|  |  |
| --- | --- |
| \\polyfile\Images\Logos\bonne resolution\logo group Universiapolis\logo polytechnique universia.png |  |
| **3 ème ANNEE CYCLE INGÉNIEUR DE L’ECOLE POLYTECHNIQUE D’AGADIR**  **FILIERE GENIE INFORMATIQUE**  [**INF5021-IoT PROGRAMMING AND DATA PROCESSING**](https://moodle.e-polytechnique.ma/course/view.php?id=404) | |
| **sMART TRACKING** | |
|  | |

**Encadré par :**

Mme. LACHHAB Fadwa

**Réalisé par :**

BENSSIMMOU Khaoula

JABOU Hanane

ECHCHAKHAR Hajar

KARMAOUI Moncef

ASSABAR Taoufik

ADARDOUR Anwar

|  |
| --- |
| Année universitaire 2022/2023 |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**REMERCIEMENTS**

Nous tenons à exprimer notre sincère reconnaissance à tous ceux qui ont rendu possible ce projet. Nous tenons à remercier notre encadrante pour sa disponibilité, sa patience et son soutien durant cette période. Son enseignement et ses conseils nous ont été précieux pour la réalisation de ce projet.

Nous remercions également l'université pour nous avoir donné l'opportunité de découvrir le monde de l’IoT. Cette expérience nous a apporté beaucoup de connaissances et de compétences qui seront certainement utiles pour notre avenir professionnel.

**RESUME**

Ce projet consiste à concevoir un système de suivi basé sur la détection du mouvement des utilisateurs qui vont se placer devant les capteurs et faire un suivi de ce mouvement.

Pour achever ceci, on a commencé par l’installation des capteurs dans différents angles sur le AI Lab, passant à la configuration des paramètres, l’envoi des données en temps réel, par la suite l’enregistrement de ces données sur une base de données et finalement la visualisation du suivi de ces données après avoir les analysés sur une page web.

Le protocole suivi donc c’est l’UART (Universal Asynchronous Receiver/Transmitter), il consiste à établir une communication série pour transmettre les données entre deux dispositifs électroniques et il est largement utilisé dans les systèmes embarqués.

TABLE DES MATIERES

[INTRODUCTION GENERALE 8](#_Toc125731826)

[CHAPITRE 1 9](#_Toc125731827)

[Contexte du projet 9](#_Toc125731828)

[**I.** **INTRODUCTION** 10](#_Toc125731829)

[**II.** **PRESENTATION DE l’AI LAB** 10](#_Toc125731830)

[2-1- Caractéristiques 10](#_Toc125731831)

[2-2- Représentation 3D des plans de la maison intelligente 10](#_Toc125731832)

[**III.** **THEMATIQUE** 13](#_Toc125731833)

[**IV.** **CONCLUSION** 13](#_Toc125731834)

[CHAPITRE 2 15](#_Toc125731835)

[Etude bibliographique 15](#_Toc125731836)

[**I-** **INTRODUCTION** 16](#_Toc125731837)

[**II-** **Etude d’existant** 16](#_Toc125731838)

[2-1- Solution : 16](#_Toc125731839)

[2-2- Spécification des besoins : 17](#_Toc125731840)

[2.2.1 Les besoins fonctionnels : 17](#_Toc125731841)

[2.2.1. Les besoins non fonctionnels : 17](#_Toc125731842)

[2.4.1. Environnement matériel : 17](#_Toc125731843)

[**III-** **Les différents capteurs pour le suivi des mouvements :** 18](#_Toc125731844)

[3-1- Hc-sr04 18](#_Toc125731845)

[3-2- PIR : 18](#_Toc125731846)

[3-3- Comparaison 19](#_Toc125731847)

[3-4- Pourquoi HC-SR04 ? 20](#_Toc125731848)

[**IV-** **Dispositif programmable** 20](#_Toc125731849)

[4-1- Arduino Mega et Arduino Uno 20](#_Toc125731850)

[4-1-1- Arduino Mega2560 20](#_Toc125731851)

[4-1-2- Arduino Uno 21](#_Toc125731852)

[4-1-3- Comparaison 21](#_Toc125731853)

[4-1-4- Pourquoi Arduino Uno ? 22](#_Toc125731854)

[4-2- ESP32 et NodeMCU 22](#_Toc125731855)

[4-2-1- ESP32 22](#_Toc125731856)

[4-2-2- NodeMCU 23](#_Toc125731857)

[4-2-3- Comparaison 23](#_Toc125731858)

[4-2-4- Pourquoi ESP32 ? 24](#_Toc125731859)

[4-3- Beagle Bone et Raspberry pi 24](#_Toc125731860)

[4-3-1- Beagle Bone 24](#_Toc125731861)

[4-3-2- Raspberry pi 25](#_Toc125731862)

[4-3-3- Comparaison 25](#_Toc125731863)

[4-3-4- Pourquoi Raspberry pi ? 26](#_Toc125731864)

[**V.** **Base de données** 27](#_Toc125731865)

[5-1- MongoDB 27](#_Toc125731866)

[5-2- Pourquoi MongoDB ? 27](#_Toc125731867)

[**VI.** **CONCLUSION** 28](#_Toc125731868)

[CHAPITRE 3 29](#_Toc125731869)

[Développement & analyse 29](#_Toc125731870)

[**I-** **INTRODUCTION** 30](#_Toc125731871)

[**II-** **LES MISSIONS EFFECTUÉES** 30](#_Toc125731872)

[**1.** **Besoins et objectifs du projet :** 30](#_Toc125731873)

[**2.** **Implémentation du projet :** 31](#_Toc125731874)

[**3.** **Architecture Hardware :** 31](#_Toc125731875)

[**4.** **Architecture Software :** 32](#_Toc125731876)

[**5.** **Schéma fritzing :** 32](#_Toc125731877)

[**III-** **REALISATION** 32](#_Toc125731878)

[**1.** **Arduino et ESP32 :** 32](#_Toc125731879)

[**2.** **Raspberry Broker :** 35](#_Toc125731880)

[**3.** **Stockage des données :** 35](#_Toc125731881)

[**4.** **La visualisation des données :** 36](#_Toc125731882)

[**IV-** **Conclusion** 36](#_Toc125731883)

[CONCLUSION GENERALE 37](#_Toc125731884)

[BIBLIOGRAPHIE 39](#_Toc125731885)

[WEBOGRAPHIE 40](#_Toc125731886)

[ANNEXES 41](#_Toc125731887)

TABLE DES ILLUSTRATIONS

**Liste des figures**

[Figure 1 AI LAB 10](#_Toc125811297)

[Figure 2 Kitchen 11](#_Toc125811298)

[Figure 3 Dining Room 11](#_Toc125811299)

[Figure 4 Living Room 12](#_Toc125811300)

[Figure 5 Bedroom 12](#_Toc125811301)

[Figure 6 hc-sr04 17](#_Toc125811302)

[Figure 7 PIR 17](#_Toc125811303)

[Figure 8 Arduino Mega2560 19](#_Toc125811304)

[Figure 9 Arduino Uno 20](#_Toc125811305)

[Figure 10 ESP32 21](#_Toc125811306)

[Figure 11 NodeMCU 22](#_Toc125811307)

[Figure 12 Beagle Bone 23](#_Toc125811308)

[Figure 13 Raspberry pi 24](#_Toc125811309)

[Figure 14 MongoDB 25](file:///C:\Users\admin\Downloads\SMART%20TRACKING%20(1).docx#_Toc125811310)

[Figure 15: Implémentation du projet 29](#_Toc125811311)

[Figure 16: Schema Fritzing 30](#_Toc125811312)

[Figure 17: Code Arduino 31](file:///C:\Users\admin\Downloads\SMART%20TRACKING%20(1).docx#_Toc125811313)

[Figure 18: Code Arduino 32](file:///C:\Users\admin\Downloads\SMART%20TRACKING%20(1).docx#_Toc125811314)

[Figure 19: Code Arduino 33](file:///C:\Users\admin\Downloads\SMART%20TRACKING%20(1).docx#_Toc125811315)

[Figure 20: Interface de l'application 34](#_Toc125811316)

**Liste des tableaux**

[Tableau 1 hc-sr04 et PIR 18](#_Toc125811317)

[Tableau 2 Arduino Uno et Arduino Mega 20](#_Toc125811318)

[Tableau 3 NodeMCU et ESP32 22](#_Toc125811319)

[Tableau 4 Beagle Bone et Raspberry Pi 24](#_Toc125811320)

# INTRODUCTION GENERALE

L’introduction permet de présenter l’idée directrice à travers deux/trois idées maîtresses. Pour acquérir une bonne et parfaite qualité, la formation théorique seule ne suffit pas, il est donc nécessaire de suivre une démarche réelle permettant de voir comment se déroulent les tâches dans la vie réelle.

Ce rapport a pour but de décrire les détails de la mise en place d'un système de détection et suivi de mouvement dans le AI LAB.

Ce système permet de détecter les mouvements à l'intérieur de la maison et de les suivre afin de fournir des informations utiles pour améliorer le confort et la sécurité des occupants.

Nous décrirons les différents composants utilisés pour mettre en place ce système, ainsi que les algorithmes utilisés pour détecter et suivre les mouvements.

Enfin, nous présenterons les résultats obtenus lors des tests effectués sur ce système et les améliorations possibles pour les futures versions.

Dans les sections suivantes, on va décrire en détail les différents aspects de la conception de ce système, notamment les capteurs utilisés, la méthode de suivi des utilisateurs en mouvement, et l’interface utilisateur.

