

Rapport Application IOT

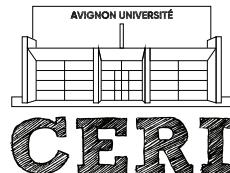
Aïda DIEDHIOU
Amine Boudra

5 janvier 2024

Master2 Informatique
Systèmes Informatiques Communicants (SICOM)
UE SYSTEMES EMBARQUES POUR LES OBJETS CONNECTES
ECUE architectures et protocoles de communication

Responsables
Marc Silanus
Philippe Gozlan

UFR
SCIENCES
TECHNOLOGIES
SANTÉ



CENTRE
D'ENSEIGNEMENT
ET DE RECHERCHE
EN INFORMATIQUE
ceri.univ-avignon.fr

Sommaire

Titre	1
Sommaire	2
1 Introduction	3
2 Cahier des charges	3
3 Problématique	4
4 Architecture matérielle et logiciel	4
4.1 Le matériel utilisé	5
4.1.1 Arduino	5
4.1.2 Raspberry pi	6
4.1.3 Capteur AD8232	7
4.1.4 LoRa et The Things Network (TTN)	7
4.2 Les logiciels utilisés	8
4.2.1 Nodered	8
4.2.2 Gestionnaire de base de donnée : MariaDB	9
5 Configurations	10
5.1 Éléments de configuration matérielle	10
5.1.1 Connexion des composants	10
5.1.2 The Thing Network	12
5.2 Configuration côté client (Raspberry)	14
5.3 Configuration côté serveur (Medecin)	18
6 Conclusion	21
Bibliographie	23

1 Introduction

Au cœur des enjeux de la santé, la surveillance cardiaque à long terme représente un impératif crucial pour anticiper, diagnostiquer et traiter les troubles cardiaques. Dans ce contexte, l'avènement des technologies de l'Internet des Objets (IoT) offre des opportunités révolutionnaires pour la médecine moderne. L'électrocardiogramme (ECG), en tant qu'outil fondamental pour l'évaluation de l'activité cardiaque, devient plus accessible que jamais grâce à des dispositifs portables miniaturisés. Ce projet vise à concrétiser cette avancée en élaborant un Holter connecté, capable de surveiller de manière continue et non invasive l'ECG d'un individu sur une période prolongée. Au-delà de l'innovation technologique, l'impact potentiel de ce dispositif sur le domaine médical est immense, offrant aux professionnels de la santé une fenêtre inédite sur les variations subtiles de l'activité cardiaque, permettant ainsi une détection précoce des anomalies et une personnalisation accrue des traitements. Ce rapport documente le développement d'une Application IoT visant à créer un Holter connecté dédié à la mesure de l'électrocardiogramme (ECG).

Ce projet s'inscrit dans le cadre de l'unité d'enseignement Application objets connectés de l'université d'Avignon, il exige la mise en œuvre de compétences techniques avancées pour concevoir un dispositif capable de capturer, transmettre et stocker de manière sécurisée et efficace des données cardiaques. Le Holter connecté élaboré ici propose une solution innovante pour l'enregistrement à long terme de l'ECG, permettant aux professionnels de la santé d'analyser des données en dehors du contexte clinique traditionnel.

Dans ce rapport, on va détailler les étapes clés de la conception, du développement et de la mise en œuvre du Holter connecté, mettant en lumière les fonctionnalités essentielles facilitant la surveillance locale et distante des données cardiaques. L'architecture du système, fondée sur la technologie LoRa et utilisant des dispositifs tels que l'Arduino, THE THINGS, UNO et le Raspberry Pi 3, sera présentée en détail, illustrant ainsi l'intégration harmonieuse des composants pour assurer un fonctionnement optimal. Toutes nos réalisations sont disponibles sur notre dépôt GitHub à l'aide du lien suivant <https://github.com/madjiguene-cloud/application-objets-connect-s.git>[7].

2 Cahier des charges

Au cœur des enjeux médicaux contemporains, notre projet de Holter Connecté répond à une problématique cruciale : comment améliorer la surveillance cardiaque à long terme de manière non invasive et continue ? Cette interrogation émane des limites des méthodes traditionnelles, souvent ponctuelles, dans la détection précoce des anomalies cardiaques.

Le cahier des charges qui guide notre projet est élaboré avec minutie pour concrétiser cette réponse. Ainsi il nous a été demandé de créer un prototype fonctionnel capable de remonter des événements significatifs, tels que les dépassements de seuils de fréquence cardiaque, tout en enregistrant de manière locale l'électrocardiogramme (ECG) du patient sur une période étendue. La solution sera basée sur la version The Things Network, avec une station de mesure de l'ECG comprenant Arduino THE THINGS UNO, Raspberry Pi 3, une carte d'amplification AD8232 ECG et un jeu d'électrodes. Conditionnée dans un boîtier prototypé à l'imprimante 3D, la solution offrira un accès à distance aux données de surveillance via un site web, tandis qu'un site web local permettra l'accès en direct aux enregistrements de l'ECG avec la possibilité de régler les seuils limites à distance.

Les fonctionnalités obligatoires comprennent le développement d'interfaces de super-

vision et configuration, tant au niveau local que distant, avec la capacité d'accéder aux mesures directes de l'ECG, de tracer en direct son évolution, de consulter les enregistrements historisés, de surveiller la liaison LoRa, et de régler à distance les seuils limites pour la surveillance du rythme cardiaque.^[1]

3 Problématique

Au cœur des préoccupations médicales contemporaines, notre projet de Holter Connecté émerge en réponse à une problématique majeure : comment améliorer de manière significative la surveillance cardiaque continue pour une détection précoce et une gestion proactive des troubles cardiovasculaires ?

Les méthodes traditionnelles, souvent ponctuelles et limitées dans le temps, ne parviennent pas à fournir une vision exhaustive et en temps réel de l'activité cardiaque, laissant ainsi échapper des fluctuations subtiles et potentiellement significatives. Ces méthodes rencontrent également des obstacles pratiques tels que l'inconfort pour les patients et une fréquence limitée des mesures. C'est dans ce contexte que notre Holter connecté se positionne comme une solution innovante. En permettant une surveillance continue et non intrusive de l'électrocardiogramme sur des périodes étendues, il vise à surmonter les limitations inhérentes aux approches conventionnelles.

Cette problématique trouve sa pertinence dans la nécessité d'introduire des dispositifs technologiques sophistiqués pour offrir aux professionnels de la santé des informations précises, constantes, et en temps réel sur l'activité cardiaque des patients, ouvrant ainsi la voie à des stratégies de soins plus personnalisées et efficaces.

4 Architecture matérielle et logiciel

L'architecture du projet repose sur une combinaison de composants matériels et logiciels interconnectés pour réaliser la surveillance cardiaque continue.^[2]

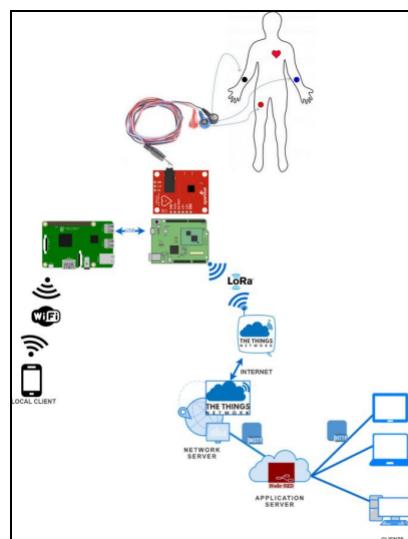


Figure 1. architecture du projet

4.1 Le matériel utilisé

4.1.1 Arduino

Arduino est une plateforme open-source de prototypage électronique qui permet aux amateurs, aux ingénieurs et aux développeurs de créer des projets interactifs. Il est composé d'un matériel, généralement basé sur un microcontrôleur, et d'un environnement de développement logiciel (IDE) facilitant la programmation.^[3]

Les principales caractéristiques d'Arduino incluent :

- Microcontrôleur : Les cartes Arduino sont équipées de microcontrôleurs, généralement de la famille Atmel AVR, dotés de diverses broches d'entrée/sortie permettant de connecter des capteurs, actionneurs et autres composants électroniques.
- Environnement de Développement : L'IDE Arduino fournit un ensemble d'outils pour programmer le microcontrôleur. Il utilise un langage de programmation simplifié basé sur le langage C/C++.
- Facilité d'Utilisation : Arduino est reconnu pour sa facilité d'utilisation, rendant la programmation et le prototypage électronique accessibles même aux débutants. Il utilise un ensemble de bibliothèques pré-écrites qui simplifient le développement.
- Polyvalence : Les cartes Arduino peuvent être utilisées dans une variété de projets, tels que la domotique, les objets connectés, les projets artistiques interactifs, les robots, etc.

