ECOLE
POLYTECHNIQUE
DE BRUXELLES

ULB

UNIVERSITÉ LIBRE DE BRUXELLES

Conception et réalisation d'une solution permettant la centralisation et la visualisation de données de monitoring provenant de centres de santé en RDC

Mémoire présenté en vue de l'obtention du diplôme
d'Ingénieur Civil Informaticien à finalité Web

Martin Delobbe

Directeur
Professeur Antoine Nonclercq

Co-Promoteur
Professeur Jean-Michel Dricot

Superviseur
Loïc Vaes

Service
BEAMS

Année académique
2017 - 2018

Remerciements

J'adresse mes remerciements aux personnes qui m'ont soutenu dans la réalisation de ce mémoire.

En premier lieu, je tiens à remercier mon superviseur, Loïc Vaes, responsable de projet chez AEDES, qui m'a toujours donné un avis très critique et dont la disponibilité et l'attention ont toujours été particulièrement présentes.

Je remercie également mon directeur et mon co-promoteur de mémoire, les professeurs Antoine Nonclercq et Jean-Michel Dricot qui m'ont permis de satisfaire aux attentes académiques et sans qui ce mémoire n'aurait pas pu voir le jour.

Je tiens aussi à remercier la cellule Codepo qui m'a soutenu et m'a permis de combler les démarches administratives pour la réalisation de la mission à Kinshasa.

Mes remerciements s'étendent de même à la société MaisOrdi qui m'a accueilli lors de mon arrivée à Kinshasa, qui m'a permis d'avoir un environnement propice au développement de ce mémoire et qui m'a fourni tous les feedbacks nécessaires à sa finalisation.

Enfin, je remercie mes parents, Eric Delobbe et Marinette Cloos, qui ont toujours été présents et dont les conseils m'ont permis de passer à travers les doutes qui m'ont parcourus dans les moments incertains.

Résumé

Le système CERHIS, développé par AEDES, permet d'informatiser les centres de santé dans les pays où l'accès à l'électricité est incertain et où la tenue des registres papier devient un calvaire. CERHIS est composé de multiples éléments hardware qui demandent à être surveillés. Cette surveillance est réalisée par un système de monitoring qui génère des SMS en cas d'alerte ou de rapport quotidien. Ces SMS sont ensuite acheminés sur une plateforme de centralisation et de visualisation. Cette plateforme, qui est le centre de ce mémoire, a été réalisée sous la forme d'une application web, comprenant plusieurs fonctionnalités dont une gestion des abonnements qui permet de redistribuer les SMS de monitoring aux personnes abonnées aux centres de santé. Les perspectives quant à l'utilisation de cette application web s'inscrivent autant dans une optique technique, puisque elle permettra de surveiller l'ensemble des centres et d'identifier les problèmes rapidement, que dans une optique plus large, puisqu'elle pourra servir de façade pour ce projet de coopération au développement en présentant des données collectées en temps réel sur les différents centres.

Mots-clés :

CERHIS - Monitoring - SMS - Application Web - Visualisation - Coopération au développement

Table des matières

1 Introduction **1**

1.1 Contexte et objectifs du mémoire 1

1.2 AEDES [1] 2

1.2.1 CERHIS [2] 2

1.3 Codepo [3] 3

1.3.1 La Collaboration AEDES-Codepo 3

1.3.1.1 2011-2012 4

1.3.1.2 2013-2014 4

1.3.1.3 2015-2016 4

1.3.1.4 2016-2017 5

1.3.1.5 2017-2018 6

1.4 Structure et méthodologie 7

1.5 Notations 8

2 Etat de l'art **9**

2.1 Les systèmes de monitoring par SMS 9

2.1.1 Introduction 9

2.1.2 Les types d'architectures 9

2.1.3 Les contraintes et avantages du SMS 11

2.2 La visualisation de données 12

2.2.1 Introduction 12

2.2.2 Outils existants 13

2.2.2.1 Les outils de Business Intelligence 13

2.2.2.2 Les outils de Dashboard 14

2.2.2.3 Les bibliothèques issues de D3 . . . 14
2.3 La coopération au développement . . . 15

3 Cahier des charges **17**
3.1 Introduction . . . 17
3.2 Objectifs . . . 18
3.2.1 Hiérarchisation et organisation des objectifs . . . 20
3.3 Description et limitation du périmètre . . . 20
3.3.1 Système de visualisation de données . . . 20
3.3.2 Serveur SMS . . . 21
3.3.3 Application technicien . . . 22
3.3.4 Application de monitoring . . . 22
3.4 Planning . . . 23
3.5 Résumé . . . 23

4 Système de centralisation et de visualisation de données **24**
4.1 Description technique . . . 24
4.1.1 Type de solution . . . 24
4.1.2 Stack utilisé . . . 25
4.1.2.1 Le Frontend . . . 26
4.1.2.2 Le Backend . . . 27
4.1.2.3 La Base de données . . . 27
4.2 Description fonctionnelle . . . 28
4.2.1 L'authentification . . . 28
4.2.2 La gestion des utilisateurs . . . 30
4.2.3 Le système d'abonnements . . . 31
4.2.4 La visualisation des données . . . 31
4.2.5 Le système de permissions . . . 32
4.2.6 Le traitement des SMS de monitoring . . . 33
4.3 Implémentation . . . 33
4.3.1 Architecture . . . 33
4.3.2 Bibliothèques utilisées . . . 35
4.3.3 Mise en production . . . 36

5 Application de monitoring **39**

5.1 Changements techniques . . . 39

5.1.1 Design . . . 39

5.1.2 Base de données . . . 40

5.1.3 Fonctions en arrière-plan . . . 41

5.2 Implémentation . . . 41

5.2.1 Architecture . . . 41

5.2.2 Mise en production . . . 42

6 Implémentation sur le terrain **44**

6.1 Organisation . . . 44

6.2 Application de monitoring . . . 45

6.3 Application web . . . 45

6.4 Résumé . . . 46

7 Résultats **47**

7.1 Bilan . . . 47

7.2 Chiffres . . . 47

7.2.1 Application Android . . . 47

7.2.2 Application Web . . . 49

7.3 Erreurs commises . . . 51

7.4 Résumé . . . 52

8 Conclusion **54**

8.1 Améliorations possibles et perspectives . . . 55

A Modes d'emploi **58**

A.1 Application de monitoring . . . 58

A.2 Application de visualisation . . . 59

B Mise en production **63**

B.1 Serveur Amazon Web Service 63

B.1.1 Création d'une instance EC2 63

B.1.2 Connexion et mise à jour 64

B.1.3 Installation du serveur Nginx 65

B.1.4 Mise en place de Node.js 65

B.1.5 Configuration de Nginx 65

B.2 Configuration de Couchbase 66

B.3 Serveur SMS Twilio 66

B.4 Plugin Fabric 67

C Documents et programme de la mission **69**

C.1 Termes et références 69

C.2 Gantt de la mission 73

Chapitre 1

Introduction

1.1 Contexte et objectifs du mémoire

La République démocratique du Congo (RDC) est un pays de 2.345.000 km carrés et compte près de 64 millions d'habitants. Membre de l'Organisation Mondiale de la Santé (OMS), la RDC a souscrit, en 1980, à la charte africaine de développement sanitaire par l'adoption de la stratégie des Soins de Santé Primaires visant la Santé pour Tous à l'an 2000. La bonne gestion des structures de santé nécessaires à la mise en œuvre de ces objectifs reste difficile, et la production de données sanitaires de qualité indispensables pour prendre des décisions efficaces dans les politiques de santé est également problématique.

Afin de répondre à ces défis, les responsables d'une structure de santé pilote, le Centre Hospitalier Mutualiste de Kinshasa (CHMK), ont décidé d'exploiter des outils informatiques qui ont été initiés dans des projets de la CODEPO et développés par la société AEDES. Le CHMK est un centre de santé de petite taille qui offre des soins aux membres d'une mutualité locale, grâce à la présence d'un médecin, d'un pharmacien, d'un laborantin, et de quelques infirmières. Il est localisé dans la commune de KasaVubu, au sud du centre-ville de Kinshasa. Le système d'information hospitalier CERHIS, utilisant des tablettes tactiles déployées au sein d'un réseau Wi-Fi local, est en phase pilote au sein de la structure de santé.

La solution logicielle déjà développée n'est correctement exploitable que si elle est installée avec du matériel de qualité, protégé, et bien entretenu. À cette fin, les étudiants du projet CODEPO/AEDES 2015-2016 ont conçu une armoire de rangement, chargement et protection des tablettes tactiles. Des éléments modulables contiennent des batteries, les tablettes, ainsi qu'un serveur local et un routeur. Le projet-pilote du CHMK étant appelé à s'étendre prochainement à d'autres structures de santé en République Démocratique du Congo, les étudiants du projet CODEPO/AEDES 2016-2017 ont développé une solution de monitoring permettant de prévenir les techniciens en cas de panne du système. Cette solution permet de ne pas charger le personnel de santé d'un travail de surveillance technique qui ne fait pas partie de ses priorités ni nécessairement

de leurs compétences et d'agir rapidement en cas de problème.

Le projet CERHIS visant à s'étendre à toute la région et même à tout le pays, l'envoi de SMS aux techniciens depuis un grand nombre de centres peut rapidement s'apparenter à du spam. En outre, AEDES souhaitant rester informé à propos du fonctionnement du système, il faudrait prévoir des coûts pour l'envoi de SMS à l'étranger. Cette solution n'étant pas flexible, il faut mettre au point une solution permettant de centraliser tous ces SMS. Une fois ces SMS centralisés, une solution permettant de visualiser l'information contenue dans les SMS doit être mise sur pied. En outre, un système d'abonnement permettant de ne recevoir les SMS que de certains centres hospitaliers est à prévoir. La réalisation d'une application web est proposée comme piste de travail.

1.2 AEDES [1]

Créée en 1985, l'Agence Européeen pour le Développement et la Santé est une entreprise de consultance spécialisée dans le domaine de la santé publique. De nombreuses thématiques du secteur de la santé sont couvertes par son expertise. Depuis 1985 jusqu'à aujourd'hui AEDES est progressivement devenue un acteur majeur en santé publique et ce aussi bien au niveau européen que dans les pays à faibles et moyens revenus.

La mission d'AEDES est de contribuer à l'amélioration de la qualité et de l'accès aux soins de santé partout dans le monde. Pour cela, AEDES favorise le partage et la transmission des connaissances, en mettant à disposition des moyens humains expérimentés et en prestant des services de conseil et de gestion dans le domaine de la santé et de la sécurité alimentaire.

AEDES intervient dans les pays industrialisés, intermédiaires ou en développement, pour des acteurs privés et publics. AEDES s'efforce de promouvoir les principes du développement durable dans ses trois dimensions : économique, sociale et environnementale.

1.2.1 CERHIS [2]

CERHIS est un système intégré qui comprend :

- Un logiciel à interface tactile pour le suivi des soins des patients et la gestion des structures de santé.
- La fourniture de matériel adapté aux conditions d'utilisation : tablettes tactiles et mise en réseau, hébergés dans une armoire de chargement et de protection spécialement conçue à cette fin.
- La gestion de l'alimentation par une combinaison d'énergie solaire, électricité du réseau, et/ou d'un générateur.
- Des services d'installation, de configuration aux besoins de chaque structure et de formations des utilisateurs

- Un partenariat local pour l'entretien régulier du système et un suivi technique de proximité à coûts mutualisés.

Les objectifs de ce système permettent de répondre à une série de problématiques : Les besoins au niveau périphérique c'est-à-dire la prise de décision, les soins ou encore la gestion - les besoins en information aux niveaux intermédiaires - la taille du pays et les conditions variables dans les structures de santé - les problèmes de communication internet - les problèmes d'alimentation en électricité - les moyens financiers limités - la disponibilité et rotation du personnel ainsi que la connaissance limitée du personnel des systèmes d'exploitation Windows ou Linux.

1.3 Codepo [3]

Codepo est la cellule de coopération au développement de l'école polytechnique de Bruxelles. Gérée l'ingénieur Cédric Boey (conseillé pédagogique) et les professeurs Antoine Nonclercq et Benoît Haut de l'ULB, elle a pour mission première de proposer aux étudiants de Master une première expérience dans le domaine de la coopération au développement. Ses objectifs sont multiples :

- **Pédagogique** : Cette expérience mettant en jeu des contraintes tant techniques que culturelles rencontrées à l'étranger, elle participe à la formation des jeunes ingénieurs et à leur prise de conscience face aux défis de projets concrets.
- **La recherche scientifique et technique** : Proposer des solutions innovantes est un enjeu clé et ces dernières sont diffusées dans des journaux scientifiques ou lors de conférences.
- **L'éducation au développement** : En offrant une ouverture aux étudiants de l'Ecole sur la coopération au développement et les principales problématiques qui y sont liées afin d'en faire des citoyens et ingénieurs du monde.

Chaque année, ce sont ainsi environ 20 étudiants qui, dans le cadre d'un projet ou de leur mémoire, partent à l'étranger afin de s'impliquer dans le développement de procédés de conservation des aliments, de la télémédecine, des énergies renouvelables et de valorisation de la biodiversité des pays émergents.

1.3.1 La Collaboration AEDES-Codepo

La collaboration entre AEDES et la Codepo est un travail reparti sur plusieurs années et dont l'origine trouve sa source dans la solution d'informatisation des centres de santé développée dans le mémoire de Loïc Vaes. Le système CERHIS décrit dans la section 1.2.1 n'est autre que le fruit de cette collaboration et du travail fait en amont par l'entreprise elle-même.

1.3.1.1 2011-2012

La première collaboration entre L'AEDES et la CODEPO a débuté durant l'année académique 2011-2012. L'objectif de ce projet était d'améliorer la prise en charge et le suivi des patients hospitalisés dans un hôpital de zone de santé en introduisant la notion de fiches d'hospitalisation uniques (en abordant déjà l'unicité des dossiers patients, facilitant sa mise en place lors d'une étape future). L'introduction d'un outil informatique de gestion de données patients au sein de l'hôpital permet, à terme, de rendre les dossiers accessibles au travers de tout l'hôpital ainsi qu'à la division provinciale. Les principaux objectifs étaient les suivants :

- Informatisation des registres au sein de chaque service
- Synchronisation des bases de données par voie USB
- Synchronisation des bases de données via un réseau local

1.3.1.2 2013-2014

Durant l'année académique 2013-2014, l'équipe d'étudiants avait pour objectif d'étendre le projet de l'année précédente en développant une application web codée avec Django, un framework Python. En effet, l'administration de l'hôpital était transcrite sur format papier. Ce système étant peu efficace, il causait de nombreuses erreurs et le suivi de l'administrateur était difficile et laborieux. Le but de ce projet était de pouvoir convertir toutes les tâches administratives réalisées avec un format papier en version digitale, c'est-à-dire :

- Les prescriptions
- La gestion des stocks
- Les factures
- Les fiches patients

Actuellement, l'application originelle n'existe plus. L'AEDES a développé une application ayant exactement le même objectif, mais codée dans un autre langage de programmation, Java, développé dans l'IDE Android Studio.

