Dossier WORKSHOP 2025

Participants :

Tom CLEMENT
Clément SALINGUE
William COLLE
Leo BERGEAUD

Projet encadré par :

Gregory Boudringhin

Projet Workshop - Deuxième Année

EPSI — Ecole de l'ingénierie informatique, Arras

Contexte :

Le MI6 a perdu Q, le génie derrière les gadgets de James Bond. Résultat : plus de gadgets modernes, et Bond se retrouve avec du vieux matériel dépassé.
Pour réagir, le MI6 relance un laboratoire appelé Q-Lab Nouvelle Génération, où les nouvelles recrues (nous et notre équipe) doivent imaginer et concevoir des gadgets espions numériques.

Notre mission :

- Créer en équipe un prototype fonctionnel d'un gadget espion numérique.
- Ce gadget doit être réalisable avec les moyens du Q-Lab.
- Il doit avoir une fonction d'espionnage (discrétion, communication secrète, détection, diversion, etc.).

En résumé : nous sommes les nouveaux inventeurs du MI6, et nous devons prouver votre créativité en fabriquant un gadget espion numérique crédible.

Présentation du gadget :

Time2Spy

Time2Spy est un gadget d'espionnage discret, conçu sous la forme d'une montre, qui permet de rester en contact permanent avec le QG.

Fonctionnalités principales :

- Système de communication intégré : un micro miniature placé au centre des aiguilles permet de transmettre des messages directement au QG.

- Bouton d'alerte : en cas de danger ou de mission urgente, un simple appui envoie instantanément un signal d'alerte.

- Double LED de confirmation : deux voyants lumineux indiquent en temps réel si le QG peut ou non venir en aide.

Objectif :

Assurer une communication rapide, discrète et fiable entre l'agent sur le terrain et son QG, sans éveiller de soupçons.

Contexte réaliste :

James Bond infiltre une réception hautement surveillée. Time2Spy, dissimulée en montre, transmet discrètement un mot de code au QG grâce au micro au centre des aiguilles. Soupçonné, Bond dit des mots de codes comme oiseau , plante ou orange : les deux LEDs confirment la réception et déclenchent l'exfiltration. Signal discret, extraction réussie, preuve récupérée et mission accomplie.

