

Dossier de fabrication IOT S3

Capteurs / ESP 32 et Raspberry

Réalisation :

Batard Jean-Baptiste

Maurize Lucas

Bouffart Victor

Groupe B1 GEII 2^{ère} année

IUT de l'Aisne, Cuffies, 2022

- I) Matériel nécessaire
- II) Partie Raspberry
- III) Partie ESP 32 et capteurs

- I) Matériel nécessaire

- Raspberry Pi
- Carte SD
- Adaptateur Carte SD vers USB
- Cordon d'alimentation USB-C
- Câble HDMI vers micro HDMI
- Câble micro-USB
- PC
- Clavier
- Souris
- Ecran
- ESP 32
- Capteurs : LDR / DHT22 / PIR / LM35 / MQ135
- Assistant d'installation linux pour Raspberry : Raspberry PI Imager
- Logiciel Arduino
- Camera avec carte wifi tapo
- lampe connecté tapo

Notre projet commun porte sur la réalisation d'un système automatisé et connecté, permettant le suivi des conditions à l'intérieur d'une pièce ainsi que le suivi de l'activité d'une lampe par une caméra . Il comprend l'utilisation de différents capteurs (luminosité, taux de CO₂, température, humidité et présence) et des actionneurs tels qu'une lampe et une caméra. Il se divise en trois parties : une électronique et programmation sur Arduino, une deuxième informatique sur un système d'exploitation Linux et des différents actionneurs et une troisième avec le lien entre la raspberry et l'utilisateur.

La partie sur Arduino concerne la récupération de toutes les données des capteurs, l'envoi de ces informations sur l'appareil Linux en MQTT.

La partie Linux installée sur une Raspberry Pi récupère, stocke et diffuse ces informations ainsi que celle des objets connectés sur internet pour les rendre accessibles à l'utilisateur.

Cet ensemble permet alors à un utilisateur de connaître en temps réel la température, l'humidité, la luminosité et la présence ou non d'un individu dans

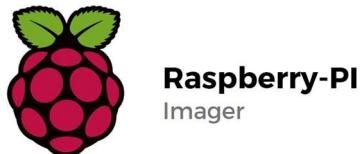
une pièce de la maison et la situation sur place via la caméra. Mais aussi de pouvoir actionner la lampe depuis n'importe quelle liaisons internet.

II) Partie Raspberry

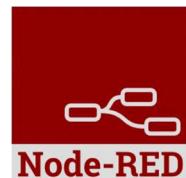
Déroulement de la réalisation :

Points critiques :

- 1) Installation de LINUX sur la Raspberry Pi (OS Ubuntu)



- 2) Installation et paramétrage du service MQTT Mosquitto



- 3) Installation de Node-RED

Node-RED

- 4) Programmation en « bloc » sur Node-RED

- 5) Création de l'affichage (ui) prévu pour smartphone(application)

- 6) utilisation de motion eye



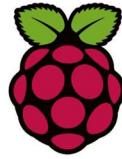
- 7) Installation et paramétrage ngrok
(Pont internet pour un accès à distance)

ngrok

ou openVPN

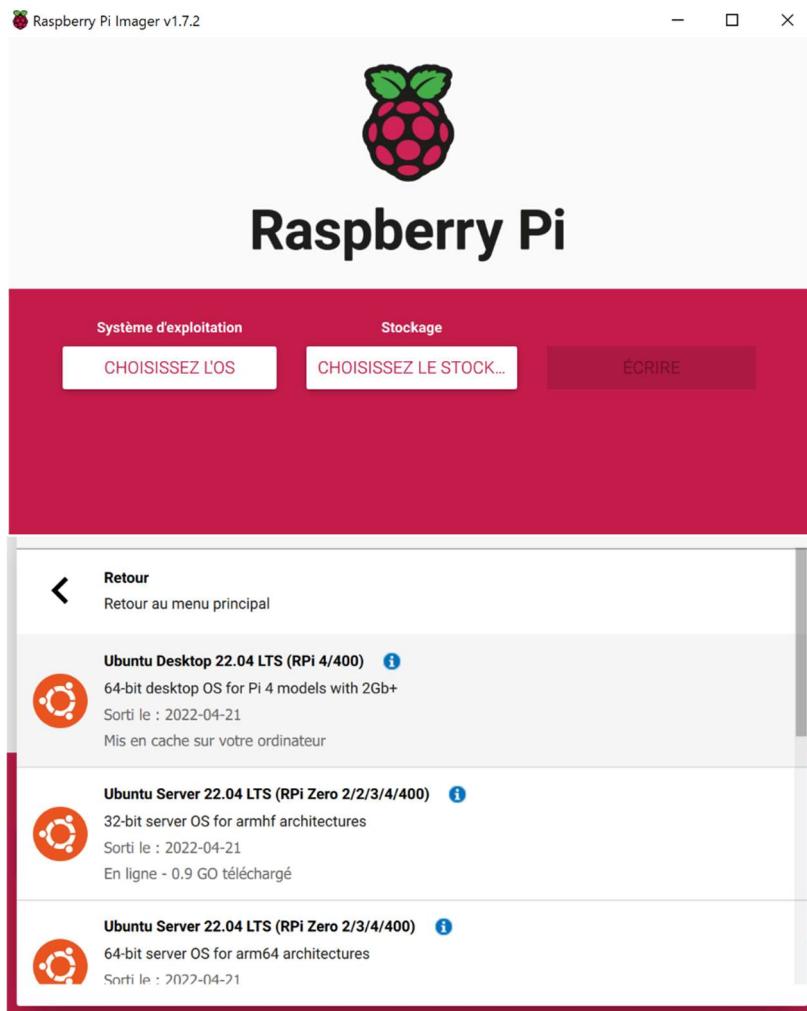
Réalisation

1) Installation de LINUX sur la Raspberry Pi (OS Ubuntu)



Raspberry-Pi
Imager

Pour installer LINUX sur la Raspberry nous avons utilisé un logiciel dédié nommé Raspberry Pi imager, puis avec celui-ci, nous avons formaté une carte SD avec l'OS LINUX UBUNTU.



Une fois la carte SD formatée avec l'OS LINUX, il faut l'insérer dans la Raspberry Pi préalablement reliée à un écran en HDMI et à deux périphériques clavier et souris.

En démarrant la Raspberry on tombe alors sur l'assistant d'installation linux où l'on choisit les paramètres voulu pour l'OS. Il faut donc sélectionner ce que l'on souhaite et choisir d'installer Ubuntu.

Installation terminée, on passe à Mosquitto.



2) Installation et paramétrage du service MQTT Mosquitto

L'installation de Mosquitto se fait à l'aide du terminal Linux. Ouvrir donc le terminal puis saisir les commandes suivantes.

Commandes préliminaires :

sudo apt update

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo apt update
[sudo] password for lenovo:
Hit:1 http://fr.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://fr.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://fr.archive.ubuntu.com/ubuntu focal-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Fetched 114 kB in 1s (144 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
154 packages can be upgraded. Run 'apt list --upgradable' to see them.
lenovo@lenovo-ThinkCentre-M72e:~$ apt list --upgradable
Listing... Done
```

Puis

sudo apt upgrade

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1
  libgstreamer-plugins-bad1.0-0 libva-wayland2 linux-headers-5.11.0-27-generic
  linux-hwe-5.11-headers-5.11.0-27 linux-image-5.11.0-27-generic
  linux-modules-5.11.0-27-generic linux-modules-extra-5.11.0-27-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  libfwupdplugins5
The following packages will be upgraded:
  alsamixer-conf apt apt-utils base-files bind9-host bluez bluez-cups
```

Pour mettre à jour les outils de Linux

Lorsqu'il est demandé « voulez-vous continuer » appuyer sur « Y » ou « O » pour poursuivre

Commandes d'installation Mosquitto:

sudo apt install -y mosquitto mosquitto-clients

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo apt install -y mosquitto mosquitto-clients
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1
  libgstreamer-plugins-bad1.0-0 libva-wayland2 linux-headers-5.11.0-27-generic
  linux-hwe-5.11-headers-5.11.0-27 linux-image-5.11.0-27-generic
  linux-modules-5.11.0-27-generic linux-modules-extra-5.11.0-27-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libdlt2 libev4 libmosquitto1 libwebsockets15
The following NEW packages will be installed:
  libdlt2 libev4 libmosquitto1 libwebsockets15 mosquitto mosquitto-clients
0 upgraded, 6 newly installed, 0 to remove and 0 not upgraded.
```

sudo systemctl enable mosquitto.service

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo systemctl enable mosquitto.service
Synchronizing state of mosquitto.service with SysV service script with /lib/systemd/
/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable mosquitto
lenovo@lenovo-ThinkCentre-M72e:~$
```

systemctl status mosquitto

```
lenovo@lenovo-ThinkCentre-M72e:~$ systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker
   Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-06-09 21:08:52 CEST; 1min 31s ago
     Docs: man:mosquitto.conf(5)
           man:mosquitto(8)
 Main PID: 38524 (mosquitto)
    Tasks: 3 (limit: 4419)
   Memory: 1.2M
      CPU: 0.000 CPU(s) (idle)
     CGroup: /system.slice/mosquitto.service
             └─38524 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

juin 09 21:08:51 lenovo-ThinkCentre-M72e systemd[1]: Starting Mosquitto MQTT v3.1/>
juin 09 21:08:52 lenovo-ThinkCentre-M72e mosquitto[38524]: [ 991.167213]~DLT~38524>
juin 09 21:08:52 lenovo-ThinkCentre-M72e systemd[1]: Started Mosquitto MQTT v3.1/v3.1.1 Broker.
lines 1-14/14 (END)
```

sudo reboot

Pour redémarrer la Raspberry

Puis aller dans le fichier .conf en saisissant la commande suivante.

sudo nano /etc/mosquitto/mosquitto.conf

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo nano /etc/mosquitto/mosquitto.conf
```

Ajouter :

listener 1883

allow_anonymous true

```
lenovo@lenovo-ThinkCentre-M72e: ~
GNU nano 4.8          /etc/mosquitto/mosquitto.conf      Modified
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

pid_file /var/run/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

listener 1883

allow_anonymous true

File Name to Write: /etc/mosquitto/mosquitto.conf
^G Get Help      M-D DOS Format    M-A Append      M-B Backup File
^C Cancel       M-M Mac Format    M-P Prepend     ^T To Files
```

Une fois fait appuyer sur CTRL-X pour quitter puis « Y » ou « O » pour valider et Entrer.

systemctl restart mosquitto

Si une erreur apparaît alors le problème se trouve dans le code dans le .conf. Vérifier alors le code.

systemctl status mosquitto