1.3.1.3 2015-2016

La collaboration entre AEDES et la CODEPO a continué durant l'année académique 2015-2016. Ce deuxième projet consistait à modéliser un système basé sur deux axes :

- Structure mécanique et stockage des composants du système
- Alimentation du système

La partie mécanique devait pouvoir accueillir et recharger 10 tablettes dans un environnement robuste, modulable et assurant une sécurité face aux vols.

FIGURE 1.1 – Application CERHIS

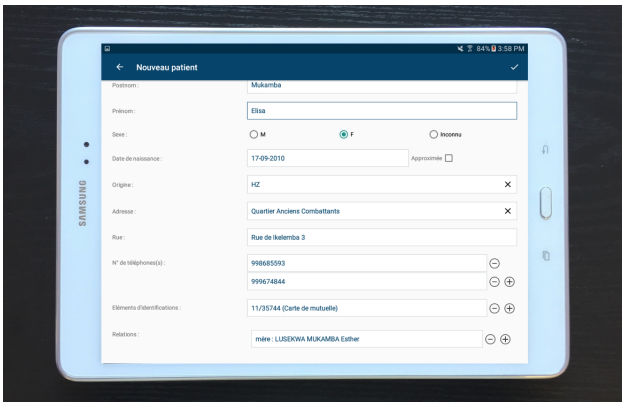


FIGURE 1.2 – Application CERHIS

Pour finir, la partie alimentation avait pour objectif de protéger les composants électroniques des surtensions, et des transitoires du réseau électrique et de dimensionner une batterie, et des panneaux solaires en cas de délestage du réseau électrique. Ces délestages sont en effet assez fréquents à Kinshasa.

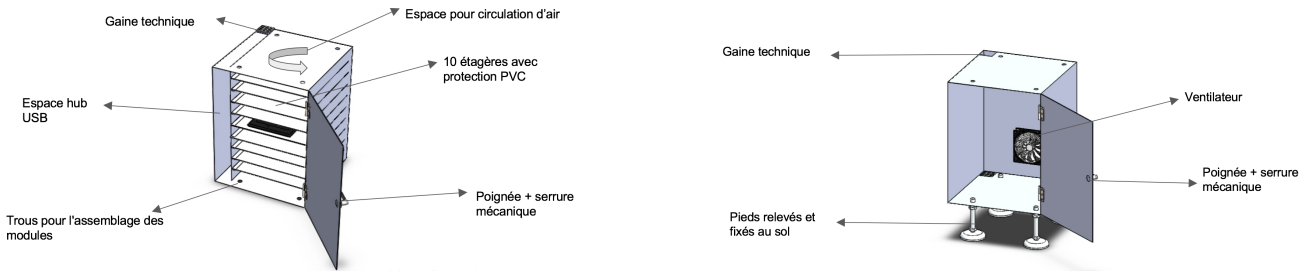


FIGURE 1.3 – Armoire médicale

L'installation du dispositif a été réalisée lors du mois de juillet 2016. Le résultat du projet peut être observé sur les photos 1.4.

1.3.1.4 2016-2017

Le projet de l'année 2016-2017 avait pour but de développer une solution de monitoring du système d'information hospitalier du CHMK. Cette solution devait pouvoir envoyer un rapport quotidien

FIGURE 1.4 – Armoire médicale et panneaux solaires du projet 2015-2016

et des alertes de pannes par SMS.

Les objectifs étaient de :

- Surveiller des éléments hardware et software du système hospitalier
- Transmettre des rapports quotidiens d'activité par SMS
- Transmettre des avis de détection de panne par SMS
- Fournir une résistance à la chaleur, à l'humidité et à la poussière
- Dimensionner la batterie du dispositif à installer
- Définition du codage optimal de l'information dans les SMS
- Avoir une capacité de production/assemblage future en RDC

1.3.1.5 2017-2018

Le projet CODEPO 2017-2018 avait pour objectif de permettre une installation des projets des années précédentes, le système informatique CERHIS avec armoire de chargement solaire et système de monitoring, dans de nouveaux centres de santé en fournissant des données spécifiques à chacun de ces centres pour adapter le système informatique selon les coûts et besoins.

Les données d'output à fournir étaient les suivantes :

- Disponibilité et qualité du réseau électrique
- Opérateur téléphonique et internet optimaux
- Consommation et besoins en énergie solaire
- Estimation de la durée de vie des composants du système informatique
- Caractéristique des batteries à mettre en place

1.4 Structure et méthodologie

La particularité de ce mémoire réside dans sa grande composante technique ainsi que dans son orientation coopération au développement. Le développement d'un tel système peut se réaliser par le biais d'un grand nombre de technologies et c'est la raison pour laquelle les choix effectués sont primordiaux et explicités de manière récurrente. Afin de garder une homogénéité dans la présentation du problème et de ses solutions, les différents chapitres comprennent tout d'abord une présentation générale du sujet. Par la suite, ce sujet est traité plus en détail dans les sections qui suivent selon des caractéristiques qui lui sont propres et ce, de manière à atteindre un niveau de granularité permettant de prendre pleinement conscience des enjeux techniques, économiques ou sociaux.

Afin d'avoir un aperçu de différents sujets pris en charge par ce mémoire, plusieurs architectures de monitoring par SMS ainsi que différentes solutions de visualisation de données sont présentées dans le chapitre 2. Une brève introduction sur la coopération au développement, son histoire ainsi que la situation actuelle et plus particulièrement en Belgique est également donnée dans ce même chapitre.

Partant des connaissances acquises dans le chapitre 2, il a été possible de mettre sur pied un cahier des charges. Le partenariat avec AEDES rendant ce mémoire très concret, il a été important de délimiter le périmètre du projet et de prendre en compte les différents critères vitaux pour la pérennité du projet. Ainsi, une présentation des différentes fonctionnalités à implémenter est faite dans le chapitre 3 ce qui permet de fixer le mémoire à la fois dans l'espace, le temps et de prendre conscience des ressources à investir pour son bon déroulement.

Le chapitre 4 fait état de la partie centrale de ce mémoire, à savoir la centralisation et la visualisation de données de monitoring. Il reprend en détail l'architecture et les fonctionnalités implémentées ainsi que les choix effectués pour mettre sur pied une solution robuste, utilisable et pérenne. Les différentes étapes de la mise en production sont également explicitées et des modes d'emploi ont été fournis en annexe.

L'application de monitoring et les changements effectués pour satisfaire aux nouvelles exigences du système de centralisation sont décrits dans le chapitre 5. Ce dernier met en avant l'importance des changements qu'il y avait à effectuer au vu du rôle que possède cette application de monitoring. Elle est la source des données du système de centralisation et de visualisation et bien qu'elle soit décrite en détail dans le rapport du projet Codepo de l'année précédente il était important de mettre en avant les changements effectués sur cette nouvelle version.

L'implémentation sur le terrain faisait partie intégrante de la dynamique de ce mémoire. Afin de garder ce mémoire centré sur le contenu scientifique, l'expérience sur place est décrite de manière formelle et centrée sur les objectifs de la mission qu'AEDES avait posés avant le départ sur le terrain.

Le chapitre 7 reprend les différents résultats des tests effectués lors de l'implémentation sur le terrain et du développement en amont. Ces résultats appuient la validation ou non des différentes composantes du système de monitoring dans son ensemble. Des conclusions et recommandations sont également proposées afin d'évaluer les résultats et de faire des recommandations pour le futur concernant la direction prise par les solutions proposées pour la mise en place de ce système.

Enfin, le chapitre 8 clôture ce mémoire en donnant une vision d'ensemble sur les tâches qui ont été accomplies ainsi que sur le résultat de l'implémentation du système et de ses possibles extensions.

1.5 Notations

Dans ce travail, le terme *flux* sera utilisé pour décrire le chemin parcouru par les données d'un point A à un point B. Le terme *dashboard* définira les pages proposant des outils de visualisation pour différents paramètres observés. Le terme *abonnement* représente le lien établi entre une personne et un service surveillé. L'acronyme *CRUD* est utilisé pour définir les opérations de base dans une base de données, à savoir create-read-update-delete. Le terme *middleware* est utilisé pour décrire un logiciel tiers qui crée un réseau d'échange d'informations entre différentes applications informatiques ou parties d'applications informatiques.

Chapitre 2

Etat de l'art

2.1 Les systèmes de monitoring par SMS

2.1.1 Introduction

De nos jours, l'utilisation de SMS dans le domaine de la surveillance se décline sous trois aspects. Dans le premier, le SMS reçu par l'utilisateur a pour but de récolter de l'information sur lui et c'est donc l'humain lui-même qui se trouve être monitoré. Ce type de pratique est fréquente dans le domaine médical. Le patient reçoit un SMS lui demandant par exemple d'encoder sa température et le suivi médical peut ainsi se faire à distance. Un autre aspect est l'avertissement lors d'un danger ou d'une alerte. Ces SMS permettent de prévenir l'utilisateur d'une alerte particulière ou d'un rapport d'activité. Ce genre de monitoring existe soit pour prévenir la population d'un évènement important soit lorsque la personne est responsable d'un système critique. Dans ces cas là, une réponse par SMS de la personne en charge n'est pas nécessairement attendue. Enfin, il existe un dernier type de monitoring par SMS où le SMS n'est pas utilisé pour être lisible par un humain, mais bien simplement pour transporter les données entre deux machines. Ce type de monitoring existe également dans le domaine médical [4] et permet le monitoring de signes vitaux en temps réel [5].

2.1.2 Les types d'architectures

Au vu des différents aspects que peut prendre le monitoring par SMS, il est possible d'en extraire plusieurs architectures, avec nombres de déclinaisons en fonction des intermédiaires et dépendant de si le SMS se trouve être à la source du système ou à son embouchure. Cependant dans la figure 2.1 a été reprise l'architecture la plus générale possible permettant à la fois de représenter les cas avec présence d'un serveur SMS ainsi que sans.

Différents chemins peuvent être empruntés en fonction de la complexité de l'application :

3-8-1-2 Représente le chemin typiquement pris pour la surveillance de patient à distance

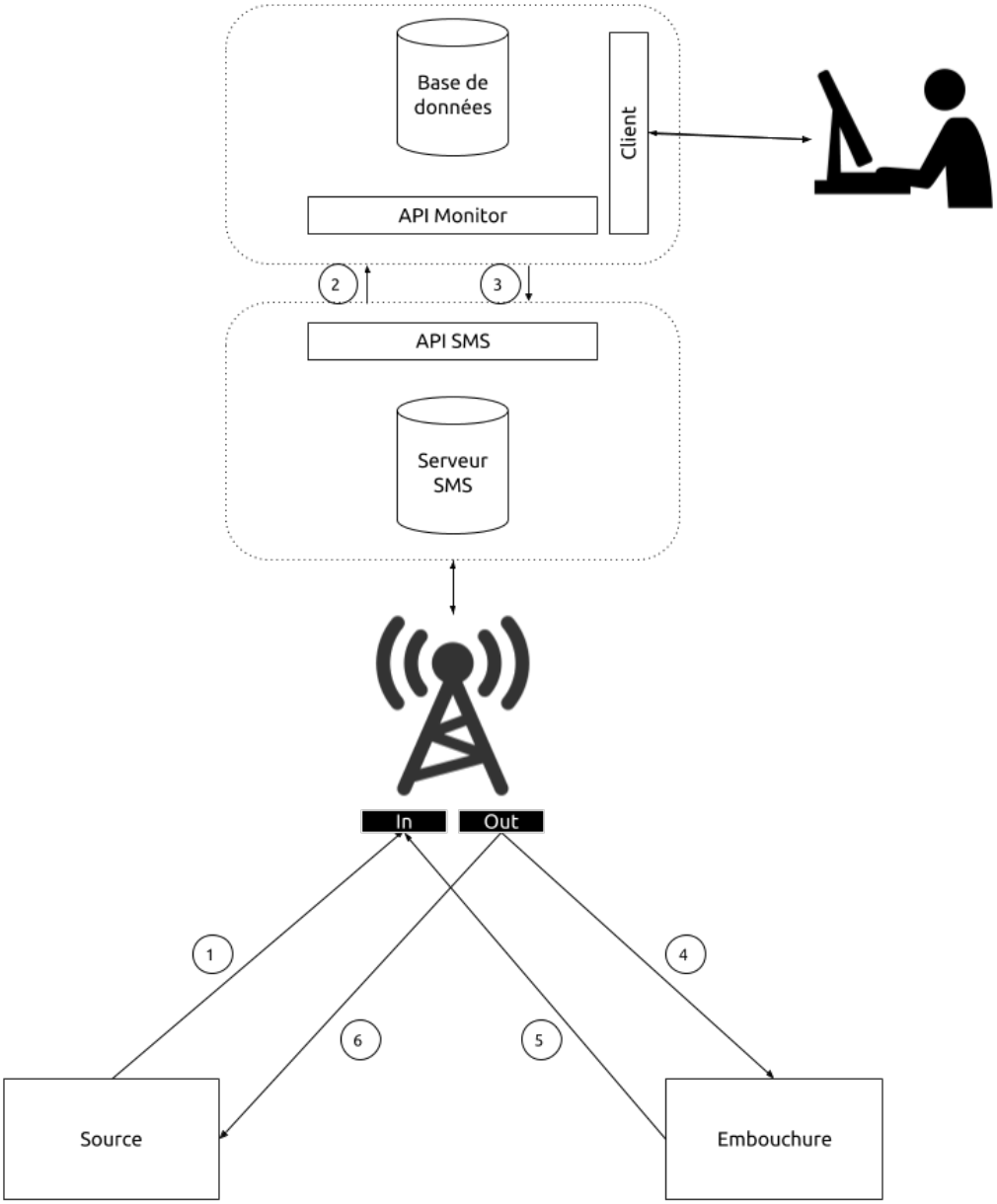


FIGURE 2.1 – Architecture générale d'un système de monitoring par SMS

via une interaction avec ces derniers. L'utilisateur/l'application responsable de récolter des informations sur les patients envoie une requête (**3**), le serveur SMS achemine les SMS aux différents patients (**6**), le patient répond ensuite au SMS (**1**) et l'information est traitée sur la plateforme (**2**) [6].

3-4 Peut se rencontrer dans les cas où la population doit être prévenue par l'état en cas d'alerte. L'état responsable d'informer la population envoie une requête (**3**), le serveur SMS fait ensuite parvenir cette requête sous forme de SMS à tous les membres de la population (**4**) [7].

1-4-5-6 S'observe dans les cas où il n'y a pas de système de centralisation de données, mais uniquement un système d'alerte ou de rapport directement acheminés aux personnes responsables du système. Cela peut s'observer dans le cas d'installation dans les maisons de particuliers. La source enclenche une alerte pour prévenir que l'utilisation de l'électricité, par exemple, est trop

importante (**1**), la personne responsable reçoit le SMS (**4**) et peut ensuite répondre via une commande SMS propre au système (**5**) afin de couper certains appareils utilisant le réseau électrique. Cette commande est ensuite interprétée par la source (**6**) qui réalise l'action en conséquence [8].

1-2-3-6 Est une architecture typiquement observée en télémédecine. Des capteurs posés sur le patient se chargent de récolter des données pour les transmettre à un personal gateway (PG), souvent le smartphone du patient, et ces informations sont transmises directement aux médecins en charge de la personne (**1**) et (**2**). Il peut également être possible de réaliser une requête soit pour récolter des données soit pour configurer l'appareil sur le patient (**3**) (**6**) [4].

