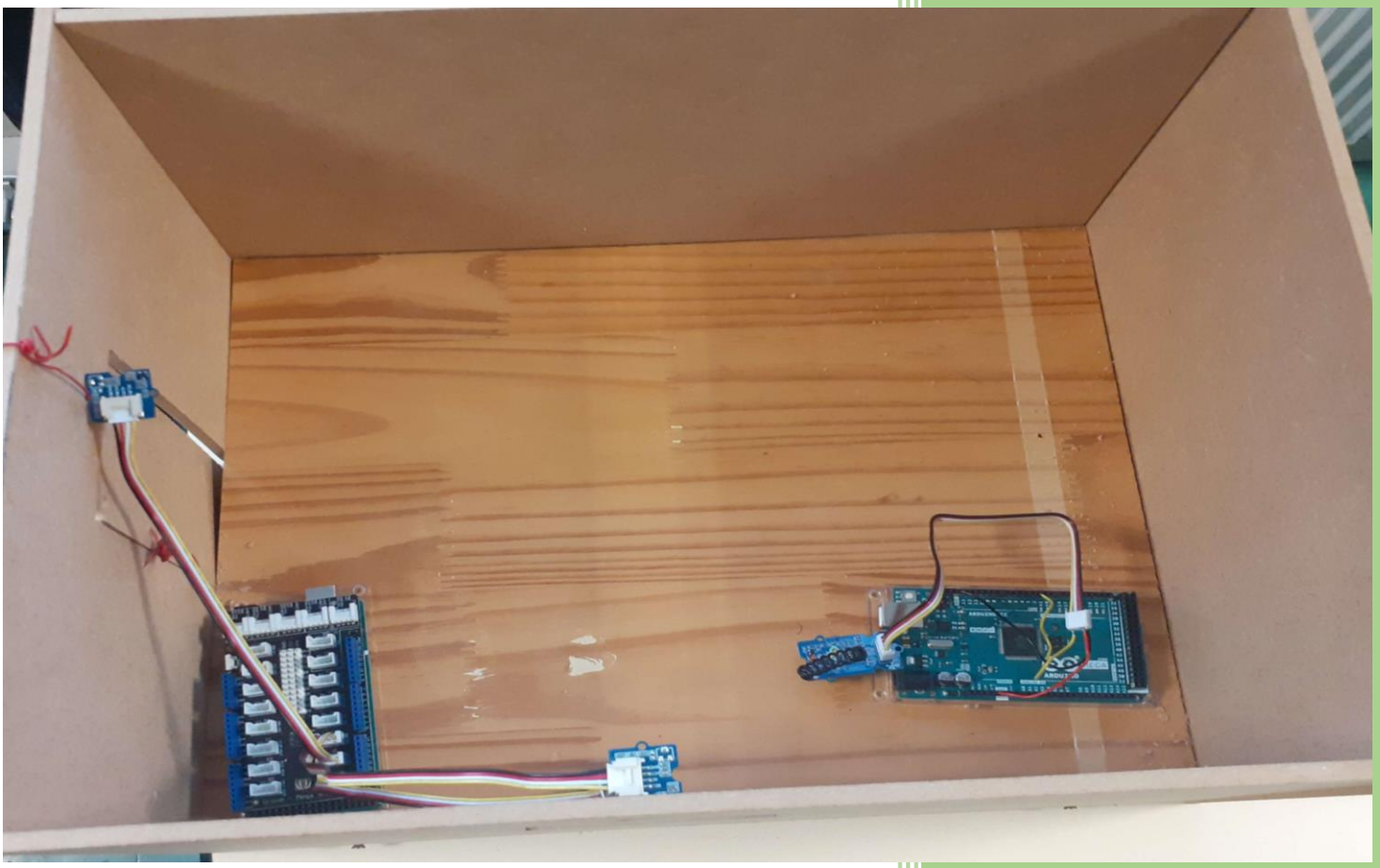
THIAM Mohammed

TOGBOSSI Kevin

SAE 5

Réalisation d'un système d'alarme domestique



THIAM Mohamed

TOGBOSSI Kevin

A-02

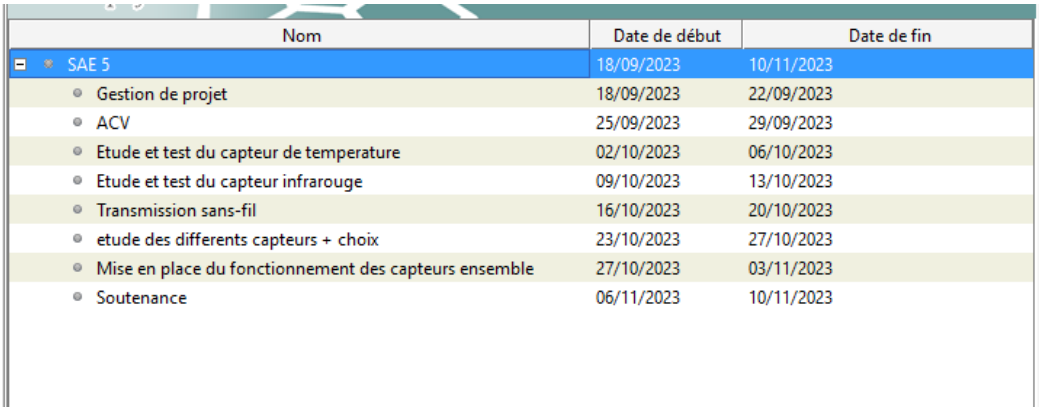
Sommaire :

Introduction

Le projet a pour but de réaliser une maquette de maison, avec un système d'alarme domestique avec des cartes Arduino et différents capteurs.

Pour réaliser ce projet nous allons nous organiser en faisant un gantt.