# 

# CHAPITRE 1

# Contexte du projet

1. **INTRODUCTION**

Ce chapitre représente une vue générale sur l’environnement de ce projet, une description de l’AI LAB, ses caractéristiques et les systèmes adaptés au sein de l’AI LAB.

1. **PRESENTATION DE l’AI LAB**

### Caractéristiques

Le AI LAB est un laboratoire qui travaille sur le déploiement d’un dôme aménagé de type studio équipé d’objets connectés (IoT) et d’intelligence artificielle (IA) au cœur du campus d’Universiapolis. Le AiLab est doté de plusieurs systèmes adaptés, tels que :

* Smart lighting
* Smart Air Conditionner
* Smart plug
* Smart Assistance: real-time alerts, energy management, appliance control, smart access.

### Représentation 3D des plans de la maison intelligente

* The house

****

Figure 1 AI LAB

* Kitchen

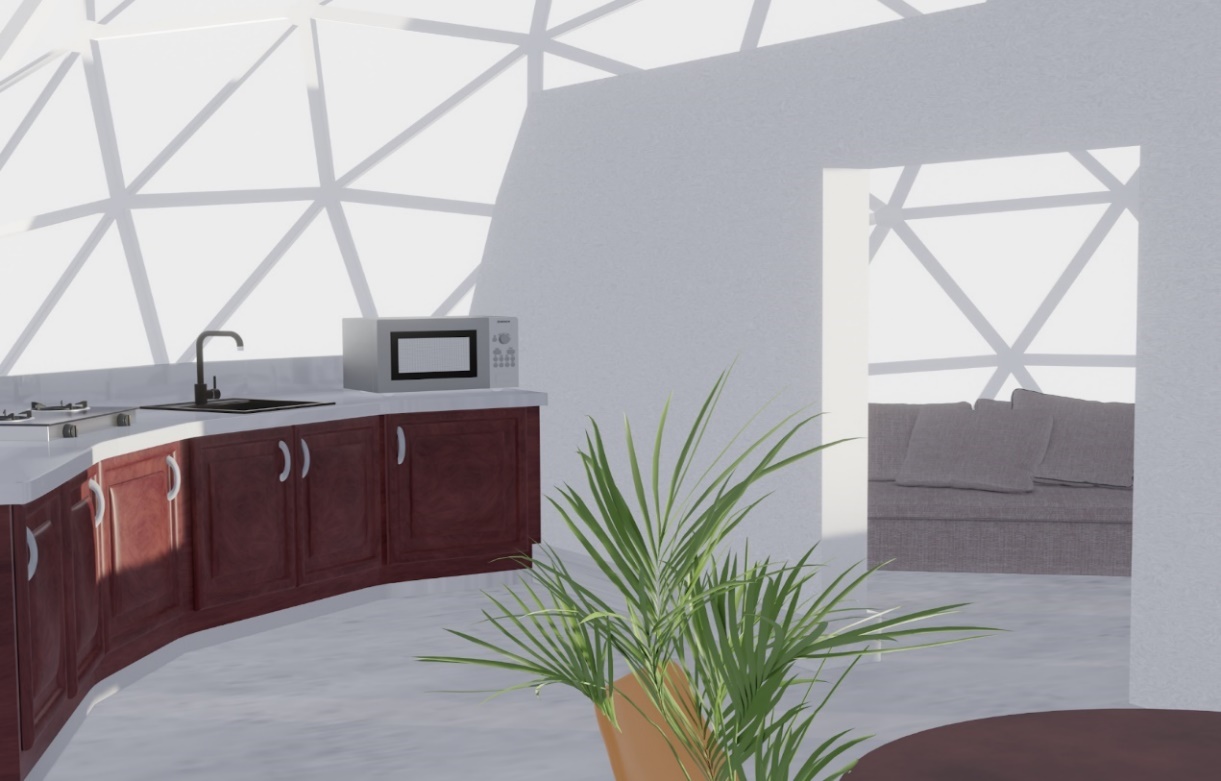
****

Figure 2 Kitchen

* Dining Room

****

Figure 3 Dining Room

* Living Room  
  ****

Figure 4 Living Room

* Bedroom

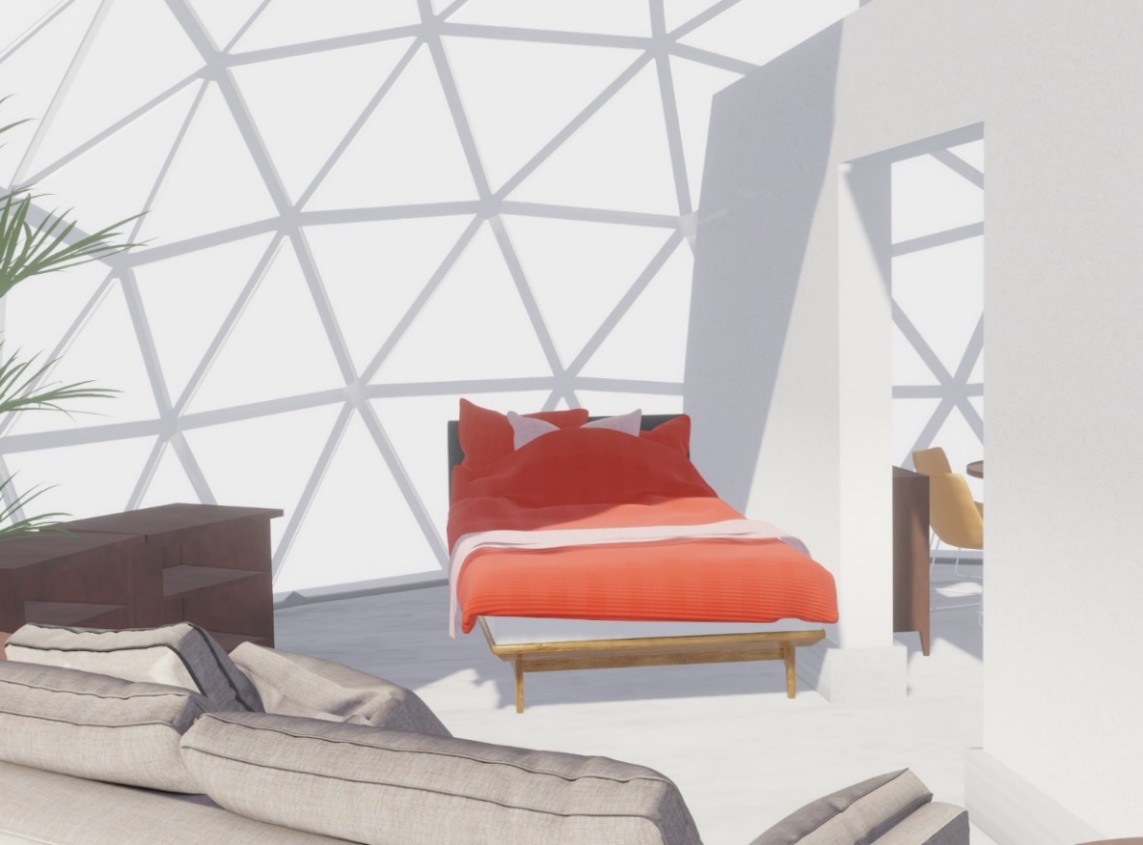


Figure 5 Bedroom

1. **THEMATIQUE**

Le besoin exprimé par le laboratoire est de développer et d’analyser un système de suivi basé sur la détection de mouvement. Pour ce faire, on va placer 12 détecteurs hc-sr04 sur différents angles de la maison tout en les connectant à une carte Arduino qui va par la suite envoyer les données au NodeMCU Esp32.

Par la suite on va utiliser un raspberry pi comme un broker, ce dernier va faire une subscription sur le NodeMCU qui va par la suite faire un publish des données récupérées au préalable sur le broker.

Après les données récupérées seront stockées dans une base de données MongoDB.

Cette base de données sera liée à notre application web pour la visualisation du suivi en temps réel.

Pour mieux comprendre la thématique, on va schématiser l’emplacement des capteurs sur les figures ci-dessous :

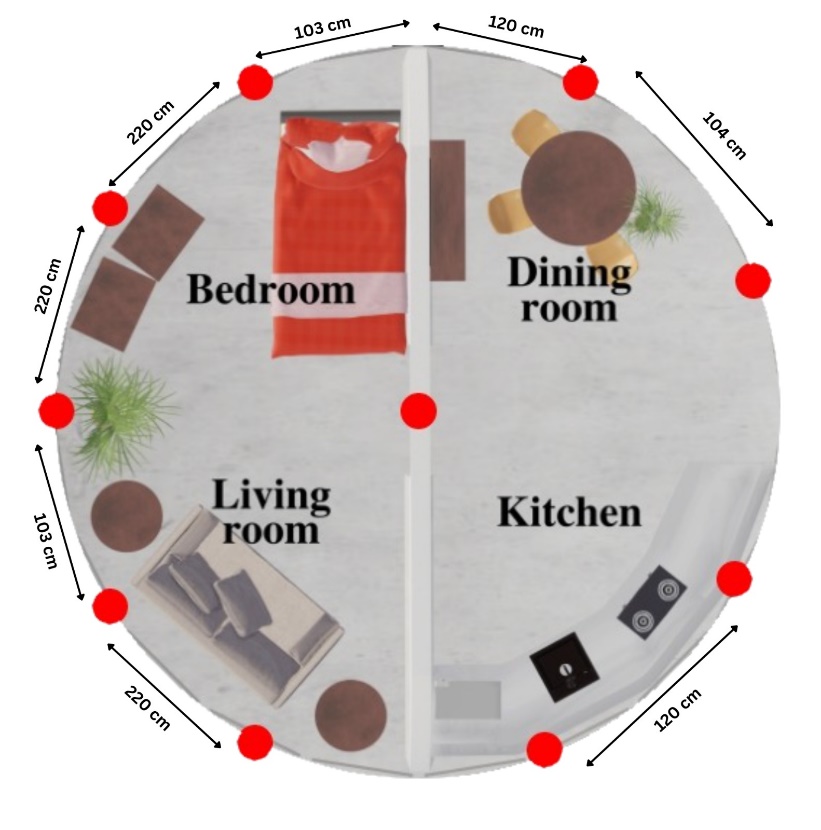


Figure Emplacement des capteurs dans le AiLAB.