1-2-3-4-5-2-3-6 Système complet comprenant une source, une plateforme de centralisation et de visualisation et une embouchure. La source envoie un SMS en cas de détection d'une erreur/alerte/rapport (**1**), cette information est ensuite traitée au niveau du serveur pour pouvoir être visualisée par un utilisateur (**2**). Par après, en fonction de certains abonnements configurés sur la plateforme, une requête est automatiquement envoyée au serveur SMS (**3**) pour transmettre le SMS d'erreur/alerte/rapport aux personnes concernées (**4**). Le flux peut ensuite procéder dans le sens inverse, de l'embouchure à la source afin de configurer ou faire réaliser une action particulière à la source (**5**) (**2**) (**3**) (**6**) [9].

Ces différentes architectures peuvent également s'accompagner de variantes dans la mesure où certaines fonctionnalités peuvent être présentes ou non. En effet, certaines applications ne permettent pas d'interaction entre l'embouchure et la source. Ainsi, l'étape **6** n'existe pas et le flux ne se fait que dans un sens.

2.1.3 Les contraintes et avantages du SMS

Le SMS, Short Message Service, est une technologie de la communication mobile originellement inventée lors du partenariat franco-allemand pour la mise en place du protocole GSM en 1984 par Friedhelm Hillebrand et Bernard Ghillebaert [10]. Son utilisation a été popularisée à partir des années 2000 et lors du jour de Noël en 2006, plus de 205 millions de SMS ont été envoyés pour le Royaume-Uni seulement [11].

Le SMS est un protocole dont la taille d'un segment est limitée à 140 bytes. Cela signifie qu'en fonction de l'alphabet utilisé, il est possible d'encoder 160 caractères (alphabet de 7 bits), 140 caractères (alphabet de 8 bits) ou 70 caractères (alphabet de 16 bits) [12]. Afin d'obtenir des messages plus longs, les segments peuvent être concaténés auquel cas, chaque message commencera avec un User Data Header (UDH) contenant l'information nécessaire au format et à l'ordonnancement du SMS. Cet UDH possède une taille de 6 bytes et les nombres maximums de caractères dans un message deviennent alors 153 (alphabet de 7 bits), 134 (alphabet de 8 bits) et 67 (alphabet de 16 bits). Bien que le SMS peut théoriquement concaténer jusqu'à 255 segments, la nature des mécanismes de transmission ne facilite pas l'acheminement de message aussi large. En effet, les SMS peuvent ne pas être délivrés (dû à leur période de validité) et le réseau GSM ne garantit pas

que les SMS sont délivrés au destinataire dans l'ordre auquel ils ont été envoyés [13]. Tout cela impacte la taille maximum pratique des SMS concaténés. Ainsi, les problèmes observés autour de la fiabilité de la transmission, de la période de validité des messages (TTL), les garanties de transmission non faites par le réseau et le manque d'accusé de réception formels doivent être pris en compte lors de l'utilisation du SMS comme moyen de communication. Par ailleurs, la sécurité de l'information doit être gardée à l'esprit lorsque l'on travaille avec des données personnelles ou des données sensibles en règle générale.

Le SMS présente l'avantage d'être peu cher et très répandu. Cette technologie est accessible presque universellement à n'importe qui possédant un téléphone mobile de seconde génération ou supérieure. Avec sa capacité à encoder un grand nombre d'informations en concaténant les segments, le SMS offre une excellente capacité de transmission de données et est adapté pour l'envoi de données discrètes. Dans le cas de données critiques où l'échec de la transmission n'est simplement pas une option, le SMS n'est pas un choix idéal comme médium au vu du manque de garanties fournies par le réseau GSM.

2.2 La visualisation de données

2.2.1 Introduction

L'explosion de la quantité de données générées par une entreprise est aujourd'hui un phénomène récurrent et s'applique non seulement aux domaines de l'IT mais également dans chaque cas où il est nécessaire d'utiliser une base de données pour le stockage de l'information. Une bonne gestion de ces données est donc un enjeu clé pour permettre un suivi efficace et des analyses, parfois très poussées, des relations existantes entre ces dernières. Afin qu'un humain puisse réaliser ces analyses facilement, une bonne visualisation de ces données est un point fondamental. Plusieurs études montrant les bénéfices apportés par une bonne visualisation de données ont été réalisées et ci-dessous se trouvent les résultats de l'une d'entre elles. La table 2.1 reprend ces bénéfices selon le pourcentage correspondant dans différents domaines du monde de l'entreprise [14].

De bonnes pratiques sont à respecter dans le domaine de la visualisation :

- Ne pas oublier de traiter les meta-données, car ces dernières peuvent être très révélatrices.
- Les outils de visualisation doivent être interactifs et l'implication de l'utilisateur est importante.
- Encourager cette interactivité. Les outils où les données sont statiques ne mènent pas autant à la découverte de pattern/problèmes que les outils où les données sont interactives [15].

Différentes fonctionnalités peuvent être mises en oeuvre pour obtenir une visualisation interactive :

Bénéfices	**Pourcentage** (%)
Amélioration de la prise de décision	77
Meilleure analyse de données ad hoc	43
Amélioration du partage d'information/collaboration	41
Possibilité de self-service pour les utilisateurs	36
Augmentation du retour sur investissement	34
Gains de temps	20
Réduction des charges de l'IT	15

TABLE 2.1 – Bénéfices de l'utilisation d'outils de visualisation [14]

- La sélection : La sélection de sous-ensembles de données en fonction de l'intérêt de l'utilisateur
- La mise en relation : Pouvoir relier des informations au travers de plusieurs vues
- Le filtrage : Permettre à l'utilisateur de ne traiter qu'un nombre restreint de données
- Le réarrangement : L'aménagement spatial est très important dans le cas de cartes et permet de fournir de nombreux aperçus

2.2.2 Outils existants

De nombreux outils spécialisés dans la visualisation de données existent déjà sur le marché. Ils permettent également des interactions avec ces données et peuvent donc être vus comme des modèles dans le domaine. Il faut différencier trois types d'outils qui sont utilisés dans trois cas différents :

- Les outils de Business Intelligence
- Les outils de Dashboard
- Les bibliothèques issues de D3

Ces trois outils sont décrits plus en détail dans les sections suivantes.

2.2.2.1 Les outils de Business Intelligence

Les outils de Business Intelligence sont des outils d'aide à la décision permettant de traiter un nombre très important de données et de les restituer afin de permettre à un décideur d'avoir une vue d'ensemble de l'activité traitée. Ils participent ainsi à faciliter l'interprétation des Big Data.

Des logiciels, tels que Tableau, Sisense ou Looker, proposant des fonctions de visualisation et d'interaction pour interagir avec les données ont été développés. Ils ne requièrent que peu de

compétences en programmation et proposent un large panel pour intégrer ces fonctionnalités dans tous types de projet. Cependant, bien que les outils de Business intelligence peuvent traiter des ZB (zettabytes) et PB (pettabytes) ils ne peuvent que rarement proposer de visualiser toutes ces données. De plus, le prix des licences pour utiliser ce type d'outils est souvent élevé (i.e : 70$/utilisateur/mois pour Tableau).

2.2.2.2 Les outils de Dashboard

Les outils de Dashboard sont des outils de visualisation permettant à un utilisateur de créer des tableaux de bord avec des fonctionnalités de visualisation de données. Ces outils, contrairement aux outils de Business Intelligence, se limitent à la visualisation et à l'analyse de ces données.

Des outils tels que Kibana ou Grafana proposent ces fonctionnalités et possèdent souvent une partie open source. Cependant, ils imposent parfois le type de stack à utiliser (ElasticSearch pour Kibana) et requièrent plus de compétences en programmation que pour les outils de Business Intelligence.

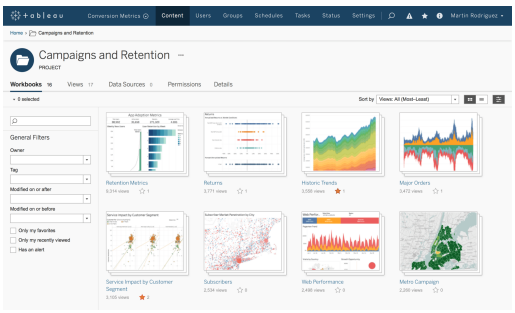


FIGURE 2.2 – Dashboard généré par Tableau

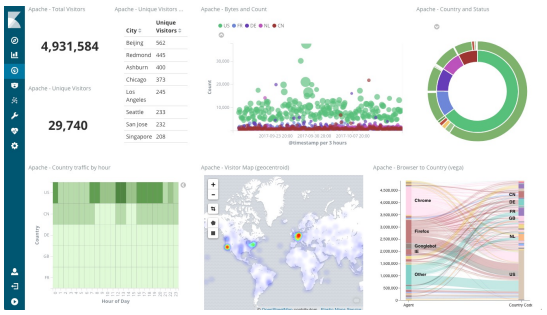


FIGURE 2.3 – Dashboard généré par Kibana

2.2.2.3 Les bibliothèques issues de D3

D3 est une bibliothèque JavaScript orientée web permettant de manipuler des documents orientés données. D3 permet de lier des données arbitraires au DOM (Document Object Model) et ensuite d'appliquer des transformations au document. Contrairement à beaucoup d'autres bibliothèques, D3 permet un contrôle important sur le résultat visuel final [16]. Cette bibliothèque trouve sa source en 2009 dans le cadre du mémoire de Mike Bostock et cette dernière s'appelait alors Protovis [17]. Son développement se popularisa en 2011, à la sortie de la version 2.0 en août 2011 [18]. En août 2012, la bibliothèque avait atteint la version 2.10.0 et aujourd'hui, D3.js et les librairies issues de D3 sont utilisées dans plusieurs centaines de milliers de sites web [19].

Il est à noter que des compétences avancées en programmation sont requises pour utiliser ces bibliothèques et que les contraintes liées à l'intégration dans une application plus conséquente sont à traiter au cas par cas (format des données, types de langage, ...).

2.3 La coopération au développement

La coopération au développement possède 4 caractéristiques principales :

- Son objectif est de contribuer aux priorités nationales et internationales en développement
- Elle est sans but lucratif
- Elle discrimine positivement en faveur des pays émergeant
- Elle est basée sur des relations de coopération cherchant à augmenter l'appropriation des pays en développement

Par ailleurs, en 2016 l'ONU a adopté de nouveaux objectifs du millénaire pour le développement. Ces objectifs font suite aux 8 premiers énoncés lors de la "Déclaration du millénaire de l'Organisation des Nations unies" à New York le 8 septembre 2000 et sont maintenant au nombre de 17 (voir figure 2.4) [20].

En Belgique, le 1er décembre 2011, un accord de gouvernement stipule : "En Afrique centrale, la Belgique continuera à promouvoir activement l'Etat de droit en luttant contre l'impunité, surtout concernant les violences sexuelles. La transparence dans l'exploitation des ressources naturelles sera promue au profit des populations locales. La Belgique poursuivra ses efforts en matière de soutien à la bonne gouvernance et de lutte contre la corruption. Enfin, elle encouragera la coopération régionale."

Par cette déclaration [21], la Belgique place l'Afrique, et plus particulièrement l'Afrique centrale, en haut de l'agenda de la politique étrangère belge. Par ailleurs, la CTB (i.e : coopération technique belge, qui a entre temps changé de nom pour s'appeler Enable) place les secteurs de la santé et de la digitalisation comme des priorités dans son programme ce qui inscrit tout à fait ce mémoire dans les lignes de la CTB [22]. Plus particulièrement, en République Démocratique du Congo, trois secteurs sont priorisés : L'agriculture et le développement rural, l'éducation et la santé. Dans le secteur de la santé, la CTB vise à [23] :

> "[...] améliorer l'accès physique et financier de la population aux soins de santé de qualité adaptés à leurs besoins. Cela implique non seulement un appui aux centres de santé et hôpitaux, mais également un appui au management du système de santé.[...]"

Le management des centres de santé par leur digitalisation est le coeur du projet CERHIS et de ce mémoire ce qui inscrit ce dernier à la fois dans les objectifs du millénaire ainsi que dans les directions prises par la CTB en République Démocratique du Congo.

1. Éliminer la pauvreté sous toutes ses formes et partout dans le monde
2. Éliminer la faim, assurer la sécurité alimentaire, améliorer la nutrition et promouvoir l'agriculture durable
3. Permettre à tous de vivre en bonne santé et promouvoir le bien-être de tous à tout âge
4. Assurer l'accès de tous à une éducation de qualité, sur un pied d'égalité, et promouvoir les possibilités d'apprentissage tout au long de la vie
5. Parvenir à l'égalité des sexes et autonomiser toutes les femmes et les filles
6. Garantir l'accès de tous à l'eau et à l'assainissement et assurer une gestion durable des ressources en eau
7. Garantir l'accès de tous à des services énergétiques fiables, durables et modernes à un coût abordable
8. Promouvoir une croissance économique soutenue, partagée et durable, le plein emploi productif et un travail décent pour tous
9. Bâtir une infrastructure résiliente, promouvoir une industrialisation durable qui profite à tous et encourager l'innovation
10. Réduire les inégalités dans les pays et d'un pays à l'autre
11. Faire en sorte que les villes et les établissements humains soient ouverts à tous, sûrs, résilients et durables
12. Établir des modes de consommation et de production durable
13. Prendre d'urgence des mesures pour lutter contre les changements climatiques et leurs répercussions
14. Conserver et exploiter de manière durable les océans, les mers et les ressources marines aux fins du développement durable
15. Préserver et restaurer les écosystèmes terrestres, en veillant à les exploiter de façon durable, gérer durablement les forêts, lutter contre la désertification, enrayer et inverser le processus de dégradation des terres et mettre fin à l'appauvrissement de la biodiversité
16. Promouvoir l'avènement de sociétés pacifiques et ouvertes à tous aux fins du développement durable, assurer l'accès de tous à la justice et mettre en place, à tous les niveaux, des institutions efficaces, responsables et ouvertes à tous
17. Renforcer les moyens de mettre en œuvre le Partenariat mondial pour le développement durable et le revitaliser

FIGURE 2.4 – Objectifs du millénaire de la coopération au développement

Chapitre 3

Cahier des charges

3.1 Introduction

Le projet CERHIS fait aujourd'hui face à plusieurs problèmes. Tout d'abord, le monitoring des centres hospitaliers génère un grand nombre de données et il est impossible d'y accéder à distance. Ensuite, monitorer par SMS pose un problème de spam chez les techniciens et représente des coûts trop importants dès que les SMS doivent être envoyés à l'étranger. Ce mémoire a donc pour but de fournir un système permettant de gérer les informations collectées sur les centres de santé. De plus, une solution pour centraliser toutes les données en un même lieu doit également être mise sur pieds. Enfin, un système de redistribution de ces SMS vers les smartphones des techniciens devra être pensé. Il est donc question d'un système end-to-end comprenant une source, une plateforme de centralisation et de visualisation et une embouchure à laquelle les SMS sont redistribués.