Grâce à ses caractéristiques conviviales et à sa polyvalence, Arduino est devenu un outil populaire dans le domaine du prototypage électronique et de l'apprentissage de l'électronique et de la programmation, son micro-contrôleur permet, à partir d'événements détectés par des capteurs, de programmer et commander des actionneurs.

Dans notre projet son rôle consiste à interpréter les signaux électriques physiologiques provenant des capteurs AD8232. En utilisant le puissant microcontrôleur de l'Arduino, les variations subtiles dans le signal ECG sont traitées et préparées pour la transmission ultérieure.

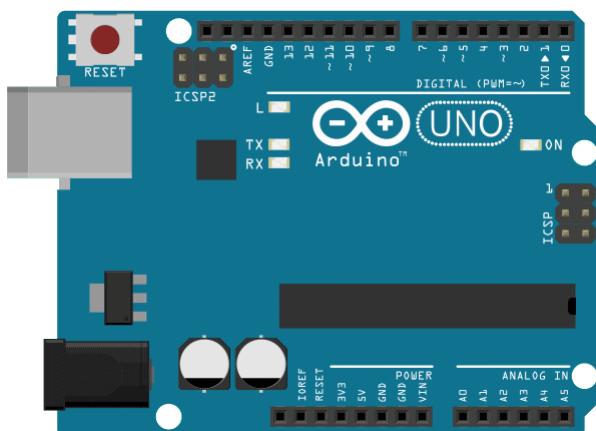


Figure 2. arduino

4.1.2 Raspberry pi

Le Raspberry Pi 3 est un ordinateur monocarte (single-board computer) de petite taille et à faible coût développé par la Fondation Raspberry Pi. Lancé en février 2016, le Raspberry Pi 3 constitue la troisième génération de la série Raspberry Pi. [4]



Figure 3. raspberry

Voici quelques caractéristiques clés du Raspberry Pi 3 :

- Processeur : Le Raspberry Pi 3 est équipé d'un processeur quad-core ARM Cortex-A53 cadencé à 1,2 GHz, offrant une puissance de traitement accrue par rapport aux modèles précédents.
- Mémoire RAM : Il dispose de 1 Go de mémoire RAM, ce qui améliore les performances globales et permet d'exécuter des applications plus complexes.
- Connectivité sans Fil : Le Raspberry Pi 3 intègre des fonctionnalités de connectivité sans fil avec un module Wi-Fi 802.11n et un module Bluetooth 4.2 intégré, éliminant ainsi le besoin d'adaptateurs externes.
- Ports d'Entrée/Sortie : Il propose une variété de ports d'entrée/sortie, y compris des ports USB, un port Ethernet, des ports HDMI, un port audio/vidéo, et des broches GPIO (General Purpose Input/Output) pour connecter des composants électroniques.
- Stockage : Pour le stockage, le Raspberry Pi 3 utilise des cartes microSD, offrant une solution pratique et amovible.
- Système d'Exploitation : Il prend en charge une gamme de systèmes d'exploitation, dont Raspbian (basé sur Linux), permettant aux utilisateurs d'exécuter diverses applications et projets.

Dans notre projet, la Raspberry Pi joue un rôle central en tant que plateforme de traitement local des données physiologiques provenant de l'Arduino. Voici ses différentes fonctionnalités :

- Réception des Données depuis l'Arduino : Elle est directement connectée à l'Arduino, recevant en temps réel les données physiologiques telles que les signaux électriques de l'électrocardiogramme (ECG).
- Traitement Local des Données : elle effectue un traitement initial des données localement. Cela inclut des opérations telles que l'amplification, la filtration, et d'autres

traitements nécessaires pour garantir la qualité des données avant leur utilisation ultérieure.

- Stockage Local des Données : La Raspberry Pi assure le stockage local des données, permettant la constitution d'un historique des mesures. Cette fonctionnalité garantit l'accès aux données même en l'absence de connectivité avec d'autres réseaux.
- Interfaces Utilisateur Locales : Gérant des interfaces utilisateur locales, elle offre la possibilité aux utilisateurs de superviser en temps réel l'évolution de l'ECG, d'accéder aux mesures historisées et de surveiller d'autres paramètres. Cela contribue à une utilisation autonome du dispositif.

En résumé, la Raspberry Pi agit comme le cerveau central du dispositif, assurant la réception, le traitement, le stockage local des données, et facilitant l'interaction avec les utilisateurs.

4.1.3 Capteur AD8232

AD8232 ECG Module Capteur de rythme cardiaque est utilisé pour mesurer l'activité électrique du cœur et représenter cette activité sous forme d'électrocardiogramme (ECG). Le capteur est un bloc de conditionnement de signal intégré pour les mesures biopotentielles, notamment pour l'ECG.

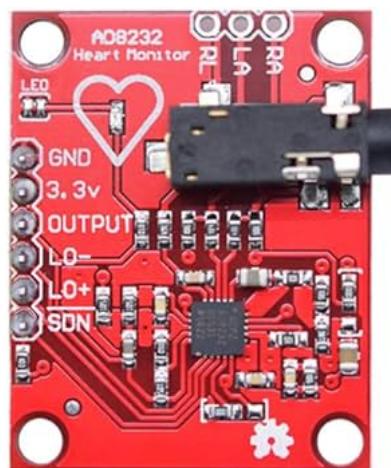


Figure 4. capteurs

Il est conçu pour extraire, amplifier et filtrer les petits signaux biopotentiels en présence de bruit, tels que ceux créés par le mouvement ou le placement des électrodes à distance.

4.1.4 LoRa et The Things Network (TTN)

LoRa (Long Range) est une technologie de communication sans fil conçue pour permettre la transmission longue portée de données avec une consommation d'énergie réduite. C'est une technologie idéale pour les applications Internet des objets (IoT) qui nécessitent une communication à faible débit sur de longues distances. LoRa utilise des fréquences radio non autorisées et permet une communication bidirectionnelle entre les noeuds, offrant ainsi une solution pratique pour des applications telles que la surveillance à distance, la télémétrie et d'autres systèmes IoT.

The Things Network (TTN) est une infrastructure ouverte et décentralisée basée sur la technologie LoRaWAN (LoRa Wide Area Network). TTN fournit une infrastructure gratuite et communautaire pour les appareils IoT basés sur LoRa, permettant une connectivité à faible coût sur de longues distances. La communauté TTN met en place des passerelles (gateways) LoRaWAN, qui sont des points d'accès au réseau, permettant aux dispositifs LoRa de transmettre leurs données à des applications hébergées dans le cloud.^[5]

En résumé, LoRa est la technologie sous-jacente qui permet la communication longue portée et à faible consommation d'énergie, tandis que The Things Network est une plateforme communautaire qui exploite LoRaWAN pour offrir une connectivité IoT à grande échelle, couvrant des zones étendues grâce à des passerelles déployées par la communauté. Dans le contexte de votre projet, l'utilisation de LoRa et The Things Network pourrait permettre la transmission sans fil des données collectées par le holter connecté vers un serveur distant.

