

Tutorial du projet Crazy Baby

EISE-5

Huiling BAO - Yingshan LIU - Mingda WANG

A. Généralité du projet

Notre projet vise à réaliser la fonction de surveillance des bébés, à l'aide de ESP32 CAM et de certains [capteurs](#) et [actionneurs](#).

Les parents peuvent visualiser la surveillance de la vidéo du bébé en temps réel , ainsi que les paramètres environnementaux (tels que la température et l'humidité) où se trouve dans la chambre du bébé, et l'accélération du berceau. Lorsque les pleurs continus du bébé ou l'amplitude anormale du balancement du berceau sont détectés, les parents reçoivent un rappel sonore à distance(via un buzzer). De plus, les parents peuvent également contrôler à distance les lumières de la chambre de bébé via la plateforme (parce que nous n'avons pas pu acheter un interrupteur intelligent/connecté, nous utilisons temporairement une LED à la place).

B. Matériaux utilisés dans ce projet

<i>Matériaux</i>	<i>Model</i>	<i>Quantité</i>	<i>Lien</i>
Microcontrôleur	ESP32	2	https://www.gotronic.fr/art-module-nodemcu-esp32-28407.htm
Module caméra	ESP32 CAM	1	https://www.gotronic.fr/art-module-esp32-cam-32630.htm
Raspberry Pi	3b+	1	https://www.gotronic.fr/art-carte-raspberry-pi-3-b-27826.htm
Capteur de température/d'humidité	DHT11	1	https://www.gotronic.fr/art-capteur-de-t-et-d-humidite-dht11-st052-26117.htm
Capteur de pression de l'air	BMP 280	1	https://www.gotronic.fr/art-capteur-de-pression-bmp280-sen0372-32856.htm
Accéléromètre	GY521	1	https://www.amazon.com/MPU-6050-MPU6050-Sensors-Accelerometer-Arduino/dp/B07K8TVD2H
Capteur Sonore	GT1146	1	https://www.gotronic.fr/art-capteur-sonore-gt1146-26144.htm

Buzzer	KY-006 Module	1	https://sensorkit.fr/joy-it.net/index.php?title=KY-006_Module_buzzer_passif
Lipo Rider	Carte LiPo Rider Plus	2	https://www.gotronic.fr/art-carte-lipo-rider-plus-106990290-31762.htm
LiPo	3.7V	2	https://www.gotronic.fr/art-accu-lipo-3-7-v-1000-mah-pr523450-5813.htm
Fils de cavalier	Génériques	-	-
LED	-	1	-
Breadboard	Générique	-	-

Liste des matériaux

C. Préparation du projet

V1 - Réalisation par Home Assistant

Connaissance et l'installation du Home Assistant



Home Assistant

Ce système est basé sur la plateforme Home Assistant. Home Assistant est un logiciel d'automatisation et de gestion d'objets connectés. Il est gratuit, récent et populaire et est simple à mettre en œuvre.

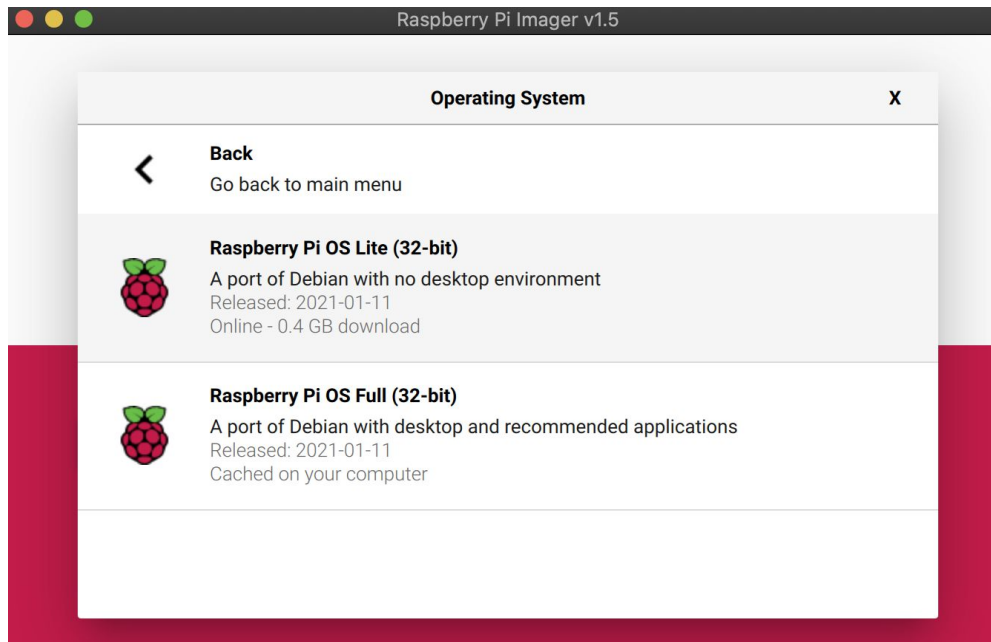
Voici l'introduction de Home Assistant: <https://www.home-assistant.io/>

Nous avons choisi d'installer d'abord le système Raspberry Pi sur le Raspberry Pi, puis d'installer le package de support de l'assistant domestique sur sa base (méthode 3). Sur le site officiel de Home Assistant, nous pouvons trouver une variété de méthodes d'installation. Y compris l'installation du logiciel sur le système Raspberry Pi, ou l'installation directe du système Home Assistant. Consultez plus d'informations sur le site suivant:

<https://www.home-assistant.io/installation/raspberrypi>

1. Installation du Raspberry Pi OS.

Nous avons utilisé le **Raspberry Pi Imager** : <https://www.raspberrypi.org/software/> et choisi la version lite. Il suffit d'insérer la carte SD et d'écrire.