```
lenovo@lenovo-ThinkCentre-M72e:~$ systemctl restart mosquitto
lenovo@lenovo-ThinkCentre-M72e:~$ systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker
   Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor pre>
   Active: active (running) since Thu 2022-06-09 21:20:04 CEST; 6s ago
     Docs: man:mosquitto.conf(5)
           man:mosquitto(8)
     Main PID: 2216 (mosquitto)
        Tasks: 3 (limit: 4413)
       Memory: 1.0M
      CGroup: /system.slice/mosquitto.service
              └─2216 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

juin 09 21:20:04 lenovo-ThinkCentre-M72e systemd[1]: Starting Mosquitto MQTT v3>
juin 09 21:20:04 lenovo-ThinkCentre-M72e mosquitto[2216]: [ 480.665620]-DLT~ 2>
juin 09 21:20:04 lenovo-ThinkCentre-M72e systemd[1]: Started Mosquitto MQTT v3.>
```

S'assurer que le serveur MQTT est bien « ACTIVE », si oui on passe au test du serveur MQTT.

Test de Mosquitto

D'abord installer l'outil net-tools

```
sudo apt install net-tools
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo apt install net-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1
  libgstreamer-plugins-bad1.0-0 libva-wayland2 linux-headers-5.11.0-27-generic
  linux-hwe-5.11-headers-5.11.0-27 linux-image-5.11.0-27-generic
  linux-modules-5.11.0-27-generic linux-modules-extra-5.11.0-27-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 196 kB of archives.
After this operation, 864 kB of additional disk space will be used.
```

Puis

```
Ifconfig
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
  inet 192.168.1.4 brd 192.168.1.255 netmask 255.255.255.0
    inet6 fe80::c842:d135:e6fa:dcfb brd ff02::1 scopeid 0x20<link>
      ether 00:23:24:3a:04:97 txqueuelen 1000 (Ethernet)
        RX packets 4593 bytes 2676307 (2.6 MB)
        RX errors 0 dropped 815 overruns 0 frame 0
        TX packets 3335 bytes 673202 (673.2 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
  inet 127.0.0.1 brd 127.0.0.0 netmask 255.0.0.0
    inet6 ::1 brd ff00::1 scopeid 0x10<host>
      loop txqueuelen 1000 (Local Loopback)
        RX packets 903 bytes 92690 (92.6 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 903 bytes 92690 (92.6 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Récupérer l'IP du poste ex : 192.168.1.4.

Envoyer et recevoir sur Mosquitto (ligne de commande)

Sub Lorsqu'on veut recevoir une valeur

Pub Lorsqu'on veut envoyer une valeur

-h Adresse IP à laquelle on envoie la valeur

-t Topic sur lequel on envoie la valeur

-m Message ou valeur que l'on envoie

L'adresse IP s'écrit normalement contrairement au topic entre guillemet ainsi qu'au message.

Ouvrir deux pages de terminal distinctes, sur l'une d'entre elle entrer :

```
mosquitto_sub -h 192.168.1.4 -t "test"
```

Sur l'autre :

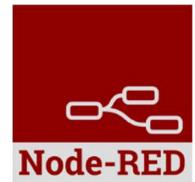
```
mosquitto_pub -h 192.168.1.4 -t "test" -m "boujour"
```

The screenshot shows two terminal windows side-by-side. The left window has the title bar 'lenovo@lenovo-ThinkCentre-M72e: ~'. It contains the command 'mosquitto_sub -h 192.168.1.4 -t "test"' followed by the message 'boujour' on a new line. The right window also has the title bar 'lenovo@lenovo-ThinkCentre-M72e: ~'. It contains the command 'mosquitto_pub -h 192.168.1.4 -t "test" -m "boujour"' followed by a new line. A small text box at the bottom left of the left terminal says 'mosquitto_sub -h 192.168 sur l'autre'.

Si tout va bien le message se transmet d'une fenêtre à l'autre. Pour quitter une fenêtre : CTRL-C

Tant que le « SUB » est présent, le topic est sur écoute. Si tout fonctionne comme prévu on passe à l'installation de Node-RED.

The screenshot shows two terminal windows side-by-side. The left window has the title bar 'lenovo@lenovo-ThinkCentre-M72e: ~'. It contains the command 'mosquitto_sub -h 192.168.1.4 -t "test"' followed by four 'boujour' messages on new lines. The right window also has the title bar 'lenovo@lenovo-ThinkCentre-M72e: ~'. It contains the command 'mosquitto_pub -h 192.168.1.4 -t "test" -m "boujour"' followed by eight 'boujour' messages on new lines. A small text box at the bottom left of the left terminal says 'mosquitto_sub -h 192.168 sur l'autre' and another box below it says 'mosquitto_pub -h 192.168 si tout va bien un le message'. A note at the bottom of the page says '^C'.



3) Installation de Node-RED

Avant Node-RED on installe nodejs, saisir dans le terminal les commandes suivantes :

Installation Node-RED

Installation de nodejs

```
sudo apt install nodejs
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo apt install nodejs
Reading package lists... Done
Building dependency tree
Reading state information... Done
nodejs is already the newest version (10.19.0~dfsg-3ubuntu1).
nodejs set to manually installed.
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1
  libgstreamer-plugins-bad1.0-0 libva-wayland2 linux-headers-5.11.0-27-generic
  linux-hwe-5.11-headers-5.11.0-27 linux-image-5.11.0-27-generic
  linux-modules-5.11.0-27-generic linux-modules-extra-5.11.0-27-generic
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
lenovo@lenovo-ThinkCentre-M72e:~$
```

Installation de l'outil npm

```
sudo apt install npm
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ sudo apt install npm
[sudo] password for lenovo:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1
  libgstreamer-plugins-bad1.0-0 libva-wayland2 linux-headers-5.11.0-27-generic
  linux-hwe-5.11-headers-5.11.0-27 linux-image-5.11.0-27-generic
  linux-modules-5.11.0-27-generic linux-modules-extra-5.11.0-27-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  binutils binutils-common binutils-x86_64-linux-gnu build-essential dpkg-dev
  fakeroot g++ g++-9 gcc gcc-9 gyp javascript-common libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libatomic1
  libbinutils libc-ares2 libc-dev-bin libc6-dev libcrypt-dev libctf-nobfd0
```

Installation de node red

```
sudo npm install -g --unsafe-perm node-red
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ npm install -g --unsafe-perm node-red
npm WARN deprecated multer@1.4.4: Multer 1.x is affected by CVE-2022-24434. This is fixed in v1.4.4-lts.1 which drops support for versions of Node.js before 6. Please upgrade to at least Node.js 6 and version 1.4.4-lts.1 of Multer. If you need support for older versions of Node.js, we are open to accepting patches that would fix the CVE on the main 1.x release line, whilst maintaining compatibility with Node.js 0.10.
[.....] / loadDep:vary: sill resolveWithNewModule vary@1.1.2 checkin
```

```
sudo npm install -g npm@8.10.0
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ npm install -g npm@8.10.0
npm WARN checkPermissions Missing write access to /usr/local/lib
npm WARN notsup Unsupported engine for npm@8.10.0: wanted: {"node":">=12.13.0 || ^14.15.0 || >=16"} (current: {"node":"10.19.0","npm":"6.14.4"})
npm WARN notsup Not compatible with your version of node/npm: npm@8.10.0

npm ERR! code EACCES
npm ERR! syscall access
npm ERR! path /usr/local/lib
npm ERR! errno -13
npm ERR! Error: EACCES: permission denied, access '/usr/local/lib'
npm ERR! { [Error: EACCES: permission denied, access '/usr/local/lib']
npm ERR!   stack:
npm ERR!     'Error: EACCES: permission denied, access \'/usr/local/lib\'',
npm ERR!     errno: -13,
npm ERR!     code: 'EACCES',
npm ERR!     syscall: 'access',
npm ERR!     path: '/usr/local/lib' }
```

Mise à jour de Node-RED. Une fois arrivé ici on doit être en mesure de lancer Node-RED. Saisir la commande suivante dans le terminal pour lancer l'application :

```
node-red
```

```
lenovo@lenovo-ThinkCentre-M72e:~$ node-red
9 Jun 22:10:26 - [info]

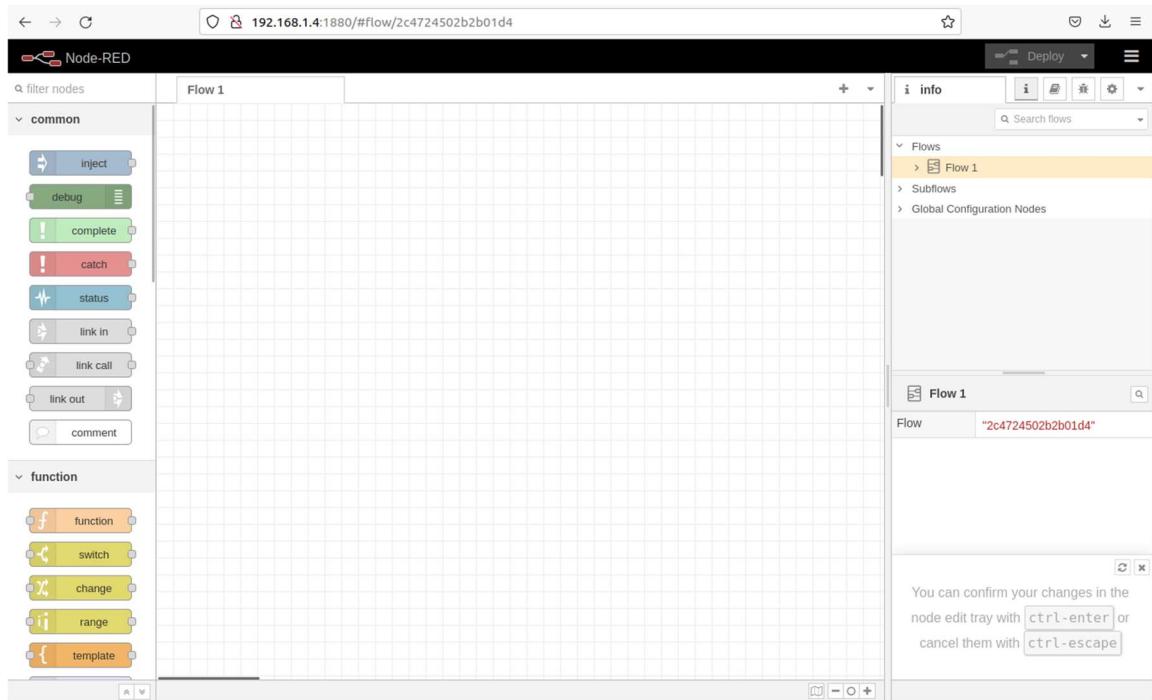
Welcome to Node-RED
=====
9 Jun 22:10:26 - [info] Node-RED version: v2.2.2
9 Jun 22:10:26 - [info] Node.js version: v10.19.0
9 Jun 22:10:26 - [info] Linux 5.13.0-48-generic x64 LE
9 Jun 22:10:27 - [info] Palette editor disabled : npm command not found
9 Jun 22:10:27 - [info] Loading palette nodes
9 Jun 22:10:28 - [info] Settings file : /home/lenovo/.node-red/settings.js
9 Jun 22:10:28 - [info] Context store : 'default' [module=memory]
9 Jun 22:10:28 - [info] User directory : /home/lenovo/.node-red
9 Jun 22:10:28 - [warn] Projects disabled : editorTheme.projects.enabled=false
9 Jun 22:10:28 - [info] Flows file : /home/lenovo/.node-red/flows.json
9 Jun 22:10:29 - [info] Creating new flow file
```

Pour arrêter Node-RED à tout moment appuyer sur CTRL-C.