4.2 Les logiciels utilisés

4.2.1 Nodered

Node-RED est un outil de développement low-code basé sur les flux pour la programmation visuelle, développé à l'origine par IBM pour connecter des périphériques matériels, des API et des services en ligne dans le cadre de l'Internet des objets.

Node-RED fournit un éditeur de flux basé sur un navigateur Web, qui peut être utilisé pour créer des fonctions JavaScript. Les éléments des applications peuvent être sauvegardés ou partagés pour être réutilisés. Le runtime est basé sur Node.js. Les flux créés dans Node-RED sont stockés à l'aide de JSON. Depuis la version 0.14, les nœuds MQTT peuvent établir des connexions TLS correctement configurées.

Dans notre projet, Node-RED joue un rôle essentiel en tant que plateforme de développement visuel pour la gestion des flux de données, la récupération, le traitement et l'acheminement des informations provenant du capteur AD8232 et de l'Arduino. Voici ses principales fonctions dans notre projet :

- Récupération des Données : Node-RED facilite la mise en place d'un flux permettant de récupérer les données émises par le capteur AD8232 via l'Arduino. Il peut être configuré pour surveiller en continu les données provenant de l'Arduino, assurant ainsi une acquisition régulière des signaux cardiaques.
- Traitement des Données : Une fois les données récupérées, Node-RED offre la possibilité de les traiter en temps réel. On peut mettre en place des opérations de traitement telles que l'amplification, la filtration ou la conversion des données, garantissant ainsi leur qualité et leur cohérence avant de les transmettre à d'autres composants du système.
- Connectivité avec d'Autres Services : Node-RED propose une variété de nœuds (nodes) prédéfinis qui simplifient l'intégration avec d'autres services, bases de données ou plateformes cloud. On peut configurer des nœuds pour envoyer les données traitées vers un serveur distant, une base de données ou tout autre service requis dans votre architecture.
- Interface Utilisateur : Node-RED peut être utilisé pour créer des interfaces utilisateur visuelles permettant aux utilisateurs de surveiller en temps réel les données cardiaques.

Cela inclut la visualisation des signaux ECG, des données historiques et d'autres informations pertinentes pour la surveillance cardiaque.

- Flexibilité et Personnalisation : L'approche visuelle de Node-RED permet une configuration rapide et une adaptation facile du flux en fonction des besoins spécifiques du projet.

On peut dire que Node-RED agit comme un orchestrateur central dans le projet, facilitant la récupération, le traitement et la transmission des données cardiaques tout en offrant une flexibilité et une facilité d'utilisation significatives dans le développement de votre système de surveillance cardiaque connecté.

4.2.2 Gestionnaire de base de donnée : MariaDB

MariaDB est un système de gestion de base de données relationnelles (SGBDR) open source. Il a été développé en tant que fork de MySQL. MariaDB partage une compatibilité élevée avec MySQL, ce qui signifie que les utilisateurs peuvent généralement migrer de MySQL vers MariaDB sans beaucoup de modifications au niveau des applications. Il est distribué sous une licence open source, ce qui signifie que son code source est librement accessible, modifiable et distribuable par n'importe qui. MariaDB sert de référentiel central pour stocker toutes les données collectées par le capteur AD8232 et traitées par Node-RED. Cela inclut les mesures de l'électrocardiogramme (ECG), les informations sur la fréquence cardiaque, et d'autres données physiologiques pertinentes.

5 Configurations

5.1 Éléments de configuration matérielle

5.1.1 Connexion des composants

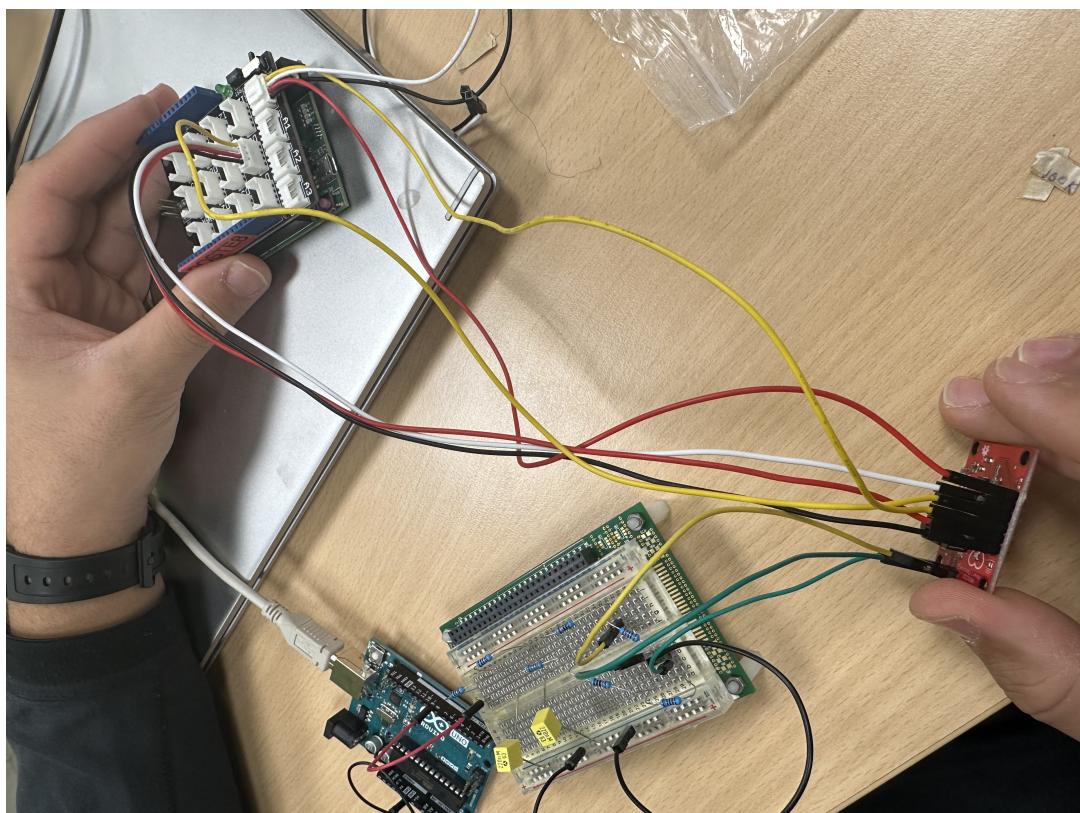
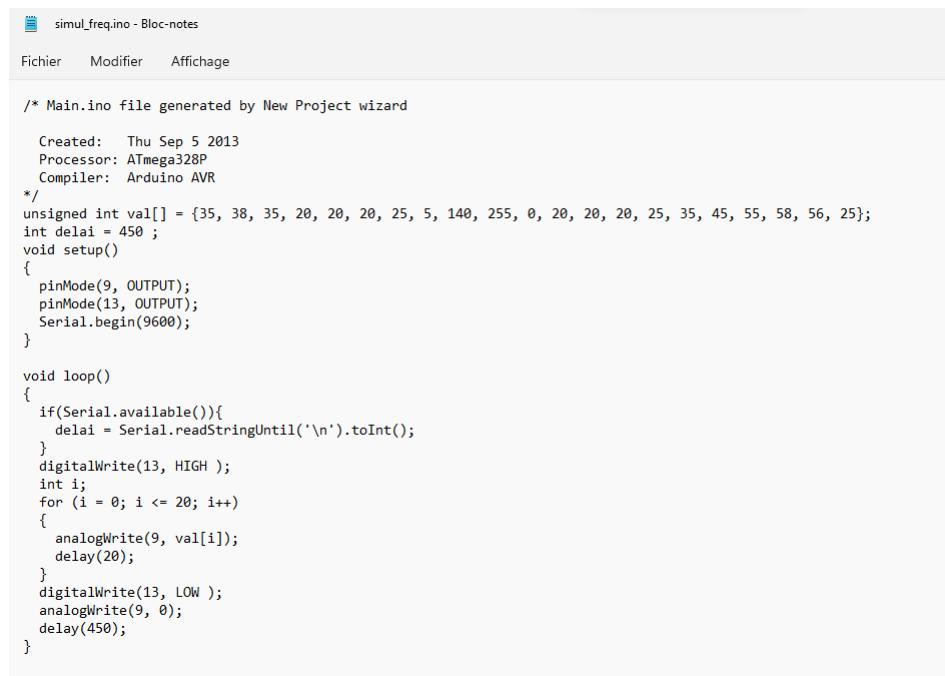


