

---

# PROJET DE FIN D'ANNEE

*Présenté à*

**L'Ecole Nationale d'Electronique et des  
Télécommunications de Sfax**

**Génie des Systèmes Electroniques de Communication**

*Par*

**Syrine Dabbeche  
Khouloud Matri**

---

**Titre**

**Système d'irrigation intelligent**

---

*Soutenu le 29 mai 2023, devant la commission*

**Mme** Achwek Ben Saied

*Examineur*

**Mme** Houda Daoud Damak

*Encadrant*

## Remerciement

En préambule à ce rapport, nous souhaitons adresser tous nos remerciements aux personnes qui nous ont apporté leur aide et qui ont ainsi contribué à l'élaboration de ce rapport.

Nous tenons à exprimer également notre profonde reconnaissance à madame Houda Daoud qui nous a encadré durant ce projet de fin d'année. Pour l'aide et les conseils concernant les missions évoquées dans ce projet, qu'elle nous a apporté lors des différents suivis.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.

Enfin, Nous tenons également à remercier toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

# Table des matières

Introduction générale

## Chapitre 1: Etude de l'art sur le système d'irrigation intelligent

I.Introduction .....	10
II.Internet Des Objets. ....	10
II.1Définition.....	10
II.2Les avantages. ....	10
II.3 Les trois couches d'un modèle IOT.....	10
II.3.1 La couche de perception .....	11
II.3.2La couche de réseau .....	11
II.3.3 La couche d'application .....	11
III.Les systèmes d'irrigation .....	20
III.1. Système d'irrigation classique .....	11
III.2. Système d'irrigation moderne .....	12
IV.Context et Objectif du projet.....	12
IV.1Description du système d'irrigation intelligent actuel.....	12
IV..2 Identification des défis et des limites .....	13
IV..3 Solution proposée.....	13
V.Conclusion.....	13
<b>Chapitre 2 : Etude et conception du système .....</b>	<b>14</b>
I.Introduction : .....	15
II. Architecture du système .....	15
III.Environnement matériel .....	15
III.1.Description de la carte esp32 .....	15

III.2.Capteur d'humidité et température DHT11 .....	16
III.3.Capteur d'humidité du sol(Soil Moisture Sensor).....	16
III.4. Relais .....	17
III.5.Pompe à eau .....	17
IV.Environnement Logiciel.....	18
IV.1. L'environnement de programmation(Arduino IDE).....	18
IV.2.Fritzing interface .....	18
IV.3. Le protocole MQTT .....	18
IV.4.HiveMQ Cloud.....	19
IV.5.Node Red.....	19
V.Conclusion :.....	20
 <b>Chapitre 3 : Développement et test de système d'irrigation</b> .....	<b>21</b>
I Introduction .....	22
II.Organigramme .....	22
III.Etape de conception .....	23
III.11 <sup>ère</sup> partie : Conception de partie electronique .....	23
III.1.1 Description .....	23
III.1.2 Configuration de la carte esp32 .....	23
III.1.3 cablage du circuit .....	24
III.1.3.1 Test du capteur d'humidité et température DHT11.....	24
III.1.3.2 Test du capteur d'humidité du sol .....	25

III.1.3.3 Test de la pompe.....	25
III.1.3.4 Assemblage du circuit .....	26
IV. 2 <sup>ème</sup> Partie : Communication et transmission des données .....	26
IV.1 Développement de Dashbord avec node red.....	26
IV.1.1 Installation.....	26
IV.1.2 Création du tableau de bord .....	27
IV.2 Communication du dashboard avec HiveMQ Broker (partie configuration).....	28
IV.2.1 Configuration de HiveMQ .....	28
IV.2.2 L'envoi des données par le broker MQTT : Méthode publish .....	29
IV.2.3 Test de la méthode publish.....	29
IV.3 La réception des données par le MQTT broker : méthode subscribe .....	31
IV.3.1 Test de la méthode subscribe .....	31
IV.4 .Test final du système d'irrigation .....	33
V.Evaluation des performances du système d'irrigation intelligent. ....	34
VI.Conclusion. ....	34
Conclusion générale .....	35
Références Bibliographiques.....	36

## Liste des figures

Figure 1 : Les trois couches d'internet des objets .....	11
Figure2 : Les méthodes classiques d'irrigation.....	12
Figure3 : Architecture du système .....	15
Figure 4: Carte esp32 .....	16
Figure 5: Capteur DHT11 .....	16
Figure 6:Capteur d'humidité du sol .....	17
Figure 7: Relais .....	20
Figure 8: Pompe à eau.....	20
Figure 9: Arduino IDE .....	20
Figure 10: Fritzing interface .....	20
Figure 11: MQTT-Broker .....	20
Figure 12 :HiveMQ.....	20
Figure 13 : Node-RED .....	20
Figure 14: Organigramme du système .....	22
Figure 15 : Test de fonctionnement de la carte .....	23
Figure 16 : Câblage de DHT11 .....	24
Figure 17 : Test de fonctionnement du capteur DHT11 .....	24
Figure 18 : Câblage du capteur d'humidité du sol .....	25
Figure 19 : Test de fonctionnement de soil moisture.....	25
Figure 20 : Test de fonctionnement de la pompe.....	25
Figure 21: Assemblage des composants .....	26
Figure 22: Lancement de Node red.....	27
Figure 23: Développement avec Node red.....	28
Figure 24: Notification avec Node red.....	28

Figure 25: Configuration de l'output pompe .....	29
Figure 26 : Connexion aux topics .....	30
Figure 27 : Connexion réussite aux topics .....	30
Figure 28 : Affichage des messages dans HiveMQ .....	30
Figure 29: Affichage des messages reçus par les capteurs sur la partie Debug .....	31
Figure 30: L'envoi manuel sur le topic khouloudsyrine/hum.....	32
Figure 31 : Affichage des données sur le dashboard .....	32
Figure 32 : Affichage des messages du système dans HiveMQ .....	33
Figure 33 : Affichage des données du système sur le dashboard.....	33

## Liste des abréviations

Iot : Internet of things

Wi-Fi: Wireless Fidelity

NFC: Near Field Communication

RAM: Random Access Memory

MQTT : Message QueingTelemetry Transport

IDE: environnement de développement intégré

QoS : Quality of Service

SSL: Secure Socket Layer

TLS: Long Term Support

LDAP : Lightweight Directory Access Protocol

API :Application Programming Interface

URL :Uniform Resource Locator

LED : Light Emitting Diode

VCC: Voltage Common Collector

GND:ground



## Introduction générale

Un système d'irrigation est un ensemble de dispositifs et de techniques utilisés pour fournir de l'eau aux plantes de manière contrôlée et efficace. Il vise à maintenir un niveau d'humidité adéquat dans le sol, en fournissant la quantité d'eau nécessaire pour répondre aux besoins des cultures. Cependant, les méthodes traditionnelles d'irrigation présentent des problèmes qui peuvent compromettre son efficacité et entraîner un gaspillage des ressources en eau. Parmi ces problèmes, on peut citer le manque de précision dans la quantité d'eau fournie, une réponse inadaptée aux besoins réels des plantes et une utilisation inefficace des ressources disponibles. Ces facteurs ont un impact négatif sur les rendements agricoles et peuvent entraîner des conséquences économiques et environnementales.

L'utilisation des technologies de L'IoT offre de nouvelles opportunités pour résoudre ces problèmes et améliorer la gestion de l'irrigation.

Et c'est dans ce contexte, que s'inscrit notre projet de réaliser « Un système d'irrigation intelligent » qui vise principalement à collecter, analyser et le partage de données en temps réel, ce qui permet une surveillance précise des conditions environnementales et des besoins des plantes, en intégrant des capteurs de sol et des systèmes de communication.

Notre projet consiste à développer un système de contrôle du système d'irrigation qui assure à l'utilisateur la possibilité de suivre instantanément les différentes variables du système d'irrigation, via les capteurs implémentés, pour pouvoir ainsi anticiper les problèmes potentiels. Les systèmes d'irrigation intelligents basés sur l'IoT peuvent ajuster automatiquement la quantité d'eau fournie en fonction des besoins spécifiques des cultures et des conditions environnementales changeantes.

Dans ce cadre, notre travail est organisé comme suit :

Le premier chapitre présente plus en détail les avantages, les défis et les solutions proposées par les systèmes d'irrigation intelligents basés sur l'IoT. Nous présenterons une étude de cas et des résultats expérimentaux pour illustrer les bénéfices potentiels de cette approche.

Le deuxième chapitre est dédié à l'étude du système proposé. Ensuite on présente les Technologies et les langages de programmation utilisé.

Le troisième chapitre porte sur la réalisation expérimentale et la validation du système.

# Chapitre1 : Etat de l'art sur les systèmes d'irrigation

## I. Introduction :

L'irrigation joue un rôle crucial dans l'agriculture, permettant d'optimiser la croissance des cultures et d'améliorer leur rendement. Ce chapitre présente un état de l'art détaillé des systèmes d'irrigation, mettant en évidence les technologies, les méthodes pratiques qui ont révolutionné le domaine de l'irrigation.

## II. Internet Des Objets :

### II.1 Définition :

L'IoT est un concept qui décrit la connectivité des objets physiques, tels que les appareils électroménagers, les véhicules, les capteurs, les caméras de surveillance, les systèmes de sécurité, et autres, à Internet. Les objets connectés sont capables de collecter et d'échanger des données entre eux et avec les systèmes informatiques, en utilisant des technologies sans fil telles que le Wi-Fi, le Bluetooth, le NFC (Near Field Communication) ou des réseaux de capteurs sans fil. L'IoT permet de créer des environnements intelligents, dans lesquels les objets peuvent être contrôlés et automatisés à distance, et où les données collectées peuvent être analysées pour prendre des décisions éclairées [1]

### II.2 Les avantages :

L'IoT est un concept qui offre de nombreux avantages pour les entreprises et la société en général. Grâce à la connectivité des objets physiques, l'IoT permet d'automatiser les tâches, de collecter des données en temps réel et d'optimiser les processus, ce qui peut améliorer l'efficacité et réduire les coûts. En utilisant des capteurs et des analyses de données en temps réel, l'IoT permet également de prendre des décisions plus rapides et plus précises, basées sur des données fiables.[2]

L'automatisation des tâches et des processus réduit les risques d'erreurs humaines, ce qui peut améliorer la sécurité et la qualité des produits et services.

De plus, l'IoT peut aider à surveiller les performances et à optimiser les processus pour améliorer la productivité des employés, ainsi qu'à offrir une meilleure expérience client en

permettant des interactions plus fluides et plus personnalisées, ainsi que des produits et services plus adaptés aux besoins des clients. Enfin, l'IoT peut contribuer à la protection de l'environnement en surveillant et en réduisant la consommation d'énergie et de ressources. En résumé, l'IoT offre de nombreux avantages pour les entreprises et la société en général, et sa mise en œuvre peut contribuer à améliorer l'efficacité, la qualité, la sécurité et la durabilité des processus et des produits.

