***Dédicaces***

***Remerciement***

***Sommaire***

***Liste des figures***

***Liste des tableaux***

Introduction GÉNÉRALE

L'évolution rapide, la généralisation et l’usage des technologies relatives à l’Internet des objets **IoT** (**I**nternet **o**f **t**hings), réinventent de nos jours les différents environnements industriels pour créer un nouveau monde de fabrication ou de production en pouvant être désormais autonome, intelligent, et connecté. L’IoT représente ainsi le tremplin indispensable qui permet de transformer les industries historiques traditionnelles en systèmes intelligents, productifs, connectés, qualifiés et hautement automatisés. En effet, l’adoption des capteurs, des modules de communication, ou encore des analyses des données, permettent à l’IoT de garantir l’accomplissement d’une supervision de mesure des performances en temps réel, la maintenance prédictive, ou une prise de décision optimisée autant de moyens permettant un meilleur rendement, en sécurité, et en excellence opérationnelle.

À l’échelle mondiale, de nombreuses organisations industrielles recourent à des solutions IoT, c’est-à-dire embarquant des capteurs et des objets connectés, pour améliorer leur processus, pour réduire leur temps d’indisponibilité, et par conséquent pour réduire leurs dépenses d’exploitation. La possibilité d’avoir des données réelles sur le fonctionnement des machines, et sur le contexte dans lequel elles sont employées, permet d’identifier les pannes possibles, avant même qu’elles arrivent, pour éviter les arrêts coûteux, et améliorer le cycle de vie de machines. Ce changement d’état d’esprit favorise le maintien de la machine, en favorisant la démarche préventive au détriment des techniques de maintenance standard, à la fois en développant l’anticipation des évolutions de l’équipement, et en limitant sa période d’indisponibilité annuelle. De plus, avec ce changement, un exploitant peut imaginer beaucoup moins d’interactions humaines différentes, ce qui lui permet de réaffecter ces ressources à d’autres investissements humains stratégiques.

Au sein de cette perspective, l’objet de ce travail présenté dans le rapport va consister à spécifier un dispositif IoT à très grande échelle pour la surveillance préventive industrielle. Ce dispositif a pour but de maintenir un alliage de surveillance de l’état des machines et dans la situation sous tension de la machine afin de détecter des anomalies éventuellement précurseur de défaillances ou des conditions de fonctionnement dangereux.

Dans le premier chapitre je présente brièvement la définition et le domaine d’utilisation des différents types de capteurs. Pour passer par la suite au cahier de charge du système.

En suite le deuxième chapitre est consacré à la présentation des différents composants utilisés dans ce projet et la réalisation des différents câblages.

Dans le troisième chapitre je montre la programmation de notre système.

En fin le dernier chapitre sera pour la conclusion générale et les perspectives de continuité.

CHAPITRE I :

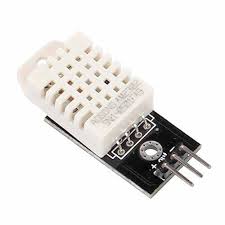
Environnement et problématique

1. **Introduction**

Dans ce contexte, le projet présenté dans ce rapport vise à détailler un système IoT à grande échelle axé sur la surveillance industrielle préventive. En permettant la détection continue de l’état de fonctionnement des composants en fonctionnement de l’installation, le système permet une détection précoce des anomalies, déclenche donc un signal d’alerte précoce lors de mode de fonctionnement mal caractérisé dans un contexte en particulier, et éventuellement en fonction d’interactions mal comprises entre les machines. Ce projet est avant tout axé sur des fonctionnalités systématiques, tournées vers l’utilisateur et sur leurs applications industrielles ; le premier apport étant un réseau multi capteurs permettant l’évaluation des conditions environnementales globales, et du contrôle d’un nombre beaucoup plus grand de paramètres : température, humidité, concentration de gaz… fonctionnalités que les systèmes de surveillance mono usage ne peuvent pas fournir. D’autre part, on a un écran LCD interactif procurant quatre fonctions utilisateurs : définition des limites des capteurs, lecture historique sur « ThingSpeak » , affichage des valeurs mesurées et limites des capteurs en temps réel localement ou sur un autre écran, et transfert des données vers un PC au traitement ultérieur.

1. **Les capteurs : Définitions et domaines d’utilisations**
2. **Le capteur de température et d’humidité**

Un capteur de température et d'humidité est un dispositif électronique qui mesure et signale simultanément la température et l'humidité de l'air ou d'un matériau. Ces capteurs sont largement utilisés dans divers domaines, notamment la gestion des bâtiments, l'automatisation industrielle (Ils sont utilisés dans les processus de fabrication, de séchage, et dans les chambres climatiques pour assurer des conditions stables et éviter les dégradations de produits), la climatisation, et la surveillance environnementale, pour optimiser les conditions et prévenir les dommages liés à des niveaux d'humidité inappropriés.



1. **Le capteur de tension**

Un capteur de tension, ou capteur de voltage, est un dispositif électronique qui mesure la différence de potentiel électrique entre deux points d'un circuit et la convertit en un signal mesurable. Ce signal peut être utilisé pour surveiller, contrôler ou analyser le circuit électrique. Ils sont essentiels dans de nombreuses applications, notamment l'électronique (Contrôle de l'alimentation, protection des circuits, mesure de la tension dans les cartes électroniques), l'industrie (Surveillance de la consommation d'énergie, contrôle de moteurs électriques, protection contre les surcharges) et l'automobile (Suivi de la consommation électrique dans les véhicules électriques et hybrides).



1. **Le capteur de gaz**

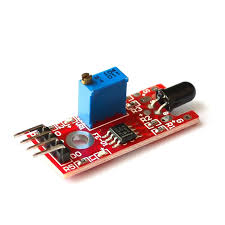
Un capteur de gaz, également appelé détecteur de gaz, est un appareil qui mesure la concentration de gaz dans l'atmosphère. Il est utilisé pour identifier et quantifier la présence de gaz potentiellement dangereux ou toxiques. Ces capteurs sont essentiels pour assurer la sécurité dans de nombreux domaines, allant de l'industrie à l'environnement, en passant par les environnements domestiques.



1. **Le capteur de flamme**

Un capteur de flamme est un dispositif électronique conçu pour détecter la présence de flammes ou de feux ouverts. Il fonctionne en reconnaissant les caractéristiques spécifiques de la lumière émise par une flamme, telles que le rayonnement ultraviolet (UV) ou infrarouge (IR). Ces capteurs sont essentiels dans les systèmes de sécurité pour la détection précoce des incendies et la protection des équipements et des personnes.

Les capteurs de flamme sont largement utilisés dans divers domaines telques : Industrie pétrolière et gazière, Industrie chimique, Chauffage et chaudières, Automatisation industrielle, Systèmes d'alarme incendie.



1. **Le capteur de vibration**

Un capteur de vibration, également appelé détecteur de vibrations, est un dispositif qui mesure les vibrations mécaniques d'un objet ou d'une machine. Il convertit ces vibrations en signaux électriques, permettant ainsi leur analyse et l'évaluation de l'état de l'équipement. Ces capteurs sont largement utilisés dans la maintenance prédictive et la surveillance de l'état des machines, notamment dans l'industrie.



1. **Le capteur d’obstacles**

Un capteur de détection d'obstacles est un dispositif conçu pour identifier et signaler la présence d'objets sur la trajectoire d'un robot ou d'un véhicule en mouvement.



1. **Le capteur de niveau**

Un capteur de niveau est un dispositif électronique qui mesure la hauteur d'un liquide ou d'un solide dans un récipient, tel qu'un réservoir ou une cuve. Il est utilisé pour surveiller et contrôler les niveaux de matière afin d'éviter les débordements, les manques, ou pour déclencher des actions de remplissage ou de vidange. Les capteurs de niveau trouvent leur utilité dans divers domaines, allant de l'industrie à l'électroménager, en passant par l'automobile et la surveillance environnementale.



1. **Cahier de charge**

Le but de ce projet est de réaliser un système embarqué IoT qui permet de contrôler plusieurs paramètres et renvoyer un signal en cas d’anomalie. Ce système nous facilite à détecter la production des pannes et à intervenir avant l’accident. Ce système à un grand rôle dans le gain en temps lors de la maintenance et surtout garantir la sécurité humaine et matérielle.

Notre système sera composé :

* De 7 capteurs pour détecter les différents paramètres.
* Un microcontrôleur ESP32, un écran LCD, un clavier matriciel et un module CJMCU-4051.

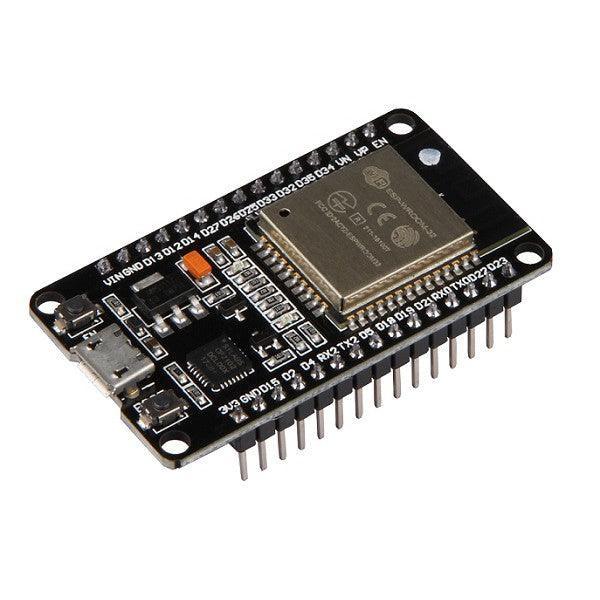
1. **Conclusion**

Ce premier chapitre nous donne une idée sur les différents capteurs et leurs domaines d’applications ainsi que la présentation du cahier de charge.

Une étude sur les différents composants électroniques et leurs principes de câblage sera consacré pour le deuxième chapitre.

