Livrable H : Prototype III et rétroaction du client

Cours: GNG 1503
Section: D03
N° équipe: 10
Nom et prénom de l'équipage :

- Soussi Naima #300141438
- Roy Sébastien #300123281
- Torjmen Karim #300156574
- Sanspoil Florian #300207641

● Introduction :

"Tout obstacle renforce la détermination. Celui qui s'est fixé un but ne change pas."
L.De Vinci.

En effet, ce dernier livrable consacré aux ultimes prototypes, nous a causé quelques difficultés. Au cours de ce livrable, nous avons pu aboutir au prototype physique du capteur et de l'afficheur en les réunissant en un seul système. Cependant établir une synchronisation et une fluidité correcte était plus complexes que nos attentes. Mais en réunissant nos forces et en mettant en avant notre détermination, nous sommes parvenus à un résultat satisfaisant, tant au niveau du système de détection composé des capteurs et de l'afficheur, qu'au système externe, l'application. Nous avons donc au cours de ce livrable, effectué les tests finaux de fonctionnement et de synchronisation du système de détection de personnes. Nous avons résumer le tout dans notre tableau du plan de test. Ce livrable suit donc notre plan de confection du système à savoir dans un premier temps nous avons le système de détection avec les capteurs. Deuxièmement, nous avons notre prototype d'affichage et enfin nous terminerons par notre application.

Tableau du plan de test des Prototypes :

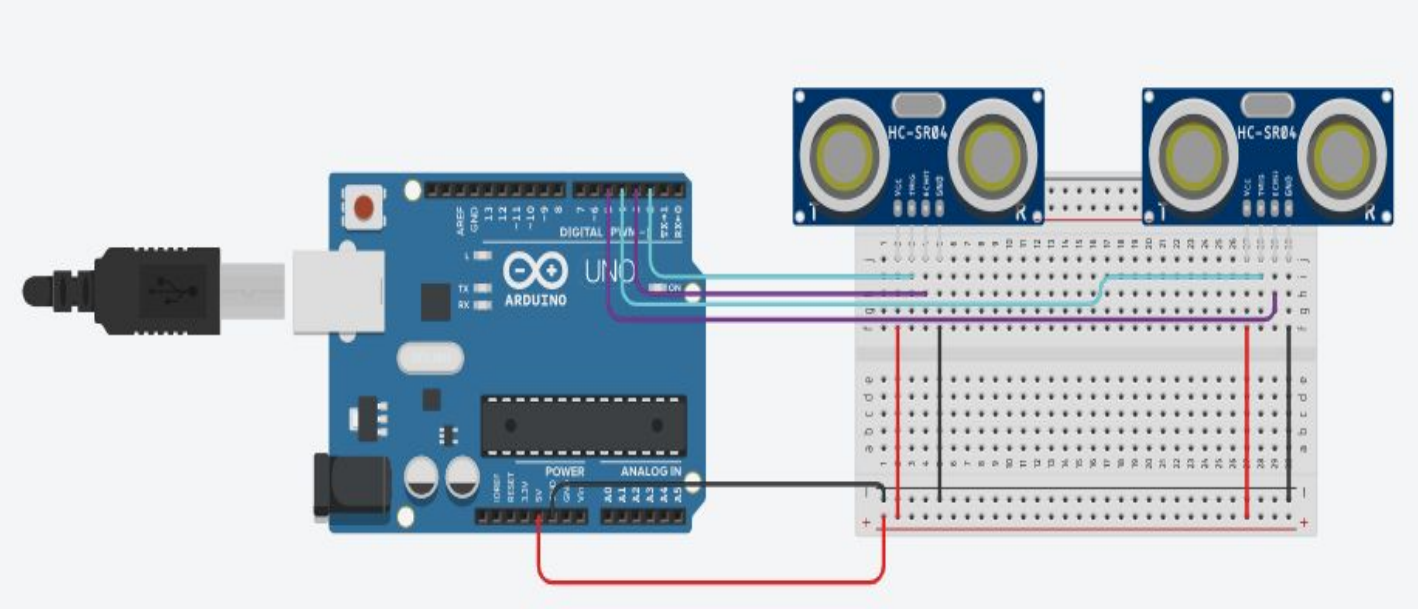
N° de Test	***Objectif du Test*** (Pourquoi)	***Description du Prototype Utilisé et de la Méthode de Test de Base*** (Quoi)	***Description des Résultats à Documenter et Comment ces Résultats seront Utilisés*** (Comment)	***Durée Estimée du Test et Date Prévue du Début du Test*** (Quand)
1	Confirmer la réception des mouvements par les capteurs	Nous avons utilisé les capteurs physique pour s'assurer qu'il n'est pas de bris électronique	Nous avons fait passer différents items devant les capteurs pour s'assurer que ceux-ci détectent une différence dans le retour de l'onde ultrasonique. Nous avons conclu que ceux-ci captent bien la présence d'objet dans leur champ de vision.	Nous avons testé ceci le 10 Novembre pour environ 1h
2	Confirmer communication entre les deux capteurs	Nous avons encore une fois utilisé les capteurs physiques pour s'assurer que le code permet une bonne communication entre les capteurs et retourne les valeurs voulues.	En effet, nous avons eu quelques difficultés ici, mais nous sommes arrivés à un code qui permet une communication raisonnable entre les capteurs.	Ce test à été étalé sur environ 1 semaine commençant le 18 Novembre.
3	Tester la synchronisation entre les capteurs et l'afficheur	Branchement de l'afficheur sur la carte Arduino avec les capteurs et le haut-parleur.	L'afficheur a su inscrire les données collectées par les capteurs.	Environ 5 heures, tests débutent le 11 Novembre