### II.3 Les trois couches d'internet des objets :

Le concept de l'Internet des objets a été l'objet des recherches depuis plus d'une décennie, mais même si, encore de nombreux aspects ne sont pas clairement définis. Par exemple, aujourd'hui il n'y a pas une architecture standardisée et spécifique pour l'IoT. Malgré ce manque de compatibilité, il y a une architecture à trois couches (figure 1) bien connu qui est généralement accepté, ces couches sont : la couche de perception, la couche réseau et la couche d'application.[3]

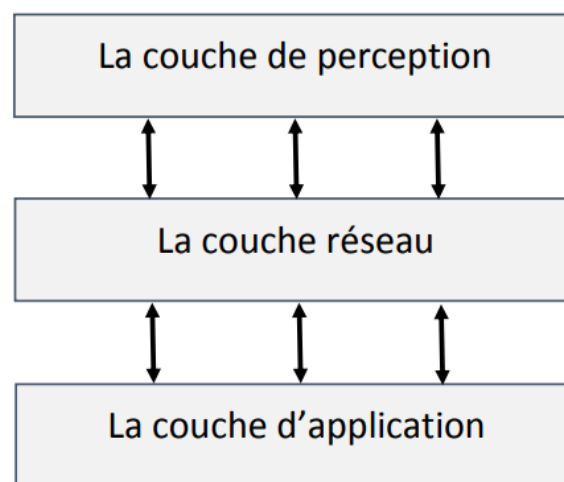


Figure1 : Les trois couches d'internet des objets

#### II.3.1 La couche de perception :

Cette couche est composée de capteurs et d'actuateurs qui permettent de collecter des données à partir de l'environnement physique. Les capteurs sont chargés de mesurer les données telles que la température, l'humidité, la pression, la vitesse, la position, etc. tandis que les actuateurs sont responsables d'effectuer des actions sur l'environnement physique telles que le contrôle de l'éclairage, le verrouillage des portes, l'ouverture des fenêtres, etc.

### II.3.2 La couche de réseau :

Cette couche est responsable de la communication des données collectées entre les objets physiques et les systèmes d'information. Les protocoles de communication couramment utilisés dans l'IoT sont le Wi-Fi, le Bluetooth, le Zigbee, le LoRaWAN, le Sigfox, etc.

### II.3.3 La couche d'application :

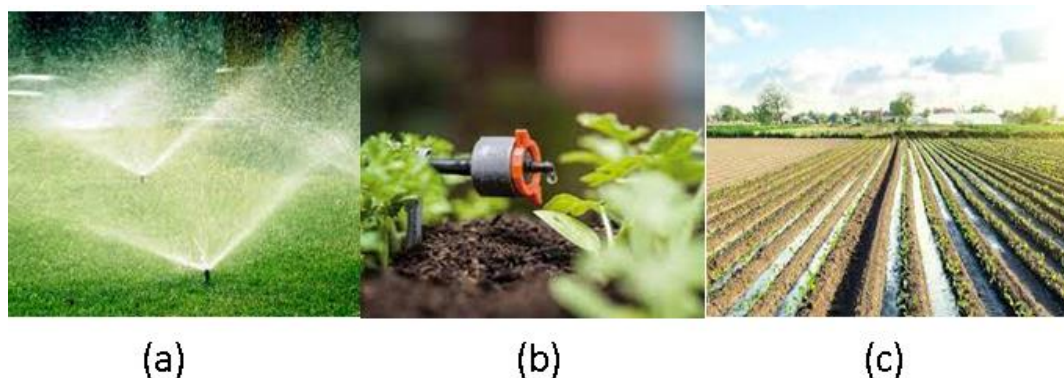
Cette couche est chargée de l'analyse des données collectées par les capteurs afin d'en extraire des informations utiles. Cette couche peut être mise en œuvre à différents niveaux, allant de l'analyse locale à l'analyse cloud en passant par l'analyse de bord. Elle utilise également des techniques d'apprentissage automatiques pour améliorer la qualité et la précision de l'analyse des données. Les résultats de l'analyse peuvent être utilisés pour prendre des décisions en temps réel ou pour optimiser les processus à long terme.

## III. Les systèmes d'irrigation :

### III.1 Système d'irrigation classique :

Le système d'irrigation classique repose sur l'utilisation d'arroseurs pour fournir de l'eau aux cultures. Les arroseurs sont généralement disposés de manière fixe ou mobile dans les champs ou les jardins. Voici quelques méthodes d'irrigation classique couramment utilisées [4] :

- Arrosage par aspersion : Les arroseurs peuvent être fixes, semi-mobiles ou mobiles. Ils sont généralement placés en hauteur pour couvrir une zone étendue. L'eau est pulvérisée dans l'air et retombe sous forme de fines gouttelettes. Cette méthode permet une distribution relativement uniforme de l'eau.
- Arrosage par goutte-à-goutte : L'eau est distribuée lentement et de manière contrôlée, goutte à goutte, permettant une utilisation précise de l'eau. Cette méthode est efficace pour économiser l'eau car elle réduit les pertes dues à l'évaporation et à l'arrosage de zones non cultivées.
- Irrigation par submersion : Cette méthode est principalement utilisée dans les cultures en riziculture. Elle implique l'inondation des champs avec une couche d'eau qui recouvre les plants de riz. Cette méthode permet une irrigation uniforme et fournit également des nutriments supplémentaires aux plants.



**Figure2 : Les méthodes classique d'irrigation (a) : Arrosage par aspersion  
(b) Arrosage par goutte-à-goutte(c) Irrigation par submersion**

La figure 2 présente les différentes méthodes pour l'irrigation classique.

### **III.2 Système d'irrigation moderne :**

Les systèmes d'irrigation modernes améliorent la gestion de l'irrigation en utilisant des technologies avancées telles que des capteurs, des contrôleurs et des systèmes de gestion des données. Ces systèmes optimisent l'utilisation de l'eau, s'adaptent aux besoins spécifiques des cultures et augmentent les rendements agricoles. Ils peuvent être connectés à des réseaux IoT, ce qui permet une surveillance en temps réel et une gestion à distance. Grâce à cette connectivité, il est possible de contrôler le système d'irrigation depuis un ordinateur, un smartphone ou une tablette, où que l'on se trouve. De plus, la connectivité permet de recevoir des alertes en cas de problèmes et de réaliser des analyses de données. [5]

## **IV.Contexte et Objectif du projet :**

### **IV.1 Description du système d'irrigation intelligent actuel :**

Le système d'irrigation intelligent actuel repose sur l'intégration de technologies avancées telles que l'Internet des objets (IoT), l'analyse des données et l'automatisation pour améliorer l'efficacité et la durabilité de l'irrigation. Les capteurs sont essentiels dans le système d'irrigation intelligent. Ils sont installés dans le sol pour mesurer l'humidité, la température, la conductivité électrique et d'autres paramètres environnementaux. Ces capteurs collectent des données en temps réel, fournissant des informations précises sur l'état du sol et des plantes. Les données collectées sont ensuite transmises à un système centralisé via des réseaux sans fil, permettant une surveillance continue et une analyse approfondie.

Parmi les systèmes d'irrigation intelligents les plus reconnus l'utilisation des cartes Arduino et des plateformes comme Blynk .Ce système est un système automatisé qui permet de contrôler et de gérer l'arrosage des cultures de manière efficace et précise. Il utilise la plateforme Arduino, qui est un microcontrôleur programmable, pour surveiller les conditions environnementales et activer/désactiver les vannes d'irrigation en conséquence.

#### **IV.2 Identification des défis et des limites :**

Malgré les avantages qu'il offre, le système d'irrigation intelligent actuel présente également certaines limites qu'il est important de prendre en compte.

Parmi les limites, Les cartes Arduino de base ne disposent pas de connectivité intégrée, telle que le Wi-Fi ou le Bluetooth. Cela peut rendre difficile la communication en temps réel avec d'autres appareils ou l'accès à des services en ligne. Des modules supplémentaires peuvent être ajoutés pour ajouter des fonctionnalités de connectivité, mais cela peut augmenter la complexité et les coûts du système. Ainsi, les cartes Arduino ont une mémoire limitée pour le stockage du programme et des données.

En outre, La version gratuite de Blynk impose des limites sur le nombre de widgets pour l'utilisation dans un projet et sur le nombre de projets que nous avons créé.

#### **IV.3 Solution proposée :**

Le système d'irrigation basé sur l'ESP32 offre des fonctionnalités plus avancées, une connectivité intégrée et une puissance de calcul supérieure par rapport aux microcontrôleurs Arduino traditionnels. Cette carte dispose d'un processeur plus puissant. Elle est basée sur une architecture à double cœur et possède une fréquence d'horloge plus élevée,

Contrairement aux cartes Arduino de base, l'ESP32 dispose de connectivité Wi-Fi et Bluetooth intégrée. Cela facilite l'intégration des projets IoT et permet une communication sans fil avec d'autres appareils ou services en ligne.

Elle dispose généralement de plus de mémoire (RAM et flash). Cela permet de stocker et de manipuler plus de données et de programmes plus volumineux. [6]

La solution proposée est à un système d'irrigation intelligent basé sur l'esp32 en utilisant HiveMQ et Node-Red.

Cette combinaison est possible de construire des systèmes IoT complets et résoudre une grande variété de problèmes. HiveMQ fournit une infrastructure robuste pour la mise en place

de la communication bidirectionnelle entre les appareils IoT, ce qui permet de résoudre des problèmes tels que la collecte de données, le contrôle à distance et la coordination des appareils. Node-RED permet de connecter et de manipuler facilement des appareils, des services et des API différents en utilisant une interface graphique conviviale. et ESP32 offre une connectivité et une capacité de traitement locales. Ensemble, ces outils peuvent être utilisés pour créer des solutions personnalisées et flexibles pour les problèmes spécifiques auxquels sont confrontés les projets IoT. [7]

## **V.Conclusion :**

On peut concevoir et construire un système d'irrigation intelligent qui répond aux besoins spécifiques des utilisateurs. Grâce à l'utilisation de capteurs, d'algorithmes intelligents et de contrôleurs programmables, il permet une gestion précise de l'eau, favorisant ainsi l'efficacité, la durabilité et la productivité de l'agriculture.

# Chapitre 2 : Etude et Conception du système intelligent

## I. Introduction :

En comprenant les avancées technologiques et les meilleures pratiques, les agriculteurs peuvent prendre des décisions éclairées pour adopter les systèmes d'irrigation les plus adaptés à leurs besoins. Dans ce chapitre, nous présentons les outils matériels et environnements logiciels utilisés pour développer notre système et ainsi que les différentes plateformes d'exécution de ses différentes parties.

## II. Architecture du système :

Le système d'irrigation comprend des capteurs (humidité du sol et DHT11) connectés à l'ESP32, qui joue le rôle de client MQTT qui se connecte au broker et échange des messages avec lui. Le broker MQTT dans notre projet, agit comme un intermédiaire entre les différents clients MQTT connectés, y compris l'ESP32, DHT11, Capteur de sol et pompe.

