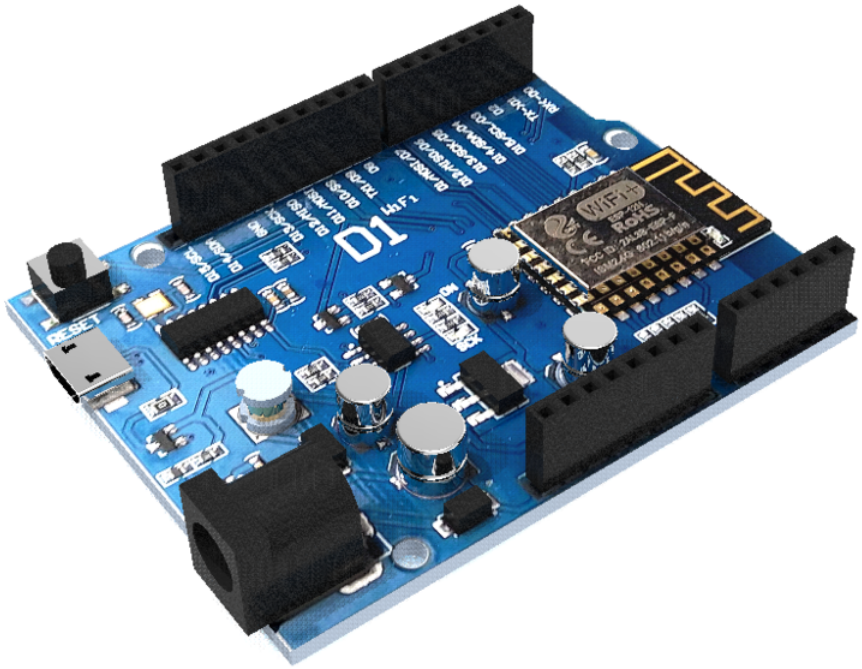
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Compte-rendu de BE de programmation orientée objet : Protech(t), a Health Monitoring App



Réalisé par
Assia NGUYEN
Agathe LIEVRE

Dans le cadre du cours de
Programmation Orientée Objet

Travail présenté à
Thierry Monteil et Raphael Deau
Du département de génie électronique et informatique

Table des matières

1 Introduction

En cette période difficile où la santé est au centre de notre vie, nous avons voulu développer un projet pour aider les personnes les plus à risque : les personnes âgées. Nous voulons leur donner une solution simple pour surveiller leur santé et prévenir les secours en cas de besoin. C'est ainsi que nous avons eu l'idée d'un *health monitoring* qui surveille le rythme cardiaque, la température et les chutes.

Ce projet entre parfaitement dans les conditions voulues du BE de C++ car il utilise une multitude de capteurs et d'actionneurs. En effet, nous utilisons :

— un oxymètre,
— un capteur de température,
— un accéléromètre,
— des boutons,
— un buzzer,
— et une LED.

De plus, pour chaque capteur, nous avons créé une classe en C++. Cela permet de faire entrer en jeu l'héritage et les relations d'amitié entre classes. Nous avons également utilisé les mécanismes d'exceptions et de redéfinition d'opérateur.

Dans ce rapport, nous décrirons dans un premier temps nos objectifs. Ensuite, nous allons exposer les cas d'utilisation de notre système de surveillance et le déroulement de notre projet. Enfin nous aborderons les problèmes rencontrés lors de ce projet ainsi que les perspectives d'utilisations envisagées.

2 Conception et scénarios d'utilisation

2.1 Choix des fonctions et objectifs

Nous avons commencé notre conception en décidant quelles fonctions nous voulions implémenter dans notre projet. Notre but était de surveiller le rythme cardiaque, la température et l'éventuelle chute d'une personne âgée et afficher toutes ces informations sur un écran. De plus, si la personne âgée présente des risques, nous voulons l'informer par une alarme visuelle et sonore et prévenir les secours pour qu'ils interviennent le plus rapidement possible.

De plus, nous avons décidé d'implémenter une fonctionnalité liée au module WiFi de l'ESP8266 : celle qui permet à la personne âgée de prévenir les secours en un clic de bouton pour n'importe quelle autre raison. Le bouton sert aussi à éteindre l'alarme si l'urgence n'en est finalement pas une.

