الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة أبي بكر بلقايد– تلمسان

Université Aboubekr Belkaïd-Tlemcen

كلية التكنلوجيا

Faculté de Technologie

Département de Génie Electrique et Electronique (GEE)

Filière : Electronique

MASTER INSTRUMENTATION

PROJET DE FIN D'ETUDES

Présenté par : Messaoud Mohammed Riad **&** Zehar Mohammed

Intitulé du Sujet

Etude et réalisation d'un système d'appel malade pour centre hospitalier

Soutenu le 01/07/2021, devant le jury composé de :

Mme BOUAZZA née GUEN Ahlem	**Pr Univ.**	Univ. Tlemcen	**Président**
Mr BRIXI NIGASSA M.E.A	**MCB**	Univ. Tlemcen	**Encadrant**
Mr SLAMI Ahmed	**Doctorant**	Univ. Tlemcen	**Co-Encadrant**
Mr MOULAI KHATIR Ahmed Nassim	**MCB**	Univ. Tlemcen	**Examinateur**

Année Universitaire 2020-2021

Remerciements :

Nous remercions tout d'abord Dieu pour ses bienfaits inestimables, de nous avoir illuminé le chemin vers le savoir et la science et de nous avoir permis de terminer nos études ainsi que ce projet dans de bonnes conditions.

Nous tenons à exprimer toute notre reconnaissance à nos encadreurs Monsieur **Brixi Nigassa Mohammed El Amine Amine et** Monsieur **Slami Ahmed** nous les remercions de nous avoir encadrés, orientés, aidés et conseillés.

On est reconnaissants pour le temps qu'ils nous ont accordé, leurs qualités pédagogiques et scientifiques, leur franchise et leur sympathie. On a beaucoup appris d'eux et on leur adresse notre gratitude pour tout cela.

Nos remerciements s'adressent aussi à **Mme BOUAZZA née GUEN Ahlem, professeur** de la Faculté de technologie Université d'Abou-Bekr Belkaid Tlemcen, pour l'attention qu'il a bien voulu porter à ce travail en acceptant de le jurer et le discuter.

On tient aussi à remercier **Mr MOULAI KHATIR Ahmed Nassim**, professeur à l'Université d'Abou Bekr Belkaid Tlemcen, pour l'intérêt qu'il a accordé à ce travail en acceptant de l'examiner.

Nos vifs remerciements s'adressent à Messieurs **Benomari Houari, Réda, Mohammed Beloufa, Khalifa Mohammed** pour leurs précieux conseils, leurs orientations fructueuses, leurs encouragements et leurs grandes contributions dans cette étude.

Merci de votre aide chaleureuse, veillez trouver ici l'expression de notre reconnaissance et de notre vive gratitude.

Enfin nous remercions tous ceux qui ont contribué de près ou de loin à

L'aboutissement de ce travail

Dédicace

Nous dédions ce modeste mémoire à :

Nos chers parents : Que nulle dédicace ne puisse exprimer ce que nous leurs

Devons, pour tous leurs sacrifices, leur bienveillance, leur amour, leur tendresse, leur soutien et leurs prières tout au long de nos études.

Que ce travail soit témoignage de notre profond amour et notre grande reconnaissance « Que Dieu vous garde ».

Nos chères sœurs et nos chers frères : pour leurs encouragements permanents, et leur soutien moral Nous leur dédions ce modeste travail en témoignage de notre grand amour et notre gratitude infinie.

Toutes nos Familles : à toutes les familles MESSAOUD & BELBACHIR, pour leur soutien tout au long de mon parcours universitaire, Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fuit de votre soutien infaillible.

Tous nos amis : Pour leur aide et leur soutien moral durant l'élaboration de ce

travail de fin d'étude.

Messaoud Mohammed Riad

Dédicace

Je dédie ce modeste travail :

Aux êtres qui me sont les plus chers ma mère et mon père. Que Dieu préserve bonne santé et longue vie. Qui ont tous fait pour m'encourager durant les années de mes études

A mes chères sœurs

A mes Grands Parents que Dieu les protège et à toute ma famille. A mes Oncles et mes Tantes.

A mes chers cousins :Ali,Badiss, Anouar, Aymen, Abdelhadi, Abderrahmane,...

A mon cher ami : Riad

A mes chers amis : Redouane , Anes , Amine , Mohammed

Ainsi que tout le groupe de sport et la promotion Master GEE sans oublier tous mes enseignants durant tout mon cursus.

Zehar Mohammed

Sommaire

Liste des figures

Introduction générale

La santé est un droit fondamental à chaque citoyen. Chaque pays dispose d'un système de santé permettant de soigner ses malades dans les meilleurs conditions et conforts. Néanmoins, la crise sanitaire par laquelle est passé le monde ces deux dernières années a mis à mal tous les systèmes hospitaliers et le personnel soignant.

Afin de permettre une meilleure gestion de ce personnel médical ainsi qu'une meilleure fluidité et réactivité de ce dernier, plusieurs systèmes existent. L'un des plus connus et les plus utilisés est le système d'appel malade (nurse call en anglais). Des systèmes d'appel malade, il en existe plusieurs types sur le marché : filaire et sans fil. Ce que nous proposons dans ce travail c'est un appel malade composé d'une carte électronique disposant d'un module Wifi et communiquant avec une application Android installée sur les smartphones de l'ensemble du personnel soignant.

L'objectif ici est que lorsqu'un malade a besoin d'assistance, il n'aura qu'à appuyer sur un bouton installé à son chevet. Cette action est récupérée par la carte électronique et transférée par Wifi vers les smartphones du personnel médical. Le soignant prenant en charge le malade n'aura alors plus qu'à désactiver via son smartphone l'alerte afin de dire à ses collègues de ne pas se déplacer à la chambre du malade sur place. En plus, un afficheur OLED sera installé au niveau de la réception du centre hospitalier, ceci permettra d'alerter aussi la réception en cas de problème de réseau ou en cas de retard de prise en charge.

Pour mener à bien ce travail, nous avons divisé ce travail en 3 chapitres :

Dans le premier chapitre, nous commencerons par un petit historique sur la création du système d'appel malade. Nous définirons par la suite ce dernier avant de citer quels sont les types d'appel malade qui existent. Nous montrerons aussi où est-ce qu'on peut trouver de tels systèmes et quelques dispositifs commerciaux existants sur le marché.

Dans le deuxième chapitre, nous détaillerons les composants matériels et logiciels qui nous ont été nécessaires pour mener à bien notre projet. Nous commencerons par présenter l'Arduino de manière générale avant de parler de la carte NodeMCU ESP8266 intégrant un module Wifi utilisé dans notre projet. Nous parlerons par la suite de l'afficheur OLED I2C utilisé. Nous entamerons après cela la description de l'outil MIT App Inventor (AI2) que nous avons choisi pour développer notre application Android. Nous terminerons par quelques notions sur l'outil Firebase de Google qui nous a été nécessaire pour la création de notre base de données.

Dans le troisième chapitre, nous détaillerons le concept de notre système d'appel malade en présentant dans un premier temps le circuit réalisé à base de carte NodeMCU ESP8266. Nous détaillerons aussi l'application Android ainsi que la base de données Firebase créées. Nous terminerons ce chapitre en montrant le fonctionnement de notre système.

Ce travail sera clôturé par une conclusions générale et quelques perspectives en vue d'améliorer ce travail.

I. CHAPITRE 1 : Généralités sur l'appel malade

I.1 Introduction :

Dans ce premier chapitre, nous commencerons par définir le système d'appel malade en donnant un petit historique sur la création de ce dernier. Nous verrons pourquoi nous utilisons de tels systèmes et dans quels buts. Quelques systèmes déjà existant seront cités avant de donner et présenter le concept de notre système d'appel malade.

I.2 Histoire de l'appel malade :

Dans le milieu des années 1800, durant la guerre de Crimée, une brillante et charismatique infirmière Britannique nommée « **Florence Nightingale** » remarque la nécessité pour ses patients de faire appel au personnel soignant à distance. Pour pallier à ce problème, elle a eu la brillante idée à l'époque d'installer une sonnette au chevet de ses patient, comme ça ils pouvaient lui faire appel à n'importe quel moment juste en actionnant la cloche [1].

Figure I-1 – Cloche utilisée par « Florence Nightingale » [2].

Grâce à ce système, Florence Nightingale avait inventé ce que l'on appelle aujourd'hui « appel malade ». Elle sortira même une phrase très célèbre qui restera dans les annales, elle dira : "**Without a system of this kind, a nurse is converted to a pair of legs**"[3], ce qui veut dire que Sans un système de ce type, une infirmière est convertie en une paire de jambes pour monter et descendre les escaliers.

Ce système a subi au fil du temps plusieurs innovations et améliorations jusqu'à arriver en 1980 où les cloches ont pour la première fois été remplacées par un système de notification sonore et lumineuse et d'une unité centrale de traitement [2]. De tels systèmes sont toujours très utilisés aujourd'hui, néanmoins, l'arrivé de nouvelles technologies de communications ainsi que de l'informatique n'ont cessé d'améliorer ces systèmes et plusieurs fabricants aujourd'hui

proposent des systèmes très perfectionnés, même si souvent cette perfection a le plus souvent un coût [4].

Avant de présenter quelques-uns de ces systèmes, nous allons déjà commencer par définir l'appel malade.

I.3 Définition de l'appel malade :

L'appel malade (ou appel infirmière tiré du mot en Anglais Nurse Call) est un système dont le rôle est simple, faire communiquer un patient avec le personnel soignant en cas de besoin ou nécessité. En effet, le patient peut actionner à n'importe quel moment un matériel par exemple (une poire, une télécommande, un bouton) disponible à son chevet ou au sanitaires afin d'alerter le personnel médical à distance qu'il a besoin d'assistance [5].

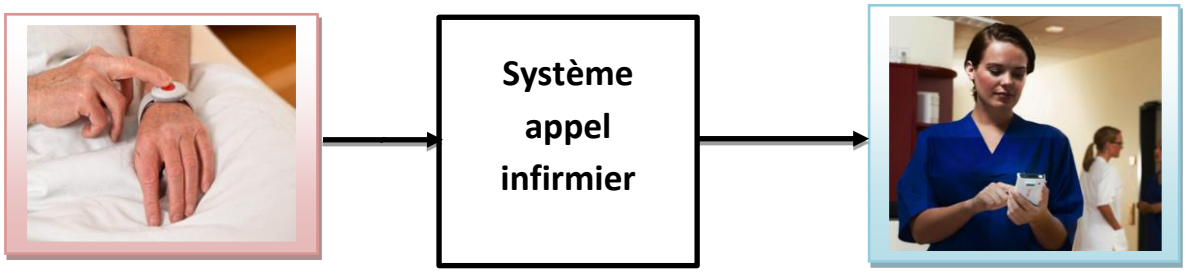


Figure I-2 : Principe de fonctionement d'un système d'appel malade.

Le personnel médical peut être ainsi alerté de différentes manières par exemple par le biais d'une signalisation lumineuse et sonore comme montre la **figure 1.3**

Figure I-3 : Système d'appel malade type HESTEC [6].

I.4 Types d'appel malade :

Il existe deux principaux types de systèmes d'appel malade : filaire et sans fil.

I.4.1 Appel malade filaire (Wired Nurse Call) :

Ce type d'appel malade repose comme son nom l'indique sur une architecture filaire. Ceci impose une installation longue et parfois lourde pour un établissement de santé. Néanmoins, ce type d'appel malade est considéré comme très fiable, permettant de s'affranchir des interférences électromagnétiques [7].

Figure I-4 : Exemple de système d'appel malade filaire (ASCOM) *[8]*.

I.4.2 Appel malade sans fil (Wireless Nurse Call) :

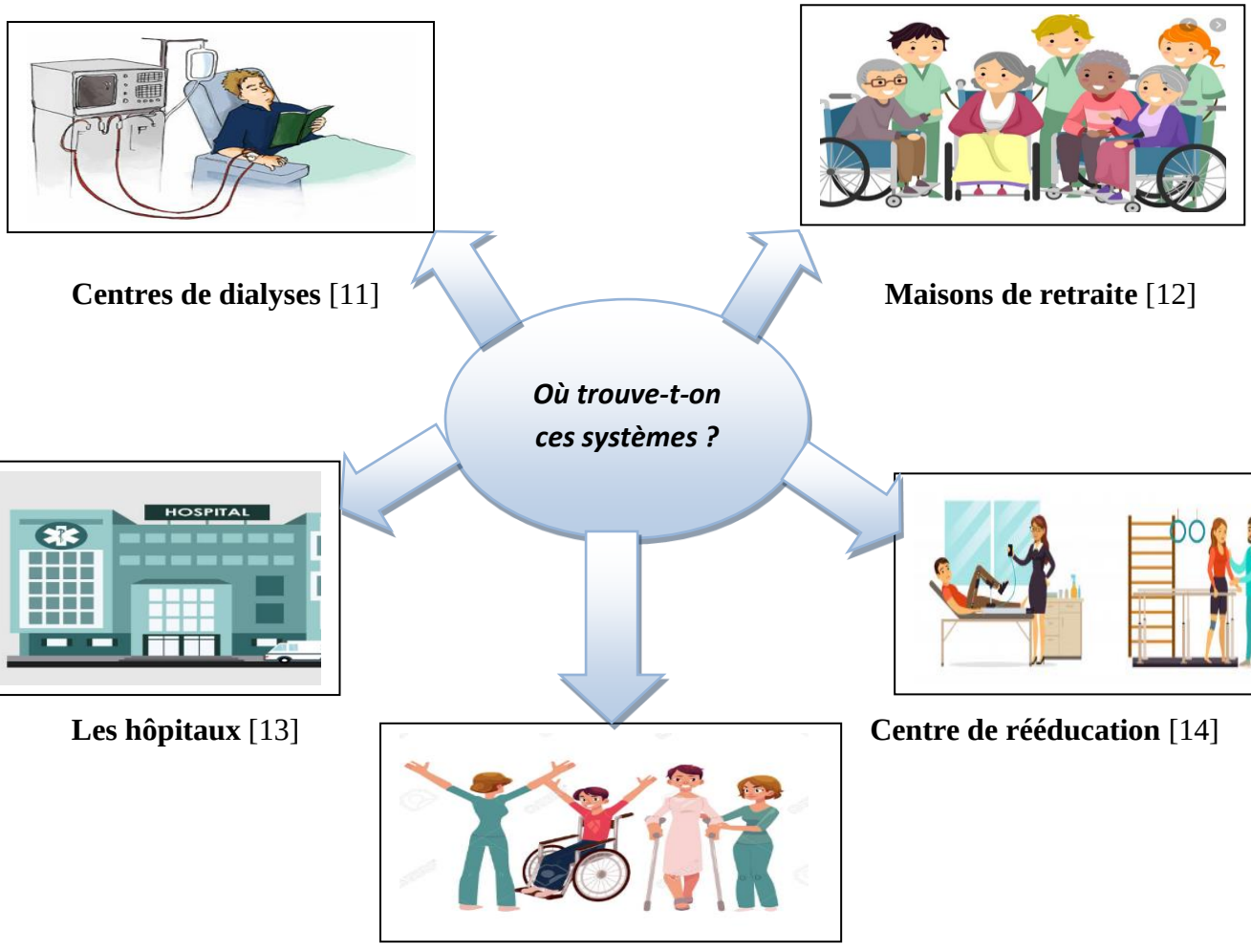
Ce type d'appel malade repose quant à lui sur une architecture sans fil utilisant le plus souvent des ondes radiofréquences, ce qui lui permet de communiquer sur une distance longue portée [9].

Figure I-5 : Exemple de système d'appel malade sans fil (Legrand) [10].

I.5 Où trouve-t-on ces systèmes d'appel malade ?

Contrairement à ce que l'on pourrait penser, un système d'appel malade n'est pas destiné uniquement aux hôpitaux. Ainsi, ces systèmes sont présents dans (voir figure I-6) :

- Les hôpitaux et cliniques privées.
- Les maisons de retraite.
- Les centres d'hébergements pour personnes handicapées.
- Les centres de Soins de Suite et Rééducation.
- Les centres de dialyses.
- Les résidences séniores.



Centres de dialyses [11]

Maisons de retraite [12]

Les hôpitaux [13]

Centre de rééducation [14]

Centre pour personnes handicapées [15]

Figure I-6 : Exemples d'établissements utilisant des systèmes d'appel malade.

I.6 Système d'appel malade dans le commerce :

Comme nous l'avons cité précédemment, il existe des systèmes d'appels malades filaire et sans fil. Nous allons voir dans ce qui suit quelques exemples d'appels malades existant déjà dans le commerce.

I.6.1 Système d'appel malade filaire

I.6.1.1 Système d'appel malade téléphonique :

Ce système est basé sur l'installation d'un standard téléphonique au niveau de la réception d'un centre hospitalier. Lorsqu'un patient demande de l'aide, la personne présente à la réception reçoit son appel sur son standard téléphonique.

(a) (b) (c)

Figure I-7 :Les types de standard téléphonique central : (a) SCHRACK[16]. (b) Yealink T46S [17]. (c) ICall 220 [18].

I.6.1.2 Les hublots de couloir

Les hublots de couloir sont des afficheurs à base de LEDS associées à la chambre du patient. La demande d'assistance du patient sera sous la forme d'un signal lumineux reconnu par les soignants.

(a) (b)

Figure I-8 : Les types de signalisation hublots de couloir : (a) Ackermann [19]. (b) MOCAIS [20].

I.6.2 Système d'appel malade sans fil :

I.6.2.1 Téléphone sans fil compatible :

Le téléphone appel malade compatible (figure I.9) est un dispositif autonome avec un support de charge. Le soignant équipé de ce téléphone pourra recevoir non seulement la sollicitation du patient sous forme d'appel, mais aussi d'autres informations relatives à son état de santé.

(a) (b)

Figure I-9 : Type de téléphone appel malade compatible. (a) Kl-QC08 [21]. (b) KlQC06 [22].

I.6.2.2 Système d'appel malade Médaillon :

C'est est un petit appareil sans fil qui permet aux patients de lancer un appel en cas de besoins n'importe où dans l'établissement.

Les patients sont ainsi équipés d'un médaillon qui peut être porté autour du coup ou autour du poignet.

(a) **b)** **(c)**

Figure I-10 : Les type Appel malade Médaillon. (a) Instasys [23]. (b) MEYY[24]. (c) LEGRAND [25].

