UNIVERSITÉ PARIS 13

Institut Galilée
UNIVERSITÉ PARIS 13

INTERNET OF THINGS

Projet SMART MAILBOX

Groupe LOFE
Ferroudja DJELLALI
Lounas HADJ ALI
Lotfi HAMICHE

99 avenue Jean-Baptiste Clément 93430 Villetaneuse

Sommaire

Introduction

Internet des objets est l'interconnexion entre Internet et des objets, des lieux et des environnements physiques, il repose avant tout sur les objets connectés. Un objet connecté a la capacité de capter une donnée et l'envoyer, via le réseau Internet ou autre technologies, pour que celle-ci soit analysée et visualisée sur des tableaux de bord dédiés.Les objets connectés interagissent avec leur environnement par biais de capteurs : température, vitesse, humidité, vibration…
L'IOT est en partie responsable d'un accroissement exponentiel du volume de données généré sur le réseau, à l'origine du big data.
Internet des objets touche tous les secteurs, il n'y a pas de limite à ce que l'on peut connecter en disposant sur des objets des capteurs.
On peut par exemple contrôler la température de notre maison juste en cliquant sur un bouton sur notre téléphone, connecter des miroirs, des voitures, des montres et même des boites aux lettres.

Problématique: *Peut-t-on recevoir ses courriers comme on reçoit les emails?*

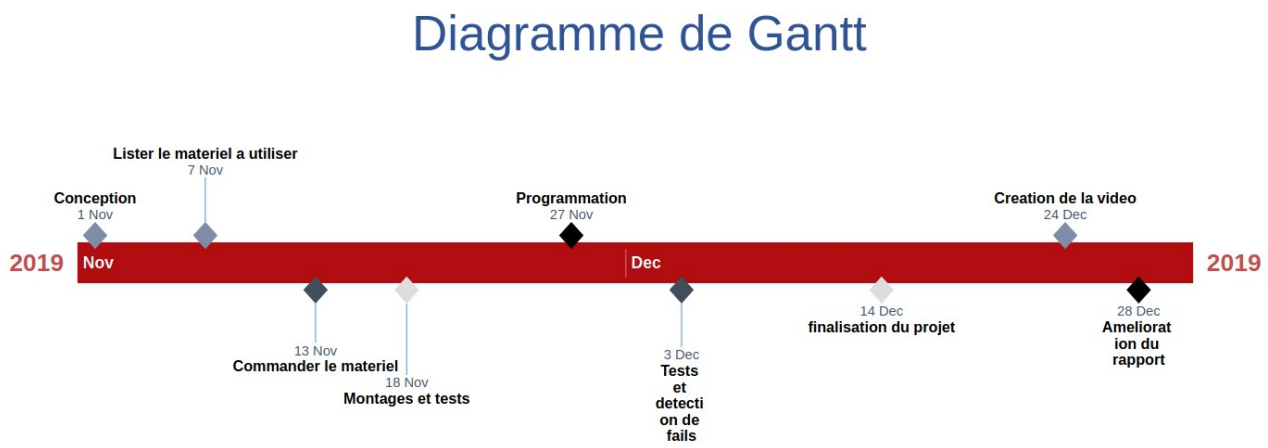
La plupart d'entre nous ont l'habitude de vérifier leur boite aux lettres en rentrant ou en sortant de la maison, mais il y a des moments ou l'on attend un courriers important ou même des colis, mais à chaque fois ou on jette un œil sur ses courriers, on ne trouve toujours rien, ce qui n'est pas pratique surtout pour les maisons ou la boite au lettres se trouve à l'extérieur, coté rue. Puis on ne peut pas passer notre temps à surveiller le passage du facteur.

