

# PRESENTATION

# PROJET COMPTE PERSONNES

Centre d'Enseignement et de Recherche Informatique d'Avignon

*12 mai 2025*

## Auteurs

Mathis Hernandez  
Amel Naak

## Encadrants

M. Gozlan  
M. Silanus



# Sommaire

Introduction	3-5
• Problématique et Etat de l'art	3
• Notre solution	4
• Notre organisation	5
Partie de Mathis	6-24
• Programmation Arduino	6
• Conception 3D	14
• Assemblage et peinture	18
Partie d'Amel	25-33
• Création du site web	26
• Réalisation du graphe	29
• Communication des données via USB	31
Conclusion	34

# Problématique

Comment mettre à disposition en présentiel et à distance l'information du nombre de personnes présentes dans une salle ?

## Etat de l'art : solutions existantes

Capteurs Infrarouges



Caméras 3D

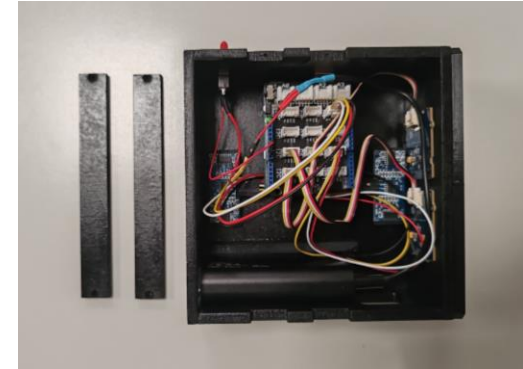
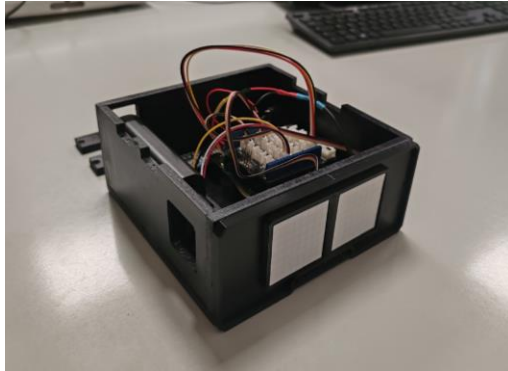


Tapis connectés



# Solution développée

## TransitTracker



- Dispositif positionné sur la partie supérieure d'un encadrement de porte
- Alimenté par batterie
- Détecte les entrées et sorties
- Affiche en présentiel le nombre de personnes dans la pièce
- Envoie l'information à un site Web hébergé sur un ordinateur relié par câble

Besoin	Composant
Détecter une entrée / sortie	2 capteurs à ultrasons HC-SR04
Afficher le nombre de personnes présentes	2 matrices LED RGB Grove 8x8
Traiter les informations	Carte Arduino Uno + Grove Base Shield
Partager les informations via USB	Fil USB qui transmet des données
Alimenter le système	Batterie 5200mAh sortie 5V

# Notre organisation

## Répartition des tâches

Responsable	Tâche
Mathis	Programmation
	Conception 3D
	Assemblage et peinture
Amel	Communication des informations via USB
	Stockage dans une base de données MySQL
	Création du site Web
	Réalisation d'un graphe