Chapitre II : étude théorique

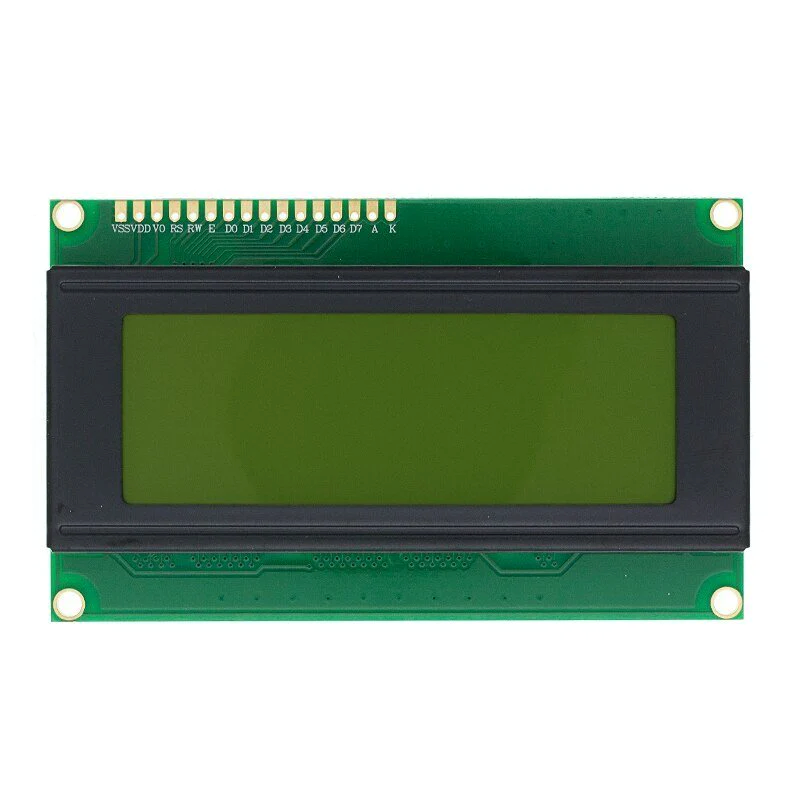
1. **Matériels utilisés**
2. **Le Microcontrôleur ESP32**

Le microcontrôleur ESP32 est le processeur principal de mon système IoT, offrant un haut niveau de puissance de calcul, de connectivité multi usage pour les applications IoT embarquées. Il présente de très bonnes performances en proposant par ailleurs du multitâche, permettant ainsi de traiter en temps réel les données utiles à la gestion avancée des capteurs et l’accès au réseau. L’atout principal de l’ESP32 est d’intégrer dans la puce les modules Wi-Fi et Bluetooth (Classique et BLE) permettant ainsi d’assurer la communication sans fil, ce qui permet d’éviter des composants supplémentaires.

Dans mon projet, l'ESP32 rend le système capable de collecter des données à partir de capteurs, de naviguer sur Internet, d'envoyer des données au serveur (Thinkspeek) et de communiquer avec d'autres appareils. De plus, il traite les données et les compare à des seuils pour déterminer les problèmes potentiels. Il communique avec l'écran LCD pour afficher les données à l'utilisateur et avec le clavier pour lire vos entrées. Via Wi-Fi, il envoie des données au PC pour créer un fichier Excel contenant la date du capteur et leur courbe.

1. **L’écran LCD**

J'ai utilisé un écran LCD I2C de 20x4 caractères, qui est l'une des interfaces de sortie essentielles fournissant aux utilisateurs des données et des retours en temps réel. C'est un écran qui peut afficher 4 lignes avec 20 caractères par ligne (simultanément). La couleur des caractères est blanche et l'écran a un fond bleu. Cela le rend également lisible dans l'obscurité.  
Il affiche des données de capteurs en temps réel, des annonces système et des sélections de menu, ce qui rend mon système interactif et facile à utiliser. L'écran LCD se connecte à l'ESP32 via le protocole I2C, un système simple à deux fils (SDA pour les données et SCL pour l'horloge) qui réduit l'encombrement des câbles et les échecs de communication.



1. **Le clavier matriciel**

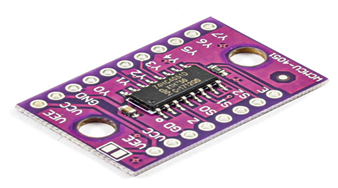
La configuration IoT fait appel à un clavier matriciel 4x4 en tant que principal élément d’entrée pour l’utilisateur, constitué de 4 lignes de rangées interfacées avec des broches GPIO de l’ESP32 en mode sortie, et de 4 lignes de colonnes interfacées avec des broches GPIO en mode entrée disposant de résistances de tirage interne actives. La détection d’entrée de ce clavier est pertinente dans l’interface utilisateur du système pour naviguer dans le menu présenté sur l’afficheur (LCD), définir, paramétrer la limite d’alarme du capteur et saisir les valeurs du champ date à prendre en compte pour les requêtes de données. On sera par exemple amené à utiliser les touches ‘#’ pour valider ou ‘\*’ pour annuler les différentes actions. Le système IoT peut ainsi atteindre un contrôle interactif avec une bonne ergonomie malgré la banalité de son équipement matériel.



1. **Module CJMCU-4051**

Tout en étant un produit dérivé de l’IC multiplexeur analogique 74HC4051 (pour lequel il existe plusieurs équivalents), le module CJMCU-4051 voit son emploi à des fins de connexion dans le système IOT, au profit de la multiplication des entrées de capteurs analogiques disponibles pour le microcontrôleur ESP32. Il est composé de huit canaux d’entrée pouvant être sélectivement adressés vers un canal de sortie communément disponible pour l’une des broches ADC de l’ESP32.

Pour faciliter l’exposition des capteurs à la liberté du scan par commutateur de canaux à haut débit, le CJMCU4051, tout en tenant compte des dispositions de multiplexage matériel, permet d’effectuer l’amélioration en réalité sur des capteurs gaz, béance, vibration, détection d’obstacle, niveaux de flottaison (ce qui est en soi un ajout du niveau de lecture), capable de partager une ligne d’entrée analogique commune sans qu’il n’y ait d’encombrement sur un circuit moins complexe de câblage extensible.



Le câblage au niveau matériel pour le multiplexeur CJMCU-4051 est donc en mesure d’intégrer plusieurs capteurs au sein d’un même système avec le microcontrôleur ESP32 permettant de lire les informations dans un temps limité et d’obtenir des relevés sur le taux d’audio et la luminosité ambiante. Les broches de contrôle qui permettent de déterminer le canal de capteur actif sont bien reliées à l’ESP32 qui commande donc la sélection d’un canal (d’un capteur) à l’autre en ayant recours à un système d’adresse binaire. La sortie commune de l’émetteur multiplexé est reliée à une entrée analogique sur l’ESP32, configurée pour des lectures du type des capteurs, sur une zone de tension choisie sous un voltage maximal fournissant suffisamment un niveau d’analogique à partir des capteurs. Les huit sorties possibles du multiplexeur sont reliées aux sorties analogiques de différents capteurs, tel que le capteur de gaz MQ-5, le capteur de flamme KY-026, le capteur de vibration SW-420, le capteur d’obstacle infrarouge FC-51 et un capteur de niveau à flotteur, toutes permettant la surveillance des paramètres mesurés. Via l’unique entrée analogique de l’ESP32, il est donc possible de surveiller plusieurs paramètres relevés par une large pléthore de capteurs dans une unique carte électronique.

1. **Les capteurs**
2. Capteur Température et Humidité ( DH22) :

Caractéristiques

* Alimentation : 3,3 à 6 Vcc
* Consommation maxi : 1,5 mA
* Consommation au repos : 50 µA
* Plage de mesure :  
  - température : -40 à +80 °C  
  - humidité : 0 à 100 % RH
* Précision :  
  - température : ± 0,5 °C  
  - humidité : ± 2 % RH
* Dimensions : 25 x 15 x 9 mm

Fonction

Dans le système que j'ai établi le DHT22 nous donne les mesures de température et d'humidité dans l'air ambiant près des machines industrielles et quand l'une ou l'autre atteint un seuil anormal le système déclenche une alerte simple et informative.

1. Capteur Voltage:

Caractéristiques :

* Alimentation : 5 V ou 3,3 V
* Tension d'entrée : DC 0 à 25 V
* Tension de détection : DC 0,02445 V à 25 V
* Résolution analogique : 0,00489 V
* Tension minimale détectable : 0,02445 V

Fonction: Pour mesurer la tension, il convient de brancher le capteur en parallèle dans le circuit en laissant passer le courant librement en évitant d’ajouter une résistance notable. Le capteur transforme alors cette tension en signal analogique pour un microcontrôleur.

1. Capteur Gas (MQ5):

Caractéristiques :

* Interface: n'importe quel port analogique peut être utilisé.
* Tension de fonctionnement: DC 5V
* Courant de travail: 150mA
* Détection de concentrations: 300 ~ 5000ppm
* Humidité fonctionnante: 95% hr (humidité nominale 65%)
* Dimensions: 32 x 21 x 20 mm
* poids: 9.00g

Fonction: Le capteur de gaz MQ-5 est fabriqué à partir de SnO2, le matériau sensible, qui permet d’obtenir une très faible conductivité en l’absence de gaz et qui augmente avec la concentration de gaz combustibles (butane, propane ou méthane) dans l’environnement. En utilisant un circuit peu complexe, les utilisateurs peuvent utiliser la variation de conductivité du capteur pour la convertir en un signal de sortie puis en niveaux de gaz. Ce capteur est très sensible à la concentration de butane, propane et méthane (quasi simultanément) mais également à d’autres gaz combustibles, notamment le GPL (propane).

1. Capteur Flamme (KY-026):

Caractéristiques :

* Alimentation : 3,3 V à 5,5 V
* Plage de mesure :
  + Longueur d'onde infrarouge : 760 nm à 1100 nm
  + Angle de détection : 60°
* Précision : Non spécifiée (à préciser si disponible)
* Dimensions : 1,5 cm x 3,6 cm [0,6 po x 1,4 po]
* Composants :
  + LED de réception infrarouge de 5 mm
  + Comparateur différentiel double LM393
  + Potentiomètre de réglage 3296 W
  + Six résistances
  + Deux LED indicatrices

Fonction: Le KY-026 est un module qui regroupe une photodiode sensible à l’infrarouge. Il permet de détecter une flamme en recueillant son rayonnement. En présence d’une flamme, la résistance de la photodiode est modifiée. Cette variation est convertie en un signal de sortie de détection qui peut être interprété par un microcontrôleur.

1. Capteur Vibration (SW-420):

Caractéristiques :

* Alimentation : 3,3 V à 5 V
* Consommation : 15 mA
* Dimensions : 3,2 cm x 1,4 cm
* Composants :
  + Comparateur de tension large LM393

Fonction: Le SW-420 est constitué d’un interrupteur à bascule sensible aux vibrations qui se ferme lorsque le capteur vibre et modifie la résistance du circuit. Un comparateur LM393 permet de déceler cette modification de résistance et de produire un signal de sortie numérique utilisable par un microcontrôleur.

1. Capteur Obstacle (FC-51):

Caractéristiques :

* Distance de détection : de 2 à 30 cm
* Dimensions : 3.1 cm \* 1.5 cm
* Alimentation : 3.3 – 5 V.

