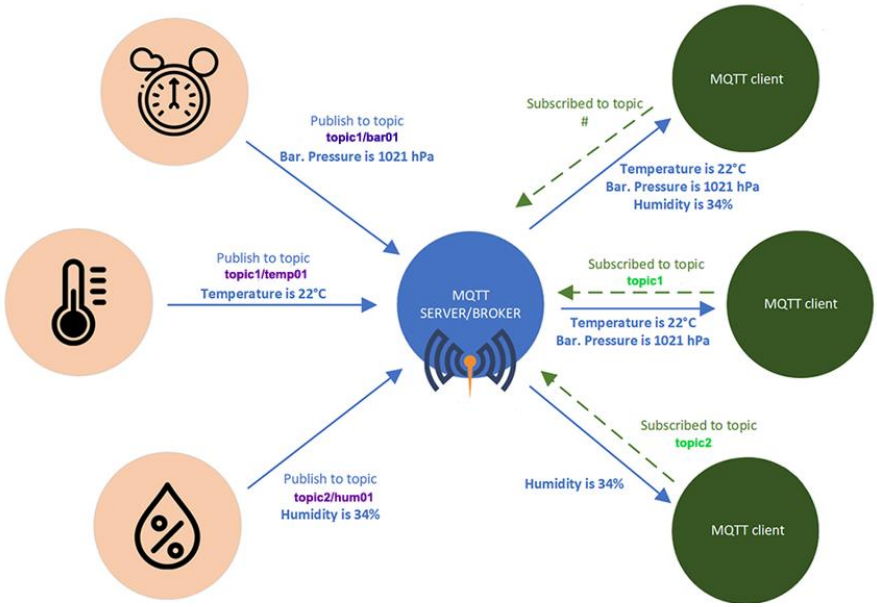
Mise en œuvre de MQTT et Node-Red

URL de référence : https://www.youtube.com/watch?app=desktop&v=FU6Henjf_Qs

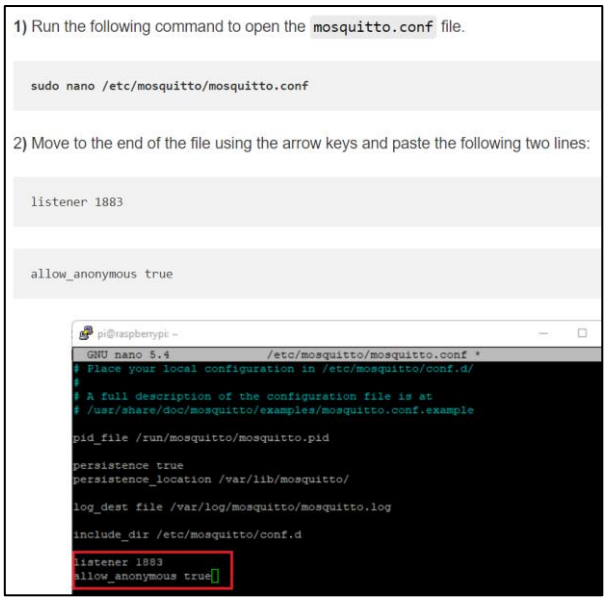
Table des matières



1 Installer un broker sur Raspberry Pi

- sudo apt-get update
- sudo apt-get install mosquitto
- sudo apt-get install mosquitto-clients
- sudo reboot

Autoriser l'accès distant : https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/




```
sudo systemctl restart mosquitto
```

2 Premier test

Le broker MQTT écoute sur le port 1883

- Je souscris à un topic sur le broker situé à l'adresse 192.168.1.45
- Le nom du topic est **`vincent/test`**

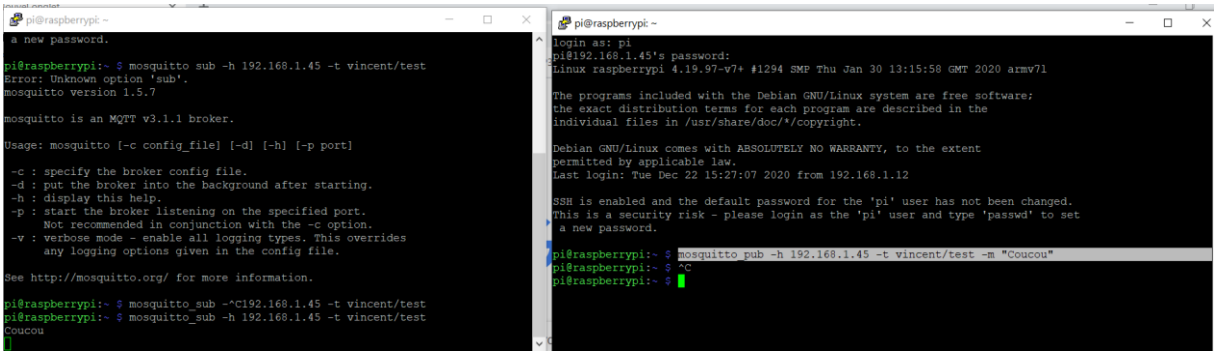
```
mosquitto_sub -h 192.168.1.45 -t vincent/test
```

⇨ le module se met en attente de recevoir des données du topic vincent/test

Sur un autre terminal, on écrit :

```
mosquitto_pub -h 192.168.1.45 -t vincent/test -m "Coucou"
```