## Supports de communication

- WhatsApp
- E-mail
- Dépôt GitHub

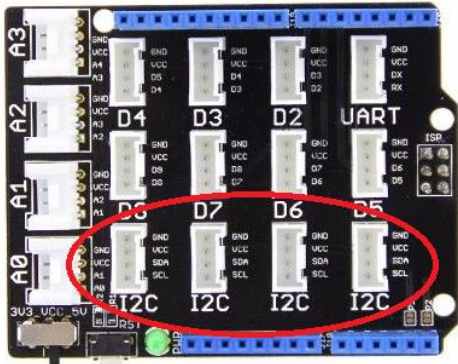
# Partie de Mathis

**1.**

## **Programmation Arduino**

## Première étape : Affichage simple

Branchement sur  
bus I2C



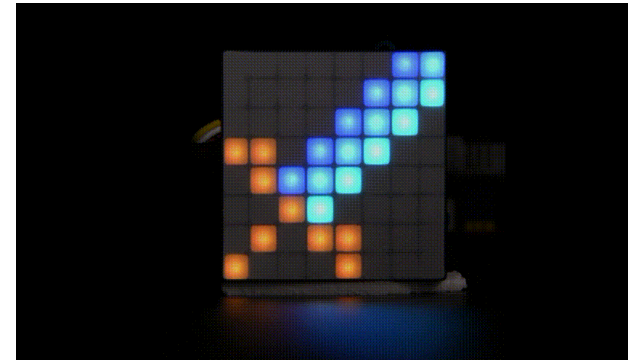
Bibliothèque officielle  
GroveTwoRGBLedMatrixClass

```
#include "grove_two_rgb_led_matrix.h"

GroveTwoRGBLedMatrixClass matrix;

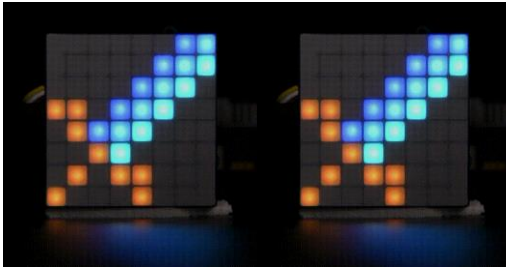
void loop() {
  for (int i = 0; i < 35; i++) {
    matrix.displayEmoji(i, 5000, true);
    delay(5000);
  }
}
```

Affichage  
sympathique

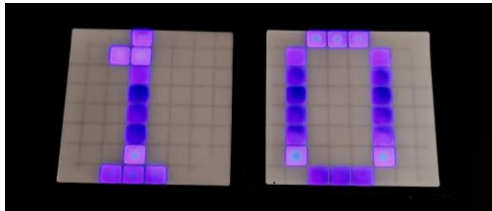


## Problème

Affichages identiques



Affichages différents



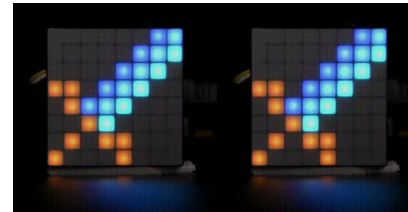
## Solution

Instancier deux objets de la classe et donner à chacun une adresse différente :

```
GroveTwoRGBLedMatrixClass matrix1(GROVE_TWO_RGB_LED_MATRIX_DEF_I2C_ADDR, 1);  
GroveTwoRGBLedMatrixClass matrix2(GROVE_TWO_RGB_LED_MATRIX_DEF_I2C_ADDR + 1, 1);
```

### Même problème

Affichages toujours identiques



### Solution



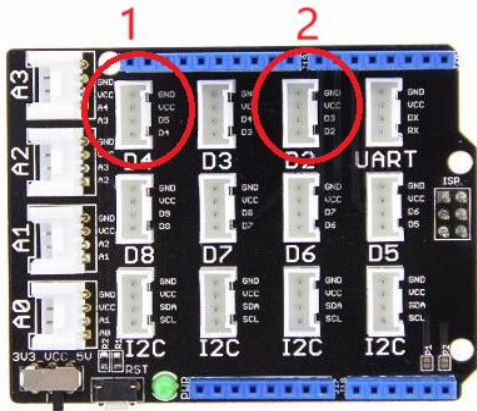
### "Solution"

Contournement du problème  
Nous avons continué avec d'autres matrices qui ne posaient pas ce problème



## Deuxième étape : recevoir les informations des capteurs

Branchement sur les broches  
3, 2 pour le premier capteur  
5, 4 pour le deuxième



Bibliothèque  
*HCSR04 ultrasonic sensor*

```
#include <HCSR04.h>

HCSR04 hc1(3, 2); //(trig pin , echo pin)
HCSR04 hc2(5, 4);

void setup() {...}

void loop()
{
    Serial.println(hc1.dist());
    Serial.println(hc2.dist());
    delay(60);
}
```