Les valeurs des capteurs sont utilisées par les actionneurs pour prendre des décisions en fonction du code. Si l'humidité du sol est inférieure à un seuil défini, la pompe est activée automatiquement via un relais. Il est également possible d'activer manuellement la pompe en mode non automatique. Les données des capteurs sont envoyées à un broker MQTT (HiveMQ), et qui sont affichées dans un dashboard (Node Red) via Internet. Le système est commandable à distance grâce au module WIFI ESP32, qui reçoit les commandes, les exécute et envoie les états des capteurs et actionneurs. Un relais contrôle le courant de la pompe. La figure\_3 présente le schéma de principe du projet et les connexions entre l'ESP32 et les autres



composants.

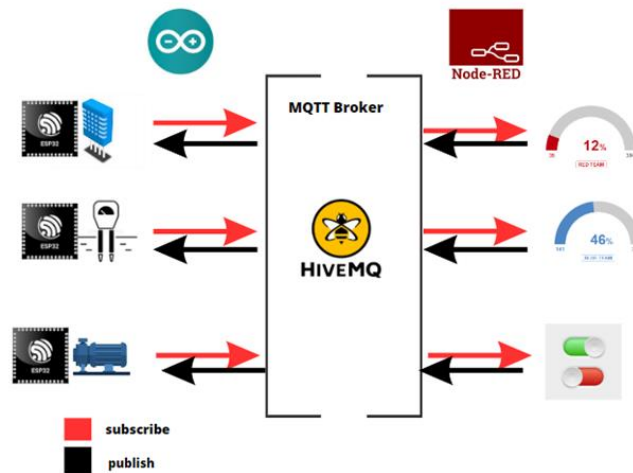


Figure3 : Architecture du système

### III. Environnement matériel :

#### III.1 Description de la carte esp32 :

La carte esp32 qui est représenté par la figure\_4 a des bénéfices dans ces caractéristiques. Elle est de faible coût et à faible consommation d'énergie. Elle offre une grande variété de fonctionnalités, notamment une connectivité Wi-Fi et Bluetooth, une grande quantité de mémoire vive (RAM) expESP32-S0WD dispose de 128 Ko de RAM et une puissance de traitement suffisante pour exécuter des tâches complexes. [8]

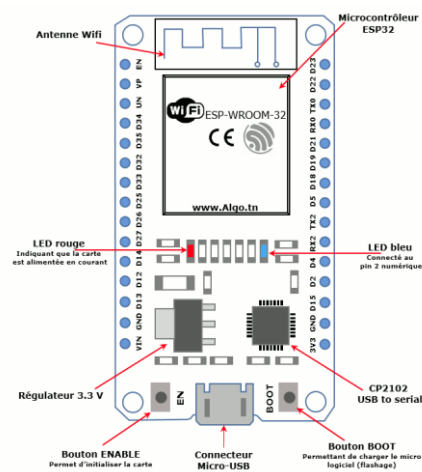
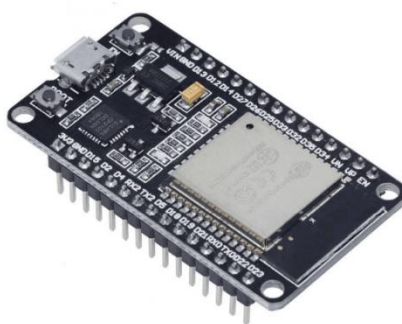


Figure 4: Carte esp32

## IV. Capteur d'humidité et température DHT11 :

Le DHT11 est un capteur d'humidité et de température numérique basé sur un élément capacitif. Il peut mesurer la température de l'air dans une plage de  $-20^{\circ}\text{C}$  à  $50^{\circ}\text{C}$  avec une précision de  $\pm 2^{\circ}\text{C}$ , et l'humidité relative de 20% à 80% avec une précision de  $\pm 5\%$ . Le capteur est facile à utiliser car il dispose de seulement 3 broches comme indique la figure 5 : la broche d'alimentation, la broche de données et la broche de terre.[9]

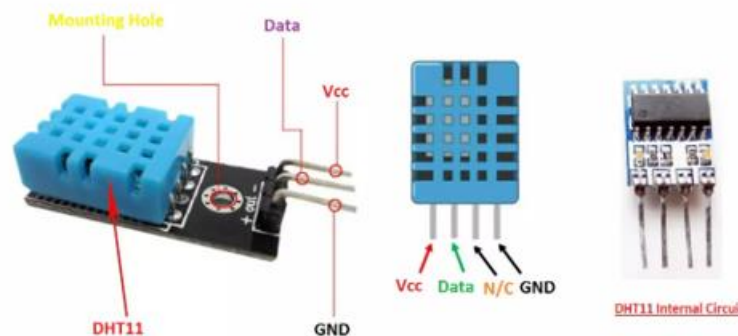


Figure 5: Le capteur DHT11

### IV.1 Capteur d'humidité de sol:

Le capteur d'humidité de sol, également appelé SoilMoistureSensor, qui présenté par la figure\_6 mesure le niveau d'humidité du sol en mesurant la résistance électrique entre deux électrodes insérées dans le sol. Une conductivité électrique plus élevée correspond à un sol plus humide, ce qui se traduit par une résistance électrique plus faible. Le capteur est alimenté en courant continu à une tension de 3,3 ou 5 volts et fournit généralement un signal analogique qui peut être mesuré par un microcontrôleur tel que l'ESP32. La plage de valeurs de ce signal varie en fonction du capteur et des conditions environnementales, mais elle est généralement entre 0 et 1023 pour un signal analogique de 10 bits [10]



Figure 6: Capteur d'humidité du sol

## IV.2 Relais :

Un relais est un dispositif électromécanique qui permet de commuter une tension ou un courant électrique, en utilisant une faible tension ou un courant de commande. Il est souvent utilisé pour isoler une charge électrique du circuit de commande.[11]

Les relais comme indique la figure\_7 sont largement utilisés dans de nombreuses applications électroniques, y compris les systèmes de contrôle de moteurs, les alimentations électriques, les systèmes de commande de climatisation, les systèmes de sécurité et le système d'irrigation etc.



Figure 7: Relais

## IV.3 Pompe à eau :

Une pompe d'irrigation montrée au-dessus par la figure\_8 sert à transporter l'eau de la source d'approvisionnement jusqu'à une zone spécifique de terre aride. Elle est souvent utilisée pour soutenir la croissance des cultures, entretenir les pelouses, la végétation et les champs.[12]

Pour son fonctionnement dans un système d'irrigation, elle utilise une source d'énergie pour aspirer l'eau de la source d'approvisionnement et la déplacer à travers le système d'irrigation jusqu'à la zone cible.



Figure 8:Pompe à eau

## V. Environnement Logiciel :

### V.1 L'environnement de programmation (Arduino IDE) :

L'IDE Arduino (Environnement de Développement Intégré) connu par le symbole marqué dans la figure\_9 est une plate-forme logicielle utilisée pour programmer et développer des logiciels pour les microcontrôleurs Arduino.

Il possède une interface utilisateur graphique conviviale qui permet aux utilisateurs de créer, de téléverser et de déboguer facilement des programmes pour les cartes Arduino.

Ce Logiciel est multiplateforme. Il est compatible avec différents systèmes d'exploitation tels que Windows, Linux et macOS. [13]



Figure9: Arduino IDE

### V.2 Fritzing interface :

Fritzing introduit dans la figure\_10 est un logiciel open-source pour la conception de circuits électroniques. Il est conçu pour aider les débutants à apprendre l'électronique et la programmation. Il propose également des fonctionnalités pour l'impression et la publication de vos circuits électroniques, ainsi que la possibilité de créer des images haute résolution pour une utilisation dans des présentations et des documents.[14]



Figure 10: Fritzing interface

### V.1 Le protocole MQTT :

MQTT (Message Queuing Telemetry Transport) est un protocole de communication de messagerie léger basé sur le modèle de publication/abonnement (pub/sub) comme indique la figure 11.

Il est conçu pour les appareils IoT (Internet des objets) et permet la communication entre des appareils distants avec une faible bande passante et une faible consommation d'énergie.

MQTT est composé de plusieurs éléments clés :

- **Broker** : le broker MQTT est le serveur centralisé qui gère la communication entre les appareils. Il reçoit les messages publiés par les éditeurs et les transmet aux abonnés intéressés.
- **Client** : un client MQTT peut être soit un éditeur, soit un abonné. Les clients peuvent être des appareils IoT, des applications mobiles ou des ordinateurs.
- **Sujet (Topic)** : un sujet MQTT est un nom de canal logique utilisé pour identifier le contenu des messages. Les éditeurs publient des messages sur un sujet et les abonnés s'abonnent à des sujets spécifiques pour recevoir les messages.
- **QoS (Quality of Service)** : MQTT prend en charge trois niveaux de qualité de service pour garantir la fiabilité et la distribution des messages. Les niveaux de QoS vont de 0 (at most once) à 2 (exactly once).
- **Message** : un message MQTT est une unité de données qui est publiée sur un sujet et transmise aux abonnés intéressés. Le message peut contenir des données telles que des valeurs de capteurs, des commandes ou des informations de statut.

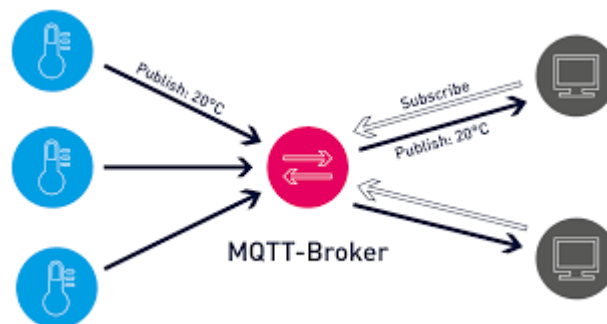


Figure 11: MQTT-Broker

## V.2 HivenMQ Cloud :

HiveMQ manifesté par la figure\_12 est une plateforme de broker MQTT (Message Queuing Telemetry Transport) conçue pour faciliter la communication entre les dispositifs IoT (Internet des Objets). HiveMQ est un broker MQTT hautement performant, capable de supporter un grand nombre de clients et un grand volume de messages.

Il propose plusieurs fonctionnalités avancées pour les entreprises, notamment la gestion de la sécurité et la haute disponibilité. La plateforme est capable de gérer des connexions MQTT sécurisées en utilisant des protocoles de sécurité tels que SSL/TLS. Elle prend également en charge l'authentification et l'autorisation des clients et peut intégrer des annuaires LDAP (Lightweight Directory Access Protocol) pour la gestion des utilisateurs.[15]



Figure 12: HiveMQ

### V.3 Node Red :

Node-RED avec un son symbole indiqué par figure\_13 est un outil open-source de programmation visuelle basé sur le langage JavaScript, qui permet de créer des flux de traitement de données pour l'Internet des Objets (IoT). Il permet de connecter des périphériques, des services et des API afin de créer des applications IoT. Il dispose d'une large bibliothèque de nœuds prêts à l'emploi, qui peuvent être facilement intégrés dans des flows pour réaliser des tâches spécifiques, tels que la récupération de données à partir de capteurs, la transformation des données, l'envoi de messages à des dispositifs, ou encore la visualisation des données.[16]



Figure 13: Node Red

## VI.Conclusion :

L'intégration de capteurs environnementaux précis et des logiciels avancés dans un système d'irrigation offre de nombreux avantages en termes d'optimisation des ressources, de productivité agricole et de durabilité environnementale. La combinaison de ces éléments permet une irrigation plus précise, basée sur les besoins réels des cultures.