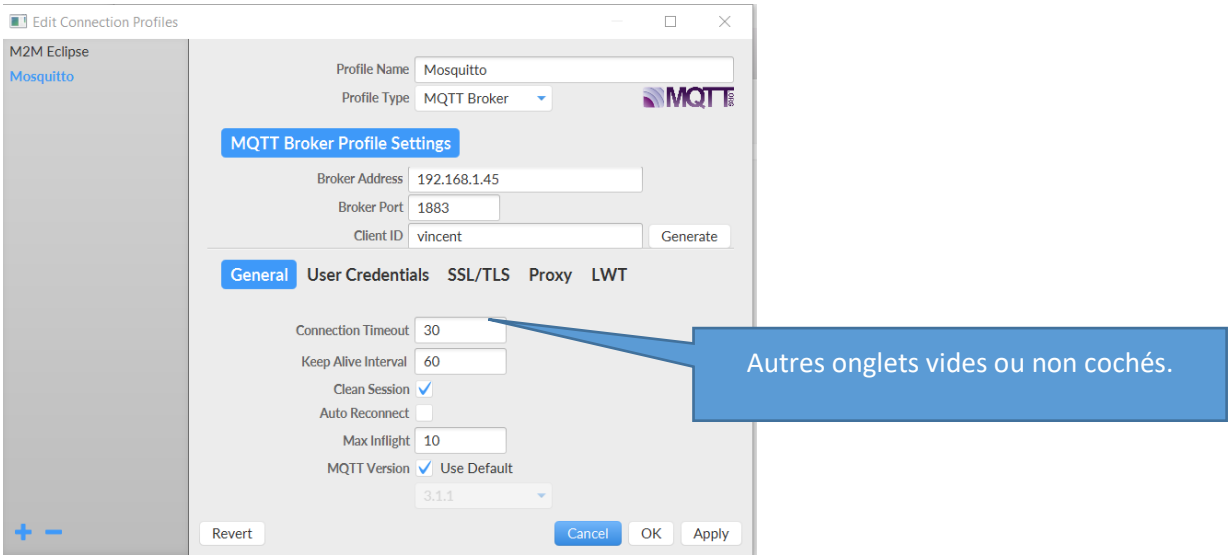
Le 1er terminal reçoit alors coucou



S'il faut redémarrer mosquitto

```
sudo systemctl enable mosquitto.service
```

3 Test d'émission/réception avec MQTT.FX



4 Sécurisation : à 23 minutes 40 de la vidéo https://www.youtube.com/watch?v=ubqzvbox5dc

On édite le fichier `/etc/mosquitto/mosquitto.log`

On efface la dernière ligne : `include_dir /etc/mosquitto/conf.d`

On la remplace par

```
allow_anonymous false
password_file /etc/mosquitto/pwfile
```

Maintenant, on définit le nom de l'utilisation et du mot de passe

```
sudo mosquitto_passwd -c /etc/mosquitto/pwfile NomDeUtilisateur
```

- il demande le password

Pour subscrire à un topic :

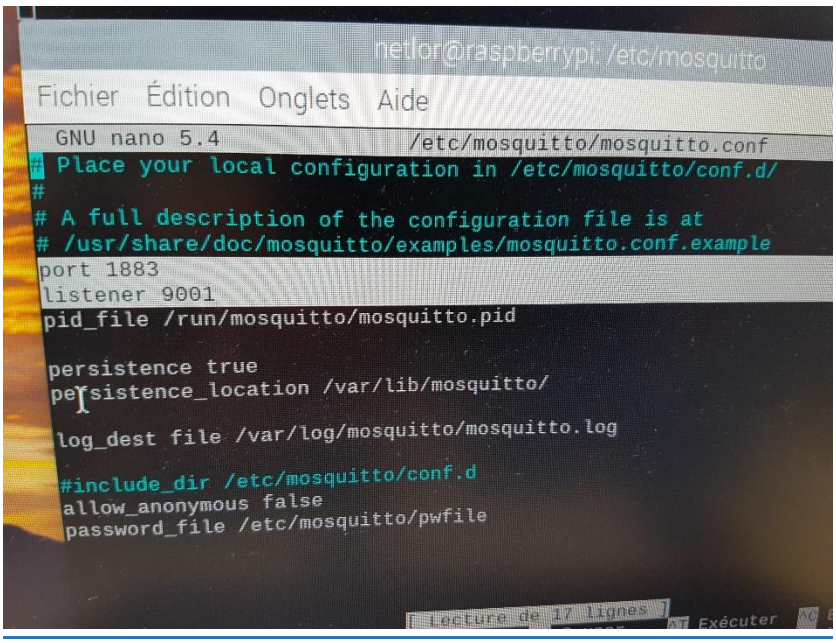
```
mosquitto_sub -d -u NomUtilisateur -P motDePasse -t VotreTopic
```

Pour envoyer une donnée :

```
mosquitto_pub -d -u NomUtilisateur -P motDePasse -t VotreTopic -m VotreMessage
```

Remarque :

Un test réalisé par des étudiants n'a pas fonctionné sur le réseau local. Ils ont dû ajouter le numéro de port et le listener au fichier **/etc/mosquitto/mosquitto.conf**



5 NodeRed

Outil de programmation par blocs.

Node-red est installé de base sur Raspbian, mais il est conseillé de le réinstaller ou le mettre à jour.

