

IMT Atlantique a
Projet PRONTO – S6 2024-2025
Campus de Brest



Livrable n°3 : Rapport final PRONTO

PROJET PRONTO n°84 : PULSE LIGHT

Auteur 1 : Justine BRISSART

Auteur 2 : Sara EL BARI

Auteur 3 : Niamatellah LAHKIM

Auteur 4 : Nihal LACHGUER

Auteur 5 : Justine BRISSART

Auteur 6 : Judith MONCHAUX



PULSE LIGHT

Encadré par : Sylvie KEROUEDAN

30/04/2025

Table des matières

Résumé	1
Introduction	1
1 Conception	2
a. Rappel des fonctionnalités attendues (cahier des charges fonctionnel)	2
b. Choix de conception	6
c. Relations entre les différentes sections	8
2 Réalisation	9
a. Choix des composants	9
b. Tests unitaires de chaque composant	9
c. Segmentation du code des différents modes	12
d. Tests unitaires du code et des modes	14
e. Assemblage des composants	15
f. Développement de l'interface administrateur	17
3 Intégration et validation	18
a. Assemblage physique final (circuit électronique + boîte)	18
b. Relation entre interfaces virtuelle et physique : Exploitation des données	20
c. Relation entre interfaces virtuelle et physique : Commande à distance	21
d. Validation de l'ensemble	23
4 Conclusion et perspectives	23
a. Livrables scientifiques attendus	23
b. Validation des scénarios	24
c. Axes d'amélioration	26
d. Retours d'expériences du projet	27
5 Références bibliographiques	29
6 Annexe	30
a. Retour d'expérience - Judith	30
b. Retour d'expérience - Tinhinen	31
c. Retour d'expérience - Justine	32
d. Retour d'expérience - Sara	33
e. Retour d'expérience - Niamatallah	33
f. Retour d'expérience - Nihal	34

Résumé

Objectif du document : Rendre compte de l'état final du projet selon différents axes : technique et organisationnel

Le projet PulseLight est un projet d'ingénierie, ayant pour but un suivi simple et intuitif de différentes capacités. Nous avons donc cherché à concevoir un jeu à mode, afin d'analyser les capacités de réflexe et de mémoire des joueurs. Pour cela, nous avons :

- Conçu un **circuit électronique**, composé de boutons
- Ecrit le **code C++** pour différents modes
- Développé une **interface html** pour l'administrateur
- Conçu une **boîte de jeu compacte**
- Intégré un **capteur de rythme cardiaque**

Nous avons obtenu trois modes de jeu fonctionnels, ainsi qu'une boîte de jeu, et un système de jeu électronique. L'interface administrateur nous permet de communiquer des commandes à distance, afin de lancer un mode de jeu particulier, et de récupérer les données enregistrées par notre dispositif.

Dans ce document, nous allons donc rendre compte de l'avancée du projet, ainsi que détailler chacune des étapes que nous avons suivies durant toute la durée du projet PulseLight. Nous allons donc pouvoir approfondir les étapes de Conception, Réalisation et d'Intégration du projet, afin de conclure sur l'état final de notre dispositif, et la validation ou non de nos objectifs.

Introduction

Nous avons mené durant 4 mois un projet d'ingénierie, dont le but final était d'**obtenir un outil de suivi de patients par leurs médecins, sur différentes capacités, comme la mémoire ou les réflexes**. C'est donc un outil qui s'adresse initialement à des organismes médicaux, tels que des maisons de retraite, des hôpitaux ou encore des cliniques. L'utilisateur cible est donc plutôt une personne âgée, qui cherche à maintenir ou entraîner ces capacités. Nous avons également envisagé les personnes victimes d'AVC comme utilisateur cible potentiel.

Afin de pouvoir assurer un suivi simple à mettre en place, notre dispositif se doit d'être **intuitif** et **compact**. Pour que le suivi soit également agréable aux utilisateurs, nous avons cherché à rendre le jeu PulseLight ludique et **simple** à jouer. Ainsi, le projet PulseLight répond à la problématique suivante :

Comment concevoir un outil de suivi de différentes capacités mentales (réflexe, mémoire) simple d'utilisation pour un usager âgé ?

Annonce du plan :

Tout d'abord, nous reviendrons sur les différentes étapes de **conception** du projet. Nos choix de conception nous ont amenés à segmenter le projet, il a donc été nécessaire de réfléchir aux relations entre les différentes sections du projet.

Ensuite, nous détaillerons chaque choix effectué lors de l'étape de **réalisation** du projet, selon les différentes parties. Nous pourrons donc expliquer plus en profondeur les parcours séparés de chaque aspect du projet, de l'électronique à l'interface administrateur, en passant par le code des modes.

Puis, une fois chaque section détaillée séparément, nous allons revenir sur **l'intégration** de tous les différents éléments ensemble. Nous pourrons ainsi valider chacune des relations essentielles du projet, entre les différents aspects : matériel, informatique, électronique.

Finalement, nous **conclurons** sur les résultats de notre projet. Nous reviendrons sur nos objectifs, afin de vérifier si ils ont bien été atteints. Pour cela, nous ferons donc un bilan des différents attendus initiaux, ainsi que des étapes validées précédemment. Nous ferons également un bilan général du projet, pour en déduire des axes d'amélioration, mais aussi pour faire notre retour d'expérience.

1 Conception

a. Rappel des fonctionnalités attendues (cahier des charges fonctionnel)

FP : Fonction Principale

FS : Fonction Secondaire

FC : Fonction de Contrainte

Objectif général du système:

Créer un dispositif médical interactif permettant d'évaluer et d'améliorer les capacités cognitives et motrices telles que les réflexes, la mémoire, la précision et l'acuité auditive des utilisateurs, tout en collectant leurs données physiologiques pour une utilisation ultérieure par les professionnels de santé.

Public cible:

- Personnes âgées ou victimes d'AVC
- Administrateurs (médecins, infirmiers)

1) Interface Joueur

Numéro des fonctions	Fonction	Condition de validité	Niveaux	Priorité
FC.1	Prise en main rapide	- Navigation simple et intuitive	Compréhensible, utilisable par tout type de personne	1
FC.2	Lisibilité de l'interface	- Ecran clair/visible - Boutons colorés identifiables facilement	Visible par une personne non daltonienne, non malvoyante	1
FC.3	Générer des feedback visuel instantanés	- Ecran LCD avec des leds réactives qui marchent bien	Temps d'affichage des couleurs lors d'une partie (<1s) et des scores (<5s)	1
FS.1	Générer des feedback sonores instantanés	- Génération de son rapide et réactive	Temps de génération des sons lors d'une partie (<2s)	3
FC.4	Confort et réduction du stress pendant le jeu	- Boutons larges, bien colorés, bien espacés. - Temps de réponse suffisant - Possibilité de répéter les instructions/de revenir en arrière.	Utilisation agréable pour tout type de personne (étudiant, adulte, personne âgée) Temps de réponse : le jeu se poursuit seulement quand le patient a appuyé sur un bouton en tant que réponse	2
FP.1	Afficher les résultats	- Scores lisibles et clairs sur écran	Lecture facile des résultats, par une personne non daltonienne et non malvoyante	1
FC.5	Aspect esthétique/attractive	- Couleurs apaisantes et différentes - Contraste adapté - Police bien lisible - Interface non surchargée	Appréciation de l'attractivité esthétique du jeu par une personne extérieure	2

2) Interface Commandes et Données:

Numéro	Fonction	Condition de validité	Niveaux	Priorité
FP.1	Être capable de changer de mode	Facilité du changement de mode	Temps de changement de mode < 3 min	1
FP.2	Restituer la fréquence cardiaque au cours des mesures	Affichage de la fréquence cardiaque du joueur sur l'interface	Obtenir des valeurs comprises entre 40 et 200 BPM (valeurs normales) Histogramme de l'évolution de la FC	Sur ordi : priorité 1 Sur écran : priorité 2
FS.2	Restituer le niveau de stress du joueur	Affichage du niveau de stress du joueur sur l'interface	Histogramme de l'évolution du niveau de stress	-Sur ordi : priorité 2 -Sur écran : priorité 2
FP.3	Restituer le temps de réponse du joueur	Affichage du score du joueur avec le temps de réaction	Histogramme de l'évolution du temps de réponse.	1
FS.4	Identifier l'administrateur	Il faut que l'administrateur puisse se connecter avec son identifiant avant avoir accès aux données	Identifiant et mot de passe	2

3) Modes du Jeu

Le jeu doit proposer plusieurs modes permettant de tester différentes capacités cognitives et réflexes des utilisateurs

	Fonction	Niveaux	Priorité
FP.4	Tester la capacité de mémorisation. <i>L'utilisateur doit reproduire une séquence lumineuse affichée sur les leds autour de l'écran.</i>		
FP.4.1	Les couleurs s'affichent dans une séquence aléatoire	Temps d'affichage de chaque couleur : 1s	1

FP4.2	La difficulté augmente progressivement	Affichage de 5 couleurs puis 7 puis 10	1
FP4.3	Les erreurs sont gérées	si erreur: le joueur peut continuer d'appuyer sur les boutons jusqu'à ce qu'il atteigne le nombre maximal de boutons (10 sur une séquence de 10 couleurs à mémoriser)	1
FP.5 Tester les réflexes du joueur			
<i>L'utilisateur doit réagir le plus rapidement possible aux stimuli lumineux</i>			
FP5.1	Évaluer le temps de réaction	Mesure du temps de réaction à 0.1 s près	1
FP5.2	Affichage de plus en plus rapide des stimuli lumineux	Affichage du stimulus lumineux: 1; 0,75; 0,5; 0,25 s	1
FP5.3	Les erreurs sont gérées	Si erreur : réaffichage de la couleur pendant 0,5 s Affichage du nombre d'erreurs	1
FS3 Tester la précision du joueur			
<i>Appuyer sur le bon bouton parmi plusieurs avec précision</i>			
FS3.1	Chaque partie a une durée définie	partie de 15 indications de boutons	3
FS3.2	L'écran indique le bouton sur lequel appuyer	numéro de boutons/couleurs (ex: bleu 4, rouge 3, vert 2, orange 1)	3
FS3.3	Les erreurs sont gérées	Si erreur: affichage sur l'écran du bouton sur lequel il faut appuyer Récupération du nombre d'erreurs	3
FS4 Tester la réaction au son du joueur			
<i>Appuyer sur le bouton désigné par un certain signal sonore</i>			
FS4.1	Un son indiquant que la couleur est générée	Prononciation correcte de la couleur	3
FS4.2	Le temps de réaction est mesuré	Mesure du temps de réaction à 0.01 seconde près	3

		si erreur: répétition de la génération du son nombre	
FS4.3	Les erreurs sont gérées	si erreur: répétition de la génération du son récupération du nombre d'erreurs	3

4) Capteurs et collecte des données physiologiques:

- **Capteur fréquence cardiaque** : Plage mesuré (40-200 BPM).
- **Stockage et restitution des données** : Sur ordinateur.

b. Choix de conception

1) Choix techniques

A. Microcontrôleur : Arduino UNO R3

Nous avons choisi l'Arduino UNO R3, facile à utiliser et compatible à la majorité des appareils électroniques, contrairement à la R4. En plus, ses nombreuses broches sont disponibles pour connecter les boutons, LEDs, écran LCD, capteurs physiologiques (fréquence cardiaque), et le buzzer, tout en assurant une gestion efficace des différents modes de jeux et du stockage temporaire des données.