Gantt prévisionnel :



Nom	Date de début	Date de fin
SAE 5	18/09/2023	10/11/2023
Gestion de projet	18/09/2023	22/09/2023
ACV	25/09/2023	29/09/2023
Etude et test du capteur de temperature	02/10/2023	06/10/2023
Etude et test du capteur infrarouge	09/10/2023	13/10/2023
Transmission sans-fil	16/10/2023	20/10/2023
etude des differents capteurs + choix	23/10/2023	27/10/2023
Mise en place du fonctionnement des capteurs ensemble	27/10/2023	03/11/2023
Soutenance	06/11/2023	10/11/2023

I- Etude des capteurs

Pour réaliser nous allons essayer plusieurs capteurs afin de déterminer lesquelles prendre pour réaliser notre système d'alarme.

1) Capteur de mouvement

Pour le capteur de mouvement nous allons utiliser le Grove – mini PIR Motion Sensor. Le capteur permet de détecter généralement un mouvement humain dans sur une portée d'un peu prés 2 mètres.

Nous allons utiliser deux capteurs que nous avons câblé sur la broche digital 2 pour celui de la porte et la broche digital 4 pour celui de la fenêtre. Ensuite, nous avons pris le programme de base du capteur puis nous l'avons modifié pour afficher les différentes détections sur le moniteur série (**Annexe *1* : Programme capteur de mouvement**)

Bilan : le capteur fonctionne et détecte bien un mouvement. On remarque que le capteur quand il détecte un mouvement reste à l'état haut pendant 5 secondes avant de redescendre dans sa position basse. Le capteur est adapté pour notre projet.

2) Capteur de vibration

Nous allons utiliser le capteur Grove - Piezo Vibration Sensor. Le capteur de vibrations Grove-Piezo convient aux mesures de flexibilité, de vibration, d'impact et de toucher. Le module est basé sur le capteur de film PZT LDT0-028. Lorsque le capteur se déplace d'avant en arrière, une certaine tension sera générée par le comparateur de tension à l'intérieur de celui-ci. Une large plage dynamique (0,001 Hz ~ 1 000 MHz) garantit d'excellentes performances de mesure.

Nous avons câblé le capteur sur la broche digital 2 puis nous avons repris le programme de la détection du mouvement en le modifiant pour afficher la détection si le capteur détecte une vibration ou non (**Annexe *2* : Programme pour le capteur de vibration.**).

Bilan : Le capteur ne détecte pas les vibrations et détecte uniquement des impacts de forte intensité. Le capteur ne sera pas adapté pour notre projet car la porte ne va pas émettre un gros impact sur le capteur lors de son ouverture.

3) Capteur d’air

Pour le capteur d’air nous allons utiliser le capteur grove air quality sensor V1.3. Ce capteur est conçu pour une surveillance complète de la climatisation intérieure. Il réagit à un large éventail de gaz nocifs, tels que le monoxyde de carbone, l'alcool, l'acétone, les diluants, le formaldéhyde. En raison du mécanisme de mesure, ce capteur ne peut pas produire de données spécifiques pour décrire quantitativement les concentrations des gaz cibles.

Nous avons câblé le capteur d’air sur la broche analogique A0 puis nous avons pris le programme de base du capteur avec sa bibliothèque (**Annexe *3* : Programme du capteur air**).

Bilan : Le capteur d’air fonctionne. On remarque qu’il est impossible d’avoir de l’air pollué avec le matériel qu’on a la valeur reste dans la zone de l’air propre avec une valeur entre 20 et 30. La valeur baisse quand le capteur détecte du vent et monte quand il n’y a pas de vent. Cependant le capteur est toujours à la même valeur même si on le met dans une boite. Le capteur n’est pas adapté à notre projet car vu que la maison n’est pas totalement hermétique l’air extérieur est la même que celle de l’intérieur de la maison.

4) Capteur de son

Le capteur Grove - Sound Sensor peut détecter l'intensité sonore de l'environnement. Le composant principal du module est un microphone basé sur l'amplificateur L358 et un microphone à électret.

Nous avons câblé le capteur de son sur la broche analogique A0 puis nous avons pris le programme de base du capteur (**Annexe *4* : Programme du capteur de son**).

Bilan : Le capteur fonctionne. On remarque quand on met un son ou une voix sur le microphone le capteur varie entre 130 et 180. Cependant le capteur à une trop grande sensibilité et fait des pics à supérieur à 130 qui rend le capteur pas très fiable. Le capteur n'est pas adapté pour notre projet.

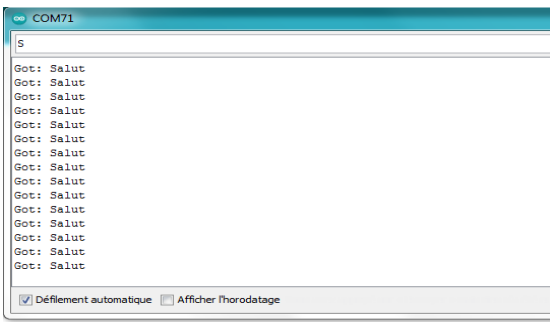
II- Transmission sans fil

1) Grove - 433MHz Simple RF Link Kit

Pour la transmission et la réception sans fil nous allons utiliser une transmission FM avec le Grove - 433MHz Simple RF Link Kit. Les modules émetteur et récepteur reposent tous deux sur un seul fil pour la communication. Il faut d'utiliser la bibliothèque VirtualWire qui utilise Amplitude Shift Keying pour une modulation qui offre une meilleure communication.

Pour faire la transmission et la réception on utilise deux cartes Arduino Méga où on câble le récepteur sur la broche digital d2 sur l'une et sur l'autre la transmission sur la broche digital d2.

Bilan : La communication fonctionne bien, on arrive à émettre un message avec le programme de test fournie (**Annexe *5*** : **Programme de la transmission et de la réception 433 Mhz**).



2) Grove - Long Range 868MHz

Le Grove - Long Range 868MHz est le même que le Grove 433 Mhz avec une meilleure distance de transmission et de réception.

Pour faire la transmission et la réception on utilise deux cartes Arduino Méga où on câble le récepteur sur la broche digitale 10 sur l'une et sur l'autre la transmission sur la broche digitale digital 10. On utilise le programme de base fourni avec la bibliothèque « RH RF495 » qu'on va modifier pour effectuer les tests de communication (***Annexe 6 : Programme du Grove long range 868 MHz***).