Figure 5. Connexion des composants

Nous avons mis en place une simulation de battements de cœur à l'aide de la carte Arduino Uno. Ce dernier génère des signaux simulés grâce au code suivants :



The screenshot shows a code editor window titled "simul_freq.ino - Bloc-notes". The menu bar includes "Fichier", "Modifier", and "Affichage". The code itself is an Arduino sketch:

```
/* Main.ino file generated by New Project wizard

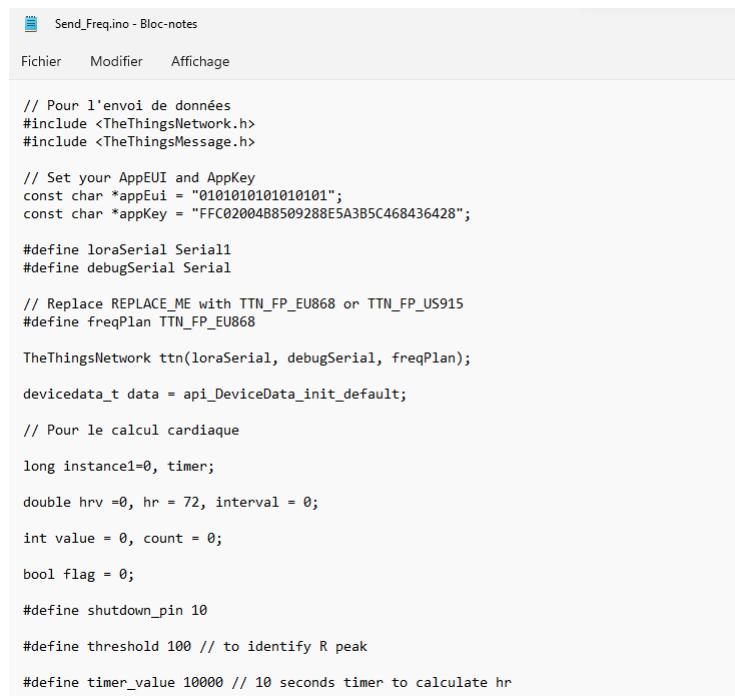
Created: Thu Sep 5 2013
Processor: ATmega328P
Compiler: Arduino AVR
*/
unsigned int val[] = {35, 38, 35, 20, 20, 20, 25, 5, 140, 255, 0, 20, 20, 20, 25, 35, 45, 55, 58, 56, 25};
int delai = 450 ;
void setup()
{
  pinMode(9, OUTPUT);
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  if(Serial.available()){
    delai = Serial.readStringUntil('\n').toInt();
  }
  digitalWrite(13, HIGH );
  int i;
  for (i = 0; i <= 20; i++)
  {
    analogWrite(9, val[i]);
    delay(20);
  }
  digitalWrite(13, LOW );
  analogWrite(9, 0);
  delay(450);
}
```

Figure 6. Simulations de battements de coeur

Ce code utilise la modulation PWM sur la broche 9 pour générer un signal analogique simulant les variations du signal électrique du cœur.

Le capteur, connecté à la carte Arduino Uno et à la carte Arduino Leonardo, capture le signal et le transmet à la Leonardo, qui commence par lire la fréquence cardiaque. Ensuite, elle envoie ces données au TTN (The Things Network) et aux ports série. La tâche de la Arduino Leonardo consiste donc à lire, traiter les données, calculer la fréquence cardiaque, et les transmettre au TTN pour les professionnels de la santé (serveurs) et aux ports série pour la Raspberry Pi, destinée aux patients. Le code téléchargé sur la carte Arduino Leonardo est présenté ci-dessous.



```

Send_Freq.ino - Bloc-notes

Fichier Modifier Affichage

// Pour l'envoi de données
#include <TheThingsNetwork.h>
#include <TheThingsMessage.h>

// Set your AppEUI and AppKey
const char *appEui = "0101010101010101";
const char *appKey = "FFC0200488509288E5A3B5C468436428";

#define loraSerial Serial1
#define debugSerial Serial

// Replace REPLACE_ME with TTN_FP_EU868 or TTN_FP_US915
#define freqPlan TTN_FP_EU868

TheThingsNetwork ttn(loraSerial, debugSerial, freqPlan);

devicedata_t data = api_DeviceData_init_default;

// Pour le calcul cardiaque

long instance1=0, timer;

double hrv =0, hr = 72, interval = 0;

int value = 0, count = 0;

bool flag = 0;

#define shutdown_pin 10

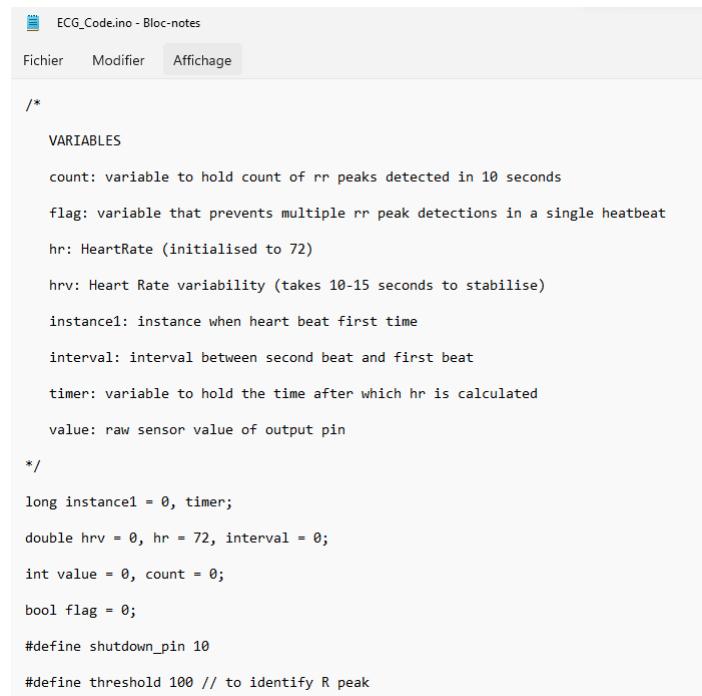
#define threshold 100 // to identify R peak

#define timer_value 10000 // 10 seconds timer to calculate hr

```

Figure 7. Envoie de fréquence

Le code suivant nous permet de mesurer le rythme cardiaque (Heart Rate – HR) et la variabilité du rythme cardiaque (Heart Rate Variability – HRV) à partir d'un capteur d'ECG.



```

ECG_Code.ino - Bloc-notes

Fichier Modifier Affichage

/*
VARIABLES
count: variable to hold count of rr peaks detected in 10 seconds
flag: variable that prevents multiple rr peak detections in a single heartbeat
hr: HeartRate (initialised to 72)
hrv: Heart Rate variability (takes 10-15 seconds to stabilise)
instance1: instance when heart beat first time
interval: interval between second beat and first beat
timer: variable to hold the time after which hr is calculated
value: raw sensor value of output pin
*/
long instance1 = 0, timer;

double hrv = 0, hr = 72, interval = 0;

int value = 0, count = 0;

bool flag = 0;

#define shutdown_pin 10

#define threshold 100 // to identify R peak

```

Figure 8. Calcul HR

5.1.2 The Thing Network

Pour établir la connectivité entre notre dispositif IoT et le réseau LoRaWAN via The Things Network (TTN), nous avons suivi une série d'étapes clés, détaillées ci-dessous :

Création d'un Compte TTN : Nous avons créé un compte sur la plateforme TTN, accessible à l'adresse <https://eu1.cloud.thethings.network/console/>. Les identifiants utilisés étaient "m2-sicom-iot" pour le nom d'utilisateur et "m2-sicom-iot" pour le mot de passe.