4) Programmation en « bloc » sur Node-RED

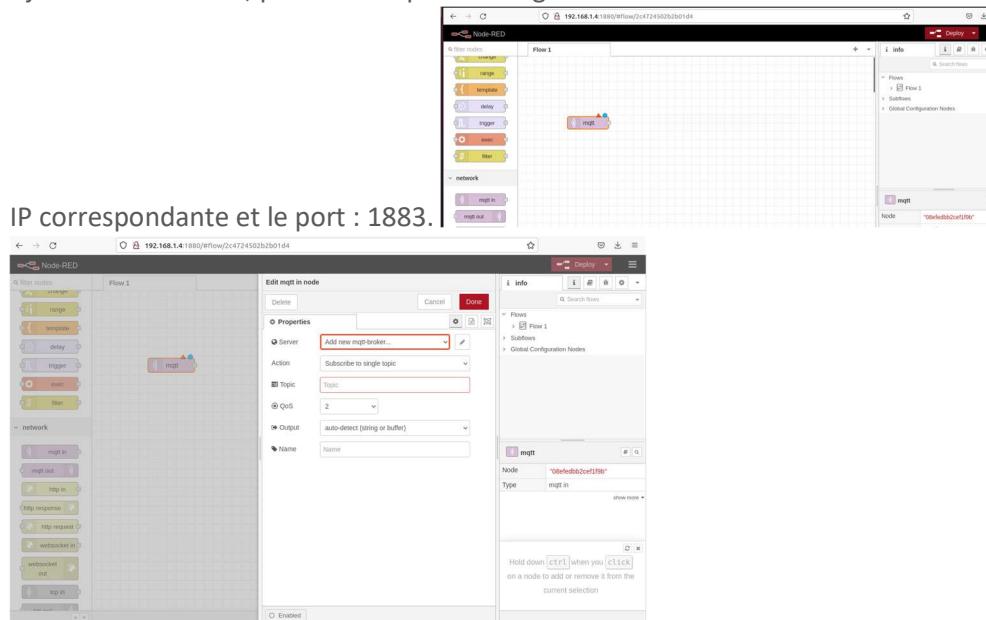
Configuration et programmation en « bloc »

Pour afficher Node-RED, lancer un navigateur et saisir par exemple : 192.168.1.4:1880, la forme IP :1880. Au lieu de l'IP, on peut saisir : 127.0.0.1 :1880. Si tout fonctionne, la page de configuration Node-RED s'affiche. Une fois la page sur le navigateur on peut commencer la programmation par bloc.

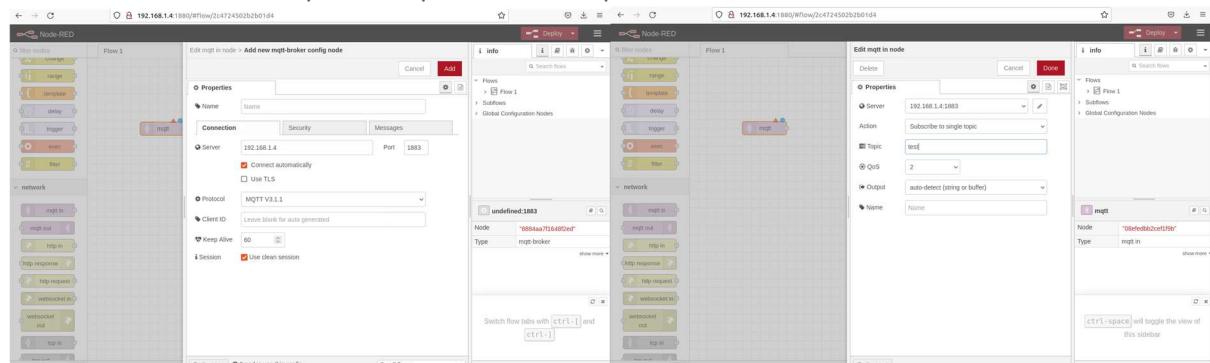


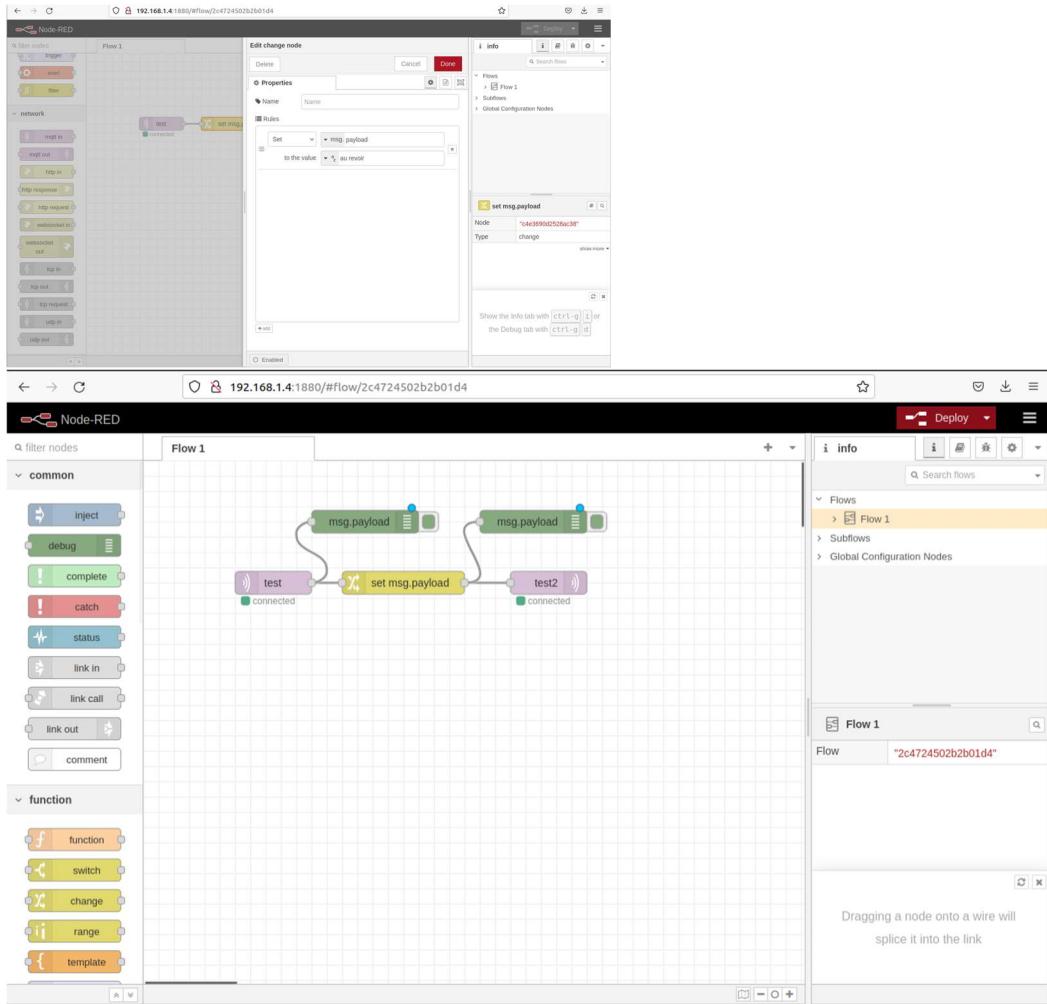
Petit exemple :

Ajouter un MQTT in, puis dans le paramétrage du bloc : « add new MQTT broker » et entrer l'adresse IP correspondante et le port : 1883.

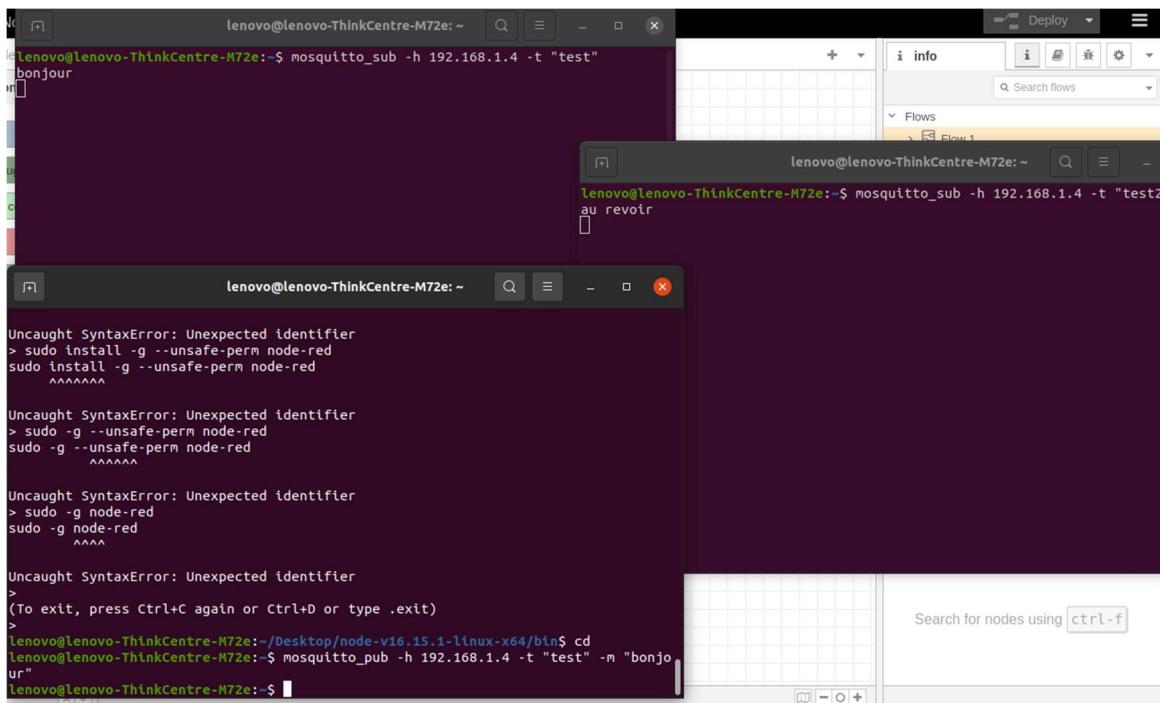


Enfin saisir un nom de topic sur lequel on va faire passer les informations.