Choix du moyen de transmission :

On va utiliser le Grove 433 MHz Simple RF Link Kit car il n'y avait plus assez de Grove Lora Long range 868 MHz

III- Réalisation du projet

1) Choix des capteurs + bon de commandes

Pour réaliser le projet d'un système d'alarme domestique nous allons choisir 2 capteurs de mouvement qui sont les plus adapté à notre projet selon nos tests effectuer précédemment. Nous allons les placer au niveau de la porte et de la fenêtre. Pour la communication nous allons utiliser le Grove 433 MHz.

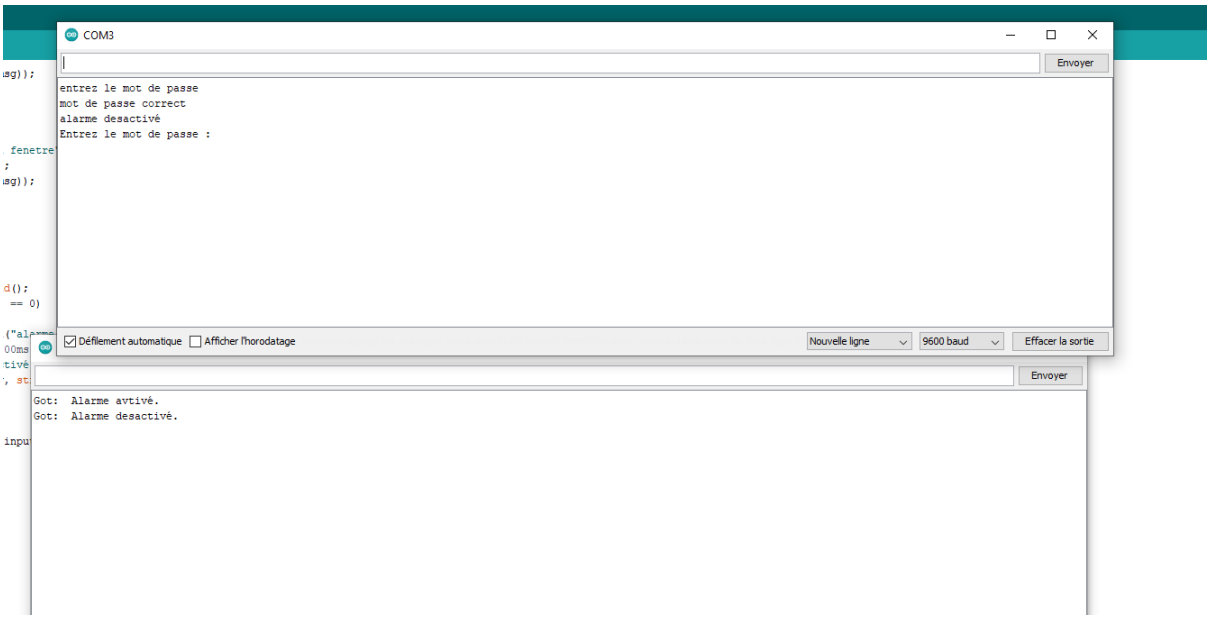
On va câbler le capteur de mouvement de la porte sur la broche digital 4 et le capteur de mouvement sur la broche digital 6 sur une carte Arduino Méga 2560. On va utiliser un module Grove Méga Shield qui est une carte d'interface permettant de raccorder facilement les capteurs de mouvement. Pour la transmission on va utiliser une deuxième carte Arduino Méga 2560 où on va câbler l'émetteur sur la carte Arduino Méga où il y a les capteurs de mouvements sur la broche digital 2 et le récepteur sur la deuxième carte Arduino Méga sur la broche digital 2.

Nous allons mettre en place un programme pour que quand l'utilisateur part de la maison, il rentre un mot de passe pour activer l'alarme et quand il revient il rentre un mot de passe pour le désactiver (**Annexe 7 : Programme mot de passe**).

Test du programme mot de passe :

Pour activer l'alarme on entre le mot de passe brown dans le moniteur série.

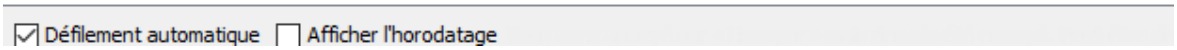
Pour désactiver l'alarme on entre le mot de passe « desac » dans le moniteur série.

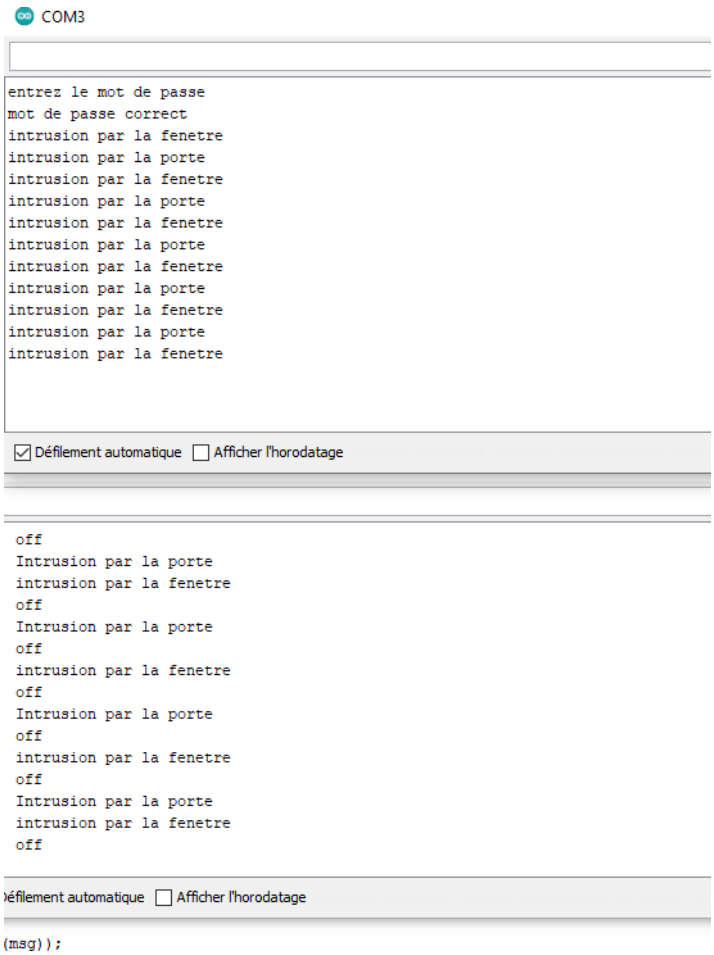


Nous allons rajouter au programme quand l'alarme sera activée les détections des capteurs de mouvements afin de prévenir l'utilisateur quand il y a une intrusion chez lui (***Annexe 8 : Programme intrusion dans la maison***).

