

UNIVERSITÉ DE NAMUR

RAPPORT DE PROJET : BINNY, LA POUBELLE
INTELLIGENTE

Rapport de laboratoire



1^{er} quadrimestre 2024-2025

Groupe 3

GENTILE Donato
ALARCON Diego
DEVIGNE Matteo
BARBIEUX Arthur
YANDO DJAMEN Rodrigue



Table des matières

1	Introduction	2
2	Objectifs du Projet	2
3	Description de Binny	2
4	Fonctionnalités Clés de Binny	3
5	Matériel	4
6	Technologies Utilisées	7
6.1	Scala et Akka (INFOM451)	7
6.1.1	Les acteurs	7
6.1.2	L'animation	7
6.1.3	Le websocket	7
6.2	Python et Intelligence Artificielle (INFOM450, INFOM453)	7
6.3	Architecture REST Maillée	8
7	Évolutions possibles	9
8	Conclusion	9

1 Introduction

Dans le cadre des cours INFOM450, INFOM451 et INFOM453, nous avons développé un projet innovant intitulé **Binny**, une poubelle intelligente capable de trier automatiquement les déchets tout en proposant une expérience éducative et ludique pour les enfants. Ce projet répond aux besoins d'un persona qui souhaite adopter des pratiques écologiques tout en optimisant son temps, notamment pour passer plus de moments de qualité avec ses enfants.

2 Objectifs du Projet

Le projet avait pour but principal de :

- Répondre aux besoins d'un persona soucieux de l'environnement mais disposant de peu de temps en raison de son travail.
- Proposer une solution technologique innovante permettant d'encourager le tri des déchets de manière ludique et éducative.
- Utiliser des technologies modernes, adaptées aux besoins spécifiques du projet.

3 Description de Binny

Binny est une poubelle intelligente équipée d'une caméra capable de détecter et d'identifier les déchets. Grâce à un modèle d'intelligence artificielle, elle trie automatiquement les déchets montrés à sa caméra. En outre, elle est connectée à un modèle de langage (LLM) qui permet d'interagir avec les enfants en leur racontant des blagues sur le tri des déchets et ainsi de les éduquer sur l'importance du recyclage et de leur apprendre à trier de manière efficace.



FIGURE 1 – Binny la poubelle intelligente

4 Fonctionnalités Clés de Binny

- **Tri Automatique des Déchets** : Grâce à la caméra et à l'algorithme d'IA, Binny identifie et trie les déchets en temps réel.

- **Interaction Éducative** : Binny engage les enfants dans des conversations amusantes et éducatives sur le recyclage.
- **Connectivité Fiable** : L'architecture maillée garantit une utilisation fluide, qu'elle soit locale ou connectée à un serveur distant.

5 Matériel

Pour la réalisation de Binny, nous avons utilisé les éléments matériels suivants :

- **Raspberry Pi 4 2Gb** : Utilisé comme unité centrale de traitement.
- **Écran LCD GPIO pour Raspberry Pi** : Fournissant une interface utilisateur visuelle.
- **Poubelle en plastique** : Structure principale pour le projet.
- **Servo moteur connecté à un module de contrôle Phidget** : Permettant l'ouverture et la fermeture automatique de la poubelle.
- **Hub de contrôle de capteur Phidget** : Centralisant la gestion des différents capteurs.
- **Capteur de distance Phidget** : Mesurant le remplissage de la poubelle.
- **Capteur d'infrarouges Phidget** : Identifiant la présence d'un pied pour l'ouverture automatique.
- **Capteur de force Phidget** : Mesurant la pression exercée sur le couvercle de la poubelle.
- **Capteur de touché Phidget** : Fournissant un bouton tactile pour l'ouverture de la poubelle.
- **Camera** : Utilisée pour faire une capture du déchet à jeter dans la poubelle.
- **Microphone** : Utilisé pour l'interaction avec l'assistant vocal.