Création d'une Application : Au sein de notre compte TTN, nous avons créé une application dédiée à notre projet, que nous avons nommée "project-boudra-diedhiou". Cette application a servi de conteneur logique pour regrouper tous nos end devices et gérer les flux de données générées par notre dispositif connecté.

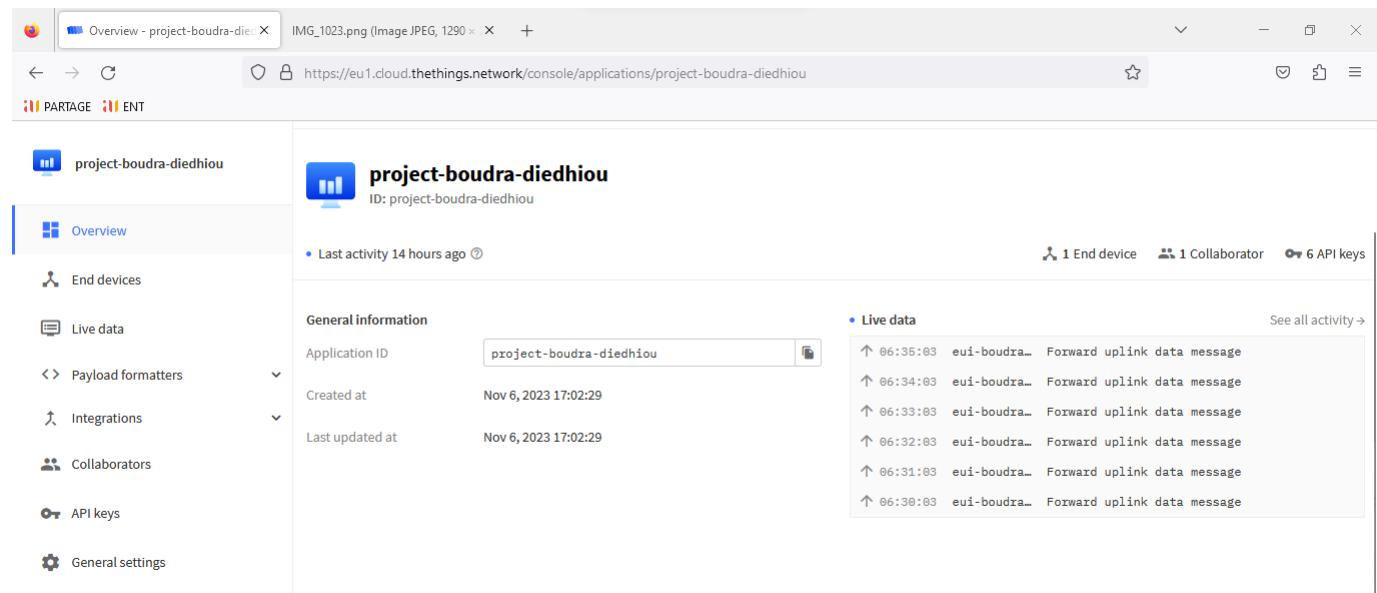


Figure 9. Application TTN

Ajout de l'End Device (Arduino) : Nous avons ajouté un end device représentant notre Arduino équipé du capteur AD8232 à l'intérieur de notre application TTN. Cette étape a impliqué la génération des clés d'identification nécessaires, telles que l'AppEUI, DevEUI et AppKey.

Nous avons intégré les clés d'identification générées (AppEUI, DevEUI, et AppKey) directement dans le code Arduino. Cette démarche stratégique permet à notre Arduino, équipé du capteur AD8232, de s'authentifier de manière sécurisée sur le réseau The Things Network (TTN) lors de chaque transmission de données. Ces clés d'identification sont essentielles pour établir une connexion fiable et sécurisée, garantissant que seuls les dispositifs autorisés peuvent communiquer avec le réseau TTN.

Rapport Application IOT

The screenshot shows the 'End devices' section of the The Things Network console. On the left, a sidebar lists 'project-boudra-diedhiou' and various configuration tabs: Overview, End devices (selected), Live data, Payload formatters, Integrations, Collaborators, API keys, and General settings. The main panel displays the device 'eui-boudra-diedhiou-ecg' with its ID. It shows activity statistics: 65 uplinks and 0 downlinks, with the last activity being 14 hours ago. Below this are tabs for Overview, Live data, Messaging, Location, Payload formatters, and General settings. The 'Overview' tab is selected. Under 'General information', fields include End device ID (eui-boudra-diedhiou-ecg), Frequency plan (Europe 863-870 MHz (SF9 for RX2 - recommended)), LoRaWAN version (LoRaWAN Specification 1.0.2), and Regional Parameters version (RP001 Regional Parameters 1.0.2 revision_1). The 'Live data' tab shows a list of recent messages, with the first few entries being: '↑ 06:35:03 Forward uplink data message', '↑ 06:35:03 Successfully processed data message', '↑ 06:34:03 Forward uplink data message', '↑ 06:34:03 Successfully processed data message', '↑ 06:33:03 Forward uplink data message', and '↑ 06:33:03 Successfully processed data message'. A 'See all activity' link is at the top right of this section.

Figure 10. END DEVICE

Configuration de La passerelle : la passerelle LoRa joue un rôle fondamental en tant que point d'accès, facilitant la transmission bidirectionnelle des données entre les dispositifs IoT de notre projet, notamment l'Arduino doté du capteur AD8232, et le réseau The Things Network (TTN). En agissant comme un relais intelligent, elle permet aux dispositifs distants de transmettre leurs données au réseau TTN et vice versa.

The screenshot shows the 'Gateway' section of the The Things Network console. The gateway is identified as 'eui-58a0cbfffe803e40-boudra' with ID: eui-58a0cbfffe803e40-boudra. It shows activity statistics: 51 uplinks and 0 downlinks, with the last activity being 7 minutes ago. The 'General information' tab is selected, displaying fields for Gateway ID (eui-58a0cbfffe803e40-boudra), Gateway EUI (58 A0 C8 FF FE 80 3E 40), Gateway description (None), Created at (Jan 2, 2024 14:16:53), Last updated at (Jan 4, 2024 14:23:25), and Gateway Server address (eu1.cloud.thethings.network). The 'Live data' tab shows a single entry: '↑ 20:31:20 Receive uplink message DevAddr: 0F 32 B5 EF F0n1'. A 'See all activity' link is at the top right. The 'Location' tab shows a map of Europe with a marker indicating the gateway's location. A 'Change location settings' link is at the top right of the map area.

Figure 11. passerelle TTN

5.2 Configuration côté client (Raspberry)

Dans cette section du rapport, nous allons détailler comment la configuration côté client, spécifiquement sur le Raspberry Pi, est réalisée pour récupérer les données provenant de l'Arduino via les ports série.

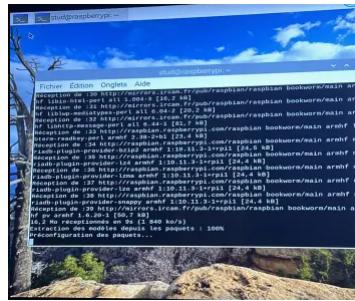


Figure 12. Demarrage de rapsberry

Nous avons configuré MariaDB sur le Raspberry Pi afin de mettre en place un environnement spécifique destiné au stockage des données cardiaques.