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

Pour lancer Node-Red

```
pi@raspberrypi:~ $ node-red-start

      Node-RED

Once Node-RED has started, point a browser at http://192.168.1.45:1880
On Pi Node-RED works better with the Firefox or Chrome browser

Use    node-red-stop                          to stop Node-RED
Use    node-red-start                         to start Node-RED again
Use    node-red-log                           to view the recent log output
Use    sudo systemctl enable nodered.service  to autostart Node-RED at every boot
Use    sudo systemctl disable nodered.service to disable autostart on boot

To find more nodes and example flows - go to http://flows.nodered.org
```

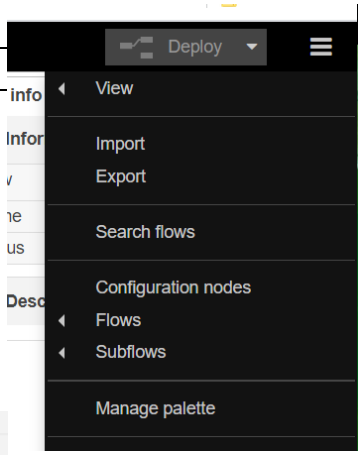
S'il manque des paquets, il faut les installer :

```
sudo apt-get install npm
```

- Redémarrer node-red

Dans un navigateur Web, taper l'URL fournie

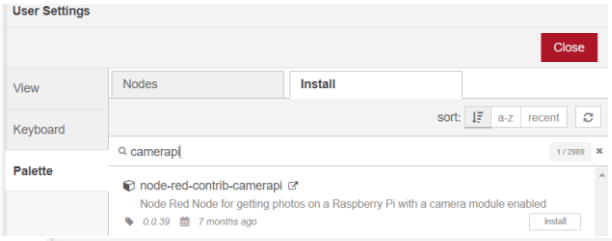
Aller dans le menu "Manage Palette"



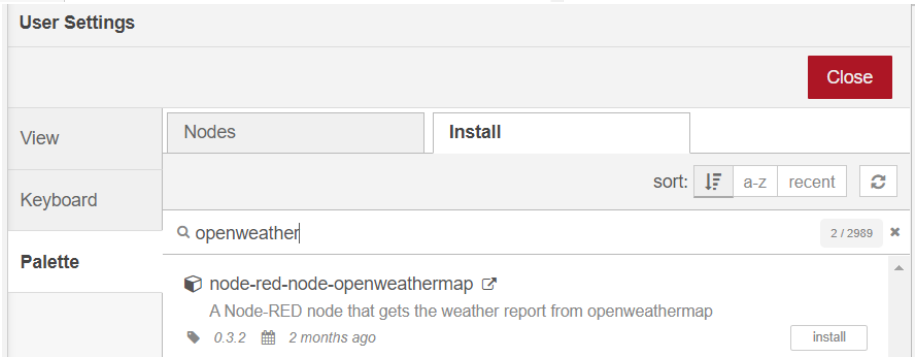
Installer le dashboard (node-red-dashboard)

⇨ Ensuite, à gauche de l'écran, on va trouver l'onglet dashboard

Si on veut ajouter la Raspberry-pi, on ajoute camera pi

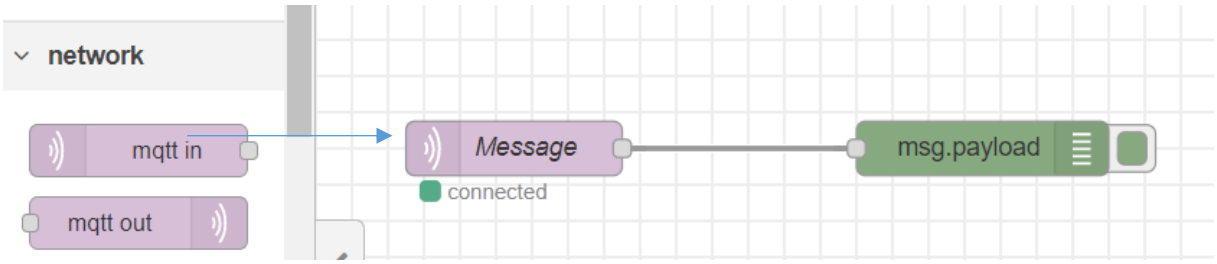


Si on veut récupérer les prévisions météo de la journée, on peut les trouver sur openweather

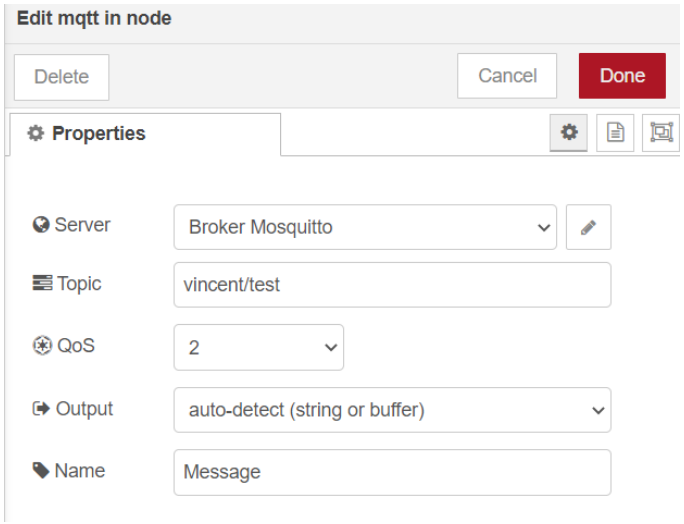


6 Test de MQTT et NODE-RED

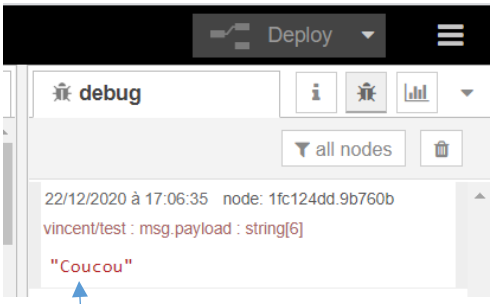
Sous Node-Red, réaliser l'architecture suivante :



