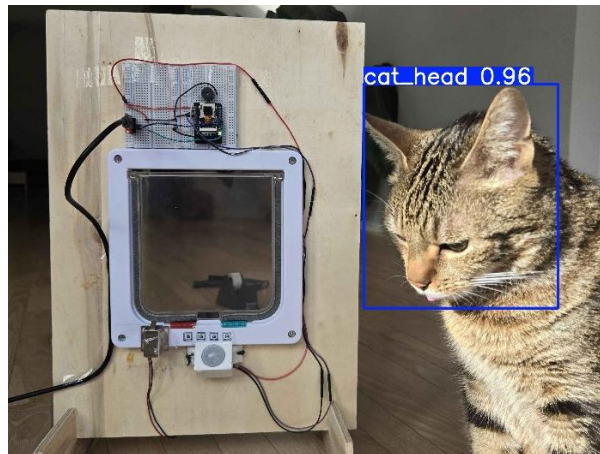

Conception d'une chatière connectée avec détection et blocage automatique des proies

*Travail de fin d'étude présenté en vue de l'obtention du diplôme de bachelier en
Technologies de l'informatique*



Patrycja DREWNOWSKA

Rapporteur : Arnaud DEWULF

Année Académique 2025 – 2026

<p>Dans cette étude, le rôle de l'IA générative a été :</p> <p><input type="checkbox"/> A) J'ai écrit l'intégralité de mon texte sans avoir eu recours à un outil d'IA générative ;</p> <p><input checked="" type="checkbox"/> B) J'ai rédigé le contenu de mon travail mais j'ai sollicité un outil d'IA générative pour améliorer</p>	
<p><input checked="" type="checkbox"/> L'orthographe</p> <p><input checked="" type="checkbox"/> La grammaire</p> <p><input checked="" type="checkbox"/> La syntaxe</p>	
<p><input type="checkbox"/> C) J'ai consulté un outil d'IA générative pour m'inspirer et puiser des idées de rédaction au niveau du contenu ou de la structure ;</p> <p><input type="checkbox"/> D) J'ai construit des idées que j'ai ensuite soumises à un outil d'IA générative qui m'a aidé à développer mon texte sur base de ces idées ;</p> <p><input type="checkbox"/> E) J'ai sollicité un outil d'IA générative à des fins de traduction ;</p> <p><input type="checkbox"/> F) J'ai confronté plusieurs propositions de contenu produit par l'IA générative pour en sélectionner les passages les plus pertinents, et j'ai édité et amélioré pour la plupart ;</p> <p><input type="checkbox"/> G) J'ai édité et amélioré une proposition de contenu produit par l'IA générative ;</p> <p><input type="checkbox"/> H) Une ou des parties de mon travail ont été intégralement produites au moyen d'un outil d'IA générative sans apport original de ma part.</p>	

Remerciements

Je souhaite tout d'abord remercier mon rapporteur, Monsieur Arnaud Dewulf, pour son accompagnement, ses conseils et sa disponibilité tout au long de la réalisation de ce travail de fin d'études.

Je remercie également ma cliente, Madame Jolanta Drewnowska, pour m'avoir proposé une problématique concrète, basée sur un besoin réel, et pour m'avoir permis de travailler sur un projet utile, technique et motivant.

Enfin, je tiens à remercier mon père, Wieslaw Drewnowski, pour son aide dans la conception du prototype en bois de la chatière.

Table des matières

Remerciements	2
1 – Introduction	6
2 – Cahier des charges.....	7
2.1. Contexte et problématique	7
2.2. Identification des intervenants.....	7
2.3. Identification des fonctionnalités.....	7
2.4. Les contraintes	10
3 – Analyse technique.....	12
3.1. Organisation du travail	12
3.1.1. Planning	12
3.1.2. Echanges avec la cliente	13
3.1.3. MVP	13
3.2. Choix des composants	13
3.3. Comparaison avec les produits existants	14
4 – Conception de la solution.....	15
4.1. Architecture globale	15
4.1.1. Le schéma du circuit électrique de la chatière	16
4.1.2. Problèmes techniques et solutions apportées	17
4.2. La base de données	19
4.3. Entraînement du modèle IA	19
Pour que ma chatière soit capable de détecter si un chat tient une proie dans sa bouche, j’ai dû entraîner une intelligence artificielle. Au départ, j’ai recherché des modèles déjà préentraînés, mais je n’ai trouvé aucun modèle correspondant précisément à mon besoin. J’ai donc choisi d’utiliser YOLOv26 afin d’entraîner mon propre modèle.	19
4.3.1. Choix des classes	19
4.3.2. Création du dataset	19
4.3.3. L'amélioration du dataset.....	20
5 – La réalisation	21
5.1. Développement du MVP	21
5.2. Montage du prototype physique.....	21
5.3. Problèmes rencontrés	24

5.4. Développement de l'application	24
6 – Tests et validation.....	26
6.1. Test et fiabilité de l'IA.....	26
6.1.1. Analyse de la matrice de confusion	26
6.1.2. Analyse du processus d'apprentissage	27
6.2. Tests sur le fonctionnement de la chatière	28
7- Législation et sécurité.....	29
7.1. Législation	29
7.2. Sécurité	29
8 – Conclusion	30
9 – Bibliographie.....	31

1 – Introduction

Les chats sont des animaux très indépendants. De nombreuses personnes installent des chatières pour leur permettre d'entrer et de sortir librement de la maison. Mais cette liberté peut parfois amener des situations pas agréables à l'intérieur du domicile.

En effet, il est fréquent que les chats ramènent leurs proies, comme des souris ou de petits oiseaux. C'est surtout problématique lorsque le chat libère sa proie encore vivante dans la maison, où elle peut alors se déplacer librement. Cette situation n'est agréable pour personne.

L'objectif de ce travail de fin d'études est donc de concevoir une chatière connectée capable de détecter si un chat transporte ou non une proie dans sa bouche. En cas de détection, la chatière se bloque automatiquement, et seul un chat sans proie est autorisé à entrer.

Ce rapport présente le cahier des charges, l'analyse des solutions technologiques, la conception du système, la réalisation du prototype ainsi que les différents tests effectués afin de vérifier la fiabilité du produit.

2 – Cahier des charges

2.1. Contexte et problématique

L'objectif est de répondre au besoin de Madame Jolanta Drewnowska, propriétaire de plusieurs chats qui sortent à l'extérieur et rapportent régulièrement des proies, telles que des souris ou des oiseaux, à l'intérieur de la maison. Cette dernière souffre d'une grande phobie des souris, ce qui lui provoque beaucoup de stress lorsqu'elle en voit chez elle. Cela pose également des problèmes d'hygiène et de confort.