Les commandes utilisés sont les suivantes :

Créer Base de données `ecg_data` :

`CREATE DATABASE ecg_data` Créer les 2 tables `freq` et `freq_sup` :

`CREATE TABLE freq (Date DATETIME, BPM INT)`

`CREATE TABLE freq_sup (Date DATETIME, BPM INT)`

Créer utilisateur et lui donner les droits :

`CREATE USER 'nouveau_utilisateur'@'localhost' IDENTIFIED BY 'mot_de_passe';`

`GRANT ALL PRIVILEGES ON *.* TO 'nouveau_utilisateur'@'localhost';`

`FLUSH PRIVILEGES`

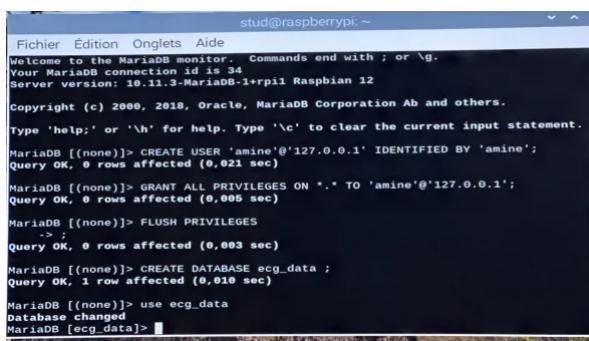


Figure 13. Creation de base de données

Node-RED a été installé sur le Raspberry Pi, offrant une interface conviviale pour la création de flux de traitement de données. Les noeuds spécifiques à la communication série ont été intégrés dans Node-RED pour garantir une interaction fluide avec les ports série de l'Arduino.

les commandes utilisés pour installer node-red sont les suivantes :

```
curl -sL https://deb.nodesource.com/setup_14.x |  
sudo -E bash -  
sudo apt update sudo apt install nodejs  
npm -v  
apt install npm  
sudo npm install -g --unsafe-perm node-red
```

Sur nodered dans manage palette il faut installer les options suivantes : Mysql, Moment,

Dashboard, Serial et Ui-Table.

- MySQL : est utilisé pour se connecter à une base de données MySQL depuis Node-RED. Il permet d'exécuter des requêtes SQL et de manipuler les données stockées dans une base de données MySQL.
- Moment : est utilisé pour manipuler et formater des dates et heures dans Node-RED. Il facilite le traitement des données temporelles en utilisant la bibliothèque JavaScript Moment.js.
- Dashboard : permet de créer un tableau de bord (dashboard) dans Node-RED. Il offre une interface utilisateur graphique pour visualiser et interagir avec les données de manière plus conviviale. Vous pouvez créer des tableaux de bord interactifs avec des graphiques, des boutons, des jauge, etc.
- Serial : Le nœud Serial est utilisé pour la communication série avec des périphériques matériel tels que des microcontrôleurs, des capteurs, etc. Il permet à Node-RED de recevoir et d'envoyer des données via des ports série.
- Ui-Table : est utilisé pour afficher des données sous forme de tableau dans le tableau de bord. Il peut être utilisé pour présenter des données tabulaires de manière structurée et permet souvent aux utilisateurs de trier, filtrer ou rechercher des données dans le tableau.

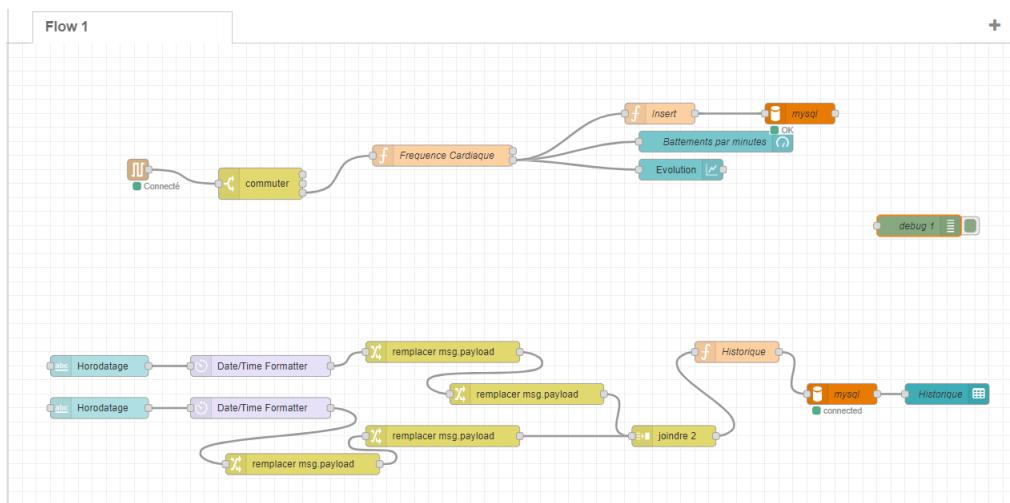


Figure 14. le nodered de la raspberry

Dans le flux "fréquence cardiaque", nous récupérons la fréquence cardiaque et la stockons dans notre base de données. Nous affichons également le nombre de battements par minute ainsi que l'évolution des battements.

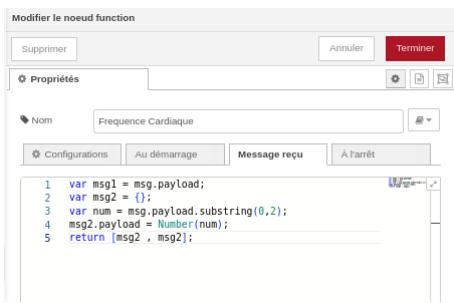


Figure 15. recuperer la fréquence cardiaque

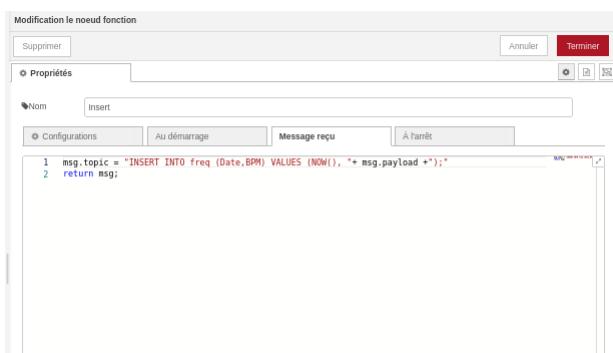


Figure 16. insérer les fréquence dans la base de donnée

Le flux "horodatage" offre à l'utilisateur la possibilité de saisir une date et de consulter l'historique. Lorsque nous récupérons les données saisies, elles sont initialement sous forme numérique. L'option Moment nous permet de formater ces données en remplaçant les chiffres par une date. Après le formatage dans le flux "historique", une requête SQL est effectuée pour afficher l'historique correspondant à la date demandée par l'utilisateur.[6]

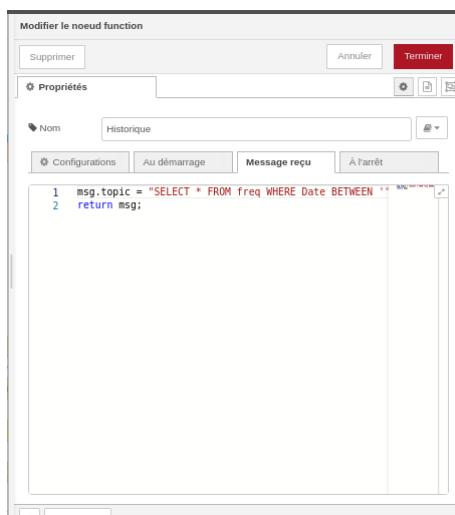


Figure 17. Requête pour afficher l'historique

Sur la figure suivante, l'affichage visible par le patient en local est présenté. Il lui permet d'observer sa fréquence cardiaque en temps réel, d'effectuer des recherches dans l'historique des données, et de suivre l'évolution de sa fréquence cardiaque au fil du temps.