Objectif du projet

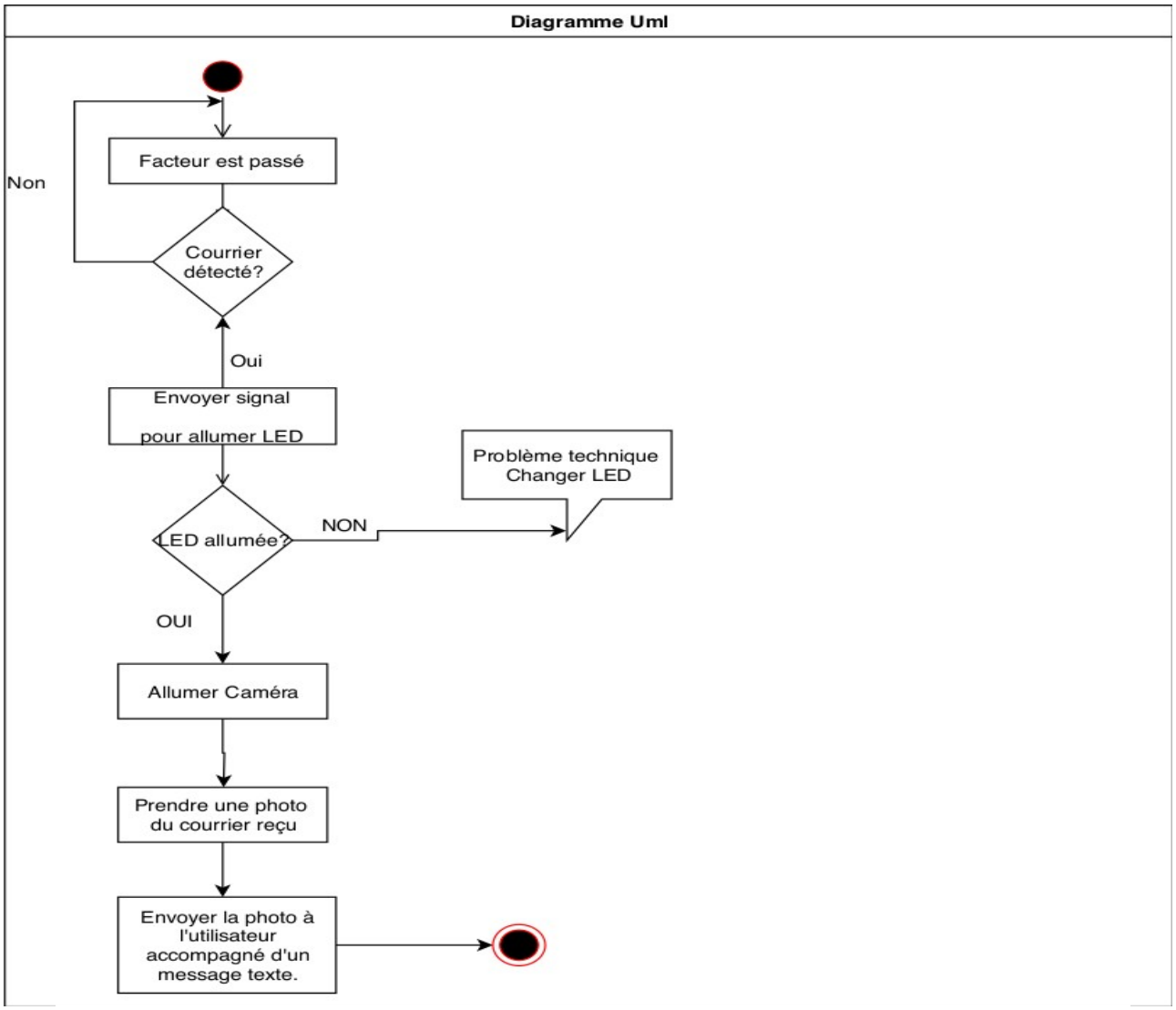
Afin d’être informé de la réception d’un courrier ou colis sans se déplacer pour consulter notre boite au lettre, nous avons l’idée de rendre cette dernière plus intelligente, en la connectant via le réseau wifi. Cela nous permettra de recevoir, en temps réel des notifications sur notre smartphone dès la réception d’un courrier ou d’un colis.

Organisation initiale

Avant de se lancer dans le projet, nous avons réalisé le diagramme de Gantt.
Cela nous permet de bien s’organiser, respecter la contrainte de limite de temps et mieux ce retrouver entre le projet, les tps et les autres cours.



Nous avons aussi réalisé le diagramme de cas d’utilisateur et diagramme UML pour mieux expliquer le contenu et le objectifs du projet



