



Documentation technique MedTrackr



UNIVERSITÉ
CÔTE D'AZUR

Réalisé par :

Buonocore Lucas

Delemarre Antoine

Durand Théo

Hallouli-Errafik Amin

Pujos Pierre

BUT3 Informatique

Table des matières

Introduction	4
Présentation du logiciel	4
Objectifs de la documentation	4
Architecture de l'application	5
Architecture globale	5
Front-end	6
Back-end	8
Stratégie pour collecter les données	10
Stratégie pour stocker les données côté serveur	10
Stratégie pour échanger les données entre smartphone et serveur	11
Stratégie de déploiement de l'application :	11
Fonctionnalités	13
Cas d'usages	13
Installation et configuration.....	14
Lancement local	14
Connexion au cloud.....	14
Fonctionnalités.....	15
Générales.....	15
Patient	Error! Bookmark not defined.
Médecin.....	Error! Bookmark not defined.
Proche	Error! Bookmark not defined.
Guide de l'utilisateur :.....	18
Interface médecin.....	20
Interface patient	21
Comment contribuer à notre projet.....	26

Introduction

Présentation du logiciel

Pour les personnes âgées et les personnes en situation de handicap nécessitant un suivi constant de son état de santé, MedTrackr est un logiciel de monitoring qui permet à l'utilisateur d'avoir en temps réel son état de santé et envoie ces données à son médecin traitant. Le projet vise à créer une application permettant à l'utilisateur de voir ses données de santé en temps réel et de les envoyer régulièrement à son médecin et à ses proches. Le but est d'aider les patients à recevoir un meilleur suivi médical.

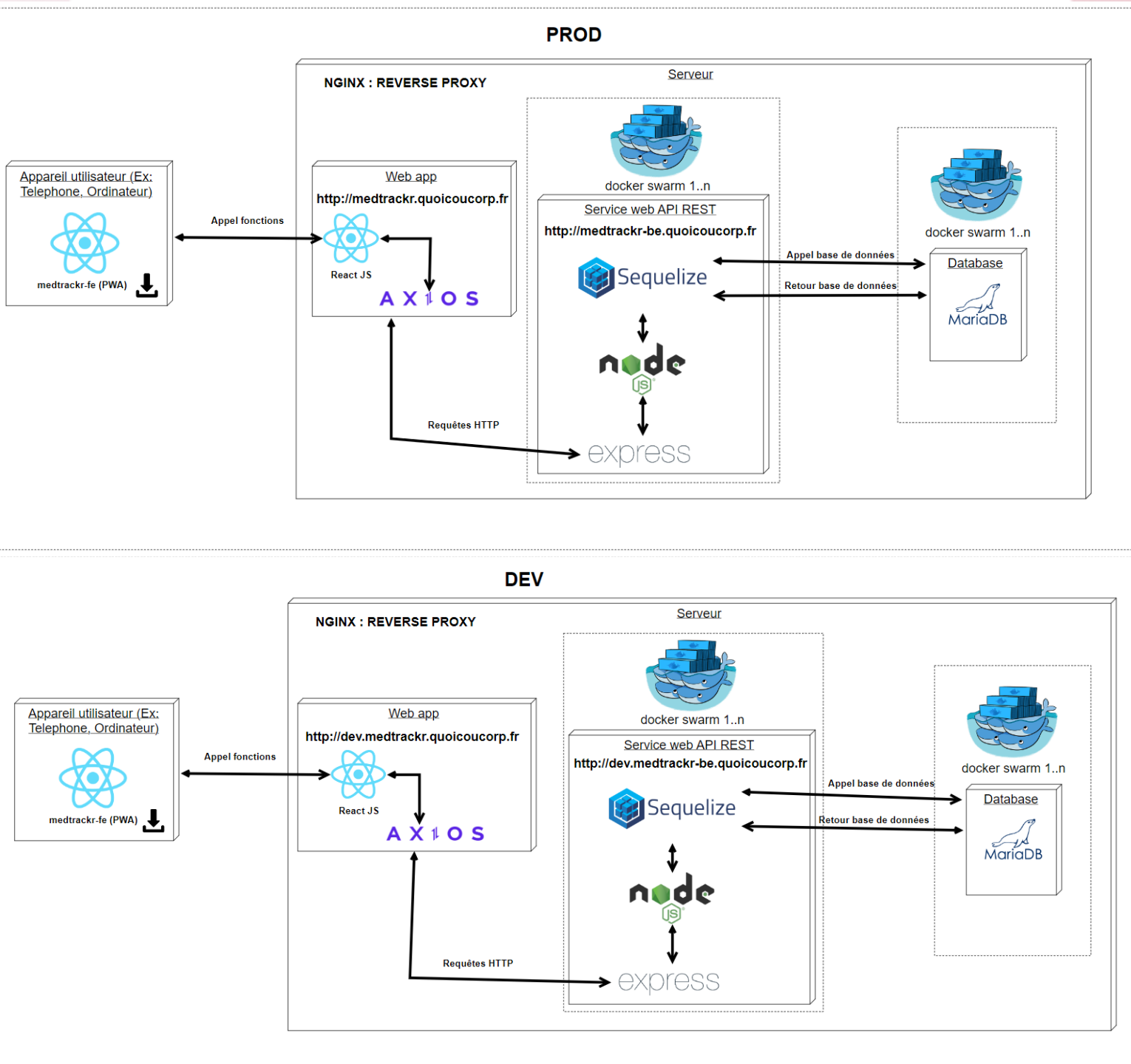
Objectifs de la documentation

Cette documentation s'adresse aux développeurs souhaitant contribuer ou réutiliser le code de notre application. Elle est également conçue pour les utilisateurs afin de les guider dans l'utilisation de l'application à l'aide du guide fourni ultérieurement.

Architecture de l'application

Architecture globale

MedTrackr est basé sur une architecture de type REST qui lui permet d'être **moderne** et **évolutive** et de bien séparer les différentes parties du code.



Notre logiciel a été déployé sur un serveur distant, hébergeant le back-end, le front-end et la base de données.

L'accès à l'application (front-end) en production se fait directement depuis le navigateur via l'adresse <http://medtrackr.quoicoucorp.fr>.

De même, l'accès au back-end de l'application en production est possible directement depuis le navigateur à l'adresse <http://medtrackr-be.quoicoucorp.fr>.

Il n'y a pas de différence entre l'architecture de l'environnement de production et de l'environnement de développement, seul les adresses http changent.

