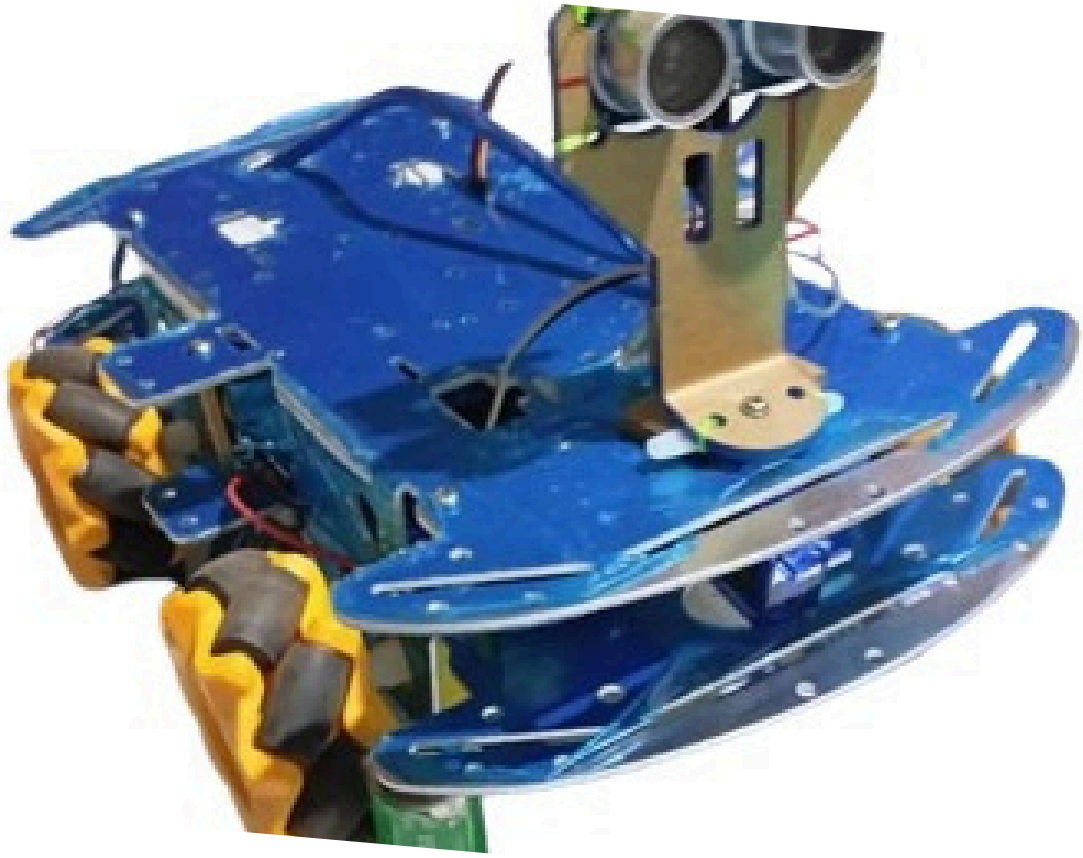
ECOLE NATIONALE SUPERIEURE POLYTECHNIQUE DE YAOUNDE
DEPARTEMENT DU GENIE INFORMATIQUE

UE: Electronique et Interfaçage

SIGHTEYES

The New Vision



RAPPORT MODÉLISATION

Sous la supervision de:

Dr. CHANA Anne Marie
Dr NGOUNOU Guy Merin

Année Académique **2024-2025**

ECOLE NATIONALE SUPERIEURE POLYTECHNIQUE DE YAOUNDE
DEPARTEMENT DU GENIE INFORMATIQUE

UE: Electronique et Interfaçage

MEMBRES DU GROUPE

- ATABONG Stéphane
- DJOUNKENG Eléonor (Chef)
- FOMEKONG Jonathan
- MBOCK Jean Daniem
- NGAH NDONGO Estelle (Vice-chef)
- NGHOGUE Franck
- NGO BASSOM Anne Rosalie
- NGOUPAYE DJIO Thierry
- NOMO Gabriel
- WANDJI Emmanuel

TABLE DES MATIÈRES

INTRODUCTION

Dans le cadre de notre projet pour l'unité d'enseignement Électronique et Interfaçage, nous proposons une solution innovante répondant à un enjeu actuel de notre société, nous l'avons nommé: **Sight-Eyes**, vise à développer un robot capable d'observer son environnement, faire l'analyse d'images et détecter un objet ayant un certain nombre de caractéristiques et de diriger vers cet objet.

Ce projet fait appel à l'utilisation de technologies électroniques, de computer vision et de programmation avancée pour répondre à des besoins concrets. Il s'inscrit dans une démarche d'innovation et de développement durable, en cherchant à améliorer notre quotidien tout en optimisant l'utilisation des ressources.

I. DESCRIPTION DU PROJET

1. Contexte du projet

Perdre des objets comme des lunettes, un livre, un pendentif ou une balle est une expérience courante pour tout le monde. Cependant, pour certaines personnes, notamment celles ayant des limitations physiques, comme les malvoyants ou les personnes âgées, la recherche de ces objets peut devenir une tâche difficile, voire impossible. C'est dans ce contexte que notre projet, **Sight Eyes**, vise à offrir une solution d'assistance accessible à tous, en particulier à ceux qui rencontrent des obstacles dans leurs capacités de recherche.

2. Problématique

Etant donné la difficulté que présentent certains individus à accomplir le travail de recherche, il devient de plus en plus nécessaire de leur apporter assistance au quotidien. Cependant, pour un Homme, apporter une aide quotidienne serait trop éprouvant. Ainsi, il est impératif de pouvoir automatiser cette assistance, en d'autres termes de créer un robot pour faire ce travail à la place de l'Homme. Dès lors, **comment concevoir un tel mécanisme ?**

3. Démarche et solution proposée

Ce projet d' IoT a pour objectif de **concevoir un robot destiné à assister les personnes âgées ou malvoyantes dans la recherche de leurs objets perdus**. Équipé d'une caméra pour la détection des objets, le robot est capable de se déplacer automatiquement vers l'objet recherché tout en évitant les obstacles. Il émet également un **signal sonore** pour alerter l'utilisateur. De plus, le projet intègre un système de communication via un **bot Telegram**, permettant ainsi de notifier les utilisateurs.

4. Modules et Composants utilisés

Notre projet se divise en plusieurs modules interconnectés entre eux à savoir :

1. Module Central (Raspberry Pi)

- Composant principal : Raspberry Pi (recommandé pour sa puissance).
- Rôle :
  - Centraliser le traitement des images.
  - Contrôler les moteurs.
  - Gérer la communication entre les différents modules.
- Communication :
  - Avec l'ESP 32-CAM pour recevoir le flux vidéo via Wi-Fi.
  - Avec les moteurs pour le contrôle des déplacements.
  - Avec le module de signalisation (buzzer/LEDs).

2. Module Caméra (ESP32-CAM)

- Composant principal : ESP32-CAM.
- Rôle : Capturer et envoyer le flux vidéo au Raspberry Pi pour analyse.
- Communication : Utilisation du Wi-Fi pour transmettre le flux vidéo vers le Raspberry Pi.
- Avantages : L'ESP 32-CAM offre une solution économique et sans fil pour capturer des images et les transmettre.

3. Module de Rotation de la Caméra

- Composants nécessaires : Servomoteurs pour le contrôle de la caméra sur deux axes (vertical et horizontal), Support de rotation pour la caméra.
- Rôle : Ajuster la direction de la caméra pour mieux cibler l'objet.
- Communication : Le Raspberry Pi envoie des commandes de position aux servomoteurs pour orienter la caméra.

4. Module de Mouvement (Moteurs)

- Composants nécessaires :

  - Moteurs DC pour le déplacement du robot.

  - Contrôleur de moteur L298N pour gérer les moteurs.