La solution doit donc permettre d'identifier le chat, de détecter s'il porte une proie et de bloquer l'accès si nécessaire.

2.2. Identification des intervenants

Pour la suite de ce document, les intervenants du projet sont définis comme suit :

- **La Propriétaire** : Madame Jolanta Drewnowska, cliente et utilisatrice principale de la solution.
- **Les Chats** : utilisateurs dont l'accès doit être contrôlé.
- **L'Administrateur** : le concepteur du système, c'est-à-dire moi-même, en charge de la réalisation et de la maintenance du produit.

2.3. Identification des fonctionnalités

Voici un tableau résumant toutes les fonctionnalités et dans l'ordre de priorité :

F1	La détection d'objet dans la bouche du chat	Fait partie du MVP
F2	Un système de verrouillage de la chatière	Fait partie du MVP
F3	La reconnaissance du chat	
F4	La gestion du passage du chat	Fait partie du MVP
F5	La possibilité de consulter chaque passage des chats	
F6	La définition d'horaires	
F7	Un système d'alerte	

F1 : La détection d'objet dans la bouche du chat

User story : En tant que propriétaire, je veux empêcher mon chat d'entrer avec une proie pour éviter d'avoir des animaux morts ou vivants dans la maison.

Critères d'acceptation :

- La chatière détecte automatiquement si le chat a un objet en bouche ou pas.
- Si aucun objet n'est détecté, la chatière s'ouvre.
- Si un objet est détecté, la chatière reste fermée.
- Si la chatière a un doute sur le fait que le chat a une proie en bouche, alors elle reste fermée.
- Une image est enregistrée lorsqu'un chat est détecté avec une proie.

Complexité : Élevée

Importance : Très élevée (fonctionnalité principale)

F2 : Un système de verrouillage de la chatière

User story : En tant que propriétaire, je veux que la chatière puisse se verrouiller afin de contrôler les entrées et sorties du chat.

Critères d'acceptation :

- La chatière se verrouille automatiquement après chaque passage du chat.
- La chatière se déverrouille correctement lors du passage du chat autorisé.
- Le verrou est fiable et s'ouvre à chaque fois qu'il le faut.

Complexité : Élevée

Importance : Très élevée (fonctionnalité principale)

F3 : La reconnaissance du chat

User story : En tant que propriétaire, je veux que seuls mes chats puissent utiliser la chatière pour éviter l'entrée d'animaux inconnus.

Critères d'acceptation :

- La chatière identifie le chat via sa puce électronique.
- Si le chat est reconnu alors le chat est autorisé à rentrer.
- Si le chat n'est pas reconnu alors l'accès est refusé.

Complexité : Moyenne

Importance : Élevée

F4 : La gestion du passage du chat

User story : En tant que propriétaire, je veux que l'utilisation de la chatière soit simple et intuitive pour mon chat.

Critères d'acceptation :

- La chatière émet un son lorsqu'elle est déverrouillée pour signaler au chat qu'il peut rentrer.
- La chatière émet un son lorsque l'accès est refusé.
- Lorsque la chatière est déverrouillée, le chat peut ouvrir la porte en la poussant légèrement.

Complexité : Faible**Importance :** MoyenneF5 : La possibilité de consulter chaque passage des chats

User story : En tant que propriétaire, je veux consulter l'historique des passages afin d'analyser les habitudes de mes chats.

Critères d'acceptation :

- Chaque passage est enregistré avec : la date, l'heure, le nom du chat et s'il a une proie ou non.
- Les données sont consultables dans l'application mobile.

Complexité : Faible**Importance :** MoyenneF6 : La définition d'horaires

User story : En tant que propriétaire, je veux définir des horaires durant lesquels mes chats ne peuvent pas sortir, notamment la nuit.

Critères d'acceptation :

- L'utilisateur peut définir des plages horaires via l'application.
- En dehors de ces plages, la sortie est bloquée.
- L'utilisateur peut choisir si le chat peut entrer, sortir ou les deux.
- Les horaires peuvent être modifiés facilement.

Complexité : Moyenne**Importance :** Faible

Remarque : Cette fonctionnalité n'a pas été implémentée dans le prototype, car le système de verrouillage choisi ne permet pas de gérer indépendamment les entrées et

les sorties. Elle reste néanmoins réalisable dans une version future du projet et avec un changement du système de verrouillage de la chatière.

F7 : Un système d'alerte

User story : En tant que propriétaire, je souhaite recevoir des notifications sur l'application mobile en cas de situation anormale afin de réagir rapidement.

Critères d'acceptation :

Une notification est envoyée si :

- Une proie est détectée
- Un chat inconnu tente d'entrer
- Le chat reste bloqué dehors trop longtemps

La notification contient :

- La date et l'heure
- Une image ou vidéo

Complexité : Moyenne

Importance : Moyenne

2.4. Les contraintes

Pour répondre aux attentes de ma cliente, j'ai identifié quatre contraintes :

1. **Consommation d'énergie :** C'est l'une des contraintes les plus importantes, car ma cliente souhaite que la chatière puisse fonctionner sur piles et qu'il n'y ait pas un gros câble qui sorte de sa chatière. Pour faire cela, il est nécessaire de correctement choisir les composants afin que la consommation soit la plus faible possible.
2. **Fiabilité de la détection :** La chatière doit être extrêmement fiable sur la détection de proie. L'objectif est qu'elle ne laisse passer aucune proie dans la maison. Il est donc préférable que le système génère un "faux positif", c'est-à-dire bloquer le chat par erreur s'il y a un doute, plutôt qu'un "faux négatif", c'est-à-dire laisser passer une souris. La propreté de la maison dans ce projet est plus importante qu'une petite attente dehors pour le chat.
3. **Sécurité du chat :** Le système de verrouillage doit être adapté au comportement imprévisible de l'animal. Un chat peut hésiter, s'arrêter au milieu de la porte ou mettre du temps à entrer. Il est donc nécessaire de prévoir un mécanisme qui ne risque pas de le blesser ou de le pincer si la porte se referme.

4. **Solidité et durabilité de la chatière** : Comme la chatière est en contact direct avec l'extérieur, elle doit être solide et durable. Tout le mécanisme électronique, c'est-à-dire la caméra et les capteurs, doit être bien protégé contre l'humidité, la poussière et les changements de température. Le boîtier doit donc être assez robuste pour garantir que le système fonctionne correctement, peu importe la météo.

3 – Analyse technique

3.1. Organisation du travail

Pour réaliser ce projet, j'ai choisi de diviser le travail en plusieurs étapes. Étant donné la complexité du projet, il me semble important d'avancer par petites étapes pour ne pas être dépassée par l'ampleur et la difficulté du travail.