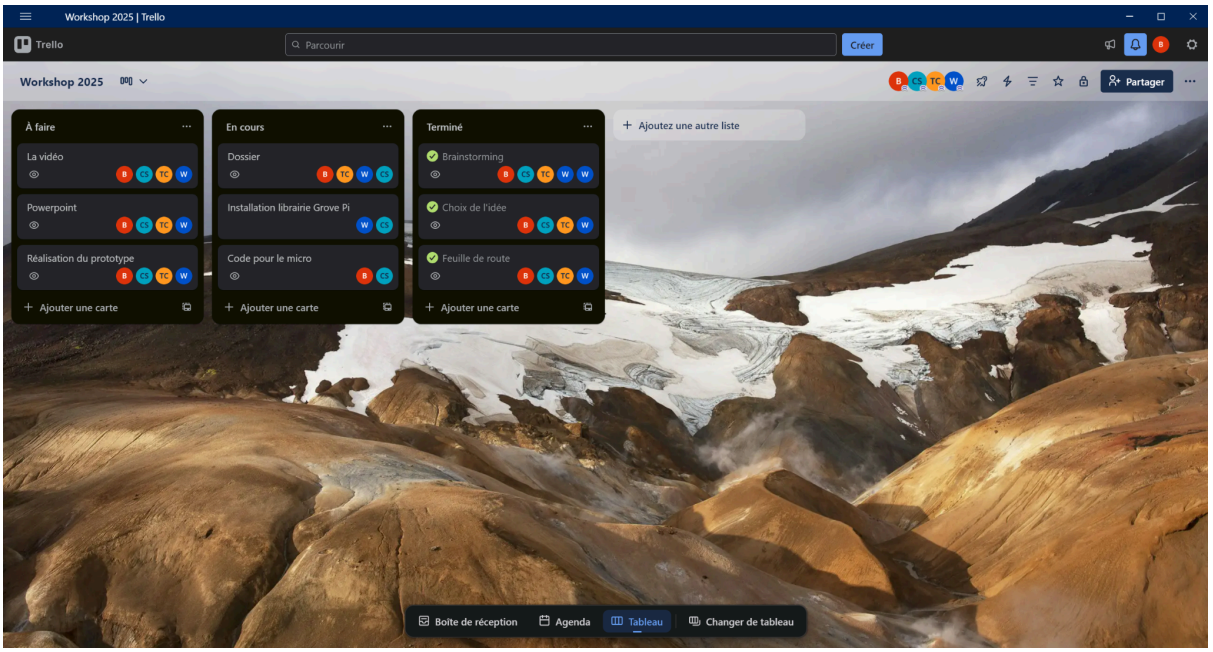
1. Planning de la semaine

Jours	**Tâches principales**
Lundi	**Brainstorming + choix de l'idée + feuille de route**
Mardi	**Début de la conception + code du micro + installation librairie Grove Pi + doc + Lié microSD et raspberry pi**
Mercredi	**Doc + initialisation du ppt + communication Raspberry pi et Grove pi**
Jeudi	**Codage des LED et Micro Raspberry + powerpoint + conception du gadget final**
Vendredi	**Passage à l'oral + doc**

Organisation de l'équipe :

TRELLO

Afin d’organiser plus facilement la réalisation de notre projet, nous avons décidé d’utiliser l’outil de gestion Trello. Nous avons donc créé différentes tâches, et chaque tâche a été attribuée à un ou plusieurs membres du groupe. Nous avons également séparé les tâches en trois catégories, la catégorie “A faire” qui, comme son nom l’indique, représente les tâches qu’il nous reste à commencer. Il y a ensuite les tâches “En cours” qui sont les tâches que nous sommes en train d’effectuer et qui peuvent aussi être peaufinés, et enfin, il reste les tâches “Terminées”, qui sont donc toutes les tâches que nous avons fini et nous n’avons plus besoin de nous y attarder.



2. Composants électroniques principaux

Matériel	Quantité	Utilité
Raspberry Pi 4 modèle B (4 ou 8 Go)	1	Ordinateur principal du projet
Carte microSD (32 Go)	1	Contient le système d’exploitation et les scripts

Boîtier Powerbank (ou batterie externe USB)	1	Alimentation portable et discrète dans le
Bouton v1.2	1	si l'agent ne peut pas parler il contact le QG avec le bouton
Microphone USB	1	Enregistre 5 secondes de son
Led socket Kit	2	Faire fonctionner les leds
Led Rouge / Vert	2	Interprétation du message

3. Accessoires pour les branchements

Matériel	Quantité	Utilité
Fils Dupont Mâle-Femelle	3	Pour relier le PIR au Raspberry Pi sans breadboard
Grove Pi	1	
câble usb / usb-c	1	Pour relier la batterie aux raspberry pi

4. Support physique (la montre)

Matériel	Quantité	Utilité
Une horloge (montre trop petite pour le prototype)	1	Cache l'ensemble du système (caméra, capteurs ou Pi)
Une ceinture	1	Représente les bracelets de la montre