Fonction: Ce capteur émet un faisceau de lumière infrarouge qui se réfléchit sur tout obstacle dans son champ d’atterrissage accessible, ce dernier étant alors détecté au moment où le faisceau est de retour au capteur, ce qui nous conduit à conclure que l’obstacle est présent. Cela se traduit par une sortie numérique possible qui vient à être gérée par la sortie logique (0 ou 1) de manière à faire une mesure d’occupation, occupation qui peut être réglée en fonction de la distance effective de coupe de l’obstacle pour que ce soit son retour de signal qui soit géré par un potentiomètre du capteur.

1. Capteur de niveau

Caractéristiques :

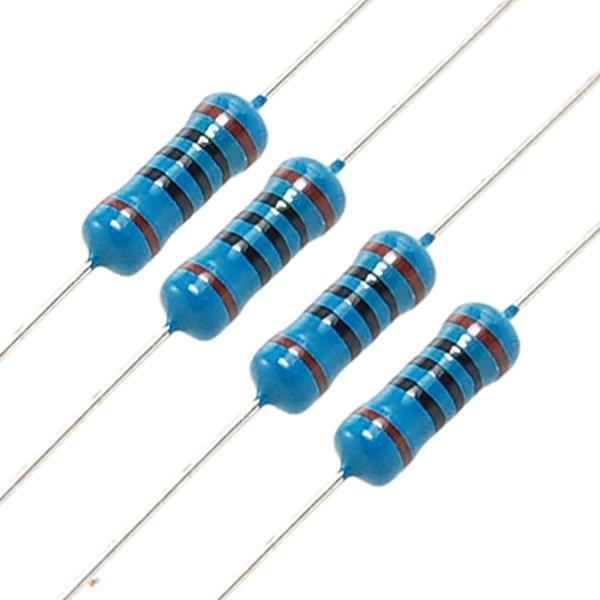
* Plage de mesure : -10°C à +60°C
* Résistance de contact (max) : 100 mΩ
* Dimensions : Taille compacte
* Matériau : PP (Polypropylène)
* Capacités :
  + Capacité de contact (max) : 100 W
  + Tension de commutation (max) : 100 V CC
  + Courant de commutation (max) : 0,5 A
  + Tension de claquage (max) : 220 V CC
  + Courant de transport (max) : 1 A

Fonction: Un flotteur immergé dans le liquide suit avec précision ses variations de niveau : il est relié à un interrupteur ou un capteur qui actionne ou non lorsque le flotteur atteint un seuil prédéfini selon le niveau du liquide.

1. **Les actionneurs**

**Resistance :**

J’ai utilisé des résistances pour protéger mes LEDs de 5 mm, j’ai choisi des résistances de 150 Ω chacune en série pour limiter le courant à 20 mA avec une alimentation de 5 V. Ces résistances sont essentielles pour assurer la stabilité et la sécurité du système.

****

**Buzzer:**

Le buzzer est un dispositif d'alerte sonore crucial conçu pour fournir à l'utilisateur des notifications immédiates et audibles. Le buzzer a une tension de fonctionnement comprise entre 3V et 24V, ce qui le rend très compatible avec l'alimentation du système et flexible dans différentes conditions de déploiement.



L'ajout de ce buzzer actif renforce l'interface homme-machine du système en ajoutant une notification audible immédiate et sans ambiguïté, complémentaire aux signaux visuels des LED.

**LEDs:**

Utilisation de LED multicolore de 5 mm comme indicateurs visuels pour fournir une notification visuelle contrastante et immédiate sur l'état du système et l'alerte environnementale. Parmi ces trois LED essentielles (blanche, jaune, rouge), elles sont intégrées comme indicateurs principaux de statut sur l'interface matérielle du système.  
Ces LED sont alimentées directement par les broches de sortie numérique du microcontrôleur et sont alimentées de manière programmatique pour signaler visuellement des niveaux de gravité d'alerte de plus en plus distincts déclenchés par le traitement des données des capteurs.



Fonction:

1. LED blanche: Alimentée par la broche de sortie du microcontrôleur , ce capteur détecte des

alarmes mineures ou de faible niveau comme de petites fluctuations de température, d’humidité ou de tension en dehors des limites attendues. Sa lumière signale au système avoir détecté un écart par rapport à la condition normale qui doit être communiqué à l’utilisateur mais n’affectera pas la sécurité et l’intégrité du système.

1. LED jaune: Alimentée par la broche de sortie du microcontrôleur, ce capteur détecte des alarmes mineures ou de faible niveau comme de petites fluctuations de température, d’humidité ou de tension en dehors des limites attendues. Sa lumière signale au système avoir détecté un écart par rapport à la condition normale qui doit être communiqué à l’utilisateur mais n’affectera pas la sécurité et l’intégrité du système.
2. LED rouge: cette alimentation provient de la broche de sortie du microcontrôleur. Le rouge LED émettra des événements avertisseurs critiques; cela se produira si le statut du capteur est à un niveau alarmant, comme une exposition au gaz ou à un feu. Cela peut ne pas être dangereux à lui seul, mais contient un élément de menace ou d’urgence qui doit être traité par un utilisateur dans un court laps de temps.
3. **Autres Composants**

**Bouton Poussoir:**

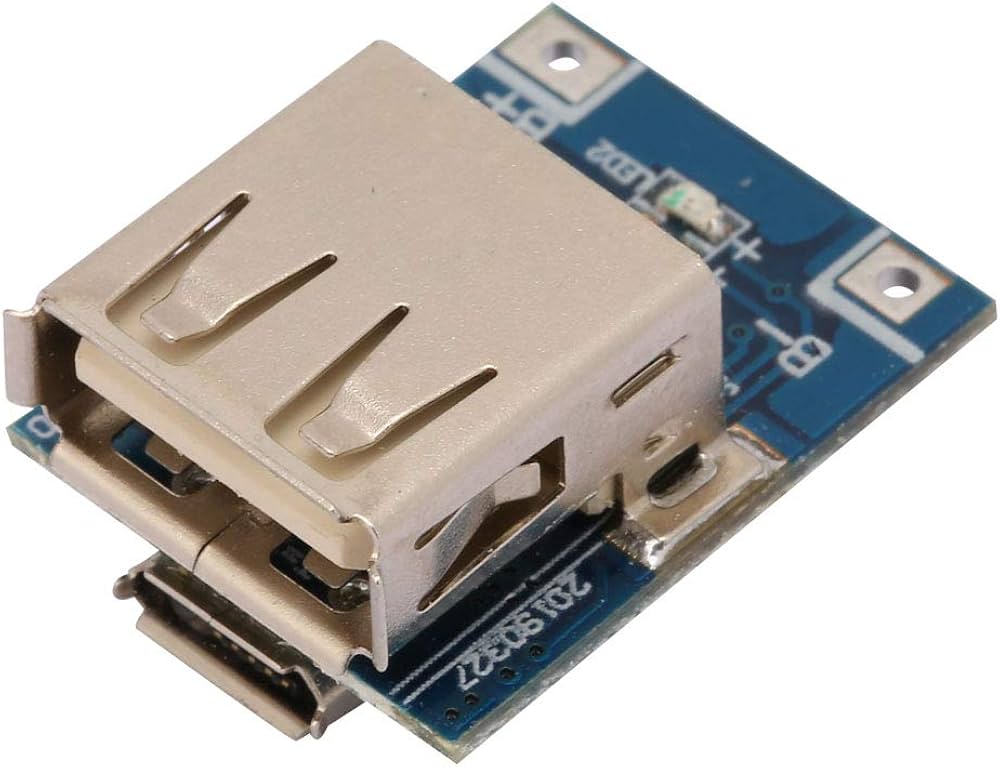
Le bouton poussoir est construit en métal solide, garantissant sa résistance mécanique, mais aussi il est capable de résister à un environnement difficile. Il a un diamètre réduit de 16 mm, qui est le mieux adapté au panneau du système embarqué où la compacité pour le moindre millimètre est essentielle. Le fait qu’il travaille entre 12 et 24 volts DC, compatible avec les exigences d’alimentation du système, permet son intégration sans ajout d’un ou plusieurs composants pour assurer la régulation de tension.



Déclencher le bouton entraîne un signal toge associé à un état LOW traité par l’intermédiaire du "system\_active" fonction qui gère l’exécution du programme ; en revanche, la relâcher initie immédiatement le processus d’extinction du système à l’aide de HIGH. C’est pourquoi l’architecture rend aisé le contrôle exercé par l’utilisateur sur le cycle de vie opérationnel de l’appareil IoT qui garantit l’intégrité du circuit et la consommation d’énergie.

**Module de charge:**

Module Charge, Décharge et Protection Dans notre projet IoT, la gestion d’une alimentation sûre et fiable est nécessaire à celui des dispositifs connectés. Cette carte permet de charger des batteries lithium-polymère (Lip) en utilisant une alimentation USB 5 volts, pouvant délivrer un courant de charge maximum de 1 ampère, et acceptant des capacités de batteries, allant des petits systèmes intégrés, à des moyens systèmes (IoT). La protection de la batterie contre des situations dégradantes (cour circuit, surcharge, décharge excessive, surintensité) est l’un de ses rôles essentiels.

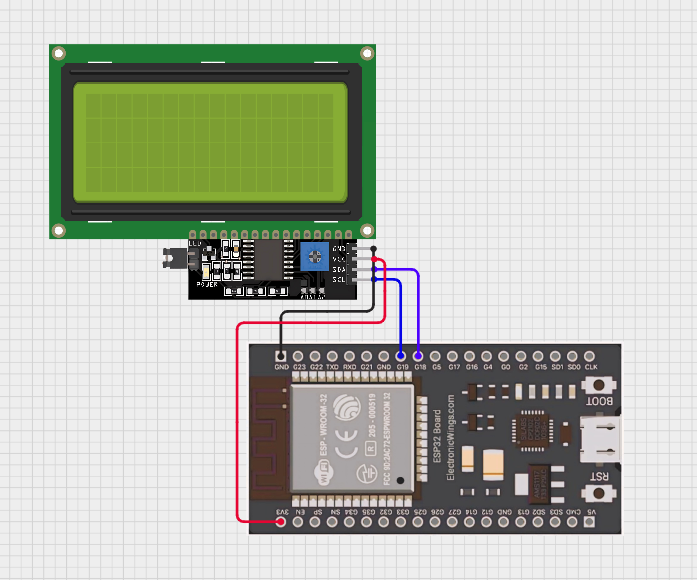


La carte se laisse facilement alimenter via une prise USB classique de 5V, ce qui simplifie la tâche puisque l’on ne dispose d’aucun moyen particulier de charge. Ce module possède également les caractéristiques d’un instrument de haute précision par le biais du développement d’un microcontrôleur ESP32 en montage avec des capteurs capables de détecter en automatique des paramètres à surveiller tout à fait adaptés au milieu de la surveillance d’aujourd’hui.

