REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DR. YAHIA FARES DE MEDEA

FACULTE DE TECHNOLOGIE



Faculté de Technologie

Mémoire de fin d'étude
En vue de l'obtention de diplôme de Master
Filière : Génie électrique

Spécialité : Mécatronique

Conception et réalisation d'un système de contrôle et supervision de la température d'un four géré par ESP8266 et Protocol MQTT

PRÉSENTÉ PAR :

- ✓ Gheribi Ahmed
- ✓ Kermezli Abdeldjalil

PROPOSE ET DIRIGÉ PAR :

Rebhi Redha

Promotion : 2021/2022

REMERCIEN

Toute notre gratitude, grâce et remerciement à Dieu le tout puissant qui nous a donné sa foi, la patience et la sagesse.

Nous remercions les membres de jury qui nous font l'honneur de présider et d'examiner ce modeste travail.

A nos enseignant, docteur et encadreur : Mr : rabhi redha qui nous a donné l'éducation, la science et le savoir.

A nos mères qui nous porté et soutenu durant toute notre vie, à nos pères qu'ont veillé sur nous de jour comme de nuit.

A nos M. mahfoud zouaouine et copains et copines, nos cousins et cousines, et surtout nos frères et sœurs pour nous avoir acoquiné et nous donner le courage à continuer et préserver.

Nous adressons de chaleureux remerciements à tous les enseignants qui on contribue à notre réussite dans nos études, depuis notre première année scolaire jusqu'à la dernière année universitaire.

Enfin, Nous remercions cette mémoire, qui sans leur aide, leur soutien et leurs conseils nous ne serons pas arrivé à ce stade et peut-être peu d'avantages.

Gheribi et kermezli

Dédicace

Je dédie ce mémoire à ma mère et à mon père, A mes frères et ma sœur, et mes amis.

A tous mes oncles et tantes, cousins et cousines. A mes collègues de travail.

A tous mes amis que je n'ai pas cités et à tous ceux qui me connaissent, les étudiants de ma promotion Master2 2022 mécatronique de Collège des sciences et de la technologie de l'université Yahia Fares Medea.

A tous qui m'aidaient pour terminer ce travail.

A la fin, je veux dédier ce modeste travail à tous les enseignants et enseignantes qu'ils m'ont aider à arriver ici depuis le début de mes études.

Ahmed Gheribi

Dédicace

Je dédie ce mémoire à ma mère et à mon père, A mes frères et mes sœurs,

A tous mes oncles et tantes, cousins et cousines. A mes collègues de travail.

A tous mes amis que je n'ai pas cités et à tous ceux qui me connaissent, les étudiants de ma promotion Master2 2022 mécatronique de Collège des sciences et de la technologie de l'université Yahia Fares Medea.

A tous qui m'aidaient pour terminer ce travail.

A la fin, je veux dédier ce modeste travail à tous les enseignants et enseignantes qu'ils m'ont aider à arriver ici depuis le début de mes études.

Abdeldjalil Kermezli

RESUME

L’internet a connu, depuis son apparition, un grand essor et a touché pratiquement à tous les domaines de notre vie de tous les jours : politique, économique et socioculturel, et donc il est devenu une principale source de l’information.

L’internet classique tel que nous le connaissons, à savoir l’accès à une multitude de site web à travers d’un navigateur, repose sur plusieurs protocoles. Un premier protocole utilisé est HTTP (Hyper Texte Transfer Protocol). Il s’agit du protocole de couche applicative utilisé pour la communication web. Comme son nom l’indique, son rôle est de transférer des données hypertextes, du serveur jusqu’au client final qui souhaite cette ressource. Donc, l'internet classique consiste principalement à l’internet des données.

Le IoT (Internet of Things) est plus vaste et plus varié puisqu’elle exploite des innovante technologies et des normes que l'internet classique. L’apparition de l’IoT offre un grand potentiel pour le développement de nouveaux services et d’applications immergeant le monde physique et le monde virtuel, une perspective qui n'était pas possible auparavant.

De nombreuses applications de l’Internet des Objets mènent des projets pour améliorer la vie des gents grâce aux objets et infrastructures connectés. La ville de futur s'organise autour de plusieurs contraintes : environnement, contraintes énergétiques, collaboration entre les gents, les collectivités et les opérateurs de la ville, développement des modèles économiques orientés sur les usagés et l’intégration numérique dans la ville.

Ce projet vise à la mise en œuvre d’une application d’IoT sur la façon d'utiliser MQTT comme protocole de communication pour le déploiement d'applications IoT et la gestion à distance de ces applications, de sorte qu'il peut fonctionner dans toutes les conditions du réseau.

L’objectif de cette application est l’implantation de protocole MQTT dans les applications IOT et prendre en compte les valeurs transmises par les différents dispositifs situés dans l'environnement (capteur de température), et visualiser ces données a distance sur un l'ordinateur.

Mots clés : Internet of Things, ordinateur, IoT, MQTT.

ملخص

لقد عرف الإنترنت ، منذ ظهوره ، طفرة كبيرة وطال عمليا جميع مجالات حياتنا اليومية: السياسية والاقتصادية والاجتماعية والثقافية ، وبالتالي أصبح مصدرًا رئيسيًا للمعلومات.

يعتمد الإنترنت الكلاسيكي كما نعرفه ، أي الوصول إلى العديد من مواقع الويب من خلال متصفح ، على عدة بروتوكولات. أول بروتوكول مستخدم هو HTTP (Hyper Text Transfer Protocol). إنه بروتوكول طبقة التطبيق المستخدم للاتصال عبر الويب. كما يوحي اسمها ، يتمثل دورها في نقل بيانات النص التشعبي ، من الخادم إلى العميل النهائي الذي يريد هذا المورد. لذا ، فإن الإنترنت الكلاسيكي يتكون أساسًا من إنترنت البيانات.

إنترنت الأشياء (IoT) هو أوسع وأكثر تنوعًا لأنه يستخدم تقنيات ومعايير مبتكرة من الإنترنت التقليدي. يوفر ظهور إنترنت الأشياء إمكانات كبيرة لتطوير خدمات وتطبيقات جديدة تغمر العالم المادي والعالم الافتراضي ، وهو احتمال لم يكن ممكنًا من قبل.

تقود العديد من تطبيقات إنترنت الأشياء مشاريع لتحسين حياة الناس من خلال الكائنات والبنية التحتية المتصلة. يتم تنظيم مدينة المستقبل حول عدة قيود: البيئة ، وقيود الطاقة ، والتعاون بين الناس والمجتمعات ومشغلي المدينة ، وتطوير نماذج اقتصادية موجهة للمستخدم والتكامل الرقمي في المدينة.

يهدف هذا المشروع إلى تنفيذ تطبيق إنترنت الأشياء حول كيفية استخدام MQTT كبروتوكول اتصال لنشر تطبيقات إنترنت الأشياء والإدارة عن بُعد لهذه التطبيقات ، بحيث يمكن أن تعمل في جميع ظروف الشبكة.

الهدف من هذا التطبيق هو تنفيذ بروتوكول MQTT في تطبيقات إنترنت الأشياء ومراعاة القيم التي تنقلها الأجهزة المختلفة الموجودة في البيئة (مستشعر درجة الحرارة) ، وتصور هذه البيانات عن بُعد على جهاز كمبيوتر.

الكلمات الرئيسية: إنترنت الأشياء ، الكمبيوتر ، إنترنت الأشياء ، MQTT.

Abstract

The Internet has known, since its appearance, a great boom and has touched practically all areas of our daily life: political, economic and socio-cultural, and therefore it has become a main source of information.

The classic Internet as we know it, namely access to a multitude of websites through a browser, is based on several protocols. A first protocol used is HTTP (Hyper Text Transfer Protocol). It is the application layer protocol used for web communication. As its name suggests, its role is to transfer hypertext data, from the server to the end customer who wants this resource. So, the classic internet consists mainly of the internet of data.

The IoT (Internet of Things) is broader and more varied since it uses innovative technologies and standards than the traditional Internet. The emergence of IoT offers great potential for the development of new services and applications immersing the physical world and the virtual world, a prospect that was not possible before.

Many Internet of Things applications are leading projects to improve people's lives through connected objects and infrastructure. The city of the future is organized around several constraints: environment, energy constraints, collaboration between people, communities and city operators, development of user-oriented economic models and digital integration in the city.

This project aims at the implementation of an IoT application on how to use MQTT as a communication protocol for the deployment of IoT applications and the remote management of these applications, so that it can work in all network conditions.

The objective of this application is the implementation of MQTT protocol in IOT applications and to take into account the values transmitted by the various devices located in the environment (temperature sensor), and to visualize these data remotely on a computer.

Keywords: Internet of Things, computer, IoT, MQTT.

Table des matières

Introduction générale 1

CHAPITRE I : GENERALITE SUR L'IOT 1

I-Introduction : 3

I-1- Définition de l'internet des objets : 3

I-2- L'IoT aujourd'hui : 4

I-3- Domaine d'application des systèmes d'IOT : 6

I-3-1- Les Villes Intelligentes : 6

I-3-2- La santé intelligente : 7

I-3-3- L'énergie intelligente : 8

I-3-4- L'environnement intelligent : 9

I-3-5- L'industrie intelligente : 9

I-3-6- Le Transport et La mobilité intelligente : 10

I-4- Architecture de l'IOT : 11

I-4-1- Les quatre couches de L'IOT : 12

I-4-2- Les sept niveaux du modèle référence pour l'IoT : 12

I-5- Les Défis del'IOT : 18

I-5-1- La Découverte Automatique : 18

I-5-2- L'Interopérabilité : 19

I-5-3- La Sécurité et La Confidentialité : 19

I-5-4- La Tolérance Aux Pannes : 19

I-5-5- L'Auto-Organisation : 19

I-6- Les bénéfices de l'IoT : 19

I-7- Protocole de communication : 20

I-7-2 Le protocole MQTT : 21

Conclusion : 23

Chapitre II : Protocol MQTT 3

II-1-Le protocole MQTT : 24

II-1-1- Définition : 24

II-1-2- Mode de fonctionnement MQTT : 24

..... 25

II-1-3- Abonnements et Publications : 25

1. **SUBSCRIBE** 25
2. **UNSUBSCRIBE** 25
3. **PUBLISH** 26

II-1-4 Les topics : 26
II-1-5- Qualité de service : 26
1. QoS 0 « Livraison une fois maximum ». 26
2- QoS 1 « Livraison au moins une fois ». 26
3-QoS 2 « Livraison exactement une fois ». 27
III-1-6- Le broker MQTT : 28
1-Publier (envoyer) un message MQTT depuis le terminal : 28
2-Recevoir (souscrire) des messages : 28
II-1-7- Sécurité : 28
-L'identification....28
-L'authentification. 29
-L'autorisation....29
II-2Arduino : 29
II.2.1 Introduction : 29
II-2-2 Le logiciel : 32
II-2-2-1 L'interface : 33
II-2-3 Le langage Arduino : 35
II.3 Node-red : 37
II-3-1 Installation : 38
II-3-2 interface : 39
II-3-3 Ajout d'une catégorie de nodes à la palette : 40
II-3-4 Principe de la programmation : 41
Les composants utilisés 42
III.1 Introduction : 43
III-2 Esp8266 NodeMCU : 43
III-2-1- Définition : 43
III-2-2 L'ESP8266 peut se programmer de plusieurs façons : 44
III-2-3 Description technique d'ESP8266 44
III-2-4- Les différents composant d'ESP8266 : 45
III-2-5- Support de carte NodeMCU : 45
III-2-6 Schéma interne d'ESP8266 : 46
III-2-7 Les broches de l'ESP8266 12E : 46
III-2-8 Les avantages d'ESP8266 : 47
III-2-9 Comparaison entre Arduino_Uno et ESP8266 12E: 47
III-2-10 Installation _ ESP8266_sur IDE Arduino : 47

III-3 Le relais électromécanique : 52
III-3-1 Constitution : 53
III-3-2 Fonctionnement III-3-2-1 Fonctionnement monostable : 54
III-3-2-2Fonctionnement bistable à une bobine 55
III-3-2-3Fonctionnement bistable à deux bobines : 55
III-3-3 Inconvénients du relais électromécanique : 55
III-4 La résistance : 56
III-5 Le capteur LM35 : 57
III-5-1 Définition du LM35 : 58
III-5-2 Calibration du lm35 : 60
Chapitre IV 61
IV.1 Introduction : 61
IV.2 Les étapes de la réalisation : 61
IV.3 Etude technique : 61
IV.3.1 Principe et mode de fonctionnement : 61
IV.3.2 Outils de programmations : 61
IV.4 Schéma synoptique de système : 63
IV.4.1 Schéma Bloc du client MQTT1 : 64
IV. 5 Schéma électrique de montage : 65
IV.6 Régulation et Contrôle de la température : 66
IV.7 Etude software : 67
IV.7.1 Programmation Node-red : 67
IV.7.2 Programmation Arduino : 68
IV.8 Quelques photos de la réalisation : 72
IV.8.1 Câblage et montage : 72
IV.8.2 Bloc de Client MQTT1 : 72
IV.8.3 Bloc de commande : 73
IV.8.4 Bloc Heater: 73
IV.8.5 Bloc Fan: 74
IV.8.6 Bloc de client MQTT2 : 74
IV.8.7 Bloc d'alimentation : 75
Conclusion : 76
Conclusion générale : 78
Liste des références bibliographiques : 79
Annex: 81

LISTE DES FIGURES :

Figure I-1 Internet des objets……..4

Figure I-2 L'évolution des objets connecté……5

Figure I-3 Les constituants d'une ville intelligente……... 6

Figure I-4 Le mécanisme de la surveillance distants des patients……..7

Figure I-5 Les constituants d'une Smart Grid……8

Figure I-6 Les constituants de l'environnement intelligent……9

Figure I-7 Les constituants de l'industrie intelligente……..10

Figure I-8 Quelques aspects du transport intelligent……11

Figure I-9 Le Modèle de référence de L'IoT……13

Figure II-1 Modèle pour transférer des données entre clients (publish / subscribe) à travers le broker……25

Figure II-2 Niveaux de Qualité de Service MQTT……..27

Figure II-3 Arduino UNO……30

Figure II-4 interface du programme Arduino……...30

Figure II-5 Carte (Uno et Duemilanove) et (Carte Mega)……31

Figure II-6 interface de Arduino IDE……33

Figure II-7 Le menu File de Arduino……34

Figure II-8 Les boutons d'action……35

Figure II.8 Node-red logo……..37

Figure II.9 Node-red fonctionne sur tout type de systèmes……38

Figure II.10 Sous linux……...……38

Figure II.11 Sous Windows……38

Figure II.12 Execution de node-red dans le terminal……..39

Figure II.13 IHM de node-red……..39

Figure II.14 Accéder au gestionnaire de palette……..40

Figure II.15 Ajout de la catégorie Dashboard à la Palette……..40

Figure II.16 Jauge pilotée par un slider……..41

Figure II.17 Ajout d'un node Numeric……...42

Figure II.18 Ajout d'un node Chart pour obtenir un graph de l'historique……42

Figure III-1 : Réalisation de système…….43

Figure III-2 La carte NodeMCU (ESP8266)…….44

Figure III-3 Les différents composant d'esp8266…….45

Figure III-4 Support de carte NodeMCU……..45

Figure III-5 Le Support avec la carte NodeMCU…….45

Figure III-6 Composition d'une carte NodeMCU (ESP8266)…….46

Figure III-7 Les broches de l'ESP8266 12E…….46

FigureIII.8 Comparaison entre Arduino_Uno et ESP8266 12E……47

Figure III.9 Schéma de Télèverserensuite le code Blink sur notre ESP8266……48

Figur III.10 : composants de relais…….53

Figur III.11 Le système de commutation……53

Figur III.12 : SUMBOLES DE RELAIS……..…….54

Figur III.13 : Schéma commandé et non commandé de relais……..…….54

Figur III.14 : 2 petites animations de relais……..……..55

Figure III.15 : Symbole d'une résistance électrique……..…….56

Figure III.16 : Résistance a 4 anneaux ……..…….57

Figure III.17 : Capteur LM35……..……57

Figure III.18 : brochage du lm35…….……59

Figure III.19 : calibration du lm35……60

FigureIV.1 : node-red……62

FigureIV.2 : Schéma synoptique de système……63

Figure IV.3 : Schéma Bloc du client MQTT1……...64

FigureIV.4 : Schéma électrique……65

FigureIV.5 : Régulation et Contrôle de la température……66

Fig IV.5 : Photo de la réalisation……72

Fig IV.6 : Bloc de Client MQTT1……..72

Fig IV.7 : Bloc Commande……73

Fig IV.8 : Bloc Heater……73

Fig IV.9 : Bloc Fan……74

Fig IV.10 : Bloc de Client MQTT2 (Node-red)……74

Fig IV11 : Bloc de Client MQTT2 (Node-red Dashboard)……75

Fig IV.12 : Bloc d'Alimentation……75

Introduction générale

L'Internet est devenu en quelques années le vecteur principal de diffusion de l'information. Il s'est imposé dans de nombreux domaines comme une infrastructure essentielle pour les individus, les entreprises et les institutions. Toutefois, ses capacités d'extension, au-delà des seuls ordinateurs et terminaux mobiles, sont encore considérables car il devrait permettre l'interaction d'un nombre croissant d'objets entre eux ou avec nous-mêmes.

Internet transforme progressivement en un réseau étendu, appelé « Internet des Objets », reliant plusieurs milliards d'êtres humains mais aussi des dizaines de milliards d'objets [1].

Mais l'image que projette IoT de demain est beaucoup plus vaste que celle d'aujourd'hui. On parle plutôt d'objets en mode Plug and Play, capables de se découvrir et de communiquer entre eux. Et de réseaux d'objets connectés qui embarquent de l'intelligence qui est capables de prendre des décisions et de faire des actions de façon autonome [2].

De toute évidence, Internet est l'un des plus importants et des plus puissantes créations dans toute l'histoire humaine.

Ce travail consiste à réaliser une plateforme IoT permettant de suivre la consommation d'énergie dans des locaux pour habitation ou entreprise. Le système, permet de suivre la consommation d'énergie électrique en temps réels, et permet à long terme d'avoir un profil de consommation personnalisé. Ce profil sert à prédire les consommations futures et détecter d'éventuelles surconsommations.

Le système est en mesure de renseigner l'utilisateur sur les horaires de pics de consommations, et ainsi l'aider à mieux optimiser ses consommations.

La plateforme est constituée deux parties, la partie client (locaux avec les capteurs intelligents) et la partie serveur qui se charge de collecter les informations issues des capteurs du client.

Nous allons présenter notre travail dans trois chapitres :

Chapitre 1 : est consacré à définir l'internet des objets, ses différents domaines d'applications, nous expliquons l'architecture de l'IoT (les quatre couches et les sept niveaux du modèle référence), et nous présentons les protocoles et connectivité IoT et ainsi que le Protocol MQTT, et nous mentionnons les défis et les bénéfices de l'IoT.

Chapitre 2 : Dans ce chapitre, nous présentonsle protocole MQTT et expliquons le fonctionnement des programmes Arduino et Node-red

Chapitre 3: Dans ce chapitre, nous avons expliqué tous les composants utilisés dans notre projet et leur fonctionnement, nous avons également expliqué le principe du composant principal ESP8266 et comment l'installer dans le programme Arduino et nous l'avons testé

Chapitre 4 : et enfin nous présentons l'environnement de travail de notre projet, ainsi que les différentes étapes de la réalisation,Nous avons également expliqué notre programme Arduino et notre programme Node-red et fourni des photos de notre réalisation

A la fin, nous terminons notre mémoire par une conclusion générale.

CHAPITRE I :

GENERALITE SUR L'IOT

I-Introduction :

Ce chapitre sera consacré à l'étude théorique d'un système de l'internet des objets, nous Présentons l'historique de l'IoT et les défèrent domaine d'application, ensuite nous citerons l'architecture de l'IoT, ainsi que les défis et les bénéfices de l'IoT.

I-1- Définition de l'internet des objets :

Historiquement le terme IoT a été utilisé pour la première fois en 1999 par Kevin Ashton, un ingénieur britannique, pour décrire un système où les objets physiques sont connectés à internet.

Au fil du temps, le terme a évolué et il englobe maintenant tout l'écosystème des objets connectés. L'IoT commence ainsi dans le monde physique avec les capteurs qui recueillent les informations, elles sont ensuite transmises grâce à la connexion et l'intégration des systèmes entre eux, les données sont enfin traitées et stockées pour être analysées et exploitées.

On pense que l'IoT améliorera l'efficacité énergétique, la surveillance à distance et le contrôle des actifs physiques et la productivité grâce à des applications aussi diverses que la sécurité à domicile pour surveiller la condition sur le plancher de l'usine.

Maintenant, l'IoT a été utilisé sur les marchés dans le domaine des soins de santé à domicile appareils et bâtiments, marchés de détail, entreprises énergétiques et manufacturières, mobilité et le transport, les entreprises de logistique et les médias.

Les équipements sont de plus en plus numérisés et connectés, créant des réseaux entre les machines, les humains et Internet, conduisant à la création de nouveaux écosystèmes qui permettent une productivité plus élevée, une meilleure efficacité énergétique et une rentabilité plus élevée [3].

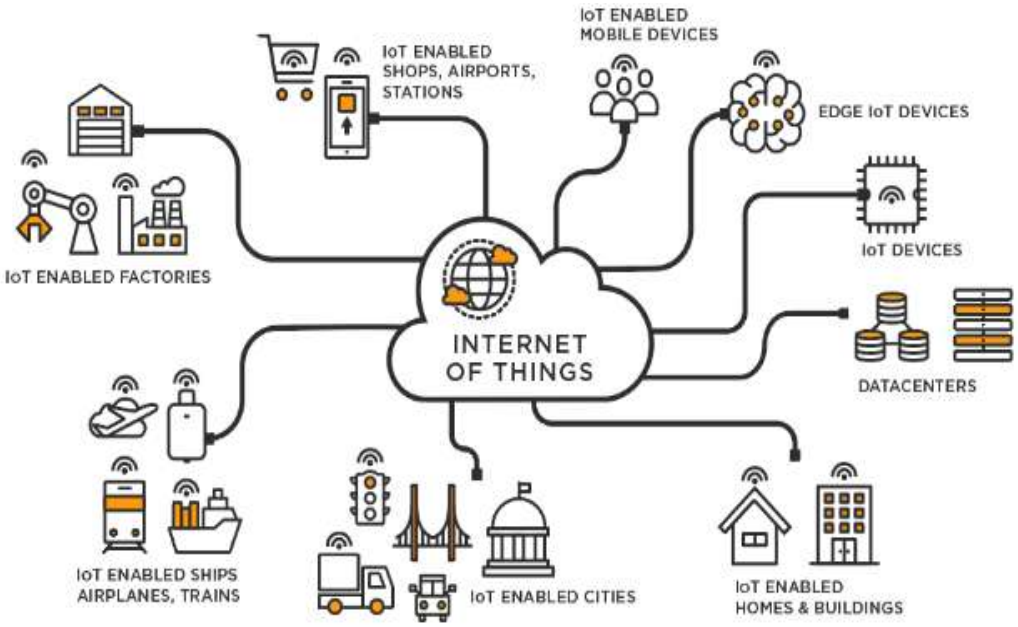


Figure I-1 Internet des objets

I-2- L'IoT aujourd'hui :

Avant de parler de l'état actuel de l'IoT, il est important de s'entendre sur une définition. Selon Cisco Internet Business Solutions Group (IBSG), l'IoT correspond simplement au moment où il y a eu plus « de choses ou d'objets » connectés à Internet que de personnes.

En 2003, il y avait environ 6,3 milliards de personnes vivant sur la planète et 500 millions appareils connectés à Internet. En divisant le nombre d'appareils connectés par la population mondiale, nous constatons qu'il y avait moins d'un (0,08) appareil pour chaque personne. Selon la définition de Cisco IBSG, l'IoT n'existait pas encore en 2003 car le nombre d'objets connectés était relativement faible. En outre, les appareils les plus répandus actuellement, et notamment les smartphones, faisaient tout juste leur apparition sur le marché.

La croissance explosive des smartphones et des tablettes PC a amené le nombre d'appareils connectés à Internet à 12,5 milliards en 2010, tandis que la population humaine mondiale est passée à 6,8 milliards, ce qui rend le nombre d'appareils connectés par personne supérieur à 1 (1,84 pour être exact) pour la première fois de l'histoire [4] .

En janvier 2009, une équipe de chercheurs basée en Chine a étudié les données qui transitaient sur Internet pendant plusieurs intervalles de six mois, entre décembre 2001 et décembre 2006. Les résultats présentaient des similitudes avec les propriétés de la loi de Moore et montraient que la taille d'Internet doublait tous les 5,32 ans. Après avoir combiné ce résultat au nombre d'appareils connectés à Internet en 2003 (soit 500 millions d'après l'institut de recherche Forrester Research) et à la population mondiale estimée par le Bureau

du recensement des États-Unis, Cisco IBSG a évalué le nombre d'appareils connectés par personne.

En affinant ces chiffres, Cisco IBSG a situé l'apparition de l'IoT entre 2008 et 2009 (voir figure I-2). Aujourd'hui l'IoT prend de l'ampleur, notamment en raison de la progression d'initiatives Cisco telles que la plate-forme Planetary Skin, les réseaux et les véhicules intelligents [4].

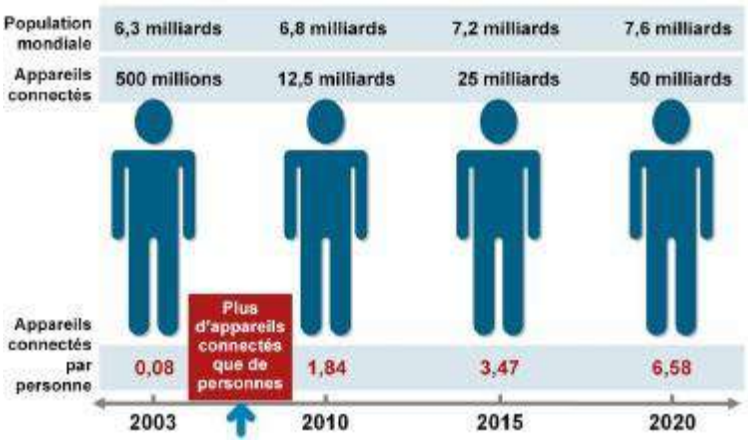


Figure I-2 L'évolution des objets connecté

En regardant vers l'avenir, Cisco IBSG avait déjà prédit qu'il y aura 50 milliards d'appareils connectés en 2020. Il est important de noter que ces estimations ne tiennent pas en compte les progrès rapides de la technologie Internet ou des appareils ; les chiffres présentés sont basés sur ce que l'on sait être vrai aujourd'hui.