Raspberry Pi Imager

2. Installation l'environnement de Home Assistant dans le Raspberry Pi OS, y compris les étapes suivant :
 - Installation des indépendances
 - Création d'un compte de Home Assistant
 - Création d'un environnement virtuel

Les instructions détaillées se trouvent dans le lien ci-dessous.

<https://www.home-assistant.io/installation/raspberrypi#install-home-assistant-core>

3. Ensuite, on démarre le serveur Home Assistant. Avec les instructions suivantes:
Attention: Les instructions varient selon le système d'exploitation (Ici, on utilise le Raspberry Pi OS)

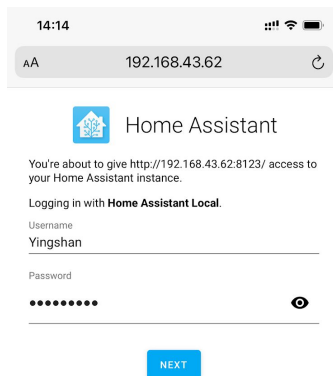
```
pi@raspberrypi:~ $ sudo -u homeassistant -H -s
homeassistant@raspberrypi:/home/pi $ cd /srv/homeassistant
homeassistant@raspberrypi:/srv/homeassistant $ source bin/activate
(homeassistant) homeassistant@raspberrypi:/srv/homeassistant $ hass
2021-02-13 18:25:49 ERROR (MainThread) [metno] https://aa015h6buqvi86i1.api.met
.no/weatherapi/locationforecast/2.0/complete returned
2021-02-13 18:25:49 WARNING (MainThread) [homeassistant.bootstrap] Support for t
he running Python version 3.7.3 is deprecated and will be removed in the first r
elease after December 7, 2020. Please upgrade Python to 3.8.0 or higher.
```

Démarrage du serveur HA

4. On visite l'interface d'utilisateur via l'adresse IP et le port du serveur (Voir l'adresse IP du Raspberry Pi). Il vous demandera de créer un utilisateur et un mot de passe


lorsque vous vous connectez pour la première fois, puis vous pourrez sélectionner la connexion automatique.

<http://192.168.43.62:8123/>



14:14

AA 192.168.43.62

 Home Assistant

You're about to give <http://192.168.43.62:8123/> access to your Home Assistant instance.

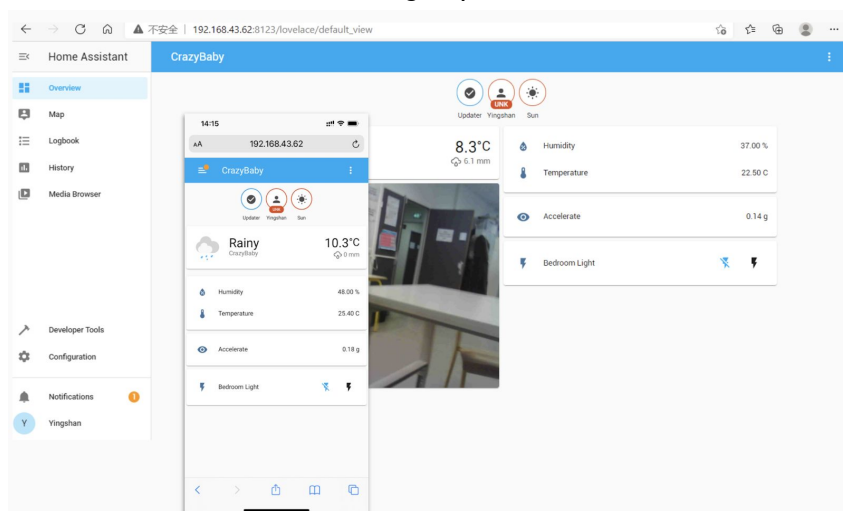
Logging in with **Home Assistant Local**.

Username
Yingshan

Password
••••••••

NEXT

Sign up



L'interface utilisateur

Préparation du broker MQTT

Home Assistant prend en charge le protocole MQTT et divers brokers MQTT. Home Assistant a un MQTT broker intégré (HBMQTT), ce qui suit est les informations du broker:

<https://hbmqtt.readthedocs.io/en/latest/>

Setting	Value
Broker	localhost
Port	1883
Username	homeassistant
Password	HA API pwd(123456789)
Websocket	8080

Broker HA

Dans le fichier de configuration `configuration.yaml`, on peut faire les configuration de MQTT, y compris, les paramètres à publier ou subscriber, ainsi que la configuration du broker. Ici on utilise le broker intégré HBMQTT.

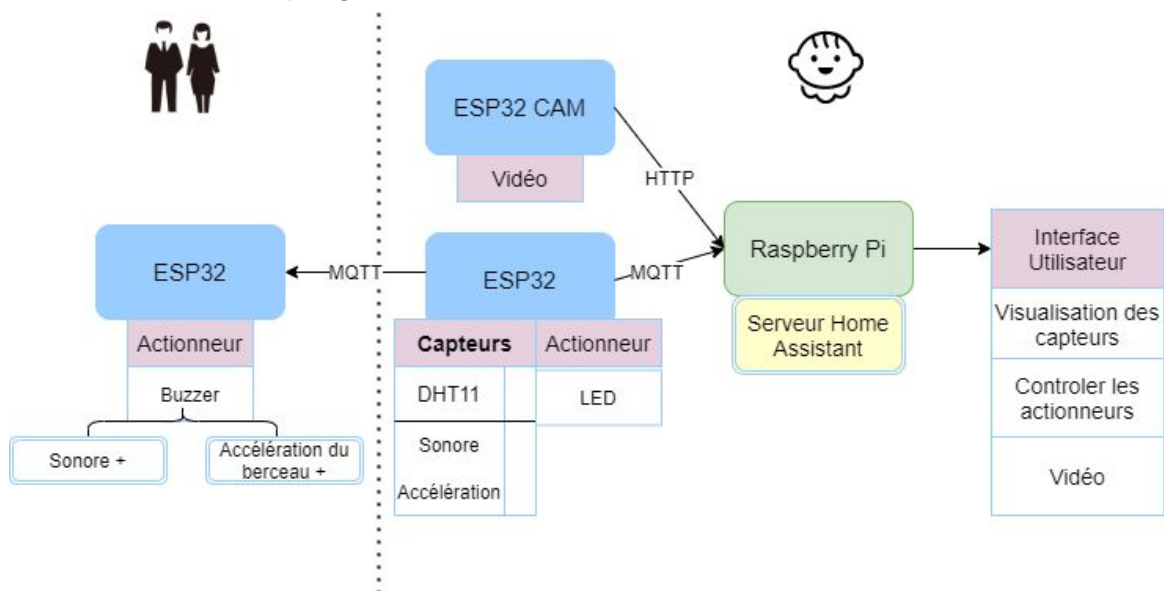
L'utilisation de HA est principalement basée sur le fichier de configuration, qui est un fichier au format `.yaml` situé à : <https://www.home-assistant.io/docs/configuration/>