Test du programme :

```
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
intrusion par la fenetre
```





Bon de commande

2 capteurs de mouvements : 6,14 euros l'unité soit 12.28 euros les deux

1 capteur d'air : 8.70 euros

2 Arduino Méga 2560 : 42 euros l'unité soit 84 euros les deux

2 Grove Méga Shield : 10 euros l'unité soit 20 euros les deux

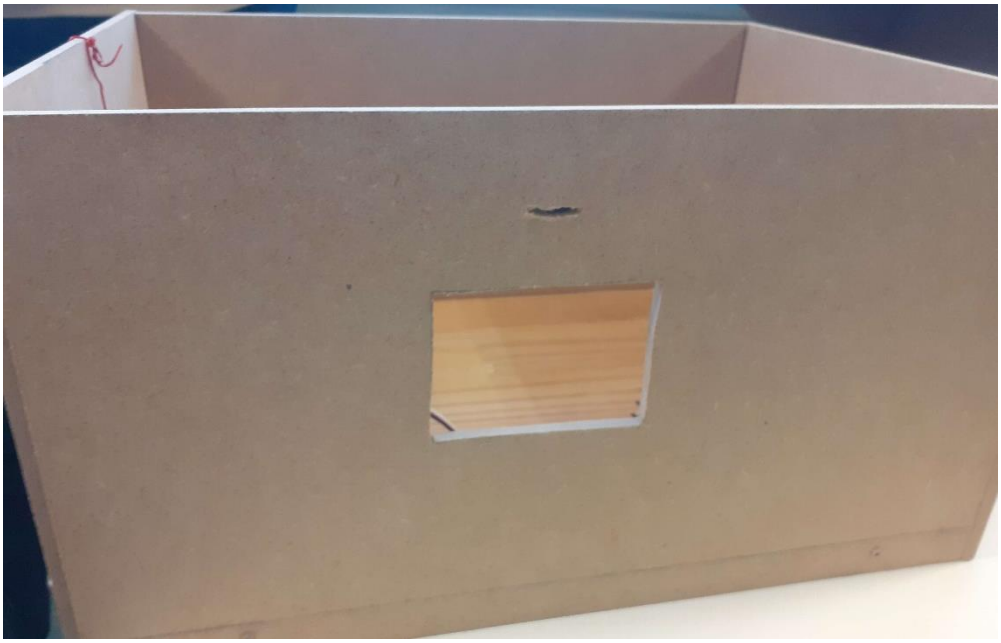
1 Grove Long range 868 MHz: 12 euros

Total: 136.98 euros

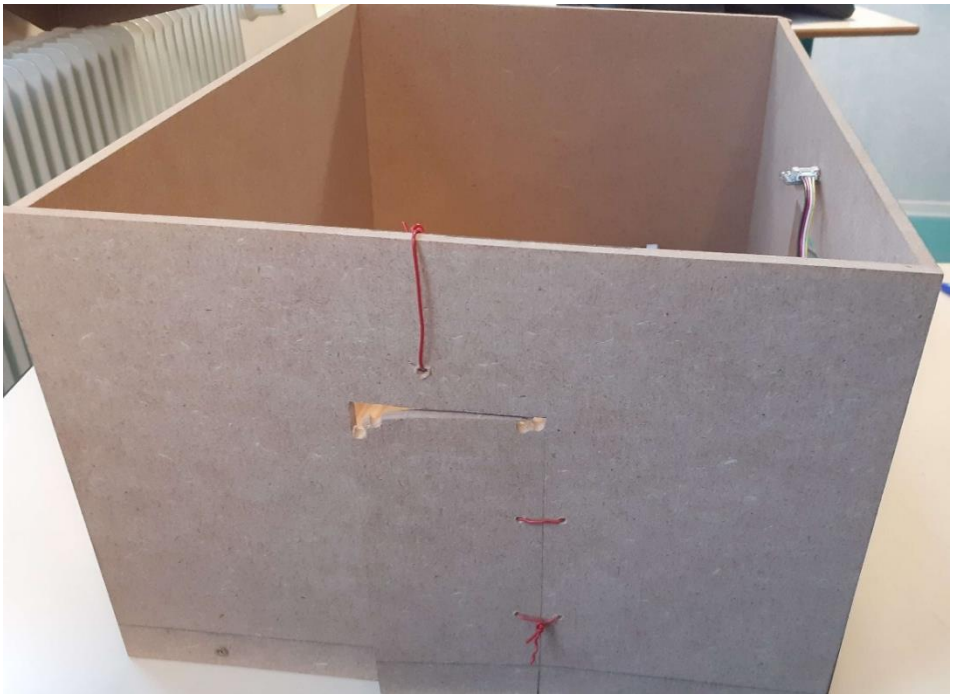
2) Maquette

Nous avons fabriqué une maison en maquette avec des planches en bois pour les 4 murs et une planche en pin Massif pour le sol. Nous avons eu pour les dimensions de la maison 30 cm de largeur et 40 cm de longueur et 30 cm de hauteur. Pour la stratégie de l'emplacement des capteurs, nous avons mis un capteur de mouvement en haut de la porte et un autre capteur de mouvement en haut de la fenêtre pour détecter les intrusions dans les 2 issues de pénétration de la maison quand l'alarme est activée.