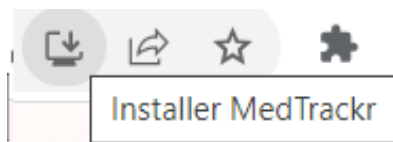
Pour accéder aux routes de l'application en développement, il suffit de rajouter « dev. » au début des adresses :

- <http://dev.medtrackr.quoicoucorp.fr> : accéder à l'application (front-end) en développement
- <http://dev.medtrackr-be.quoicoucorp.fr> : accéder au back-end de l'environnement

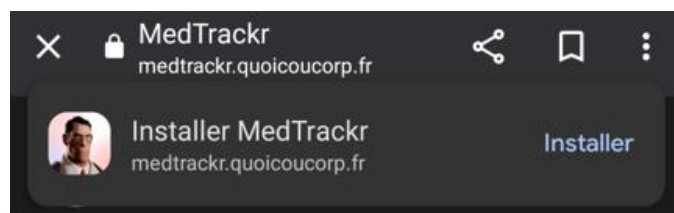
L'accès au site est sécurisé grâce à la mise en place d'un reverse proxy qui nous permet d'avoir du HTTPS sur notre site web.

Grace à cela, l'application est également installable directement depuis le navigateur.

Sur PC :



Sur téléphone (android) :



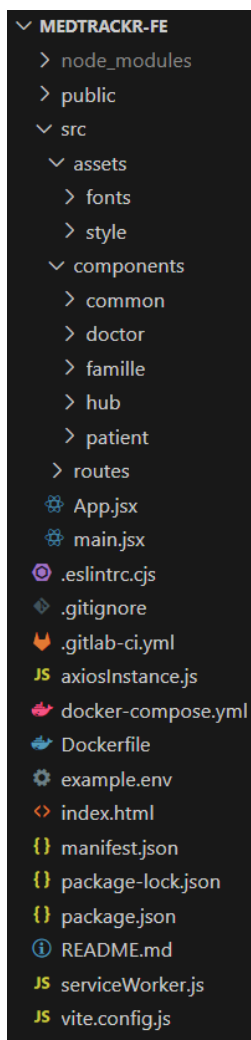
Front-end

Concernant la partie front-end de l'application le tout a été réalisé en **React** qui permet à notre application de fonctionner sans recharger la page, facilitant ainsi **la navigation** sur le site et permettant **d'améliorer l'expérience utilisateur**.

Afin d'améliorer l'expérience utilisateur, nous avons développé une application responsive, capable de s'ajuster automatiquement à la taille de l'écran de chaque utilisateur.

Concernant l'organisation du code, celui-ci est structuré en trois répertoires principaux. Le répertoire "src" contient le code React, avec les composants regroupés dans le sous-répertoire "components", organisés en fonction des types d'utilisateurs. Les fichiers CSS, les images et les polices de caractères sont stockés dans le répertoire "assets", chacun ayant son propre sous-répertoire. Le répertoire "émulateur" renferme les données des divers types de capteurs, tandis que le répertoire "axios" est dédié à la création de requêtes HTTP pour communiquer avec le back-end.

Voici la structure de notre projet front-end sur Visual Studio Code



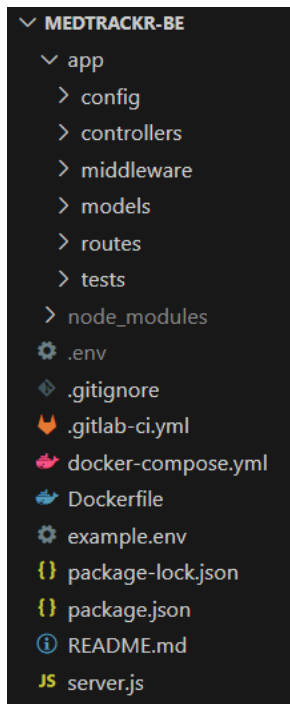
Back-end

Pour le back-end de l'application nous utilisons **Node.js** qui permet de gérer les requêtes des utilisateurs et communiquer directement avec notre base de données.

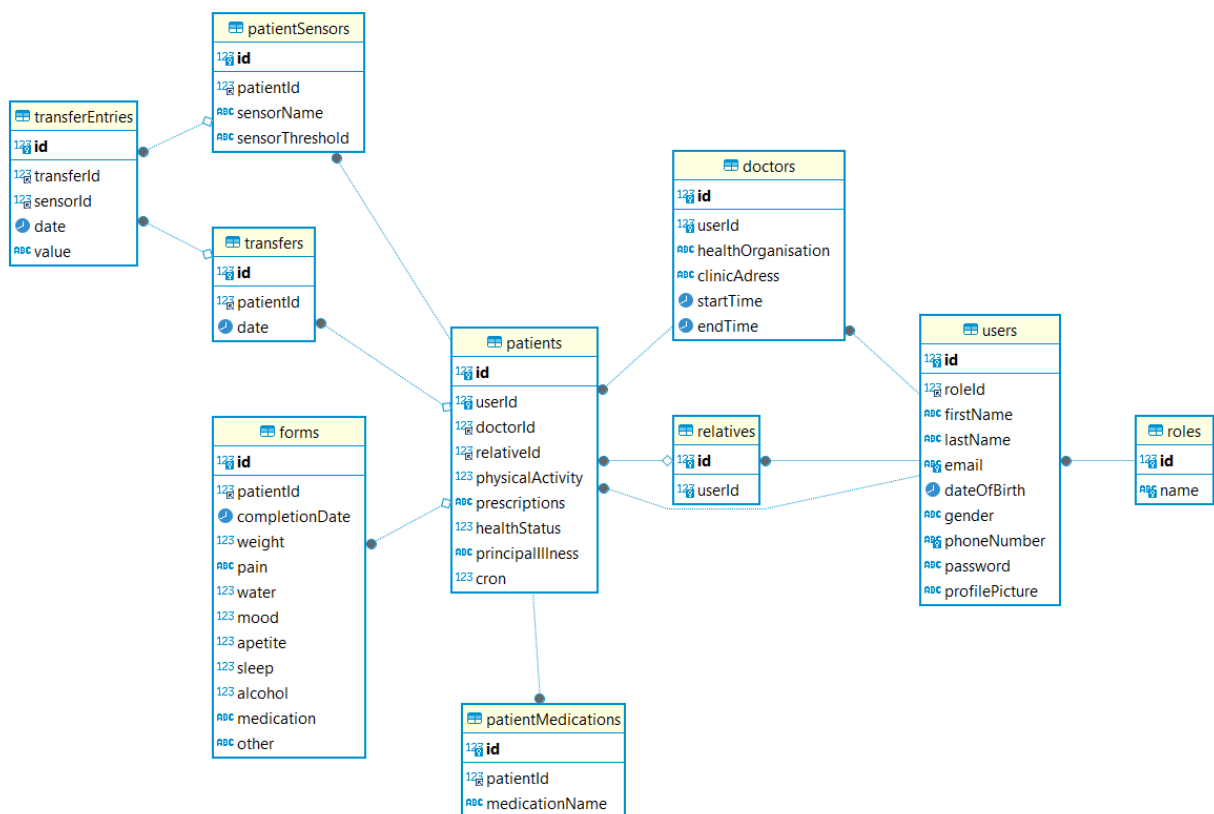
En ce qui concerne l'organisation du code, tout le code permettant de faire tourner l'application se trouve dans le répertoire "app" qui regroupe plusieurs sous répertoires :

- « config » pour tous les fichiers de configuration permettant de lancer la base de données et le serveur node.js en local
- « controllers » qui contient tous les fichiers qui gère les appels à la base de données à travers des fonctions.
- « middleware » qui permet de gérer l'authentification au site à l'aide d'un json web token et de fonctions pour s'inscrire et se connecter.
- « models » nous permet de créer les tables dans la base de données à l'aide du package sequelize.
- « routes » nous permet d'appeler les fonctions contenues dans « controllers » en fonction du type de requête HTTP et de la route demandée par le front-end
- « tests » contient tout les tests qui sont lancés dans le pipeline lors de la construction de l'application.