Le paramétrage du message est :



Déployez, puis sélectionner l'onglet "Debug" en haut à droite de Node-Red



- Publiez ensuite un message depuis un terminal :

```
mosquitto_pub -h 192.168.1.45 -t vincent/test -m "Coucou"
```

- Votre message doit apparaître dans la fenêtre "Debug" de Node-Red

7 Créer un mot de passe pour nodered : 15 minutes dans la vidéo

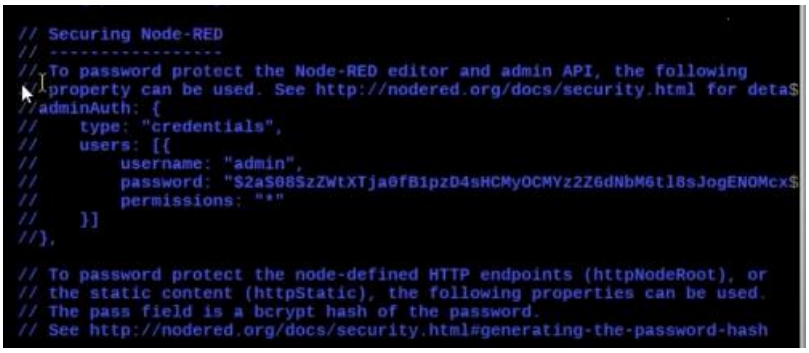
https://www.youtube.com/watch?v=9v3j-TTgG6M&t=104s

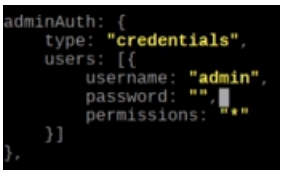
ou dans la vidéo https://www.youtube.com/watch?v=Jghd_nbY-KU

- stopper node-red

```
pi@raspberrypi:~ $ cd .node-red
pi@raspberrypi:~/.node-red $ ls
flows_raspberrypi_cred.json  lib           package.json       settings.js
flows_raspberrypi.json       node_modules  package-lock.json
```

- Editer settings.js
- Trouver le bloc "Securing Node-Red"
- Copier la section, enlever les commentaires.
- Supprimer le mot de passe





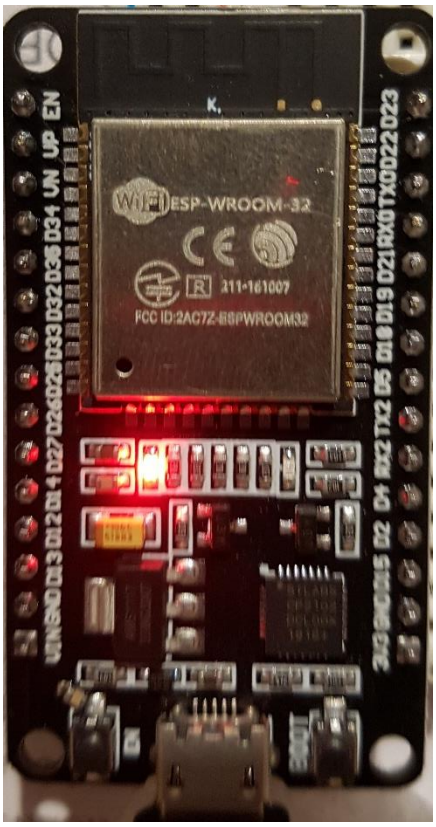
à priori pas nécessaire

- Ouvrir un autre terminal
  - installer node-red-admin avec la commande :
    - `sudo npm install -g node-red-admin`
  - Hacher le mot de passe :
    - `node-red-admin hash-pw`

    saisir le mot de passe ➔ il génère une clé que vous insérez dans le fichier settings.js
- Relancer node-red par la commande node-red-restart

8 Création de l'IHM avec Node-red : vidéo https://www.youtube.com/watch?v=N-Cko9s19uQ&t=533s

9 Réalisation d'un programme avec l'ESP-32 afin de publier des données issues d'un capteur



- Connecter votre ESP 8266 ou ESP32 à votre ordinateur
- Démarrer l'IDE Arduino
- Si ce n'est pas fait, installer le package ESP32 pour arduino

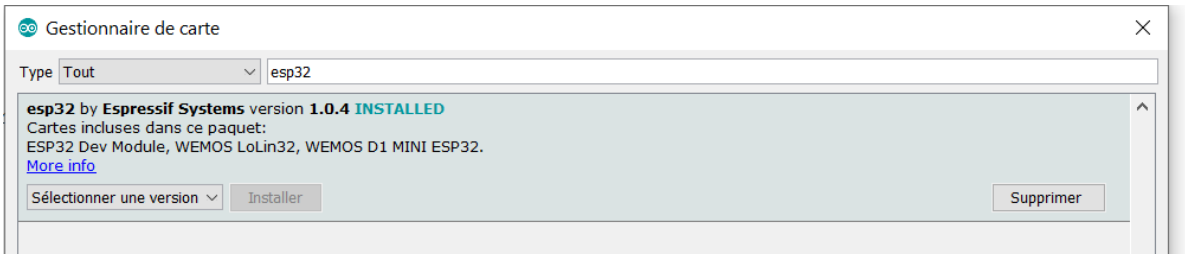
https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/