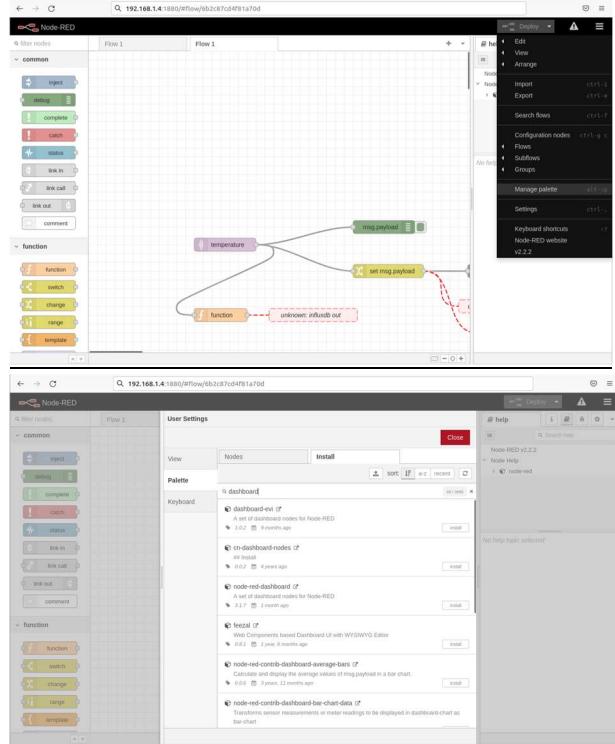
Relier l'élément à un « Change » et ajouter un bloc « Debug » en parallèle.



Enfin si on teste en envoyant un pub au serveur sur le topic « test » on reçoit bien l'information et le serveur revoie bien sur le topic « test2 » l'information qu'on lui a demandé de renvoyer.

Ajout des bibliothèques

Trois tirets puis manage palette



Dans **install** puis

→ **dashboard-evi**

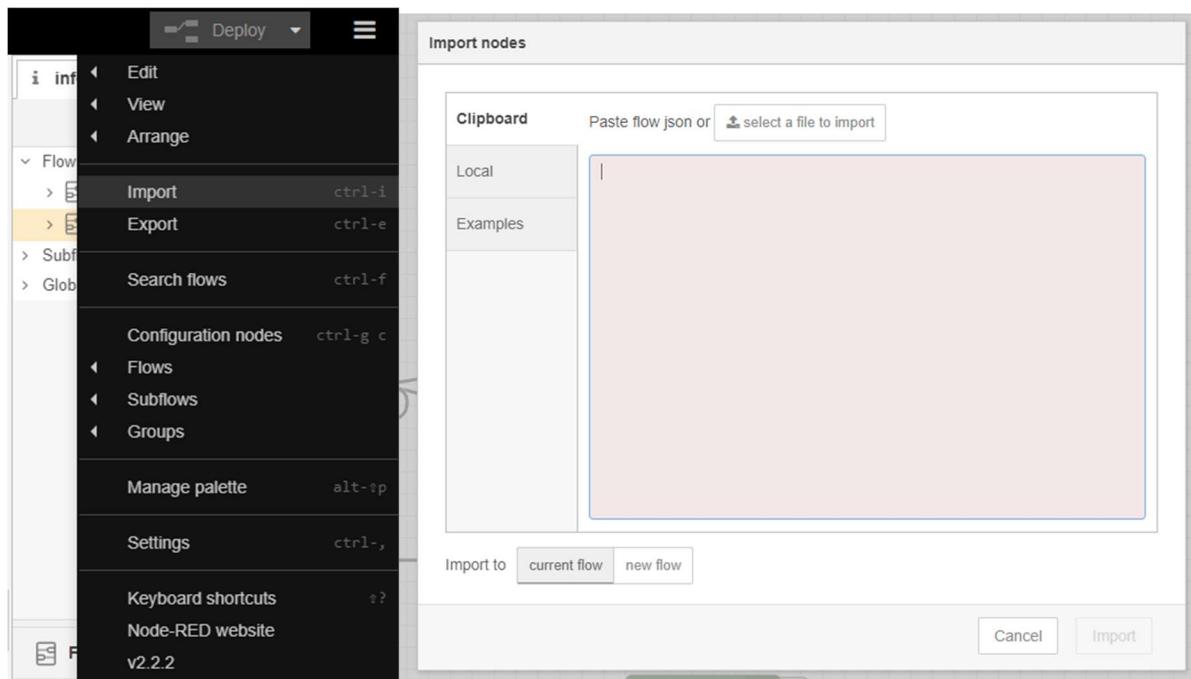
→ **node-red-contrib-ui-led**

Enfin

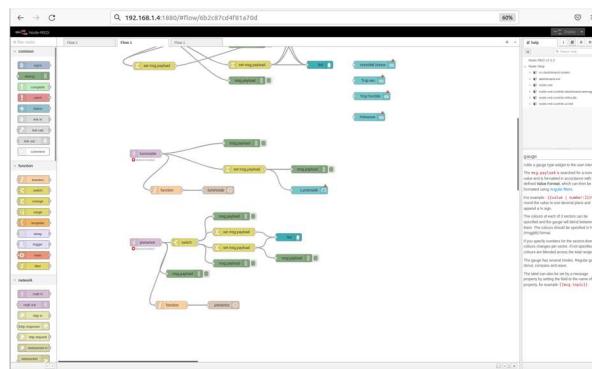
→ **node-red-contrib-influxdb** (pour la future base de données)

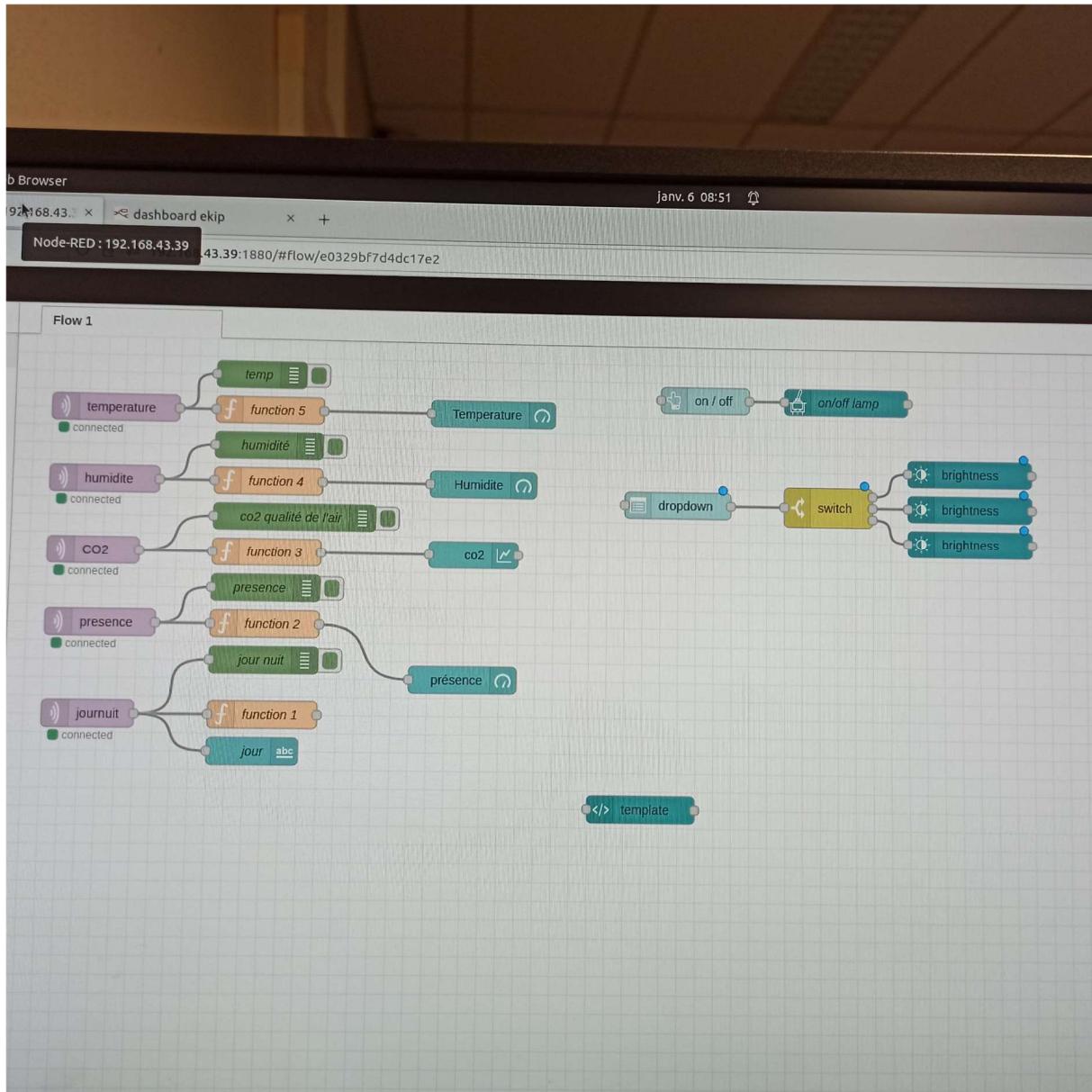
Lorsque toutes les bases de données ont été ajoutées, on peut alors saisir le programme en format .json.

Trois petits traits puis « import ».



Coller ensuite le code fournit en annexe et le coller dans la zone de texte ci-contre. Le résultat final est le suivant (penser à modifier l'IP).