- Rôle: Permettre au robot de se déplacer vers l'objet détecté.
- Communication: Les moteurs sont contrôlés directement par le Raspberry Pi via les broches GPIO.

5. Module d'Arrêt et d'Évitement d'Obstacles

- Fonction :

  - S'arrêter devant l'objet recherché.

  - Détecter et éviter les obstacles (optionnel).

- Composants :

  - Capteurs à ultrasons pour la détection des obstacles.

  - Capteur de distance infrarouge (optionnel) pour une détection plus précise de la proximité de l'objet.

- Communication :
  - Le Raspberry Pi reçoit des informations sur la distance des obstacles et ajuste les mouvements du robot pour éviter les collisions et/ou s'arrêter.

6. Module de Signalisation et Alerte

- Composants nécessaires :

  - Buzzer pour l'alerte sonore.

  - LEDs pour les indications visuelles.

  - ESP32-CAM pour envoyer les signaux via Wi-Fi.

- Rôle : Signaler la détection de l'objet et émettre une alerte.

- Communication : Activé par le Raspberry Pi selon les résultats de la détection d'objets.

7. Module d'Alimentation

- Composants nécessaires :

  - Batterie LiPo pour alimenter le Raspberry Pi et les moteurs.

  - Régulateur de tension pour assurer une tension adéquate à chaque composant.

- Rôle : Assurer l'alimentation de tous les modules.
- Communication : La batterie distribue l'énergie de manière uniforme à chaque composant du robot.

8. Module de Développement Logiciel

- Fonction: Développer le bot Telegram pour la communication avec le robot.
- Communication : Le Raspberry Pi servira d'interface pour l'envoi des données à l'application web ou au bot Telegram.

9. Châssis du Robot

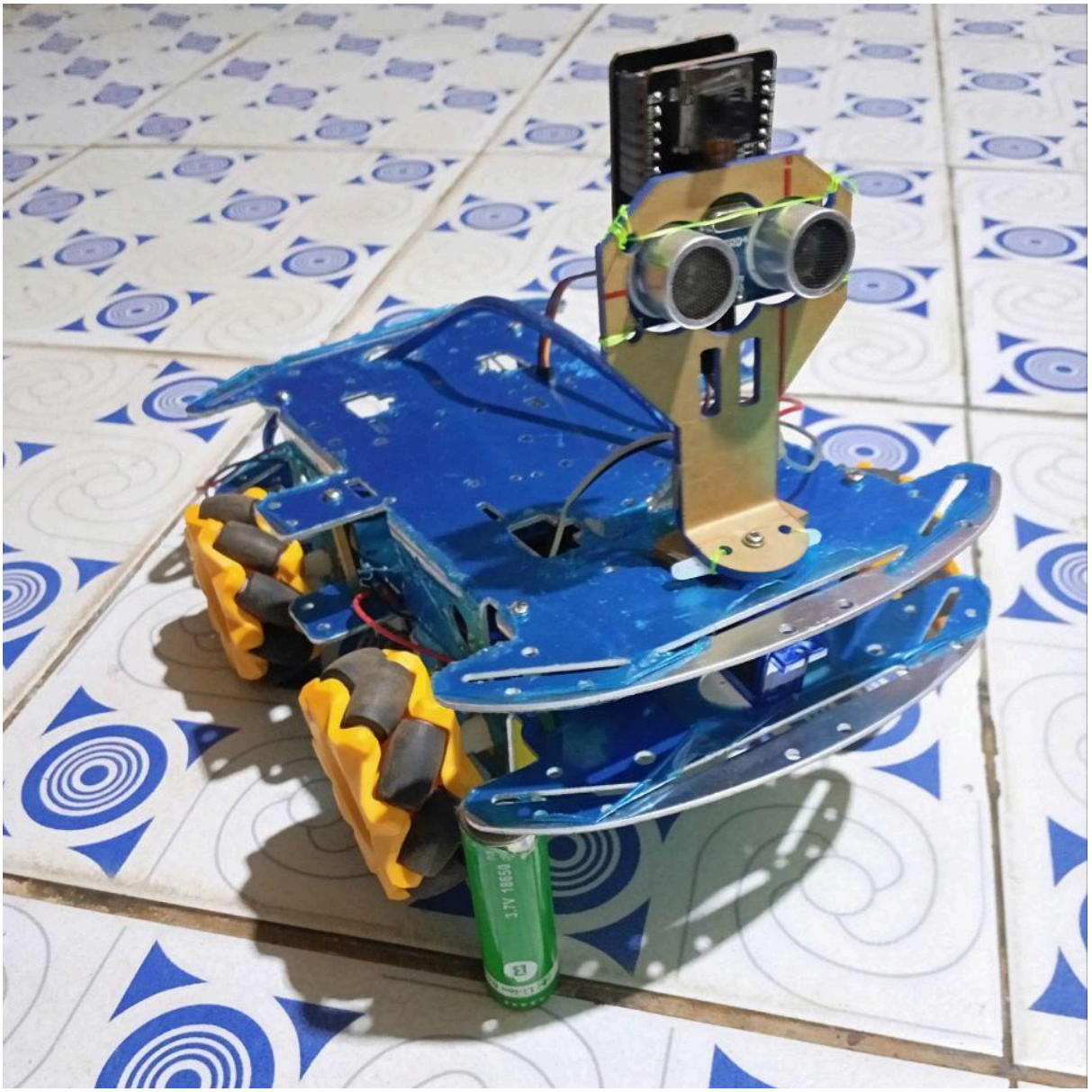
- Fonction: Supporter tous les composants et assurer la stabilité et la mobilité du robot.
- Composants : Châssis Keyestudio 4WD, intégrant les moteurs DC, offrant une plateforme pour fixer le Raspberry Pi, l'ESP 32-CAM, les capteurs, et les servomoteurs.

Voici la liste du matériel nécessaire pour la réalisation de ce projet :

- Kit Keyestudio 4WD BT Robot Car V2.0 pour Arduino.
- Kit Arduino Uno
- Raspberry Pi (module central pour le traitement).
- ESP32-CAM (ou une autre alternative pour la capture d'images).

5. Représentation du dispositif

Ceci est la maquette finale de notre projet .




-SR04
Trig
Echo
Gnd

III. **MODÉLISATION**

1.Diagrammes Structurels

1. Diagramme de Contexte

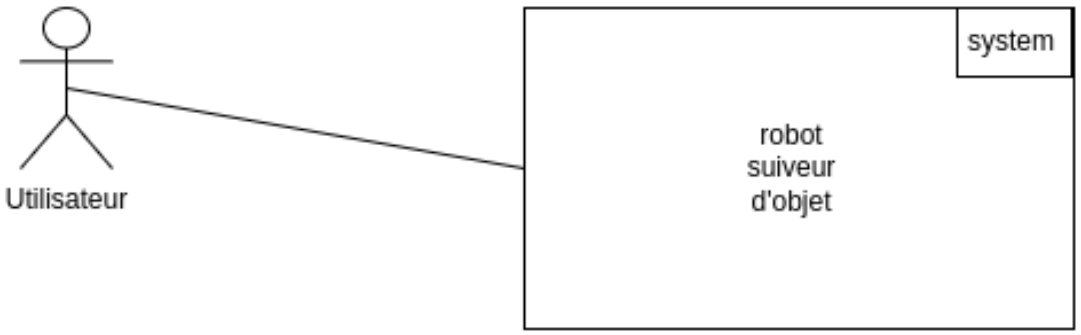
Le diagramme de contexte représente les **interactions** entre le **système sight eyes** et **les acteurs externes**, à savoir **l'utilisateur**.

-Description : L'utilisateur est la personne qui utilise le système pour retrouver des objets égarés.