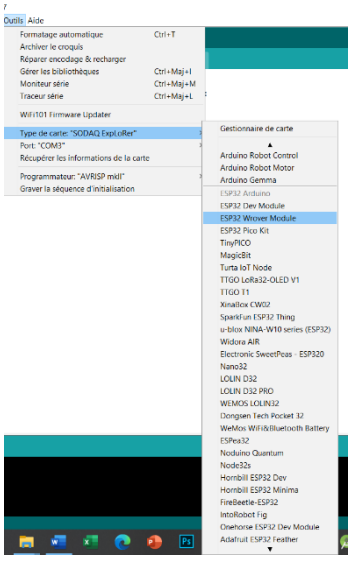
https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/

- ✓ Aller dans le menu "fichier-préférences"
- ✓ Ajouter l'URL :

https://dl.espressif.com/dl/package_esp32_index.json

- ✓ Ouvrir ensuite le gestionnaire de cartes et taper esp32 et l'installer





- ✓ Ensuite, votre carte devrait apparaître dans la liste des cartes. Choisir **"ESP32 Wrover Module"**
- ✓ installer la librairie "PubSubClient" https://github.com/knolleary/pubsubclient

- ✓ Dézipper la librairie.
- ✓ Renommer **pubsubclient-master** en **pubsubclient**
- ✓ Déplacer le répertoire **pubsubclient** dans le dossier **libraries** d'Arduino

Voici le code de test élémentaire

```
/*********************************************************************
  Programme créé par Vincent ROBERT d'après le tuto :
https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/
*********************************************************************/
#include <WiFi.h>
#include <PubSubClient.h>
#define LED 2 // Led intégrée au module ESP32

// Replace the next variables with your SSID/Password combination
const char* ssid = "*****************";
const char* password = "***************";

// Add your MQTT Broker IP address, example:
const char* mqtt_server = "192.168.1.45";

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;

/*************************************
 * SETUP
 *************************************/
void setup()
{
  Serial.begin(115200);
  pinMode(LED,OUTPUT);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);

}

/*************************************
 * Connexion au point d'accès WIFI
 *************************************/
void setup_wifi()
{
  delay(10);
  // 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.println("WiFi connected");
  Serial.println("IP address: ");  Serial.println(WiFi.localIP());
}

/*************************************************************
 * Fonction appelée dès qu'un message sur un topic auquel on a
 * souscrit est arrivé
 *************************************************************/
void callback(char* 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 += (char)message[i];
  }
```

```
  Serial.println();

  // Feel free to add more if statements to control more GPIOs with MQTT

  // Changes the output state according to the message
  if (String(topic) == "esp32/output")
  {
    Serial.print("Changing output to ");
    if(messageTemp == "on")
    {
      Serial.println("on");
      digitalWrite(LED, HIGH);
    }
    else if(messageTemp == "off"){
      Serial.println("off");
      digitalWrite(LED, LOW);
    }
  }
}

/*****************************************************
 * Tentative de reconnexion au broker MQTT
 *****************************************************/
void reconnect()
{
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // Subscribe
      client.subscribe("vincent/test");
      client.subscribe("esp32/output");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

/*********************************************
 * Boucle principale
 * On envoie un message toutes les 10 secondes
 *********************************************/
void loop()
{
  static unsigned i=1;
  String strMessage;
  char szMessage[20];

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

  long now = millis();
  if (now - lastMsg > 10000)
  {
    lastMsg = now;
    strMessage="coucou "+String(i);
    strMessage.toCharArray(szMessage,strMessage.length()+1); // conversion String en chaine C
    client.publish("vincent/test", szMessage);
    i++;

  }
}
```

Si MQTT est sécurisé par user et password, écrivez :