Voici notre architecture du projet sur Visual Studio Code :



La base de données est faite avec **MariaDB** qui assure la **sécurité** et assure une structure **organisée** de nos données. Ci-dessous vous pouvez observer la structure de notre base de données :



Stratégie pour collecter les données

Concernant la collecte des données, notre stratégie repose sur un **formulaire** que le patient remplit et transmet pour tenir informé le personnel médical ainsi que ses proches sur son état de santé de façon **régulière**. Les données qui sont collectées par les capteurs à intervalles **réguliers** seront transmises au serveur en fonction du besoin du médecin. Le patient définira un niveau d'importance de ses données qui configurera leur fréquence de transmission au médecin car il faut que le patient reste maître de ses données et sache où sont ses données personnelles et sensibles. Il nous paraît évident que si nous voulons étendre notre application au marché européen, il faut que l'application respecte le Règlement Général sur la Protection des Données.

Stratégie pour stocker les données côté serveur

Pour stocker toutes ces données nous utilisons **MariaDB**. Nous avons fait ce choix car MariaDB permet le **cryptage des données** et assure ainsi la **confidentialité**, la **sécurité** et l'**intégrité** des informations des utilisateurs. De plus, l'on y stocke des données extrêmement sensibles car il s'agit des données concernant la santé des patients qui peuvent être exploitées de façon malveillante.

De plus, MariaDB offre des **performances supérieures** et est plus adaptée que MySQL pour travailler avec des **données sous format JSON** comme ce que nous retournent nos capteurs.

MariaDB nous permettra aussi d'avoir une bonne **scalabilité horizontale** pour facilement être mis à l'échelle en fonction de la croissance du nombre d'utilisateurs.

Grâce à MariaDB, les données sont **interconnectées** et bénéficient d'une architecture relationnelle qui permet une organisation **logique, fluide** et **efficace**. Nos requêtes sont simplifiées et les données sont stockées de façon **cohérentes** pour être consultées par les utilisateurs et utilisées par notre back-end.

Stratégie pour échanger les données entre smartphone et serveur

Pour faciliter l'échange de données entre le smartphone et le serveur nous utilisons des requêtes **HTTP** pour les communications entre le front-end et le back-end.

L'avantage de cette approche par rapport à une connexion à l'aide d'un websocket est que les requêtes HTTP sont bien plus rapides et ont une meilleure scalabilité. Il n'y a pas besoin de créer un socket par utilisateur, ils appellent tous la même route pour obtenir le même résultat.

Ainsi, notre API est **intuitive** et permet une facilitation de l'intégration de multiples utilisateurs.

Notre stratégie permet une **bonne gestion des réponses et des erreurs** car les requêtes HTTP ont une variable de statut qui indiquent l'échec ou le succès de la requête. Ce statut est utilisé directement par les contrôleurs du back-end pour indiquer le **succès** ou l'**échec** de nos opérations. De plus, nos réponses et nos requêtes offrent une **flexibilité accrue** pour de **futurs développements** car pour ajouter une nouvelle fonctionnalité à l'application, il suffit de rajouter une route.

Grâce à notre stratégie nous pouvons concevoir une **application durable** qui nous permettra un ajout aisé de nouvelles fonctionnalités.

Stratégie de déploiement de l'application :

Notre stratégie de déploiement de MedTrackr repose sur l'utilisation de **Docker Swarm**. L'application est déployée avec l'aide de la commande :

docker stack deploy

Cette commande permettant de déployer plus **facilement** plusieurs instances du back-end et de la base de données.

Nous avons la possibilité d'utiliser des **indicateurs** tels que :

-h

afin de spécifier un serveur hôte et :

-c

dans le but de définir un fichier docker-compose.yml en tant que base du service. Cette commande est intégrée dans un pipeline Gitlab, offrant ainsi la flexibilité

d'une configuration avancée et personnalisée. Cela simplifie l'automatisation et la gestion du déploiement.

En effet, Gitlab Runner permet de déclencher et **automatiser** le processus de déploiement en réponse à des événements spécifiques
notre pipeline d'intégration comprend 5 étapes :

- test-node : va installer les dépendances du projet et lancer les tests
- build-docker : va créer un Dockerfile contenant le code du projet et va l'uploader sur un hub local (aminhelloworld/medtrackr-be)
- deploy-docker-dev : va setup le ssh et lancer un docker stack qui va déployer une instance du projet sur l'environnement de développement
- deploy-docker-prod : va setup le ssh et lancer un docker stack qui va déployer une instance du projet sur l'environnement de production

Cela nous permet de faire de **l'intégration** et du **déploiement continu**.

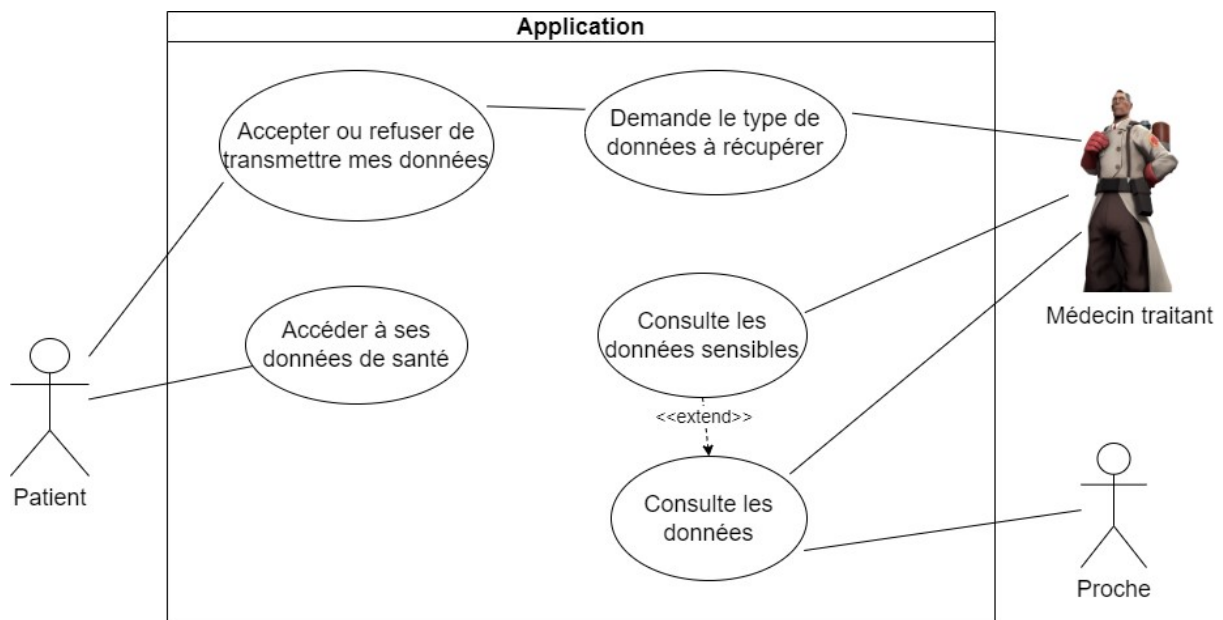
L'avantage de notre stratégie réside dans le fait que notre service est **facilement scalable** à l'aide de Docker Swarm qui permet une **haute scalabilité** et une **haute disponibilité** de l'application.