Le projet comporte également plusieurs parties différentes, notamment l'électronique, l'intelligence artificielle, la communication entre appareils et l'application mobile. Il était donc important de ne pas essayer de tout réaliser en même temps, mais plutôt d'avancer par étapes.

L'objectif est de commencer par un prototype minimal fonctionnel (MVP), puis d'ajouter progressivement les fonctionnalités supplémentaires.

La première version fonctionnelle, ou MVP, doit répondre au besoin principal de ma cliente : détecter un chat, prendre une photo, analyser la présence éventuelle d'une proie et décider si la chatière doit s'ouvrir ou rester fermée.

Les fonctionnalités plus avancées, comme la gestion complète des horaires ou l'application mobile, sont prévues pour plus tard.

3.1.1. Planning

Le planning suivant a été défini à titre indicatif pour structurer le projet. Il ne doit pas nécessairement être suivi dans l'ordre : certaines étapes peuvent prendre plus ou moins de temps que prévu. Ce calendrier sert avant tout de fil conducteur pour structurer mon travail et me permettre de ne pas m'éparpiller.

	Durée	Objectif
1	1 à 2 semaines	Analyse des besoins de la cliente et choix des composants
2	1 semaine	Tests avec l'ESP32-CAM, création d'un script permettant de prendre des photos lorsqu'un mouvement est détecté avec un capteur de mouvement
3	1 à 2 semaines	Entraînement d'un modèle capable de détecter une proie dans la bouche d'un chat avec YOLO
4	1 semaine	Communication entre l'ESP32-CAM et le Raspberry Pi
5	2 semaines	Développement de l'application mobile et création de la base de données
6	1 semaine	Ajout d'un verrou à la chatière
7	1 semaine	Ajout d'un lecteur de puce RFID
8	1 semaine	Assemblage de tous les composants et création d'un prototype de vraie chatière

3.1.2. Echanges avec la cliente

Tout au long du projet, j'ai régulièrement échangé avec la cliente, ce qui m'a permis de mieux comprendre ses besoins et d'améliorer la solution.

Lors d'une démonstration du prototype, on s'est rendu compte qu'il manquait un retour sonore pour indiquer l'état de la chatière. J'ai donc décidé d'ajouter un buzzer pour signaler les différents cas : lorsque la chatière s'active, quand le chat est autorisé à entrer ou quand l'accès est refusé.

Cette amélioration, basée sur le retour de la cliente, permet de rendre le système plus clair et plus facile à utiliser.



3.1.3. MVP


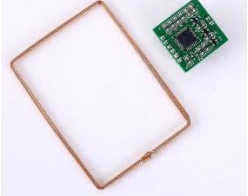


Pour le MVP, c'est-à-dire la version minimale fonctionnelle de la chatière, j'ai décidé de réaliser un premier montage avec un ESP32-CAM, un détecteur de mouvement, un verrou et un buzzer. Pour cette étape, j'ai utilisé une intelligence artificielle que j'ai entraînée avec YOLO et qui tourne sur mon ordinateur.

Le fonctionnement est le suivant : lorsque le capteur détecte un mouvement, l'ESP32-CAM prend une photo et l'envoie via Wi-Fi à mon PC. L'IA, entraînée pour détecter des proies dans la bouche du chat, analyse ensuite l'image. Le PC renvoie alors une réponse pour indiquer s'il faut autoriser ou non l'ouverture de la chatière, et le verrou ne s'ouvre que s'il reçoit cette autorisation.

Cette première version m'a permis de mettre en place les fonctionnalités les plus importantes du projet, à savoir la détection du chat, la capture d'image, la communication Wi-Fi, la détection de proie et le verrouillage de la chatière. La validation de ce fonctionnement a été une étape importante, car elle m'a permis de confirmer la faisabilité du projet avant d'intégrer le Raspberry Pi. Une fois cette base fonctionnelle terminée, j'étais bien lancée pour la suite du projet.

3.2. Choix des composants

	Composant	Son utilité	Photo
1	ESP32-CAM	Il prend en photo le chat et contrôle l'intégralité des composants attachés à la chatière	
2	Raspberry Pi 4	Permet de faire tourner l'IA qui détecte les souris, et permet de stocker les images prises par la chatière	

3	HC-SR501 PIR Capteur	Permet de détecter s'il y a un mouvement et ainsi allumer l'ESP32-CAM	
4	Lecteur RFID 134.2khz	Permet d'identifier la puce qui se trouve sous la peau du chat. Ce lecteur permet de lire des puces qui sont à une fréquence de 134.2 kHz qui est la fréquence des puces qu'ont les animaux	
5	Verrou électromagnétique	Permet de garder la porte fermée et de l'ouvrir lorsque c'est nécessaire	
6	Buzzer passif	Permet de signaler au chat s'il peut rentrer ou non dans la maison	

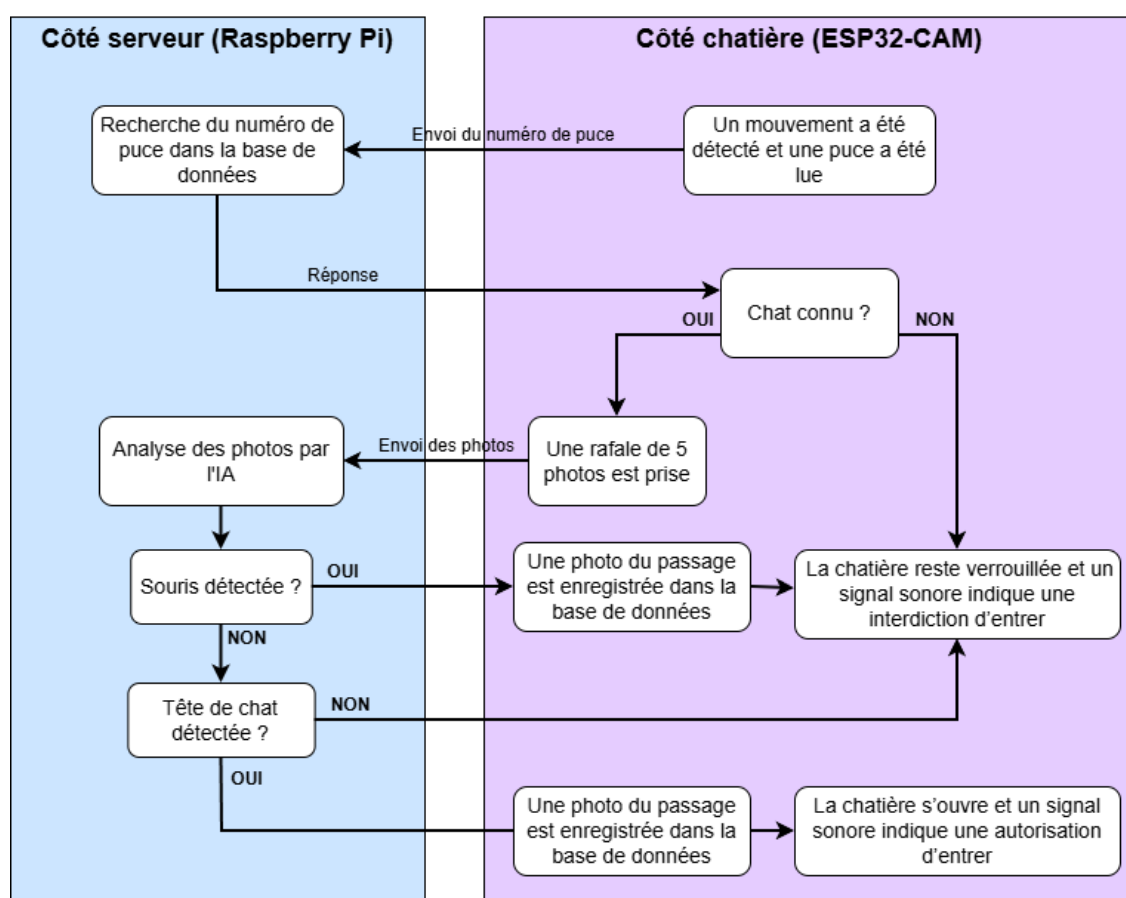
3.3. Comparaison avec les produits existants