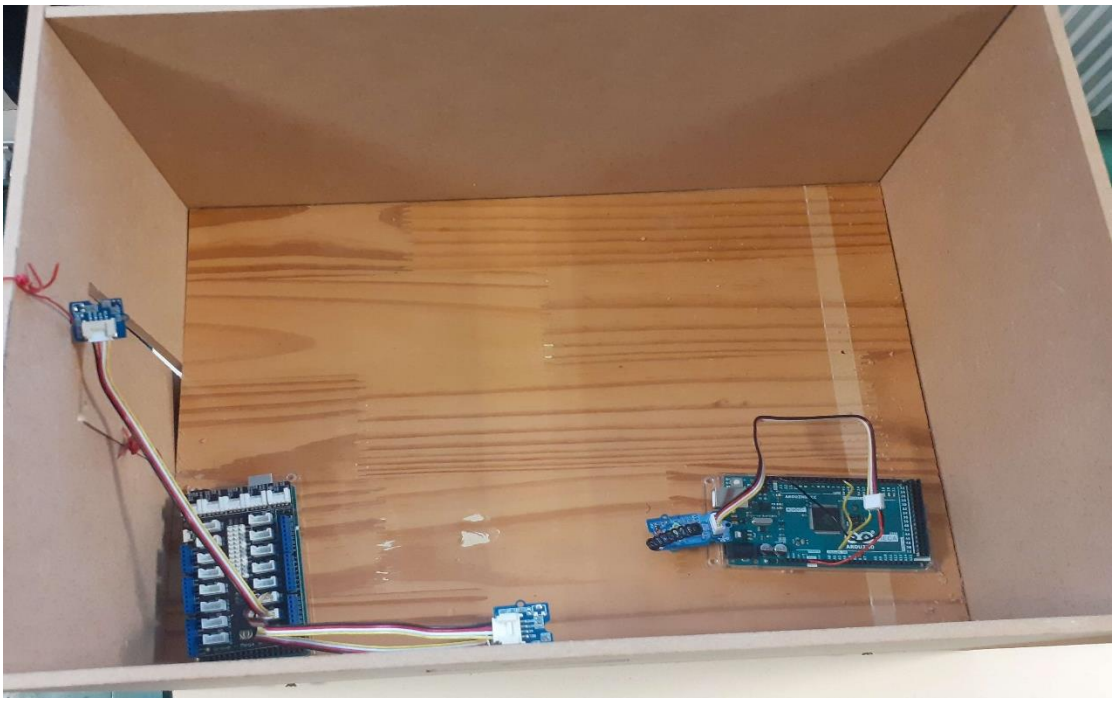
Vue extérieur côté fenêtre :



Vue extérieur côté porte :



Vue intérieure avec les emplacements des capteurs :



IV- ACV

1) Critères généraux

L'Analyse du Cycle de Vie (ACV) est une méthodologie utilisée pour évaluer l'impact environnemental d'un produit, d'un service ou d'un processus tout au long de son cycle de vie, de la production des matières premières à la fin de vie. Les critères généraux qui guident une ACV sont les suivants :

1) Délimitation du système : Identifier clairement les limites du système étudié, en déterminant quelles étapes du cycle de vie seront prises en compte. Cela peut inclure l'extraction des matières premières, la production, la distribution, l'utilisation et la fin de vie.
2) Inventaire des flux : Collecter toutes les données pertinentes sur les entrées (matières premières, énergie, etc.) et les sorties (émissions, déchets, etc.) du système étudié. Ces données peuvent être quantitatives ou qualitatives.
3) Classification des impacts : Catégoriser les impacts environnementaux potentiels en fonction de critères tels que les émissions de gaz à effet de serre, la consommation d'énergie, la pollution de l'air, de l'eau et du sol, la perte de biodiversité, etc.
4) Évaluation des impacts : Évaluer l'importance relative des différents impacts environnementaux en utilisant des indicateurs spécifiques. Certains impacts peuvent être pondérés plus fortement en raison de leur gravité ou de leur importance environnementale.
5) Interprétation des résultats : Analyser et interpréter les résultats de l'inventaire et de l'évaluation des impacts pour fournir des informations significatives aux parties prenantes. Cela peut inclure des comparaisons avec d'autres produits ou systèmes similaires.
6) Sensibilité et incertitude : Identifier et évaluer la sensibilité des résultats aux variations des données d'entrée et des méthodes utilisées. Reconnaître les incertitudes inhérentes à l'ACV et fournir des informations sur la fiabilité des résultats.
7) Communication des résultats : Présenter les résultats de manière claire et compréhensible pour un public varié, en utilisant des graphiques, des tableaux et des rapports. La transparence et l'accessibilité sont essentielles pour assurer une communication efficace.
8) Amélioration continue : Utiliser les résultats de l'ACV pour identifier des opportunités d'amélioration environnementale tout au long du cycle de vie du produit, du service ou du système. Encourager l'innovation et l'adoption de pratiques plus durables.

2) Critères prise en compte

L'Analyse du Cycle de Vie (ACV) du système d'alarme domestique avec maquette en bois et capteurs Grove – mini PIR Motion Sensor, utilisant le Grove - 433MHz Simple RF Link Kit et deux cartes Arduino méga 2560, permet d'évaluer l'impact environnemental tout au long du cycle de vie du produit.

1. Production des composants électroniques :

La fabrication des capteurs de mouvements Grove – mini PIR Motion Sensor et du Grove - 433MHz Simple RF Link Kit implique l'extraction des matériaux, la production des circuits imprimés, et l'assemblage des composants électroniques. La fabrication des cartes Arduino méga 2560 suit un processus similaire. L'empreinte carbone de cette phase dépend des pratiques de fabrication et de l'utilisation de matériaux respectueux de l'environnement.

2. Production des éléments en bois :

La production des planches en bois pour les murs (40 cm x 30 cm x 30 cm) et la planche en pin massif pour le sol implique l'abattage des arbres, la transformation du bois, et le transport. L'utilisation de bois provenant de sources durables peut réduire l'impact environnemental de cette phase.

3. Assemblage du système :

L'assemblage du système comprend la construction de la maquette en utilisant les planches en bois et l'intégration des capteurs de mouvements, du kit RF Link, et des cartes Arduino. Cette étape peut générer des déchets et consommer de l'énergie. L'efficacité du processus d'assemblage peut influencer l'empreinte carbone totale.