5. Matériel pour la configuration initiale

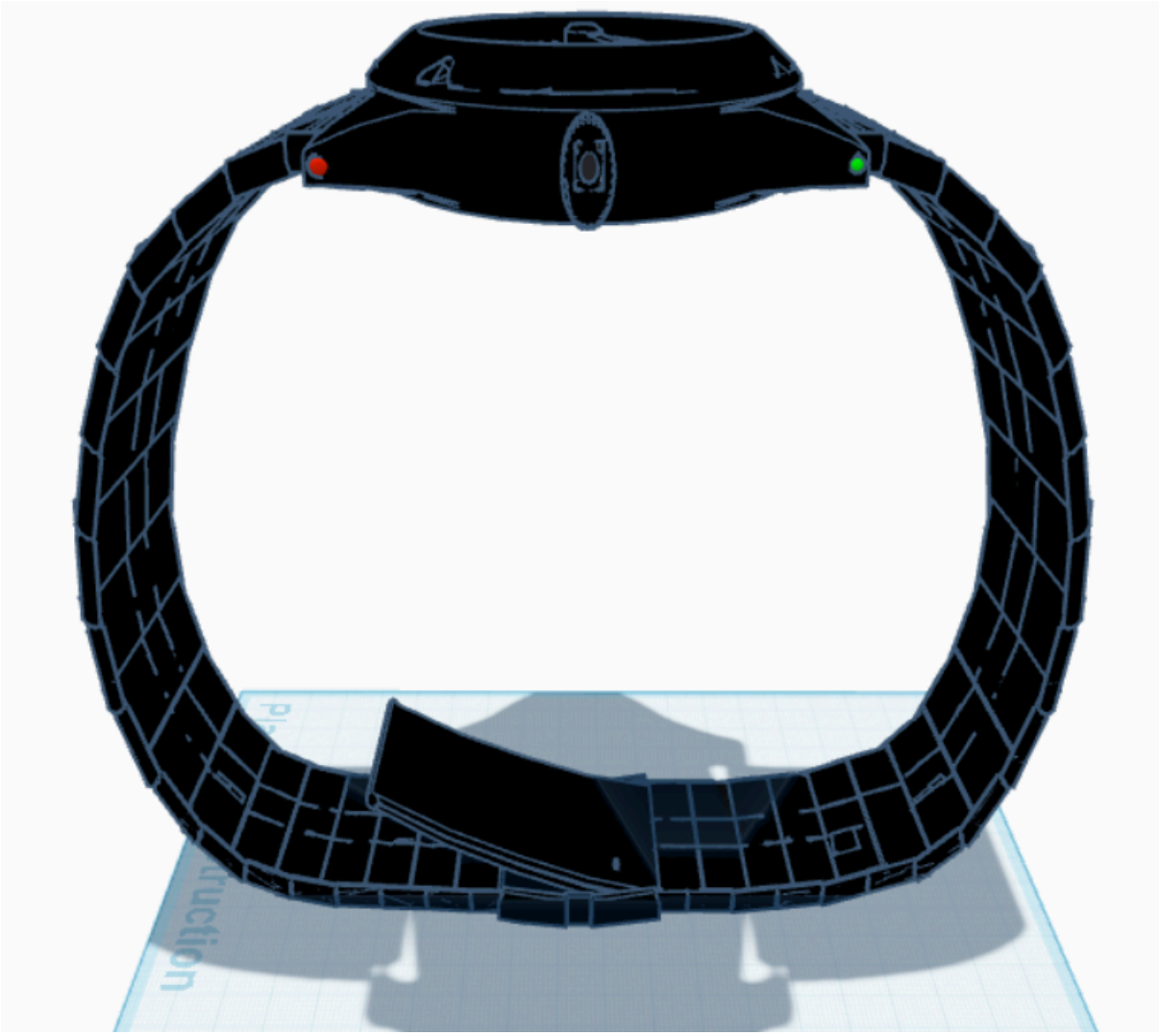
Matériel	Quantité	Utilité
PC portable	1	Pour préparer la carte microSD
Clé USB adaptable au carte microSD	1	Pouvoir une carte microSD sans port
Clavier + souris	1 de chaque	Pour contrôler le Pi lors de l'installation