Notons bien que les zones en rouges représentent les 12 capteurs qu’on doit mettre en place.

Sur notre prototype on a pu câbler la première partie de l’AiLAB (kitchen et Dining room) et voici le câblage sur le plan réel :



Figure Câblage Kitchen.



Figure Câblage Dining Room

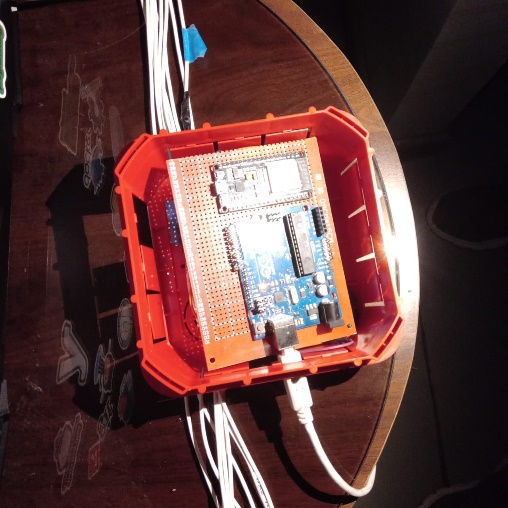


Figure Procédure câblage.

1. **CONCLUSION**

Le présent chapitre a été consacré à un aperçu général sur l’AILAB, ses caractéristiques, son architecture ainsi que le contexte pédagogique et l'intérêt du projet.

Dans le chapitre suivant, une bibliographie sera présentée sur les méthodes qu’on va utiliser pour réussir ce projet.

# CHAPITRE 2

# Etude bibliographique

1. **INTRODUCTION**

Dans ce chapitre, nous allons examiner tous les matériels logiciels existants afin que nous puissions avoir une idée sur la façon de procéder pour résoudre ce problème, et dire que notre projet a trois composantes principales, tout d'abord détection du mouvement à travers l’écho d’un signal sonore, ensuite stocker les mouvements détectés au préalable dans une base de données et faire un suivi de ce mouvement pour toute personne existante dans le AI LAB.

1. **Etude d’existant**

L'étude de l'existant d'un projet de suivi intelligent des occupants vise à évaluer les solutions existantes pour surveiller les activités des occupants dans le AI LAB. Cela peut inclure l'analyse des technologies telles que les capteurs de mouvement.

L'objectif de cette étude est de déterminer les solutions les plus efficaces et les plus adaptées aux besoins du projet, ainsi que d'identifier les éventuels défis et opportunités liés à la mise en œuvre de ces solutions. Il est également important de prendre en compte les aspects de la vie privée et de la protection des occupants.

L'étude de l'existant est une étape cruciale pour le développement réussi d'un projet de suivi intelligent des occupants, en permettant de comprendre les solutions disponibles, les opportunités et les défis liés à la mise en œuvre de ces solutions, et en garantissant que les solutions choisies sont conformes aux lois et aux normes en matière de protection de la vie privée et des données.

### Solution :

Afin de résoudre ce problème, on a pensé à adopter un système de suivi qui utilise des technologies avancées, pour suivre les mouvements et les données des personnes. Une solution qui, à la fois, résoudra le problème du temps et de la sécurité.

### Spécification des besoins :

### Les besoins fonctionnels :

Le système de smart tracking peuvent varier en fonction de l'application, mais certains besoins courants incluent :

* La localisation précise : le système de smart tracking utilise généralement des technologies de localisation, pour suivre les mouvements d'une personne.
* La collecte de données : le système de smart tracking collecte des données sur la vitesse, la distance parcourue.
* La visualisation de suivi : Application web pour visualiser le suivi.
* La sécurité : Ce système garantit la sécurité des données et des communications pour protéger la confidentialité des utilisateurs.

### Les besoins non fonctionnels :

Les besoins non fonctionnels d'un système de smart tracking peuvent inclure :

* La disponibilité : Les systèmes de smart tracking doivent être disponibles en permanence pour assurer un suivi en continu.
* La scalabilité : Capables de gérer un grand nombre d'utilisateurs et de dispositifs connectés.
* La performance : Rapides et réactifs pour assurer une expérience utilisateur satisfaisante.
* La fiabilité : Fiables pour garantir que les données collectées sont exactes et fiables.
* La sécurité : Garantir la sécurité des données et des communications pour protéger la confidentialité des utilisateurs.
* La flexibilité : Flexibles pour s'adapter aux besoins changeants et évolutifs de l'application.

### Environnement matériel :

L’équipement mis à notre disposition pour la réalisation du projet se compose :

* ESP32
* 12 capteurs : hc-sr04
* Raspberry Pi
* 2 Arduino Uno
* Écran LCD

1. **Les différents capteurs pour le suivi des mouvements :**

### Hc-sr04

Le capteur HC-SR04 est un capteur de distance ultrasonique communément utilisé pour mesurer les distances avec une précision relativement élevée. Il fonctionne en émettant un signal ultrasonique à haute fréquence, qui est ensuite réfléchi par les objets environnants et capté par le capteur. En mesurant le temps entre l'émission du signal et la réception de son écho, le capteur peut calculer la distance à l'objet.



Figure 10 hc-sr04

Il est souvent utilisé dans des applications de robotique, de capteurs de proximité, de détection de mouvement et de mesure de distance pour les projets de bricolage. Il est facile à utiliser car il n'a qu'un minimum de composants (émetteur, récepteur et un microcontrôleur) et il est disponible à un prix abordable. Il est généralement connecté à un microcontrôleur ou à un ordinateur pour la lecture des données de distance.

### PIR :

Le capteur PIR (Passive Infrared) est un capteur de mouvement communément utilisé pour détecter les mouvements humains ou animaux. Il fonctionne en mesurant les variations de température dans son champ de vision, causées par les mouvements des objets chauds (comme les humains et les animaux).

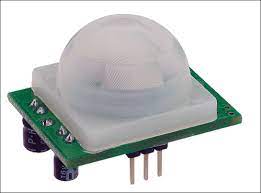


Figure 11 PIR

Il est souvent utilisé dans des applications de sécurité, de domotique, de surveillance et de contrôle d'éclairage. Il est facile à utiliser car il n'a besoin que de quelques composants pour fonctionner (le capteur PIR et un microcontrôleur) et il est disponible à un prix abordable. Il est généralement connecté à un microcontrôleur ou à un ordinateur pour la lecture des données de mouvement.

Il est important de noter que le capteur PIR ne détecte pas directement les mouvements, il détecte les variations de température qui accompagnent les mouvements, et il peut donc être affecté par des sources de chaleur environnantes, comme des radiateurs, des climatiseurs, des appareils électroniques, des lumières, etc. Il est également important de noter que ces capteurs ne fonctionnent pas à travers les murs ou les objets et leur champ de vision est généralement limité.

### Comparaison

Tableau 1 hc-sr04 et PIR

|  |  |
| --- | --- |
| hc sr04 | PIR |
| Plage de détection | 2cm à 4m | 3m à 4m |
| Angle de détection | 15 degrés | 110 degrés |
| Temps de réponse | 2ms | 1 à 2s |
| Alimentation | 5V | 3V à 5V |
| Connectivité | Microcontrôleur ou ordinateur via un port série | Microcontrôleur ou ordinateur via un port série |
| Température de fonctionnement | -20 à +70 degrés Celsius | -20 à +70 degrés Celsius |

### Pourquoi HC-SR04 ?

Le capteur HC-SR04 est un capteur de distance ultrasonique qui mesure la distance en émettant un signal ultrasonique et en mesurant le temps entre l'émission et la réception de l'écho. Il est utilisé pour mesurer les distances avec une précision relativement élevée et est souvent utilisé dans des applications de robotique et de mesure de distance.

Le capteur PIR est un capteur de mouvement passif qui mesure les variations de température causées par les mouvements des objets chauds. Il est utilisé pour détecter les mouvements humains et animaux et est souvent utilisé dans des applications de sécurité, de domotique et de surveillance.

En comparant les deux capteurs, voici quelques points :

* Le capteur HC-SR04 mesure les distances avec une précision relativement élevée, tandis que le capteur PIR détecte les mouvements humains et animaux.
* Le capteur HC-SR04 a une plage de mesure plus courte que le capteur PIR.
* Le capteur HC-SR04 a un angle de détection plus étroit que le capteur PIR.
* Le capteur HC-SR04 a un temps de réponse plus rapide que le capteur PIR.
* Le capteur HC-SR04 est plus sensible aux réflexions d'écho et aux perturbations environnementales que le capteur PIR.

