pixels)
Interface d'affichage	SPI
Capacité de stockage	Flash intégré (jusqu'à plusieurs mégaoctets)
Mémoire RAM	Variable (généralement entre 520KB et 4MB)
Interfaces d'E/S	GPIO, I2C, SPI, UART, ADC, DAC, etc.
Alimentation	3.3V DC (peut varier selon le modèle)/manuel
Dimensions	Variables, mais généralement compactes
Langages de programmation	Supporte Arduino IDE, MicroPython, etc.
Compatibilité avec IDE	Arduino IDE, PlatformIO, etc.
Systèmes d'exploitation	FreeRTOS, etc.

Vous trouverez le datasheet de l'ESP32 WROOM OLED dans la figure suivante :

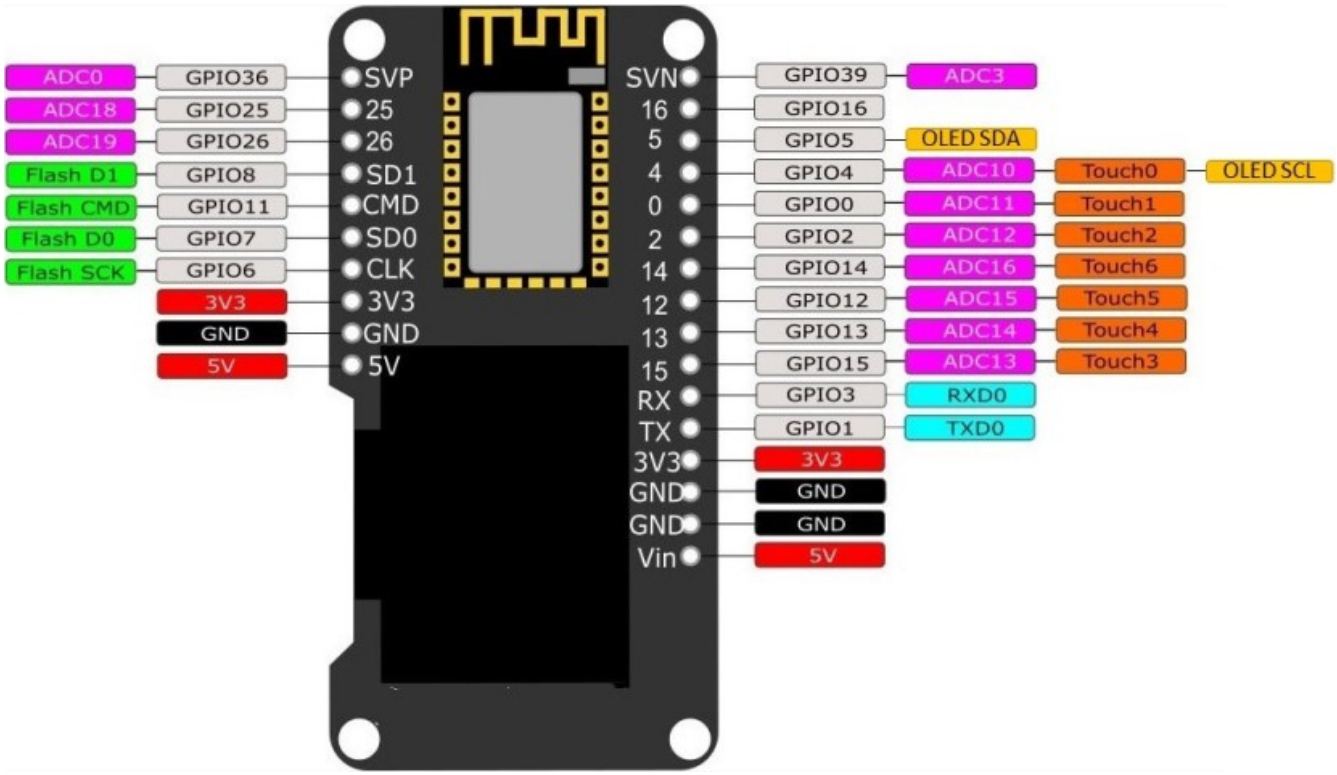


FIGURE 2.1 – ESP32 Wroom Oled

2.1.2 Le capteur de température et d'humudité : DHT22

Nous avons choisi le capteur DHT22 pour son aptitude à mesurer avec précision la température et l'humidité, répondant ainsi parfaitement à nos besoins. Sa fiabilité dans les mesures était essentielle pour garantir des données fiables. De plus, sa compatibilité avec notre plateforme de développement préférée et la disponibilité de bibliothèques de code bien établies ont facilité son intégration. En résumé, le DHT22 était l'option optimale pour notre projet, offrant une solution robuste et efficace pour la surveillance de ces paramètres environnementaux.

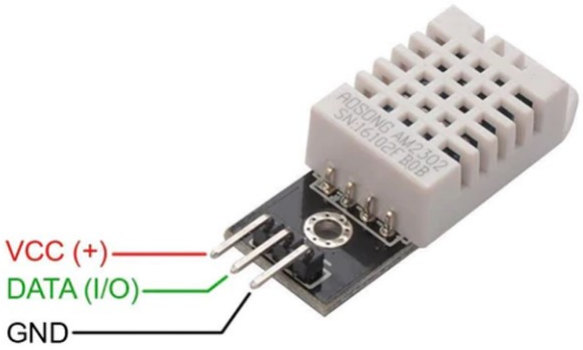


FIGURE 2.2 – Le brochage du capteur de température et d'humudité : DHT22

Ce tableau fournit une vue d'ensemble des spécifications et caractéristiques importantes du capteur DHT22.

TABLE 2.2 – Carectéristiques du capteur de température et d'humudité : DHT22

Caractéristique	**Description**
Type de capteur	Capteur de température et d'humidité
Plage de température	Température : -40°C à 80°C
Plage d'humidité	Humidité : 0% à 100% RH
Précision	Température ±0.5°C Humidité : ±2% RH
Temps de réponse	< 2 secondes
Alimentation	3.3V à 5V DC
Consommation d'énergie	1.5 mA (typique)
Interface de communication	Numérique (1-Wire ou bus série)
Résolution	Température : 0.1°C Humidité : 0.1% RH
Plage de tension	3V à 5.5V
Dimensions	15.1 mm x 25.1 mm x 7.7 mm (L x l x H)
Humidité de fonctionnement	0% à 100% RH
Compatibilité avec IDE	Arduino IDE, PlatformIO, etc.

2.1.3 Capteur de gaz : MQ7

Ce capteur de gaz est spécialement conçu pour détecter le monoxyde de carbone (CO), ce qui est crucial pour assurer la sécurité de notre ruche. C'est un capteur robuste et fiable, sa sensibilité élevée et sa réponse rapide en font un choix idéal pour notre application et son faible coût et sa facilité d'intégration avec notre système de contrôle et de surveillance ont également été des facteurs déterminants dans notre décision de l'incorporer dans notre dispositif. De plus, le MQ7 offre une grande plage de détection et une résolution suffisante pour répondre à nos besoins spécifiques.

FIGURE 2.3 – Capteur de gaz : MQ7

Ce tableau présenter des spécifications du capteur de gaz MQ7 et des informations utiles pour son intégration dans notre projet pour la surveillance de la qualité de l'air.