6. Connexion réseau

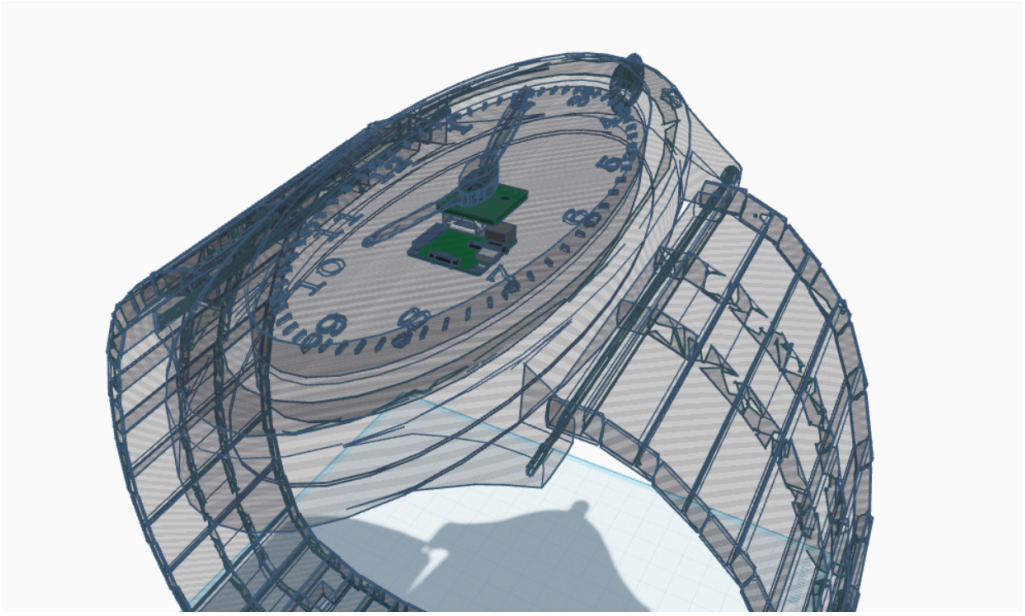
Matériel	Quantité	Utilité
Connexion WI-FI stable	1	Pour que le Pi puisse envoyer le message secret
Accès à un webhook discord ou telegram	1	Pour recevoir les alertes secrètes à distance

Maquette 3D

Conception d'une montre gadget en 3D.



Sur cette première image nous pouvons remarquer qu’il à était rajouter un deux led (rouge / vert). L’objectif est de reçevoir un message lumineux si les secours ou autres arrivent en fonction de la demande de james bond. De plus, un bouton a été ajouté en cas de situation ou james bond ne puissent pas parler.



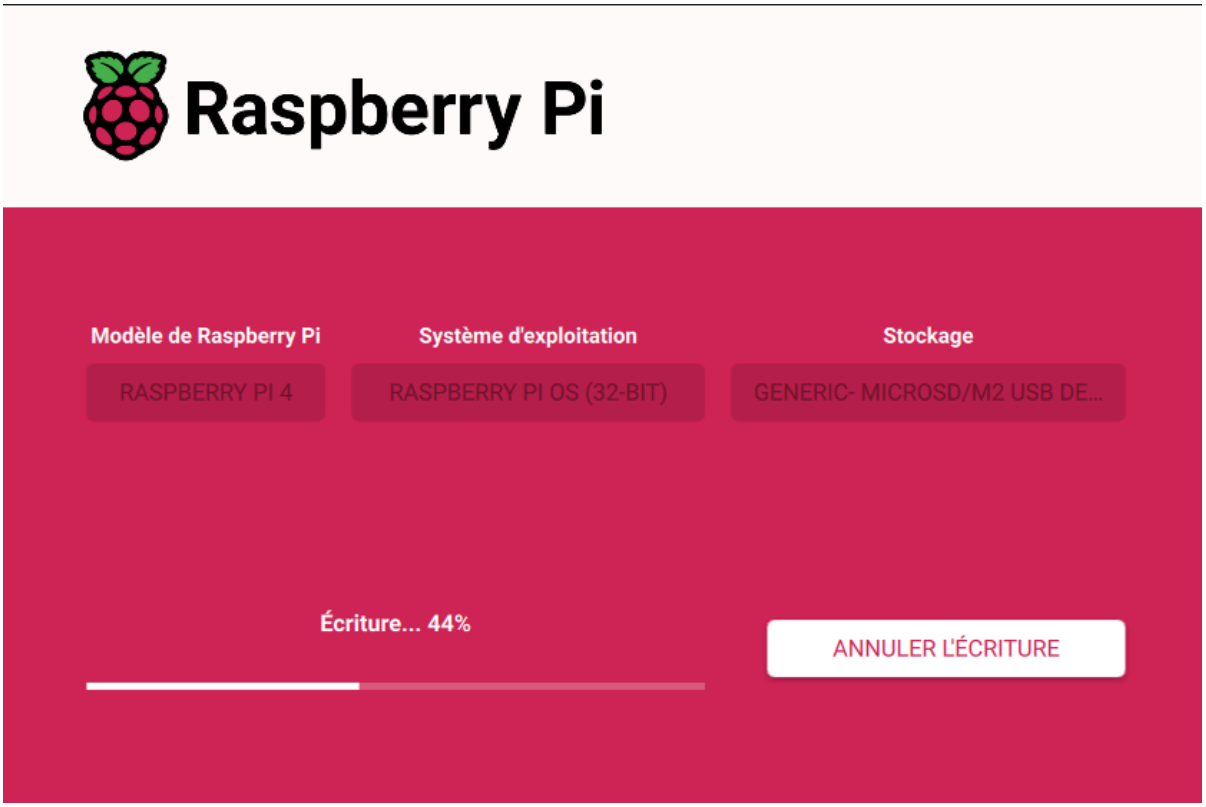
Sur cette seconde image, Nous apercevons au sein de la montre un système composé d’un Raspberry PI 4 modèle B et du micro intégré dans le centre des aiguilles pour obtenir la meilleure discrétion.