B. Capteurs physiologiques:

- Capteur de fréquence cardiaque (Pulse Sensor Amped) :

Nous avons choisi le Pulse Sensor Amped pour sa simplicité d'intégration avec les cartes Arduino et sa capacité à fournir un signal analogique clair et exploitable, et enfin, sa précision reconnue dans la mesure des battements par minute (40-200 BPM). Ce capteur répond parfaitement à nos contraintes d'encombrement avec un diamètre de 16 mm et une épaisseur 3 mm. Son faible encombrement et sa faible consommation sont des atouts pour une intégration compacte dans notre boîtier.

C. Module sonore : Buzzer piézoélectrique

Nous avons choisi le buzzer piézoélectrique pour sa capacité à émettre facilement des mélodies simples depuis un fichier de notes. Il nous permet de fournir des stimuli auditifs compréhensibles par l'utilisateur et de l'informer de l'avancée de sa partie.

D. Écran LCD

On a choisi un écran LCD I2C assurant une lisibilité optimale des consignes, des stimuli visuels et des résultats, facilitant une interaction pour les utilisateurs de tout âge ou ayant subi un AVC. De

plus, contrairement à un écran LCD classique, il peut être utilisé avec seulement 4 fils à brancher, ce qui représente un avantage dans notre projet contenant déjà de nombreux composants dont 16 boutons et 16 résistances.

E. Stockage des données:

Pour un stockage et une analyse approfondie des données à long terme, les informations sont transférées périodiquement sur un ordinateur via une connexion câblée (USB).

2) Architectures techniques

A. Architecture générale du système:

Le système PulseLight adopte une architecture organisée autour du microcontrôleur Arduino UNO R3. Les entrées (boutons, capteurs) et les sorties (LEDs, écran LCD, haut-parleur via DFPlayer Mini) sont indirectement connectés avec le microcontrôleur via des platines d'essai, facilitant la synchronisation rapide entre la mesure, la réaction du joueur et le feedback visuel et auditif. Les données enregistrées sont ensuite transférées vers un ordinateur pour stockage durable et analyse complémentaire.

➤ Circuit imprimé personnalisé (PCB):

Un circuit PCB sera réalisé pour optimiser la compacité du dispositif, il intégrera toutes les composantes générales (boutons, LEDs, capteurs) en minimisant les câblages externes, ce qui augmente la robustesse du système. De plus, ce dernier sera composé d'une carte centrale et de 4 cartes contenant chacune 4 boutons : cela permet d'adapter facilement la position des boutons sur le boîtier.

➤ Boîtier en bois:

Bien que l'impression 3D offre une grande personnalisation, nous avons choisi un boîtier en bois afin de faciliter les ajustements physiques lors du prototypage. Le bois est plus facile à percer ou modifier à la main, ce qui permet d'ajouter ou déplacer des composants (LEDs, capteurs, boutons) rapidement en fonction des besoins. Il offre également une structure rigide et résistante aux chocs.

3) Justification des choix

Notre dispositif a comme pour but principal d'assurer ces objectifs principaux tout en anticipant les besoins d'utilisateurs et patients dans l'environnement médical:

Ainsi, nos choix techniques ont été réalisés afin de répondre aux aspects primordiaux suivants :

- **Polyvalence et portabilité**
- **Précision et fiabilité des données**
- **Interaction utilisateur optimisée et confortable**
- **Durabilité et robustesse** : par la réalisation PCB personnalisé

c. Relations entre les différentes sections

Nous avons choisi de diviser notre projet en quatre sections principales :

- électronique
- code des modes
- code de l'interface administrateur
- conception du boîtier

Afin d'assurer que toutes les sections du projet étaient compatibles entre elles, il nous a été nécessaire de vérifier les différentes relations individuellement.

1) Relation électronique - code des modes

Il s'agit de la première relation à avoir été mise à l'épreuve, car nous avons dû nous assurer de la cohérence des deux parties dès le début du projet. Pour effectuer le moindre test sur le code des modes, il était nécessaire d'avoir un circuit fonctionnel et compatible le plus vite possible.

Dans un premier temps, pour accélérer les étapes de tests unitaires, nous avons pu **mettre en commun les tests unitaires** de code pour chacun des composants, afin de vérifier à la fois leur fonctionnement, mais aussi les fonctions codées basiques, utilisant ces composants.

Il a été indispensable de mettre au point un **document Entrées - Sorties**, afin de pouvoir assurer la cohérence des PIN d'entrée et de sortie utilisées par le code, et ceux utilisés réellement par le circuit électronique, que ce soit pour le capteur, les LEDs, l'écran ou les boutons.

Nous avons également dû choisir en accord un système de branchement pour les **registres**. Nous avons finalement opté pour les brancher **en série**, afin de faciliter le codage. Nous avons également dû **communiquer la position de chaque bouton**, et sa couleur, dans le circuit, pour pouvoir les associer correctement à leur équivalent dans le codage des modes.

2) Relation code des modes - interface administrateur

L'interface administrateur propose deux fonctionnalités : elle nous permet de récupérer et d'exploiter les données, et elle nous permet d'opérer le dispositif à distance à l'aide de commande.

- Exploitation des données : il a fallu choisir un **format type d'enregistrement des données**, afin qu'elles puissent correctement être exploitées
- Commande à distance : il a fallu **décider d'un système de commande à distance**, pour segmenter le code de manière appropriée

3) Relation interface administrateur - électronique

Le choix des composants a été crucial pour la relation entre l'interface administrateur et le circuit électronique : c'est ce choix qui a décidé du mode de communication entre le dispositif et l'interface. En l'occurrence, nous avons opté pour une **carte Arduino R3**, pour privilégier une compatibilité avec le plus d'utilisateurs possibles.

4) Relation boîte - électronique

Afin de concevoir la **boîte** qui accueille le dispositif, il a été indispensable de connaître les **dimensions exactes du circuit**, afin d'être sûr que la boîte serait assez grande pour contenir le circuit entier, mais aussi qu'elle serait la plus petite possible, dans le but d'obtenir un dispositif final le plus compact possible.

De plus, de nombreuses **ouvertures** doivent être présentes dans la boîte, pour que le jeu fonctionne correctement. Il faut ainsi prévoir des emplacements pour les boutons, l'écran et les leds, afin que l'interface de l'utilisateur soit agréable et pratique. Mais nous avons également dû ajouter des ouvertures pour avoir accès à la carte Arduino : pour l'**USB**, pour pouvoir lui envoyer des codes, et également pour **son alimentation**, pour qu'elle fonctionne sans branchement à un ordinateur.

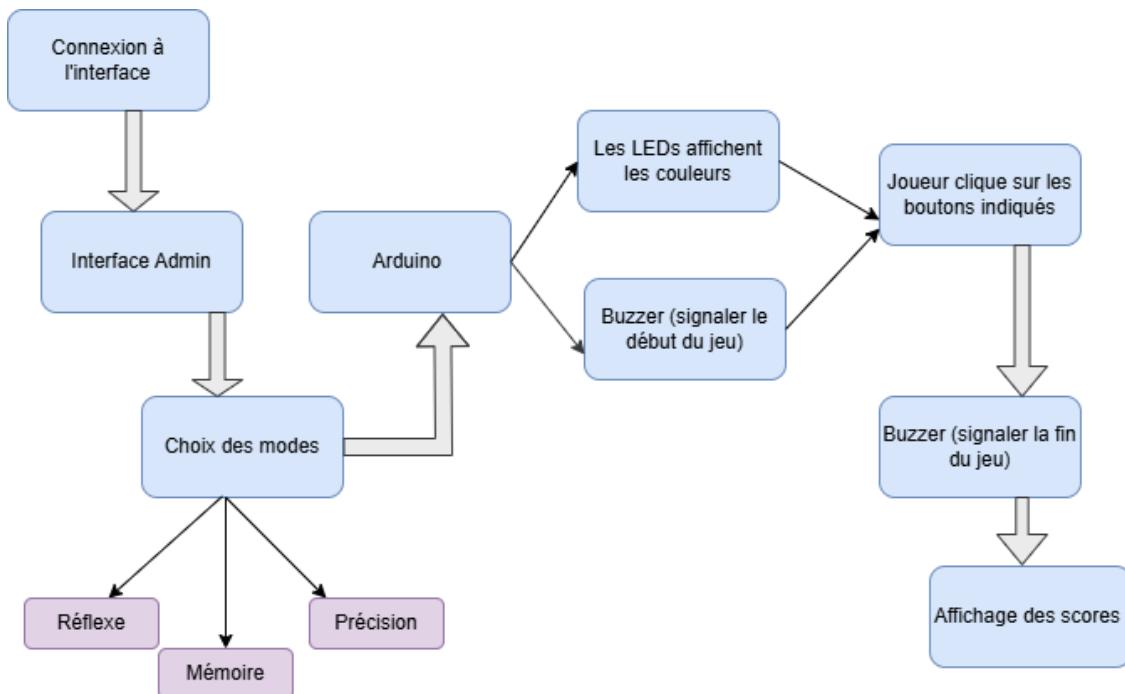


figure n°1: schéma bloc de Pulse Light

2 Réalisation

a. Choix des composants

- **Écran LCD I2C** : Affiche les commandes à réaliser, indique la couleur du bouton sur lequel appuyer et l'endroit à cliquer pour le mode "précision".
- **Boutons poussoirs**: 4 couleurs et 4 boutons pour chaque couleur .
- **LEDs NeoPixel** : Au départ, nous voulions utiliser des LEDs classiques, mais le choix s'est rapidement tourné vers les NeoPixels afin de faciliter la synchronisation.
- **Carte Arduino Uno R3**
- **Registre à décalage** : Utilisé pour regrouper les boutons par blocs de 4 et ainsi réduire le nombre d'entrées nécessaires sur l'Arduino.
- **Module son** : Permet de jouer des musiques de succès ou d'échec.

- Résistances.

b. Tests unitaires de chaque composant

Les tests unitaires des composants assurent le bon fonctionnement de ces derniers mais aussi une détection plus rapide et simple des erreurs futures.

1) Test unitaire de l'écran

Un montage minimalisté est utilisé pour tester l'écran en le branchant directement sur la carte Arduino et en utilisant un code simple pour afficher le message “**HELLO PRONTO**”, comme on peut le voir sur la figure n°1.