TABLE 2.3 – Carectéristiques du capteur de gaz : MQ7

Caractéristique	Description
Type de capteur	Capteur de gaz
Gaz détecté	Monoxyde de carbone (CO)
Plage de mesure	10 à 1000 ppm (parties par million)
Sensibilité	200 à 10000 ppm (selon la concentration en CO)
Temps de réponse	$<$ 10 secondes
Temps de préchauffage	24 à 48 heures
Température	-10°C à 50°C
Tension d'alimentation	5V DC
Courant	$<$ 150 mA (typique)
Sortie analogique	Tension proportionnelle à la concentration en CO
Sortie numérique	Signal numérique (TTL compatible)
Durée de vie	$>$ 5 ans
Dimensions	Variables, généralement compactes
Fiabilité	Haute

2.2 Schéma électrique de la ruche intelligente

Pour la mise en place de notre ruche connectée, nous avons suivi un schéma électrique détaillé. Dans ce schéma, nous avons commencé par alimenter le capteur de gaz MQ7 ainsi que le capteur de température et d'humidité, en veillant à respecter scrupuleusement les spécifications techniques fournies dans leurs fiches respectives. Ces capteurs ont été reliés à une source d'alimentation appropriée, tenant compte des besoins en tension et en courant pour assurer leur bon fonctionnement.

Ensuite, nous avons intégré une résistance chauffante et un relais pour contrôler le

ventilateur. La résistance chauffante a été connectée au relais de manière à ce que le microcontrôleur puisse activer ou désactiver la chauffe en fonction des données captées par les capteurs. De même, le ventilateur a été associé au relais pour permettre au microcontrôleur de réguler son fonctionnement en réponse aux conditions environnementales mesurées. Cette configuration globale, combinant capteurs, actionneurs (résistance chauffante et ventilateur) et microcontrôleur, a été planifiée pour garantir une gestion efficace de l'environnement interne de la ruche, facilitant ainsi la collecte et l'analyse des données pour une surveillance optimale.

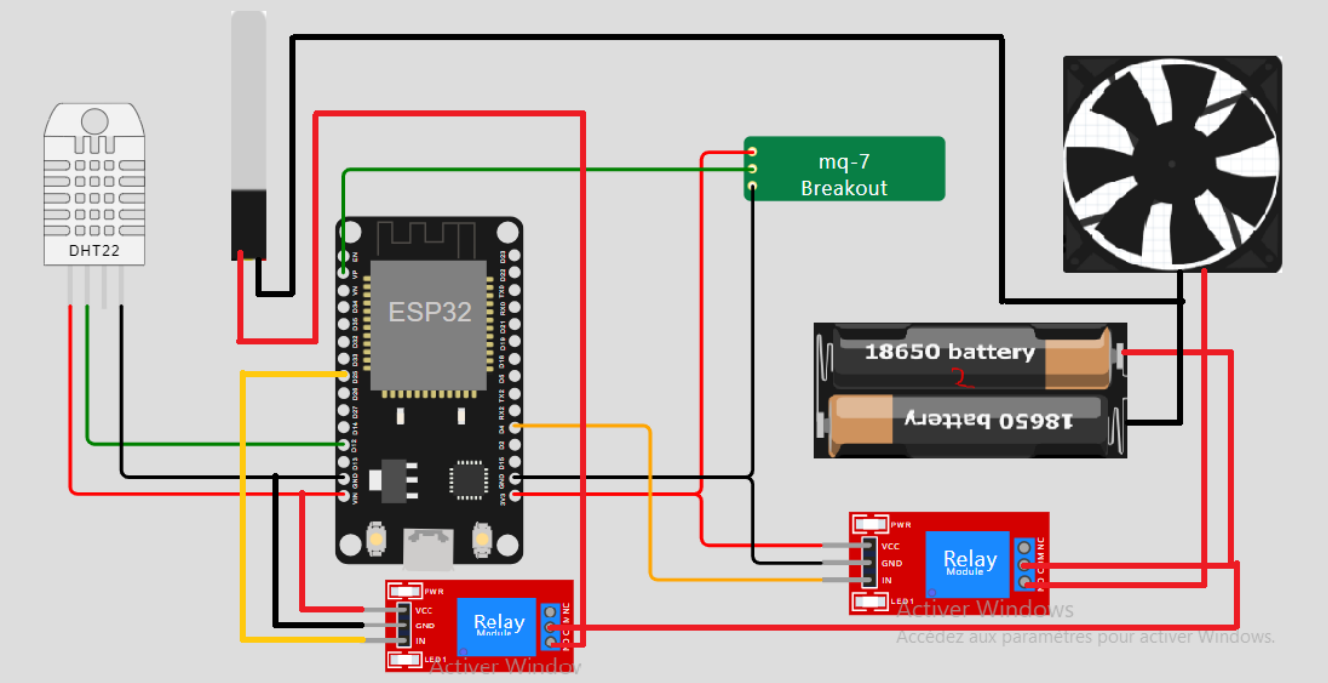


FIGURE 2.4 – Montage électrique de la ruche intelligente

2.3 IoT dans la Ruche Connectée

Dans le cadre de notre projet de ruche connectée, l'Internet des Objets (IoT) est essentiel pour surveiller et contrôler la ruche à distance et comment cette technologie peut être appliquée de manière pratique pour améliorer la santé et la productivité des colonies d'abeilles. Les capteurs intégrés à la ruche mesurent des paramètres tels que la température, l'humidité et la qualité de l'air. Ces capteurs transmettent ensuite les

données collectées à un dispositif de traitement ou à une passerelle IoT, qui les envoie à un serveur distant via Internet. Grâce à l'IoT, les apiculteurs ont une vision en temps réel de l'état de la ruche et des activités des abeilles, ce qui facilite la gestion et la surveillance des colonies.

2.3.1 Architecture IoT

L'architecture IoT mise en œuvre pour notre projet de ruche connectée repose sur un serveur distant hébergé dans le cloud. Ce choix stratégique garantit une scalabilité et une disponibilité optimales pour gérer efficacement les données des capteurs collectées par les passerelles ESP32 déployées dans chaque ruche. La connexion entre les passerelles IoT et le serveur distant est assurée par une connexion Wi-Fi, permettant ainsi la transmission sécurisée des données prétraitées. Cette infrastructure cloud offre également une sécurité avancée avec le chiffrement des données et les contrôles d'accès, tout en simplifiant la gestion grâce à des fonctionnalités automatisées de sauvegarde et de mise à jour. En intégrant notre architecture IoT à un service cloud de premier plan, nous nous assurons de disposer d'une solution fiable, évolutive et rentable pour surveiller et analyser en temps réel les informations vitales provenant de nos ruches connectées.

2.3.2 Protocole MQTT dans la Ruche Connectée

Nous utilisons le protocole MQTT pour échanger des données entre les capteurs de la ruche et le serveur distant. Chaque capteur publie des données sur des sujets MQTT spécifiques, tels que "température", "humidité", etc. Le serveur ou la passerelle IoT s'abonne à ces sujets pour recevoir les données publiées par les capteurs. Cette architecture de publication/abonnement (pub/sub) de MQTT permet une communication asynchrone efficace entre les différents composants de notre système de ruche connectée.