I.6.2.3 Système d'appel malade avec récepteurs sans fil :

Les récepteurs d'Appel malade sans fil sont des systèmes totalement autonomes et complets. Lors d'un appel d'urgence le soignant sera averti avec le récepteur par un affichage de position du patient attaché d'un son et d'une vibration.

(a) (b) (c)

Figure I-11 : Les types de récepteur d'appel malade sans fil. (a) CRMS 868 MHz [26]. (b) ASCOM a71[27]. (c) ascom 914T [28].

I.7 Le concept de notre système d'appel malade

Comme nous l'avons vu précédemment, il existe déjà sur le marché des systèmes d'appel malade. L'objectif de ce travail est donc de proposer un système reprenant certains aspects des appels malades traditionnels tout en apportant quelques améliorations. Pour ce faire, nous allons nous baser sur une carte de développement reposant sur une plateforme Arduino et intégrant un module Wifi. L'objectif visé ici est que lorsqu'un malade appuie sur un bouton, cette information est envoyée via Wifi vers les smartphones du personnel soignant. Ces derniers disposant d'une application Android, ils pourront choisir dans la mesure du possible prendre en charge ou non le malade.

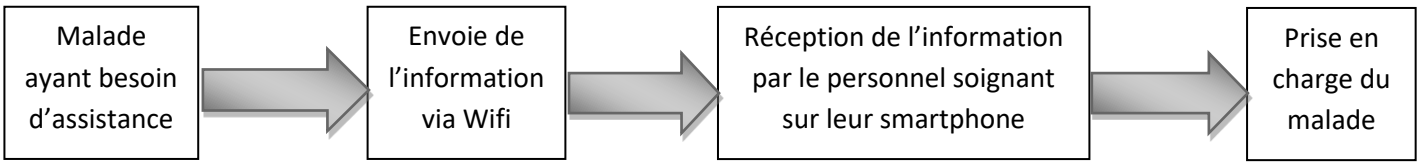


Figure I-12 :Principe de fonctionnement de notre système d'appel malade.

I.8 Conclusion

Dans ce chapitre, nous avons commencé par présenter un petit historique sur l'appel malade et les raisons qui ont poussées à son invention. Nous avons ensuite défini ce qu'était ce système avant de citer les différents types existants. Nous avons aussi vu où est-ce qu'on pouvait trouver de tels systèmes et quels sont les dispositifs déjà présents dans le commerce avec leurs particularités. A partir de là, nous avons imaginé un système d'appel malade faisant appel à une carte électronique intégrant un module Wifi et une application Android peu couteux et facile à mettre en place. Dans le chapitre qui suit, nous allons présenter les différents outils qui nous ont été nécessaires pour mener à bien ce travail.

II. CHAPITRE 2 : Outils matériels et logiciels nécessaires à la réalisation de notre appel malade

II.1 Introduction :

Après avoir établi le contexte et la problématique de notre travail dans le premier chapitre, nous allons détailler dans ce deuxième chapitre les différents composants matériels et logiciels composant notre appel malade. Nous commencerons par une présentation sur l'Arduino et le module Wifi ESP8266 avant de détailler la carte de développement NodeMCU utilisée dans notre projet. Nous présenterons par la suite l'outil MIT App Inventor utilisé pour créer notre application Android. Nous finirons par quelques notions sur l'outil Firebase de Google et comment l'utiliser.

II.2 Concept de notre système d'appel malade :

Le concept proposé de notre appel malade est basé sur quatre composants : carte Arduino, module Wifi ESP8266, application Android et l'outil base de données Firebase de Google. L'objectif ici est de créer une application Android recevant une demande d'assistance d'un patient résidant dans une structure médicale. Cette demande est générée par l'action du malade sur un bouton, elle est ensuite transférée via Wifi (NodeMCU) vers les smartphones du personnel soignant. L'infirmier ou le médecin prenant en charge le malade devra alors notifier à ses collègues que le malade a bien été pris en charge, ceci permettra d'éviter le déplacement inutile d'un autre soignant chez le malade.

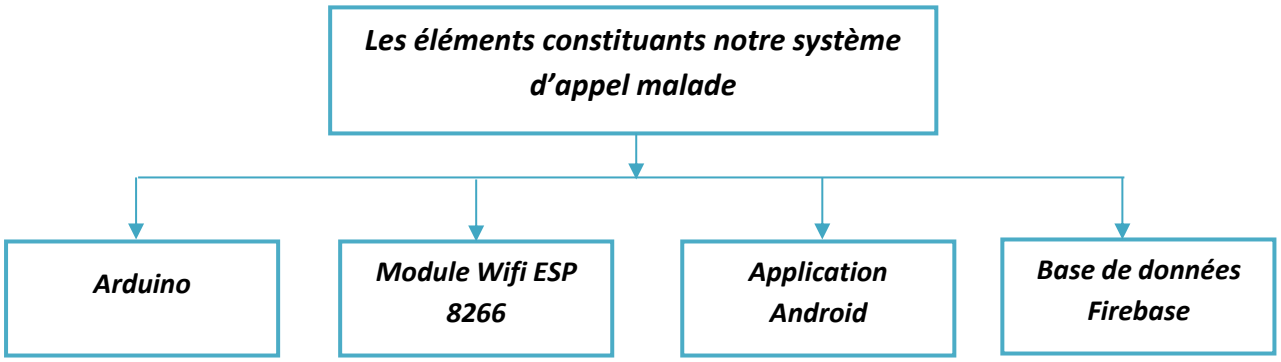


Figure II-1 : Les composants du système d'appel malade proposé.

Pour détailler les composants de notre appel malade, nous avons choisi de diviser ces derniers en deux grandes parties : partie matérielle et partie logicielle.

II.3 Partie matérielle de notre appel malade :

Pour cette partie, il s'agira de définir les composants électroniques constituant notre système d'appel malade.

II.3.1 Arduino :

Arduino est une plate-forme électronique de développement et prototypage open source. Cette plateforme intègre deux composants essentiels : carte électronique et IDE (logiciel de programmation). Il existe une grande variété de cartes Arduino, comme le montre la figure II-2 [29].

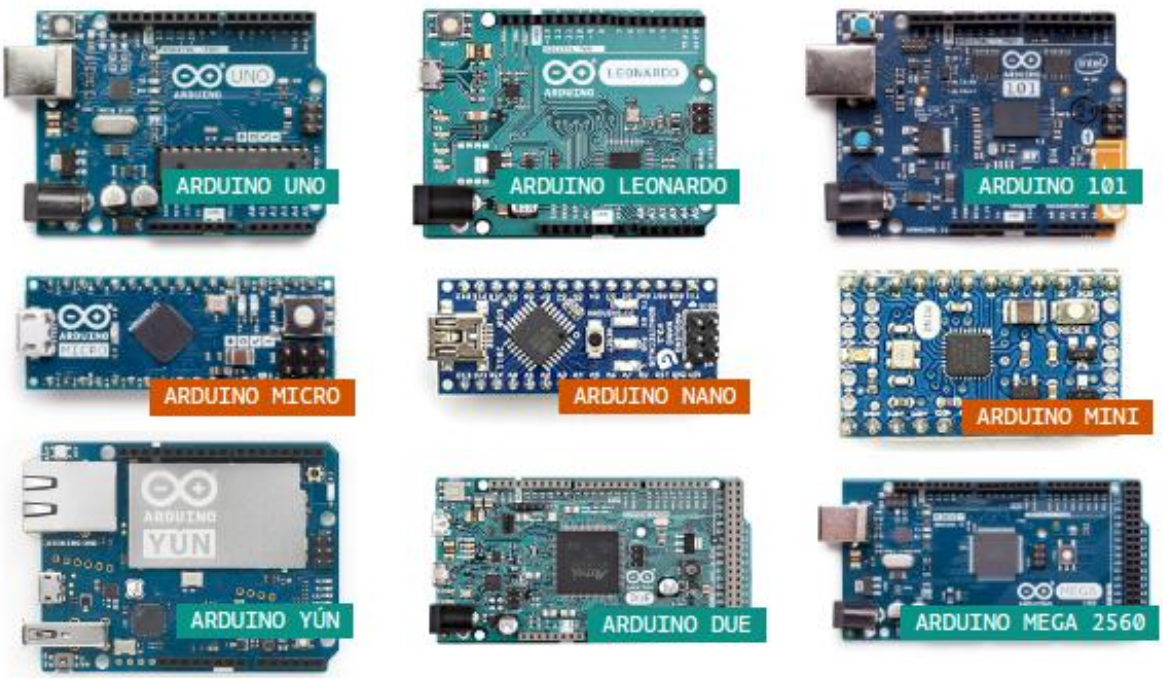


Figure II-2 : Quelques exemples de cartes Arduino.

Comme nous le verrons par la suite, notre choix se tournera vers une carte basée sur Arduino et intégrant un module Wifi.

II.3.1.1 IDE Arduino (Integrated Development Environment):

L'IDE Arduino est un logiciel multiplateforme (Windows, Linux, Mac) gratuit et open source permettant l'édition, la compilation et le téléversement de programmes sur la carte Arduino afin de la faire fonctionner [30]. L'IDE Arduino comporte plusieurs composants comme le montre la figure II-3.

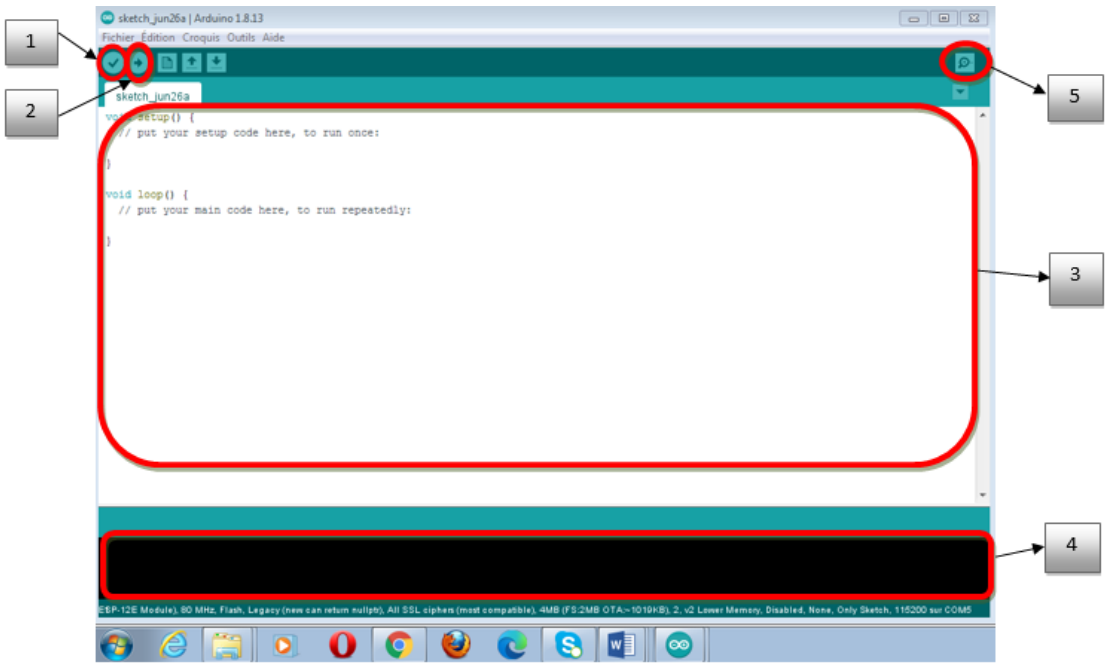


Figure II-3 : Interface de l 'IDE Arduino.

- Compilateur (1) : Pour compiler et vérifier la conformité du programme après édition.
- Télé-versement (2) : Permet de télécharger le programme sur la carte Arduino.
- Un espace d'édition : Destiné à l'écriture du programme (3).
- Une zone de message pour afficher les informations relatives à la compilation et le télé-versement du programme (4).
- Moniteur série (5) : Permet la communication entre la carte Arduino et l'IDE.

II.3.2 Module Wifi ESP8266 :

Wifi ou Wi-Fi (Wireless Fidelity) est une technologie de réseau informatique sans fil basée sur les normes IEEE 802.11 conçue afin que plusieurs équipements ou appareils puissent communiquer entre eux. Elle est de nos jours utilisée dans les réseaux locaux ainsi que pour se connecter à internet [31].

Dans notre projet, nous avons besoin d'un module Wifi pouvant être relié à une carte Arduino afin d'assurer la communication sans fil avec l'application Android, plusieurs choix s'offraient alors à nous. Soit partir sur une solution basée sur une carte Arduino et un module Wifi séparé, soit partir sur une solution combinée.

Figure II-4 : Exemples de modules Wifi ESP8266.

Nous avons opté pour une solution combinant une carte électronique reposant sur la plateforme Arduino et un module Wifi ESP8266. Cette solution s'appelle NodeMCU ESP8266.

II.3.3 NodeMCU ESP8266 :

Le NodeMCU (Node Micro Controller Unit) est une plateforme de développement matériel et logiciel open-source conçue autour d'un SoC (System On a Chip) appelé ESP8266. Ce SoC conçu et fabriqué par Expressif System reste un excellent choix pour des projets faisant appel à l'Internet des objets (IoT) de toutes sortes [32].

Figure II-5 : Carte NodeMCU ESP8266.

Le NodeMCU se compose de 30 broches chacune remplissant une fonction bien déterminée comme le montre la figure II-6 qui suit.

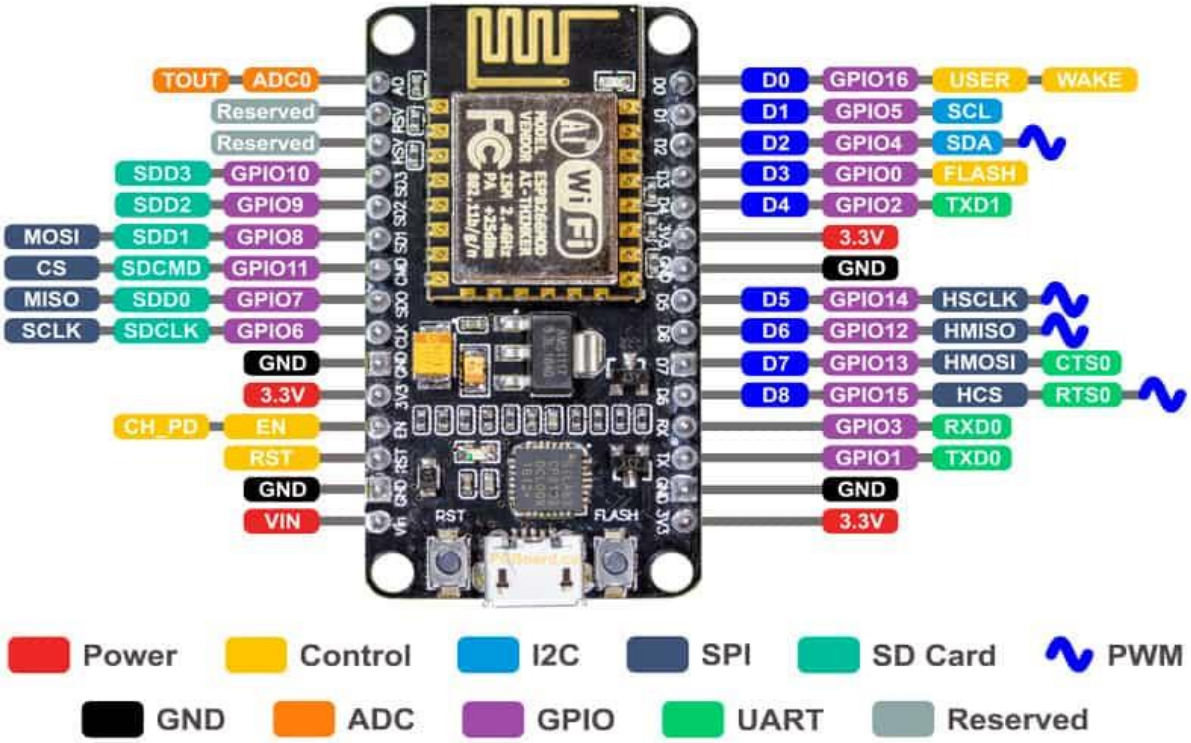


Figure II-6 : Brochage (PIN) NodeMCU esp8266 [32].

Les fonctions détaillées de chaque broche du NodeMCU sont présentées dans le tableau Figure II-7 qui suit.

GND	Masse
Broches d'alimentation	Il y a quatre alimentations. Une broche VIN et trois broches 3.3V.
Broches GPIO	La carte NodeMCU dispose de 17 broches GPIO qui peuvent être affectées à des fonctions telles que I2C, télécommande IR, lumière LED et bouton par programmation.
ADC Chanel	Le NodeMCU est intégrée à un ADC SAR de précision 10 bits. Les deux fonctions peuvent être implémentées à l'aide de l'ADC. Test de la tension d'alimentation de la broche VDD3P3 et test de la tension d'entrée de la broche TOUT.
Broches UART :	NodeMCU/ESP8266 dispose de 2 interfaces UART (UART0 et UART1) qui fournissent une communication asynchrone (RS232 et RS485) et peuvent communiquer jusqu'à 4,5 Mbps.
Broches SPI :	NodeMCU/ESP8266 dispose de deux SPI (SPI et HSPI) pour fonctionner en modes esclave ou maître
Broches SDIO :	NodeMCU/ESP8266 dispose d'une interface d'entrée/sortie numérique sécurisée (SDIO) qui est utilisée pour interfacer directement les cartes SD.
Broches PWM :	La carte dispose de 4 canaux de modulation de largeur d'impulsion (PWM). La sortie PWM peut être implémentée par programmation et utilisée pour piloter des moteurs numériques et des LED.
Broches de contrôle	Sont utilisées pour contrôler le NodeMCU/ESP8266. Ces broches comprennent la broche d'activation de la puce (EN), la broche de réinitialisation (RST) et la broche WAKE.
FR	La puce ESP8266 est activée lorsque la broche EN est à l'état haut. Lorsqu'elle est l'état bas, la puce fonctionne à une puissance minimale.
RST	la broche RST est utilisée pour réinitialiser la puce ESP8266.
WKE	La broche de réveil est utilisée pour réveiller la puce du sommeil profond

Figure II-7 : Fonctions des broches de la carte NodeMCU ESP8266.

L'une des principales caractéristiques de la carte NodeMCU est qu'elle peut être programmée en utilisant tout simplement l'IDE Arduino. L'utilisateur de ce fait n'est pas obligé d'apprendre un autre langage de programmation. Néanmoins, pour fonctionner correctement avec l'IDE Arduino, il faudra rajouter les bibliothèques ESP8266 à ce dernier.