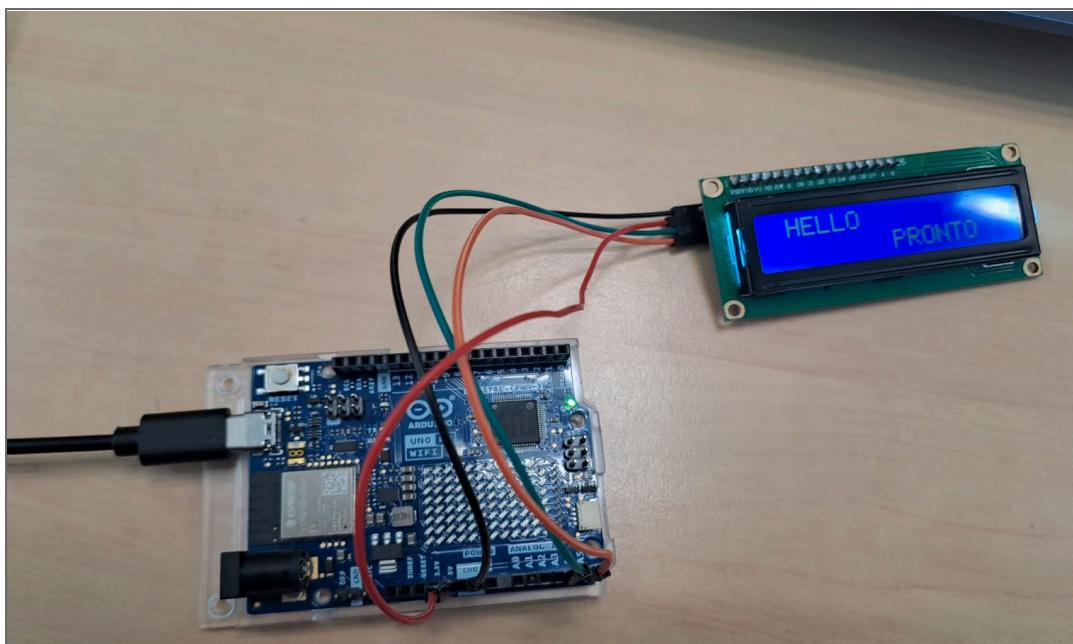


figure n°2 : Test unitaire Ecran LCD

2) Test unitaire du bouton

Les 16 boutons sont testés les uns après les autres en utilisant un montage simple incluant une résistance, une carte Arduino et un code qui vérifie si le bouton est appuyé. Cela a été utile pour éliminer de nombreux boutons qui ne fonctionnaient pas. (voir figure n°3)

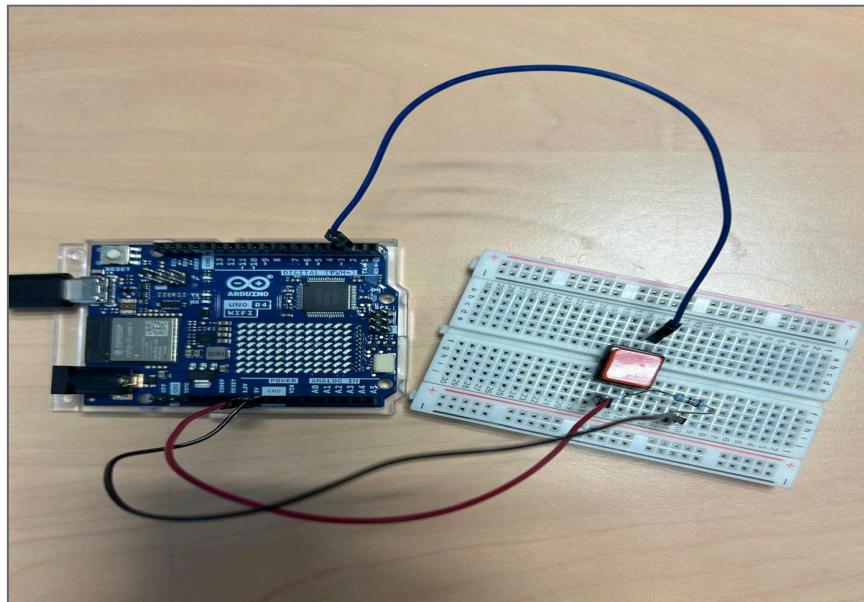


figure n°3 : Test unitaire bouton

3) Test unitaire du neopixel

Les LEDs Neopixel étaient branchées directement sur l'arduino, différentes couleurs ont été affichées sur la LED pour une meilleure prise en main du composant.

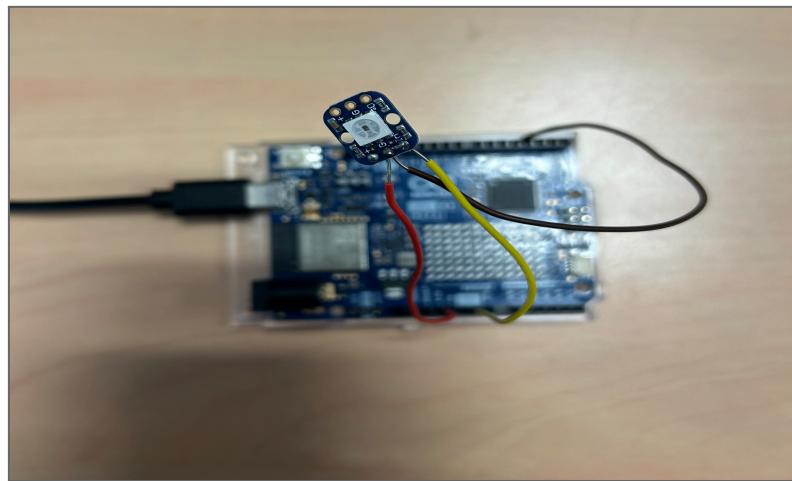


figure n°4 : Test unitaire Neopixel

4) Test unitaire du capteur

Nous avons dû nous passer d'un capteur après des tests unitaires, faute de fragilité de ce dernier dont les fils se cassaient à tout mouvement. Nous avons donc opté pour un capteur plus résistant et fonctionnel.

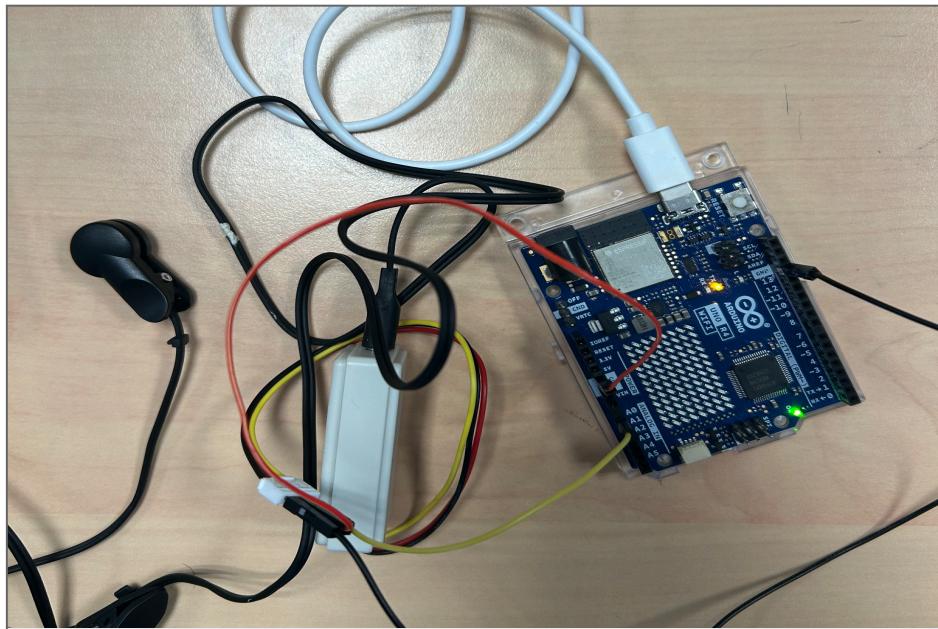


figure n°5 : Test unitaire du capteur

5) Test unitaire du buzzer

Le test du buzzer piézoélectrique a été concluant. Ce dernier a bien émis les notes attendues lors de notre test.

c. Segmentation du code des différents modes

Afin de garantir une gestion claire, le code est structuré en modules distincts selon les différents modes du jeu:

Trois modes pour le moment :

- Mode réflexe
- Mode mémoire
- Mode précision

La segmentation du code en complet permet sa compréhension optimale et facilite ainsi à la fois son développement, sa maintenance et son évolution future.

1) Gestion du son:

Cette partie du code regroupe les fonctions qui permettent d'émettre des stimuli sonores en fonction du contexte et de l'avancée du joueur dans le jeu :

- **Début du jeu:** Une fonction génère un son pour signaler au joueur le démarrage de la partie.
- **CountDown:** Dans le mode mémoire, avant chaque nouvelle séquence, le son CountDown est joué.
- **Succès:** Une fonction qui s'active uniquement si le joueur réalise correctement l'action.
- **Échec:** Génère un son différent de Succès, pour indiquer une erreur due à l'action de la partie.
- **Fin de la partie:** cette fonction marque la fin d'une partie par un son spécifique.

2) Gestion de l'Écran:

Ce module comprend des fonctions permettant une gestion précise de l'affichage textuel des informations du jeu :

- **Affichage du score :** Indique les résultats en temps réel (dans le mode réflexe et précision) et les résultats finaux (pour tous les modes).
- **Affichage textuel de la couleur :** Affiche la couleur à appuyer par le joueur (immédiatement pour le mode réflexe, à la fin de la séquence pour le mode mémoire).
- **Affichage de l'emplacement (mode précision) :** Présente visuellement et clairement l'emplacement précis du bouton sur lequel l'utilisateur doit appuyer.
- **Message de fin :** Affiche un message final clair indiquant la fin de la partie et proposant les résultats obtenus.

3) Gestion des LEDs

Les fonctions associées aux LEDs sont conçues pour offrir une signalisation visuelle intuitive et rapide :

- **Affichage d'une couleur spécifique :** Active les LEDs à la couleur précise nécessaire à l'action en cours.
- **Affichage neutre (couleur blanche) :** Utilisée pour un état par défaut (démarrage, fin, entre chaque couleur)

4) Gestion des Boutons

Elle regroupe les fonctions liées à l'interaction physique du joueur avec les boutons du dispositif :

- **Traduction des entrées des registres :** Une fonction traduit les informations des registres en identifiant quel bouton a été activé.
- **Détection du bouton appuyé :** Fonction permettant la détection de manière précise et rapide des pressions des boutons par le joueur.

5) Gestion des Données

Ces fonctions assure une structuration organisée des données captées ou générées pendant les parties :

- **Données capteurs** : Fonction utilisée pour la récupération des mesures physiologiques (fréquence cardiaque) provenant du capteur de rythme cardiaque.
- **Données de réussite** : Enregistrement des résultats obtenus par le joueur (succès, échec, score final).
- **Données temporelles** : Gestion des données de temps de réaction réel

6) Gestion de la logique du Jeu

Ces fonctions regroupent la logique de jeu. La boucle coordonne les différentes fonctions décrites plus haut.

- **Sélection de la zone** : Une fonction dédiée qui sélectionne aléatoirement la zone de couleur selon le mode de jeu en cours (elle choisit une zone parmi 4 pour le mode réflexe et mémoire, une zone parmi 16 pour le mode précision)
- **Vérification du bon bouton** : une fonction valide si le bouton pressé par le joueur correspond bien à l'action attendue
- **Boucle principale** : Elle supervise le déroulement général et complet d'une partie, assurant l'enchaînement synchronisé des différentes fonctions citées plus haut (interaction avec les boutons, stimuli visuels et sonores, gestion du temps, récupération et sauvegarde des données).

d. Tests unitaires du code et des modes

Les tests unitaires des fonctions permettent de s'assurer que chaque fonction remplit correctement sa tâche en l'isolant des autres fonctions. Ces tests garantissent la fiabilité du système avant d'intégrer l'ensemble du code :

1) Tests sur les fonctions des Leds

- Vérifier l'allumage précis des LEDs selon les couleurs attendues et leur extinction au bon moment.
Méthode : Tester individuellement l'allumage de chaque couleur (rouge, vert, bleu, jaune) et le vérifier visuellement.
- Tester les fonctions d'affichage neutre (blanc)

2) Test des fonctions de l'écran

- Vérifier l'exactitude, la précision et la lisibilité de l'affichage textuelle sur l'écran.