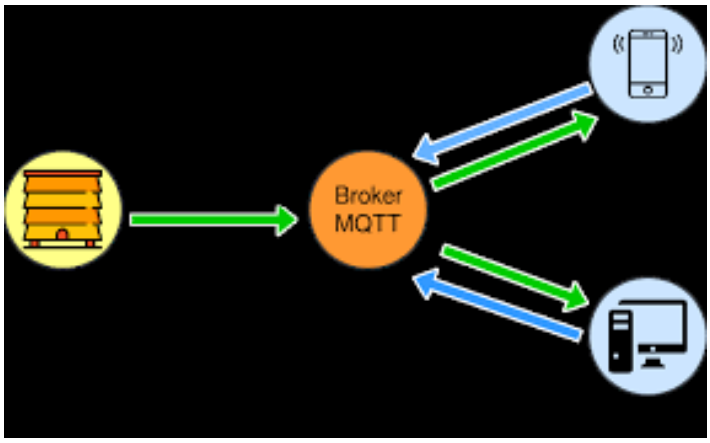


FIGURE 2.5 – Protocole MQTT

2.3.3 Rôle de l'IoT et de MQTT dans le Projet de Ruche Connectée

En résumé, l'Internet des Objets (IoT) et le protocole MQTT jouent un rôle crucial dans notre projet de ruche connectée d'abeilles. Ils permettent la collecte, la transmission et l'analyse des données générées par la ruche, offrant ainsi aux apiculteurs des informations précieuses pour la gestion et la surveillance des colonies d'abeilles. Cette combinaison de technologies modernes contribue à améliorer l'efficacité et la durabilité de l'apiculture en fournissant un accès en temps réel aux données environnementales des ruches.

Conclusion

En conclusion, le choix du matériel, la conception et l'intégration des technologies IoT sont des étapes cruciales dans le développement de notre solution pour la ruche connectée. L'utilisation de cartes comme l'ESP32 WROOM OLED, associée à des capteurs tels que le DHT22 et le MQ7, nous permet de surveiller efficacement les conditions environnementales et de garantir la sécurité des abeilles. L'architecture IoT mise en place, avec le protocole MQTT pour la communication des données, offre une plateforme robuste pour la gestion à distance des ruches.

Chapitre 3

Présentation des outils et processus de développement

Introduction

Dans ce chapitre, nous allons d'abord présenter les logiciels utilisés, puis aborder la partie programmation. Enfin, nous détaillerons l'utilisation de l'outil Node-RED.

3.1 Logiciels utilisés

Dans notre projet, nous avons travaillé sur de nombreux logiciels que nous traitons un par un.

3.1.1 Outil de rédaction de rapport : Latex

Nous avons choisi comme éditeur de texte le logiciel "Latex" car il nous offre de nombreuses fonctionnalités et nous permet une typographie de haute qualité et un rapport professionnelle.

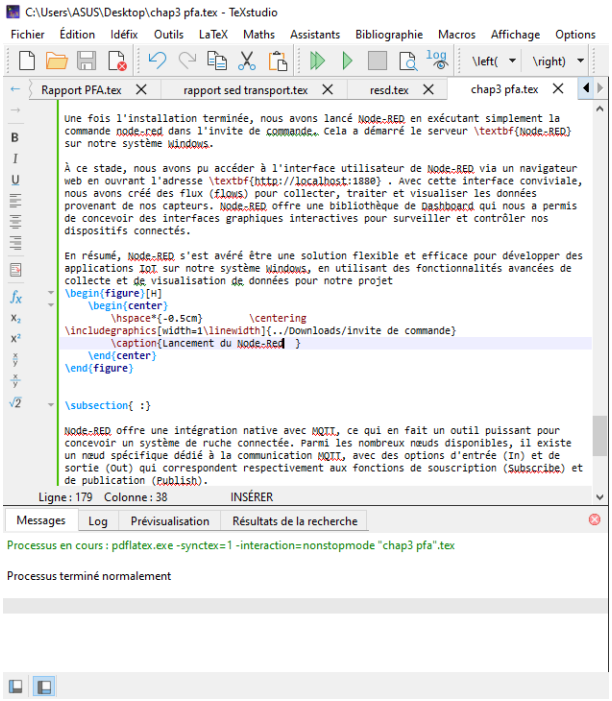


FIGURE 3.1 – Interface de Latex

3.1.2 outils de planification et de suivi : Github

C'est un service web que nous avons utilisé pour la centralisation et la gestion de notre rapport puisqu'il nous garantit la sauvegarde des versions que nous avons traité précédemment et le gain en temps à travers la répartition des tâches.

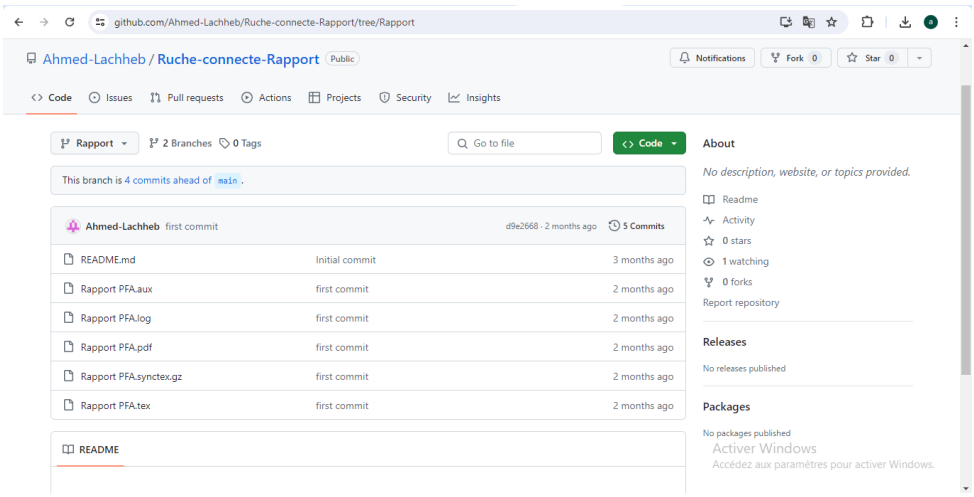


FIGURE 3.2 – Platforme Github

3.1.3 Outil de création de diagrammes et d'organigrammes : Lucidchart

Nous avons recours à "Lucidchart" pour la réalisation des organigramme des capteurs utilisé.

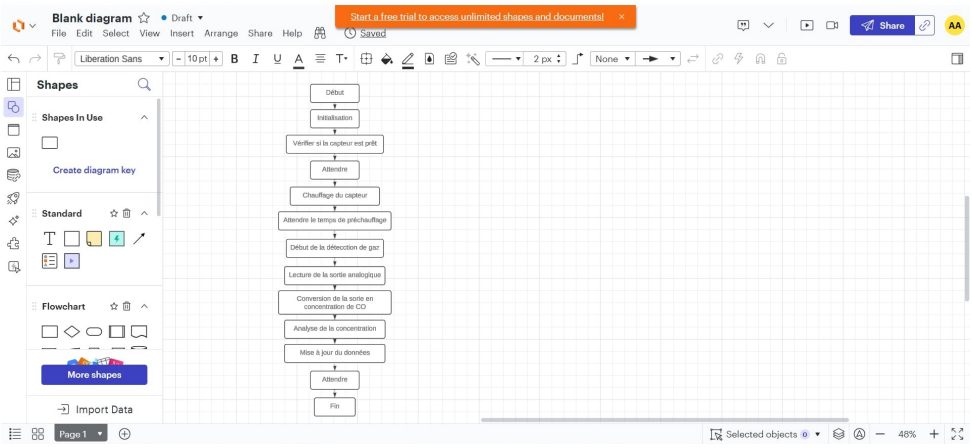


FIGURE 3.3 – Platforme Lucidchart

3.1.4 Arduino IDE

Le logiciel de programmation Arduino, également connu sous le nom d'IDE (Integrated Development Environment), est utilisé pour écrire et éditer du code dans un langage de programmation proche du C. Une fois que le programme est saisi ou modifié dans l'IDE Arduino, il peut être transféré et téléchargé sur la carte Arduino via une connexion USB. Ce câble USB sert à la fois à alimenter la carte en énergie et à transférer le programme vers la carte.