# Chapitre 3 : Développement et test de système d'irrigation

## I.Introduction :

Dans cet ultime chapitre, on visualise les tests de chaque partie pour garantir la fiabilité et le bon fonctionnement du système. Nous commençant par illustrer l'organigramme du système. Ensuite, en première partie nous traitant les développements des différents capteurs et en deuxième partie on présente l'application qu'on a réalisée.

## II.Organigramme de système:

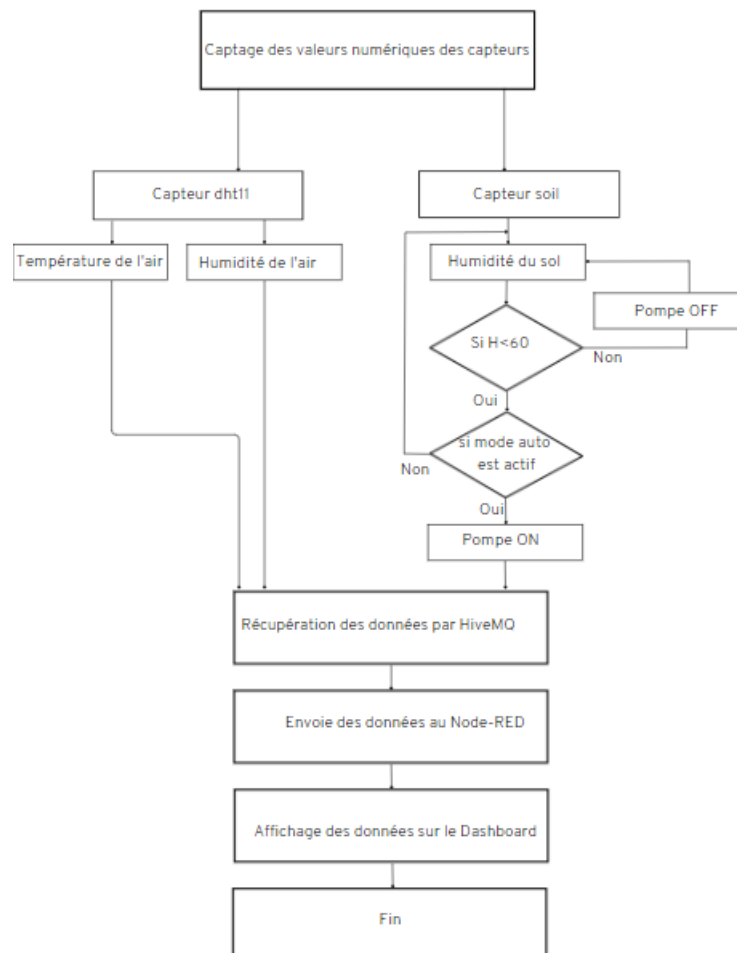


Figure 14: Organigramme du système

Une fois le système est alimenté, l'ESP lit les valeurs de température et d'humidité à partir des capteurs DHT11 et capteur d'humidité du sol. si la valeur d'humidité du sol mesurée est inférieure à 60 et le mode auto de système est désactivé le système allume la pompe et si elle est supérieure, il l'éteint.

Dans le mode manuel, la pompe peut être activée à l'aide d'un bouton switch disponible dans le dashboard.

Toutes ces valeurs sont affichées sur le dashboard et sont mises à jour continuellement. La figure 14 représente le fonctionnement global des capteurs connectés à la carte ESP32 et l'envoi des données mesurées par les capteurs.

### **III.Etapes de conception:**

#### **III.1 1ère partie : Conception de la partie électronique :**

##### **III.1.1 Description :**

Au début de notre projet, nous avons entrepris le câblage de chaque capteur individuellement : le capteur DHT11 ainsi le capteur de sol. Cette approche a permis de bien comprendre le fonctionnement de chaque capteur, ainsi que son raccordement à l'esp32.

Une fois cette étape achevée, nous avons procédé à l'affectation du rassemblement total. Cette méthode a permis de s'assurer que tous les capteurs étaient correctement raccordés et que les données étaient transmises avec précision. Ce processus de câblage rigoureux a permis d'éviter les erreurs et de garantir un fonctionnement optimal du circuit.

##### **III.1.2 Configuration de la carte esp32 :**

On commence tout d'abord la configuration de l'environnement de développement pour les cartes ESP32, il est nécessaire d'installer les outils de développement spécifiques à cette carte. Pour ce faire, premièrement on ajoute l'URL du gestionnaire de cartes ESP32 implémenté par l'Annexe1 dans l'IDE Arduino. Cette action permet de télécharger les outils et les bibliothèques nécessaires. Ensuite on règle le gestionnaire de cartes : on clique sur Outils → Type de carte → Gestionnaire de carte comme l'indique l'Annexe2. Dès le choix de type de carte dev module, On installe la bibliothèque esp32 comme la montre l'annexe3.

On test après le fonctionnement de notre carte. Chaque carte est préalablement programmée avec un code qui fait clignoter la LED bleue située sur la carte et qui scanne les réseaux Wi-Fi à proximité. Le moniteur série donne ce résultat qui est présenté par la figure 15.



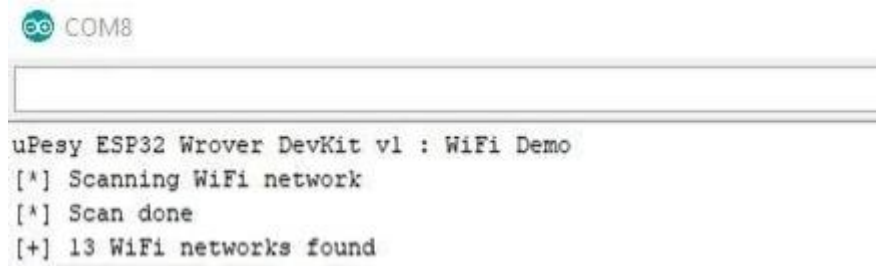


Figure 15 : Test de fonctionnement de la carte

Après La vérification maintenant notre logiciel Arduino IDE est prêt pour réalise des projets avec cette carte.

### III.1.3 Câblage du circuit :

#### III.1.3.1 Test de capteur d'humidité et température DHT11:

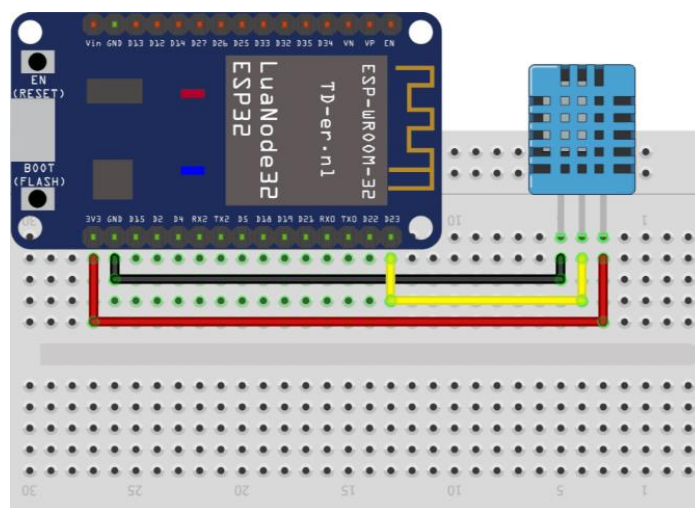


Figure 16 : Câblage de DHT11

Pour câbler ce capteur, premièrement on inclut la bibliothèque DHT11 en cliquant sur "Croquis" -> "Inclure une bibliothèque" -> "Gérer les bibliothèques".

Recherchez la bibliothèque "DHT sensorlibrary" et installez-la.

Ensuite, On branche le capteur avec esp32 par D4 et on relie le VCC et le GND (figure 16).

Finalement on Téléverse le code sur la carte ESP32 en utilisant l'IDE Arduino et on ouvre le moniteur série pour voir les résultats et la figure\_17 le montre clairement :

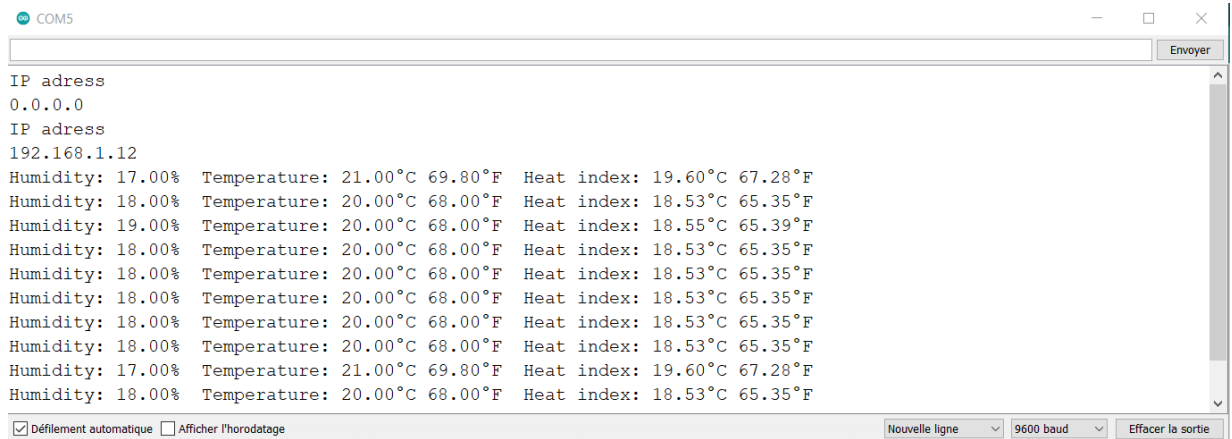


Figure 17 : Test le fonctionnement du capteur DHT11

### III.1.3.2 Test de Capteur d'humidité de sol :

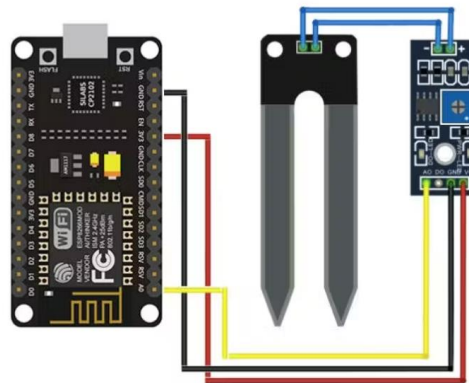


Figure 18 : Câblage du capteur de sol

Pour Connectez le capteur d'humidité du sol à la carte ESP32, il faut tout d'abord relier la broche de sortie du capteur à une broche analogique de la carte (figure 18).

Après coder le capteur on téléverse le code sur la carte et on ouvre le moniteur série pour voir les résultats. On affiche les valeurs sous forme de pourcentage après conversion.

Le moniteur série donne ce résultat présenté par la figure\_19. Ce qui affirme le bon fonctionnement de ce capteur.

