```
/* BOITE AUX LETTRES CONNECTEE
*   création Eric, Yola, Anne  - nov 2017
*   Le facteur glisse des lettres dans la boîte,
*   une voix enregistrée lui dit merci,
*   et un message nous parvient par courrier électronique, pour nous indiquer que nous avons du courrier  !
*   En rentrant à la maison, nous savons donc que nous avons quelque chose dans la boîte.

*/

// inclusion des librairies

// capteur de distance - a priori rien de spécifique

// connexion Wifi via ESP8266Wifi
#include <ESP8266WiFi.h>

// pour émettre le son Grove-Serial MP3 Player V2.0 - https://github.com/Seeed-Studio/Grove_Serial_MP3_Player_V2.0
#include <SoftwareSerial.h>
#include <MP3Player_KT403A.h>



// Déclaration des variables et constantes

// valeurs pour le WiFi
const char* ssid = "Mon_réseau_Wifi";       // ATTENTION c'est chez Eric
const char* password = "Mon_Mot_de_passe";  // ATTENTION c'est chez Eric

// valeurs pour le serveur Web
const char* host     = "maker.ifttt.com";

// Note pour le son : You must define a SoftwareSerial class object that the name must be mp3,
// but you can change the pin number according to the actual situation.

SoftwareSerial mp3(13, 15);  //  Rx pin3 et Tx pin1 cf. doc de l'ESP ou D9 et D10 physique sur l'ESP car là c les pin logiques equivalent arduino

// pour le capteur de distance

const int pinLightSensor = A0;
int SensorValue = 0;
const byte TRIGGER_PIN = D1; // Broche TRIGGER
const byte ECHO_PIN = D2;    // Broche ECHO
/* Constantes pour le timeout */
const unsigned long MEASURE_TIMEOUT = 25000UL; // 25ms = ~8m à 340m/s
/* Vitesse du son dans l'air en mm/us */
```

```
const float SOUND_SPEED = 340.0 / 1000;

// pour le reste à modifier selon la taille de la BàL
int d = 320;    // représente en mm la distance où, si un objet passe en-dessous, on considère qu'on a reçu une lettre



void setup() {
// phase d'initialisation
// initialisation du capteur de distance

Serial.begin(115200);

/* Initialise les broches */
pinMode(TRIGGER_PIN, OUTPUT);
digitalWrite(TRIGGER_PIN, LOW); // La broche TRIGGER doit être à LOW au repos
pinMode(ECHO_PIN, INPUT);


// Initialisation pour la carte son

mp3.begin(9600);
delay(100);

SelectPlayerDevice(0x02);       // SD card comme player par défaut
SetVolume(0x1E);                // définit le volume, the range is 0x00 to 0x1E.

// Initialisation de la communication série

Serial.begin(115200);
Serial.print("Connexion au WiFi ");
Serial.println(ssid);

WiFi.begin(ssid, password);    // On se connecte

while (WiFi.status() != WL_CONNECTED) {
// On attend
delay(500);
Serial.print(".");
}
Serial.println("");  // on affiche les paramètres
Serial.println("WiFi connecté");
Serial.print("Adresse IP du module ESP: ");
Serial.println(WiFi.localIP());
Serial.print("Adresse IP de la box : ");
Serial.println(WiFi.gatewayIP());
```

```
}

void loop() {
// corps du programme

distance (); //appelle la fonction distance - à revoir, je ne suis pas sure de la syntaxe


//si la distance est inférieur à d
// vérifier la valeur, on serait plutôt à 30 mm

if(SensorValue < d) {
// print out the value you read:
Serial.print("Alerte :  ");
Serial.println(SensorValue);
delay(500);
envoimail();  // alors on envoie la requête ifttt
Serial.println("Message début  ");
SpecifyMusicPlay(1);  // lit a première piste de la carte SD
Serial.println("Message fin  ");
}
}



void distance()
{
/* 1. Lance une mesure de distance en envoyant une impulsion HIGH de 10µs sur la broche TRIGGER
*/
digitalWrite(TRIGGER_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIGGER_PIN, LOW);

/* 2. Mesure le temps entre l'envoi de l'impulsion ultrasonique et son écho (si il existe) */
long measure = pulseIn(ECHO_PIN, HIGH, MEASURE_TIMEOUT);

/* 3. Calcul la distance à partir du temps mesuré */
float distance_mm = measure / 2.0 * SOUND_SPEED;

/* Affiche les résultats en mm, cm et m */
Serial.print(F("Distance: "));
Serial.print(distance_mm);
Serial.print(F("mm ("));
Serial.print(distance_mm / 10.0, 2);
Serial.print(F("cm, "));
Serial.print(distance_mm / 1000.0, 2);
Serial.println(F("m)"));
SensorValue = distance_mm;
```

```
/* Délai d'attente pour éviter d'afficher trop de résultats à la seconde */
delay(200);
}

void envoimail()
{
SpecifyMusicPlay(1);  // lit a première piste de la carte SD
Serial.print("Connexion au serveur : ");
Serial.println(host);
// On se place dans le rôle ndu  client en utilisant WifiClient
WiFiClient client;
// le serveur Web attend tradionnellement sur le port 80
const int httpPort = 80;
// Si la connexio échoue ca sera pour la prochaine fois
if (!client.connect(host, httpPort)) {
Serial.println("connection failed");
return;
}

// La connexion a réussie on forme le chemin
String url = String("/trigger/test_mailbox/with/key/Ma_Key_Perso_IFTTT");
Serial.print("demande URL: ");
Serial.println(url);
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
   "Host: " + host + "\r\n" +
   "Connection: close\r\n\r\n");

// On attend 10 secondes
delay(10000);
// On lit les données reçues, s'il y en a
while(client.available()){
String line = client.readStringUntil('\r'); // découpe ligne par ligne
Serial.print(line);
}
// plus de données
Serial.println();
Serial.println("connexion fermée");
}
```