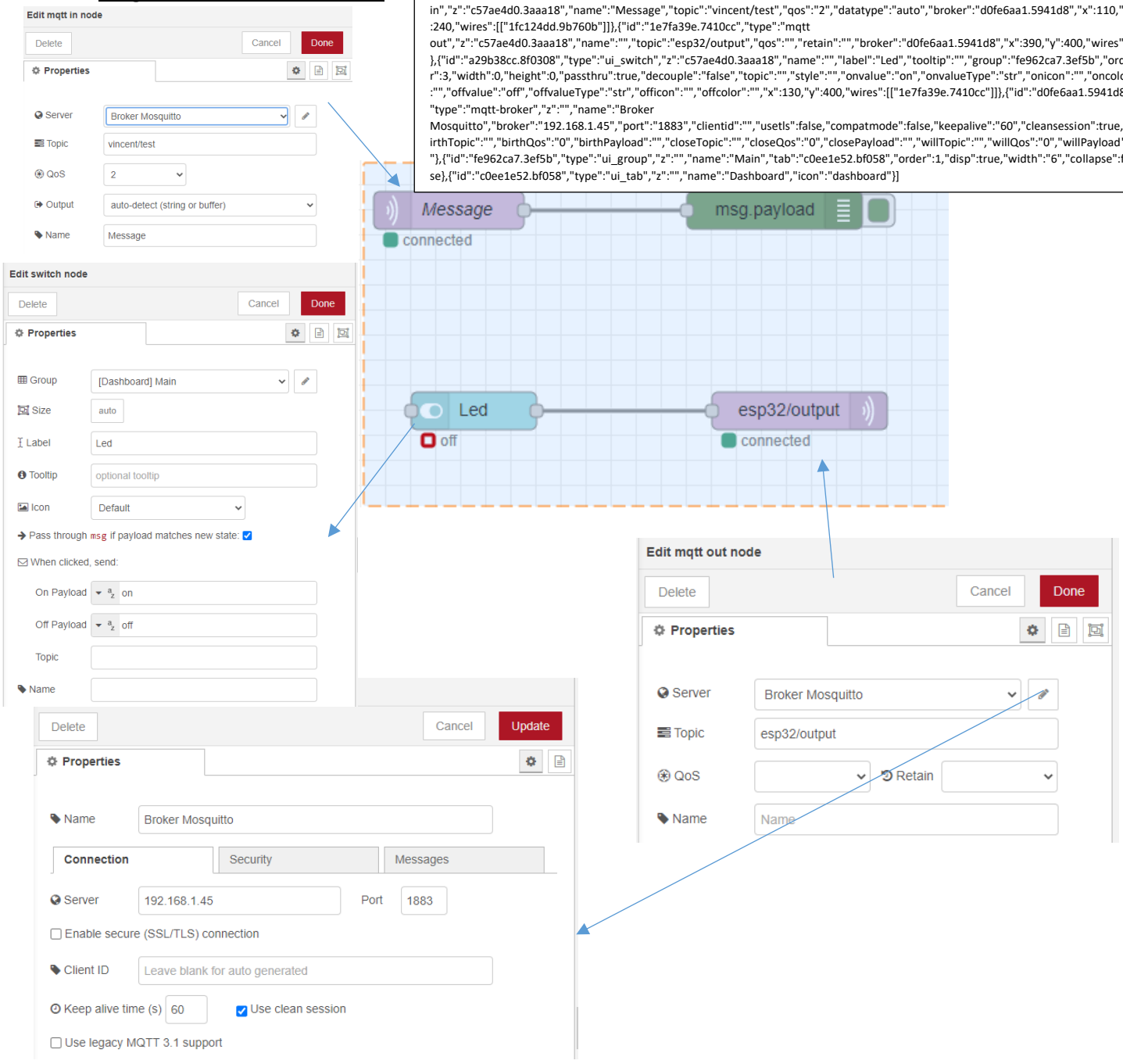
```
if (client.connect("ESP8266Client","user","password"))
```

Toutes les 5 secondes, ce programme envoie "coucou x" sur le topic "vincent/test". On peut le voir dans la console "Debug" de Node-Red.

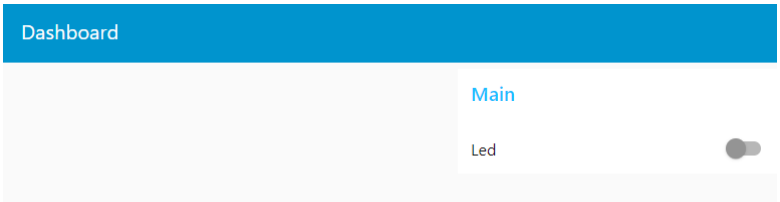
Maintenant, depuis la console, publiez le message "on" ou "off" sur le topic "esp32/output". Vous devriez voir apparaître le message dans le terminal de l'IDE Arduino et si vous connectez une led, vous pourrez l'allumer ou l'éteindre.

Programme Node-Red associé

[{"id":"c57ae4d0.3aaa18","type":"tab","label":"Flow 1","disabled":false,"info":""},{"id":"1fc124dd.9b760b","type":"debug","z":"c57ae4d0.3aaa18","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":370,"y":240,"wires":[]},{"id":"41223bca.b64fe4","type":"mqtt in","z":"c57ae4d0.3aaa18","name":"Message","topic":"vincent/test","qos":"2","datatype":"auto","broker":"d0fe6aa1.5941d8","x":110,"y":240,"wires":[["1fc124dd.9b760b"]]},{"id":"1e7fa39e.7410cc","type":"mqtt out","z":"c57ae4d0.3aaa18","name":"","topic":"esp32/output","qos":"","retain":"","broker":"d0fe6aa1.5941d8","x":390,"y":400,"wires":[]},{"id":"a29b38cc.8f0308","type":"ui_switch","z":"c57ae4d0.3aaa18","name":"","label":"Led","tooltip":"","group":"fe962ca7.3ef5b","order":3,"width":0,"height":0,"passthru":true,"decouple":"false","topic":"","style":"","onvalue":"on","onvalueType":"str","onicon":"","oncolor":"","offvalue":"off","offvalueType":"str","officon":"","offcolor":"","x":130,"y":400,"wires":[["1e7fa39e.7410cc"]]},{"id":"d0fe6aa1.5941d8","type":"mqtt-broker","z":"","name":"Broker Mosquitto","broker":"192.168.1.45","port":"1883","clientid":"","usetls":false,"compatmode":false,"keepalive":"60","cleansession":true,"birthTopic":"","birthQos":"0","birthPayload":"","closeTopic":"","closeQos":"0","closePayload":"","willTopic":"","willQos":"0","willPayload":""},{"id":"fe962ca7.3ef5b","type":"ui_group","z":"","name":"Main","tab":"c0ee1e52.bf058","order":1,"disp":true,"width":"6","collapse":false},{"id":"c0ee1e52.bf058","type":"ui_tab","z":"","name":"Dashboard","icon":"dashboard"}]