De plus, le nombre d'appareils connectés par personne peut sembler faible. C'est parce que le calcul est basé sur l'ensemble de la population mondiale, dont une grande partie n'est pas encore connectée à Internet. En réduisant l'échantillon de population aux personnes réellement connectées à Internet, le nombre d'appareils connectés par personne augmente considérablement. Par exemple, nous savons qu'environ 2 milliards de personnes utilisent Internet aujourd'hui. En utilisant ce chiffre, le nombre des appareils connectés par personne passe à 6,25 en 2010, au lieu de 1,84 Bien sûr, nous savons que rien ne reste statique, surtout en ce qui concerne Internet. Avec des initiatives et des avancées telles que la plate-forme Planetary Skin de Cisco, le projet CeNSE (Central Nervous System for the Earth) d'HP et le

système smartdust (parfois traduit « poussière intelligente »), le nombre de capteurs connectés à Internet pourrait augmenter de plusieurs millions, voire de plusieurs milliards. En effet tout va être connectés, à mesure que les vaches, les conduites d'eau, les personnes et même les chaussures, les arbres et les animaux deviennent connecté à l'IoT, le monde a le potentiel de devenir un endroit meilleur [4] .

I-3- Domaine d'application des systèmes d'IOT :

L'internet offre de nombreuses applications à ses utilisateurs. Parmi ces applications nous citons :

I-3-1- Les Villes Intelligentes :

Beaucoup de grandes villes ont été soutenues par des projets intelligents, comme Séoul, New York, Tokyo, Shanghai, Singapour, Amsterdam et Dubaï. Les villes intelligentes peuvent encore être considérées comme des villes de l'avenir et la vie intelligente, et par le taux d'innovation de la création de villes intelligentes d'aujourd'hui, il sera devenu très faisable pour entrer la technologie IoT dans le développement des villes.

La demande exige une planification minutieuse à chaque étape, avec l'appui de l'accord des gouvernements, citoyens à mettre en œuvre la technologie de l'Internet des objets dans tous les aspects. Par l'IoT, les villes peuvent être améliorées à plusieurs niveaux, en améliorant les infrastructures, en améliorant les transports [5].



Figure I-3 Les constituants d'une ville intelligente

I-3-2- La santé intelligente :

L'internet des objets a rapidement transformé la prestation de soins. Les équipements et les capteurs sont de plus en plus « intelligents » et génèrent toujours plus de données nécessaires aux équipements médicaux, aux professionnels et profitant ainsi aux patients, en réduisant les coûts et en améliorant leur satisfaction. Les données ainsi collectées facilitent, adaptent, améliorent, anticipent ou réorganisent les soins des patients.

-**Surveillance des patients :** Surveillance des conditions des patients dans les hôpitaux et chez les personnes âgées.

-**Réfrigérateurs médicaux :** Contrôle des conditions à l'intérieur des congélateurs stockant des vaccins, médicaments et éléments organiques.

-**Détection Automne :** Assistance pour personnes âgées ou handicapées vivant indépendants.

-**Dentaire :** Bluetooth connecté avec une brosse à dents, avec une application Smartphone permettant l'analyse de brossage et donne des informations sur les habitudes de brossage sur le Smartphone. Ces informations peuvent être utiles pour les utilisateurs (vie privée) ou pour montrer des statistiques au dentiste.

-**Surveillance de l'activité physique :** Capteurs sans fil placés sur le matelas détectant de petits mouvements comme la respiration et la fréquence cardiaque. En lançant et en tournant pendant le sommeil, fournissant des données disponibles via une application sur le Smartphone [5].



Figure I-4 Le mécanisme de la surveillance distants des patients

I-3-3- L’énergie intelligente :

L’Energie intelligente peut apporter une amélioration à la qualité de vie des habitants et faire des économies.

-Smart Grid :

Surveillance et gestion de la consommation d'énergie, Eoliennes / Centrale électrique : Surveillance et analyse du flux d'énergie des éoliennes et des centrales électriques, et communication bidirectionnelle avec des compteurs intelligents des consommateurs pour analyser les habitudes de consommation.

-Contrôleurs d’alimentation :

Contrôleur pour Alimentations AC-DC qui déterminent l’énergie requise et améliorent l'efficacité énergétique avec moins de gaspillage d'énergie pour les alimentations liées aux ordinateurs, aux télécommunications et aux applications électroniques grand public.

-Installations photovoltaïques :

Surveillance et optimisation des performances des centrales Solaires [5].



Figure I-5 Les constituants d’une Smart Grid

I-3-4- L'environnement intelligent :

L'environnement intelligent s'appuie sur un ensemble de mesures :

- **Surveillance de la pollution de l'air.**
- **Contrôle des émissions de CO2 :** des usines et des gaz toxiques générés dans les fermes
- **Détection des incendies de forêt :** Surveillance des gaz de combustion et des conditions Préemptives d'incendie.
- **Surveillance météorologique.**
- **Surveillance de l'humidité, température, pression, vitesse du vent et pluie, séisme.**
- **Détection précoce, qualité de l'eau** : Étude de la qualité de l'eau dans les rivières et dans la mer pour l'admissibilité à l'usage potable [5].

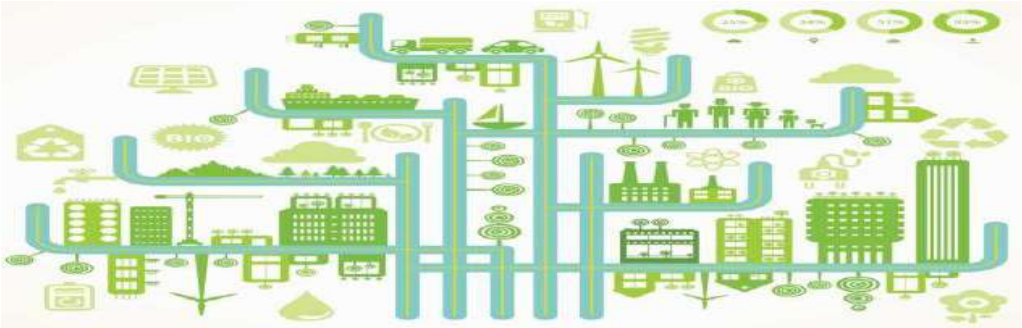


Figure I-6 Les constituants de l'environnement intelligent

I-3-5- L'industrie intelligente :

L'industrie intelligente va changer fondamentalement, comment les produits sont inventés, fabriqués et expédiés. En même temps, cela améliorera la sécurité des travailleurs et protège l'environnement un faible incident de fabrication. Ces progrès dans la façon dont les machines et autres objets communiquent, et la manière dont la prise de décision passe des humains aux systèmes techniques signifient que la fabrication devient "plus intelligente".

Gaz explosifs et dangereux :

Détection des niveaux de gaz et des fuites dans les environnements industriels, environnants des usines chimiques et des mines,

Surveillance des niveaux de gaz toxique et d'oxygène dans les usines chimiques :

Pour assurer la sécurité des travailleurs et des biens,

Surveillance des niveaux d'eau, de pétrole et de gaz :

-Réservoirs et Citerne.

-Entretien et réparation.

Les prédictions précoces :

Sur les défaillances d'équipement et la maintenance du service peuvent être automatiquement programmées avant une défaillance réelle de la pièce en installant des capteurs à l'intérieur de l'équipement pour surveiller et envoyer des rapports [5] .



Figure I-7 Les constituants de l'industrie intelligente

I-3-6- Le Transport et La mobilité intelligente :

Le développement du transport est l'un des facteurs qui indiquent le bien-être de pays. Une application de surveillance de l'état des routes et d'alerte est l'un des applications les plus importantes de l'IoT. Le processus a besoin de l'identification de l'utilisateur et son trajectoire souhaité dans son application sur son téléphonique intelligent.

Le transport intelligent est confronté à deux problématiques principales qui sont, l'analyse des transports, le contrôle des véhicules connectées. L'analyse de transport représente l'analyse de la prédiction de la demande et de détection des anomalies. Le routage des véhicules et le contrôle de la vitesse en plus de la gestion du trafic sont tous connu comme le contrôle du transport qui sont étroitement lié aux véhicules connectés [5].



Figure I-8 Quelques aspects du transport intelligent

I-4- Architecture de l'IOT :

Il n'y a pas de consensus unique sur l'architecture de l'IoT, qui est universellement accepté. Différentes architectures ont été proposées par différents chercheurs.

Comme nous l'avons vu, nous pouvons utiliser l'IoT dans plusieurs domaines et cette technologie porte un grand potentiel en tant que nouveau paradigme des télécommunications sans fil modernes. Nous considérons en particulier la visualisation de l'IoT dans un modèle de référence développé par Cisco. Ce modèle a été développé pour fournir des définitions et des descriptions claires qui peuvent être appliquées avec précision aux éléments et aux fonctions des systèmes et des applications IoT. Le modèle aide à décomposer les systèmes complexes discutés ci-dessus, afin que chaque partie soit plus compréhensible, et fournit également des informations supplémentaires pour identifier avec précision les niveaux IoT et établir une relation avec les systèmes de produits et services. Grâce à ce modèle, nous pouvons également identifier où les types spécifiques de traitement sont optimisés dans différentes parties du système. Cela permettra aux agents de créer des produits e/ou services IoT fonctionnant les uns avec les autres, ce qui rendra l'IoT réel et accessible, plutôt que simplement conceptuel.

Dans un système IoT, les données sont générées par plusieurs types de dispositifs, traitées de manières différentes et transmises dans différents emplacements par des applications. L'architecture orientée aux services a été appliquée comme technologie habilitante par (Xu, He, et Li, 2014) qui divisait l'IoT en quatre couches [6].

I-4-1- Les quatre couches de L'IOT :

L'IoT comporte quatre couches distinctes à savoir :

-Couche de détection : Cette couche est intégrée avec le hardware existant (RFID, capteurs, actionneurs, etc.) et sert à détecter/contrôler le monde physique et acquérir des données.

-Couche de mise en réseau : Cette couche fournit un soutien réseau de base et le transfert de données par un réseau sans ou avec fil.

-Couche de service : cette couche crée et gère les services et fournit des services pour satisfaire les besoins des utilisateurs.

-Couche d'interface : cette couche fournit des méthodes d'interaction aux utilisateurs et aux autres applications [6].

I-4-2- Les sept niveaux du modèle référence pour l'IoT :

Le modèle de référence IoT que nous proposons comprend sept niveaux. Chaque niveau est défini avec une terminologie qui peut être standardisée pour créer un cadre de référence. Le modèle de référence IoT ne se limite pas à la portée ou à localisation de ses composants. Par exemple, d'un point de vue physique, chaque élément pourrait résider dans un seul rack d'équipement ou pourrait être distribué par ailleurs au monde.

Les réseaux traditionnels de communication de données ont de multiples fonctions, comme en témoigne le modèle de référence à 7 couches de l'Organisation internationale de normalisation (ISO). Cependant, un système IoT complet contient de nombreux niveaux en plus du réseau de communication. Le modèle de référence IoT permet également de varier les traitements effectués à chaque niveau de trivial à complexe, en fonction de la situation. Le modèle décrit comment les tâches à chaque niveau doit être traité pour maintenir la simplicité, permettre une grande évolutivité et assurer un soutien. Enfin, le modèle définit les fonctions nécessaires pour qu'un système IoT soit complet [6].

La figure 9 illustre le modèle de référence IoT et ses niveaux. Il est important de noter que dans l'IoT, les données circulent dans les deux sens. Dans un modèle de contrôle, les informations de contrôle circulent du bas du modèle (niveau 7) vers le haut (niveau 1).

Dans un modèle de surveillance, le flux d'informations est l'inverse. Sur la plupart des systèmes, le flux sera bidirectionnel [6].

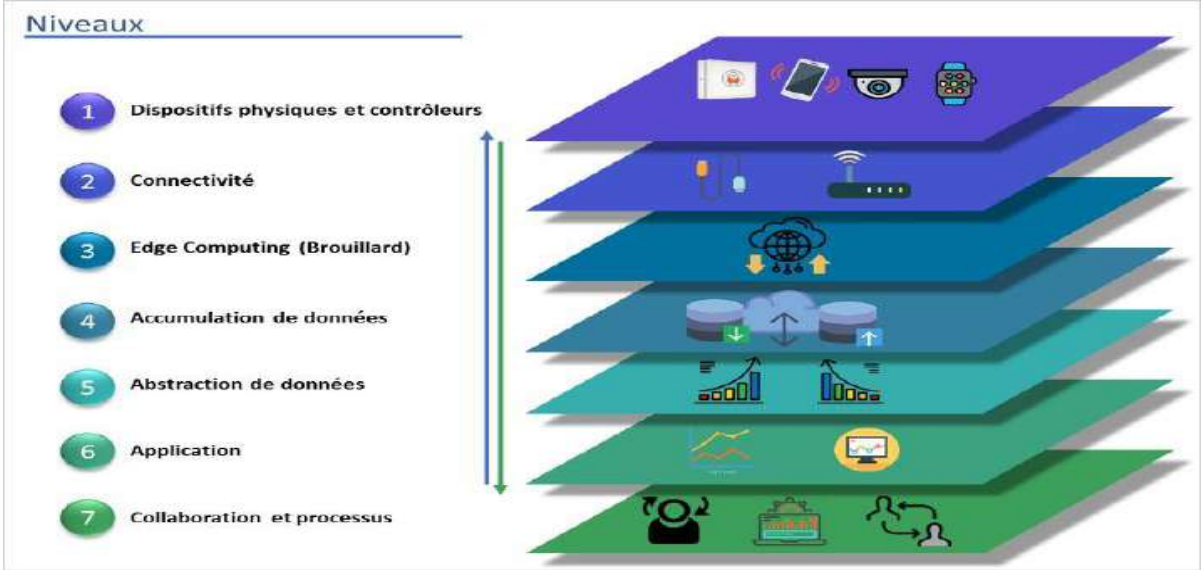


Figure I-9 Le Modèle de référence de L'IoT [6].

Le modèle de référence IoT en sept couches décrit deux flux :

•Dans un modèle de contrôle, un flux descendant, de la couche 7 à la couche 1

•Dans un modèle de surveillance, un flux montant, de la couche 1 à la couche 7

Niveau 1 : Dispositifs physiques et contrôleurs :

Le modèle de référence IoT commence par la couche 1 : périphériques physiques et contrôleurs pouvant contrôler plusieurs périphériques. Ce sont les « objets » de l'Internet des objets, et ils comprennent une large gamme d'appareils de terminal qui envoient et reçoivent des informations. Aujourd'hui, la liste des appareils est déjà étendue. Il deviendra presque illimité, car plus d'équipements sont ajoutés à l'IoT au fil du temps [6].

Niveau 2 : Connectivité

Les communications et la connectivité sont concentrées au niveau 2. La fonction la plus importante du niveau 2 est la transmission fiable et opportune de l'information.

Cela comprend les transmissions entre les appareils (niveau 1) et le réseau, entre le réseau (niveau 2) et le traitement des informations de faible niveau se trouvant au niveau 3 et à travers les réseaux (est-ouest) [6].

Niveau 3 : Edge Computing (Brouillard)

Les fonctions de niveau 3 sont dictées par la nécessité de convertir les flux de données du réseau en informations appropriées pour le stockage et le traitement de niveau 4 (hiérarchisé). Cela signifie que les activités de niveau 3 se concentrent sur l'analyse sur place des données, et leur transformation à haut volume. Par exemple, un appareil de détection de niveau 1 peut générer des échantillons de données plusieurs fois par seconde, 24 heures sur 24, 365 jours par an. Un principe de base du modèle de référence IoT est que le système plus intelligent commence à traiter les informations le plus tôt possible à partir des limites du réseau. Ceci est parfois appelé cloud computing. Au niveau 3, c'est là que cela se produit. Étant donné que les données sont généralement soumises à un équipement de réseau de niveau de connectivité (niveau 2) par des dispositifs dans de petites unités, le traitement de niveau 3 est effectué sur une base par paquet. Ce traitement est limité, car il n'y a que la connaissance des unités de données et non pas des « sessions » ou des « transactions ». Le traitement de niveau 3 peut couvrir de nombreux exemples, tels que [6] :

•Évaluation : évalue des données pour déterminer si elles doivent être traitées à un niveau supérieur.

•Mise en forme : reformate des données pour un traitement cohérent de haut niveau.

•Expansion/décodage : manipule de données cryptiques avec un contexte supplémentaire (comme l'origine).

•Distillation/Réduction : réduit et/ou synthétise des données afin de minimiser l'impact des données et du trafic sur le réseau et les systèmes de traitement de niveau supérieur.

•Évaluation : détermine si les données représentent un seuil ou une alerte ; cela peut inclure la redirection de données vers des destinations supplémentaires.

Niveau 4 : Accumulation de données

Les systèmes de réseau sont conçus pour déplacer les données en toute sécurité. Les données sont en mouvement. Avant le niveau 4, les données circulent sur le réseau et le taux et à l'organisation sont déterminées par les périphériques qui génèrent les données. Le modèle est piloté par les événements. Comme défini précédemment, les périphériques de niveau 1 n'incluent pas les ressources informatiques. Cependant, certaines activités de calcul peuvent se produire au niveau 2, telles que la traduction de protocole ou l'application de la politique de sécurité réseau. D'autres tâches informatiques peuvent être effectuées au niveau 3, telles que

l'inspection d'un paquet. La réalisation de tâches de calcul aussi proches que possible de l'IoT, avec des systèmes hétérogènes répartis sur plusieurs domaines de gestion, représente un exemple de cloud computing. Les services de brouillard et de cloud computing sont une caractéristique distinctive de l'IoT [6].

La plupart des applications ne peuvent ou n'ont pas besoin de traiter des données à la vitesse du réseau. Les applications supposent généralement que les données sont « Au repos » ou immuables en mémoire ou sur disque. A ce niveau, les données en mouvement sont converties en données au repos. Le niveau 4 détermine :

Si les données présentent un intérêt pour les niveaux supérieurs : si c'est le cas, le traitement de niveau 4 est le premier niveau configuré pour répondre aux besoins spécifiques d'un niveau supérieur.

•Les données doivent-elles être persistantes ? Les données doivent-elles être conservées sur disque dans un état non volatile ou accumulées en mémoire pour une utilisation à court terme ?

•Le type de stockage requis : La persistance nécessite-t-elle un système de fichiers, un système de données volumineux ou une base de données relationnelle ?

•Si les données sont organisées correctement : les données sont-elles organisées de manière appropriée pour le système de stockage requis ?

•Si les données doivent être recombinées ou recalculées : les données peuvent être combinées, recalculées ou agrégées avec des informations précédemment stockées, dont certaines peuvent provenir de sources non IoT.

Au fur et à mesure que le niveau 4 capture les données et les met en veille, il devient possible d'utiliser ces données par des applications en temps non réel. Les applications accèdent aux données en cas de besoin. En bref, le niveau 4 convertit les données basées sur des événements en traitement basé sur une requête. C'est une étape cruciale pour surmonter les différences entre le monde des réseaux en temps réel et le monde des applications en temps non réel [6]

Niveau 5 : Abstraction de données

Les systèmes IoT devront évoluer au niveau de l'entreprise ou même au niveau mondial et nécessiteront plusieurs systèmes de stockage pour prendre en charge les données de données

IoT provenant des systèmes ERP (« Entreprise Resource Planning »), HRMS (« Human Resource Management System »), CRM (« Customer Relationship management ») et d'autres systèmes d'entreprise traditionnels. Les fonctions d'abstraction des données de niveau 5 se concentrent sur le rendu des données et leur stockage de manière à permettre le développement d'applications plus simples et améliorées [6].

Avec plusieurs périphériques générant des données, il existe de nombreuses raisons qui expliquent pourquoi ces données ne peuvent pas se retrouver dans le même magasin de données :

Il peut y avoir trop de données à mettre dans un endroit.

•Le transfert de données vers une base de données peut consommer beaucoup de puissance de traitement, de sorte que sa récupération doit être séparée du processus de génération de données.

•Les bases de données et les entrepôts de données.

•Les appareils peuvent être géographiquement séparés et le traitement est optimisé localement.

•Les niveaux 3 et 4 peuvent séparer les « flux de données brutes continues » des « données représentant un événement ».

•Le stockage de données pour le streaming de données peut être un système de données volumineux. Le stockage des données d'événement peut être un système de gestion de base de données relationnelle avec des temps de requête plus rapides.

•Différents types de traitement de données peuvent être requis. Par exemple, le traitement en magasin se concentrera sur des choses autres que le traitement sommaire dans tous les magasins.

Pour ces raisons, le niveau d'abstraction des données doit traiter beaucoup de choses différentes. Ceux-ci incluent :

Concilier différents formats de données provenant de différentes sources.

•Assurer une sémantique des données cohérente entre les sources.

•Vérifiez que les données sont complètes pour l'application de niveau supérieur [6].

Niveau 6 : Application

Le niveau 6 est l'endroit où l'interprétation de l'information se produit. Le logiciel à ce niveau interagit avec le niveau 5 et les données au repos, il n'a donc pas besoin de fonctionner à des vitesses réseau. Le modèle de référence IoT ne définit pas strictement une application. Les applications varient en fonction des marchés verticaux, de la nature des données de l'appareil et des besoins de l'entreprise. Par exemple, certaines applications se concentreront sur la surveillance des données de l'appareil. Certains se concentreront sur le contrôle de l'appareil. Certains vont combiner l'appareil et non les données de l'appareil. Les applications de surveillance et de contrôle représentent de nombreux modèles d'application, modèles de programmation et différentes piles logicielles, conduisant à des discussions sur les systèmes d'exploitation, la mobilité, les serveurs d'applications, les hyperviseurs, le « multithreading », etc. Ces sujets dépassent la portée de notre discussion sur le modèle de référence IoT. Il suffit de dire que la complexité de l'application varie considérablement [6].

Les exemples incluent :

•Des applications entrepreneuriales critiques telles que des solutions ERP (« Enterprise Resource Planning ») généralisées ou des solutions sectorielles spécialisées ;

•Des applications mobiles qui traitent d'interactions simples ;

•Des rapports Business Intelligence, dans lesquels l'application est le serveur BI ;

•Des applications analytiques qui interprètent les données pour les décisions d'affaires

•Des applications de gestion de systèmes/centres de contrôle qui contrôlent le système IoT lui-même et n'agissent pas sur les données produites par celui-ci.

Si les niveaux 1-5 sont correctement architecturés, la quantité de travail requise par le niveau 6 sera réduite. Si le niveau 6 est correctement conçu, les utilisateurs pourront mieux faire leur travail [6].

Niveau 7 : Collaboration et processus

L'une des principales distinctions entre l'IoT et le système IoT est que l'IoT inclut les personnes et les processus. Cette différence devient particulièrement claire au niveau 7

(Collaboration et processus). Le système IoT et les informations qu'il crée ont peu de valeur à moins de produire une action qui nécessite souvent des personnes et des processus.

Les applications exécutent une logique d'affaires pour former les personnes. Les gens utilisent des applications et des données associées pour leurs besoins spécifiques. Souvent, plusieurs personnes utilisent la même application pour différentes raisons.

Par conséquent, l'objectif n'est plus l'application, mais de permettre aux gens de mieux faire leur travail. Les applications (niveau 6) donnent aux entrepreneurs les bonnes données, au bon moment, afin qu'ils puissent prendre la bonne décision.

Mais souvent, l'action requise nécessite plus d'une personne. Les gens devraient être capables de communiquer et de collaborer, parfois en utilisant l'Internet traditionnel, pour rendre l'IoT utile. La communication et la collaboration nécessitent souvent plusieurs étapes. Et cela transcende généralement plusieurs applications. C'est pourquoi le niveau 7 représente un niveau supérieur à la place d'une seule application [6].

Les modèles de distribution, les standards et les études sur l'IoT et ses applications, y compris les villes et communautés intelligentes, fournissent la plateforme de normalisation spécialisée de l'IoT nécessaire à cette convergence vers un ensemble cohérent de normes internationales. Avec la participation des participants dans le domaine des technologies de l'information et de la communication, ce type d'étude aura une incidence sur la promotion du développement de « systèmes de systèmes » très efficaces qui contribueront à réduire la fracture numérique et à rendre le monde plus connecté [6].

I-5- Les Défis del'IOT :

Il y a des défis à l'application de l'Internet des objets de point de vu coût de la mise en œuvre du fait que l'attente de la technologie doit être disponible à faible coût avec un grand nombre d'objets. De plus, l'IOT est également face à de nombreux autres défis qui se résument dans cinq sous sections suivantes [7].

I-5-1- La Découverte Automatique :

Dans les environnements dynamiques les services appropriés pour les objets doivent être identifié automatiquement. Ceci nécessite des moyens sémantiques appropriés pour décrire leur fonctionnalité afin de les exploiter d'une manière efficace [7].

I-5-2- L'Interopérabilité :

Chaque type d'objets intelligents dans l'Internet des objets possède différentes capacités de stockage d'information, de traitement et de communication. Différents objets intelligents seraient également soumis à des conditions différentes telles que la disponibilité de l'énergie et les besoins en bande passante de communications. A cet effet, pour faciliter la communication et coopération de ces objets, des normes communes sont requises.

I-5-3- La Sécurité et La Confidentialité :

Les aspects de sécurité et de protection d'Internet tel que la confidentialité des communications, l'intégrité des messages ainsi que l'authenticité et la fiabilité de partenaires de communication doivent être assurés dans un environnement IoT. A titre d'exemple dans certain cas il y a un besoin d'accéder à certains services pour accomplir une tâche ou d'empêcher de communiquer avec d'autres objets du système IoT pour des raisons préventives de sécurité. De plus, d'autres exigences selon la particularité du domaine d'application, doivent être aussi considérées comme par exemple dans les transactions commerciales des objets intelligents doivent être protégés des yeux des concurrents [7].

I-5-4- La Tolérance Aux Pannes :

Les objets dans IOT sont beaucoup plus dynamiques et mobiles que les ordinateurs Internet, et dans un changement rapide et de manière inattendue. Pour structurer Internet des objets d'une manière robuste et digne de confiance on utilise la redondance en plusieurs niveaux pour l'adaptation automatique au changement de conditions.

I-5-5- L'Auto-Organisation :

Les objets intelligents ne devraient pas être gérés comme des ordinateurs qui nécessitent leurs utilisateurs pour les configurer et les adapter à des situations particulières. Les objets mobiles, qui sont souvent utilisées de façon sporadique, doivent établir spontanément des liens et pouvoir être organisé et configurer par eux-mêmes en fonction de leur environnement d'exécution [7].

I-6- Les bénéfices de l'IoT :

L'Internet des objets promet ainsi une forte plus-value à chaque organisme. En connectant objets, personnes et environnements, il devient possible de développer des améliorations qui ne pourront être que bénéfiques.

Les principaux bénéfices que peut avoir une entreprise dotée d'un système IoT sont :

Amélioration de la productivité : l'IoT permet la surveillance, le monitoring et le contrôle des différents process, ce qui optimise les différentes opérations qui augmentent la productivité et l'efficacité.

•Analyses prédictives : grâce à la collecte de nombreuses données, les nouvelles technologies de l'IoT permettent d'examiner les patrons récurrents et contribuent à l'analyse prédictive qui peut être principalement utilisée en maintenance. Ces informations précises vont servir à améliorer les process et les services existants.