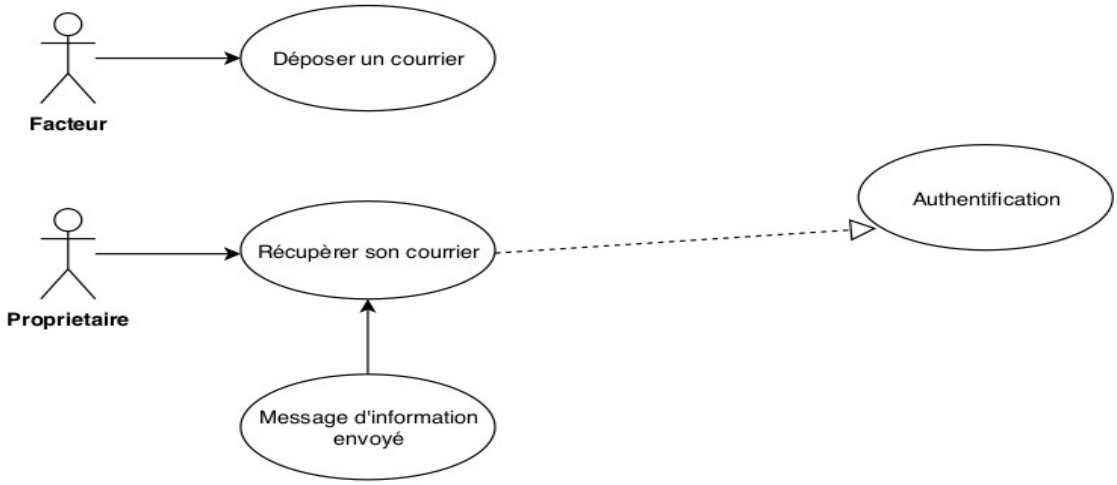
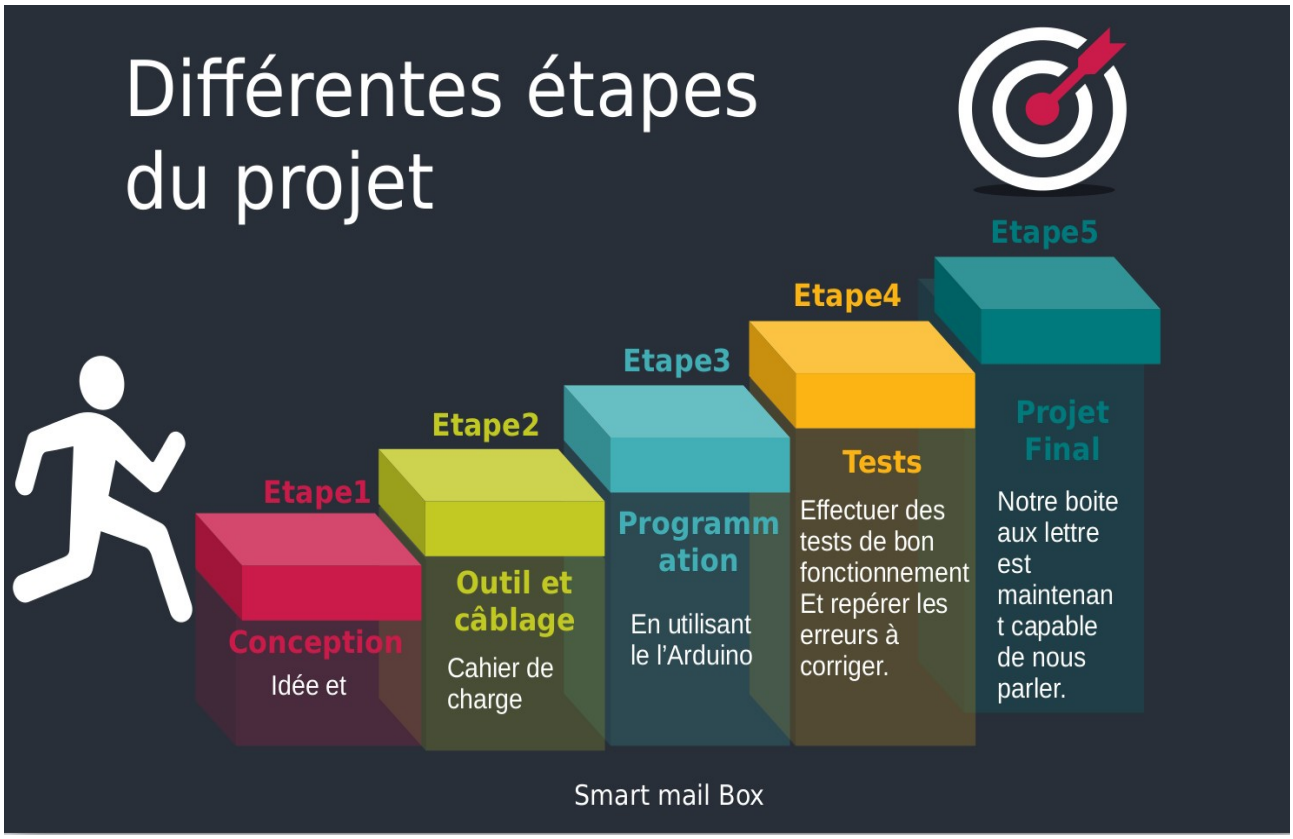


DIAGRAMME DE CAS D'UTILISATION

Composants utilisés

- Un ESP32
- Des capteurs ultrason HC SR04
- La carte Arduino
- Des câbles
- Une camera (ESP-Camera)
- Une boite en carton
- La breadboard
- Résistances