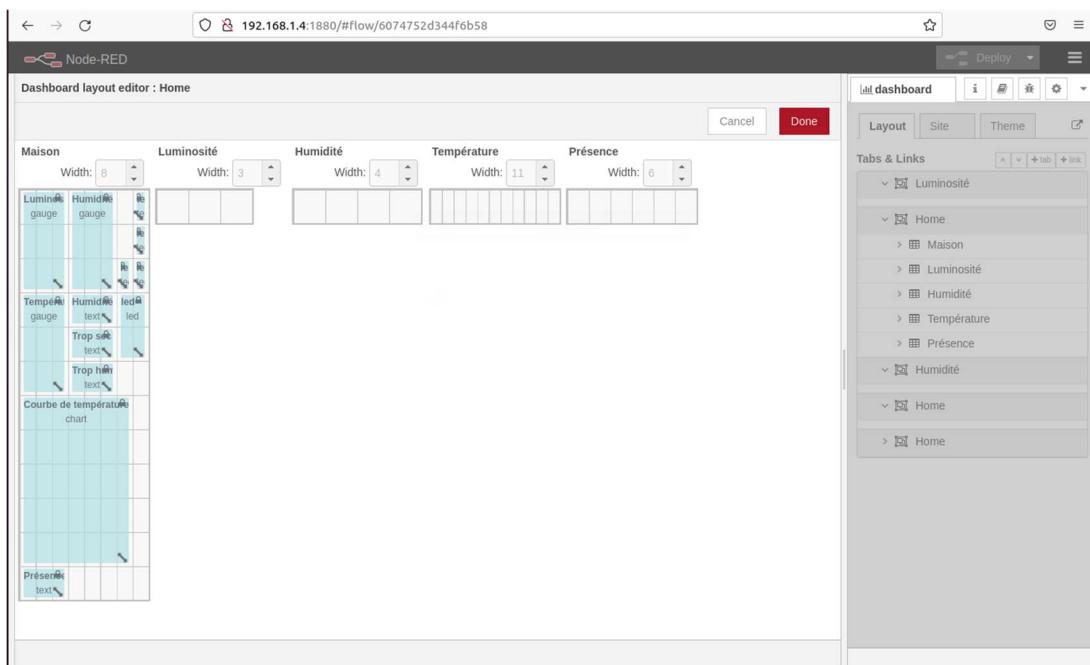
5) Création de l'affichage (ui) prévu pour smartphone

Pour accéder à l'affichage / Dashboard :

IP :1880/ui

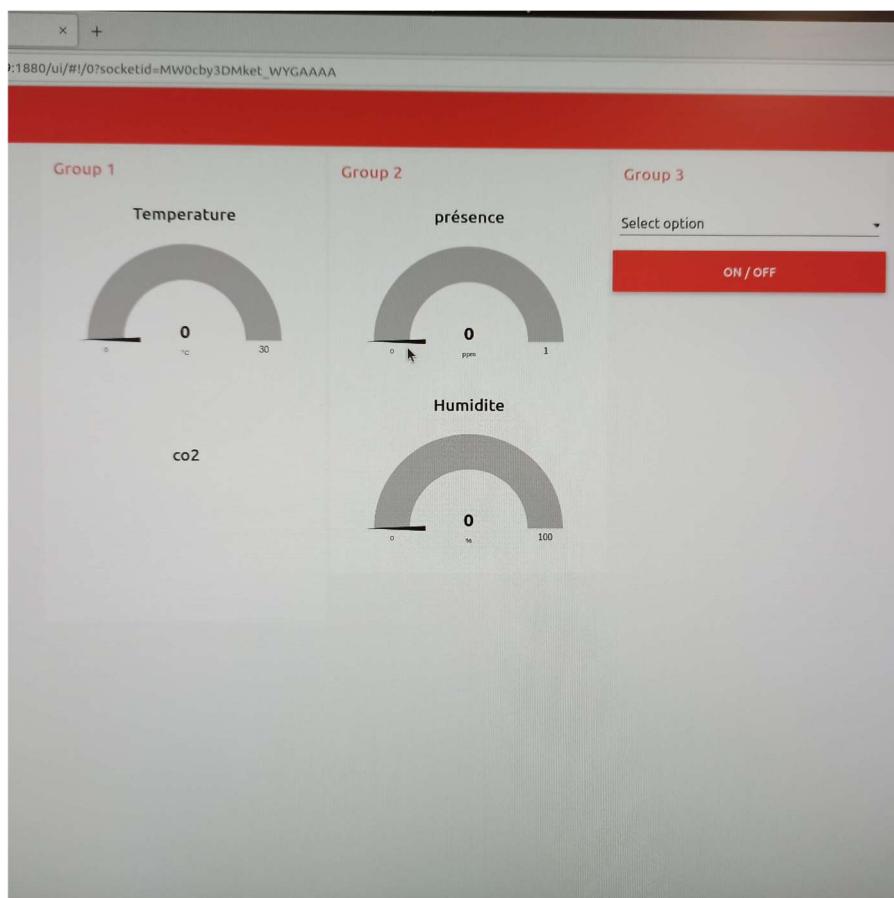
Exemple : 192.168.1.4:1880/ui

Pour le paramétrage aller dans : « Flèche vers le bas » en haut à droite puis « Dashboard », puis « Home » et « Layout » une fenêtre s'affiche. Dans cette fenêtre on peut disposer les éléments dans la configuration voulue.



Sans modifications (sur le .json fournit) on obtient cela :

Résultat final



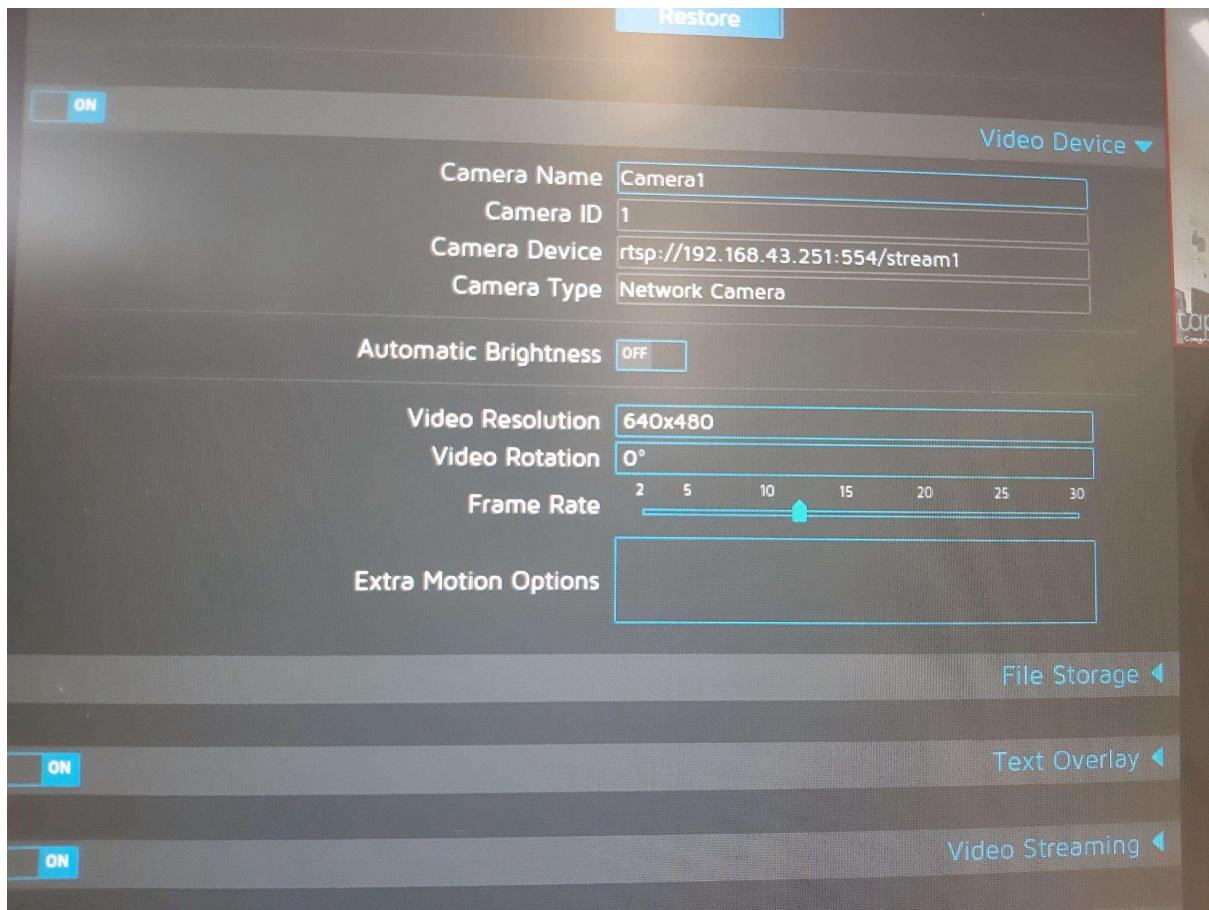
6) utilisation de motion eye (réception stream caméra)

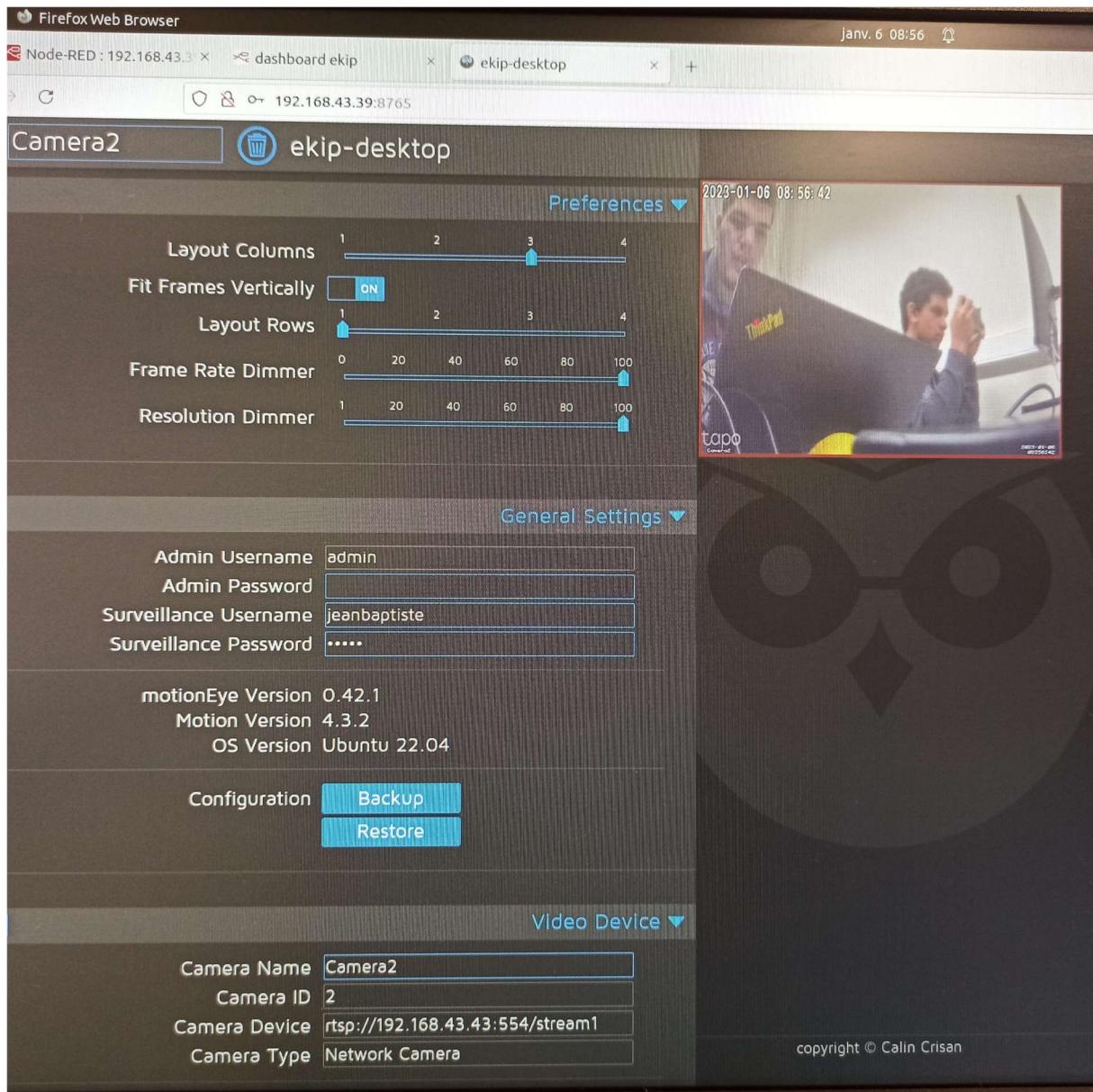
durant cette partie nous avons rencontré un grand nombre de problèmes au niveau du système à utiliser pour l'obtention du flux video de la camera (vlc,domoticz, spyeye,...) il s'est avéré qu'un grand nombre d'entre eux nécessite des installation de systeme spéciaux ou bien même de différent OS ,motion eye offrait une option simple si nous parvenions à l'installer sur notre raspberry.