1. **Dispositif programmable**

### Arduino Mega et Arduino Uno

### Arduino Mega2560

L'Arduino Mega2560 est une carte de développement électronique basée sur le microcontrôleur ATmega2560. Il dispose de 54 entrées/sorties numériques, 16 entrées analogiques, 4 ports série, une horloge à quartz de 16 MHz, et est compatible avec les logiciels de développement Arduino. Il est souvent utilisé pour les projets de contrôle de moteur, de robotique et d'automatisation.

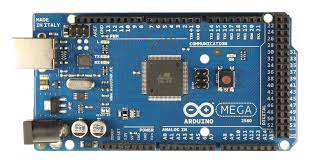


Figure 12 Arduino Mega2560

### Arduino Uno

L'Arduino Uno est une carte de développement électronique basée sur le microcontrôleur ATmega328P. Il dispose de 14 entrées/sorties numériques, 6 entrées analogiques, 1 port série, une horloge à quartz de 16 MHz, et est compatible avec les logiciels de développement Arduino. Il est souvent utilisé pour les projets de débutant en électronique et robotique, et pour les projets de domotique, de capteur et d'IoT. Il est considéré comme un des modèles les plus populaires de la famille Arduino.



Figure 13 Arduino Uno

### Comparaison

Tableau 2 Arduino Uno et Arduino Mega

|  |  |  |
| --- | --- | --- |
|  | **Arduino Uno** | **Arduino Mega** |
| **Microcontrôleur** | Atmega328P | Atmega2560 |
| **Fréquence d’horloge** | 16 MHz | 16 MHz |
| **Entrées/sorties numériques** | 14 | 54 |
| **Entrées analogiques** | 6 | 16 |
| **Ports série** | 1 | 4 |
| **Mémoire flash** | 32 KB | 256 KB |
| **SRAM** | 2 KB | 8 KB |
| **EEPROM** | 1 KB | 4 KB |

### Pourquoi Arduino Uno ?

Il y a plusieurs raisons pour lesquelles nous avons choisi l'Arduino Uno plutôt que l'Arduino Mega pour notre projet :

* Taille et coût : L'Arduino Uno est plus petit et moins cher que l'Arduino Mega, ce qui en fait une option plus abordable pour les projets à petit budget.
* Nombre d'entrées/sorties : Notre projet n'a pas besoin de beaucoup de ports d'entrée/sortie, l'Arduino Uno pourrait suffire. Il a 14 ports d'entrée/sortie, tandis que l'Arduino Mega en a 54.
* Mémoire : L'Arduino Uno a moins de mémoire flash et SRAM que l'Arduino Mega. Notre projet n'a pas besoin de beaucoup de mémoire, l'Arduino Uno pourrait suffire.
* Taille de la communauté : L'Arduino Uno est le plus populaire des deux, il y a donc plus de bibliothèques, de tutoriels et de documentation disponibles pour l'Arduino Uno que pour l'Arduino Mega.
* Facilité d'utilisation : L'Arduino Uno est plus facile à utiliser, car il a une courbe d'apprentissage plus plate que l'Arduino Mega.

### ESP32 et NodeMCU

### ESP32

L'ESP32 est un microcontrôleur de la société Espressif qui est basé sur l'architecture Xtensa. Il est souvent utilisé dans les projets IoT (Internet des objets) en raison de ses fonctionnalités de connectivité sans fil, telles que WiFi et Bluetooth. Il est également populaire pour ses capacités de traitement et de mémoire élevées, ainsi que pour sa faible consommation d'énergie.



Figure 14 ESP32

### NodeMCU

NodeMCU est un système sur puce (SoC) basé sur l'ESP8266, un microcontrôleur Wi-Fi développé par Espressif Systems. NodeMCU est un projet open-source qui ajoute un firmware à l'ESP8266 pour faciliter l'utilisation de ce dernier en lui fournissant une interface de programmation basée sur Lua Script. Cela permet aux utilisateurs de contrôler facilement les fonctionnalités de l'ESP8266, telles que la connectivité Wi-Fi, les entrées/sorties, les capteurs, etc.

NodeMCU est également équipé d'un certain nombre de modules de communication intégrés tels que I2C, SPI, UART, PWM, et GPIO pour permettre une communication avec d'autres composants électroniques.

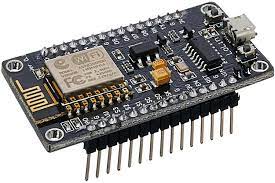


Figure 15 NodeMCU

### Comparaison

Tableau 3 NodeMCU et ESP32

|  |  |  |
| --- | --- | --- |
|  | **NodeMCU** | **ESP32** |
| **Processeur** | ESP8266 (80 MHz à 160 MHz) | Dual-Core Xtensa LX6 (240 MHz) |
| **Mémoire** | 64 Ko de SRAM, 4 Mo de mémoire flash | 520 Ko de SRAM, 4 Mo de mémoire flash |
| **Connectivité** | WiFi 802.11b/g/n (2,4 GHz) | WiFi 802.11b/g/n (2,4 GHz) et Bluetooth 4.2 |
| **Entrées/Sorties** | 16 broches | 40 broches |
| **Alimentation** | 3.3 V via un port micro-USB ou une batterie | 3.3 V via un port micro-USB ou une batterie |
| **Dimensions** | Compacte, environ 34 mm x 18 mm | Compacte, environ 25 mm x 18 mm |

### Pourquoi ESP32 ?

* Puissance de traitement : l'ESP32 est plus puissant que le NodeMCU, il a deux cœurs de processeur contrairement à un seul cœur pour le NodeMCU, ce qui le rend plus adapté pour les projets nécessitant des calculs intensifs.
* Connectivité : l'ESP32 a une connectivité Wi-Fi et Bluetooth intégrée, alors que le NodeMCU utilise un module ESP8266 qui a seulement la connectivité Wi-Fi.
* Mémoire : L'ESP32 a plus de mémoire flash et SRAM que le NodeMCU, ce qui le rend plus adapté pour les projets nécessitant beaucoup de mémoire.
* Durabilité : l'ESP32 est plus robuste et durable que le NodeMCU, il peut résister à des températures plus élevées et à des conditions environnementales plus extrêmes.
* Soutien de la communauté : bien que le NodeMCU soit populaire, l'ESP32 a une communauté de développement en croissance et de plus en plus de bibliothèques et de documentation disponibles.

### Beagle Bone et Raspberry pi

### Beagle Bone

BeagleBone Black est un ordinateur de développement open-source basé sur un processeur Texas Instruments AM3358 (ARM Cortex-A8). Il est conçu pour les projets électroniques et informatiques, et il est souvent utilisé dans les applications de contrôle de moteur, de capteurs et de réseaux. Il est une alternative au Raspberry Pi, similaire en termes de prix et de fonctionnalités. Il est plus performant que le Raspberry Pi mais avec un peu moins de communauté, il est donc moins populaire. Il possède une connectivité Ethernet, USB, et un grand nombre d'entrées-sorties (GPIO) pour connecter des capteurs et des actionneurs. Il est souvent utilisé pour les projets de robotique, de vision par ordinateur, d'apprentissage automatique et de contrôle industriel.



Figure 16 Beagle Bone

### Raspberry pi

Le Raspberry Pi est un ordinateur de la taille d'une carte de crédit qui utilise un système d'exploitation basé sur Linux. Il a été développé par la fondation Raspberry Pi pour promouvoir l'enseignement de l'informatique dans les écoles. Il est équipé d'un processeur, de mémoire vive, de ports USB et Ethernet, et peut être connecté à un écran, un clavier et une souris pour fonctionner comme un ordinateur de bureau traditionnel. Il est également populaire auprès des développeurs et des passionnés de la technologie pour ses capacités de programmation et de contrôle de matériel.



Figure 17 Raspberry pi

### Comparaison

Tableau 4 Beagle Bone et Raspberry Pi

|  |  |  |
| --- | --- | --- |
|  | **Beagle Bone** | **Raspberry pi** |
| **Processeur** | Texas Instruments AM3358 Cortex-A8 à 1 GHz | Broadcom BCM2711, Cortex-A72 Quad Core à 1,5GHz |
| **Mémoire RAM** | 512 Mo DDR3L | 2 Go, 4 Go ou 8 Go LPDDR4-3200 SDRAM (selon le modèle) |
| **Stockage** | 4 Go de mémoire flash eMMC intégrée | Slot pour carte SD (jusqu'à 256 Go) |
| **Connectivité** | Ethernet Gigabit, USB 2.0 client, USB 2.0 hôte, HDMI, et ports série | Ethernet Gigabit, Wi-Fi 802.11ac, Bluetooth 5.0, BLE, 2 ports USB 3.0, 2 ports USB 2.0, 2 ports micro-HDMI |
| **Entrées/sorties** | 65 broches | 40 broches |
| **Alimentation** | Entrée DC de 7 à 15 V avec une consommation électrique de 0,5 A à 1 A | Entrée DC de 5V via un connecteur USB-C (minimum 3A) |
| **Système d'exploitation** | Linux embarqué pré-installé | Linux ou Windows 10, à installer par l'utilisateur |
| **Dimensions** | 3,4 x 2,1 pouces | 88 x 58 x 19,5 mm |

### Pourquoi Raspberry pi ?

Il y a plusieurs raisons pour lesquelles on a choisi le Raspberry Pi plutôt que le BeagleBone :

* Coût : Le Raspberry Pi est généralement plus abordable que le BeagleBone, ce qui en fait un choix plus économique pour les projets à petit budget.
* Facilité d’utilisation : Le Raspberry Pi est conçu pour être facile à utiliser, même pour les débutants. Il a un grand nombre de tutoriels et de projets disponibles en ligne, ainsi qu'une communauté active pour aider les utilisateurs à résoudre les problèmes. Le BeagleBone, bien qu'il soit plus puissant, peut-être plus difficile à utiliser pour les utilisateurs débutants.
* Popularité : Le Raspberry Pi est très populaire et largement utilisé dans les projets éducatifs, les projets de robotique, les projets de média center, etc. Il existe donc une grande quantité de documentation, de tutoriel, de codes sources et de communauté pour l'utiliser.
* Taille : Le Raspberry Pi est plus petit que le BeagleBone, ce qui le rend plus facile à intégrer dans les projets de robotique, les projets de domotique, etc.
* Système d’exploitation : Le Raspberry Pi utilise un système d'exploitation basé sur Linux qui peut être installé par l'utilisateur, ce qui permet de choisir son système d'exploitation préféré. Le BeagleBone, quant à lui, utilise un système d'exploitation embarqué pré-installé.