Méthode : Afficher des messages standards basiques comme les scores finaux et intermédiaires, la fin de jeu, les couleurs, les emplacements précis d'une couleur, et tester avec différentes couleurs aléatoires pour s'assurer de l'affichage correct.

3) Tests des fonctions des registres et boutons

- Assurer une détection précise des pressions sur les boutons

Méthode : Vérifier que chaque bouton correspond au bon registre et produit la couleur attendue lors de appui. Tester le dispositif en stimulant l'appui ou plusieurs pressions rapides d'appui sur les boutons.

4) Tests de l'enregistrement des données

- Vérifier que toutes les données sont correctement captées et sauvegardées sur l'ordinateur.

Méthode : Jouer à un mode de jeu, et vérifier le format du document enregistré

5) Tests des différents modes de jeu

- Vérifier que chaque mode de jeu fonctionne correctement

Méthode : Pour le mode Mémoire, vérifier la génération des séquences, aléatoire, suivant la difficulté de la partie et la gestion des erreurs.

Pour le mode Réflexe, vérifier que le bouton pressé est celui attendu et chronométrier pour s'assurer de la fiabilité des valeurs mesurées.

e. Assemblage des composants

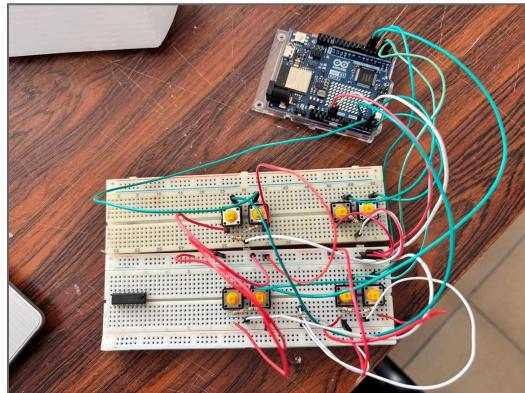


figure n°6 : Circuit électronique partiel

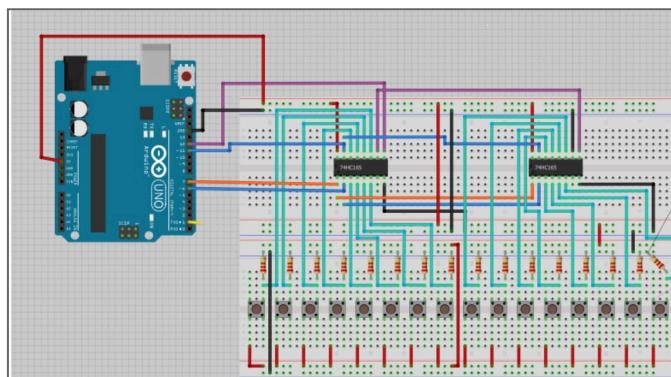


figure n°7 : Circuit électronique recréé sur logiciel : deux registres en série

Dans notre projet Pronto basé sur Arduino, la limitation du nombre de broches d'entrée/sortie s'est rapidement révélée contraignante [figure 6], notamment en raison de la nécessité de connecter 16 boutons poussoirs. Pour surmonter cette contrainte, nous avons opté pour l'utilisation de registres à décalage parallèle-in série-out, en particulier le 74HC165N, qui permet de lire jusqu'à 8 entrées numériques en n'utilisant que 3 broches Arduino (horloge, chargement et données série), contre une broche par bouton en connexion directe.

Un avantage clé de ce composant est sa capacité à être chaîné [figure 7 &8], permettant ainsi la lecture de 16, 24 ou davantage d'entrées avec le même principe. Dans notre cas, nous avons donc connecté deux registres en cascade afin de lire l'état des 16 boutons tout en préservant un nombre minimal de broches Arduino.

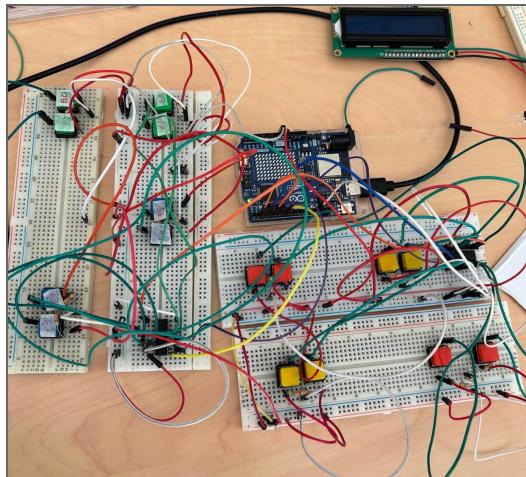


figure n°8 : Circuit électronique complet

Nous avons également été confrontés à la difficulté d'identifier les sources d'erreurs dans le montage. Un simple faux contact suffisait à perturber le fonctionnement du système, ce qui nous obligeait à vérifier manuellement chaque connexion, une par une, afin de localiser le problème. Cette étape s'est révélée à la fois chronophage et délicate, car il fallait faire preuve de rigueur et de méthode pour éviter d'aggraver la situation en introduisant de nouvelles erreurs involontaires.

Le câblage dense et l'enchevêtrement des fils [figure 8] compliquaient encore davantage l'intervention, rendant le dépannage parfois difficile. La solution que nous avons envisagée est d'imprimer le circuit sur un PCB (Printed Circuit Board). Cela permet d'éliminer les problèmes de connexions instables et de soudures fragiles. Ce PCB a été réfléchi pour être le plus pratique possible dans la mesure où les 16 boutons doivent être disposés en 4 groupes de 4 à des endroits distincts de la boîte. Le PCB est décrit plus en détail dans la suite du rapport.

Difficultés rencontrées:

- Composants fragiles: Certains composants, comme le capteur de rythme cardiaque, ont des connexions fines et difficiles à manipuler.
- Connexions instables: Les câbles ont tendance à se détacher ou à faire de mauvais contacts ce qui entraîne des pertes de signal et des courts circuits.
- Soudures instables et défectueuses: Certaines soudures étaient difficiles à réaliser et il a fallu les refaire à plusieurs reprises pour assurer une bonne conductivité.

f. Développement de l'interface administrateur

Le développement de l'interface administrateur s'est basé sur **React**, une bibliothèque JavaScript permettant de construire des interfaces dynamiques et réactives. L'objectif était de concevoir une interface simple, intuitive et fonctionnelle permettant à l'administrateur de gérer à distance les modes de jeu et d'avoir un retour immédiat sur les performances du système physique Arduino.

a. Connexion sécurisée

L'interface démarre par une **page de connexion** demandant un **mot de passe administrateur**. Ce mot de passe est ensuite vérifié côté serveur via un appel **POST** vers l'endpoint `/api/loginAdmin`. Si la validation est réussie, l'administrateur accède au tableau de bord. Cette étape ajoute une couche de sécurité minimale pour éviter tout déclenchement non autorisé des modes de jeu.

b. Sélection des modes de jeu

Une fois connecté, l'administrateur voit trois **boutons cliquables** : Réflexe, Mémoire, Précision. Chaque bouton correspond à un **mode de jeu implémenté sur la carte Arduino**. En cliquant sur l'un d'eux :

- Une requête HTTP est envoyée via **Axios** à `/api/selectGameMode`.
- Le serveur Node.js traduit ce choix en une commande série (`start_reflexe`, `start_memoire`, etc.).
- Cette commande est envoyée à l'Arduino via le port Série, qui démarre le mode demandé.

Chaque clic sur un bouton permet donc de **contrôler physiquement** l'appareil Arduino à distance, ce qui établit une relation directe entre le front-end React et le monde réel.

c. Récupération et affichage des résultats

Une fois la partie jouée, l'administrateur peut cliquer sur le bouton "**Afficher les scores**". Cela déclenche un appel vers un autre endpoint (`/api/getReflexeResults`, `/api/getMemoireResults`, etc. selon le mode choisi).

Le serveur renvoie alors les données stockées après la dernière session :

- Score total obtenu
- Temps moyen (calculé côté serveur)
- Et, si applicable, le détail des essais

Ces données sont ensuite **affichées directement dans l'interface**, sans rechargement de page, grâce à la logique réactive de React. La structure de l'état (`useState`) permet d'afficher dynamiquement ces informations dès leur réception.

d. Ergonomie et design

L'interface a été stylisée avec **CSS personnalisé** et utilise la police Orbitron, des couleurs sombres, les boutons colorés pour rappeler l'univers technologique.

Chaque section (connexion, sélection de mode, résultats) est **clairement séparée visuellement**, et les erreurs de communication (mot de passe incorrect, absence de données, échec serveur) sont toutes gérées et affichées proprement à l'utilisateur avec un message explicite.

3 Intégration et validation

a. Assemblage physique final (circuit électronique + boîte)

A la fin de ce projet, nous obtenons et obtiendrons deux versions finales du prototype du jeu PulseLight. En effet, la conception du PCB étant longue, nous avons préféré avoir une première version du prototype à présenter dans le cas où le circuit imprimé ne serait pas prêt à la date butoire.

La première version finale du prototype se présente sous la forme d'un fond de boîte découpé au laser dans lequel sont placées la carte Arduino Uno R3 ainsi que les deux platines d'essais contenant les boutons poussoirs. L'écran LCD, la LED NeoPixel et le buzzer piézoélectrique sont fixés sur les bords de ce fond de boîte. Le capteur de pouls reste mobile grâce à son long fil de connexion.

Ajouter Photo

Cette première version nous permettra de faire une démonstration dans de bonnes conditions de notre jeu.

Ensuite, notre deuxième version du prototype physique sera dans la continuité de la première. Dans le même fond de boîte, nous placerons une PCB principale [figure 9], contenant les deux registres et des headers pour la connexion aux autres composants, la carte Arduino Uno R3 et 4 autres PCB contenant 4 boutons poussoirs et 4 LEDs chacune [figure 10].