Pour résumer, notre approche stratégique de déploiement permet de rendre **robuste** et **flexible** notre application pour le **déploiement continu** de l'application. Cela nous donne l'opportunité de **centraliser**, de **personnaliser** et d'**automatiser** le déploiement de MedTrackr. Nous garantissons grâce à cela la **stabilité**, la **performance** et le **bon fonctionnement** de l'application.

Fonctionnalités

Cas d'usages

Voici les différents cas d'usage actuellement présent dans l'application :



Les cas d'utilisations liés au patient à savoir « Accepter ou refuser de transmettre mes données » et « Accéder à ses données de santé » sont 2 fonctionnalités primordiales que l'application doit implémenter car sans cela, MedTrackr ne peut pas être considéré comme une application de suivi de santé.

Pour ce qui est du médecin, il doit pouvoir demander le type de données à récupérer et consulter les données sensibles du patient. Sans cela, il n'y a aucune raison d'implémenter le rôle de médecin dans l'application.

Pour finir, les proches doivent pouvoir consulter les données de la personne qu'ils ont renseigné sans quoi notre application ne leur est pas utile.

Installation et configuration

Lancement local

Front-end

Pour lancer le front-end en local sur votre machine, il faut d'abord cloner le répertoire du projet :

```
git -c http.sslVerify=false clone https://iut-git.unice.fr/bl1116261/medtrackr-fe.git
```

Puis il vous suffit de taper les commandes suivantes pour lancer le front-end :

```
npm install  
npm start
```

Par défaut, le front-end tourne sur <http://localhost:5173/>

Back-end

Pour lancer le back-end en local sur votre machine, il faut d'abord cloner le répertoire du projet :

```
git -c http.sslVerify=false clone https://iut-git.unice.fr/ha110840/medtrackr-be.git
```

Ensuite il vous suffit de taper les commandes suivantes pour lancer le back-end :

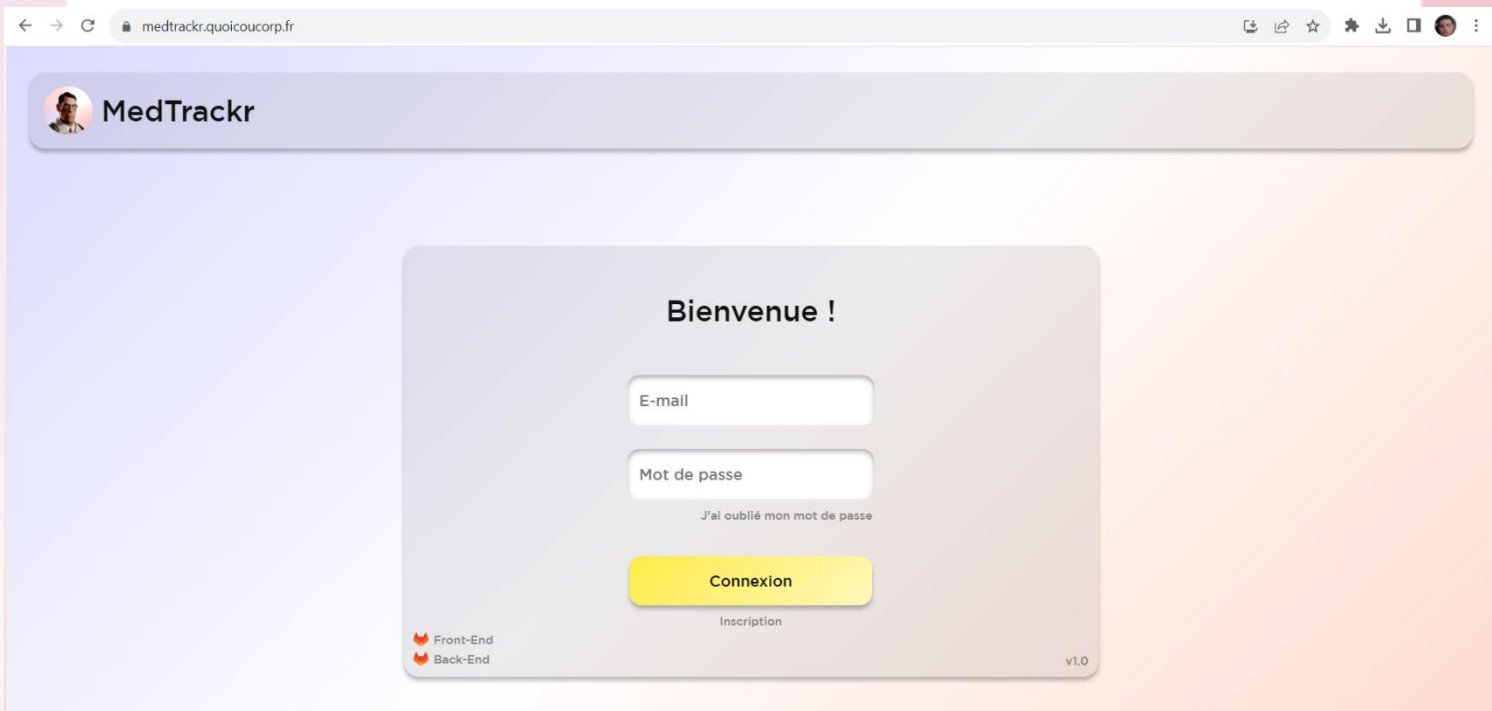
```
npm install  
node .\server.js
```

Par défaut, le serveur node.js expose le back-end sur <http://localhost:8081>

Connexion au cloud

Pour accéder au site web, il vous suffit d'aller sur

<http://medtrackr.quoicoucorp.fr> ou <https://medtrackr.quoicoucorp.fr>



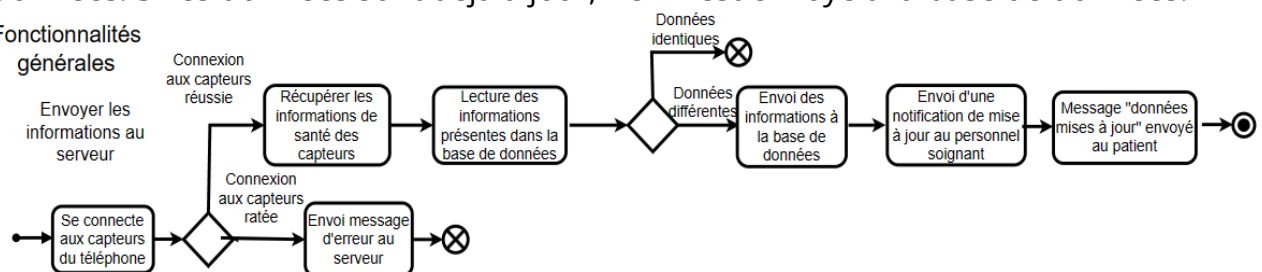
Fonctionnalités

Générales

Envoyer les informations au serveur

Pour réussir à envoyer les informations au serveur il faut que le capteur se connecte au téléphone puis qu'il mette à jour les informations de la base de données. Si les données sont déjà à jour, rien n'est envoyé à la base de données.