La figure 3.1 reprend les centres de santé (**1**) desquels sont envoyés les SMS vers le serveur SMS (**2**). Ces données peuvent ensuite être observées sur une plateforme de visualisation (**3**) et les SMS à destination des techniciens sont redirigés du serveur SMS vers les smartphones des personnes en charge (**4**).

Pour l'instant, seule la solution au niveau du centre de santé existe (voir figure 3.2 et se référer au projet de l'année 2016-2017 pour plus de détails). A ce niveau, le flux de données se fait directement du centre de santé vers les smartphones des techniciens. L'objet de ce mémoire reprend donc les deux autres éléments du schéma à savoir l'application web et le serveur SMS. Des modifications sur l'application de monitoring sont également à prévoir au vu de ce passage à l'échelle.

La solution de monitoring présentée dans la figure 3.2 et mise en place dans les centres de santé permet de surveiller les différents éléments vitaux le fonctionnement de CERHIS. Certains ne sont pas représentés sur le schéma par souci de lisibilité, mais sont monitorés de la même manière que la batterie. Dans ces éléments il faut compter :

- Le wifi

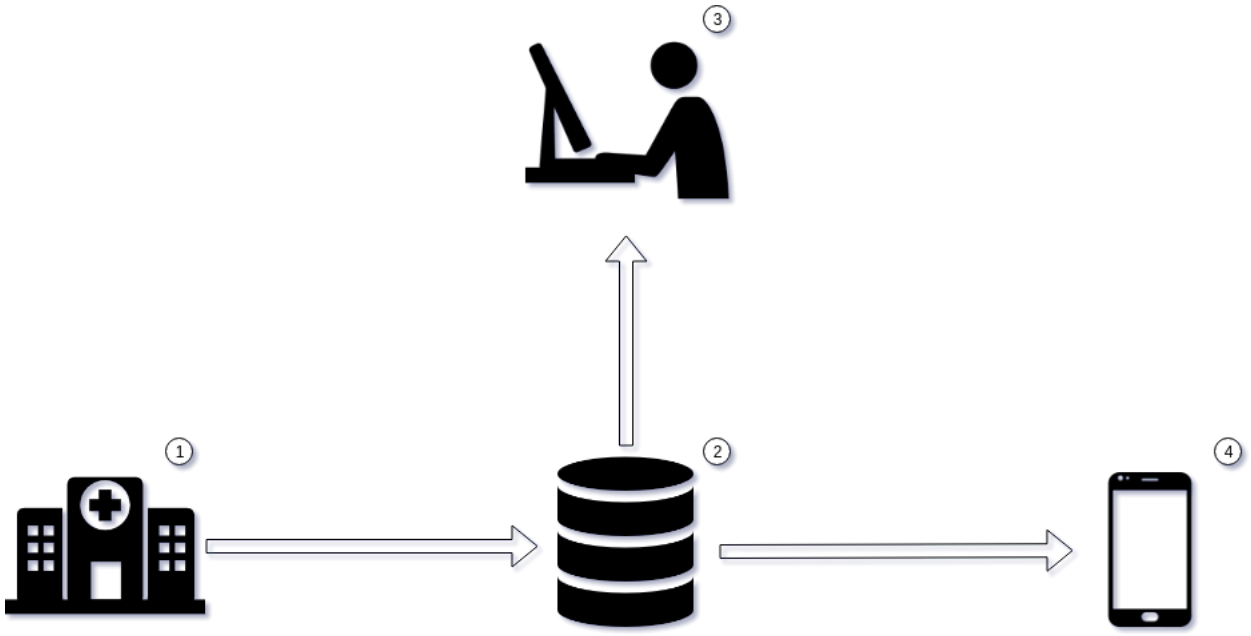


FIGURE 3.1 – Flux de base de l'information provenant des centres de santé

- Les tablettes (et plus généralement les éléments connectés au wifi)
- L'état du serveur
- La charge de la batterie
- La charge du smartphone
- Le réseau électrique
- Les panneaux solaires

Tous ces éléments sont monitorés par l'application Android qui permet d'envoyer des SMS aux techniciens en cas d'alerte. Ce sont les SMS provenant de ces structures de santé qu'il faut pouvoir centraliser et visualiser sur le système à développer.

3.2 Objectifs

Beaucoup d'éléments peuvent être développés à partir du processus de base. Après plusieurs brainstorming et grâce à l'expérience déjà acquise sur le terrain durant l'été 2017, il a été possible de déceler différents objectifs et fonctions qu'il serait intéressant d'implémenter. Cependant, étant donné les contraintes de temps, des priorités ont été mises afin de classer les objectifs en fonction de leur importance et de l'impact que l'implémentation de ces solutions pourrait apporter aux bénéficiaires du projet. Ci-dessous, se trouvent les objectifs dégagés lors des premières réunions. Par après, ces objectifs sont également classés en fonction de leur degré de priorité.

1. **Système de visualisation de données**

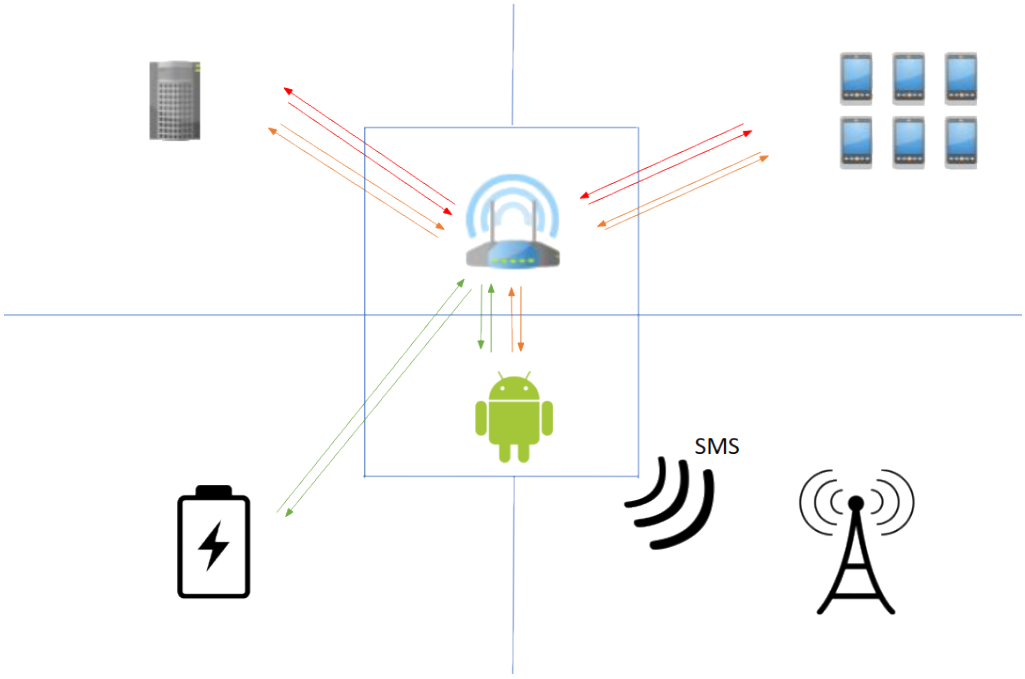


FIGURE 3.2 – Fonctionnement du monitoring dans les centres de santé

1.1. Réception des données du serveur SMS et présentation du contenu

1.2. Authentification

1.3. Gestion des utilisateurs

1.4. Abonnement aux flux de SMS

1.5. Visualisation graphique des données

1.6. Modularité en fonction des serveur SMS

1.7. Disponibilité indépendante de la localisation

1.8. Synchronisation avec les changements réalisés sur l'application de monitoring

1.9. Système de permissions et privilèges

2. **Application technicien permettant la gestion des SMS**

2.1. Traitement des SMS de monitoring et visualisation sur une carte

2.2. Système d'authentification

3. **Application de monitoring**

3.1. Gestion des tâches de fond

3.2. Encodage de l'information

3.3. Refactorisation en fonction des besoins du système de visualisation

3.4. Synchronisation avec les changements réalisés sur le système de visualisation de données

4. **Serveur SMS**

4.1. Déterminer le fournisseur d'accès

3.2.1 Hiérarchisation et organisation des objectifs

Les tâches ont été triées en fonction des priorités estimées par l'entreprise AEDES. Ces dernières sont présentées dans la pyramide ci-dessous. Elles sont ordonnées de telle sorte que plus les tâches sont proches de la base, plus elles sont prioritaires. De façon similaire, plus elles sont proches du sommet, moins elles seront favorisées dans un premier temps.

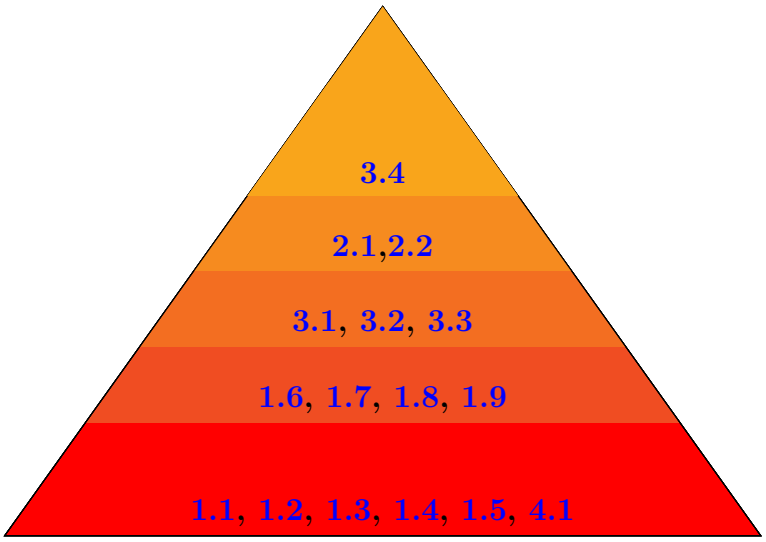


FIGURE 3.3 – Pyramide des priorités du mémoire

La priorité a été mise sur la conception du système de visualisation de données et la mise à jour de l'application de monitoring. En effet, le but premier d'AEDES est de développer un MVB (minimum viable product) de telle sorte que le processus illustré à la figure 3.2 soit opérationnel.

3.3 Description et limitation du périmètre

3.3.1 Système de visualisation de données

La réalisation de mockups (aussi appelé wireframing) pour illustrer le fonctionnement d'un logiciel et de ses fonctionnalités est une pratique courante dans le monde professionnel. De la sorte, un utilisateur peut directement donner son feedback sur son expérience avant même qu'une seule ligne de code soit écrite. C'est la raison pour laquelle le lien suivant reprend le fruit des discussions avec AEDES sur l'utilisation du logiciel. Des captures d'écran du mockup sont également proposées à la figure 3.4

https://marvelapp.com/32fg259/screen/35982955

La seule contrainte imposée par AEDES concernant les spécifications techniques du projet est l'utilisation de CouchbaseDB pour le stockage des informations. Cette contrainte est intégrée et spécifiée dans la partie 4.1.2.3.

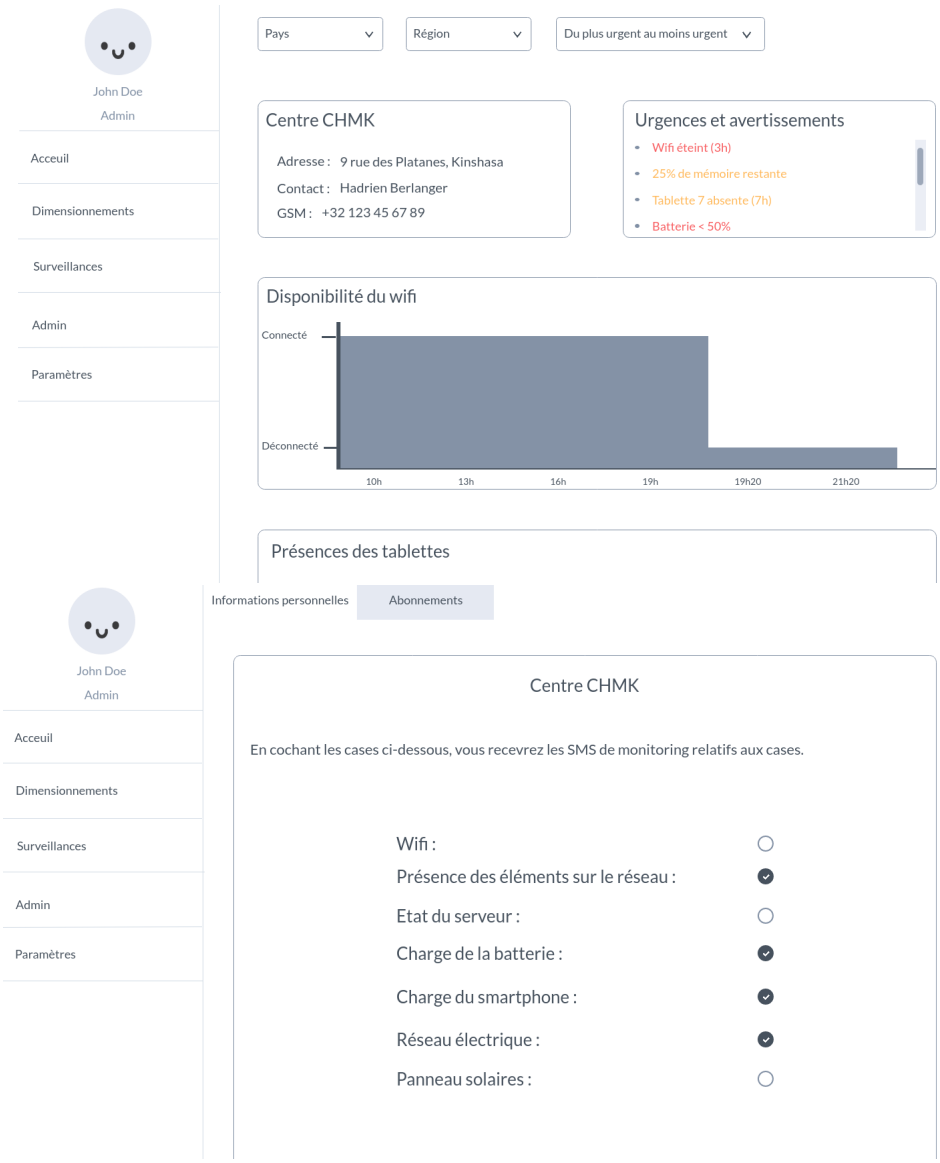


FIGURE 3.4 – Captures d'écran du mockup réalisé pour l'application web

Les personnes susceptibles d'utiliser ce système et/ou d'y avoir trait sont les **techniciens informatiques** des systèmes hospitaliers, les **responsables** des centres hospitaliers ainsi que les **responsables de projet** chez AEDES. Il est également probable que des **membres du ministère de la santé** aient à utiliser ce sytstème.

3.3.2 Serveur SMS

Bien que ce projet vise à s'étendre à plusieurs pays d'Afrique centrale, ce mémoire se concentrera sur la solution en République Démocratique du Congo. Ainsi, même si le système de redistribution des SMS doit être modulable en fonction des différentes API des fournisseurs de Serveur SMS, il ne sera question ici que d'un seul fournisseur propre à la RDC, le travail ne devant qu'être répété pour les autres pays.