		Inclusion du fonctionnement de l'afficheur dans le code.		
4	Tester l'habilité d'utilisation de l'application	Les 2 versions d'applications ont été fluidifiées et sont désormais opérationnelles. Nous nous sommes donc concentré sur la synchronisation entre l'application et les données du smartphone	L'application à été synchronisée avec les données du smartphones, l'heure d'actualisation est affiché en direct ainsi que le renvoie de données de l'application vers le smartphone	Environ 1 heure. Le test à été réalisé le 24 novembre

1)Prototype III du capteur:

-Capteur ultrasonic hc-sr04 :

-Câblage avec carte arduino:



- **Principe du capteur:**

Le détecteur HC-SR04 utilise les ultrasons pour déterminer la distance à laquelle se trouve un objet. Peu importe l'intensité de la lumière, la température ou le type de matière, le capteur pourra facilement détecter s'il y a un obstacle devant lui. Tout de fois, il peut être contraint sur certains types de couleurs tel que le noir (contraste), ou encore sur la matière comme le textile. Son champ de vision est de 90° environ selon l'environnement. Si une impulsion de plus de 10µS est détectée, alors le capteur envoie une série de 8 impulsions à ultrason de 40 kHz et attend le réfléchissement du signal. Ensuite, en ayant en tête la vitesse du son, il effectue un rapide calcul pour déterminer la distance.

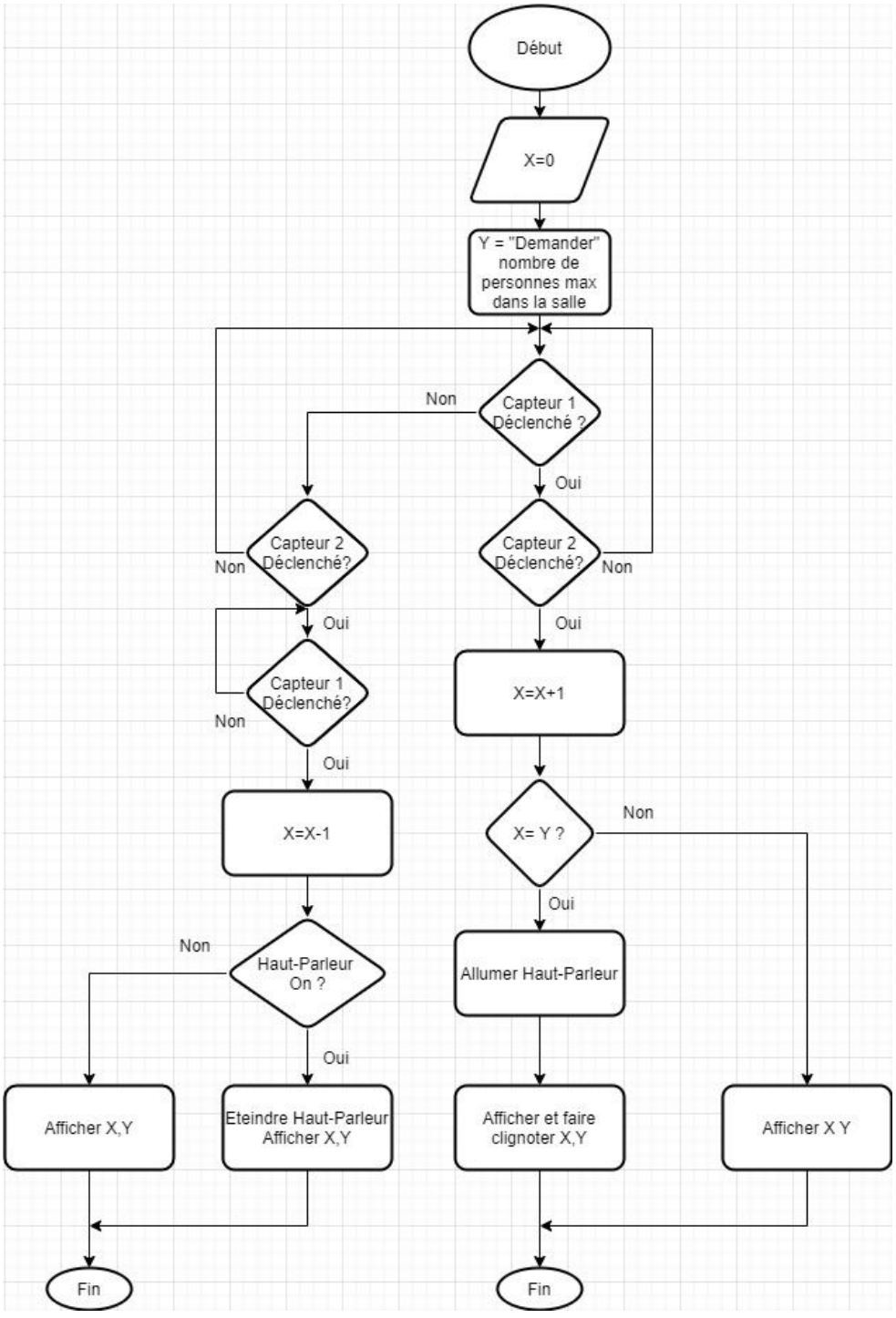
- **Fonctionnement du capteur pour notre cas:**