L'IDE Arduino offre une interface de programmation flexible et conviviale qui peut être exécutée sur divers systèmes d'exploitation tels que Windows, macOS et Linux. Ce logiciel est spécialement conçu pour simplifier le processus de développement sur les cartes Arduino en utilisant le langage de programmation C/C++.

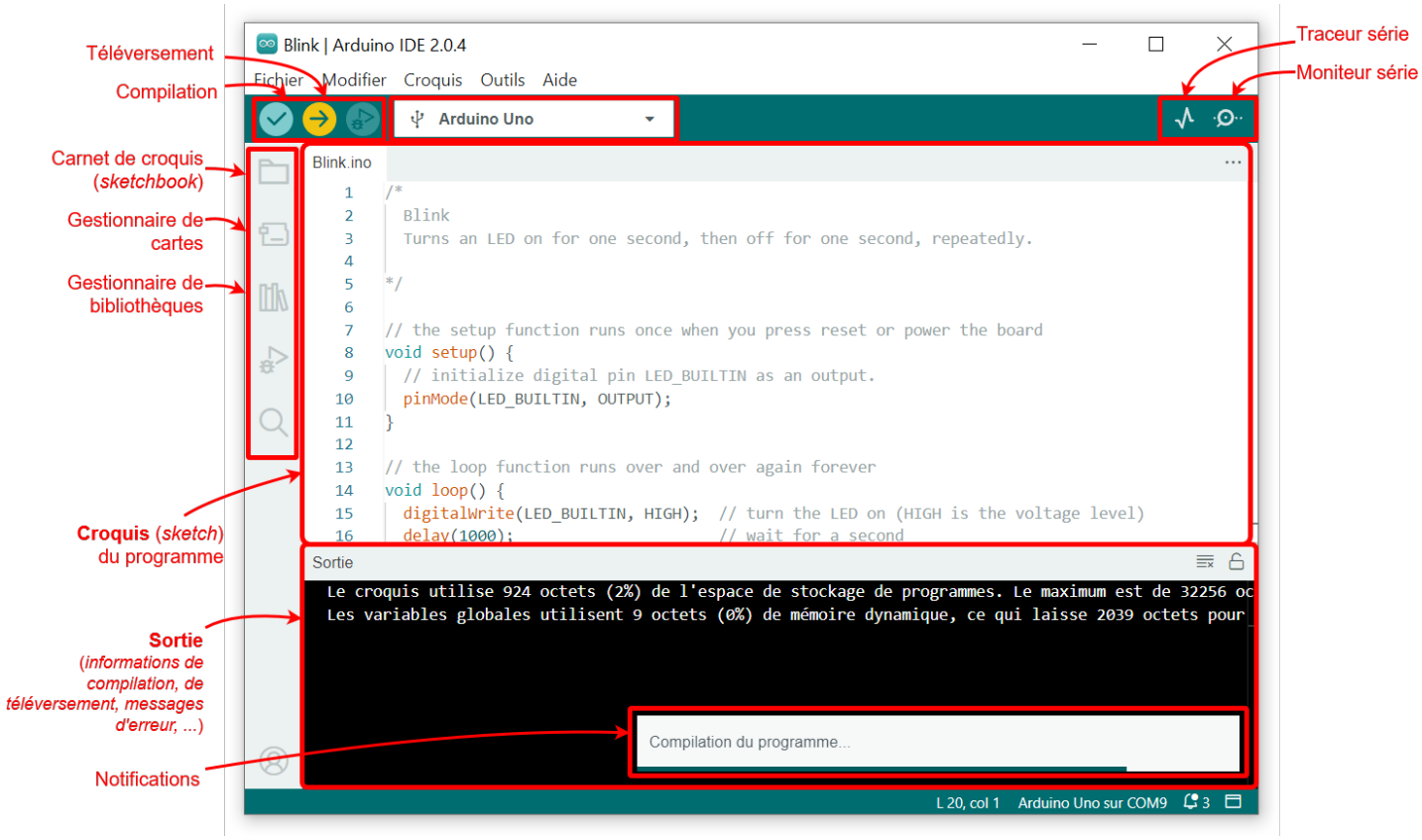


FIGURE 3.4 – Platforme Arduino IDE

3.1.5 Node-Red

Node-RED est un outil puissant qui peut être déployé sur divers systèmes d'exploitation tels que Windows, Linux ou même dans un conteneur Docker pour une utilisation locale. Dans notre projet, nous avons choisi d'installer Node-RED sur un système Windows pour exploiter ses capacités de collecte de données à partir de capteurs connectés.

3.2 Le code principal

La carte ESP32 doit être capable de :

- Se connecter au réseau wifi.

- Se connecter au serveur MQTT avec un ID unique.

- Collecter les données délivrer par les capteurs.

- Envoyer les données collectées vers le tableau de bord.

Le code est organisé en trois parties distinctes : l'inclusion des bibliothèques nécessaires, la déclaration et l'initialisation des variables et des composants, et enfin les fonctions qui définissent le comportement global du système. Ces fonctions principales jouent un rôle clé dans la mise en œuvre et le contrôle des différentes opérations du système.

Dans cette approche, nous avons détaillé chaque partie du code en expliquant ses fondements et son rôle dans le fonctionnement global du système.

Pour commencer notre projet sur la carte ESP32, nous avons débuté par l'inclusion de trois bibliothèques essentielles : WiFi.h, PubSubClient.h et DHTesp.h. Ces bibliothèques sont nécessaires pour gérer respectivement la connexion WiFi, la communication MQTT et l'interaction avec le capteur DHT22.

Ensuite, nous avons configuré les broches utilisées par les capteurs DHT22 (pour la température et l'humidité) et MQ-7 (pour la détection de gaz) en leur attribuant les numéros de broche appropriés sur l'ESP32.

Nous avons également défini et déclaré les informations réseau nécessaire pour établir une connexion WiFi, telles que le nom du réseau (SSID) et le mot de passe associé.

De plus, nous avons spécifié les détails de connexion pour le serveur MQTT, y compris son adresse IP, son port et un identifiant unique (mqtt_client_id) pour notre ESP32. Cet identifiant permet d'assurer une connexion MQTT sécurisée et d'identifier de manière unique notre dispositif sur le réseau.

Enfin, nous avons déclaré des variables pour stocker les données recueillies par les capteurs, notamment la température, l'humidité et le dernier message. Ces variables seront utilisées pour collecter et transmettre les données vers notre tableau de bord ou tout autre service de surveillance.

