1. Introduction

- Présentation du Projet

Dans le cadre de l'exploration des capacités du développement en C++ orienté objet et de l'intégration de matériel électronique, ce rapport se penche sur un projet de jeu interactif impliquant l'usage de divers composants électroniques et la programmation sur une plateforme Arduino. Le jeu, conçu comme un challenge de mémoire et de réflexe, utilise des LED, des boutons poussoirs, un écran LCD, et un contrôleur Arduino pour créer une expérience ludique et éducative.

- Idée du projet

L'objectif principal de ce jeu est double. D'une part, il vise à fournir une plateforme interactive où les utilisateurs peuvent tester et améliorer leur mémoire et leurs réflexes en suivant et en reproduisant des séquences de lumières. D'autre part, du point de vue du développement, il sert d'exemple concret pour illustrer les concepts avancés de programmation en C++ dans un environnement embarqué. Le jeu est conçu pour être évolutif et modulable, permettant ainsi des modifications ou des extensions futures, telles que l'ajout de niveaux de difficulté, l'intégration de nouvelles fonctionnalités, ou même la refonte pour s'adapter à différents types de matériel.

2. Scénario / USE CASE

Démarrer le Jeu :
- L'utilisateur appuie sur le bouton de démarrage.
- Le jeu initialise le niveau et affiche les instructions sur l'écran LCD.

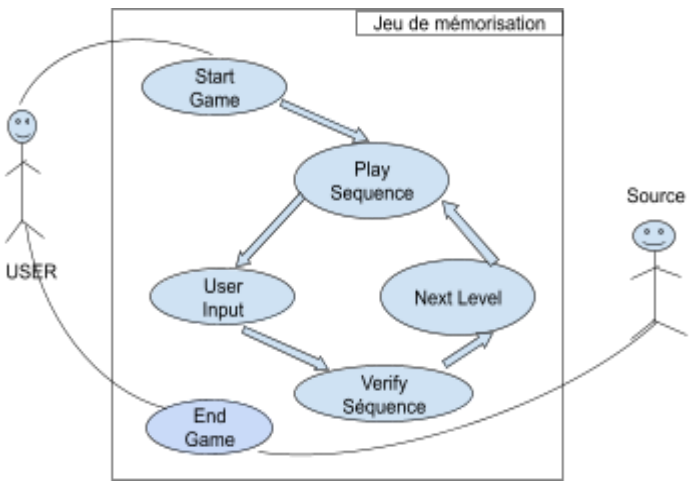
Jouer une Séquence :
- Le jeu allume successivement les LED dans un ordre spécifique pour créer une séquence.
- L'utilisateur observe cette séquence.

Entrée Utilisateur :
- L'utilisateur appuie sur les boutons correspondant à la séquence des LED.
- Le jeu reçoit et enregistre l'entrée de l'utilisateur.

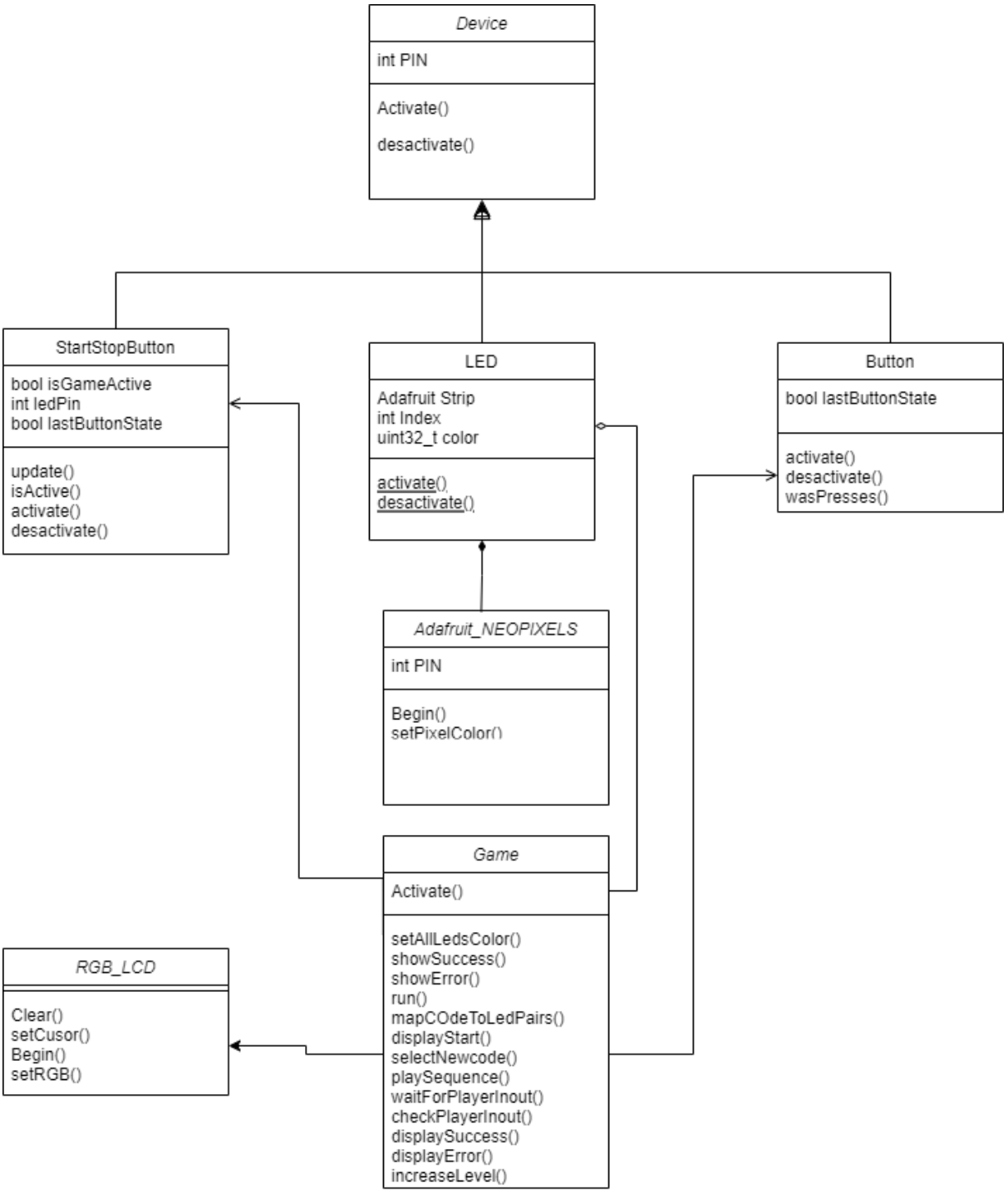
Vérification de la Séquence :
- Le jeu compare la séquence entrée par l'utilisateur avec la séquence initiale.
- Le jeu affiche le résultat succès (LED tout en VERT) ou échec (LED tout en ROUGE)