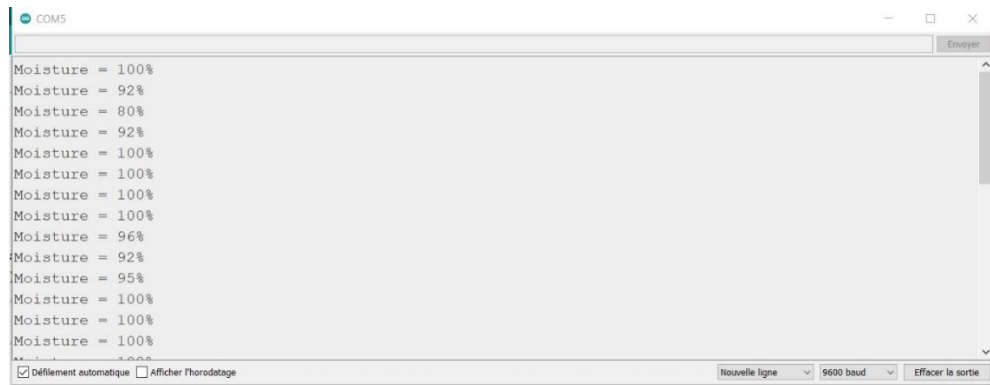


Figure 19 : Test de fonctionnement du capteur de sol

### III.1.3.3 Test de La pompe :

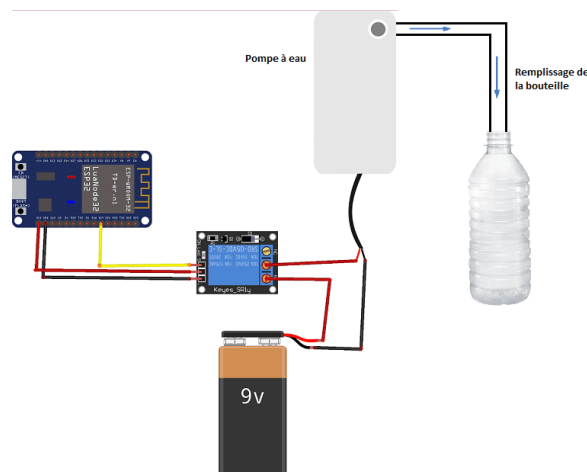


Figure 20 : Test de fonctionnement de la pompe

Afin de connecter la pompe à notre carte ESP32 on utilise un relais et une pile de 9V. On choisit une sortie analogique « D5 » et on relie le relais avec GND et VCC comme la figure 20 montre clairement. Ensuite, on connecte la borne du relais à la borne positive de la pile et l'autre borne du relais à la borne positive de la pompe et la borne négative de la pile à la borne négative de la pompe.

Dans la fonction `loop()` de l'arduino IDE la pompe est activée en écrivant HIGH sur sa broche, puis le relais est activé en écrivant HIGH sur sa broche, ce qui permet à l'alimentation d'atteindre la pompe.

### III.1.3.4 Assemblage du circuit :

La figure 21 représente le schéma explicatif par Fritzing de montage final qui nous montre la connexion entre les différents modules avec la carte Esp 32.

- DHT 11 se brancher avec l'Esp32 par 1 fil (D4) + VCC, GND.
- Capteur d'humidité du sol se branche avec l'Esp32 par 1 fil (A0) + VCC, GND.
- Relais de pompe se brancher avec l'Arduino par 1 fil (D5) + VCC, GND.

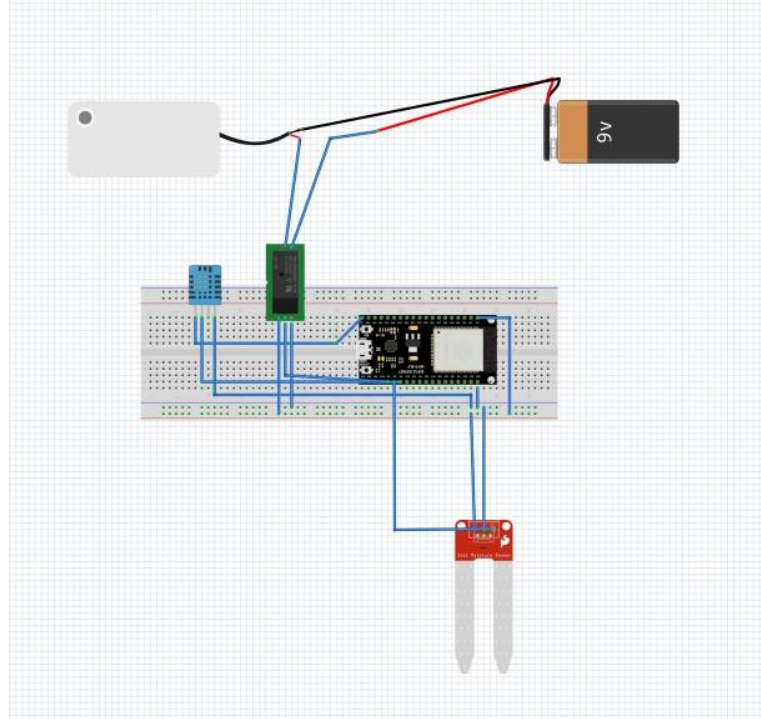


Figure 21: Assemblage des composants

## IV.2ème Partie : Communication et transmission des données :

### IV.1 Développement de Tableau de bord avec node red :

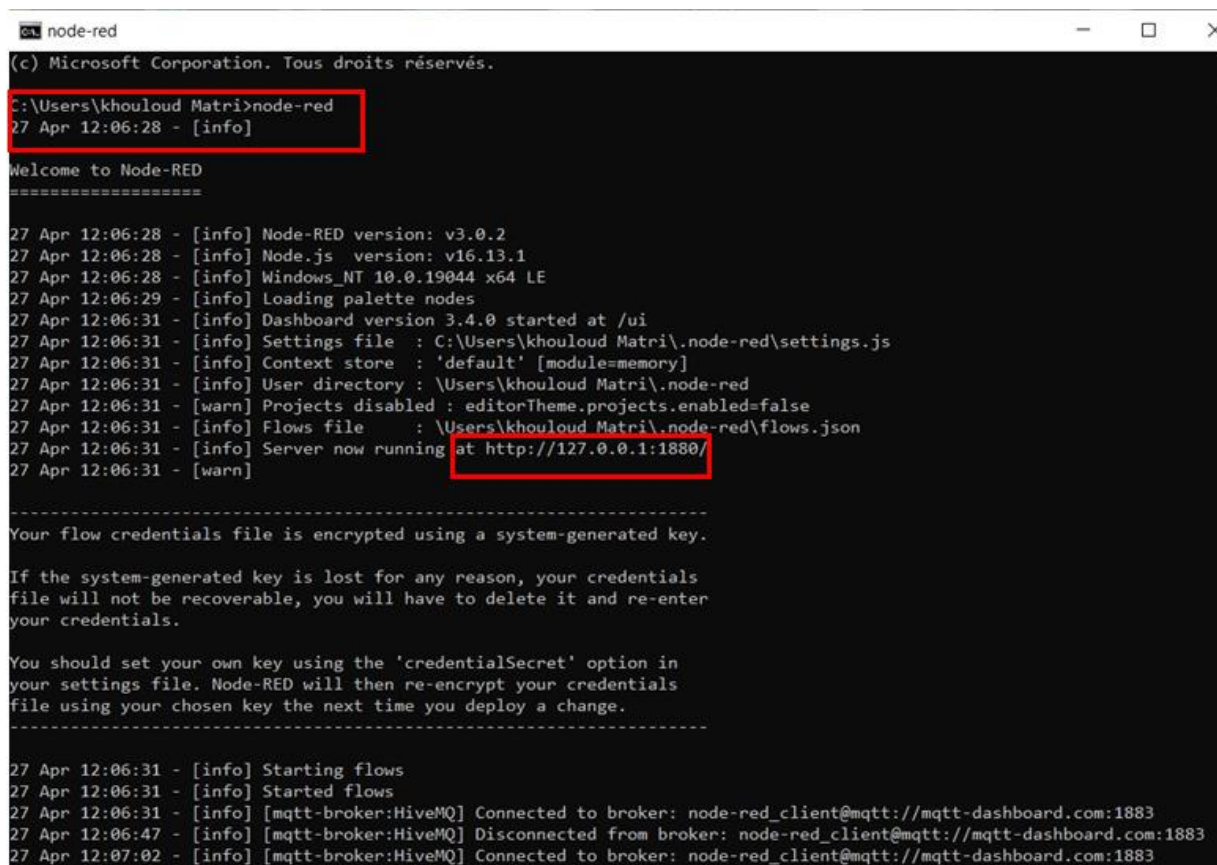
#### IV.1.1 Installation :

Pour créer un tableau de bord de base dans Node-RED, vous pouvez suivre les étapes suivantes :

Pour commencer, tous d'abord on doit avoir Node.js installé sur notre système.

Une fois que Node.js est installé, on ouvre une interface de ligne de commande ou un terminal et on exécute la commande suivante : `npm install -g --unsafe-perm node-red` (comme montre la figure 22)

Pour lancer Node-RED on tape simplement la commande suivante dans notre terminal :



```
node-red
(c) Microsoft Corporation. Tous droits réservés.
C:\Users\khouloud Matri>node-red
27 Apr 12:06:28 - [info]

Welcome to Node-RED
=====
27 Apr 12:06:28 - [info] Node-RED version: v3.0.2
27 Apr 12:06:28 - [info] Node.js version: v16.13.1
27 Apr 12:06:28 - [info] Windows_NT 10.0.19044 x64 LE
27 Apr 12:06:29 - [info] Loading palette nodes
27 Apr 12:06:31 - [info] Dashboard version 3.4.0 started at /ui
27 Apr 12:06:31 - [info] Settings file : C:\Users\khouloud Matri\.node-red\settings.js
27 Apr 12:06:31 - [info] Context store : 'default' [module=memory]
27 Apr 12:06:31 - [info] User directory : \Users\khouloud Matri\.node-red
27 Apr 12:06:31 - [warn] Projects disabled : editorTheme.projects.enabled=false
27 Apr 12:06:31 - [info] Flows file : \Users\khouloud Matri\.node-red\flows.json
27 Apr 12:06:31 - [info] Server now running at http://127.0.0.1:1880/
27 Apr 12:06:31 - [warn]

-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----
27 Apr 12:06:31 - [info] Starting flows
27 Apr 12:06:31 - [info] Started flows
27 Apr 12:06:31 - [info] [mqtt-broker:HiveMQ] Connected to broker: node-red_client@mqtt://mqtt-dashboard.com:1883
27 Apr 12:06:47 - [info] [mqtt-broker:HiveMQ] Disconnected from broker: node-red_client@mqtt://mqtt-dashboard.com:1883
27 Apr 12:07:02 - [info] [mqtt-broker:HiveMQ] Connected to broker: node-red_client@mqtt://mqtt-dashboard.com:1883
```

Figure 22: Lancement de Node Red

Pour accéder à l'interface utilisateur de Node-RED, on tape cette url sur le navigateur <http://127.0.0.1:1880/>

#### IV.1.2 Création de tableau de bord :

Commençant par l'installation du package "node-red-dashboard" via la gestion des palettes. Puis on ajoute les nœuds de tableau de bord à notre flux et on les configure selon le besoin. On connecte correctement les nœuds pour interagir avec les données.