II.3.4 Installation des bibliothèques ESP8266 dans l'IDE :

Après avoir ouvert l'IDE Arduino, il suffit d'aller dans le menu « Préférences ». Dans la partie URL pour gérer les cartes supplémentaires, nous allons introduire l'URL suivante (voir figure II-8 :

http://arduino.esp8266.com/stable/package_esp8266com_index.json

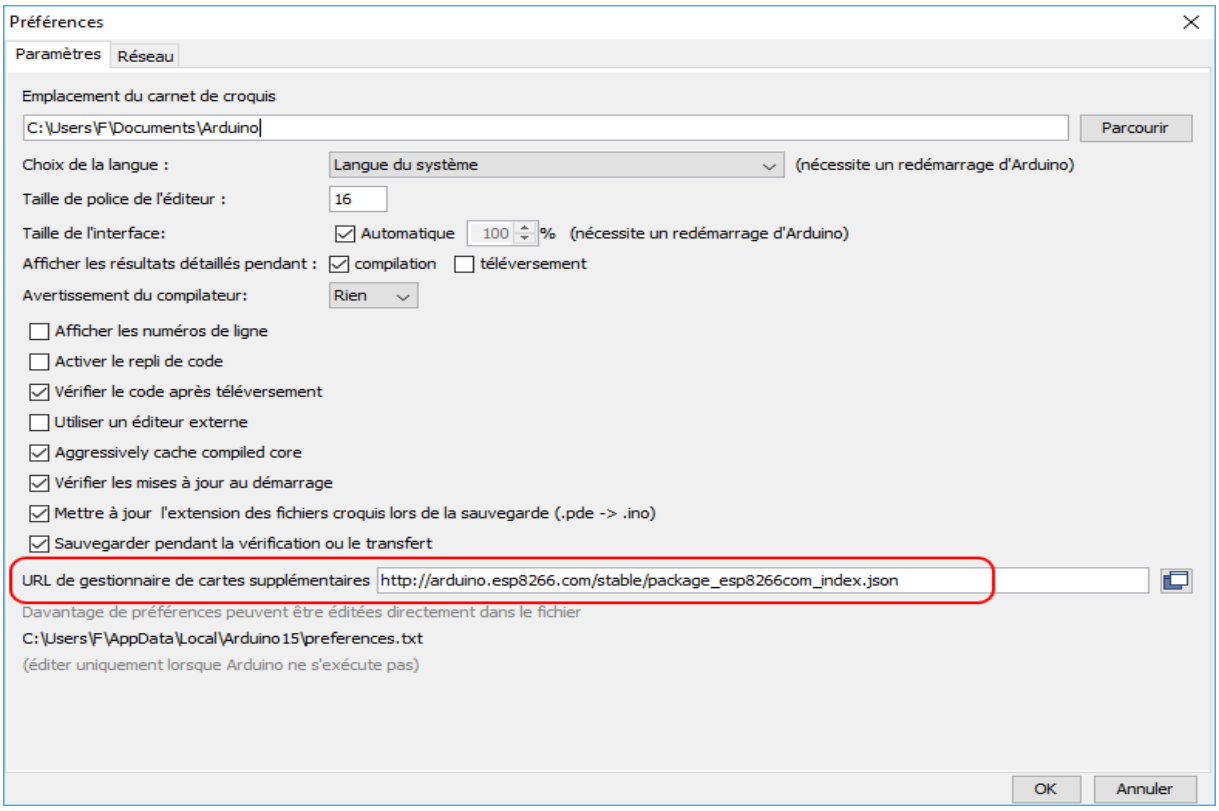
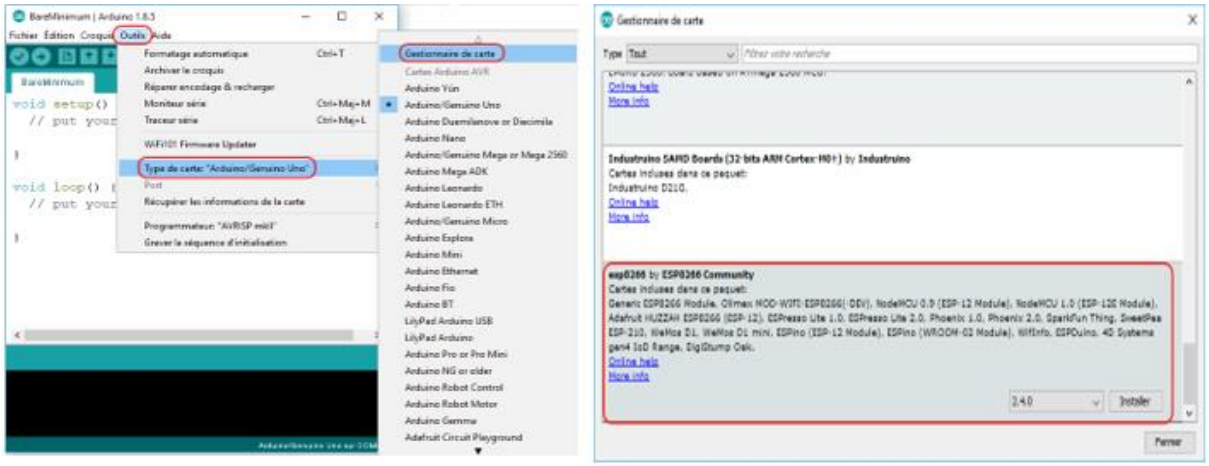


Figure II-8 : Insertion du lien d'installation de la bibliothèque ESP8266 dans l'IDE .

Il faudra ensuite aller dans le menu outil, chercher le Type de carte : Arduino/Genuine Uno et sélectionner le Gestionnaire de cartes. Une nouvelle fenêtre s'affiche, il suffira ensuite de lancer une recherche sur l'ESP8266 pour l'installer.



(A) (B)

Figure II-9 : (A) Menu type de carte de l'IDE. (B) Installation de la bibliothèque ESP8266.

La bibliothèque ESP8266 installée, il ne nous reste plus qu'à sélectionner la carte NodeMCU 1.0 (ESP-12E Module) dans le menu type de carte comme le montre la figure II-10.

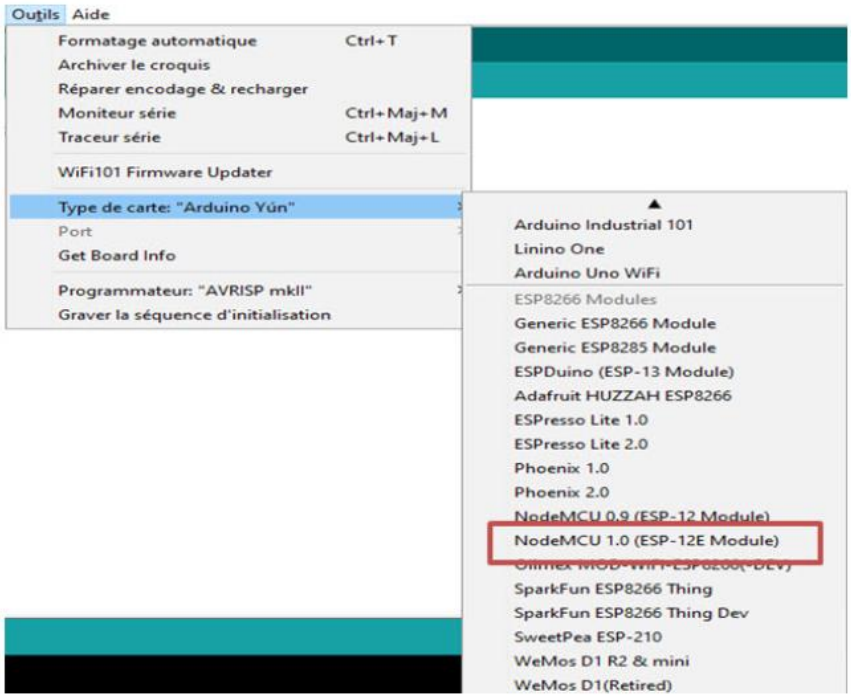


Figure II-10 : Choix de la version de la carte NodeMCU.

II.4 Afficheur OLED I2C (Organic Light Emitting Diode):

Il s'agit d'un module d'affichage monochrome de 0,96 pouce avec 128 × 64 pixels OLED I2C. Ce module d'affichage OLED peut être alimenté de 3,3 à 5 volts, ce qui le rend compatible avec les cartes prenant en charge des tensions de 3,3 volts, telles que NodeMCU ESP8266, ESP32 [35].

Figure II-11 : Afficheur OLED I2C [36].

II.4.1 Brochage OLED I2C :

Ce module d'affichage dispose d'un total de 4 broches qui sont étiquetées : VCC, GND, SCL et SDA.

Figure II-12 : Broches de l'afficheur OLED I2C.

II.4.2 Exemple de brochage d'un afficheur OLED I2C avec une carte Arduino UNO :

La figure II-13 qui suit montre un exemple de comment est branché un afficheur OLED I2C avec une carte Arduino UNO.

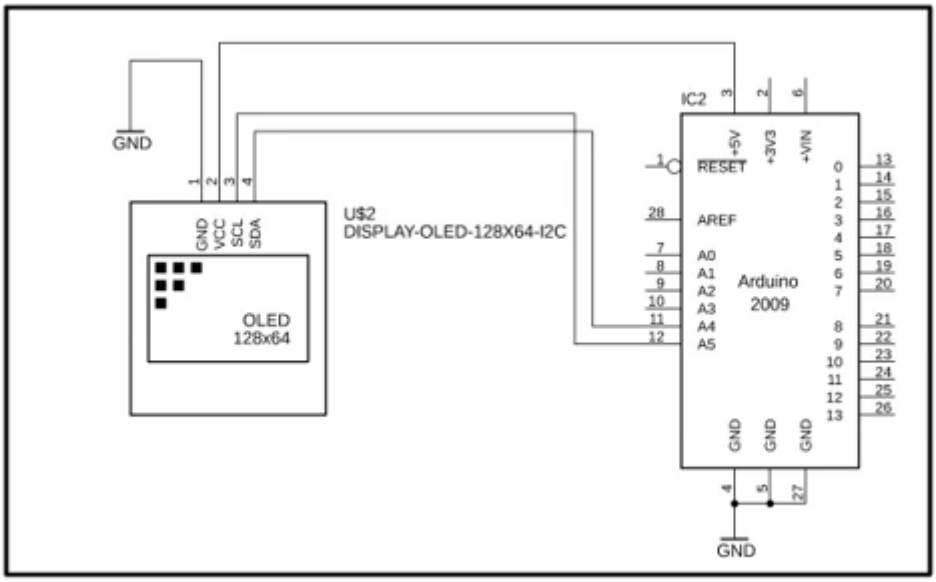


Figure II-13 : OLED I2C avec carte arduino UNO**.** [37]

Les broches VCC et GND du module d'affichage OLED I2C sont connectées respectivement au 5V et à la masse de l'Arduino. Les broches SCL et SDA sont quant à elles connectées aux broches analogiques A5 et A4 de l'Arduino.

II.5 Partie Software :

Dans cette partie, nous commencerons par présenter l'outil de création d'application Android MIT App Inventor. Nous verrons par la suite la partie base de données basée sur l'outil Firebase de Google.

II.5.1 Application Android :

Android est un système d'exploitation Open-source destiné aux appareils mobiles tels que les smartphones, tablettes, ordinateurs, objets connectés, voitures, etc... Il permet l'interaction entre l'utilisateur et son appareil par le biais d'écrans tactiles et différentes applications. Ces applications Android peuvent être créés en utilisant plusieurs outils. Nous nous intéresserons dans notre cas à l'outil MIT App Inventor.

II.5.1.1 MIT App Inventor :

MIT App Inventor est une plateforme de développement en ligne pour smartphones et tablettes destiné à créer des applications fonctionnant sous le système d'exploitation Android. C'est une application web à l'origine fournie et développée par Google et actuellement maintenue par le MIT (Massachusetts Institute of Technologie) [33].

II.5.1.2 Environnement MIT App Inventor :

Avant de pouvoir utiliser MIT App Inventor, il est nécessaire de créer un compte sur la plateforme, ceci permet entre autres de sauvegarder tous ses projets sur et d'y avoir accès à n'importe quel moment.

L'environnement de MIT App Inventor se compose de deux fenêtres principales : Designer et blocks.

- **Designer :** Cette partie permet de créer l'interface de notre application Android. Cette fenêtre Designer se compose de 4 sous fenêtres comme le montre la figure II-14.

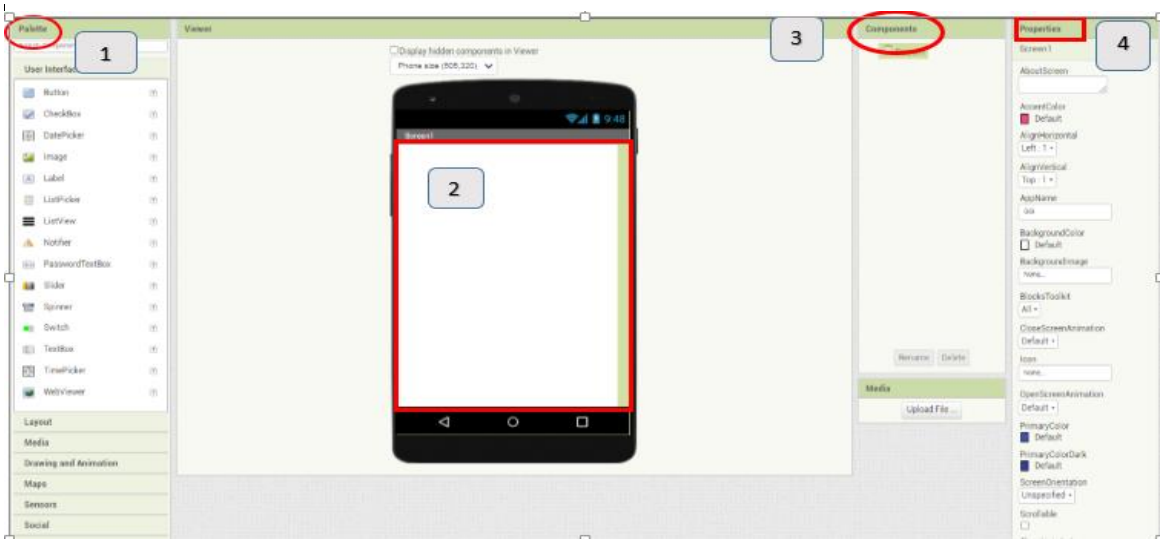


Figure II-14 : Fenêtre Designer sous MIT App Inventor.

1. **Palette** : Cette partie contient tous les éléments nécessaires à la création de l'interface de l'application Android par exemple une zone de saisie de texte, bouton, image, etc...
2. **Viewer** : Cette partie permet de montrer l'interface de l'application en temps réel.
3. **Components :** Contiennent tous les composants que nous avons ajoutés à notre application.

4. **Properties :** Les propriétés des différents éléments utilisés peuvent être paramétrés par exemple : la couleur du texte, la taille d'un bouton, texture, etc…

- **Blocks :** Cette fenêtre permet la programmation sous forme de blocs de l'application Android en fonction des composants ajoutés à son interface (figure 2-15).

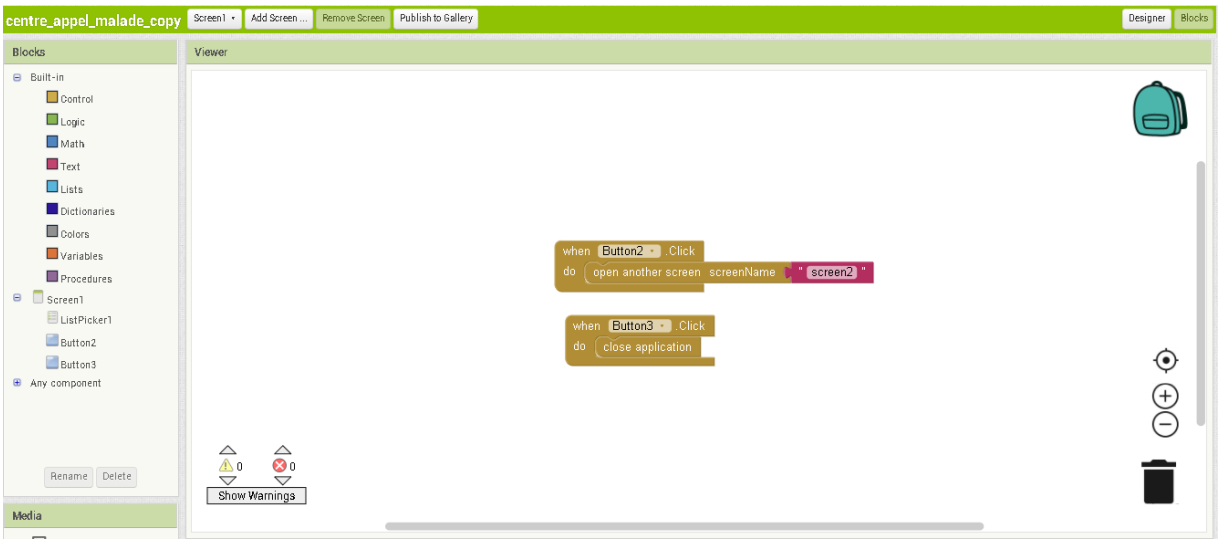


Figure II-15 : Fenêtre Blocks sous MIT App Inventor.

Une fois que l'on a terminé la programmation de l'application Android, cette dernière est automatiquement sauvegardée. On peut alors tester l'application de 3 manières différentes avant l'installation :

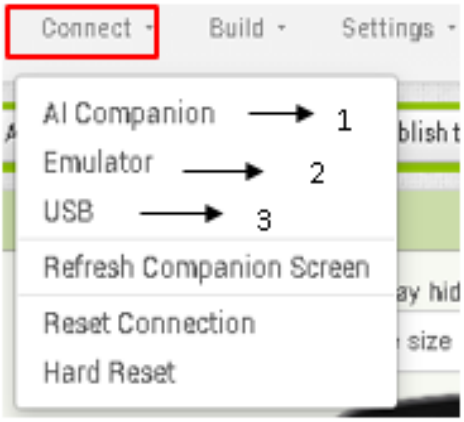


Figure II-16 : Onglet connect dans MIT App Inventor.