Chapitre III : étude PRATIQUES

1. PRINCIPE DE CABLAGE
2. Ecran LCD :

J’ai câblé l’écran LCD en utilisant une interface I2C avec le module SoftI2C. J’ai connecté les broches SDA et SCL respectivement aux broches GPIO16 (SDA) et GPIO17 (SCL) de la carte, avec une fréquence de 400 000 Hz. J’ai défini l’adresse I2C à 0x27 et configuré l’écran pour 4 lignes et 20 colonnes. J’utilise une alimentation de 5 V via les broches VCC et GND connectées aux alimentations correspondantes de la carte.



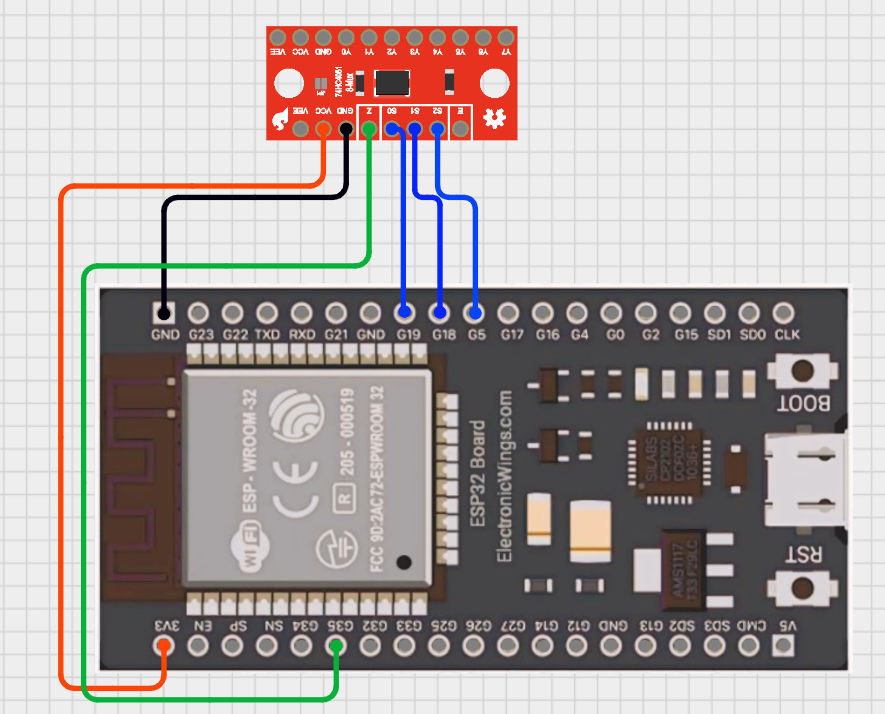
1. Clavier Matriciel :

J’ai câblé le clavier matriciel avec quatre lignes et quatre colonnes. J’ai connecté les lignes aux broches GPIO32, GPIO33, GPIO25 et GPIO26 en mode sortie, tandis que j’ai relié les colonnes aux broches GPIO27, GPIO14, GPIO12 et GPIO13 en mode entrée avec résistance de pull-up. J’utilise cette configuration pour détecter chaque touche par une mise à niveau basse des lignes et une lecture des états des colonnes.



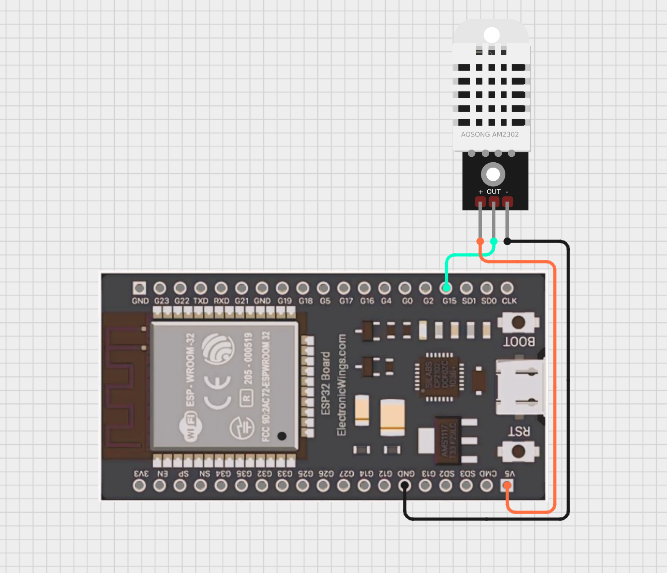
1. Module CJMCU :

J’ai câblé le module CJMCU comme un multiplexeur analogique en connectant les broches de sélection S0, S1 et S2 respectivement aux GPIO5, GPIO18 et GPIO19 en mode sortie. J’ai relié la sortie analogique du multiplexeur (Z) à la broche ADC GPIO35. J’utilise une alimentation de 5 V via les broches VCC et GND connectées aux alimentations correspondantes de la carte.



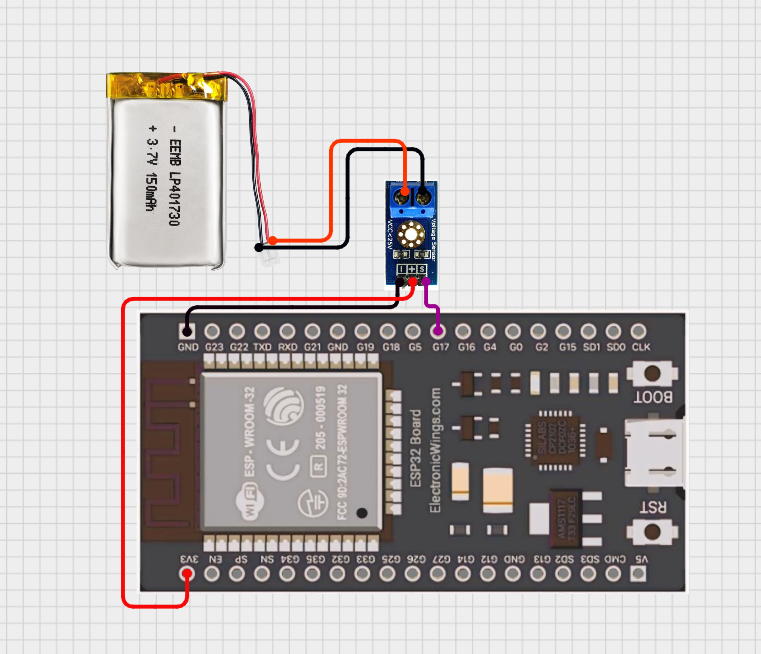
1. Capteurs :
2. Capteur de Température et Humidité (DHT22) :

J’ai câblé le capteur DHT22 en connectant sa broche de données à la broche GPIO23 de la carte en mode entrée. J’ai ajouté une résistance de pull-up (nécessaire pour son fonctionnement) et j’utilise une alimentation de 5 V via les broches VCC et GND pour mesurer la température et l’humidité à intervalles réguliers.



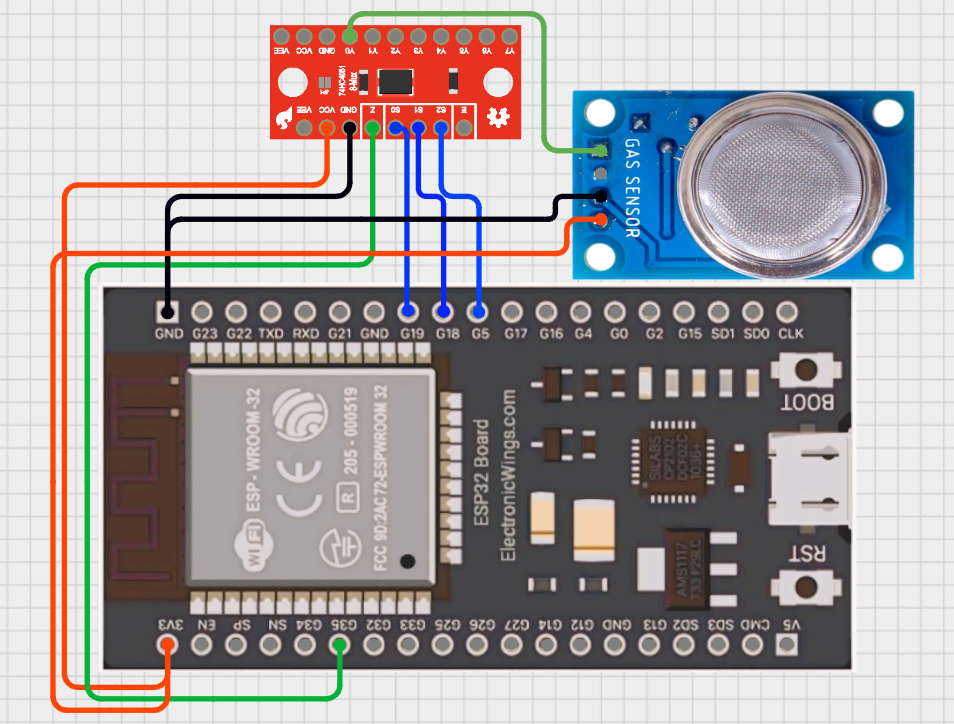
1. Capteur de Tension :

J’ai câblé le capteur de tension en reliant sa sortie analogique à la broche ADC GPIO34 de la carte, configurée avec une atténuation de 11 dB pour supporter une tension d’entrée maximale de 25 V grâce à un diviseur de tension. J’utilise une alimentation de 5 V via les broches VCC et GND pour assurer une mesure précise.



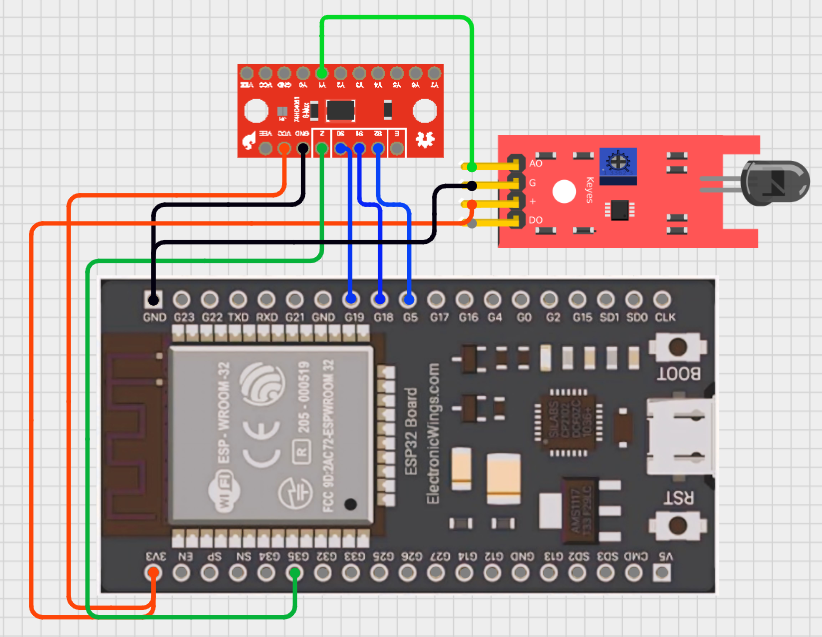
1. Capteur de Gaz (MQ5) :

J’ai câblé le capteur de gaz MQ5 via le multiplexeur CJMCU, sélectionné sur le canal 0, et j’utilise une alimentation de 5 V via VCC et GND. Je détecte la présence de gaz avec un seuil de 700.