1. Ajoutez les configurations ci-dessous du broker au **`configuration.yaml`**.

```
mqtt:
  broker: 192.168.43.62
  port: 1883
  client_id: homeassistant
  keepalive: 60
  username: homeassistant #mqtt username
  password: 123456789 #mqtt password
  discovery: true
  tls_insecure: false
```

Configuration du MQTT

Architecture du projet

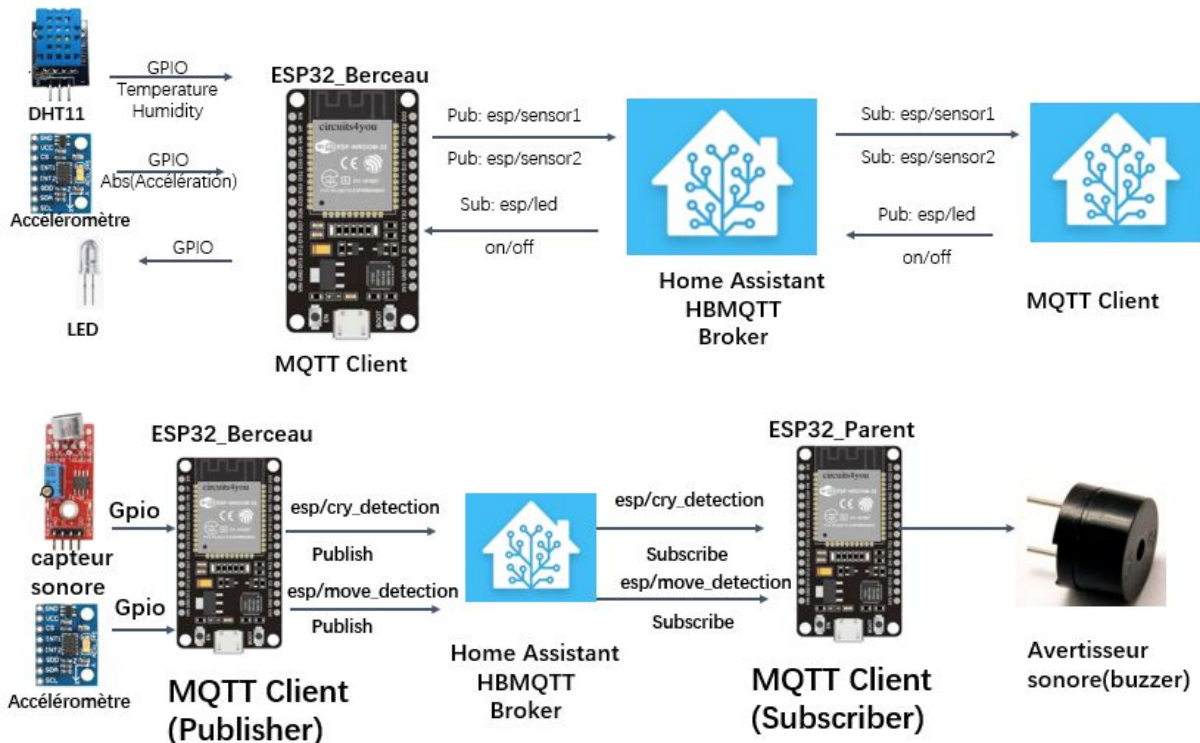


On prend le Raspberry Pi comme serveur, on installe Home Assistant ici. On a un ESP32 CAM, on peut voir la vidéo en visitant l'adresse IP du ESP32 CAM.

On a un ESP32 comme le microcontrôleur, il peut communiquer avec le Raspberry Pi par le protocole MQTT, il envoie l'humidité, température et l'accélération du berceau au serveur Home Assistant. Par contre, les utilisateurs peuvent contrôler l'état du LED dans l'ESP 32

sur l'interface d'utilisateur de Home Assistant. Les communications sont toutes réalisées par MQTT, on utilise le broker HBMQTT, qui est intégré au Home Assistant.

Pour informer les parents à distance lorsque les cas sont particuliers, on met un ESP32 au parent. L'ESP 32 du berceau peut détecter l'accélération du berceau et le pleurs du bébé selon le son. Lorsqu'ils dépassent les seuils, L'ESP 32 du berceau envoie(publish) un message MQTT au ESP32 du parent. Et l'ESP32 des parents subscribe ce message. Lorsque un message est bien reçu, le buzzer est activé.