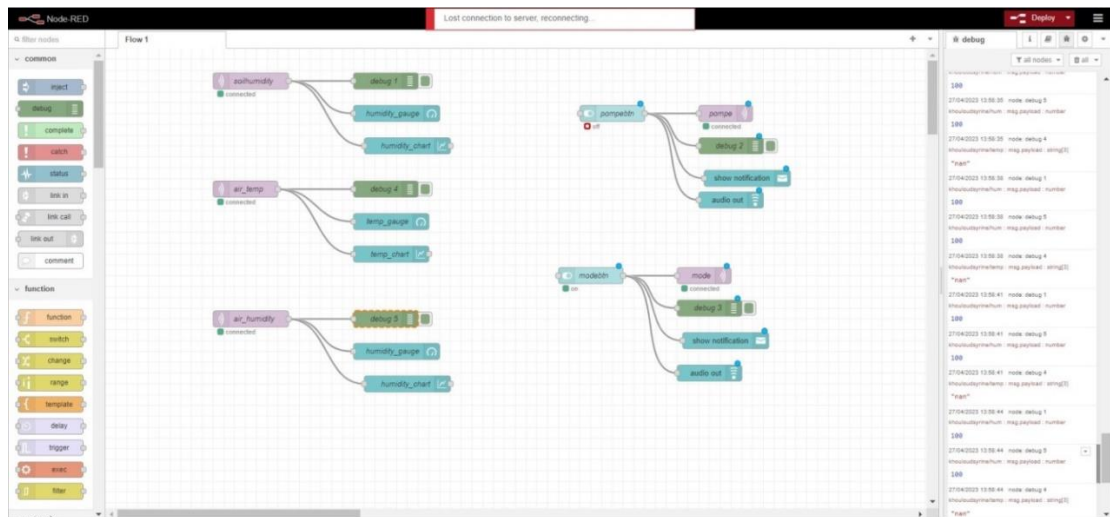


Figure 23: Développement avec Node red

On crée ensuite une notification sur le dashboard. Chaque fois qu'on appuie sur un bouton il indique son état avec un effet audio. Les nœuds sont indiqués par la figure\_24

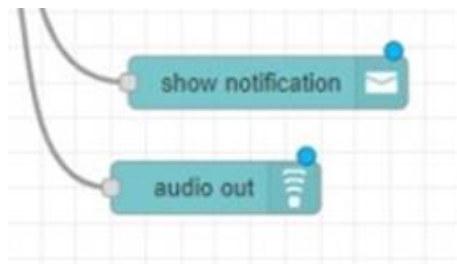


Figure 24: Notification avec Node red

Une fois le flux est déployé, on accède au tableau de bord en utilisant l'adresse IP et le port fournis par Node-RED. On tape cet url sur le navigateur <http://127.0.0.1:1880/ui>

Le dashboard que nous créons affiche les données de deux manières différentes à travers :

- une jauge.
- une courbe temporelle.

## IV.2 Communication du dashboard avec HiveMQBroker (partie configuration :

### IV.2.1 Configuration HiveMQ :

Nous utilisons "broker.hivemq.com" comme une adresse de broker MQTT public fournie par HiveMQ.

Nous connectant que client MQTT et par l'abonnement à des topics cela nous permet de publier des messages et d'exécuter des instructions différées.

Pour que notre dashboard puisse connecter avec le HiveMQ broker il faut entrer ces paramètres dans la configuration des composants comme indique l'annexe4.

#### IV.2.2 L'envoi des données par le broker MQTT : Méthodepublish :

La méthode "publish" dans notre dashboard est utilisée pour envoyer des données à un broker MQTT. Elle permet d'envoyer des messages contenant les informations à afficher ou à traiter. Ces messages peuvent être reçus par des clients MQTT abonnés au même topic sur le broker HiveMQ.

Dans notre cas, on a deux méthodes publish. Chaque fois que l'état des deux boutons (pompe, mode-auto) change, on va le publier sur les topics du HiveMQ choisis.

« khouloudsyryne/mode » pour l'état du bouton mode-auto.

« khouloudsyryne/pompe » pour l'état du bouton pompe.

Cette méthode qui est présenté par la figure 25 nous donne un output qu' il faut choisir, dans la construction de notre dashboard, « mqtt out node » qu'on va lier avec un bouton et puis compléter la configuration requise pour la communication.

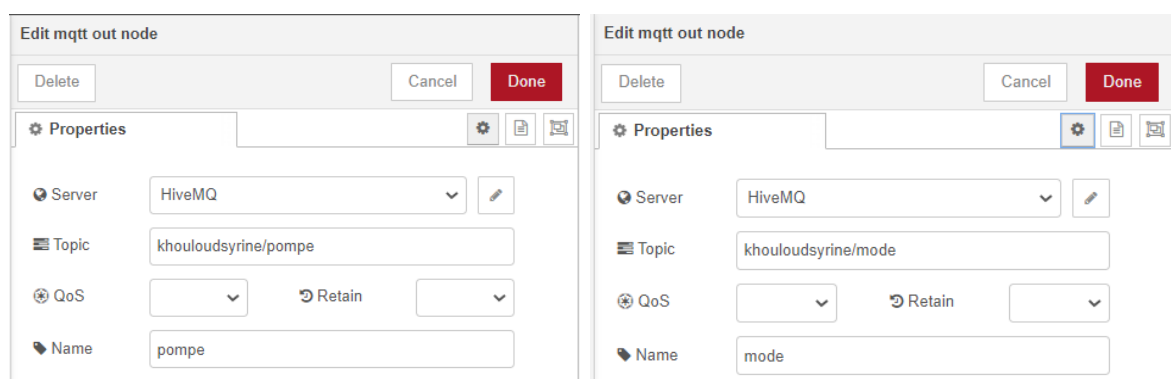


Figure 25 : Configuration de l'output pompe

#### IV.2.3 Test de la méthode publish:

Ouvrir tout d'abord le HiveMQ broker <https://www.hivemq.com/demos/websocket-client/>

Se connecter sur le host « mqtt-dashboard.com ».



S'inscrire sur les deux topics « khouloudsyrene/pompe » et « khouloudsyrene/mode » affiché par la figure 26 pour suivre les messages envoyés par les boutons du dashboard.

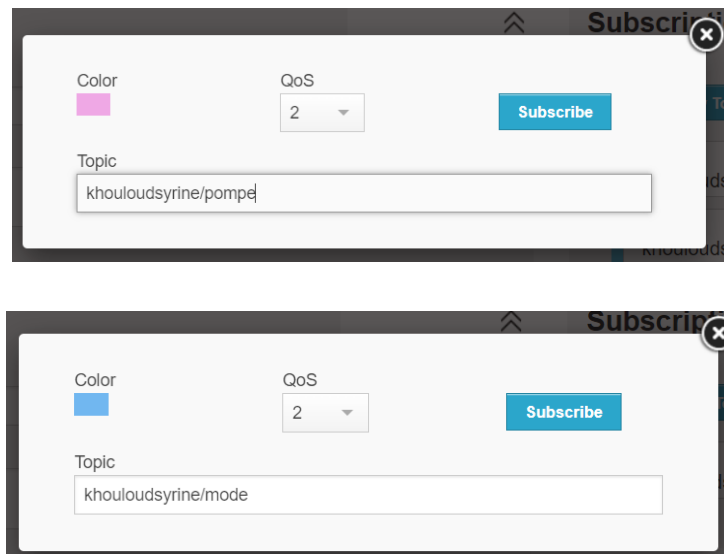


Figure 26 : Connexion aux topics

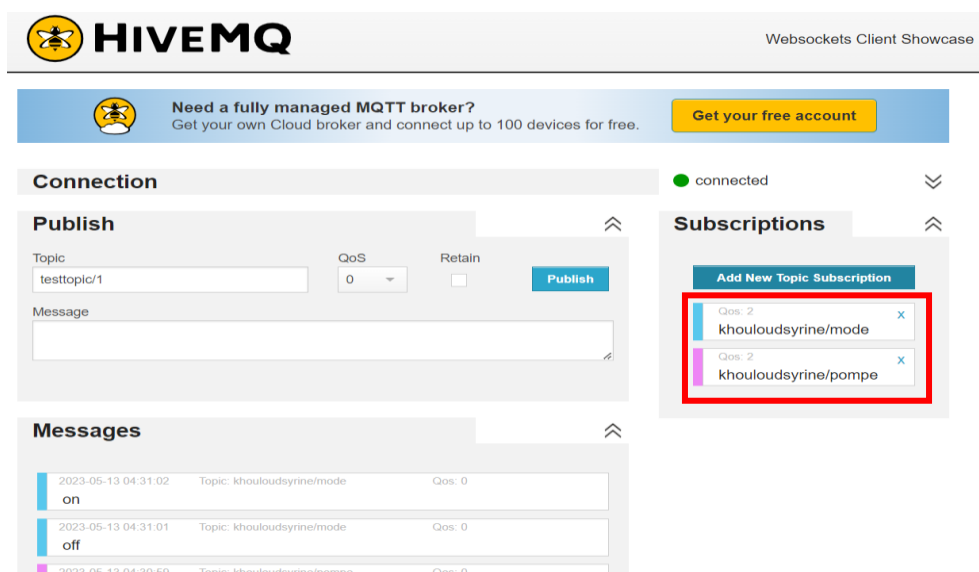


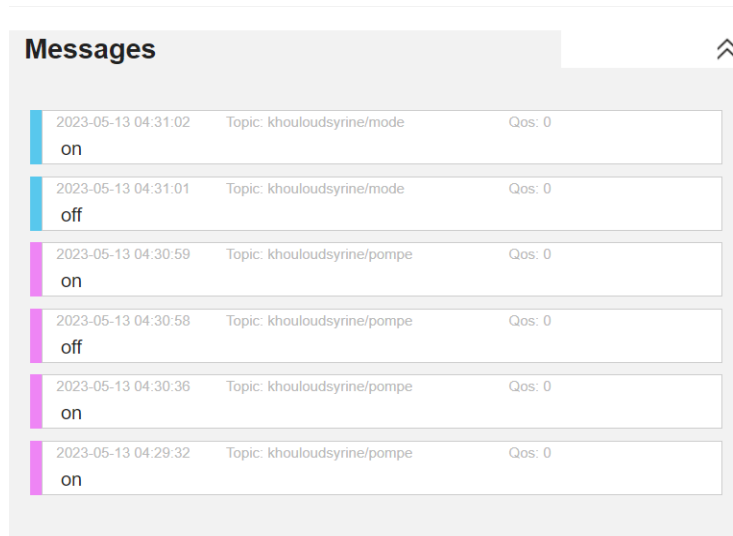
Figure 27: Connexion réussite aux topics

La connexion est prête, comme l'indique la figure 27. Maintenant on peut suivre les messages envoyés :

On change l'état en cliquant sur le bouton qui est présenté et on observe ce changement en temps réel sur le HiveMQ broker.



On reçoit dans l'interface de HiveMq les messages reçus à partir de dashboard affiché par la figure\_28.