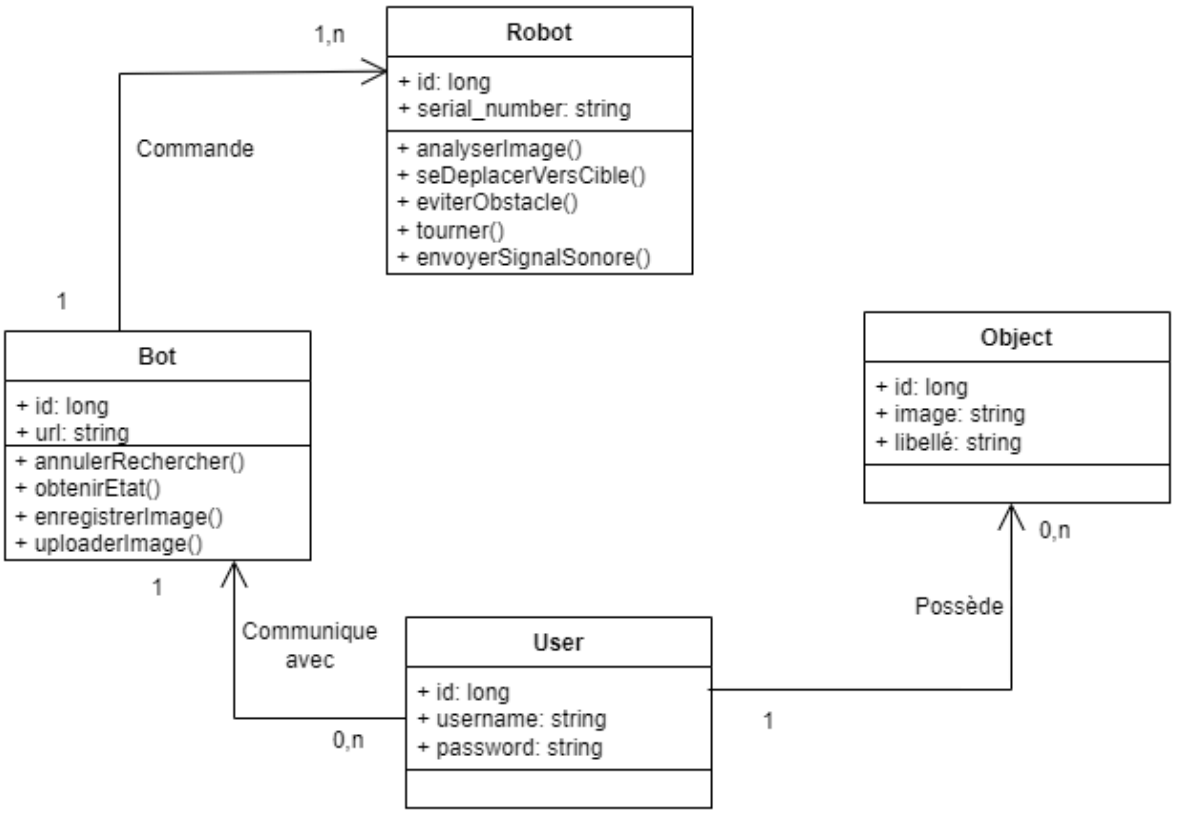
- Interactions :

- L'utilisateur envoie des commandes au système pour initier la recherche d'un objet.

- L'utilisateur reçoit des notifications lorsque l'objet est trouvé ou en mouvement



2. Diagramme de classes



Pour étudier en détail le robot Sight Eye, nous aurons besoin d'analyser les classes suivantes :

- **La classe Robot"** : Représente le robot dans son ensemble. Il contient des attributs pour les composants matériels (capteurs, caméras, moteurs,....), des attributs pour l'identifier(id,numéro de série,version) et des méthodes pour contrôler ses actions. Ses principales méthodes incluent :
  - **analyserImage():** Traite l'image capturée pour reconnaître l'objet cible.
  - **seDeplacerVersCible():** Ordonne aux moteurs de déplacer le robot vers l'objet.
  - **eviterObstacle():** Permet d'éviter des obstacles détectés durant le mouvement.
  - **tourner()**: Change la direction du robot en fonction de la position de l'objet**.**

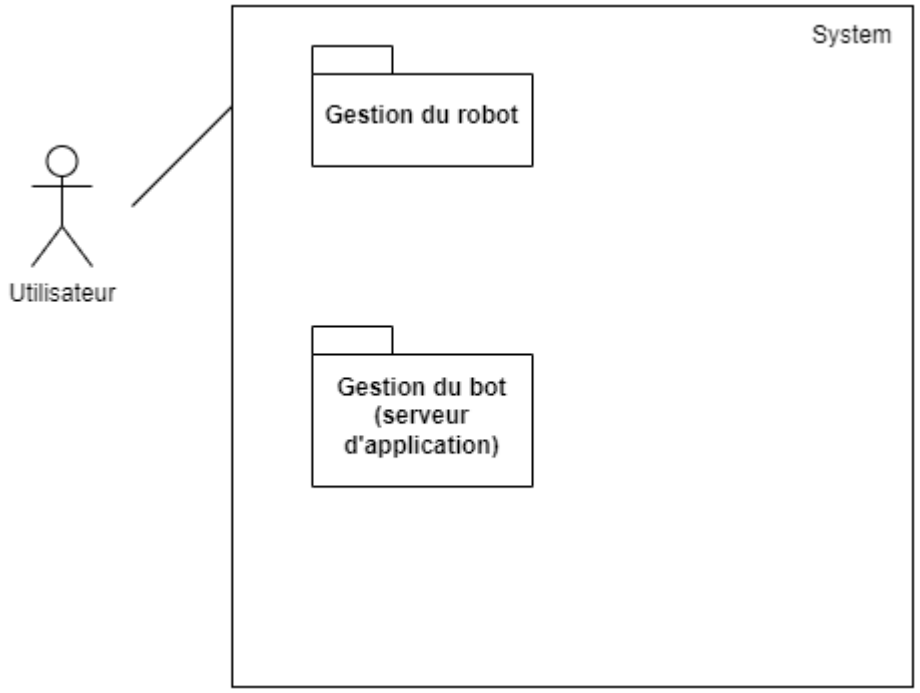
- **envoyerSignalSonore()**:permet au robot d'envoyer un signal déclenché par le user.

❖ **La classe "Object"** : Représente l'objet recherché par le robot. Attributs :
- position: La position estimée de l'objet dans l'espace.
- forme, couleur,image,numéro de série: caractéristiques de l'objet à reconnaître.

❖ **La classe "User ":** Celle-ci fait référence à l'utilisateur du robot. Il est caractérisé par un id,un username et un password.

❖ **La classe "Bot ":** Celle ci fait référence au Bot telegramm.C'est par l'intermédiaire de ce dernier que le user commande le robot.Il est caracterise par son id et son url et contient des méthodes telles que:
- **annulerRecherche():** permet d'annuler la recherche en cours.
- **obtenirEtat():** permet d'obtenir l'état actuel du robot et la renvoyer à l'utilisateur
- **declencherSignalSonore():** permet de déclencher un signal chez le rebot
- **enregistrerImage()**: permet au user d'enregistrer une image dans notre base de données
- **uploaderImage()**: permet à un utilisateur de fournir une image pour une recherche.