Passage au Niveau Suivant :
- En cas de succès, le jeu augmente le niveau de difficulté et génère une nouvelle séquence.

Fin du Jeu :
- L'utilisateur peut arrêter le jeu à tout moment en appuyant sur le bouton d'arrêt.



3. Diagramme de Classe



Le projet de jeu repose sur plusieurs classes essentielles : `Device`, qui sert de base abstraite pour les périphériques, et ses dérivées `LED`, `Button`, et `StartStopButton`, chacune gérant des aspects spécifiques du jeu comme les LEDs, les boutons, et le contrôle du jeu. La classe `Game` coordonne l'ensemble du jeu, reliant toutes ces composantes.

4. Redéfinition d'Opérateur

La redéfinition de l'opérateur `==` dans le projet permet la comparaison directe entre deux objets, comme des boutons, pour vérifier s'ils sont identiques en fonction de critères définis, tels que leur état ou position. Cette approche simplifie le code et rend les comparaisons plus intuitives.

5. Utilisation de la STL (Standard Template Library)

Le projet utilise la Standard Template Library (STL) de C++ pour la gestion des données, notamment std::vector pour les collections d'objets tel que les LEDS et BOUTONS et std::string pour les chaînes de caractères comme dans la manipulation des séquences de jeu ou l'affichage sur l'écran LCD, améliorant ainsi la flexibilité et l'efficacité du code.

6. GESTION DES ERREURS OU EXCEPTIONS

La programmation Arduino, basée sur C++, ne supporte généralement pas les exceptions dans le sens traditionnel de la programmation orientée objet.
On implémente une sorte de gestion d'erreurs en utilisant des vérifications conditionnelles et des messages d'erreur.
Vérification de l'état du bouton : Dans la classe `Button`, On ajoute des vérifications pour s'assurer que l'état du bouton est valide.
Validation de l'entrée du joueur : Dans la méthode `waitForPlayerInput` de la classe `Game`, On s'assure que l'entrée du joueur est valide (par exemple, pas en dehors des limites d'index).

7. CONCLUSION

Les objectifs atteints sont d'avoir réussi à créer un jeu interactif avec un ESP8266, en respectant toutes les directives du projet jusqu'à obtenir un jeu fonctionnel, ludique et amusant. Nous avons également réussi à implémenter les bibliothèques et à appliquer le principe d'héritage.
Durant le développement, on a dû faire face à pas mal de problèmes. Par exemple, utiliser une boucle `while` dans la fonction `loop` posait un risque de blocage de l'ESP8266, mais on a réglé ça en utilisant `yield()`, qui permet au système de rester stable et réactif. Aussi, afficher des messages sur l'écran LCD était compliqué, surtout pour les parties plus complexes du jeu.
Pour l'avenir, le projet a plein de potentiel pour s'améliorer et s'agrandir. On pense notamment à ajouter plus de LEDs et de boutons pour rendre le jeu plus complexe et intéressant. On pourrait aussi varier les niveaux de difficulté, améliorer les effets sonores et visuels, et rendre l'interface utilisateur plus facile à utiliser.