•Rapidité d'action : les données permettent de suivre en temps réel et même à distance les systèmes mis en place. Elles facilitent l'optimisation des interventions de maintenance, mais aussi donnent un avantage stratégique à l'entreprise dans le suivi de l'évolution des marchés.

•Diminution des erreurs humaines : grâce à la complémentarité des technologies comme l'intelligence artificielle, l'IoT permet de minorer les erreurs humaines dues à des tâches mondaines ou répétitives.

L'IoT constitue ainsi un atout dans l'aide à la prise de décision et au rendement de l'entreprise. Il représente un avantage stratégique face au marché concurrentiel. C'est une tendance qui ne cessera de croître, puisqu'on estime qu'il y aura plus de 20 milliards d'objets connectés dans le monde d'ici 2020, Face à leur futur prometteur, il serait temps d'en profiter et commencer à transformer son business [8].

I-7- Protocole de communication :

-HTTP : signifie Hypertext Transfer Protocol (protocole de transfert hypertexte). Ce protocole définie la communication entre un client (exemple : navigateur) et un serveur sur le World Wide Web (WWW). Est un protocole client/serveur qui utilise le principe requête/réponse. Nous avons choisi le http pour interagir avec le service de base de données « Cloudant ». On utilise la méthode GET pour demander une ressource et la méthode POST pour stocker les données. Nous avons choisi le protocole http pour plusieurs raisons :
•Comme le service Cloudant utilise les bases de données NoSQL et stocke les données en tant que documents avec le format JSON, il est facile d'interroger les données avec une demande http. Ces données sont traitées alors dans les différentes applications.
La réponse de Cloudant est un fichier texte avec le format JSON. Donc Un http est la solution idéale pour obtenir les données.

•La plupart des langages de programmation tels que java prennent en charge le protocole http sans avoir besoin de bibliothèques externes.

-MQTT : signifie Message Queuing Telemetry Transport. Il utilise le principede « Publisher / Subscriber » pour connecter les systèmes entre eux.

-AMQP: signifie Advanced Message Queuing Protocol. Un protocole pour les systèmes de messagerie orientés messages (MOM).

-STOMP: signifie Simple Text Oriented Messaging Protocol. Est un protocole textuel au-dessus de TCP conçu pour permettre l'interaction avec un middleware orienté messages.

I-7-2 Le protocole MQTT :

MQTT permet concrètement aux objets d'envoyer des informations sur un sujet donné à un serveur qui fonctionne comme un broker de messages. Le broker pousse ces informations vers les clients qui se sont précédemment abonnés [9].

Nous avons choisi le protocole MQTT pour de nombreuses raisons :

•Le MQTT échange les données en temps réel ce qui nous aide à créer et à envoyer des notifications en même temps qu'une chose anormale détectée.

•Il fonctionne pour les appareils à faible puissance, dans notre cas c'est l'Arduino.

•Un protocole facile à utiliser en raison du principe de publication / abonnement.

A-Publication / abonnement :

Le protocole MQTT est basé sur le principe de la publication de messages et de l'abonnement à des sujets, ou "pub / sub". Plusieurs clients se connectent à un courtier et s'abonnent aux sujets qui les intéressent. Les clients se connectent également à courtier et publier des messages sur des sujets. De nombreux clients peuvent s'abonner aux mêmes sujets et faire avec les informations comme ils veulent. Le courtier et MQTT agissent comme une interface simple et commune pour tout ce à quoi se connecter [10].

B-Qualité de service :

MQTT définit trois niveaux de qualité de service (QoS). La QoS définit à quel point le courtier / client essayera d'assurer qu'un message est reçu. Les messages peuvent être envoyés à n'importe quel niveau de qualité de service, et les clients peuvent tenter de s'abonner aux

sujets à n'importe quel niveau de QoS. Cela signifie que le client choisit la QoS maximale qu'il recevra. Par exemple, si un message est publié à QoS 2 et qu'un client est abonné à QoS 0, le message sera livré à ce client avec QoS 0. Si un deuxième client est également abonné au même sujet, mais avec QoS 2, il recevra le même message mais avec QoS 2. Pour un deuxième exemple, si un client est abonné à QoS 2 et qu'un message est publié sur QoS 0, le client recevez-le sur QoS 0. Des niveaux plus élevés de QoS sont plus fiables, mais impliquent une latence plus élevée et ont des exigences de bande passante plus élevées [10].

0: Le courtier / client remettra le message une fois, sans confirmation.

1: Le courtier / client remettra le message au moins une fois, avec confirmation requise.

2: Le courtier / client livrera le message exactement une fois en utilisant une poignée de main en quatre étapes

C-Le serveur :

1883 : MQTT, unencrypted

8883 : MQTT, encrypted

8884 : MQTT, encrypted, client certificate required 8080 : MQTT over WebSockets, unencrypted 8081 : MQTT over WebSockets, encrypted

Les ports cryptés prennent en charge TLS v1.2, v1.1 ou v1.0 avec des certificats x509 et requièrent le support du client pour se connecter. Dans tout cas, vous devez utiliser le fichier d'autorité de certification mosquitto.org.crt pour vérifier la connexion au serveur. Port 8884 exige que les clients fournissent un certificat pour authentifier leur connexion. Si vous souhaitez obtenir un certificat client, S'il vous plaît, faites-le toucher [10].

D-Principe de fonctionnement :

Contrairement au principe du client/serveur utilisé sur le Web, MQTT utilise celui de la publication/souscription : plusieurs clients se connectent à un serveur unique (appelé broker) pour soit publié des informations, soit souscrire à leur réception [10].

Conclusion :

Dans ce chapitre, nous avons présenté une étude détaillée sur l'internet des objets, sa définition, leur domaine d'application, l'architecture, les défis, et enfin les bénéfices de l'IoT.

L'IoT incarne la prochaine évolution de l'Internet. Sachant que l'être humain progresse et évolue en transformant les données en informations, en connaissances et en savoir, l'IoT à le potentiel d'améliorer le monde tel que nous le connaissons. La rapidité à laquelle nous y parviendrons ne dépend que de nous.

Dans le chapitre suivant nous allons détailler les différents Protocol de l'IoT.

Chapitre II : Protocol MQTT

II-1-Le protocole MQTT :

II-1-1- Définition :

MQTT (Message Queue Telemetry Transport) est un protocole M2M (Machine-to-Machine). C'est un protocole de messagerie basé sur le publish/subscrib à la fois extrêmement simple et léger idéal pour l'Internet des objets. Il a été inventé par Andy Stanford-Clark (IBM) et Arlen Nipper (Arcom, maintenant Cirrus Link) en 1999, lorsque leur cas d'utilisation était de créer un protocole avec les objectifs suivants :

1- Simple à mettre en œuvre.

2- Fournir une qualité de service de livraison de données.

3- Léger et faible consommation de la bande passante.

Par la suite, la version 3.1 a été libérée de droits en 2010, et le protocole est standardisé par l'OASIS en 2014 dans sa version 3.1.1. Il devient un standard ISO (ISO/IEC 20922) en 2016[11].

Aujourd'hui, avec l'essor de l'IOT, MQTT suscite un intérêt grandissant, dont on peut se rendre compte en observant l'évolution du nombre de recherches pour MQTT.

Les qualités de MQTT font de lui un excellent candidat pour les communications au sein de l'Internet des Objets, mais il a également été utilisé avec succès par Facebook pour être au cœur de leur système de messagerie à cause de sa performance, comme il est utilisé par Amazon web services dans ses passerelles pour permettre la diffusion des données à plusieurs abonnés sur un sujet donné.

La norme MQTT décrit le comportement attendu des brokers et clients MQTT, ainsi que les détails du format binaire employé pour les échanges [11].

II-1-2- Mode de fonctionnement MQTT :

MQTT est un protocole de publication et de souscription de message. Les clients ne communiquent pas directement entre eux, ils publient des messages sur un broker (appelé aussi courtier), les messages sont composés d'un contenu et d'un sujet (topic). Le broker garde en mémoire le dernier message pour chaque sujet. Les clients qui sont intéressés par les messages d'un sujet peuvent les récupérer en se connectant au broker. L'avantage de cette solution est qu'elle permet à plusieurs clients de communiquer même s'ils ne sont jamais connectés en même temps au broker. Ce modèle consiste à organiser les messages par sujets

(topics) et à gérer leur distribution selon le principe d'abonnement. Ainsi, toute application peut publier ses messages sur les topics de son choix, tandis que les applications intéressées par les messages d'un topic donné peuvent s'abonner pour recevoir tous les nouveaux messages publiés sur ce topic [12].

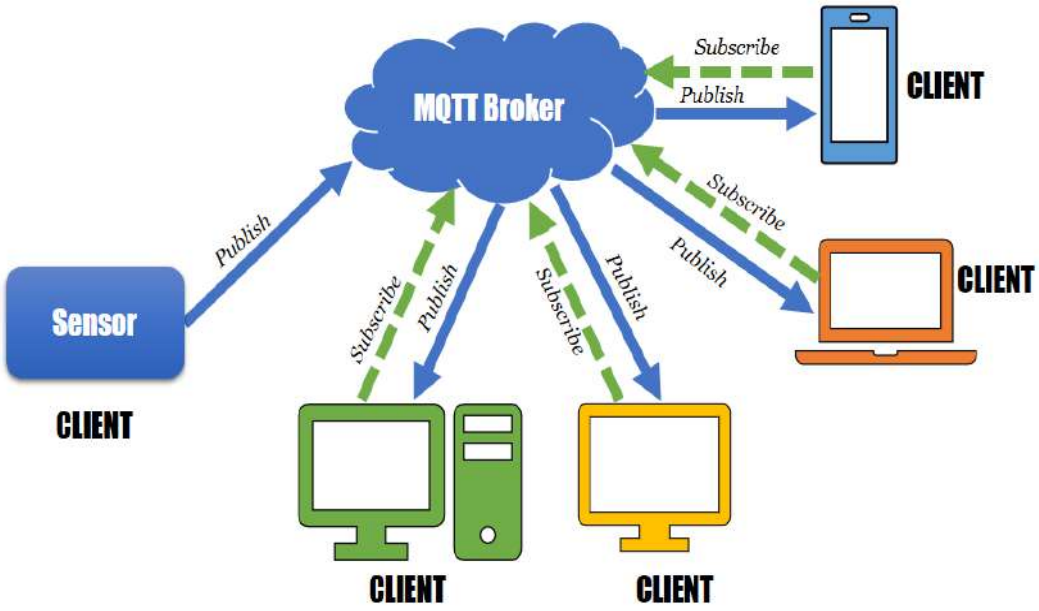


Figure II-1 Modèle pour transférer des données entre clients (publish / subscribe) à travers le broker

II-1-3- Abonnements et Publications :

Chaque message publié est nécessairement associé à un sujet, ce qui permet sa distribution aux abonnés. Les sujets peuvent être organisés en hiérarchie arborescente, ainsi les abonnements peuvent porter sur des motifs de filtrage.

La gestion des abonnements est très simple et consiste en trois commandes essentielles :

1. **SUBSCRIBE :** Permet à un client de s'abonner à un topic, une fois abonné il recevra par la suite toutes les publications concernant ce sujet. Un abonnement définit également un niveau de qualité de service. La bonne réception de cette commande est confirmée par le broker par un SUBACK portant le même identifiant de paquet.

2. **UNSUBSCRIBE** : Donne la possibilité d'annuler un abonnement, et ainsi ne plus recevoir les publications ultérieures. La bonne réception de cette commande est confirmée par le broker par un UNSUBACK portant le même identifiant de paquet.

3. PUBLISH : Initié par un client, permet de publier un message qui sera transmis par le broker aux abonnés éventuels. La même commande sera envoyée par le broker aux abonnés pour délivrer le message Si la qualité de service requise est supérieure à zéro, des messages seront échangés pour confirmer la prise en charge de la publication [13].

II-1-4 Les topics :

Les clients de souscription MQTT s'enregistrent auprès du broker sur des topics, des sortes de chemins d'accès à une ressource. Ils demandent ainsi à être notifiés lorsque quelqu'un publie sur ces topics [14].

Cela peut être un topic de température par exemple :

/sensor/1/temperature. On peut souscrire à un ensemble de topics en utilisant des wildcards # ou +.

Par exemple, si un client publie sur les topics /sensor/1/temperature et /sensor/1/humidity, un autre client peut écouter ces deux topics à la fois : /sensor/1/#.

Si plusieurs clients publient leurs températures et humidités en intercalant leur numéro de client sur leur topic, un autre client peut écouter toutes les températures ainsi :

/sensor/+/temperature. Il recevra alors les températures du client 1(/sensor/1/temperature), et du client 2 (/sensor/2/temperature), etc [14].

II-1-5- Qualité de service :

Trois niveaux de qualité de service (QoS) sont définis pour la publication des messages :

1. QoS 0 « Livraison une fois maximum ».

Les messages sont envoyés en fonction des capacités maximum du réseau TCP/IP sous- jacent. Aucune réponse n'est attendue. Aucune sémantique de relance n'est définie dans le protocole. Par conséquent, le message ne parvient pas du tout ou une seule fois au broker de destination. Le niveau QoS 0 est également connu comme fire and forget.

2- QoS 1 « Livraison au moins une fois ».

L'arrivée d'un message QoS 1 au broker est reconnue. En cas d'échec identifiable de la liaison ou de l'unité d'envoi, ou bien après une certaine période de temps sans réception du

message d'accusé de réception, l'expéditeur envoie à nouveau une copie du message. Par conséquent, le message est sûr d'arriver, mais il peut le faire plusieurs fois.

3-QoS 2 « Livraison exactement une fois ».

Pour le niveau QoS 2, d'autres flux de protocoles sont utilisés au-delà de QoS 1 pour que des messages ne soient pas envoyés en double à l'application de réception. Il s'agit du niveau de service le plus élevé qui sert lorsque des messages en double ne sont pas appropriés. Il existe évidemment des conséquences en matière de trafic réseau, mais cet impact est souvent acceptable sachant l'importance du contenu du message.

La qualité de service peut être sélectionnée message par message, ce qui permet la publication des messages d'importance mineure avec le niveau QoS 0 et la distribution des messages plus importants avec QoS 2.

Ces niveaux sont mis en œuvre par des échanges supplémentaires entre l'expéditeur et le récepteur, et plus la qualité demandée est élevée, plus il faudra d'échanges pour valider une publication comme illustré dans la figure. Pour tous les niveaux supérieurs à zéro, un identifiant est associé au message pour permettre son suivi, MQTT prévoyant la possibilité d'avoir au plus 65535 messages en attente (l'identifiant de message tenant sur 16 bits) [15].

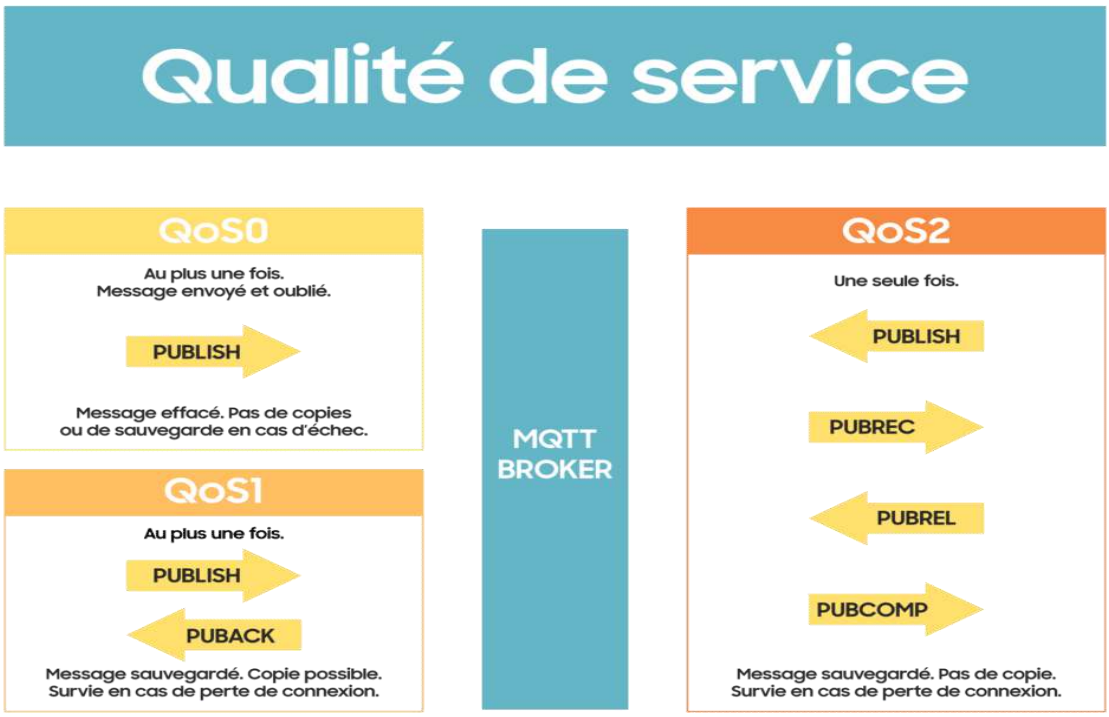


Figure II-2 Niveaux de Qualité de Service MQTT

III-1-6- Le broker MQTT :

MQTT (Message Queuing Telemetry Transport) est un protocole de messagerie qui fonctionne sur le principe de souscription / publication qui a été développé à la base pour simplifier la communication entre les machines. Pour fonctionner il faut installer un serveur, qui est désigné par Broker.

Mosquitto est un très bon choix. Il est disponible sur toutes les plateformes et le plus utilisé dans les projets DIY [16].

1-Publier (envoyer) un message MQTT depuis le terminal :

La commande mosquitto_pub permet de publier simplement un message MQTT. Voici quelques options utiles à connaître :

- h nom du broker (ou IP) sur lequel se connecter
- p port, par défaut 1883
- u nom d'utilisateur (optionnel)
- P mot de passe (optionnel)
- t topic sur lequel publié le message
- m message (payload) à envoyer. Le payload peut être une valeur ou un JSON.
- r indique au broker de retenir (ou pas) le message

2-Recevoir (souscrire) des messages :

La commande mosquitto_sub permet de souscrire, c'est à dire de recevoir des messages publiés sur un broker MQTT. Les options sont presque identiques à mosquitto_pub.

II-1-7- Sécurité :

Trois concepts sont fondamentaux pour la sécurité MQTT : identification, authentification et autorisation.

-L'identification : consiste à nommer le broker et le client auquel on donne des droits d'accès.

-L'authentification : cherche à prouver mutuellement l'identité du client et du broker.

-L'autorisation : consiste à gérer les droits du client. Le broker MQTT s'identifie auprès du client avec son adresse IP et son certificat numérique. Le client MQTT utilise le protocole SSL pour authentifier le certificat envoyé par le broker. Un client authentifie un broker à l'aide du protocole SSL. Un broker MQTT authentifie un client à l'aide du protocole SSL,d'un mot de passe, ou des deux. L'autorisation ne fait pas partie du protocole MQTT. Elle est fournie par les brokers MQTT. Ce qui est autorisé ou non dépend de ce que fait le broker [17].

II-2Arduino :

II.2.1 Introduction :

Le système Arduino donne la possibilité d'allier les performances de la programmation à celles de l'électronique. Plus précisément, pour programmer des systèmes électroniques. Le gros avantage de l'électronique programmée c'est qu'elle simplifie grandement les schémas électroniques et par conséquent, le coût de la réalisation, mais aussi la charge de travail à la conception d'une carte électronique [18].

Le système Arduino permet de :

- contrôler les appareils domestiques
- fabriquer votre propre robot
- faire un jeu de lumières
- communiquer avec l'ordinateur
- télécommander un appareil mobile (modélisme)
- etc.

Le système Arduino est composé de deux choses principales **:** le matériel et le logiciel.

- **Le matériel :**

Il s'agit d'une carte électronique basée autour d'un microcontrôleur Atmega du fabricant Atmel, dont le prix est relativement bas pour l'étendue possible des applications.

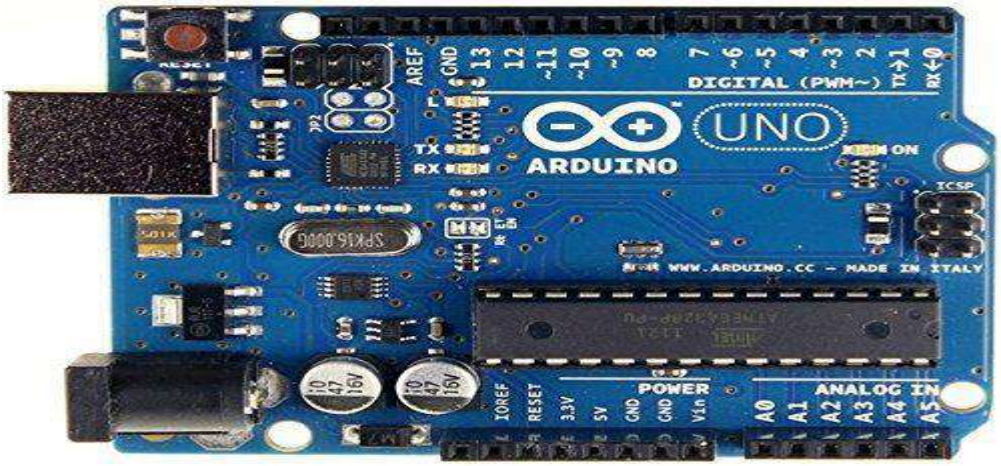


Figure II-3 Arduino UNO

. **Le logiciel :**

Permet de programmer la carte Arduino. Il offre une multitude de fonctionnalités

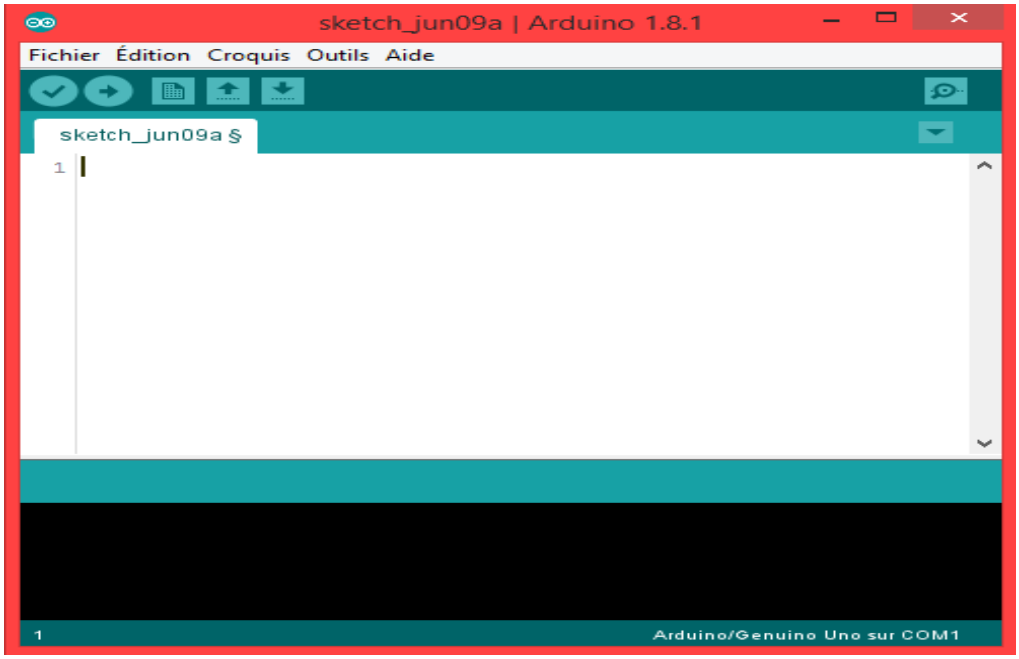


Figure II-4 interface du programme Arduino

Il y a trois types de cartes :

- Les « officielles » qui sont fabriquées en Italie par le fabricant officiel : Smart Projects
- Les « compatibles » qui ne sont pas fabriqués par Smart Projects, mais qui sont totalement compatibles avec les Arduino officielles.
- Les « autres » fabriquées par diverse entreprise et commercialisées sous un nom différent (Freeduino, Seeduino, Femtoduino, ...) [18].

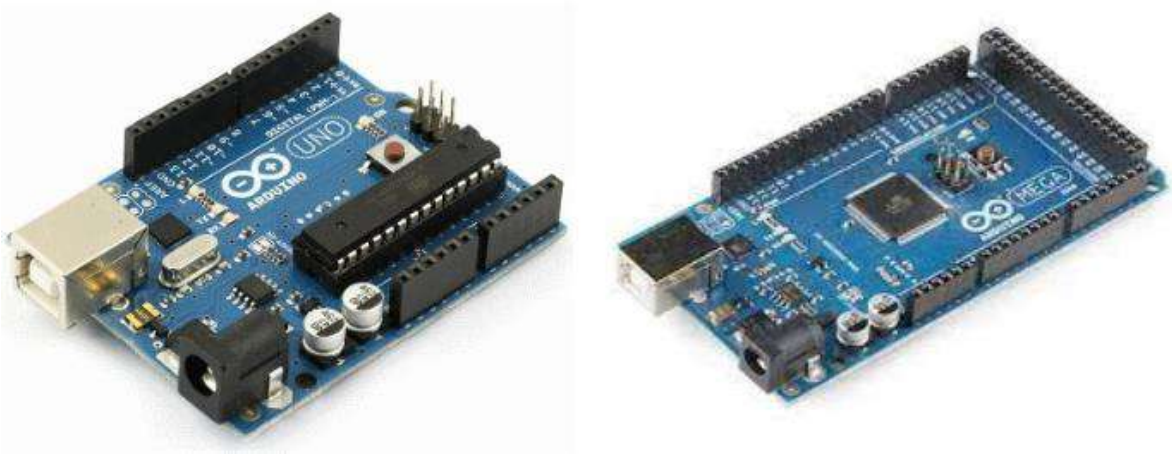


Figure II-5 Carte (Uno et Duemilanove) et (Carte Mega)

La carte Arduino est équippé d'un microcontrôleur. Le microcontrôleur est un composant électronique programmable. On le programme par le biais d'un ordinateur grâce à un langage informatique, souvent propre au type de microcontrôleur utilisé [18].

Un microcontrôleur est constitué par un ensemble d'éléments qui ont chacun une fonction bien déterminée. Il est en fait constitué des mêmes éléments que sur la carte mère d'un ordinateur :

1. La mémoire :

Il en possède 5 types :

- La mémoire Flash :

C'est celle qui contiendra le programme à exécuter.Cette mémoire est effaçable et ré-inscriptible.

- RAM :

C’est la mémoire dite "vive", elle va contenir les variables de votre programme. Elle est dite "volatile" car elle s'efface si on coupe l'alimentation du micro-contrôleur.

- EEPROM :

C'est le disque dur du microcontrôleur. Vous pourrez y enregistrer des infos qui ont besoin de survivre dans le temps, même si la carte doit être arrêtée. Cette mémoire ne s'efface pas lorsque l'on éteint le microcontrôleur ou lorsqu'on le reprogramme.

- Les registres :

C’est un type de mémoire utilisé par le processeur.