4 – Conception de la solution

4.1. Architecture globale

La solution finale est constituée de deux parties : la chatière, contrôlée par l'ESP32-CAM et le serveur IA, hébergé sur le Raspberry Pi. Cette séparation permet de garder une chatière légère et peu énergivore, tout en déléguant les traitements plus lourds au Raspberry Pi.

Le fonctionnement de la chatière est le suivant :

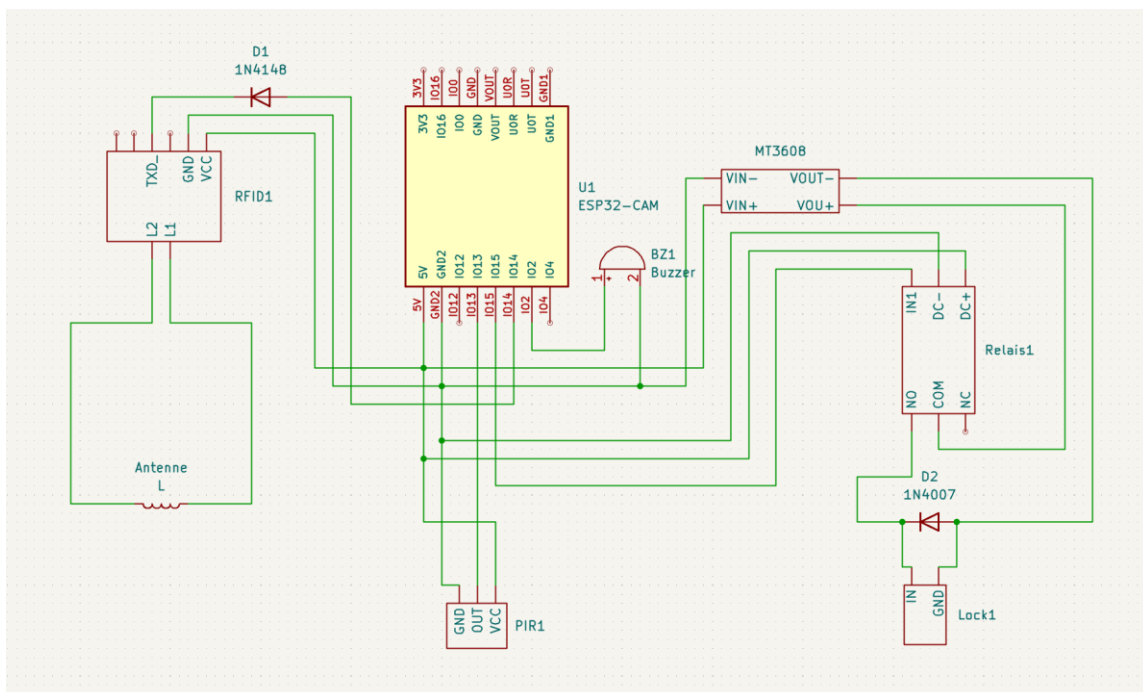


1. **Détection et allumage :** Pour minimiser la consommation d'énergie, la chatière reste en veille en permanence. Lorsqu'un mouvement est détecté par le capteur, elle s'active et initialise le lecteur RFID.
2. **Lecture de la puce :** Dès qu'une puce est détectée, la chatière envoie son identifiant au Raspberry Pi via le Wi-Fi.
3. **Identification de la puce :** Le Raspberry Pi vérifie dans la base de données si la puce est connue :

- a. **Chat inconnu** : le serveur envoie une réponse négative. La chatière reste fermée et une photo de l'animal est prise et enregistrée.
 - b. **Chat reconnu** : le serveur envoie le nom du chat pour indiquer à la chatière que le chat est connu et l'analyse des photos peut commencer.
4. **Capture et analyse du chat** : La chatière prend une rafale de 5 photos et les envoie au Raspberry Pi pour analyse par l'IA :
- a. **Proie détectée** : dès qu'une proie est détectée, la prise de photos s'arrête immédiatement. La chatière reste verrouillée et un signal sonore d'interdiction est émis.
 - b. **Aucune proie détectée** : si aucune proie n'est détectée mais qu'au moins une tête de chat est reconnue, la chatière s'ouvre.
 - c. **Aucune proie ni tête de chat détectée** : dans ce cas, la chatière reste fermée par sécurité. Il est possible que le chat soit mal positionné et que la proie ne soit pas visible. C'est pourquoi l'IA vérifie également si la tête du chat a bien été détectée avant d'autoriser l'ouverture.

4.1.1. Le schéma du circuit électrique de la chatière

Le schéma ci-dessous présente l'ensemble des connexions entre les différents composants de la chatière, c'est-à-dire l'ESP32-CAM, le capteur de mouvement, le lecteur RFID, le buzzer ainsi que le système de verrouillage.



Le circuit électronique de la chatière

4.1.2. Problèmes techniques et solutions apportées

Durant la conception, j'ai dû résoudre deux problèmes majeurs liés aux différences de tension entre les composants.

A. La puissance du verrou 12V

Le verrou électromagnétique choisi nécessite une alimentation en 12V pour fonctionner correctement. Cependant, le reste du circuit est alimenté en 5V via l'ESP32-CAM.