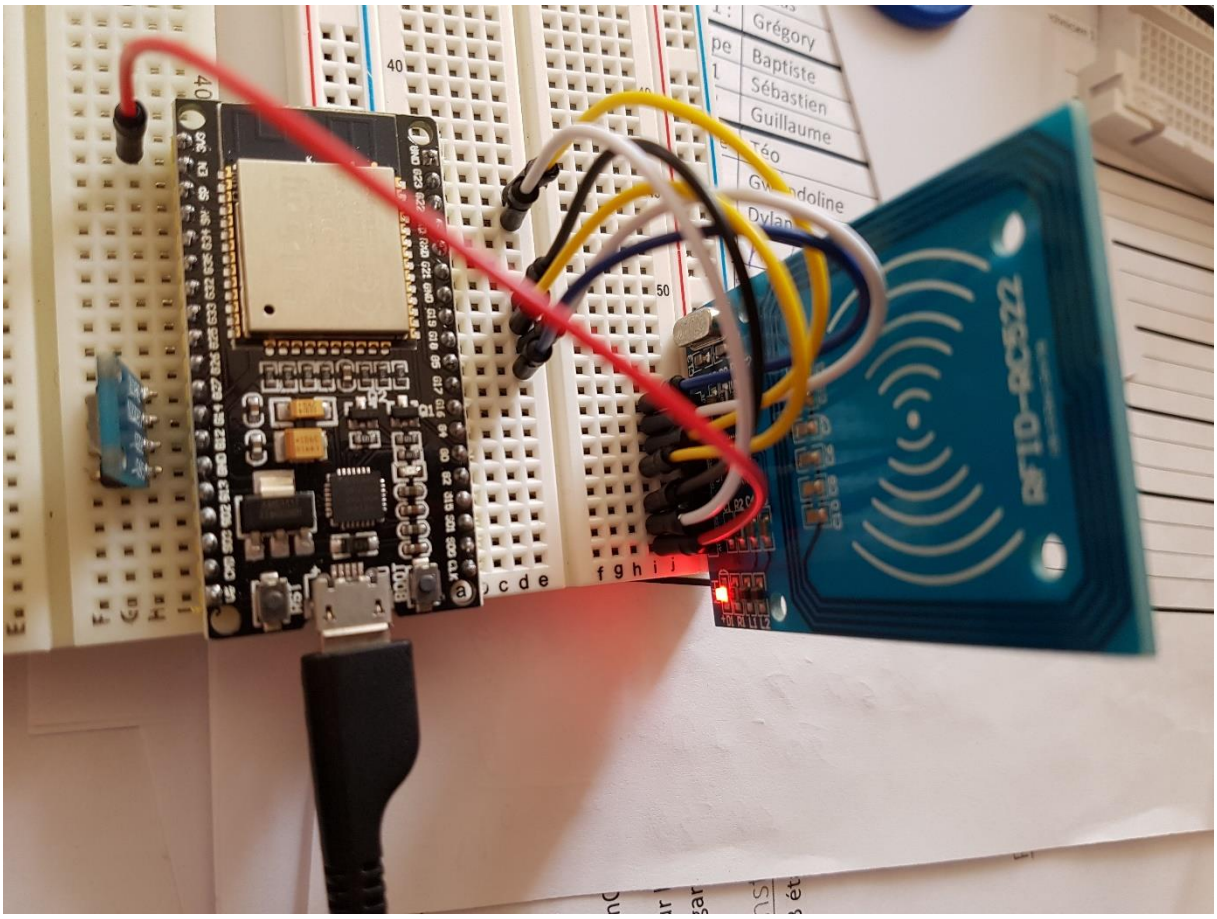


Pour lancer le dashboard, il faut écrire : http://192.168.1.45:1880/ui



10 Programme ESP32 avec fonction callback

badge passé :	ESP32 → MQTT	topic : rfid	message : code carte en hexa
Accusé de réception	MQTT → ESP32	topic : rfid/ar	si message ="ok", on allume led verte sinon on allume led rouge.




```
#include <WiFi.h>
#include <PubSubClient.h>
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN         22          // Configurable, see typical pin layout above
#define SS_PIN          5         // Configurable, see typical pin layout above
#define DEL_ROUGE       12
#define DEL_VERTE       14
#define DEL_BLEUE       27

// Replace the next variables with your SSID/Password combination
const char* ssid = "*****************";
const char* password = "******************";

// Add your MQTT Broker IP address, example:
//const char* mqtt_server = "192.168.1.144";
const char* mqtt_server = "192.168.1.45";

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;
```

```
MFRC522 rfid(SS_PIN, RST_PIN); // Instance of the class

MFRC522::MIFARE_Key key;

// Init array that will store new NUID
byte nuidPICC[4];

//=================================================
void allume(int rouge,int vert, int bleu, int duree)
{
  digitalWrite(DEL_ROUGE,rouge);
  digitalWrite(DEL_VERTE,vert);
  digitalWrite(DEL_BLEUE,bleu);
  delay(duree);
  digitalWrite(DEL_ROUGE,LOW);
  digitalWrite(DEL_BLEUE,LOW);
  digitalWrite(DEL_VERTE,LOW);
}

//=================================================
void setup_wifi() {
  delay(10);
  // 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.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

//=================================================
void setup()
{
  Serial.begin(115200);
  SPI.begin(); // Init SPI bus
  rfid.PCD_Init(); // Init MFRC522

  // Init des E/S TOR
  pinMode(DEL_ROUGE,OUTPUT);
  pinMode(DEL_VERTE,OUTPUT);
  pinMode(DEL_BLEUE,OUTPUT);

  allume(HIGH,LOW,LOW,2000);


  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback); // Fonction qui reçoit les messages MQTT
}
//=================================================================
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // Subscribe
      client.subscribe("rfid");
      client.subscribe("rfid/ar");
    } else {
```