Les capteurs 1 et 2 sont deux capteurs ultrasons HC-SR04, ils permettent de mesurer la distance qui les sépare d'un obstacle (personne) et ce allant jusqu'à une distance de 4 à 6 mètres. On va considérer que le largeur de l'entrée de la toilette fait 1 mètre de large, c'est-à-dire que si les capteurs indiquent tous les 2 une distance inférieure ou égale à 1 mètre alors une entrée ou une sortie doit être comptabilisée. La variable nombre de personnes démarre à 0. Si capteur 1 réagit avant capteur 2 et que les deux sont franchis ALORS il s'agit d'une entrée donc on va faire un +1 sur notre variable nombre de personnes. Si capteur 2 réagit avant capteur 1 et que les deux sont franchis ALORS il s'agit d'une sortie donc on va faire un -1 sur notre variable nombre de personnes.

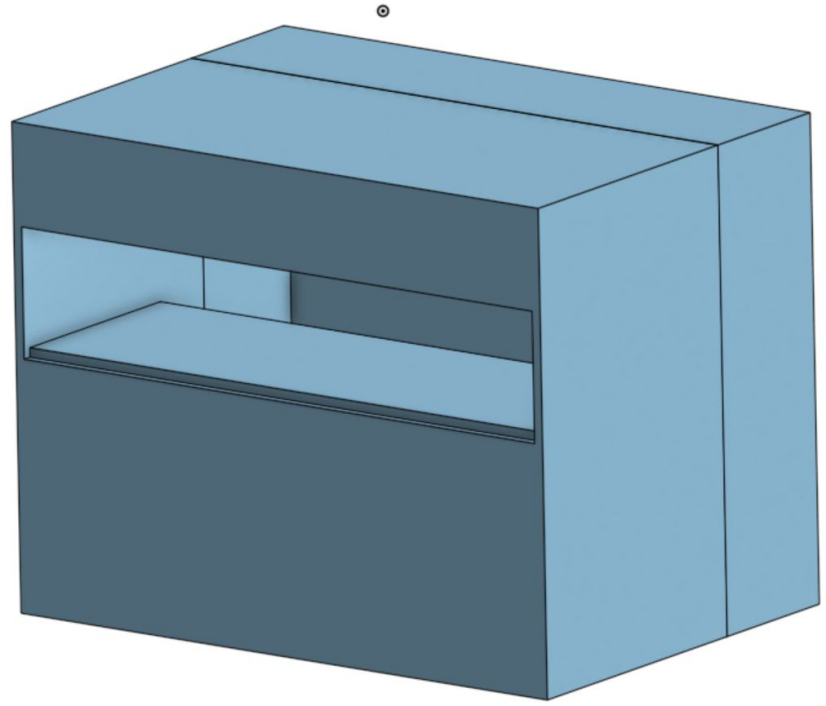
Caractéristiques techniques du HC-SR04

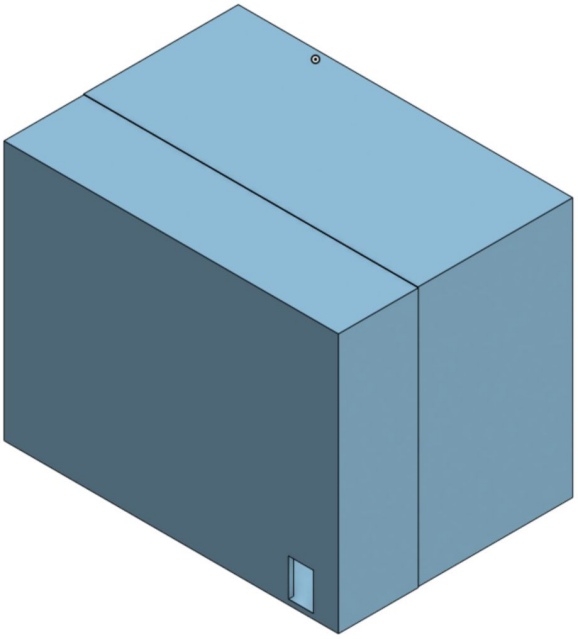
1. Plage de mesure de distance : 2cm à 450 cm (4,5m)
2. Précision de mesure : 0,3cm.
3. Tension d'alimentation : 5V.
4. Sortie numérique : PWM.
5. Poids : 9g.
6. Dimensions : 45mm x 20mm x 18mm