- La mémoire cache :

C’est une mémoire qui fait la liaison entre les registres et la RAM.

2. Le processeur :

C'est le composant principal du micro-contrôleur. C'est lui qui va exécuter le programme qu'on lui donnera à traiter. On le nomme souvent le CPU.

Pour que le microcontrôleur fonctionne, il lui faut une alimentation ! Cette alimentation se fait en générale par du +5V. D'autres ont besoin d'une tension plus faible, du +3,3V.

En plus d'une alimentation, il a besoin d'un signal d'horloge. C'est en fait une succession de 0 et de 1 ou plutôt une succession de tension 0V et 5V. Elle permet en outre de cadencer le fonctionnement du microcontrôleur à un rythme régulier. Grâce à elle, il peut introduire la notion de temps en programmation [18].

II-2-2 Le logiciel :

Au jour d'aujourd'hui, l'électronique est de plus en plus remplacée par de l'électronique programmée. On parle aussi d'électronique embarquée ou d'informatique embarquée.

II-2-2-1 L'interface :

L'interface du logiciel Arduino se présente de la façon suivante :

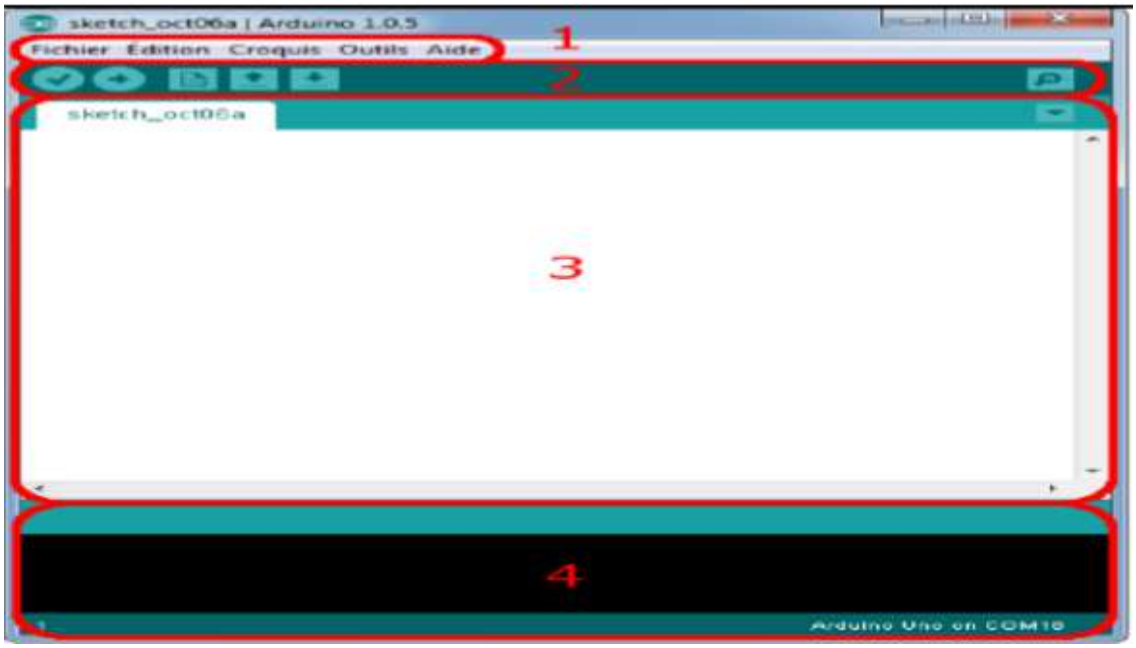


Figure II-6 interface de Arduino IDE

1. options de configuration du logiciel

2. boutons pout la programmation des cartes

3. programme à créer

4. débogueur (affichage des erreurs de programmation)

1. Menu :

Les différents éléments du menu permettent de créer de nouveaux sketchs (programmes), de les sauvegarder, et de gérer les préférences du logiciel et les paramètres de communication avec votre carte esp8266 Node MCU.

Le menu comprend :

•**Fichier :** Pour créer, sauvegarder en spécifiant la destination, et d'appeler un programme.

•**Edition :** Pour couper, copier, coller, supprimer, sélectionner…, etc.

•**Croquis :** Regroupe les fichiers réalisés.

•**Outils :** Pour spécifier le type de la carte, le port série, formater, recharger et réparer l'encodage, graver la séquence d'initialisation, de la carte branchée sur l'ordinateur [19].

Le menu File dispose d'un certain nombre de choses qui vont être très utiles :

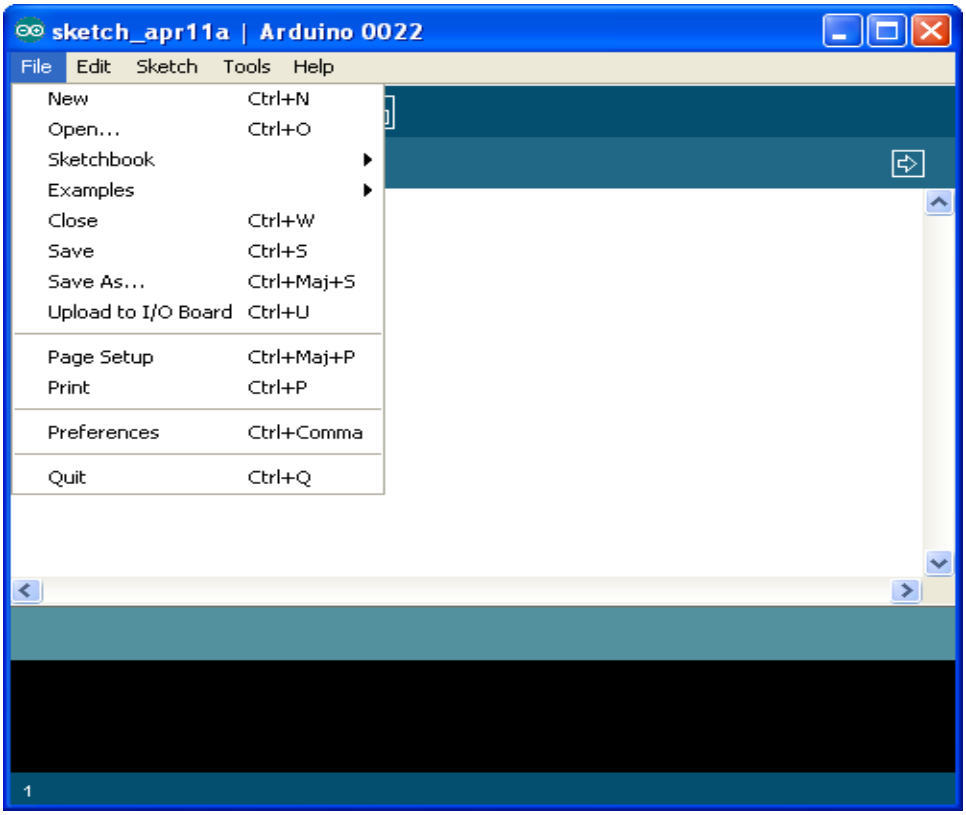


Figure II-7 Le menu File de Arduino

- New (nouveau) : va permettre de créer un nouveau programme. Quand on appuie sur ce bouton, une nouvelle fenêtre, identique à celle-ci, s'affiche à l'écran.

- Open... (ouvrir) : avec cette commande, on peut ouvrir un programme existant.

- Save / Save as... (enregistrer / enregistrer sous...) : enregistre le document en cours / demande où enregistrer le document en cours.

- Examples (exemples) : ceci est important, toute une liste se déroule pour afficher les noms d'exemples de programmes existant [18].

2. Les boutons :



Figure II-8 Les boutons d'action

- **Bouton 1 :** Ce bouton permet de vérifier le programme, il actionne un module qui cherche les erreurs dans le programme
- **Bouton 2 :** Charge (téléverse) le programme dans la carte esp8266.
- **Bouton 3 :** Cree un nouveau fichier.
- **Bouton 4 :** Ouvre un fichier.
- **Bouton 5 :** Enregistre le fichier.
- **Bouton 6 :** Ouvre le moniteur série [20].

NOTE : Le moniteur série est un outil de communication entre le pc et la carte esp8266 qui sont connecté via le port USB, utilisé généralement pour tester le fonctionnement des programmes.

II-2-3 Le langage Arduino :

Le projet Arduino était destiné à l'origine principalement à la programmation multimédia interactive en vue de spectacle ou d'animations artistiques. C'est une partie de l'explication de la descendance de son interface de programmation de Processing.

Processingest une librairie java et un environnement de développement libre. Le logiciel fonctionne sur Macintosh, Windows, Linux, BSD et Android.

Références :

- Le langage Java.

- Le langage C.
- L'algorithmique.

Cependant, le projet Arduino a développé des fonctions spécifiques à l'utilisation de la carte qui ont été listées ci-dessous. Vous obtiendrez la description de chacune d'elles dans le manuel de référence[18].

Structure	Constants	Functions
•setup() •loop()	•HIGH, LOW •INPUT, OUTPUT, INPUT_PULLUP •LED_BUILTIN	E/S numérique •pinMode() •digitalWrite() •digitalRead() E/S analogique •analogReference() •analogRead() •analogWrite() - PWM E/S avancée •tone() •noTone() •shiftOut() •shiftIn() •pulseIn() Temps •millis() •micros() •delay() •delayMicroseconds() Bits et octets •lowByte() •highByte() •bitRead() •bitWrite() •bitSet() •bitClear() •bit() Interruptions externes •attachInterrupt()

		•detachInterrupt() Interruptions •interrupts() •noInterrupts() Communication •Serial •Stream

II.3 Node-red :

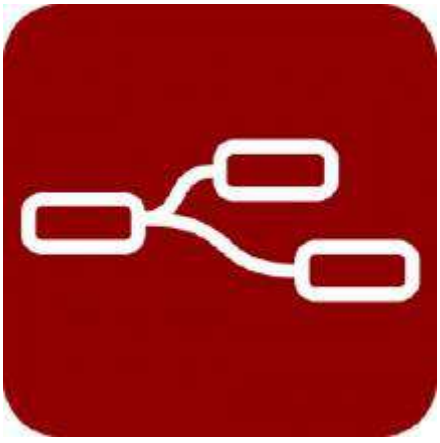


Figure II.8 Node-red logo

Node-red : est un outil puissant pour construire des applications de l'Internet des Objets en mettant l'accent sur la simplification de la programmation qui se fait grâce à des blocs de code prédéfinis, appelés nodes pour effectuer des tâches. Il utilise une approche de programmation visuelle qui permet aux développeurs de connecter les blocs de code ensemble.

Les noeuds connectés, généralement une combinaison de noeuds d'entrée, de noeuds de traitement et de noeuds de sortie, lorsqu'ils sont câblés ensemble, constituent un flow.

Node-red : est construit sur Node.js, tirant pleinement parti de son modèle non bloquant piloté par les événements. Cela le rend idéal pour fonctionner en périphérie du réseau sur un serveur d'application qui peut être un matériel à faible coût tel que le Raspberry Pi ou un serveur dans le cloud [21].

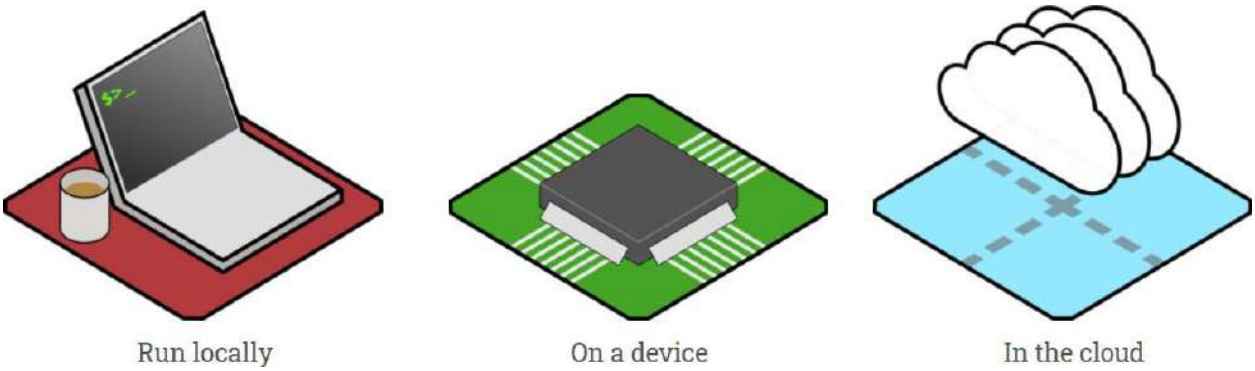


Figure II.9 Node-red fonctionne sur tout type de systèmes

II-3-1 Installation :

Consultez le guide de démarrage de node-red pour procéder à l'installation qui correspond à votre cas.

https://nodered.org/docs/getting-started/

Nous utiliserons ici node-red localement sur notre ordinateur. L'installation consiste donc à installer préalablement node.js :

- Installation de `node.js`
- Installation de **node-red**

démarrage et arrêt

•Pour démarrer node-red, utilisez la commande node-red dans un terminal.

```
$ node-red

Welcome to Node-RED
===================
```

Figure II.10 Sous linux

```
node-red

C:\>node-red
12 Jan 12:10:10 - [info]

Welcome to Node-RED
===================
```

Figure II.11 Sous Windows

II-3-2 interface :

On accède à l'interface de développement par un navigateur en utlisant l'URL : http://127.0.0.1:1880 comme l'indique node-red dans le terminal :

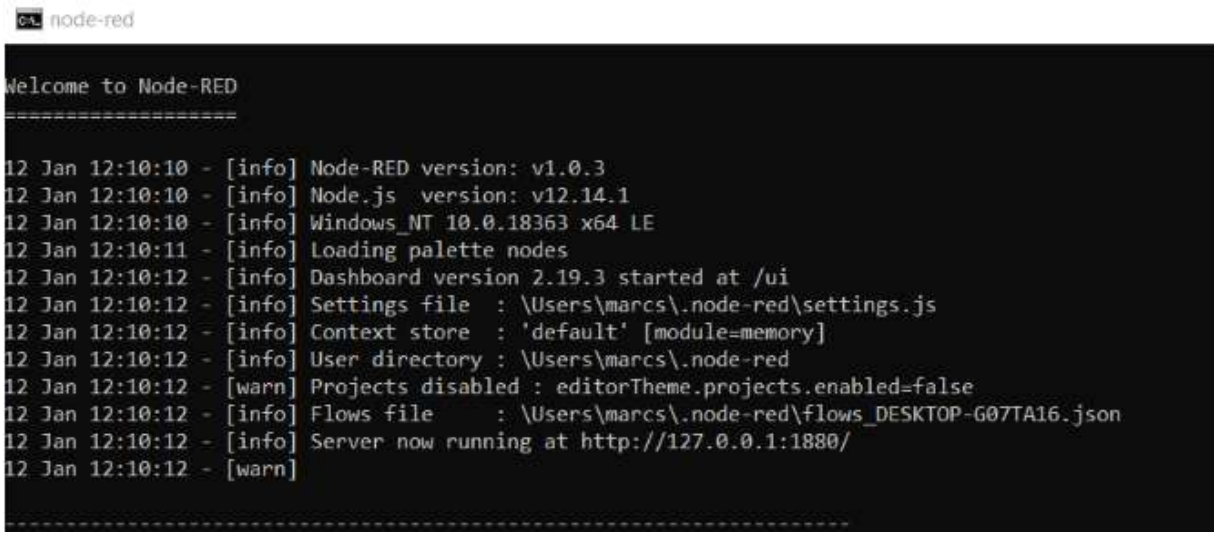


Figure II.12 Execution de node-red dans le terminal

On y accède également depuis n'importe quel poste du réseau pour peu que le pare-feu soit correctement configuré.

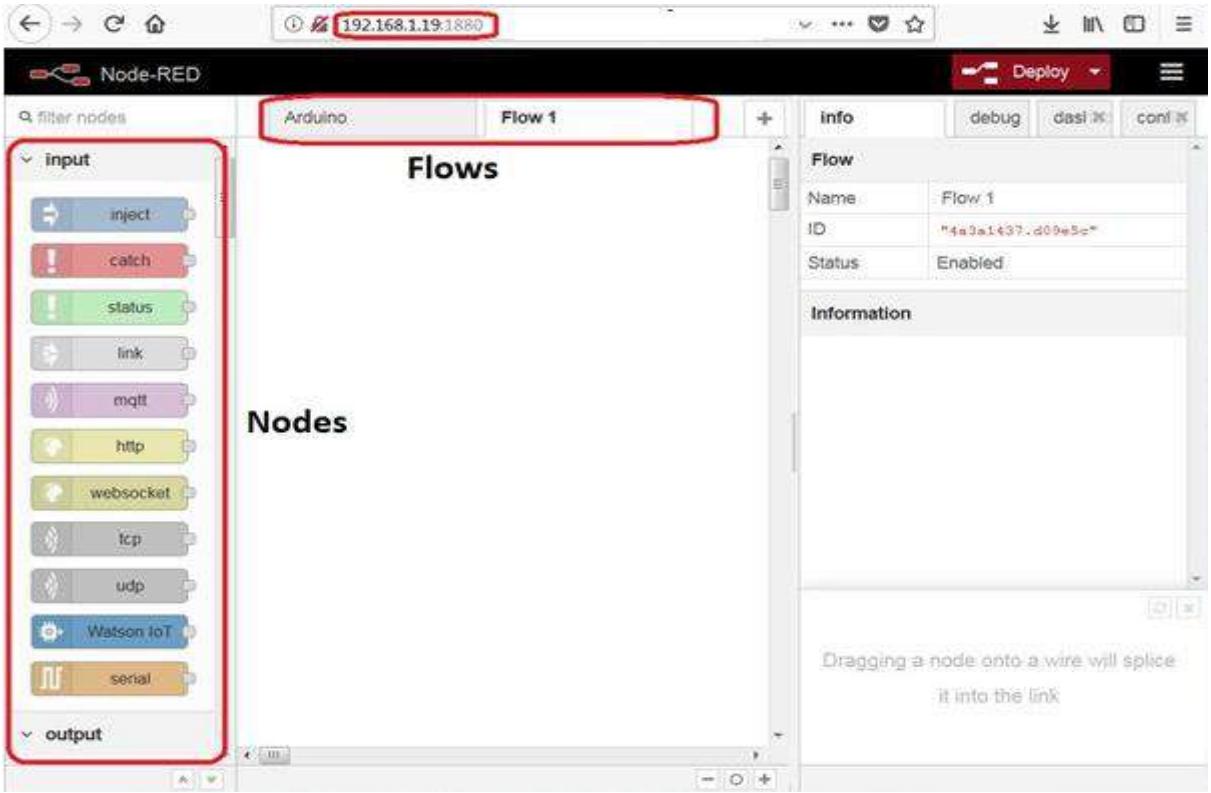


Figure II.13 IHM de node-red

II-3-3 Ajout d'une catégorie de nodes à la palette :

La palette de nodes d'origine est déjà bien fournie mais il manque au moins une catégorie essentielle qui va permettre de réaliser l'interface web. Elle se nomme Dashboard [21].

Dans le menu, cliquez sur Manage Palette puis recherchez la catégorie de nodes à installer. Une nouvelle catégorie est ensuite disponible :

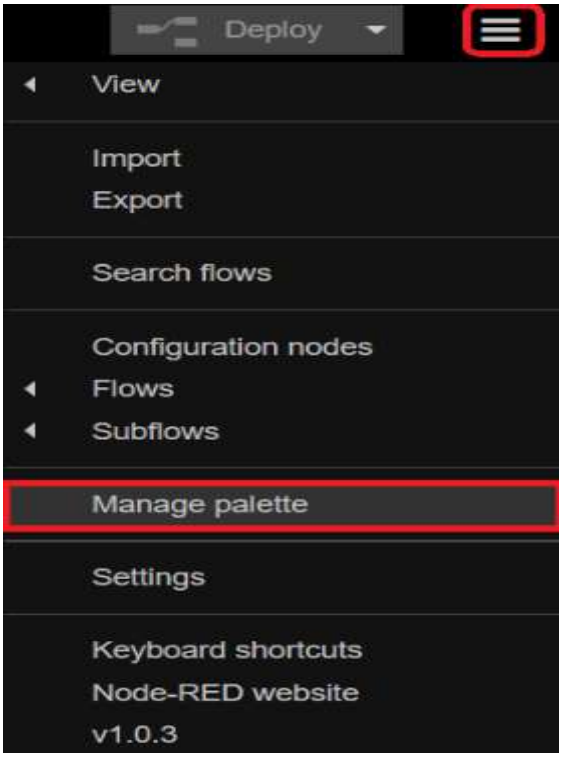


Figure II.14 Accéder au gestionnaire de palette

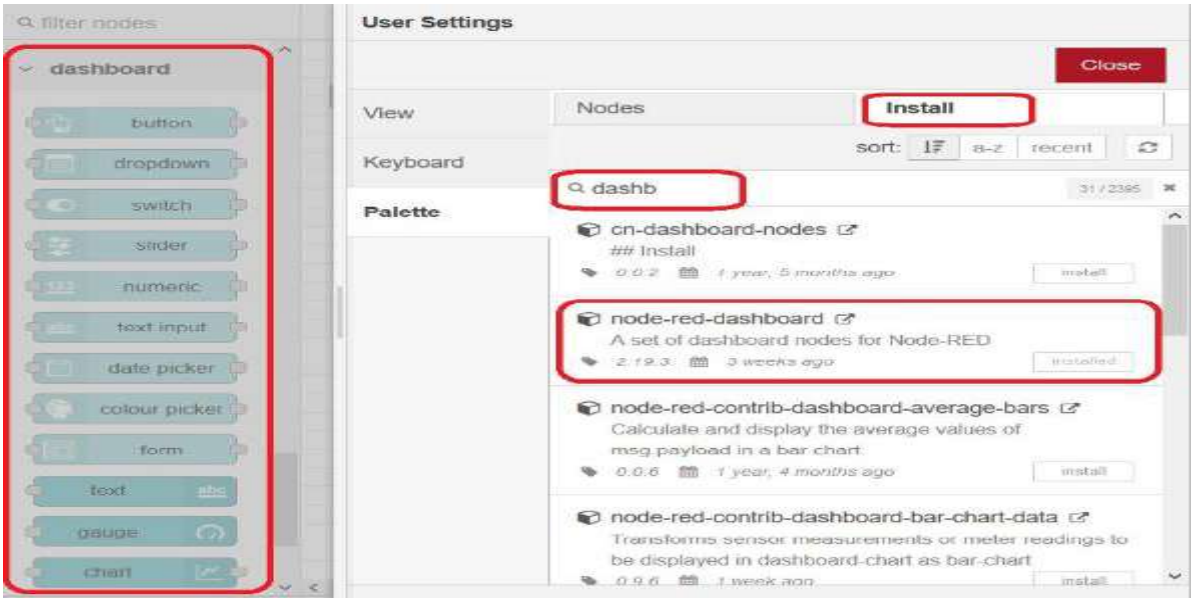


Figure II.15 Ajout de la catégorie Dashboard à la Palette

II-3-4 Principe de la programmation :

•La programmation consiste à glisser-déplacer les nodes sur le flow et à les relier entre eux. Les données échangées sont des objets JSON et doivent posséder la propriété payload (charge utile).

•Le payload peut être un nombre, une chaine de caractères ou un objet JSON.

•Certains nodes peuvent avoir plusieurs sorties. Dans ce cas, le node retourne un tableau d'objets JSON dont le premier élément sort par la première sortie, le deuxième par le second,

Commençons par réaliser un flow très simple composé de :

•un Slider

•une Gauge.

Pour modifier la configuration par défaut des nodes, il suffit de double-cliquer dessus.

Pour que ces composants soient bien disposés sur la page web, il faut les mettre dans un Tab, puis dans un Groupe :

•Dans la boite de sélection du groupe, cliquez sur **Add new ui_group**,

•Créer un groupe (ici : "Le groupe") et faite de même pour le Tab (ici : "Test").

•Cliquez ensuite sur le bouton Done, puis déployer le flow en cliquant sur le bouton Deploy.

•On accède au site web généré par l'url : http://127.0.0.1:1880/ui [21].

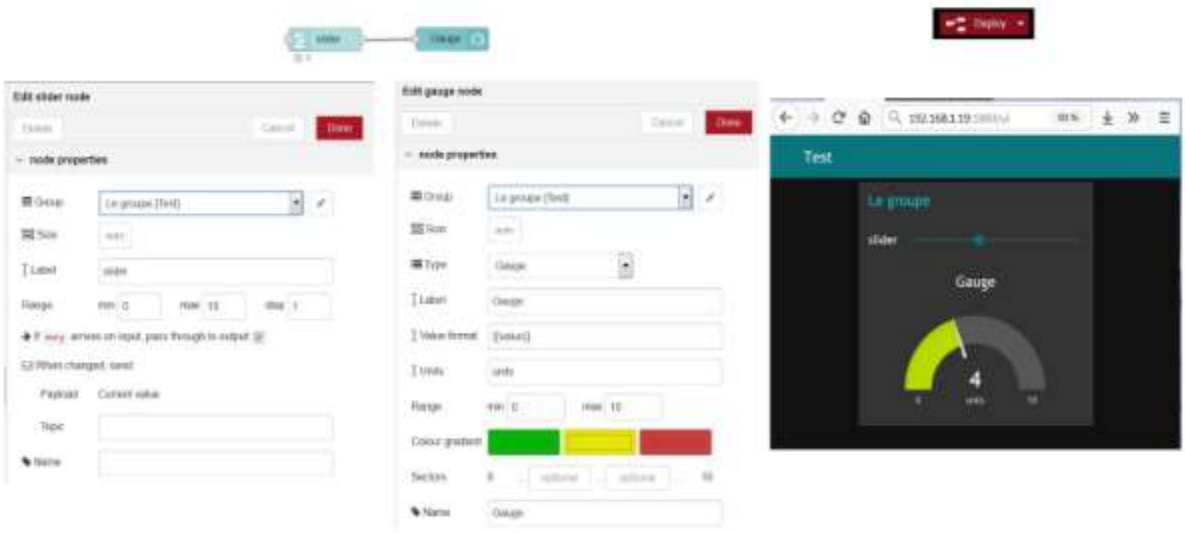


Figure II.16 Jauge pilotée par un slider

•Actionnez le Slider et constatez l'action sur la jauge (Gauge).