Comment formater la clé composé de la carte microSD :

Pour préparer la carte microSD il faut l'insérer dans une clé usb puis branchez cette clé dans un pc. Une fois cette étape réalisé sur le pc:

- télécharger – Raspberry Pi imager
- lancer Raspberry Pi imager
- Choisir L'iso dans "choose iso"
- Il est recommandé de prendre Raspberry Pi 0S 32
- choisir le storage et bien prendre celui de la carte microSD et non le disque sur le pc
- Dans les réglages il faut activer le SSH pour avoir un accès à distance
- personnalisé le nom d'utilisateur et le mot de passe
- puis faire modifier enregistrer

Après que les étapes sont finies, on obtient un téléchargement. Une fois qu'il est terminé nous pouvons débrancher la clé et remettre la microSD dans le Raspberry pi 4.



Explication du projet :

Le projet met en scène un agent secret, inspiré de James Bond, en mission sur le terrain. Sa mission nécessite de communiquer de manière sécurisée avec son quartier général (le QG), sans éveiller les soupçons. Pour cela, un système a été conçu permettant à l'agent, lorsqu'il est en difficulté, de transmettre un message codé. Ce message est envoyé discrètement, sans révéler son contenu à des observateurs extérieurs.

Ce message chiffré est ensuite reçu par le QG, où il est déchiffré par un opérateur à l'aide d'un mot de passe. Une fois le message validé, le QG peut répondre par une instruction à distance, en allumant une LED sur le Raspberry Pi de l'agent : une LED verte si une exfiltration est possible, ou une LED rouge si ce n'est pas le cas.

Le système repose sur plusieurs technologies complémentaires. D'abord, le Raspberry Pi sur le terrain est capable d'écouter en continu grâce à un micro. Il est programmé pour détecter des mots de code prédéfinis comme "oiseau", "plante" ou "orage". Lorsqu'un de ces mots est détecté, il est immédiatement chiffré à l'aide d'un algorithme AES combiné à un mot de passe secret. Le message chiffré est alors envoyé en TCP vers le PC du QG, qui agit comme serveur.

De son côté, le QG dispose d'une interface graphique développée avec Tkinter. Lorsqu'un message chiffré arrive, un bip est joué pour alerter l'opérateur, et celui-ci est invité à entrer le mot de passe correspondant pour tenter de déchiffrer le message. Si le mot de passe est correct, le QG affiche le message réel – par exemple "J'ai besoin d'aide" – ainsi que le lieu où se trouve l'agent. L'opérateur peut alors décider s'il est possible ou non d'exfiltrer l'agent. Selon la décision, une instruction est envoyée au Raspberry Pi : soit pour allumer une LED verte, soit pour allumer une LED rouge.