Les distances renvoyées par les deux  
capteurs sont utilisables

Output Serial Monitor X

Message (Enter to send message to 'Arduino Uno' on 'COM4')

Time	Distance 1 (cm)	Distance 2 (cm)
16:02:16.737	85.22	85.22
16:02:16.771	86.05	86.05
16:02:16.817	85.19	85.19
16:02:16.849	85.61	85.61
16:02:16.883	85.61	85.61
16:02:16.917	86.12	86.12
16:02:16.950	85.22	85.22
16:02:17.029	85.75	85.75
16:02:17.029	85.68	85.68
16:02:17.064	85.77	85.77
16:02:17.100	85.71	85.71

## Troisième étape : combiner les deux Programme final

### Principe de fonctionnement

Boucle de 50ms

Deux variables pour  
détecter un passage :

- **hc1\_detect** = *false*
- **hc2\_detect** = *false*



Couloir

Salle

## Troisième étape : combiner les deux Programme final

### Principe de fonctionnement

#### Passage

Deux variables pour  
détecter un passage :

- **hc1\_detect** = *true*
- **hc2\_detect** = *false*

Couloir



Salle

## Troisième étape : combiner les deux Programme final

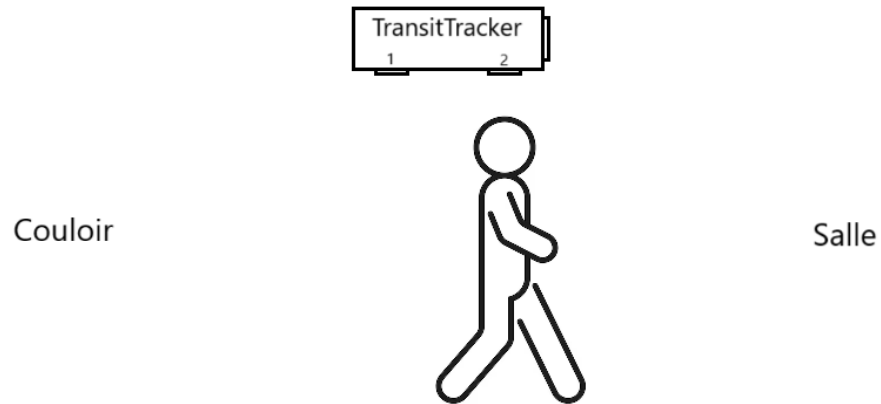
### Principe de fonctionnement

#### Passage

Deux variables pour  
détecter un passage :

- **hc1\_detect** = *true*
- **hc2\_detect** = *true*

Compteur + 1



## Troisième étape : combiner les deux Programme final

### Principe de fonctionnement

Reset toutes les  
secondes

Deux variables pour  
détecter un passage :

- **hc1\_detect** = *false*
- **hc2\_detect** = *false*



Couloir

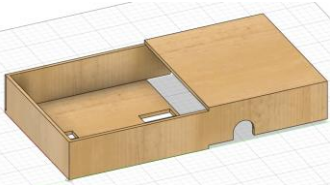
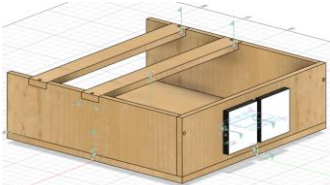

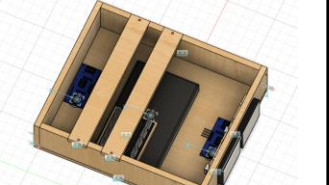
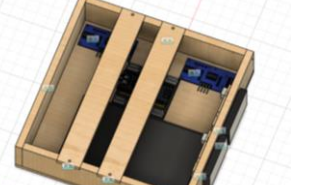
Salle