•Modifiez le flow en ajoutant un node Numeric, déployez et constatez le résultat :



Figure II.17 Ajout d'un node Numeric

•Pour conserver un historique des valeurs affichées, on peut utiliser un node Chart :

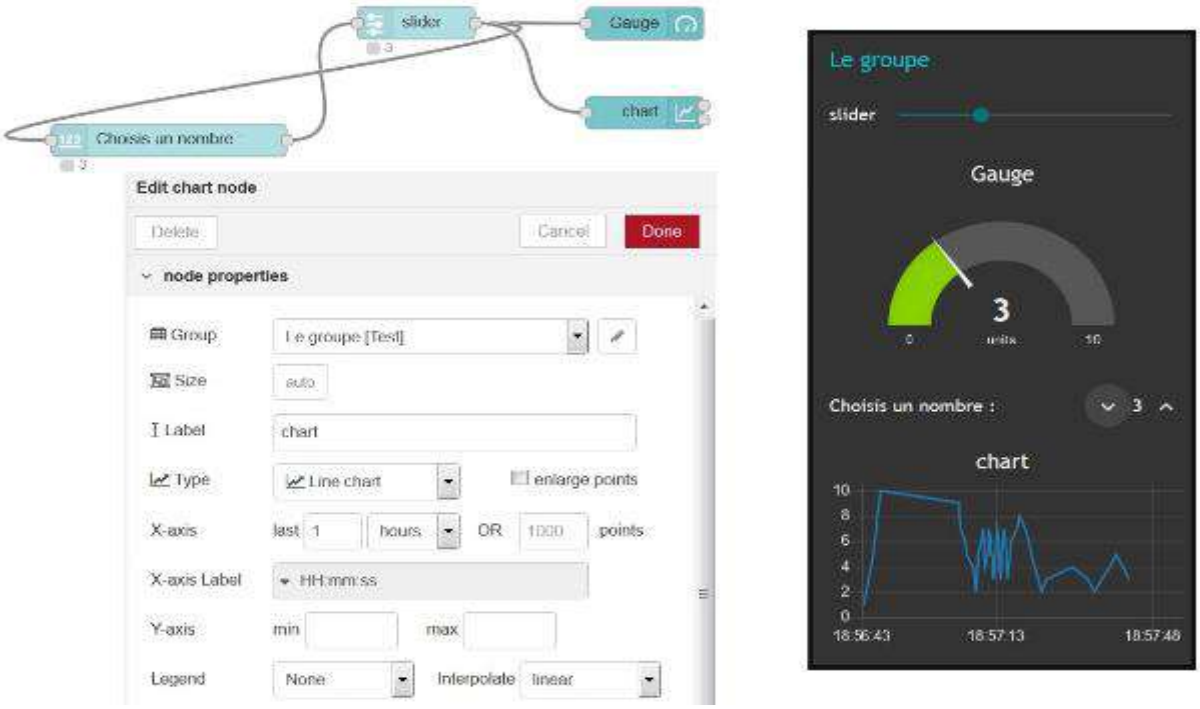


Figure II.18 Ajout d'un node Chart pour obtenir un graph de l'historique

CHAPITRE III :

Les composants utilisés

III.1 Introduction :

Dans ce chapitre nous expliquerons tous les composants utilisée dans notre projet et leur fonctionnement, nous avons utilisé Esp8266,NodeMCU ,Relais,Capteur,Heater et Fan.

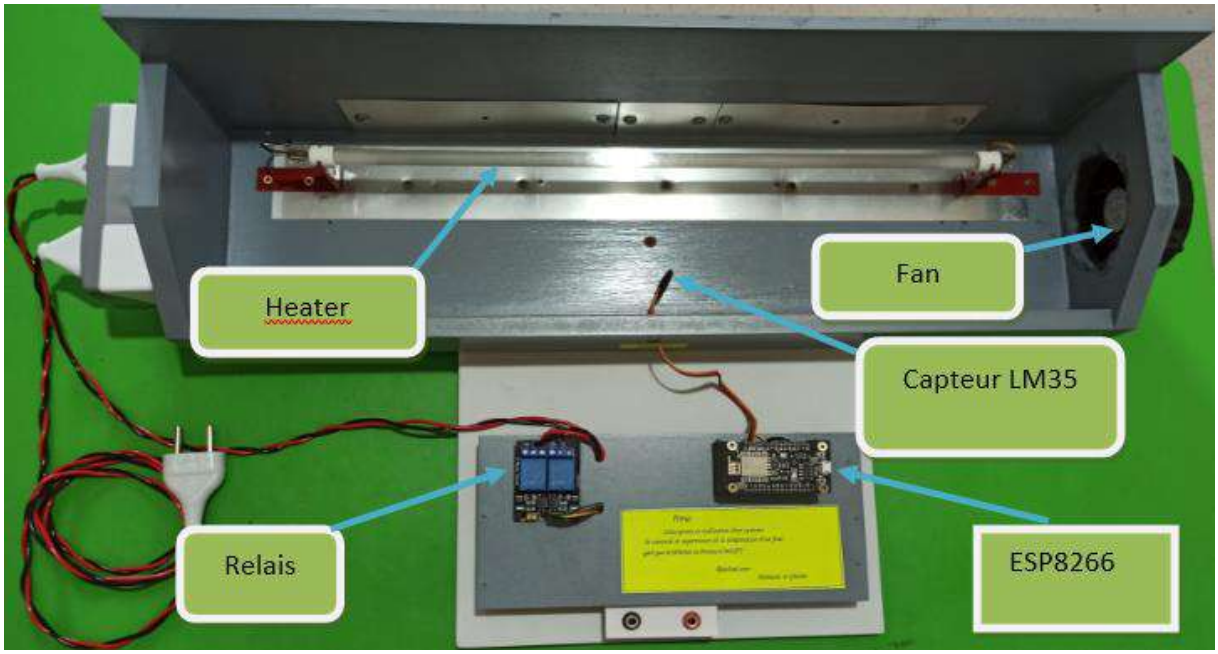


Figure III-1 : Réalisation de système

III-2 Esp8266 NodeMCU :

III-2-1- Définition :

NodeMCU est une plate-forme open source, matérielle et logicielle basée sur un SoC Wi-Fi ESP8266 fabriqué par Espressif Systems. Le terme 'NodeMCU' se réfère par défaut au firmware plutôt qu'aux kits de développement. Le firmware, permettant nativement l'exécution de scripts écrits en Lua.

L'ESP 8266 est un circuit intégré à microcontrôleur avec connexion Wi-Fi de taille réduite, permet de connecter un microcontrôleur à un réseau Wi-Fi et d'établir des connexions TCP/IP avec des commandes AT.

Il existe à ce jour plus de 12 versions de modules qui ont été construits à partir de ce composant. Chaque version est identifiée par une nomenclature sous la forme : ESP-01, ESP-02 …ESP-12 [22].

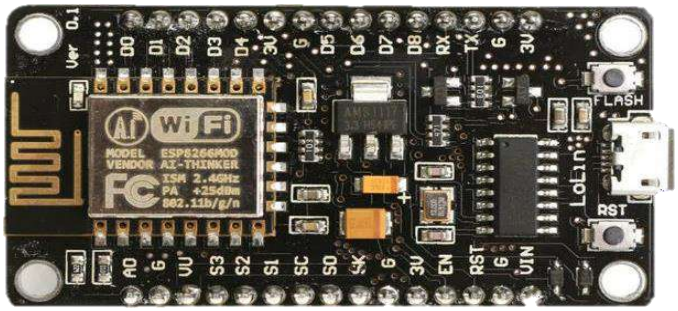


Figure III-2 La carte NodeMCU (ESP8266)

III-2-2 L'ESP8266 peut se programmer de plusieurs façons :

- Avec des scripts Lua, interprétés ou compilés, avec le firmware NodeMCU.
- En C++, avec l'IDE Arduino,
- En JavaScript, avec le firmware Espruino,
- En Micro Python, avec le firmware MicroPython5 ou Circuit Python.
- En C, avec le SDK d'Espressif.
- En C, avec le SDK esp-open-sdk7 basé sur la chaîne de compilation GCC.
- En Go, avec le Framework Gobot.

III-2-3 Description technique d'ESP8266

Les spécifications principales de l'ESP8266 sont :

- WIFI direct 802.11 b/g/n,
- Une pile de protocoles TCP/IP intégrés.
- Processeur 32 bits intégré de faible puissance.
- Il fonctionne à 3,3 V et ne possède pas de régulateur de tension, il faudra donc bien veiller à toujours l'alimenter en 3,3 V et non en 5 V ou depuis un pack de piles.
- Le processeur est cadencé à 80 MHz (contre 16 MHz pour carte Arduino UNO) et possède 80 KB de RAM (contre 2,5 KB pour une carte Arduino UNO).
- Il intègre une mémoire flash externe d'environ 400 KB (contre 32 KB pour une carte Arduino UNO) [22].

III-2-4- Les différents composant d'ESP8266 :

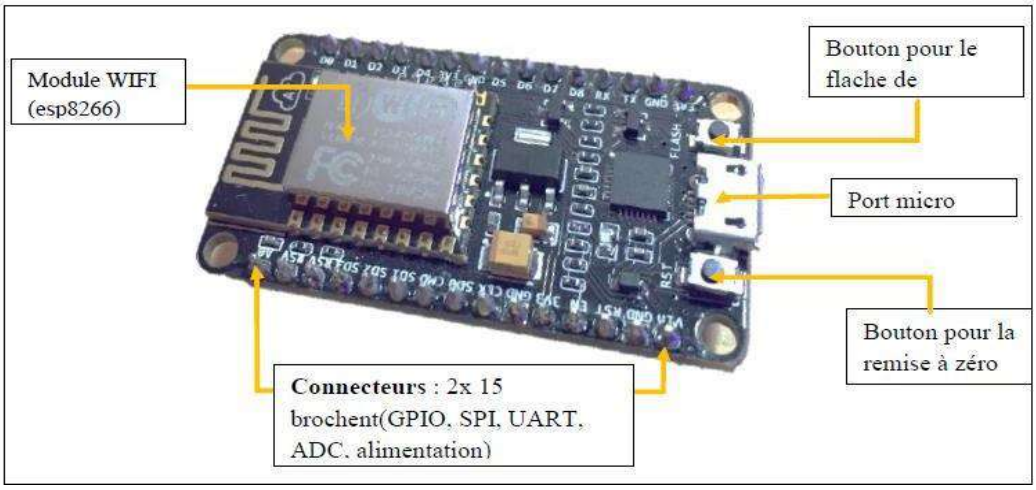


Figure III-3 Les différents composant d'esp8266

III-2-5- Support de carte NodeMCU :

C'est une base sur laquelle le NodeMCU est installé et présente plusieurs avantages :

- Résolution du problème des dimensions de la carte et se passer de la plaque de fixation des parties électroniques.
- Nous nous permettons d'utiliser une source d'alimentation externe (batteries) avec une valeur de tension allant Entre 6 et 24 volts.
- Il dispose d'une prise dédiée pour connecter des sources d'alimentation (batteries).

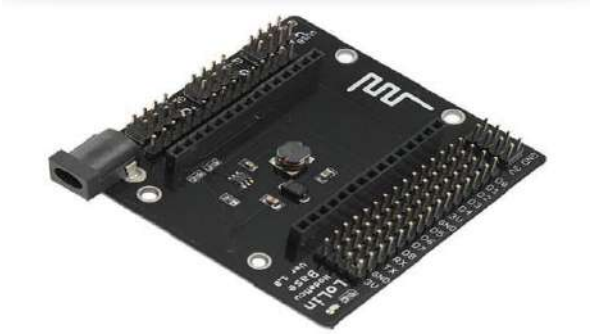


Figure III-4 Support de carte NodeMCU.



Figure III-5 Le Support avec la carte NodeMCU.

III-2-6 Schéma interne d'ESP8266 :

L'ESP8266 possède plusieurs entrées-sorties comme le montre le schéma ci-dessous :

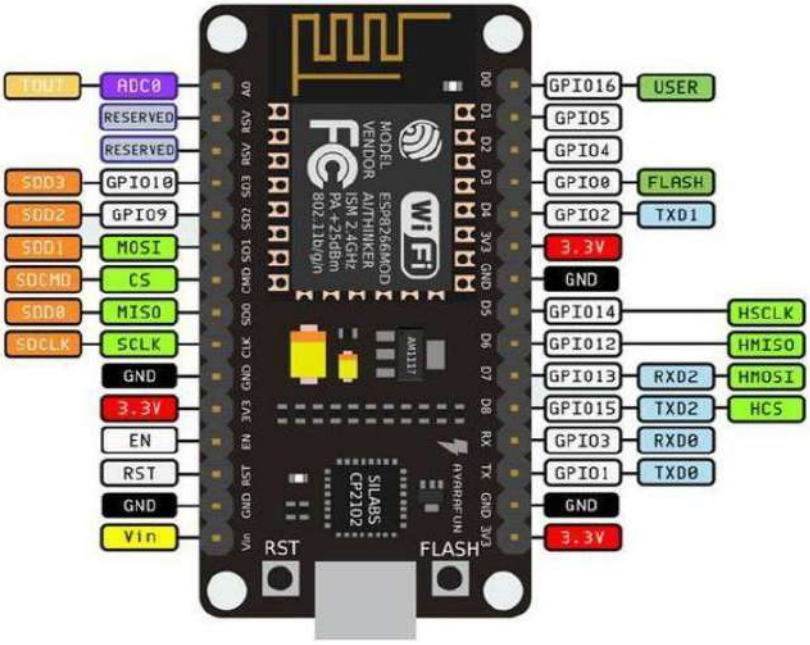


Figure III-6 Composition d'une carte NodeMCU (ESP8266)

III-2-7 Les broches de l'ESP8266 12E :

Le numéro GPIO ne correspond pas à l'étiquette sur la sérigraphie de la carte. Par exemple, D0 correspond à GPIO16 et D1 correspond à GPIO5.

Label	GPIO	Input	Output	Notes
D0	GPIO16	no interrupt	no PWM or I2C support	HIGH at boot used to wake up from deep sleep
D1	GPIO5	OK	OK	often used as SCL (I2C)
D2	GPIO4	OK	OK	often used as SDA (I2C)
D3	GPIO0	pulled up	OK	connected to FLASH button, boot fails if pulled LOW
D4	GPIO2	pulled up	OK	HIGH at boot connected to on-board LED, boot fails if pulled LOW
D5	GPIO14	OK	OK	SPI (SCLK)
D6	GPIO12	OK	OK	SPI (MISO)
D7	GPIO13	OK	OK	SPI (MOSI)
D8	GPIO15	pulled to GND	OK	SPI (CS) Boot fails if pulled HIGH
RX	GPIO3	OK	RX pin	HIGH at boot
TX	GPIO1	TX pin	OK	HIGH at boot debug output at boot, boot fails if pulled LOW
A0	ADC0	Analog Input	X	

FigureIII.7 Les broches de l'ESP8266 12E

III-2-8 Les avantages d'ESP8266 :

- ✓ Pas cher.
- ✓ Environnement de programmation clair et simple (Arduino).
- ✓ Outils de développement Multiplateforme : tourne sous Windows, Macintosh et Linux.
- ✓ Nombreuses librairies disponibles avec diverses fonctions implémentées.
- ✓ Logiciel et matériel open source et extensible.
- ✓ Nombreux conseils, tutoriaux et exemples en ligne (forums, sites persos, etc.).
- ✓ Existence de « Shield » (boucliers en français) : ce sont des cartes supplémentaires [22].

III-2-9 Comparaison entre Arduino_Uno et ESP8266 12E:

	ArduinoUno	**ESP8266**
Micro-contrôleur	ATmel ATmega328p	Tensilica Xtensa LX106
Espace d'adressage	8bits	processeur 32bits RISC
Fréquence d'horloge	16 MHz	80 MHz – 160 MHz (1)
Mémoire flash	**32 Ko**	aucune de base et de 512 Ko à **4 Mo** selon les modèles améliorés (2)
Mémoire RAM	2 Ko	96 Ko
E/S numérique	20 (5V) dont 6 capables de PWM	16 (3,3V), toutes capables de PWM
Entrées analogiques	6 (10 bits)	1 (10 bits, 1V max (3))
Communication	1x UART;1x I2C; 1x SPI	1,5x UART (4);1x I2C; 1x SPI

FigureIII.8 Comparaison entre Arduino_Uno et ESP8266 12E

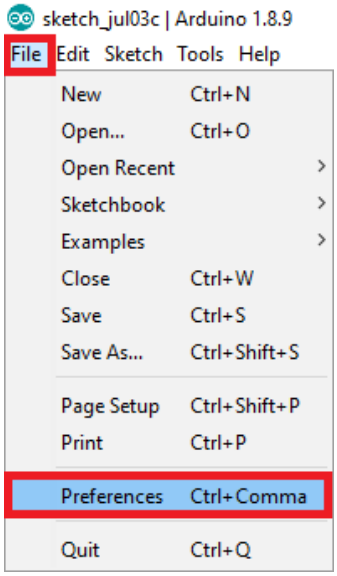
III-2-10 Installation _ ESP8266_sur IDE Arduino :

Pour programmer l'ESP8266 à l'aide de l'IDE Arduino et de son langage de programmation on doit d'abordl'installer sur cet environnement de programmation IDE Arduino.

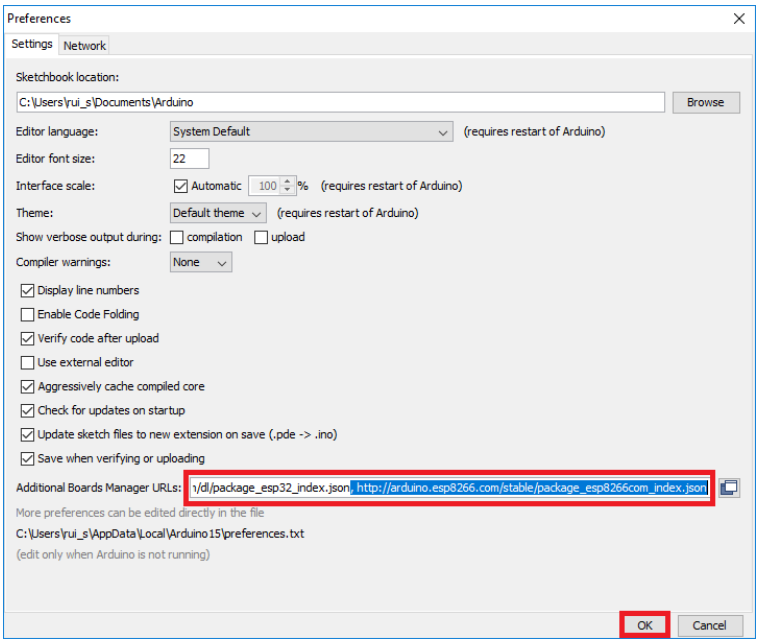
NB : assurons-nous que la version récente de l'IDE Arduino est installée sur notre ordinateur

Pour installer la carte ESP8266 dans notre IDE Arduino, suivons ces instructions suivantes :

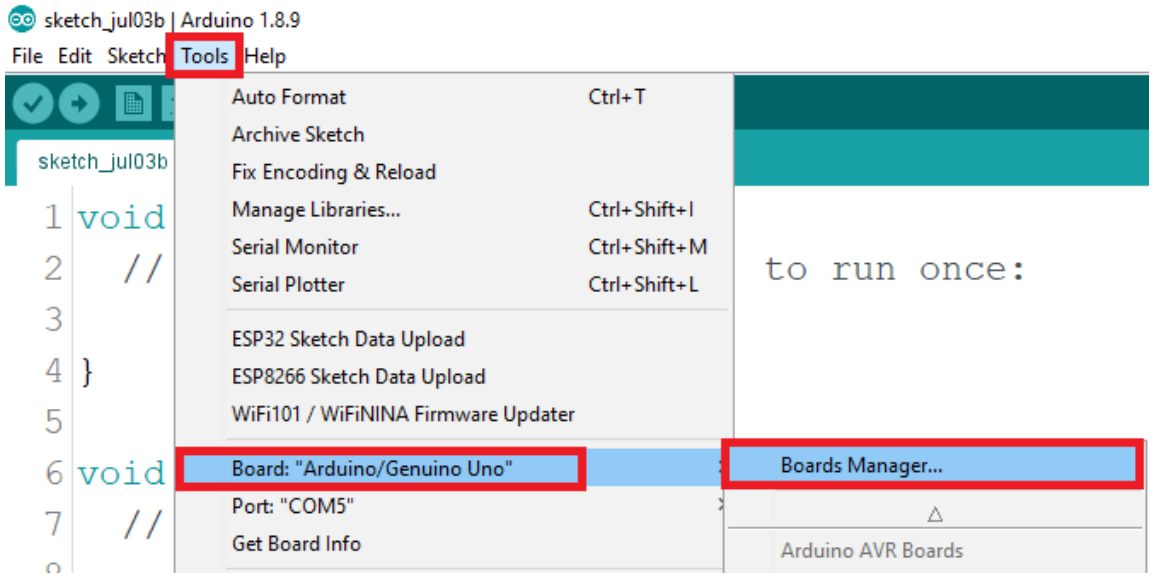
-Dans votre IDE Arduino, allez dans **Fichier** > **Préférences**



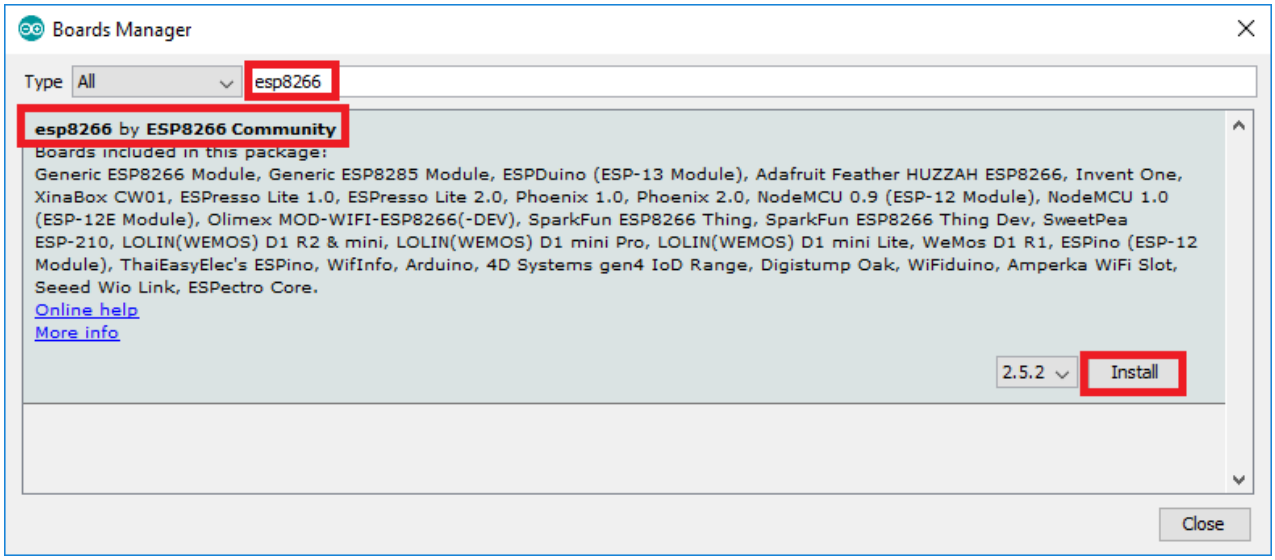
-Entrez **http://arduino.esp8266.com/stable/package_esp8266com_index.json** dans le champ "AdditionalBoards Manager URLs" comme indiqué dans la figure ci-dessous. Ensuite, cliquez sur le bouton « OK » :



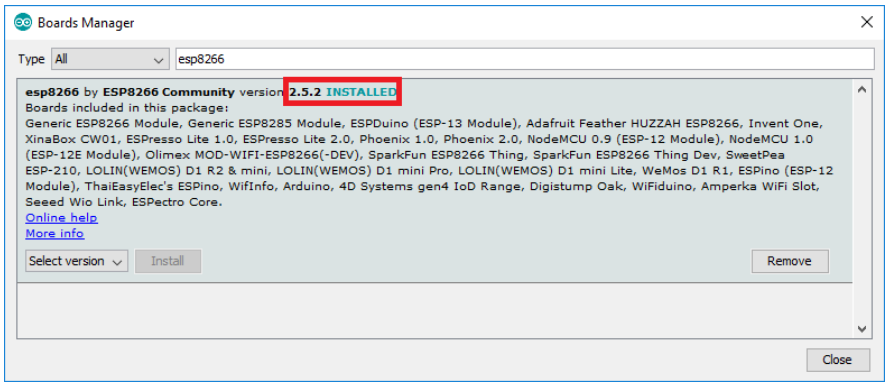
-Ouvrez le gestionnaire de tableaux. Allez dans **Outils** > **Tableau** > **Gestionnaire de tableaux…**



-Recherchez **ESP8266** et appuyez sur le bouton d'installation pour la « **communauté ESP8266 par ESP8266** » :



-L’installation devra être être installé après quelques secondes.