1. **Base de données**

### MongoDB

MongoDB est un système de gestion de base de données NoSQL, c'est-à-dire qu'il ne suit pas les conventions d'un système de gestion de base de données relationnelle classique (comme MySQL ou PostgreSQL). Au lieu de cela, MongoDB utilise un format de données orienté document qui permet de stocker des données sous forme de documents au lieu de lignes et de colonnes. Il est souvent utilisé pour les applications web et les projets de données volumineux.

Figure 18 MongoDB

### Pourquoi MongoDB ?

Les raisons pour lesquelles on a choisi MongoDB pour leurs projets :

* Flexibilité de la structure de données : MongoDB utilise un format de données orienté document, ce qui permet de stocker des données sous forme de documents JSON plutôt que de lignes et de colonnes. Cela permet de stocker des données de différentes formes et tailles dans la même collection.
* Évolutivité : MongoDB est conçu pour gérer des données volumineuses, il est facilement évolutif en termes de stockage et de puissance de calcul.
* Performance : MongoDB utilise un système d'indexation intégré qui permet de rechercher rapidement des données dans les collections.
* Facilité d'utilisation : MongoDB est facile à apprendre et à utiliser, il a une syntaxe simple et intuitive pour les requêtes, et il est compatible avec de nombreux langages de programmation populaires.
* Réplication et haute disponibilité : MongoDB prend en charge la réplication de données à plusieurs nœuds, ce qui permet de garantir la haute disponibilité et la tolérance aux pannes.

1. **CONCLUSION**

Donc pour conclure, nous avons examiné certaines des principales composantes du projet, dans le but de bien connaître le processus, et d'arriver à un point où nous pouvons décider quelle méthode est la meilleure et la plus efficace.

# CHAPITRE 3

# Développement & analyse

* 1. **INTRODUCTION**

Dans cette partie nous présentons la procédure de réalisation de notre système de  suivi intelligent des utilisateurs. Ce système permettra aux utilisateurs de suivre leur mouvement physique au sein de AI LAB.

Les données seront collectées par des capteurs intégrés au sein de chaque chambre de AiLAB et transmises à MongoDB. Les utilisateurs pourront accéder aux données à travers une application web.

* 1. **LES MISSIONS EFFECTUÉES**

### Besoins et objectifs du projet :

Les besoins et les objectifs d'un projet de suivi intelligent des utilisateurs peuvent inclure :

* **Détection de la position :** Il est important de pouvoir détecter la position de l'utilisateur pour pouvoir suivre ses mouvements. Les capteurs de localisation tels que les capteurs GPS, les capteurs de proximité et les capteurs de balises Bluetooth peuvent être utilisés pour cela.
* **Suivi en temps réel :** Il est important d'être en mesure de suivre les utilisateurs en temps réel pour pouvoir réagir rapidement en cas de besoin.
* **Stockage des données :** Il est important de pouvoir stocker les données de suivi des utilisateurs pour pouvoir les analyser ultérieurement. Les modules de stockage de données tels que les cartes SD peuvent être utilisés pour cela.
* **Communication sans fil :** Il est important d'être en mesure de transmettre les données de suivi des utilisateurs à un serveur ou à une application pour pouvoir les consulter à distance. Les modules de communication sans fil tels que les modules WiFi, les modules GSM et les modules LoRa peuvent être utilisés pour cela.

### Architecture Hardware :

L’implémentation du projet Smart tracking of users est comme suit :

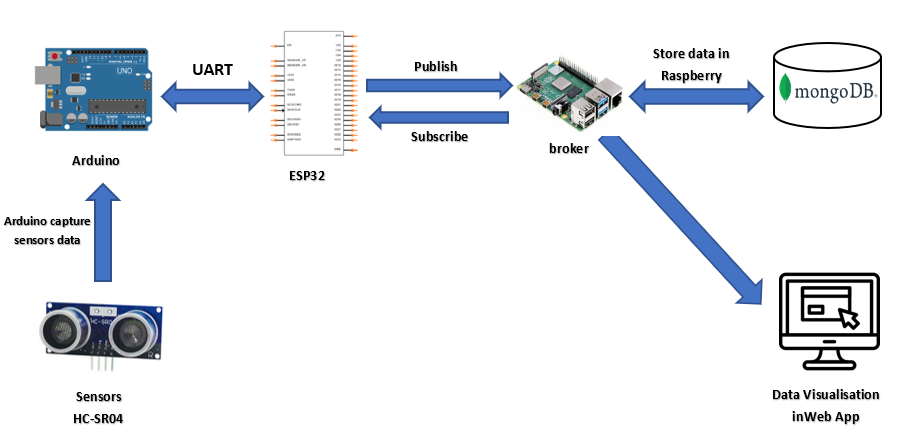
****

Figure 19: Implémentation du projet

|  |  |
| --- | --- |
| **Composants** | **Qté** |
| **Capteur HC-SR04** | 12 |
| **Arduino** | 2 |
| **ESP32** | 2 |
| **Broker Raspberry** | 1 |
| **Écran LCD** | 1 |

**Capteur HC-SR04 :** Ce capteur est utilisé pour mesurer la distance entre lui et un objet. Il est relié au ESP32 ou Arduino pour collecter les données et les envoyer au Broker.

**ESP32, Arduino :** Ces microcontrôleurs sont utilisés pour collecter les données des capteurs HC-SR04, les traiter et les envoyer au Broker Raspberry pour une utilisation ultérieure. Il peut être utilisé pour lire les données des capteurs, effectuer des calculs, envoyer les données à un serveur ou une base de données.

**Broker Raspberry :** Il sert de point central pour la réception, la distribution, et l'envoi des données collectées par les capteurs. Il peut être utilisé pour stocker les données collectées dans une base de données pour une utilisation ultérieure. Il peut également être utilisé pour effectuer des calculs en temps réel sur les données et pour envoyer des alertes.

**Écran LCD :** Il est utilisé pour afficher les informations récupérées par les capteurs HC-SR04. Il peut être utilisé pour afficher les données en temps réel, l'historique des données, les alertes, etc.

### Architecture Software :

Les logiciels de traitement de données sont utilisés pour analyser les données collectées par les capteurs. Ceux utilisés sont présentés dans le schéma ci-dessous :

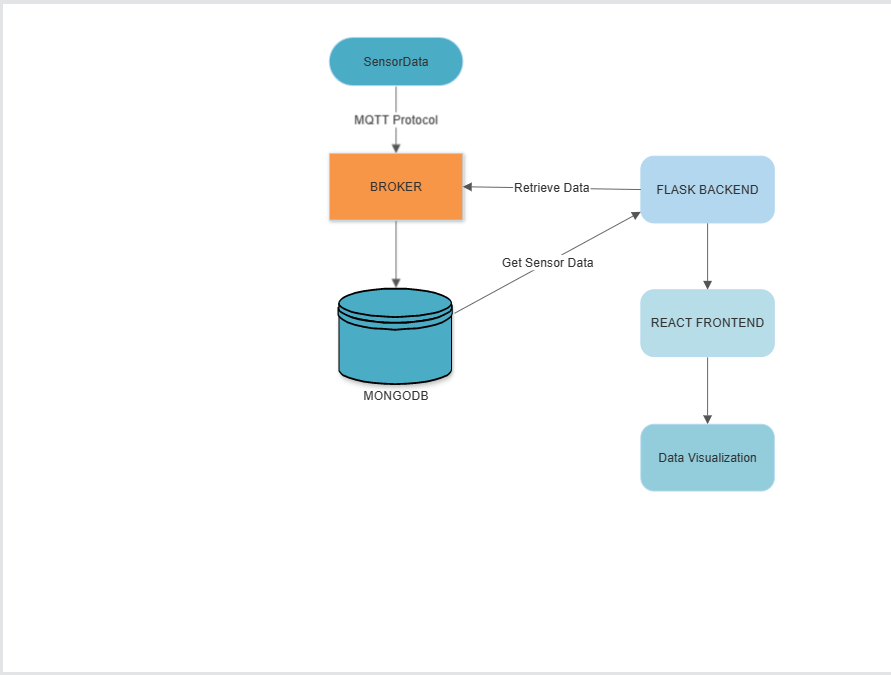


Tableau 5 Architecture Software

* **Arduino** (écrire le code pour obtenir les données collectées par les capteurs HC-SR04).
* **MongoDB** (stocker les données dans une base de données).
* **Reactjs** (sous forme d’une application WEB qui affiche les données obtenues).
* **Flask** (Framework de web python).