Cette organisation initiale nous permet de préparer l'environnement de développement de notre programme, en configurant les paramètres nécessaires à la communication réseau

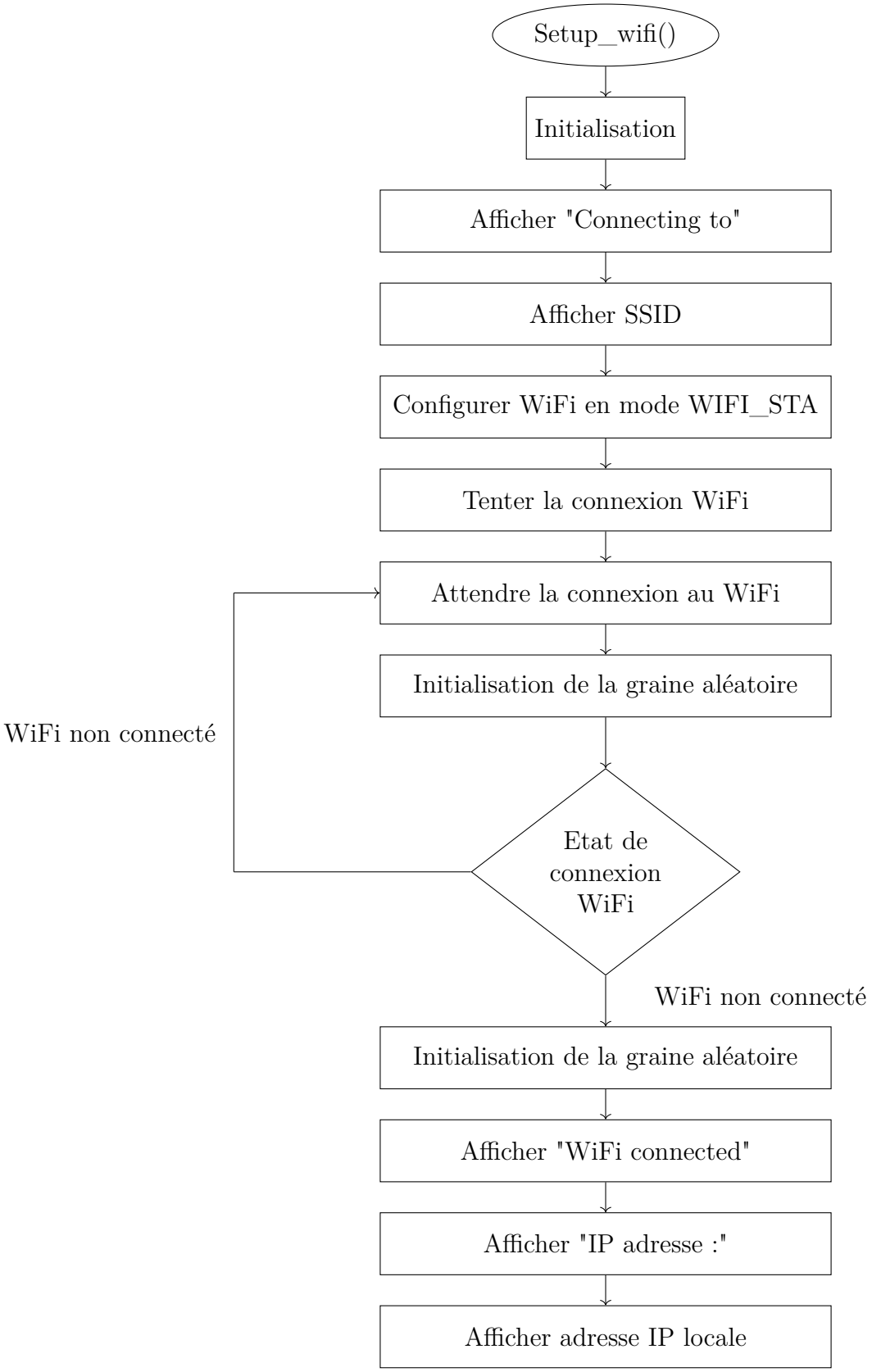
et à la collecte de données à partir des capteurs. La prochaine étape consiste à définir les fonctions qui géreront la connexion WiFi, la connexion MQTT, la lecture des capteurs et l'envoi des données collectées vers le tableau de bord.

FIGURE 3.5 – Inclusion des bibliothèques et des variables dans le code principal

Notre code comprend cinq fonctions importantes que nous allons identifier et détailler ci-dessous.

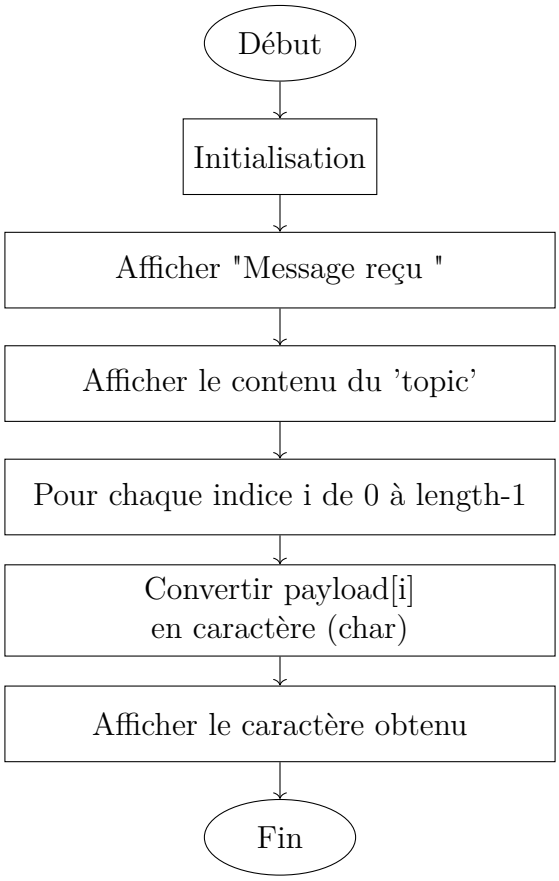
3.2.1 La fonction setup_wifi()

L'organigramme ci_dessous commence par le bloc de démarrage, représentant le début du programme, suivi de l'initialisation de la connexion Wi-Fi. Ensuite, il y a une décision en boucle qui vérifie si la connexion est établie ou non. Si la connexion n'est pas encore établie, le flux revient à la boucle d'attente et continue à vérifier. Une fois que la connexion est établie, le programme génère une graine aléatoire et affiche les messages de succès. Enfin, le flux se termine. Cet organigramme offre une représentation visuelle claire du processus de connexion Wi-Fi de la fonction setup_wifi().