L'interface du QG intègre également une carte visuelle qui représente le plan du site, dans un style rappelant le jeu Cluedo. Cette carte affiche les différentes salles, la position actuelle de l'agent ainsi que celle des ennemis, qui sont générées aléatoirement pour chaque message. De plus, tous les messages traités sont enregistrés dans un fichier journal (log.txt), afin de garder une trace des échanges et

décisions. Si un mauvais mot de passe est entré, le système affiche un faux message appelé leurre, comme "Le colis est sécurisé" ou "Zone sans incident", afin de ne pas éveiller les soupçons.

Voici un exemple de scénario complet : l'agent Bond, équipé du Raspberry, prononce discrètement le mot "oiseau". Le système sur le Raspberry chiffre ce mot à l'aide du mot de passe "bleu123", puis l'envoie automatiquement au QG sur l'adresse IP du serveur, via le port 5050. Dès réception, le QG joue un son pour prévenir l'opérateur. Celui-ci entre le mot de passe "bleu123", permettant de déchiffrer le message et d'afficher "J'ai besoin d'aide" ainsi que le lieu correspondant, ici "Salle 5". L'opérateur analyse alors la situation sur la carte, et choisit d'autoriser l'exfiltration. En réponse, une LED verte s'allume sur le Raspberry Pi, signalant à l'agent que le feu vert est donné. Si au contraire, la situation ne le permet pas, une LED rouge s'allume à la place.

Ce projet est donc une simulation immersive et complète, combinant reconnaissance vocale, cryptographie, communication réseau, interface graphique, interaction physique via les LEDs, et scénarisation digne d'un film d'espionnage. Il illustre à la fois des compétences techniques variées et une bonne coordination entre le monde virtuel et réel.

Explication du code :

Le code principal se trouve du côté du QG (PC). Il s'agit d'un programme Python qui fait plusieurs choses à la fois : recevoir un message, le déchiffrer, afficher une interface graphique et envoyer une réponse au Raspberry Pi.

1) Réception des messages du terrain

La fonction suivante démarre un serveur TCP sur le QG :

```
def start_server():
    HOST = "192.168.137.1"
    PORT = 5050
    ...
    s.bind((HOST, PORT))
    s.listen()
```

Elle attend qu'un message arrive depuis le Raspberry sur le port 5050. Dès qu'un message arrive, il est affiché dans la console et stocké pour être déchiffré.

2) Déchiffrement du message

Quand l'utilisateur tape un mot de passe dans l'interface et clique sur "Déchiffrer", on appelle :

```
def traiter_dechiffrement():
    mot_code = dechiffrer_message(dernier_message_chiffre, mdp)
```

Cette fonction vérifie si le mot de passe est bon en essayant de décrypter le message chiffré avec AES. Si c'est bon, on obtient un mot-clé (comme `"oiseau"`) qui est associé à un vrai message ("J'ai besoin d'aide") et un lieu ("Salle 5").

3) Interface Cluedo (Tkinter)

Grâce à la fonction `dessiner_plan()`, on crée un plan graphique des salles du bâtiment. Il s'agit d'un petit plan style Cluedo, dessiné avec des rectangles dans un `Canvas`.

On y affiche aussi :

- Le lieu de l'agent (en bleu)
- Les ennemis (des "X" rouges)
- Le message déchiffré dans un champ texte

4) Réponse par LED sur le Raspberry

Une fois le message affiché, on demande si l’exfiltration est possible :

```
def afficher_choix_led():
    ...
    envoyer_instruction_led("vert")
```

En cliquant sur "Oui" ou "Non", le QG envoie une instruction réseau au Raspberry sur un autre port (6060) pour lui dire d’allumer une LED verte ou rouge. Le Raspberry allume la LED grâce à ses ports GPIO.