### Schéma Fritzing :

Pour concevoir le schéma électronique de votre projet IoT nous avons utilisé Fritzing pour pouvoir visualiser facilement comment les différents composants sont connectés les uns aux autres et identifier les erreurs éventuelles avant de construire le prototype physique.

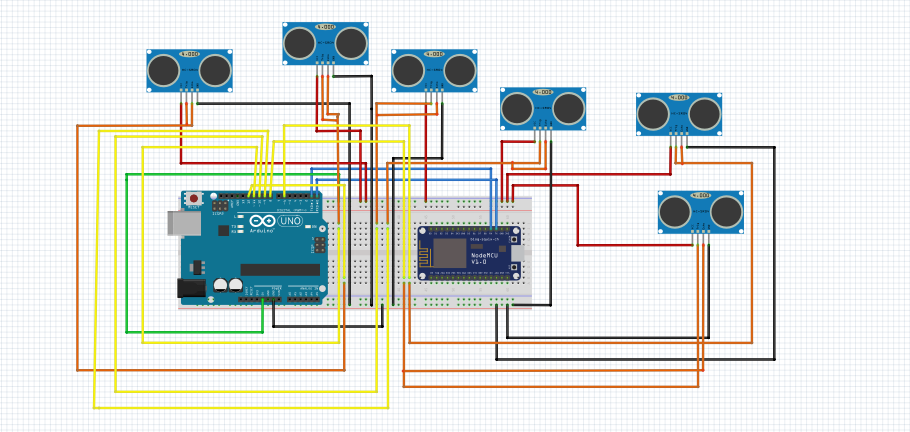


Figure 20: Schéma Fritzing

L’image ci-dessus représente le schéma électronique montre comment les différents composants sont connectés les uns aux autres.

Tableau 6: Rôles des câbles schématisés

|  |  |
| --- | --- |
| **Les câbles** | **Rôles** |
|  | Alimentation des capteurs |
|  | Alimentation de l’Arduino |
|  | Les pins des capteurs |
|  | Ground |
|  | RX & TX |

* 1. **REALISATION**

### Arduino et ESP32 :

La partie développement de l'Arduino et de l'ESP32 utilisant une communication UART (Universal Asynchronous Receiver / Transmitter) dans notre projet de suivi intelligent des utilisateurs peut inclure les étapes suivantes :

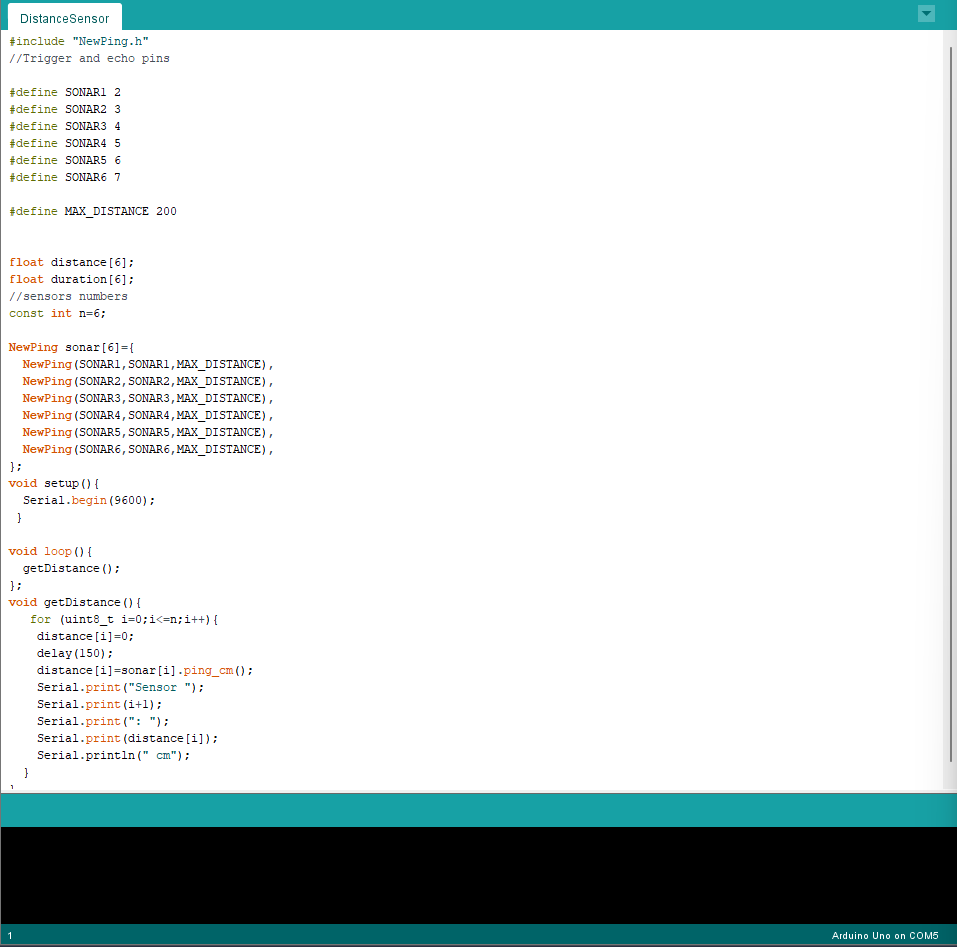
* **Conception du circuit :** Cette étape consiste à concevoir le circuit électronique qui connecte les capteurs HC-SR04, l'ESP32 et l'Arduino. Il est important de vérifier que tous les composants sont correctement connectés et fonctionnent correctement.
* **Écriture du code Arduino :** Cette étape consiste à écrire le code Arduino qui permet de collecter les données des capteurs HC-SR04 et de les envoyer à l'ESP32 en utilisant une communication UART. Il est important de vérifier que les données sont collectées correctement et sont envoyées à l'ESP32.

Figure 21: Code Arduino

* **Écriture du code ESP32 :** Cette étape consiste à écrire le code ESP32 qui permet de traiter les données reçues de l'Arduino en utilisant une communication UART et de les envoyer à Raspberry Broker.
* **Tests :** Cette étape consiste à tester le système en utilisant des données réelles pour vérifier que toutes les fonctionnalités sont implémentées correctement. Il est important de tester toutes les fonctionnalités clés comme la collecte de données, l'analyse de données, la prise de décision, la communication avec les utilisateurs et les systèmes externes. Il est également important de documenter les résultats des tests, les éventuels problèmes rencontrés et les solutions apportées. Il est important de valider si les fonctionnalités clés ont été implémentées correctement et si les résultats obtenus répondent aux exigences et aux objectifs du projet.



Figure 22: Code Arduino

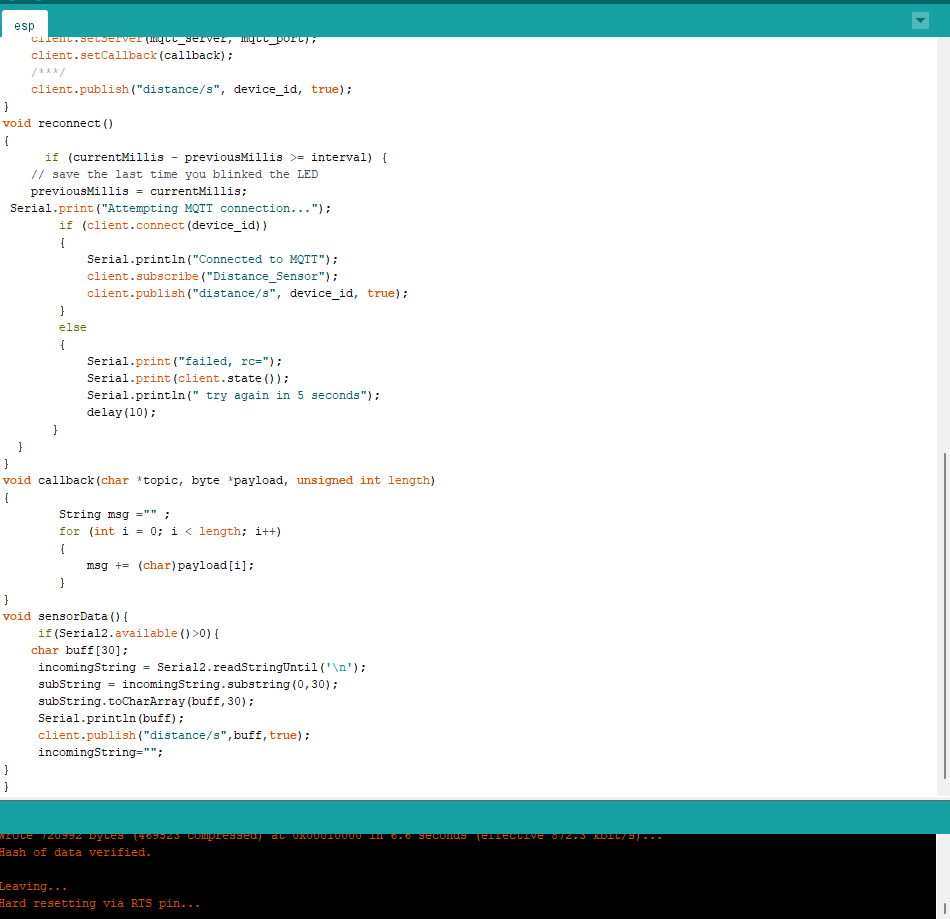


Figure 23: Code Arduino

* + 1. **Raspberry Broker :**

**Conception du Raspberry Broker :** Nous avons choisi d'utiliser un Raspberry Pi comme broker pour gérer les données en provenance des capteurs et des microcontrôleurs.