4. Utilisation du système :

Pendant la phase d'utilisation, le système d'alarme est alimenté en électricité. Les capteurs de mouvement et les cartes Arduino consomment de l'énergie régulièrement. L'impact environnemental dépend de la source d'énergie utilisée. L'utilisation de composants écoénergétiques peut contribuer à réduire la consommation électrique.

5. Fin de vie du système :

La fin de vie du système implique le remplacement ou la mise au rebut des composants électroniques obsolètes et la gestion des éléments en bois. Le recyclage des composants électroniques et le traitement responsable du bois en fin de vie sont cruciaux pour minimiser l'impact environnemental.

6. Scénarios d'amélioration :

Utilisation de matériaux électroniques recyclés ou à faible impact environnemental.

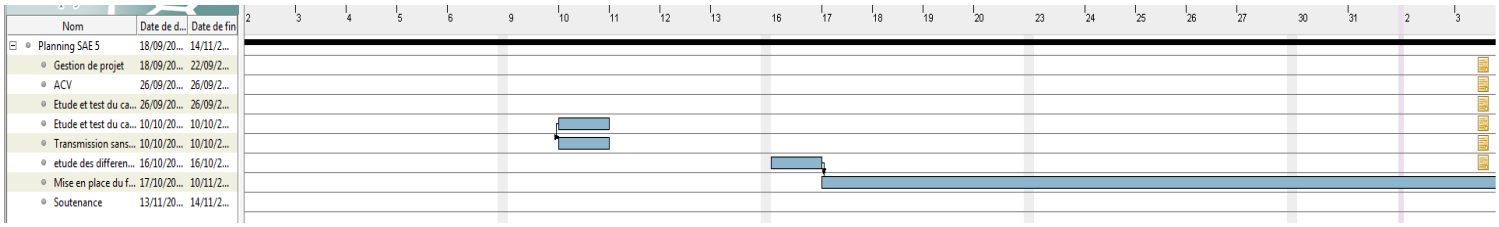
Incorporation de technologies écoénergétiques pour réduire la consommation électrique.

Utilisation de bois certifié FSC (Forest Stewardship Council) pour assurer des pratiques forestières durables.

V- Conclusion

Pour conclure lors de la mise en œuvre du projet système d'alarme domestique avec notre programme final (***Annexe 9 : Programme final***) et l'installation des capteurs nous avons observé que tout fonctionne correctement. Cependant notre programme peut être améliorer en ajoutant une attente au moment de l'activation de l'alarme quand une personne rentre pour le laisser le temps de rentrer le mot de passe.

Gantt réel :



Analyse des écarts :

Comparer à notre Gantt Prévisionnel vu dans l'introduction (**Introduction)** on a eu des complications avec une semaine qu'on n'a pas eu à cause du COVID du 2 octobre au 6 octobre qui nous a fait déplacer tout d'une semaine. Sinon le planning mis en place a été respecter.

VI- Annexes

```
int porte = digitalRead(2); // capteur de mouvement cabler sur le port digital 2
int fenetre = digitalRead(4); // capteur de mouvement cabler sur le port digital 4
void setup()
{
  Serial.begin(9600);
}
void loop()
{
 capteur();

}
void capteur()
{
    Serial.println(porte);
    Serial.println(fenetre);
    if(porte == HIGH)
    {
      Serial.println("presence a la porte");
      alarme();
    }

    else
    {
      Serial.println("aucune présence a la porte");
    }
    delay(200);
    if(fenetre == HIGH )
    {
      Serial.println("presence a la fenetre");
    }
    else
    {
      Serial.println("aucune presence a la fenetre");
    }
    delay(200);
  }
```

Annexe 1 : Programme capteur de mouvement

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int porte = digitalRead(2); // Capteur câblé sur la broche digital 2
  Serial.println(porte); // retourne la valeur du capteur sur le moniteur serie
  delay(200);

  if(porte == HIGH)
  {
    Serial.println("intrusion par la porte");
  }
  else
  {
    Serial.println("aucune présence a la porte");
  }
    delay(200);

```

Annexe 2 : Programme pour le capteur de vibration.

```ino
#include"AirQuality.h"
#include"Arduino.h"
AirQuality airqualitysensor;
int current_quality =-1; // variable mis à -1 qui correspond au capteur éteint ou un non fonctionnement du capteur

ISR(TIMER1_OVF_vect)
{
  if(airqualitysensor.counter==61)// timer de 2 seconde pour lancer l'initialisation du capteur
  {

    airqualitysensor.last_vol=airqualitysensor.first_vol;
    airqualitysensor.first_vol=analogRead(A0);
    airqualitysensor.counter=0;
    airqualitysensor.timer_index=1;
    PORTB=PORTB^0x20;
  }
  else
  {
    airqualitysensor.counter++;
  }
}

void setup()
{
    Serial.begin(9600);
    airqualitysensor.init(14);
}
void loop()
{
    current_quality=airqualitysensor.slope();
    if (current_quality >= 0)// si le capteur fonctionne
    {
        if (current_quality==0)
            Serial.println("Forte pollution force le signal à s'activer");
        else if (current_quality==1)
            Serial.println("Forte pollution!");
        else if (current_quality==2)
            Serial.println("basse pollution!");
        else if (current_quality ==3)
            Serial.println("Air propre");
    }
}
```

Annexe 3 : Programme du capteur air

```
int loudness;
void setup()
{
    Serial.begin(9600);
}

void loop()
{
    loudness = analogRead(0); // capteur câbler sur la broche A0
    Serial.println(loudness);
    delay(1000);
    if(loudness > 130)
    {
      Serial.println("la porte est ouverte");
    }
}
```

Annexe 4 : Programme du capteur de son

```
#include <VirtualWire.h>

int RF_TX_PIN = 2;

void setup()
{
  vw_set_tx_pin(RF_TX_PIN); // Setup transmit pin
  vw_setup(2000); // Transmission speed in bits per second.
}

void loop()
{
  const char *msg = "salut";
  vw_send((uint8_t *)msg, strlen(msg));  // Send 'hello' every 400ms.
  delay(400);