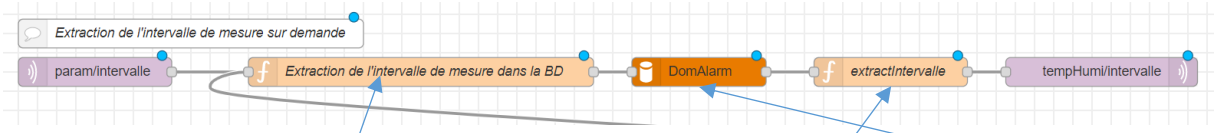
```
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

//=======================================================
void callback(char* topic, byte* message, unsigned int length) {
  Serial.print("Message MQTT : ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

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

  // Si un message est reçu avec le topi rfid/ok et si le
  // message correspond au dernier badge passé, on allume la led en
  // vert
   if (String(topic) == "rfid/ar")
   {
      if (String(messageTemp)=="ok")
          allume(LOW,HIGH,LOW,5000);  // vert
      else
          allume(HIGH,LOW,LOW,5000); // rouge
   }
 }

//================================================
void loop()
{
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  readRFID();
}

//================================================
void readRFID()
{
  // Reset the loop if no new card present on the sensor/reader. This saves the entire process
when idle.
  if ( ! rfid.PICC_IsNewCardPresent())
    return;

  // Verify if the NUID has been readed
  if ( ! rfid.PICC_ReadCardSerial())
    return;


  // Store NUID into nuidPICC array
  String strRfid;
  for (byte i = 0; i < 4; i++)
  {
    nuidPICC[i] = rfid.uid.uidByte[i];
    strRfid += String(nuidPICC[i],HEX);
  }

  char szRfid[8+1];
  strRfid.toCharArray(szRfid,9);
  client.publish("rfid",szRfid);

  Serial.println(F("The NUID tag is:"));
  Serial.print(F("In hex: "));
```

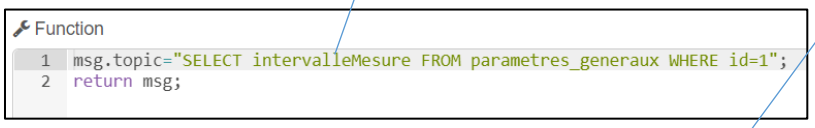
```
  printHex(rfid.uid.uidByte, rfid.uid.size);
  Serial.println();
  Serial.print(F("In dec: "));
  printDec(rfid.uid.uidByte, rfid.uid.size);
  Serial.println();
  allume(LOW,LOW, HIGH,2000);

  // Halt PICC
  rfid.PICC_HaltA();

  // Stop encryption on PCD
  rfid.PCD_StopCrypto1();
}

/**=================================================================
 * Helper routine to dump a byte array as hex values to Serial.
 =================================================================*/
void printHex(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], HEX);
  }
}

/**=================================================================
 * Helper routine to dump a byte array as dec values to Serial.
 =================================================================*/
void printDec(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], DEC);
  }
}
```

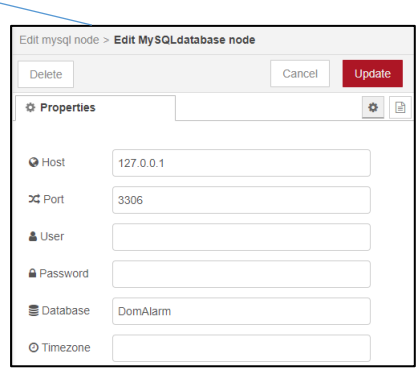
11 Connexion avec une base de données mySql

- Installer le plugin "node-red-node-mysql"

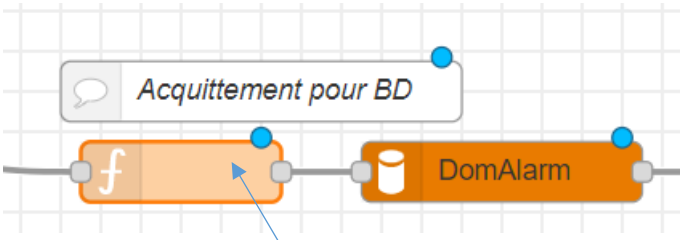
11.1 Exemple 1 : extraction de l'intervalle de mesure dans la base de données







11.2 Exemple 2 : enregistrement en base de données




```
var dt=new Date();


// Je formate la date pour la rendre compatible avec MySql

var strDateHeure=dt.getFullYear().toString()+"-"+(dt.getMonth()+1).toString()+"-
"+dt.getDate().toString()+"
"+dt.getHours().toString()+":"+dt.getMinutes().toString()+":"+dt.getSeconds().toString();


// Je crée la requete SQL

msg.topic="INSERT INTO `acquittements_rfid`(`dateHeure`) VALUES (\""+strDateHeure+"\")";


return msg;
```