III-2-10-1 Tester l’installation à l’aide d’un exemple :

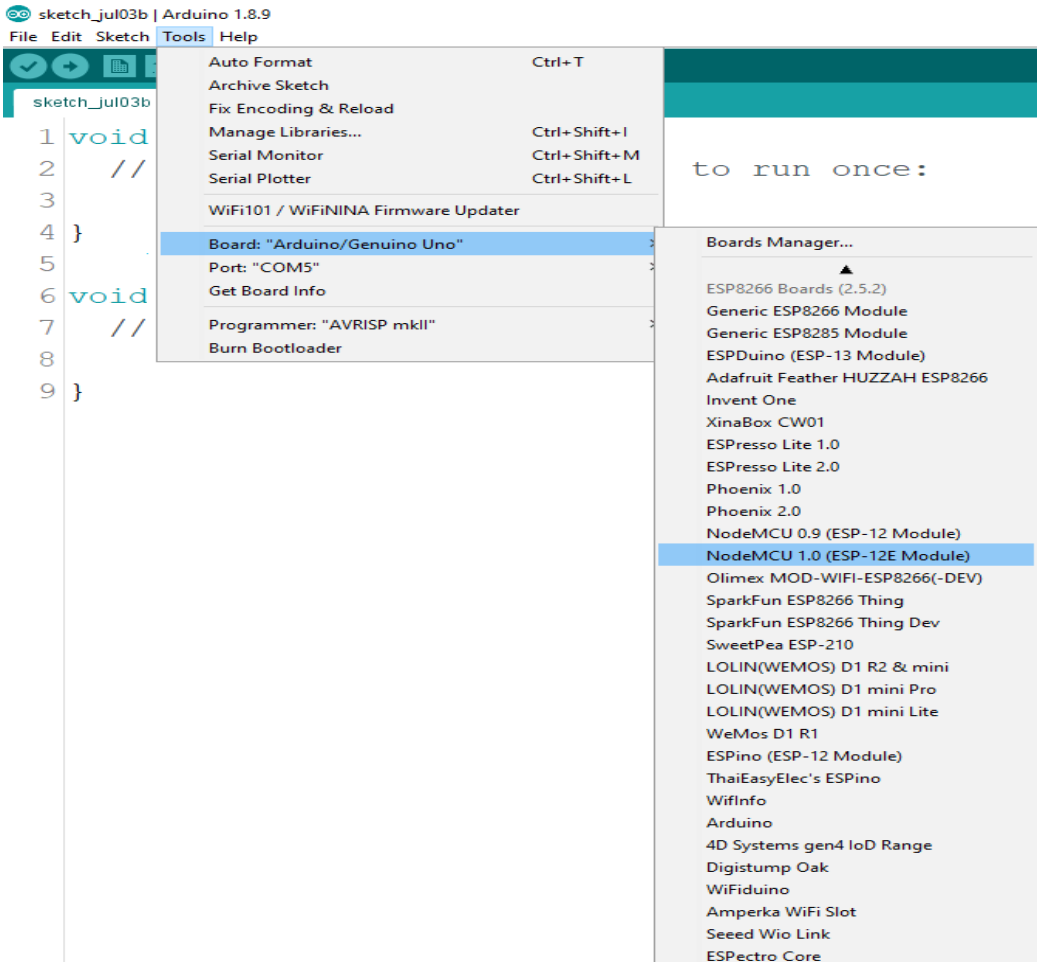
Soit le programme Blink suivant :

```
int pin =2;

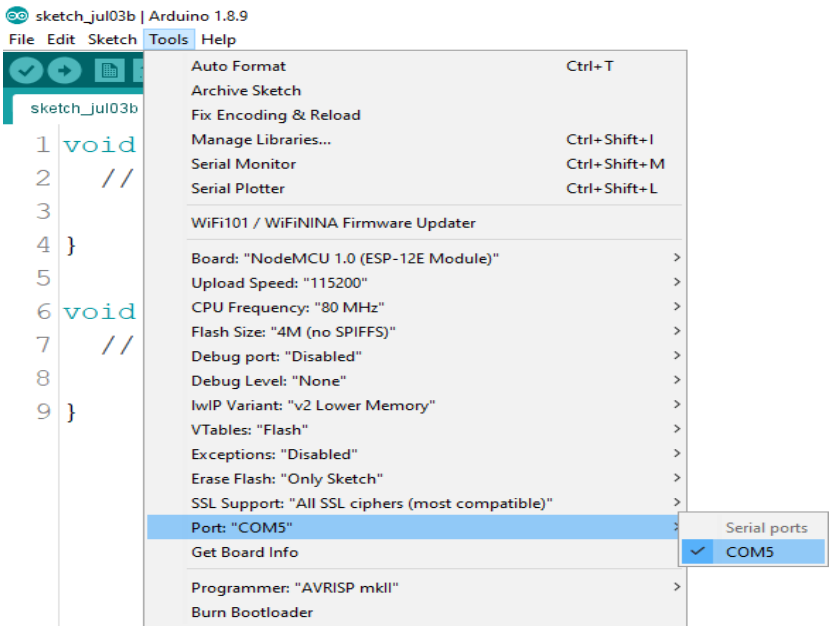

voidsetup(){
// initialize GPIO 2 as an output.
pinMode(pin, OUTPUT);
}


// theloopfunctionruns over and over againforever
voidloop(){
digitalWrite(pin, HIGH);// turn the LED on (HIGH is the voltage level)
delay(1000);// wait for a second
digitalWrite(pin, LOW);// turn the LED off by making the voltage LOW
delay(1000);// wait for a second
}
```

Branchons la carte ESP8266 à notre ordinateur. Puis on sélectionne la bonne carte :



Sélectionner le Port COM :



Télèverserensuite le code Blink sur notre ESP8266, puis voyons le clignotement de la LED selon le schéma de la figure suivante :

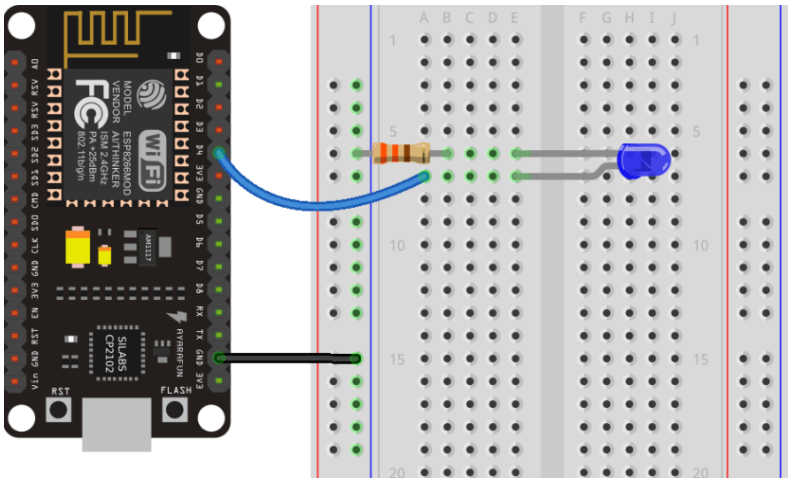


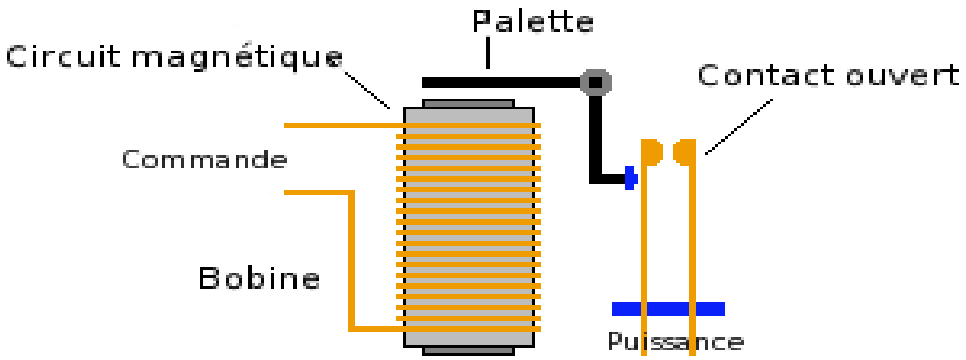
Figure III.9 Schéma de Télèverserensuite le code Blink sur notre ESP8266

III-3 Le relais électromécanique :

Un relais est un interrupteur qui se commande avec une tension continue ou alternative de faible puissance. La partie interrupteur sert à piloter des charges secteur de forte puissance (plusieurs dizaines d'ampères) [23].

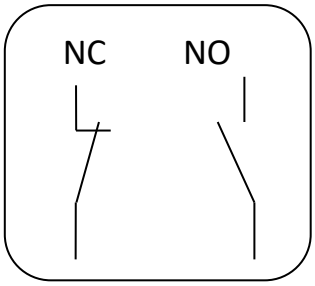
III-3-1 Constitution :

- Un relais est composé principalement d'un électroaimant, qui lorsqu'il est alimenté, transmet une force à un système de commutation électrique : les contacts [23].



Figur III.10 : composants de relais

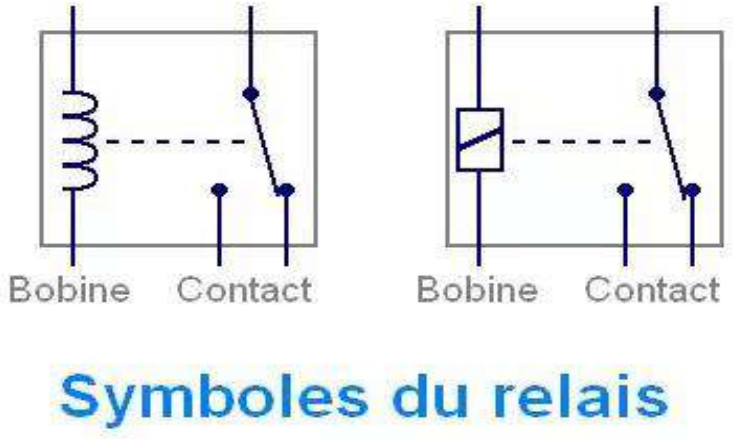
- L'électroaimant (la bobine) peut être, suivant les spécifications et besoins, alimenté en TBT (Très Basse Tension) (moins de 12 V, 24 V, 48 V) continu ou alternatif ou en BT (Basse Tension) (230 V, 400 V).



Figur III.11 Le système de commutation

- Le système de commutation peut être composé d'un ou plusieurs interrupteurs simple effet appelés contacts normalement ouverts (NO) ou normalement fermés (NF ou NC). Ces commutateurs sont adaptés aux courants et à la gamme de tensions à transmettre à la partie puissance.
- Divers systèmes mécaniques ou pneumatiques peuvent créer un retard à l'enclenchement ou au relâchement (on dit alors que les relais sont "temporisés").

- Dans les systèmes mettant en œuvre une grande puissance, on appelle les relais des "contacteurs" [23].



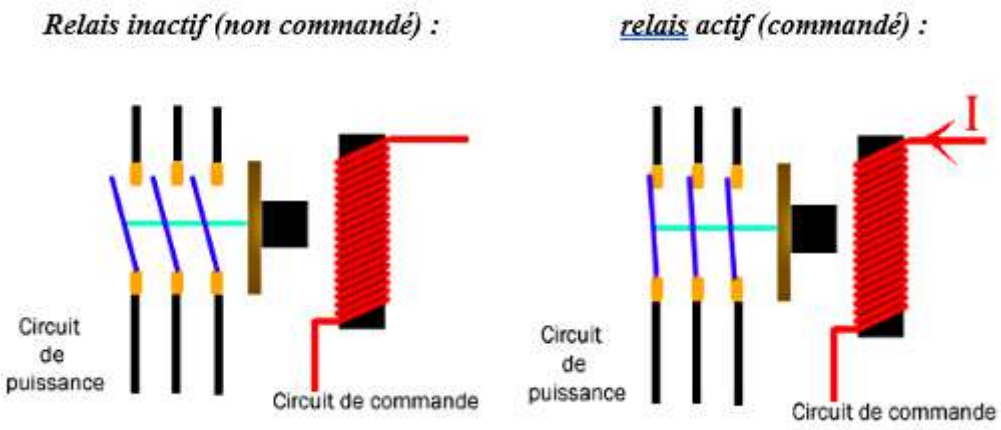
Figur III.12 : SUMBOLES DE RELAIS

III-3-2 Fonctionnement :

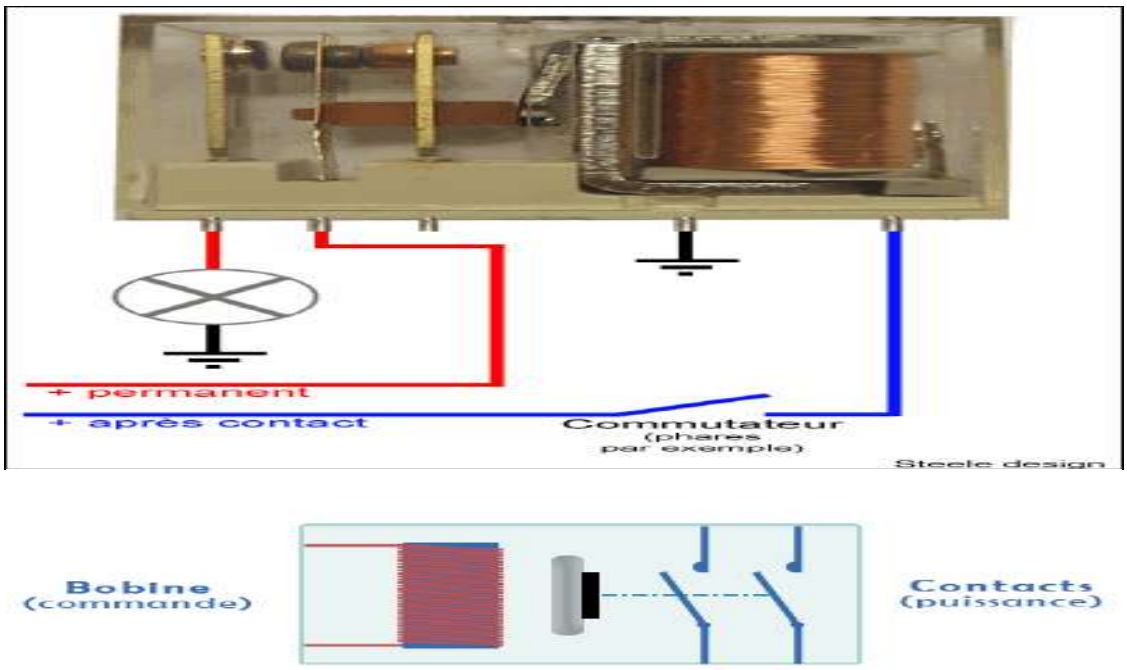
Un relais peut être monostable ou bistable :

III-3-2-1 Fonctionnement monostable :

Les contacts commutent quand la bobine est alimentée et le retour à l'état initial se fait quand la bobine n'est plus alimentée (à l'aide d'un ressort en général). C'est le relais le plus répandu [23].



Figur III.13 : Schéma commandé et non commandé de relais



Figur III.14 : 2 petites animations de relais

III-3-2-2Fonctionnement bistable à une bobine :

On alimente la bobine pour que les contacts commutent : l'état ne change pas quand la bobine n'est plus alimentée, un système mécanique bloque le retour. Pour revenir à l'état initial, on alimente à nouveau la bobine pour débloquer le mécanisme, dans certain cas en inversant la polarité de l'alimentation (dans le bâtiment on appelle ce type de relais un "télérupteur) [23].

III-3-2-3Fonctionnement bistable à deux bobines :

On alimente la première bobine pour que les contacts commutent : l'état ne change pas quand la bobine n'est plus excitée. Pour revenir à l'état initial, on alimente la deuxième bobine.Quand un courant circule dans la bobine (circuit de commande), celle-ci crée un champ magnétique qui fait se fermer les contacts (circuit de puissance). Quand le courant est coupé, un ressort rouvre les contacts [23].

III-3-3 Inconvénients du relais électromécanique :

- Il peut être relativement bruyant

- Il a une durée de vie limitée car les pièces en mouvement s'usent. Quand on doit commander très souvent un élément on utilise plutôt des composants électroniques (relais statique, transistors, ...)

III-4 La résistance :

La résistance considérée parmi les composants les plus utilisés dans le monde de l'électronique, est un dipôle qui s'oppose au passage du courant électrique qui la traverse, plus une résistance possède une valeur élevée et plus celle-ci résiste à la circulation du courant. Elle est notamment utilisée pour protéger les diodes électroluminescentes (DEL) qui ne supportent qu'une tension faible. Sur un schéma électronique, son symbole est le suivant [24]:

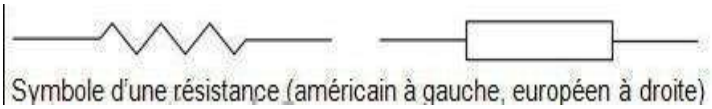


Figure III.15 : Symbole d'une résistance électrique

La résistance a les caractéristiques suivantes :

- ✓ notée par le symbole : **R**.
- ✓ La résistance **R** d'un conducteur ohmique est le quotient de la tension **U** entre ses bornes par l'intensité du courant **I** qui le traverse : **R** = **U** (voltage) / **I** (l'ampérage).
- ✓ L'unité de la résistance électrique est le **Ohm** de symbole Ω.
- ✓ L'appareil de mesure de la résistance électrique est l'**ohmmètre**.

Le plus souvent, la résistance se présente avec des bagues de couleurs (anneaux) autour de celle-ci. Chaque couleur correspond à un chiffre. La correspondance entre les chiffres et les couleurs des anneaux constitue ce qu'on appelle le code des couleurs et permet de déterminer la valeur d'une résistance ainsi que sa tolérance. Il existe trois types de résistances : les résistances à 4, 5 et 6 anneaux (couleurs) [25].

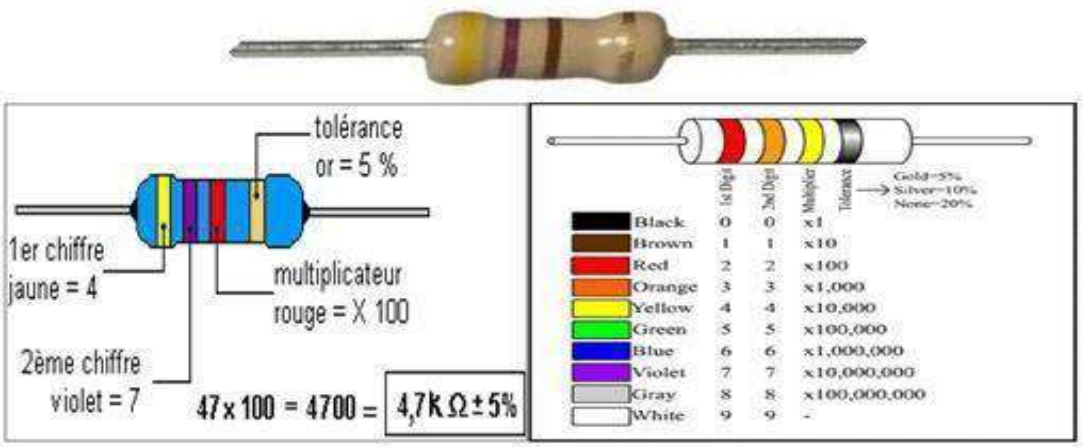
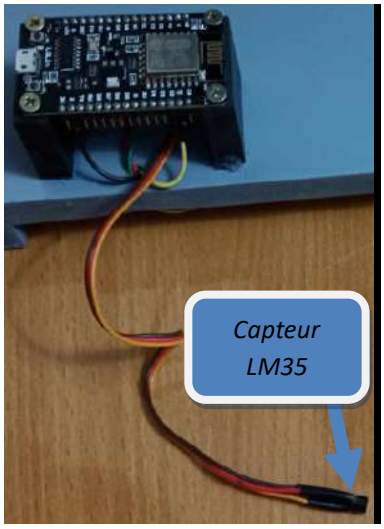
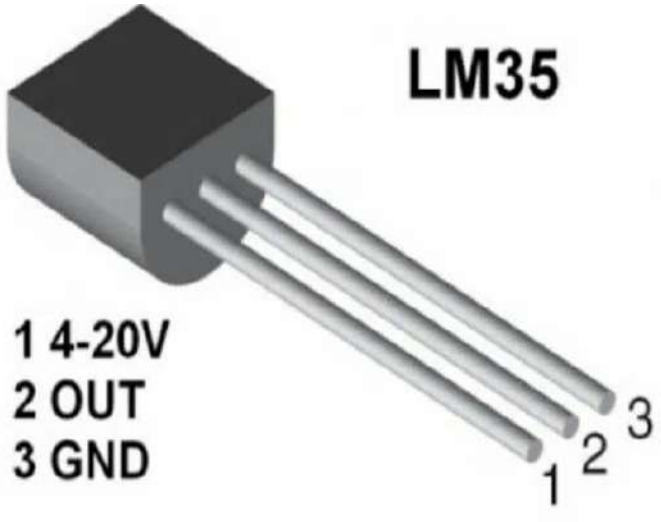


Figure III.16 : Résistance a 4 anneaux

III-5 Le capteur LM35 :





Figur III.17 :Capteur LM35

III-5-1 Définition du LM35 :

Le capteur de température LM35 est un capteur analogique de température fabriqué par Texas Instruments. Il est extrêmement populaire en électronique, car précis, peu couteux, très simple d'utilisation et d'une fiabilité à toute épreuve.

Le capteur de température LM35 est capable de mesurer des températures allant de -55°C à +150°C dans sa version la plus précise et avec le montage adéquat, de quoi mesurer n'importe quelle température.

N.B. Les versions grand publiques sont capables de mesurer des températures comprises entre -40°C et +110°C.

La sortie analogique du capteur est proportionnelle à la température. Il suffit de mesurer la tension en sortie du capteur pour en déduire la température. Chaque degré Celsius correspond à une tension de +10mV.

Le capteur LM35 supporte des températures assez extrêmes (jusqu'à -55°C / +150°C), mais il n'en est pas de même pour les cartes Arduino.

Si vous soumettez une carte Arduino "classique" à ces températures extrêmes, elle va purement et simplement cesser de fonctionner ou se dégrader très rapidement. Cela est valable pour les cartes Arduino, mais aussi pour tous autres circuits électroniques fabriqués suivant les standards "grand public" (et non "industriels").

Pour information, les gammes de température classique en électronique sont les suivantes :

Grand public : 0°C ~ 70°C

Industrie : -40°C ~ 85°C

Militaire : -55°C ~ 125°C

Une des grandes forces du capteur LM35, qui fait sa popularité, c'est sa pré calibration en sortie d'usine. Tous les capteurs LM35 sont calibrés en degré Celsius lors de la fabrication. Cela signifie que vous n'avez absolument rien à faire pour calibrer le capteur, il l'est déjà au moment où vous le sortez de son sachet !

La précision garantie par le fabricant est de +/-1°C à 25°C et +/-1.5°C à -55°C ou +150°C pour la version la moins précise, ce qui est largement suffisant pour la plupart des

applications. La version plus précise du LM35 (nommée "LM35A") a une précision garantie de +/-0.5°C à 25°C et +/-1°C à -55°C ou +150°C.

Une autre des grandes forces du capteur LM35, c'est sa linéarité exemplaire : moins de 1°C d'erreur sur la plage complète de -55°C à +150°C. Comme chaque degré Celsius correspond à 10mV (soit 0.01 volt) et que la sortie du capteur est (quasi) parfaitement linéaire, convertir une mesure en température se résume à faire un bête produit en croix.

Pour en finir avec les compliments, le capteur LM35 fonctionne avec n'importe quelle tension d'alimentation comprise entre 4 volts et 30 volts, ce qui permet de l'utiliser dans virtuellement n'importe quel montage numérique ou analogique.

Passons maintenant aux points négatifs, ceux qui fâchent. Le capteur LM35 ne fonctionne pas en dessous de 4 volts, donc oubliez l'utilisation d'un LM35 avec des cartes Arduino 3.3 volts (Due, Zéro, etc.), sans alimentation 5 volts externe, ça ne marche pas.

N.B. Le capteur LM35 peut être utilisé avec une entrée analogique 3v3, car 150°C = 1,5V, cependant, une alimentation 5v reste obligatoire pour que le capteur fonctionne.

Autre souci, le capteur LM35 nécessite une alimentation négative pour mesurer des températures en dessous de 0°C. La sortie est proportionnelle à la température, par conséquent -10°C équivaut à -0,1 volt.

De fait, avec une simple alimentation 5v, comme celle des cartes Arduino, il n'est possible de mesurer que des températures positives [26].

Brochage du lm 35 :

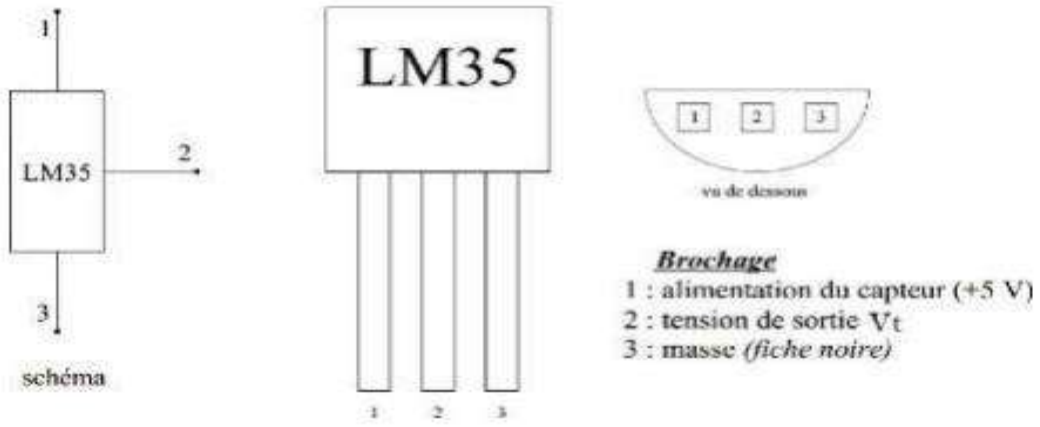


Figure III.18 : brochage du lm35

III-5-2 Calibration du lm35 :

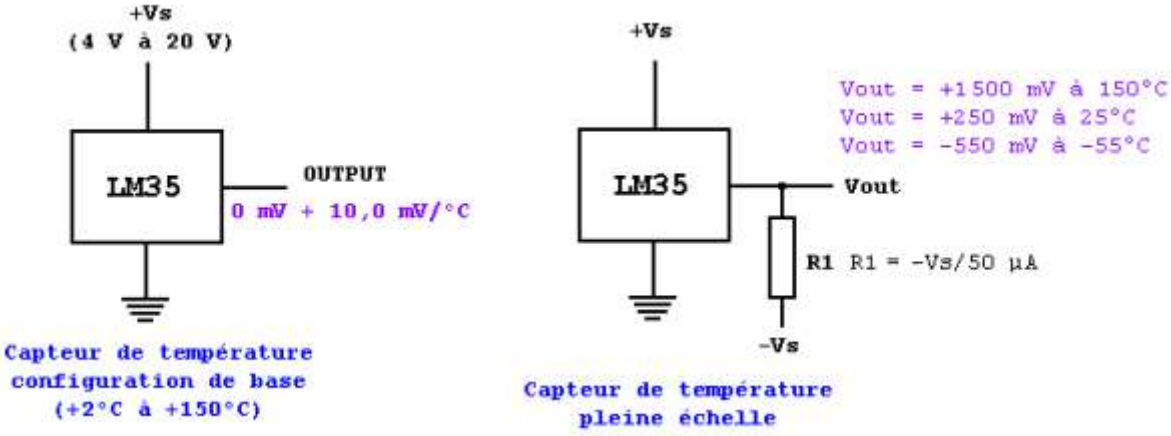


Figure III.19 : calibration du lm35.

Chapitre IV

Etude technique

Et pratiqu

IV.1 Introduction :

Dans ce chapitre, nous allons développer les étapes nécessaires pour aboutir à notre objectif et faciliter la réalisation de notre projet ainsi que les Outils de programmation (Arduino,node-red).

IV.2 Les étapes de la réalisation :

Première étape :

C'est l'étape la plus difficile car elle consiste en la recherche des Informations bibliographiques et techniques à utiliser pour aboutir à l'objectif.

Deuxième étape :

Etablir le circuit électrique dans l'environnement node-red puis programmer ESP8266 par son IDE arduino (logiciel Arduino).

Troisième étape :

Réalisation du montage complet.

IV.3 Etude technique :

IV.3.1 Principe et mode de fonctionnement :

Le principe de fonctionnement de notre projet est conception et réalisation d'un systeme de control et supervision de la température d'un four a base de esp8266 via node-red.