3. Diagramme de package



Nous avons décidé de découper les fonctionnalités de notre projet en deux grands packages: **gestion du robot** et **gestion du bot.**

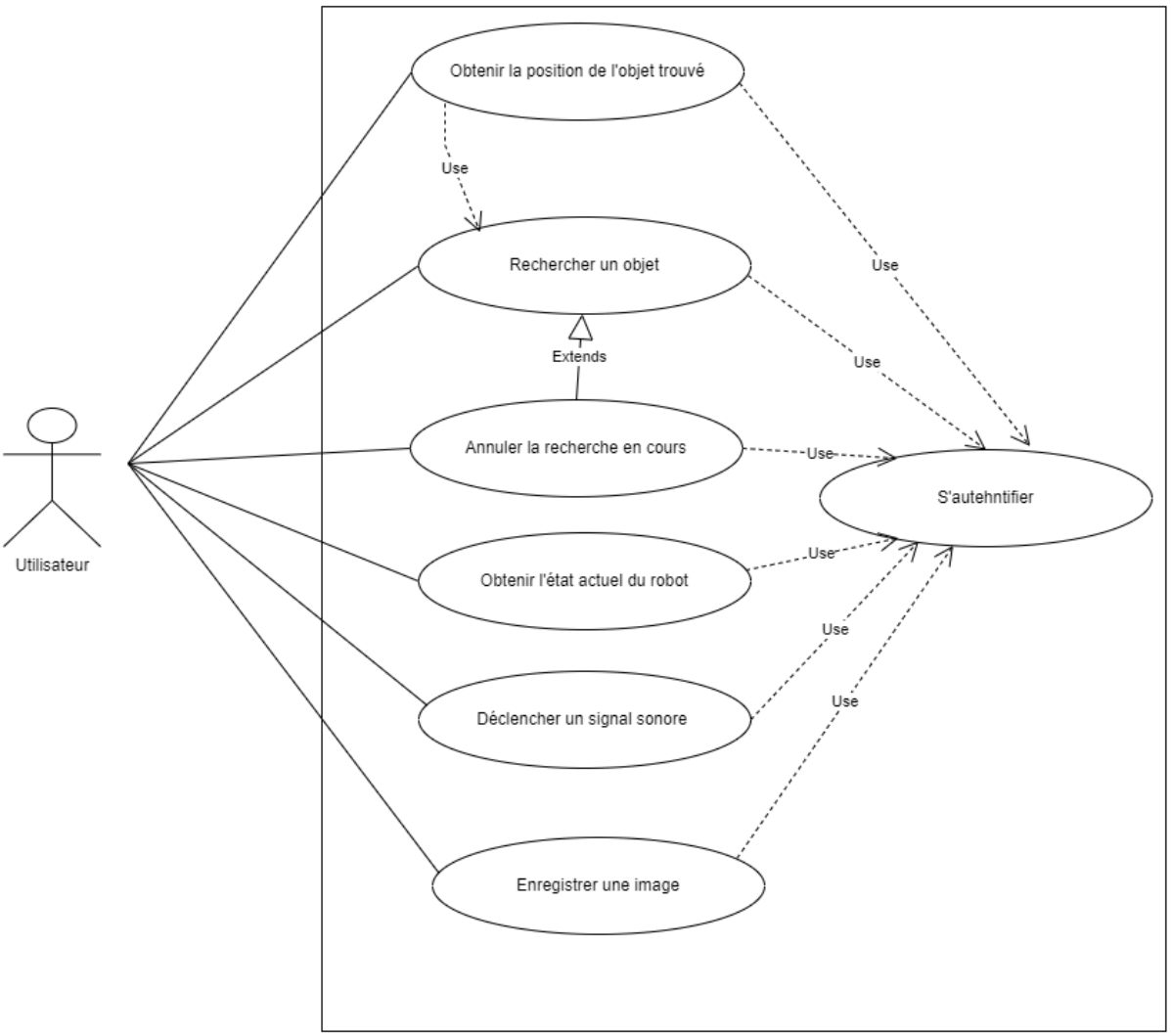
Gestion du robot:c'est ici que nous allons implémenter les fonctionnalités techniques qui concernent le robot et ses composants.

Gestion du bot: c'est ici que nous allons implémenter tout ce qu'il faut pour se connecter au bot telegramm et manipuler le robot à distance.

2.Diagrammes Comportementaux

1. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation présenté vise à modéliser les interactions entre les acteurs et le système du robot d'assistance. Ce diagramme permet d'identifier les différents scénarios d'interaction et de mettre en lumière les besoins des utilisateurs. En définissant les relations entre les acteurs et les cas d'utilisation, nous pouvons mieux comprendre les exigences fonctionnelles et les objectifs du projet.



- **Description Textuelle des cas d'utilisation**

1. Rechercher un objet

- **Objectif** : Permettre à l'utilisateur d'effectuer une recherche d'objet spécifique.
- **Acteurs** : Utilisateur.
- **Pré-conditions** : L'utilisateur est authentifié et le robot est opérationnel.
- **Post-conditions** : La recherche est active et le robot commence à rechercher l'objet.
- **Scénario Nominal** :

1. L'utilisateur sélectionne l'option "Rechercher un objet ".
2. Il saisit les caractéristiques de l'objet à rechercher.
3. Le robot commence à analyser l'environnement pour trouver l'objet.
4. Lorsque l'objet est retrouvé, un message de confirmation est envoyé à l'utilisateur.

- **Scénario Alternatif** :
    1. Si l'objet ne peut pas être identifié, le robot informe l'utilisateur et lui demande de spécifier d'autres caractéristiques.

2. Annuler la Recherche en Cours

- **Objectif** : Permettre à l'utilisateur d'annuler une recherche en cours.
- **Acteurs** : Utilisateur
- **Pré-conditions** : Une recherche est actuellement active.
- **Post-conditions** : La recherche est arrêtée et le robot revient à un état d'attente.
- **Scénario Nominal :**
    1. L'utilisateur sélectionne l'option "Annuler la recherche".
    2. Le robot cesse la recherche et se met en état d'attente
- **Scénario Alternatif :**
    3. Si aucune recherche n'est en cours, le robot informe l'utilisateur qu'il n'y a rien à annuler.

3. Obtenir la position de l'objet trouvé

- **Objectif** : Permettre à l'utilisateur de connaître la position de l'objet.
- **Acteurs** : Utilisateur
- **Pré-conditions** : Le robot est opérationnel et il a retrouvé l'objet.
- **Post-conditions** : L'utilisateur sait où se trouve son objet.
- **Scénario Nominal** :
    1. L'utilisateur demande la position de l'objet trouvé.
    2. Le robot envoie ses coordonnées à l'utilisateur.
- **Scénario Alternatif** :
    1. S'il ya un problème de partage de position , un message d'erreur est renvoyé à l'utilisateur.

4. **Obtenir l'État Actuel du Robot**

- **Objectif** : Fournir à l'utilisateur l'état actuel du robot.
- **Acteurs** : Utilisateur
- **Pré-conditions** : Le robot est en fonctionnement.
- **Post-conditions** : L'utilisateur reçoit des informations sur l'état du robot.
- **Scénario Nominal** :
    1. L'utilisateur demande l'état actuel du robot.
    2. Le robot vérifie son statut (en recherche, inactif, erreur, etc.).
    3. L'état est envoyé à l'utilisateur.
- **Scénario Alternatif :**
    1. Si le robot ne peut pas déterminer correctement sa position, il informe l'utilisateur de cette erreur et suggère de relancer la commande de localisation

5. **Déclencher un Signal Sonore**

- **Objectif** : Permettre au robot d'émettre un signal sonore pour attirer l'attention de l'utilisateur.
- **Acteurs** : Utilisateur
- **Pré-conditions** : Le robot est opérationnel.
- **Post-conditions** : Un signal sonore est émis par le robot.
- **Scénario Nominal :**
    1. L'utilisateur sélectionne l'option pour déclencher un signal sonore.
    2. Le robot émet un signal sonore.
    3. L'utilisateur est informé que le signal a été déclenché.
- **Scénario Alternatif :**
    1. Si le robot rencontre une erreur lors de l'émission du signal, il informe l'utilisateur.