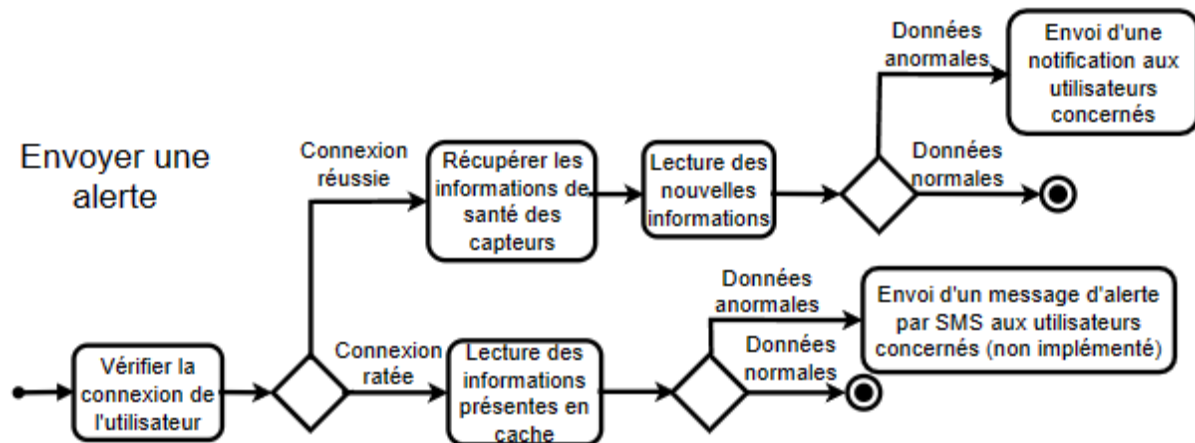
Fonctionnalités générales



Afin d'envoyer une alerte aux utilisateurs, on vérifie d'abord que le patient soit connecté à l'application puis on récupère les nouvelles informations des capteurs. Si elles sont anormales, une alerte est envoyée au médecin et aux proches du patient. Si les données sont normales, il n'y a pas d'alerte envoyée. Si le patient n'est pas connecté, on récupère les données en cache et on vérifie

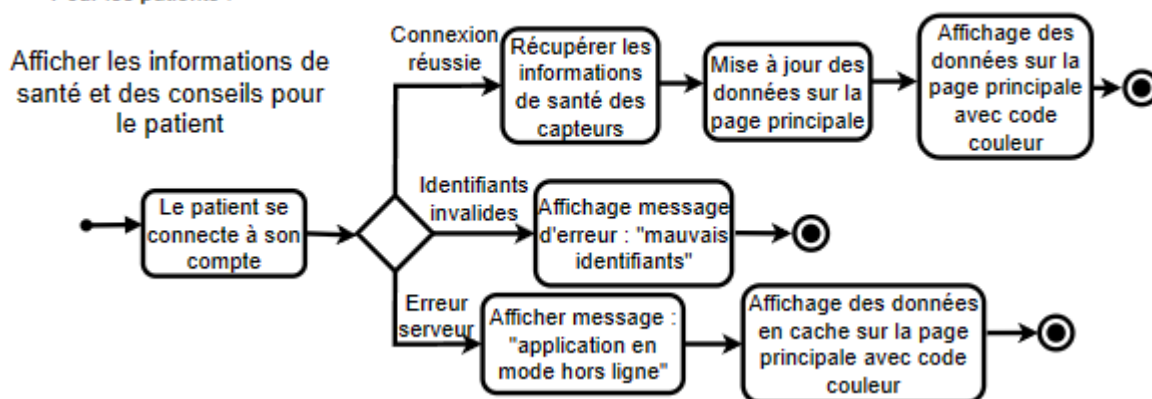
qu'elles ne soient pas anormales. Malheureusement, pour le moment si les données en cache sont anormales l'on ne peut pas encore envoyer de sms afin de prévenir les proches.

Voici le diagramme illustrant ce fonctionnement :

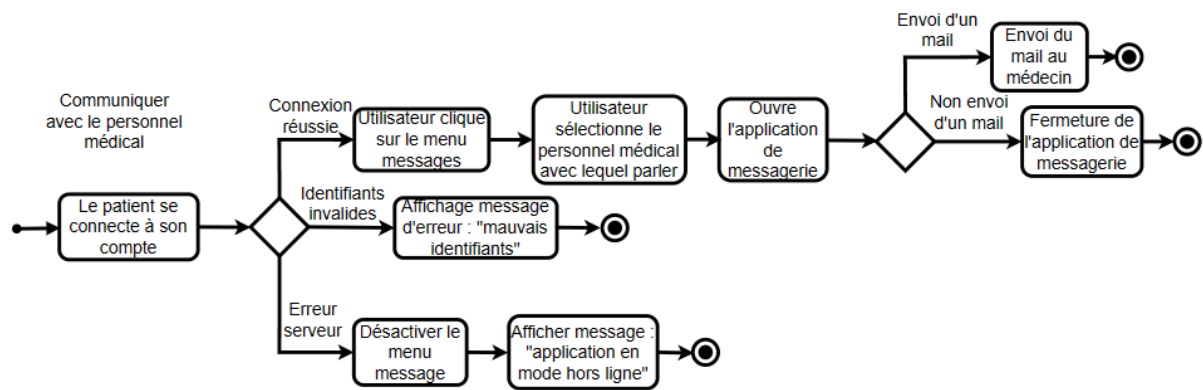


Pour afficher les données du patient il faut d'abord que le patient se connecte. Lors de la connexion du patient à son compte, si la connexion aboutit l'on récupère les données des capteurs et l'on met à jour les données. Ensuite en fonction de la normalité de la donnée l'on affiche les données avec un code couleur (rouge = critique; orange = à surveiller; vert = normale) comme l'illustre le diagramme ci-dessous.