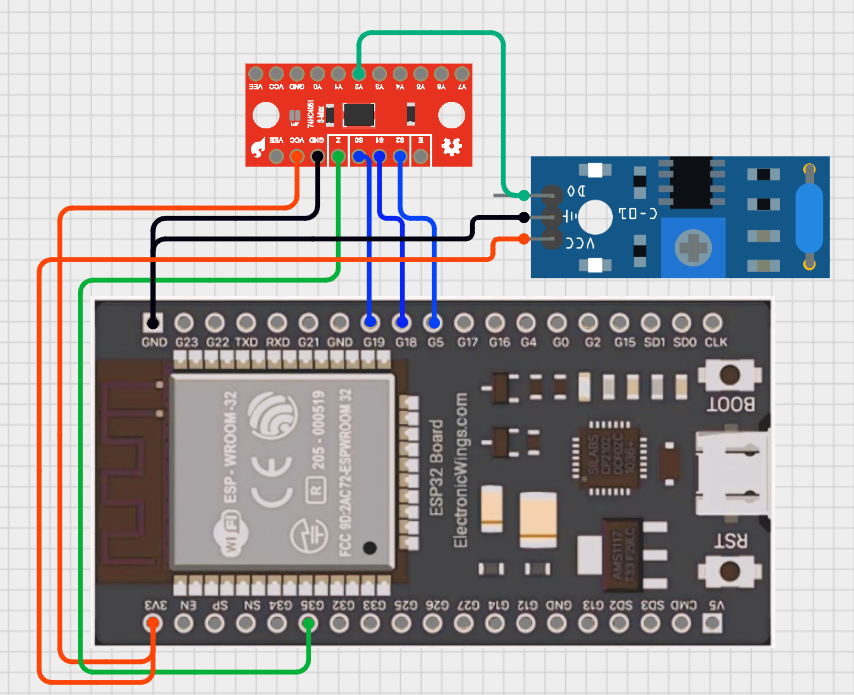
1. Capteur de Flamme (KY026):

J’ai câblé le capteur de flamme KY026 via le multiplexeur CJMCU sur le canal 1, et j’utilise une alimentation de 5 V via VCC et GND. Je détecte une flamme avec un seuil inversé de 2000.



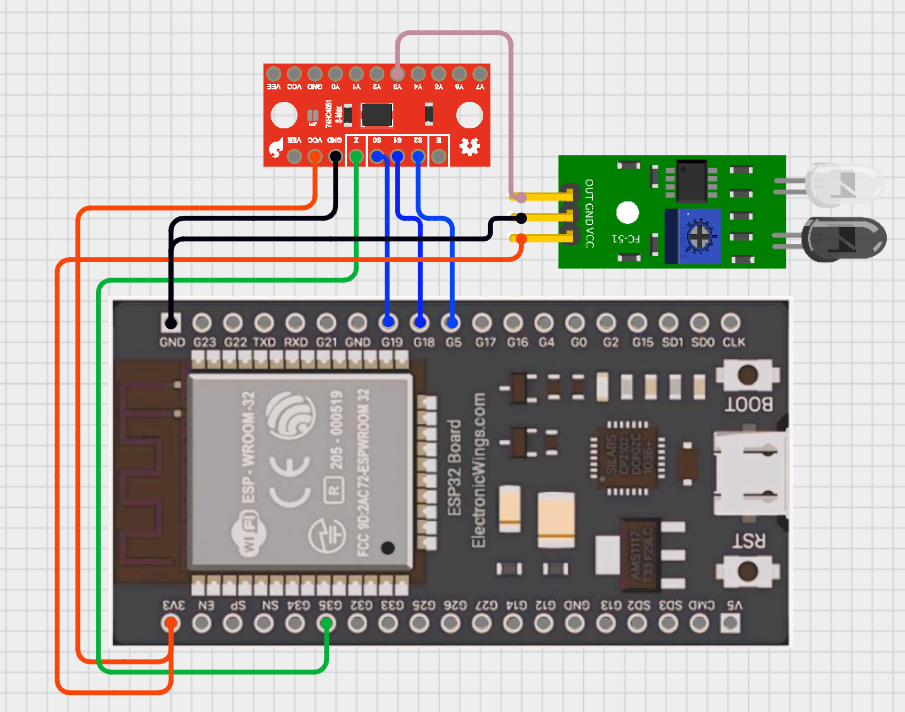
1. Capteur de Vibration (SW420) :

J’ai câblé le capteur de vibration SW420 via le multiplexeur CJMCU sur le canal 2, et j’utilise une alimentation de 5 V via VCC et GND. Je détecte les vibrations avec un seuil de 700.



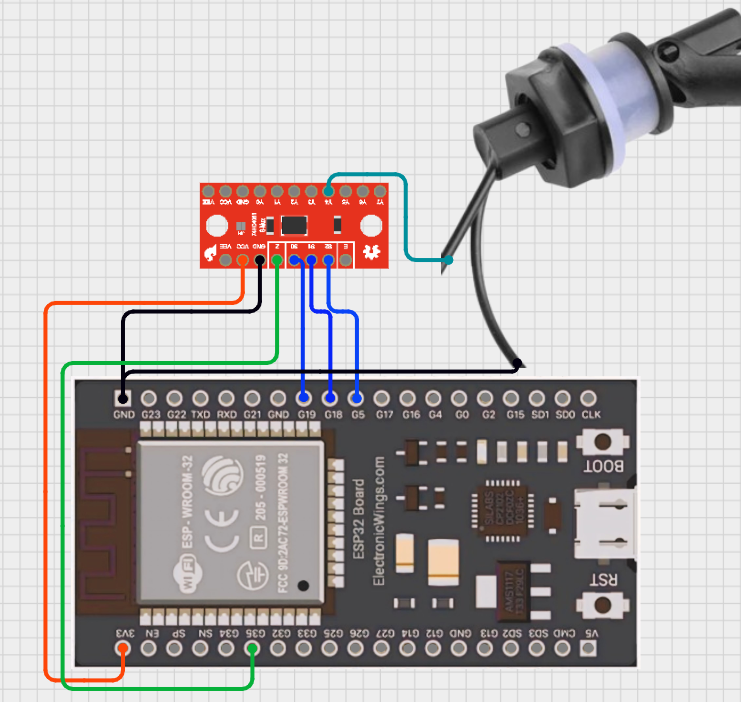
1. Capteur d’Obstacle (FC51) :

J’ai câblé le capteur d’obstacle FC51 via le multiplexeur CJMCU sur le canal 3, et j’utilise une alimentation de 5 V via VCC et GND. Je détecte les obstacles avec un seuil inversé de 2000.



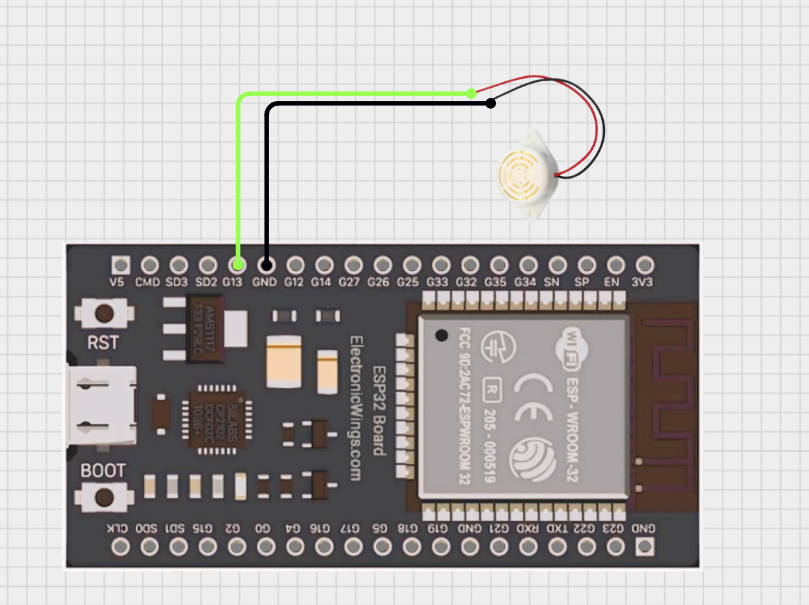
1. Capteur de Niveau :

J’ai câblé le capteur de niveau via le multiplexeur CJMCU sur le canal 4, et j’utilise une alimentation de 5 V via VCC et GND. Je détecte le niveau de liquide avec un seuil de 100.



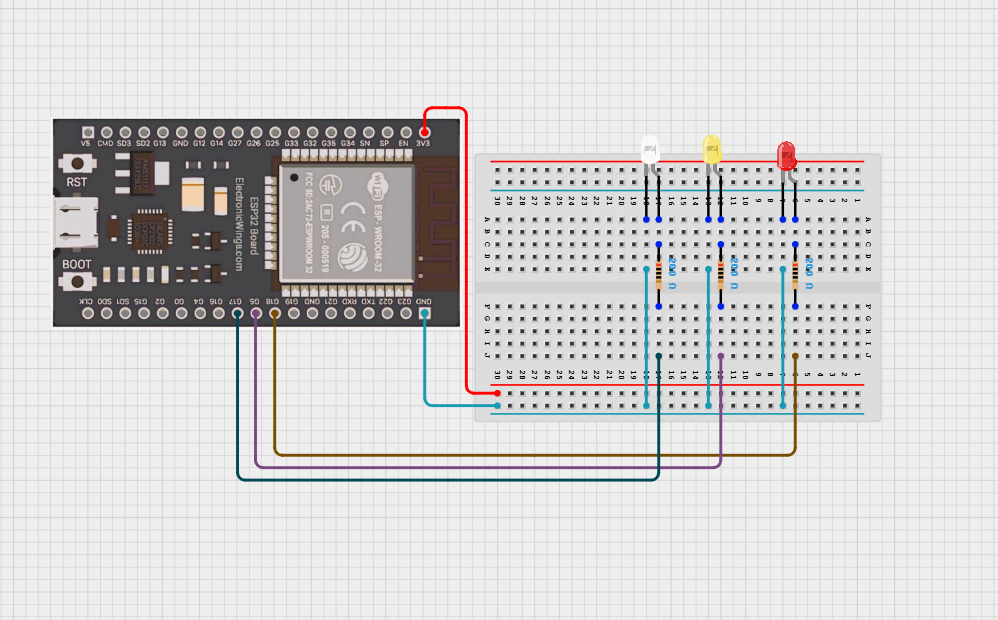
1. Buzzer :

J’ai câblé le buzzer à la broche GPIO15 en mode PWM pour contrôler la fréquence et l’intensité sonore (0 à 2000 Hz). J’ai connecté les broches VCC et GND à une alimentation de 5 V, et j’utilise ce montage pour activer différents niveaux d’alerte (0 à 3) selon la gravité.