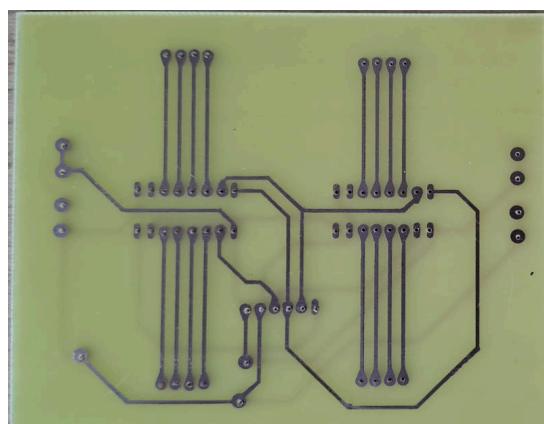


figure n°9 : PCB principal

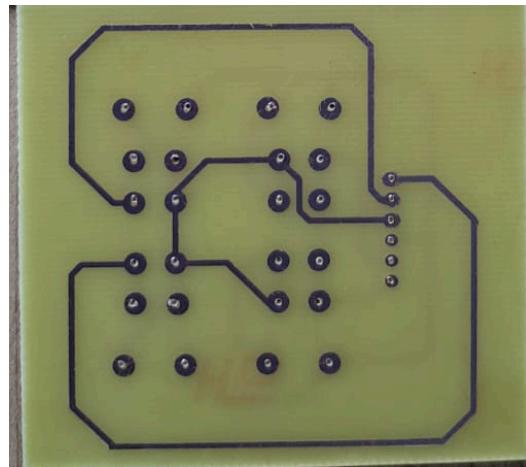


figure n°10 : Exemplaire du PCB correspondant aux boutons (l'empreinte des boutons ne correspond pas à la réalité)

Comme la figure 10 l'indique, les bornes VCC, GND et l'entrée de chaque bouton poussoir seront reliées à la carte PCB principale via des fils soudés. Ensuite, la connexion de la carte principale à l'Arduino se fera via 6 fils au total. Cela permet de diminuer drastiquement le nombre de connexions entre les composants, tout en octroyant une certaine flexibilité quant à la maintenance technique du jeu: si un dysfonctionnement est repéré concernant un boutons, seule une carte PCB reliée à trois autres boutons sera à manipuler pour établir un diagnostic du problème.

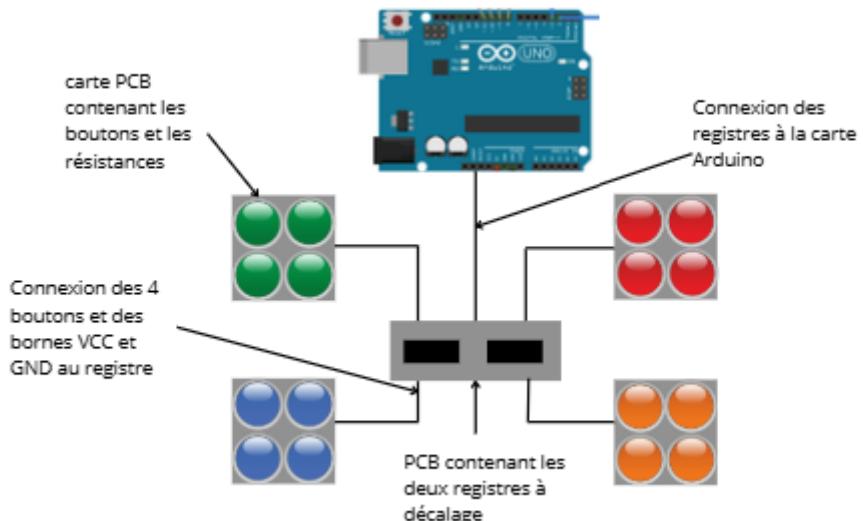


figure n°11 : Schéma simplifié de l'organisation des composants électroniques im

De plus, les 4 cartes PCB contenant les boutons pourront être disposées et surélevées sur différentes zones de la boîte.

Enfin, concernant l'intégration de l'électronique au sein d'un boîtier en bois et découpé au laser, dont le fond est présenté dans la figure 12. Les différents boards et l'Arduino pourront être

disposés à différents endroits et à différentes hauteurs dans le boîtier. Un trou a été percé sur le côté pour pouvoir brancher l'Arduino à nos ordinateurs.



figure n°12 : Fond du boîtier PulseLight.

Nous n'avons pas encore terminé l'intégration de l'électronique dans le dispositif physique, néanmoins la figure 13 donne un aperçu du prototype qui sera obtenu. Les 4 zones de 4 boutons seront placées de part et d'autre de l'écran et de la barre de LEDs Neopixel.

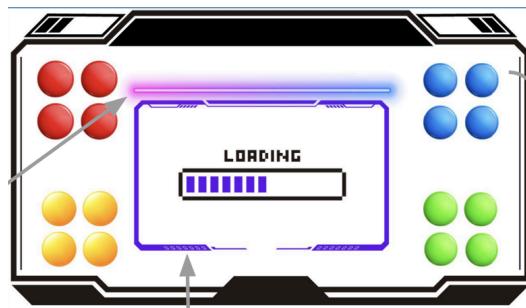


figure n°13 : Prototype du dispositif final.

b. Relation entre interfaces virtuelle et physique : Exploitation des données

L'interface web React que nous avons développée permet une **visualisation en temps réel** des performances du joueur pour chaque mode. Une fois connecté en tant qu'administrateur (via un mot de passe sécurisé), l'utilisateur accède à un tableau de bord interactif avec trois boutons : **Réflexe**, **Mémoire**, et **Précision**.

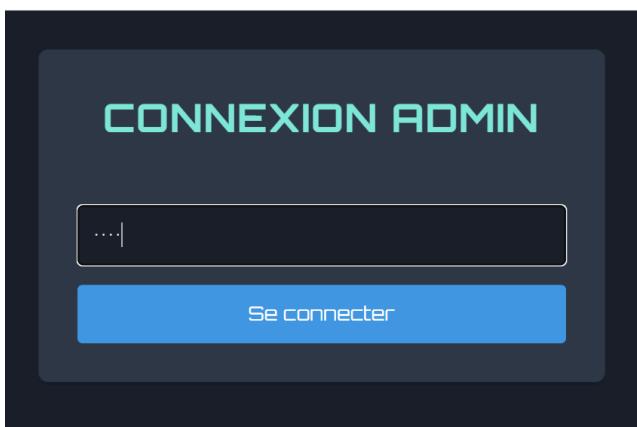


figure 14

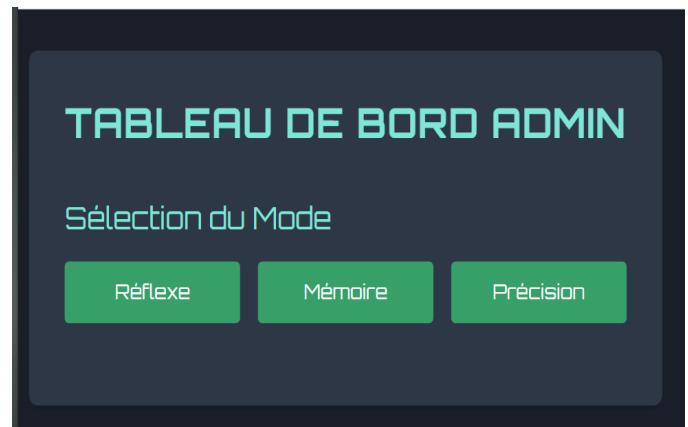


figure 15

Lorsque l'un de ces boutons est cliqué, le mode de jeu correspondant est lancé sur la carte Arduino via une commande série. Après l'exécution complète du jeu sur le système physique (réactions du joueur, affichages sur l'écran LCD, changement de LED, sons, etc.), un message contenant les résultats est automatiquement envoyé de l'Arduino vers le serveur Node.js.

Ce message est intercepté, parsé (décodé), et les informations suivantes sont extraites :

- Score total
- Temps moyen (si disponible, sinon "N/A")

Ces données sont ensuite mises à jour dans l'état React de l'interface et **affichées dans la section "Statistiques du Joueur"**, juste en dessous du bouton de mode sélectionné. Un simple clic sur le bouton **"Afficher les scores"** permet de charger les derniers résultats disponibles.

Par exemple :

- Si l'utilisateur clique sur "**Mémoire**", l'interface affiche :
Mode sélectionné : Mémoire
puis après un clic sur "**Afficher les scores**", l'écran affiche :
Score Total : 120,
Temps Moyen : 2.83s
et les détails (si disponibles) sont listés sous forme de séquences réussies ou échouées.

L'interface utilise également des codes couleurs et des séparateurs visuels pour rendre les résultats **faciles à lire**, même pour un non-technicien. En arrière-plan, chaque appel API est asynchrone, et un système d'erreurs gère proprement les cas d'échec de récupération des données ou de connexion au serveur.

c. Relation entre interfaces virtuelle et physique : Commande à distance

L'aspect bidirectionnel de ce projet est essentiel. L'interface web React permet non seulement de visualiser les résultats, mais aussi de **contrôler directement le système physique Arduino à distance**, via une simple connexion série.

Concrètement :

- L'interface contient trois boutons (un par mode de jeu) : **Réflexe, Mémoire, Précision**
- Lorsqu'un utilisateur clique sur l'un de ces boutons, une requête **POST** est envoyée au serveur Express/Node.js via Axios.
Exemple : **POST /api/selectGameMode** avec le corps { mode: "Réflexe" }
- Le serveur mappe ce mode à une commande série (**start_reflexe, start_memoire, etc.**) et l'envoie à la carte Arduino via le port COM8.

```
→ Command sent: start_reflexe  
← From Arduino: Mode Reflexe demande
```

- L'Arduino reçoit cette commande, réinitialise les états internes, démarre les composants nécessaires (écran LCD, LEDs, sons, capteurs, chronos...) et exécute le jeu.

Pendant ce temps, l'écran LCD affiche un message comme "**Demarrage Reflexe**", et les LEDs changent de couleur pour indiquer l'état du jeu. L'utilisateur, depuis l'interface web, ne fait **aucune interaction physique directe avec l'Arduino** : tout se fait **à distance via l'interface**.



figure 16

À la fin du jeu, les résultats sont envoyés par Arduino → Node.js, puis renvoyés à React qui les affiche instantanément dans la section des scores.



figure 17



figure 18

Conclusion : cette architecture met en place une **chaîne complète de communication virtuelle ↔ physique**. Un clic web déclenche un comportement physique réel, et une réponse physique est transformée en **données numériques intelligibles** et visualisées dans une interface web moderne.

d. Validation de l'ensemble

Après intégration de chaque élément au dispositif, nous avons obtenu un prototype qui satisfait plusieurs de nos critères initiaux, nous permettant de proposer plusieurs modes à l'utilisateur. PulseLight est contrôlé à distance (depuis l'ordinateur), grâce à une interface administrateur simple d'utilisation, et sur laquelle on peut récupérer les données du joueur. Nous disposons d'une boîte

Nous avons effectué une phase de tests, une fois le prototype du dispositif finalisé. Nous avons pu prendre de nombreuses vidéos, afin d'assurer la progression des différentes étapes. Nous avons également préparé une étape de démonstration, afin de pouvoir constater la validation de l'ensemble.

4 Conclusion et perspectives

a. Livrables scientifiques attendus

Le projet a été jalonné de plusieurs livrables scientifiques, qui ont permis de suivre nos avancements et nos progrès.

- **Présentation de projet :** Présenter le projet au client en détaillant ses divers aspects fonctionnels ainsi que le matériel spécifique nécessaire à sa mise en œuvre.:
- **Document de cadrage du projet :** Décrire l'ensemble du projet en synthétisant les éléments clés du projet incluant le contexte, les besoins fonctionnels, la méthodologie, la structure de l'équipe, la planification, les livrables et moyens de communication et action à venir.

Déposé le 27/02/2025.

