```
// Bibliotheque //
#include<IRremote.h>
#include<Ultrasonic.h>
#include<SoftwareSerial.h>

// Declaration variable //
#define capteur_presence 2 //Grove receiver infrared
#define emetteur_infra 3 //Grove emitter infrared
#define fdchaut 4 //Grove
#define fdcbas 5 //Grove
#define capteur_distance 6 //Grove PIR
#define BP_ouverture 7 //Grove button
#define BP_fermeture 8 //Grove button
#define commande_moteur_horaire 9 //Grove relay
#define commande_moteur_horaire2 11 //Grove relay
#define commande_moteur_antihoraire 10 //Grove relay
#define commande_moteur_antihoraire2 12 //Grove relay
#define alarme_visuel 14 //Grove LED //Broche A0
#define alarme_sonore 15 //Grove Buzzer //Broche A1
#define detecteur_mur_fond 16 //Grove Ultrasonic
//Broche A2
#define RX 17 //Grove bluetooth //Broche A3
#define TX 18 // Grove bluetooth //Broche A4

SoftwareSerial mySerial(RX,TX);

Ultrasonic ultrasonic(capteur_distance);
Ultrasonic ultrasonic2(detecteur_mur_fond);

IRrecv irrecv(emetteur_infra);
IRsend irsend;

char bluetooth;
int commande_bluetooth;
```

```
unsigned long date_courante = 0;
int etat_alarme = 0;
longdistance_mur;

void setup()
{
   Serial.begin(9600);
   mySerial.begin(9600);

   irsend.enableIROut(38);
    irsend.mark(0);
   pinMode(capteur_presence, INPUT);

    pinMode(fdchaut, INPUT);
    pinMode(fdcbas, INPUT);

   pinMode(BP_fermeture, INPUT);
   pinMode(BP_ouverture, INPUT);

  pinMode(commande_moteur_horaire, OUTPUT);
  pinMode(commande_moteur_horaire2, OUTPUT);
  pinMode(commande_moteur_antihoraire, OUTPUT);
  pinMode(commande_moteur_antihoraire2, OUTPUT);

   pinMode(alarme_visuel, OUTPUT);
   pinMode(alarme_sonore, OUTPUT);
}
void loop()
{
    // Distance de securite //
    long distance_sol;
  distance_sol = ultrasonic.MeasureInCentimeters();
    Serial.print("distance sol : ");
   Serial.print(distance_sol);
```

```
   Serial.println(" cm");

  Serial.println(capteur_presence);

   // Ordre de marche bluetooth //
   Serial.print("\t");
   Serial.print("commande bluetooth :");
  Serial.print(commande_bluetooth);
   Serial.print("\t");
   if (mySerial.available()>0)
    {
     bluetooth = mySerial.read() ;
     if (bluetooth == 'A') //Ouverture
      {
       commande_bluetooth = 10;
      }
     else if (bluetooth == 'D') // Fermeture
      {
       commande_bluetooth = 20;
      }
     else if (bluetooth == 'Z') // Stop
      {
       commande_bluetooth = 30;
      }
    }

   // Commande moteur ouverture //
  if((digitalRead(BP_ouverture) ||
commande_bluetooth == 10) && !digitalRead(fdchaut))
    {
    digitalWrite(commande_moteur_horaire,HIGH);
    digitalWrite(commande_moteur_horaire2,HIGH);
    digitalWrite(commande_moteur_antihoraire,LOW);
    digitalWrite(commande_moteur_antihoraire2,LOW);
```

```
      commande_bluetooth = 10;
     }

   else if(digitalRead(capteur_presence) && !
digitalRead(fdchaut))
     {
      if(!digitalRead(fdcbas))
       {
      digitalWrite(commande_moteur_horaire,HIGH);
     digitalWrite(commande_moteur_horaire2,HIGH);
     digitalWrite(commande_moteur_antihoraire,LOW);
     digitalWrite(commande_moteur_antihoraire2,LOW);
       commande_bluetooth =10;
       }
       else
       {
       digitalWrite(commande_moteur_horaire,LOW);
      digitalWrite(commande_moteur_horaire2,LOW);
      digitalWrite(commande_moteur_antihoraire,LOW);
      digitalWrite(commande_moteur_antihoraire2,
LOW);
       commande_bluetooth = 30;
       }
     }

    // Commande moteur fermeture //
   else if((digitalRead(BP_fermeture) ||
commande_bluetooth == 20) && !digitalRead(fdcbas))
     {
      digitalWrite(commande_moteur_horaire,LOW);
      digitalWrite(commande_moteur_horaire2,LOW);
      digitalWrite(commande_moteur_antihoraire,
HIGH);
      digitalWrite(commande_moteur_antihoraire2,
```

```
HIGH);
      commande_bluetooth = 20;
    }

    // arrêt des moteur //
   else if(commande_bluetooth == 30 ||
digitalRead(fdchaut)||digitalRead(fdcbas))
    {
     digitalWrite(commande_moteur_horaire,LOW);
     digitalWrite(commande_moteur_horaire2,LOW);
     digitalWrite(commande_moteur_antihoraire,LOW);
     digitalWrite(commande_moteur_antihoraire2,
LOW);
      commande_bluetooth = 30;
    }

    // radar de recul //
   distance_mur = ultrasonic2.
MeasureInCentimeters();
   Serial.print("distance mur :");
   Serial.print(distance_mur);
   Serial.println(" cm");
   if(distance_mur <= 2)
    {
   digitalWrite(alarme_visuel, HIGH);
   digitalWrite(alarme_sonore, HIGH);
    }
   else if(distance_mur > 2 && distance_mur <=5)
    {
     if((millis() - date_courante) > 200)
      {
       etat_alarme = !etat_alarme;
      digitalWrite(alarme_visuel, etat_alarme);
      digitalWrite(alarme_sonore, etat_alarme);
```

```
       date_courante = millis();
      }
    }
   else if(distance_mur > 5 && distance_mur <=8) //
proche du mur
    {
     if((millis() - date_courante) > 400)  //
temporisation d'une seconde
      {
       etat_alarme = !etat_alarme;
      digitalWrite(alarme_visuel, etat_alarme);
      digitalWrite(alarme_sonore, etat_alarme);
       date_courante = millis();
      }
    }
   else if(distance_mur > 8 && distance_mur <=11)
// proche du mur
    {
     if((millis() - date_courante) > 600)  //
temporisation d'une seconde
      {
       etat_alarme = !etat_alarme;
      digitalWrite(alarme_visuel, etat_alarme);
      digitalWrite(alarme_sonore, etat_alarme);
       date_courante = millis();
      }
    }
   else if(distance_mur > 11 && distance_mur <=14)
// proche du mur
    {
     if((millis() - date_courante) > 600)  //
temporisation d'une seconde
      {
       etat_alarme = !etat_alarme;
```

```
      digitalWrite(alarme_visuel, etat_alarme);
      digitalWrite(alarme_sonore, etat_alarme);
       date_courante = millis();
      }
    }
   else if (distance_mur >15)
    {
     digitalWrite(alarme_visuel, LOW);
     digitalWrite(alarme_sonore, LOW);
    }
}
```