Organigramme du code à programmer :



Boîtier du capteur :





Code Operationnel obtenu jusqu'ici :

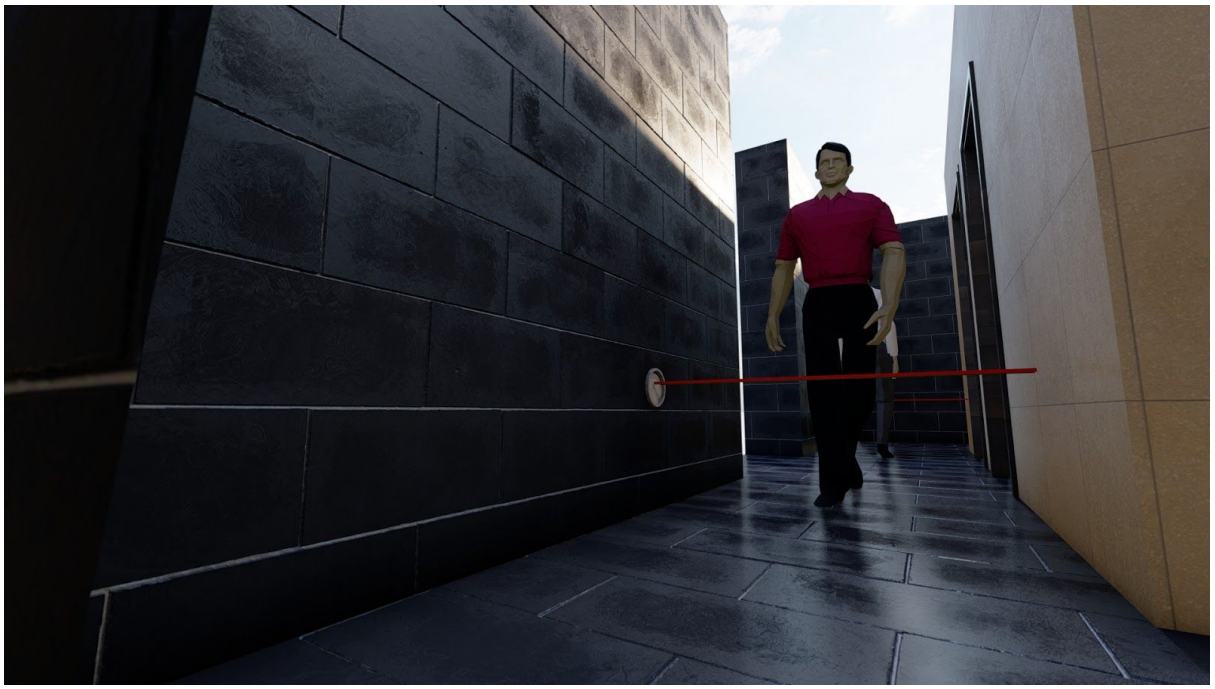
```
//--------------------------------------------------------------------------------
//  Partie 1 du programme : La détection


// Définition des bibliothéque utilisé (Télécharger bibliothéque Md PArola et MD_Max72xx pour l'afficheur :
// Inclut les bibliothèques Arduino requises:
#include <MD_Parola.h>
#include <MD_MAX72XX.h>
#include <SPI.h>
// Définissez le type de matériel, la taille et les broches de sortie:
#define HARDWARE_TYPE MD_MAX72XX :: FC16_HW
#define MAX_DEVICES 4
#define CS_PIN 10
// Créer une nouvelle instance de la classe MD_Parola avec une connexion SPI matérielle:
 // MD_Parola myDisplay = MD_Parola ( HARDWARE_TYPE, CS_PIN, MAX_DEVICES ) ;
// Configuration pour le logiciel SPI:
#define DATAPIN 11
#define CLK_PIN 13
 MD_Parola myDisplay = MD_Parola (HARDWARE_TYPE, DATAPIN, CLK_PIN, CS_PIN, MAX_DEVICES);
// initialisation de l'afficheur :

// Modifier le nombre limite ici :
int y = 2 ;
int i = 0;
int currentState = 0;
int previousState = 0;
int currentState2 = 0;
int previousState2 = 0;

#define trigPin 2
#define echoPin 3
#define trigPin2 4
#define echoPin2 5

void setup() {
  Serial.begin (9600); // on va utiliser notre ordinateur afin de lire ce que dit la carte Arduino
  pinMode(trigPin, OUTPUT); // on active la borne trigger du capteur 1
  pinMode(echoPin, INPUT); // on active la borne echo du capteur 1
  pinMode(trigPin2, OUTPUT); // on active la borne trigger du capteur 2
  pinMode(echoPin2, INPUT); // on active la borne echo du capteur 2
  Serial.println("Demarrage du programme"); // cette ligne est inutile c'est juste pour faire beau

  // Parie pour l'afficheur
  // Initialisez l'afficheur :
```