Les personnes susceptibles d'interagir directement avec le Serveur SMS sont **les développeurs**

chargés de la maintenance du système.

3.3.3 Application technicien

De même que pour le système de visualisation de données, un mockup a été réalisé afin d'avoir un premier rendu graphique interactif de ce que l'application pourrait donner. Ce mockup est disponible à l'adresse ci-dessous. De plus, des captures d'écran sont également proposées à la figure 3.5.

https://marvelapp.com/2i21g53/screen/42037467

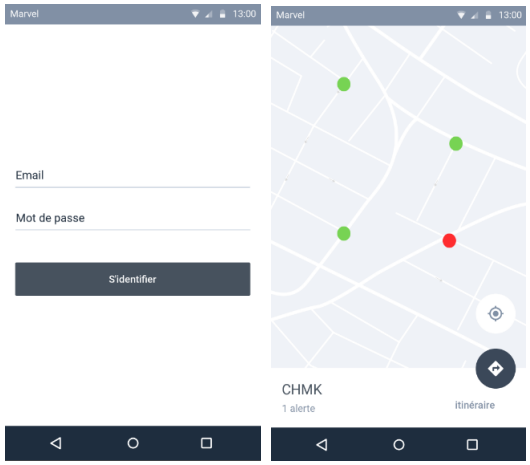


FIGURE 3.5 – Captures d'écran du mockup réalisé pour l'application technicien

Les personnes susceptibles d'utiliser l'application et/ou d'y avoir trait sont les **techniciens informatiques** des systèmes hospitaliers

3.3.4 Application de monitoring

L'application de monitoring a été le centre du sujet du projet Codepo 2016-2017 (voir section 1.3.1.4). Au terme de ce projet, la gestion des services en tâche de fond n'était pas optimale ce pourquoi dans le cadre du projet et de la mission à venir, il est demandé dé l'améliorer. Il est également important de se rendre compte que beaucoup de travail sur la refactorisation est à prévoir. Notamment pour les fonctionnalités de synchronisation des bases de données.

Le développement se fait en Java sur Android Studio qui est la référence en matière d'IDE pour Android.

Les personnes susceptibles d'utiliser l'application et/ou d'y avoir trait sont les **techniciens informatiques** des systèmes hospitaliers

3.4 Planning

De début novembre à fin janvier
Prise de contact avec AEDES pour établir un cahier des charges complet. Un premier MVB serait idéal pour la fin janvier. Prise de contact avec des personnes chez les fournisseurs de serveur SMS pour établir un partenariat. Ce partenariat doit pouvoir fournir un serveur de test en Belgique ainsi qu'un serveur en RDC lorsque le système devra être installé sur place.

De début février à fin mars
Intégration d'une visualisation plus esthétique des données sur l'application web. Gestion des utilisateurs avec permissions, abonnements à certains flux de SMS. Refactoriser l'application de monitoring et amélioration de cette dernière en fonction de feedbacks reçus quant à son utilisation.

Avril
Implémentation du système lors d'une expérience pilote sur le terrain, en République Démocratique du Congo. Ce séjour constitue évidemment une étape clé du projet. Le transfert des solutions développées devra se faire en harmonie avec les partenaires locaux.

De début mai à fin juin
Rédaction et finalisation du mémoire. La documentation du projet sur le wiki d'AEDES est une étape clé pour permettre la pérennité du projet.

3.5 Résumé

Ce mémoire s'inscrivant dans le cadre d'un projet propre à une entreprise, il est important de spécifier le périmètre pour satisfaire les objectifs techniques et s'assurer que les objectifs académiques seront alignés avec les attentes de l'entreprise.

Ainsi, il a été possible de prioriser certaines fonctionnalités afin d'obtenir le MVB voulu tout en mettant l'accent sur le centre de ce mémoire et des attentes académiques, à savoir le système de visualisation et de centralisation des SMS de monitoring. Chacun des éléments a été décrit succinctement et des précisions plus techniques sont données dans la suite du mémoire. Enfin, les délais ont été posés afin de garder le développement à l'oeil de fixer des objectifs à plus ou moins long terme.

La suite de ce mémoire reprend donc les éléments du MVB pour AEDES, c'est-à-dire le système de centralisation et de visualisation des données de monitoring et l'application de monitoring ainsi que toutes les fonctionnalités reprises dans les trois premiers niveaux de la pyramide des priorités, figure 3.3. Une description technique est ainsi donnée et une explication sur l'implémentation sur le terrain est également décrite. Enfin des résultats sont tirés et présentés dans le chapitre 7.

Chapitre 4

Système de centralisation et de visualisation de données

4.1 Description technique

4.1.1 Type de solution

Au vu des outils existants présentés dans la partie 2.2.2 il est à même de se demander quelle approche prendre pour décider du type de système de visualisation de données. Vaut-il mieux utiliser des outils de Business Intelligence, intégrer des outils de Dashboard dans une application ou développer une application en utilisant les bibliothèques issues de D3 ?
Les trois solutions possèdent leurs avantages et leurs inconvénients. Les outils de Business Intelligence sont des outils puissants qui permettent non seulement l'analyse de Big Data mais ils offrent également tout un panel d'outils d'aide à la décision. Cependant, ces derniers peuvent être coûteux et les besoins en visualisation spécifiés dans le cahier des charges ne sont pas critiques assez que pour imposer l'utilisation de tels outils. De plus, bien que la part de développement soit plus faible que pour les deux autres possibilités, des fonctionnalités telles que la gestion des permissions ou la gestion des abonnements ne sont pas présentes. Concernant les outils de Dashboard, ils sont plus légers que les outils de Business Intelligence et offrent également des visualisations très élaborées. De plus, ces derniers offrent souvent un grand nombre de fonctionnalités Open Source. Malgré cela, la gestion des abonnements reste impossible en n'utilisant que ces outils et une application tierce reste tout de même à prévoir. Enfin, la réalisation d'une application en utilisant les bibliothèques issues de D3 représente beaucoup de développement et un gros travail sur la modularité des composants ainsi que pour implémenter les fonctionnalités de base (authentification, gestion des utilisateurs) mais elle est la seule permettant de satisfaire à toutes les contraintes du cahier des charges. De ce fait, le système de visualisation de données consistera à développer une application web faisant usage des bibliothèques issues de D3 et satisfaisant toutes les contraintes définies.

En outre, le besoin de pérennité du projet étant un facteur critique de ce mémoire, permettre une utilisation simple et logique du système est fondamental. En effet, au vu des différents profils des personnes susceptibles d'utiliser le système, une bonne expérience utilisateur est très importante. C'est la raison pour laquelle une attention particulière sera portée au design et à la facilité d'utilisation de l'application.

Le tableau 4.1 reprend une comparaison entre les différentes technologies présentées à la section 2 ainsi qu'une pondération en fonction de leur pertinence par rapport aux critères définis. La pertinence par rapport aux critères relevés est comprise entre 1 (peu pertinent) et 5 (très pertinent). Ces pondérations prennent à la fois en compte le travail à fournir pour mettre en place le critère et la faisabilité de ce dernier.

	Outils BI	**Outils Dashboard**	**Application web**
Visualisation des données	5	4	2
Analyse des données	5	3	1
Coût	1	3	4
Centralisation des données	2	3	5
Modularité (API tierces)	1	3	5
Fonctionnalités	1	2	5
Expérience utilisateur	1	2	5
Total	16	20	27

TABLE 4.1 – Tableau comparatif des différentes solutions potentielles

Les critères "Visualisation de données" et "Analyse des données" rendent compte du fait que ces fonctionnalités sont déjà implémentées dans certains outils et que les développer soi-même représente un travail conséquent. Le critère "Coût" ne demande pas plus d'explications (se référer au chapitre 2 pour plus d'informations). Le critère "Centralisation des données" prend en compte le fait qu'il y a un besoin d'application en temps réel et que certaines solutions ne proposent qu'une mise à jour discrète et non continue des données. La "Modularité" représente la facilité d'intégrer des API tierces dans la solution, notamment pour ce qui est de l'utilisation d'un serveur SMS. Les "Fonctionnalités" font état de la facilité d'intégrer les fonctionnalités décrites dans le chapitre 3 quant à l'"Expérience utilisateur" elle représente la facilité d'adapter l'expérience utilisateur en fonction du besoin des personnes impliquées.

4.1.2 Stack utilisé

Choisir un stack pour le développement d'une application web est un choix important, qui doit être réfléchi, autrement le risque est de subir un stack inadapté, que cela soit à cause de l'habitude ou à cause de ce qui est déjà en place. Ce choix est d'autant plus important que les outils disponibles

sont de plus en plus nombreux, et changent de plus en plus rapidement. Etant donné que cette application web comprendra des fonctionnalités avancées et nécessite la mise en place d'une base de données, il faut pouvoir déterminer le type de technologie utilisée pour :

- *le Frontend* : La partie présentée à l'utilisateur. Le premier niveau d'interaction avec les données.
- *le Backend* : La partie présente sur le serveur hébergeant l'application. Le second niveau d'interaction avec les données.
- *la Base de données* : Le stockage des données brutes. Le dernier niveau d'interaction avec les données.

4.1.2.1 Le Frontend

Ces dernières années ont vu un apparaître un réel essor des technologies web et de bibliothèques propres à chacune de ces technologies. Dans ces dernières, les technologies Frontend occupent une grande part et se déclinent soit sous la forme de frameworks HTML/CSS ou de frameworks Javascript.

Dans les frameworks HTML/CSS nous retrouvons des outils tels que Twitter Bootstrap, Materialize CSS, Bulma, Zurb Foundation ou encore Skeleton. Tous ces frameworks permettent de limiter le développement HTML et CSS de composants et de conserver une certaine cohérence entre les différents éléments composant l'application web.
Au niveau des frameworks Javascript, il existe différents outils apparus ces dernières années permettant de générer les pages web uniquement à partir d'un fichier bundle Javascript. Ainsi, React.js, Vue.js, Angular.js et d'autres frameworks moins populaires partagent le même but avec certaines subtilités propres à chacun. L'avantage de ces frameworks/librairies Javascript est l'apport d'une structure clairement définie par rapport aux framworks HTML/CSS. C'est la raison pour laquelle le choix a été porté sur l'utilisation de ces frameworks Javascript dans le cadre du développement de l'application.

Au sein des frameworks Javascript, un choix s'impose et doit pouvoir prendre en compte les contraintes des différents outils. Cette comparaison est réalisée ci-dessous :

Angular.js est un framework publié en octobre 2010 par Google. Il est utilisé par de nombreuses grosses entreprises et est basé sur TypeScript ce qui lui donne une approche OO plus robuste dans le cas d'application conséquente. C'est un framework qui propose beaucoup de structure. Les templates Angular sont faits à base d'HTML avec du Javascript intégré ce qui peut faciliter le travail avec un designer.

React.js est une librairie publiée en mars 2013 par Facebook. Cette librairie est la plus utilisée des trois et possède un très grand écosystème. Elle est reconnue pour sa flexibilité et peut également être utilisée avec TypeScript. Tout en React.js est du Javascript, même les templates

(JSX).

Vue.js est un framework publié en février 2014 et est décrit comme le framework avec la plus grosse croissance ces deux dernières années. Il permet d'avoir un code très propre et est le framework le plus simple à apprendre des trois. De plus, il permet une séparation claire des différents composants d'une application.

Au vu des différentes caractéristiques de ces frameworks/bibliothèques Javascript, le choix s'est porté sur l'utilisation de React.js au vu de sa grande communauté, sa modularité ainsi que sa robustesse lorsque ce dernier est utilisé avec Typescript. De plus, un pipeline de production sera installé. Celui-ci se compose notamment de webpack qui participe à modulariser/optimiser le code ainsi que babel qui assure une compatibilité du code avec les browsers existants.

4.1.2.2 Le Backend

Un langage de programmation backend est habituellement choisi en parallèle avec un framework. Ces frameworks sont très utiles puisqu'ils fournissent au développeur des fonctionnalités permettant de ne pas réinventer la roue et d'abstraire toute une série de processus.

Language	**Framework**
Ruby on rails	Ruby
Python	Django
Javascript	Node.js
PHP	Laravel
C#	.NET

TABLE 4.2 – Langages et Frameworks populaires

Dans le cadre de ce mémoire, la technologie backend choisie est Node.js avec le package *Express*. Node.js possède un package manager qui offre un très large panel d'outils open source. De plus, Node.js est un environnement rapide et facilement extensible à une échelle plus importante. C'est en règle générale une technologie très utilisée dans le cas d'applications en temps réel.

4.1.2.3 La Base de données

Le choix de la base de données est également un choix important lors du développement d'une application web. Cependant, cette donnée ayant été une contrainte lors de l'attribution du mémoire, le système stockera l'information sur CouchbaseDB, une base de données NoSql.

Cette contrainte s'inscrit dans l'idée de faciliter la maintenance en utilisant des stacks technologiques communs. Cependant, ce choix en amont est motivé par les exigences du projet CERHIS. Le choix d'une base de données NoSql orientée document se justifie ainsi :

- Cela correspond bien avec le modèle de données qu'ils utilisent, notamment le fait qu'ils informatisent des documents papier
- Le processus de développement impose beaucoup de refactoring en prenant en compte ce que souhaitent les médecins. Ils avaient donc besoin d'une très grande flexibilité
- Les informations changent beaucoup d'un pays à l'autre et d'une année à l'autre, chaque fois qu'un programme médical veut changer les données collectées
- La vision NoSQL est très utile pour la vision de traitement de données à l'aide de métadonnées éclatées (ils utilisent d'ailleurs tout un bucket « metadata » pour la définition dynamique de toutes les données médicales qui sont dans le bucket « data ») – par exemple dès qu'une mise à jour est faite à distance du bucket de metadata. Ainsi la prochaine fois que l'application s'ouvrira, le formulaire pour les consultations aura été mis à jour sans recompilation.

Entre les différentes options NoSQL / NoSQL orienté document, Couchbase s'est imposé grâce à la facilité de gestion des bases de données désynchronisées sur applications mobiles. Les exigences du projet imposent la possibilité pour les tablettes de fonctionner quand elles ne sont pas connectées au serveur, et Couchbase rend ça très simple, entre autres pour la resynchronisation et la gestion de conflits. Il y a maintenant des concurrents de Couchbase comme Firebase mais dans le cas de CERHIS il est aussi impératif que le serveur puisse être local et pas dans le cloud.

4.2 Description fonctionnelle

4.2.1 L'authentification

L'authentification est une fonctionnalité souvent présente au sein d'une application web. La figure 4.1 reprend les différentes étapes à travers les différents niveaux du stack (client-serveur-base de données).