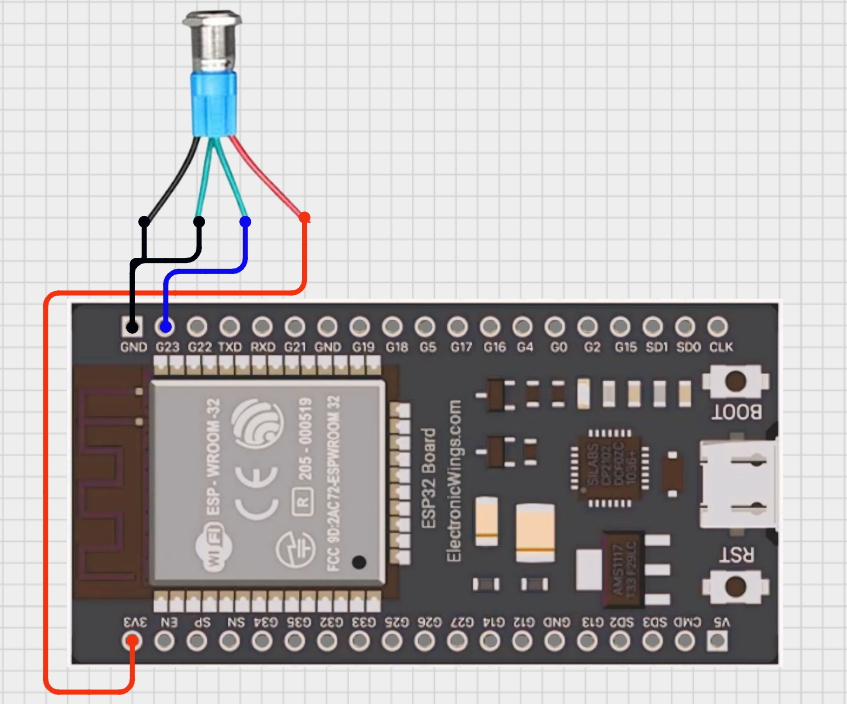
1. LEDS :

J’ai câblé les LED comme suit : la LED verte à GPIO21, la LED jaune à GPIO4 et la LED rouge à GPIO2, toutes en mode sortie. J’ai ajouté une résistance appropriée entre chaque broche et l’alimentation de 5 V, avec la masse (GND) commune, et j’utilise ces LED pour indiquer les niveaux d’alerte (vert, jaune, rouge).



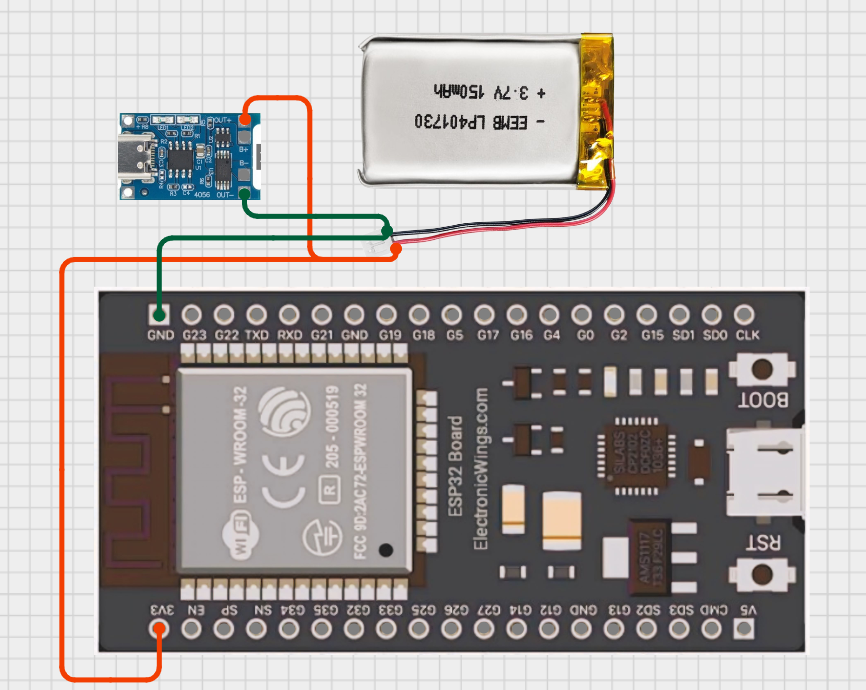
1. Bouton Poussoir :

J’ai câblé le bouton poussoir à la broche GPIO22 en mode entrée avec résistance de pull-up. J’ai connecté les broches à l’alimentation de 5 V et à la masse (GND) via le bouton, et j’utilise cette configuration pour activer ou désactiver manuellement le système IoT.



1. Module de Charge :

J’ai câblé le module de charge pour alimenter le système en le connectant aux broches VCC et GND de la carte 5 V. J’utilise une alimentation externe (batterie) pour assurer une source stable d’énergie à tous les composants