Retour:

- Bonne cohésion, et une envie de progresser dans les différents aspects du projet.
- Planning est bien détaillé et la liste des livrables est complète.
- Manque de gestion des risques.
- **Revue de projet et d'équipe** : Faire un point sur l'avancement du projet, incluant les mises à jour du planning, des livrables et des contributions techniques (montage électronique et programmation). Présenter les éléments réalisés et l'état d'avancement du projet au client.

Déposé le 27/03/2025.

Retour :

- Bonne mise à jour de la planification temporelle, manque de prise en compte des jours fériés de mai.
- Bon travail d'équipe, et un investissement individuel important.
- Une bonne application de la méthodologie de tests unitaires et de tests globaux.
- Bon avancement global du projet.

- **Livrable recette du projet** : Présentation du prototype final, description détaillée de toutes les étapes réalisées et des résultats obtenus.

Déposé le 23/05/2025.

b. Validation des scénarios:

➤ Scénario de base :

- **Première version du scénario** : L'utilisateur allume le boîtier. Un test automatique des capteurs et composants est effectué, suivi d'un affichage du menu principal sur l'écran LCD. En utilisant les boutons situés à droite de l'écran, l'utilisateur sélectionne un mode de jeu (par exemple, le mode mémoire). Les LEDs s'allument alors dans une séquence aléatoire que l'utilisateur doit reproduire en appuyant sur les boutons correspondants. En fin de partie, le score, le temps réactionnel moyen et les données physiologiques mesurées (niveau de stress, fréquence cardiaque) sont affichés sur l'écran LCD. L'utilisateur peut sauvegarder ses résultats dans la mémoire.
- **Scénario validé** : L'utilisateur allume le boîtier. Un test automatique des capteurs et composants est effectué, une musique annonce le début du jeu , le mode est choisi via l'ordinateur. Les LEDs s'allument alors dans une séquence aléatoire que l'utilisateur doit reproduire en appuyant sur les boutons correspondants. En fin de partie, le score, le temps réactionnel, les données physiologiques sont sauvegardés

sur l'ordinateur. Concernant les données physiologiques, nous récoltons seulement la fréquence cardiaque. En effet, la mesure du niveau de stress avec un capteur de conductance cutanée a été jugée peu pertinente et source de dépenses inutiles.

➤ Scénario d'analyse des données :

- **Première version du scénario** : Dans ce mode, l'utilisateur accède à l'historique des données via le menu. L'écran LCD présente un histogramme des variations du stress sur plusieurs parties ainsi qu'une table récapitulative des fréquences cardiaques moyennes.
- **Scénario validé** : l'histogramme se fait manuellement sur l'ordinateur.

➤ Scénario mode réflexe :

- **Première version du scénario** : Dans ce mode, les LEDs s'allument en succession rapide à divers endroits du boîtier. L'utilisateur doit appuyer sur les boutons associés aussi rapidement que possible. La difficulté augmente progressivement en réduisant le temps d'allumage des LEDs. Si l'utilisateur manque une LED ou appuie sur le mauvais bouton, une alerte sonore est émise. En fin de partie, le système affiche le nombre de réactions réussies et le temps moyen de réaction.
- **Scénario validé** : Dans ce mode, les LEDs s'allument dans une couleur particulière. L'utilisateur doit appuyer le plus rapidement possible sur un bouton de cette couleur. Quand le bouton est correctement appuyé (seule une bonne réponse permet de passer à la suite), le temps mis s'affiche sur l'écran. Au bout d'un certain nombre d'essais (actuellement, 10), la partie se termine, et les données sont transmises à l'interface administrateur.

➤ Scénario mode précision :

- **Première version du scénario** : Dans ce mode, une zone dense (située sur une autre zone du boîtier que celle des LEDs/boutons poussoirs des autres modes) de LEDs et de boutons est activée. L'une des LEDs s'allume aléatoirement, et l'utilisateur doit appuyer rapidement sur le bouton correspondant. Le système mesure à la fois la rapidité et la précision de l'utilisateur. Si l'appui est correct, une LED verte s'allume et un son positif est joué. Si l'utilisateur se trompe ou est trop lent (≥ 1 s), une LED rouge s'allume et un son d'échec est émis. En fin de partie, un score est calculé en fonction du nombre de réponses correctes et du temps moyen de réaction.
- **Scénario validé** : le mode se fait sur la même zone que les autres modes et une musique annonce le début. La LED s'allume aléatoirement sur une couleur et l'écran LCD affiche une zone de 4 boutons disposés en carré. Sur l'écran, un des quatre

boutons clignote. L'utilisateur doit appuyer avec précision sur le bouton correspondant à la couleur et à la position indiquée par la LED et l'écran.

➤ Scénario réaction au son :

- **Première version du scénario :** Dans ce mode, le système génère un son aléatoire (par exemple, « GO! ») via le module DFPlayer Mini. L'utilisateur doit appuyer sur un bouton désigné dans un temps imparti (≤ 1 ou 2 s). En cas de réussite, une LED verte s'allume et le système passe à la prochaine réaction. En cas d'échec, une LED rouge s'allume et une alerte sonore est émise. Les résultats finaux affichent le nombre de réactions correctes et le score global.
- **Scénario non validé :** Le mode son n'a pas été validé par manque de temps et par souci de pertinence.

➤ Scénario limite :

- **Première version du scénario :** Si un capteur est hors plage ou ne fonctionne pas correctement (par exemple, une mesure de stress aberrante ou un signal physiologique absent), le système affiche un écran de diagnostic. Cet écran identifie l'élément défectueux et propose des solutions (redémarrage, réinstallation du capteur, etc.). Si le problème persiste, un mode de jeu limité aux fonctionnalités disponibles est activé.
- **Le scénario n'est pas validé par manque de temps et par soucis de complexité.**

c. Axes d'amélioration

La version actuelle n'est qu'une première version du projet final , dans une démarche d'amélioration continue plusieurs axes d'amélioration sont envisageables:

Amélioration de composant :

1. Opter pour un écran RGB et se passer des LEDs en affichant directement les couleurs sur l'écran.
2. faire une version qui fonctionne sans connexion à un ordinateur, grâce à une alimentation sur batterie.
3. Remplacer l'arduino R3 par une carte arduino R4 et faire la connexion par Bluetooth.

Amélioration de fonctionnement:

1. Ajouter une LED de feedback: Une LED qui s'allume en rouge si le mauvais bouton est appuyé, et en vert dans le cas contraire, pour donner un retour visuel immédiat à l'utilisateur.
2. Ajouter une fonction pause /reprendre :Offrir la possibilité de suspendre une partie en cours et de la reprendre sans tout recommencer.
3. Afficher les différentes informations de pulsation sur l'écran.

Pour aller plus loin :

1. Ajouter un quatrième mode “sensoriel” : ce mode pourrait indiquer les couleurs à voix haute et donner le signal de départ oralement (“Go”).
2. Prendre en compte le scénario limite (détecter le dysfonctionnement d'un capteur)

d. Retours d'expériences du projet

1) Observation du Diagramme de Gantt :

Construction du boîtier // Montage du circuit électronique // Code des modes principaux // Interface joueur // Interface administrateur // Code des capteurs // Tests finaux

- Début de projet

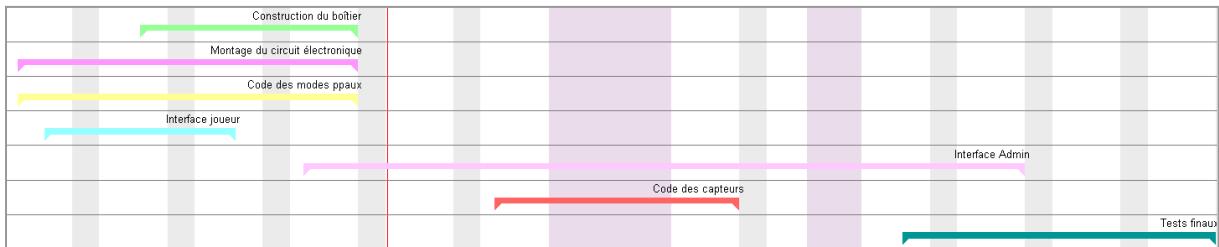


figure n° 19 : Diagramme de Gantt initial

- Mi-parcours

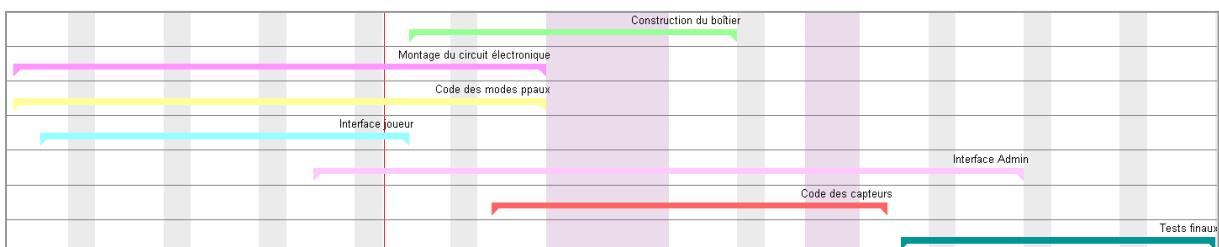


figure n°20 : Diagramme de Gantt de mi-projet

- Fin de projet

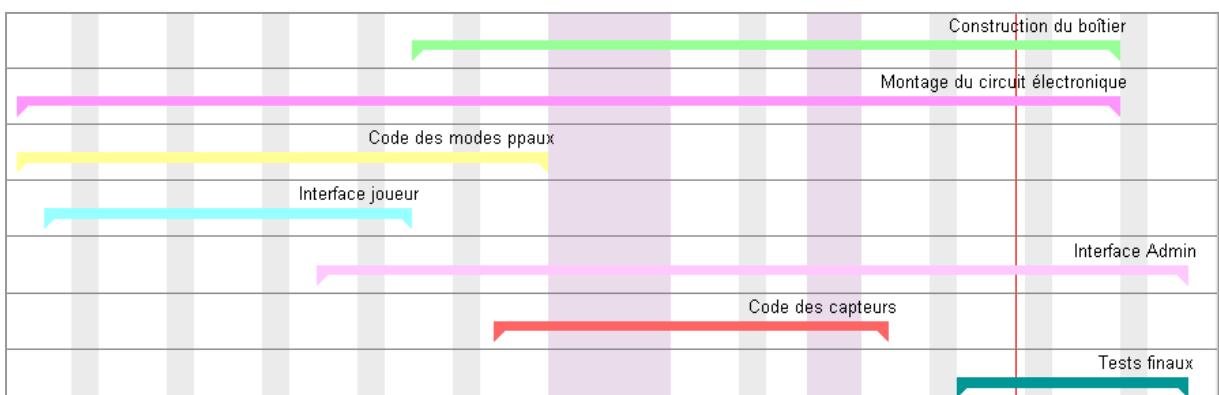


figure n°21 : Diagramme de Gantt de fin de projet

L'organisation des tâches et leur durée a beaucoup évolué durant notre projet. La plupart des tâches ont dû être allongées, certaines raccourcies. Les durées allouées à l'**interface joueur**, le **code des modes principaux** ainsi que des **capteurs** n'ont pas été fortement modifiées, et leur réalisation a suivi majoritairement le **plan initial**. Cependant, certaines tâches ont dû être gérées de manière très différente, par rapport à ce que le diagramme de Gantt initial prévoyait, et même par rapport aux rectifications de mi-parcours.