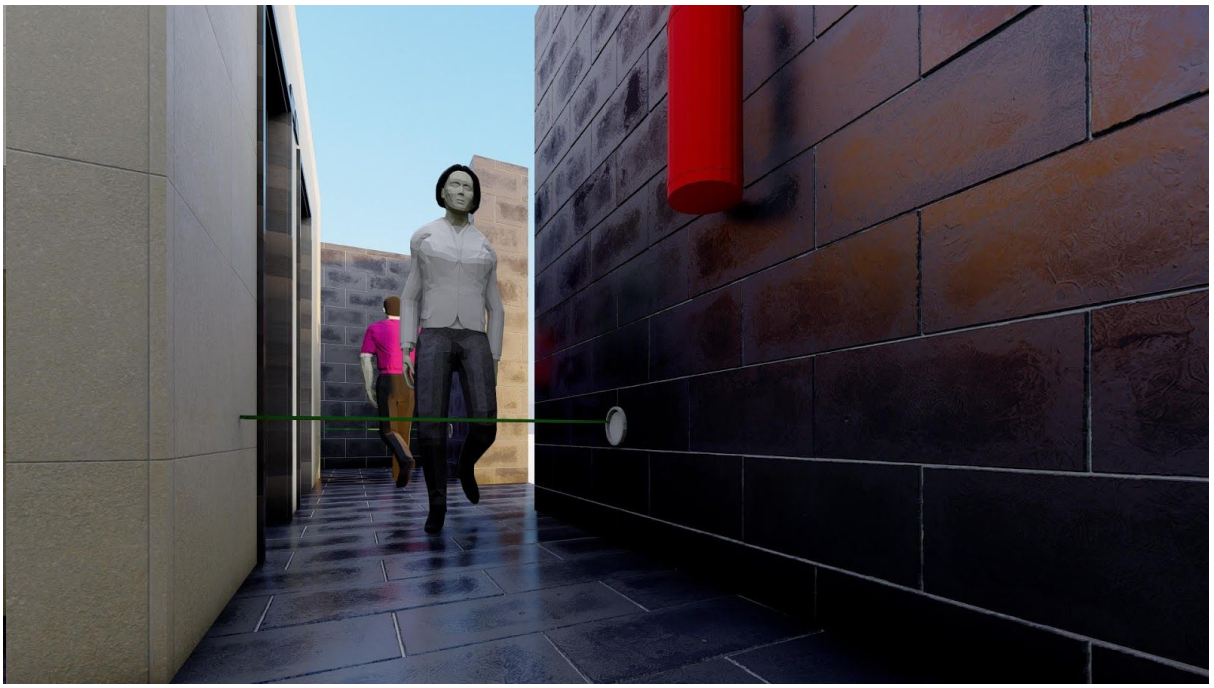
```
  myDisplay. begin () ;
  // Réglez l'intensité (luminosité) de l'affichage (0-15):
  myDisplay. setIntensity ( 8 ) ;
  // Effacer l'affichage:
  myDisplay. displayClear () ;
}
unsigned int visitor = 0; // de base il y a personne dans notre salle de bain et visitor ne peut
prendre une valeur negative
void loop() {

 long duration, distance;
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = (duration/2) / 29.1;
 long duration2, distance2;
 digitalWrite(trigPin2, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin2, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin2, LOW);
 duration2 = pulseIn(echoPin2, HIGH);
 distance2 = (duration2/2) / 29.1;
 // object entering in the system
 if (distance <= 10){
 currentState = 1;
 }
 else {
 currentState = 0;
 }
   delay(100);
 if(currentState != previousState){
 while(currentState == 1) {
 long duration2, distance2;
 digitalWrite(trigPin2, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin2, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin2, LOW);
 duration2 = pulseIn(echoPin2, HIGH);
 distance2 = (duration2/2) / 29.1;
 // object exiting the system
 if (distance2 <= 10){
 currentState2 = 1;
 }
```