Etapes à suivre

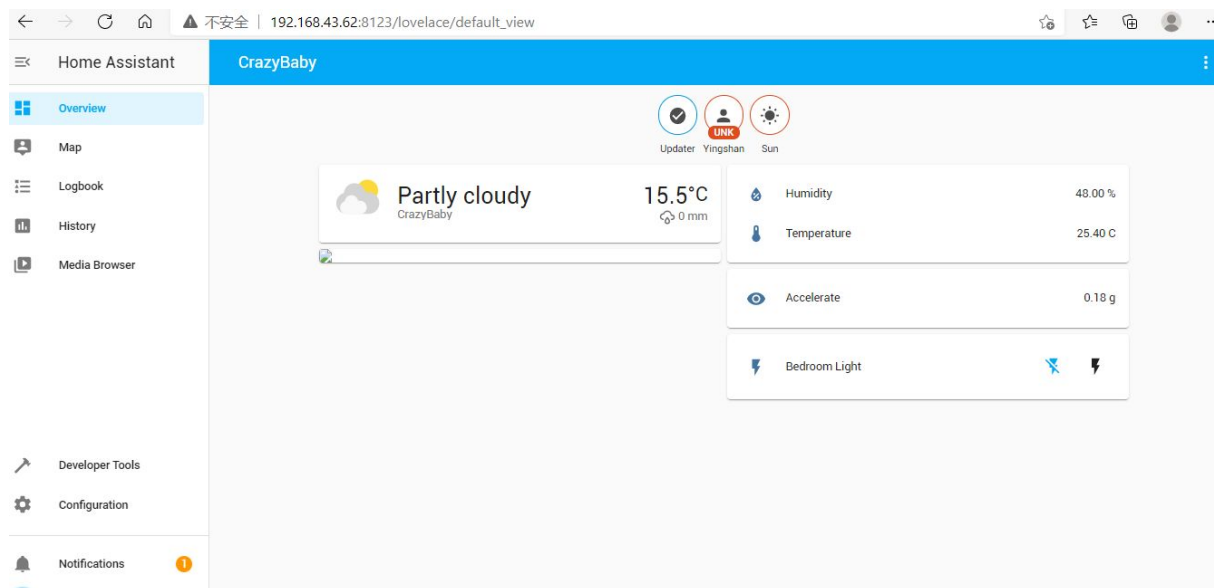
Visualiser le vidéo depuis Home Assistant

On utilise un microcontrôleur ESP32 CAM pour la capture vidéo. Le ESP32 CAM sert comme un serveur vidéo, le code du ESP32 CAM se trouve dans ce chemin du Github. [El-SE5_2020-2021_CrazyBaby/v1_HAServeur/codeSource_Arduino/homeassis_video/](https://github.com/El-SE5_2020-2021_CrazyBaby/v1_HAServeur/codeSource_Arduino/homeassis_video/)

```
//Replace with your network credentials
const char* ssid = "Yingshanwifi";
const char* password = "123456789";
```

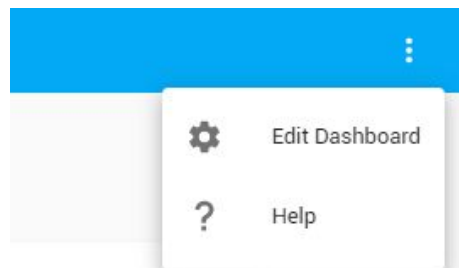
Ici, on utilise le module wifi pour le connecter avec l'internet. On remplace les deux lignes par notre propre id et password.

En visitant l'adresse IP du ESP32 CAM, on peut voir la vidéo. Ensuite, on démarre le serveur Home Assistant. Après la visite de l'IP du raspberry pi, on peut voir le page web du Home Assistant.



Interface utilisateur

On ajoute le fenêtre du vidéo, cliquer Edit Dashboard:



On ajoute un picture entity, ensuite, sur la fenêtre de configuration, on ajoute l'adresse IP du ESP32 CAM :

Picture Card Configuration

Image Path (Required)

http://192.168.43.139

Tap Action (Optional)

No Action

Hold Action (Optional)

No Action

Theme (Option...

SHOW CODE EDITOR

CANCEL

CLOSE

Ajouter le Picture Card

Ensuite, on peut visualiser la vidéo sur le Dashboard.

Réalisation du communication MQTT du ESP32 du berceau

Le code du ESP32 du berceau se trouve dans ce chemin.à

[EI-SE5_2020-2021_CrazyBaby/v1_HAServeur/codeSource_Arduino/homeassistant_sensor/](#)

On configure les paramètres du réseau, ainsi que le broker MQTT qu'on utilise. Ici on utilise le broker intégré dans le platform Home Assistant.

```
// Network params
const char* ssid = "Yingshanwifi";
const char* password = "123456789";
const char* mqtt_server = "192.168.43.62";
const char* mqtt_user = "homeassistant";
const char* mqtt_password = "123456789";
```

Le code réalise les mesures des capteurs, ainsi que les publications des données par les topics correspondants. Il a aussi une subscription du topic d'état du LED. Quand on reçoit un message qu'on subscribe, dans la fonction callback, il contrôle le LED selon le contenu du message.

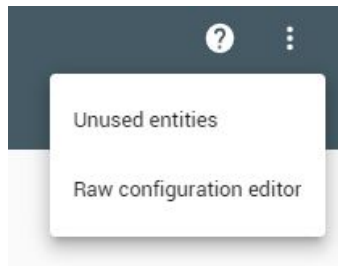
Visualiser les valeurs des capteurs par MQTT

1. Il faut d'abord ajouter les variables à visualiser dans le fichier de configuration du Home assistant. DHT11: l'humidité, la température; Accéléromètre: accélération; Comme les données sont transmises via mqtt, il faut ajouter le champ "platform: mqtt"

```
sensor:
  - platform: mqtt
    state_topic: "esp/sensor1"
    name: 'Temperature'
    unit_of_measurement: 'C'
    value_template: '{{ value_json.Temperature }}'
    device_class: temperature
  - platform: mqtt
    state_topic: "esp/sensor1"
    name: 'Humidity'
    unit_of_measurement: '%'
    value_template: '{{ value_json.Humidity }}'
    device_class: humidity
  - platform: mqtt
    state_topic: "esp/sensor2"
    name: 'Accelerate'
    unit_of_measurement: 'g'
    value_template: '{{ value_json.Accelerate }}'
```

Configuration des variables dans configuration.yaml

2. Démarrer le serveur et accéder à l' UI. Cliquez sur le champ "Unused entities" dans le coin supérieur droit. Vous pouvez trouver des entités qui ont été ajoutées mais qui n'y sont pas encore utilisées.