Enfin, un bouton permet d'éteindre et d'allumer l'écran à tout moment.

2.2 Diagramme de classe

En annexe, dans la figure 1, vous pouvez voir notre tout premier diagramme de classe qui a bien changé depuis le début du projet.

Puis, à la figure 2, vous pouvez voir le diagramme de classe final avec les changements et améliorations inclus.

Nous avons aussi implémenté des surcharges d'opérateurs et des exceptions. Dans la classe *Menu*, nous avons surchargé l'opérateur d'indiçage [] et dans la classe *Monitoring*, l'opérateur ++ et −−. Ces opérateurs ++ et −− incrémente ou décrémente l'attribut `cursorPosition`. De plus, nous avons implémenté deux exceptions :

- la première `invalid_argument`, à la création d'un objet Menu, vérifie que le nombre de choix est bien supérieur ou égal au numéro du dernier choix possible et que le numéro du premier choix est inférieur ou égal au numéro du dernier choix,
- la deuxième `out_of_range`, lors de la surcharge de l'opérateur d'indiçage, vérifie que l'indice est bien compris dans la taille du tableau.

2.3 Design des menus

En annexe, à la figure 4

2.4 Scenarios d'utilisation

Au démarrage, notre système de surveillance demande dans un premier temps l'âge du patient. Nous avons borné l'âge de l'utilisateur entre 10 et 119 ans.

Ensuite, un écran d'affichage mentionne en continu les informations suivantes : la fréquence cardiaque, la température, l'accélération verticale, l'âge, la fréquence cardiaque maximale calculée selon l'âge de l'utilisateur et une option "Call for help" et "Settings". Cet âge peut être modifié à tout instant par l'utilisateur en selectionnant cette option affichée en bas sur le menu d'affichage.

Si l'utilisateur se sent en détresse, il peut sélectionner l'option "Call for help". Un SMS est alors envoyé à un numéro d'urgence. Dans notre simulation, le numéro d'urgence est configuré pour être celui d'Assia Nguyen via une applet créée sur IFTTT. (voir figure 5 en annexe)

Ce message est envoyé dans 2 autres scénarios. D'abord, si la température captée est en dessous de 36° ou au dessus de 40°. Ensuite, si l'accéléromètre détecte une chute. Un écran d'appel à l'aide dédié est affiché. Dans ce cas spécifique, si l'utilisateur appuie sur le bouton de sélection dans les 15 secondes suivant la détection de la chute, le sms ne sera pas envoyé. Dans le cas contraire, l'utilisateur sera considéré comme en danger.

Si la température est entre 38° et 40° un écran d'avertissement est affiché pour inciter l'utilisateur à consulter un médecin.

Que ce soit pour le cas d'avertissement ou d'appel à l'aide, une alarme constituée d'un buzzer et du LED se déclenche. Cette alarme se désactive en appuyant sur le bouton de sélection.

A tout moment, l'utilisateur peut éteindre puis rallumer l'écran avec le bouton associé et une page d'adieu va apparaître à l'écran. Au rallumage, on ne réaffiche pas le menu de l'âge mais directement l'écran d'affichage des caractéristiques des utilisateurs.

L'ensemble des cas d'utilisation sont démontrés dans des vidéos disponibles sur le lien Google drive donné au 4.3.

Note : durant notre démonstration, pour éviter le déclenchement de l'alerte si on est en dessous de 36°, nous avons enlevé cette limite dans le code. En effet, le capteur lit la température de la pièce.

3 Déroulement du projet

3.1 Répartition du travail et implémentation

Nous avons commencé l'implémentation en créant toutes les classes en C++ après avoir créé un premier jet de diagramme d'utilisation. Pour être le plus efficace possible dans le projet, nous avons identifié les principales tâches à réaliser et nous les sommes réparties. Assia était chargée des méthodes et classes des différents capteurs et Agathe s'est chargée de la mise en place du display et de la mise en commun de toutes les classes. Après avoir testé chacune des fonctionnalités indépendamment, nous avons rassemble le code pour avoir le système complet fonctionnel.

3.2 Câblage

Nous avons mis un schéma du câblage en annexe, à la figure 6.