6. **Enregistrer une image**

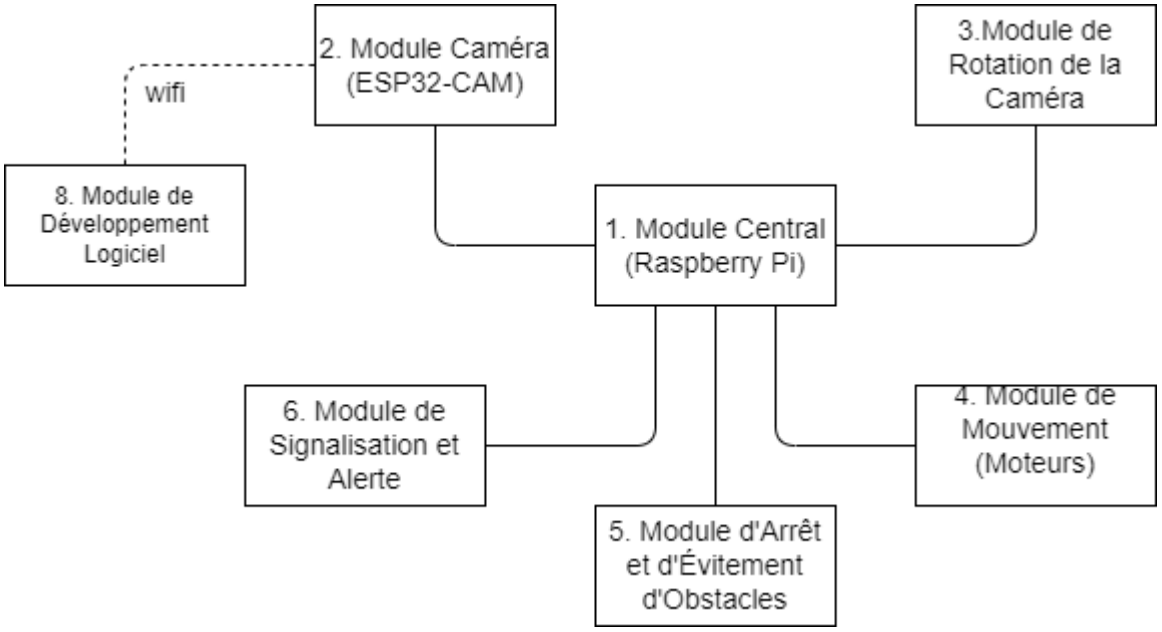
- **Objectif** : Permettre à l'utilisateur de pré-enregistrer les images de ses objets afin de les chercher ultérieurement en cas de perte.
- **Acteurs** : Utilisateur
- **Pré-conditions** : L'utilisateur est connecté.
- **Post-conditions** : Les images de l'utilisateur sont enregistrées.
- **Scénario Nominal :**
    1. L'utilisateur sélectionne l'option pour enregistrer une image.
    2. Il upload ses images et sélectionne l'option 'Enregistrer'.
- **Scénario Alternatif :**
    1. Si le système rencontre une erreur lors de l'enregistrement, il informe l'utilisateur.

7. S'authentifier

- **Objectif** : Permettre à l'utilisateur d'accéder à l'interface du système afin de pouvoir utiliser ses fonctionnalités.
- **Acteurs** : Utilisateur
- **Pré-conditions** : L'utilisateur possède un compte valide .
- **Post-conditions** :L'utilisateur est authentifié et a accès aux fonctionnalités autorisées du robot.
- **Scénario Nominal :**
    1. L'utilisateur accède à l'interface de connexion.
    2. Il entre ses informations de connexion.
    3. Le système d'authentification vérifie les informations fournies.
    4. Si les informations sont correctes, l'utilisateur est authentifié et redirigé vers l'interface principale.
- **Scénario Alternatif :**
    1. Si le système rencontre une erreur lors de l'authentification (informations incorrectes,compte verrouillé, système indisponible), l'utilisateur est informé.

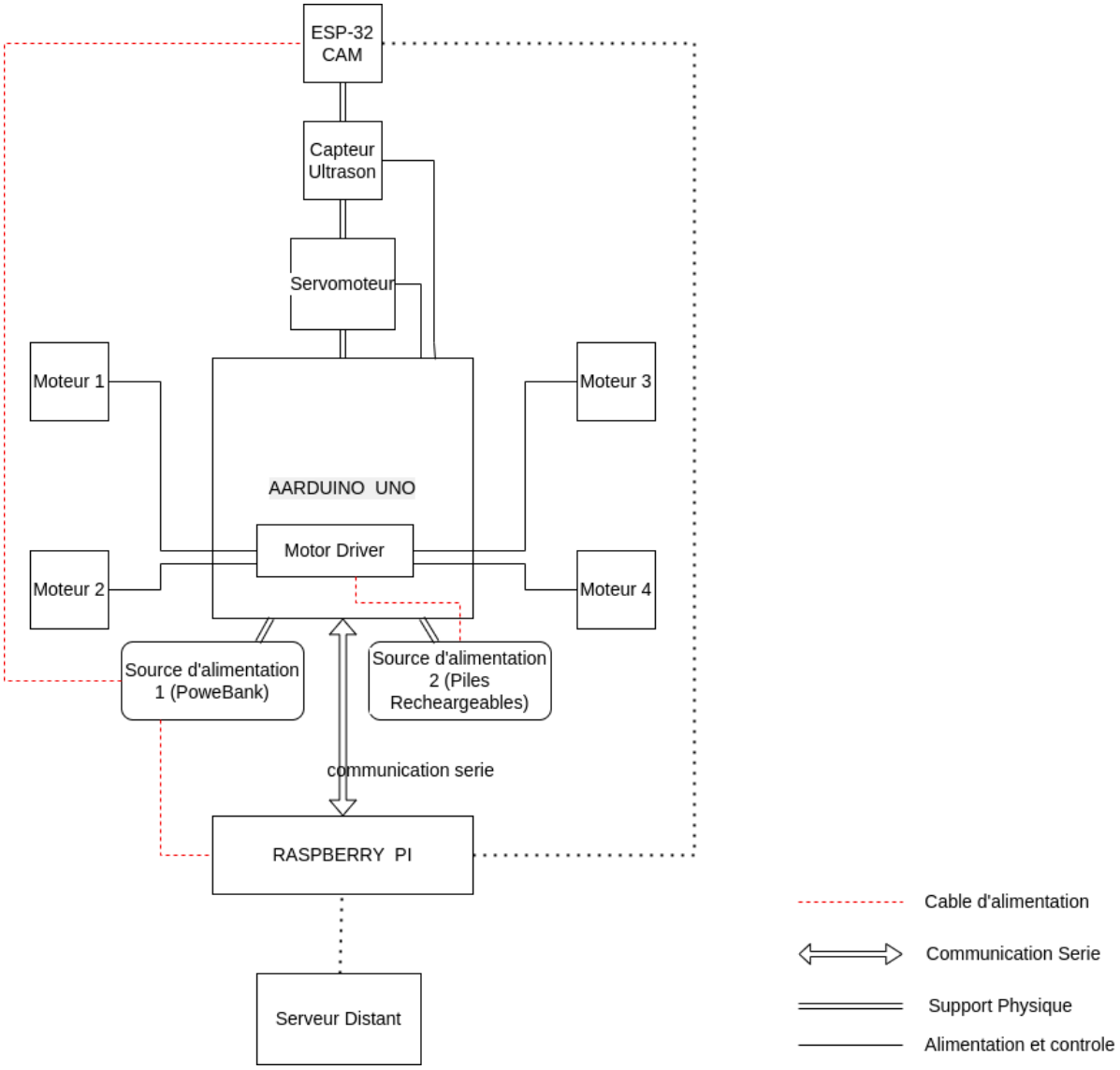
2. Illustration de l'interaction entre les modules

Ce diagramme nous permet de visualiser les differents modules de notre projet comme des boites noir liees entre elles en fonction de si elles communiquent directement ou non.



IV. ARCHITECTURE MATÉRIELLE ET LOGICIEL

1.Architecture matérielle



Ce schéma représente l'architecture matérielle de motre robot motorise basé sur le kit Keyestudio Mecanum, organisé autour de plusieurs composants électroniques interconnectés pour assurer la navigation et le contrôle. Voici une description textuelle détaillée

ESP32-CAM : Placé au sommet du schéma, ce module est équipé d'une caméra permettant la capture vidéo pour la reconnaissance visuelle. Il est connecté à un capteur à ultrasons, qui détecte les obstacles ou mesure des distances.