➢ **Al Companion :**

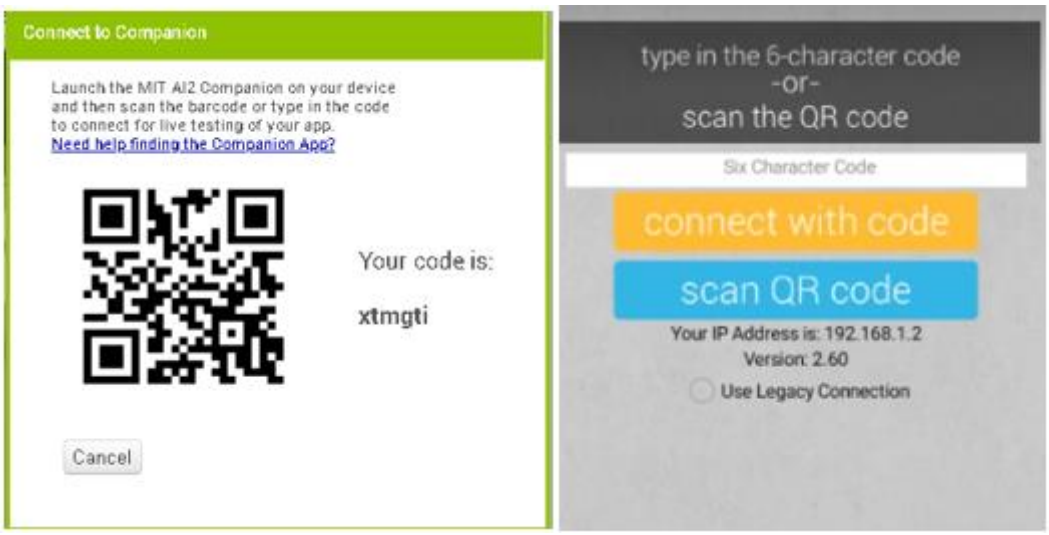


Figure II-17 : Code QR à scanner sur MIT App Inventor pour tester l'application.

➢ **USB :**

Cette option permet de tester l'application sur un smartphone connecté à un ordinateur via un câble USB en utilisant le logiciel « aiStarter ».

➢ **L'émulateur :**

Il permet de tester l'application sur un émulateur Android « aiStarter » installé sur un ordinateur.

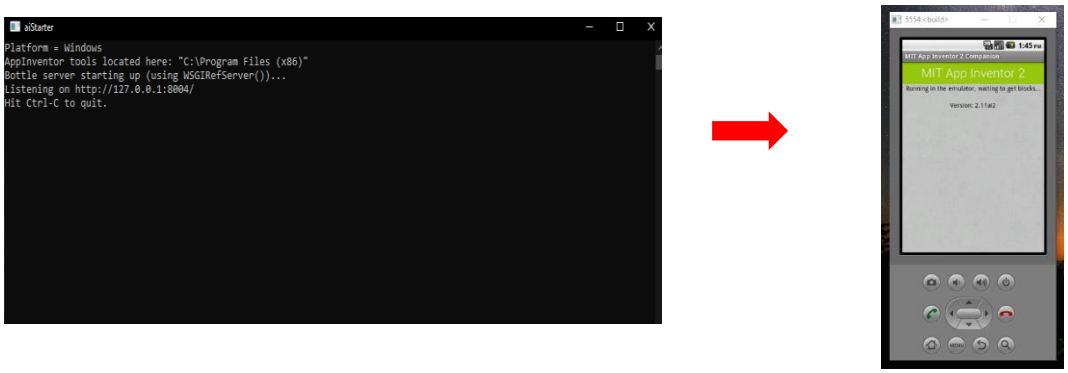


Figure II-18 : Emulateur de MIT App Inventor.

Si les tests de l'application sont fructueux, on peut alors installer l'application sur un smartphone. MIT App Inventor permet l'installation d'une application de deux manières différentes :

- En scannant un QR code à partir de l'application AiCompanion installée sur le smartphone.

- En téléchargeant le fichier « .APK » sur son ordinateur, le copier et l'exécuter sur le smartphone.

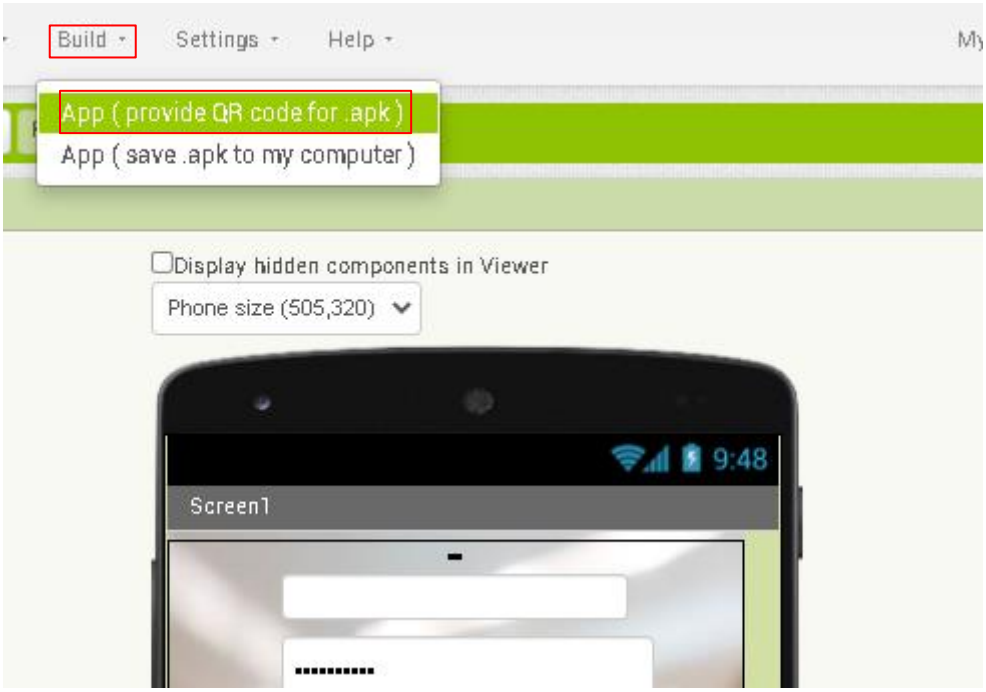


Figure II-19 : Choix du type d'installation de l'application dans le menu « Build » de MIT App Inventor.

II.5.2 Outil Firebase de Google :

Firebase est une plate-forme de développement fournissant des outils et services destinés à la création d'applications pour mobiles et pour le web. Firebase utilise une API simple pour fournir à l'application des données en temps réel et permet leur actualisation de manière automatique. Grâce à cela, la plateforme permet entre autres de gérer l'authentification des utilisateurs, de tester ses applications sur toutes les plateformes (web, iOs, Android), d'effectuer des mises à jour à distance, d'obtenir et d'analyser des rapports de crash, etc… [34].

II.5.3 Création d'un serveur Firebase :

Pour pouvoir créer un serveur, il faut accéder au site : https://firebase.google.com/. La page d'acceuil de Firebase s'affiche alors et on clique sur le bouton commencer afin d'initialiser un projet (figure II-20)

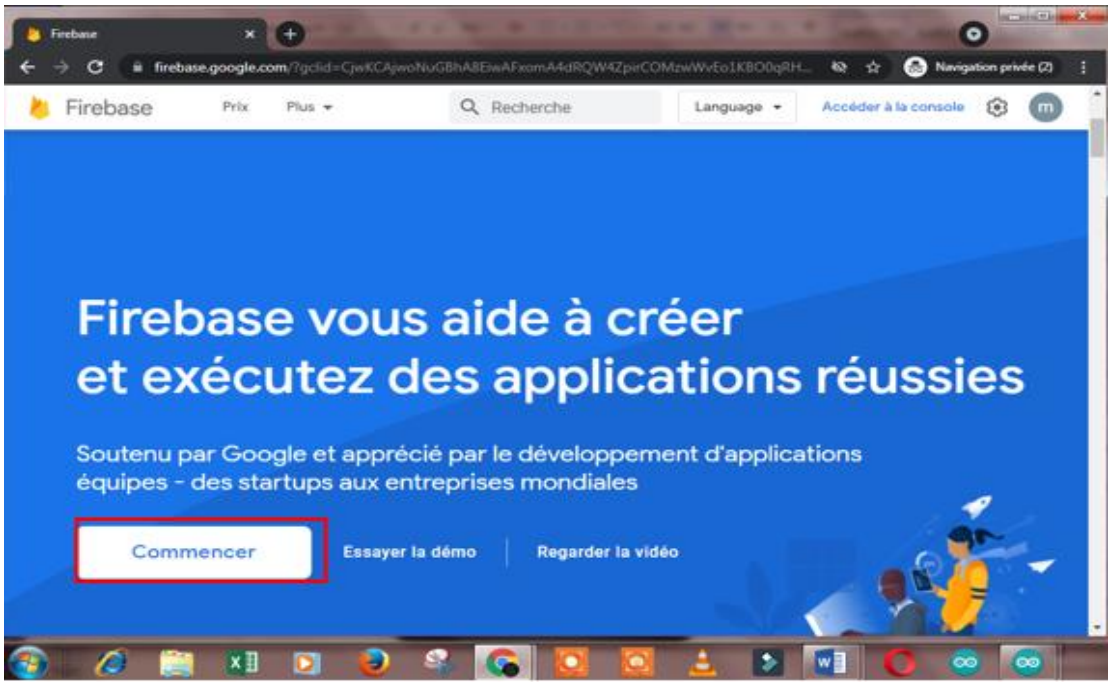


Figure II-20 : Page d'accueil de Firebase.

Il faudra après cela s'inscrire en utilisant son compte Gmail pour pouvoir utiliser ce service. Une fois connecté, une nouvelle page s'affiche (figure II-21), on peut alors éditer un projet existant ou en créant un autre.

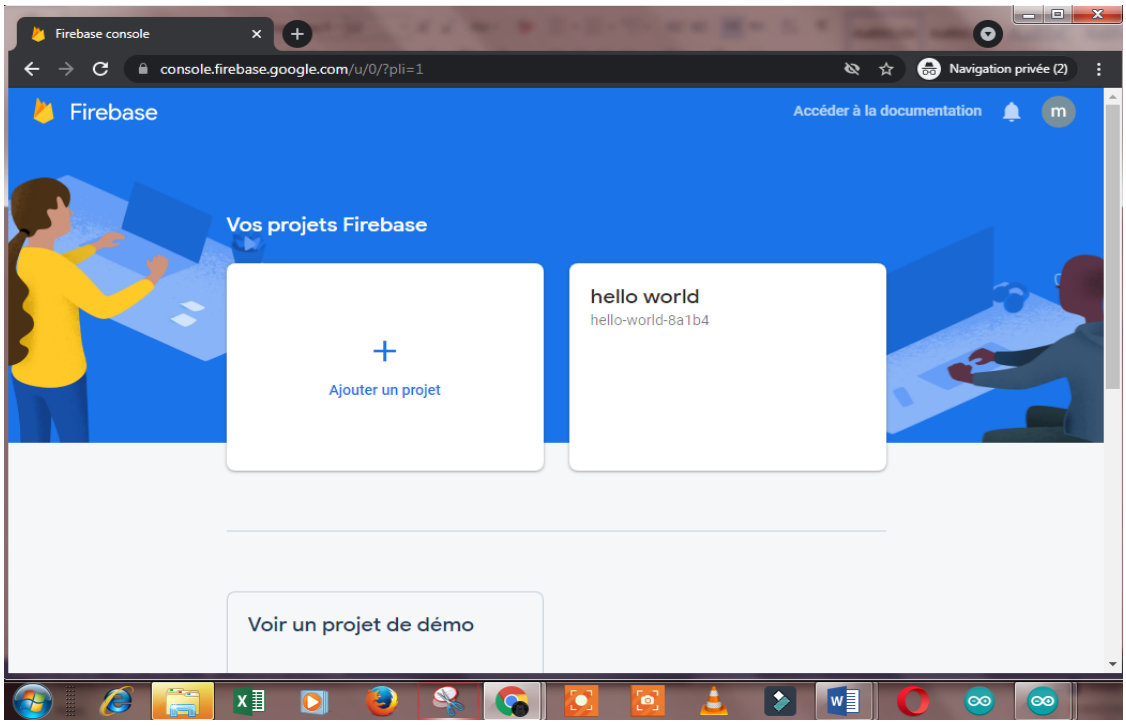


Figure II-21 : Fenêtres des projets créés sur Firebase.

L’étape suivante maintenant consiste à créer votre base de données (figure II-22).

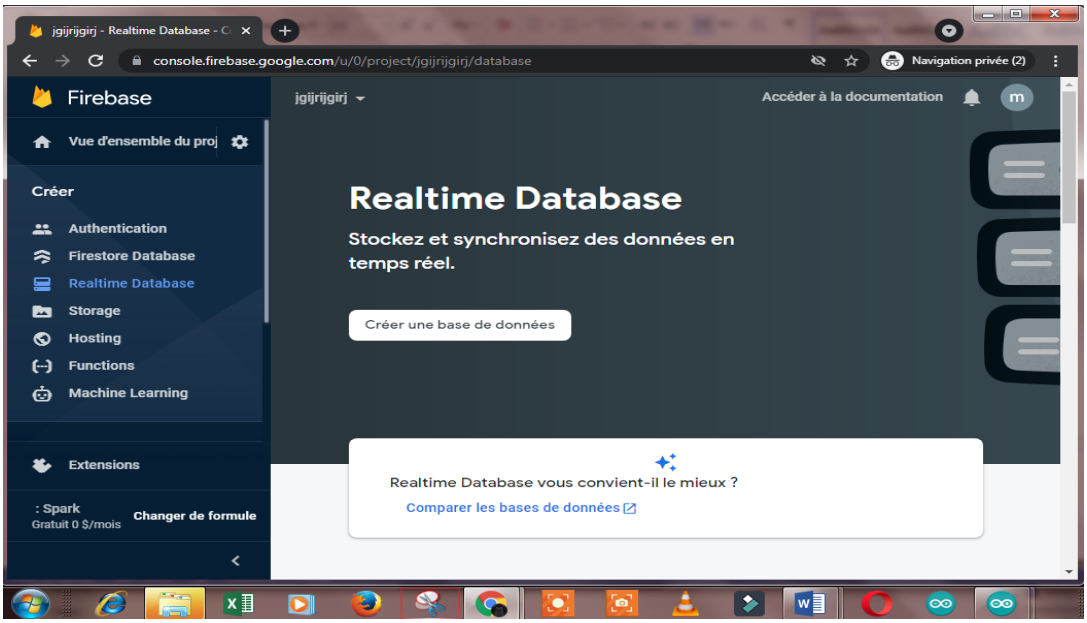


Figure II-22 : Fenêtre de création de base de données dans Firebase.

Après avoir créé la base de données, il faut récupérer les liens de connexion et d’authentification pour les introduire dans MIT App Inventor. Pour cela, il suffit d’aller dans Realtime Database et sélectionner l’onglet créer une base de données comme le montre la figure II-23.

- **Le lien de la base de données « https://hello-world-8a1b4-default-rtdb.firebaso.com/ »**

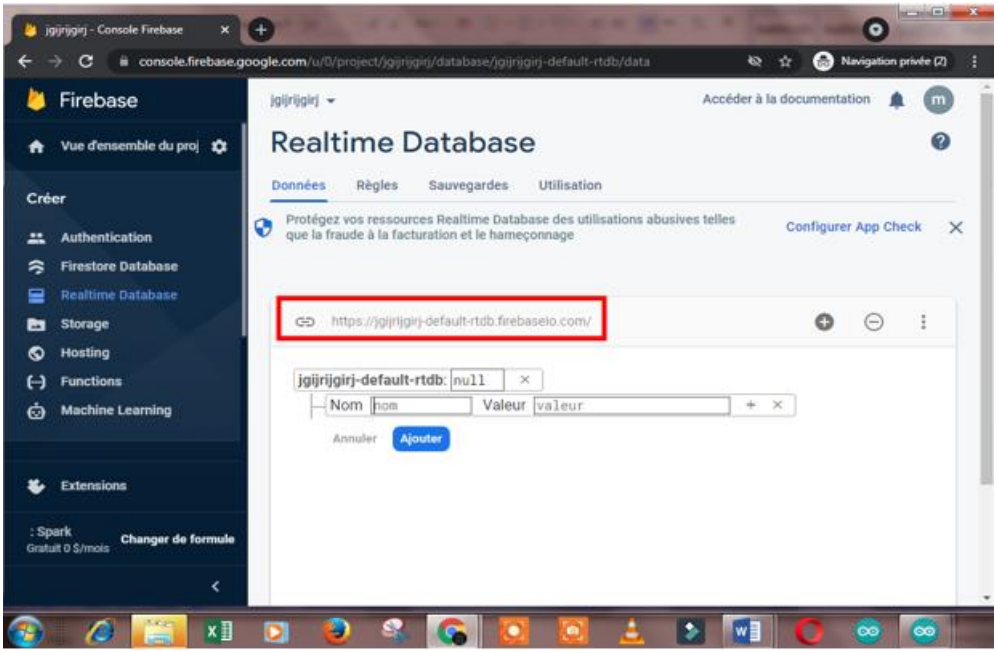


Figure II-23 : Lien de la base de donnée.