**Partie de Mathis**

# **2. Conception 3D**

## Versions non retenues

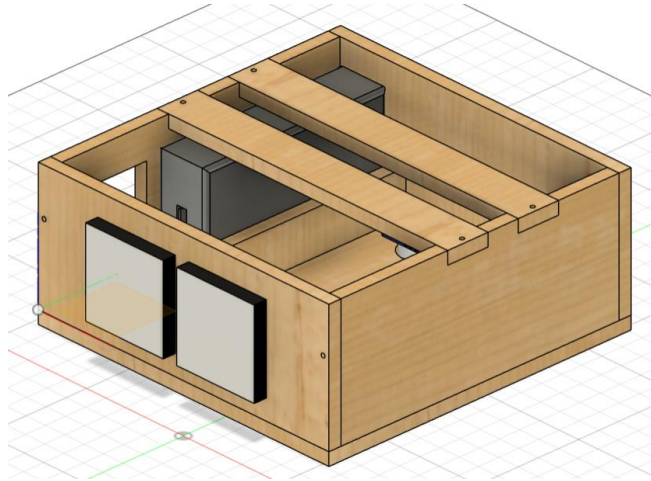
### Fusion 360

1	2	3	4	5
				

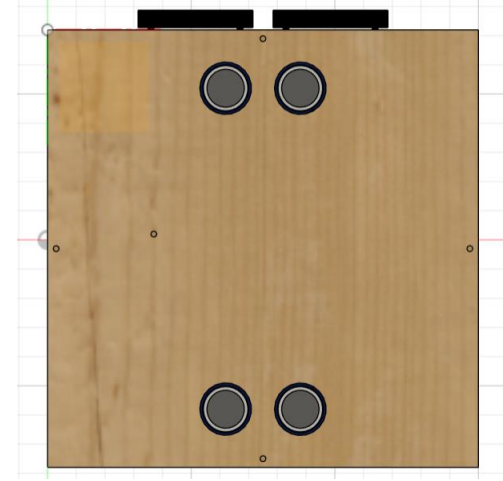
## Version finale



Vue de dessus



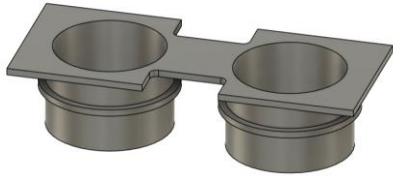
Vue d'ensemble



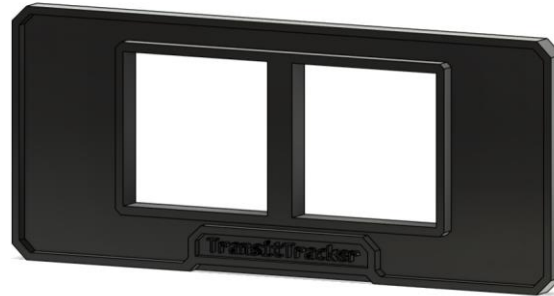
Vue de dessous



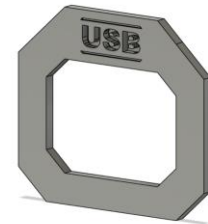
## Design et finitions



Supports / Rehausseurs pour capteurs



Design de la face avant

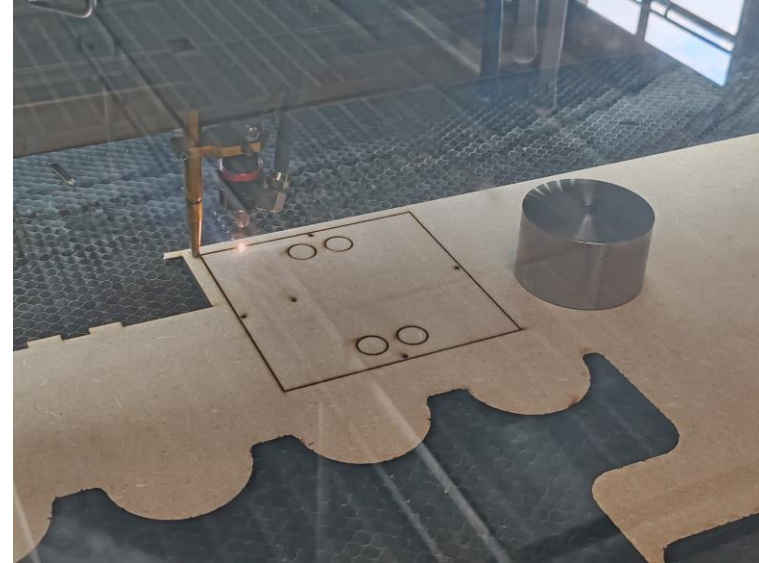


Contour

Partie de Mathis

# 3. Assemblage et peinture

## Découpe laser



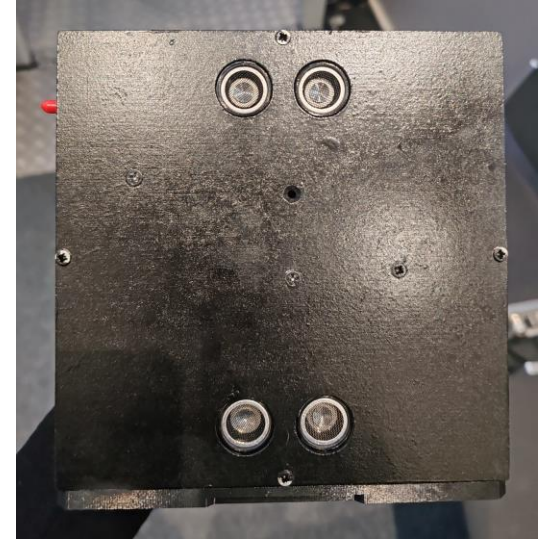
## Ponçage



## Peinture



Après 2 couches



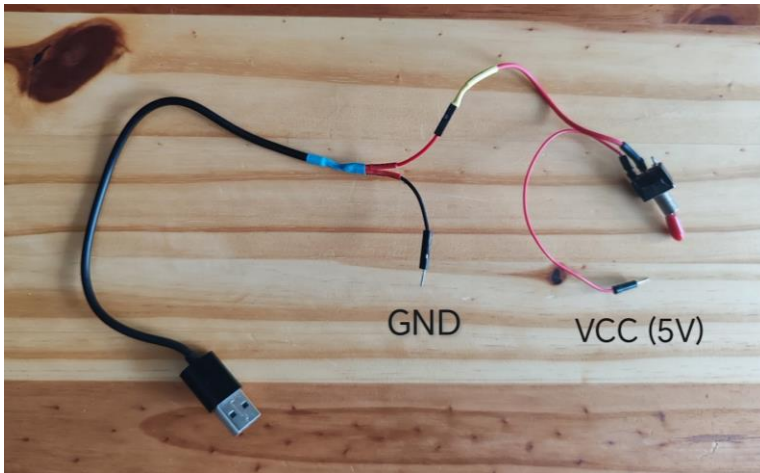
Après 4 couches

## Mise en évidence du Nom TransitTracker





## Ouvertures pour bouton Marche/Arrêt et accès USB Arduino



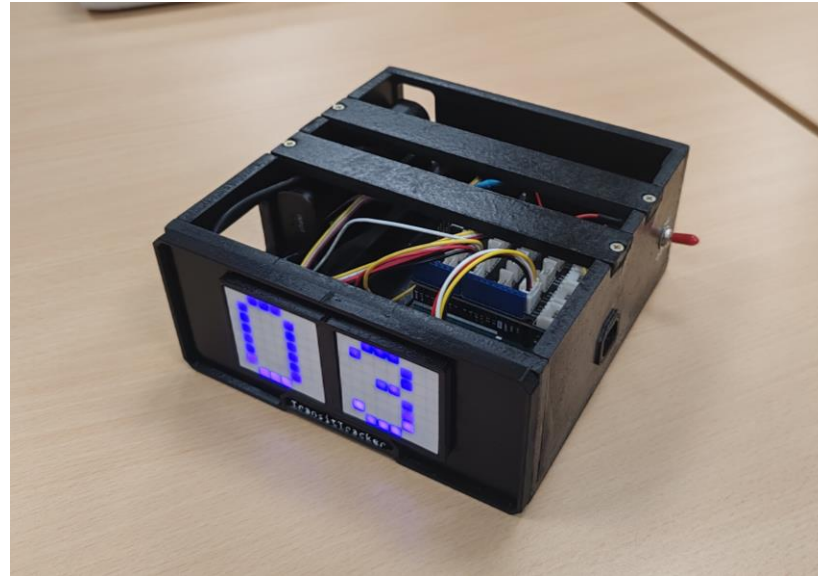
Mise en place du bouton M/A



Ouverture accès Arduino et bouton M/A installé