Voici un petit schéma montrant comment tout est mis en place :

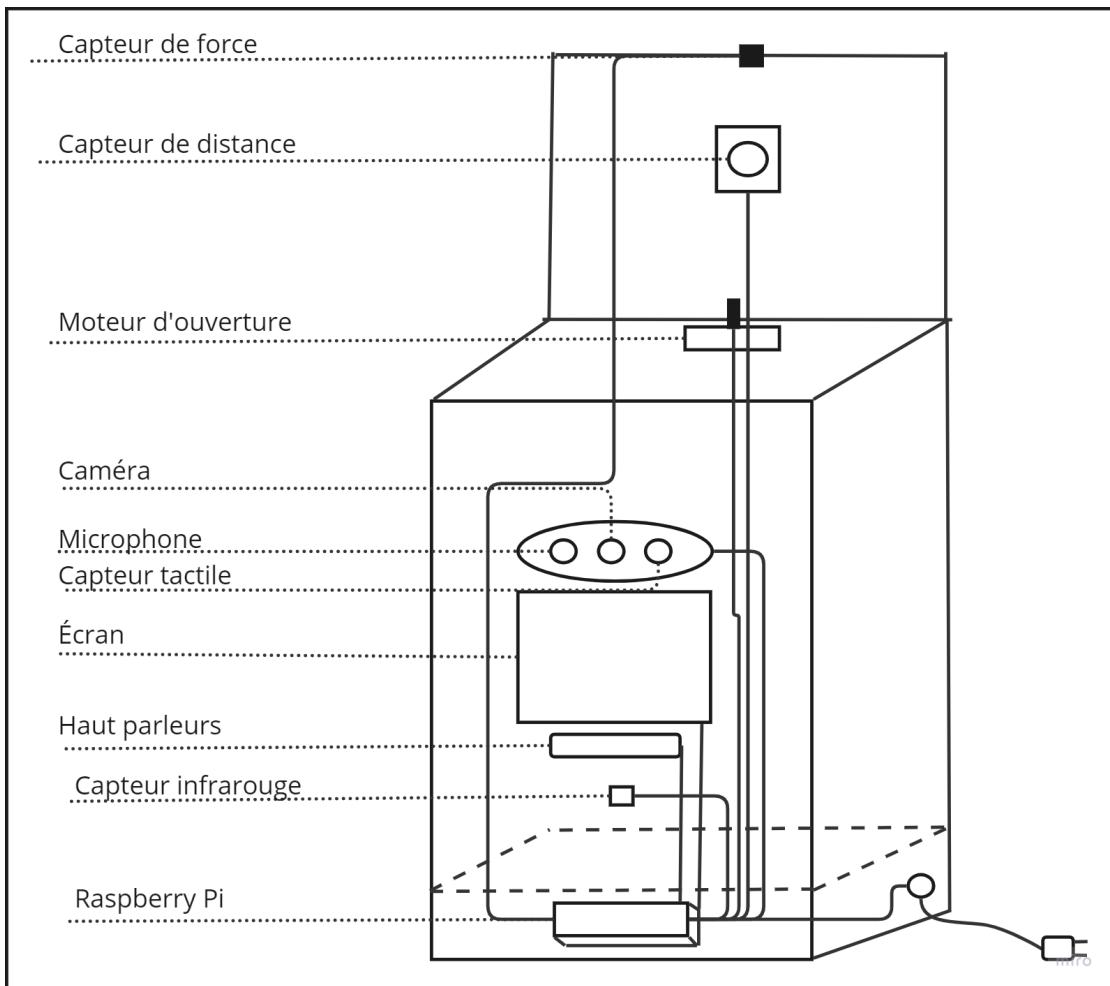


FIGURE 2 – Schéma matériel Binny

Et voici ce que cela donne à l'intérieur une fois que c'est monté :