3.3 Problèmes rencontrés et dysfonctionnements

Nous avons rencontré quelques difficultés lors de ce projet. Le plus notable est lié à l'oxymètre. Au bout de quelques utilisations il dissipait énormément de chaleur même alimenté sur la tension indiquée sur la datasheet. Au bout d'un certain temps, il ne renvoyait plus aucune valeur en analogique et en numérique. Cependant, nous avons décidé de laisser dans le code et les diagrammes car notre limitation était purement un défaut ou panne technique du capteur. De plus, il y avait des fois où l'accéléromètre renvoyait des valeurs d'accélération en z nulles. Pour remédier à ce problème, nous avons ajouté des conditions dans la détection de chute pour ignorer ces valeurs et éviter de déclencher une alarme sans le vouloir.

4 Conclusion

4.1 Résultats

En conclusion nous sommes très satisfaites à l'issu de ce bureau d'études car nous avons réussi à mener notre projet à bien en créant un système de surveillance fonctionnel et efficace. Le fait d'avoir pu choisir nous-mêmes notre sujet nous a donné une motivation exacerbée lors de notre travail orienté objet.

4.2 Perspectives

Pour améliorer notre système, il serait intéressant d'insérer un oxymètre fonctionnant correctement dans le circuit.

Pour avoir un système fonctionnel en réalité il faudrait embarquer le système totalement pour que le patient puisse le porter en permanence. Une autre amélioration qui pourrait être amenée serait l'utilisation de l'accéléromètre pour détecter des crises d'épilepsie ou un chancellement inhabituel.

On pourrait envisager en plus d'envoyer un SMS à la personne de confiance du patient en situation de risque, d'appeler un numéro de secours comme les urgences. Enfin, on pourrait insérer un écran qui affiche l'état de recherche du WiFi et une mention qui indique si le système est bien connecté à un réseau WiFi sur le display.

4.3 Liens utiles

Lien du repository GIT :
https://github.com/Assianguyen/BE_Cpp

Lien drive des vidéos de démonstration :
https://drive.google.com/drive/folders/1GCRVf0l4m1-jvIOpZmPmu1rzHZRewxsx?usp=sharing
Lire avec le lecteur VLC