Messages		
2023-05-13 04:31:02	Topic: khouloudsyrine/mode	Qos: 0
on		
2023-05-13 04:31:01	Topic: khouloudsyrine/mode	Qos: 0
off		
2023-05-13 04:30:59	Topic: khouloudsyrine/pompe	Qos: 0
on		
2023-05-13 04:30:58	Topic: khouloudsyrine/pompe	Qos: 0
off		
2023-05-13 04:30:36	Topic: khouloudsyrine/pompe	Qos: 0
on		
2023-05-13 04:29:32	Topic: khouloudsyrine/pompe	Qos: 0
on		

**Figure 28 : Affichage des messages dans HiveMQ**

#### IV.2.1 La réception des données par le MQTT broker: méthode subscribe :

La méthode "subscribe" dans une dashboard sur HiveMQ est utilisée pour s'abonner à des topics MQTT spécifiques sur le broker. Cela permet à la dashboard de recevoir les messages publiés par d'autres clients sur ces topics. Les données reçues peuvent ensuite être utilisées pour afficher des informations ou déclencher des actions sur la dashboard.

Dans notre cas, on a trois méthodes subscribe. On va lire et afficher sur le dashboard les données reçues « température, humidité de l'air, humidité du sol ».

Cette méthode nous donne un input donc il faut choisir un « mqtt in node » qu'on va lier avec une gauge et une courbe. Il ne faut pas oublier de compléter la configuration requise pour la communication.

La composante debug nous permet d'afficher les données reçues ou envoyées sur un terminal pour suivre l'avancement de notre dashboard montré par la figure 29 :

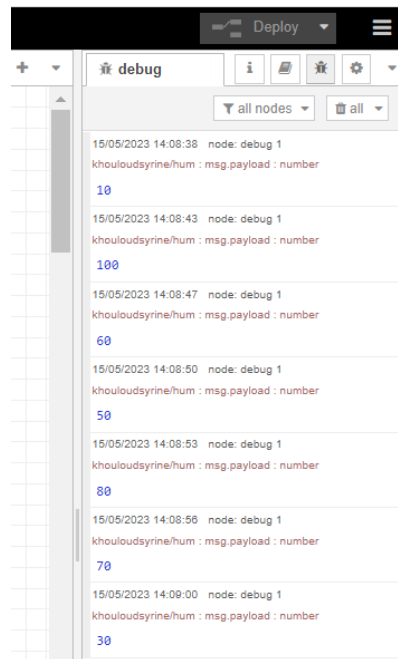


Figure 29 : Affichage des messages reçus par les capteurs sur la partie Debug

#### IV.2.1 Test de La méthode subscribe:

On connecte sur l'host « mqtt-dashboard.com » puis on publie manuellement des valeurs sur les topics comme la montre la figure 30 pour le topic « khouloudsyryne/hum »

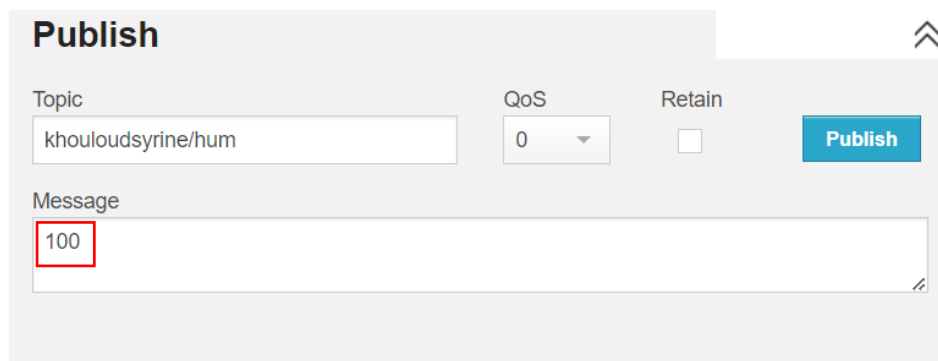
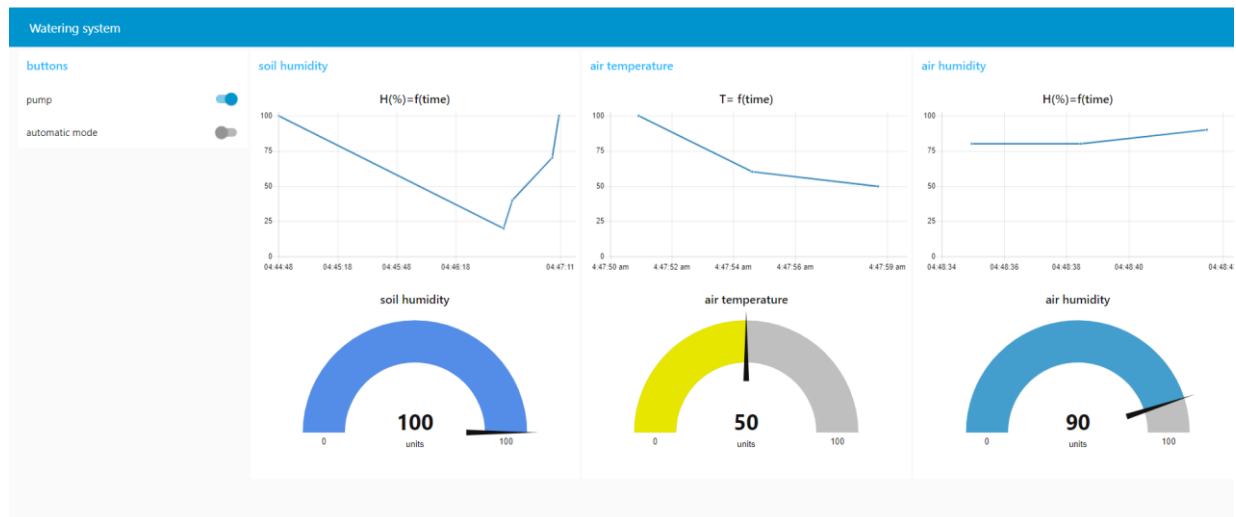


Figure 30: L'envoi manuel sur le topic khouloudsyryne/hum

Ensuite on suit l'affichage sur le dashboard web. on répète le même test pour les autres topics réservés pour la température et l'humidité d'air.

On obtient ce résultat :



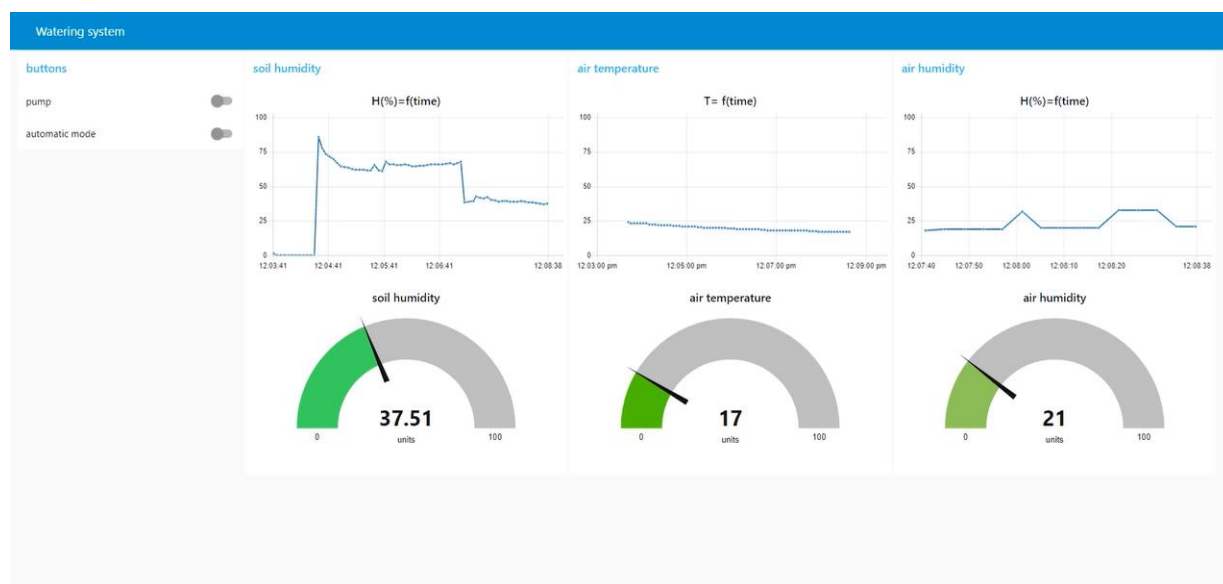
**Figure 31 : Affichage des données sur le dashboard**

La communication entre la dashboard et HiveMQ s'effectue avec succès, permettant ainsi l'échange de messages et de données. Les messages publiés par la dashboard sont transmis au broker HiveMQ et peuvent être reçus par les clients MQTT abonnés aux topics correspondants. Ainsi que les messages envoyés par les autres clients MQTT seront affichés sur notre dashboard. Cette interaction fluide garantit une communication efficace et fiable entre la dashboard et le système MQTT.

### IV.3 Test final du système d'irrigation :

Messages			
2023-05-26 12:56:08	Topic: khouloudsyryne/temp	Qos: 0	19.8
2023-05-26 12:55:28	Topic: khouloudsyryne/hum	Qos: 0	30.1
2023-05-26 12:54:59	Topic: khouloudsyryne/airh	Qos: 0	30
2023-05-26 12:54:33	Topic: khouloudsyryne/temp	Qos: 0	20
2023-05-26 12:54:15	Topic: khouloudsyryne/hum	Qos: 0	30.8
2023-05-26 12:53:52	Topic: khouloudsyryne/airh	Qos: 0	32
2023-05-26 12:53:42	Topic: khouloudsyryne/temp	Qos: 0	24
2023-05-26 12:53:28	Topic: khouloudsyryne/hum	Qos: 0	51.2
2023-05-26 12:53:01	Topic: khouloudsyryne/airh	Qos: 0	30.4
2023-05-26 12:52:49	Topic: khouloudsyryne/temp	Qos: 0	25
2023-05-26 12:52:40	Topic: khouloudsyryne/hum	Qos: 0	50.3

**Figure 32 : Affichage des messages du système dans HiveMQ**



**Figure 33 : Affichage des données du système sur le dashboard**

Après avoir programmé le système, il est temps de tester son fonctionnement. Nous allons activer le mode automatique, où la pompe d'irrigation sera activée via le relais. La pompe se mettra en marche lorsque le niveau d'humidité descendra en dessous de 60%, la marge idéale pour ce capteur entre 60% et 70%, et s'arrêtera dès qu'il dépassera ce seuil prédéfini. Cela permettra de confirmer si le système réagit de manière appropriée aux variations d'humidité. En cas de situation d'urgence, notamment lorsque le capteur présente un dysfonctionnement, il est possible de basculer vers le mode manuel du système d'irrigation intelligent. Pour ce faire, il suffit de désactiver le mode automatique et d'utiliser le bouton dédié à la pompe pour effectuer un test de fonctionnement. L'observation de l'évolution des courbes de mesure nous a permis d'évaluer la performance du capteur d'humidité et de température de l'air qui varie au cours de temps. la plage idéale de température est comprise entre 0 et 50 ° C (+/- 2 ° C) et la plage d'humidité entre 20 et 90% (+/- 5%). En analysant les résultats des tests et en prenant en compte les observations faites, il sera possible d'apporter les ajustements nécessaires au programme du système d'irrigation intelligent. Ces ajustements visent à optimiser son fonctionnement en garantissant une réactivité adéquate aux variations d'humidité et en assurant la précision des mesures fournies par le capteur.