Pour résoudre ce problème, un module élévateur de tension (MT3608) a été utilisé pour convertir le 5V qui sort de l'ESP32-CAM en 12V.

Comme l'ESP32-CAM ne peut pas fournir suffisamment de courant pour piloter directement le verrou, un relais a également été ajouté. Celui-ci agit comme un interrupteur : lorsqu'il reçoit un signal de l'ESP32-CAM, il permet d'alimenter le verrou en 12V.

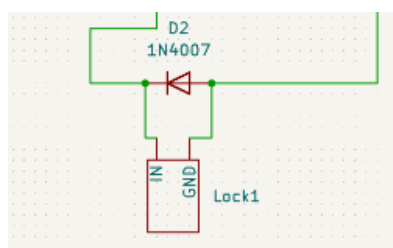
Protection du circuit :

Pour protéger le circuit, j'ai ajouté une diode 1N4007 en parallèle du verrou électromagnétique.

Le verrou étant une charge inductive (il contient une bobine), il stocke de l'énergie lorsqu'il est alimenté. Lorsque le courant est coupé, cette énergie peut provoquer une surtension appelée retour de courant.

Cette surtension peut endommager les composants du circuit, comme le relais ou l'ESP32-CAM.

La diode agit comme une diode de roue libre : elle ne conduit pas pendant le fonctionnement normal, mais lors de la coupure, elle donne un chemin au courant résiduel. Cela permet de dissiper l'énergie progressivement et d'éviter un pic de tension dangereux qui pourrait endommager les composants électroniques du circuit, notamment le relais ou l'ESP32-CAM.



Diode de protection

Le problème du double verrouillage :

Lors de la conception, j'ai rencontré un problème lié au système de verrouillage. Initialement, j'avais prévu d'utiliser un seul verrou électromagnétique, sans prendre en compte le fait que je devais bloquer à la fois l'entrée et la sortie de la chatière.

En ajoutant un second verrou, je me suis rendu compte que l'alimentation n'était plus suffisante pour faire fonctionner l'ensemble du système. Lors de l'activation des verrous, la consommation devenait trop importante, ce qui provoquait une chute de tension. Cela entraînait un redémarrage de l'ESP32-CAM, rendant le système instable.

Solution pour un meilleur verrouillage :

Le modèle de verrou choisi est efficace pour bloquer un seul côté de la chatière. Cependant, pour bloquer à la fois l'entrée et la sortie, il est nécessaire d'en utiliser deux, ce qui, comme expliqué au paragraphe précédent, n'est pas compatible avec le système actuel.

Une première piste d'amélioration consisterait à modifier l'embout du verrou afin de permettre un blocage simultané des deux côtés avec un seul mécanisme.

La solution la plus pertinente que j'ai identifiée est l'utilisation d'un verrou à vis sans fin. Ce type de mécanisme fonctionne sous une tension de 3V et consomme beaucoup moins d'énergie.

Cette alternative permet de supprimer le convertisseur de tension (MT3608), de simplifier le circuit et de réduire la consommation globale. Elle rend également le système plus adapté à une alimentation sur batterie.

Malheureusement, cette solution a été identifiée trop tard dans le projet et n'a donc pas pu être implémentée.

B. Le lecteur RFID

Le lecteur RFID fonctionne en 5V. Il envoie donc ses données avec une tension de 5V, alors que les broches de l'ESP32-CAM sont limitées à 3,3V. Si le lecteur envoie en continu un signal en 5V, cela peut endommager l'ESP32-CAM. Il a donc fallu trouver une solution pour abaisser cette tension.

J'ai d'abord essayé d'utiliser un pont diviseur de tension avec des résistances afin d'abaisser le signal de 5V à 3,3V. Cependant, cette solution n'était pas fiable : la tension en sortie était trop faible, ce qui empêchait l'ESP32-CAM de détecter correctement les informations.

Pour protéger l'ESP32-CAM tout en conservant un signal exploitable, j'ai utilisé une diode 1N4148. Cette diode permet de limiter la tension appliquée à l'entrée de l'ESP32 en empêchant le signal de dépasser une certaine valeur. Ainsi, le microcontrôleur peut recevoir les informations du lecteur RFID sans risque de surtension.

4.2. La base de données

Une base de données a été créée afin de regrouper toutes les informations liées aux chats comme le numéro de la puce, les photos, l'historique des passages, ...

La base de données a été pensée de manière évolutive. Ainsi, si le produit est amené à être utilisé par d'autres personnes ou à évoluer dans le futur, il sera plus facile d'ajouter de nouveaux utilisateurs et de nouvelles fonctionnalités.

4.3. Entraînement du modèle IA

Pour que ma chatière soit capable de détecter si un chat tient une proie dans sa bouche, j'ai dû entraîner une intelligence artificielle. Au départ, j'ai recherché des modèles déjà préentraînés, mais je n'ai trouvé aucun modèle correspondant précisément à mon besoin. J'ai donc choisi d'utiliser YOLOv26 afin d'entraîner mon propre modèle.

4.3.1. Choix des classes

J'ai décidé d'entraîner le modèle avec deux classes : « **cat_head** » et « **prey** ». J'ai fait le choix de ces deux classes car je voulais être sûr que l'IA voit bien l'entièreté du visage du chat avant de prendre une décision. Si je n'avais pas créé la classe « cat_head », l'IA pourrait analyser uniquement des photos du dos du chat, ne voir aucune proie, et ouvrir la porte alors que la souris est juste cachée. En forçant la détection de la tête du chat, je m'assure d'une détection de proie plus fiable.

4.3.2. Création du dataset

Afin d'avoir un modèle qui est entraîné le mieux possible selon mes besoins, j'ai dû créer mon propre dataset, c'est-à-dire les différentes photos sur lesquelles va s'entraîner l'IA.

Pour faire cela, j'ai récupéré plusieurs datasets sur Roboflow que j'ai fusionnés ensemble et j'ai également ajouté mes propres photos.

Étant donné que les images viennent de différents datasets j'ai dû tous réannoter manuellement. C'est un travail qui a pris du temps mais c'était nécessaire pour que l'entraînement corresponde aux besoins du projet.

Au total mon dataset contient **743** photos : **743** avec la classe « **cat_head** » et **370** avec la classe « **prey** ».

Idéalement il aurait été mieux d'avoir plus de photos mais c'est malheureusement assez difficile de trouver des photos de chats avec une proie en bouche, et comme je voulais avoir un nombre équitable de photo avec un chat seul et un chat avec une proie, j'ai entraîné mon modèle avec que 743 photos. Si je voulais ajouter mes propres photos de chats avec une souris, j'aurais dû passer plusieurs mois à récolter ces images car elles ne sont pas faciles à obtenir.