Unused entities

These are the entities that you have available, but are not in your Lovelace UI yet. Select the entities you want to add to a card and then click the add card button.

☒ ☐ ☐

↑ Entity

Entity ID

Entités non utilisé

Réaliser le contrôle du LED

Le contrôle du LED est aussi réalisé par le protocole MQTT. Les opérations de réalisation similaire à l'étape précédente, mais ici le platform Home Assistant faire publish des topics.

```
switch:
  - platform: mqtt
    command_topic: "esp/led"
    name: "Bedroom Light"
    payload_on: "ON"
    payload_off: "OFF"
```

Configuration des variables dans configuration.yaml

On ajoute un topic à publish de type switch dans le fichier configuration.yaml. Ensuite, on ajoute le entity unused dans le dashboard, comme l'étape précédente.

Réalisation des alarmes aux parents en utilisant le buzzer

Le code du ESP32 du parents se trouve dans ce chemin.à
EI-SE5_2020-2021_CrazyBaby/v1_HAServeur/codeSource_Arduino/parent_receive/

L'ESP32 du parent sert comme un entité à subscribed des topics d'alarme, L'ESP32 du berceau sert comme un entité à publish des topics d'alarme, lorsque la variation de fréquence de son ou l'accélération du berceau dépasse certaines valeurs.

On configure les paramètres du réseau, ainsi que le broker MQTT qu'on utilise . Ici on utilise le broker intégré dans le platform Home Assistant.

Dans l'ESP32 du parent, lorsqu'il reçoit un message, il active le buzzer en modes différents selon le contenu du message. Ici '1' correspond au pleur, et 2 correspond au tremblement du berceau.

```
if((char)payload[0] == '1')
    buzzer1();
else if((char)payload[0] == '2')
    buzzer2();
```

On configure les modes de son du buzzer en changeant les fréquences et volumes du signal.

```
void buzzer1(){
    for(int i=0; i<3;i++){
        ledcWriteTone(channel, 2000);
        ledcWrite(channel, 125);
        delay(600);
        ledcWrite(channel, 0);
        delay(400);
    }
}

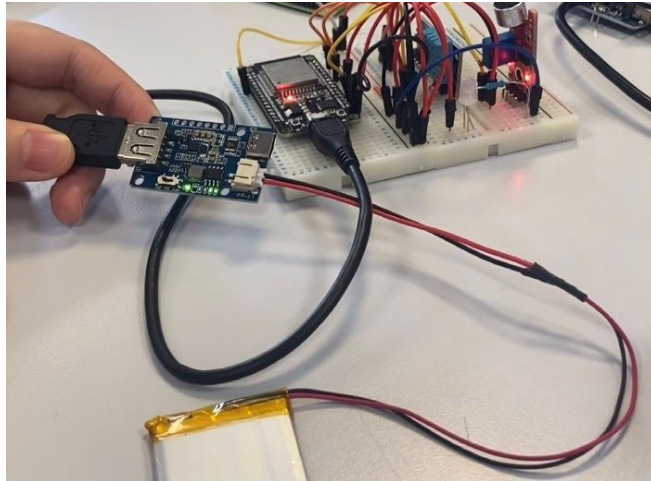
void buzzer2(){
    for(int i=0; i<10;i++){
        ledcWriteTone(channel, 2000);
        ledcWrite(channel, 125);
        delay(100);
        ledcWrite(channel, 0);
        delay(100);
    }
}
```

La fonction `void ledcWrite(uint8_t channel, uint32_t duty)` contrôle le PWM du son(volume).

La fonction `double ledcWriteTone(uint8_t channel, double freq)` contrôle la fréquence du son(tonne).

Alimentation des microcontrôleurs

Enfin, afin de parvenir à la portabilité de l'appareil, on a décidé d'ajouter de la puissance mobile aux deux ESP32. On a choisi deux cartes LiPo Rider Plus qui ont une sortie de 5V, utilisent la pile ou l'alimentation de type C comme entrée. Puis connectez avec le LiPo. On peut donc utiliser un câble USB pour l'alimenter

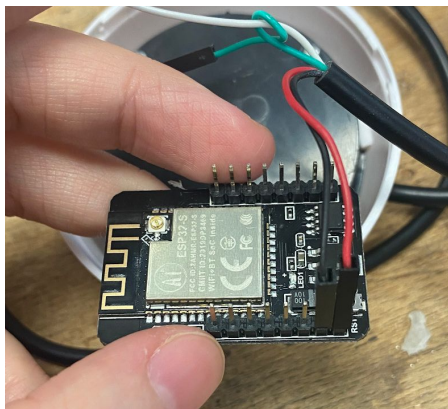


Alimentation du ESP32

Décoration du ESP32 CAM

Afin de rendre notre caméra plus réaliste et d'améliorer la sécurité de l'équipement, nous avons acheté un boîtier de caméra et installé notre ESP32 CAM à l'intérieur.

Nous avons collé le CAM à l'intérieur du boîtier et utilisé un cordon d'alimentation pour l'alimenter.