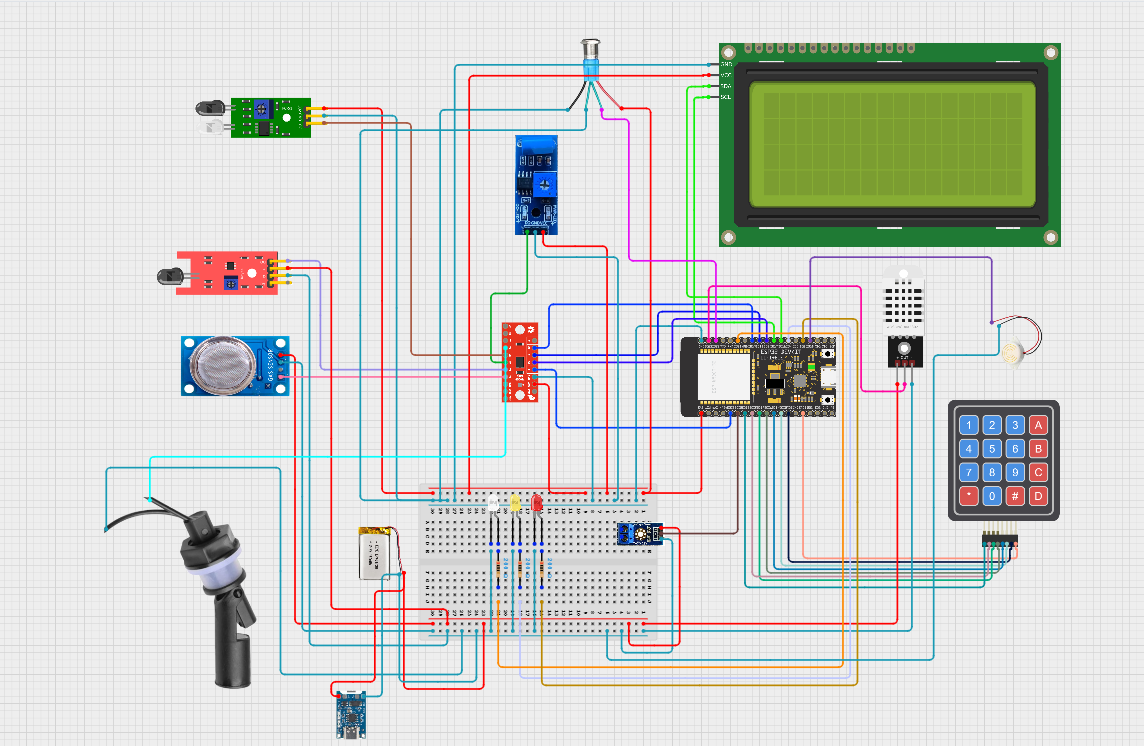


1. Câblage Complet :

Bouton Poussoir

Ecran LCD

SW-420



M. Charge

C . Niveau

Clavier

Buzzer

DHT22

ESP32

C. Volt

CJMCU-4051

MQ5

KY-026

FC-51

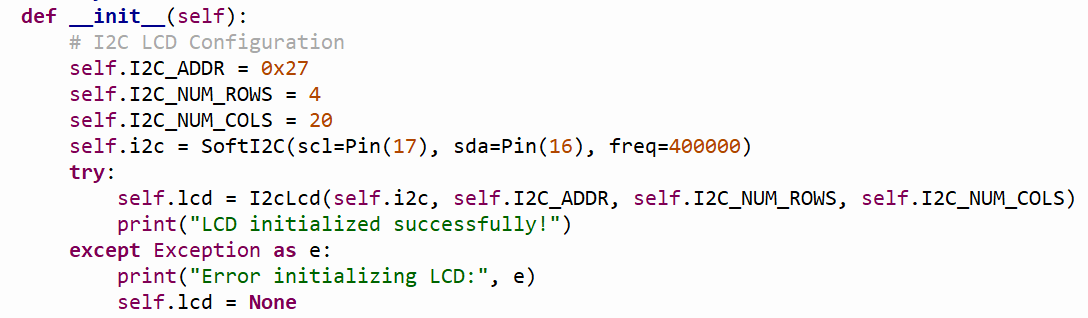
M. Charge

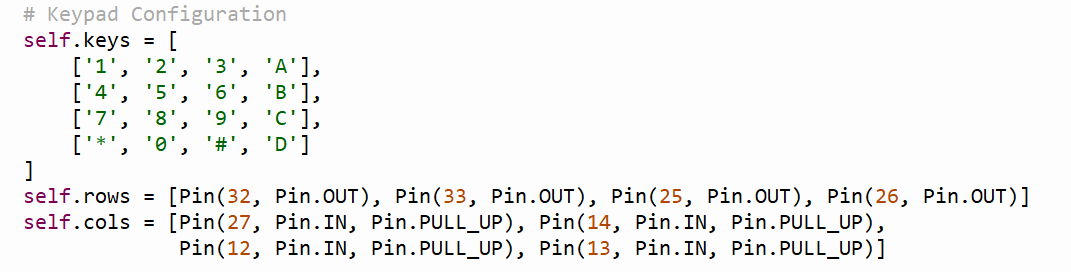
1. PROGRAMMATION

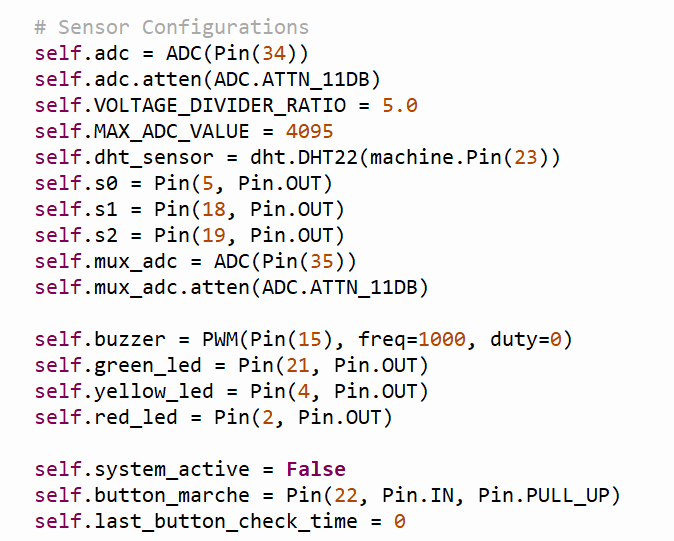
Le développement du système IoT a nécessité une programmation détaillée en Python, utilisant la plateforme Micro Python pour contrôler les capteurs, l’affichage, les alertes et la connectivité. Voici les principales parties du code que j’ai implémentées :

1. Initialisation du System :

j’associe le clavier matriciel constitué d’une matrice 4x4 sur les GPIO32-33 et GPIO27-14, puis les capteurs : DHT22 sur GPIO23, tension sur GPIO34, multiplexer sur GPIO35, puis le buzzer sur GPIO15, les LED fonctionnant en mode entrée/sortie, hum… la LED sur le GPIO21, la LED sur le GPIO4, la LED sur le GPIO2, le bouton sur le GPIO22. Cette phase est fondamentale, car on jette les bases du fonctionnement du cadre dans lequel on va faire fonctionner le reste de la plate-forme, ce qui implique que tous les composants soient correctement mis en service et prodiguent les longues séries de passionnantes frustrations de recoupement d’erreurs potentielles dans les développements ultérieurs jusqu’à ce que l’écran LCD fonctionne convenablement, je ne dis pas que le reste a lui-aussi de petites désillusions contagieuses de bas niveau mais bon au moins on s’attend à comprendre autrement la suite.

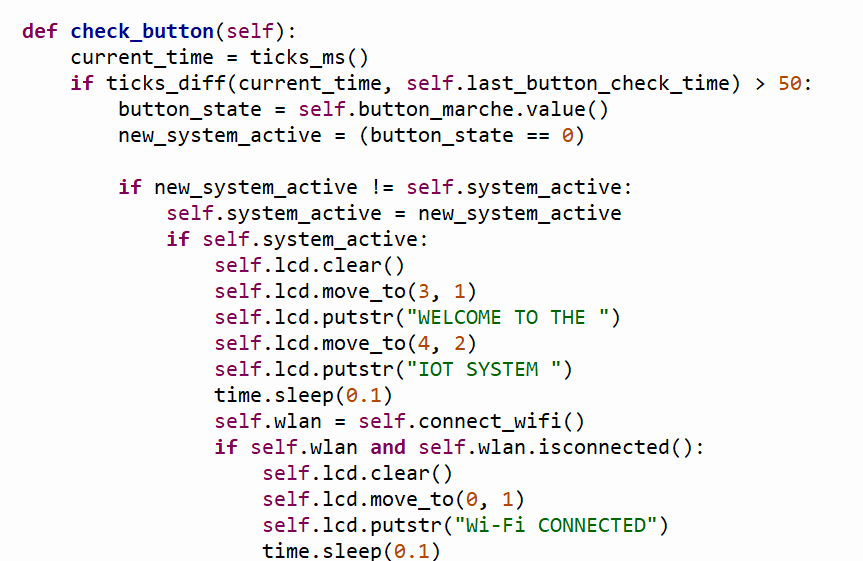


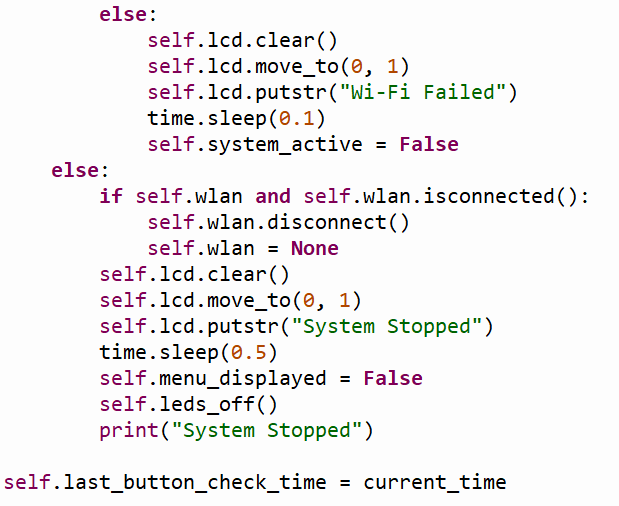




1. Gestion du Bouton Poussoir :

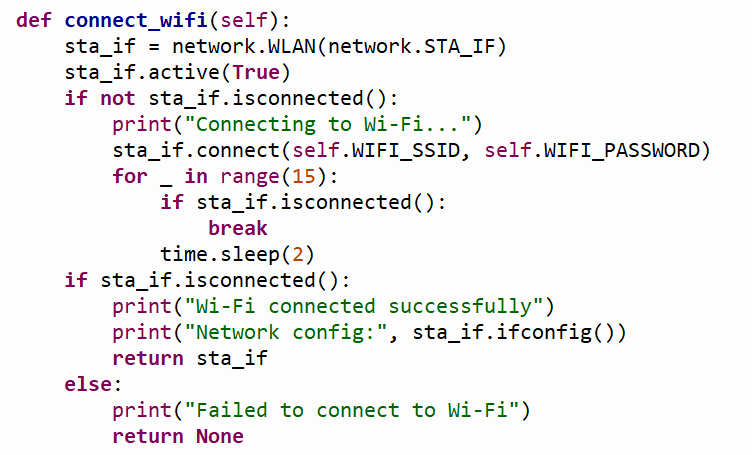
La fonction **check\_button()** est censée surveiller le bouton connecté à GP22 20 fois par seconde. S’il est pressé (valeur basse), je mets system\_active à 1 pour afficher « **WELCOME TO THE IOT SYSTEM** » avant d’appeler **connect\_wifi()**. Si ce dernier réussit, le message « **Wi-Fi CONNECTED** » apparaît ; sinon c’est « **Wi-Fi Failed** », puis le système s’arrête après avoir déconnecté le Wi-Fi et affiché « **System Stopped** ». Cette fonction permet au système d’être piloté manuellement avec une meilleure transparence.





1. Connection WIFI :

J’ai programmé **connect\_wifi()** pour activer le Wi-Fi et se connecter au réseau "WIFI\_SSID" avec le mot de passe "WIFI\_PASSWORD", tentant pendant 15 secondes. Si réussi, j’affiche les détails de configuration ; sinon, je signale une erreur. Cette fonction est cruciale pour établir la connectivité, base de la transmission des données.



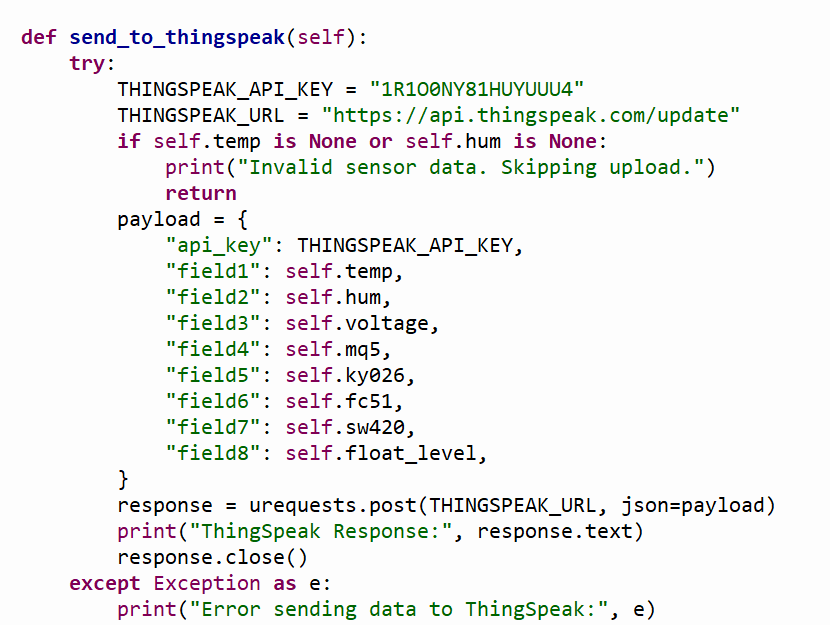
1. Lecture des capteurs :

J'ai implémenté dans le programme la fonction **read\_dht22()** pour effectuer les mesures de température et d'humidité au moyen du GPIO23, **read\_voltage()** pour convertir les valeurs de l'ADC du GPIO34 en tensions avec un diviseur de 5:1 et **select\_channel()** pour le contrôle du multiplexeur (GPIO5, 18, 19). J'ai conçu aussi **read\_digital\_channel()** et **read\_digital\_channel\_2()** pour la lecture des capteurs au GPIO35, en intégrant des lectures lues sur 5 mesures différant par des seuils (700, 2000, 100), la seconde inverse la logique. L'appel à **read\_all\_sensors()** est prévu de façon à succéder à la récupération des valeurs par messagerie toutes les 5 secondes, tout en affichant la mise à jour pour le débogage. Cette architecture étant essentielle à la robustesse de la supervision.