4.3.3. L'amélioration du dataset

Pour que mon IA soit plus efficace, j'ai ajouté des modifications aux photos. J'ai appliqué des rotations, du flou, du bruit ou encore des changements de saturation.

L'objectif est de rendre le modèle plus robuste, afin qu'il puisse reconnaître une proie même si le chat est en mouvement, si l'image est floue ou si les conditions de luminosité sont mauvaises.

5 – La réalisation

Dans cette partie, je vais expliquer comment j'ai fabriqué le prototype, en partant de mes premiers montages jusqu'au résultat final.

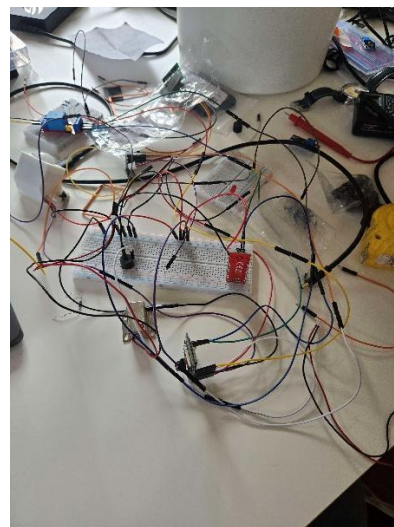
5.1. Développement du MVP

Le but de mon MVP était de valider le fonctionnement de base : la chatière détecte un mouvement, prend une photo, l'envoie à l'intelligence artificielle, puis celle-ci décide si la porte doit s'ouvrir ou non.

Pour commencer, j'ai réalisé ce montage sur une breadboard. À ce moment-là, j'avais déjà ma chatière fixée dans une planche en bois, mais j'ai préféré faire les branchements à côté pour tester si tout marchait bien. Avec le recul, je pense que j'aurais dû tout installer directement sur la planche en bois dès le début. Cela m'aurait permis de mieux visualiser l'emplacement des composants et d'avoir un résultat concret plus rapidement.



Planche en bois avec la chatière



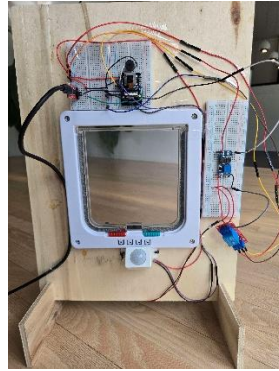
Montage sur breadbord

5.2. Montage du prototype physique

Pour le montage final, j'ai fixé l'ensemble du circuit sur la planche en bois représentant la porte. J'ai réalisé plusieurs essais afin d'optimiser la disposition des composants pour obtenir un système compact. L'objectif était de pouvoir cacher l'ensemble du mécanisme dans un boîtier imprimé en 3D.



Première version du prototype



Deuxième version du prototype (l'avant)



Deuxième version du prototype (l'arrière)

Au début, comme on peut le voir sur les photos, le lecteur de puce n'était pas encore intégré à la chatière car j'ai mis du temps à réussir à le faire fonctionner correctement. Une fois que tous les composants fonctionnaient ensemble, j'ai tout assemblé pour que ce soit le plus propre possible.

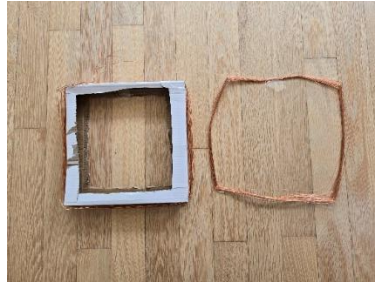


Ajout du lecteur de puce, première version



Ajout du lecteur de puce, deuxième version

Mon lecteur de puce est arrivé avec une très petite antenne et une très faible distance de détection. J'ai initialement pensé à en fabriquer une plus grande moi-même pour qu'elle fasse tout le tour de la chatière. Mais après plusieurs essais qui n'ont pas fonctionné j'ai dû abandonner car c'était trop complexe et cela m'aurait pris trop de temps. J'ai donc gardé l'antenne d'origine. Elle fonctionne très bien, mais il faut que le chat soit assez proche de la chatière pour que sa puce soit lue.



Essais de création d'antenne

Enfin, pour faire mes tests sans embêter mon chat toutes les 5 minutes, j'ai acheté plusieurs puces qu'on injecte normalement sous la peau des animaux. Elles sont vendues dans des seringues. J'en ai pris quelques-unes que j'ai scotchées sur un bout de carton, ce qui m'a permis de simuler le passage d'un chat et de tester si la chatière s'ouvrait correctement.

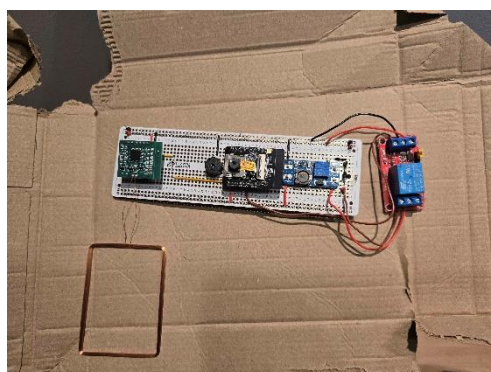


Seringues avec les puces



Puces de test associé à des chats

La version finale de mon prototype je l'ai soudé sur une plaque de prototypage, il aurait été préférable de réaliser un PCB mais j'ai malheureusement manqué de temps.



Prototype soudé



Prototype sur la chatière

5.3. Problèmes rencontrés

Lors de l'installation du système sur la planche, j'ai rencontré un problème lié au système de verrouillage. Initialement, j'avais prévu d'ajouter un deuxième verrou afin de bloquer à la fois l'entrée et la sortie de la chatière.

Cependant, après plusieurs essais, je me suis rendu compte que ce système ne fonctionnait pas comme prévu et qu'il nécessiterait une modification importante de l'architecture globale. Étant donné l'état d'avancement du projet, il était trop tard pour repenser entièrement cette partie.

J'ai donc choisi de conserver un seul verrou fonctionnel. Bien que cette solution ne soit pas optimale, elle reste fiable et constitue une base solide pour de futures améliorations.

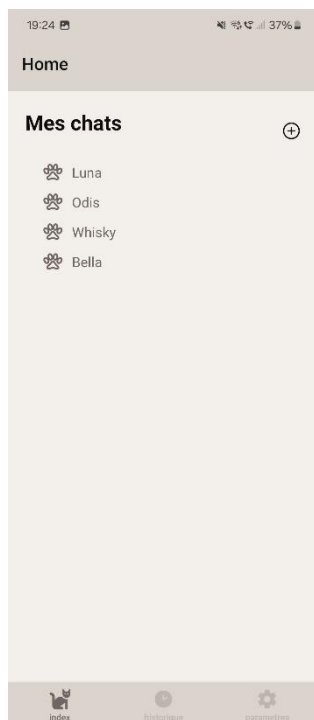
5.4. Développement de l'application

Pour que ma cliente puisse suivre l'activité de ses chats, j'ai créé une application mobile. Elle permet de voir l'historique des passages avec plusieurs informations telles que l'heure, le nom du chat et si l'IA a détecté une proie ou non.