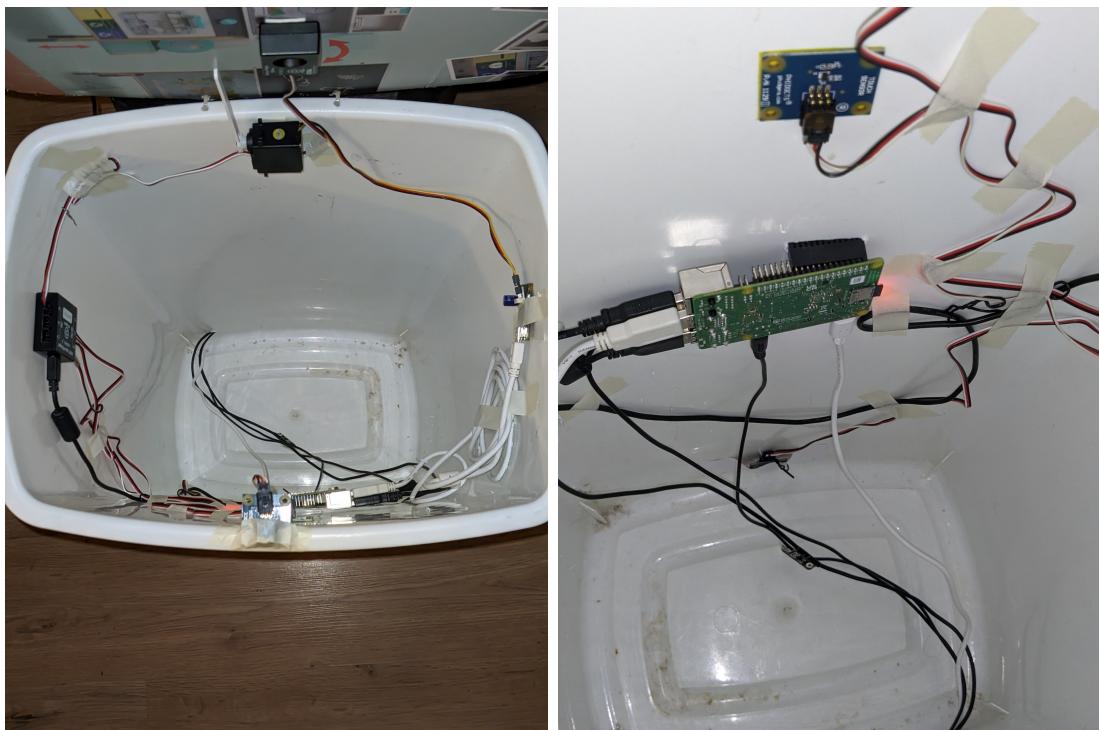


FIGURE 3 – Intérieur de Binny avec ses capteurs et son moteur montés



FIGURE 4 – Extérieur de Binny avec ses capteurs et son moteur montés

6 Technologies Utilisées

6.1 Scala et Akka (INFOM451)

La gestion des capteurs et des acteurs a été réalisée en **Scala**, notamment en utilisant le framework **Akka**. Nous avons pu ainsi implémenter toutes les exigences du cours, montrant notre compréhension du langage. Les capteurs faisant partie des équipements **Phidget**, il a été aisément de tous les connecter et de lire leurs données avec l'API fournie par le fabricant des capteurs.

Le projet Scala se divise en 3 sous-modules. Les acteurs, l'animation et le websocket.

6.1.1 Les acteurs

Nous utilisons une approche événementielle pour l'obtention et le monitorage des nouvelles données entrantes. Une fois un événement capturé, nous envoyons directement les informations dans une classe de partage pour que chaque acteur puisse accéder aux informations des autres acteurs en temps réel.

6.1.2 L'animation

Pour l'animation de notre poubelle intelligente avec l'écran LCD connecté au Raspberry Pi, nous utilisons JavaFX et sa couche de conversion en Scala, ScalaFX. L'animation sert principalement à humaniser la poubelle pour qu'elle soit plus attractive pour les enfants, mais aussi pour indiquer à l'utilisateur de la poubelle le niveau de remplissage de celle-ci grâce à une barre de progression qui se remplit au fur et à mesure du remplissage de la poubelle. La détection du remplissage est opérée par le capteur de distance qui mesure, à l'intérieur, la distance entre le couvercle et le fond de la poubelle.

6.1.3 Le websocket

Ce serveur sert principalement comme communication entre le projet Python et le projet Scala. Le projet Python envoie des messages via ce tunnel websocket au projet Scala pour indiquer à la poubelle quel couvercle ouvrir selon le besoin de l'utilisateur détecté par l'intelligence artificielle.

6.2 Python et Intelligence Artificielle (INFOM450, INFOM453)

Pour la partie intelligence artificielle, nous avons choisi **Python**. Ce choix s'explique par :

- La richesse de l'écosystème Python pour le développement d'algorithmes d'apprentissage automatique.

- La simplicité et la rapidité de prototypage qu'il offre, particulièrement adaptées au traitement d'images et au traitement du langage naturel (NLP).

De plus, nous utilisons le tout nouveau modèle de reconnaissance d'image YoloV11 pour la détection des objets présentés par l'utilisateur, ainsi que le nouveau modèle d'inférence de langage Llama 3.2 dans sa version 3B pour des performances d'inférence rapides et précises.

Pour ce qui est de la technologie de synthèse et de reconnaissance vocale, nous utilisons la reconnaissance vocale de Google, là encore, il est possible d'obtenir un modèle de reconnaissance vocale en local, mais le matériel ne nous permet pas de le faire à l'instant, et un modèle de synthèse de voix tiré du jeu "Portal" avec son antagoniste "GLaDOS". Le modèle de synthèse de texte et le modéliseur vocal ont été entraînés au préalable par [R2D2FISH](#) et sont exécutés par un serveur qui renvoie l'audio, une fois généré, à la poubelle pour la reproduction. Il y a aussi la détection de wake word qui est totalement en local et entraînée sur le modèle [OpenWakeWord](#) en utilisant un langage phonétique (hey beenn_i) pour générer quelques milliers de segments audios avec la prononciation du mot clé "Hey Binny" augmentés de bruits de fond divers et variés pour éviter les faux positifs et ensuite entraîner le modèle sur 50 000 epochs pour qu'il soit robuste.