Servomoteur : Le capteur à ultrasons est monté sur un servomoteur, permettant à ce dernier de pivoter pour couvrir une zone plus large dans l'environnement.

Moteurs : Quatre moteurs (Moteurs 1 à 4) équipés de roues Mecanum sont répartis pour le déplacement du robot. Ils sont reliés à un module de contrôle des moteurs (Motor Driver), qui reçoit des instructions pour piloter leur mouvement.

Arduino Uno : Situé au centre du schéma, l'Arduino Uno sert de microcontrôleur principal, gérant la communication entre les différents composants matériels (moteurs, capteurs, etc.) et assurant le traitement des commandes.

Sources d'alimentation :

Source d'alimentation 1 : Un Power Bank fournit de l'énergie à l'Arduino et à certains composants électroniques.

Source d'alimentation 2 : Des piles rechargeables alimentent principalement les moteurs via le Motor Driver.

Raspberry Pi : Connecté à l'Arduino Uno via une communication série, le Raspberry Pi joue un rôle de supervision, permettant les traitements avancés et les échanges avec un serveur distant.

Serveur distant : Représenté en bas, ce serveur est utilisé pour des tâches telles que le stockage des données collectées par le robot ou le contrôle à distance via des commandes.

Connexions :

- Câbles d'alimentation : Représentés en pointillés rouges, ils transportent l'énergie entre les différents composants.
- Communication série : Symbolisée par des flèches à double sens, elle assure l'échange de données entre l'Arduino et le Raspberry Pi.
- Supports physiques : Les lignes noires unies représentent les liaisons matérielles directes entre les composants.
- Alimentation et contrôle : Les lignes épaisses montrent les connexions utilisées pour transmettre l'énergie et les commandes aux moteurs.

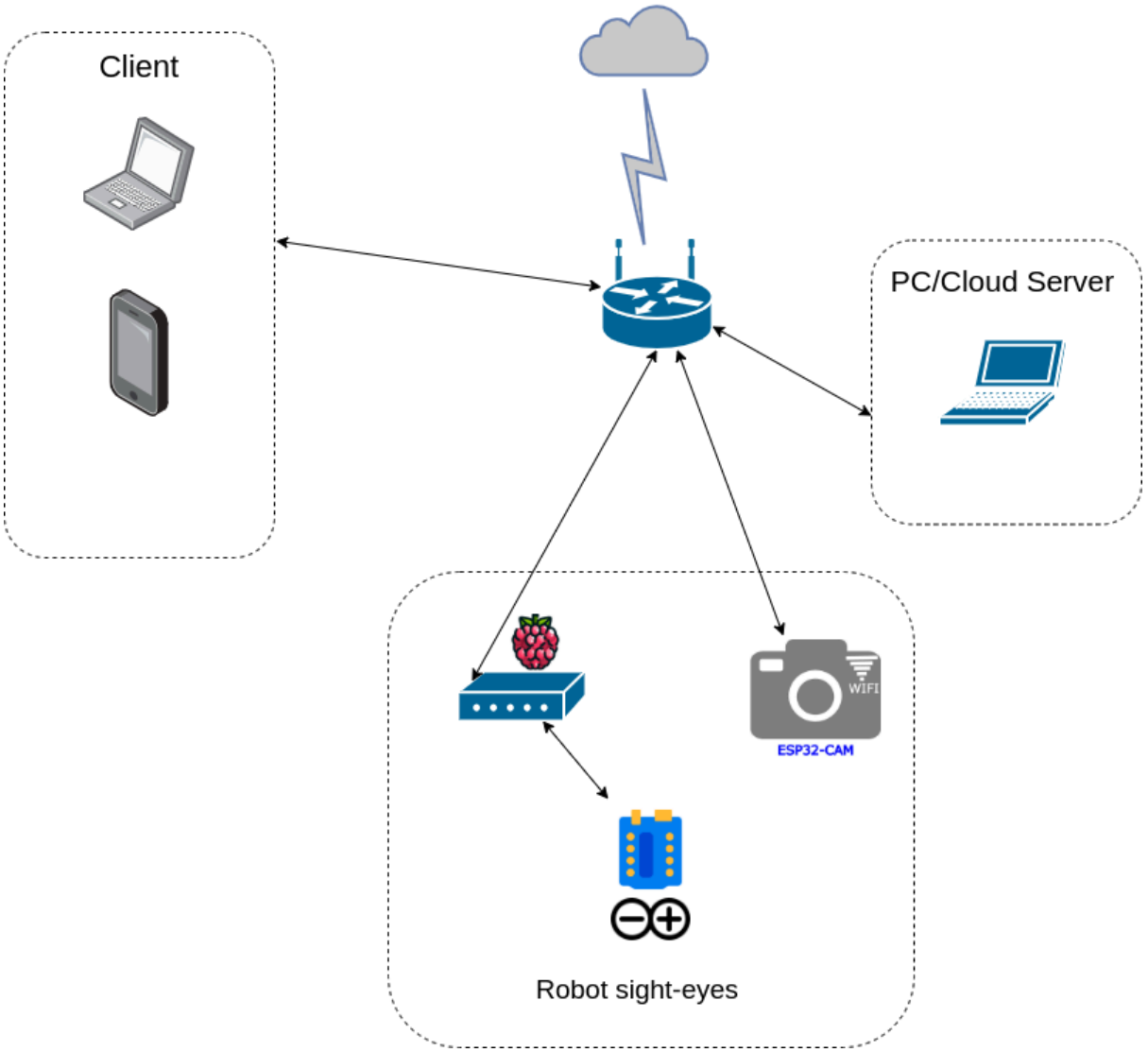
Cette configuration est typique d'un robot mobile capable d'effectuer des tâches telles que la navigation autonome, l'évitement d'obstacles et la reconnaissance d'objets.

2.Architecture Logiciel

L'architecture qui suit n'est pas celle planifiée dès le départ, elle est le résultat final d'ajustement, de résolution de problèmes et d'optimisation de communication entre les composants.

A. Schéma IOT

Commençons par une présentation de la topologie réseau des éléments.



Ce schéma est composé de plusieurs entités interconnectées, permettant la capture, le traitement et la transmission de données en quasi temps réel.

Composants et interconnexions

L'architecture repose sur un routeur central qui joue un rôle clé dans la communication entre les différentes entités. Ce routeur connecte les composants du système, y compris un serveur distant (PC/Cloud Server), un module de vision embarqué sur un robot (ESP32-CAM), et un micro ordinateur (Raspberry Pi) qui communique via une communication port série avec notre arduino uno.

Client et interaction utilisateur

Un client, représenté par un ordinateur portable et un smartphone, se connecte à internet via un routeur et ce dernier lui permet d'avoir accès au réseau local. À l'état actuel du prototype il faut que tous les composants y compris le serveur qui exécute le modèle doivent être dans le même réseau. Ce client peut envoyer des commandes à travers un bot télégram à notre système.

Module robotique de vision

Le robot est équipé de plusieurs composants électroniques pour l'acquisition de l'image, le traitement des commandes. L'ESP32-CAM, un module caméra Wi-Fi, capture des images et les transmet via le réseau au serveur qui exécute le model. Ce module est en communication avec le routeur, qui relaie les données vers un serveur distant ou un client.