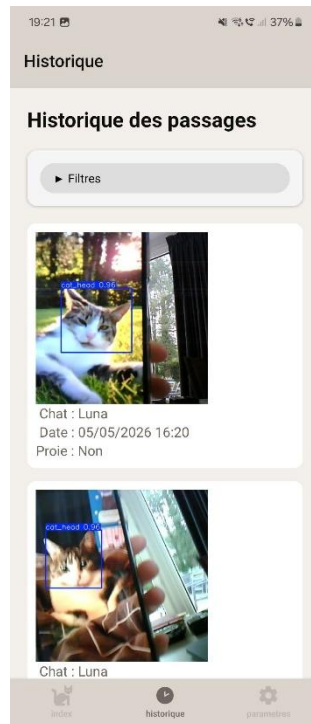
L'application a aussi d'autres fonctions pratiques :

- **Ajouter un chat** : La cliente peut enregistrer un nouveau chat et l'associer à la chatière.
- **Notifications** : Si un chat inconnu se présente devant la maison, l'application envoie une alerte. La cliente peut choisir d'activer ou de désactiver cette option selon ses envies.

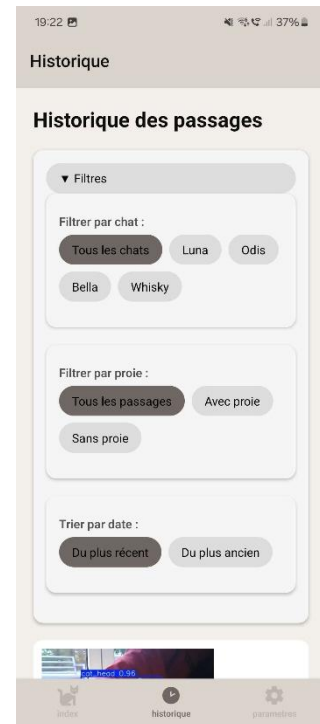
L'application n'a pas une grande utilité pratique, elle permet surtout de voir les passages des chats, d'ajouter ou supprimer des chats.



Page d'accueil qui regroupe tous les chats



Page avec les passages des chats



Option qui permet de filtrer les passages des chats

6 – Tests et validation

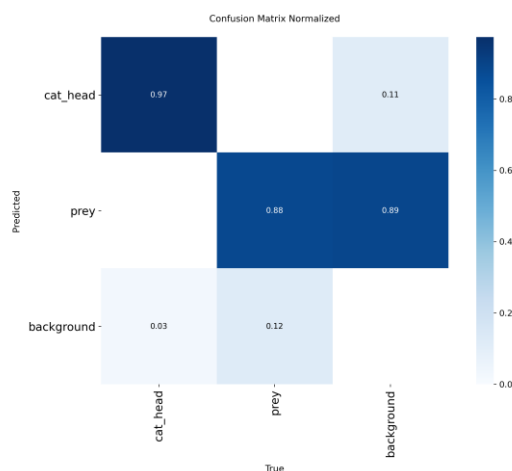
6.1. Test et fiabilité de l'IA

Pour entraîner mon IA qui détecte si un chat a une souris en bouche j'ai utilisé YOLO26. J'ai fait plusieurs entraînements pour arriver aux meilleurs résultats possibles. Après chaque entraînement, j'ai analysé les résultats, ce qui était bien et ce qui l'était moins pour que je puisse améliorer mon modèle.

Ce que j'ai appris durant la réalisation de ce projet, c'est que le dataset, c'est-à-dire les données avec lesquelles j'entraîne l'IA, est la chose la plus importante. Plus j'ai d'images de bonnes qualité, variées, pertinentes meilleurs seront les résultats. Mon modèle final que j'ai entraîné pour ce prototype n'est pas parfait, mais il est déjà très efficace pour détecter des proies et voici pourquoi.

6.1.1. Analyse de la matrice de confusion

Lorsqu'une IA est entraînée, il existe plusieurs moyens d'évaluer sa précision. Dans ce paragraphe, j'analyse le graphique « Confusion Matrix Normalized ».



Dans un premier temps, on peut lire les résultats en diagonale :

- **La tête du chat « cat_head »** : Elle est détectée dans 97 % des cas. C'est un excellent score qui montre que le modèle reconnaît la tête d'un chat presque à tous les coups.
- **Les proies « prey »** : Elles sont détectées à 88 %. C'est un bon résultat, car une proie est plus petite et plus difficile à voir qu'un chat, il est donc normal d'avoir un résultat moins bon.
- **Les erreurs ou faux négatifs** : En bas du graphique, on voit que seulement 3 % des têtes de chats ne sont pas vues, et 12 % des proies sont manquées. C'est pour cette raison que ma chatière prend une rafale de plusieurs photos :

si l'IA rate la souris ou la tête du chat sur la première image, elle a une autre chance de les voir sur la deuxième ou la troisième.

On peut ensuite analyser la colonne de droite :

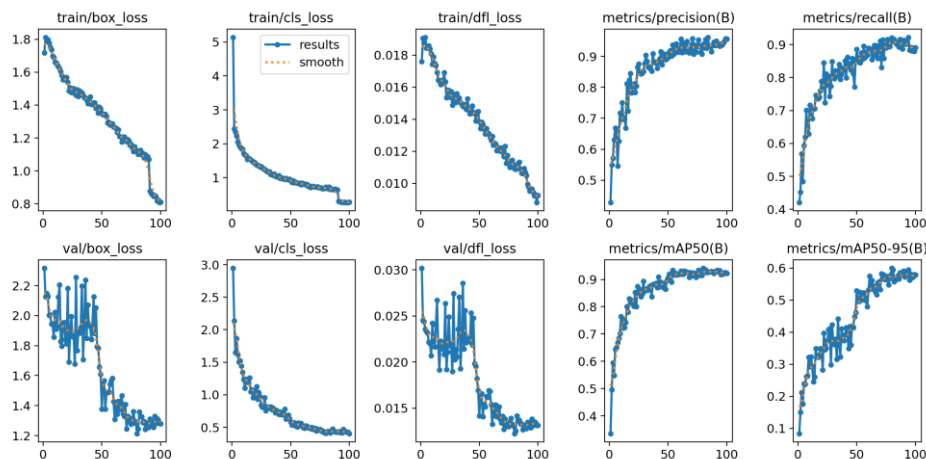
Si on regarde cette colonne on peut voir que dans 11% des cas une tête de chat est détectée alors qu'il n'y en a pas. Et plus inquiétant encore, dans 89% des cas l'IA détecte une proie alors qu'il n'y en a pas.