L'utilisateur interagit avec un formulaire d'authentification similaire à celui présenté dans le mockup de la partie 3 au sein duquel il rentre son adresse mail, unique, et son mot de passe. Les input sont ensuit vérifiés au niveau du client et un message d'erreur est affiché si ces derniers ne correspondent pas au format requis (erreur non représentée sur le schéma par souci de lisibilité). Une fois l'input validé, une requête http post contenant les informations rentrées est effectuée vers le serveur. Ces informations sont ensuite à nouveau vérifiées sur le serveur pour éviter qu'une requête personnalisée par l'utilisateur puisse tout de même envoyer un format non valide (même type de vérification que sur le client). La base de données est ensuite interrogée et si l'utilisateur avec l'adresse mail envoyée existe dans la base de données, il est retourné au serveur autrement une erreur de statut 401, utilisateur non identifié, est renvoyée (erreur non représentée sur le schéma par souci de lisibilité). Une fois cet utilisateur renvoyé, les hash des mots de passe sont

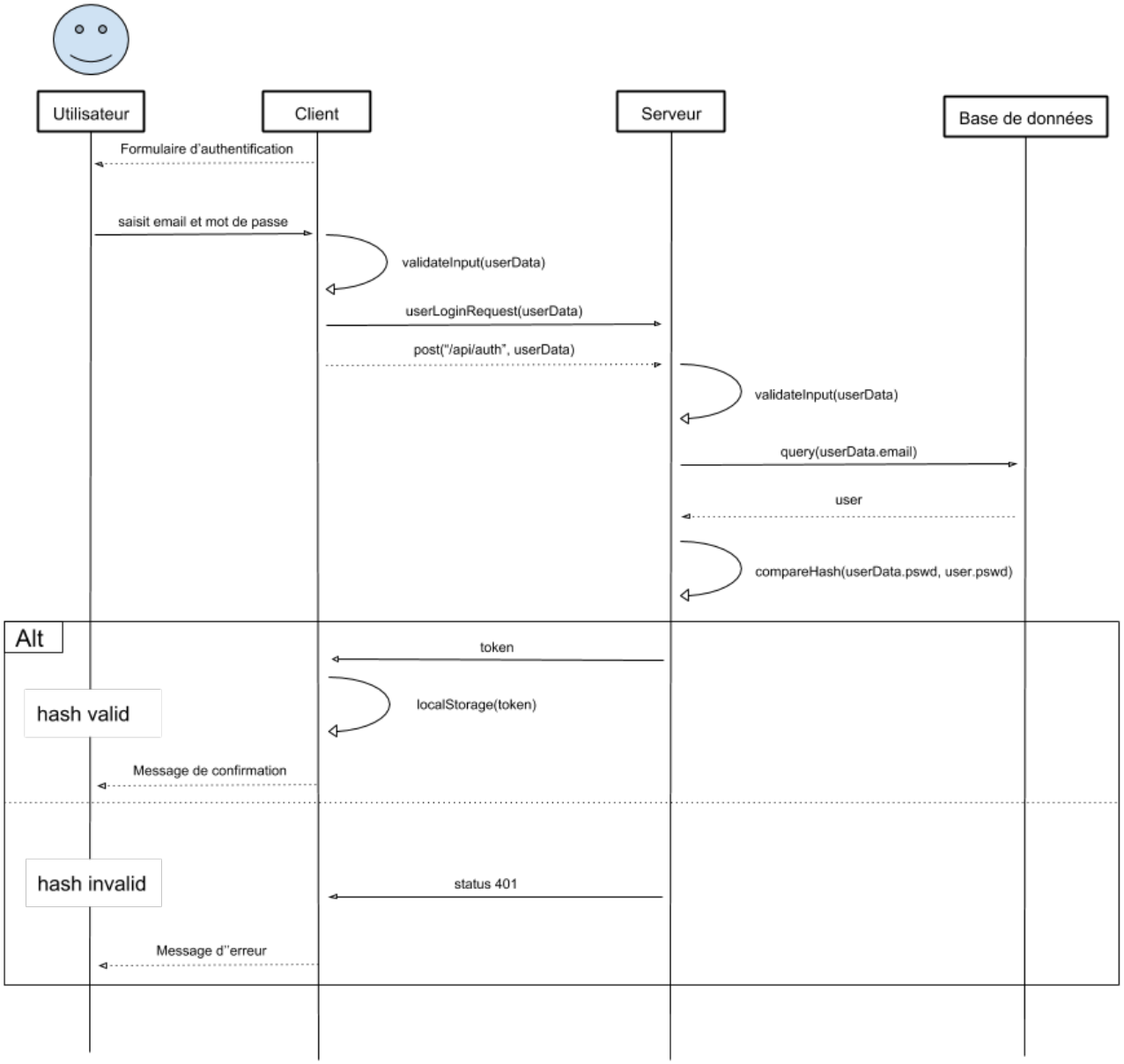


FIGURE 4.1 – Diagramme de séquence de l'authentification

comparés et s'ils correspondent, un token d'authentification est retourné au client, enregistré dans la mémoire local, l'état de l'application est mis à jour et un message de confirmation est renvoyé à l'utilisateur. Si les hash ne correspondent pas, une erreur de statut 401, utilisateur non identifié, est retournée au client et l'utilisateur reçoit un message d'erreur.

Lors de chaque requête client (changement de route) l'authentification de l'utilisateur est vérifiée avec une fonction retournant le composant de la route si l'utilisateur est authentifié soit qui le redirige vers la page d'authentification. De plus, lors de chaque requête vers le serveur (requête http get/post), l'authentification de l'utilisateur est vérifiée via un middleware.

4.2.2 La gestion des utilisateurs

La gestion des utilisateurs est une fonctionnalité qui s'inscrit dans l'idée que les utilisateurs peuvent avoir accès aux données des autres utilisateurs et les changer pour telle ou telle raison. La figure 4.2 reprend le flux pour une opération de mise à jour, la plus technique des opérations CRUD.

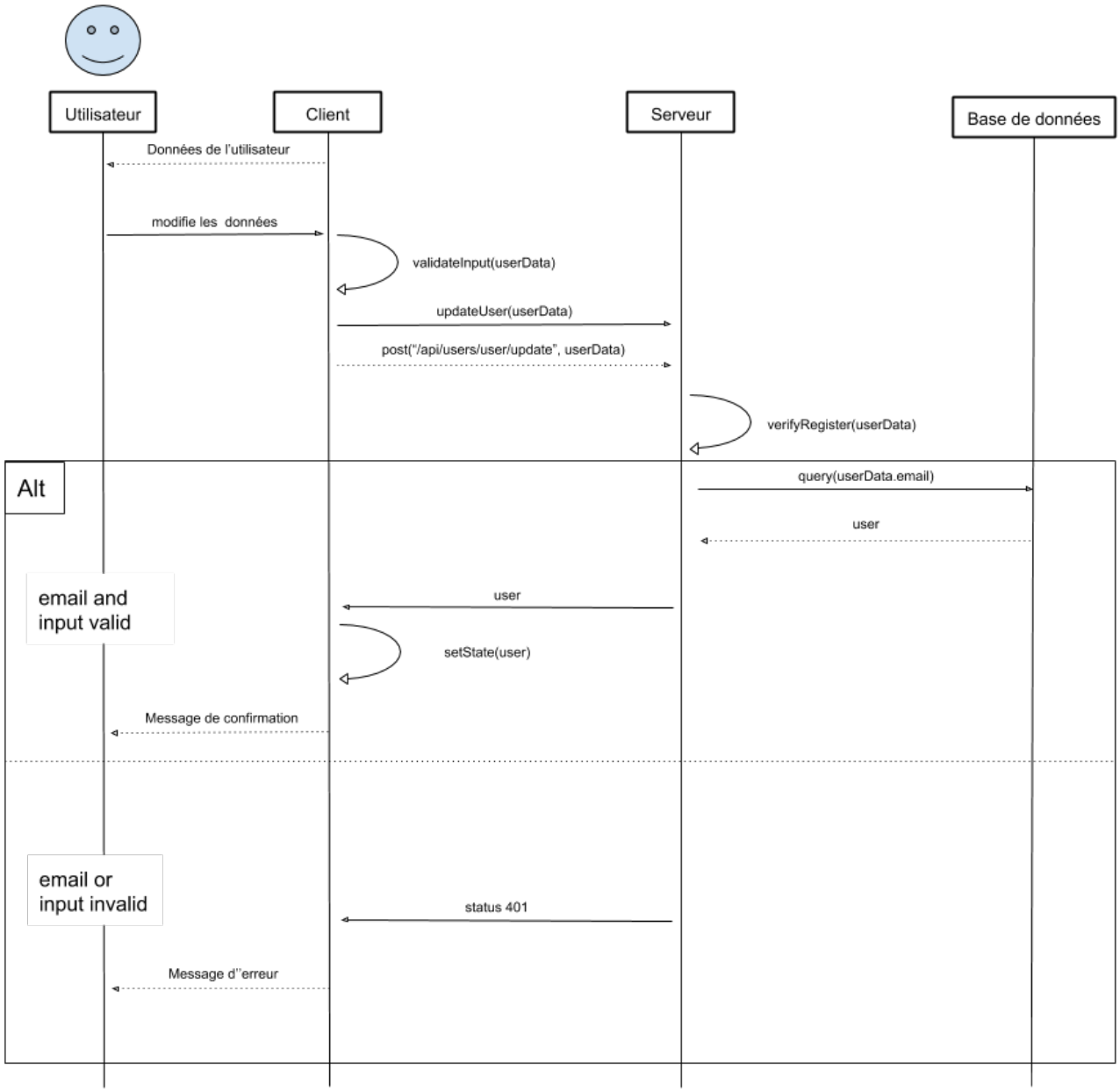


FIGURE 4.2 – Diagramme de séquence d'une mise à jour d'un utilisateur

Le flux jusque *verifyRegister(userData)* est similaire à celui de l'authentification 4.2.1. *verifyRegister(userData)* est une fonction de callback permettant de vérifier non seulement l'input de l'utilisateur, mais également si l'adresse mail rentrée n'existe pas déjà dans la base de données. Si l'adresse email est validée alors l'utilisateur est renvoyé vers le client qui met à jour l'état de son reducer correspondant (Redux, voir section 4.3.2). Autrement, une erreur 403 (accès refusé) est renvoyée et un message d'erreur s'affiche sur l'écran de l'utilisateur.

4.2.3 Le système d'abonnements

Le système d'abonnement permet aux utilisateurs de recevoir ou non des SMS de rapport et d'alerte des centres de santé. Un centre peut contenir plusieurs services monitorés (wifi, tablettes, réseau électrique, ...). Un utilisateur abonné à ce centre recevra les SMS de rapport du centre et un utilisateur abonné aux services du centre recevra en plus les SMS d'alerte. Ces informations sont modélisées sous la forme d'un extrait de schéma entité-relation dans la figure 4.3.

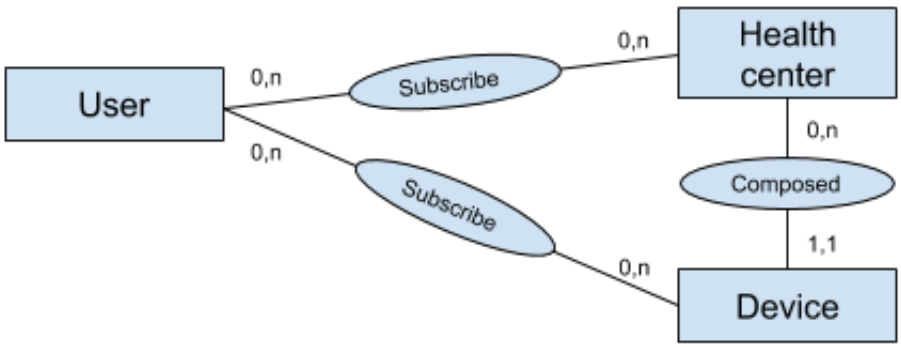


FIGURE 4.3 – Schéma du système d'abonnements

Il est à noter qu'un utilisateur ne peut pas s'abonner aux services du centre s'il n'est pas de base abonné au centre lui-même. De plus, ce schéma relationnel mène à la possibilité de gérer les abonnements soit à partir des centres de santé soit à partir des utilisateurs.

4.2.4 La visualisation des données

Plusieurs types de données sont à visualiser :

- Les alertes des centres
- Les retours à la normale
- Les centres de santé et leurs informations
- La localisation des centres
- Les données des services enregistrés dans les centres

Les alertes des centres et les retours à la normale sont visualisés sous forme de label, précisant si un problème est survenu dans un centre ou si tout est normal. Les centres de santé et leurs informations sont repris dans une section mise en avant et au début de la page. La localisation des centres est visible sur l'API de google maps sur la page principale. Les marqueurs en rouge indiquent les centres où un problème est survenu et les marqueurs en vert, les centres où tout est normal. Les données des services enregistrés dans les centres sont reportées sur des graphes et le type de graphe dépend du type de données. Si la donnée est binaire, le graphe est en barre sinon, il est linéaire.

Chacune de ces visualisations est interactive d'une manière ou d'une autre. Les alertes/retours à la normale des centres sont mises en relation avec les graphes des données correspondantes, les centres de santé et leurs informations peuvent à la fois être sélectionnés et filtrés en fonction des alertes, la localisation des centres peut être réarrangée dans l'espace grâce à l'API interactive de Google Maps et les données des services enregistrés dans les centres peuvent être filtrées grâce à une fonctionnalité de zoom. Cette philosophie rejoint les fonctionnalités explicitées dans la partie Etat de l'art, section 2.2.

4.2.5 Le système de permissions

Le système de permissions est un système où les utilisateurs possèdent un rôle et en fonction de ce rôle ils peuvent utiliser différentes fonctionnalités de l'application. Les rôles sont divisés en 4 catégories : technicien - responsable - admin - root. Le technicien peut consulter les données des différents centres auquel il est abonné, modifier ses données de profil, décider de recevoir ou non des SMS d'alerte. Le responsable peut consulter les données de tous les centres et modifier ses données de profil. L'admin peut, en plus des droits propres au technicien et au responsable, décider de recevoir ou non des SMS journaliers, créer, mettre à jour ou même supprimer un utilisateur et gérer les abonnements des différents utilisateurs. Le root possède les mêmes droits que l'admin mais est impossible à supprimer de la base de données. Ces rôles sont donc représentables de manière pyramidale, chaque niveau équivalent à des droits supplémentaires.

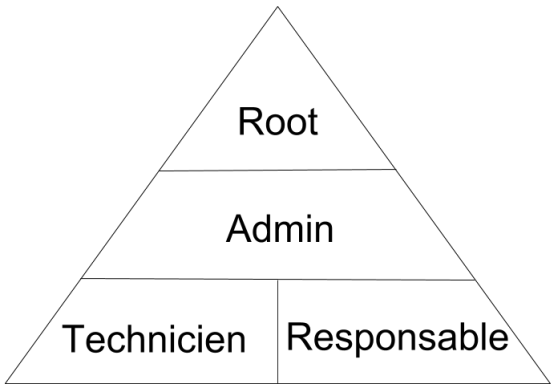


FIGURE 4.4 – Pyramide des permissions

Ces permissions sont vérifiées de la même manière que pour l'authentification à savoir ; lors de requêtes clients demandant certains privilèges (changement de route, typiquement pour aller dans les onglets réservés aux admin) les permissions sont vérifiées avec une fonction retournant le composant de la route si l'utilisateur possède les permissions requises soit qui le redirige vers la page d'accueil. De plus, lors de requêtes demandant certains privilèges vers le serveur (requête http get/post), les permissions de l'utilisateur sont vérifiées via un middleware.