Description générale des étapes du projet

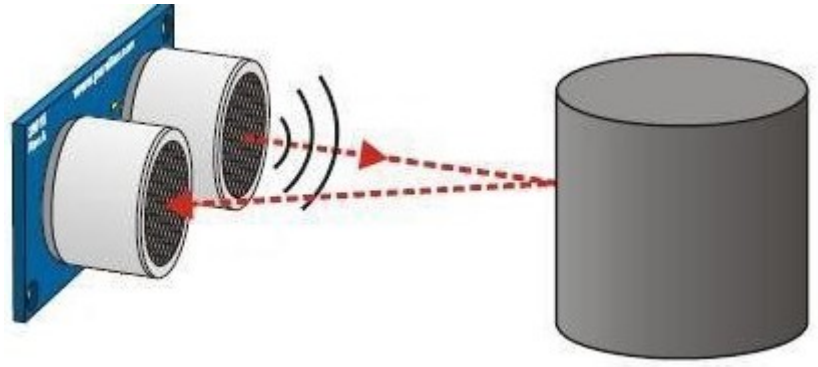


Description détaillée

Etape 1 : Détecter la présence d'une lettre grâce à un capteur ultrason

Nos capteurs fonctionnent avec une alimentation 5v.

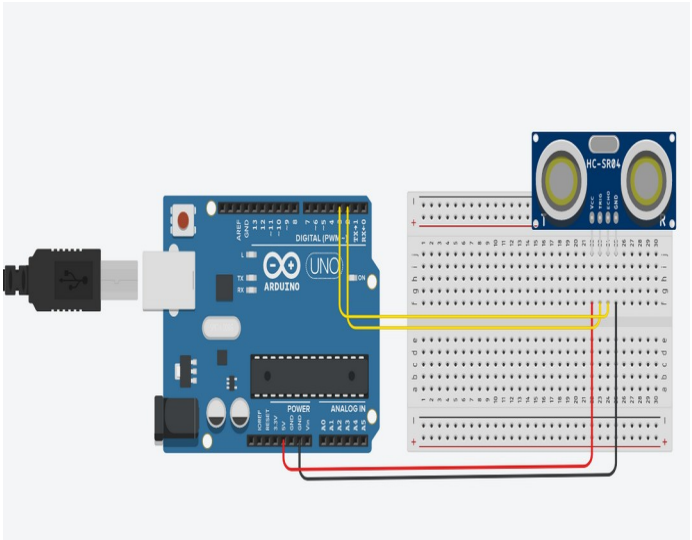
Les capteurs à ultrasons nous permettent de mesurer une distance en créant une impulsion sur une des broches. Grâce à cette mesure, nous pouvons détecter une variation de la distance lorsqu'un courrier est inséré dans la boîte aux lettres. Nous envoyons alors l'information grâce à un MessageCallback.
Pour le faire fonctionner, nous avons relié le capteur à notre carte uno en plaçant les quatre branches du capteur aux pins de l'uno.