}
```

Transmission

```
#include <VirtualWire.h>

  int RF_RX_PIN = 2;

  void setup()
  {
    Serial.begin(9600);
    Serial.println("setup");
    vw_set_rx_pin(RF_RX_PIN);  // Initialise la broche D2 .
    vw_setup(2000); // la vitesse de transmission.
    vw_rx_start(); // lance la réception.
  }

  void loop()
  {
    uint8_t buf[VW_MAX_MESSAGE_LEN];
    uint8_t buflen = VW_MAX_MESSAGE_LEN;
    if(vw_get_message(buf, &buflen))
    {
      int i;
      // renvoi d'un message de reçu
      Serial.print("reçu ");
      for(i = 0; i < buflen; ++i)
      {
        Serial.print(buf[i], HEX);
        Serial.print(" ");
      }
      Serial.println("");
    }
  }
```

Réception

Annexe 5 *:* **Programme de la transmission et de la réception 433 Mhz**

L’émetteur :

```
#include <RH_RF95.h>
// Initialisation des ports
#ifdef __AVR__
    #include <SoftwareSerial.h>
    SoftwareSerial SSerial(10, 11); // RX, TX
    #define COMSerial SSerial
    #define ShowSerial Serial

    RH_RF95<SoftwareSerial> rf95(COMSerial);
#endif
// Initialisation de l'émetteur
void setup() {
    ShowSerial.begin(9600);
    ShowSerial.println("RF95 client test.");

    if (!rf95.init()) {
        ShowSerial.println("init failed");
        while (1);
    }


    rf95.setFrequency(868.0);
}

void loop() {
    ShowSerial.println("en cours d'envoi au recepteur ");
    // envoie un message sur le récepteur
    uint8_t data[] = "ALARME : INTRUSION DANS LA MAISON !";
    rf95.send(data, sizeof(data));

    rf95.waitPacketSent();

    delay(1000);
}
```

Recepteur :

```
#include <RH_RF95.h>
// initialisation des ports
#ifdef __AVR__
    #include <SoftwareSerial.h>
    SoftwareSerial SSerial(10, 11); // RX, TX
    #define COMSerial SSerial
    #define ShowSerial Serial
    RH_RF95<SoftwareSerial> rf95(COMSerial);
#endif
// Initialisation du récépteur
void setup() {
    ShowSerial.begin(9600);
    ShowSerial.println("RF95 server test.");

    if (!rf95.init()) {
        ShowSerial.println("init failed");
        while (1);
    }
    rf95.setFrequency(868.0);
}

void loop()
{
  // Condition de quand la réception est activé
    if (rf95.available())
    {
      // Reçoit le message de la transmission et l'affiche sur le moniteur serie
        uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
        uint8_t len = sizeof(buf);
        if (rf95.recv(buf, &len))
        {
            ShowSerial.print("got: ");
            ShowSerial.println((char*)buf);

        }
        else
        {
            ShowSerial.println("erreur de récepetion");
        }
    }
}
```

Annexe 6 : Programme du Grove long range 868 MHz

```
#include <VirtualWire.h>
#include"Arduino.h"
int RF_TX_PIN = 2;
const char *mdp = "brown"; // Définissez votre mot de passe
const char *mdp2 = "desac"; // Définnisez votre mot de passe
char input_mdp[20]; // Une chaîne pour stocker le mot de passe entré par l'utilisateur
char input_mdp2[20]; // Une chaîne pour stocker le mot de passe entré par l'utilisateur
int mdpIndex = 0; // Index pour parcourir la chaîne du mot de passe
int mdpIndex2 = 0; // Index pour parcourir la chaîne du mot de passe
const char *msg;
int x = 1;

void setup()
{
  vw_set_tx_pin(RF_TX_PIN); // Initialisation du PIN trasmission
  vw_setup(2000); // vitesse de transmission.
  Serial.begin(9600);
}

void loop()
{
if(x== 1) // condition qui permet de faire des étapes dans le programme
  {
   Serial.println("entrez le mot de passe"); // demande à l'utilisateur de rentrer son mot de passe
    x=2; // passe à l'étape suivante
  }
   while (Serial.available() > 0 ) // Tant que le moniteur série est ouvert
    {
     char inputChar = Serial.read(); // Lis le mot de passe entrer par l'utilisateur
      if (inputChar == '\n' || inputChar == '\r')
      { // Si la chaine de caractére fini par le caractére Entrée ou un espace
       input_mdp[mdpIndex] = '\0'; // Ajoutez un caractère nul pour terminer la chaîne
       mdpIndex = 0; // Réinitialisez l'index du mot de passe
       // Vérifiez si le mot de passe entré correspond au mot de passe défini
       if (strcmp(input_mdp, mdp) == 0) // Condition pour comparer les caracteres du mot de passe entrer ave
       {
         Serial.println("mot de passe correct");
             msg = " Alarme avtivé."; // envoie Alarme activé au récépteur
         vw_send((uint8_t *)msg, strlen(msg));
             while (Serial.available() > 0) // Pour désactiver l'alarme l'utilisateur rentre un mot de passe
               {
                 char inputChar2 = Serial.read(); // Lis le mot de passe entrer par l'utilisateur
                 if (strcmp(input_mdp2, mdp2) == 0)
                   {
                             Serial.println("alarme desactivé"); // envoie Alarme desactivé au récépteur
                             msg = " Alarme desactivé.";
                           vw_send((uint8_t *)msg, strlen(msg));Q
                           x=3; // passe à l'étape suivante
                   }
                   else {
                    input_mdp2[mdpIndex2] = inputChar2; // Ajoute caractère par caractére le mot de passe c
                  mdpIndex2++; }// incrémente la variable mdpIndex pour rentrer le prochain caractére
                  }
        }
         }
         else
         {
         Serial.println("Mot de passe incorrect. Réessayez.");
         }
        Serial.println("Entrez le mot de passe :"); // Demandez à l'utilisateur un nouveau mot de passe
       } else {
      // Ajoute caractère par caractére le mot de passe dans la table imput_mdp
      input_mdp[mdpIndex] = inputChar;
      mdpIndex++; // incrémente la variable mdpIndex pour rentrer le prochain caractére
}
```

Annexe 7 : Programme mot de passe