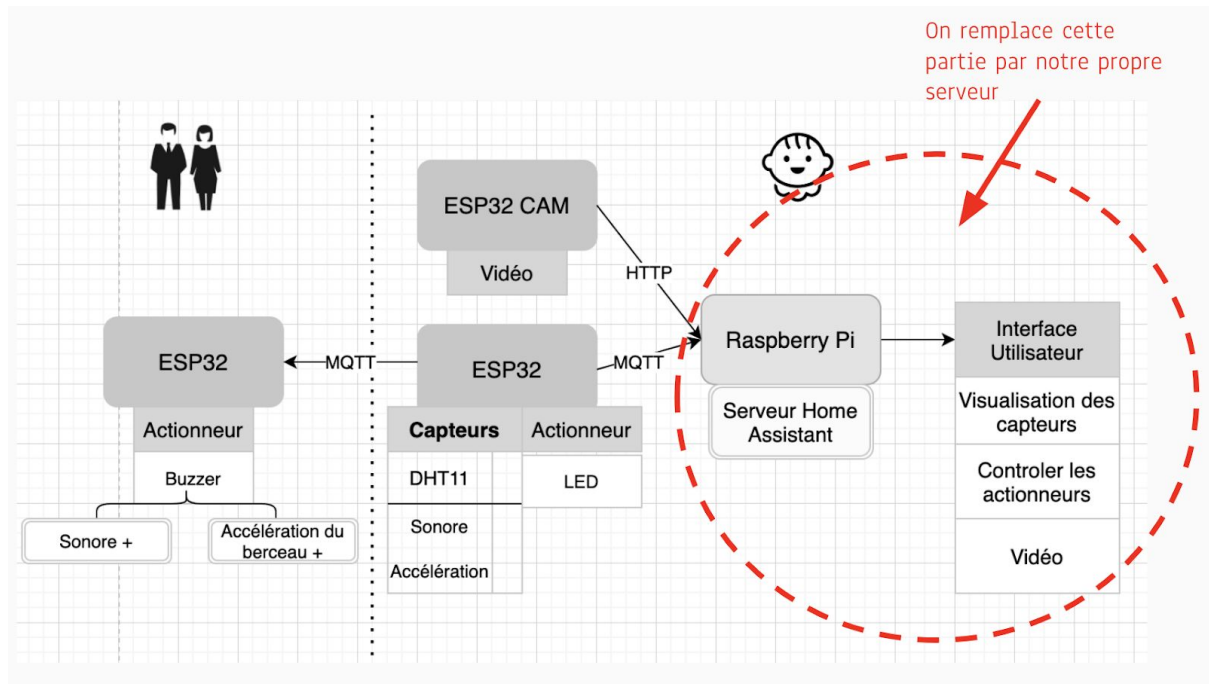


Alimentation de la caméra

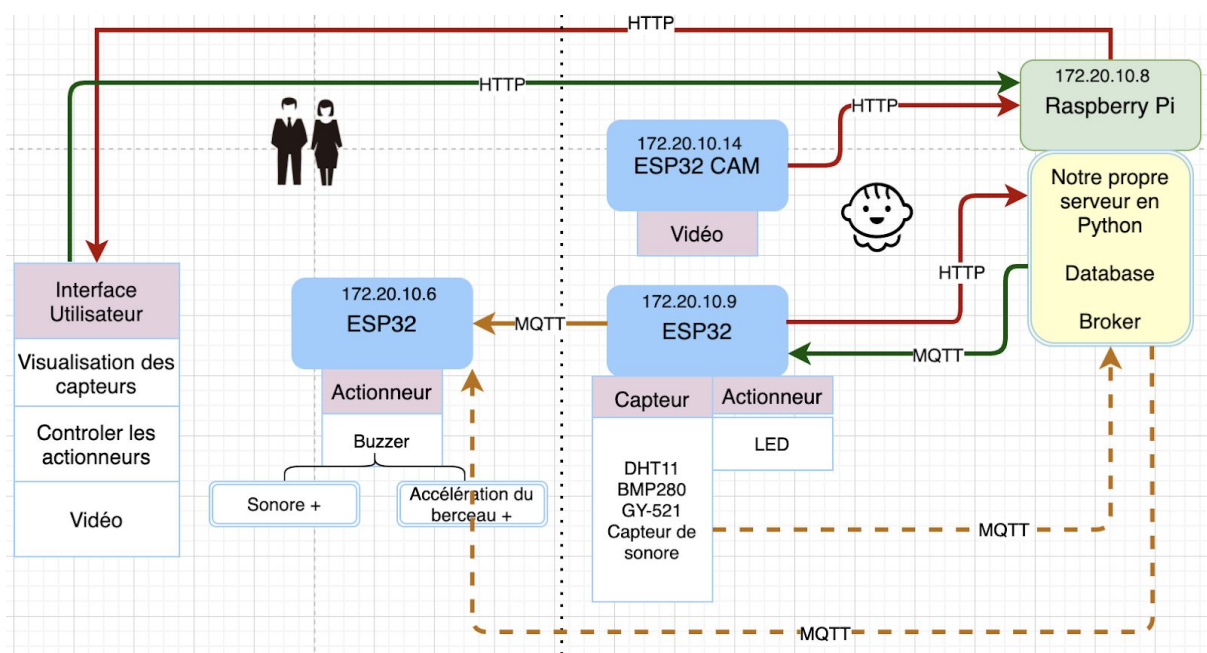
V2 - Réaliser ce système par nous-même totalement

Explication de notre propre système

Jusqu'à maintenant on a fini le système pour surveiller le bébé, mais on veut réaliser tout le système par nous-même, c'est-à-dire on va faire le serveur, le database et les pages Web par nous-même mais pas utiliser le **HomeAssistance**.



Et notre système est comme ça:



Il y a deux parties. A droit c'est la partie de bébé, il y a un **esp32** pour mesurer les paramètres environnementaux, un **esp32 CAM** pour surveiller le bébé en utilisant le vidéo et un raspberry qui contient un broker **mosquitto**, le serveur écrit par nous-même et un database. A gauche la partie de parent, il y a un **esp32** qui est relié avec un buzzer pour alarmer le parent quand le bébé pleure ou bouge et un interface d'utilisateur que l'on peut utiliser pour visiter le serveur par page web.

Maintenant on explique comment on transmet les données. D'abord c'est le ligne rouge : le **esp32** mesure les paramètres environnementaux et puis les transmet vers le serveur en utilisant le protocole **http**, et le **esp32 CAM** aussi envoie le vidéo de bébé au serveur en utilisant le protocole **http**, puis l'utilisateur peut visiter le serveur en utilisant une page web par le protocole **http**.