Nous avons choisi de **décaler la construction du boîtier** à la fin du projet, plutôt qu'initialement, pour nous laisser une grande marge de manœuvre sur le montage du circuit électronique. Au niveau du circuit, nous avons eu **du mal à estimer le temps nécessaire** à la conception d'un PCB, ce que nous cherchions à faire pour réduire la taille du montage. Cette étape a pris plus de temps qu'estimé, retardant ainsi la fabrication finale du boîtier.

De plus, une autre tâche nous a confronté à de la difficulté. En effet, **le développement de l'interface administrateur a connu un obstacle** : la commande à distance. La récupération des données, et l'interface en elle-même ont pu suivre le planning prévu initialement, mais la commande à distance a été plus compliquée à mettre en place. Nous avons dû lui accorder plus de temps.

Nous avons donc choisi de **réduire la durée des tests**, quitte à annuler entièrement la campagne de tests prévue initialement, et privilégier les tests privés, afin de pouvoir proposer un projet le plus complet possible. Cependant, cette campagne de tests était déjà dès le départ prévue comme réalisable seulement dans un scénario idéal, et nous avions conscience de **sa possible annulation selon les circonstances**. Ce n'était donc pas un point important du projet.

Finalement, les différentes tâches ont toutes pu, au moins partiellement, être satisfaites. Cependant, nous avons dû limiter nos tests afin de pouvoir rendre le projet à temps. Les tâches qui nous ont posé **le plus de problème sur leur estimation** sont des missions que nous n'avions jamais faites, comme la conception d'un PCB ou la commande à distance d'une carte Arduino via une interface administrateur. Nous n'avions **pas accordé assez de temps ni de marge à ces tâches**, ce qui nous a mis dans une situation de risque.

2) Gestion des risques

A. Risques envisagés initialement

➤ Manque de communication et d'organisation

Pour éviter ce risque, nous avons décidé de nous concerter régulièrement pour faire des mises au points fréquentes. Nous avons décidé de tenir **un rapport de compte-rendus**, où chacune a la possibilité d'écrire ses succès et ses difficultés, afin de que tout le monde puisse être au courant de l'avancée des autres, et leur ressenti.

Pour prévenir un potentiel risque de mauvaise organisation, nous avons prêté beaucoup d'attention au début du projet à **la répartition des tâches, et à leur ordre de priorité**.

➤ Suppression des fichiers de travail (codes, plannings, schémas, ...)

Nous avons choisi de travailler avec un **Drive partagé**, afin que chacune puisse échanger ses fichiers facilement, et afin de pouvoir toutes travailler sur les mêmes documents en même temps. Cependant, pour éviter une erreur de suppression qui entraînerait alors la suppression pour toutes, nous avons chacune des versions de notre travail **sur nos ordinateurs personnels**, ce qui permet de réduire fortement le risque de suppression accidentelle.

➤ Dysfonctionnement des composants

Il y a toujours un risque que certains composants ne fonctionnent pas comme ils le devraient. Dans certains cas, nous ne pouvons pas savoir si l'erreur vient de notre compréhension du composant

ou du composant. Nous avons donc choisi de **fixer une date limite**, après laquelle nous demanderions de l'aide à l'encadrant, et nous chercherions une solution de repli, si le composant était effectivement **défectueux**.

➤ Non-compatibilité des différentes parties

Comme nous opérons par sections, avec chacune une partie principale à réaliser, nous avons dû faire très attention à la compatibilité des différentes parties, en communiquant ensemble régulièrement, en tenant un document nous assurant la compatibilité des entrées et sorties, mais aussi en travaillant chacune avec les autres de temps à autre, afin de connaître un minimum toutes les parties du projets.

B. Risques auxquels nous avons également été confronté

➤ Non-compatibilité des outils de travail

Par exemple, la carte **Arduino R4 Wifi** que nous voulions initialement utilisée, afin d'avoir une liaison sans fil avec l'interface administrateur **n'est pas compatible** avec de nombreux ordinateurs. Nous avons donc choisi d'utiliser une Arduino Uno R3 à la place.

➤ Non-disponibilité du matériel

Chacune des parties requiert **des tests** : pour cela, il est nécessaire d'avoir le dispositif du circuit. Ainsi, il était régulier d'être plusieurs à devoir travailler en même temps sur le même objet, nous avons donc dû nous mettre d'accord sur les **ordres de priorité des tâches**, et sur les **créneaux de disponibilité** du dispositif.

➤ Retards (manque de prise en compte de certains facteurs)

Le projet a pris un peu de retard par rapport aux planifications initiales. Nous n'avions pas suffisamment bien estimé la durée de certaines tâches. La présence de **nombreux jours fériés en mai**, ainsi que des **vacances** au milieu du projet, et même la **diminution du nombre de séances** de projet durant le mois d'avril, ont représenté un grand risque pour le retard du projet. Cependant, grâce à la marge que nous avions choisi de prendre au début du projet, et notre prise en compte des vacances et des moindres séances dans notre planification initiale, nous avons pu **rester dans des limites raisonnables**.

➤ Problème de communication entre interface administrateur et Arduino

Comme la liaison entre l'interface administrateur et le circuit électronique ne pouvait être effectuée qu'en toute fin du projet, tout problème soulevé à cette étape représentait une situation de risque important, car en fin de projet.

5 Références bibliographiques :

Abigail, G. Boostez votre esprit : Les jeux de réflexion comme alliés de la santé mentale. 18 décembre 2024.

[<https://regles-et-jeux.fr/explorer/les-bienfaits-des-jeux-de-reflexion-sur-la-stimulation-mentale/>]

cpstest.org.Reaction Time Test - Test your reaction speed. s. d.
[<https://cpstest.org/reaction-time-test/>]

Human Benchmark. Reaction Time Test. s. d.[<https://humanbenchmark.com/>]

Lumosity. Turbo Mode - Jeux de réflexe. s. d. [<https://www.lumosity.com/train/turbo/odp/3/play>]

Steven, & Steven. 2 Jeux de réflexe - RobDomo. 22 mai 2022.
[<https://robdomo.com/projets-diy/projets-robdomo/mini-jeux-arduino/2-jeux-de-reflexe/>]

Suzie. Les 6 bienfaits des jeux de réflexion pour la santé mentale. 30 août 2023.
[<https://www.dynseo.com/les-6-bienfaits-des-jeux-de-reflexion-pour-la-sante-mentale/>]

Suzie. Les jeux de réflexion pour améliorer vos compétences cognitives. 25 juillet 2023.
[<https://www.dynseo.com/les-jeux-de-reflexion-pour-ameliorer-vos-competences-cognitives/>]

6 Annexe :

a. Retour d'expérience - Judith

Question : Racontez-moi une situation dans ce projet où vous étiez confronté.e à un problème qui avait plusieurs solutions possibles.

Dans le cadre du projet PRONTO, réalisé au sein de l'école durant 4 mois, nous avons décidé de concevoir un jeu de réflexe et de mémoire, à destination des personnes âgées. J'ai été chargée d'écrire plusieurs modes, le mode réflexe et le mode mémoire. Pour pouvoir vérifier les codes, il était nécessaire de les tester sur le circuit électronique, et la carte Arduino associée. Hors, cela ne marchait pas du tout initialement.

Pour résoudre la situation, j'ai tenté plusieurs solutions possibles :

- Vérification du code

En testant chacune des parties du code individuellement, sur un autre circuit moins complexe, j'ai pu vérifier que chaque partie fonctionnait correctement. Cependant, je ne pouvais pas tester le code dans son intégralité sans le tester sur le circuit électronique.

- Vérification du circuit

A l'aide de mes camarades de projet, nous avons vérifié le circuit dans son intégralité, pour être sûres que l'erreur ne provenait pas d'un faux contact ou d'un composant défaillant. Finalement, nous avons changé la carte Arduino pour vérifier si l'erreur ne provenait pas de celle-ci. En effet l'erreur provenait de la carte Arduino, qui n'était pas compatible avec mon matériel.

- Compatibilité Carte Arduino

J'ai donc dû me renseigner sur la carte Arduino que nous utilisions, et sur sa compatibilité avec différents types d'appareils. Il en est ressorti que la carte R4 Wifi que nous souhaitions utiliser n'était pas compatible avec de nombreux appareils, donc mon ordinateur et celui de plusieurs de mes camarades de projet. J'ai donc choisi de modifier le circuit en remplaçant la carte R4 Wifi par une

Arduino Uno R3, celle-ci étant bien compatible. Il s'agissait d'abord d'une mesure temporaire, qui a finalement été adoptée pour le restant du projet.

Cela m'a permis de réaliser un problème dont nous n'avions pas connaissance, c'est-à-dire la compatibilité de notre projet avec les appareils personnels des utilisateurs. Pour permettre à notre projet d'être ouvert à un maximum de personnes, nous avons donc choisi de conserver cette carte R3, compatible avec tous les appareils. De plus, cela m'a grandement appris sur tous les différents problèmes liés aux compatibilités, et les spécificités de chaque composant, puisque j'ai dû y accorder beaucoup de temps. Cela m'a permis de trouver des méthodes alternatives pour vérifier mon travail.

b. Retour d'expérience - Tinhinen

Question : Racontez-moi une situation dans ce projet où vous étiez confronté.e à un problème qui avait plusieurs solutions possibles.

"Pendant la réalisation de notre projet Pronto, au sein de notre école, nous devions développer un jeu interactif reposant sur une interface composée de boutons de différentes couleurs, il fallait donc trouver une solution permettant d'indiquer visuellement sur quel bouton appuyer, en affichant la couleur correspondante à l'aide de LEDs.

Plusieurs options ont été explorées pour résoudre ce problème technique :

- Première solution : une LED par bouton
Chaque bouton aurait sa propre LED qui s'allume lorsque l'utilisateur doit appuyer dessus. Mais cela impliquait l'utilisation de 16 LEDs, soit 16 entrées supplémentaires sur l'arduino, un câblage complexe et beaucoup de lignes de code à maintenir.
- Deuxième solution : LEDs autour de l'écran
On a ensuite pensé à placer des LEDs standards autour de l'écran, qui s'allumeront selon la couleur du bouton concerné. Cette solution permettait une visualisation simple, mais chaque LED devait être commandée individuellement, ce qui restait long et répétitif à coder.
- Troisième solution : bandes de LEDs NeoPixel autour de l'écran
Les NeoPixels étant adressables individuellement via une seule entrée, cela facilitait la gestion du code et réduisait les branchements. Cependant, pour bien encadrer l'écran, il fallait souder plusieurs bandes ensemble, et ces soudures étaient trop fragiles, causant des dysfonctionnements.
- Quatrième solution (retenue) : une bande de NeoPixel en haut de l'écran
Cette dernière option offrait une solution à la fois simple, stable et efficace. Elle permettait de conserver tous les avantages des NeoPixels (pilotage simplifié, visualisation claire), tout en évitant les problèmes de soudure fragile et de positionnement difficile. Le câblage était plus propre, le code plus clair, et l'esthétique du dispositif restait cohérente avec l'interface du jeu.