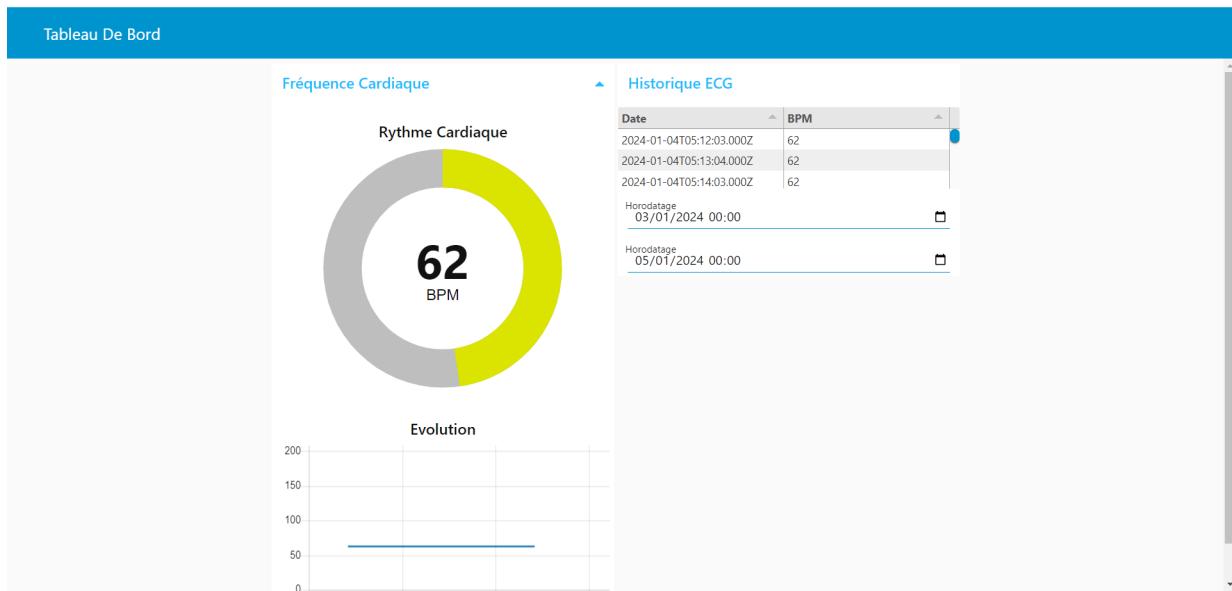


Figure 18. Affichage chez le patient

5.3 Configuration côté serveur (Medecin)

La configuration côté serveur est presque identique à celle en local, à la différence près que cette fois-ci, les données sont récupérées depuis The Things Network (TTN).

Nous avons installé MariaDB pour le stockage des données, et nous avons créé la base de données nommée "ecg_data".

```
[root@kali: ~]# mysql
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 62
Server version: 10.1.6-MariaDB-1 Debian 10
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> CREATE DATABASE ecg_data ;
```

Figure 19. ecg_data

Nous créons la table "freq" qui va stocker les fréquences enregistrées à une date donnée.

```
Database changed
MariaDB [ecg_data]> CREATE TABLE freq ( Date DATETIME , BPM INT ) ;
```

Figure 20. création de table

Affichons le contenu de la table freq

```
470 rows in set (0.013 sec)
MariaDB [ecg_data]> select * from freq_sup ;
+-----+-----+
| Date | BPM |
+-----+-----+
| 2024-01-03 10:17:18 | 62 |
| 2024-01-03 10:36:18 | 62 |
| 2024-01-04 01:28:03 | 62 |
+-----+-----+
3 rows in set (0.000 sec)

MariaDB [ecg_data]>
```

Figure 21. Affichage table

Nous avons également installer Nodered avec les commandes suivantes :

```
curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
sudo apt update
```

```

sudo apt install nodejs
npm -v
apt install npm
sudo npm install -g -unsafe-perm node-red
apt install docker.io
docker run -it -p 1880:1880 --name mynodered nodered/node-red

```

Nous allons maintenant configurer Node-RED pour établir une connexion avec The Things Network (TTN) et récupérer les données.



Figure 22. Link entre TTN et Nodered

La capture ci-dessous présente les flux que nous avons mis en place.

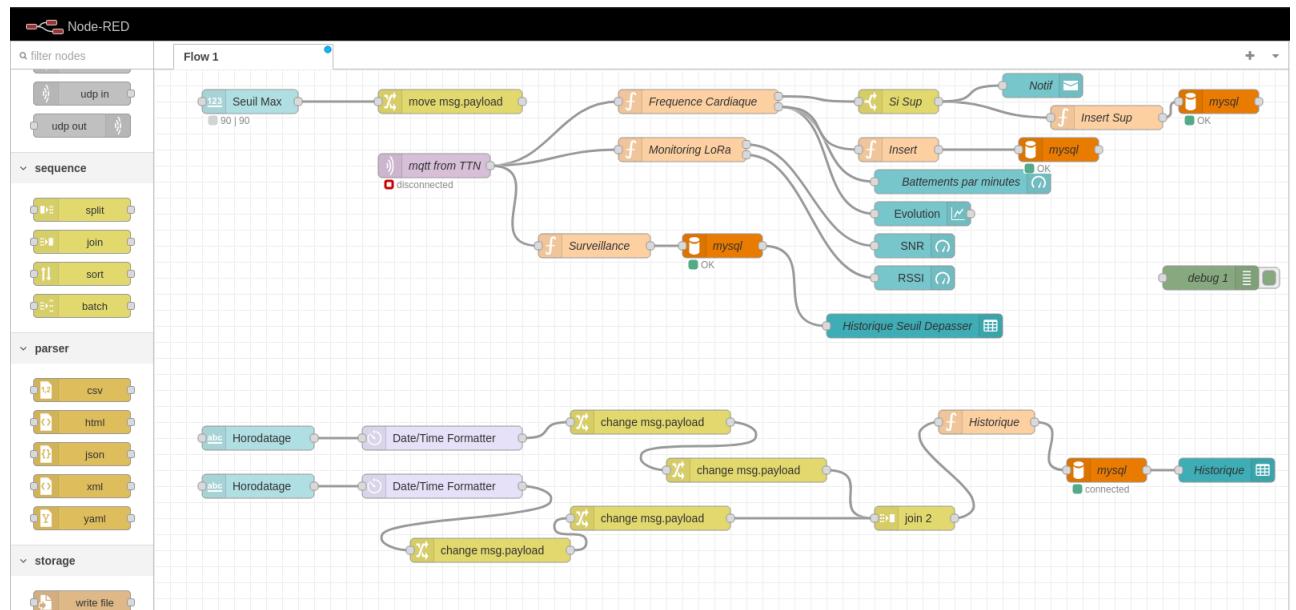


Figure 23. nodered medecin

Nous avons récupéré les fréquences cardiaques depuis TTN à l'aide de la fonction suivante, puis nous les avons stockées dans la base de données.

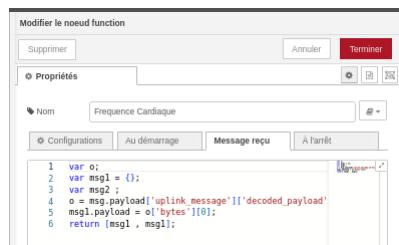


Figure 24. récupération fréquence cardiaque

le flux "Monitoring Lora" nous permet de surveiller la qualité du signal de la liaison LoRa et affichons le SNR (Rapport Signal sur Bruit) et le RSSI (Indicateur de Force du Signal Reçu) dans le tableau de bord.