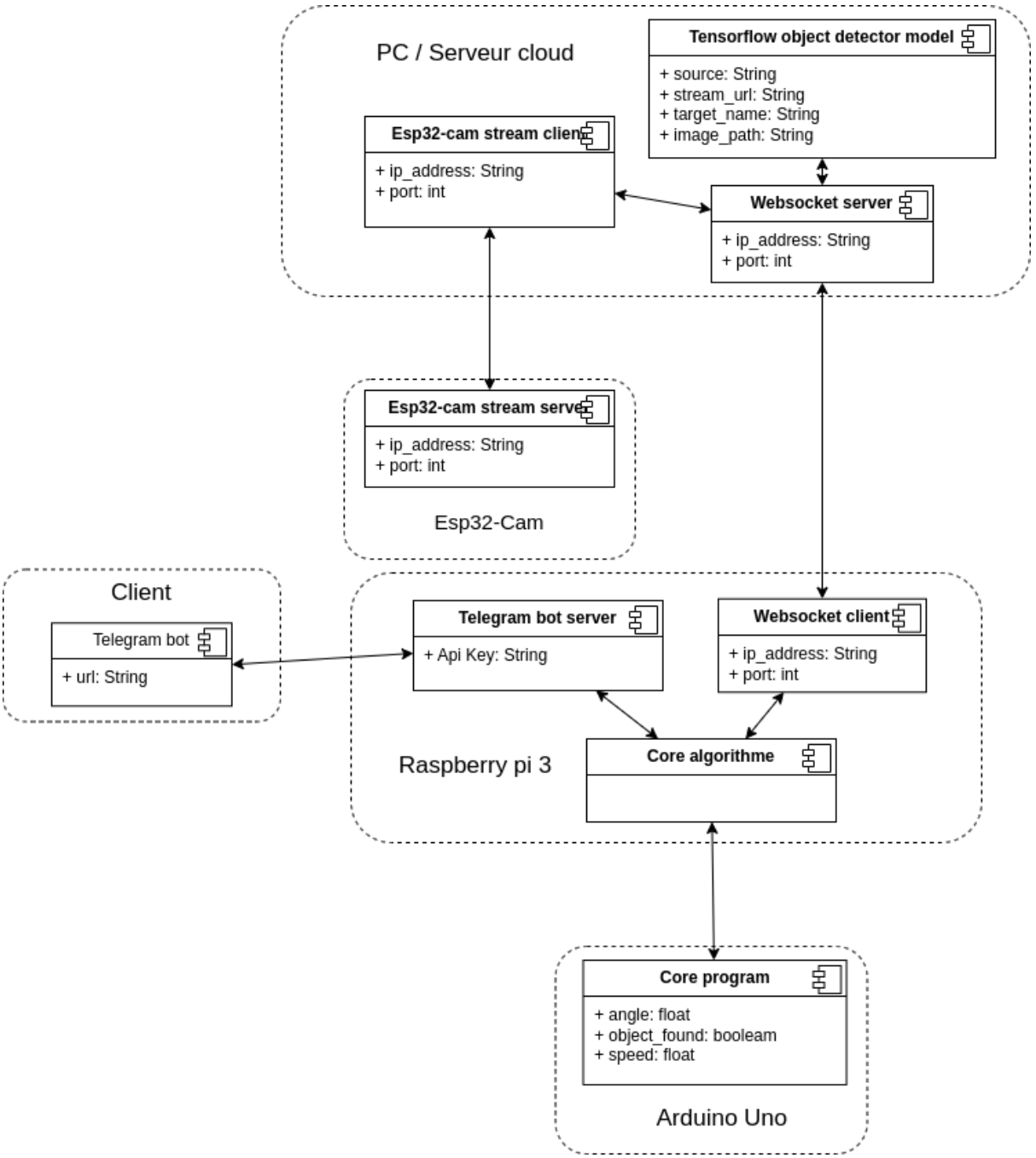
Traitement et contrôle embarqué

Un Raspberry Pi est également présent dans le système et se connecte à un switch réseau, assurant la gestion des communications internes au robot. Il peut traiter les données issues de la caméra, exécuter des algorithmes de reconnaissance d'images et transmettre les résultats vers un serveur distant ou un client.

Serveur distant et traitement avancé

Un PC ou un serveur cloud est connecté au réseau pour stocker, analyser et traiter les données capturées par le robot. Il peut héberger des algorithmes avancés de vision par ordinateur, permettant une analyse plus approfondie des images et la prise de décision en fonction des résultats.

B. Diagramme de composant



PC / Serveur Cloud

Le serveur distant (qui peut être remplacé par un serveur cloud) joue un rôle central dans l'analyse et le traitement des flux vidéo envoyés par l'ESP32-CAM. Il contient plusieurs composants :

- **Esp32-cam stream client** : Ce composant reçoit le flux vidéo depuis l'ESP32-CAM via un serveur de streaming et le transmet à d'autres services pour traitement.

- **Websocket server** : Il établit une communication bidirectionnelle en temps réel avec d'autres composants via le protocole WebSocket. Il est utilisé pour envoyer des informations au Raspberry Pi 3.
- **Tensorflow object detector model** : Ce modèle basé sur TensorFlow traite les flux vidéo entrants, détecte les objets spécifiques et extrait les informations pertinentes (source vidéo, URL du flux, nom de la cible et chemin de l'image capturée).

ESP32-CAM

L'ESP32-CAM est un module caméra connecté qui capture et diffuse un flux vidéo via un serveur de streaming. Il comprend :

- **Esp32-cam stream server** : Ce composant sert le flux vidéo en continu sur un port spécifique et une adresse IP définie, permettant à d'autres entités du système d'y accéder.

Raspberry Pi 3

Le Raspberry Pi 3 agit comme un centre de contrôle du système, en recevant des informations du serveur cloud et en communiquant avec l'Arduino Uno pour piloter des actions. Il intègre plusieurs composants :

- **Telegram bot server** : Il interagit avec un client Telegram et reçoit les requêtes des utilisateurs via une clé API. Il permet l'envoi d'alertes et d'informations sur l'état du robot.
- **Websocket client** : Ce composant établit une connexion avec le serveur WebSocket hébergé sur le serveur cloud pour recevoir en temps réel les données de détection d'objets.
- **Core algorithm** : C'est l'algorithme principal du Raspberry Pi qui analyse les données reçues (par WebSocket ou Telegram), prend des décisions et envoie des commandes à l'Arduino Uno pour ajuster les actions du système.

Arduino Uno

L'Arduino Uno est responsable du contrôle physique du robot en fonction des données envoyées par le Raspberry Pi. Il héberge le **Core program**, qui gère :

- **L'angle de déplacement** (variable flottante).
- **La détection d'objets** (booléen indiquant si un objet a été détecté).
- **La vitesse du mouvement** (variable flottante représentant la vitesse de déplacement du robot).

Client (Bot Telegram)

Un client distant peut interagir avec le système via un bot Telegram, qui envoie et reçoit des informations grâce à :

- **Telegram bot** : Il utilise une URL spécifique pour transmettre les requêtes de l'utilisateur au serveur Telegram sur le Raspberry Pi.

3.Technologie utilisés

Protocoles de Communication

Pour permettre l'échange de données entre les différents composants, plusieurs protocoles sont utilisés :

- **WebSocket** : Ce protocole assure une communication bidirectionnelle en temps réel entre le serveur cloud, le Raspberry Pi 3 et d'autres composants. Il permet une transmission rapide et efficace des données issues de la caméra et du modèle de détection d'objets.
- **HTTP et API REST** : Utilisé notamment par le bot Telegram, il permet aux utilisateurs d'envoyer des requêtes et de recevoir des informations en réponse.
- **Streaming vidéo** : L'ESP32-CAM utilise un serveur de streaming pour diffuser son flux vidéo en temps réel au serveur cloud.

Intelligence Artificielle et Traitement d'Images

Le système repose sur l'analyse des flux vidéo en provenance de l'ESP32-CAM grâce à l'intelligence artificielle :

- **TensorFlow et OpenCV** : Le serveur cloud exécute un modèle de détection d'objets basé sur TensorFlow, qui traite les images envoyées par l'ESP32-CAM pour identifier les objets pertinents. OpenCV peut être utilisé pour le prétraitement des images avant analyse.
- **Python** : Langage principal du modèle d'intelligence artificielle et des scripts d'analyse.

Plateformes et Frameworks Embarqués

Les différents composants matériels du projet sont pilotés par des systèmes embarqués :

- **Arduino IDE** : Utilisé pour programmer l'**Arduino Uno**, qui exécute le programme de contrôle moteur en fonction des données reçues.