5 Annexes

5.1 Conception

5.1.1 Diagrammes de classe

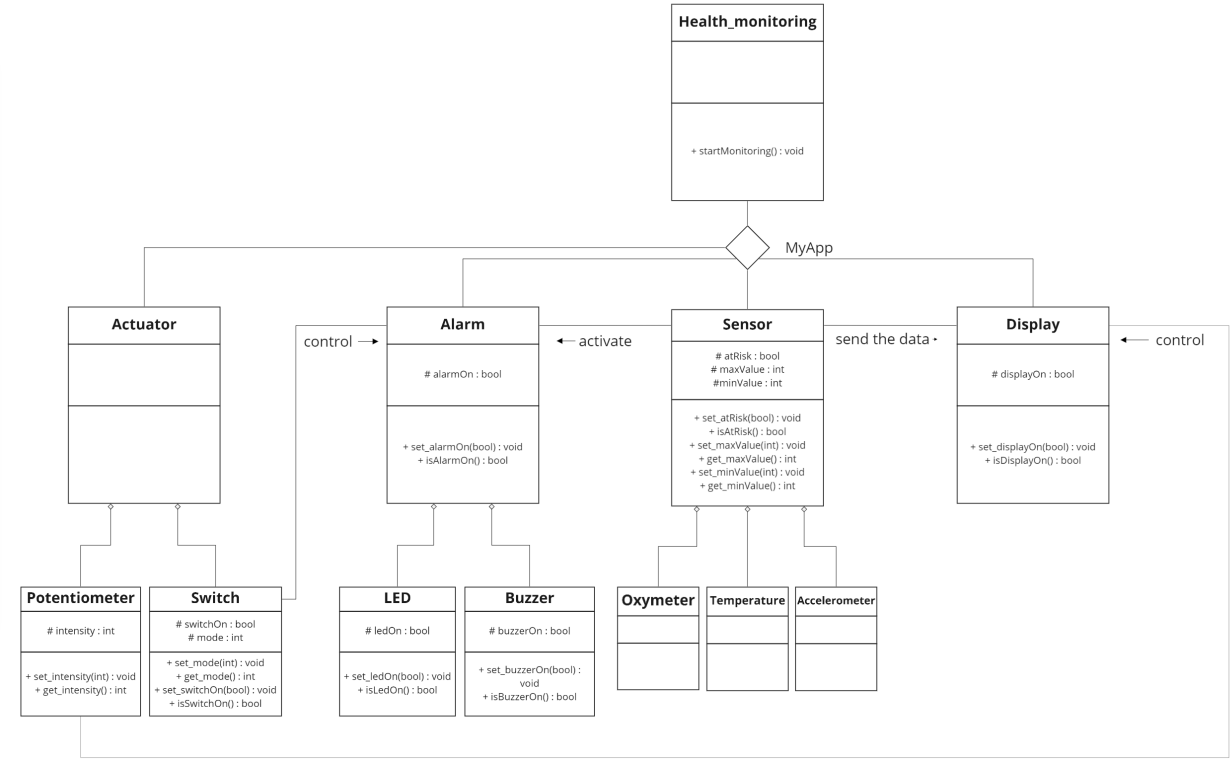


FIGURE 1 – Premier diagramme de classe

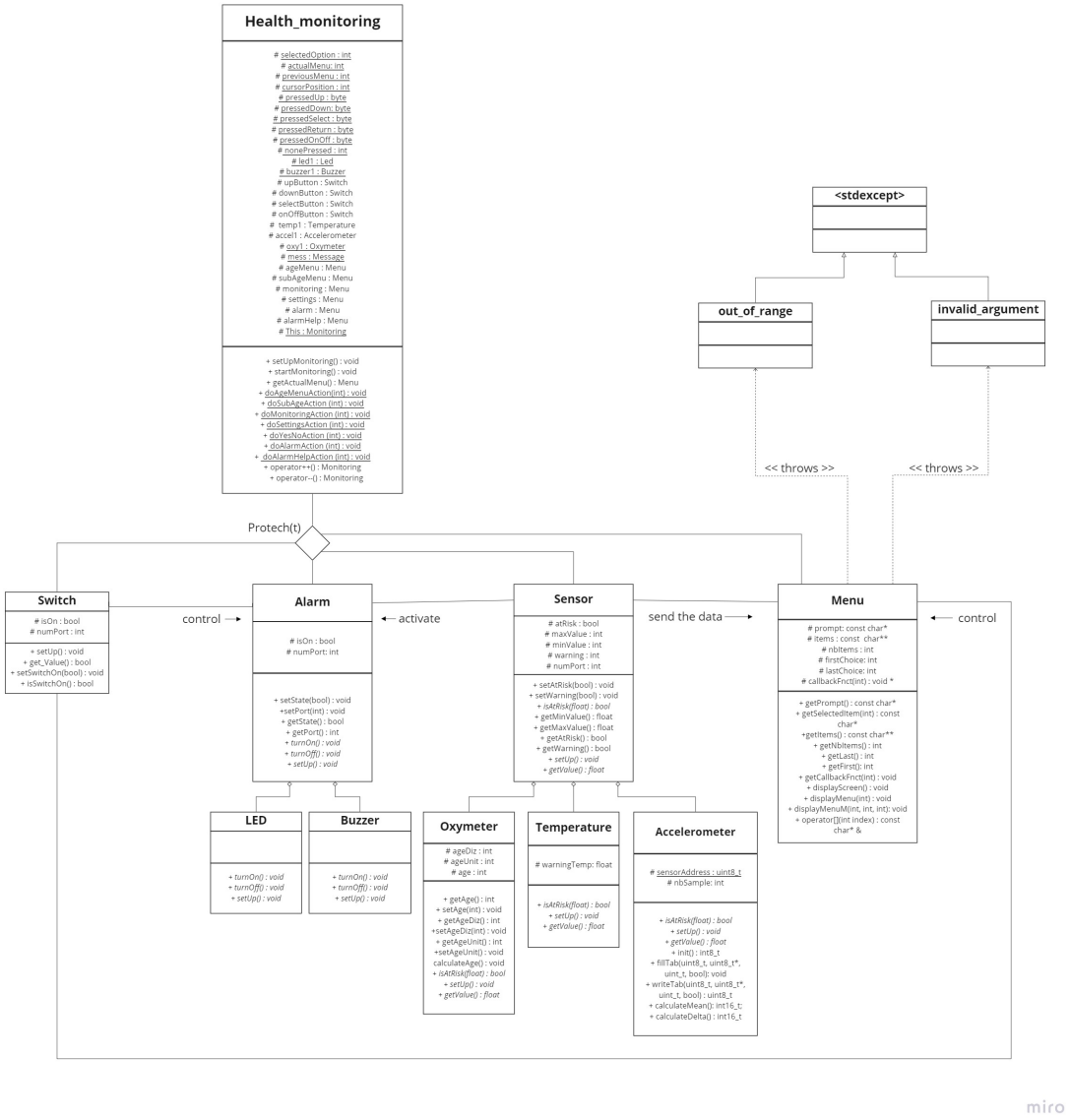