Setup_wifi()
Initialisation
Afficher "Connecting to"
Afficher SSID
Configurer WiFi en mode WIFI_STA
Tenter la connexion WiFi
Attendre la connexion au WiFi
Initialisation de la graine aléatoire
WiFi non connecté
Etat de connexion WiFi
WiFi non connecté
Initialisation de la graine aléatoire
Afficher "WiFi connected"
Afficher "IP adresse :"
Afficher adresse IP locale

3.2.2 La fonction callback()

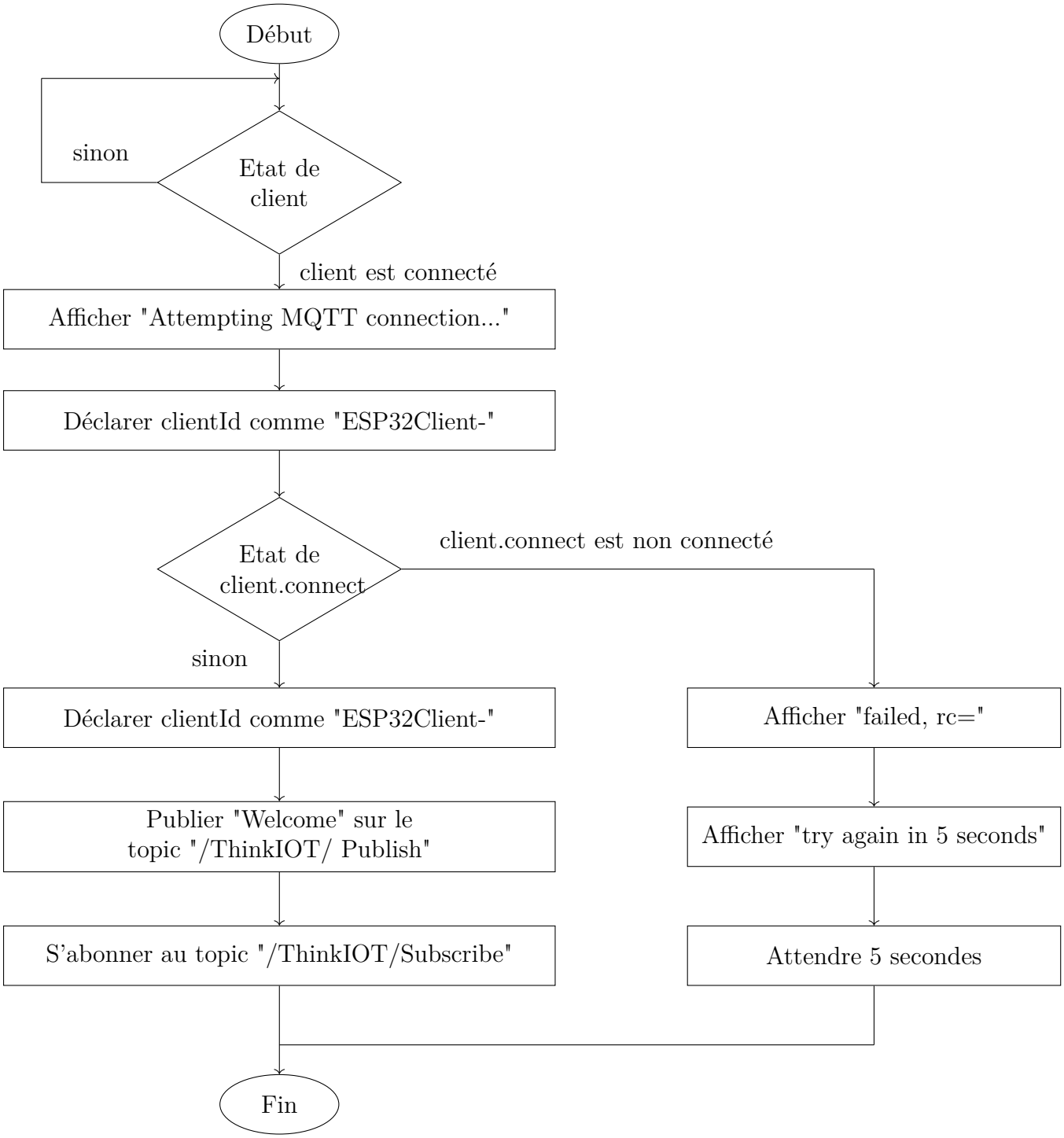
L'organigramme suivant débute par le déclenchement de la fonction callback() avec les paramètres du message MQTT reçu. Ensuite, elle imprime le message "Message arrived [topic]" sur le port série. Après cela, elle itère à travers chaque byte du payload et l'imprime en tant que caractère sur le port série. Une fois tous les bytes du payload traités, le processus se termine. Ce diagramme offre une représentation visuelle claire du fonctionnement de la fonction callback() dans la gestion des messages MQTT.



3.2.3 La fonction-reconnect()

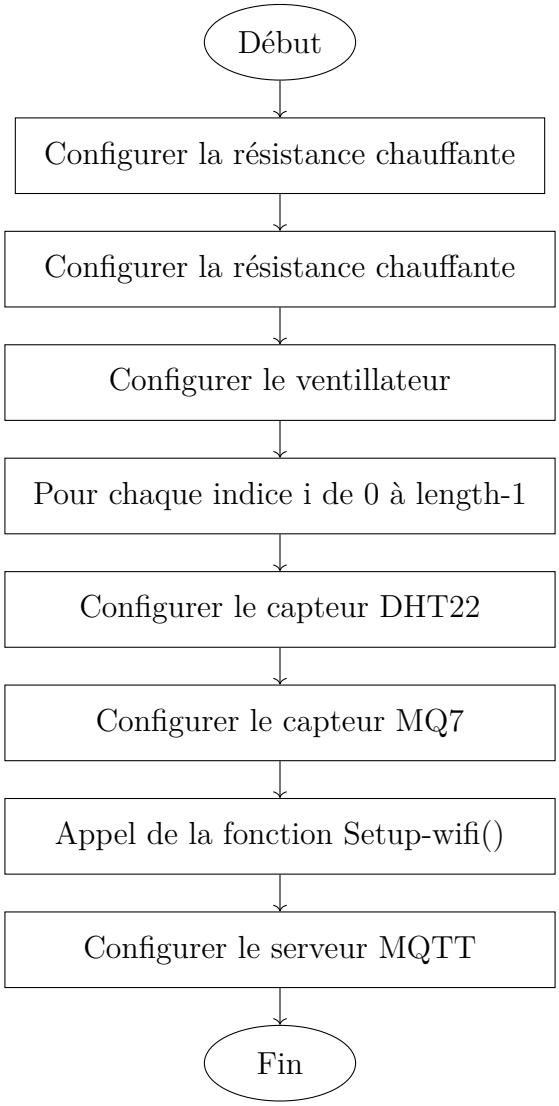
La fonction reconnect() gère la reconnexion au broker MQTT en cas de déconnexion. Elle entre dans une boucle jusqu'à ce que le client MQTT soit connecté. À l'intérieur de cette boucle, elle tente de se reconnecter en générant un identifiant client unique. Si la

connexion réussit, elle publie un message de bienvenue sur le sujet "/ThinkIOT/Publish" et s'abonne à "/ThinkIOT/Subscribe". En cas d'échec, elle affiche un message d'échec avec le code de retour du client MQTT et attend 5 secondes avant de réessayer.