Grâce à cette solution, nous avons pu réduire le nombre d'entrées nécessaires sur l'arduino, écrire un code plus modulaire et maintenable et limiter les risques de panne liés aux soudures fragiles.

c. Retour d'expérience - Justine

Question : Racontez-moi une situation dans ce projet où vous étiez confronté.e à un problème qui avait plusieurs solutions possibles.

En fin de projet, j'ai travaillé sur la conception et la création du Printed Circuit Board (PCB), le circuit électronique imprimé qui avait pour avantage sa taille réduite et la suppression des nombreux fils de connexion. Cette étape s'est révélée laborieuse car nous n'avions jamais fait de PCB et notre circuit électronique est particulièrement compliqué. En effet ce dernier comprend de nombreux composants avec lesquels le joueur doit pouvoir interagir (écran, LED, boutons, capteur de pouls). De plus, la disposition de ces composants est aussi un facteur de difficulté: les 16 boutons doivent être répartis en 4 groupes de 4 boutons autour de l'écran et de la LED.

Ainsi mon problème a été celui de créer un PCB adapté à notre dispositif physique en n'ayant aucune expérience préalable en création de circuits imprimés.

J'ai donc décidé d'agir en suivant tout d'abord les conseils donnés par notre tutrice Mme KEROUEDAN. Cette dernière nous a redirigé vers Mme LE GALL qui m'a guidé sur les premières étapes de conception du PCB. J'ai donc suivi les tutoriels qu'elle m'a envoyés et je me suis aidée de recherches internet pour mieux comprendre certaines fonctionnalités du logiciel EAGLE. J'avais alors pour idée de construire un PCB qui se clipserait comme un shield à notre carte Arduino. Après un rendez-vous de suivi avec Mme LE GALL, j'ai réalisé que recréer fidèlement notre circuit électronique en version imprimée n'allait pas nous permettre de respecter la disposition prévue de nos composants sur le boîtier physique. J'ai donc consulté mes coéquipières pour que l'on puisse établir une solution ensemble. Malheureusement, notre manque de connaissances sur le processus de création d'un PCB nous a limités et nous n'avons pas trouvé de solution pour adapter notre PCB à notre boîtier.

Nous avons donc solliciter une nouvelle fois notre tutrice pour établir ensemble le plan de notre futur PCB. Cette dernière a réfléchi aux contraintes d'interactions et de placement des candidats et nous a suggéré d'imprimer 5 circuits au total. Le premier contiendrait les deux registres à décalage tandis que les 4 autres contiendraient 4 boutons poussoirs chacun. Les 4 petits PCB seraient reliés par quelques connexions filaires simples au PCB central qui serait lui-même relié en 6 points à l'Arduino. Cette solution permet de disposer les 4 groupes de 4 boutons autour de notre écran tout en limitant le nombre de fils à connecter. De plus, elle simplifie aussi le processus puisqu'il n'y a pas à anticiper la correspondance entre le PCB central et l'Arduino pour les clipser. Ainsi pour m'assurer d'avoir bien solutionner le problème avec l'aide de mon groupe et de ma tutrice, j'ai réalisé un dessin schématique de l'ensemble du dispositif et des PCB. Cela m'a permis aussi de m'assurer de ne pas faire d'erreurs de connexion lors de la conception du PCB. Cependant j'ai encore eu quelques soucis concernant les empreintes des boutons poussoirs OMRON B3F. Mme LEGALL m'a aidé à trouver l'empreinte correcte pour imprimer des PCB adaptés.

Concernant le résultat, à l'heure où ce rapport est écrit le PCB (avec les empreintes correctes)des boutons n'a pas été imprimé en raison de délai et de disponibilité des machines requises mais grâce aux tutoriels de prise en main du logiciel Eagle et aux conseils de conception donnés, j'ai pu créer le PCB principal contenant les registres. Celui-ci est fonctionnel et a été

percé. Cependant, arriver à ce résultat m'a demandé de nombreuses heures car le logiciel Eagle a une interface difficile à prendre en main.

d. Retour d'expérience - Sara

Question : Racontez-moi une situation dans ce projet où vous étiez confronté.e à un problème qui avait plusieurs solutions possibles.

Dans ce projet, j'ai été chargé du développement des codes des modes, en étroite coordination avec les autres pôles du projet. J'étais en binôme avec Judith, qui était également chargée de la partie tests sur le matériel, ainsi à chaque retour ou dysfonctionnement de sa part, je modifiais et corrigeais le code en conséquence. J'étais aussi en communication avec l'équipe responsable du circuit électronique. L'équipe me tenait informé des solutions techniquement viables ou des aspects trop compliqués à mettre en œuvre, et j'adapterais le code pour qu'il soit à la fois fonctionnel et en phase avec la réalité du montage électronique. Cette collaboration m'a permis d'ajuster les scripts de manière à répondre au mieux aux exigences concrètes du projet.

Pendant l'élaboration du mode Précision, j'ai rencontré un défi technique : comment signaler au joueur non seulement la couleur ciblée, mais aussi l'emplacement précis du bouton à presser, étant donné qu'il y a quatre boutons de la même couleur disposés à divers emplacements sur l'écran.

J'ai examiné diverses options :

1- Utiliser quatre LED NeoPixel disposées autour de l'écran LCD, une pour chaque orientation (haut, bas, gauche, droite). Ceci permettait de signaler la position de façon lumineuse et intuitive.

2- L'utilisation d'une LED unique pour la couleur et l'affichage de la position en texte sur l'écran LCD (par exemple : « Appuyer sur VERT HAUT GAUCHE »).

3- Activer les quatre LED de la même couleur, à l'exception d'une LED qui clignote pour signaler la position correcte.

Chaque option présentait ses propres atouts :

La première était très intuitive pour le joueur, mais trop difficile à réaliser en termes de câblage et d'alimentation selon l'équipe d'électronique.

La troisième option était innovante, mais demandait une administration plus sophistiquée (chronomètres, clignotement autonome).

-> La seconde, qui était techniquement plus facile, offrait aussi une meilleure fiabilité avec les moyens à notre disposition.

Nous avons finalement opté pour la seconde option, qui impliquait :

L'utilisation d'une unique LED NeoPixel pour la couleur, et montrer exactement l'emplacement sur l'écran LCD mais avec quatres cercles animés qui montrent la position exacte du bouton.

Cette décision nous a donné la possibilité de garantir une clarté optimale pour l'utilisateur, tout en tenant compte des limitations matérielles et temporelles du projet.

e. Retour d'expérience - Niamatallah

Question : Racontez-moi une situation dans ce projet où vous étiez confronté.e à un problème qui avait plusieurs solutions possibles.

Mon plus grand problème au début de ce projet, c'est que je ne savais pas coder en JavaScript ni utiliser des outils comme Node.js, React ou les API HTTP. J'ai dû tout apprendre toute seule en parallèle du projet, alors que ces notions ont été abordées bien plus tard en cours. Ça m'a clairement mis en difficulté dès le départ, parce que pendant que d'autres attendaient les séances de TD, moi je me battais déjà avec des erreurs de compilation, des problèmes de communication série, et une interface web vide qui ne réagissait à rien.

L'un des moments les plus complexes a été lorsqu'on a voulu afficher les résultats des jeux (réflexe, mémoire, précision) dans l'interface admin web. Le terminal me montrait bien les résultats en provenance de l'Arduino, donc la communication fonctionnait côté hardware. Mais côté web, aucune donnée ne s'affichait, ce qui m'a bloquée plusieurs jours.

J'ai analysé le problème et identifié plusieurs solutions :

1. Créer une variable globale (`gameStats`) côté serveur pour stocker les résultats reçus.
2. Utiliser un système d'événements asynchrones avec `EventEmitter` pour faire réagir l'interface dès qu'une ligne est reçue.
3. Mettre en place un buffer d'attente ou un système de polling pour attendre la prochaine donnée avant d'afficher quoi que ce soit.

Chaque solution avait ses risques. J'ai finalement choisi la première parce qu'elle était plus stable et plus simple à maintenir, surtout dans un contexte débutant. Une fois les résultats bien stockés dans `gameStats`, j'ai créé des routes API dédiées (`/api/getReflexeResults`, `/api/getMemoireResults`, etc.) et modifié l'interface React pour aller chercher ces données via axios.

Grâce à cette solution, j'ai pu :

- sécuriser la communication entre l'Arduino et l'interface web,
- afficher les scores, les temps moyens, et masquer certains détails inutiles selon le mode,
- rendre l'interface fluide et lisible pour un admin non technique

f. Retour d'expérience - Nihal

Questions: Décrivez le problème dans son contexte. Comment avez-vous réagi initialement ? Comment avez-vous déterminé la solution à choisir ? Quel a été le résultat de votre choix ? Qu'avez-vous retenu de cette expérience ?

Dans le cadre de notre projet PRONTO, nous avons décidé de concevoir un jeu dont l'objectif final est de fournir un outil de suivi pour les médecins, afin d'évaluer différentes capacités chez leurs patients, telles que la mémoire ou les réflexes. Pour cela, nous devons concevoir un circuit électronique intégrant 16 boutons, un écran, des LEDs et un capteur. Cependant, le nombre de broches disponibles sur une carte Arduino standard est très limité.

Au début du projet, je faisais partie du groupe chargé de la réalisation du circuit. Après la répartition des tâches, nous avons rapidement été confrontés à ce problème majeur et avons cherché une solution.

Pour y remédier, nous avons entrepris plusieurs démarches :

- Recherche d'informations :

Nous avons exploré différentes méthodes pour augmenter le nombre de broches disponibles sur une carte Arduino. Nos recherches se sont concentrées sur des solutions directement liées à la carte, telles que :

- L'utilisation de dispositifs externes permettant d'augmenter le nombre d'entrées ;
- Le recours à une carte Arduino Mega, qui offre 16 entrées analogiques et plus de 50 entrées/sorties digitales.

- Demande de conseils à notre encadrante :

Grâce à son intervention, nous avons découvert une solution à la fois plus simple et plus pratique, ne nécessitant pas l'achat de nouveau matériel : l'utilisation de registres à décalage. Cette technique nous a permis de connecter tous les boutons à la carte Arduino, sans avoir à augmenter physiquement le nombre d'entrées.

On a choisi d'utiliser des registres à décalage qui nous a permis de connecter les 16 boutons à la carte Arduino sans dépasser le nombre de broches disponibles. Le circuit a fonctionné correctement et a répondu aux besoins du projet.

Nous avons appris à chercher des solutions techniques efficaces, à tirer parti des conseils de notre encadrante, et à optimiser les ressources matérielles auxquelles on a accès.

OUR WORLDWIDE PARTNERS UNIVERSITIES - DOUBLE DEGREE AGREEMENTS



3 CAMPUS



IMT Atlantique Bretagne-Pays de la Loire – <http://www.imt-atlantique.fr/>

Campus de Brest

Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 3
France
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes

4, rue Alfred Kastler
CS 20722
44307 Nantes Cedex 3
France
T +33 (0)2 51 85 81 00
F +33 (0)2 99 12 70 08

Campus de Rennes

2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
France
T +33 (0)2 99 12 70 00
F +33 (0)2 51 85 81 99