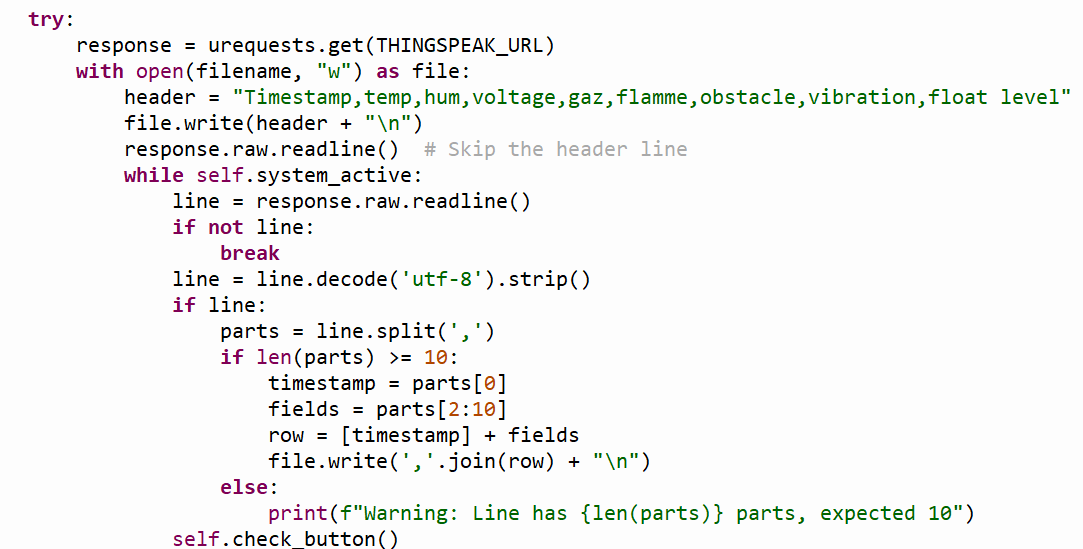


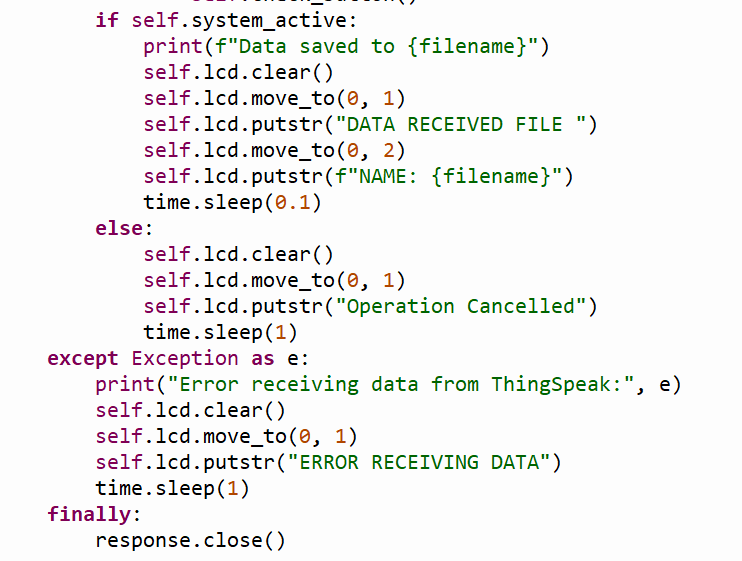
1. Envoi et Réception de Données :

J’ai programmé **send\_to\_thingspeak()** pour envoyer les données des capteurs à ThingSpeak toutes les 60 secondes avec la clé "1R1O0NY81HUYUUU4", vérifiant leur validité pour éviter les erreurs. J’ai aussi implémenté **receive\_from\_thingspeak()** pour télécharger des données historiques avec des dates ou les 100 dernières entrées, les sauvegardant dans un fichier CSV. Ces fonctions sont essentielles pour la connectivité et l’analyse des données, avec des messages LCD comme "**DATA RECEIVED FILE**" pour la rétroaction.



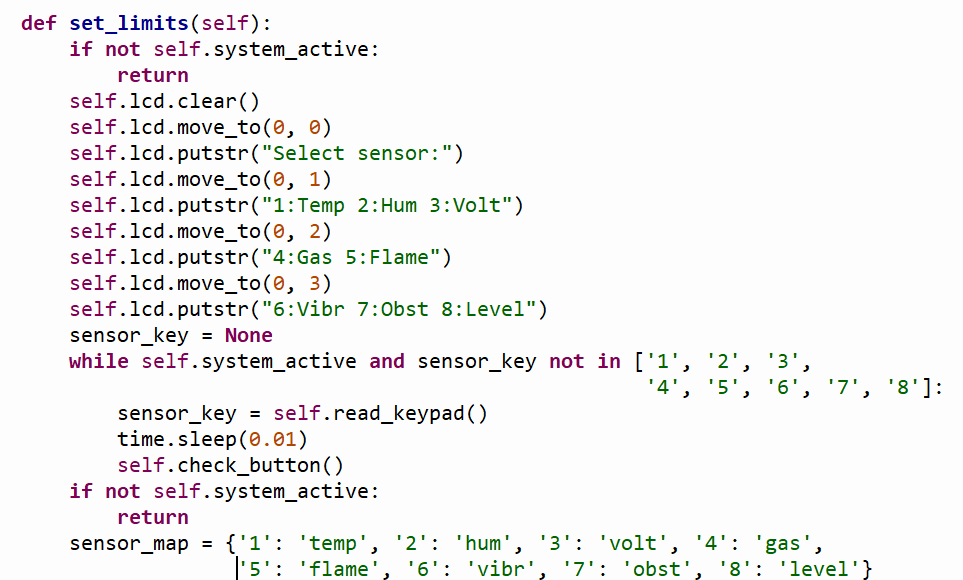


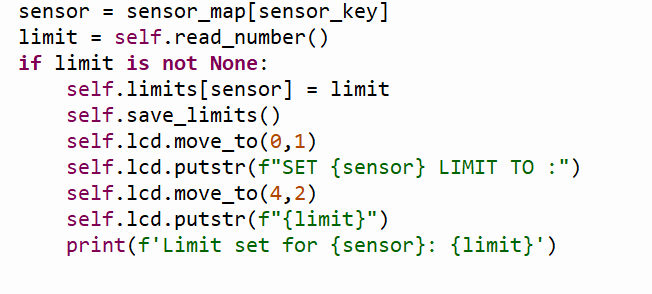




1. Définitions des Limites :

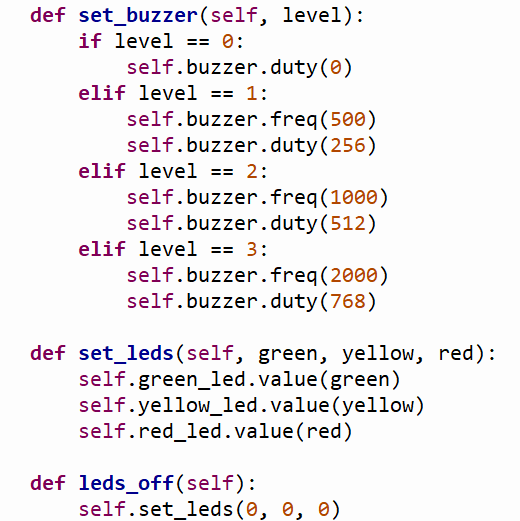
Sous la responsabilité de la fonction **save\_limits(),** les limites sont donc stockées dans limits.json. La fonction **read\_number()** permet d’entrer un nombre au moyen du clavier, où l’on peut supprimer (\*) et confirmer (#). La fonction **set\_limits()**, pour sélectionner un capteur (1 à 8) et définir son seuil, affiché à l’écran. Ces trois fonctions sont accessibles par l’option "SET LIMIT" sur le LCD, permettant ainsi d’ajuster les alertes : une fonctionnalité indispensable.





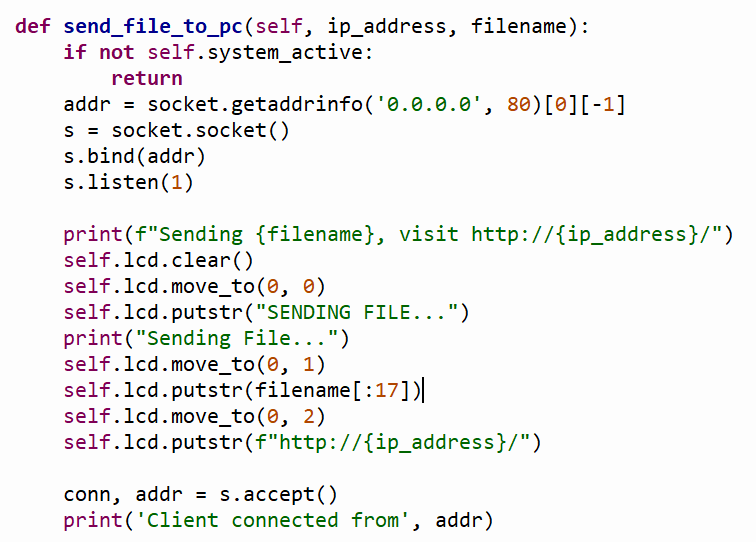
1. Gestion LEDs et Buzzer :

J’ai programmé **set\_buzzer()** pour contrôler le buzzer sur GPIO15 avec des fréquences et intensités croissantes (0 à 2000 Hz) selon les niveaux d’alerte (0 à 3), et **set\_leds()/leds\_off()** pour allumer/éteindre les LEDs. Ces fonctions sont clés pour signaler visuellement et sonores les alertes, améliorant la sécurité.



1. Envoi fichiers au PC :

Sur commande dans l’exposé, j’ai programmé **send\_file\_to\_pc()** pour recevoir un fichier CSV sélectionné via option «SEND FILE » au bon moment en chargeant un serveur local sur le port 80, à partir duquel « **SENDING FILE…** » est affiché avec l’URL (ex. http://192.168.x.x/). Le fichier est alors envoyé ; dès lors, il est supprimé et « **FILE SENT & DELETED** » apparait. Cette fonction contribue à partager les données avec un PC et ainsi à la flexibilité du système.







1. Boucle Principale :

Je me suis attelé à la programmation de **main\_loop()** primaire et ai fait appel à **check\_button()**, **read\_all\_sensors()**, **check\_limits\_and\_alert()** tous les 5 secondes ainsi que **send\_to\_thingspeak()** toutes les 60 secondes s’il est connecté. J’ai aussi créé un menu sur l’écran LCD avec « 1 : SET LIMIT » pour paramétrer les seuils, « 2 : RECEIVE DATA » pour télécharger des données historiques, « 3 : SENSORS OPTIONS » pour lire les données ou limites et « 4 : SEND FILE » pour envoyer un fichier CSV. Ces options étaient sous la compétence du clavier, rendant le système interactif et personnalisable. Or cette souplesse dans son utilisation comme un véritable outil constitue une fonctionnalité capitale pour son exploitation.