6.3 Architecture REST Maillée

Nous avons implémenté une architecture **REST** très maillée avec **quatre serveurs** fonctionnant en parallèle. Cela offre plusieurs avantages :

- La possibilité d'utiliser la poubelle de manière locale.
- Une continuité de service même si certains serveurs sont hors ligne.
- La possibilité de changer de système d'entrée/sortie facilement.
- Scalabilité et maintenabilité du projet améliorées.

Voici un diagramme de séquence montrant comment le projet Scala communique avec le projet Python :

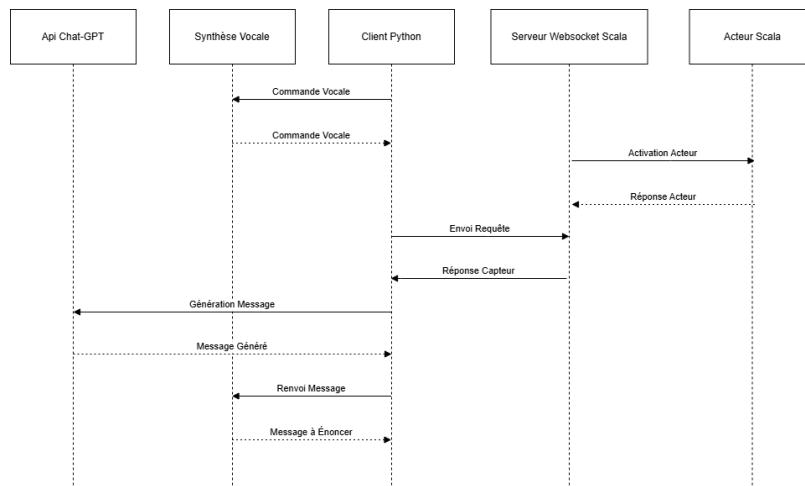


FIGURE 5 – Diagramme de séquence montrant la communication entre les différents serveurs

Le projet rencontre actuellement plusieurs problèmes :

1. Nécessité d'entraîner le modèle de synthèse vocale en français.
2. Problème pour l'arrêt de Binny lorsqu'elle parle.
3. Problème de feed-back de la poubelle dans le microphone lors de l'input de commandes utilisateur vocales.
4. Précision de la reconnaissance des objets tenus par l'utilisateur. Il nous manque de la puissance de calcul, il faudrait +70 heures de calcul d'une NVIDIA P100 pour faire un entraînement correct de 300 epochs. Et dans l'idéal, il faudrait +-1000 epochs d'entraînement pour avoir quelque chose de robuste (soit +-240 heures d'entraînement).
5. Temps de réponse trop long à cause de l'obsolescence du matériel actuel.
6. Les capteurs de distance et d'infrarouge ne fonctionnent pas sur des surfaces noires/sombres.
7. Les animations du visage de Binny sont très lentes à cause du taux de rafraîchissement de l'écran.

7 Évolutions possibles

Comme indiqué dans la section précédente, le projet a encore plusieurs problèmes, il faudrait donc commencer par les résoudre. Ensuite, une fois tous ces problèmes résolus, nous pouvons envisager les améliorations suivantes au projet :

- Ajout d'un bouton physique pour l'arrêt de la reproduction de l'audio.
- Ajout de boutons pour régler le volume de la poubelle.
- Création d'une application mobile pour gérer l'ouverture, la fermeture et le remplissage de la poubelle.
- Ajout d'un écran/visage pour Binny lorsqu'elle est pleine.
- Utilisation d'un écran plus grand et tactile pour l'implémentation d'une interface graphique greffée à la poubelle.

8 Conclusion

Le projet **Binny** a permis de combiner innovation technologique et impact écologique positif. En utilisant Scala pour la gestion des capteurs et Python pour l'intelligence artificielle, nous avons tiré parti des forces de ces technologies pour offrir une solution robuste et performante. Grâce à ses fonctionnalités avancées et son interactivité, Binny répond efficacement aux besoins du persona ciblé tout en encourageant les pratiques écoresponsables.