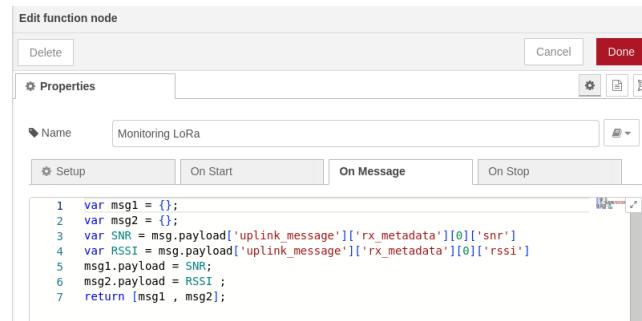


Figure 25. afficher qualité du signal

Les fréquences cardiaques récupérées sont placées sous surveillance. Si le nombre de battements par minute (BPM) dépasse un seuil prédéfini, la fréquence est stockée dans la base de données. Cette information est également affichée, et le médecin peut consulter ces données. Le seuil peut être défini par le médecin en fonction des besoins spécifiques du patient.

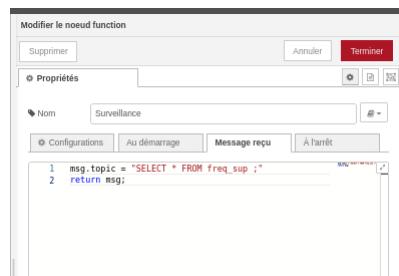


Figure 26. le flux surveiller fréquence

Le flux "horodatage" offre au medecin la possibilité de saisir une date et de consulter l'historique. Lorsque nous récupérons les données saisies, elles sont initialement sous forme numérique. L'option Moment nous permet de formater ces données en remplaçant les chiffres par une date. Après le formatage dans le flux "historique", une requête SQL est effectuée pour afficher l'historique correspondant à la date demandée par l'utilisateur.

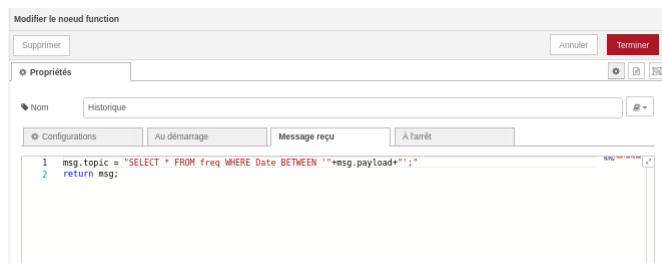


Figure 27. recherche de l'historique

Le médecin dispose dans son tableau de bord de deux tableaux distincts. Le premier lui permet de surveiller les données en fonction d'un seuil défini, tandis que le second lui offre la possibilité de rechercher dans l'historique des électrocardiogrammes (ECG) à partir d'une date spécifique. De plus, le médecin a la possibilité de surveiller la qualité du signal LoRa, lui permettant ainsi de déterminer la fiabilité des résultats avant d'émettre un diagnostic.

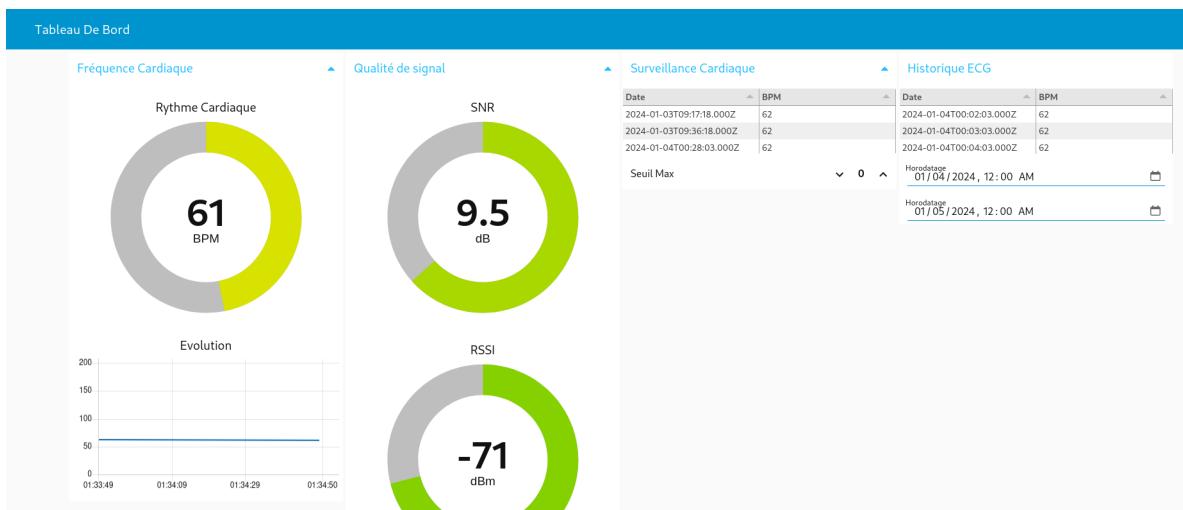


Figure 28. Le dashboard du medecin

6 Conclusion

La réalisation de ce projet dédié à la conception d'un holter connecté pour la surveillance cardiaque continue représente une étape significative dans le domaine de la santé connectée et de l'Internet des Objets (IoT). À travers l'intégration de technologies telles que Arduino, Raspberry Pi, Node-RED, LoRa, The Things Network, et MariaDB, nous avons élaboré un prototype fonctionnel répondant aux exigences de la surveillance cardiaque à long terme.

Ce projet a permis d'explorer en profondeur les aspects techniques et pratiques liés à la mise en œuvre d'un dispositif médical IoT. La station de mesure de l'électrocardiogramme (ECG) construite autour de l'association Arduino THE THINGS UNO / Raspberry Pi 3, du capteur AD8232 ECG, et du bus de communication LoRa, démontre la viabilité de notre approche.

Les fonctionnalités développées, tant au niveau local avec la supervision et configuration via une interface dédiée, qu'au niveau distant avec l'accès aux données via The Things Network, offrent une solution complète pour la surveillance et l'analyse du rythme cardiaque. L'utilisation de MariaDB pour le stockage des données contribue à la pérennité de l'information médicale.

Cependant, malgré les succès obtenus, des perspectives d'amélioration subsistent. Des développements futurs pourraient se concentrer sur l'optimisation de l'interface utilisateur, l'ajout de fonctionnalités avancées d'analyse des données, et la conformité aux normes de sécurité médicale.

En conclusion, ce projet a été une expérience enrichissante, nous permettant d'appliquer nos connaissances théoriques à des défis pratiques, et ouvrant la voie à des innovations futures dans le domaine de la télémédecine et de la surveillance cardiaque connectée. Nous sommes convaincus que cette solution pourrait contribuer positivement à l'amélioration des soins de santé, offrant une surveillance cardiaque continue et accessible, améliorant ainsi la qualité de vie des patients.

Références

- [1] Rapp. tech. url : <https://circuitdigest.com/microcontroller-projects/understanding-ecg-sensor-and-program-ad8232-ecg-sensor-with-arduino-to-diagnose-various-medical-conditions>.
- [2] Rapp. tech. url : https://www.researchgate.net/publication/324157382_AD8232-based_Smart_Healthcare_System_using_Internet_of_Things_IoT.
- [3] Rapp. tech. url : <https://www.positron-libre.com/electronique/arduino/arduino.php>.
- [4] Rapp. tech. url : <https://www.raspberrypi.com/documentation/>.
- [5] Rapp. tech. url : <https://www.thethingsnetwork.org/docs/applications/>.
- [6] Rapp. tech. url : <https://nodered.org/docs/user-guide/>.
- [7] Aïda DIEDHIOU et Amine BOUDRA. Rapp. tech. url : <https://github.com/madjiguene-cloud/application-objets-connect-s.git>.