## **V.Evaluation des performances du système d'irrigation intelligent :**

Le système d'irrigation intelligent basé sur ESP32, HiveMQ et Node-RED se distingue par ses performances supérieures par rapport à d'autres systèmes d'irrigation traditionnels. Son

évaluation met en évidence plusieurs points forts qui le rendent plus efficace et plus avancé technologiquement.

Dans notre travail, l'objectif était de déterminer l'efficacité et la fiabilité de ce système dans des conditions réelles d'utilisation.

Nous avons mesuré le temps de réponse du système, qui représente le délai entre la collecte des données d'humidité du sol et l'activation des actionneurs d'irrigation. Nos résultats ont montré que le système était capable de fournir des temps de réponse rapides et cohérents, permettant ainsi une irrigation précise et opportune. Cette réactivité est essentielle pour garantir que les plantes reçoivent la quantité d'eau nécessaire au bon moment.

En ce qui concerne la communication, nous avons constaté que la liaison entre ces technologies était fiable et robuste. Les messages MQTT étaient transmis sans erreur, et la connexion était stable même dans des environnements perturbés. Cela garantit une transmission de données précise et cohérente entre les différents composants du système.

En termes de gestion de la charge, le système a démontré une capacité à gérer efficacement un grand nombre de capteurs d'humidité du sol. Nous avons pu connecter plusieurs capteurs au système sans rencontrer de problèmes de performance ou de saturation. Cela permet une surveillance étendue des conditions du sol, ce qui est crucial pour une irrigation précise et ciblée.

## **VI.Conclusion :**

En conclusion, ce chapitre a permis de mettre en place un système d'irrigation complet en réalisant le câblage de la partie électronique et en développant un tableau de bord dédié. Grâce à cette intégration efficace, la communication entre les deux parties du système a été établie avec succès. En résumé, ce travail démontre notre capacité à créer une solution d'irrigation fonctionnelle et cohérente en mettant en œuvre des composantes électroniques et logicielles interconnectées.

## Conclusion générale :

En conclusion, l'utilisation d'un système d'irrigation intelligent offre de nombreux avantages. Ce système permet une gestion précise, efficace et durable de l'irrigation agricole grâce à l'intégration des capteurs, du contrôleur et de systèmes de gestion des données.

Pour concevoir ce projet, nous avons fait un tri des meilleurs composants pour notre système ainsi que des logiciels qui nous ont permis une meilleure gestion de la base de données et les interfaces développées. En effet, nous avons opté pour l'utilisation de :

L'ESP32, en tant que microcontrôleur puissant et polyvalent, offre une connectivité Wi-Fi et une capacité de traitement permettant de collecter et d'analyser les données des capteurs.

Node-RED permet de créer des flux de données personnalisés, d'intégrer des règles de décision et d'automatiser les tâches liées à l'irrigation.

HiveMQ, en tant que broker MQTT, assure une communication fiable et sécurisée entre les différents composants du système. Il permet l'échange de données en temps réel entre l'ESP32, Node-RED.

Notre travail nous a mené à bien apercevoir de nouveaux horizons dans la réalisation et le perfectionnement des systèmes de commande à distance comme il nous a conduit à s'aviser de la nécessité de la mise en place d'un système de sécurité permettant le transfert parfait des données collectées. Ce système offre des temps de réponse rapides, une communication fiable et une capacité de gestion de charge adéquate. Il représente une solution prometteuse pour l'automatisation de l'irrigation agricole, permettant une utilisation efficace des ressources en eau et une croissance optimale des plantes

Concernant les éventualités de perfectionnement de notre travail, plusieurs démarches peuvent être considérées. Nous citons comme exemple les tâches suivantes :

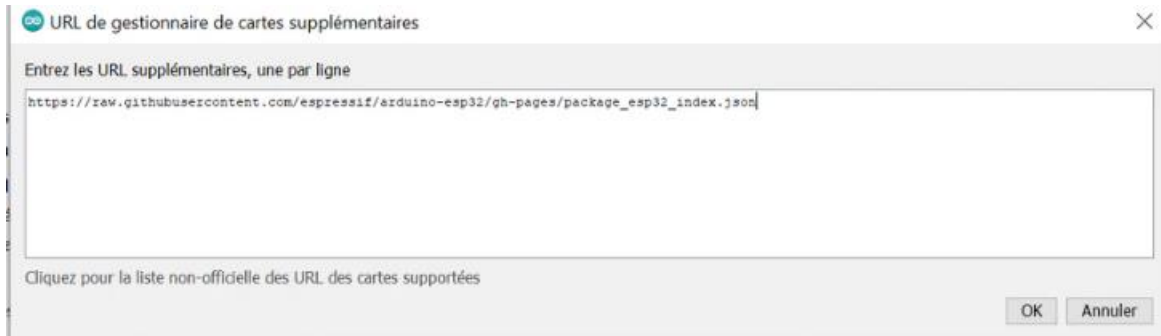
- Accorder aux utilisateurs l'opportunité de modifier certaines fonctions selon leurs besoins.
- Installer une deuxième pompe d'eau qui sera fonctionnelle systématiquement en cas de panne de la pompe principale.
- Mesurer la vitesse de croissance des plantes afin d'ajuster les conditions de culture pour garantir un rendement meilleur.

## Bibliographie

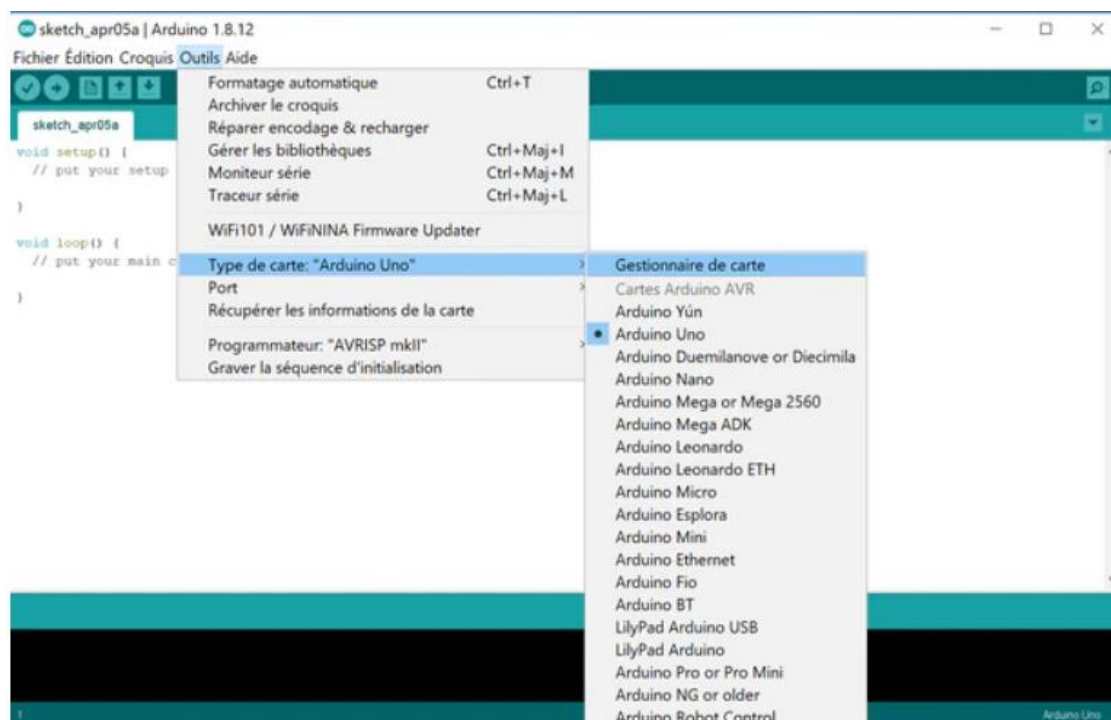
- [1] « That 'Internet of Things' Thing. » Ashton, K. (2009).
- [2] « The Internet of Things: A survey. » Atzori, L., Iera, A., & Morabito, G. (2010).
- [3] « Internet of Things: From Research and Innovation to Market Deployment. » River Publishers. Vermesan, O., & Friess, P. (Eds.). (2014).
- [4] "Traditional Irrigation Systems: An Overview" - R. Kaur, et al. (2019)
- [5] "IoT-based Smart Irrigation Systems: Challenges, Opportunities, and Future Directions" - M. Al-Fuqaha, et al. (2018)
- [6] "10 Reasons to Use ESP32 for IoT Projects" - S. Gupta (2019)
- [7] "5 Reasons to Use HiveMQ for Your MQTT Projects" - M. Krieg (2019)
- [8] <https://www.espressif.com/en/products/socs/esp32>
- [9] <https://www.adafruit.com/product/386>
- [10] <https://anrcatalog.ucanr.edu/pdf/8423.pdf>
- [11] <https://www.mouser.com/>
- [12] <https://circuitdigest.com/microcontroller-projects/water-pump-control-using-relay-and-arduino>
- [13] <https://www.arduino.cc/>
- [14] <https://fritzing.org/>
- [15] <https://www.hivemq.com/>
- [16] <https://nodered.org/>

## Annexes

### Annexe1 : Url de gestionnaire de la carte esp32

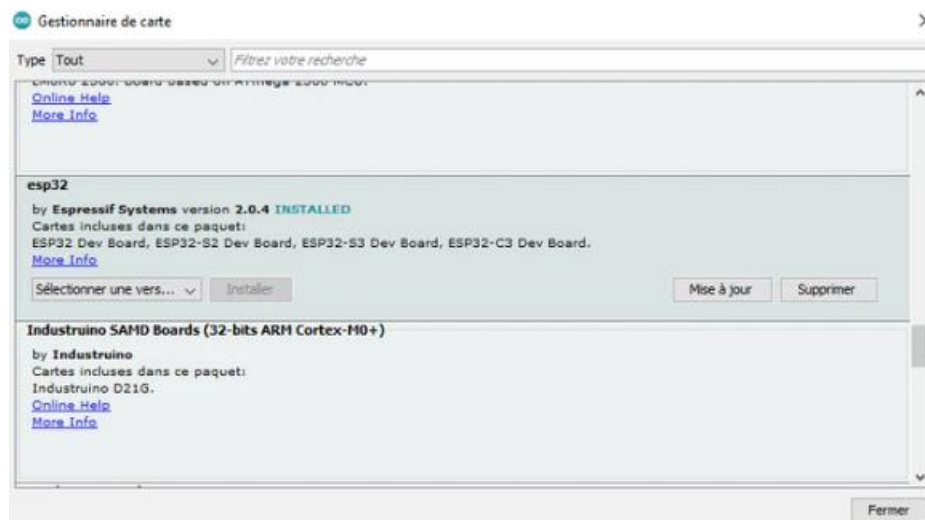


### Annexe2: Gestionnaire de la carte esp32





### Annexe3 : Installation de la bibliothèque esp32



### Annexe4: Configuration de la connexion avec HiveMQ