**Protocoles de communication :** Nous avons utilisé le protocole MQTT qui est un protocole de messagerie public/abonné qui permet aux clients de publier des messages sur des canaux (appelés "sujets") auxquels d'autres clients s'abonnent pour recevoir ces messages. Le backend peut utiliser MQTT pour recevoir des données de capteur et les publier dans un sujet spécifique. Les applications frontales peuvent ensuite s'abonner à cette rubrique pour recevoir des mises à jour de données en temps réel.

* **Stockage des données :**

Il est important de pouvoir stocker les données de suivi des utilisateurs pour pouvoir les analyser ultérieurement. Pour stocker les données des capteurs, nous avons utilisé MongoDB comme base de données NoSQL.

**3.** **La visualisation des données :**

La partie backend consiste à utiliser Flask pour créer des API RESTful et MQTT pour transmettre des données. Flask permet de définir des chemins pour accéder à des données spécifiques. Cela signifie que lorsque l’application ReactJs envoie une requête à l'API, Flask peut renvoyer les données demandées au format JSON.

La partie frontend utilise React.js pour créer une interface utilisateur en temps réel. React utilise un modèle de données unique pour mettre à jour automatiquement l'interface utilisateur en fonction des modifications apportées aux données. L'application frontend utilise également une boucle de rappel "useEffect" pour actualiser les données de manière périodique en utilisant une requête fetch vers l'API Flask.

React met à jour l'interface utilisateur en modifiant l'état de chaque point (capteur) lorsque les données sont actualisées. Chaque capteur représenté sur l’interface s’allumera lorsqu’il détectera une personne . La position de la personne sera déterminée à partir des coordonnées (x,y) spécifiées dans les données, ces coordonnées seront calculées en utilisant les formules :

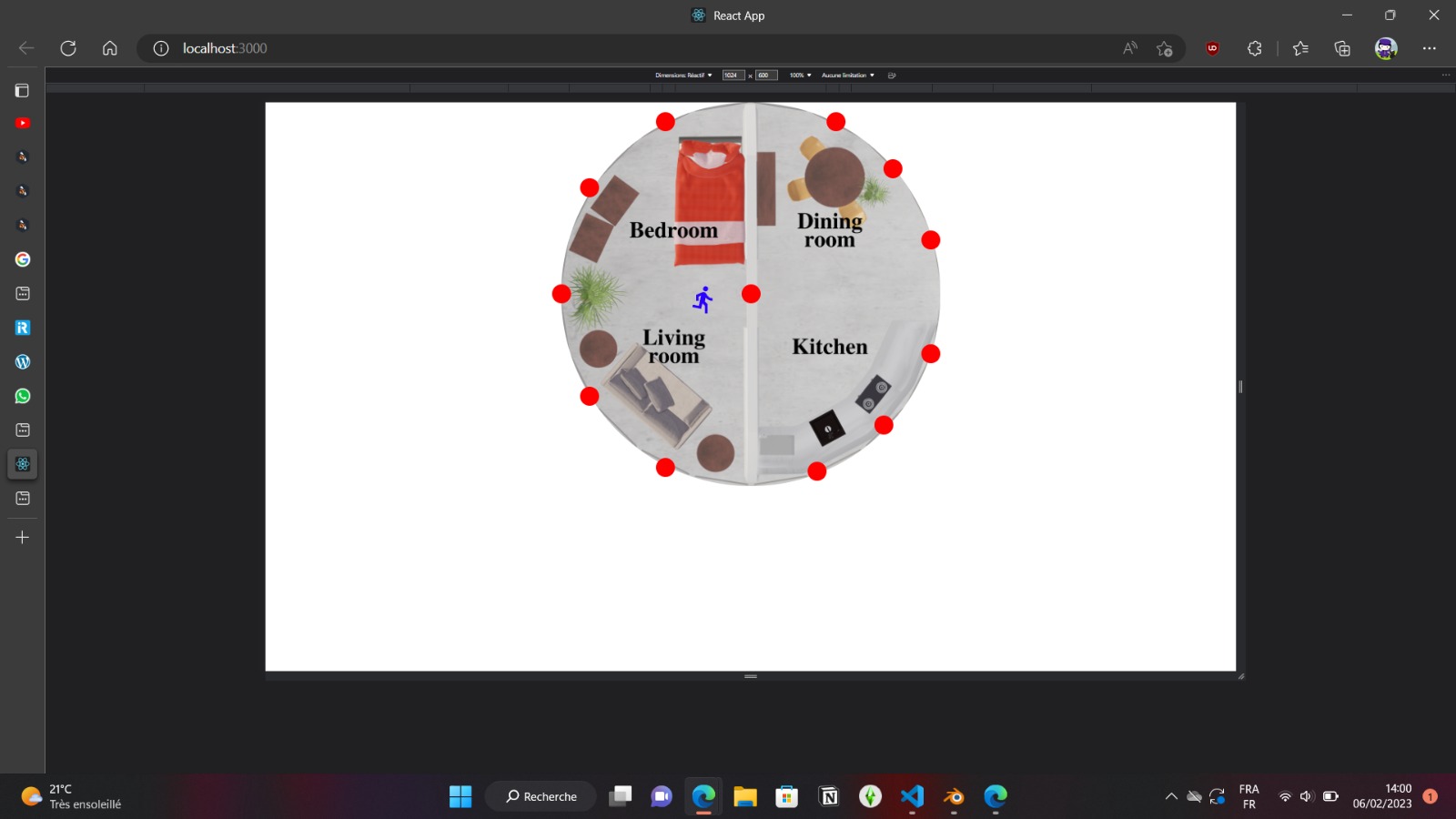


Figure 24: Interface de l'application

* 1. **Conclusion**

La partie développement a été cruciale pour la réalisation de ce projet de suivi intelligent des utilisateurs. Nous avons utilisé des technologies telles que Arduino et ESP32 pour la collecte de données à partir des capteurs, utilisé le Raspberry Broker pour la communication entre les différents éléments du système, utilisé MongoDB pour stocker les données et React JS pour la visualisation des données. Tous ces éléments ont été développés et testés avec soin pour s'assurer qu'ils fonctionnent correctement et contribuent à la réalisation des objectifs du projet.

# CONCLUSION GENERALE

Ce projet s’est révélé très enrichissant dans la mesure où il a consisté en une approche concrète du métier d’ingénieur. En effet, la prise d’initiative, le respect des délais et le travail en équipe seront des aspects essentiels de notre futur métier.

Dans Le cadre général du Projet de fin de session et son implication dans le cursus des études d'ingénierie ainsi que dans la vie professionnelle et personnelle. Nous avions eu l’occasion d’acquérir des nouvelles compétences dans notre carrière afin de pouvoir réaliser un tel travail collectif. Ce projet consiste à réaliser un système de suivi basé sur la détection des mouvements à travers des dispositifs électroniques qui communiquent entre eux en admettant le protocol UART, une base de données et une page web.

Nous résumons ce que nous avons fait dans le 2ème chapitre, et pourquoi nous l'avons fait, nous avons examiné certaines des principales composantes du projet, dans le but de bien connaître le processus, et d'arriver à un point où nous pouvons décider quelle méthode est la meilleure et la plus efficace.

Le projet a réussi à mettre en place un système de suivi basé sur la détection de mouvement. Les capteurs de mouvement installés ont permis de détecter les mouvements dans les zones ciblées et de stocker les données capturées dans une base de données. Le projet a également montré que la détection de mouvement est un moyen efficace de surveiller les zones et de protéger les personnes et les biens. Il est donc recommandé de continuer à explorer et à développer

Nous avons compris qu’il est important d’évaluer les connaissances de chacun, et de faire un point sur nos tâches régulièrement. Notre organisation était plutôt bonne. Il y avait une bonne cohésion de groupe. Nous avons appris à gérer un projet et à faire face aux difficultés ensemble.

De l’avis général, nous avons consolidé nos connaissances générales et appris à faire des applications plus attrayantes et plus orientées pour le monde du travail. Nous sommes globalement satisfaits de ce que nous avons réalisé. Au niveau de la gestion du projet en équipe, nous avons réussi à bien nous répartir les tâches afin de réaliser nos objectifs dans les temps et l'ambiance générale du groupe était une très bonne expérience à renouveler.

Ce projet nous a aussi offert la chance de découvrir et travailler dans un contexte réel et affronter de vraies problématiques. Ces dernières nous ont aidé à acquérir une expérience très riche et diversifiée. Ainsi, on a travaillé avec de différents composants électroniques complexes en termes d’établissement de communication et transmission de données et avec tout ce que cela implique en termes de complexité du projet.

Durant ce projet, on a pu raffiner nos capacités d’abstraction et de conception ainsi que notre méthodologie de travail, on a été confronté à plusieurs contraintes et difficultés qu’on a pu surmonter. On a aussi développé nos connaissances métiers. Par ailleurs, ce travail nous a permis d’acquérir une double compétence, technique et une culture de bonnes pratiques.

Parmi les perspectives ouvertes à ce projet, l’utilisation d’autres méthodes de reconnaissance, et la combinaison avec d’autre technologie biométrique comme l’empreinte digitale ou l’iris pour finaliser l’application multimodale et fiabiliser le système en diminuant la sensibilité aux conditions d’éclairage par de nouvelles normalisations.

Parmi les perspectives ouvertes à ce projet, l’intégration d’autres technologies comme la détection et reconnaissance de la parole, c’est à dire identifier les personnes à partir de leur voix, intégration de la reconnaissance faciale, intégration de la vidéo-analyse pour identifier les comportements suspects etc.

Parmi les perspectives qui concerne notre projet, il peut inclure l’amélioration de la détection du mouvement tout en ajoutant des algorithmes d’apprentissage automatique, ainsi la mise en place d’une plateforme de surveillance en cloud pour permettre à l’utilisateur de visualiser le suivi à distance des zones ciblées, ce qui peut être particulièrement utiles pour les sites industriels, les bâtiments commerciaux et les infrastructures critiques.

Il peut aussi inclure l’analyse de données avancés pour la détection des anomalies dans les données des mouvements sonores, ce qui peut aider à améliore la sécurité et à prévenir les incidents.

# BIBLIOGRAPHIE

Raspberry Pi Cookbook - Software and Hardware Problems and Solutions

Projets créatifs avec Arduino, Bruno Affagard, Jean-Michel Géridan, Jean-Noël Lafargue, éd

datasheet-capteur-ultrasons-hc-sr04.pdf

Home Appliance Hack-and-IoT Guidebook

# WEBOGRAPHIE

<https://forums.raspberrypi.com/viewtopic.php?t=273351>

<https://thedatafrog.com/fr/articles/mongodb-remote-raspberry-pi/>

<https://gist.github.com/LarsBergqvist/8d9414aeadf667216df81f41140aa72c>

<https://www.rgot.org/212-2/>

<https://www.gotronic.fr/art-module-de-detection-us-hc-sr04-20912.htm>

<https://fr.wikipedia.org/wiki/MongoDB>

<https://www.aranacorp.com/en/programming-an-esp32-nodemcu-with-the-arduino-ide/>

# ANNEXES

**Backend:**

from flask import Flask, request, jsonify,render\_template

from flask\_mqtt import Mqtt

from datetime import datetime

import pymongo

import uuid

from math import \*

app = Flask(\_\_name\_\_)

app.config['MQTT\_BROKER\_URL'] = "192.168.10.19"

app.config['MQTT\_BROKER\_PORT'] = 1883

app.config['MQTT\_KEEPALIVE'] = 5  # Set KeepAlive time in seconds

app.config['MQTT\_TLS\_ENABLED'] = False

ids=uuid.uuid1()

topic = "distance\_s"

session=pymongo.MongoClient("mongodb+srv://Moncef:12345@sensor.un2niwv.mongodb.net/?retryWrites=true&w=majority")

db=session.SensorData

collection=db.DistanceSensor

client = Mqtt(app)

captors = {

    'capteur\_1': {'x': 412, 'y': 10},

    'capteur\_2': {'x': 332, 'y': 80},

    'capteur\_3': {'x': 302, 'y': 192.2624969482422},

    'capteur\_4': {'x': 332, 'y': 300},

    'capteur\_5': {'x': 412, 'y': 375},

    'capteur\_6': {'x': 572, 'y': 379.5375061035156},

    'capteur\_7': {'x': 642, 'y': 330},

    'capteur\_8': {'x': 692, 'y': 255},

    'capteur\_9': {'x': 692, 'y': 135},

    'capteur\_10': {'x': 652, 'y': 60},

    'capteur\_11': {'x': 592, 'y': 10},

    'capteur\_12': {'x': 582, 'y': 192.2624969482422}

}

@client.on\_connect()

def handle\_connect(client, userdata, flags, rc):

    if rc == 0:

        print('Connected successfully')

        client.subscribe(topic) # subscribe topic

    else:

        print('Bad connection. Code:', rc)

@client.on\_message()

def handle\_mqtt\_message(client, userdata, message):

    global data

    if float(message.payload.decode()[10:])>=2:

        data = dict(

            capteur=message.payload.decode()[0:9],

            Distance=float(message.payload.decode()[10:]),

            time=datetime.today()

        )

@app.route("/data")

def index():

    xf = 0

    yf = 0

    dataa=[]

    for captor in data.keys():

        dataa=[]

        captor\_dict = captors[data['capteur']]

        print(captor\_dict)

        captor\_dict['Distance'] = data['Distance']

        captor\_dict['captor'] = data['capteur']

        xf = captor\_dict['x']

        yf = captor\_dict['y']

        d = captor\_dict['Distance']

        d = d\*0.6

        if captor\_dict['captor'] == "capteur\_1":

            y = d/(sqrt(1+tan(30)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf+x, 'y': yf+y,'name': captor\_dict['captor'], 'distance': d})

        elif captor\_dict['captor'] == "capteur\_2":

            y = d/(sqrt(1+tan(30)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf+x, 'y': yf+y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_3":

            y = d/(sqrt(1+tan(15)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf+x, 'y': yf,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_4":

            y = d/(sqrt(1+tan(15)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf+x, 'y': yf-y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_5":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf+x, 'y': yf-y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_6":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf-x, 'y': yf-y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_7":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf-x, 'y': yf-y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_8":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf-x, 'y': yf-y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_9":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf-x, 'y': yf+y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_10":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf-x, 'y': yf+y,'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_11":

            y = d/(sqrt(1+tan(60)\*\*2))

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf-x, 'y': yf+y, 'name': captor\_dict['captor']})

        elif captor\_dict['captor'] == "capteur\_12":

            y = d

            x = sqrt(d\*\*2-y\*\*2)

            dataa.append({'x': xf+x, 'y': yf , 'name': captor\_dict['captor']})

    return jsonify(dataa)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(host='127.0.0.1/', port=5000)

**Frontend:**

import Ai from "./Images/ai-lab.png"

import { useEffect, useState } from 'react';

import { Icon } from "@iconify/react"

const Map = () => {

  const [index, setIndex] = useState(0);

  const [data, setData] = useState([]);

// Using useEffect for single rendering

useEffect(() => {

  const intervalId = setInterval(() => {

  fetch("/data")

  .then(res => res.json())

  .then(data => {

  setData(data);

  })

  .catch(error => console.error("Error:", error));

  }, 1000);

  return () => clearInterval(intervalId);

  }, []);

  const [point1State, setPoint1State] = useState("default");

  const [point2State, setPoint2State] = useState("default");

  const [point3State, setPoint3State] = useState("default");

  const [point4State, setPoint4State] = useState("default");

  const [point5State, setPoint5State] = useState("default");

  const [point6State, setPoint6State] = useState("default");

  const [point7State, setPoint7State] = useState("default");

  const [point8State, setPoint8State] = useState("default");

  const [point9State, setPoint9State] = useState("default");

  const [point10State, setPoint10State] = useState("default");

  const [point11State, setPoint11State] = useState("default");

  const [point12State, setPoint12State] = useState("default");

  useEffect(() => {

  data.forEach(d => {

  switch (d.name) {

  case "capteur\_1":

  setPoint1State("highlighted");

  break;

  case "capteur\_2":

  setPoint2State("highlighted");

  break;

  case "capteur\_3":

  setPoint3State("highlighted");

  break;

  case "capteur\_4":

  setPoint4State("highlighted");

  break;

  case "capteur\_5":

  setPoint5State("highlighted");

  break;

  case "capteur\_6":

  setPoint6State("highlighted");

  break;

  case "capteur\_7":

  setPoint7State("highlighted");

  break;

  case "capteur\_8":

  setPoint8State("highlighted");

  break;

  case "capteur\_9":

  setPoint9State("highlighted");

  break;

  case "capteur\_10":

    setPoint10State("highlighted");

    break;

    case "capteur\_11":

  setPoint11State("highlighted");

  break;

  case "capteur\_12":

    setPoint12State("highlighted");

  break;

  default:

  break;

  }

  setTimeout(() => {

    setPoint1State("default");

    setPoint2State("default");

    setPoint3State("default");

    setPoint4State("default");

    setPoint5State("default");

    setPoint6State("default");

    setPoint7State("default");

    setPoint8State("default");

    setPoint9State("default");

    setPoint10State("default");

    setPoint11State("default");

    setPoint12State("default");

    // ... and so on for other points

  }, 1000);

  });

  }, [data]);

    return(

    <div  className="map" style={{backgroundSize:"contain",backgroundRepeat:"no-repeat",backgroundPosition:"center",objectFit:"cover"}}>

      <img className="imageback" src={Ai} alt=""></img>

      <div className={`capteur-1 ${point1State}`}></div>

      <div className={`capteur-2 ${point2State}`}></div>

      <div className={`capteur-3 ${point3State}`}></div>

      <div className={`capteur-4 ${point4State}`}></div>

      <div className={`capteur-5 ${point5State}`}></div>

      <div className={`capteur-6 ${point6State}`}></div>

      <div className={`capteur-7 ${point7State}`}></div>

      <div className={`capteur-8 ${point8State}`}></div>

      <div className={`capteur-9 ${point9State}`}></div>

      <div className={`capteur-10 ${point10State}`}></div>

      <div className={`capteur-11 ${point11State}`}></div>

      <div className={`capteur-12 ${point12State}`}></div>

      {data.map((d, i) => (

              <Icon  key={i} style={{ left: d.x, top: d.y }} icon="material-symbols:directions-run" className="location-icon"  />

      ))}

      </div>

    )

}

export default Map;