5) Sécurité et log

Le programme enregistre tout dans un fichier `log.txt` :

- le message chiffré
- s’il a été déchiffré
- la réponse choisie

Et si le mot de passe est faux, il affiche un message leurre (ex. : "Le colis est sécurisé.") pour protéger la vraie mission.

L'AES (Advanced Encryption Standard) est un système qui permet de rendre un message illisible sans le bon mot de passe. On peut l’imaginer comme un coffre-fort numérique : on y place un mot ou une phrase importante, puis on le verrouille avec un mot de passe. Par exemple, si l’agent dit "oiseau", ce mot est chiffré avec un mot de passe comme "bleu123".

Le résultat devient un code totalement incompréhensible, tel que (nT4uWE==), à moins d'avoir exactement le bon mot de passe pour le décrypter. Ce chiffrement est utilisé dans le projet pour que même si le message est intercepté, il soit impossible à comprendre sans autorisation.

L'interface graphique du QG est créée avec Tkinter, un outil de Python qui permet d'afficher une vraie fenêtre, avec des boutons, des champs de texte, et même des dessins. Grâce à Tkinter, le QG peut facilement voir les messages reçus, entrer le mot de passe pour les déchiffrer, et décider s'il faut autoriser l'extraction de l'agent. L'affichage est clair, agréable, et comprend un plan des salles stylisé, comme dans un jeu d'enquête.

Pour que le Raspberry Pi et le QG puissent communiquer à distance, le projet utilise ce qu'on appelle des sockets TCP. Cela permet à deux ordinateurs de s'envoyer des messages par le réseau, un peu comme s'ils avaient chacun une boîte aux lettres numérique. Le Raspberry envoie son message vers une adresse IP fixe du QG sur un port spécifique (le port 5050), et le QG écoute ce port pour recevoir les messages. Ensuite, il peut répondre à son tour au Raspberry via un autre port (6060), pour lui donner une instruction.

Enfin, le Raspberry Pi joue le rôle de l'agent sur le terrain. Il écoute ce que dit l'agent via le micro, et dès qu'il reconnaît un mot de code important comme "oiseau", il le chiffre automatiquement avec AES. Ce message codé est ensuite envoyé au QG, de manière totalement discrète et sécurisée. Une fois le message reçu, le QG demande à l'utilisateur de taper le bon mot de passe. Si celui-ci est correct, le message est déchiffré et affiché, et le QG peut alors décider s'il faut extraire l'agent. Si l'exfiltration est possible, une LED verte est allumée à distance sur le Raspberry. Si ce n'est pas le moment d'agir, c'est une LED rouge qui s'allume. Cela permet à l'agent de recevoir une réponse claire et silencieuse, uniquement par une lumière.

PROBLÈMES RENCONTRÉS

- perte de temps en raison des compétences

  Pendant le développement du projet, plusieurs difficultés ont été rencontrées, qui ont occasionné une perte de temps significative. L'une des premières sources de ralentissement a été liée au niveau de compétence technique initial. Certains aspects du projet, notamment la communication réseau entre machines, la manipulation des modules de cryptographie ou encore l'utilisation de bibliothèques spécifiques comme Tkinter ou `socket`, ont nécessité un temps d'apprentissage et de recherche non négligeable. Cette montée en compétence, bien que bénéfique à long terme, a ralenti l'avancement global lors des premières étapes du projet.

- problème de connexion

  Par ailleurs, des problèmes de connexion ont fréquemment perturbé les tests, notamment entre le Raspberry Pi et le poste du QG. Il a parfois été difficile d'établir une communication stable en TCP, que ce soit à cause de la configuration réseau, ou encore des restrictions d'accès aux ports. Le Raspberry ne disposant pas toujours d'une IP fixe, il a fallu à plusieurs reprises reconfigurer les paramètres pour garantir que les messages soient bien envoyés et reçus.