Voici le lien utiliser pour cette installation : <https://github.com/motioneye-project/motioneye/wiki/%28Install-On-Ubuntu-%2820.04-or-Newer%29>

les problème de cette installation étant un python incompatible avec la version de motion eye et requérant donc la suppression de la version de python présente pour celle demandé.

après l'avoir téléchargé nous avons créé un compte tptink pour la caméra puis la fenêtre ci-dessous s'est ouverte et nous avons entré les information de la caméra dans motion eye et celle du compte de la caméra pour lancer motion eye ecrire : systemctl start motioneye puis rejoindre le lien local ,ici : 192.168.43.39.8765 un mot de passe et un compte nous est demandé on rentre l'identifiant admin et pas de mot de passe





par la suite différentes extensions du dashboard node red ont été téléchargés afin de pouvoir utiliser la caméra et la lampe à travers la raspberry et node-red

(https://flows-nodered-org.translate.goog/node/node-red-contribtplink-tapo-connect-api?x_tr_sl=en&x_tr_tl=fr&x_tr_hl=fr&x_tr_pto=sc
node red pour la lamps
<https://flows.nodered.org/node/node-red-dashboard>)

Pour utiliser la lampe nous avons à l'instar de la caméra récupérer l'ip de la lampe et le compte tapo relié à celle ci pour la connecter directement avec les blocs du dashboard de node-red (ci-dessus).

7) Installation et paramétrage ngrok (Pont internet pour un accès à distance)

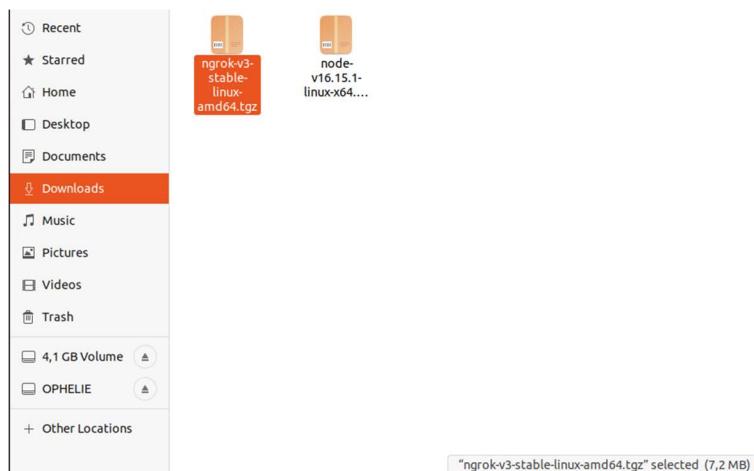


Aller sur le site ngrok, se créer un compte.

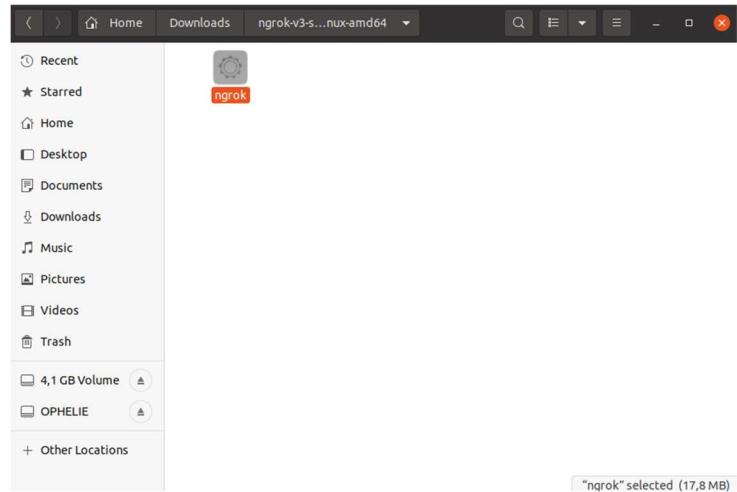
A screenshot of a web browser displaying the ngrok setup page at https://dashboard.ngrok.com/get-started/setup. The page has a dark header with the ngrok logo and a sidebar with navigation links like Getting Started, Setup & Installation, Your Authtoken, Cloud Edge, Tunnels, Events, API, Security, Team, Billing, and Settings. The main content area is titled "Download ngrok" and shows download links for Mac OS, Mac OS (ARM64), Windows (32-Bit), Linux (32-Bit), Linux (ARM), Linux (ARM64), FreeBSD, and FreeBSD (32-Bit). Below the download section, there are two numbered steps: "1. Unzip to install" with a terminal command "\$ unzip /path/to/ngrok.zip" and "2. Connect your account".

Récupérer l'authtoken.

Télécharger le fichier ngrok.zip



Extraire le fichier à l'emplacement actuel.



Déplacer l'exécutable sur le bureau puis saisir le chemin du fichier

```
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ ls
Desktop Downloads node_modules Pictures Templates
Documents Music package-lock.json Public Videos
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ cd Desktop
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ ls
ngrok node-v16.15.1-linux-x64
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$
```

```
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ ls
ngrok node-v16.15.1-linux-x64
```

Une fois l'exécutable trouvé saisir :

```
./ngrok config add-authtoken (authtoken perso)
```

```
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ ./ngrok config add-authtoken 29ZPyBdMA
```

Une validation va être donnée.

```
Authtoken saved to configuration file: /home/lenovo/.config/ngrok/ngrok.yml
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$
```

Enfin saisir :

```
./ngrok http 1880
```

```
lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ ./ngrok http 1880
```

Si tout est bien fait on tombe sur une page donnant le lien du site, lien qui ressemble à cela :
https://l'ip_deffinit.eu.ngrok.io

The terminal window shows the output of the command `ngrok`. It displays session status information including account details (Flavien, Plan: Free), version (3.0.4), region (Europe (eu)), and latency (15ms). It also shows a forwarding rule with the local port 4040 and the external URL `https://645b-176-146-91-164.eu.ngrok.io`.

```

lenovo@lenovo-ThinkCentre-M72e: ~/Desktop
ngrok
Hello World! https://ngrok.com/next-generation

Session Status          online
Account                Flavien (Plan: Free)
Version                3.0.4
Region                 Europe (eu)
Latency                15ms
Web Interface          Forwarding
Forwarding             http://127.0.0.1:4040
                        https://645b-176-146-91-164.eu.ngrok.io -> http://

Connections            ttl     opn      rt1      rt5      p50      p90
                        0       0       0.00    0.00    0.00    0.00

```


The Node-RED interface shows a flow with two parallel flows. Flow 2 is currently selected. The left sidebar contains common nodes like inject, debug, complete, catch, status, link in, link call, link out, and comment. The right sidebar shows the global configuration and the selected flow.

Mais ce lien amène vers la configuration du serveur, or nous voulons le Dashboard.

Pour faire cela on ajoute simplement `/ui` à la fin du lien tel que :

<https://645b-176-146-91-164.eu.ngrok.io/ui>

On tombe alors sur notre Dashboard

Ce lien étant accessible de n'importe quel appareil possédant un accès à internet.

Commandes Linux utiles

`./(nom du fichier)`

Permet d'exécuter un fichier lorsqu'on se situe dans le bon répertoire

`ls`



Permet de voir les fichiers se trouvant dans le dossier actuel

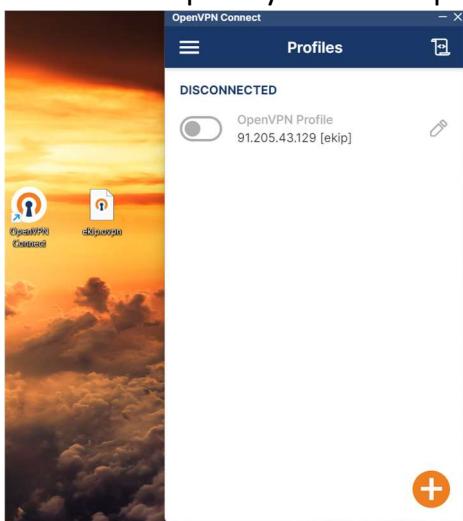
cd (*nom du dossier*)

Permet de se déplacer dans un dossier

Exemple

```
Desktop  Downloads  node_modules      Pictures  Templates
Documents  Music      package-lock.json  Public    Videos
Lenovo@lenovo-ThinkCentre-M72e:~$ cd
Lenovo@lenovo-ThinkCentre-M72e:~$ ls
Desktop  Downloads  node_modules      Pictures  Templates
Documents  Music      package-lock.json  Public    Videos
Lenovo@lenovo-ThinkCentre-M72e:~$ cd Desktop
Lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ ls
node-v16.15.1-linux-x64
Lenovo@lenovo-ThinkCentre-M72e:~/Desktop$ cd node-v16.15.1-linux-x64
Lenovo@lenovo-ThinkCentre-M72e:~/Desktop/node-v16.15.1-linux-x64$ ls
bin  CHANGELOG.md  include  lib  LICENSE  README.md  share
Lenovo@lenovo-ThinkCentre-M72e:~/Desktop/node-v16.15.1-linux-x64$ cd bin
Lenovo@lenovo-ThinkCentre-M72e:~/Desktop/node-v16.15.1-linux-x64/bin$ ls
corepack  node  npm  npx
```

Premièrement l'option de Ngrok semblait être une bonne solution or il nous est nécessaire de posséder au moins 2 tunnels pour la connexion du dashboard et la connexion de la caméra avec motion eye en ligne et Ngrok n'offrait qu'un seul tunnel et l'autotoken empêchait l'utilisation de 2 comptes pour plusieurs tunnels sur la même raspberry ,la seule solution viable qui a pu être trouvée est l'utilisation d'OpenVPN permettant plusieurs tunnels dans un même appareil (video utiliser pour son installation https://youtu.be/80owhV_5jYA) mais le problème de son utilisation à travers la raspberry nous a empêchés de parvenir à créer plusieurs tunnels.