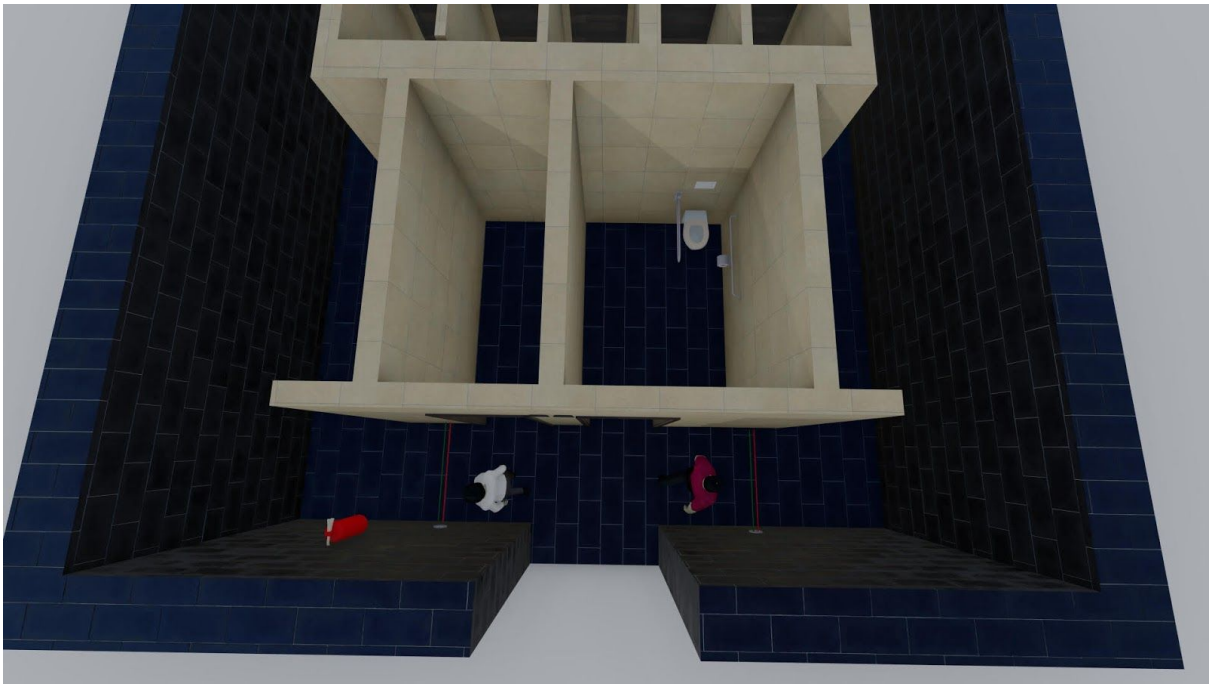
```
 else {
 currentState2 = 0;
 }
  delay(100);
 if(currentState2 != previousState2){

 if(currentState2 == 1) {
 i = i+1;

 delay(1000); // pause for 1/2 second
 }
 if ( i < 0 ) { i = 0 ; }
 Serial.println(i) ;
 return;

   }
  }
 }
 // object exit from the system
if (distance2 <= 10){
 currentState2 = 1;
 }
 else {
 currentState2 = 0;
 }
 delay(100);
 if(currentState2 != previousState2){
 while (currentState2 == 1) {
 long duration, distance;
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = (duration/2) / 29.1;
 if (distance <= 10){
 currentState = 1;
 }
 else {
 currentState = 0;
 }
 delay(100);
 if(currentState != previousState){
 if(currentState == 1) {
 i = i-1;
if ( i < 0 ) {
  i = 0 ; }
```

```
 delay(1000); // pause for 1 second
 }
Serial.println(i) ;
 return;
  }
 }
 }

//--------------------------------------------------------------------------------
//  Partie 2 du programme : L'affichage
//--------------------------------------------------------------------------------
      // si le nombre de personnes est inférieur à la limite
if (i<y){
    myDisplay. setTextAlignment ( PA_LEFT ) ;
    myDisplay. print ( i ) ;
   delay (200) ;
    myDisplay. setTextAlignment ( PA_RIGHT);
    myDisplay. print( y );
    delay (200) ;
}

if (i>=y){
    myDisplay. setTextAlignment ( PA_CENTER ) ;
    myDisplay. print ( "STOP" ) ;
    delay ( 500 ) ;
    myDisplay. setTextAlignment ( PA_LEFT ) ;
    myDisplay. print ( i ) ;
    delay (200) ;
    myDisplay. setTextAlignment ( PA_RIGHT);
    myDisplay. print ( y ) ;
   delay (200) ;

  }
 }
```

-Quelques Photos 3D







Afficheur :

Compatible avec le système Arduino, notre affichage est supposé transférer les informations recueillies par le capteur, c'est-à-dire le nombre de personnes présentes dans la salle de bain.

Cas 1 :

Tant que le nombre de personnes présentes reste inférieur au nombre maximal fixé lors de la programmation du système (dans ce cas-ci 2), il y aura affichage du nombre 0 ou 1 à gauche (donc personne à l'intérieur), et 2 à droite (donc nombre maximum à l'intérieur est de 2 personnes). Tout ceci se faisant avec clignotement alternatif entre la valeur de gauche et de droite (ceci est valeur donc pour les autres cas potentiels) (voir **images Cas 1/a et Cas 1/b)**

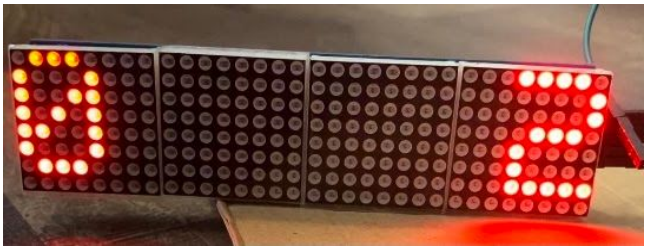
Cas 2 :