```
while (x==2 ) // Etape 2
 {
   int porte = digitalRead(4); // capteur de la porte câbler sur le port digital 4
   int fenetre = digitalRead(6); // capteur de la fenetre câbler sur le port digital 6
 if(porte == HIGH) // condition quand le capteur est à l'état haut
 {
   Serial.println("intrusion par la porte");
   msg = "Intrusion par la porte"; // envoie le message intrusion par la porte au récépteur
   vw_send((uint8_t *)msg, strlen(msg));
   delay(200);  // envoie le message toutes les 200 ms
 }
 if(fenetre == HIGH) // condition quand le capteur est à l'état haut
 {
   Serial.println("intrusion par la fenetre");
   msg = "intrusion par la fenetre"; // envoie le message intrusion par la fenetre au récépteur
   vw_send((uint8_t *)msg, strlen(msg));
   delay(200); // envoie le message toutes les 200 ms
 }
```

Annexe 8 : Programme intrusion dans la maison

Programme transmission :

```
#include <VirtualWire.h>
#include"Arduino.h"
int RF_TX_PIN = 2;
const char *mdp = "brown"; // Définissez votre mot de passe
const char *mdp2 = "desac"; // Définnisez votre mot de passe
char input_mdp[20]; // Une chaîne pour stocker le mot de passe entré par l'utilisateur
char input_mdp2[20]; // Une chaîne pour stocker le mot de passe entré par l'utilisateur
int mdpIndex = 0; // Index pour parcourir la chaîne du mot de passe
int mdpIndex2 = 0; // Index pour parcourir la chaîne du mot de passe
const char *msg;
int x = 1;

void setup()
{
  vw_set_tx_pin(RF_TX_PIN); // Initialisation du PIN trasmission
  vw_setup(2000); // vitesse de transmission.
  Serial.begin(9600);
}
```

```
void loop()
{
if(x== 1) // condition qui permet de faire des étapes dans le programme
  {
   Serial.println("entrez le mot de passe"); // demande à l'utilisateur de rentrer son mot de passe
    x=2; // passe à l'étape suivante
  }
   while (Serial.available() > 0 ) // Tant que le moniteur série est ouvert
    {
      char inputChar = Serial.read(); // Lis le mot de passe entrer par l'utilisateur
      if (inputChar == '\n' || inputChar == '\r')
      { // Si la chaine de caractére fini par le caractére Entrée ou un espace
       input_mdp[mdpIndex] = '\0'; // Ajoutez un caractère nul pour terminer la chaîne
       mdpIndex = 0; // Réinitialisez l'index du mot de passe
       // Vérifiez si le mot de passe entré correspond au mot de passe défini
       if (strcmp(input_mdp, mdp) == 0) // Condition pour comparer les caracteres du mot de passe entrer avec celui défini
       {
        Serial.println("mot de passe correct");
            msg = " Alarme avtivé."; // envoie Alarme activé au récépteur
        vw_send((uint8_t *)msg, strlen(msg));
         while (x==2 ) // Etape 2
          {
            int porte = digitalRead(4); // capteur de la porte câbler sur le port digital 4
            int fenetre = digitalRead(6); // capteur de la fenetre câbler sur le port digital 6
           if(porte == HIGH) // condition quand le capteur est à l'état haut
           {
            Serial.println("intrusion par la porte");
            msg = "Intrusion par la porte"; // envoie le message intrusion par la porte au récépteur
            vw_send((uint8_t *)msg, strlen(msg));
            delay(200);  // envoie le message toutes les 200 ms
           }
           if(fenetre == HIGH) // condition quand le capteur est à l'état haut
           {
            Serial.println("intrusion par la fenetre");
            msg = "intrusion par la fenetre"; // envoie le message intrusion par la fenetre au récépteur
            vw_send((uint8_t *)msg, strlen(msg));
            delay(200); // envoie le message toutes les 200 ms
           }

            while (Serial.available() > 0) // Pour désactiver l'alarme l'utilisateur rentre un mot de passe
              {
               char inputChar2 = Serial.read(); // Lis le mot de passe entrer par l'utilisateur
               if (strcmp(input_mdp2, mdp2) == 0)
                {
                            Serial.println("alarme desactivé"); // envoie Alarme desactivé au récépteur
                          msg = " Alarme desactivé.";
                        vw_send((uint8_t *)msg, strlen(msg));Q
                        x=1; // passe à l'étape suivante
                }
                else {
                 input_mdp2[mdpIndex2] = inputChar2; // Ajoute caractère par caractére le mot de passe dans la table imput_mdp
              mdpIndex2++; }// incrémente la variable mdpIndex pour rentrer le prochain caractére
              }
        }
         }
         else
         {
         Serial.println("Mot de passe incorrect. Réessayez.");
         }
        Serial.println("Entrez le mot de passe :"); // Demandez à l'utilisateur un nouveau mot de passe
       } else {
      // Ajoute caractère par caractére le mot de passe dans la table imput_mdp
      input_mdp[mdpIndex] = inputChar;
      mdpIndex++; // incrémente la variable mdpIndex pour rentrer le prochain caractére
}
```

Programme récepteur :

```
#include <VirtualWire.h>

    int RF_RX_PIN = 2;

    void setup()
    {
      Serial.begin(9600);
      Serial.println("setup");
      vw_set_rx_pin(RF_RX_PIN);  // Initialise la broche D2 .
      vw_setup(2000); // la vitesse de transmission.
      vw_rx_start(); // lance la réception.
    }

    void loop()
    {
      uint8_t buf[VW_MAX_MESSAGE_LEN];
      uint8_t buflen = VW_MAX_MESSAGE_LEN;
      if(vw_get_message(buf, &buflen))
      {
        int i;
        // renvoi d'un message de reçu
        Serial.print("reçu ");
        for(i = 0; i < buflen; ++i)
        {
          Serial.print(buf[i], HEX);
          Serial.print(" ");
        }
        Serial.println("");
      }
    }
```

Annexe 9 : Programme final