Ici on peut voir l'installation d' OpenVPN et le fichier de notre Raspberry ainsi que le switch d'activation (ici désactivé) du tunnel .

partie application :

Nous avons créé une application avec Android Studio, avec une webview.

```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {
    WebView webView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        webView = (WebView) findViewById(R.id.webview);

        WebSettings webSettings = webView.getSettings();
        webSettings.setJavaScriptEnabled(true);

        webView.loadUrl("https://www.google.com");
    }
}
```

Puis il faut aussi autoriser l'application à aller sur internet depuis le manifeste.

```
<uses-permission android:name="android.permission.INTERNET" />
```

III) Partie ESP 32 et capteurs

Déroulement de la réalisation :

1/ Etude des capteurs individuellement et Étude global des capteurs

2/ Typon pour l'ESP 32

3/ Gestion du MQTT

Réalisation

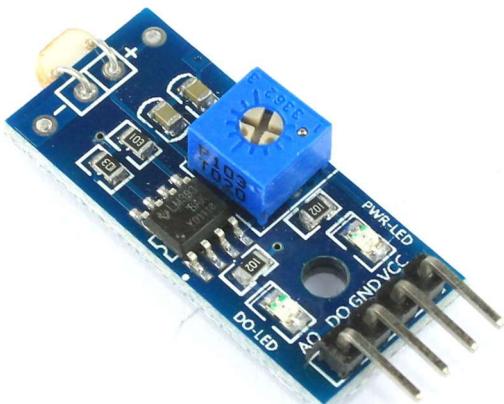
1 / Etude des capteurs individuellement :

LDR

LDR Sensor module : Ce module est parfaitement adapté à la détection de la lumière, généralement observée dans les robots de projet et les systèmes de lumière du jour. Basé sur une résistance dépendante de la lumière, un circuit diviseur de potentiel fournit une sortie analogique en réponse à la quantité de lumière disponible et à l'entrée CC. Tandis qu'un potentiomètre de déclenchement offre une sortie numérique basée sur un seul niveau de déclenchement.

Travaillant à une faible entrée de 3,3 à 5 V, le module produira des valeurs élevées lors de la détection de la lumière et des valeurs faibles sans détection. De plus, un trou de montage garantit que vous disposez d'une méthode de montage de l'unité pour toutes sortes de projets :

Flux SKU: BDAA100156	Country of Manufacture: China	Pin Pitch: 2.54mm
Length (mm): 33	MPN: BDAA100156	Mounting Hole Size: M2.5
Width (mm): 15	UPC: Does Not Apply	Output Type: Analog/Digital
Height (mm): 8	Minimum Operating Temperature: -40°C	Noted Onboard Components: Trigger potentiometer
Brand: Unbranded/Generic	Maximum Operating Temperature: 85°C	Sensor Property: Light
Main Colour: Blue	Number of Items: 1	



Programme de fonctionnalité seul pour le LDR avec les branchements en commentaire :

```
/*
 *          LDR
 * Pin 1 : Broche : +5V (V1N)
 * Pin 2 : Broche : GROUND
 * Pin 3 : Broche : D26
 */
#define LDR 26 // pin 26 LDR
int lumiere;
int journuit;
void setup() {
    // initialise la communication avec le PC
    Serial.begin(115200);
    // initialise les broches
    pinMode(LDR, INPUT);
}
void loop() {
    lumiere = analogRead(LDR); // Lis la valeur analogique de la broche 26
    if (lumiere < 600){ Serial.println("jour"); journuit=1;}
    if (lumiere > 600){ Serial.println("nuit"); journuit=0;}
    delay(1000);
}
```

MQ135

Le capteur **MQ135** permettant de détecter plusieurs types de gaz: le benzène (C₆H₆), l'ammoniaque (NH₃), le sulfure, la fumée et la pollution atmosphérique.

Caractéristiques:

- Alimentation: 5 Vcc
- Plage de mesure: 10 à 1000 ppm
- Sortie analogique et digitale (seuil ajustable via potentiomètre)
- Sensibilité: 2 à 20 kΩ
- Faible temps de réponse
- Haute sensibilité
- Température de service: -20 à 50 °C
- Compatibilité: Arduino et Raspberry Pi
- Dimensions: 52 x 20 x 13 mm
- Poids: 8 g



Programme de fonctionnalité seul pour le MQ135 avec les branchements en commentaire :

```

/*
 *           MQ135
 * Pin 1 : Broche : +5V (VIN) (droite de face)
 * Pin 2 : Broche : GROUND
 * Pin 3 : Broche : rien
 * Pin 4 : Broche : D34      (gauche de face)
 * Capteur de lumière (UV) Définition jour nuit
 */

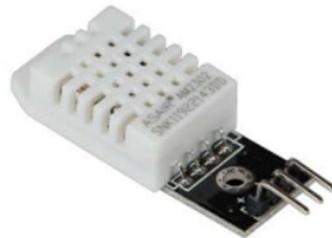
#define MQ135 34 // SI1145
void setup() {
    Serial.begin(115200);
}
void loop() {
    int sensorValue = analogRead(MQ135); // lis la valeur analogique de la broche 34
    Serial.println("The amount of CO2 (in PPM): ");
    Serial.println(sensorValue);
    delay(500);
}

```

DHT22

Caractéristiques:

- Alimentation: 3,3 ou 5 Vcc
- Consommation: 1,5 mA maxi
- Plage de mesure:
 - température: -40 à 80 °C
 - humidité: 0 à 100 % RH
- Précision:
 - température: $\pm 0,5$ °C
 - humidité: ± 2 % RH
- Fréquence d'échantillonage: 2 s
- Interface: One Wire digitale
- Sorties: S, Vcc, Gnd
- Dimensions: 45 x 15 x 10 mm



Programme de fonctionnalité seul pour le DHT22 avec les branchements en commentaire :

```

/*
 *           DHT22
 * Pin 1 : Broche : +5V (VIN)
 * Pin 2 : Broche : D15
 * Pin 3 : Broche : GROUND
 */
#include "DHTesp.h" // library DHT 22
const int DHT_PIN = 15; // Analog Pin DHT 22
float DataTemp; // Valeur température DHT22
float humidite = 0; // Humidité DHT22
void setup() {
    Serial.begin(115200); // Baud par défaut esp 32
    dhtSensor.setup(DHT_PIN, DHTesp::DHT22); // Définition du DHT avec la pin data et le type de DHT
}
void loop() {
    Serial.println("-----DHT22-----");
    TempAndHumidity data = dhtSensor.getTempAndHumidity();
    DataTemp = data.temperature; // Valeur de la température
    humidite = data.humidity; // valeur humidité
    Serial.print("Humidité: "); Serial.print(humidite); Serial.println(" %"); // Affichage de l'humidité en %
    Serial.print("La température est de : "); Serial.print(DataTemp); Serial.println(" °C");
    delay(100); // petite tempo (transition)
}

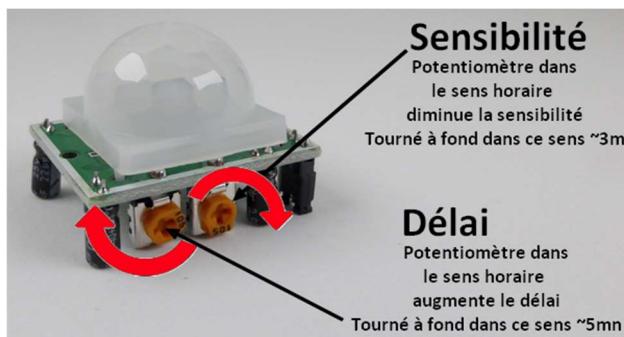
```

PIR

Voltage	4.8 V – 20 V
Current (idle)	<50 µA
Logic output	3.3 V / 0 V
Delay time	0.3 s – 200 s, custom up to 10 min
Lock time	2.5 s (default)
Trigger	repeat : L = disable , H = enable
Sensing range	<120 °, within 7 m
Temperature	-15 ~ +70 °C
Dimension	32 x 24 mm screw-screw 28 mm, M2 Lens diameter: 23 mm



Ne pas oublier de régler la sensibilité et le délai directement sur le PIR :



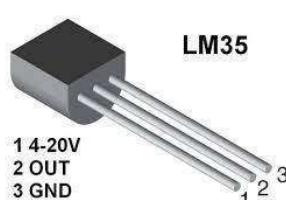
Programme de fonctionnalité seul pour le PIR HC SR501 avec les branchements en commentaire :

```
/*
 * Pin 1 : Broche : +5V
 * Pin 2 : Broche : D27
 * Pin 3 : Broche : GROUND
 */
#define PIR 27 // la data du PIR est sur la PIN 27 de l'ESP32
void setup()
{
    pinMode(PIR,INPUT); // Definition de l'entrée sur l'ESP32 sur la PIN 27
    Serial.begin(115200); // Definition des bauds
}
void loop()
{
    int PIRval = digitalRead(PIR); // récupération de la donnée digitale de présence ou non
    if(PIRval >= HIGH){ Serial.println("Presence"); } // si une présence est détecté alors afficher présence
        else {Serial.println("Aucune presence");} // sinon afficher aucune présence
}
delay(500); // attendre 0.5 ms
}
```

LM35

Capteur de température type LM35

Alimentation : 4 à 20 V
Plage de mesure : 0 / +100°C



Précision : 0,5°C (à 25°C)
Boîtier : TO92

Calculer la température avec le LM35 :

Température en degré =

$$\begin{aligned} &= ((\text{Donnée Digitale} * \text{Valeur max initiale}) / (\text{Valeur max digitale})) * \text{valeur max degré} \\ &= ((\text{Donnée Digitale} * 5V) / (4095)) * 100 \\ &= (\text{Donnée digitale} * 500) / (4095) = \text{température degré voulue} \end{aligned}$$

Programme de fonctionnalité seul pour le PIR HC SR501 avec les branchements en commentaire :

```
/*
 *      LM35
 *  V1N (broche 1) : entrée de 5V
 *  OUT (Broche 2) : Entrée Analogique D33 (GPIO33)
 *  GROUND (broche 3) : ground
 *  On affiche la valeur analogique
 *  mesurée par la broche GPIO 33
 *  de l'ESP32.
 */
float EntreAnalog = 33; // GPIO 33
float ValTemperature; // Valeur température
void setup() {
    Serial.begin(9600); // Baud définit
}
void loop() {
    ValTemperature = ((analogRead(EntreAnalog)*500.0)/4095.0); //Calcul de la température en fonction de l'entrée analogique
    Serial.print("La température est de : "); Serial.println(ValTemperature); // afficher la valeur de la température
    delay(100); //tempo de 0.1 seconde
}
```

1 / Etude Global des Capteurs :