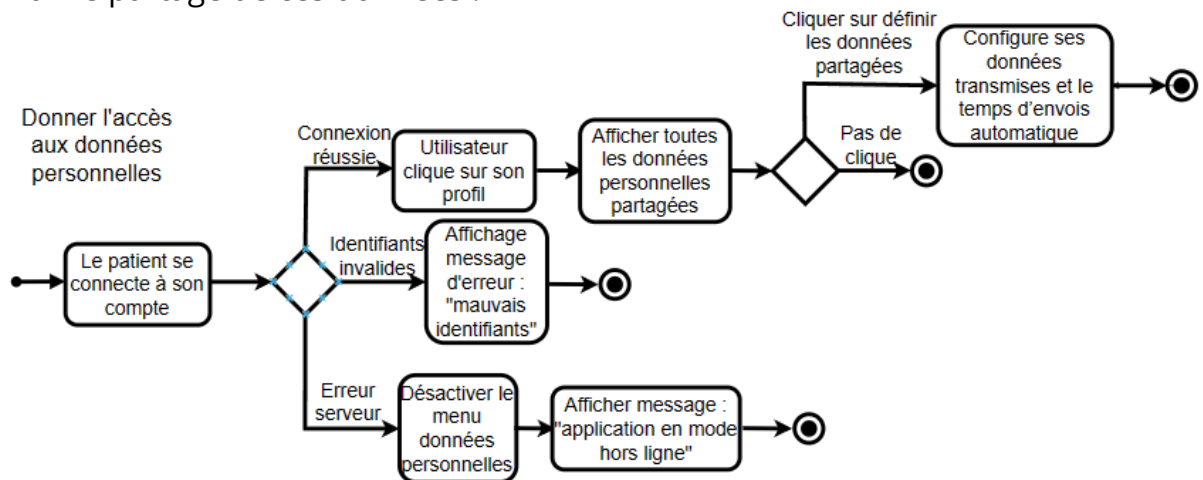
Pour les patients :



Pour communiquer avec le personnel médical il faut que le patient soit connecté et qu'il clique sur le menu message afin d'ouvrir son application de messagerie personnelle et envoie par la suite s'il le souhaite un mail à son médecin. Un diagramme illustre cette interaction ci-dessous.

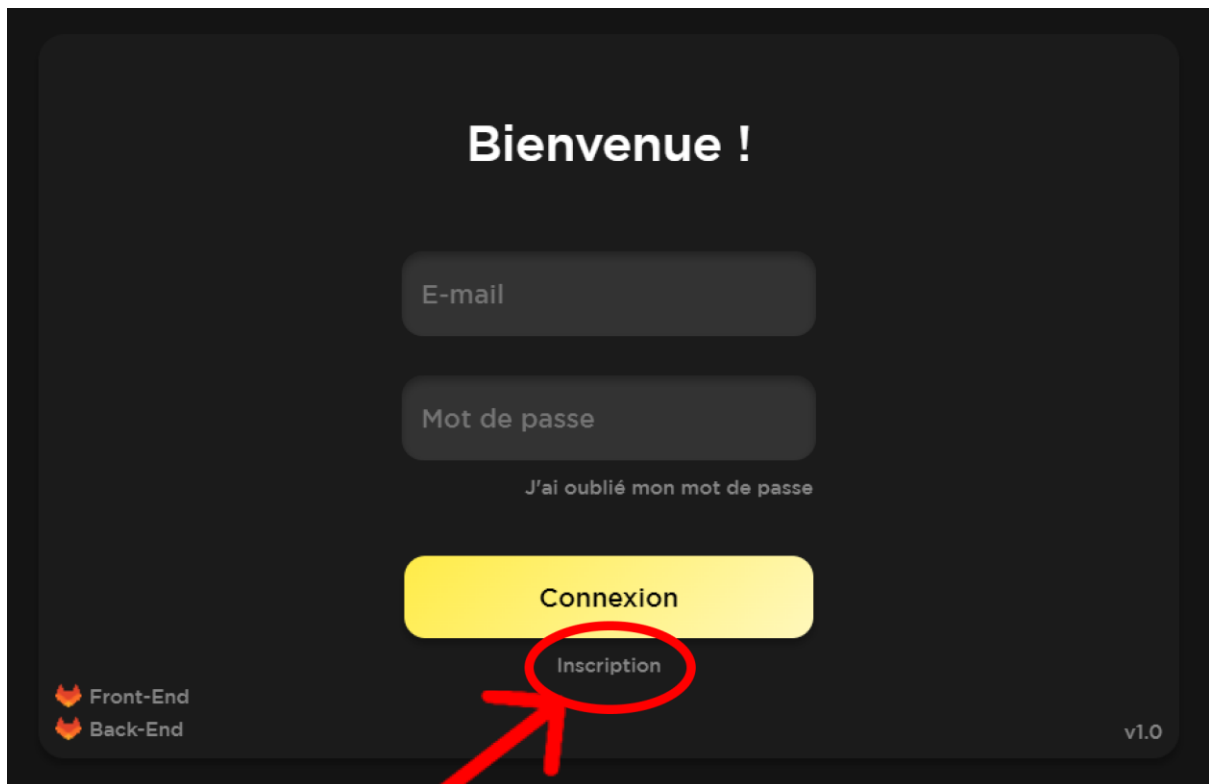


Une des fonctionnalités que l'on a choisi de ne pas inclure est la demande de données de la part du médecin au patient. L'on a par la suite opté pour une configuration d'envoi des données personnelles directement contrôlé par l'utilisateur lui-même pour lui laisser plus de liberté. Ici, lorsque le patient aurait reçu une demande de la part de son médecin il aurait eu le choix d'accepter ou non le partage de ces données :



Guide de l'utilisateur :

Une fois arrivée sur la page d'accueil il faut vous inscrire avec un nom d'utilisateur, un mot de passe et choisir quel type de compte vous souhaitez créer. Ainsi, vous arriverez sur la page d'accueil de votre compte.



Bienvenue !

E-mail

Mot de passe

[J'ai oublié mon mot de passe](#)

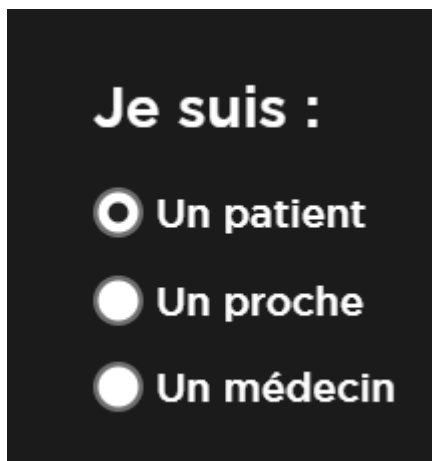
Connexion

Inscription

Front-End
Back-End

v1.0

Sélectionnez le bouton inscription



Je suis :