FIGURE 2 – Diagramme de classe final

FIGURE 3 – Diagramme de cas d'utilisation

5.1.2 Design des menus

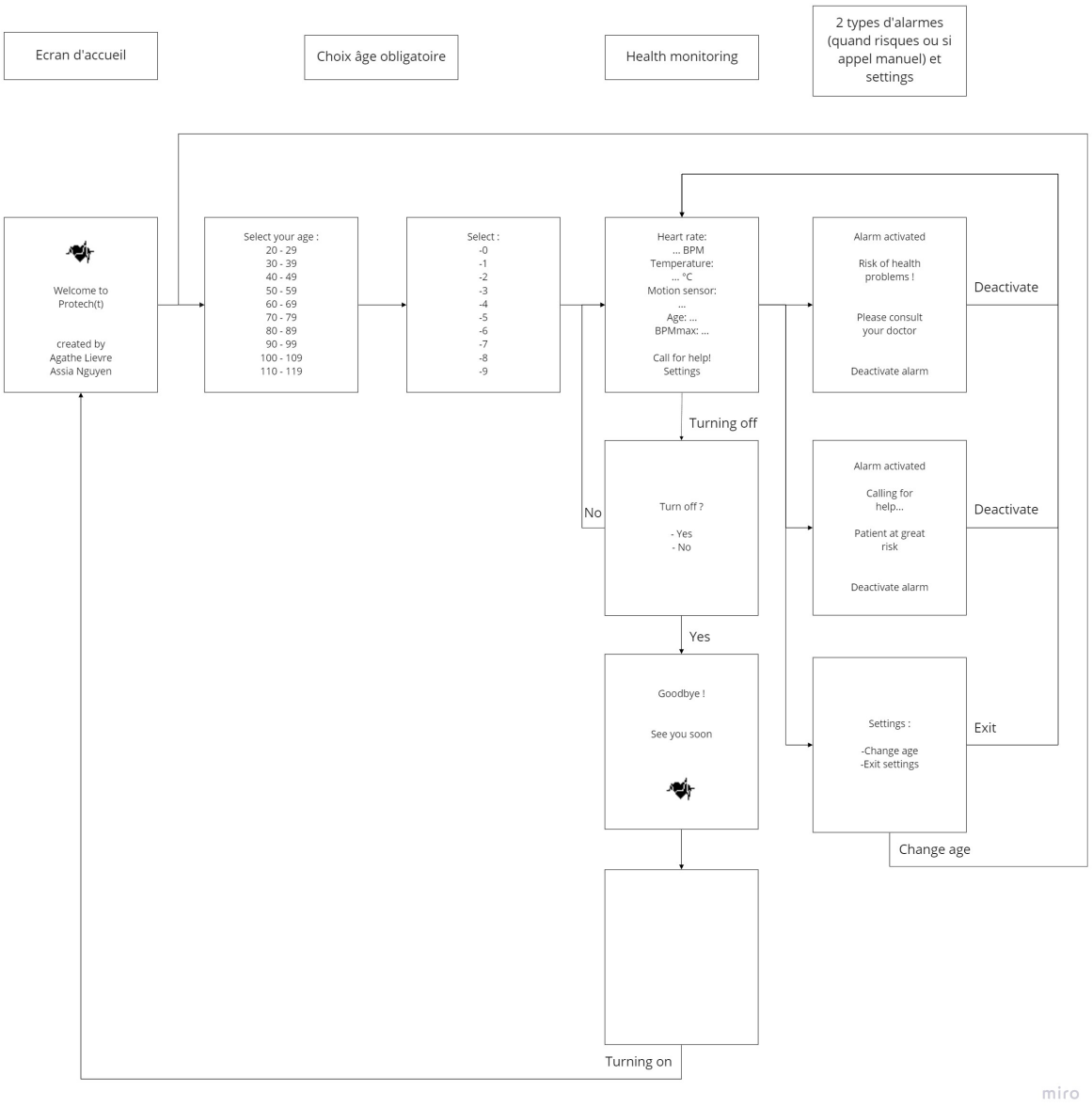


FIGURE 4 – Design du menu

5.2 Implémentation

5.2.1 Code