- **Le mot de de passe de la base de données :**

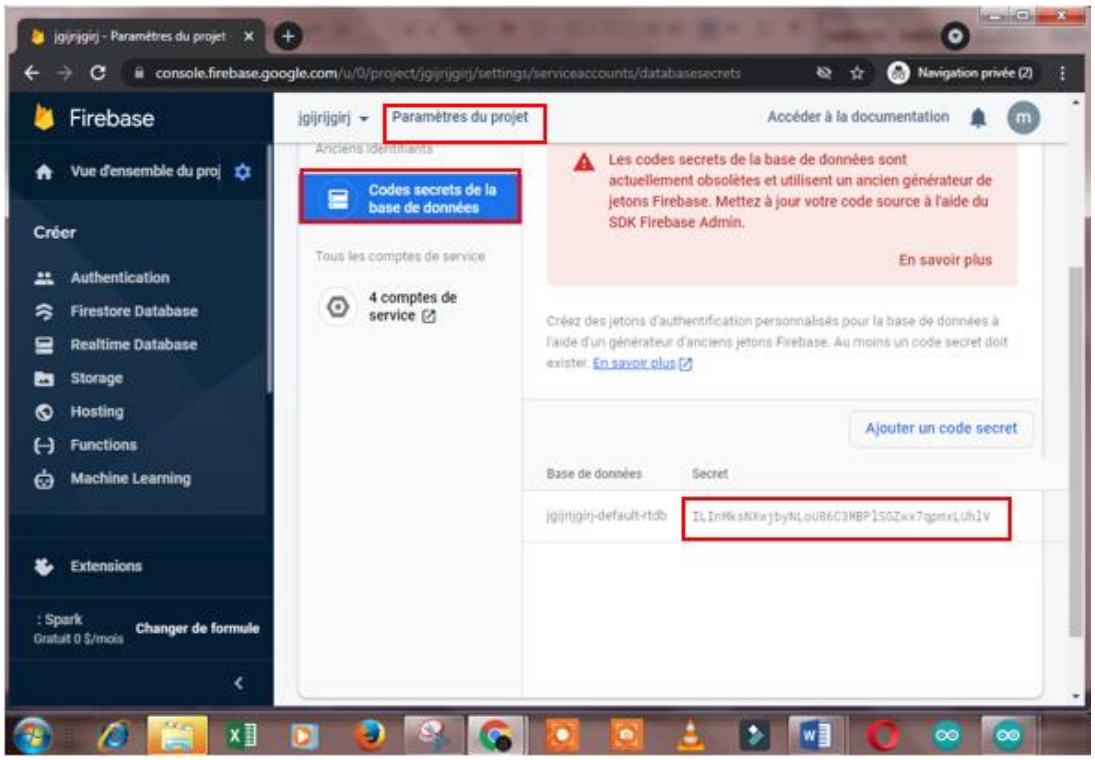


Figure II-24 : Mot de passe de la base de données.

II.6 Conclusion :

Dans ces chapitres, nous nous sommes focalisés sur la description des différents éléments et outils nécessaires à la réalisation de notre appel malade. Ainsi, nous avons divisé ce chapitre en deux parties : partie matérielle et partie logicielle. Dans la partie matérielle, nous avons présenté l'Arduino dans son ensemble avant de choisir la carte NodeMCU intégrant un module Wifi. Dans la partie logicielle, nous avons présenté l'outil de développement d'application Android MIT App Inventor ainsi que le service Firebase de Google.

Nous allons dans le chapitre suivant entamer la réalisation de notre appel malade en utilisant les outils présentés dans ce chapitre.

III. CHAPITRE 3 : La conception de système d’appel malade

III.1 Introduction :

Dans ce dernier chapitre, nous détaillerons le concept proposé de notre système d'appel malade. Nous présenterons le circuit électronique à base de carte NodeMCU utilisée avant de détailler l'application Android fonctionnant avec ce dernier. Nous finirons par quelques tests réalisés montrant le bon fonctionnement de notre système d'appel malade.

III.2 Description de notre concept d'appel malade :

La figure III-1 montre le schéma bloc de notre système d'appel malade. Le principe de fonctionnement de ce dernier est basé sur l'échange d'informations entre la carte NodeMCU, base de données FireBase et l'application Android. Ainsi, lorsqu'un malade ayant besoin d'assistance actionne un bouton Branché sur la carte NodeMCU, cette dernière transmet l'information au serveur Firebase, la valeur initialement enregistrée pour ce bouton change alors d'état. Ce changement d'état est envoyé directement à l'application Android, ce qui déclenche une alarme avertissant le personnel médical qu'un patient a besoin d'assistance. Lorsqu'un membre du personnel soignant souhaite prendre en charge le malade, il devra alors désactiver l'alarme via l'application Android.

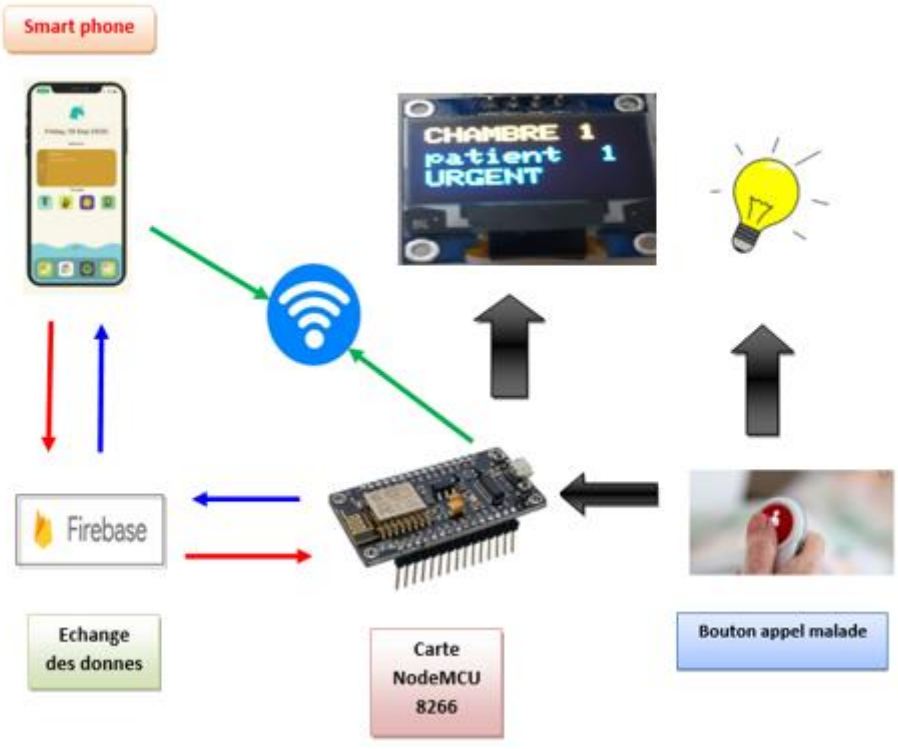


Figure III-1: Schéma bloc de notre système d'appel malade.

Parallèlement à l'application Android, un afficheur OLED est installé au niveau de la réception du centre hospitalier. Ceci permet de rajouter une sécurité en cas de problème de réseau ou bien en cas de retard induit pour la prise en charge du malade.

Nous allons dans ce qui suit montrer ce circuit réalisé ainsi que l'application Android et la base de données Firebase créée.

III.3 Le circuit électronique de notre appel malade :

Comme nous l'avons vu dans le chapitre précédent, la partie électronique de notre système d'appel malade se compose principalement d'une carte NodeMCU ESP8266 intégrant un module Wifi avec laquelle on vient brancher un bouton d'appel malade et une LED (pour chaque chambre ou patient). Le bouton va servir à alerter le personnel soignant qu'un malade a besoin d'assistance. Ainsi, dans le cas de la figure III-2, on peut voir trois boutons, donc nous sommes ici en présence d'un exemple de trois malades. L'afficheur OLED I2C va servir ici à afficher un message d'alerte au niveau de la réception du centre hospitalier. Ceci permettra de palier à d'éventuels problèmes de réseau et de s'assurer qu'il n'y a pas de retard dans la prise en charge du patient. Enfin, la LED placée devant chaque chambre de patient servira de signalisation lumineuse.

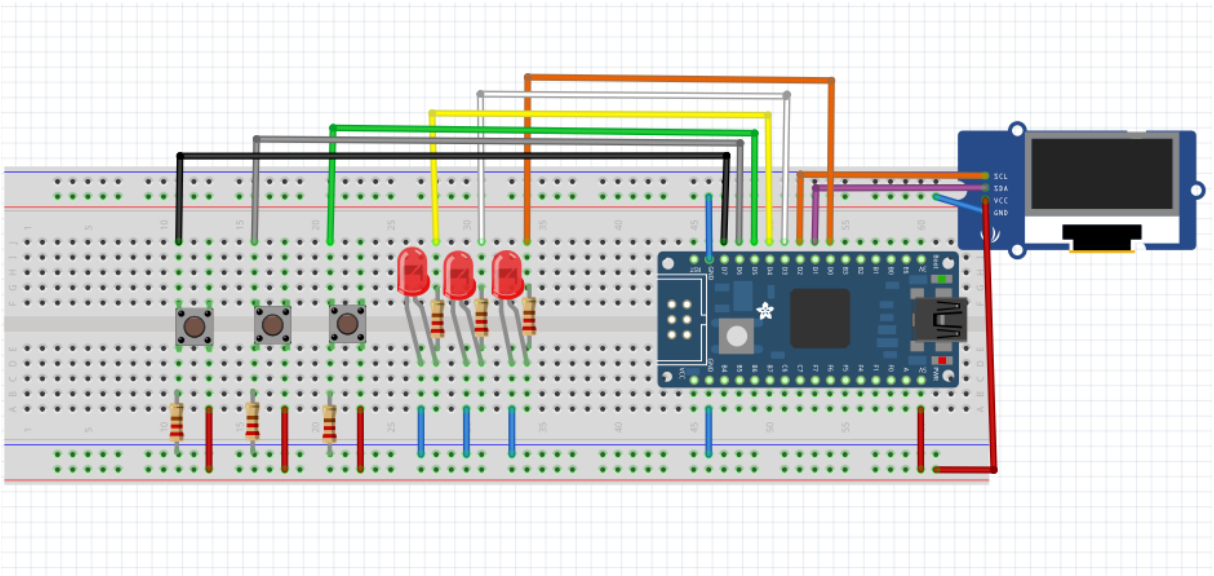


Figure III-2 : Circuit proposé pour notre système d'appel malade.

III.3.1 Programmation de la carte NodeMCU :

Afin de faire fonctionner le circuit correctement, il est nécessaire d'injecter un programme par le biais de l'IDE Arduino à la carte NodeMCU. Le programme se présente sur la figure III-3. Par soucis de clarté, nous avons pris un exemple pour un seul malade.

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <ESP8266WiFi.h>
#include "FirebaseESP8266.h"
```

Partie 1

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define FIREBASE_HOST "hospital-da611-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "WQqVE6NEj2GLoDGlt6Y4tkiQvQMDZxxxvTfk0JDi"
#define WIFI_SSID "NOM DE WIFI"
#define WIFI_PASSWORD "MOT DE PASSE"
```

Partie 2

```
FirebaseData firebaseData;
FirebaseJson json;
Adafruit_SSD1306 display (SCREEN_WIDTH , SCREEN_HEIGHT);
```

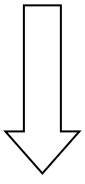
Partie 3

```
void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  Serial.begin(9600);
  pinMode(D3, INPUT); pinMode(D6, OUTPUT);
```

Partie 4

```
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  { Serial.print(".");
    delay(300);}
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);
}
```

Partie 5



```
void loop() {

 if ((digitalRead(D3)) == 0)
  { Serial.println("WORK ");
    Firebase.setFloat(firebaseData, "/hospital/btn1", digitalRead(D3)
    digitalWrite(D6, HIGH);
    display.clearDisplay();
    display.setTextSize(2);
    display.setTextColor( SSD1306_WHITE);
    display.setCursor(0, 00);
    display.print("CHAMBRE 1    ");
    display.startscrollleft(0x02 , 0x07);
    display.setCursor (0, 20);
    display.print("patient  1 ");
    display.setCursor (0, 40);
    display.print("besoin de l'aide");
    display.display();
```

Partie 6

```
    delay(9000);
    digitalWrite(D6, LOW);
    display.clearDisplay();
    display.setTextSize(2);
    display.setTextColor( SSD1306_WHITE);
    display.setCursor(0, 0);
    display.print(" WELCOM :)");
    display.setCursor (0, 20);
    display.print("To");
    display.setCursor (0, 40);
    display.print(" HOSPITAL");
    display.display();
  }
}
```

Partie 7

Figure III-3 : bloc de programmation .

III.3.1.1 Détails du programme :

Nous avons divisé notre programme en plusieurs parties comme le montre la figure III-3 précédente. Nous allons maintenant détailler le rôle de chaque partie.

- **Partie 1 :** Cette partie concerne la déclaration des bibliothèques utilisées dans notre programme.
- **Partie 2 :** Nous avons commencé par définir la taille de l'écran OLED. Nous avons ensuite défini le lien de notre base de données Firebase ainsi que le code d'authentification pour pouvoir y accéder. Nous avons terminé cette partie en définissant le réseau Wifi du centre hospitalier en question avec son nom et son mot de passe.
- **Partie 3 :** Permet de synchroniser et stocker en temps réel les données sur notre serveur Firebase. Aussi, nous l'avons défini

- **Partie 4 :** On commence l'affichage sur l'écran OLED. Après cela, on déclare les pins. Le « D3 » est notre bouton d'appel malade, le « D6 » quant à lui désigne la LED à l'entrée de la chambre.
- **Partie 5 :** Démarre la connexion de la carte NodeMCU au réseau Wifi et se connecte après cela à la base de données Firebase.
- **Partie 6 :** Si le bouton branché sur l'entrée « D3 » de la carte NodeMCU est actionné, il envoie la nouvelle valeur à la base de données Firebase, la LED branchée sur la broche « D6 » s'allume. L'afficheur OLED affiche ensuite un message sur l'écran « chambre 1 patient 1 besoin d'aide ».
- **Partie 7 :** Dans la réception, un délai est rajouté permettant de laisser le temps au personnel soignant de réagir à l'appel du malade. Après cela, la LED « D6 » s'éteint et l'afficheur affiche « Welcome to Hospital ».

Maintenant que nous avons présenté le circuit ainsi que le programme permettant de faire fonctionner notre système d'appel malade. Nous allons dans ce qui suit détailler l'application Android que nous avons créé ainsi que la base de données Firebase.

III.4 Base de données Firebase et application Android

III.4.1 Base de données FireBase

La base données Firebase va nous servir dans notre cas à stocker les identifiants et mots de passe créés ainsi que l'état des boutons d'appel malades. Dans la figure III-4, nous pouvons voir le lien de notre base de données Firebase, les identifiants enregistrés dans la section « Users » et l'état des boutons dans la section « hospital ».

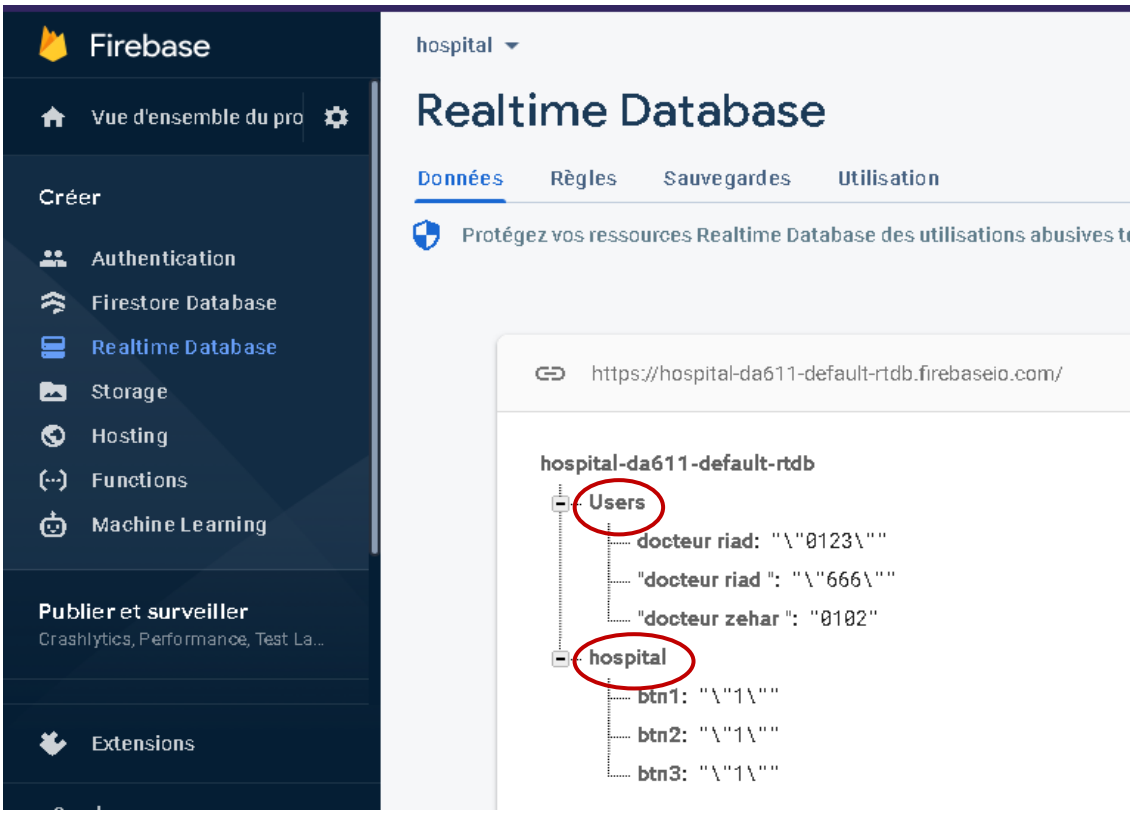


Figure III-4 : Base de données Firebase créée.

L'URL de cette base de données doit être insérée dans le programme Arduino (comme nous l'avons montré dans la section précédente) et dans l'application Android que nous allons créer. Il faudra aussi récupérer le code d'authentification (figure III.5) et l'insérer aussi dans le programme Arduino et l'application Android.

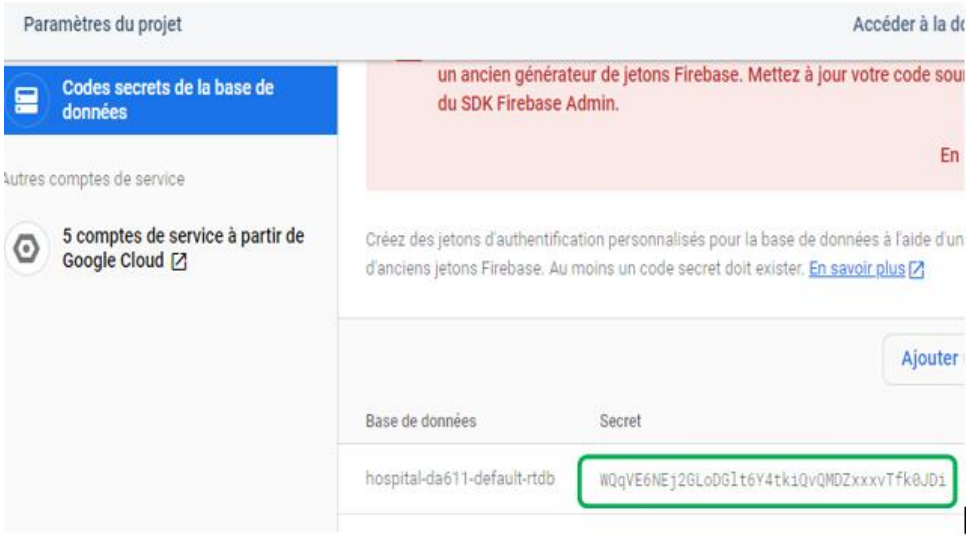


Figure III-5: Code d'authentification de notre base de données FireBase.