☒ Un patient

☐ Un proche

☐ Un médecin

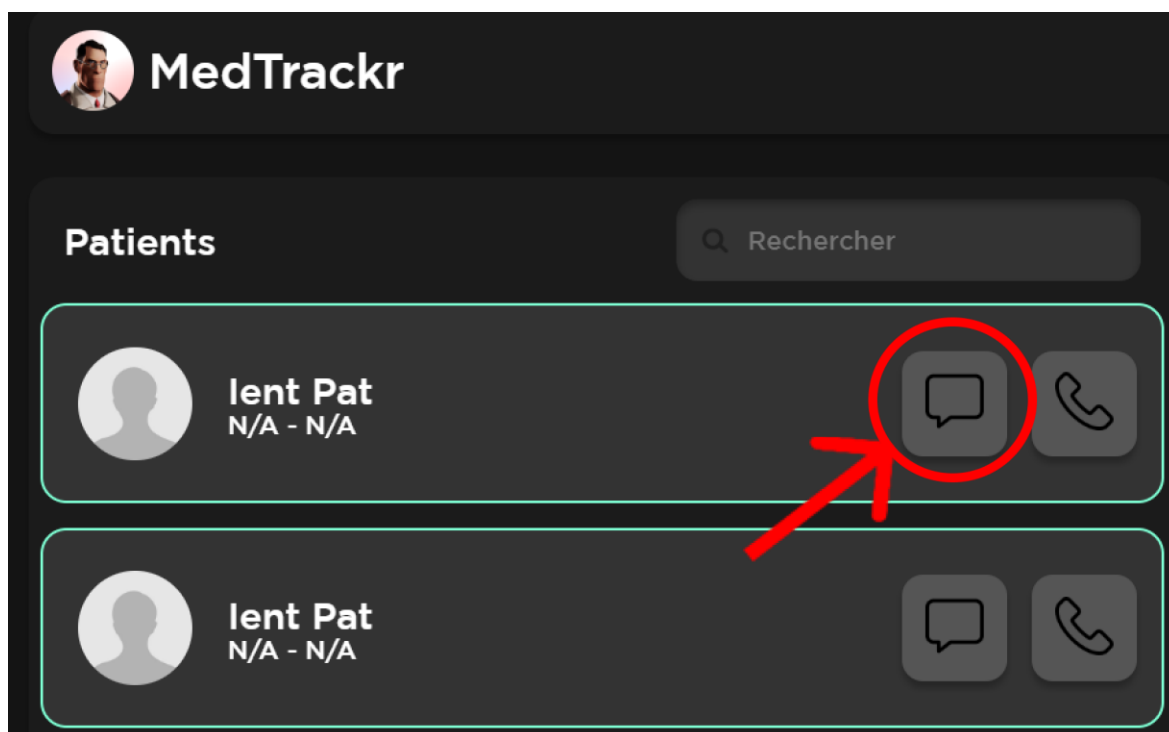
Sélectionnez ensuite votre type de profil

Inscription

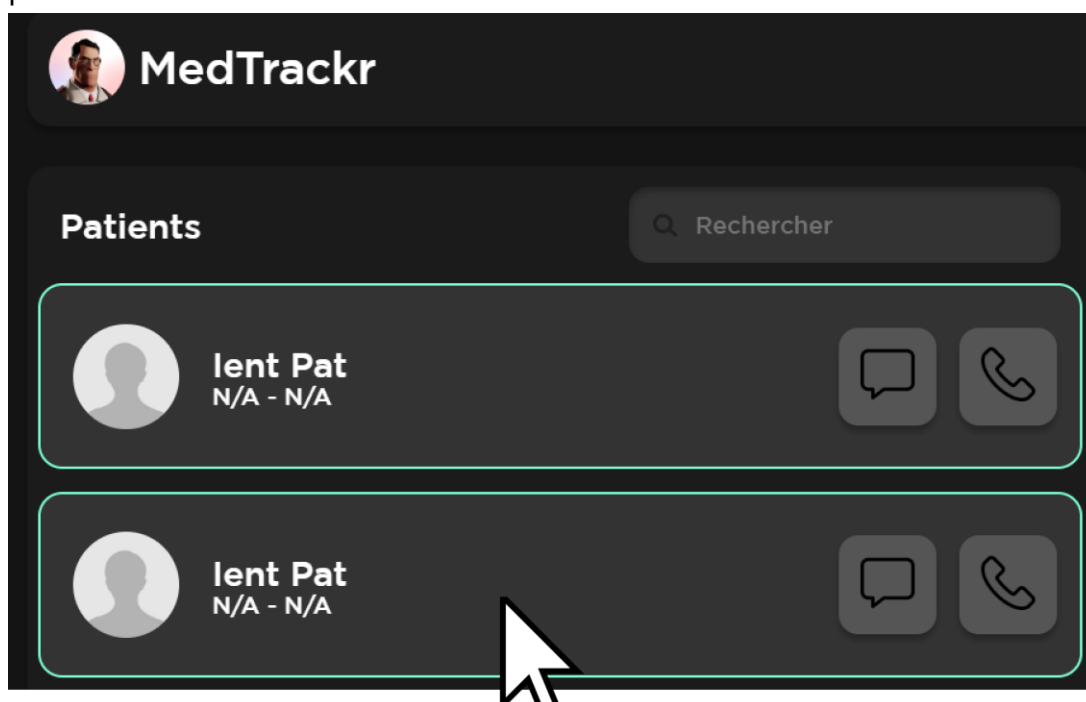
Enfin, inscrivez vous en donnant vos informations

Interface médecin

Lorsque vous êtes sur votre page d'accueil vous pouvez voir à gauche votre liste de patients avec la possibilité de les joindre directement via votre application de messagerie en ligne afin d'envoyer un mail à votre patient.

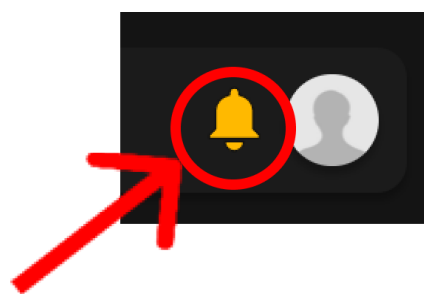


Mais le plus important reste la possibilité de cliquer sur n'importe quel patient pour visionner ces données.



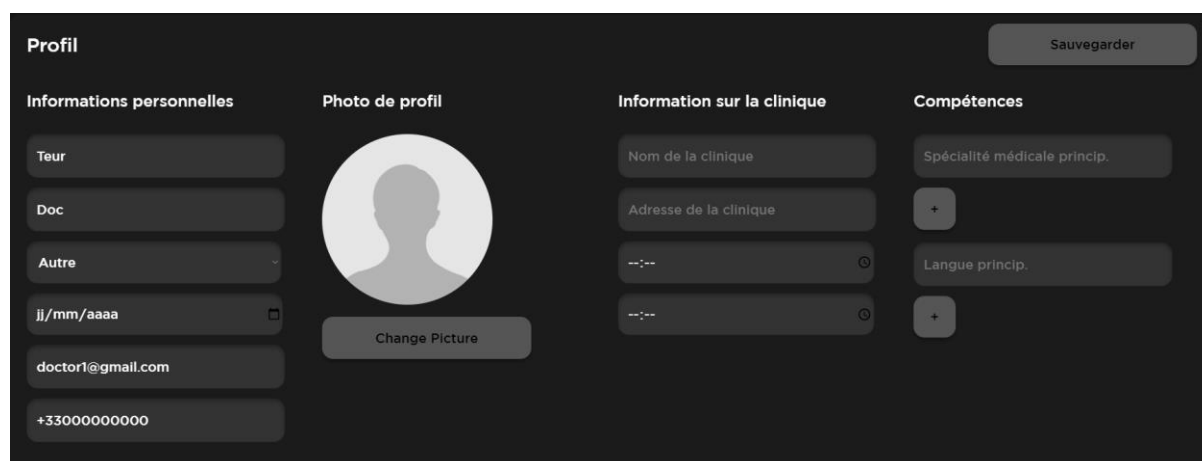
Pour qu'un de vos patients apparaisse sur ce menu il faut que lors de son inscription ou sur sa page de profil il renseigne l'adresse email avec laquelle vous avez créé votre compte dans le champ email médecin. Ou si vous le souhaitez, vous pouvez envoyer un lien d'invitation en appuyant sur inviter un patient. Vous aurez dans votre presse-papier le lien prêt à être envoyé.