À première vue, cela peut sembler problématique, mais dans mon cas, ce n'est pas forcément grave :

1. **Filtrage dans le code :** Comme l'IA est très fiable sur la détection de tête de chat, je peux définir un réglage dans mon programme que seules les proies qui se trouvent très proche de la tête d'un chat sont acceptées. Ainsi même si une fausse proie est détectée et qu'aucune tête n'a été trouvée, cette proie sera ignorée.
2. **Sécurité :** Il est préférable que la chatière se bloque par erreur plutôt que de laisser passer un chat avec une proie.

6.1.2. Analyse du processus d'apprentissage

Pour compléter l'analyse de la matrice de confusion, il est important de regarder comment l'IA a appris au fil du temps. Le graphique ci-dessous montre les 100 époques ou étapes de son entraînement :



On peut diviser ce graphique en deux grandes parties :

1. **La baisse des erreurs :** On remarque que le taux d'erreur baisse de manière très régulière. Cela signifie que l'IA a bien réussi à corriger ses erreurs au fur et à mesure des entraînements.

2. **L'évolution de la précision :** La courbe mAP50, qui représente la précision globale du modèle, augmente rapidement au début. Cela montre que l'IA apprend vite au départ. Ensuite, la courbe se stabilise autour de 90 %, ce qui signifie que le modèle devient de plus en plus précis et qu'il progresse moins rapidement. On observe aussi une petite amélioration vers la fin, autour de l'époque 90. Cela correspond au moment où l'algorithme ajuste les derniers détails pour être le plus précis possible. En résumé, l'IA apprend d'abord rapidement, puis elle se stabilise une fois qu'elle a atteint un bon niveau de performance.

Ces graphiques montrent donc que l'entraînement s'est bien déroulé. Les courbes deviennent stables vers la fin, ce qui indique que le modèle a appris tout ce qu'il pouvait à partir des données disponibles.

Si les courbes continuaient à évoluer fortement à la 100ème époque, cela aurait signifié qu'un entraînement plus long était nécessaire. Ici, le nombre d'itérations était donc adapté.

6.2. Tests sur le fonctionnement de la chatière

Dans le code qui contrôle la chatière, j'ai ajouté des commentaires afin de suivre précisément le comportement du système et d'identifier ce qui fonctionne ou non.

Ces tests m'ont permis de vérifier que la chatière réagit correctement dans les différentes situations : détection d'un mouvement, reconnaissance du chat, analyse des images par l'IA, puis décision d'ouvrir ou non la porte.

Grâce à ces vérifications, j'ai pu m'assurer que le système global fonctionne comme prévu et que les différentes parties communiquent correctement entre elles.

7- Législation et sécurité

7.1. Législation

Dans le cadre de ce projet, plusieurs aspects légaux doivent être pris en compte.

Tout d'abord, la chatière prends des photos des chats pour être analyser. Il faut faire attention avec cela car il n'est pas autorisé de capturer des personnes à leur insu. Il a été important de trouver un angle pour la caméra qui capture le moins possible le décor extérieur.

Ensuite les photos prisent par la chatière sont directement stocker chez le client, sur son Raspberry Pi, cela permet de garder les photos privé et donc personne à par la cliente n'aura accès aux photos prise devant chez elle.

7.2. Sécurité

La sécurité du système est un point essentiel dans ce projet.

D'un point de vue matériel, la chatière comporte un verrou électromagnétique. Il est donc important de s'assurer que celui-ci fonctionne correctement pour éviter toute ouverture non désirée ou tout blocage du chat. Un système fiable permet d'éviter les risques pour l'animal.

Au niveau du fonctionnement, une règle de sécurité a été mise en place dans le programme : en cas de doute dans la détection (par exemple si l'IA pense voir une proie), la chatière reste fermée. Cela permet d'éviter que le chat rentre avec une proie à l'intérieur de la maison.

8 – Conclusion

Ce projet m’a permis de concevoir une chatière intelligente capable de détecter si un chat transporte une proie avant de le laisser entrer. Pour y parvenir, j’ai combiné plusieurs technologies, notamment l’intelligence artificielle avec YOLO, un ESP32-CAM, un Raspberry Pi ainsi qu’un système de verrouillage et un lecteur RFID.

Au cours de ce travail, j’ai rencontré plusieurs difficultés, notamment lors de l’intégration des différents composants et du système de verrouillage. Certaines idées initiales n’ont pas pu être mises en place comme prévu, mais j’ai su m’adapter et trouver des solutions alternatives pour obtenir un prototype fonctionnel.

Ce projet m’a permis de développer de nombreuses compétences, autant en programmation qu’en électronique et en intelligence artificielle. J’ai également appris à mieux organiser mon travail, à tester mon système et à corriger les erreurs au fur et à mesure.

Même si le système n’est pas parfait, il constitue une bonne base d’amélioration. Par exemple, il serait possible d’augmenter encore la précision du modèle ou d’améliorer le système de verrouillage pour le rendre plus fiable.

En conclusion, ce projet a été une expérience enrichissante qui m’a permis de mettre en pratique mes connaissances tout en découvrant de nouvelles technologies.

9 – Bibliographie

- CircuitDigest, *How to Program ESP32-CAM using Arduino*, disponible sur : <https://circuitdigest.com/microcontroller-projects/how-to-program-esp-32-cam-using-arduino>
- Random Nerd Tutorials, *ESP32-CAM PIR Motion Detector with Photo Capture*, disponible sur : <https://randomnerdtutorials.com/esp32-cam-pir-motion-detector-photo-capture/>
- Roboflow Universe, *Cat with Prey Detection Dataset*, disponible sur : <https://universe.roboflow.com/pino-carezza-mgn6v/cat-with-prey-detection>
- RaspberryTips, *ESP32 vs Raspberry Pi Pico : comparatif*, disponible sur : <https://raspberrytips.fr/esp32-vs-raspberry-pico/>
- YouTube, *ESP32-CAM Tutorial* (vidéo), disponible sur : <https://www.youtube.com/watch?v=y89yJ1Fq-hQ>
- Ultralytics Documentation, *Guide de test des modèles YOLO*, disponible sur : <https://docs.ultralytics.com/fr/guides/model-testing>
- Locoduino, *Introduction aux microcontrôleurs*, disponible sur : <https://www.locoduino.org/spip.php?article30>