III.5 Application Android

Comme nous l'avons vu dans le chapitre 2, nous avons utilisé l'outil MIT App Inventor pour réaliser notre application Android. Nous avons nommé cette dernière « Smart Help ». Les étapes de création ainsi que les composants de notre application sont détaillées dans ce qui suit.

III.5.1 Création de l'application Android « Smart Help » :

L'application Android « Smart Help » a été créée en deux étapes :

- Création de l'interface.
- Programmation de l'application.

III.5.1.1 Interface de l'application Smart Help :

Notre application Smart Help se compose de deux Screen principales.

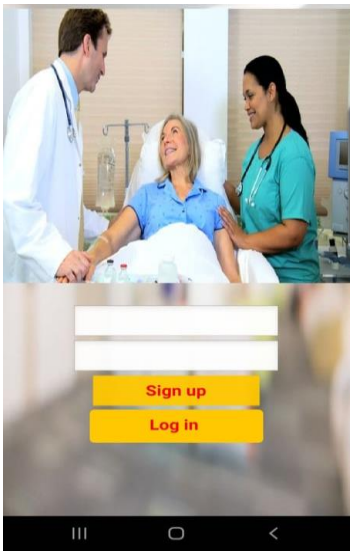


Figure III-5 : screen 1 de l'application.

Figure III-6: screen 2 de l'application.

III.5.1.2 Description des Interface de l'application Smart Help :

L'interface de notre application Smart Help se compose de deux fenêtres. Lors du premier lancement sur un smartphone, l'utilisateur doit créer son compte d'utilisateur, pour cela il doit rentrer un nom d'utilisateur et un mot de passe et cliquer sur « sign-up ». Il n'aura alors plus qu'à faire « login » après pour se connecter (figure III-7).

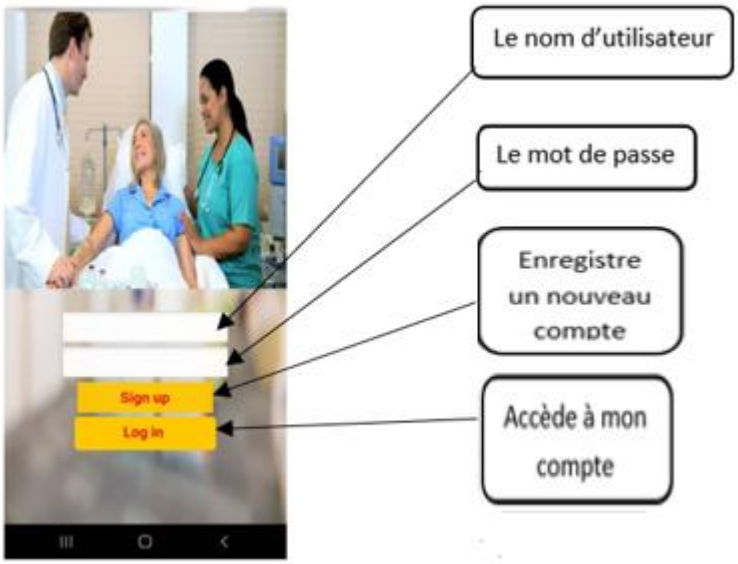


Figure III-7 : Interface d'application

Une fois enregistrées, les informations de l'utilisateur sont directement envoyées vers la base de données FireBase où ils seront stockées

Figure III-8 : les données stockées dans la base de données FireBase.

Lorsque l'utilisateur accède à l'application après le login, une deuxième fenêtre s'affiche alors montrant les chambres des patients et l'état de l'alarme du malade comme le montre la figure (III-9).

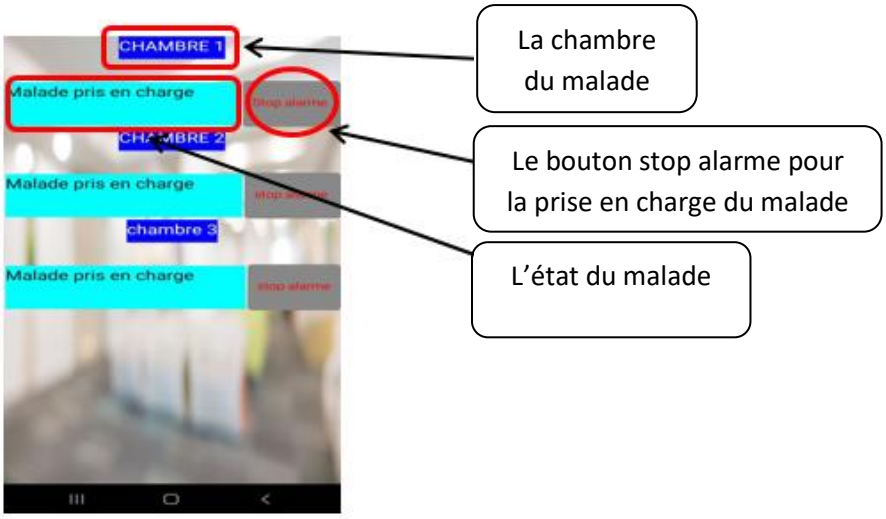


Figure III-9 :l'interface des chambres des patients

Lorsqu'une alarme est déclenchée, l'alarme de la chambre du malade change d'état comme le montre la figure III-10.

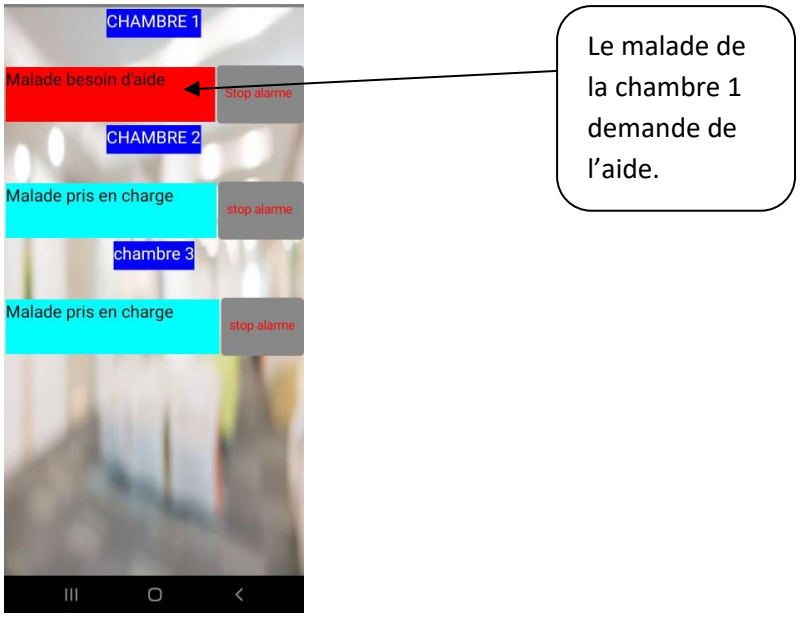


Figure III-10 : le message de demande de l'aide.

Quand l'un des membres du personnel soignant décide de prendre en charge le malade, il devra cliquer sur le bouton « stop alarme » de l'application, une notification de confirmation apparaît alors en lui demandant de confirmer ou non son action.

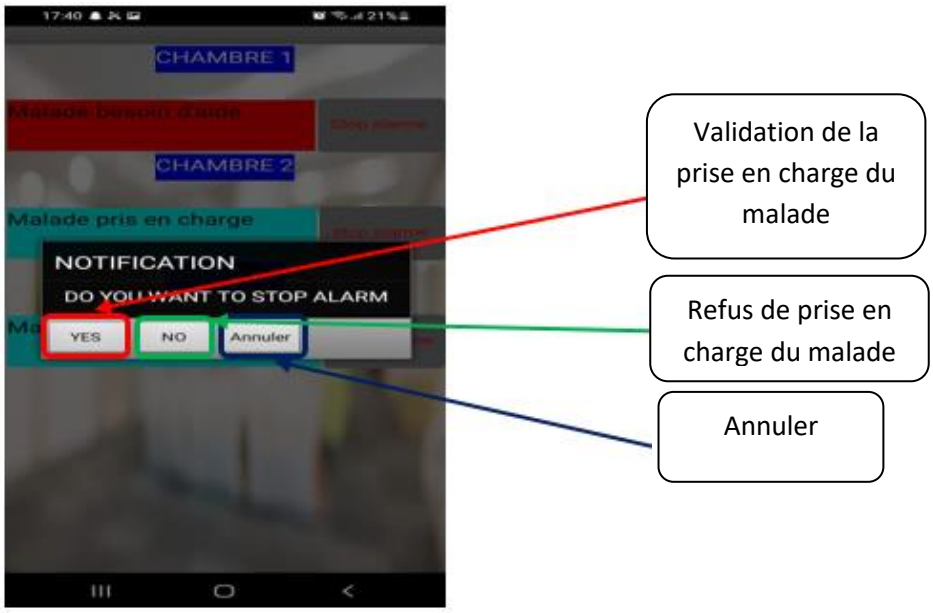


Figure III-11 : Notification de confirmation de prise en charge du malade.

Lorsque la confirmation est faite, l'application affiche que le patient a bien été pris en charge comme le montre la figure III-12.

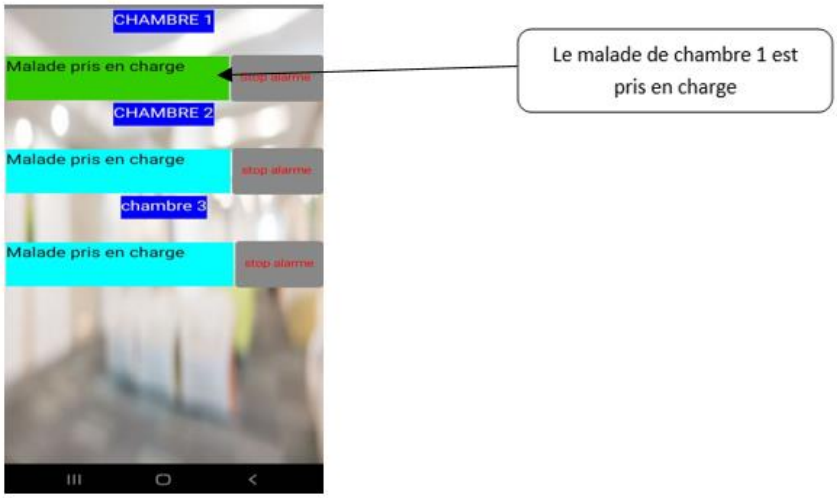
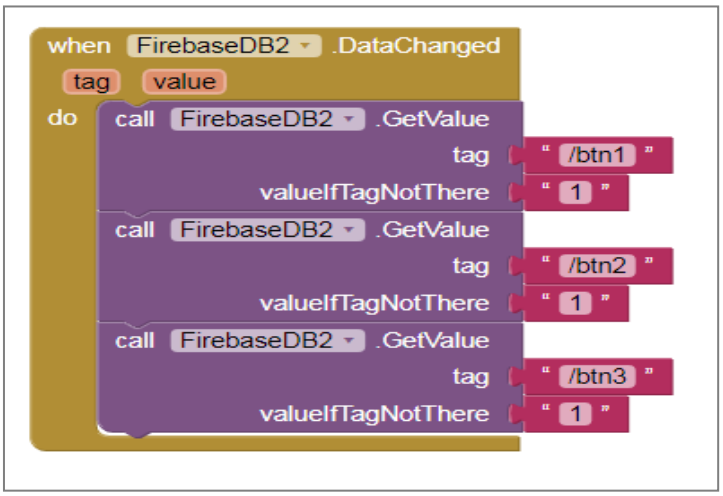


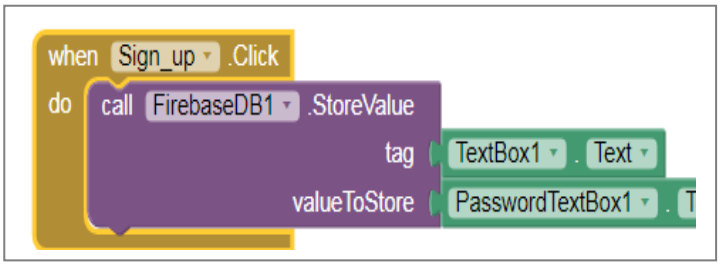
Figure III-12 : La prise en charge du malade

III.5.1.3 Programmation de l'application Smart Help :

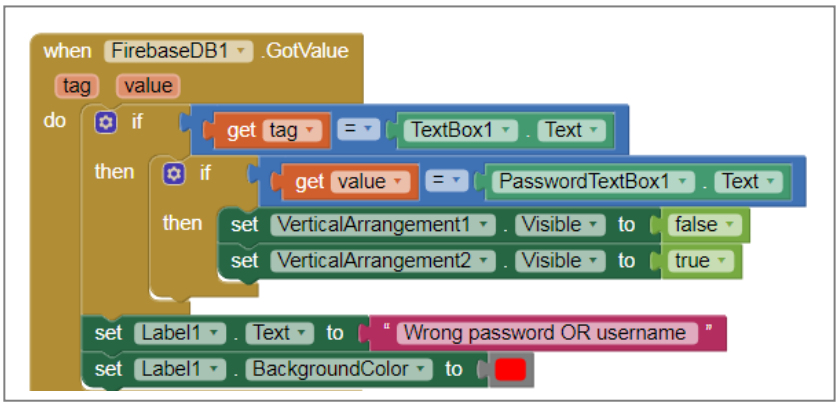
Les blocs de programmes de notre application Smart Help sont présentés dans la figure III-13.



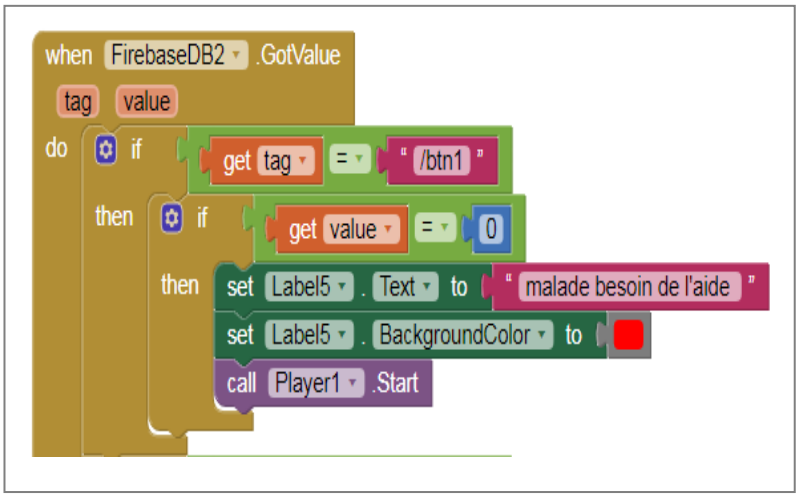
Partie 1



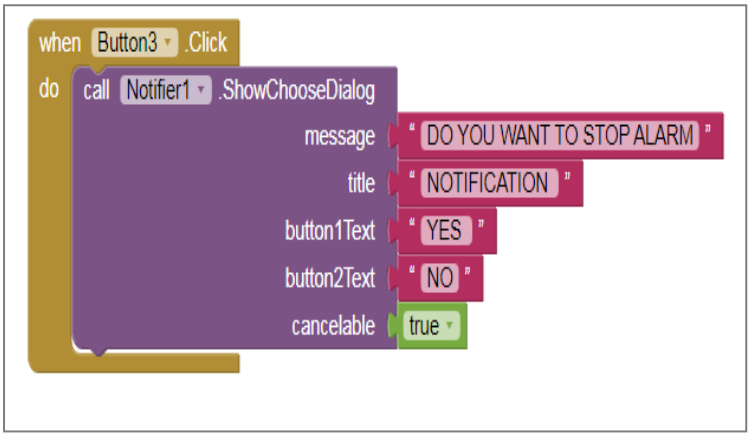
Partie 2



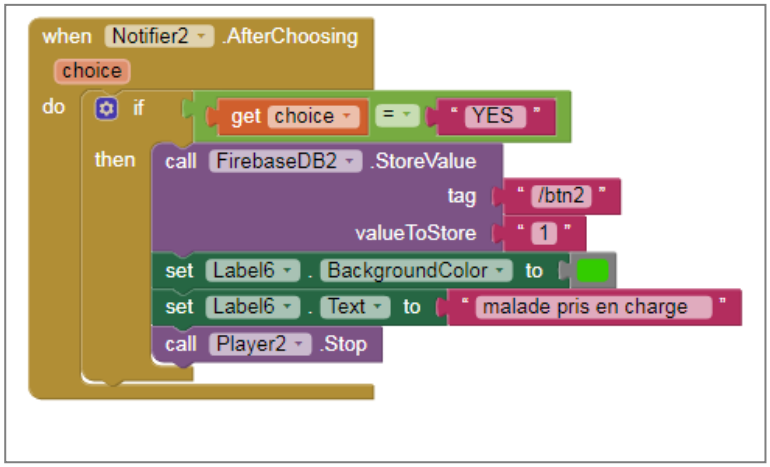
Partie 3



Partie 4



Partie 5



Partie 6

Figure III-13 Blocs de programmes de notre application Smart Help **.**

- **Partie 1 :** Dans cette partie de programme, nous allons initialiser les boutons (btn1, btn2, btn3) de chaque chambre de patient à '1' dans la base De données FireBase. Nous noterons par ailleurs que l'exemple repris ici est celui de trois chambres et trois malades.

- **Partie 2 :** Ce bloc permet d'envoyer et sauvegarder le nom d'utilisateur et le mot de passe de l'utilisateur sur la base de données FireBase.

- **Partie 3 :** Ce bloc compare la valeur récupérée pour le login avec le nom d'utilisateur et le mot de passe enregistrés sur FireBase. Si la comparaison est correcte, l'utilisateur accède à la liste des chambres. Si elle est fausse, un message ' **wrong password OR username** ' s'affiche alors.

- **Partie 4 :** Ce bloc permet de récupérer la valeur du bouton « btn1 ». Si la valeur est de 0, l'application affiche alors le message 'malade besoin de l'aide' en changeant la couleur de l'arrière-plan en rouge et déclenche l'alarme.