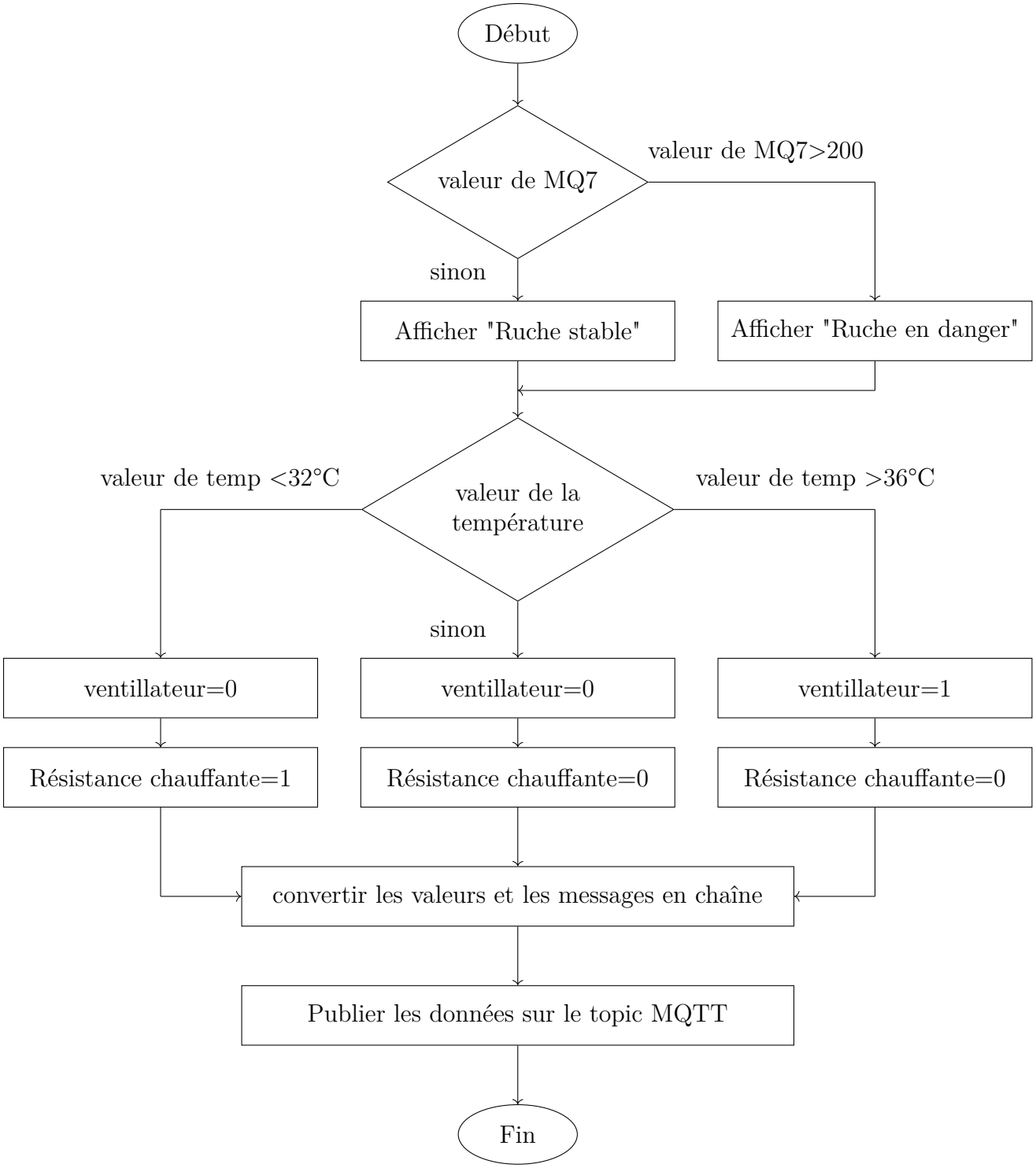
3.2.4 La fonction setup()

La fonction setup() initialise les entrées et les sorties GPIO pour contrôler la résistance chauffante et le ventilateur. Elle démarre la communication série à 115200 bauds pour le débogage et configure la connexion Wi-Fi avec setup_wifi(). De plus, elle configure le client MQTT avec l'adresse du serveur et le port, et définit la fonction de rappel pour le traitement des messages reçus. Cette étape garantit la mise en place correcte des périphériques et des connexions nécessaires avant le fonctionnement principal du programme.



3.2.5 La fonction loop()

La fonction loop() est la boucle principale d'exécution dans de nombreux programmes, et dans ce contexte particulier, elle assume plusieurs tâches cruciales. Tout d'abord, elle compare la valeur mesurée par le capteur de gaz MQ7 à une valeur seuil. Si cette valeur dépasse le seuil, elle affiche "ruche en danger" ; sinon, elle affiche "ruche stable". Ensuite, elle passe à la comparaison de la température mesurée par le capteur DHT22, qui doit être comprise entre 32 et 36 degrés Celsius, ainsi que de l'humidité par rapport à une valeur seuil. Si la température est inférieure à 32 degrés Celsius, elle active la résistance chauffante ; si elle est supérieure à 36 degrés Celsius, elle active le ventilateur. Dans le cas où la température est entre 32 et 36 degrés Celsius, la résistance chauffante et le ventilateur sont désactivés. Ensuite, elle convertit les valeurs et les messages en chaînes de caractères et les publie sur le topic MQTT approprié. Ce processus garantit que les données collectées par les capteurs sont analysées et réagissent en temps réel aux conditions environnementales spécifiées, tout en étant rendues accessibles à d'autres dispositifs connectés via MQTT.


Début
valeur de MQ7
valeur de MQ7>200
sinon
Afficher "Ruche stable"
Afficher "Ruche en danger"
valeur de la température
valeur de temp <32°C
valeur de temp >36°C
sinon
ventillateur=0
ventillateur=0
ventillateur=1
Résistance chauffante=1
Résistance chauffante=0
Résistance chauffante=0
convertir les valeurs et les messages en chaîne
Publier les données sur le topic MQTT
Fin

3.2.6 Lancement du Node-RED

Pour commencer, nous avons d'abord installé Node.js, la plateforme sur laquelle Node-RED est construit. Après avoir terminé l'installation de Node.js, nous avons ouvert l'invite de commande pour vérifier si Node.js était correctement installé. En tapant **node -v**, nous avons confirmé la version installée de Node.js, puis avec **npm -v**, nous avons vérifié la version de npm (Node Package Manager) associée.

Ensuite, nous avons procédé à l'installation de Node-RED en utilisant npm avec l'option –**unsafe** pour autoriser l'installation des modules nécessaires. La commande **npm install -g –unsafe node-red** a été exécutée dans l'invite de commande, et nous avons attendu que l'installation soit complète.

Une fois l'installation terminée, nous avons lancé Node-RED en exécutant simplement la commande node-red dans l'invite de commande. Cela a démarré le serveur **Node-RED** sur notre système Windows.

À ce stade, nous avons pu accéder à l'interface utilisateur de Node-RED via un navigateur web en ouvrant l'adresse **http ://localhost :1880**.

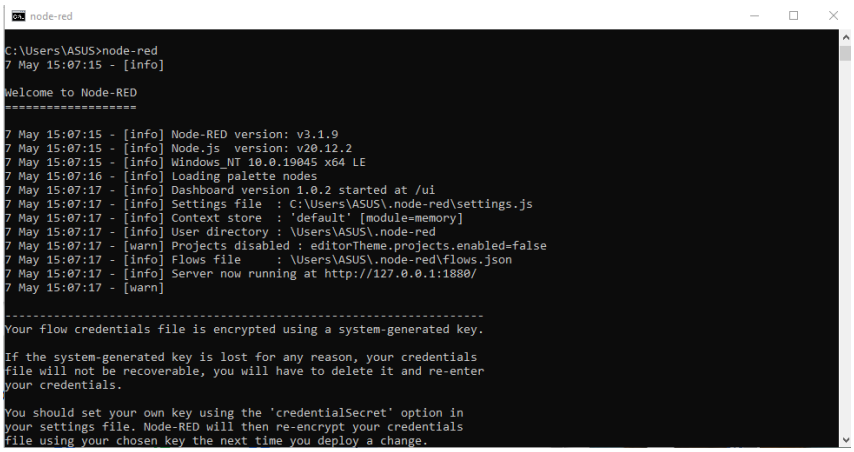


FIGURE 3.6 – Lancement du Node-Red

3.2.7 Configuration du Node-RED

Node-RED offre une intégration native avec MQTT, ce qui en fait un outil puissant pour concevoir un système de ruche connectée. Parmi les nombreux nœuds disponibles, il existe un nœud spécifique dédié à la communication MQTT, avec des options d'entrée (In) et de sortie (Out) qui correspondent respectivement aux fonctions de souscription (Subscribe) et de publication (Publish).