Quand le nombre de personnes à l'intérieur atteint le maximum fixé (2 : 2), le clignotement alternatif des deux valeurs sera suivi par l'affichage d'un signal "STOP" qui indiquera qu'il y a danger sanitaire et de ce fait une interdiction de rentrer. (voir **images Cas 2**)

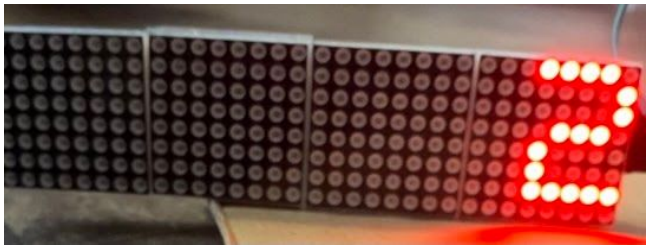
Cas 1/a : Initialement avec 0 personnes

0 personnes à l'intérieur (à gauche) :

Lors du clignotement un à un :

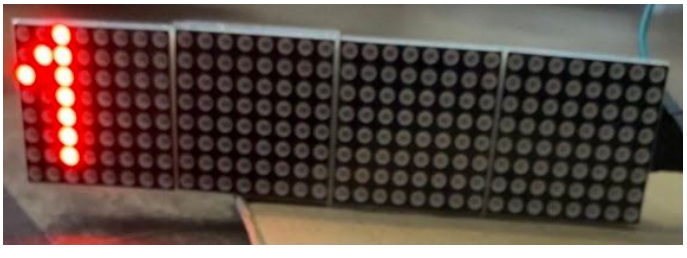


2 personnes maximum (à droite):

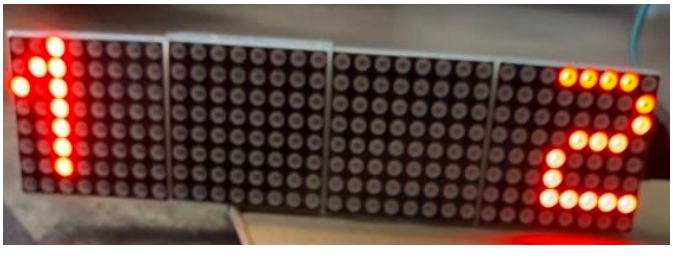


Cas 1/b : Avec 1 personne à l'intérieur

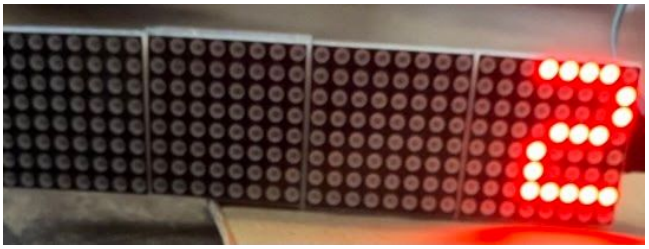
1 personne à l'intérieur (à gauche) :



Lors du clignotement un à un :



2 personnes maximum (à droite) :

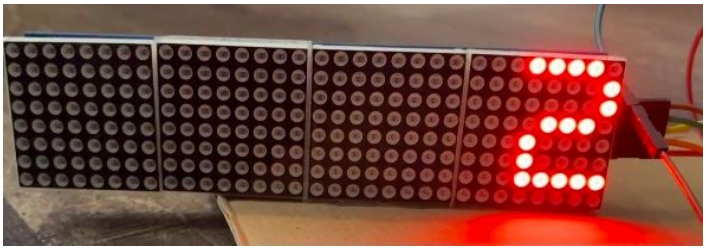


Cas 2 : Avec 2 personnes à l'intérieur

2 personnes à l'intérieur (à gauche) :

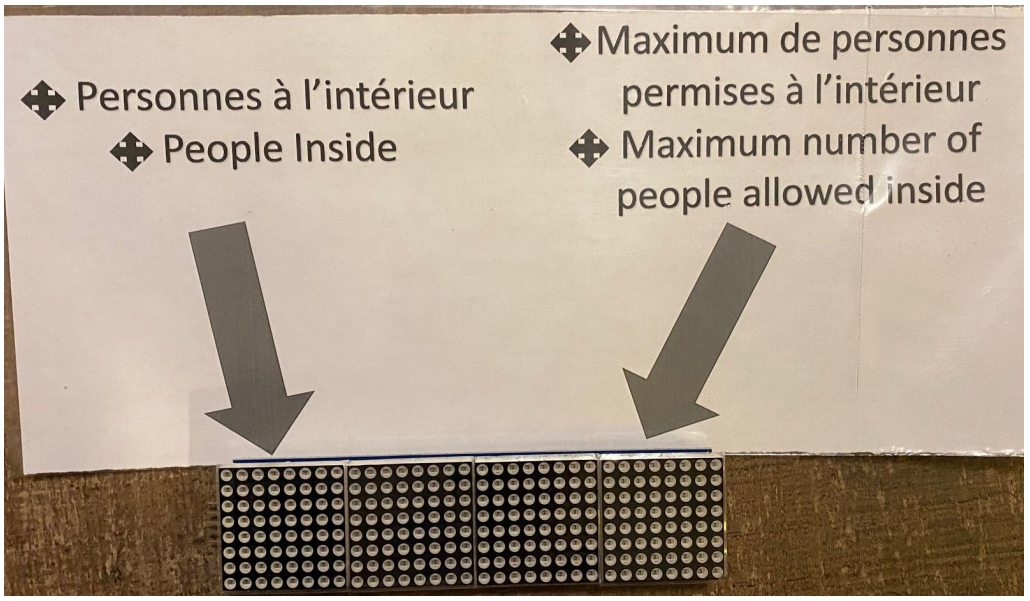
Lors du clignotement un à un :