IV.3.2 Outils de programmations :

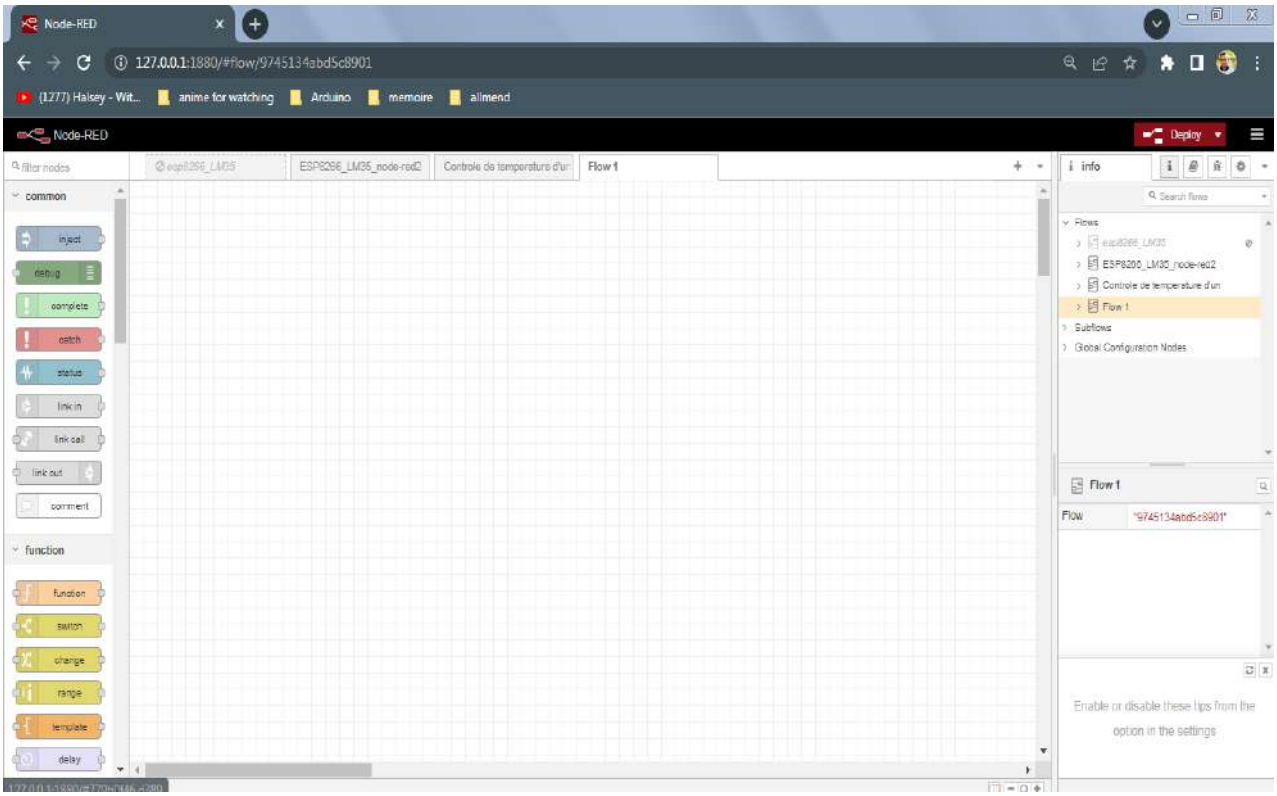
L'Arduino peut être programmé par divers langages et pour chaque langage il ya plusieurs compilateurs, nous avons choisi le langage Arduino avec son compilateur integé.

- Logiciel IDE Arduino

Avant de passer à la programmation, nous devons réaliser un organigramme qui explique le déroulement des différentes séquences, tant intérieures qu'extérieures: il comportera plusieurs boucles dont la fin d'exécution succède toujours à son commencement.

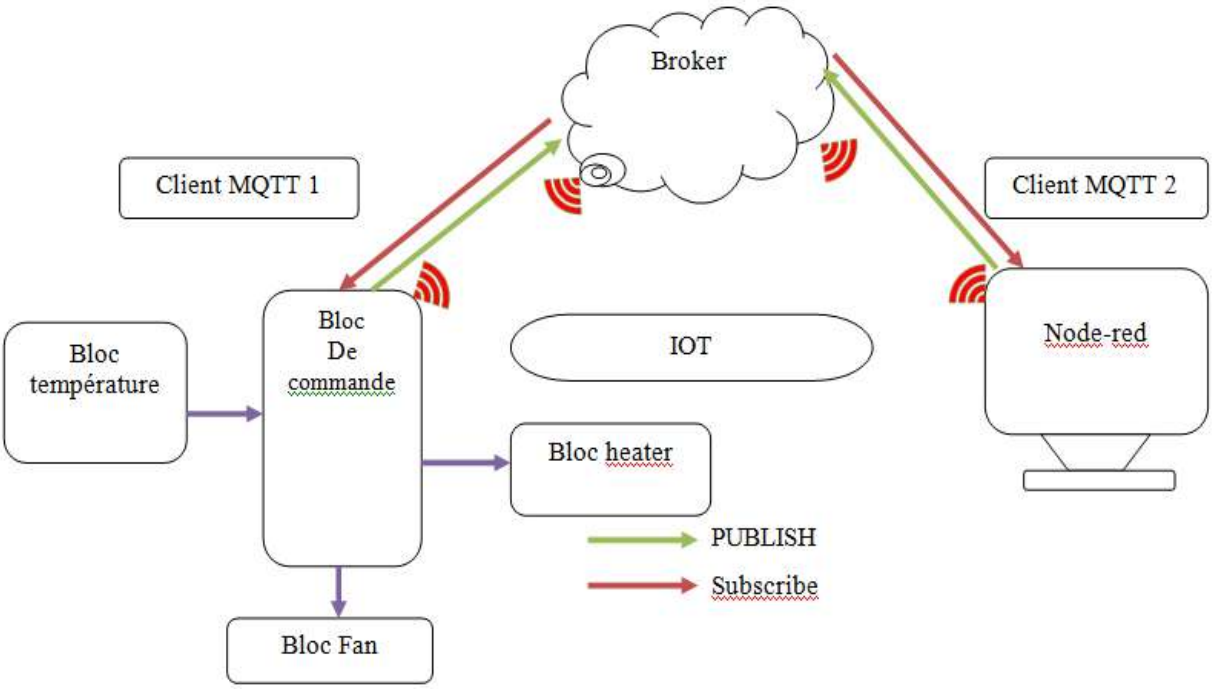
• Logiciel Node-red :

Node-red est un outil puissant pour construire des applications de l'Internet des Objets en mettant l'accent sur la simplification de la programmation qui se fait grâce à des blocs de code prédéfinis, appelés nodes pour effectuer des tâches. Il utilise une approche de programmation visuelle qui permet aux développeurs de connecter les blocs de code ensemble.



FigureIV.1 : node-red

IV.4 Schéma synoptique de système :



FigureIV.2 : Schéma synoptique de système

✓ Bloc température :

Il est représenté dans le capteur LM35, qui capte la température du four.

✓ Bloc Fan :

C'est le ventilateur qui se charge de ventiler le four pour baisser la température du four.

✓ Bloc Heater :

C'est la résistance qui se charge de chauffer le four.

✓ Bloc de commende :

C'est l'ESP8266 qui est responsable de la collecte des données du capteur LM35 et du contrôle du Heater et du Fan et qui est également responsable de la liaison au serveur du Broker en s'inscrivant sur Topic (Subsucribe), il doit être enregistré pour pouvoir envoyer des informations à BROKER (publier)

✓ Bloc de Broker :

Le Broker véritablement maitre à bord joue un rôle crucial dans l'acquisition et la distribution des données entre le capteur et le/les utilisateurs abonnés. Le principe de l'abonnement du protocole MQTT s'appuie sur le fait qu'un « publisher » va émettre une information tout en se catégorisant dans un sujet. Le Broker est capable de « ***bufferiser*** » (mise ne mémoire tampon) toutes les informations provenant de différents capteurs ou objets connectés. Ainsi, si l'utilisateur est hors ligne (Smartphone éteint par exemple), les informations émises vers le Broker pendant cette période d'inactivité remonteront immédiatement lors de la prochaine connexion (principe de **Facebook Messenger** par ex.). L'utilisateur recevra ainsi toutes les données émises pendant ce laps d'inactivité grâce au Broker capable de gérer la notion de mise en mémoire tampon. L'utilisateur en liaison constante « **Heartbeat** » avec le Broker bénéficie en temps réel des informations émises par le Broker. MQTT utilise une certaine hiérarchisation très pratique.

IV.4.1 Schéma Bloc du client MQTT1 :

Nous avons utilisé plusieurs composants interconnectés sur les broches d'entrée et de sortie de la carte ESP8266, pour assurer le fonctionnement du système, pour obtenir et contrôler la valeur de la température.

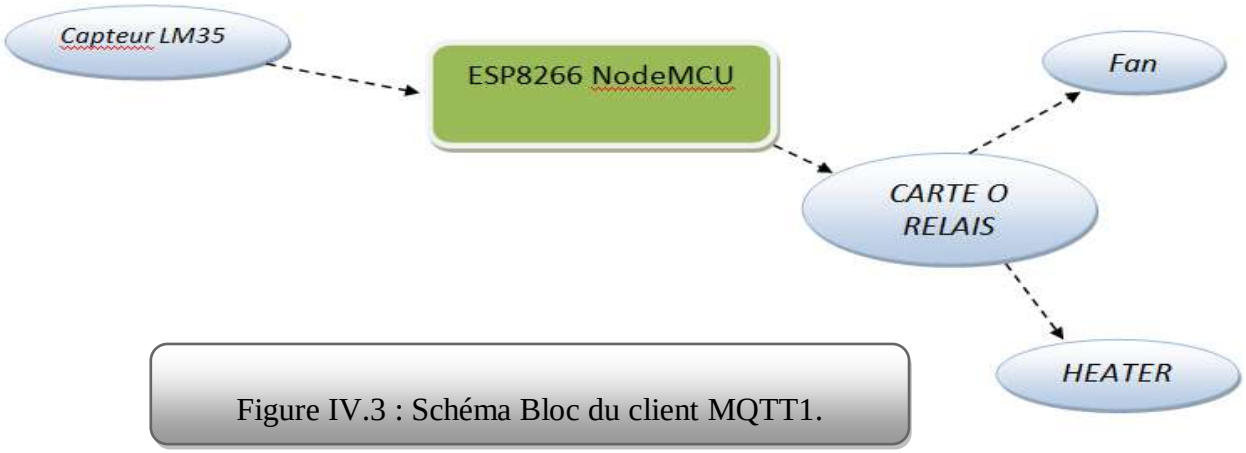
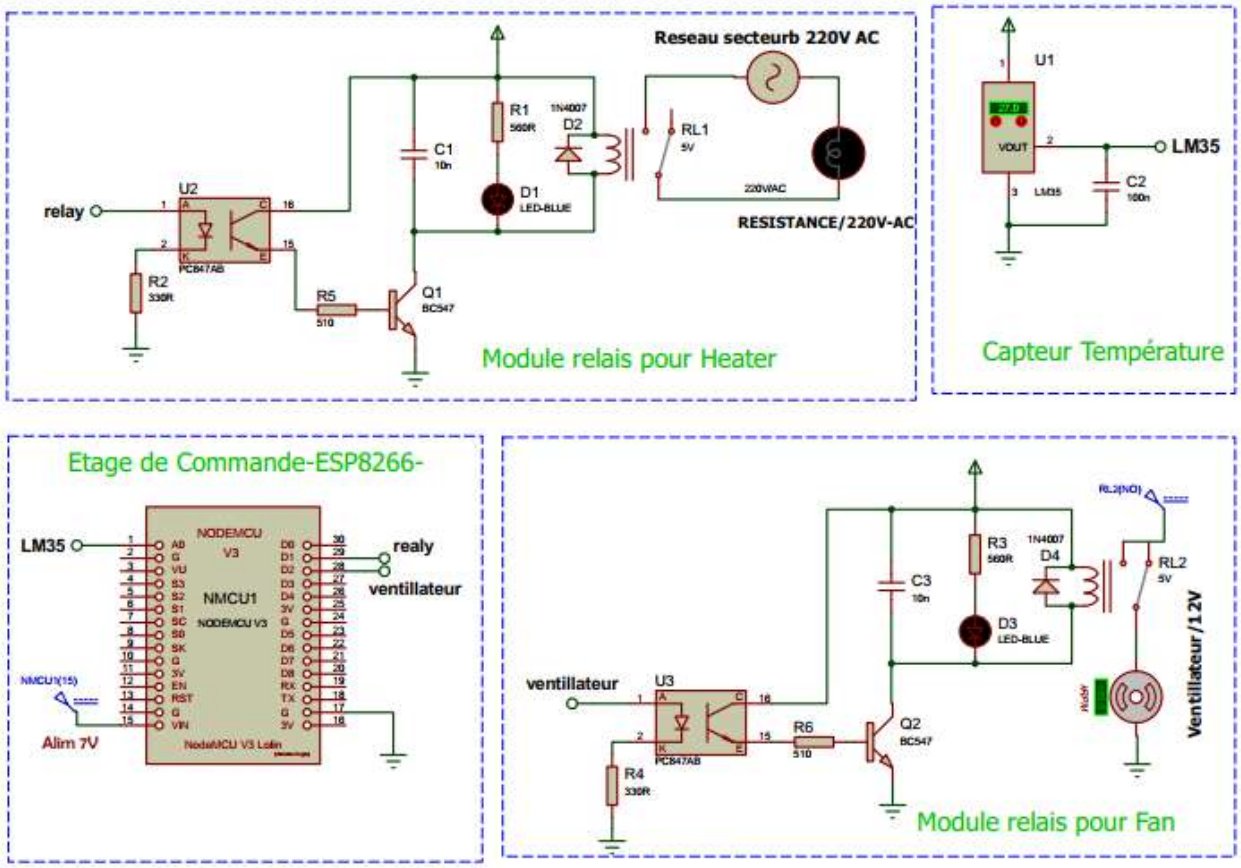


Figure IV.3 : Schéma Bloc du client MQTT1.

IV. 5 Schéma électrique de montage :

Nous avons créé un schéma à l'aide d'un programme Proteus ISIS.



FigureIV.4 : Schéma électrique

Le principe de ce montage est le suivant :

- Module relais pour heater et pour relais pour fan :

La fonction principale du relais dans notre circuit est de séparer le ventilateur et la résistance du circuit d'alimentation dans le but d'isoler et de connecter la haute tension ou le courant et d'assurer la sécurité, le relais de résistance est connecté à la broche 29(D1/GPO5) et le relais du ventilateur est connecté à la broche 28(D2/GPO4) sur l'ESP8266

- Etage de commande ESP8266 :

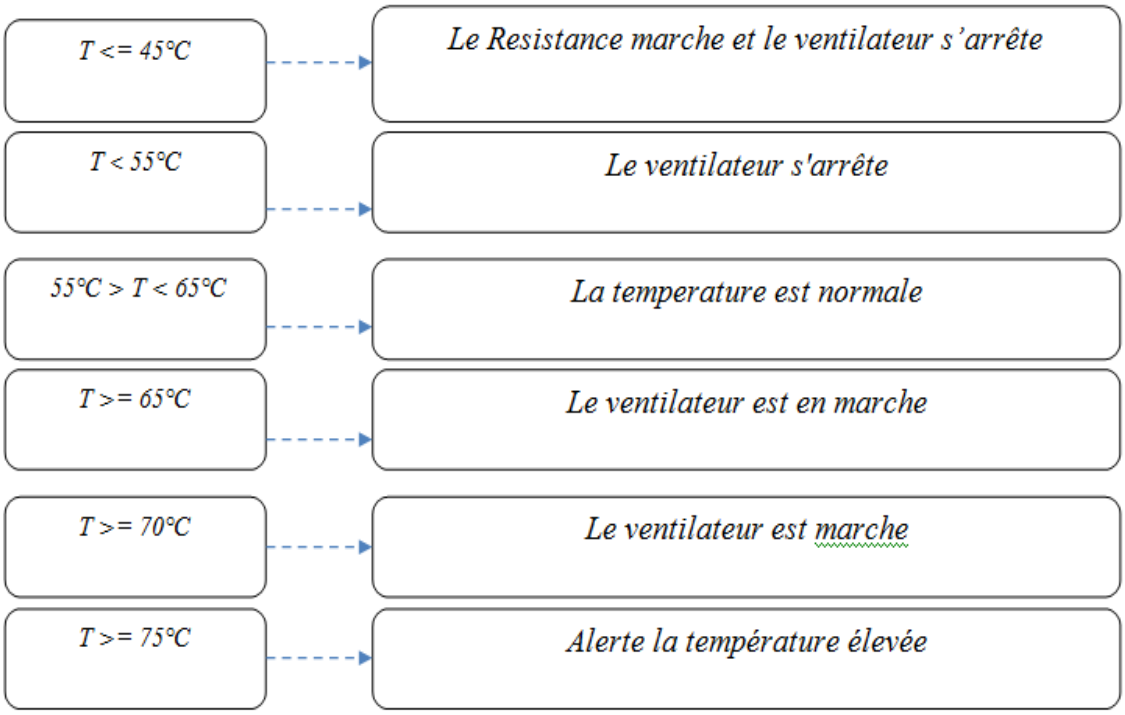
C'est la partie contrôle qui relie le Broker au WiFi en s'inscrivant dans topic, elle analyse aussi les données obtenues du capteur LM35 et contrôle la résistance et le ventilateur selon la programmation.

- Le capteur LM35 :

Le capteur LM35, qui joue le rôle de calcul de la valeur de la température, est connecté à la broche 1 (ADC0) de l'ESP8266, cette broche est configurée en entrée, c'est-à-dire qu'elle recevra la température du capteur LM35 , Étant donné que le capteur a une sortie de tension, nous devons construire une équation pour convertir la tension en une lecture correcte de la température en degrés Celsius.

IV.6 Régulation et Contrôle de la température :

Pour contrôler et régler la température dans le four, nous avons placé un capteur LM35, une résistance et un ventilateur connecté à l'ESP8266, qui contient le WIFI pour envoyer et recevoir des informations au Broker, Nous avons choisi certaines températures à étudier

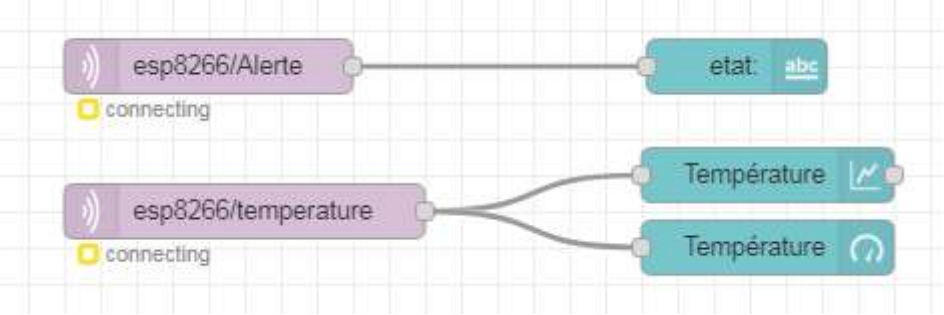


FigureIV.5 : Régulation et Contrôle de la température

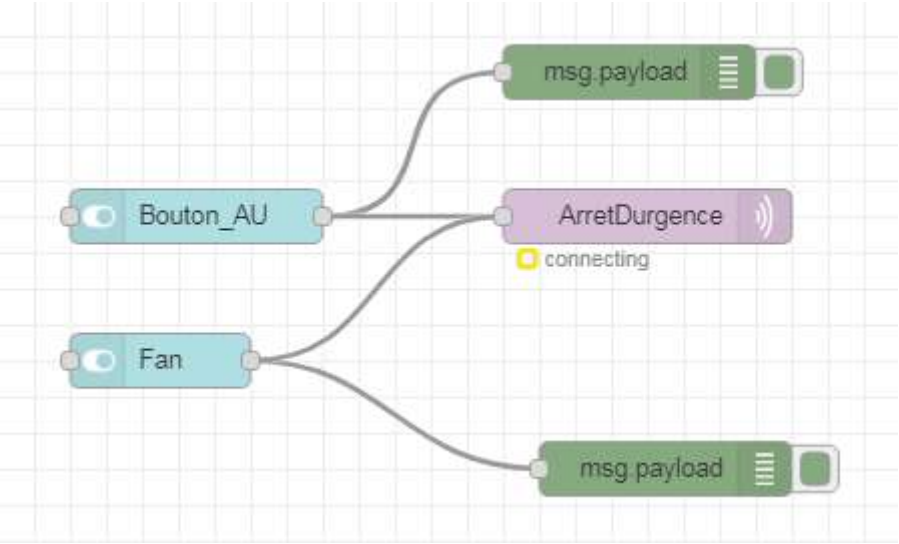
IV.7 Etude software :

IV.7.1 Programmation Node-red :

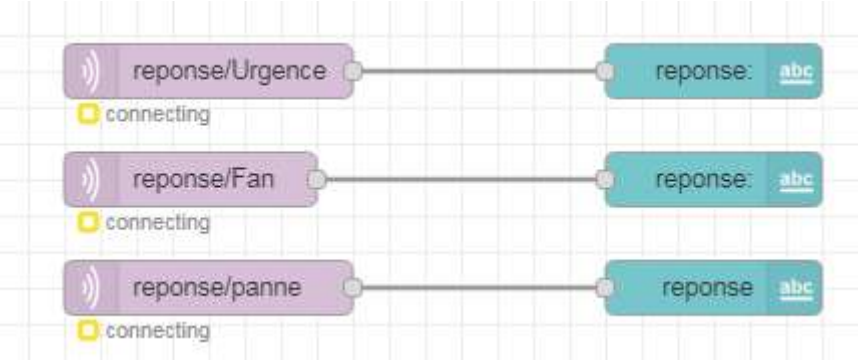
Dans cette partie nous nous sommes inscrits (subscribe) dans le topic (esp8266/Alerte) pour voir l'état de la maquette et la montrer en dushbeard Ici on se connecte au sujet ‘topic’ (esp8266/temperature) pour recevoir la température et on choisit deux façons de lire la température (jauge et chart)



Nous avons connecté le bouton d'arrêt d'urgence à TOPIC (ArretDurgence) ici MQTT PUBLISH Son travail est qu'en cas de panne du système, nous appuyons sur le bouton pour arrêter le chauffage et maintenir le ventilateur en marche



Ici nous nous sommes inscrits à TOPIC (reponse/Urgence) pour lire l'état d'Urgence, (reponse/Fan) pour lire l'état de Fan, (reponse/panne) pour lire l'état de Panne, en les connectant à un nœud nommé Texte



IV.7.2 Programmation Arduino :

Au début de la programmation il faut déclarer une partie de la librairie qui est nécessaire lors de la programmation du module ESP8266WiFi et PubSubClient, puis on crée les instances (objets utilisés) et on déclare les variables et les constantes, Choisissez le nom et le mot de passe Wi -Fi disponibles et choix de l'adresse du broker utilisé, puis consistent à initialiser les variables et constantes ainsi qu'à configurer les différents paramètres et pins esp8266 utilisés et wifi, crée une Fonction de connexion au réseau WIFI, créer une Fonction de réception de données du broker, Fonction de reconnexion au broker , puis on écrit Fonction principale du programme, dedans on écrit fonction reconnect de client , puis on crée la fonction de Publishes new temperature andhumidito toutes les 3 secondes , puis on crée une fonction Convertit la valeur flottante hic en une chaîne et l'enregistre dans le tableau température Tempe…… Voir d'Annexe

```
now = millis();
// Publishes new temperature and humidity every 3 seconds
if (now - lastMeasure > 3000) {
      lastMeasure = now;
      int analogValue = analogRead(lm35_pin);
      float millivolts = (analogValue/1024.0) * 3300;  //3300 is the voltage provided by NodeMCU
      float temp = millivolts/10;

      //Convert the float value hic to a string and save it into temperatureTemp array
      static char temperatureTemp[7];
      dtostrf(temp, 6, 2, temperatureTemp);

      // Publishes Temperature value
      client.publish("esp8266/temperature", temperatureTemp);
```

Pour s'assurer que la valeur de température est obtenue toutes les trois secondes, nous avons créé un programme pour vérifier la température toutes les trois secondes

Lors de la lecture de la valeur obtenue à partir de la sortie du capteur LM35, il s'agit d'une tension, ce qui signifie qu'il ne s'agit pas d'une valeur de température directe. Nous devons convertir cette tension en une valeur de température via une certaine équation

La tension connectée au capteur LM35 est de 3,3 V du NodeMCU. Nous convertissons cette valeur en mv et le microcontrôleur divise sa tension de 0 à 1023 divisions (parties), soit 1024 parties,

L'équation est la suivante

$$temperature = ((Analogvalue * 3300 * 100)/1024)$$

```
    //..............sending Alarm signal to node-red dashboard (MQTT Client).................
    if(EtatUrgence==false || etat_resistance==false ){
        if( temp>=70 ){  //temp>=70
digitalWrite(relay_heater_resistor, HIGH);
digitalWrite(relay_fan,LOW);
            Serial.println("Alerte!.. --> temperature elevee");
            client.publish("esp8266/Alerte", "Alerte Temperature elevee!..");

          }
        if( temp>=75 ){  //temp>=70
            Serial.println("Alerte!.. --> temperature elevee");
            client.publish("esp8266/Alerte", "Alerte Four en etat critique!..");

          }
        else if (temp<=45){  //temp<=45
          digitalWrite(relay_heater_resistor, LOW);
    digitalWrite(relay_fan,HIGH); //..............##############
          Serial.println("ventillateur en arret");
          client.publish("esp8266/Alerte", "Temperature faible");
          }
```

Si la température :

- ✓ Temp>=70 Le chauffage est éteint et le ventilateur en marche
- ✓ Temp>=75 Si la température est élevée, une alerte (four en etat critique) est envoyée lorsque le message nous apparaît.On appuie sur le bouton d'arretdergence jusqu'à ce qu'on arrête le radiateur et garde le ventilateur en marche ou qu'on arrête le système définitivement et qu'on le redémarre à nouveau
- ✓ Temp<=45 La température est basse, le chauffage est démarré et le ventilateur est éteint s'il fonctionne

```
    else if (temp>=65){
      digitalWrite(relay_fan,LOW);
      Serial.println("Ventillateur en marche");
      client.publish("esp8266/Alerte", "Ventillation en marche");
      }
      else if (temp<55){  //temp<=45
     // digitalWrite(relay_heater_resistor, LOW);
digitalWrite(relay_fan,HIGH); //..............############
      Serial.println("ventillateur en arret");
      client.publish("esp8266/Alerte", "ventillateur en arret");
      }



    else if (temp>55 && temp < 65){  //temp>50 && temp < 65
      Serial.println(" temperature normale");
      client.publish("esp8266/Alerte", "Temperature normale 55°C-65°C");
      }
}
```

- ✓ Temp>=65 le ventillateur est marche
- ✓ 55> Temp <65 le température est normale

IV.8 Quelques photos de la réalisation :

IV.8.1 Câblage et montage :



Fig IV.5 : Photo de la réalisation

IV.8.2 Bloc de Client MQTT1 :