# Installation des composants

## Dispositif TransitTracker Fonctionnel














# Partie de Amel










**1.**

**La création et la conception du site**

## Première étape : Création de la base de données

- Télécharger le serveur local MAMP (Mac + Apache + MySQL + PHP)
- La table "Salle" pour stocker les différentes salles

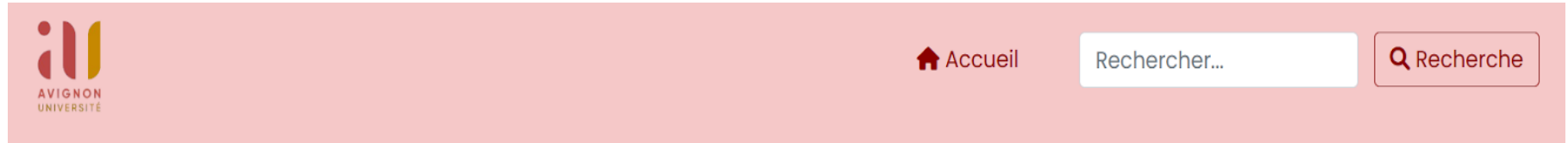
				id	capacite	nom
<input type="checkbox"/>		Éditer	 Copier	 Supprimer	1	50 Salle C024
<input type="checkbox"/>		Éditer	 Copier	 Supprimer	2	30 Salle S4 Nodes
<input type="checkbox"/>		Éditer	 Copier	 Supprimer	3	100 Amphi Blaise

				id	salle_id	nb_entrée	nb_sortie	nombre_personnes	heure
<input type="checkbox"/>		Éditer	 Copier	 Supprimer	1	1	10	3	7 2025-03-31 22:23:54
<input type="checkbox"/>		Éditer	 Copier	 Supprimer	2	1	20	4	16 2025-03-31 22:24:55
<input type="checkbox"/>		Éditer	 Copier	 Supprimer	3	1	10	4	6 2025-03-31 22:24:55

- La table "Personnes" pour stocker le nombres des personnes dans les salles

## Deuxième étape : Mise en page du site

- Le programme index.php qui se connecte à la base de données et génère cette interface en utilisant Bootstrap et HTML et CSS
- [http://localhost:8888/Site\\_compte\\_personnes/index.php](http://localhost:8888/Site_compte_personnes/index.php)



### Les salles :

Salle C024

Voir la salle

Salle S4 Nodes

Voir la salle

Amphi Blaise

Voir la salle

## Barre de recherche

- Ce bout du code permet de faire une recherche dans la table salle, à partir d'un champ de formulaire nommé nom\_salle.
- LIKE ? avec LOWER(nom) permet de chercher dans la table salle toutes les lignes dont le nom contient la chaîne tapée, sans tenir compte de la casse.

```
if (isset($_GET['nom_salle']) && !empty(trim($_GET['nom_salle'])))  
{  
    $nom_salle = strtolower(trim($_GET['nom_salle']));  
  
    /* Préparer une requête LIKE pour une recherche  
    partielle et insensible à la casse */  
    $sql = "SELECT * FROM salle WHERE LOWER(nom) LIKE ?";  
    $stmt = $conn->prepare($sql);  
  
    // Ajouter les jokers % pour recherche partielle  
    $param = '%' . $nom_salle . '%';  
    $stmt->bind_param("s", $param);  
    $stmt->execute();  
    $result = $stmt->get_result();
```