Ensuite on explique le ligne vert : une fois que l'utilisateur veut allumer ou éteindre la lumière il peut le contrôler depuis l'interface utilisateur, et la commande va être transmise par le protocole **http** vers le serveur, puis le serveur va transmettre la commande vers le **esp32** en utilisant le protocole **mqtt**.

A la fin c'est ligne jaune : une fois que le **esp32** de bébé détecte que le bébé pleure(par le capteur sonore) ou le bébé bouge(par l'accéléromètre) le **esp32** de bébé va indiquer au **esp32** de parent en utilisant le protocole **mqtt**.

Commencer faire le système

Etant donnée que les codes sont très similaire que l'on a expliqué avant donc dans cette partie on explique uniquement comment on change le code pour re-produire le système et on explique le code de serveur et le code de database

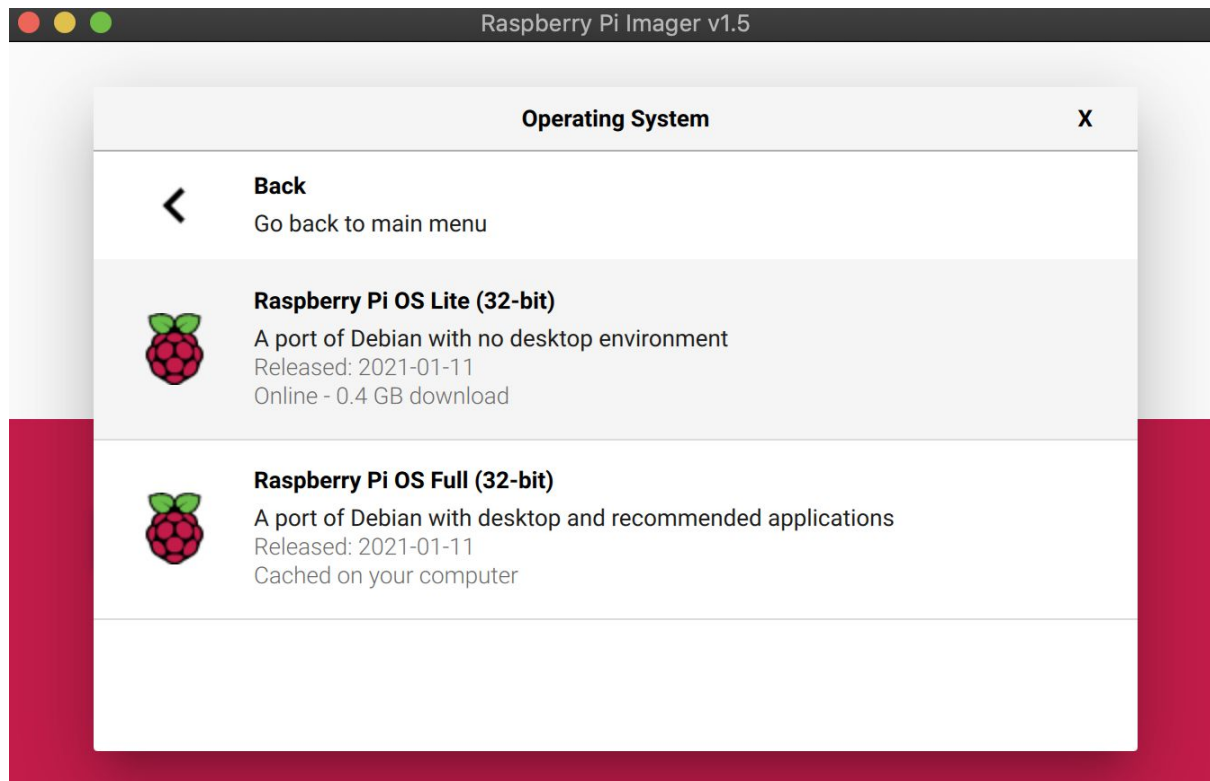
D'abord on fait la partie de raspberry

Le raspberry contient un serveur, un database et un broker.

On télécharge le système du raspberry

<https://www.raspberrypi.org/software/>

et on utilise l'OS Full 32-bits



Puis on met dans le fichier **boot** deux fichiers, un fichier **ssh** et un fichier **wpa_supplicant.config** et dans ce fichier on écrit :

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
    ssid="WiFi SSID"
    psk="mot de passe"
}
```

On peut relier le raspberry en utilisant ssh :

```
PolytechSorbonne@EISE5:~ $ ssh pi@[ip du raspberry]
```

On installe le **sqlite3** et **mosquitto**

```
pi@raspberrypi:~ $ sudo apt-get install mosquitto
pi@raspberrypi:~ $ sudo apt-get install sqlite3
```

On écrit une base de données dans votre ordinateur, pour la base de données vous pouvez voir le code **crazyBaby.sql** dans **EISE-5_2020-2021_CrazyBaby/v2_propreServeur/code**. Dans la base de données il y a des tableaux qui sauvegardent l'information de l'utilisateur, les données des capteurs et les types des capteurs.

On crée une base de données sur le raspberry


```
PolytechSorbonne@EISE5:~ $ scp crazyBaby.sql pi@[ip du raspberry]:/home/pi
pi@raspberrypi:~ $ touch crazyBaby.db
pi@raspberrypi:~ $ sqlite3 crazyBaby.db
sqlite> .read crazyBaby.sql
```

Ensuite pour faire le serveur vous pouvez voir le **serveur.py** code sur github dans **EISE-5_2020-2021_CrazyBaby/v2_propreServeur/code**.

Changez en ligne 61 l'ip de camara par votre propre ip de camara

```
61                                     <li><a href="http://172.20.10.14">Video</a></li>
```

Changez en ligne 58, 59, 60, 62, et 63 les ip par les ip corespondants

```
58                                     <li><a href="http://172.20.10.8:8888/home">Home</a></li>
59                                     <li><a href="http://172.20.10.8:8888/temprature">Temperature</a></li>
60                                     <li><a href="http://172.20.10.8:8888/humidite">Humidite</a></li>
61                                     <li><a href="http://172.20.10.14">Video</a></li>
62                                     <li><a href="http://172.20.10.8:8888/mouvement">Mouvement</a></li>
63                                     <li><a href="http://172.20.10.8:8888/airPress">Air Pressure</a></li>
```

Changez en ligne 302 l'ip et le numéro de la porte du broker par votre propre ip et numéro de porte du broker.