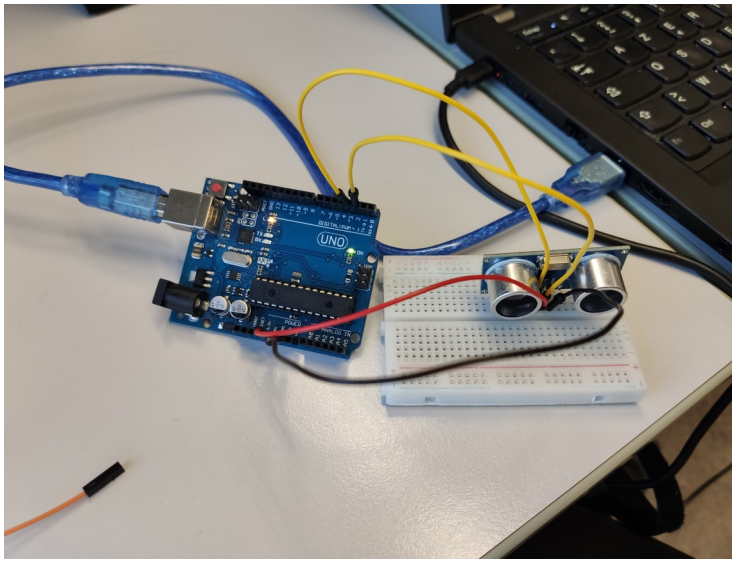


Code

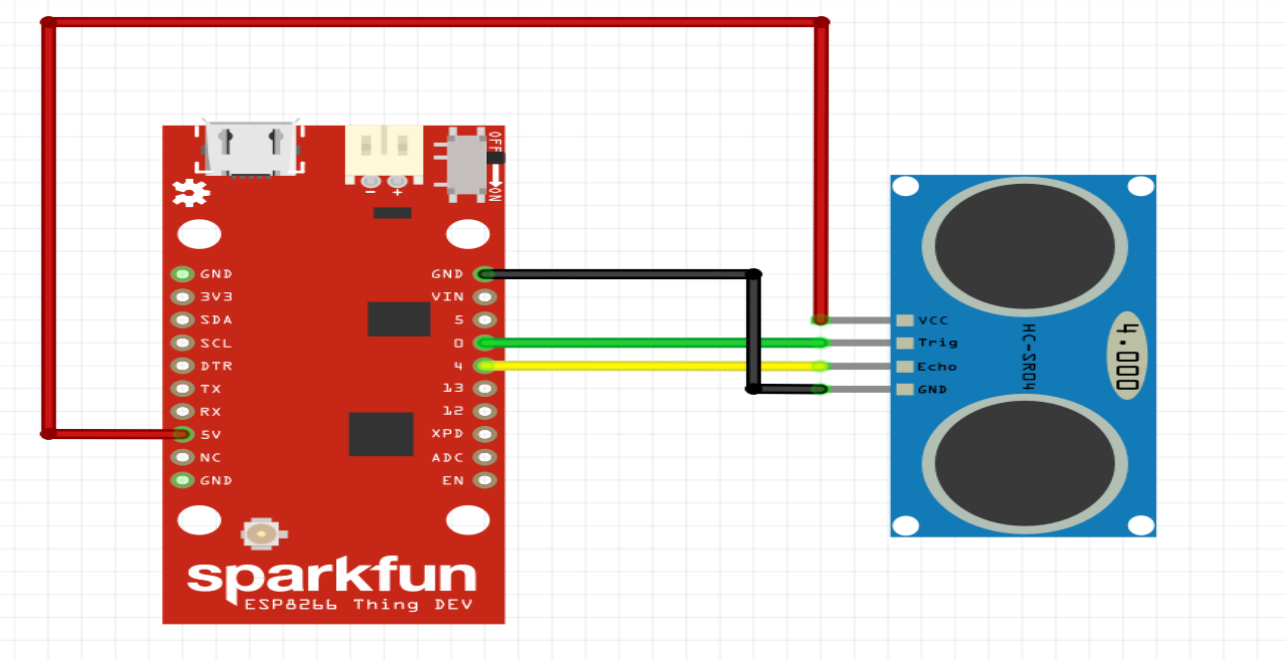
```
void loop()
{
            //partie pour le detecteur de mouvement
                digitalWrite(trigger,LOW);
                delayMicroseconds(2);
                digitalWrite(trigger,HIGH);
                delayMicroseconds(10);
               mesure=pulseIn(echo,HIGH)/58.0;
                Serial.print(" DISTANCE : ");
               Serial.println(mesure);
```

Branchement





On peut le brancher aussi sur la carte esp32 comme suit :



Etape 2: Allumer la lumière

Afin de visionner le contenu de notre boite aux lettres ou permettre a la caméra de prendre des photos, nous avons pensé à utiliser une lampe connectée.

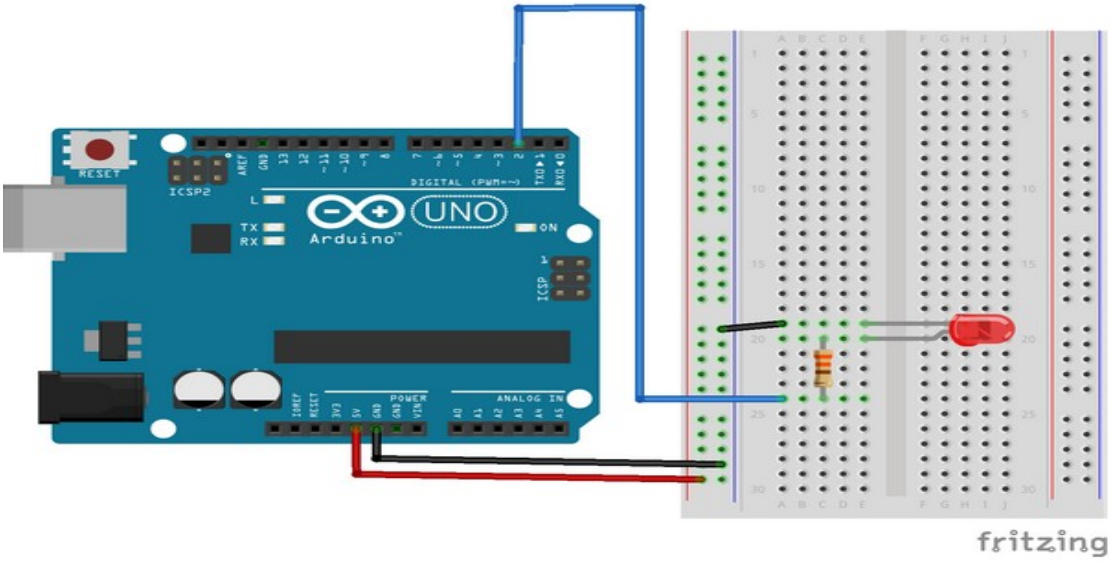
Nous avons réussi à connecter cette lampe (achetée chez action) avec notre matériel arduino.