Sur la partie droite de votre interface vous avez accès à vos alertes récentes concernant vos patients. Les patients ayant eu une alerte récente seront affichés en rouge pour pouvoir les discerner plus rapidement.



Vous aurez la possibilité de regarder vos notifications en cliquant en haut à droite de votre écran. La fonctionnalité n'étant entièrement implémenter coté back end les notification médecin ne fonctionne pas entièrement.

Et juste à côté de vos notifications se trouve l'accès à votre onglet profil. Depuis votre profil vous pouvez personnaliser vos données ou vous déconnecter.



Interface patient

Depuis votre page d'accueil vous avez instantanément accès à un formulaire que vous pouvez remplir afin de tenir informer votre médecin sur votre état de santé

ainsi que vos proches de façon régulière. Grâce à celui-ci, vous pouvez communiquer directement avec votre médecin.


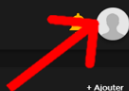
The screenshot shows the MedTrackr app interface. On the left, the 'Formulaire' (Form) section is active, indicated by a red arrow. It contains several input fields: 'Poids', 'Douleurs ressenties', 'Verres d'eau', 'Humeur' (with a dropdown menu showing 'Bonne'), 'Appétit' (with a dropdown menu showing 'Bon'), 'Qualité du sommeil' (with a dropdown menu showing 'Bonne'), and 'Consommation d'alcool' (with a dropdown menu showing 'Non'). There is also a section for 'Médicaments pris' (Medicines taken) with checkboxes for 'Medication 1', 'Medication 2', and 'Medication 3', and a text area for 'Autres remarques' (Other remarks). An 'Envoyer' (Send) button is at the bottom right of the form. On the right, the 'Capteurs connectés' (Connected sensors) section displays four sensor cards: 'Fréquence cardiaque' (Heart rate) at 62 BPM, 'Pression artérielle' (Blood pressure) at 102/64 mmHg, 'Température corporelle' (Body temperature) at 36 °C, and 'Glycémie' (Blood sugar) at 72 mg/dL. A 'Historique' (History) tab is visible between the form and the sensors. At the top right, there is a notification bell icon and a user profile icon.

Sur la partie droite de votre interface, les données que transmettent vos capteurs.

This screenshot is identical to the one above, showing the MedTrackr app interface. However, a red arrow points to the 'Capteurs connectés' (Connected sensors) section on the right. This section displays four sensor cards: 'Fréquence cardiaque' (Heart rate) at 62 BPM, 'Pression artérielle' (Blood pressure) at 102/64 mmHg, 'Température corporelle' (Body temperature) at 36 °C, and 'Glycémie' (Blood sugar) at 72 mg/dL. The 'Formulaire' (Form) section on the left is also visible, with the same input fields and 'Envoyer' button. The 'Historique' (History) tab is positioned between the two main sections. The top right corner features a notification bell icon and a user profile icon.

En haut à droite de votre écran vous pouvez accéder à vos notifications. Celle-ci vous informe sur les alertes concernant vos données de santé, des rappels pour ne pas oublier d'envoyer votre formulaire.

Un peu plus à droite vous pouvez accéder à votre profil. Depuis votre profil vous pouvez personnaliser vos informations personnelles et définir quelles informations vous souhaitez communiquer avec votre médecin et avec votre famille. Une fois les données choisies vous devrez définir la fréquence à laquelle vous souhaitez les transmettre. Vous êtes le propriétaire absolu de vos données.


MedTrackr


Formulaire

Poids

Douleurs ressenties

Verres d'eau

Humeur

Appétit

Qualité du sommeil

Consommation d'alcool

Médicaments pris :

■ Medication 1

■ Medication 2

■ Medication 3


Autres remarques

Envoyer

Historique


Capteurs connectés

+ Ajouter




Fréquence cardiaque

62 BPM




Pression artérielle

102/64 mmHg




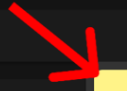


Température corporelle

36 °C



Glycémie

72 mg/dL


MedTrackr




Formulaire

Poids

Douleurs ressenties

Verres d'eau

Humeur

Appétit

Qualité du sommeil

Consommation d'alcool

Médicaments pris :

■ Medication 1

■ Medication 2


■ Medication 3

Autres remarques

Envoyer


Historique

Capteurs connectés




Fréquence cardiaque

81 BPM




Pression artérielle

null/null mmHg



Température corporelle

37.07 °C



Glycémie

86 mg/dL

Profil

Déconnexion

MedTrackr

Profil

Informations personnelles

Test

Patient

Autre

12/01/2023

patientTest@gmail.com

01 23 45 67 97

Photo de profil

Change Picture

Fréquence d'envoi

Régulière

Données de santé

Pathologie princip.

Choisir quelles données sont partagées...

Activité Physique

Régulière

Information sur le médecin

Test
Doctor
doctorTest@gmail.com
01 23 45 67 89

Information sur le proche

Test
Proche
procheTest@gmail.com
01 23 45 67 88

Sauvegarder

Si vous êtes le proche d'un patient :

Depuis votre page d'accueil vous pouvez accéder à la liste de vos proches sur la partie gauche de votre interface.

MedTrackr

Profil

Informations personnelles

Doe

John

Autre

jj/mm/aaaa

patient@gmail.com

06 06 06 06 06

Comment contribuer à notre projet

Nous serions heureux que la communauté nous aide à participer au développement de notre application pour le bien de la santé de tous. Si vous souhaitez contribuer à MedTrackr, suivez ces étapes :

Créez une branche pour votre fonctionnalité :

```
git -c http.sslVerify=false checkout -b Nouvelle_fonctionnalité
```

Vous pouvez ensuite développer une fonctionnalité.

Envoyez vos changements :

```
git add .  
git commit -m "Ma fonctionnalité"  
git -c http.sslVerify=false push origin Nouvelle_fonctionnalité
```

Après une analyse de la pertinence et du bon fonctionnement de votre code, nous l'intégrerons par la suite à notre projet.