_____ Programme intégrant chaque capteur _____

Programme fonctionnel intégrant tous les capteurs sans transmission de l'information au Raspberry PI 4 B sans MQTT :

```
*****
*      LM35
* Pin 1 : Broche : +5V (V1N) (Gauche de face)
* Pin 2 : Broche : D33 Entrée Analogique
* Pin 3 : Broche : GROUND (Droite de face)
*****
/*****
```



```
*****
*      MQ135
* Pin 1 : Broche : +5V (V1N) (droite de face)
* Pin 2 : Broche : GROUND
```

```

* Pin 3 : Broche : rien
* Pin 4 : Broche : D12      (gauche de face)
* Capteur de lumière (UV) Définition jour nuit
******/
/*****
*      PIR
* Pin 1 : Broche : +5V (V1N)(Gauche de face)
* Pin 2 : Broche : D27
* Pin 3 : Broche : GROUND  (Droite de face)
* *****/
/*****
*      DHT22
* Pin 1 : Broche : +5V (V1N)
* Pin 2 : Broche : D15
* Pin 3 : Broche : GROUND
* *****/
/*****
*      LDR
* Pin 1 : Broche : +5V (V1N)
* Pin 2 : Broche : GROUND
* Pin 3 : Broche : D26
* *****/
#include <Wire.h> //library de cablage
#include "DHTesp.h" // library DHT 22

#define MQ135 12 // MQ135
#define PIR  27 // PIR
#define LDR 26 // pin 26 LDR
const int DHT_PIN = 15; // Analog Pin DHT 22
float EntreAnalog = 33; // Broche d'entrée analogique LM35
float ValTemperature; // Valeur température LM35
bool PIRval; //Valeur présence
float DataTemp; //Valeur temperature DHT22

```

```

float temperature = 0; // Temperature DHT 22
float humidity = 0; // Humidite DHT22
int CO2=0; //CO2 (MQ135)
const int ledPin = 4;
int lumiere;
int journuit;

DHTesp dhtSensor; //library

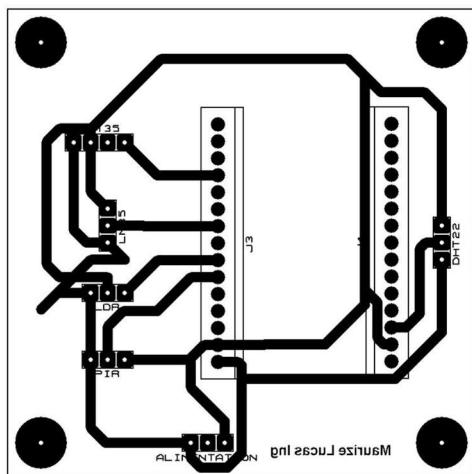
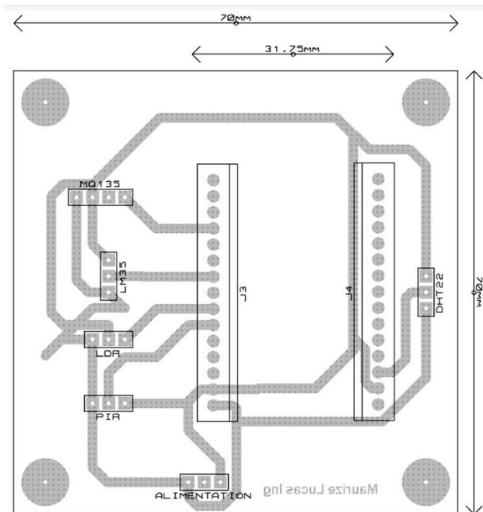
void setup() {
    Serial.begin(115200); // Baud par défaut esp 32
    pinMode(ledPin, OUTPUT);
    pinMode(LDR, INPUT);
    pinMode(PIR,INPUT); // Definition de l'entrée sur l'ESP32 sur la PIN 27
    dhtSensor.setup(DHT_PIN, DHTesp::DHT22); // Definition du DHT avec la pin data et le type de DHT
    Serial.println("-----Debut du Programme-----"); // Affichage du début du programme
}

void loop() {
    delay(2000); // attendre 0.5 secondes
    Serial.println("=====LOOP=====");
    /////////////////////////////////
    // Detecteur Température et Humidité DHT22
    Serial.println("-----DHT22-----");
    TempAndHumidity data = dhtSensor.getTempAndHumidity(); // Definition de la variable de
    récupération de tempéarture et humidité
    DataTemp = data.temperature; // Valeur de la température
    Serial.print("Humidity: ");Serial.print(data.humidity);Serial.println(" %"); // Affciage de l'humidité en %
    delay(100); // petite tempo (transition)
    /////////////////////////////////
    // Temperature LM35
    Serial.println("-----DHT22 + LM35-----");
    ValTemperature = ((analogRead(EntreAnalog)*500.0)/4095.0); //Calcul de la température en fonction
    de l'entrée analogique
    ValTemperature = ((DataTemp + ValTemperature) / 2); // Faire la moyenne avec les deux capteurs
}

```

2/ Typon pour l'ESP 32

Voici le Typon réalisé sur ISIS 8.15 pour l'utilisation de capteur avec un ESP 32 :



Le typon a été testé et est fonctionnel pour chaque capteur.

3/ Gestion du MQTT

Programme fonctionnel du MQTT :

```
*****
```

* LM35

* Pin 1 : Broche : +5V (V1N) (Gauche de face)

* Pin 2 : Broche : D33 Entrée Analogique

* Pin 3 : Broche : GROUND (Droite de face)

```
*****/
```

```
*****
```

- * MQ135
- * Pin 1 : Broche : +5V (V1N) (droite de face)
- * Pin 2 : Broche : GROUND
- * Pin 3 : Broche : rien
- * Pin 4 : Broche : D34 (gauche de face)
- * Capteur de lumière (UV) Définition jour nuit

******/

******/

- * PIR

- * Pin 1 : Broche : +5V (V1N)(Gauche de face)
- * Pin 2 : Broche : D27
- * Pin 3 : Broche : GROUND (Droite de face)

******/

******/

- * DHT22

- * Pin 1 : Broche : +5V (V1N)
- * Pin 2 : Broche : D15
- * Pin 3 : Broche : GROUND

******/

******/

- * LDR

- * Pin 1 : Broche : +5V (V1N)
- * Pin 2 : Broche : GROUND
- * Pin 3 : Broche : D26

******/

#include <Wire.h> //library de cablage

#include "DHTesp.h" // library DHT 22

#include <PubSubClient.h> // envoie de message library

#include <EspMQTTClient.h> // esp32 client library:

```
#define PIR 27 // PIR
#define LDR 26 // pin 26 LDR

// Replace the next variables with your SSID/Password combination
const char* ssid = "AndroidAPCBD0"; // Identifiant de connexion (wi-fi / 4G)
//u will eat ze bugs
//AndroidAPCBD0
//
const char* password = "12345678"; // mot de passe de connexion (wi-fi / 4G)
//codecontre8connexion
//12345678
//
const char* mqtt_server = "192.168.43.39"; // IP du serveur MQTT
//192.168.43.172
WiFiClient espClient; // definition wi-fi client pour l'esp client
PubSubClient client(espClient); // definition du client
DHTesp dhtSensor; //library
long lastMsg = 0;
char msg[50];
int value = 0;

const int DHT_PIN = 15; // Analog Pin DHT 22
float EntreAnalog = 33; // Broche d'entrée analogique LM35
float ValTemperature; // Valeur température LM35
bool PIRval; //Valeur présence
float DataTemp; //Valeur temperature DHT22
float temperature = 0; // Temperature DHT 22
float humidite = 0; // Humidite DHT22
const int ledPin = 4;
int lumiere;
```

```

int journuit;

///////////
/// Sous Programme Wi-fi
///////////

void setup_wifi() {
    delay(10); // attendre 10 ms
    Serial.println(); // Sauter une ligne
    Serial.print("Connecting to "); // afficher connecté à :
    Serial.println(ssid); // afficher l'identifiant de connexion
    WiFi.begin(ssid, password); // commencer la connexion au wi-fi
    while (WiFi.status() != WL_CONNECTED) { // tant que le wi-fi n'est pas connecté :
        delay(500); // attendre 0.5 s
        Serial.print("."); // afficher "."
    }
    // si on est connecté au wi-fi:
    Serial.println(""); // sauter une ligne
    Serial.println("WiFi connected"); // afficher connecté au wi-fi
    Serial.println("IP address: "); // afficher adresse IP :
    Serial.println(WiFi.localIP()); // afficher l'adresse IP
}

void callback(char* topic, byte* message, unsigned int length) { // répondre à un message (inutile ici)
    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();

}

//////////



/// Sous Programme Reconnexion MQTT

//////////


void reconnect() {

// Loop until we're reconnected

while (!client.connected()) { // tant que pas connecté :

    Serial.print("Address MQTT : ");

    Serial.println(mqtt_server);

    Serial.print("Attempting MQTT connection..."); // afficher tentative de connexion :

    if (client.connect("ESP32Client")) { // si il est connecté :

        Serial.println("connected"); // afficher connecté

        client.subscribe("output"); // envoyer sortie

    } else { // sinon

        Serial.print("failed, rc="); // afficher tentative de connexion loupée

        Serial.print(client.state()); // afficher l'état du client

        Serial.println(" reessayera dans 5 secondes"); // afficher reessayera dans 5 secondes

        delay(5000); // attendre 5s

    }

}

}

//////////


/// VOID SETUP

//////////


void setup() {

Serial.begin(115200); // Baud par défaut esp 32

setup_wifi(); // setup wi fi

```

```

client.setServer(mqtt_server, 1883); // definition du mqtt serveur

client.setCallback(callback); // definition du rappel de donnée

pinMode(ledPin, OUTPUT);

pinMode(LDR, INPUT);

pinMode(PIR,INPUT); // Definition de l'entrée sur l'ESP32 sur la PIN 27

dhtSensor.setup(DHT_PIN, DHTesp::DHT22); // Definition du DHT avec la pin data et le type de DHT

Serial.println("-----Debut du Programme-----"); // Affichage du début du programme

}

//////////



/// VOID LOOP

//////////

void loop() {

    if (!client.connected()) { // si le client n'est pas connecté :

        reconnect(); // se reconnecter

    }

    delay(2000); // attendre 0.5 secondes

    Serial.println("=====LOOP=====");

    ////////////


    // Detecteur Température et Humidité DHT22

    Serial.println("-----DHT22-----");

    TempAndHumidity data = dhtSensor.getTempAndHumidity(); // Definition de la variable de récupération de tempéarture et humidité

    DataTemp = data.temperature; // Valeur de la température

    humidite=data.humidity;

    Serial.print("Humidity: ");Serial.print(humidite);Serial.println(" %"); // Affcihage de l'humidité en %

    delay(100); // petite tempo (transition)

    ////////////


    // Temperature LM35

    Serial.println("-----DHT22 + LM35-----");

```



```
dtostrf(humidite, 1, 2, humString);

client.publish("humidite", humString); // envoie de la donnée humidité

char tempString[8];

dtostrf(ValTemperature, 1, 2, tempString);

// tempString[8]=(char)Valtemperature;

client.publish("temperature",tempString); // envoie de la donnée température

char CO2String[8];

dtostrf(CO2, 1, 2, CO2String);

client.publish("CO2", CO2String); // envoie de la donnée CO2

client.loop();

long now = millis();

if (now - lastMsg > 5000) {

    lastMsg = now;

}

}
```