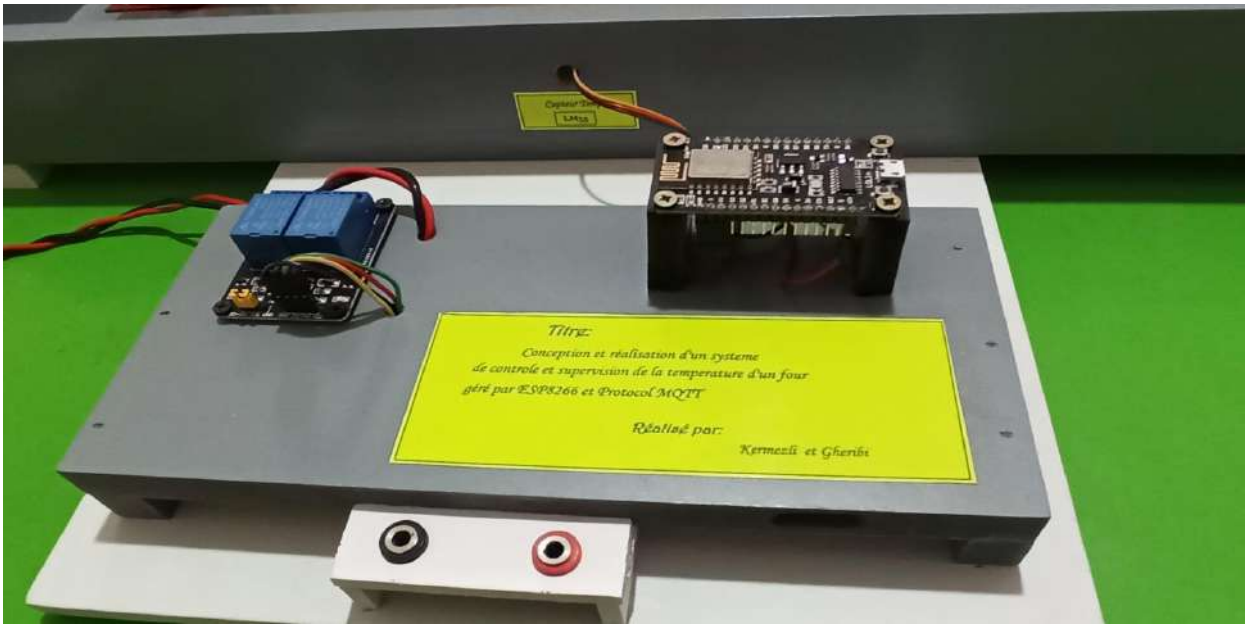


Fig IV.6 : Bloc de Client MQTT1

IV.8.3 Bloc de commande :

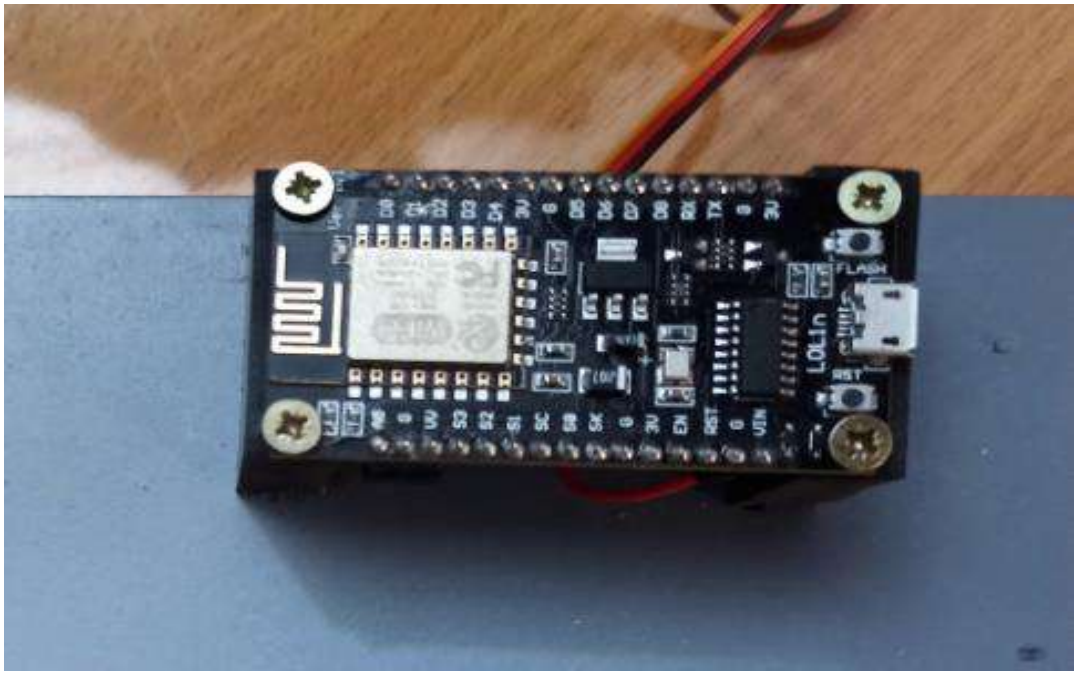


Fig IV.7 : Bloc Commande

IV.8.4 Bloc Heater:

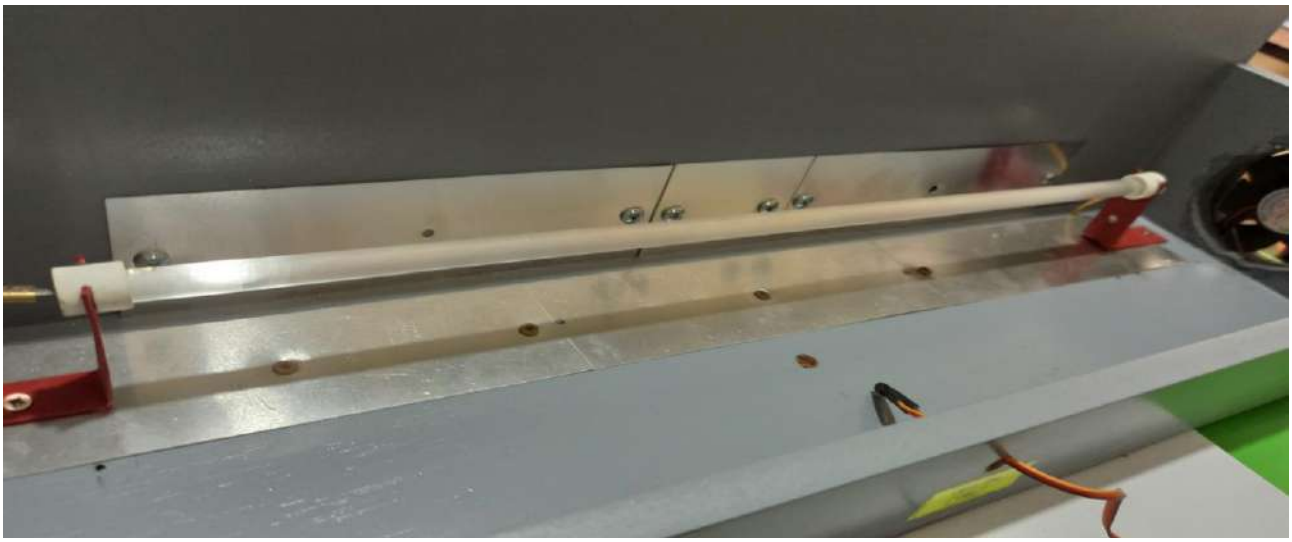


Fig IV.8 : Bloc Heater

IV.8.5 Bloc Fan:



Fig IV.9 : Bloc Fan

IV.8.6 Bloc de client MQTT2 :

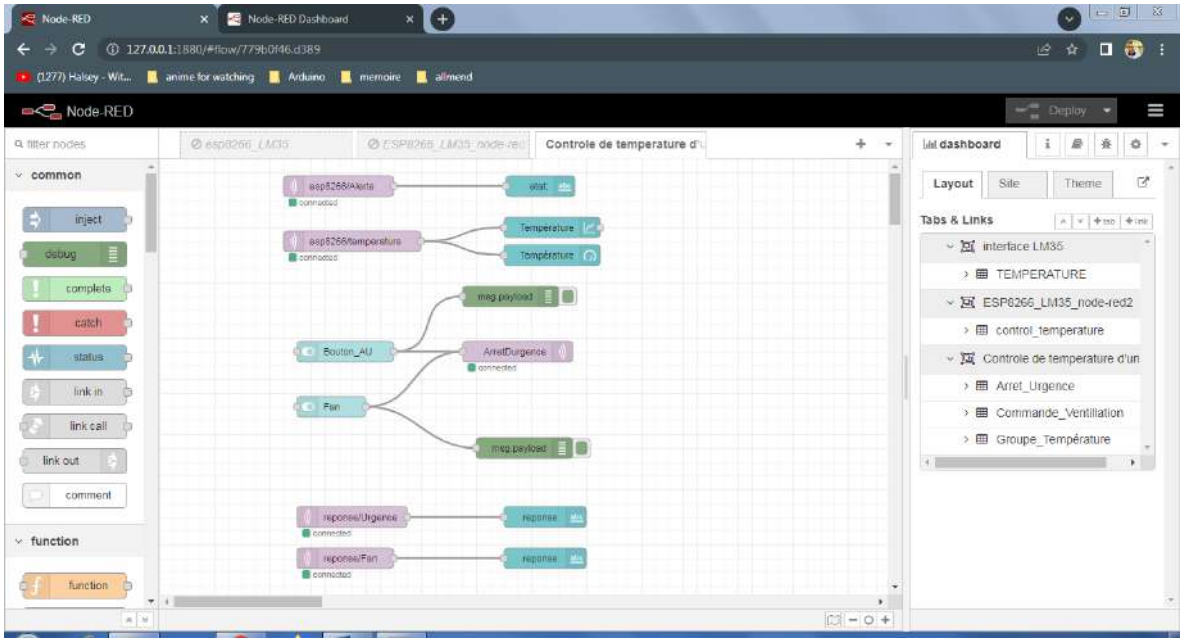


Fig IV.10 : Bloc de Client MQTT2 (Node-red)

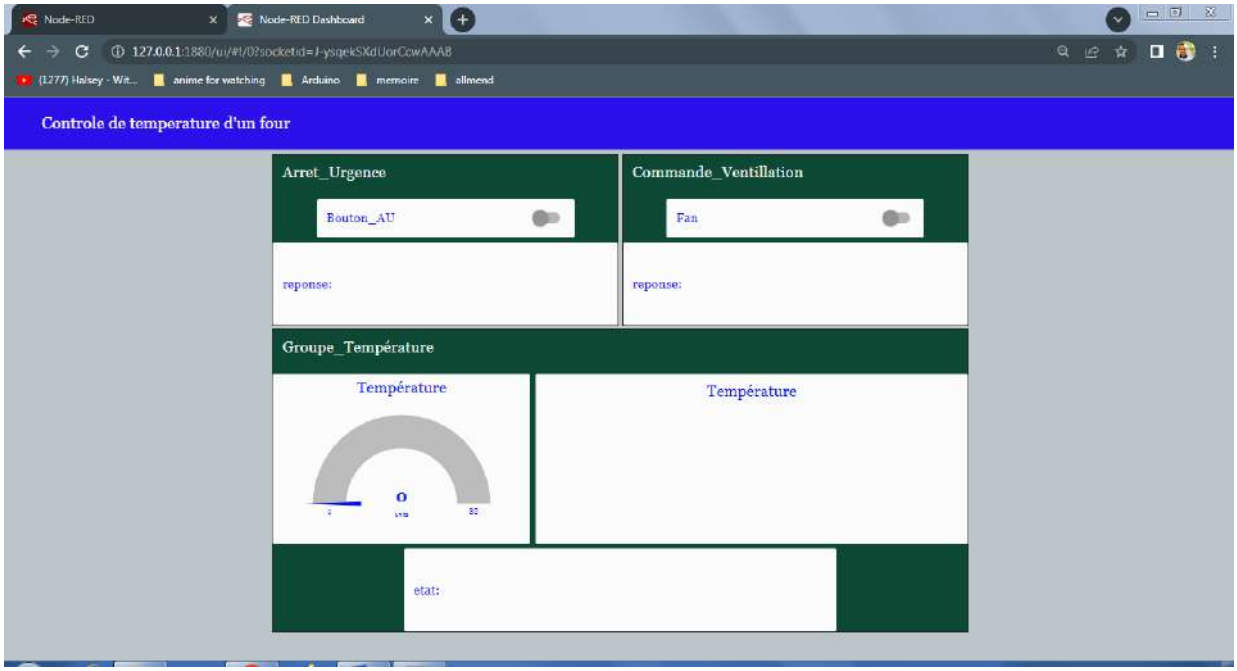


Fig IV11 : Bloc de Client MQTT2 (Node-red Dashboard)

IV.8.7 Bloc d'alimentation :

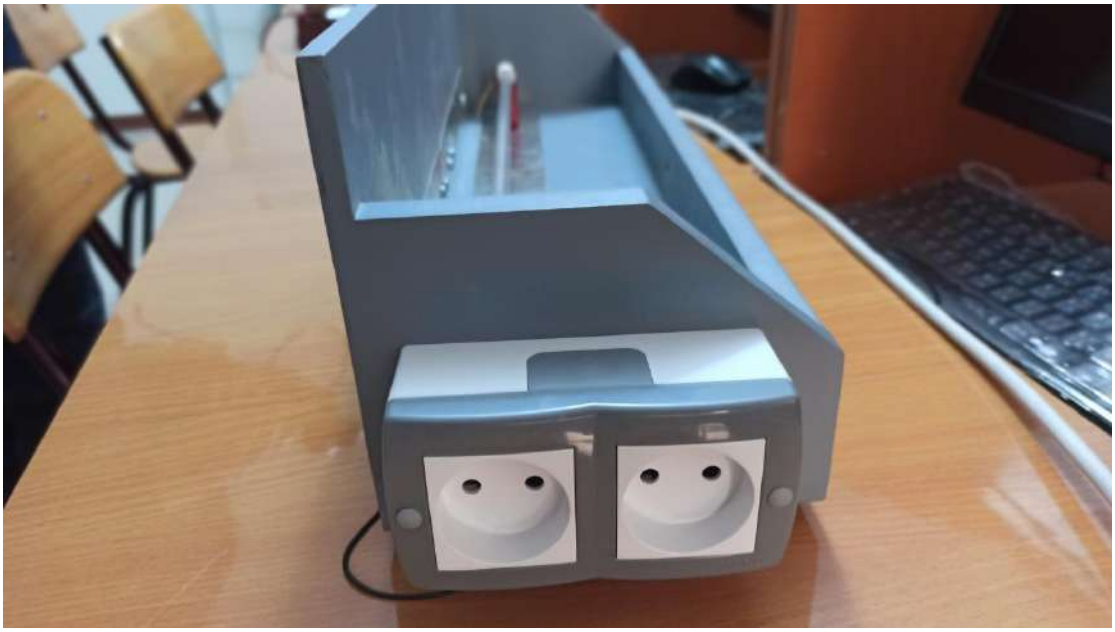


Fig IV.12 : Bloc d'Alimentation

Conclusion :

Ce dernier chapitre se conclut par une réalisation pratique en exploitant une méthodologie de régulation de la température

Dans ce chapitre, nous avons également programmé systématiquement le contrôle de la température.

Nous avons réalisé le système de contrôle de la température en utilisant le protocole MQTT, car le système détecte la température et l'envoie au programme Node-red, qui détecte également l'état et la sécurité du système.

Une fois terminé, plusieurs tests ont été exécutés. Ces tests nous permettent de valider le bon fonctionnement de notre système.

Conclusion générale :

Ce travail nous a permis d'effectuer un état de l'art dans différents domaines de recherche, à savoir l'Internet des Objets, et l'implémentation du protocole MQTT. De plus, la mise en œuvre d'un système IoT pour le contrôle de la température à distance nous a permis d'apprendre et d'exploiter diverses plates-formes, outils et langages de développement, entre autres plates-formes Arduino IDE.

La particularité de ce projet est la réalisation physique du système IoT qui nous offre la première expérience. De nombreux concepts sont obtenus, qui concernent non seulement la partie logicielle mais également la partie matérielle à partir de laquelle le système IoT est construit pour gérer les appareils.

La solution proposée dans ce manuscrit se compose de trois parties : matériel, logiciel et protocoles de communication.

• En ce qui concerne la partie physique, on perçoit un objet connecté placé dans l'environnement cible pour la capture.

Pour la partie logicielle, nous avons créé un logiciel qui nous permet de contrôler notre système à distance via NODE-RED

• Pour une partie des protocoles de communication, nous avons utilisé les protocoles suivants:

- ✓ Wifi : Pour la communication entre l'objet connecté et le support.
- ✓ MQTT : Pour la communication entre le serveur et les objets connectés.

Liste des références bibliographiques :

[1] Mehdi Nemri, l'internet des objets, France STRATEGIE,

https://www.strategie.gouv.fr/publications/demain-linternet-objets#, consulté le 11/05/2020

[2] Mekriou Ryma et Mazari Walid, Introduction à l'internet de l'objet et réalisation D'un système domotique, Mémoire de master en télécommunication, Université A/Mira de Bejaïa,2016.

[3] L'Internet des Objets (IoT), qu'est-ce que c'est ? : Synox. https://www.synox.io/internet-des-objets-iot-qu-est-ce-que-cest, consulté le 11/05/2020.

[4] Internet des Objets, Comment l'évolution actuelle d'Internet transforme-t-elle le monde ? , Dave Evans, Cisco Internet Business Solutions Group (IBSG), Avril 2011.

[5] Dr. Ovidiu et all, Internet of Things–From Research and Innovation to Market Deployment, EU, Belgium , river publishers' series in Communications, 2014.

[6] Rafael Marques, Système de produits et services basés sur l'internet des objets, Mémoire de maîtrise des sciences appliquées, école polytechnique de Montréal, juillet 2018.

[7] Zeinab Kamal Aldein Mohammed, Elmustafa Sayed Ali Ahmed, Internet of Things Applications, Challenges and Related Future Technologies. WSN 67(2) 126-148, 2017.

[8] Les bénéfices de l'IoT, site Intesens https://www.intesens.com/les-benefices-de-liot/, consulté le 20/05/2020.

[9] LeMaGiT, http : //www.lemagit.fr/conseil/Internet-des-Objets-bien-compr endre-MQTT, page consultée le 07/06/2018.

[10] Tutorial MQTT, Introduction to MQTT, May 15th 2013

[11] Edited by Andrew Banks and Rahul Gupta, MQTT Version 3.1.1, OASIS , 29 October 2014.

[12] mqtt-topics-best-practices, URL: http://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best- practices, consulté le 28 /06/2020.

[13] Benaoune Siham, Evaluation de performance de protocoles dans l'internet des objets à base de méthodes formelles, Mémoire de fin d'études, Université Mohamed Khider

– Biskra, 06/05/2017.

[14] GroupeSII Le protocole MQTT dans l'IoT,2017

https://blog.groupe-sii.com/le-protocole-mqtt-dans-liot/, consulté le 10 /07/2020.

[15] Eric J. Bruno,MQTT Programming In Depth, March 2016,

URL : http://programmingwithreason.com/article-mqtt-in-depth.html, consulté le 10/07/2020.

[16] Débuter avec le Broker MQTT Mosquitto (Raspberry Pi, Windows, macOS, Linux), DOMITIQUE ET OBJETS CONNECTE, https://projetsdiy.fr/mosquitto-broker-mqtt-raspberry-pi/, consulté le 16/08/2020.

[17] Travaux de la 4e promotion (2013-2014) du Cycle « Sécurité des usages numériques », p 28.

[18] http://projet.eu.org/pedago/sin/tutos/arduino.pdf

[19] Djafri Menad et Chelouche Djalal, Etude et Réalisation d'une Carte Arduino, mémoire de master, Université A. Mira de Bejaia ,2016.

[20] Simon Landrault (Eskimon) et Hippolyte Weisslinger (olyte), Arduino : 1er Pas en informatique embarquée, Livre, Édition du 01 juin 2014.

[21] http://silanus.fr/bts/formationIOT/node-red/nodered.pdf

[22] SAHRAOUI Seif eddine, Etude et réalisation d'une canne intelligente pour les non-voyants, Mémoire de fin d'études, Université Larbi Ben M'hidi - Oum El Bouaghi, juillet 2019.

[23] https://sti2d.ecolelamache.org/le_relais_lectromcanique.html

[24] 123RF. Stock Photo - Set of color 3 mm LED diodes isolated on white background. [en ligne]. Disponible sur : < https://www.123rf.com/photo_19706682_set-of-color-3-mm-led-diodes-isolated-on-white-background.html?undefined >. (Consulté le 19/02/2019)

[25] astuces pratiques. La résistance : Description et définition. [en ligne]. Disponible sur : https://www.astuces-pratiques.fr/electronique/la-resistance-description-et-definition (Consulté le 21/02/2019)

[26]https://www.ummto.dz/dspace/bitstream/handle/ummto/6391/AmmarkhodjaNassim.pdf?sequence=1

Annex:

```
#include <ESP8266WiFi.h>

#include <PubSubClient.h>


//

const char* ssid = "Redmi2022";

const char* password = "Gheribi2022Gheribi";


// choix de l'adresse du broker utilisé

const char* mqtt_server = "broker.hivemq.com";  //   broker.hivemq.com   test.mosquitto.org broker.emqx.io


// Initializes the espClient. You should change the espClient name if you have multiple ESPs running in your system

WiFiClient espClient;

PubSubClient client(espClient);


const int relay_heater_resistor = 5;

const int relay_fan = 4 ;


int lm35_pin=A0;

bool EtatUrgence=false,etat_resistance=false;

bool commandeFan=false;

long now = millis();
```

```
long lastMeasure = 0;


void setup() {
  Serial.begin(115200);
  pinMode(relay_heater_resistor, OUTPUT);
  digitalWrite(relay_heater_resistor,HIGH);
  pinMode(relay_fan, OUTPUT);
  digitalWrite(relay_fan,HIGH);


  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
//.....................Fonction de connexion au reseau WIFI
void setup_wifi() {
  delay(1000);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
```

```
  }
  Serial.println("");
  Serial.print("WiFi connected - ESP IP address: ");
  Serial.println(WiFi.localIP());
}
//.....................Fonction de réception de données du broker.....
void callback(String topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp = messageTemp +(char)message[i];
  }
  Serial.println();

  // If a message is received on  topic "ArretDurgence", you check if the message is either on_fanor off_fan.
  //Turns the relay  according to the message

  if(String(topic)=="ArretDurgence"){
      if(messageTemp == "on_fan" ){
```

```
      digitalWrite(relay_fan, LOW);     // if message arrived is "On"  we disable the relay
      Serial.print("Fan_On");
      //delay(300);   // delais pour l'envoie de acknownledge
      client.publish("reponse/Fan", "Ventillation en marche"); // publish a message chauffage-on- to Topic reponse/Fan
    }
    else if(messageTemp == "off_fan" ){   // if message arrived is "Off_fan"  we desactivet a relay
      digitalWrite(relay_fan, HIGH);
      Serial.print("Fan_Off");
      //delay(300);  // delais pour l'envoie de acknownledge
      client.publish("reponse/Fan", "Ventillation en arret"); // publish a message chauffage-off- to Topic reponse/Fan
    }
    else if(messageTemp == "Urgence_on"){
      EtatUrgence=true;
    }
    else if(messageTemp == "Urgence_off"){
      EtatUrgence=false;
      digitalWrite(relay_fan, HIGH);
      Serial.println("Systeme en etat normal");
      client.publish("reponse/Urgence", "fonctionnement normal"); // publish a message chauffage-off- into Topic esp8266/Lampe
    }
  }
```

```
  else if(String(topic)=="Panne/Heater"){
    if(messageTemp == "on_heater"){
          etat_resistance=true;
          digitalWrite(relay_heater_resistor, LOW);
    }
    else if(messageTemp == "off_heater"){
      etat_resistance=false;
          digitalWrite(relay_heater_resistor, HIGH);
    }
  }
  else if(String(topic)=="Panne/Fan"){
    if(messageTemp == "on_fan"){
          digitalWrite(relay_fan, LOW);
    }
    else if(messageTemp == "off_fan"){
          digitalWrite(relay_fan, HIGH);
    }
  }
  Serial.println();
}

//.....Fonction de reconnexion au broker.....

void reconnect() {
```

```
  //  on boucle jusqu'à ce que la connexion avec le Broker soit établié
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // Subscribe
      client.subscribe("ArretDurgence");
      client.subscribe("Panne/Heater");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // faisant un retard de  5 seconds
      delay(5000);
    }
  }
}
//.....Fonction principale du programme.....
void loop() {

 if (!client.connected()) {
    reconnect();
  }
```

```
  client.loop();
  if(EtatUrgence==true){
      digitalWrite(relay_heater_resistor, HIGH);  //desactiver la resistance chauffante
      digitalWrite(relay_fan, LOW);               //activer le relais du ventillateur
      Serial.println("Systeme en Arret d'urgence");
      //delay(300);   // delais pour l'envoie de acknownledge
      client.publish("reponse/Urgence", "Four en Arret d'urgence");
     // client.publish("esp8266/Alerte", "Alerte Four en etat critique!..");
    }
   if(etat_resistance==true){
      digitalWrite(relay_heater_resistor, LOW);  //desactiver la resistance chauffante
      digitalWrite(relay_fan, HIGH);              //activer le relais du ventillateur
      Serial.println("panne resistance");
      //delay(300);   // delais pour l'envoie de acknownledge
      client.publish("reponse/panne", "resistance en panne");
     // client.publish("esp8266/Alerte", "Alerte Four en etat critique!..");
    }
  now = millis();
  // Publishes new temperature and humidity every 3 seconds
  if (now - lastMeasure > 3000) {
      lastMeasure = now;
      int analogValue = analogRead(lm35_pin);
      float millivolts = (analogValue/1024.0) * 3300;  //3300 is the voltage provided by
NodeMCU
```

```
    float temp = millivolts/10;
    //Convert the float value hic to a string and save it into temperatureTemp array
    static char temperatureTemp[7];
    dtostrf(temp, 6, 2, temperatureTemp);
    // Publishes Temperature value
    client.publish("esp8266/temperature", temperatureTemp);
    //.............sending Alarm signal to node-red dashboard (MQTT Client).....
    if(EtatUrgence==false || etat_resistance==false ){
       if( temp>=70 ){  //temp>=70
digitalWrite(relay_heater_resistor, HIGH);
digitalWrite(relay_fan,LOW);
          Serial.println("Alerte!.. --> temperature elevee");
          client.publish("esp8266/Alerte", "Alerte Temperature elevee!..");

        }
       if( temp>=75 ){  //temp>=70
          Serial.println("Alerte!.. --> temperature elevee");
          client.publish("esp8266/Alerte", "Alerte Four en etat critique!..");

        }
       else if (temp<=45){  //temp<=45
        digitalWrite(relay_heater_resistor, LOW);
    digitalWrite(relay_fan,HIGH); //.............##############
        Serial.println("ventillateur en arret");
```

```
        client.publish("esp8266/Alerte", "Temperature faible");
        }

      else if (temp>=65){
        digitalWrite(relay_fan,LOW);
        Serial.println("Ventillateur en marche");
        client.publish("esp8266/Alerte", "Ventillation en marche");
        }
        else if (temp<55){  //temp<=45
       // digitalWrite(relay_heater_resistor, LOW);
    digitalWrite(relay_fan,HIGH); //.............##############
        Serial.println("ventillateur en arret");
        client.publish("esp8266/Alerte", "ventillateur en arret");
        }
      else if (temp>55 && temp < 65){  //temp>50 && temp < 65
        Serial.println(" temperature normale");
        client.publish("esp8266/Alerte", "Temperature normale 55°C-65°C");
        }
    }
    Serial.print("Temperature: ");
    Serial.print(temp);
    Serial.println(" °C ");
 }
}
```