- **Partie 5 :** Si le bouton stop alarme (button3) est actionné, une notification de confirmation 'DO YOU WANT TO STOP ALARM' s'affiche. L'utilisateur pourra ainsi sélectionner ' YES', 'NO' ou bien 'ANNULER'.

- **Partie 6 :** Si le bouton 'YES' est appuyé, la valeur de '1' pour le « btn2 » est stockée dans la base de données FireBase. On affiche alors sur l'application « malade pris en charge » et l'alarme est stoppée.

III.6 Tests pratique de notre système d'appel malade :

Dans ce qui suit nous allons montrer pratiquement le fonctionnement de notre système d'appel malade. Ces tests seront effectués pour trois chambres et trois patients.
Nous commencerons notre montage avec le câblage des leds, des boutons et le LCD avec les pins de la carte NodeMCU.
Les leds reliées aux pins D8, D9, D10 avec des résistances représentent les chambres.
Les boutons reliés aux pins D3, D4, D5 représentent les chambres.
Le LCD relié à la carte NodeMCU par ses pins : SDA, SCL, GND, VCC aux pins : D2, D1, GND, VCC par ordre
Finalement on alimente notre circuit en alimentant la carte NodeMCU par une tension de 5V du PC.

III.6.1 Test chambre 1 :

On commence notre test par l'action sur le bouton 1 comme le montre la figure III-14.

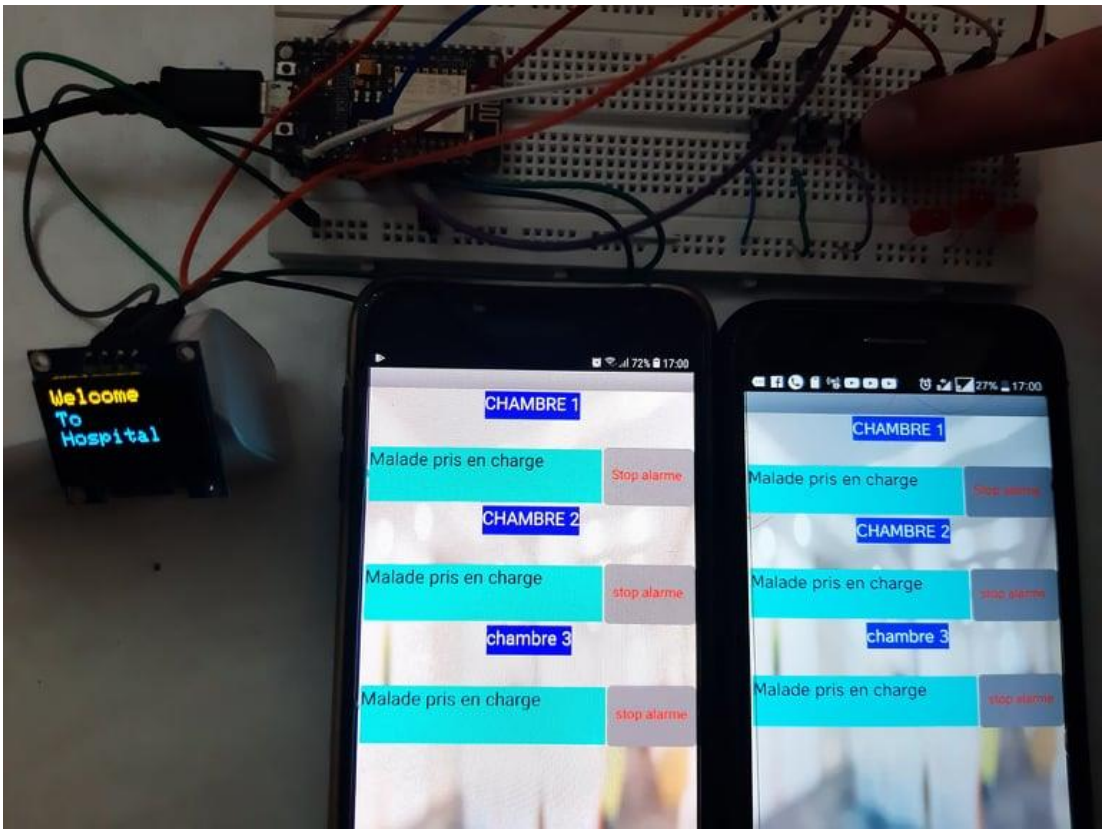


Figure III-14 : Etat initial de notre système d'appel malade.

La LED 1 commence par s'allumer, en parallèle, les smartphones de tout le personnel médical reçoivent une notification que le patient de la chambre 1 a besoin d'assistance

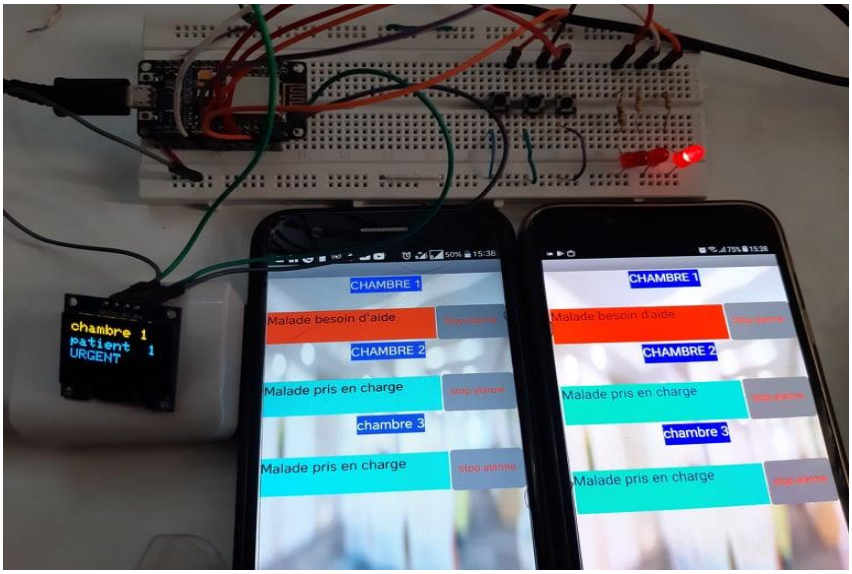
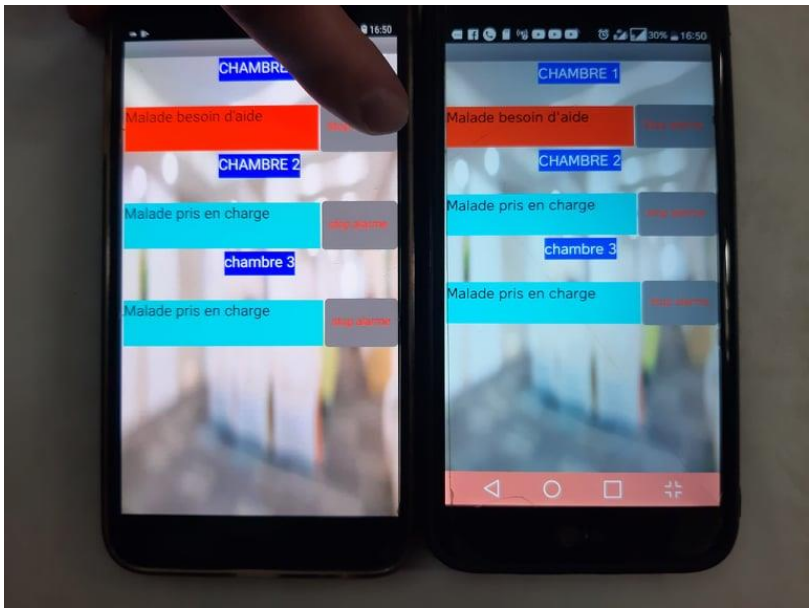


Figure III-15 : Etat du système d'appel malade après appuie sur le bouton de la chambre 1.

En même temps, l'afficheur au niveau de la réception affiche lui aussi que le malade de la chambre 1 besoin d'une assistance médicale.

Figure III-16 : Affichage du message d'assistance sur l'écran OLED pour la chambre 1.

Si l'un des membres du personnel soignant souhaite prendre en charge le patient, il désactive alors le message de la demande d'aide du patient en appuyant sur le bouton « stop alarme » correspondant à la chambre comme le montre la figure III-17.



FigureIII-17 : Stop alarme chambre 1.

Une notification de confirmation s'affiche alors (figure III-18).

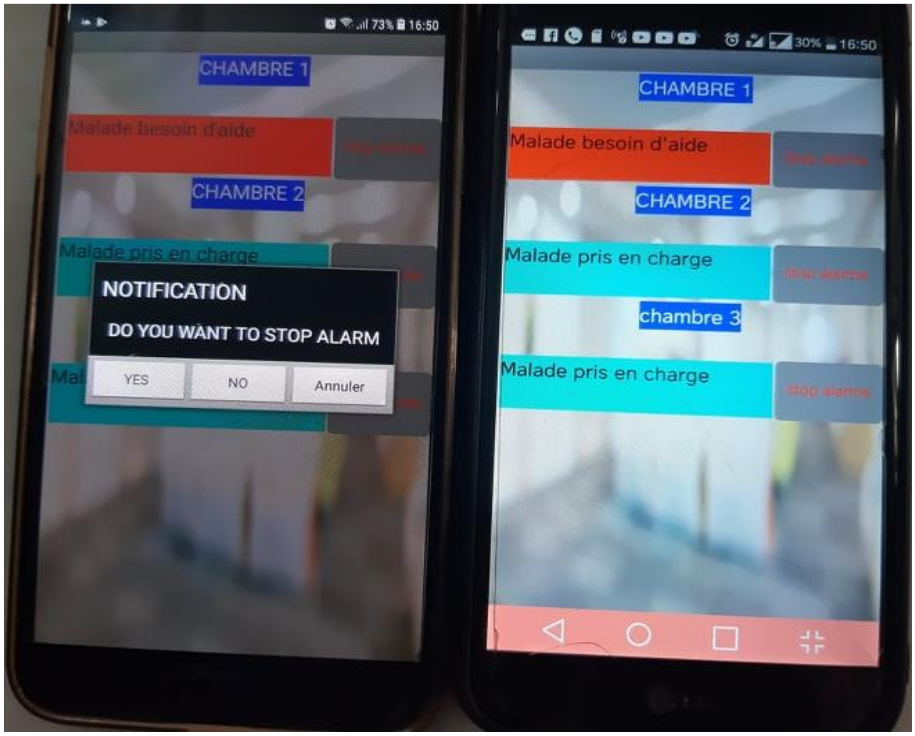


Figure III-18 : Notification de confirmation pour la chambre 1.

Si l'utilisateur appuie sur le bouton « yes » un message disant que le patient a été pris en charge s'affiche pour l'ensemble du personnel médical.

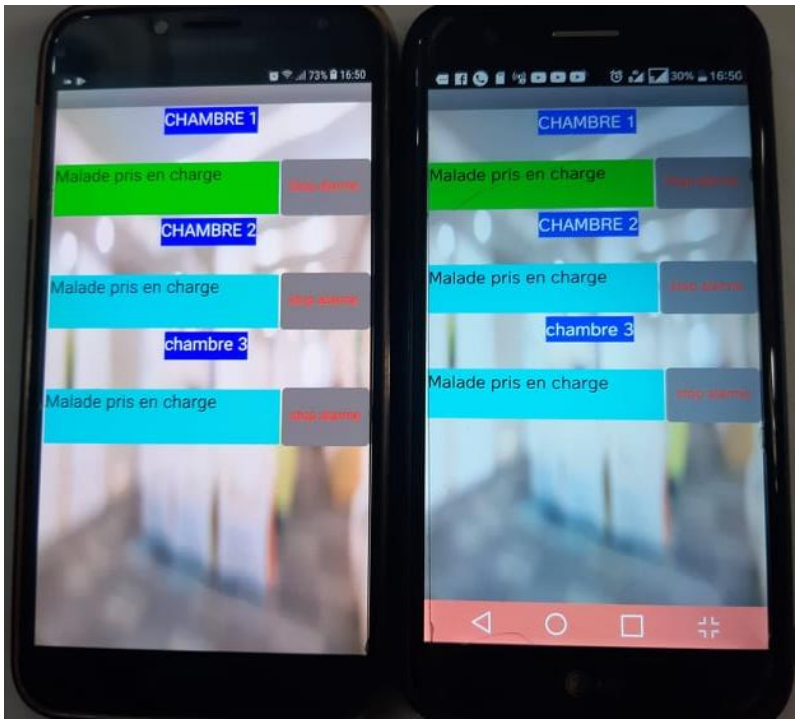


Figure III-19 : Malade de la chambre 1 pris en charge.

III.6.2 Test des autres chambres :

La même procédure est effectuée dans le cas des chambres 2 et 3 comme le montre les figures suivantes :

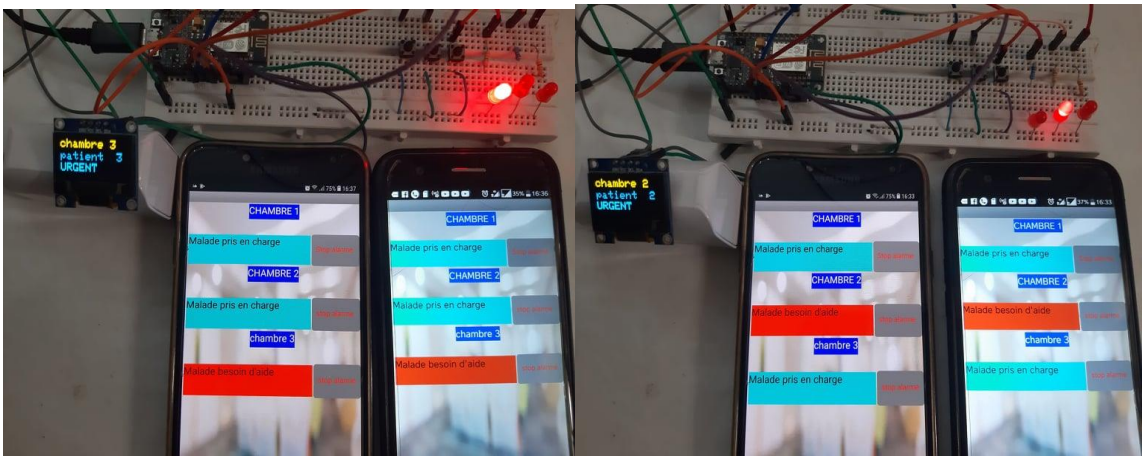


Figure III-20 : Etat du système d'appel malade après appuie sur le bouton de la chambre 2 et chambre3.

Figure III-21 : Affichage du message d'assistance sur l'écran OLED pour la chambre 2 et la Chambre 3.

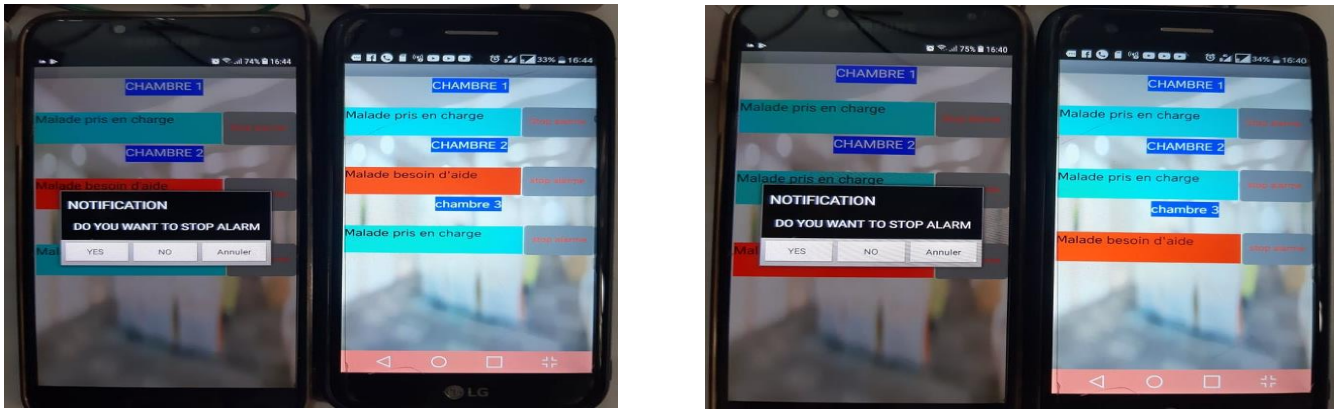


Figure III-22 : Notification de confirmation de confirmation de la prise en charge

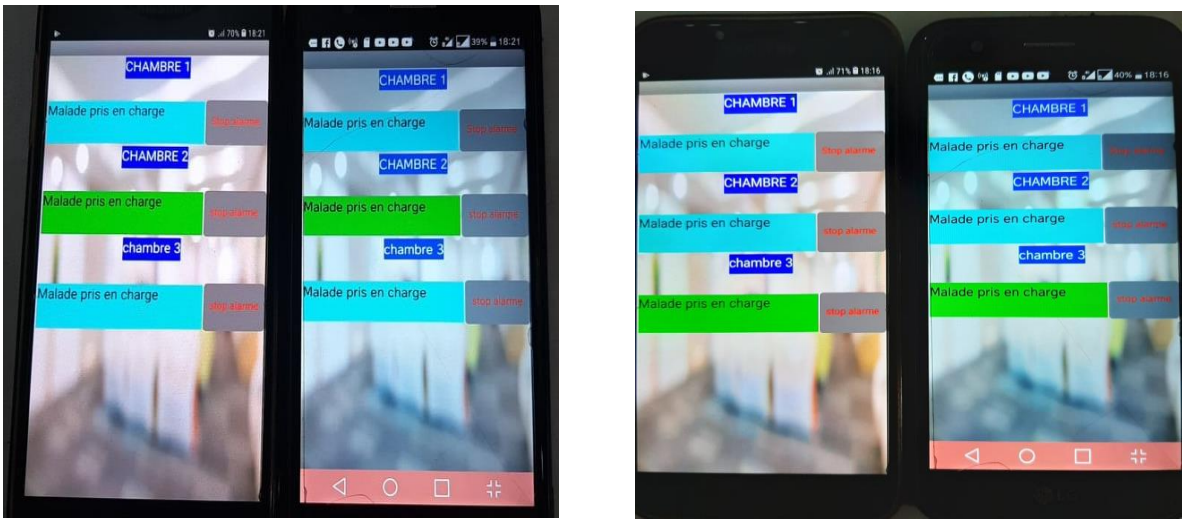


Figure III-22 : Malade de la chambre 2 et la chambre 3 pris en charge.