## **2.**

# **Réalisation du Graphe**

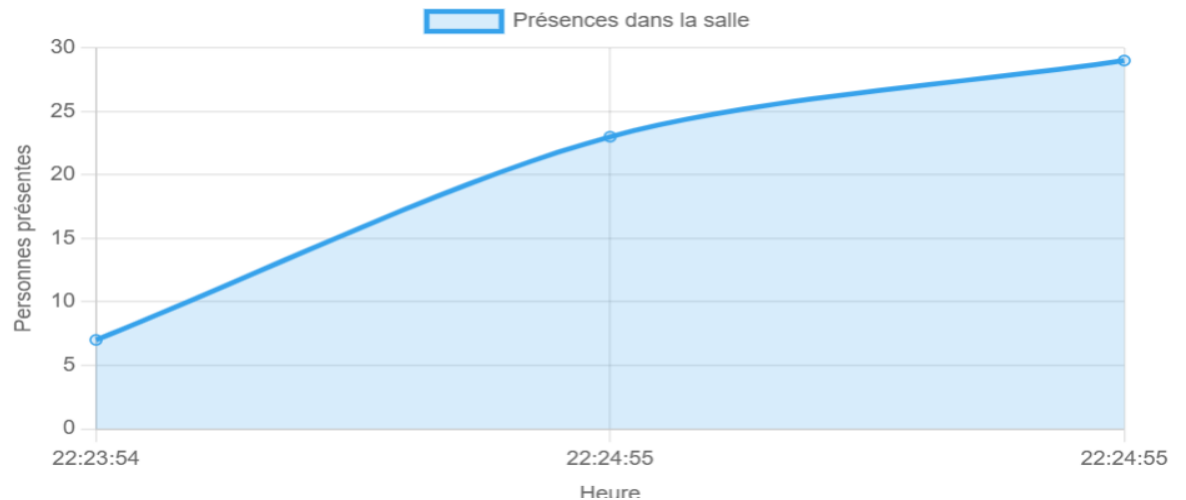
un graphique dynamique  
représentant l'évolution des  
présences au fil du temps,  
basé sur les entrées et  
sorties enregistrées c'est-à-  
dire le nombre des  
personnes dans cette salle.

## Détails de la salle : Salle C024

Capacité : 50 personnes

Nombre de personnes présents : 29

### Présence en temps réel



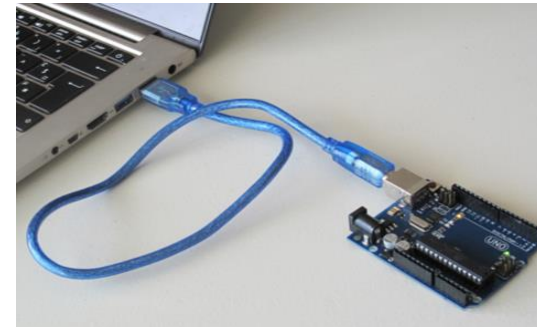
# **3.**

## **Communication des données via USB**

## Première étape : Programme Arduino

- Les deux ligne à ajouter pour envoyer la valeur du compteur via le SoftwareSerial.
- Connexion d'Arduino via USB (fil d'alimentation et de transfert de données ).

```
Serial.println("ENTREE");  
// ...  
Serial.println("SORTIE");
```





## Deuxième étape : Le programme python

- Le nom du port sur Windows est "COM6"
- La bibliothèque "**serial**" pour communiquer avec l'Arduino
- Et la bibliothèque "**mysql.connector**" pour connecter à la BDD

```
import serial
import mysql.connector
import time
import datetime

# === Configuration ===

# Port série (à adapter si nécessaire,
# ex : /dev/ttyACM0 sur ubuntu)
PORT_SERIE = "COM6"
```

# PRESENTATION

# PROJET COMPTE PERSONNES

Centre d'Enseignement et de Recherche Informatique d'Avignon

*12 mai 2025*

Auteurs  
Mathis Hernandez  
Amel Naak

## Conclusion

Encadrants  
M. Gozlan  
M. Silanus