Pour cela, nous l'avons branché comme on a l'habitude de brancher les leds fournies dans le kit arduino, sauf qu'ici, nous n'avons pas utilisé de résistances car la lampe contient déjà une résistance intégrée.

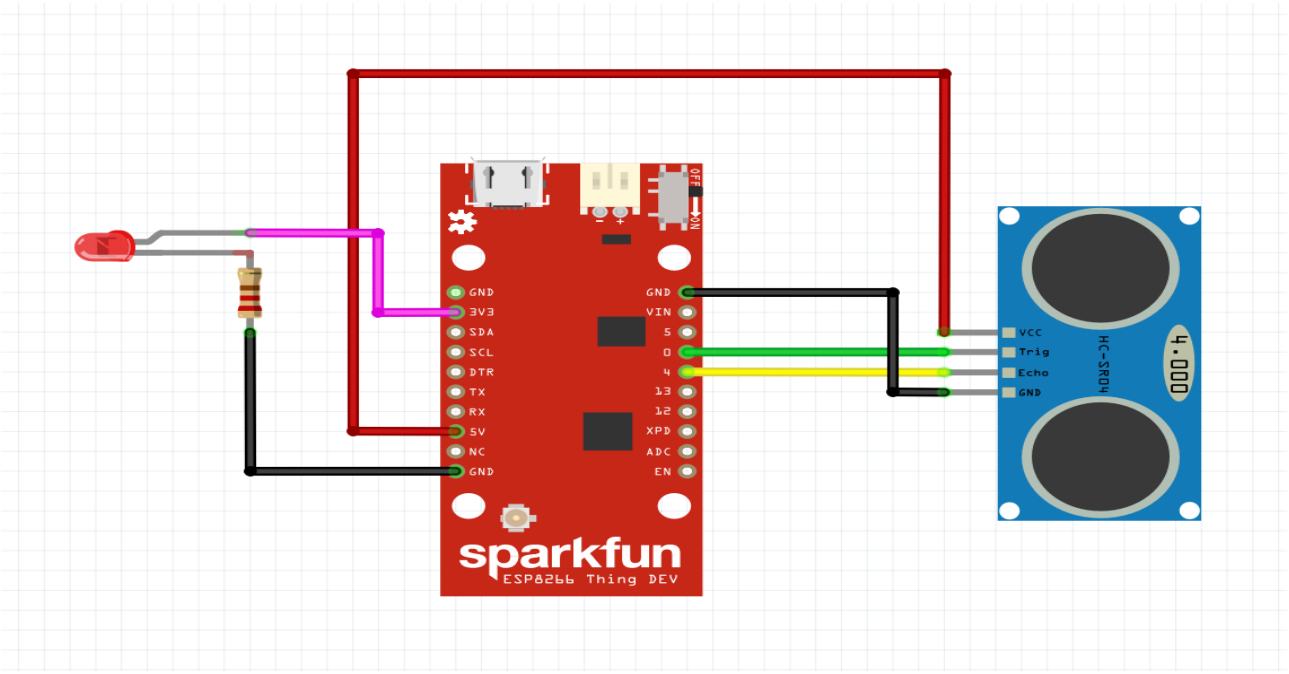
Nous avons la possibilité de l'allumer directement via notre smartphone juste en cliquant sur un bouton de l'application que nous avons créé (cela n'est utile que lorsque nous voulons consulter notre boite aux lettres à distance en utilisant notre caméra comme une camera de surveillance).

Mais l'intérêt principal est le fait quelle est reliée au capteur, ce qui fait qu'elle s'allume dès que notre capteur détecte la présence d'un courrier. Nous avons mis une condition afin de l'éteindre après un petit moment de la réception du courrier (inutile de la laisser allumer tout le temps).

Branchement



Nous avons regrouper les deux etapes (1 et 2) afin que la led s’allume à la détection d’un mouvement, en faisant le branchement suivant :



A l’issu de cette partie, nous avons obtenu le système suivant :

- Detecter un mouvemet
- Led qui s’allume
- Un mail de notifications est envoyé à l’utilisateur.