```
302                                     client.connect('172.20.10.8', 1883)
```

Changez en ligne 308 et 311 le topic par votre propre topic qui contrôle la lumière.

```
306                                     if(query['led'][0] == "on"):
307                                         print("led on")
308                                         client.publish('led0n0ff', payload='4')
309                                     elif(query['led'][0] == "off"):
310                                         print("led off")
311                                         client.publish('led0n0ff', payload='5')
```

Changez aussi en ligne 469 l'ip du serveur par votre propre ip du serveur.

```
469                                     httpd = server_class(("172.20.10.8", 8888), MyHandler)
```

Ensuit on explique le code **serveur.py**

D'abord on a les fonctions **helloPage**, **temperaturePage**, **humiditePage**, **mouvementPage**, **airPressPage** et **configurationPage**, ils sont les page Web pour que les utilisateurs peuvent visiter le serveur.

Puis on a une classe **MyHandler** qui est utilisée pour la communication en utilisant le protocole **http**. La méthode **do_GET** est utilisée pour obtenir les données et la méthode **do_POST** est utilisée pour envoyer les données.

Ensuite on a une classe **MySQL** qui est l'interface de database. Il contient deux méthodes, la méthode **select** est utilisée pour prendre les données de database et la méthode **insert** est utilisée pour insérer les données au database.

Puis on fait le code de esp32 de parent

Voir le code **version3.ino** dans **EISE-5_2020-2021_CrazyBaby/v2_propreServeur/code**.

Changez en ligne 26 et 27 pour configurer votre WiFi

```
26  const char* ssid = "mingdaWiFi";
27  const char* password = "wmd97128";
```

Changez en ligne 29 l'ip du broker par votre propre ip du broker

```
29  const char* mqtt_server = "172.20.10.8";
```

Changez en ligne 39 et 40 l'ip et numéro de poste du serveur par votre propre ip et numéro de poste du serveur.

```
39  const char* host = "172.20.10.8";
40  const int httpPort = 8888;
```

Changez en ligne 99 le topic par votre propre topic qui est utilisé pour contrôler la lumière

```
99      client.subscribe("ledOnOff");
```

Changez en lignes 144, 165, 172 et 179 le topic par votre propre topic pour indiquer que le bébé pleure ou bouge.

```
144      client.publish("cryBaby", "0");
165      client.publish("cryBaby", "1");
172      client.publish("cryBaby", "2");
179      client.publish("cryBaby", "3");
```

Ensuit on fait le code du esp32 parent

Vous pouvez voir le code **esp32Recv.ino** dans
EISE-5_2020-2021_CrazyBaby/v2_propreServeur/code

Vous devez changer en ligne 5 et 6 pour configurer votre WiFi

```
5  const char* ssid = "mingdaWiFi";  
6  const char* password = "wmd97128";
```

Changez en ligne 11 l'ip du broker par votre propre ip du broker

```
11  const char* mqtt_server = "172.20.10.8";
```

Changez en ligne 74 le topic par votre propre topic qui contrôle le buzzer

```
74      client.subscribe("cryBaby");
```

A la fin on fait le code de la camara

Vous pouvez voir le code **video.ino** dans
EISE-5_2020-2021_CrazyBaby/v2_propreServeur/code

Changez en ligne 27 et 28 pour configurer votre WiFi

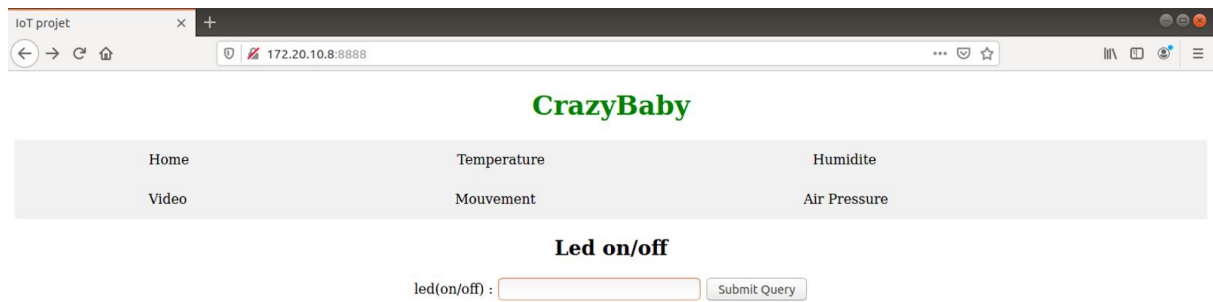
```
27  const char* ssid = "mingdaWiFi";  
28  const char* password = "wmd97128";
```

Commencer utiliser le système

On démarre le serveur en raspberry

```
pi@raspberrypi:~$ python3 serveur.py
```

Puis on visite le serveur en utilisant le navigateur



On peut allumer ou éteindre la lumière en entre **on** ou **off** au bas, et on peut aussi voir les paramètres environnementaux en cliquant sur différentes parties **Home**, **Température** etc.

Vous pouvez voir le vidéo de démonstration dans
EISE-5_2020-2021_CrazyBaby/v2_propreServeur/video

Et il y a aussi des capture d'écran pour la fonctionnalité de chaque partie dans
EISE-5_2020-2021_CrazyBaby/v2_propreServeur/capture