III.7 Concluions :

Dans ce chapitre, nous avons présenté en détails le concept de notre système d'appel malade. Nous avons commencé par donner le schéma bloc de fonctionnement de notre système d'appel malade. A partir de là, nous avons entamé la présentation du circuit électronique composant notre système avant de détailler la partie application Android « Smart Help » que nous avons créée ainsi que la base de données FireBase. Nous avons terminé ce chapitre avec des tests pratiques qui ont montré le bon fonctionnement de notre système d'appel malade.

IV. Conclusion générale :

Le personnel soignant dans son ensemble fait un travail difficile au quotidien. Afin de faciliter et fluidifier leurs activités, nous nous sommes proposés de travailler sur un système d'appel malade. L'objectif ici vise à réaliser un système d'appel malade facile à mettre en place et peu couteux. Pour mener à bien ce projet, nous nous sommes basés sur deux composantes essentielles de notre système : partie matérielle et partie logicielle. La partie matérielle contenant toute l'électronique (Carte Arduino, module Wifi, afficheur OLED). La partie logicielle quant à elle contient l'application « Smart Help » que nous avons créé ainsi que la base de données FireBase.

Pour mener à bien ce travail, nous avons divisé ce dernier en trois grandes parties :

Dans la première partie, nous avons commencé par un petit historique sur l'appel malade avant de définir ce dernier. Nous avons ensuite montré quels sont les types d'appel malade existant ? et où est-ce qu'on pouvait trouver de tels systèmes ? Nous avons fini par présenter quelques dispositifs commerciaux existants sur le marché avant de donner le concept de notre système d'appel malade.

Dans la deuxième partie, nous avons présenté les différents composants matériels et logiciels qui nous ont été nécessaires afin de mener à bien notre projet. Nous avons commencé par présenter l'Arduino de manière générale avant de décrire la carte NodeMCU intégrant un module Wifi utilisée dans notre projet. Nous avons par la suite présenté l'outils de création d'applications MIT App inventor avant de terminer avec un petit descriptif sur l'outil FireBase de Google.

Dans la troisième partie, nous avons présenté le système d'appel malade tel que nous le voyons. Nous avons commencé par donner le circuit électronique ainsi que le programme Arduino permettant de faire fonctionner notre système. Nous avons par la suite défini la base de données des noms d'utilisateurs, mots de passes et boutons dans FireBase avant de montrer comment nous avons procédé pour créer notre application et de détailler le code de cette dernière. Enfin des tests pratiques ont permis de mettre en évidence le bon fonctionnement de notre système, ce qui est très motivant en vue de continuer à développer encore plus ce projet.

A cet effet, nous proposons comme perspectives de développer encore plus l'application Android en rajoutant par exemple une fiche malade en plus d'afficher le nom du médecin prenant en charge le malade. Du côté de FireBase, cette dernière doit être remplacée par soucis de protection de données surtout dans le domaine médical.

Résumé :

Ce travail vise à réaliser un système d'appel malade simple à mettre en place et peu couteux. Pour cela nous nous basons sur une carte NodeMCU intégrant un module Wifi combinée à une application Android « Smart Help » et une base de données FireBase. Lorsqu'un malade a besoin d'aide, il devra appuyer sur bouton installé à son chevet. Cette information sera récupérée par la carte NodeMCU et envoyée vers la base de données FireBase. L'application Android « Smart Help » installée sur l'ensemble des smartphones du personnel médical reçoit par le biais de la base de donnée de FireBase qu'un patient a besoin d'aide. Le personnel soignant souhaitant ainsi prendre en charge le malade aura à désactiver la notification pour dire à ses collègues qu'il a pris en charge le patient. Un système d'affichage est rajouté au niveau de la réception du centre hospitalier afin d'alerter la réception en cas de retard de prise en charge ou défaillance réseau. Les tests que nous avons menés montrent le bon fonctionnement de notre système d'appel malade et nous encourage à développer encore plus ce projet.

ملخص :

يهدف هذا العمل إلى تحقيق نظام مكالمات مرضي سهل الإعداد وغير مكلف. لهذا نستخدم بطاقة **NodeMCU** التي تدمج وحدة **Wifi** مع تطبيق **"Android "Smart Help** وقاعدة بيانات **FireBase.** عندما يحتاج المريض إلى المساعدة ، يجب عليه الضغط على الزر الموجود بجانب سريره. سيتم استرداد هذه المعلومات بواسطة بطاقة **NodeMCU** وإرسالها إلى قاعدة بيانات **FireBase.** يتسلم تطبيق **"Android "Smart Help** المثبت على جميع الهواتف الذكية للموظفين الطبيين من خلال قاعدة بيانات **FireBase** أن المريض يحتاج إلى المساعدة. سيتعين على طاقم التمريض الراغب في رعاية المريض إلغاء تنشيط الإشعار لإخبار زملائهم بأنهم قد اعتنوا بالمريض. يتم إضافة نظام عرض على مستوى الاستقبال بمركز المستشفى لتنبيه الاستقبال في حالة حدوث تأخير في الدعم أو فشل الشبكة. تُظهر الاختبارات التي أجريناها الأداء الجيد لنظام المكالمات المرضية لدينا وتشجعنا على تطوير هذا المشروع بشكل أكبر.

الكلمات الرئيسية : Wifi، NodeMCU، FireBase، MIT App Inventor، Android.

Abstract

This work aims to achieve a sick call system that is simple to set up and inexpensive. For this we use a NodeMCU card integrating a Wifi module combined with an Android "Smart Help" application and a FireBase database. When a sick person needs help, he will have to press a button at his bedside. This information will be retrieved by the NodeMCU card and sent to the FireBase database. The "Smart Help" Android application installed on all smartphones for medical staff receives through the FireBase database that a patient needs help. Nursing staff wishing to take care of the patient will have to deactivate the notification to tell their colleagues that they have taken care of the patient. A display system has been added to the reception level of the hospital center in order to alert the reception in the event of a delay in support or network failure. The tests we have carried out show the good functioning of our sick call system and encourage us to develop this project even further.

Keywords: Wifi, NodeMCU, FireBase, MIT App Inventor, Android.

Bibliographies /Web bibliographies

[1] « History of Nurse Call », *Intercall Systems | Nurse Call Systems*, janv. 30, 2018. https://intercallsystems.com/nurse-call-systems-history-culture-context/ (consulté le juin 12, 2021).

[2] « L'histoire de l'appel malade », *Hospitalink*, mai 12, 2021. https://www.hospitalink.fr/lhistoire-de-lappel-malade/ (consulté le juin 12, 2021).

[3] F. Nightingale et M. Vicinus, *Ever Yours, Florence Nightingale: Selected Letters*. Harvard University Press, 1990.

[4] « Appel malade - Santé / Sécurité ». http://www.eshop-alliance.com/sante-securite/appel-malade.html (consulté le juin 15, 2021).

[5] « Appel malade ou appel infirmière? », *convergence.direct*. https://convergence.direct/content/109-appel-malade-ou-appel-infirmiere (consulté le mai 27, 2021).

[6] S. Factory, « Alarme appel malade sans-fil 2 médaillons d'alerte et une base de réception sonore ». https://www.prevenchute.com/alarme-appel-malade-sans-fil.htm (consulté le juin 15, 2021).

[7] « NEC APPEL MALADE FILAIRE - D3I ». http://www.d3i-fr.com/37-nec-appel-malade-filaire (consulté le juin 15, 2021).

[8] « Appel Malade, Appel Infirmière : Connex-it ». https://www.appel-malade.net/ (consulté le juin 15, 2021).

[9] « Qu'est-ce qu'un appel malade : Connex-it ». https://www.appel-malade.net/lappel-malade/ (consulté le juin 12, 2021).

[10] « Salle de surveillance des infirmières et Aides soignants - Legrand ». https://www.legrandgroup.com/fr/nos-solutions/batiment-de-sante/salle-personnel-hospitalier (consulté le juin 21, 2021).

[11] J. Z. dit, « Fer injectable en hémodialyse : nouvelles modalités d'utilisation et ce qu'elles vont changer », *Renaloo*, févr. 03, 2014. https://renaloo.com/fer-injectable-en-hemodialyse-de-nouvelles-modalites-d-utilisation-et-ce-qu-elles-vont-changer/ (consulté le juin 15, 2021).

[12] « Kine-Web.com ». https://www.kine-web.com/kine-coronavirus-ce-qu-il-faut-savoir (consulté le juin 15, 2021).

[13] « Patient En Fauteuil Roulant à L'extérieur Des Dessins Animés De L'hôpital », *Freepik*. https://fr.freepik.com/vecteurs-premium/patient-fauteuil-roulant-exterieur-dessins-animes-hopital_2504457.htm (consulté le juin 15, 2021).

[14] « Téléchargez Ensemble De Rééducation Et De Thérapie gratuitement », *Freepik*. https://fr.freepik.com/vecteurs-libre/ensemble-reeducation-therapie_9586738.htm (consulté le juin 15, 2021).

[15] « Réadaptation médicale, crunches et fauteuil roulant, aider les patients, illustration de vecteur de dessin animé sur fond blanc. Rééducation médicale, thérapie, marche avec crunchs, gymnastique en fauteuil roulant », *123RF*. https://fr.123rf.com/photo_82050673_réadaptation-médicale-crunches-et-fauteuil-roulant-aider-les-patients-illustration-de-vecteur-de-dessin-.html (consulté le juin 15, 2021).

[16] « VISOCALL IP - Système d'appel infirmière IP by Schrack Seconet AG | MedicalExpo ». https://www.medicalexpo.fr/prod/schrack-seconet-ag/product-121576-916453.html (consulté le juin 15, 2021).

[17] Techni-Contact.com, « Téléphone poste secrétariat entreprise : Devis sur Techni-Contact - Téléphone à numérotation mixte ». https://www.techni-contact.com/produits/24-9199676-telephone-poste-secretariat-entreprise.html (consulté le juin 15, 2021).

[18] « ICall 220 SIP-Touch Ecran tactil unité d'adress IP présence infirmière », *D3I*. http://www.d3i-fr.com/nec-appel-malade-ip-icall/380-icall-220-sip-touch-ecran-tactil-unite-dadress-ip-presence-infirmiere.html (consulté le juin 15, 2021).
[19] « Appel malades Ackermann », *stcom*. https://www.stcom-france.com/ackermann (consulté le juin 15, 2021).
[20] « 20160511_144735_1.pdf ». Consulté le: juin 15, 2021. [En ligne]. Disponible sur: https://www.ecataleg.be/documize/2016/5/20160511_144735_1.pdf
[21] « Imperméable à longue portée Coaster téléavertisseur pour système de radiomessagerie Restaurant Bell Kl-QC08 », *Made-in-China.com*. https://fr.made-in-china.com/co_szklvn/product_Long-Range-Waterproof-Coaster-Pager-for-Restaurant-Bell-Paging-System-Kl-QC08_uosyyreroy.html (consulté le juin 15, 2021).
[22] « Nice médicale sans fil 433.92MHz Système d'appel système de radiomessagerie Patient Pager pour aidants naturels de l'hôpital de pager Kl-QC06 », *Made-in-China.com*. https://fr.made-in-china.com/co_szklvn/product_433-92MHz-Nice-Wireless-Medical-Call-System-Caregiver-Pager-Patient-Paging-System-Pager-Hospital-Kl-QC06_uoshhhugsg.html (consulté le juin 15, 2021).
[23] « Système d'appels malades, équipement de contrôle et sécurité », *Système d'appels malades, équipement de contrôle et sécurité*. https://www.instasys.fr/systeme-appels-malades-equipement-controle.html (consulté le juin 15, 2021).
[24] « appelez l'hôpital y-a1-wr sans fil sur le bouton système système », *Fujian Huanyutong Technology Co., Ltd*. http://www.meeyicall.com/fr/hôpital-système-d-appel/système-d-appel-infirmier-sans-fil/appelez-l-hôpital-y-a1-wr.html (consulté le juin 15, 2021).
[25] « Systeme d'appel malade alarme personne agées INTERFASSE - 75 € », *ToutVendre.Fr*. https://www.toutvendre.fr/p/systeme-d-appel-malade-alarme-personne-agees-interfasse/120439862 (consulté le juin 15, 2021).
[26] « CRMS Solution d'appel infirmières, malades, contrôle de fugue, radio et/ou filaire et communication hospitalière ». https://www.crms91.com/html/produit.php?ref=CR-15 (consulté le juin 15, 2021).
[27] « DECT / BIP | Connex'it - Appel Malade ». https://www.appel-malade.net/categorie-produit/dect-bip/ (consulté le juin 15, 2021).
[28] « Appel malade sans fil », *convergence.direct*. https://convergence.direct/656-appel-malade-sans-fil (consulté le juin 15, 2021).
[29] « Arduino - Home ». https://www.arduino.cc/ (consulté le févr. 22, 2021).
[30] « Software | Arduino ». https://www.arduino.cc/en/software (consulté le juin 17, 2021).
[31] « Wi-Fi – что это такое, как работает, как пользоваться, все про вай-фай ». https://wifigid.ru/besprovodnye-tehnologii/wi-fi (consulté le juin 17, 2021).
[32] « NodeMCU ESP8266 Specifications, Overview and Setting Up ». https://www.make-it.ca/nodemcu-arduino/nodemcu-details-specifications/ (consulté le juin 05, 2021).
[33] « MIT App Inventor | Explore MIT App Inventor ». https://appinventor.mit.edu/ (consulté le mars 29, 2021).
[34] N. Laruelle, « Qu'est-ce que Google Firebase, cette technologie d'avenir dont personne ne parle ? | Joël Douillet », juin 26, 2018. https://www.joel-douillet.com/2018/06/26/google-firebase/ (consulté le juin 04, 2021).
[35] « 0.96" I2C IIC Serial OlED Module 128X64 for Arduino Raspberry PI – diymore ». https://www.diymore.cc/products/diymore-0-96-inch-i2c-iic-serial-oled-lcd-led-module-128-64-128x64-for-arduino-display-raspberry-pi-51-msp430-stm32-scr-white (consulté le juin 24, 2021).
[36] « Comment utiliser... un écran OLED I2C 128x64 0.96" ? » https://tutoduino.fr/blog-ecran-oled-i2c-128x64/ (consulté le juin 24, 2021).

[37] « Utilisation d'un Afficheur OLED 128x64 avec Arduino • AranaCorp ». https://www.aranacorp.com/fr/utilisation-dun-afficheur-oled-avec-arduino/ (consulté le juin 24, 2021).
[38] « Fritzing ». http://fritzing.org/ (consulté le juin 25, 2021).
[39] « Firebase », *Firebase*. https://firebase.google.com/?hl=fr (consulté le juin 25, 2021).
[40] « MIT App Inventor | Explore MIT App Inventor ». https://appinventor.mit.edu/ (consulté le juin 25, 2021).

Annexe

Annexe A :

Programme Arduino complet pour trois chambres et trois patients.

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#include <ESP8266WiFi.h>
#include "FirebaseESP8266.h"
#define FIREBASE_HOST "hospital-da611-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "WQqVE6NEj2GLoDGlt6Y4tkiQvQMDZxxxvTfk0JDi"
#define WIFI_SSID "wifi name"
#define WIFI_PASSWORD "wifi password"
FirebaseData firebaseData;
FirebaseJson json;
Adafruit_SSD1306 display (SCREEN_WIDTH , SCREEN_HEIGHT);
```

```
void setup() {
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
 Serial.begin(9600);
 pinMode(D3, INPUT);
  pinMode(D4, INPUT);
  pinMode(D5, INPUT);
  pinMode(D6, OUTPUT);
  pinMode(D7, OUTPUT);
  pinMode(D8, OUTPUT);

 WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
```

```
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);

}

void loop() {

  if ((digitalRead(D3)) == 0)
  { Serial.println("WORK ");
    Firebase.setFloat(firebaseData, "/hospital/btn3", digitalRead(D3));
    digitalWrite(D6, HIGH);
    display.clearDisplay();
    display.setTextSize(2);
    display.setTextColor( SSD1306_WHITE);
    display.setCursor(0,00);
    display.print("CHAMBRE 1   ");
    display.setCursor (0,20);
    display.print("patient  1 ");
    display.setCursor (0,40);
    display.print("UREGENT");
    display.display();
```

```
  delay(9000);
  digitalWrite(D6, LOW);
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor( SSD1306_WHITE);
  display.setCursor(0,0);
  display.print(" WELCOM :)");
  display.setCursor (0,20);
  display.print("To");
  display.setCursor (0,40);
  display.print(" HOSPITAL");
  display.display();
  display.startscrollleft(0x00 , 0x07);
}
 if ((digitalRead(D4)) == 0)
{ Serial.println("WORK ");
  Firebase.setFloat(firebaseData, "/hospital/btn2", digitalRead(D4));
  digitalWrite(D7, HIGH);
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor( SSD1306_WHITE);
  display.setCursor(0,00);
  display.print("CHAMBRE  2  ");
  display.setCursor (0,20);
  display.print("patient  2 ");
  display.setCursor (0,40);
  display.print("UREGENT");
  display.display();
  digitalWrite(D7, LOW);

  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor( SSD1306_WHITE);
  display.setCursor(0,00);
  display.print("CHAMBRE  2  ");
  display.setCursor (0,20);
  display.print("patient  2 ");
  display.setCursor (0,40);
  display.print("UREGENT");
  display.display();
  digitalWrite(D7, LOW);
  delay(9000);
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor( SSD1306_WHITE);
  display.setCursor(0,0);
  display.print(" WELCOM :)");
  display.setCursor (0,20);
  display.print("To");
  display.setCursor (0,40);
  display.print(" HOSPITAL");
  display.display();
  display.startscrollleft(0x00 , 0x07);
}
```

```
 if ((digitalRead(D5)) == 0)
{ Serial.println("WORK ");
  Firebase.setFloat(firebaseData, "/hospital/btn3", digitalRead(D5));
  digitalWrite(D6, HIGH);
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor( SSD1306_WHITE);
  display.setCursor(0,00);
  display.print("CHAMBRE 3   ");
  display.setCursor (0,20);
  display.print("patient  3 ");
  display.setCursor (0,40);
  display.print("UREGENT");
  display.display();
  delay(9000);
  digitalWrite(D8, LOW);
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor( SSD1306_WHITE);
  display.setCursor(0,0);
  display.print(" WELCOM :)");
  display.setCursor (0,20);
  display.print("To");
  display.setCursor (0,40);
  display.print(" HOSPITAL");
  display.display();
  display.startscrollleft(0x00 , 0x07);
}
}
```