Dans notre configuration, nous avons utilisé Node-RED comme un client Subscriber MQTT. Pour ce faire, nous avons effectué des configurations spécifiques sur le nœud MQTT afin de le connecter à notre courtier (broker) MQTT. Ensuite, nous avons intégré la palette Dashboard de Node-RED pour créer des interfaces utilisateur graphiques (UI) permettant de visualiser les données provenant des capteurs.

En utilisant le tableau de bord (Dashboard), nous avons relié chaque nœud à un widget (gauje) spécifique pour afficher les données des capteurs de manière conviviale et intuitive. Cette approche nous permet de surveiller et d'interagir avec les données des capteurs de la ruche connectée en temps réel, offrant ainsi une expérience utilisateur optimale.

En résumé, Node-RED s'est avéré être une solution flexible et efficace pour développer des applications IoT sur notre système Windows, en utilisant des fonctionnalités avancées de collecte et de visualisation de données pour notre projet.

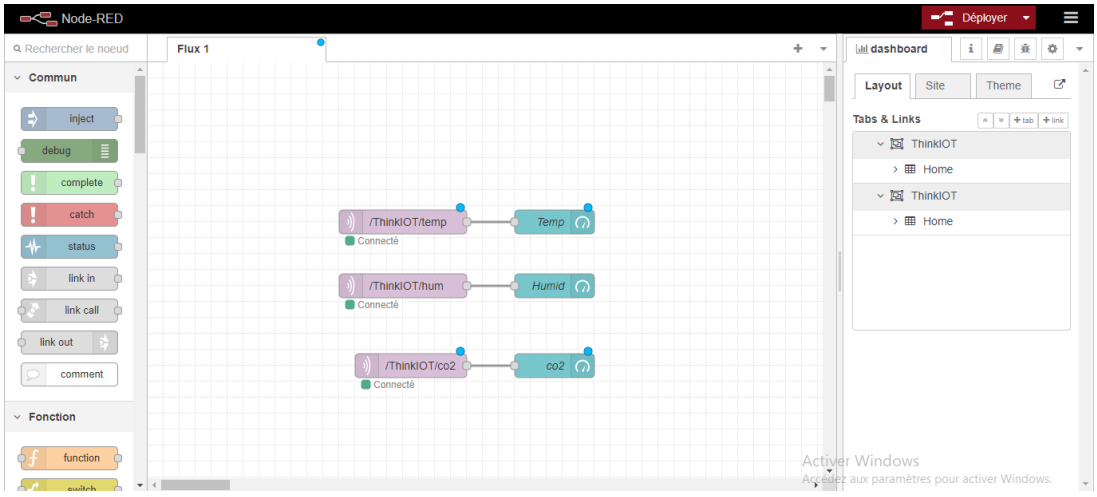


FIGURE 3.7 – Interface Node-RED

3.2.8 Visualisation

Notre objectif principal est d'améliorer le contrôle des abeilles afin de favoriser leur durée de vie, garantir la qualité du miel et augmenter les bénéfices des apiculteurs. Pour cela, nous utilisons une plateforme de visualisation de l'état des ruches à travers les données captées par des capteurs.

Initialement, nous avons utilisé le tableau de bord web de Node-RED pour cette visualisation. Cependant, après une analyse approfondie au sein de notre équipe, nous avons constaté que cette solution n'était pas suffisamment fiable pour répondre à nos exigences spécifiques. Nous avons donc décidé de migrer vers l'application mobile IoT MQTT Panel pour simplifier cette visualisation.

Ce passage vers MQTT Panel nous permet d'améliorer la fiabilité et l'accessibilité de notre système. Ainsi, il offre une expérience utilisateur optimale aux apiculteurs tout en optimisant le contrôle et la gestion des ruches grâce aux données des capteurs.

3.2.9 Problème rencontré

Lors de nos tests visant à transmettre les valeurs du capteur de monoxyde de carbone, nous avons rencontré un problème persistant où la valeur analogique du capteur restait bloquée à 0, malgré nos tentatives de changement de broche utilisée par le capteur. Après des recherches approfondies, nous avons découvert qu'une autre personne avait également rencontré ce problème et avait identifié qu'il était lié à l'ADC2 (convertisseur analogique-numérique) de la carte, qui cesse de fonctionner correctement en présence du WiFi.

3.2.10 Solution trouvée

Pour remédier au problème rencontré avec le capteur de monoxyde de carbone, nous avons opté pour une solution consistant à connecter la broche analogique du capteur à l'ADC1 de la carte au lieu de l'ADC2. Cette modification nous a permis de contourner les interférences causées par le WiFi qui affectaient l'ADC2, ce qui avait pour conséquence de

bloquer la valeur analogique du capteur à 0. En utilisant l'ADC1, nous avons pu obtenir des lectures correctes et fiables du capteur, garantissant ainsi le bon fonctionnement de notre système de surveillance. Grâce à cette approche, nous avons pu poursuivre nos expérimentations sans être impactés par les limitations de l'ADC2 en présence du WiFi, ce qui assure la stabilité et la précision des mesures de monoxyde de carbone effectuées par notre dispositif.

Conclusion

Dans ce chapitre, nous avons décrit les différents logiciels que nous avons utilisés dans notre projet. Nous avons également présenté la partie code et configuration de notre travail, ainsi que les difficultés rencontrées, notamment les erreurs que nous avons dû résoudre.

Conclusion générale

En terme de conclusion, notre projet de ruche connectée représente une convergence réussie entre tradition et technologie, offrant une approche moderne à un problème ancestral. Toutefois, notre travail ne s'arrête pas ici. Les perspectives futures incluent l'expansion de notre solution pour répondre aux besoins spécifiques des apiculteurs, ainsi que la poursuite de la recherche et du développement dans le domaine de l'apiculture intelligente. En continuant à explorer les possibilités offertes par les avancées technologiques, nous sommes confiants dans le fait que notre projet contribuera à promouvoir la durabilité de l'apiculture et à protéger les précieuses populations d'abeilles pour les générations à venir. Suite à la concrétisation de notre travail, plusieurs perspectives se sont présentées à nous dans le but de parfaire les résultats obtenus. Il serait intéressant de :

- **Capteur de Poids :** Intégré discrètement pour évaluer le poids de la ruche sans perturber les abeilles.
- **Développement de nouvelles technologies :** des nouvelles technologies émergentes telles que l'intelligence artificielle et l'apprentissage automatique qui sont appliquées pour analyser les données des ruches et fournir des informations précieuses aux apiculteurs.
- **Collaborations interdisciplinaires :** Les projets de ruches connectées offrent des opportunités de collaboration interdisciplinaire entre les apiculteurs, les scientifiques, les ingénieurs et les décideurs politiques pour aborder les défis complexes liés à la santé des abeilles et à la durabilité agricole.