2 personnes maximum (à droite) :



Signal affiché juste après clignotement un à un :

Pancarte au-dessus de l’afficheur :



Prototype 3 : Application :

Le visuel sobre et épuré de l'application a plu dès les premiers prototypages. Suite à cela, nous nous sommes concentrées afin de développer deux versions d'application, l'une offerte au grand public qui permet de savoir s'il est possible de se rendre aux toilettes ou non en affichant clairement le nombre de personnes dans la pièce ainsi que le nombre limite. Elle débute toujours de la même façon en affichant le rappel du bâtiment STEM **(Figure 1)** puis l'écran d'accueil **(Figure 2)** où le choix de la pièce est libre. Le changement dans ce dernier prototype pour cette version est uniquement la synchronisation de l'heure d'actualisation avec celle du smartphone et l'affichage de la pièce que l'on sélectionne. La seconde version dédiée au client et gestionnaire des infrastructures possède la même interface de démarrage que la version "grand public". Par la suite, l'écran d'accueil est légèrement différent tout en conservant l'authenticité de notre application. Comme nous le montre la **Figure 3**, on s'aperçoit qu'un bouton a été ajouté. En effet, il permet d'accéder au paramètres de la carte Arduino et de modifier uniquement le nombre limite de personnes d'une pièce **(Figure 4)**. Cette fonctionnalité est efficace notamment si le système est amené à être déplacé. Le branchement via un PC et des connaissances en codage d'Arduino ne sont donc pas nécessaires pour modifier cela. Les tests de communication entre l'application et la carte Arduino n'ont malheureusement pu être effectués pour le moment notamment dûe au fait que notre équipe ne soit pas dans un même lieux mais aussi des difficultés que nous avons eu sur le codage du capteur.

Figure 1

Figure 2

Figure 3

Figure 4

- ## Conclusion

Cette session de prototypage se différencie des précédentes notamment sur sa réalisation. En effet, ici nous avons assemblé, testé et confectionné notre système physique. De plus, voir notre travail aboutir à des résultats est satisfaisant. Il se démarque également sur son déroulement, lors de la confection de notre système, contrairement au prototype précédents, tous les membres de l'équipe à participer au développement des trois sous-systèmes. Ainsi nous avons pu mettre en accord nos connaissances, notre créativité et notre côté technique tout en développant de nouvelles compétences individuelles. Malgré des difficultés notamment au niveau du codage des capteurs, nous sommes parvenus au résultat que nous voulions et comme l'a dit De Vinci, chaque obstacle a su renforcer notre détermination.

Rétroaction:

Utilisateur 1 : Il a trouvé que notre concept était très bien. Il a adoré l'application et comment fonctionnelle et pratique elle pouvait être. Il a trouver l'afficheur très efficace et n'avait pas de difficulté à discerner ce que les différente animation voulait indiquer.

Utilisateur 2: Il a lui aussi trouvé que l'application était superbe et pratique. Il nous a indiqué qu'il allait la télécharger lorsqu'il allait étudier à STEM. Il a cependant trouvé que l'afficheur était quelque peut dure à comprendre, mais après quelques secondes devant celui-ci il à vite capté le concept. Il nous a ensuite indiqué que les capteurs avaient peut-être une trop grande incertitude et cela pourrait causer des problèmes.

Utilisateur 3: Il ne comprenait pas l'utilité d'une application, mais après avoir reçu quelques notifications et avoir pensé comme la gestionnaire il a vite compris qu'elle était géniale. Il a trouvé le fonctionnement de l'afficheur impeccable, mais comme l'utilisateur 2, il a quelques doutes sur les capteurs.

Réponse aux rétroactions: Nous avons adoré les commentaires sur l'application et l'afficheur nous étions très fier de ce travail. Cependant, puisque la fiabilité des capteurs est encore un problème si on se fie aux rétroactions. Nous allons revoir le code reliant les capteurs entre eux pour éliminer les sources d'erreurs.

Attribution des tache:

Florian s'est concentré sur le code de l'application et des capteurs. Naima à aider à la construction des circuits et des multiples essais techniques. Karim à aider à l'installation des capteurs et configurer tous les branchements. Sébastien a conçu le boîtier et a aussi aidé avec le code pour les capteurs.

Trello mise à jours le 25 novembre: