

SOMMAIRE

INTRODUCTION.....	3
CHAPITRE 1: CAHIER DE CHARGE.....	4
I-) PRESENTATION GENERALE DU PROJET.....	4
1. Contexte et définition du projet.....	4
2. Périmètre.....	4
3. Cibles.....	4
4. objectifs.....	5
II-) DESCRIPTIONS DES SPÉCIFICATIONS FONCTIONNELLES.....	6
1. Détection de mouvement.....	6
a) Principe de Fonctionnement:.....	6
b) Réglage du HC-SR501.....	8
c) Réglage de la sensibilité (plage).....	9
d) Réglage de la temporisation (Tx).....	9
e) Cavalier de sélection de gâchette.....	9
2. Communication.....	10
3. Surveillance.....	10
4. Interface.....	10
III-) DESCRIPTIONS DES SPÉCIFICATIONS NON FONCTIONNELLES.....	11
1. Facilité d'utilisation.....	11
2. Facilité d'apprentissage.....	11
3. Sécurité.....	11
4. Maintenance du système.....	12
5. Portabilité.....	12
IV-) SCHEMA SYNOPTIQUE.....	13
V-) DÉMARCHE.....	14
1. Analyse.....	14
2. La conception du projet.....	14
3. Evaluation du projet.....	15
Diagramme de PERT.....	15
VI-) LIVRABLES.....	16
CHAPITRE 2: CAHIER D ANALYSE ET CONCEPTION.....	17
I. PRÉSENTATION DES DIAGRAMMES INTERVENANTS.....	17
A. DIAGRAMMES D'ANALYSE.....	17
1. Diagramme de contexte.....	17
2. Diagramme de package.....	18
3. Diagramme de cas d'utilisation et description textuelle.....	19
❖ Activer le système.....	20
❖ Désactiver le système.....	21
❖ Stopper l'alarme.....	22
❖ Déclencher alarme.....	23
4. Diagramme de classe.....	24
5. Diagramme de séquence système.....	25

❖ Activer le système.....	25
❖ Désactiver le système.....	26
6. Diagramme d'activité:.....	27
❖ Activer le système.....	27
❖ Désactiver le système.....	28
❖ Déclencher alarme.....	29
❖ Stopper l'alarme.....	30
B. DIAGRAMME DE CONCEPTION.....	32
1. Diagrammes de séquence technique.....	32
❖ Stopper l'alarme.....	32
❖ Déclencher l'alarme.....	33
❖ Activer le système.....	34
❖ Désactiver le système.....	35
2. Diagramme de classe technique.....	36
3. Diagramme de composants.....	37
4. Diagramme de déploiement.....	38
CHAPITRE 3: CAHIER DE RÉALISATION.....	39
I. DESCRIPTIONS DES SPÉCIFICATIONS TECHNIQUES.....	39
a) Les choix technologiques.....	39
b) L'environnement/l'architecture/Contraintes logicielles.....	39
i. L'architecture de conception.....	39
ii. Système d'exploitation.....	39
iii. Serveur Web et Serveur d'application.....	39
c) Les exigences de programmation(langage informatique).....	40
d) L'accessibilité (compatibilité navigateurs, logiciels, appareils).....	40
e) La sécurité.....	40
II. LISTE DU MATÉRIEL.....	41
III. PROGRAMMATION DES CONSTITUANTS DU SYSTÈME.....	47
IV. SIMULATION DU CIRCUIT AVEC PROTEUS.....	49
1. PIR sensor - Arduino Uno.....	49
2. WEMOS D1 R2 Mini + Afficheur LCD + Module RTC (Real Time Clock).....	50
Montage du début avant la simulation.....	50
Montage après la simulation.....	51
IV. COMMUNICATION ENTRE TELEGRAM ET LE MODÈLE ESP32-CAM.....	52
IV. GUIDE D'INSTALLATION ET GUIDE D'UTILISATION.....	55
1. Positionner module de détection dans la zone que vous voulez sécuriser.....	55
V. PERSPECTIVE D'AMÉLIORATION.....	67
CONCLUSION.....	68
BIBLIOGRAPHIE.....	69
ANNEXES.....	70

INTRODUCTION

Le domicile est un lieu de refuge où chacun devrait se sentir en sécurité. Cependant, avec l'augmentation des cambriolages et des intrusions, il est devenu impératif de mettre en place des systèmes de sécurité efficaces pour protéger nos foyers. C'est dans ce contexte que s'inscrit le présent projet, visant à concevoir un système de sécurité complet et innovant pour les habitations. L'objectif principal de ce projet est de créer un environnement sécurisé pour les occupants d'une maison en mettant en place des dispositifs technologiques avancés. En combinant des éléments de surveillance, de détection d'intrusion, d'alerte et de contrôle à distance, ce système de sécurité vise à offrir une protection optimale et une tranquillité d'esprit aux résidents. La conception d'un tel système requiert une approche multidisciplinaire, alliant des connaissances en ingénierie électronique, en informatique et en sécurité. En effet, la complexité croissante des menaces et des techniques d'intrusion exige des solutions sophistiquées et évolutives pour garantir une protection efficace. Dans ce rapport, nous présenterons en détail les différentes phases du projet, de la planification à la mise en œuvre, en passant par le développement et les tests. Nous aborderons également les choix technologiques effectués, les défis rencontrés et les solutions apportées pour garantir la fiabilité et l'efficacité du système de sécurité proposé. En outre, nous analyserons les avantages et les limites du système conçu, en mettant en lumière ses fonctionnalités clés, sa facilité d'utilisation et sa capacité à s'adapter aux besoins spécifiques des utilisateurs. Enfin, nous discuterons des perspectives d'amélioration et des pistes pour l'évolution future de ce système de sécurité pour domicile, dans un souci constant d'innovation et de protection accrue.

CHAPITRE 1: CAHIER DE CHARGE

I-) PRESENTATION GENERALE DU PROJET

1. Contexte et définition du projet

Dans un contexte de plus en plus numérique et interconnecté, la sécurité informatique est devenue une préoccupation majeure pour les organisations. Les attaques, telles que les intrusions malveillantes peuvent entraîner de graves conséquences telles que la disparition des objets de valeur, destruction des biens matériels, ...etc. La détection d'intrusion est donc devenue essentielle pour identifier et prévenir ces menaces.

Le projet de détection d'intrusion vise à mettre en place des mécanismes et des outils de surveillance pour détecter les activités suspectes et les tentatives d'accès non autorisées aux infrastructures (maisons, entreprises, ...etc). Il implique l'analyse en temps réel des comportements des utilisateurs afin de repérer les schémas anormaux et les indicateurs d'intrusion potentiels.

2. Périmètre

Le périmètre représente les limites que nous fixons pour notre projet. Dans notre cas :

- Le système aura une portée de 3 mètres tout autour de la maison
- L'application aura une vue administrateur qui permettra au propriétaire d'être notifié de la présence d'un intrus.

3. Cibles

Ce système vise principalement les établissements et les entreprises même s'il peut être utilisé par quiconque a besoin de protéger l'accès de son domicile contre des intrus.

4. objectifs

L'objectif principal de notre système est d'assurer la protection des biens et des personnes résidant dans la maison contre les intrusions, les cambriolages, les braquages, ... etc. Cela inclut la surveillance de l'environnement, la détection précoce des menaces et la dissuasion des intrus potentiels pour garantir la sécurité et la tranquillité d'esprit des occupants.

Les objectifs secondaires d'un tel système peuvent inclure :

- **Surveillance continue** : Assurer une surveillance constante de l'extérieur de la maison à l'aide de caméras, de capteurs de mouvement et d'autres dispositifs pour détecter toute activité suspecte.
- **Alertes rapides et efficaces** : Envoyer des alertes immédiates aux occupants et aux autorités compétentes en cas d'intrusion ou d'incident, permettant une réaction rapide et appropriée.
- **Contrôle à distance** : Permettre aux propriétaires de contrôler et de surveiller leur système de sécurité à distance via une application mobile, offrant une gestion pratique et flexible.
- **Simplicité d'utilisation** : Concevoir une interface conviviale et intuitive pour permettre aux utilisateurs de gérer facilement leur système de sécurité, sans nécessiter de compétences techniques avancées.

En combinant ces objectifs principaux et secondaires, un système de sécurité pour domicile peut offrir une protection complète et efficace, adaptée aux besoins spécifiques de chaque foyer.

II-) DESCRIPTIONS DES SPÉCIFICATIONS FONCTIONNELLES

1. Détection de mouvement

Le capteur PIR Sensor ou capteur de mouvement présent dans le système est idéal pour la détection de mouvements.

PIR est l'abréviation de « **Passive Infrared** ». Fondamentalement, le capteur de mouvement PIR mesure la lumière infrarouge des objets dans son champ de vision. Ainsi, il peut détecter un mouvement en fonction des changements de rayonnement infrarouge dans l'environnement. Il est idéal pour détecter si un humain s'est déplacé dans ou hors de la portée du capteur.

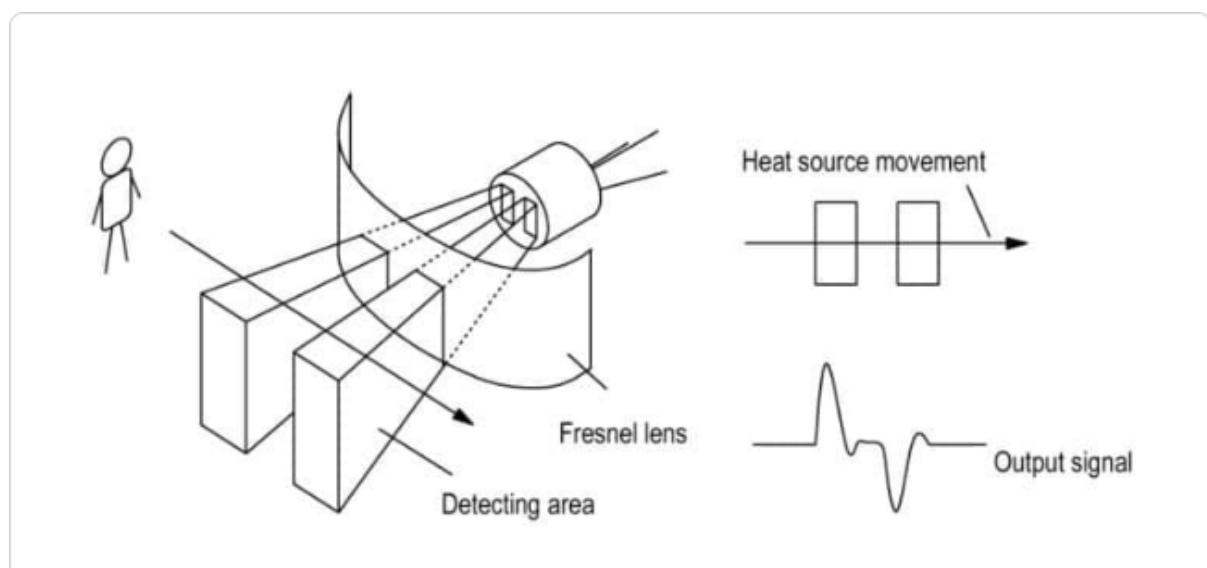
La proximité du capteur PIR est réglable entre 3m et 7m. Il existe des capteurs similaires tels que des capteurs à ultrasons, des capteurs gestuels qui peuvent également être utilisés pour la détection de mouvements.

a) Principe de Fonctionnement:

Les capteurs de mouvement PIR se composent de deux parties principales :

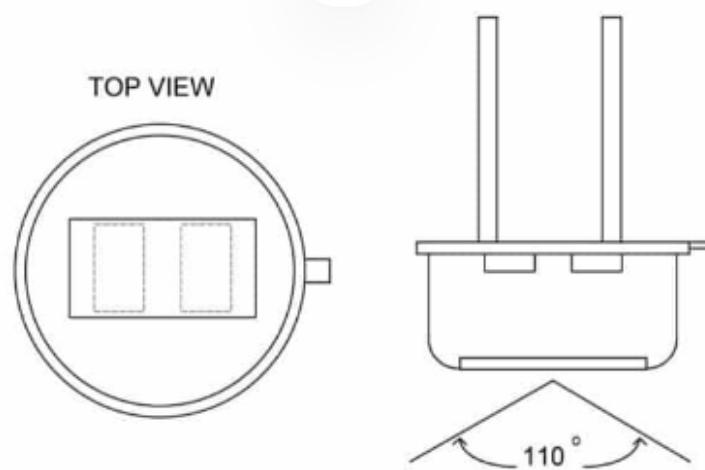
- **un élément de détection pyroélectrique**
- **une lentille de Fresnel**

L'élément de détection pyroélectrique peut détecter le rayonnement infrarouge. Tous les objets dont la température est supérieure au zéro absolu (0 Kelvin / -273,15 °C) émettent de l'énergie thermique sous forme de rayonnement infrarouge, y compris le corps humain.

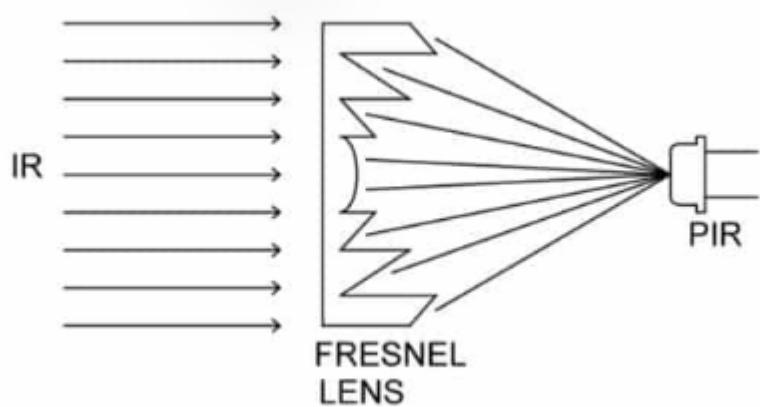


Ce capteur pyroélectrique à deux fentes rectangulaires fait d'un matériau qui laisse passer le rayonnement infrarouge. Derrière celles-ci, il y a deux électrodes de capteur infrarouge séparées, l'une chargée de produire une sortie positive et l'autre une sortie négative. La raison en est qu'on s'intéresse aux changements dans les niveaux d'IR et non dans les niveaux d'IR ambiant. Les deux électrodes sont câblées de manière à ce qu'elles s'annulent. Si une moitié voit plus ou moins de rayonnement IR que l'autre, la sortie oscillera vers le haut ou vers le bas.

Le circuit intégré de traitement du signal traite ce signal et tourne la broche de sortie du capteur vers le haut ou vers le bas en conséquence.



Le dôme blanc devant l'élément de détection est une lentille de Fresnel. Cet objectif focalise le rayonnement infrarouge sur le capteur.

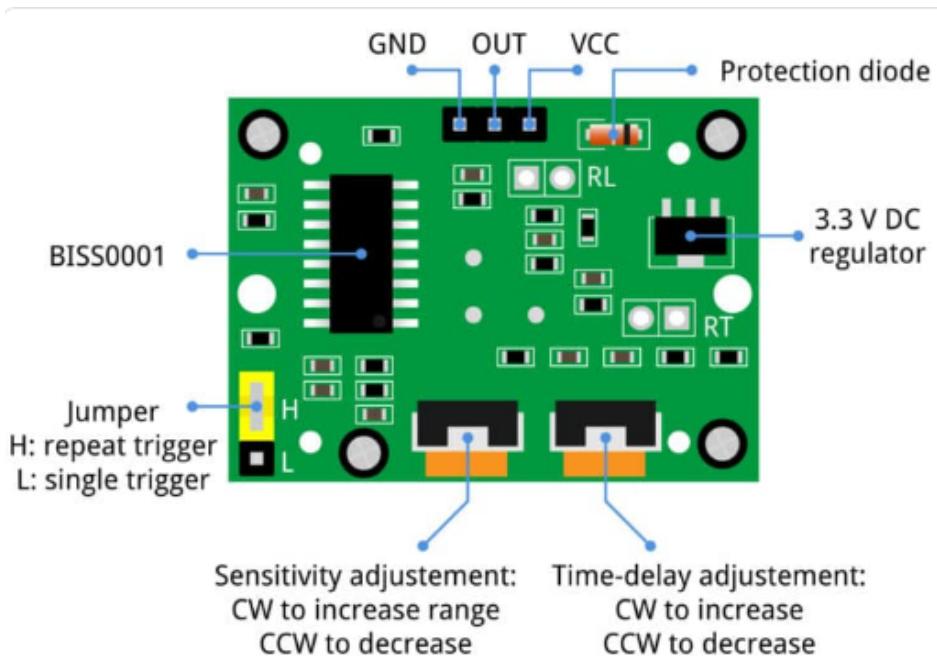


Les spécifications du HC-SR501 sont données dans le tableau ci-dessous:

Tension de fonctionnement	4,5 à 20 V
Courant de repos	50 µA
Sortie de niveau	ÉLEVÉ 3,3 V / FAIBLE 0 V
Gâchette	Gâchette simple L / Gâchette répétitive H
Temps de retard	3 à 300 s
Temps de blocage	2,5 s (par défaut)
Plage de mesure	3 – 7 m maximum
Angle de mesure	< Angle de cône de 110°
Dimensions de la lentille de Fresnel	Diamètre 15 mm x 23 mm
Température de fonctionnement	-15 à 70 °C
Dimensions du PCB	32,5 x 24 millimètre

b) Réglage du HC-SR501

À l'arrière de la carte, on trouve deux potentiomètres et un cavalier, qui ont été utilisé pour régler plusieurs paramètres :



c) Réglage de la sensibilité (plage)

Le HC-SR501 a une distance de détection maximale (portée de détection) de 7 mètres. On peut régler la distance de détection en tournant le potentiomètre de sensibilité CW ou CCW (voir image ci-dessus).

- La rotation du potentiomètre dans le sens des aiguilles d'une montre augmente la distance de détection jusqu'à un maximum de 7 mètres.
- En le tournant dans le sens inverse des aiguilles d'une montre, la distance de détection est réduite à un minimum de 3 mètres.

d) Réglage de la temporisation (Tx)

Ce potentiomètre est utilisé pour régler le temps pendant lequel la sortie reste ÉLEVÉE après la détection d'un mouvement. Au minimum, le délai est de 3 secondes et au maximum, il est de 300 secondes ou 5 minutes.

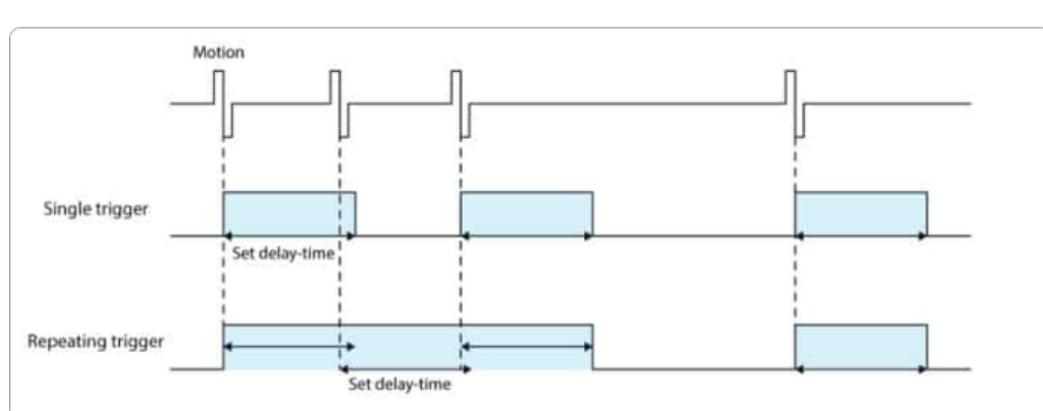
- Tourner le potentiomètre dans le sens des aiguilles d'une montre permet d'augmenter le délai
- Le tourner dans le sens inverse des aiguilles d'une montre permet de le diminuer.

e) Cavalier de sélection de gâchette

Le cavalier (jaune) est utilisé pour sélectionner l'un des deux modes de déclenchement. Il peut être réglé sur L (déclencheur unique) ou H (déclencheur répétitif) :

- **Gâchette unique:** La sortie deviendra HAUTE dès qu'un mouvement est détecté. Il restera élevé pendant le temps réglé par le potentiomètre. Tout mouvement pendant cette période n'est pas traité et ne redémarre pas le minuteur.
- **Déclencheur répétitif :** Chaque fois qu'un mouvement est détecté, la minuterie de retard est redémarrée.

La différence entre le mode de déclenchement unique et le mode de déclenchement répétitif est illustrée dans la figure ci-dessous:



2. Communication

Notre système est capable d'avertir l'utilisateur en cas de signal d'alerte. De plus, il permet de communiquer avec les utilisateurs quelle que soit leur distance de localisation par rapport au domicile (longue ou courte) et d'informer ces derniers de toute intrusion ou alerte dans un certain périmètre autour de son domicile (réglable entre 3m et 7m). Ces communications se font principalement :

- via l'application mobile dédiée: Secure Alert
- via un bot dans l'application populaire Telegram

Ainsi, pour les distances de détection définies, le buzzer est utilisé et permet d'indiquer qu'il y a une alerte via un signal sonore personnalisable dans notre système. L'interface utilisateur quant à elle a toute son importance pour des communications longue distance. L'utilisateur reçoit une notification d'alerte via l'interface utilisateur sur son mobile.

3. Surveillance

La caméra OV2640 de l'ESP32-CAM est utilisée pour surveiller la maison (enfants, animaux domestiques et étrangers). Il est responsable du streaming vidéo ainsi que de la reconnaissance et de la détection des visages. Si le capteur PIR Sensor détecte un mouvement, la caméra OV2640 de l'ESP32-CAM sera déclenchée à partir d'un mode veille et fera une capture quasi instantanée après la détection d'un mouvement. De plus, notre système prend également en charge une option de streaming vidéo dépendant de la volonté de l'utilisateur, ceci via un bouton dans l'application mobile, ce qui permet à ce dernier d'inspecter en temps réel son domicile.

4. Interface

L'interface utilisateur est un intermédiaire entre l'utilisateur et le système. Il permet aux utilisateurs d'accéder aux données de n'importe où et à tout moment. Il liste également les intrusions détectées dans le voisinage de son domicile, avec des images prises par notre système après une détection d'intrusion. Les utilisateurs peuvent se connecter à l'application pour accéder aux données.

III-) DESCRIPTIONS DES SPÉCIFICATIONS NON FONCTIONNELLES

1. Facilité d'utilisation

- L'application doit être conçue de manière conviviale et intuitive, avec une interface utilisateur simple et facile à comprendre.
- Les fonctionnalités principales doivent être accessibles en quelques clics ou taps, sans nécessiter de connaissances techniques avancées.
- Des indications et des instructions claires doivent être fournies pour guider les utilisateurs tout au long du processus d'utilisation.

2. Facilité d'apprentissage

- L'application doit être conçue selon le principe d'affordance, ce qui signifie que les actions à effectuer seront intuitives et évidentes pour les utilisateurs, sans nécessiter d'apprentissage préalable.
- Les icônes et les libellés doivent être choisis de manière à être facilement compréhensibles, même pour les utilisateurs novices.
- Des tutoriels interactifs ou des guides d'utilisation doivent être disponibles pour aider les utilisateurs à se familiariser rapidement avec le système.

3. Sécurité

- Le système sera conçu pour assurer la confidentialité et l'intégrité des données.
- Les communications entre l'application et les dispositifs du système de sécurité doivent être cryptées pour éviter toute interception ou manipulation des données.
- Des mesures de sécurité robustes doivent être mises en place pour empêcher les accès non autorisés au système, par exemple, l'utilisation de mots de passe forts et de mécanismes d'authentification à deux facteurs.
- Les données sensibles, telles que les captures photos, seront stockées de manière sécurisée et protégées contre toute tentative de vol ou de compromission.

4. Maintenance du système

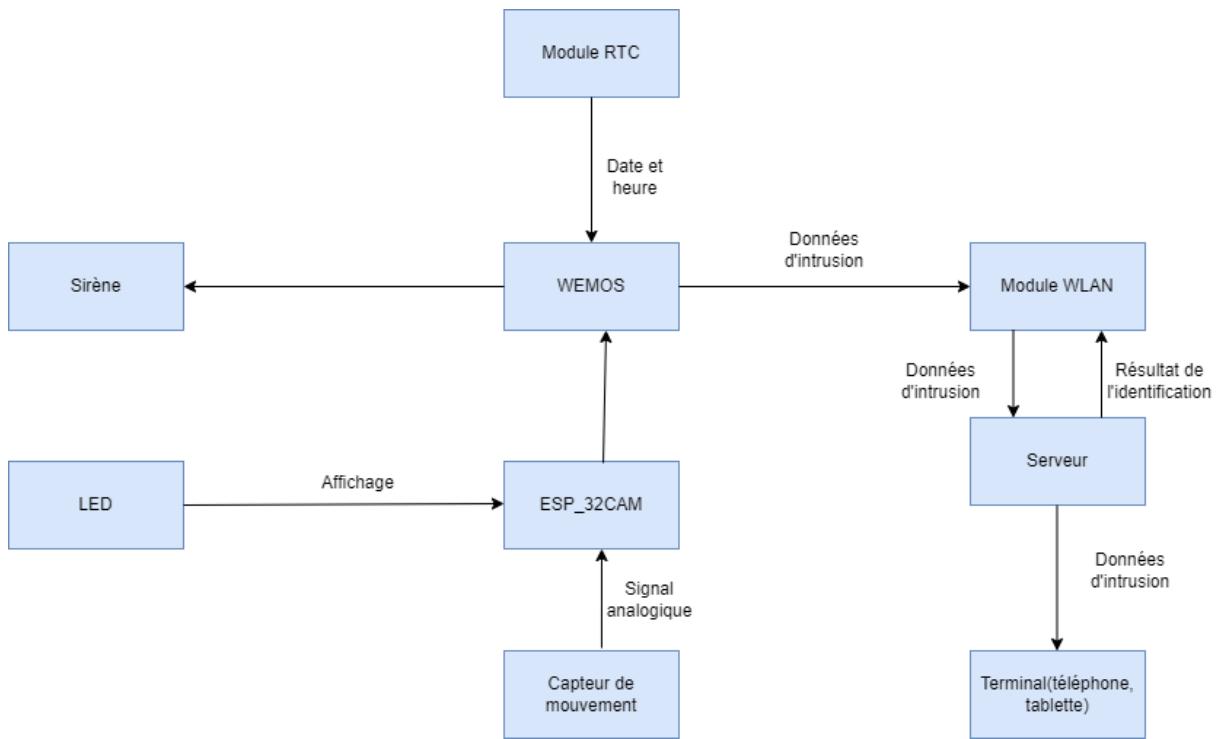
- Le développement de l'application doit être réalisé en suivant les meilleures pratiques de codage et de documentation.
- Le code source doit être commenté de manière détaillée pour faciliter la compréhension et les modifications ultérieures.
- Des procédures de maintenance doivent être documentées, ce qui permettra à un développeur externe de prendre en charge la maintenance sans difficultés majeures.
- Des mises à jour régulières du système seront fournies pour corriger les éventuels problèmes, améliorer les performances et renforcer la sécurité.

5. Portabilité

- L'application doit être compatible avec différents types de terminaux, tels que les téléphones mobiles et les tablettes.
- Elle doit être développée en utilisant des technologies multiplateformes, permettant ainsi un déploiement sur divers systèmes d'exploitation, tels que Windows, iOS et Android.
- L'interface utilisateur sera adaptable et réactive, s'adaptant automatiquement à la taille et à la résolution de l'écran du terminal utilisé.
- Les utilisateurs devront accéder à l'application depuis n'importe quel terminal pris en charge, en utilisant leurs identifiants de connexion sécurisés.

IV-) SCHEMA SYNOPTIQUE

Voici une présentation graphique simplifiée, qui permet de saisir d'un simple coup d'œil un ensemble d'informations liées à notre système.



V-) DÉMARCHE

1. Analyse

- **Diagrammes statiques d'analyse** : Heutchou, Foueguim
- **Diagrammes dynamiques d'analyse** : Mbangono, Tchakonte
 - **Suivie** : Tene, Foueguim
 - **Livrable** : Cahier d'analyse

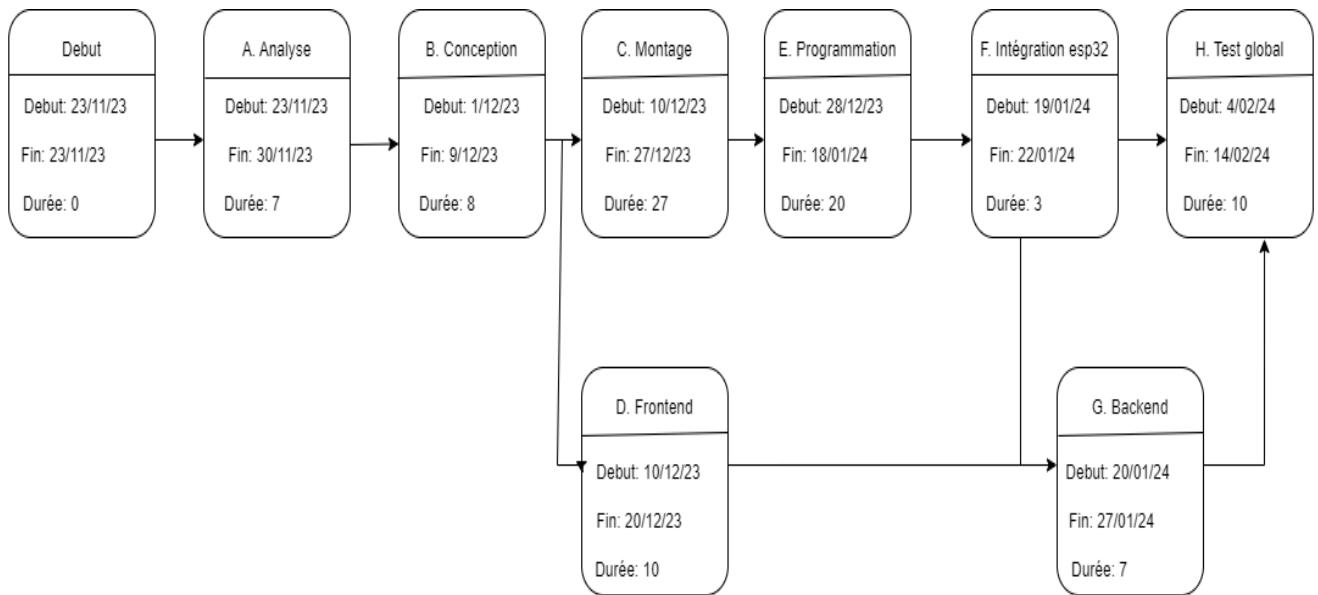
2. La conception du projet

- **Diagrammes statiques de conception** : Keugue, Chemi
- **Diagrammes dynamiques de conception** : Chedjou, Tene, Foueguim
- **Maquette** : Mbangono, Foueguim
- **Suivie** : Tene, Dada
- **Livrable** : cahier de conception

3. Evaluation du projet

Diagramme de PERT

Le diagramme de Pert suivant est illustratif de la planification des différentes activités qui ont permis à mettre ce pied ce produit final.



VI-) LIVRABLES

1. Un système de sécurité fonctionnel intégrant toutes les fonctionnalités spécifiées, comprenant les capteurs de mouvement, l'alarme, les canaux de communication et l'interface utilisateur.
2. Une documentation détaillée sur l'installation, la configuration et l'utilisation du système.
3. Un support technique si possible pour résoudre les problèmes éventuels.

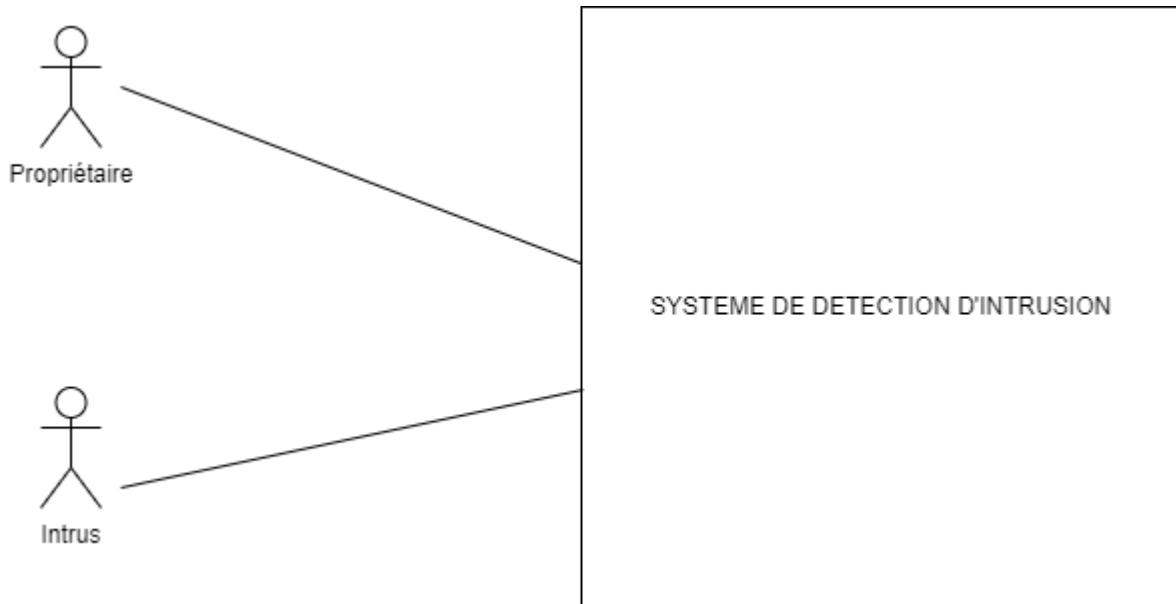
CHAPITRE 2: CAHIER D ANALYSE ET CONCEPTION

Dans cette partie, nous allons réaliser les diagrammes d'analyse et de conception de notre projet à l'aide du langage UML.

I. PRÉSENTATION DES DIAGRAMMES INTERVENANTS

A. DIAGRAMMES D'ANALYSE

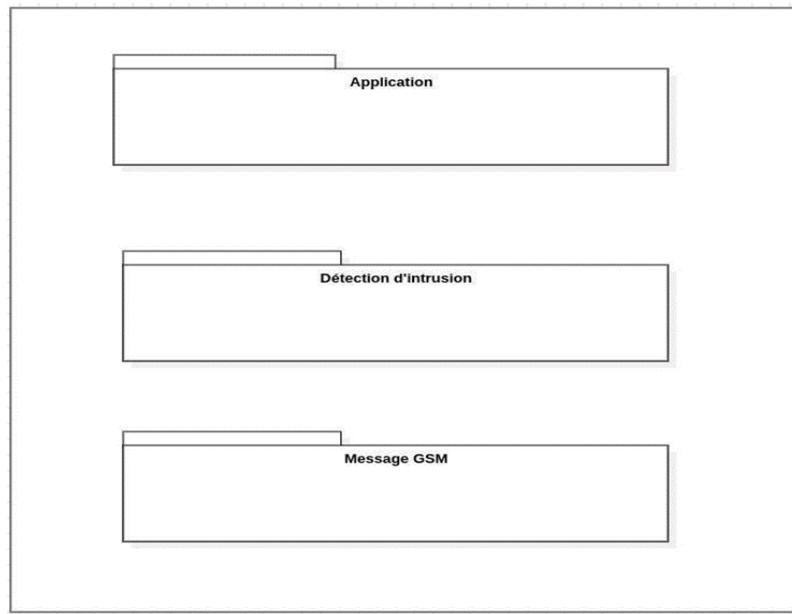
1. Diagramme de contexte



Le diagramme de contexte permet d'avoir une vue d'ensemble des interactions entre le système « Système de sécurité à domicile » et les différents acteurs. Comme acteurs interagissant avec notre système, nous avons :

- L'utilisateur qui va contrôler le système à distance.
- L'Intru qui permettra l'activation de certaines fonctionnalités du système

2. Diagramme de package

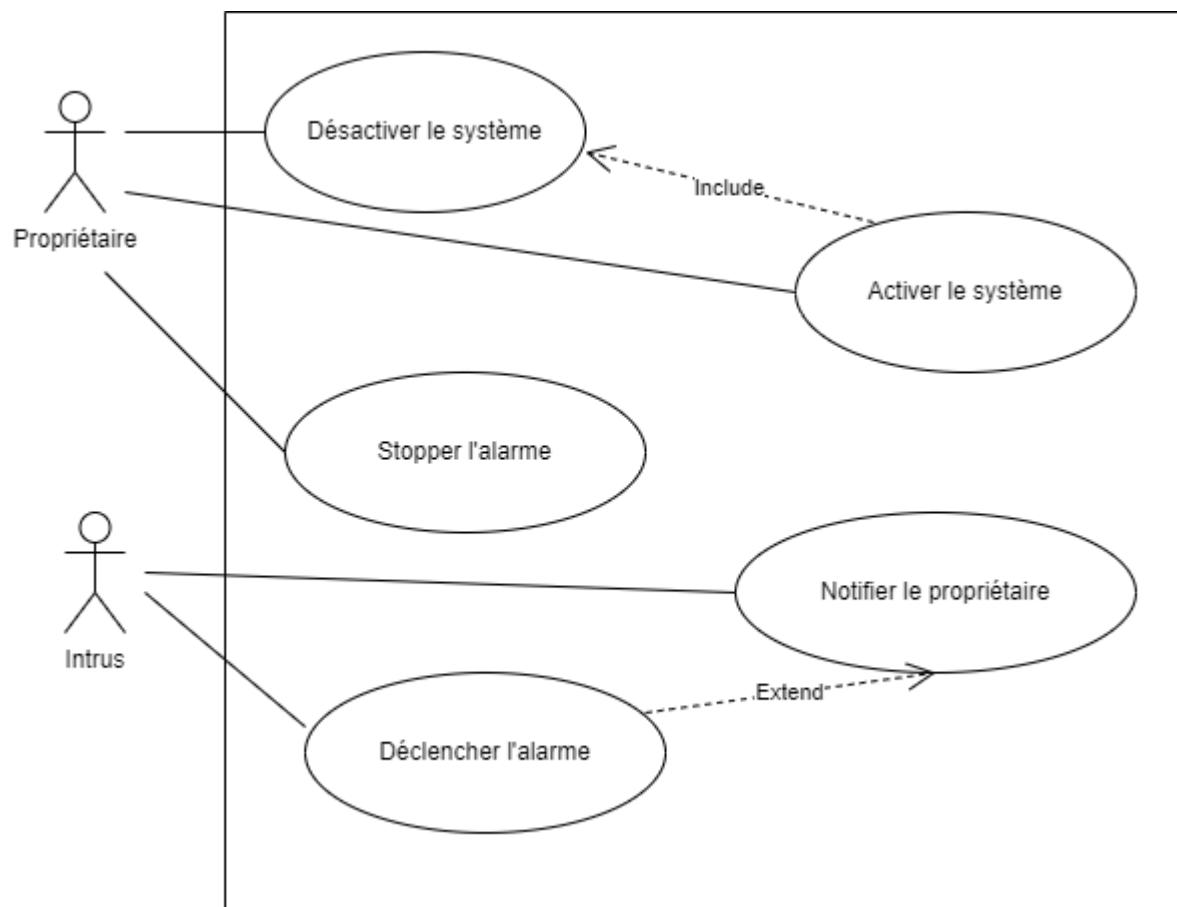


- **Application** : Ce paquet contient la logique principale de l'application. Il comprend les méthodes pour l'exécution du programme, pour détecter les intrusions, pour envoyer des messages GSM et pour afficher des alertes.
- **Détection d'intrusion** : Ce paquet est responsable de la détection des intrusions. Il comprend les méthodes pour lancer la détection, pour analyser les paquets et pour envoyer des alertes.
- **Message GSM** : Ce paquet est responsable de l'envoi des messages GSM. Il comprend les méthodes pour envoyer les messages, pour créer les messages à envoyer et pour établir la connexion GSM.

Le paquetage d'application a une dépendance d'accès vers celui de détection d'intrusion car il nécessite des fonctionnalités de détection pour afficher des informations à l'utilisateur.

Le paquetage de détection d'intrusion a une dépendance d'importation vers Message GSM car il importe les fonctionnalités d'envoi des messages pour alerter le propriétaire en cas de détection d'intrusion.

3. Diagramme de cas d'utilisation et description textuelle



Du diagramme ci-dessus, découlent les descriptions textuelles des cas d'utilisation suivantes:

❖ Activer le système

Cas d'utilisation	Activer le système
Objectifs	Mettre en marche le système de détection d'intrusion
Résultats	Notification de succès et allumage d'une Led rouge au niveau de chaque capteur
Acteurs	Propriétaire
Pré-conditions	L'utilisateur doit être connecté au système matériel d'alerte via un réseau
Post-conditions	Aucune
Scénario Nominal	1- L'utilisateur appuie sur le menu activation. 2- L'utilisateur appuie sur le bouton activer. 3- L'utilisateur valide l'activation. 4- La led au niveau de chacun des capteurs s'allume. 5- L'utilisateur reçoit une notification de succès.
Scénario alternatif	L'utilisateur appuie sur annuler : aller à 1

❖ Désactiver le système

Cas d'utilisation	Désactiver le système
Objectifs	Éteindre le système de détection d'intrusion
Résultats	Notification de succès et les leds rouge au niveau de chaque capteur s'éteignent
Acteurs	Propriétaire
Pré-conditions	L'utilisateur doit être connecté au système matériel d'alerte via un réseau
Post-conditions	Aucune
Scénario Nominal	1- L'utilisateur appuie sur le menu désactivation 2- L'utilisateur appuie sur le bouton désactiver. 3- L'utilisateur valide la désactivation. 4- La led au niveau de chacun des capteurs s'éteint. 5- L'utilisateur reçoit une notification de succès.
Scénario alternatif	L'utilisateur appuie sur annuler : aller à 1

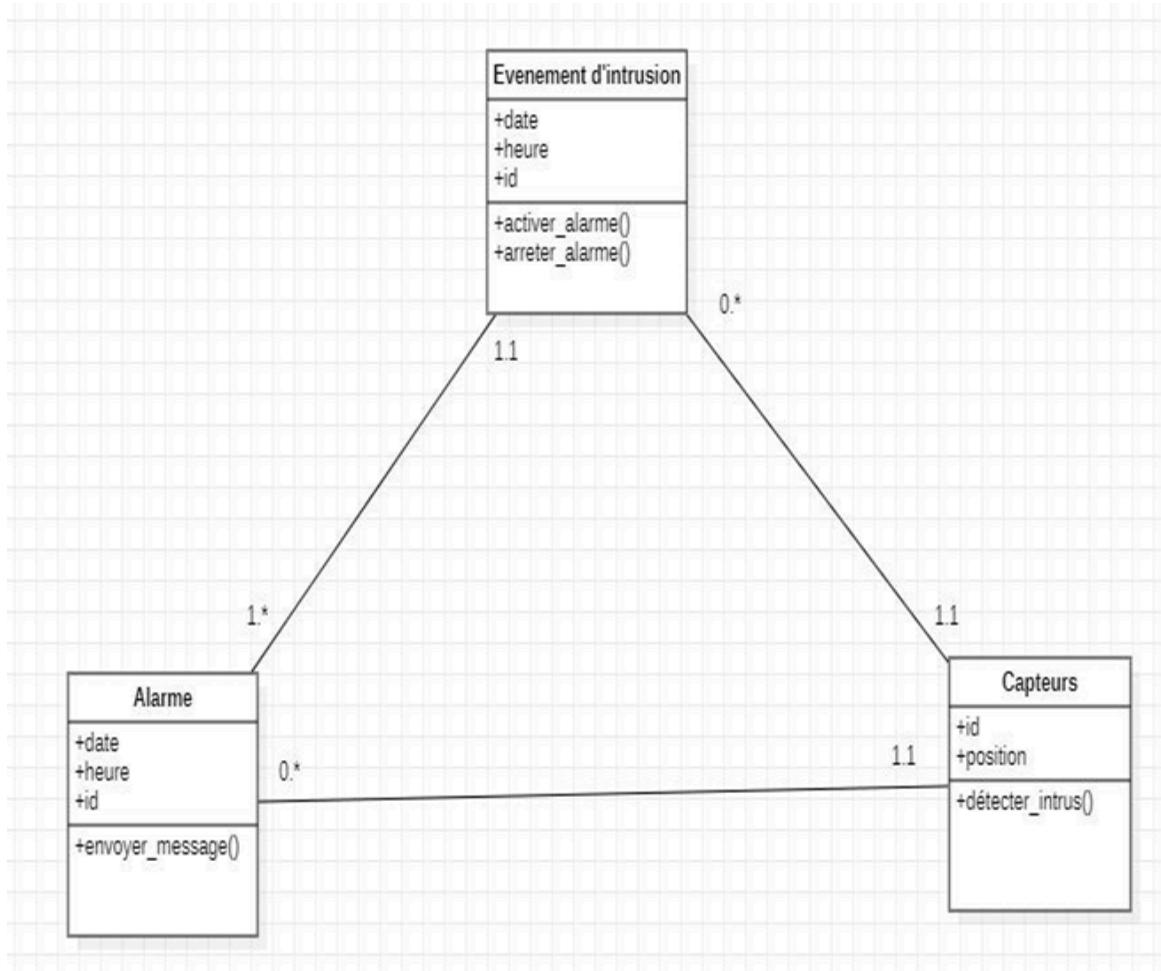
❖ Stopper l'alarme

Cas d'utilisation	Stopper l'alarme
Objectifs	Arrêter la sonnerie de l'alarme dans la maison
Résultats	L'alarme Arrête de sonner
Acteurs	Propriétaire
Pré-conditions	L'utilisateur doit être connecté au système matériel d'alerte via un réseau et l'alarme doit être en train de sonner
Post-conditions	Aucune
Scénario Nominal	<p>1- Une notification arrive pour prévenir l'utilisateur que l'alarme sonne (la page de notification à l'option stopper l'alarme et une autre laisser sonner sous forme de bouton).</p> <p>2- L'utilisateur appuie sur stopper l'alarme.</p> <p>3- L'utilisateur reçoit une notification de succès.</p>
Scénario alternatif	<p>2.a L'utilisateur ne fait rien et l'alarme s'arrête après 15 minutes grâce aux Timer</p> <p>2.b L'utilisateur appuie sur laisser sonner et l'alarme s'arrête après 30 minutes grâce au Timer</p>

❖ Déclencher alarme

Cas d'utilisation	Déclencher l'alarme
Objectifs	Faire sonner l'alarme et envoyer une notification au propriétaire
Résultats	L'alarme commence à sonner et l'utilisateur reçoit une notification
Acteurs	Intrus
Pré-conditions	Le propriétaire doit être connecté au système matériel d'alerte via un réseau
Post-conditions	Aucune
Scénario Nominal	1- L'intrus entre dans la zone de détection des capteur 2- Le système met en marche l'alarme 3- Simultanément le système envoi une notification d'alerte au propriétaire

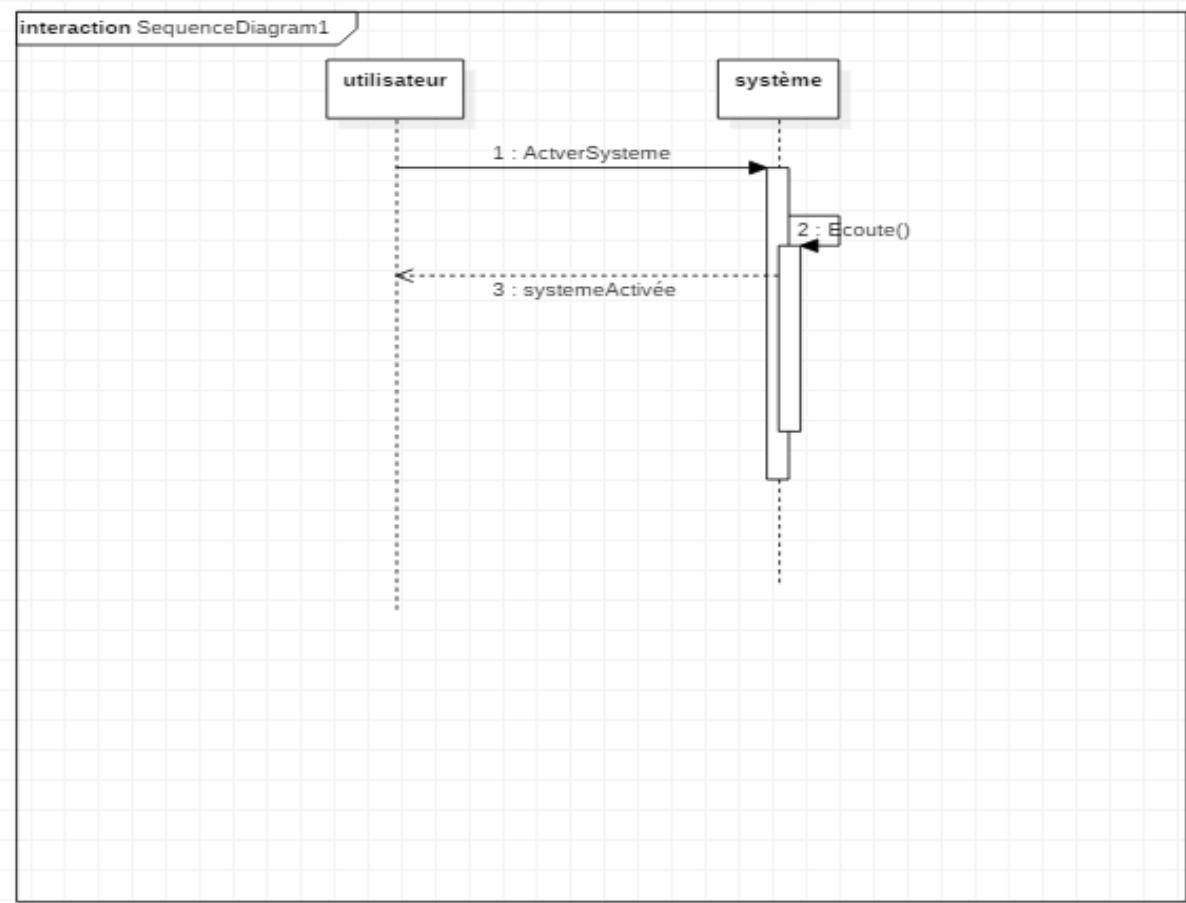
4. Diagramme de classe



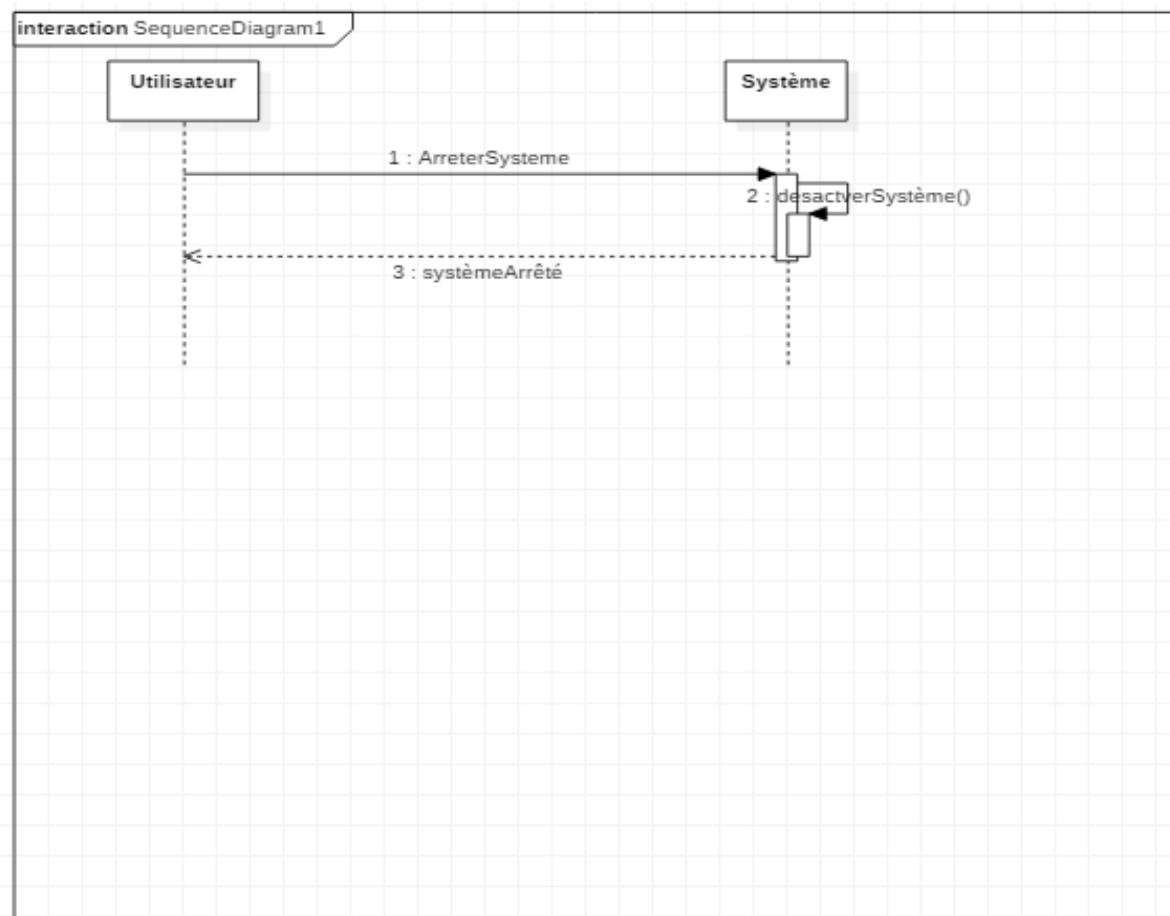
5. Diagramme de séquence système

Dans cette partie, nous allons réaliser les diagrammes de séquence système de chaque cas d'utilisation énoncés plus haut. Nous avons donc :

❖ Activer le système



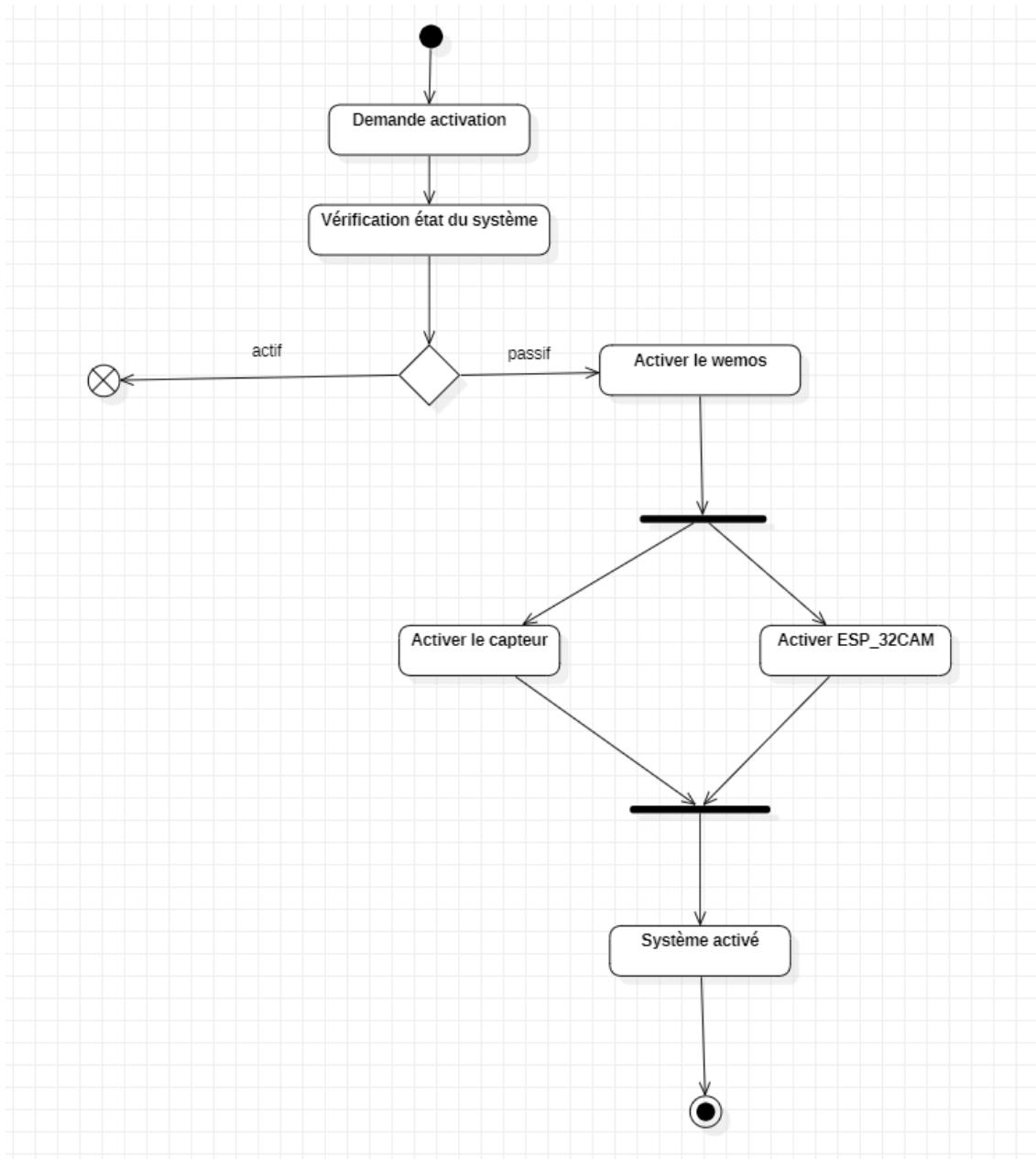
❖ Désactiver le système



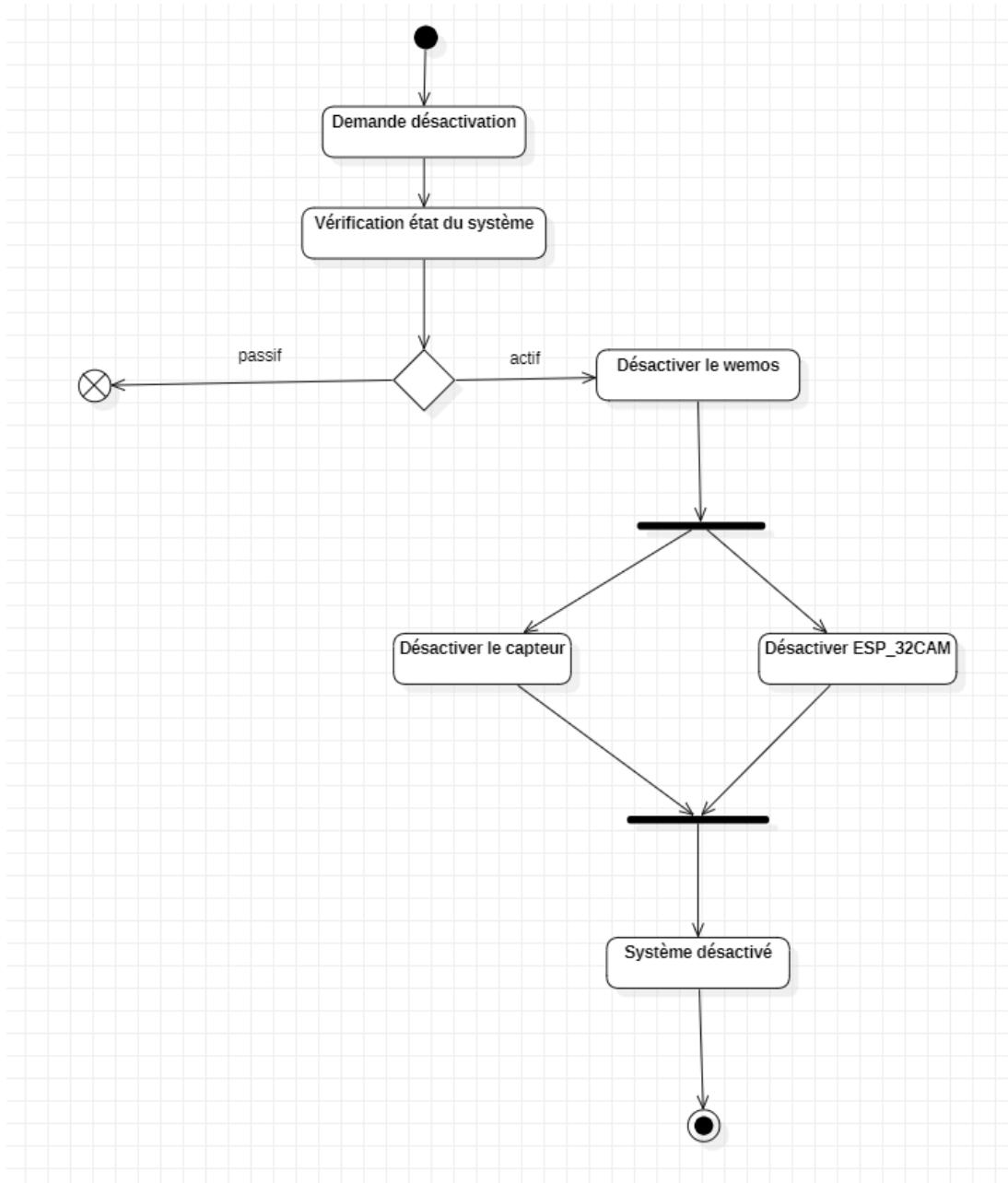
6. Diagramme d'activité:

Dans cette partie, nous allons réaliser les diagrammes d'activité de chacun des cas d'utilisation énoncés plus haut. Nous avons donc :

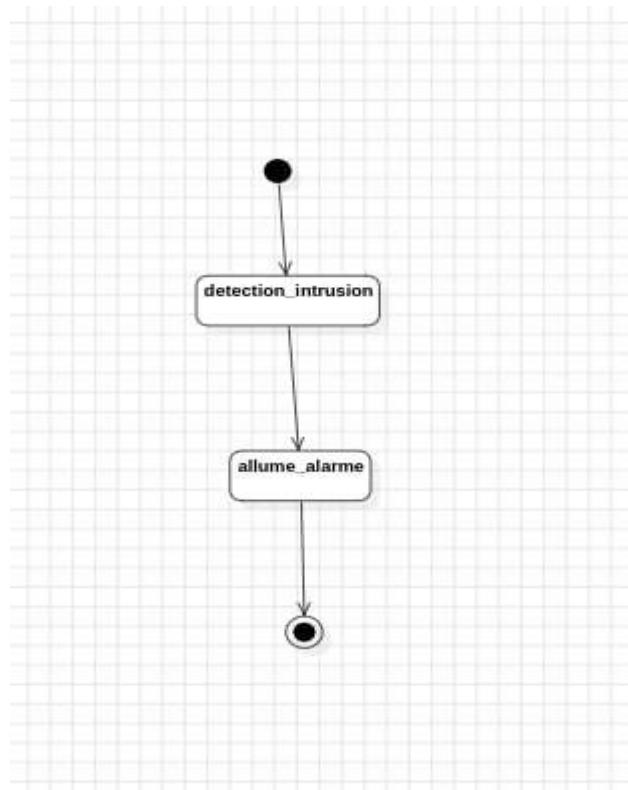
❖ Activer le système



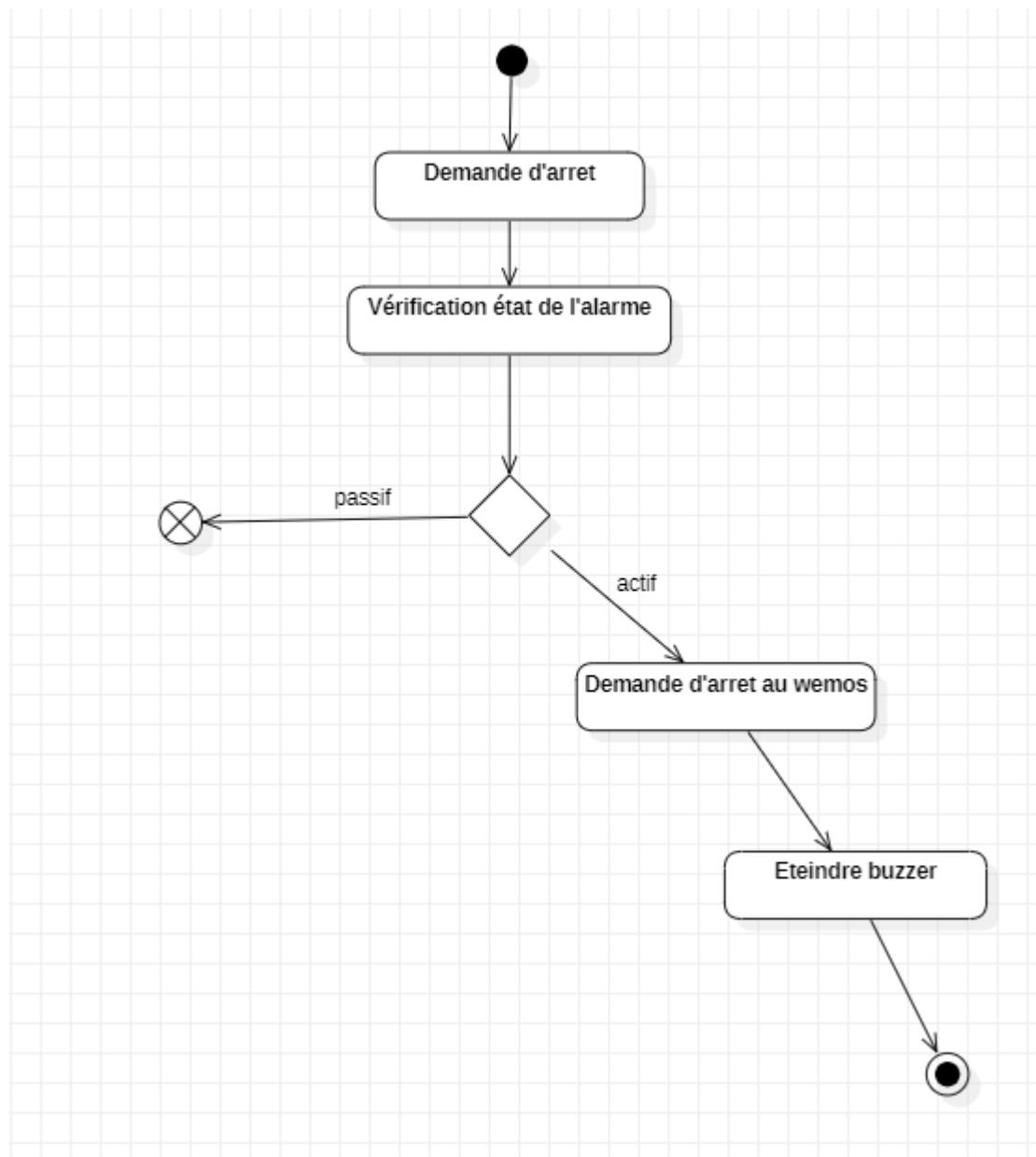
❖ Désactiver le système



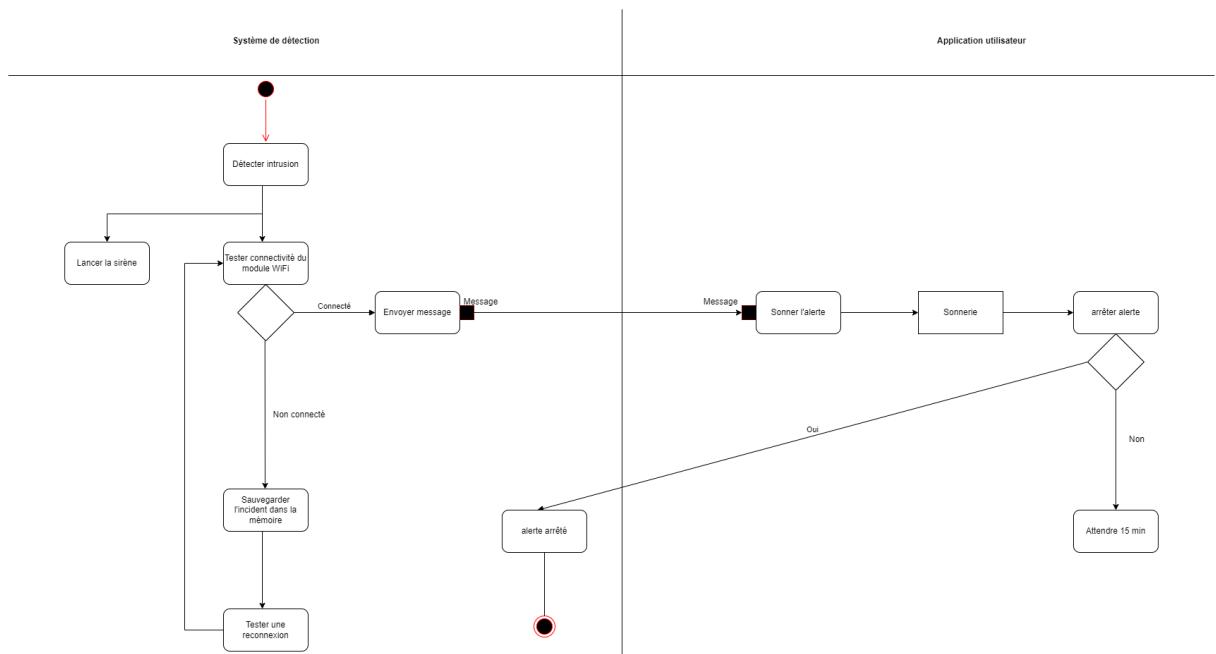
❖ Déclencher alarme



❖ Stopper l'alarme



❖ Diagramme d'activité générale

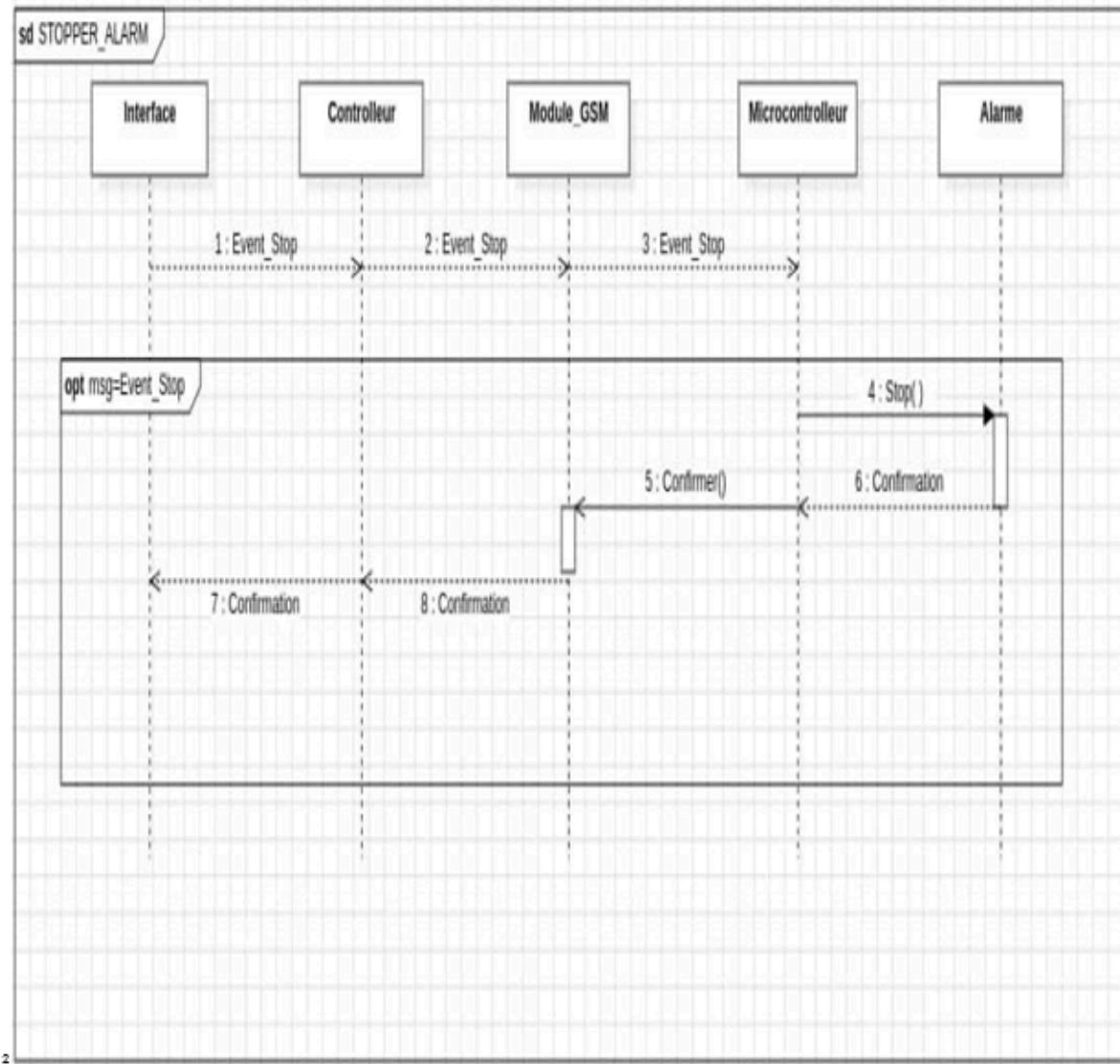


B. DIAGRAMME DE CONCEPTION

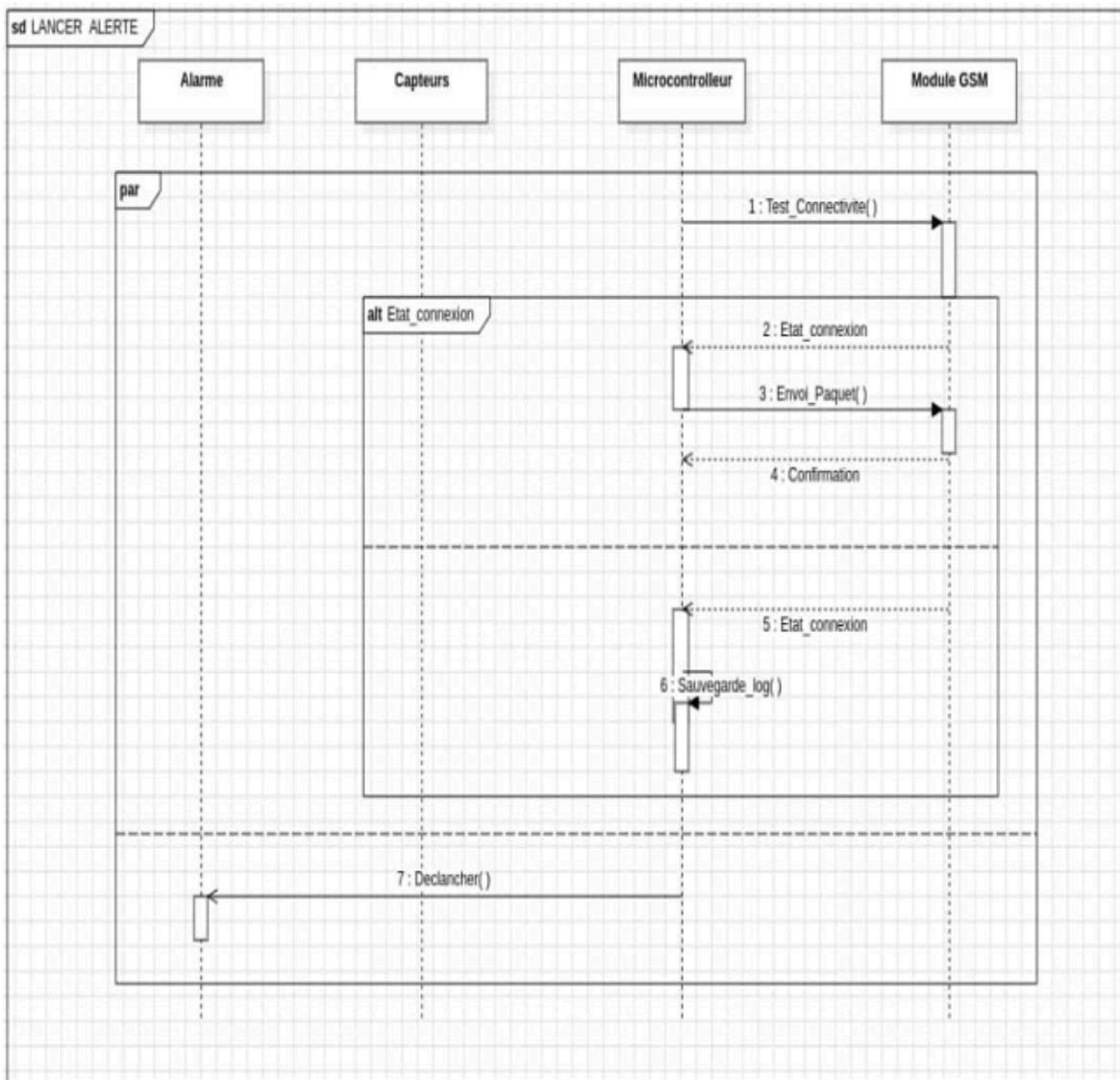
1. Diagrammes de séquence technique

Dans cette partie, nous allons réaliser les diagrammes de séquence technique de chacun des cas d'utilisation énoncés plus haut. Nous avons donc :

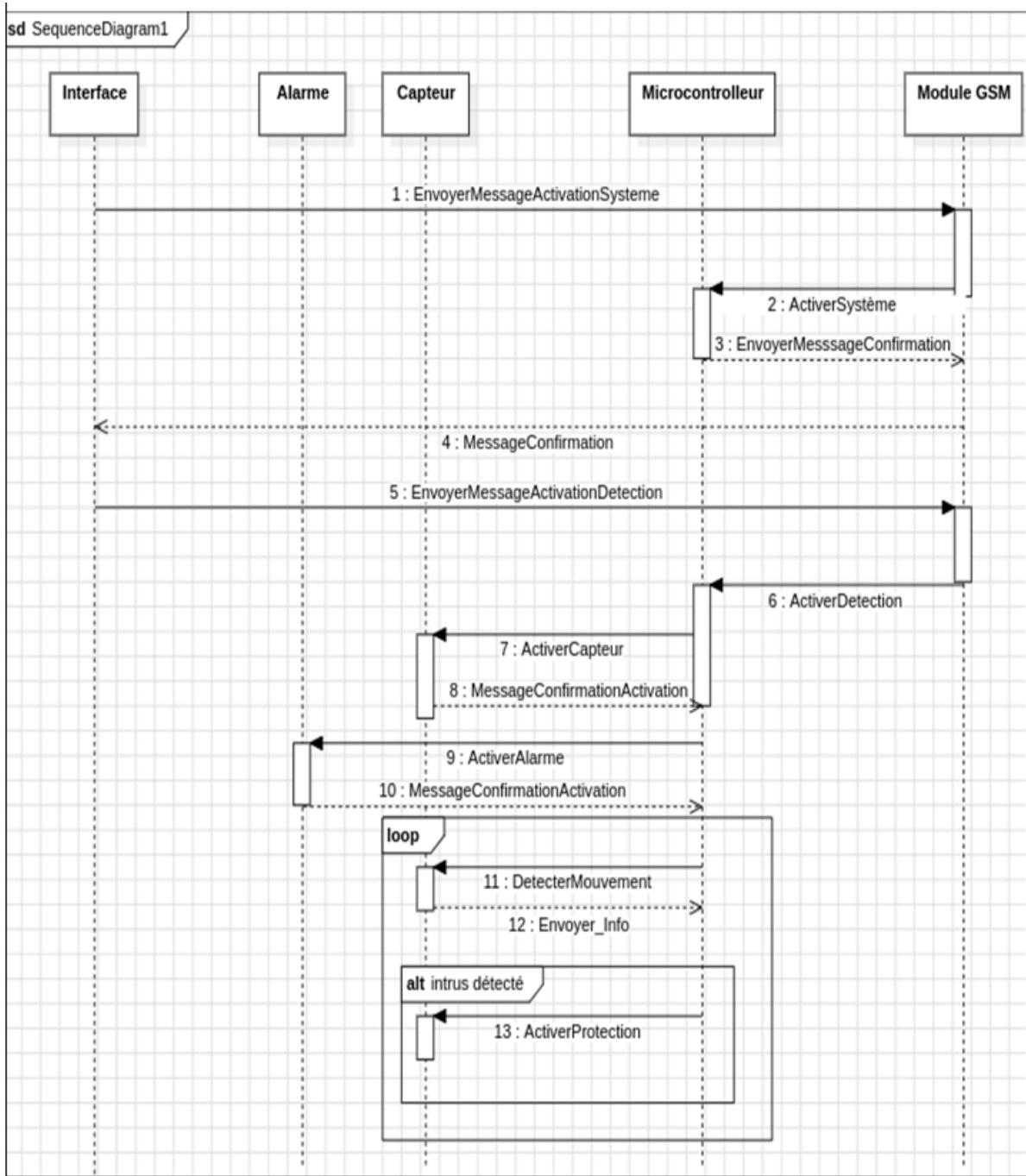
- ❖ Stopper l'alarme



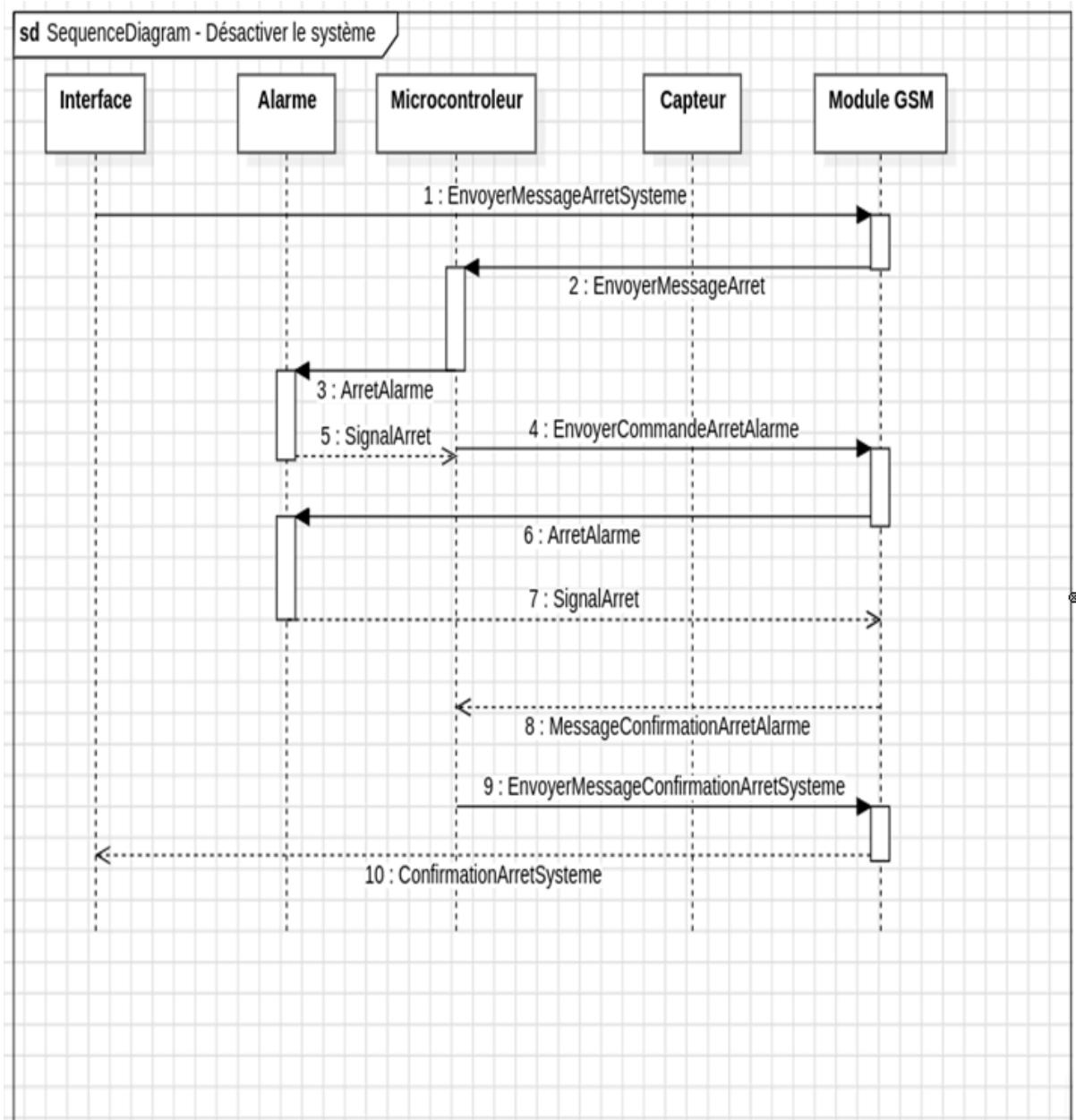
❖ Déclencher l'alarme



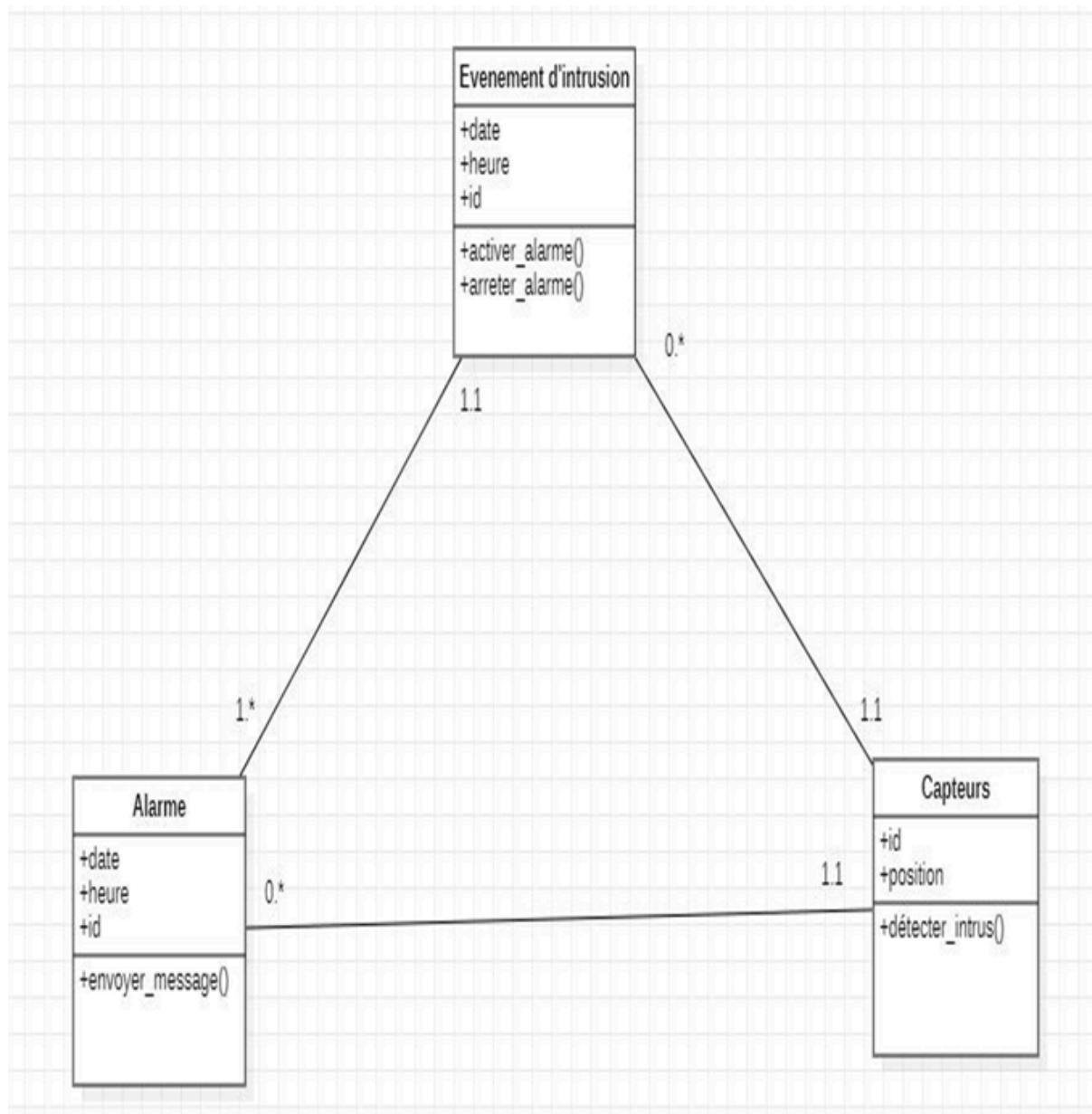
❖ Activer le système



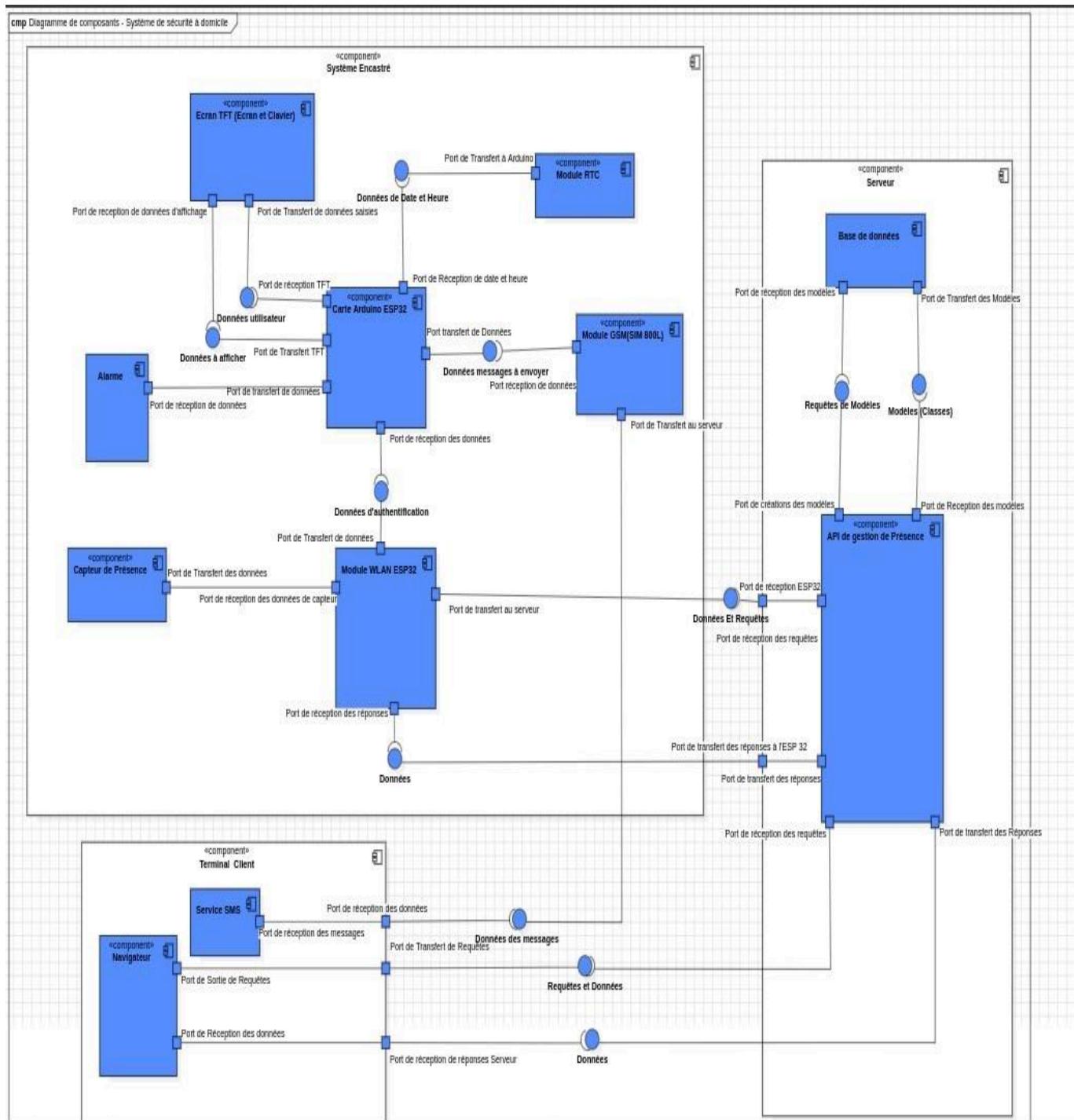
❖ Désactiver le système



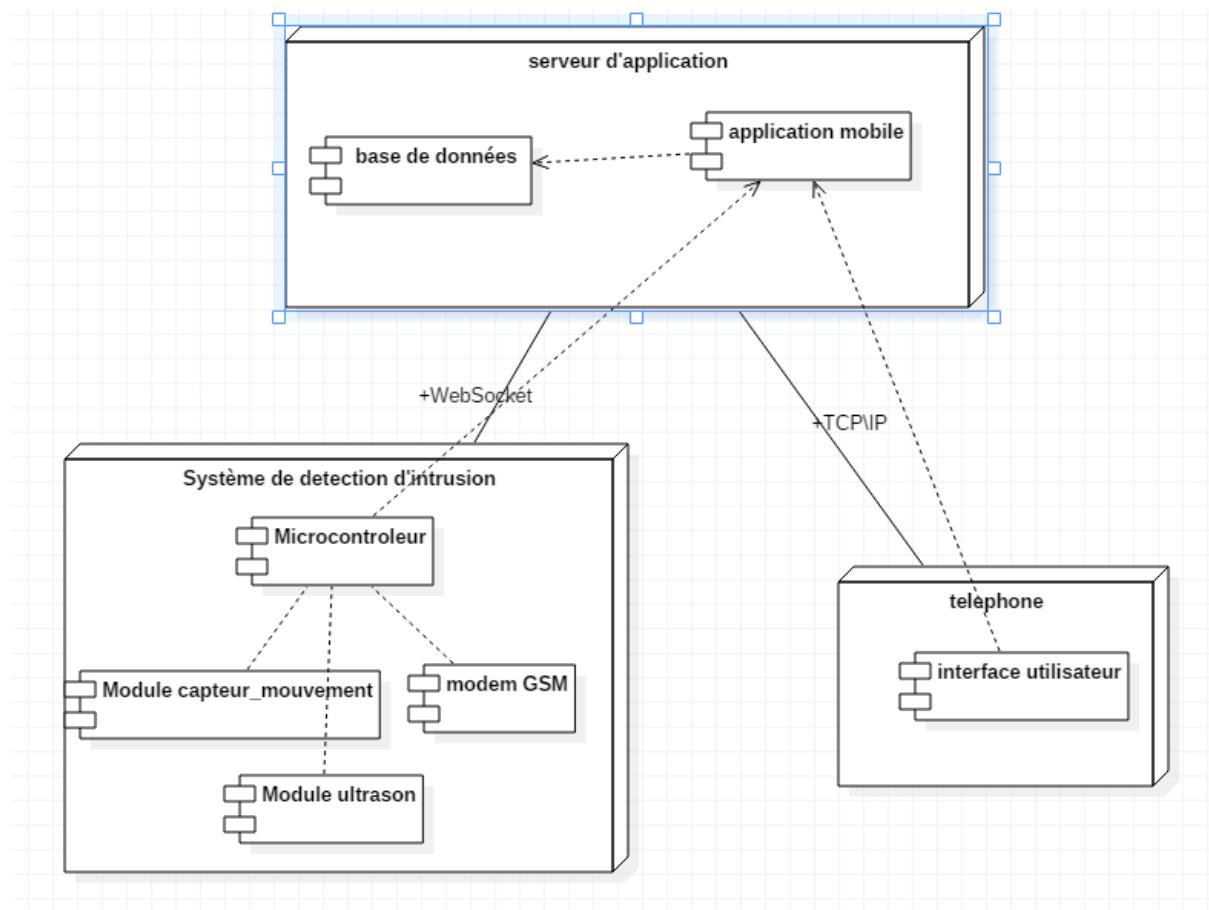
2. Diagramme de classe technique



3. Diagramme de composants



4. Diagramme de déploiement



CHAPITRE 3: CAHIER DE RÉALISATION

I. DESCRIPTIONS DES SPÉCIFICATIONS TECHNIQUES

a) Les choix technologiques

Aux vues des besoins techniques nous avons prévu pour la conception du dispositif électronique d'utiliser : Le module ESP32 qui est un microcontrôleur polyvalent utilisé dans les projets d'Internet des objets (IoT). Il dispose de fonctionnalités avancées telles que la connectivité Wi-Fi et Bluetooth intégrée, une architecture à double cœur, ainsi que des interfaces GPIO, UART, I2C, SPI, etc. Il peut être programmé avec l'IDE Arduino, C/C++, ESP-IDF ou MicroPython. Le module est économique en énergie et offre différentes options de gestion de l'alimentation.

b) L'environnement/l'architecture/Contraintes logicielles

i. L'architecture de conception

Nous utiliserons le modèle d'architecture client-serveur

ii. Système d'exploitation

L'outil devra fonctionner sur :

- Windows 10 et 11
- Linux (avec une interface graphique) quel que soit la distribution
- Android
- IOS

iii. Serveur Web et Serveur d'application

Le serveur Web et le serveur d'application doivent s'appuyer sur des standards libres de droits et ne doivent pas nécessiter l'acquisition d'outils ou de licences particulières.

c) Les exigences de programmation(langage informatique)

- Dans la conception du système de détection d'intrusion, nous utiliserons le langage arduino, étant un langage permettant une fluidité dans le traitement et l'échange de données
- Pour la conception de l'application, nous utiliserons les langages react natives et javascript
- Pour le serveur en ligne, il est développé en node js.

d) L'accessibilité (compatibilité navigateurs, logiciels, appareils)

La version mobile sera compatible avec les appareils mobiles, tablette, ordinateur portable uniquement.

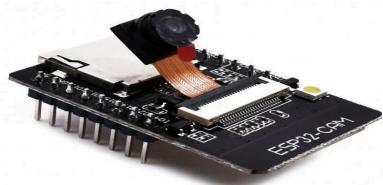
e) La sécurité

L'accès à l'application mobile sera limité au propriétaire du bien ou personnel autorisé dans le cadre d'une organisation ou dans le cadre d'une entreprise.

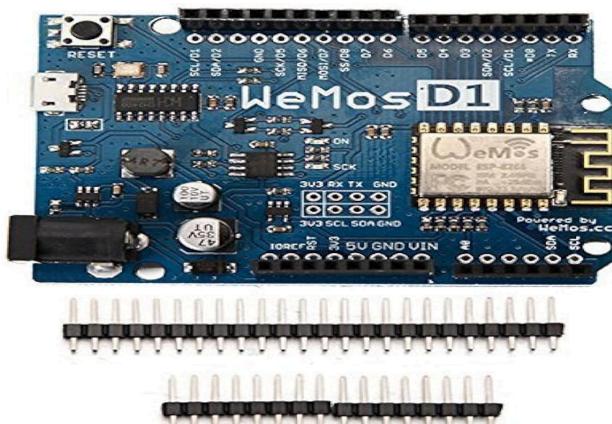
II. LISTE DU MATÉRIEL

Pour réaliser notre projet, nous avons eu à mener plusieurs analyses et réflexions en nous appuyant sur les paradigmes de programmation de chaque équipement, et la compatibilité de ceux-ci; aux termes desquelles le matériel suivants ont été retenus:

→ **2 cartes esp32-Cam AI-Thinker with OV2640 camera** : La carte Esp32-CAM AI-Thinker est embarquée avec une caméra qui peut facilement être programmé selon nos besoins pour la capture d'images et la prise de vidéos; Chaque carte permet de connecter des objets physiques au monde numérique via une connexion Wi-Fi ou Bluetooth grâce aux modules WiFi et Bluetooth qu'elle intègre. De plus, le lecteur de carte SD intégré permet de stocker les images et vidéos prises par la caméra (Mémoire de la carte SD extensible jusqu'à 8GB). On peut aussi noter que la caméra peut être changée et il existe plusieurs autres modèles de caméra compatibles avec cette carte dépendant des caractéristiques et des spécificités de l'image ou de la vidéo (la clarté, la netteté de l'image).



→ **1 carte LOLIN (Wemos) D1 R2 Mini**: Cette carte est basée sur le module ESP8266, qui est un microcontrôleur Wi-Fi intégré. Elle dispose d'un processeur à 32 bits, d'un connecteur USB pour la programmation et l'alimentation, ainsi que d'une interface de broches pour l'ajout de capteurs, d'actionneurs et d'autres périphériques. Elle est compatible avec l'environnement de développement Arduino, ce qui nous a permis de la programmer en utilisant le langage de programmation Arduino et bénéficier de la vaste bibliothèque de codes et de ressources disponibles pour Arduino. Nous l'avons appréciée pour sa facilité d'utilisation, son faible coût et sa flexibilité.



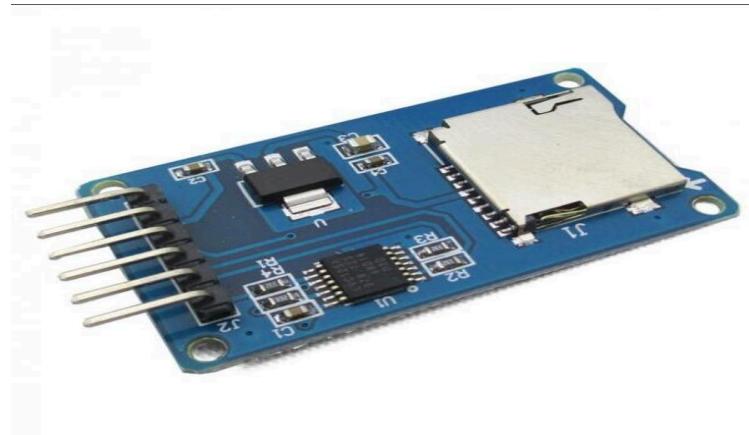
→ **2 HC-SR501 Mini PIR Sensor Capteur de présence:** C'est un capteur infrarouge passif qui détecte les variations de température causées par les mouvements. Nous l'avons utilisé pour la détection de mouvements et de présence dans un certain périmètre autour du local à protéger (3m à 7m réglable). Lorsqu'un mouvement est détecté, le capteur envoie un signal de sortie, généralement sous forme d'un signal numérique (HIGH ou LOW), indiquant la présence d'un mouvement. Ce capteur est alimenté en tension continue (5V généralement) et possède trois broches principales : VCC (alimentation), GND (masse) et OUT (sortie du signal).



→ **1 HC-SR505 Mini PIR Sensor Capteur de présence:** C'est un petit capteur utilisé pour détecter les mouvements humains dans son champ de vision(détection de présence). Nous l'avons choisi car celui-ci est compact et facile à utiliser. Il fonctionne en détectant les variations de chaleur infrarouge émises par les objets en mouvement comme son homologue HC-SR501.



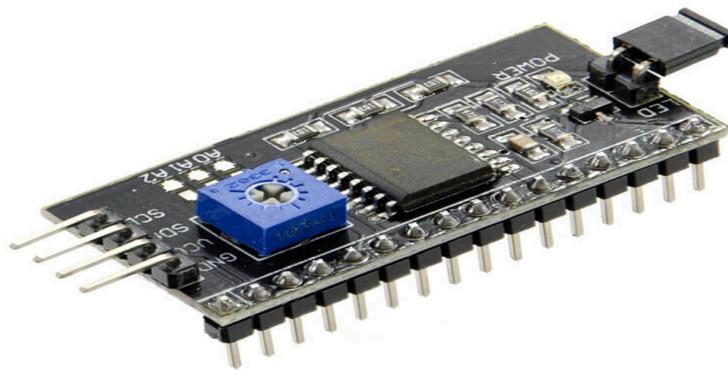
→ **1 lecteur de carte SD pour microcontrôleur:** Ce module permet à un microcontrôleur de lire et d'écrire des données sur une carte mémoire SD (Secure Digital). Il se connecte au microcontrôleur Wemos à l'aide d'une interface de communication, SPI (Serial Peripheral Interface). Il fournit une interface simple et standardisée pour le microcontrôleur afin d'interagir avec la carte SD. Il est capable de lire et d'écrire des fichiers sur la carte SD, ce qui permet de stocker des données telles que des fichiers texte, des images et des vidéos.



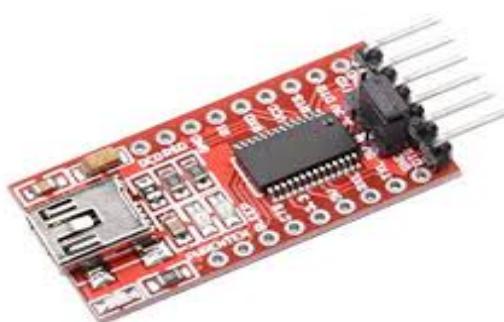
→ **1 module RTC DS1302:** il s'agit d'une horloge en temps réel couramment utilisée pour suivre l'heure. Il est utilisé pour fournir une mesure précise du temps et de la date dans les systèmes électroniques. Le DS1302 intègre un oscillateur en temps réel et une mémoire non volatile pour stocker les informations de date et d'heure, même lorsque l'alimentation est coupée. Notre module RTC DS1302 se connecte au microcontrôleur Wemos via une interface série à trois fils (Data, Clock, et Reset). Nous l'avons utilisé pour lire et de programmer l'heure et la date, ainsi que d'autres fonctionnalités telles que l'alarme et le suivi de l'alimentation.



→ **1 module I2C** : c'est un périphérique qui permet de connecter un écran LCD à un microcontrôleur via le protocole I2C. Il simplifie la connexion et le contrôle de l'écran LCD en réduisant le nombre de fils nécessaires. Le module I2C joue un rôle essentiel en convertissant les commandes et les données du microcontrôleur en signaux compréhensibles pour l'écran LCD. En utilisant un module I2C, vous économisez des broches d'E/S sur le microcontrôleur, car la communication I2C utilise seulement deux fils pour connecter plusieurs périphériques.



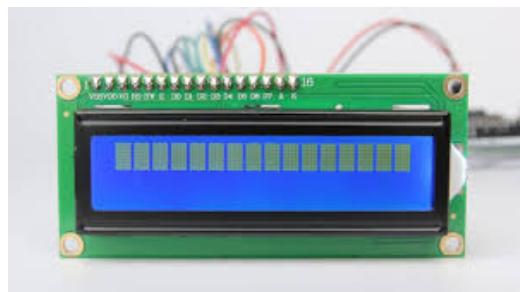
→ **USB to TTL FT232 programmer**: Il permet d'établir une connexion entre notre ordinateur et notre carte ESP32 CAM, via une interface de communication série TTL (Transistor-Transistor Logic). Pour l'ESP32 CAM, le programmeur USB vers TTL FT232 est utilisé pour téléverser le code du microcontrôleur depuis l'IDE Arduino ou d'autres outils de développement. Nous l'avons également utilisé pour afficher les messages de débogage et établir une communication série avec l'Esp.



- **Programmeur ESP32-CAM AI-Thinker MB** : utilisé pour téléverser efficacement le code dans l'ESP32 CAM.



- **1 Afficheur LCD 16*02 (16 colonnes sur 2 lignes)** : Utilisé pour afficher les messages à l'utilisateur tel que les confirmations de connexions, la date et l'heure.



- **1 Transistor BC547 NPN**



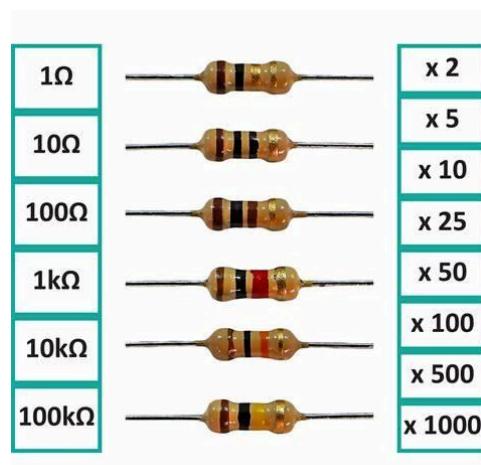
- **1 pile 2032 pour module RTC**



- Buzzer (passif et actif)



- Des résistances (220 Ohm, 1k Ohm, 10k Ohm)



- Fils d'étain
- des fils de connexion (mâle-mâle, mâle-femelle, femelle-femelle)
- 1 multimètre
- 1 fer à souder

III. PROGRAMMATION DES CONSTITUANTS DU SYSTÈME

A. ECRITURE DES PROGRAMMES POUR LES ÉQUIPEMENTS ÉLECTRONIQUES

Voici une explication pour chaque partie du programme :

1. **Fonction de configuration de la caméra [ANNEXE 1]**: Cette fonction permet de configurer les paramètres de la caméra utilisée dans le système. Cela peut inclure des réglages tels que la résolution de l'image, la fréquence d'images, etc.
2. **Fonction de capture d'image [ANNEXE 2]**: Cette fonction est responsable de capturer une image à partir de la caméra configurée. Elle déclenche le processus de capture d'une image fixe ou en continu, selon les besoins du système.
3. **Fonction d'initialisation de la carte SD [ANNEXE 3]**: Cette fonction initialise la carte SD utilisée pour stocker les données, telles que les images capturées. Elle s'assure que la carte est prête à recevoir et stocker des informations.
4. **Fonction de connexion au WEMOS [ANNEXE 4]**: Cette fonction établit une connexion avec le module WEMOS, qui peut être utilisé pour la communication sans fil ou d'autres fonctionnalités spécifiques du système.
5. **Fonction de transfert d'image au WEMOS [ANNEXE 5]**: Cette fonction permet de transférer les images capturées vers le module WEMOS. Cela est utile pour envoyer les images à l'application mobile pour un traitement ultérieur.
6. **Fonction de transfert d'image à Telegram [ANNEXE 6]**: Cette fonction gère le transfert des images capturées vers Telegram, une plateforme de messagerie populaire. Cela peut permettre aux utilisateurs de recevoir les images sur leur compte Telegram.
7. **Fonction de réception des messages [ANNEXE 7]**: Cette fonction est chargée de recevoir et traiter les messages entrants. Cela peut inclure la réception de commandes ou d'instructions à partir de Telegram ou d'autres sources pour contrôler le système.

B. ECRITURE DES PROGRAMMES POUR L'APPLICATION MOBILE

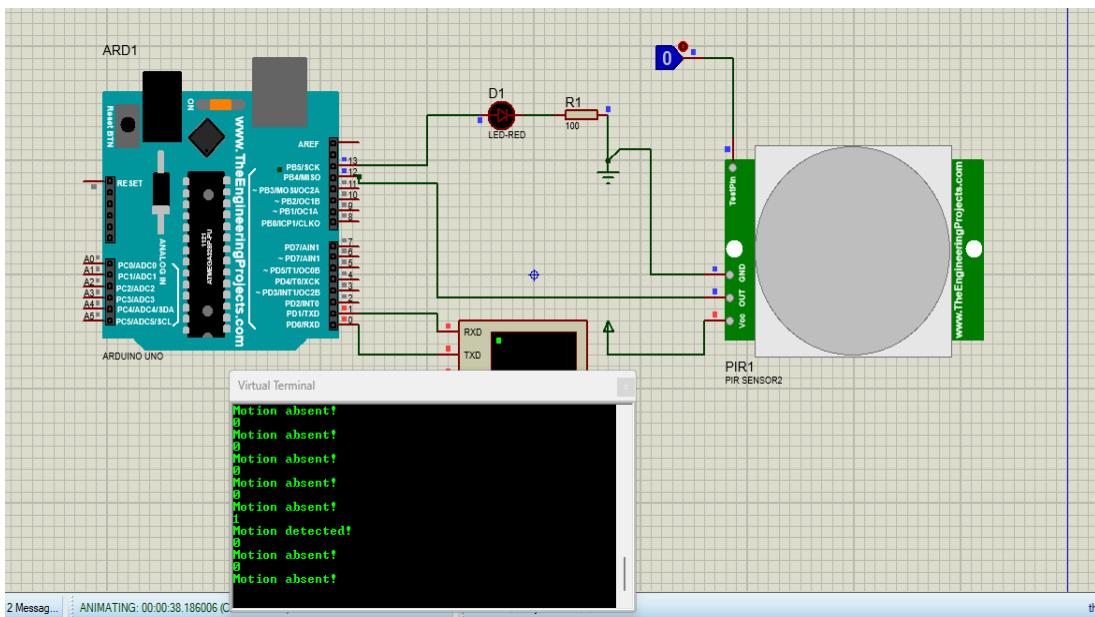
1. **Fonction qui affiche la liste de toutes les intrusions ayant eu lieu sur le domicile surveillé [ANNEXE 8]:**Cette fonction est destinée à récupérer les données relatives aux intrusions qui sont stockées sur notre base de données qui est déployée.
2. **Fonction qui affiche l'intrusion la plus récente [ANNEXE 9]:**Cette fonction affiche uniquement les informations relatives à l'intrusion la plus récente détectée par le système.
3. **Fonction qui permet la configuration automatique de notre système [ANNEXE 10]:**Par configuration automatique, nous entendons paramétriser le système de telle sorte qu'il s'active à la date du jour en cours et à l'heure à laquelle la fonction est appelée et se désactive au 31 Décembre de l'année en cours.
4. **Fonction qui permet la configuration manuelle de notre système [ANNEXE 11]:**Cette fonction active et désactive le système aux dates qui sont fixées manuellement par l'utilisateur.

IV. SIMULATION DU CIRCUIT AVEC PROTEUS

1. PIR sensor - Arduino Uno

Les composants utilisés sont les suivants :

- **Arduino Uno** : C'est la carte principale de développement qui contrôle le fonctionnement du circuit. Elle interprète les données du capteur PIR pour prendre des décisions en conséquence.
- **Diode (LED)** : Il s'agit d'un composant électroluminescent qui s'allume lorsque le courant le traverse dans un seul sens. Le passage du courant ici est provoqué par le changement d'état du logic toggle. Elle est utilisée pour afficher un signal visuel en réponse à la détection du mouvement par le capteur PIR. Elle s'allume ou s'éteint en fonction des informations reçues par la carte Arduino.
- **Résistance** : Utilisé pour limiter le courant qui traverse la diode et la carte Arduino Uno les protégeant ainsi contre une surcharge de courant.
- **Logic Toggle** : Le commutateur logique peut être utilisé pour simuler un changement d'état ou une action mutuelle dans le circuit, ce qui peut influencer le comportement du capteur PIR ou de la LED.
- **Capteur PIR (Passive Infared Sensor)** : Il détecte les mouvements en mesurant les variations de chaleur émises par les objets. Lorsqu'il détecte un mouvement, il renvoie un signal à la carte Arduino pour déclencher une action, comme allumer la LED.

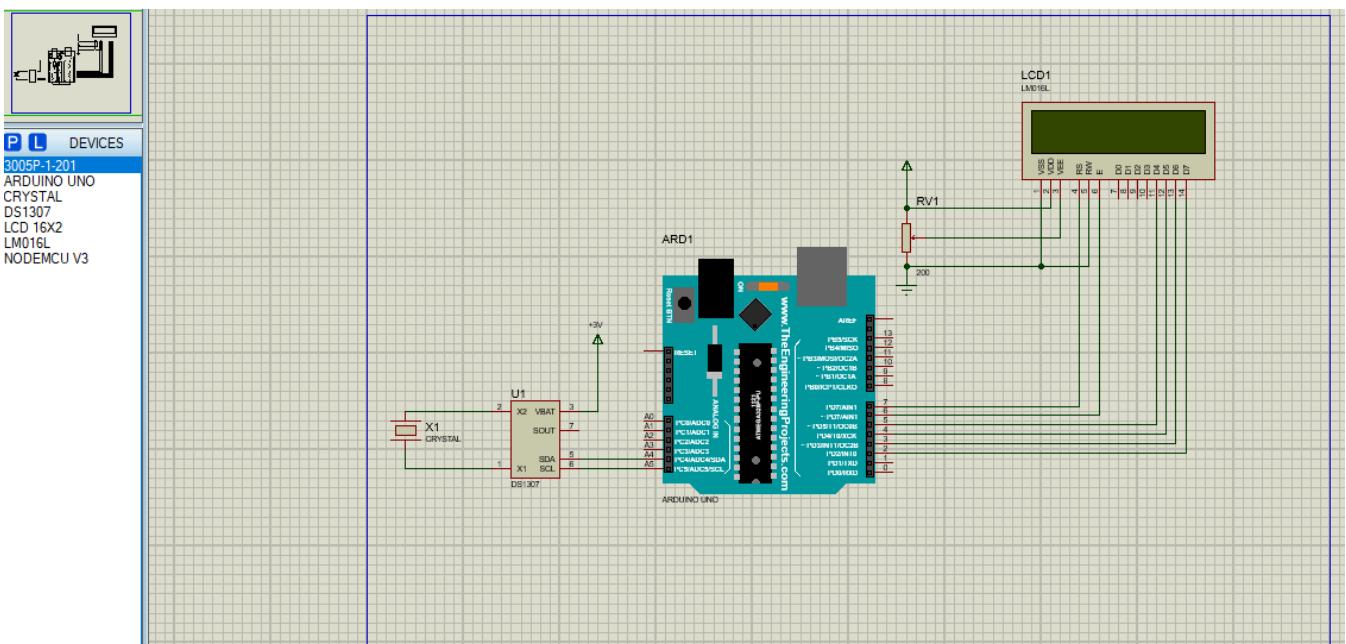


2. WEMOS D1 R2 Mini + Afficheur LCD + Module RTC (Real Time Clock)

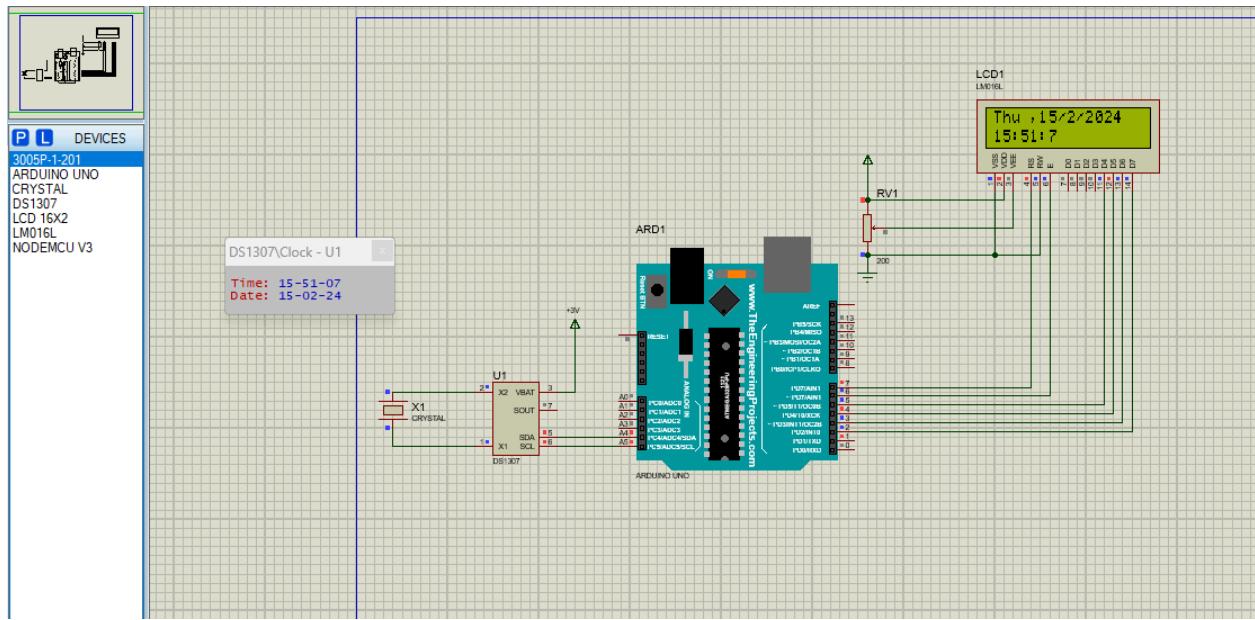
Les composants utilisés sont les suivants :

- **WEMOS D1 R2 Mini (NodeMCU V3) :** C'est un microcontrôleur basé sur ESP8266 que nous avons programmé pour contrôler le fonctionnement du circuit et interagir avec les autres composants.
- **Afficheur LCD 16-2 :** Utilisé pour afficher les informations telles que l'heure et la date du module RTC ou d'autres données venant du microcontrôleur.
- **Module RTC-DS1307 :** Il est utilisé pour garder une trace précise de l'heure et de la date, ce qui est utile pour les applications nécessitant une horloge en temps réel.
- **CRYSTAL :** Il s'agit d'un composant utilisé pour fournir une référence de temps précise au module RTC, assurant une mesure précise du temps.
- Les éléments "POWER" et "GROUND" dans le mode terminal représentent respectivement l'alimentation électrique (tension) reliée à la résistance et au module RTC et la connexion à la masse (référence de tension 0V) sur l'autre borne de la résistance. Ces éléments sont utilisés pour fournir une alimentation électrique adéquate et établir une référence de tension commune pour les composants du circuit.

Montage du début avant la simulation



Montage après la simulation



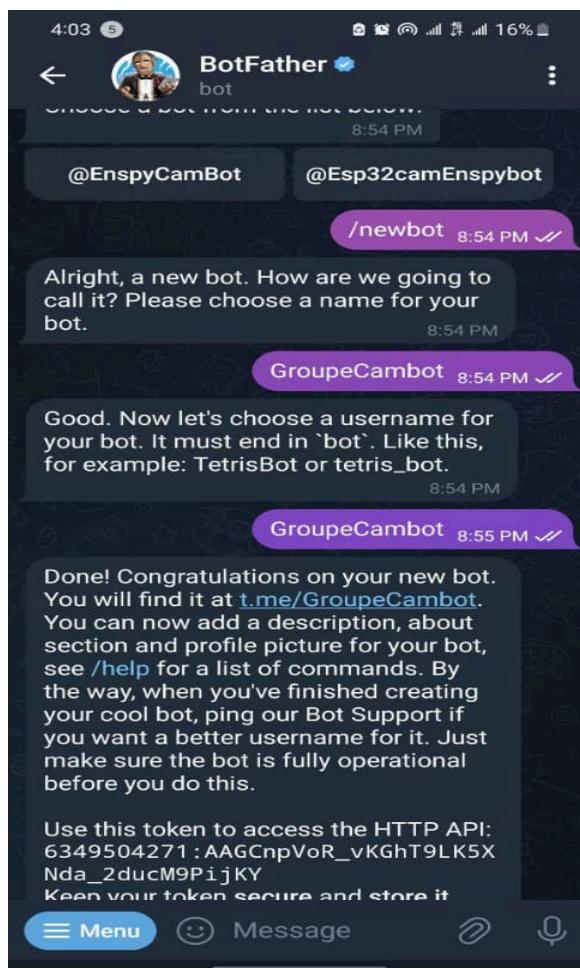
IV. COMMUNICATION ENTRE TELEGRAM ET LE MODÈLE ESP32-CAM

Lors de la détection, le modèle ESP32 enregistre une photo puis envoie au propriétaire via telegram. Ceci se réalise avec la création d'un bot qui entamera une discussion avec le propriétaire en envoyant la photo de l'individu capturé. Pour créer ce bot, on doit:

1. Ouvrez l'application Telegram et recherchez le "BotFather".



2. Commencez une conversation avec le BotFather et suivez les instructions pour créer un nouveau bot (dans notre cas, nous avons attribué le nom “GroupeCambot”).



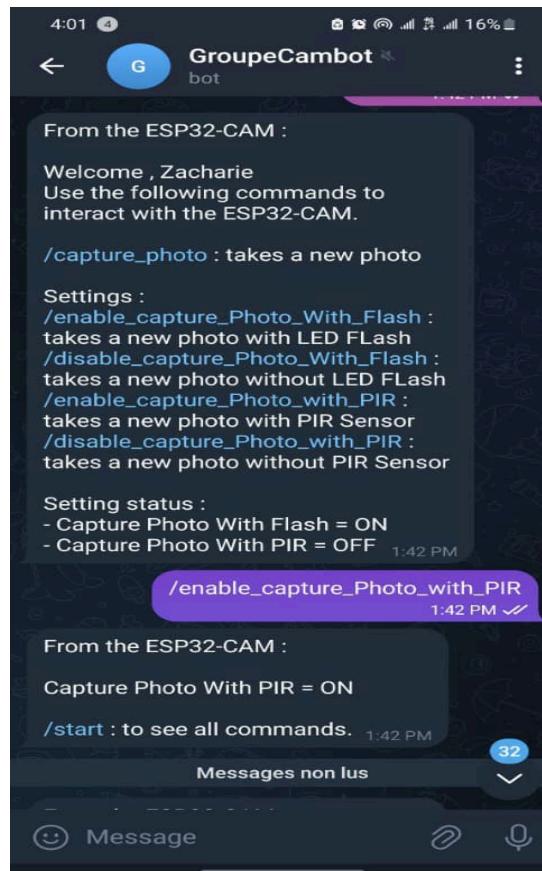
3. Une fois que vous avez créé le bot, le BotFather vous fournira un token d'accès.

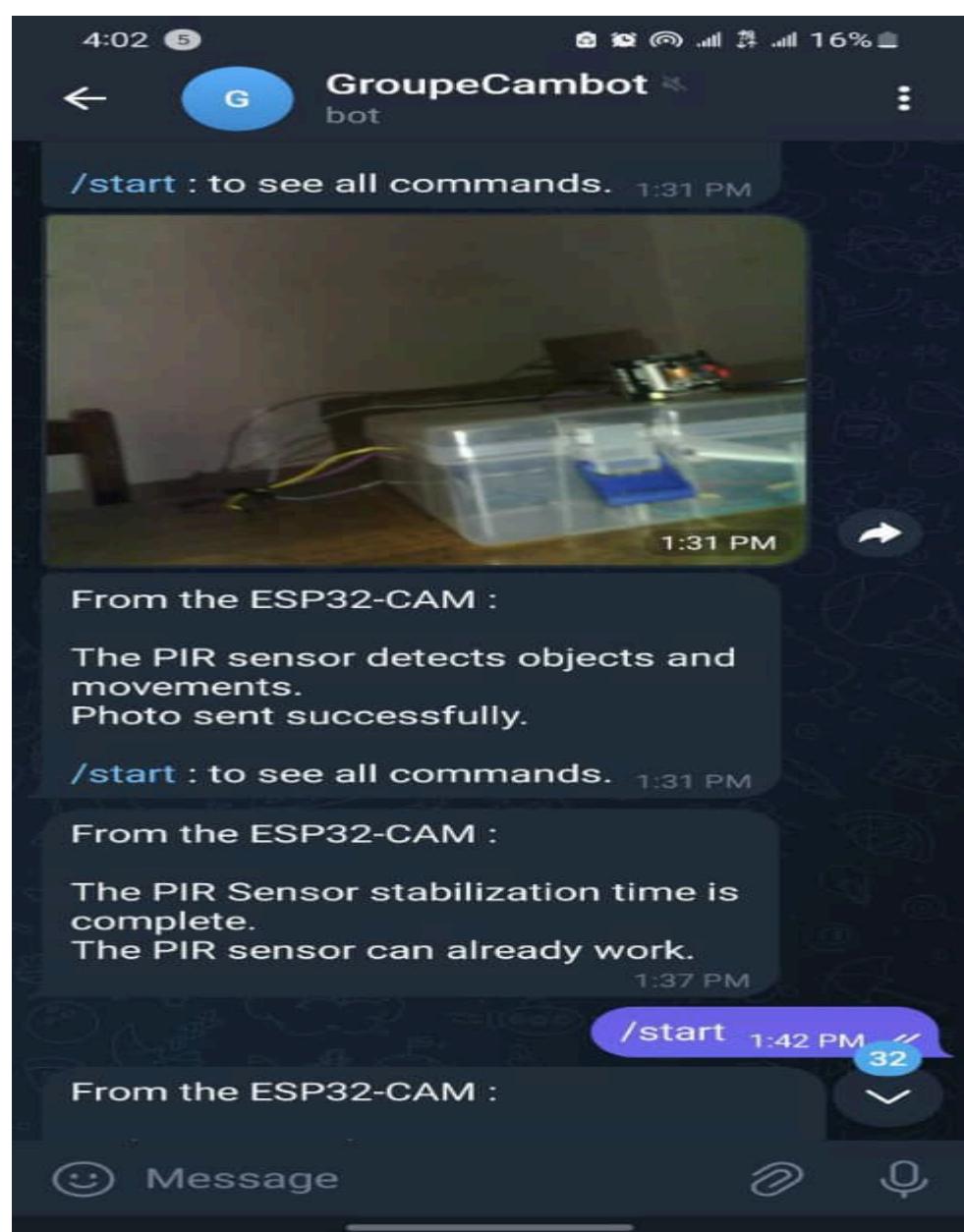
Use this token to access the HTTP API:
6349504271:AAGCnpVoR_vKGhT9LK5X
Nda_2ducM9Pi jKY

Ensuite, vous devrez installer la bibliothèque **ArduinoJSON** et la bibliothèque **UniversalTelegramBot** pour l'ESP32. Voici comment procéder :

1. Ouvrez l'IDE Arduino.
2. Allez dans "Croquis" -> "Inclure une bibliothèque" -> "Gérer les bibliothèques".
3. Recherchez "ArduinoJSON" et installez la bibliothèque.
4. Recherchez "UniversalTelegramBot" et installez également cette bibliothèque.

Par la suite, on écrit le code qui permettra la communication entre l'ESP32 et Telegram et après son exécution, ce code se connecte à un réseau Wi-Fi à l'aide des identifiants fournis (dans notre code, nous avons SSID="MTNMIFI_79AFC5" et password="925D42A7"). Suite à cela le code traite les messages entrants un à un puis envoie des réponses à notre bot créée à travers la méthode `bot.sendMessage()`, ce qui entraîne une communication entre notre bot et le propriétaire

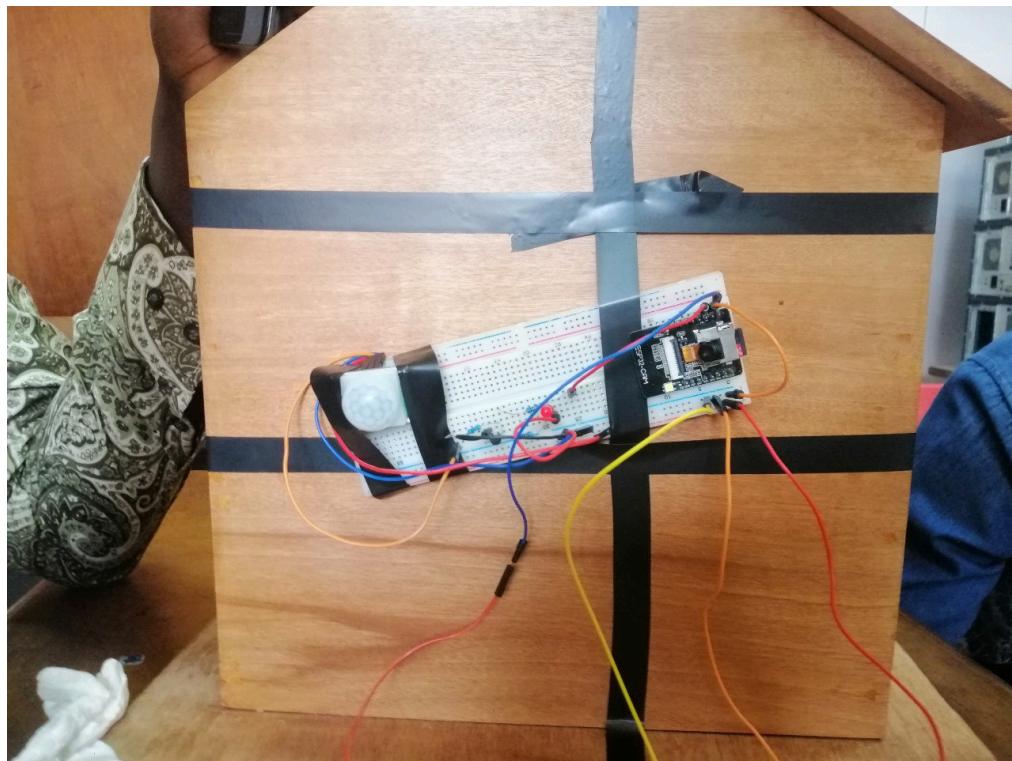




IV. GUIDE D'INSTALLATION ET GUIDE D'UTILISATION

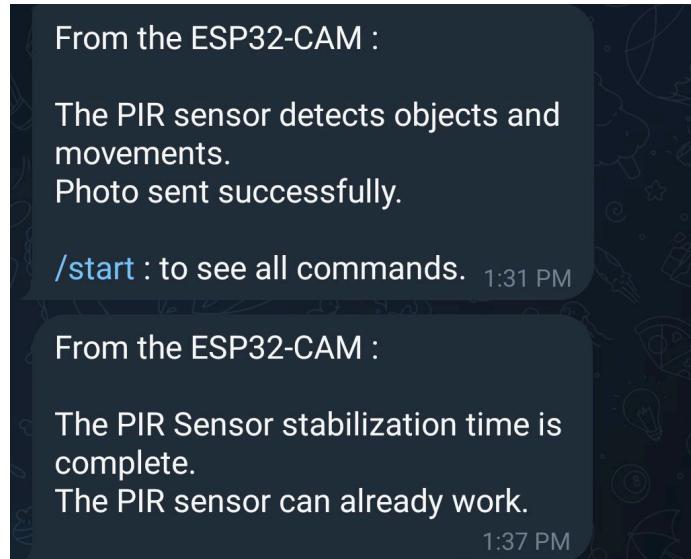
A. Manipulation à distance du système avec Telegram

1. Positionner module de détection dans la zone que vous voulez sécuriser.

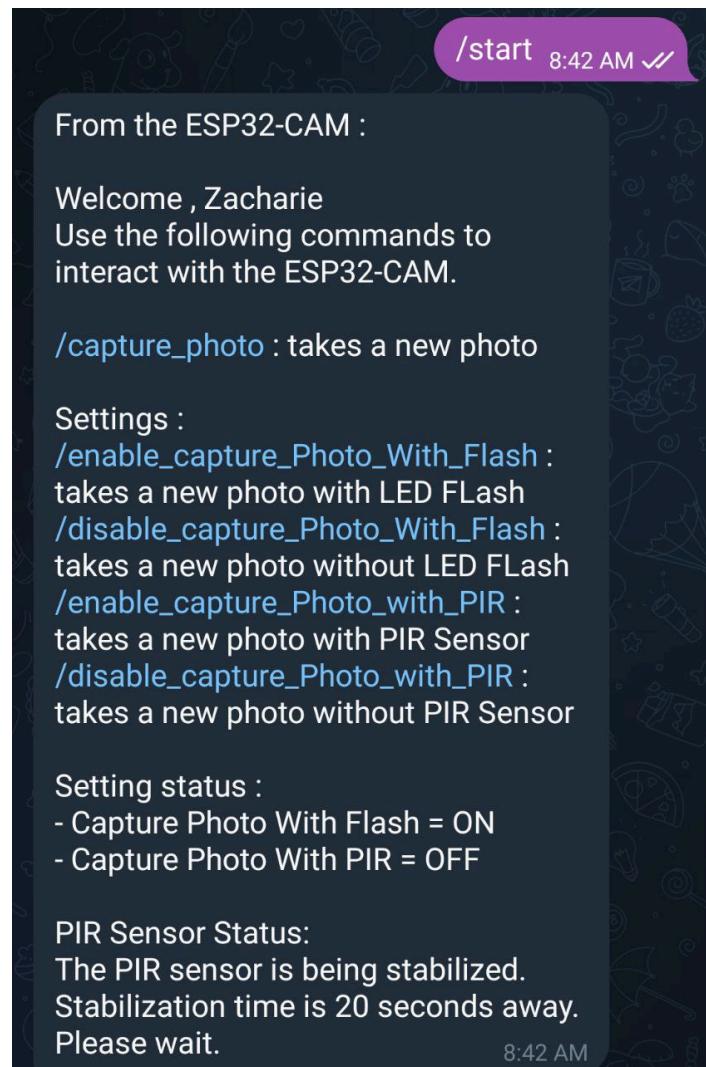


2. Lancer Telegram et faire les configurations pour avoir le bot.

3. Mettre en marche le module en branchant l'alimentation
4. Patienter une minute pour le démarrage complet et la connexion au wifi
5. Après ce temps écoulé vous allez recevoir des messages.



6. Taper la commande /start et vous recevrez le guide des commandes



7. En utilisant les commandes précédentes précédentes vous pouvez effectuer ce qui suit :
 - Activer/désactiver le flash lors de la prise de photo

/enable_capture_Photo_With_Flash :
takes a new photo with LED FFlash
/disable_capture_Photo_With_Flash :

- Activer/désactiver la prise de photo par le capteur de mouvement

```
/enable_capture_Photo_with_PIR :  
takes a new photo with PIR Sensor  
/disable_capture_Photo_with_PIR :  
takes a new photo without PIR Sensor
```

8. Après avoir effectué les configurations requises vous pouvez obtenir des résultats selon la configuration que vous avez donnée. voici un résultat obtenu apres avoir activer la prise d'image lors de la détection de mouvement.

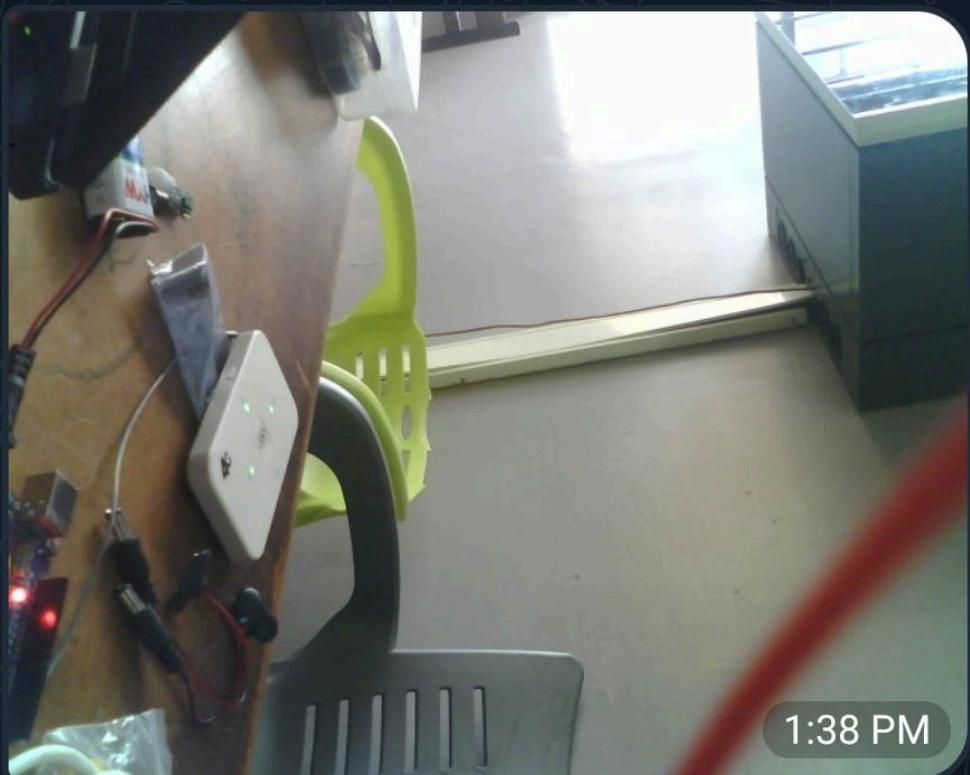


From the ESP32-CAM :

The PIR sensor detects objects and movements.

Photo sent successfully.

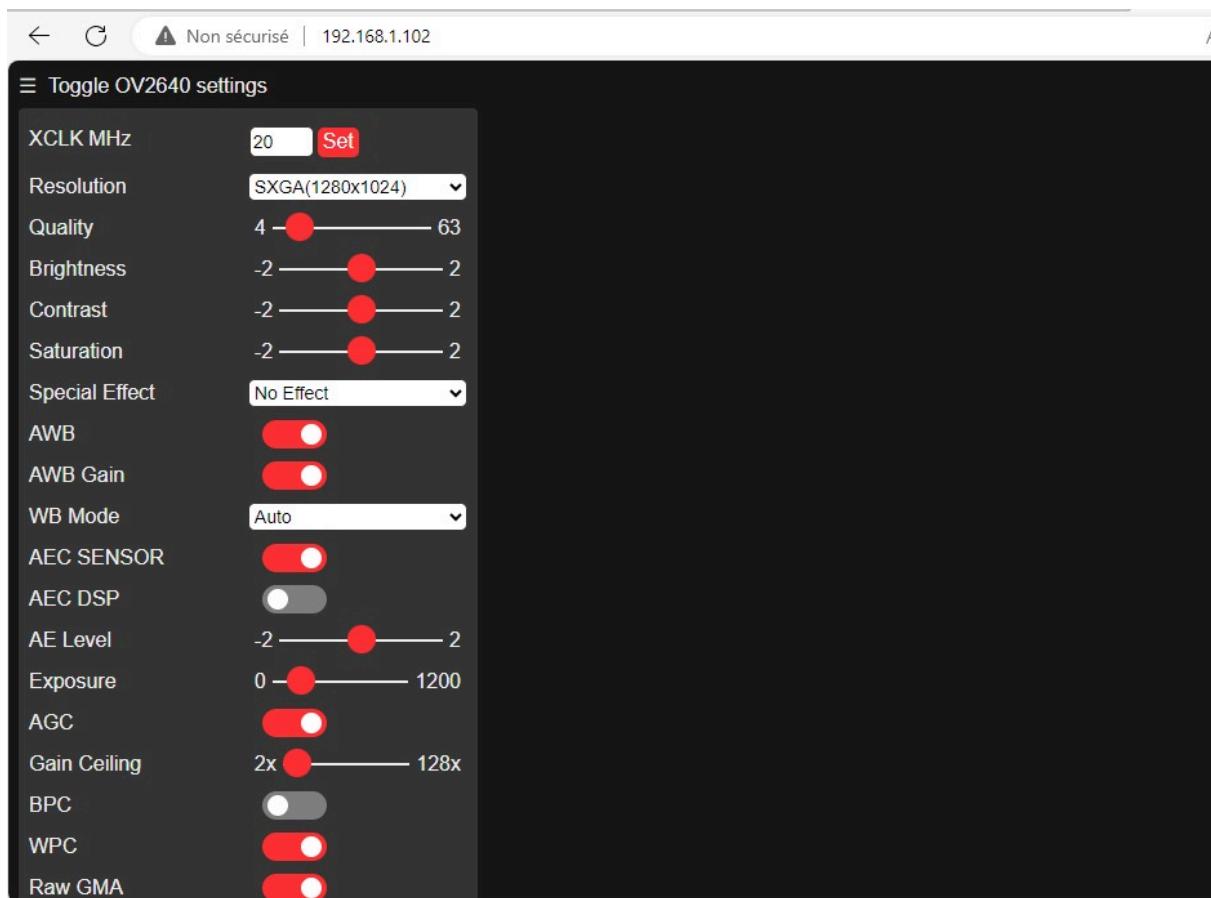
/start : to see all commands. 1:37 PM



B. Manipulation du système en Locale

Dans l'optique de donner la possibilité à l'utilisateur de pouvoir utiliser les caméras pour surveiller sa maison nous avons ajouté une option de vision en temps réel. Pour cela il faut suivre les étapes suivantes:

1. Utiliser un téléphone, une tablette ou un PC pour se connecter au même wifi que l'ESP-CAM
2. Récupérer l'adresse IP de l'ESP-CAM pour taper cela dans un navigateur. l'interface suivante s'affichera.



C. Manipulation du système à distance avec l'application développée

Pour des raisons de sécurité et de non dépendance aux outils déjà existant et pour des questions de propriété intellectuelle nous avons décidé de mettre en place une application mobile qui étend les fonctionnalités déjà présentées. Cette application présente les pages suivante:

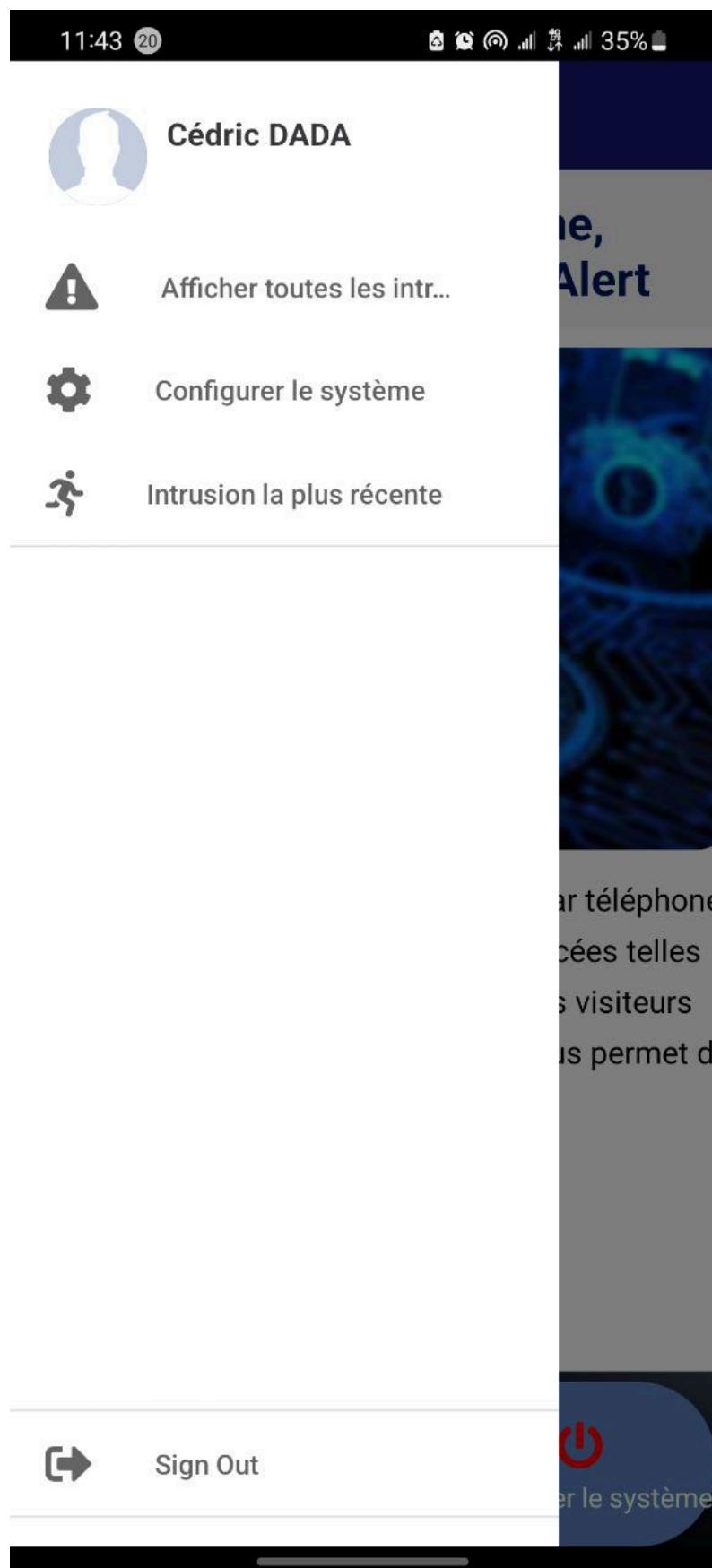
1. Une page d'accueil qui affiche un carrousel donnant les avantages d'un système de sécurité à domicile et des boutons pour activer et désactiver le système.



La visualisation en direct des caméras de sécurité sur votre téléphone vous permet de surveiller votre maison en temps réel, ce qui peut dissuader les intrus et vous offrir une tranquillité d'esprit supplémentaire



2. Une barre de navigation qui permet à l'utilisateur de changer de page facilement.



3. Une page qui affiche toutes les intrusions détectées.

The image shows a smartphone screen displaying a list of detected intrusions. The top status bar shows the time as 11:43, signal strength, battery level at 35%, and other icons. Below the status bar, the title "AllIntrusions" is displayed next to a menu icon (three horizontal lines). The main content area contains two entries, each in its own card:

Intrusion no 1

Nous avons détecté une intrusion dans la zone 1

Date: 28 February 2024: 00h00

A small video thumbnail showing a person from the chest up, wearing a dark shirt, looking upwards. The background shows an indoor setting with ceiling lights.

 Appeler la police

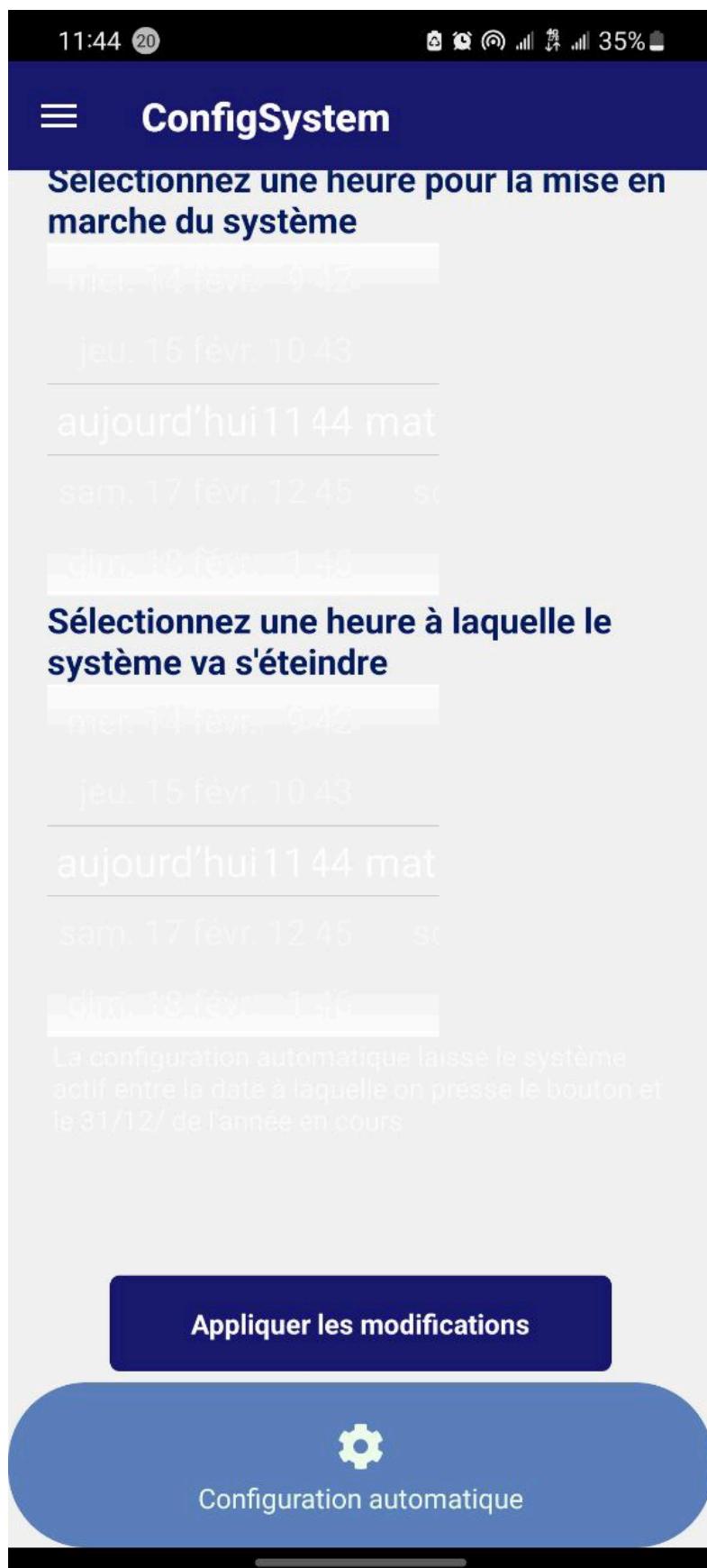
Supprimer

Intrusion no 2

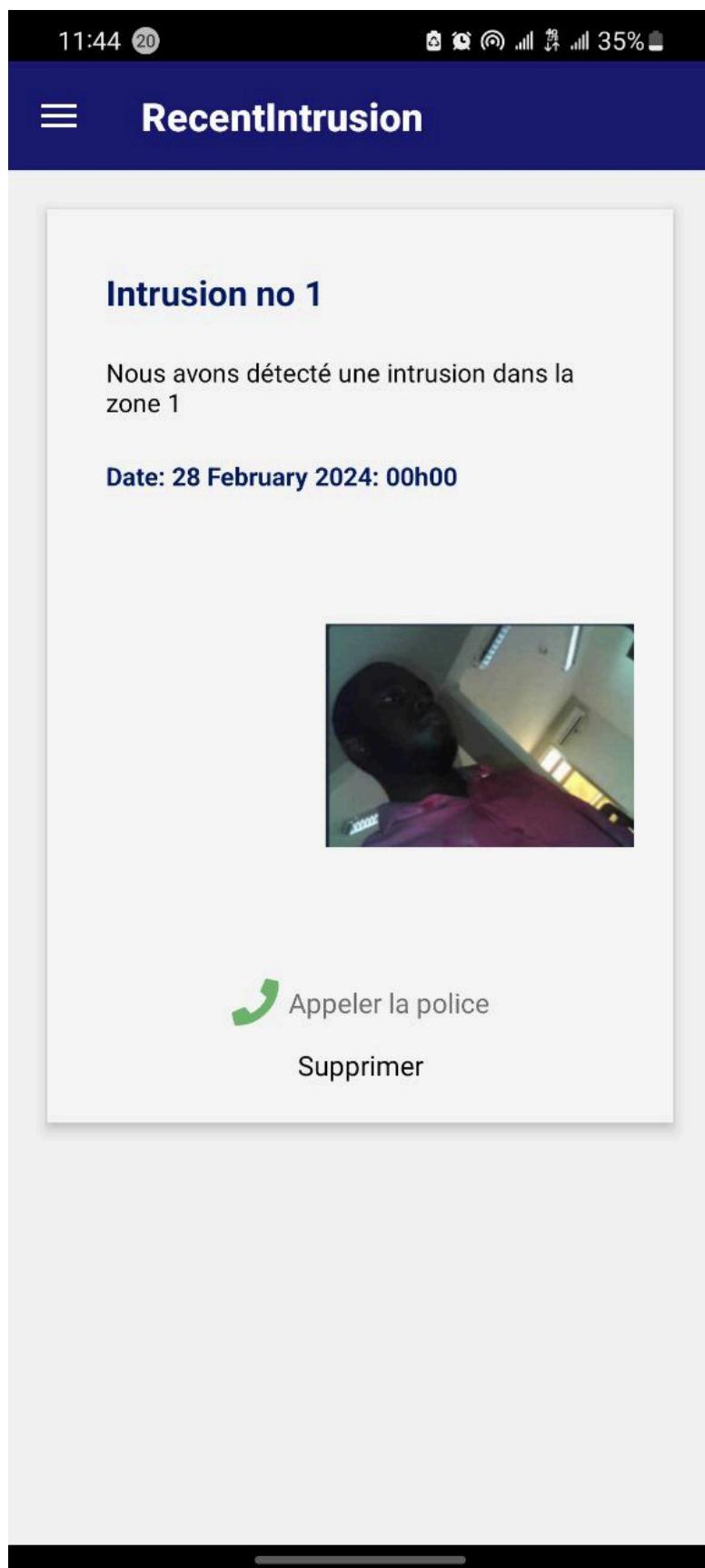
Nous avons détecté une intrusion dans la zone 2

Date: 13 February 2024: 00h00

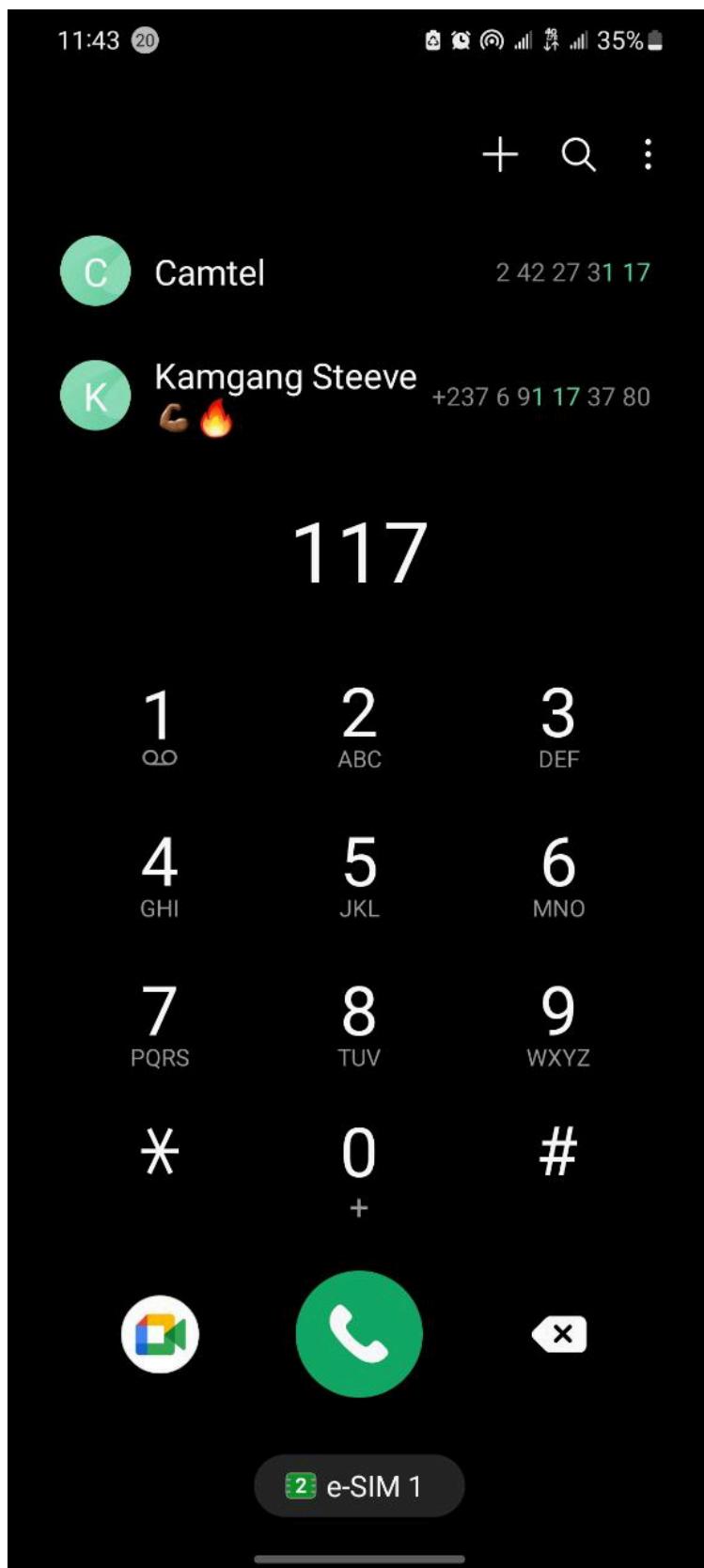
4. Une page qui permet de configurer le système



5. Une page qui présente l'intrusion la plus récente



6. Ayant détecté une intrusion on a la possibilité d'appeler la police en utilisant le bouton “Appeler la police” qui nous renvoi vers l'application d'appel du téléphone.



V. PERSPECTIVE D'AMÉLIORATION

Pour améliorer notre système de détection d'intrusion, voici quelques perspectives d'amélioration possibles :

1. **Intégration de capteurs supplémentaires** : Ajouter d'autres capteurs comme des capteurs de mouvement, des capteurs de pression ou des capteurs de température pour détecter plus efficacement les intrusions.
2. **Analyse d'images avancée** : Mettre en place une analyse d'image plus sophistiquée en utilisant des techniques d'apprentissage automatique pour détecter des schémas ou comportements suspects.
3. **Sécurisation des données** : Renforcer la sécurité des données en cryptant les informations sensibles stockées et transmises par le système pour éviter toute interception ou accès non autorisé.
4. **Intégration de caméras à vision nocturne** : Installer des caméras avec vision nocturne pour améliorer la détection d'intrusion pendant la nuit ou dans des conditions de faible luminosité.
5. **Mise en place de zones de détection personnalisées** : Permettre la configuration de zones de détection spécifiques pour concentrer l'analyse sur des zones clés et réduire les fausses alertes.

En implémentant ces améliorations, le système de détection d'intrusion pourrait devenir plus robuste, précis et efficace dans la prévention des intrusions.

CONCLUSION

Dans le cadre de ce projet de conception d'un système de sécurité pour domicile, nous avons pu constater l'importance cruciale d'assurer la protection et la tranquillité des occupants de leur foyer. En combinant expertise en ingénierie électronique, informatique et sécurité, nous avons réussi à élaborer un système complet et innovant répondant aux besoins actuels de sécurité résidentielle. À travers les différentes phases du projet, de la planification à la mise en œuvre, en passant par le développement et les tests, nous avons veillé à intégrer des dispositifs technologiques avancés pour offrir une protection optimale. Les choix technologiques effectués et les solutions apportées ont été pensés pour garantir la fiabilité et l'efficacité du système proposé, tout en prenant en compte les défis rencontrés. En analysant les avantages et les limites du système conçu, nous avons pu mettre en lumière ses fonctionnalités clés, sa facilité d'utilisation et sa capacité à s'adapter aux besoins spécifiques des utilisateurs, contribuant ainsi à leur sécurité et à leur confort au quotidien. En envisageant des perspectives d'amélioration et d'évolution future de ce système de sécurité pour domicile, nous restons engagés dans une démarche d'innovation continue visant à renforcer la protection des foyers face aux menaces émergentes. Ainsi, notre objectif demeure d'offrir des solutions toujours plus sophistiquées et évolutives pour garantir une sécurité optimale et une tranquillité d'esprit durable à tous les résidents.

BIBLIOGRAPHIE

- [1] http://arduino.esp8266.com/stable/package_esp8266com_index.json
- [2] https://dl.espressif.com/dl/package_esp32_index.json
- [3] https://github.com/esp8266/Arduino/blob/master/package/package_esp8266com_index.template.json
- [4] [Capture & Send Images With ESP32-Cam Using ESP8266 WeMos D1 R1 Wifi Processor With Uno : 7 Steps - Instructables](#)
- [5] [HC-SR501 PIR Motion Sensor Arduino Tutorial \(3 Examples\) \(makerguides.com\)](#)
- [6] [Installing ESP8266 in Arduino IDE \(Windows, Mac OS X, Linux\) | Random Nerd Tutorials](#)
- [7] [Motion Sensor Camera - ESP32-CAM project - Electronics Projects \(easyelectronicsproject.com\)](#)
- [8] [Capture & Send Images With ESP32-Cam Using ESP8266 WeMos D1 R1 Wifi Processor With Uno : 7 Steps - Instructables](#)
- [9] <https://reactnative.dev/docs/getting-started>

ANNEXES

[ANNEXE 1]

```
void configInitCamera(){
    camera_config_t config;
    config.ledc_channel = LEDC_CHANNEL_0;
    config.ledc_timer = LEDC_TIMER_0;
    config.pin_d0 = Y2_GPIO_NUM;
    config.pin_d1 = Y3_GPIO_NUM;
    config.pin_d2 = Y4_GPIO_NUM;
    config.pin_d3 = Y5_GPIO_NUM;
    config.pin_d4 = Y6_GPIO_NUM;
    config.pin_d5 = Y7_GPIO_NUM;
    config.pin_d6 = Y8_GPIO_NUM;
    config.pin_d7 = Y9_GPIO_NUM;
    config.pin_xclk = XCLK_GPIO_NUM;
    config.pin_pclk = PCLK_GPIO_NUM;
    config.pin_vsync = VSYNC_GPIO_NUM;
    config.pin_href = HREF_GPIO_NUM;
    config.pin_sscb_sda = SIOD_GPIO_NUM;
    config.pin_sscb_scl = SIOC_GPIO_NUM;
    config.pin_pwdn = PWDN_GPIO_NUM;
    config.pin_reset = RESET_GPIO_NUM;
    config.xclk_freq_hz = 20000000;
    config.pixel_format = PIXFORMAT_JPEG; /* ----- init with high specs to pre-allocate larger buffers. */
    if(psramFound()){
        config.frame_size = FRAMESIZE_UXGA; //--> FRAMESIZE_ + UXGA|SXGA|XGA|SVGA|VGA|CIF|QVGA|HQVGA|QQVGA
        config.jpeg_quality = 10;
        config.fb_count = 2;
    } else {
        config.frame_size = FRAMESIZE_SVGA;
        config.jpeg_quality = 12;
        config.fb_count = 1;
    }
} /* ----- camera init. */
```

[ANNEXE 2]

```
/* ::::::::::::::::::::: Taking a photo. */
camera_fb_t * fb = NULL;
fb = esp_camera_fb_get();
if(!fb) {
    Serial.println("Camera capture failed");
    Serial.println("Restart ESP32 Cam");
    delay(1000);
    ESP.restart();
    return "Camera capture failed";
}
/* ::::::::::::::::::::: */
```

[ANNEXE 3]

```
void initCard(){ // initialiser la carte SD
    Serial.println("Starting SD Card");
    delay(250);
    pinMode(13, INPUT_PULLUP); //--> This is done to resolve an "error" in 1-bit mode when SD_MMC.begin("/sdcard", true). Reference: https://github.com/esp8266/Arduino/issues/1030
    Serial.println("Start accessing SD Card 1-bit mode");
    if(SD_MMC.begin("/sdcard", true)){
        Serial.println("SD Card Mount Failed");
        return;
    }
    Serial.println("Started accessing SD Card 1-bit mode successfully");

    pinMode(13, INPUT_PULLDOWN);
    //*****
    //-----  

    //-----Checking SD card type
    uint8_t cardType = SD_MMC.cardType();
    if(cardType == CARD_NONE){
        Serial.println("No SD Card attached");
        return;
    }
    //-----  

    //-----Turning on the LED Flash on the ESP32 Cam Board
    pinMode(4, OUTPUT);
    digitalWrite(4, HIGH);
    delay(1000);
    //-----
```

[ANNEXE 4]

```
//voici le code de connexion à ta socket sevrer TCP,déjà inseré à la ligne 784
void connectToWemos(){
    if (client.connect(hostname, port)) // Création d'un socket
    {
        Serial.println("connection established");
    }
    else
    {
        Serial.println("Error connexion Sever");
    }
}
```

[ANNEXE 5]

```
void transfertImageToWemos(String path, camera_fb_t * fb ){

    // Construction du message JSON
    String header="";
    doc["action"] = "imageEsp";
    doc["filename"] = path.c_str();
    doc["size"] = fb->len;
    doc["id"] = pictureNumber;
    serializeJson(doc, header);
    header+="  
";
    client.println(header);

    // Envoyer les données binaires de l'image
    uint8_t *fbBuf = fb->buf;
    size_t fbLen = fb->len;

    for (size_t n=0;n<fbLen;n=n+1024) {
        if (n+1024<fbLen) {
            client.write(fbBuf, 1024);
            fbBuf += 1024;
        }
        else if (fbLen%1024>0) {
            size_t remainder = fbLen%1024;
            client.write(fbBuf, remainder);
        }
    }
}
```

[ANNEXE 6]

```
String sendPhotoTelegram() {
    const char* myDomain = "api.telegram.org";
    String getAll = "";
    String getBody = "";

    /* ----- The process of taking photos. */
    Serial.println("Taking a photo...");

    /* ::::::::::::::: Turns on LED FFlash if setting is "enable_capture_Photo_With_Flash(ON);". */
    if(capture_Photo_With_Flash_state() == ON) {
        LEDFlash_State(ON);
    }
    delay(1500);
    /* :::::::::::::: */

    /* ::::::::::::::: Taking a photo. */
    camera_fb_t * fb = NULL;
    fb = esp_camera_fb_get();
    if(!fb) {
        Serial.println("Camera capture failed");
        Serial.println("Restart ESP32 Cam");
        delay(1000);
        ESP.restart();
        return "Camera capture failed";
    }
    /* :::::::::::::: */

    /* :::::::::::::: Turn off the LED Flash after successfully taking photos. */
    if(capture_Photo_With_Flash_state() == ON) {
```

[ANNEXE 7]

```
/* ----- Subroutines to handle what to do after i
void handleNewMessages(int numNewMessages) {
    Serial.print("Handle New Messages: ");
    Serial.println(numNewMessages);

    /* ----- "For Loop" to check the contents of the newly received message. */
    for (int i = 0; i < numNewMessages; i++) {
        /* ::::::::::::::: Check ID (ID obtained from IDBot/@myidbot). */
        /*
         * If the chat_id is different from your chat ID (CHAT_ID), it means that someone (that is not you) has sent a message !
         * If that's the case, ignore the message and wait for the next message.
        */

        String chat_id = String(bot.messages[i].chat_id);
        if (chat_id != CHAT_ID){
            bot.sendMessage(chat_id, "Unauthorized user", "");
            Serial.println("Unauthorized user");
            Serial.println("-----");
            continue;
        }
        /* ::::::::::::::: */

        /* ::::::::::::::: Print the received message. */
        String text = bot.messages[i].text;
        Serial.println(text);
        /* ::::::::::::::: */

        /* ::::::::::::::: Check conditions based on commands sent from your telegram BOT. */
        // If it receives the "/start" message, we'll send the valid commands to control the ESP. This is useful if you happen !
        String send_feedback_message = "";
        String from_name = bot.messages[i].from_name;
        if (text == "/start") {
```

[ANNEXE 8]

```
const AllIntrusions = () => {
  const [isLoading, setIsLoading] = useState(false)
  const [reload, setReload] = useState(false)
  function deleteRequest(request, index) {
    setIsLoading(true)
    console.log("supprimer")
    console.log("voici la request:", request)
    const db = firebase.firestore();
    const collectionRef = db.collection("requetes");

    collectionRef.get().then((querySnapshot) => {
      querySnapshot.forEach((doc) => {
        console.log("doc.data().userId: ", doc.data().userId)
        console.log("auth().currentUser.uid", auth().currentUser.uid)
        console.log("doc.data().timestamp", doc.data().timestamp)
        console.log("request.timestamp", request.timestamp)

        if (doc.data().userId == auth().currentUser.uid &&
        doc.data().timestamp isEqual (request.timestamp)) {
          console.log("yo")
          doc.ref.delete().then(() => {
            setIsLoading(false)
            // Le document a été supprimé avec succès
            Alert.alert("Le document a été supprimé avec succès.");
            setReload(true)
            setReload(false)
          }).catch((error) => {
            // Une erreur s'est produite lors de la suppression du
            document
            setIsLoading(false)
            console.error("Erreur lors de la suppression du document
            : ", error);
          });
        }
      });
    });
  });
  setIsLoading(false)
};

const {height} = useWindowDimensions();
const navigation = useNavigation();
```

```

// const {signIn, signOut, signUp, user, setUser} =
useContext(AuthContext)

const [requests, setRequests] = useState([]);
const promises = []
const [imageUrls, setImageUrls] = useState([]);

useEffect(
() => {
  const db = firebase.firestore();
  const collectionRef = db.collection("requetes");
  collectionRef.get().then((querySnapshot) => {
    const requests = [];
    console.log("querysnapshot", querySnapshot)
    querySnapshot.forEach((doc) => {
      requests.push(doc.data());
    });
    console.log("avant :", requests)
    requests.sort((a, b) => b.timestamp.seconds - a.timestamp.seconds)
    console.log(requests)
    for(let i=0;i<requests.length;i++){
      const storageRef =
storage().ref(`images/${requests[i].userId}/${requests[i].url}.png`);
      promises.push(storageRef.getDownloadURL());
      Promise.all(promises)
      .then(urls => {
        setImageUrls(urls);
      })
      .catch(error => {
        console.error('Erreur lors de la récupération des URLs de téléchargement des images :', error);
      });
    }
    setRequests(requests);
  }).catch((error) => {
    console.error('Erreur lors de la récupération des documents :', error);
    setRequests([]);
  });
}

[, [reload]];
useEffect(() => {
  const interval = setInterval(() => {

```

```

        // Mettre à jour le state reload pour déclencher le
rechargement du composant
        setReload(true)
        setReload(false)
    , 10000); // Rafraîchir toutes les 10 secondes

        // Nettoyer l'intervalle lorsque le composant est démonté
        return () => clearInterval(interval);
    , []); // Utiliser une dépendance vide pour n'exécuter l'effet
qu'une seule fois
        return (
            <ScrollView style={styles.container}
showsVerticalScrollIndicator={true}>
            {isLoading && (
                <Modal visible={true} transparent animationType="fade">
                    <View style={styles.modalContainer}>
                        <View style={styles.modalContent}>
                            <Text>Chargement...</Text>
                        </View>
                    </View>
                </Modal>
            ) }
            {requests.map((request, index) => (
                <View key={index} style={styles.block}>
                    <Text style={styles.title}>Intrusion no {index + 1}</Text>
                    <Text
style={styles.description}>{request.description}</Text>
                    <Text style={styles.subtitle}>Date:
{moment((request.timestamp).toDate()).format('DD MMMM YYYY:
HH[h]mm')}</Text>
                    {imageUrls[index] ? (<Image source={{ uri:
imageUrls[index] }}>
                        style={[styles.logo, {height: height * 0.3}]}
                        resizeMode="contain"
                    />) : (<Text>Echec récupération image</Text>) }
                    <Pressable
                        style={{
                            padding: 10,
                            borderRadius: 5,
                            flexDirection: 'row',
                            alignItems: 'center',
                            justifyContent: 'center', // Ajout de cette ligne
                            opacity: 0.55,

```

```
        fontWeight:'bold',
    } }
    onPress={()=>{callPolice("117")}}
  >
  <FontAwesome5Icon name='phone' solid color='green'
size={24} />
  <Text style={{ color: 'black', marginLeft: 5 }}>Appeler
la police</Text>
  </Pressable>
  <TouchableOpacity onPress={() => deleteRequest(request,
index)}>
  <Text style={{ color: 'black', textAlign:'center'
}}>Supprimer</Text>
  </TouchableOpacity>
</View>
)) )
</ScrollView>
);
}
}
```

[ANNEXE 9]

```
const RecentIntrusion = () => {
  const [isLoading, setIsLoading] = useState(false)
  const [reload, setReload] = useState(false)
  function deleteRequest(request, index) {
    setIsLoading(true)
    console.log("supprimer")
    console.log("voici la request:", request)
    const db = firebase.firestore();
    const collectionRef = db.collection("requetes");

    collectionRef.get().then((querySnapshot) => {
      querySnapshot.forEach((doc) => {
        console.log("doc.data().userId: ", doc.data().userId)
        console.log("auth().currentUser.uid", auth().currentUser.uid)
        console.log("doc.data().timestamp", doc.data().timestamp)
        console.log("request.timestamp", request.timestamp)

        if (doc.data().userId == auth().currentUser.uid &&
        doc.data().timestamp isEqual (request.timestamp)) {
          console.log("yo")
          doc.ref.delete().then(() => {
            setIsLoading(false)
            // Le document a été supprimé avec succès
            Alert.alert("Le document a été supprimé avec succès.");
            setReload(true)
            setReload(false)
          }).catch((error) => {
            // Une erreur s'est produite lors de la suppression du
            document
            setIsLoading(false)
            console.error("Erreur lors de la suppression du document
            : ", error);
          });
        }
      });
    });
  });
  setIsLoading(false)
};

const {height} = useWindowDimensions();
const navigation = useNavigation();
```

```

// const {signIn, signOut, signUp, user, setUser} =
useContext(AuthContext)

const [requests, setRequests] = useState([]);
const promises = []
const [imageUrls, setImageUrls] = useState([]);

useEffect(
() => {
  const db = firebase.firestore();
  const collectionRef = db.collection("requetes");
  collectionRef.get().then((querySnapshot) => {
    const requests = [];
    console.log("querysnapshot", querySnapshot)
    querySnapshot.forEach((doc) => {
      requests.push(doc.data());
    });
    console.log("avant :", requests)
    requests.sort((a, b) => b.timestamp.seconds - a.timestamp.seconds)
    console.log(requests)
    for(let i=0;i<requests.length;i++){
      const storageRef =
storage().ref(`images/${requests[i].userId}/${requests[i].url}.png`);
      promises.push(storageRef.getDownloadURL());
      Promise.all(promises)
      .then(urls => {
        setImageUrls(urls);
      })
      .catch(error => {
        console.error('Erreur lors de la récupération des URLs de téléchargement des images :', error);
      });
    }
    setRequests(requests);
  }).catch((error) => {
    console.error('Erreur lors de la récupération des documents :', error);
    setRequests([]);
  });
}

[, [reload]];
useEffect(() => {
  const interval = setInterval(() => {

```

```

        // Mettre à jour le state reload pour déclencher le
rechargement du composant
        setReload(true)
        setReload(false)
    , 10000); // Rafraîchir toutes les 10 secondes

        // Nettoyer l'intervalle lorsque le composant est démonté
        return () => clearInterval(interval);
    , []); // Utiliser une dépendance vide pour n'exécuter l'effet
qu'une seule fois
        return (
            <ScrollView style={styles.container}
showsVerticalScrollIndicator={true}>
            {isLoading && (
                <Modal visible={true} transparent animationType="fade">
                    <View style={styles.modalContainer}>
                        <View style={styles.modalContent}>
                            <Text>Chargement...</Text>
                        </View>
                    </View>
                </Modal>
            ) }
            {requests.slice(0,1).map((request, index) => (
                <View key={index} style={styles.block}>
                    <Text style={styles.title}>Intrusion no {index + 1}</Text>
                    <Text
style={styles.description}>{request.description}</Text>
                    <Text style={styles.subtitle}>Date:
{moment((request.timestamp).toDate()).format('DD MMMM YYYY:
HH[h]mm')}</Text>
                    {imageUrls[index] ? (<Image source={{ uri:
imageUrls[index] }}>
                        style={[styles.logo, {height: height * 0.3}]}
                        resizeMode="contain"
                    />) :(<Text>Echec récupération image</Text>) }
                    <Pressable
                        style={{
                            padding: 10,
                            borderRadius: 5,
                            flexDirection: 'row',
                            alignItems: 'center',
                            justifyContent: 'center', // Ajout de cette ligne
                            opacity: 0.55,

```

```

        fontWeight:'bold',
    } }
    onPress={()=>{callPolice("117")}}
    >
    <FontAwesome5Icon name='phone' solid color='green'
size={24} />
    <Text style={{ color: 'black', marginLeft: 5 }}>Appeler
la police</Text>
    </Pressable>
    <TouchableOpacity onPress={() => deleteRequest(request,
index)}>
        <Text style={{ color: 'black', textAlign:'center'
}}>Supprimer</Text>
        </TouchableOpacity>
    </View>
)) )
</ScrollView>
);
}

```

[ANNEXE 10]

```

const autoConfig = () =>{
    setIsLoading(true)
    firebase.firestore().collection("configs").add({
        userId: auth().currentUser.uid,
        timestamp: firebase.firestore.FieldValue.serverTimestamp(),
        heureDebut: moment('01/01/2024 00:00:00', 'DD/MM/YYYY
HH:mm:ss').format('ddd MMM DD YYYY HH:mm:ss [GMT]ZZ'),
        heureFin: new Date(),
    })
    Alert.alert("Votre requête a été envoyée")
    setIsLoading(false);
}

```

[ANNEXE 11]

```
const applyChange = () =>{
    setIsLoading(true)
    console.log("hourBegin: ", selectedDateTimeBegin)
    console.log("hourEnd: ", selectedDateTimeEnd)
    firebase.firestore().collection("configs").add({
        userId: auth().currentUser.uid,
        timestamp: firebase.firestore.FieldValue.serverTimestamp(),
        heureDebut: selectedDateTimeBegin,
        heureFin: selectedDateTimeEnd,
    })
    Alert.alert("Votre requête a été envoyée")
    setIsLoading(false);
}
```