Code

1. Déclaration des variablesInitialisation

sketch_jan19a §

```
#include<WiFi.h>;
#include "ESP32_MailClient.h"
int trigger=12;
int led=2;
int transf=17;
int echo=14;
int mesure=1000;
bool b;
int cmpt=0;
char*  ssid="louness";
char* password="Ha123456789";
long d;
SMTPData smtpData;
```

2. Initialisation

sketch_jan19a §

```
void setup()
{
  //initialisation
    Serial.begin(115200);
    pinMode(led,OUTPUT);
   // digitalWrite(led,LOW);
    pinMode(trigger,OUTPUT);
    pinMode(echo,INPUT);
    pinMode(2,OUTPUT);
    WiFi.begin(ssid,password);

    Serial.print(" CONNECTION EN COURS .");
    while(WiFi.status()!=WL_CONNECTED)
    {
        Serial.print(".");
        delay(1000);
    }
//affichage de ladreess IP obtenue à l'issue de connection au WIFI
                Serial.print("   YOUR IP@ IS : ");
                Serial.println(WiFi.localIP());
}
```

3. Partie détecteur de mouvement

```
void loop()
{

            //partie pour le detecteur de mouvement
                digitalWrite(trigger,LOW);
                delayMicroseconds(2);
                 digitalWrite(led,LOW);
                  delayMicroseconds(10);
                digitalWrite(trigger,HIGH);
                delayMicroseconds(10);
               mesure=pulseIn(echo,HIGH)/58.0;
                Serial.print(" DISTANCE : ");
               Serial.println(mesure);
```

4. Envoi du mail

```
//dans le cas où on detecte une lettre on envoie le mail par le code suivant
  if(mesure<15){
             Serial.println(" detecté");
             digitalWrite(led,HIGH);
             delay(500);
                smtpData.setLogin("smtp.googlemail.com",465, "hadjalilounas@gmail.com", "Ha123456789");
                     smtpData.setSender("ESP32", "hadjalilounas@gmail.com");
                      smtpData.setPriority("High");
                         smtpData.setSubject("alerte courier");
                             smtpData.setMessage("un nouveau courier est déposé dans votre boite aux lettres",false);
                             smtpData.addRecipient("hadjalilounas@gmail.com");
                                   if (!MailClient.sendMail(smtpData))
                                       Serial.println("Error sending Email, " + MailClient.smtpErrorReason());
          // delay(2000);
           mesure=100;
  }
   }
```

Etape 3: Utilisation de la camera

Cette partie est très mportante pour notre projet, nous avons utilisé une ESP32 Camera (qui est alimentée en 5V), dont nous pouvons soit s'en servir comme une camera de surveillance pour voir le contenu de notre boite à tout moment, comme nous pouvons juste l'utiliser pour capturer le fichier reçu, et dans ce cas elle synchronise dès la détection du courrier

Code

CameraWebServer

```
#include "esp_camera.h"
#include <WiFi.h>
#include "Arduino.h"
#include "FS.h"
#include <SD.h>
#include <SPI.h>
#include "soc/soc.h"
#include "SD_MMC.h"
#include "soc/rtc_cntl_reg.h"
#include "driver/rtc_io.h"
#include <EEPROM.h>
#define CAMERA_MODEL_AI_THINKER
#define EEPROM_SIZE 1
#include "camera_pins.h"
#include "ESP32_MailClient.h"

#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

bool b;
long d;

/*const char* ssid = "yakuza";
const char* password = "01234567891";*/
const char* ssid = "yakuza";
const char* password = "01234567891";
SMTPData smtpData;
int nb_photo=0;
void startCameraServer();
void setup() {
  Serial.begin(115200);
```

CameraWebServer

```
Serial.setDebugOutput(true);
camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;
//init with high specs to pre-allocate larger buffers
if (psramFound()) {
  config.frame_size = FRAMESIZE_UXGA;
  config.jpeg_quality = 10;
  config.fb_count = 2;
} else {
  config.frame_size = FRAMESIZE_SVGA;
  config.jpeg_quality = 12;
  config.fb_count = 1;
}
// camera init
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK) {
  Serial.printf("Camera init failed with error 0x%x", err);
  return;
}
if(!SD_MMC.begin()){
   Serial.println("SD Card Mount Failed");
   return;
}
```

