**Raspbery Pi**

No Raspbery Pi correm 2 programas em simultaneo,

Mosquito - Eclipse Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 3.1 and 3.1.1. Mosquitto is lightweight and is suitable for use on all devices from low power single board computers to full servers.

The MQTT protocol provides a lightweight method of carrying out messaging using a publish/subscribe model. This makes it suitable for Internet of Things messaging such as with low power sensors or mobile devices such as phones, embedded computers or microcontrollers.

The Mosquitto project also provides a C library for implementing MQTT clients, and the very popular mosquitto\_pub and mosquitto\_sub command line MQTT clients.

Link -> https://mosquitto.org/

Node Red - Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

The light-weight runtime is built on Node.js, taking full advantage of its event-driven, non-blocking model. This makes it ideal to run at the edge of the network on low-cost hardware such as the Raspberry Pi as well as in the cloud.

Link -> <https://nodered.org/>

O Rpi é o nosso agente/intermediário de Mensagens MQTT e Servidor Web Para Fazer Interface Grafica da informação adquirida pelos Sensores

**Guia de Instalação –**

-Installing Mosquitto -> <https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/>

*sudo apt install -y mosquitto mosquitto-clients*

*sudo systemctl enable mosquitto.service*

-Installing Node Red -> https://nodered.org/docs/hardware/raspberrypi

*bash <(curl -sL https://raw.githubusercontent.com/node-red/raspbian-deb-package/master/resources/update-nodejs-and-nodered)*

*cd ~/.node-red*

*npm i node-red-dashboard*

Correr Programas:

node-red-start

Opcional, adicionar ao startup do rpi

Na directoria Home

*nano startup*

escrever no ficheiro

*node-red-start*

guardar e sair

De seguida

*sudo nano /etc/rc.local*

Adicionar ao rc.local esta linha

*su - pi -c "screen -dm -S teste /home/pi/startup"*

Nota, é necessário ter o screen package instalado

sudo apt-get install screen

**ESP8622**

Para usarmos o ESP8622 com o arduino IDE tivemos de adicionar a Board ao Programa, seguimos este instructable:<https://www.instructables.com/id/Steps-to-Setup-Arduino-IDE-for-NODEMCU-ESP8266-WiF/>

Para o nosso projecto foi necessário instalar uma biblioteca adicional: PubSubClient <https://pubsubclient.knolleary.net/> para enviarmos e receber mensagens do tipo MQTT

De seguida foi só implementação de Codigo

O Objectivo do ESP é ser o nosso Nodulo Sensorial(Sensor Node) isto é receber informação do sensor de Temperatura e Humidade (NOME\_DO\_SENSOR) e dos LDR’s(pela pic) e enviar para o rpi por wifi através do protocolo MQQT

**Definicao de MQTT:**

MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. For example, it has been used in sensors communicating to a broker via satellite link, over occasional dial-up connections with healthcare providers, and in a range of home automation and small device scenarios. It is also ideal for mobile applications because of its small size, low power usage, minimized data packets, and efficient distribution of information to one or many receivers

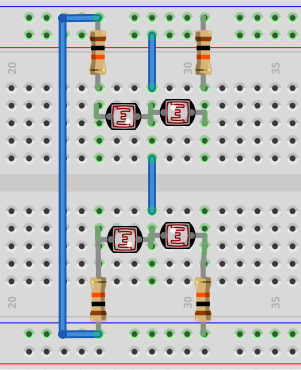
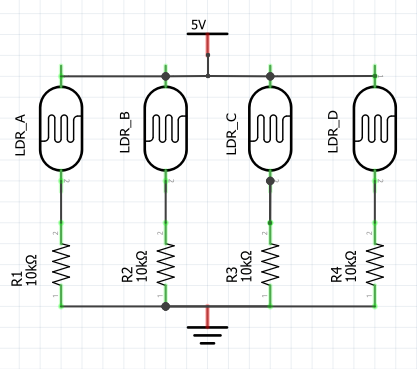
Link- <https://mqtt.org/>

More on this topic- <https://mqtt.org/faq>

PIC24

Usamos 4 Pinos da PIC, 1 para cada LDR, em modo Analogico, para obtermos o valor da tensão aos Terminais de Cada LDR e convertemos esse valor para Digital usando um ADC(Analog to Digital Converter) de 12 bit, e de seguida enviamos esses 4 valores numa string via UART para o ESP8622

Hardware

BOM: (BILL OF MATERIALS)

Resistencia 10k ohms x4

Resistencia 220 ohms x2

PhotoResistencia(LDR) x4

Sensor Humidade SHT21 x1

Peça 3D x1

Rpi

Esp8622

PIC24

Explicação do Codigo

Começando na PIC, envia os 4 valores digitais(ADC) dos LDR’s por UART com refesh rate de **XXX**ms

Esp8622, Começa por ligar se ao WIFI, com as credenciais dadas, de seguida tenta ligar se ao MQQT broker(RaspberyPi) dado o IP, apos ligação completa recebe o valor dos LDR’s , Normaliza-os segundo a calibração efectuada previamente , e envia-os para o Rpi através de Wifi com o procolo MQTT a cada (100ms).

Ao mesmo tempo Adquire a Temperatura Ambiente e a Humidade Relativa do Sensor de Humidade e Temperatura SH21 e envia também esses valores para o Rpi, mas agora com um refesh rate diferente(20.000ms=20s)