4.2.6 Le traitement des SMS de monitoring

Lorsqu'un SMS est réceptionné par l'application web, il est tout d'abord traité en fonction du serveur SMS utilisé. En effet les différentes API des fournisseurs n'encodent pas toujours l'information de la même manière et ces derniers doivent donc être formatés. Les SMS ont alors le format suivant :

```
1        {
2            id : sms.SmsSid,
3            body : sms.Body,
4            from : sms.From,
5            to : sms.To
6        }
```

Le reste du traitement sur le SMS se fera sur le *body*. Une fois ce SMS formaté, il est parsé en fonction des délimiteurs représentés par les caractères "&&". Le format du *body* des SMS est simple. Il se compose tout d'abord de l'action à réaliser (i.e : create - update - delete) et des données sur lesquelles réaliser l'action. Si les données sont de type "data" alors cela signifie qu'elle représente un SMS d'alerte, de retour à la normale ou de rapport. Dans ces trois cas, les abonnés au centre et aux services doivent être prévenus et le serveur SMS sert donc de relais. Dans le cas d'une alerte ou d'un retour à la normale, seuls les abonnés aux services en question doivent être prévenus. Si c'est un SMS de rapport alors tous les abonnés au centre doivent être prévenus.

Le diagramme logique de la figure 4.5 reprend le raisonnement expliqué plus haut. Les fonctions "Abonnés services" et "Abonnés centre" représentent l'envoi de SMS aux abonnés en question.

4.3 Implémentation

4.3.1 Architecture

L'application a été découpée en plusieurs packages de sorte à découpler les différentes parties du stack et pour faciliter la maintenance du code. Ainsi, l'application a été divisée en trois parties :

- Le client, package **client**
- Le serveur, package **server**
- La base de données, package **database**

Chaque package est subdivisé de manière identique. On retrouve les sources sous le dossier **src**, le code compilé sous le dossier **build** (voir section suivante 4.3.2), les packages utilisés sous le dossier **node_modules**, le style (quand il y en a) sous le dossier **style** et les dépendances liées aux images dans le dossier **assets**. La figure 4.6 reprend une découpe en arbre de l'application.

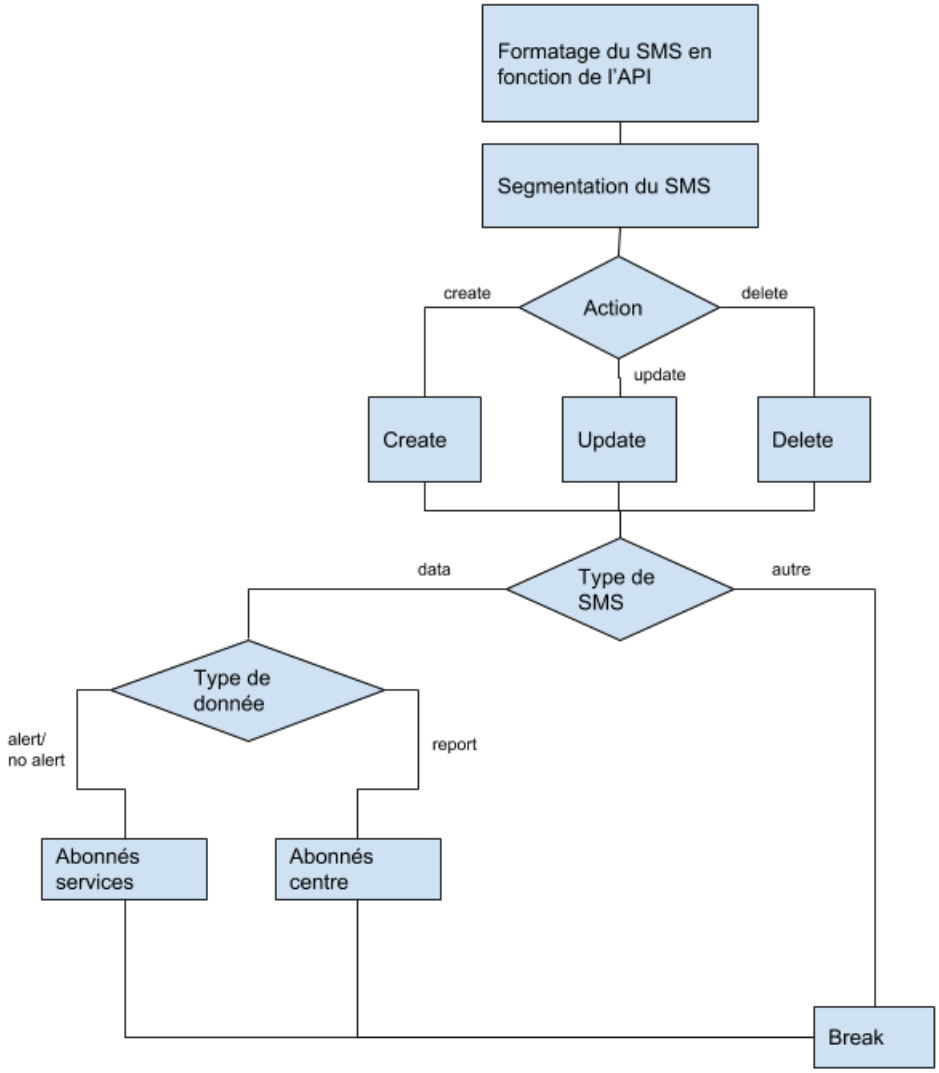


FIGURE 4.5 – Diagramme logique du traitement des SMS de monitoring

Le package **database** permet d'initialiser les données dans la base de données. Ce package a pour principal but d'insérer un premier utilisateur dans la base de données et de permettre de réaliser des tests avec des données fictives suivant un schéma particulier. Les packages **client** et **server** peuvent être utilisés sous deux modes. Soit en mode développement auquel cas un serveur de développement, permettant de détecter dès qu'une modification dans le code est réalisée, tournera pour chacun et un proxy permet de faire le lien entre les requêtes du client et les réponses du serveur. Soit en mode production pour lequel un fichier bundle du client est directement injecté dans les routes proposées par le serveur. Les différences entre ces deux modes et les bibliothèques impliquées sont explicités dans la section suivante.

La base de données a été pensée de sorte à être le plus modulable possible tout en restant facilement compréhensible. Etant donné que la base de données en question est une base de données noSql et que chaque information est stockée sous forme de document dans Couchbase, un type est attribué à chaque document. Le schéma entité-relation est repris à la figure 4.7.

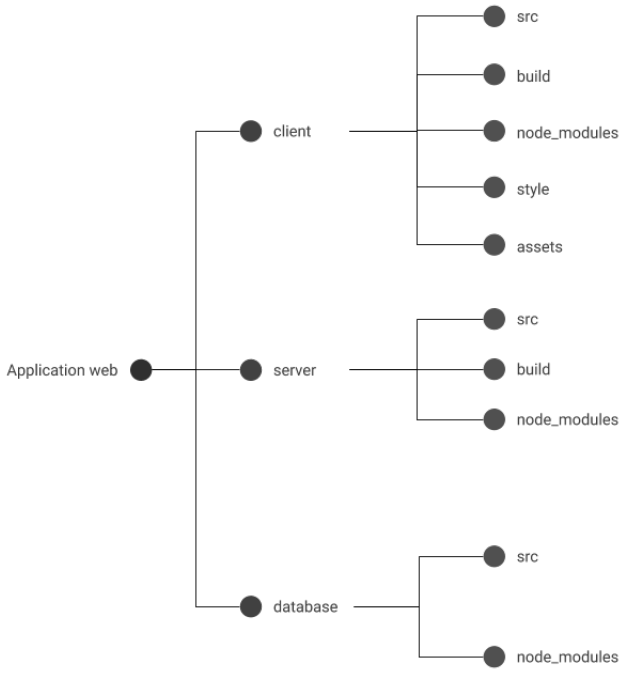


FIGURE 4.6 – Structure de l'application web

4.3.2 Bibliothèques utilisées

L'utilisation de node.js et de son gestionnaire de paquets est une pratique courante et mène souvent à l'utilisation d'un grand nombre de paquets différents lors de la conception d'une application web. Cependant, certain paquets sont plus notables que d'autres puisqu'ils impliquent certaines configurations et/ou certaines manières de coder. Pour la conception de cette application web, les paquets suivants sont à noter :

- **webpack** permet de compresser du code javascript en un seul ou plusieurs paquet(s). Quand webpack traite l'application, il va construire un graphe de dépendance et lier chaque module dont le projet à besoin pour générer ce ou ces paquet(s). C'est une bibliothèque typiquement utilisée côté client pour optimiser le temps de chargement de l'application lorsqu'elle n'est pas générée côté serveur. Il est intéressant de distinguer le mode développement du mode production. Le mode développement met à disposition un serveur de développement qui va remarquer chaque modification faite au code et recompiler ce dernier. Avoir un serveur de développement pour le client implique l'utilisation d'un proxy pour rediriger les requêtes du client vers le serveur. Le mode production propose un plus grand nombre de plugin à appliquer et donc un temps de compilation plus important, raison pour laquelle ce mode n'est utilisé que lorsque l'application doit être mise en ligne. La différence entre les deux modes s'applique aussi au serveur puisque lors du mode production, le fichier *bundle* généré pour le client sera directement injecté dans les routes du serveur.

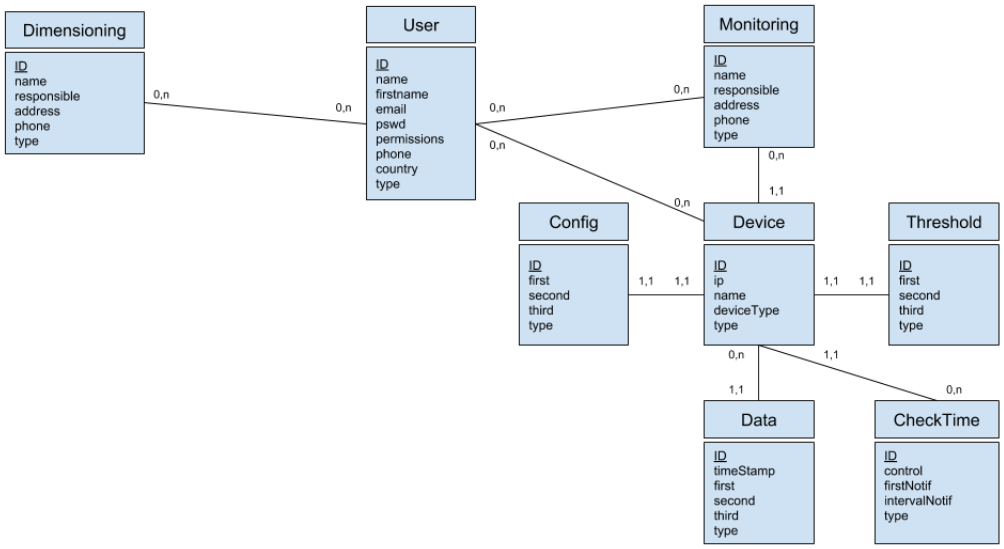


FIGURE 4.7 – Schéma relationnel de la base de données pour l'application web

- **typescript** est un sur-ensemble de Javascript qui transcompile son code en Javascript. Utiliser Typescript permet d'améliorer et de sécuriser la production de code Javascript. Il permet un typage statique optionnel des variables et des fonctions, la création de classes et d'interfaces, l'import de modules, tout en conservant l'approche non-contraignante de JavaScript. Au sein de l'application, le code est transcompilé dans le dossier build.
- **redux** est une bibliothèque frontend utilisée pour manipuler l'état de l'application. C'est une librairie compliquée dans sa conception qui tente de mixer 2 concepts, la mutation et l'asynchronisme, mais ces concepts ne seront pas détaillés ici. Cette bibliothèque est basée sur 3 principes [24] : L'état, le state, de l'application est stocké dans un arbre d'objets au sein d'un objet appelé store - La seule manière de changer l'état de l'application est d'émettre une action, c'est-à-dire une application décrivant ce qu'il se passe - Pour spécifier comment l'état est transformé par les actions, il faut écrire des reducers, c'est à dire des fonctions qui acceptent une accumulation et une valeur et qui retourne une nouvelle accumulation [25].

4.3.3 Mise en production

La mise en production de cette application web et de ses dépendances doit être séparée en 3 aspects : l'application web - le serveur SMS - la base de données. L'application est accessible via l'adresse ci-dessous. De plus, des captures d'écran de la page regroupant les centres de monitoring et de la page de gestion des abonnements sont représentées à la figure 4.8

http://34.252.56.151

L'application web a été installée sur les serveurs d'Amazon Web Service (AWS). Le serveur choisi pour cette version de l'application se situe en Irlande. Etant donné que les premiers utilisateurs de l'application sont les superviseurs du projet chez AEDES, l'emplacement choisi dépend de la vitesse de réponse entre le serveur et Bruxelles. Ci-dessous le tableau 4.3 reprend les temps de réponse des différents serveurs pour le service EC2 utilisé [26] :

Région AWS	**Ping HTTP**
Virginie du Nord	205ms
Oregon	203ms
Californie du Nord	191ms
Francfort	37ms
Irlande	35ms
Londres	42ms
Singapour	305ms
Tokyo	304ms
São Paulo	310ms
Australie	409ms
Bombay	245ms

TABLE 4.3 – Temps de latence moyen (sur 1000 essais) pour les services AWS [26]

Le serveur SMS est fourni par l'entreprise Twilio. Le serveur SMS devrait idéalement se situer en République Démocratique du Congo pour limiter les coûts d'envoi de SMS cependant très peu d'entreprises proposent ces services dans cette région ou sont restés muets face aux requêtes. De ce fait, un numéro international de Twilio est utilisé. Ce numéro se situe aux Etats-Unis, car ils offrent le plus de fonctionnalités et ne diffèrent pas en prix des autres numéros internationaux. De plus le forfait d'envoi d'un SMS de la RDC vers les numéros internationaux proposés ne diffère pas de la localisation (0,28€/SMS) [27].

La base de données Couchbase est située sur les serveurs d'AEDES, c'est à dire à Bruxelles. Les requêtes de l'application web en Irlande vers la base de données à Bruxelles sont sécurisées par le protocole SSL ce qui ajoute une couche de protection mais impacte le temps de réponse (de l'ordre de 750ms).

Les différentes étapes pour mettre un site en ligne sur les services EC2 d'AWS et la configuration des routes et de l'accès à l'API de Twilio sont reprises dans les annexes B.1 et B.3 respectivement.