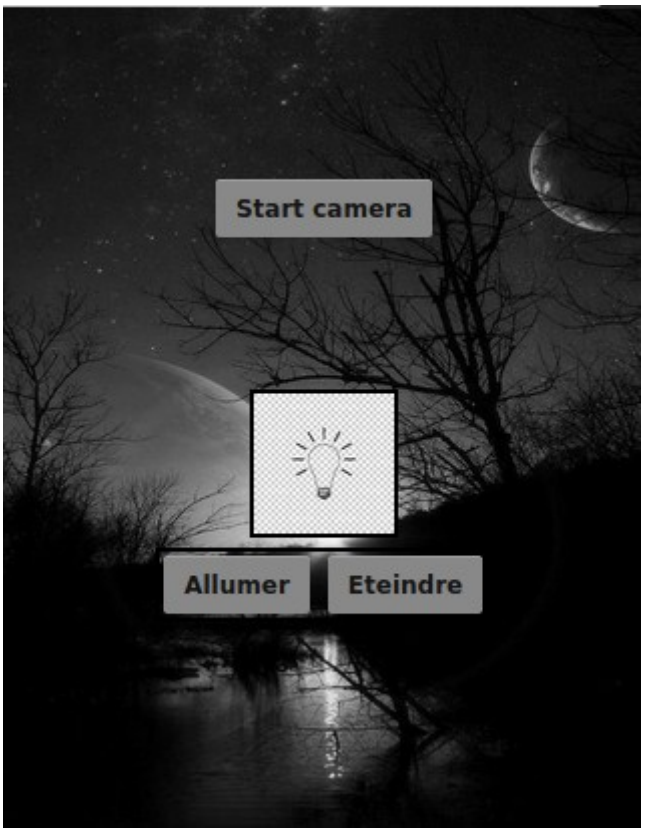
CameraWebServer

```
  }
  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  if(!SD_MMC.begin()){
     Serial.println("SD Card Mount Failed");
     return;
  }
/*  uint8_t cardType = SD_MMC.cardType();
  if(cardType == CARD_NONE){
   Serial.println("No SD Card attached");
   return;
  }*/
  sensor_t * s = esp_camera_sensor_get();
  //initial sensors are flipped vertically and colors are a bit saturated
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1);//flip it back
    s->set_brightness(s, 1);//up the blightness just a bit
    s->set_saturation(s, -2);//lower the saturation
  }
  //drop down frame size for higher initial frame rate
  s->set_framesize(s, FRAMESIZE_QVGA);


  //CONNEXION AU RESEAU WIFI
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");


    //esp_deep_sleep_start();
    // delay(2000);

 //LE CODE SUIVANT EST POUR VISIONNER LA VIDEO VIA L'ADRESSE IP FOURNI
 startCameraServer();
  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  // Serial.println("' to connect");


}
void loop() {

  delay(1000);
}
```

Etape 4: développement de l’application

Dans cette partie du projet, nous avons utilisé l’application APP INVENTOR pour développer notre application ‘**SMART MB**’,

Dans notre application, nous pouvons consulter la boite aux lettres à tout moment en utilisant l’adresse IP de la camera. Mais voir le contenu de la boite dans le noir ne sert à rien, pour cela nous avons rajouté deux boutons qui permettent d’allumer ou d’éteindre la lampe qui est bien insérée dans la boite.



Problèmes rencontrés

Tout n'était pas aussi facile que nous l'avions imaginé. Au cour de ce projet, nous avons réalisé énormément de progrès, mais aussi nous ne pouvons pas nier le fait d'avoir beaucoup de difficultés a savoir :

- **Configuration de l'esp32**
- **Connecter la caméra sur la carte uno**

Pour cela, nous avons acheté une autre camera : ESP_CAM avec module FTDI 5V, 3.5v associé, convertisseur USB à TTL.

- **Envoi d'un mail via WIFI**
- **Joindre une photo dans le mail envoyé.**
- **Développement de l'application Android**

Au début, nous avons commencé à apprendre à créer des applications mobiles en JAVA, mais c'est tout un chapitre à apprendre et malheureusement nous n'avons pas le temps suffisant pour cela

Donc nous avons opté pour une autre méthode : utiliser le logiciel APP INENTOR développé par Google qui simplifie le développement des applications Android et les rendre accessibles même pour ceux qui ne sont pas familiers avec les langages de programmations.

- **Problème de transport**

  C'est pour cette raison que nous n'avons pas pu finir la totalités des fonctions de notre projet, au début comme nous n'habitons pas à coté, l'université était fermée donc on ne pouvait pas se réunir pour travailler ensemble.

Conclusion

Ce projet a été une excellente occasion de découvrir le monde de l'internet des objets.

Nous avons réussi à transformer une boite aux lettres simple utilisée au quotidien à un système avec la même boite mais plus intelligente.

C’est à la fin de ce projet que nous réalisions le nombre de difficultés que nous avons surmonté, et toutes les choses que nous avons appris durant toutes nos recherches.

Cela était très enrichissant pour nous, c’est pour cette même raison que nous souhaitons progresser dans ce domaine plein d’avenir.