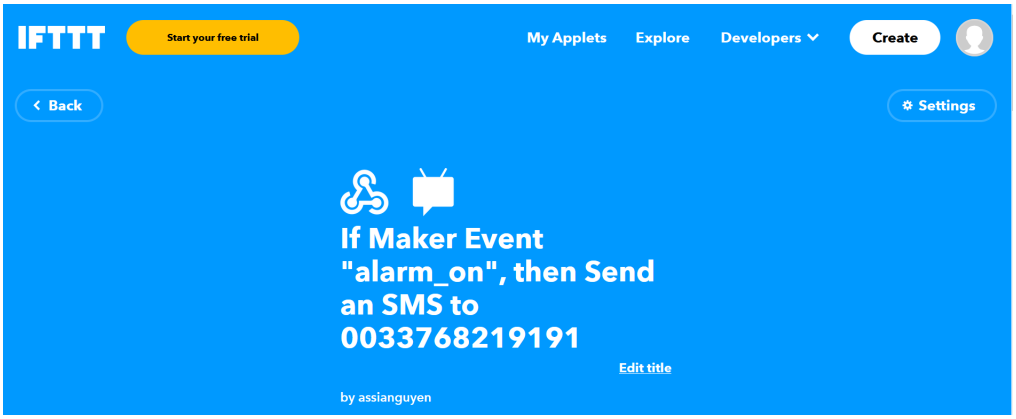


FIGURE 5 – Screen de l'applet sur IFTTT pour l'envoi de SMS

5.2.2 Câblage

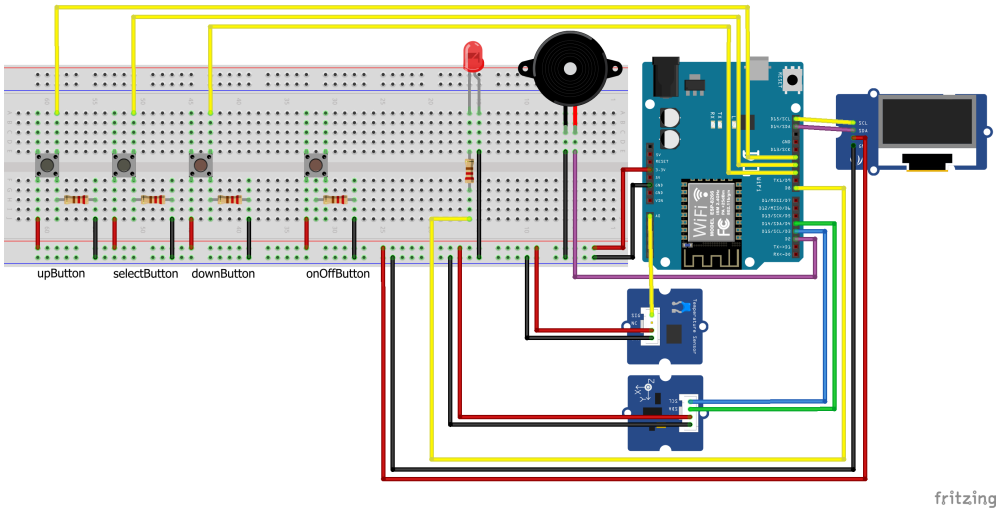


FIGURE 6 – Câblage du projet