- adaptation avec le matériel du Q-lab

L'adaptation au matériel fourni par le Q-lab a également été un défi. Chaque Raspberry Pi n'était pas forcément configuré de la même manière, certains n'ayant pas les bibliothèques nécessaires ou n'étant pas à jour. Il a fallu ajuster les scripts, vérifier la compatibilité des GPIO avec les breadboards, et parfois réinstaller certains composants ou logiciels. La configuration de la LED via les ports GPIO a demandé plusieurs essais pour s'assurer que le signal reçu soit correctement interprété par le matériel.

- difficulté dans le code pour communiquer avec le raspberry et les breadboard

Enfin, la communication entre le code Python, le Raspberry Pi et la breadboard a représenté une difficulté technique importante. Transmettre une instruction depuis le PC du QG jusqu'à une LED connectée à un GPIO du Raspberry, en passant par une couche réseau sécurisée, a demandé une coordination très fine entre les différents scripts. Il fallait s'assurer que le Raspberry soit bien en écoute, que les ports ne soient pas bloqués, et que les scripts LED puissent être déclenchés correctement sans surcharger le système. Tous ces éléments, bien que surmontables, ont représenté un véritable défi technique et logistique qui a demandé de la rigueur, de la patience et une bonne répartition des tâches.

- Grove pi

  Nous avons essayé d'utiliser la carte GrovePi pour connecter facilement des capteurs au Raspberry Pi. Le GrovePi fonctionne normalement en communiquant via le bus I²C et nécessite aussi une mise à jour de son firmware avec l'outil **`avrdude`**. Cependant, nous avons rencontré plusieurs difficultés techniques. Tout d'abord, les scripts fournis par le constructeur ont été écrits pour d'anciennes versions du système Raspberry Pi OS et beaucoup de dépendances utilisées à l'époque n'existent plus dans la version actuelle. Cela a provoqué de nombreuses erreurs lors de l'installation. Nous avons aussi eu des soucis avec **`avrdude`**, l'outil qui sert à mettre à jour le microcontrôleur du GrovePi, car la version installée par le script était trop ancienne et incompatible. Même après avoir installé une version plus récente, la communication avec le GrovePi ne fonctionnait pas car le port série nécessaire n'était pas activé sur notre Raspberry Pi. Enfin, certaines bibliothèques Python indispensables, comme **`di_i2c`**, n'étaient pas installées par défaut et leur installation échouait parfois à cause de restrictions du nouveau système.

AMELIORATIONS FUTURES

- Mise en place d’un bouton sur le côté de la montre. L’objectif est de permettre à James Bond de communiquer avec le QG si il n’a pas la possibilité de parler dans sa situation.
- Réalisé au sein de la montre un dispositif de traçage GPS pour localiser James Bond et rendre l’extraction de la meilleure des façons en terme de rapidité et simplicité.
- Conception d’une communication entre l’agent et le QG à travers un appel en direct afin de faciliter l’échange d’informations entre les deux

CONCLUSION

Ce projet mêle technologie, sécurité et interaction physique pour recréer une situation réaliste d’espionnage. Il met en œuvre des compétences variées comme la détection vocale, la cryptographie, la programmation réseau et la gestion d’interfaces graphiques et de composants électroniques.

Le Raspberry Pi, qui joue le rôle de l’agent sur le terrain, est capable de détecter des mots-clés codés via la voix, de les chiffrer de façon sécurisée avec un mot de passe, puis de les envoyer automatiquement au QG (le PC) grâce à une connexion réseau.

De son côté, le QG déchiffre le message via une interface graphique intuitive, inspirée des systèmes de surveillance, et permet à un opérateur de décider s’il faut intervenir ou non. Cette décision est renvoyée au Raspberry sous forme d’un simple signal lumineux : une LED verte pour une extraction possible, ou rouge pour un danger imminent.

Finalement, ce projet démontre comment différents domaines — audio, cryptographie, électronique, interface utilisateur et réseau — peuvent être combinés pour créer un système sécurisé, discret et interactif. Il permet une communication silencieuse mais efficace entre un agent en mission et son quartier général, le tout en restant simple d'utilisation et techniquement solide.