- **Micropython / ESP-IDF** : L'ESP32-CAM peut être programmé avec MicroPython ou l'ESP-IDF (Espressif IoT Development Framework), permettant de gérer la capture et la diffusion du flux vidéo.
- **Raspbian (Raspberry Pi OS)** : Le Raspberry Pi 3 fonctionne sous un système d'exploitation basé sur Linux, lui permettant d'exécuter les algorithmes de traitement et d'interagir avec le serveur cloud.

Automatisation et Services en Ligne

- **Telegram Bot API** : Fournit l'interface entre le Raspberry Pi et les utilisateurs via un bot Telegram. Il permet d'envoyer des commandes et de recevoir des notifications en temps réel.

V. ORGANISATION DU GROUPE

1.Postes et Responsabilités des membres du groupe

Nous avons adopté une approche structurée afin d'optimiser le travail au sein du groupe. Chaque membre a été assigné à une tâche spécifique pour garantir le bon déroulement du projet. À cet effet, nous avons mis en place cinq cellules : la cellule d'administration, le secrétariat, la cellule financière, la cellule de suivi (scrum masters), et la cellule de communication. Ainsi nous avons:

- **Cellule Administration** : Elle se charge de superviser le projet dans sa globalité et contrôler les tâches des différentes cellules. Elle organise les rencontres avec les enseignants et organise les réunions (ordre du jour, salle de travail et répartition et planification des tâches) . Comme responsables nous avons:
    - Chef projet : **DJOUNKENG NGUEFACK ELEONOR**
    - Sous-chef projet: **NGAH NDONGO ESTELLE CLOTILDE**

- **Secrétariat**: Il se charge de faire les rapport des différentes séances de travail et des réunions, garde sur écrits les remarques faites par les enseignants et organise la structure des différents rapports . Comme responsables nous avons:
    - Chef: **NGO BASSOM ANNE ROSALIE**
    - Adjoint 1: **ATABONG STEPHANE**
    - Adjoint 2: **MBOCK JEAN-DANIEL**

- **Cellule Suivi (Scrum master)**: Elle est chargée de planifier les tâches sur Jira , assurer le suivi et l'évolution de ces tâches auprès de chaque membre. Comme responsables nous avons:
    - Chef: **NOMO BODIANGA NASAIRE GABRIEL**
    - Adjoint 1: **FOMEKONG JONATHAN**

- **Cellule financière**: Elle est chargée de collecter l'argent pour l'achat ou la location du matériel et l'impression des différents rapports . Elle est

également chargée d'acheter ledit matériel. Comme responsables nous avons:

- Chef: **NGOUPAYE DJIO THIERRY**
- Adjoint 1: **WANDJI EMMANUEL**

- **Cellule communication**: Elle est chargée de véhiculer chaque information importante à tous les membres et s'assurer que tout le monde a la bonne information. Comme responsables nous avons:
  - Chef: **NGHOGUE TAPTUE FRANCK RODDIER**

2. Métrique d'évaluation

Nous avons conçu une métrique d'évaluation afin d'évaluer équitablement tous les membres du groupe, et de rendre chacun conscient de son niveau de participation. Les critères d'évaluation sont définis comme suit :

- La Discipline
- La participation active
- L'exécution des tâches

En ce qui concerne la discipline, elle inclut la gestion des présences et est notée comme suit:

- **Présence** : 1
- **Absence justifiée** : 0,75
- **Absence non justifiée** : 0
- **Retard (moins de 30 minutes)** : 0,75
- **Retard (plus de 30 minutes)** : 0,5

NB : À la fin du projet, chaque membre procédera à une évaluation mutuelle du travail des autres, laquelle sera prise en compte dans la note finale de participation.

Le pourcentage individuel sera calculé selon la répartition suivante : **30% pour la discipline** (présence, absence, et ponctualité), **10% pour la participation active**, et **60% pour l'exécution des tâches assignées**. Afin de garantir le suivi rigoureux de ce système d'évaluation, un **Google Sheet** a été mis en place et sera régulièrement mis à jour après chaque réunion

3. Outils collaboratifs pour le travail

Pour travailler de manière efficace, nous avons opté pour les outils suivants :

- **GitHub** : Dépôt pour le code source et les rapports, voici le lien: https://github.com/JonaBacho/sigh-eyes.git

- **Jira** : Outil de planification et de suivi des tâches, voici le lien: https://jonathanbachelard.atlassian.net/jira/software/projects/SCRUM/boards/1/backlog

- **Google Docs** : Rédaction collaborative du rapport et des livrables, voici le lien : https://docs.google.com/document/d/1f8tUvwfSryX23fXpkzcJhEFco83-RClL7QG0bTTjWdU/edit?usp=sharing

- **WhatsApp** : canal principal de communication pour les échanges rapides.

- **Draw.io (en ligne)** : Outil de modélisation pour la création de schémas et diagrammes, voici le lien: https://app.diagrams.net/#G1tlwq4OAcsHAh2fwqGPfyuMGhELoRUPS1#%7B%22pageId%22%3A%22rs8-hcHXhh-nhzU9TEZu%22%7D

De plus, nous avons mis en place une **métrique d'évaluation** basée sur la **discipline** et la **réalisation des tâches** pour suivre la contribution de chaque membre.

4. Calendrier de travail

Afin de travailler efficacement et permettre un suivi continu, nous avons mis sur pied ce calendrier de réalisation du projet décomposé en 5 phases et que nous avons estimé à 9 semaines pour la réalisation de notre projet

N°	Phase	Période
1	Modélisation du projet et prise en main de Arduino	14-27 octobre (2 semaines)
2	Production de la maquette 3D et identification du matériel nécessaire	28 octobre - 03 novembre (1 semaine)
3	Apprentissage et prise en main des différents modules	4 - 17 novembre (2 semaines)
4	Regroupement des modules et implémentation	18 novembre - 08 décembre (3 semaines)
5	Tests et corrections	09 décembre - 22 décembre (1 semaine)

Pour chaque semaine nous avons deux rencontres de travail:

- Mercredi: Entre 15h45 à 17h45
- Samedi: Entre 08h et 12h, extensible en fonction des objectifs à atteindre

CONCLUSION

Parvenus au terme de ce rapport, nous avons pu clairement présenter le contexte et le problème que nous souhaitons résoudre. Par la suite, nous avons présenté la solution que nous proposons qui est directement en lien avec l'électronique et l'Internet des Objets, nous avons présenté la démarche à suivre pour la réalisation dudit projet, les modules, les composants et le matériel nécessaire pour sa mise en œuvre. Après cela nous avons fait une modélisation globale du projet.

Enfin, étant donné que c'est un projet de groupe de dix membres, une organisation structurée et bien répartie a été mise en œuvre et une métrique d'évaluation que nous avons aussi présenté. Nous pensons qu'avec le calendrier mis sur pied et la discipline, notre travail acharné va concourir à la bonne réalisation de notre projet.

Références

- **Documentation officielle du Raspberry Pi:** **https://www.raspberrypi.com/documentation/computers/configuration.html**
- **Dépot github des modèles Tensorflow et leurs Documentations**

  https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

- **Communication esp32-cam et raspberry pi (ce lien ne passe pas au cameroun, nécessite d'entrer dans un VPN)**

  https://gpiocc.github.io/learn/raspberrypi/esp/ml/2020/11/08/martin-ku-stream-video-from-esp32-to-raspberry-pi.html

- **Documentation officielle de l'arduino**

  https://www.arduino.cc/

- **Documentation pour le kit Key eyes Studio:**

  https://www.keyestudio.com/products/keyestudio-4wd-mecanum-robot-car-for-arduino-stem-smart-diy-robot-car-kit

- **Communication raspberry-pi et Arduino**

  https://forum.arduino.cc/t/arduino-and-raspberry-pi-serial-communication/1161375

- **Site pour la Conception 3D :**

  https://www.tinkercad.com