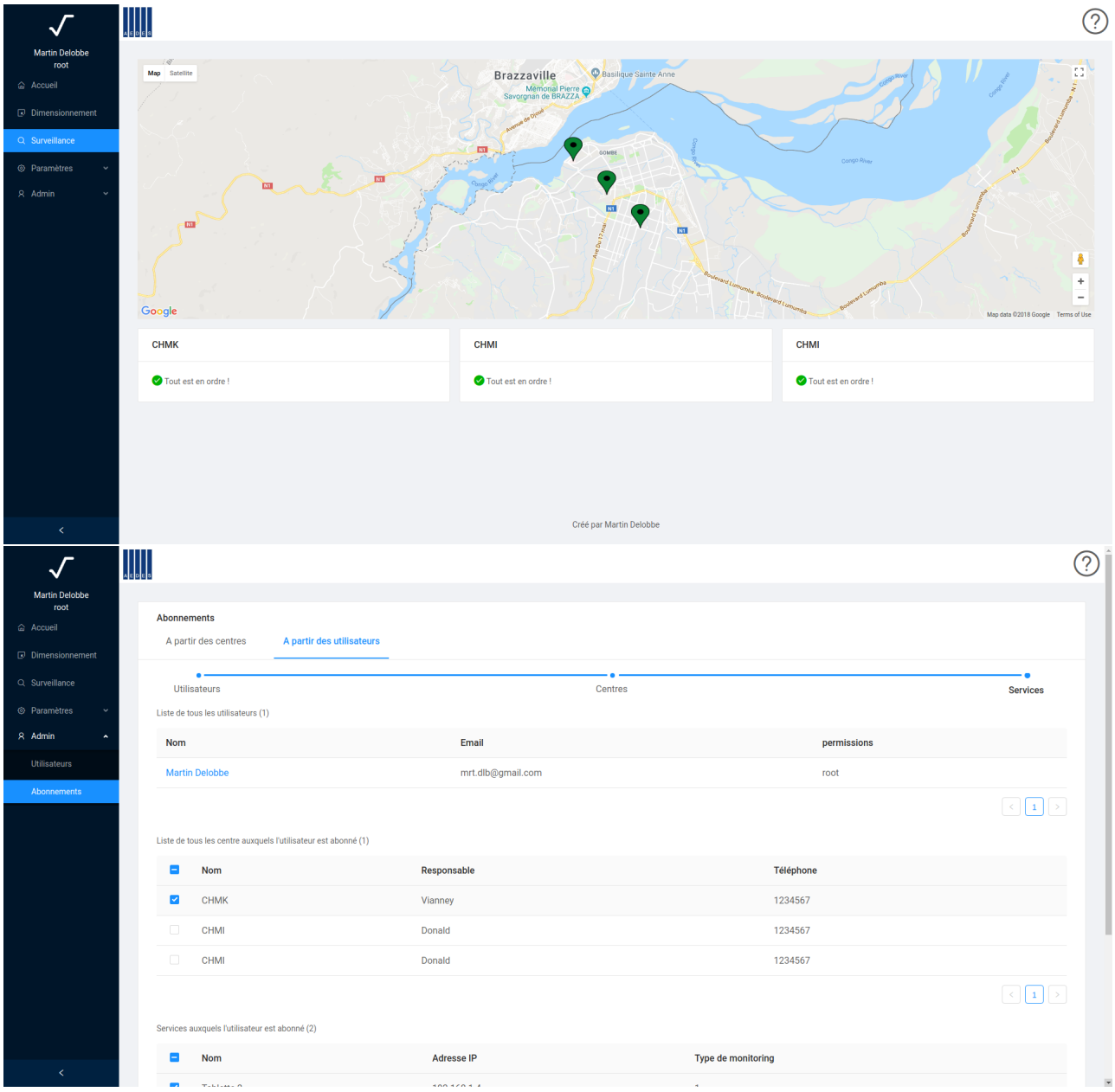


FIGURE 4.8 – Onglets montrant le monitoring des centres et la gestion des abonnements respectivement

Chapitre 5

Application de monitoring

L'application développée l'année précédente est une application Android, écrite en Java et réalisée avec l'IDE Android Studio. Cette application pouvait fonctionner de manière autonome, mais certains bugs ont été relevés par l'équipe des partenaires locaux, la base de données manquait de modularité et les services en arrière-plan ne tournaient que sous certaines conditions (écran allumé). Cette application étant le point de départ du système il était important de résoudre ces bugs et limitations.

5.1 Changements techniques

Pour une description plus détaillée de l'application réalisée l'année dernière, se référer au rapport de projet ainsi qu'au mode d'emploi repris en annexe A.1.

5.1.1 Design

Le choix de l'application Android pour le système de monitoring était entre autres motivé par la facilité d'utilisation de ce genre de technologie. Le design est donc un point important de cette solution et les changements, mineurs, entre les différentes versions sont illustrés ci-dessous. La figure 5.1 reprend les captures d'écran de la première version et la figure 5.2 reprend les captures d'écran de la seconde version.

L'écran d'accueil ne contenait que deux onglets tandis qu'aujourd'hui il en contient trois pour faciliter la gestion des abonnements des utilisateurs. Encoder un utilisateur nécessite aujourd'hui plus d'informations et une activité est donc consacrée à cet effet. Enfin, encoder un centre hospitalier a pris plus d'importance puisque ces données doivent être envoyées à l'application web. C'est la raison pour laquelle une activité est consacrée à cet encodage et les informations sont accessibles dans le troisième onglet (dernière image de la figure 5.2).

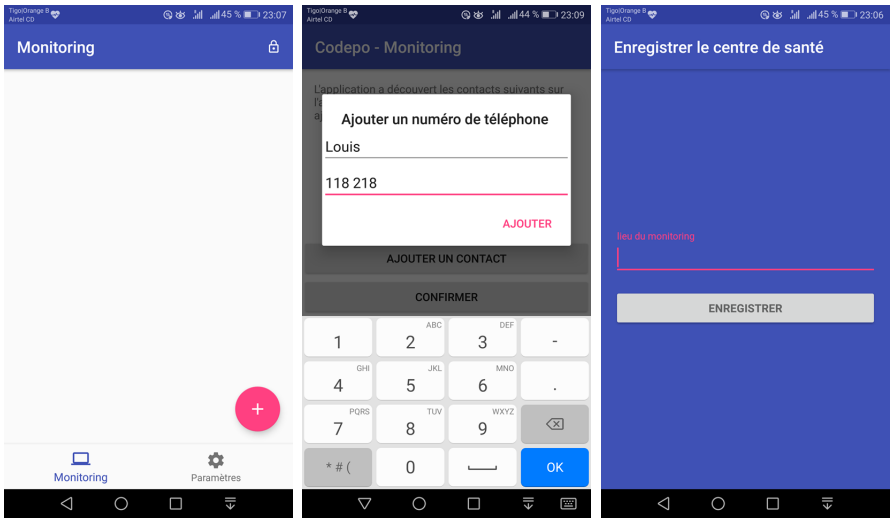


FIGURE 5.1 – Captures d'écran de la première version

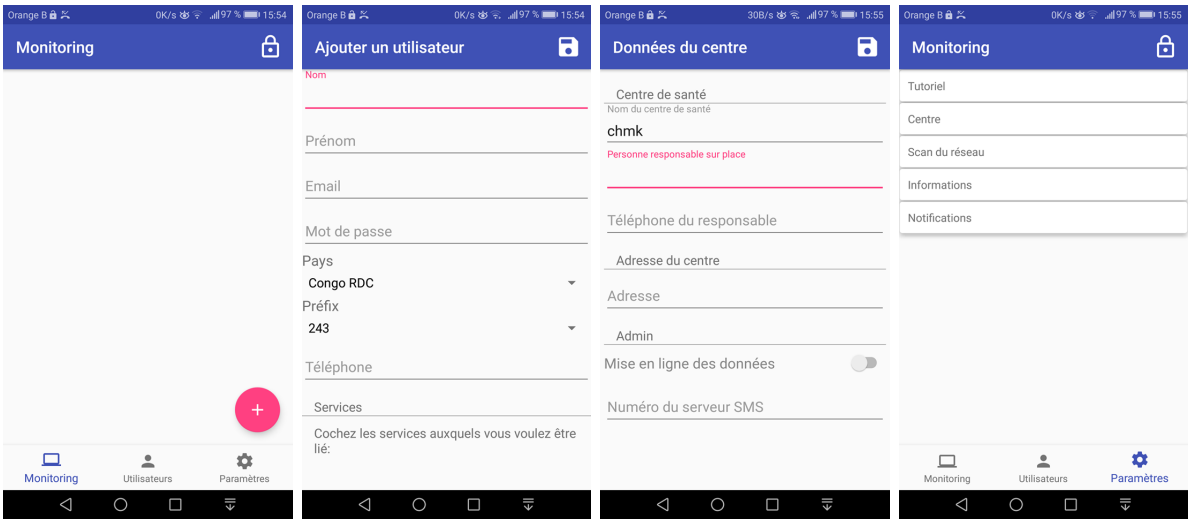


FIGURE 5.2 – Captures d'écran de la seconde version

5.1.2 Base de données

Le schéma de la base de données mise en place lors de la première version est illustré à la figure 5.3 . Cette architecture ne permettait que difficilement l'ajout d'un service et n'était pas modulable. De plus, afin de simplifier les mises à jour entre l'application Android et l'application Web, il était important de réaliser un schéma de la base de données le plus similaire possible à celui imaginé pour l'application Web (voir figure 4.7). Ainsi, rajouter des données sur l'application web revient à envoyer un fichier JSON de l'application Android directement avec les données de la base de données sans (presque) aucun formatage entre.

Les services ont été rendus plus génériques et la différence entre eux est établie avec un champ "deviceType". Les tables "PhoneNumbers" et "PhoneNumber" ont été remplacées par la table "User" et les tables "Config", "Threshold" et "Data" permettent de rajouter des caractéristiques propres à chaque type de service enregistré (voir figure 5.4).

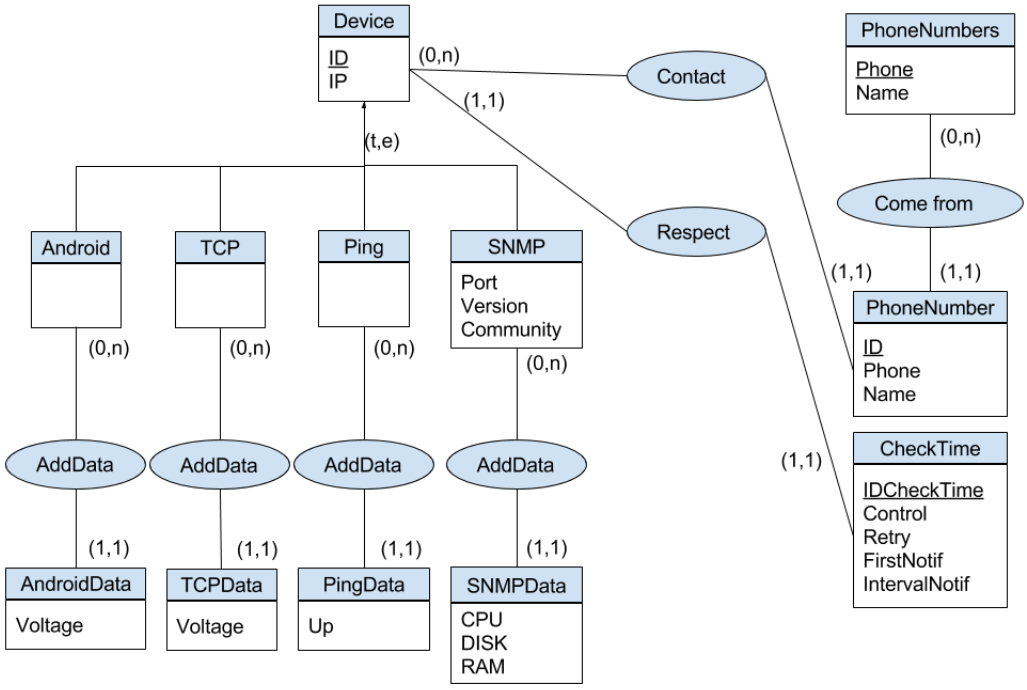


FIGURE 5.3 – Base de données de la première version de l'application Android

5.1.3 Fonctions en arrière-plan

Les fonctions en arrière-plan lors de la première version de l'application ne pouvaient tourner qu'avec l'écran allumé ce qui impliquait une forte consommation de la batterie. En effet, les fonctions en arrière-plan étaient gérées par des Thread directement lancés dans l'activité principale. L'architecture d'Android, qui vise à toujours optimiser l'utilisation de la batterie, arrête les activités à partir d'un certain temps durant lequel il n'y a pas eu d'interaction avec l'utilisateur ce qui n'est pas souhaitable pour une application de monitoring. L'implémentation de composants d'Android plus haut niveau a donc été nécessaire afin de pallier à ce problème. C'est la raison pour laquelle les Services d'Android ont été utilisés dans un premier temps. Cependant, il est plus adapté encore d'utiliser des AlarmManager dans le cas d'applications où certaines actions doivent être exécutées de manière répétitive et séparée d'une certaine durée ce qui est typiquement le cas pour les vérifications faites sur le système.

5.2 Implémentation

5.2.1 Architecture

L'application a une structure similaire à tout projet Android, à savoir un dossier java contenant le code du projet, un dossier res avec les différentes vues de l'application et un fichier manifest avec les

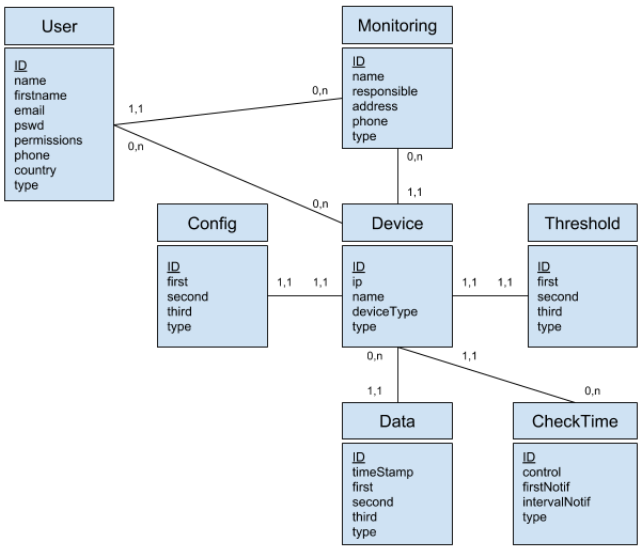


FIGURE 5.4 – Base de données de la seconde version de l'application Android

déclaration de tous les composants et permissions Android. Le dossier java est découpé en plusieurs packages pour faciliter la maintenance du code. Le package activity contient toutes les activités et fragments de l'application, le package adapter contient tous les Adappter et ViewHolder des listes présentes dans l'application. le package database contient deux sous-package ; DAO et model qui représentent respectivement toutes les interactions avec les tables et le schéma des tables en question. Enfin, le package monitoring contient deux sous packages ; devices et sms qui représentent respectivement la collecte de données pour les différents services enregistrés et l'envoi de SMS en cas d'alerte/retour à la normal/rapport. Certains fichiers utiles pour certaines fonctionnalités de l'application n'appartiennent à aucun de ces packages, mais ne sont pas pertinent pour la compréhension du code. La figure 5.5 représente la découpe du code explicitée précédemment.

5.2.2 Mise en production

La mise en production de l'application n'a pas été réalisée sur le Play Store dû à certaines conclusions tirées après l'implémentation sur le terrain (voir chapitre 6). En effet, les limitations observées dans le système Android quant à sa gestion du travail en arrière-plan remettent en cause la robustesse et la pérennité de cette solution. Cependant, le plugin Fabric a été utilisé afin de partager l'application aux testeurs et récolter les logs relatifs au fonctionnement. Fabric est un outil de monitoring pour application qui permet de récolter un panel d'informations sur l'utilisation d'une application. Une capture d'écran du dashboard principal est illustrée à la figure 5.6.

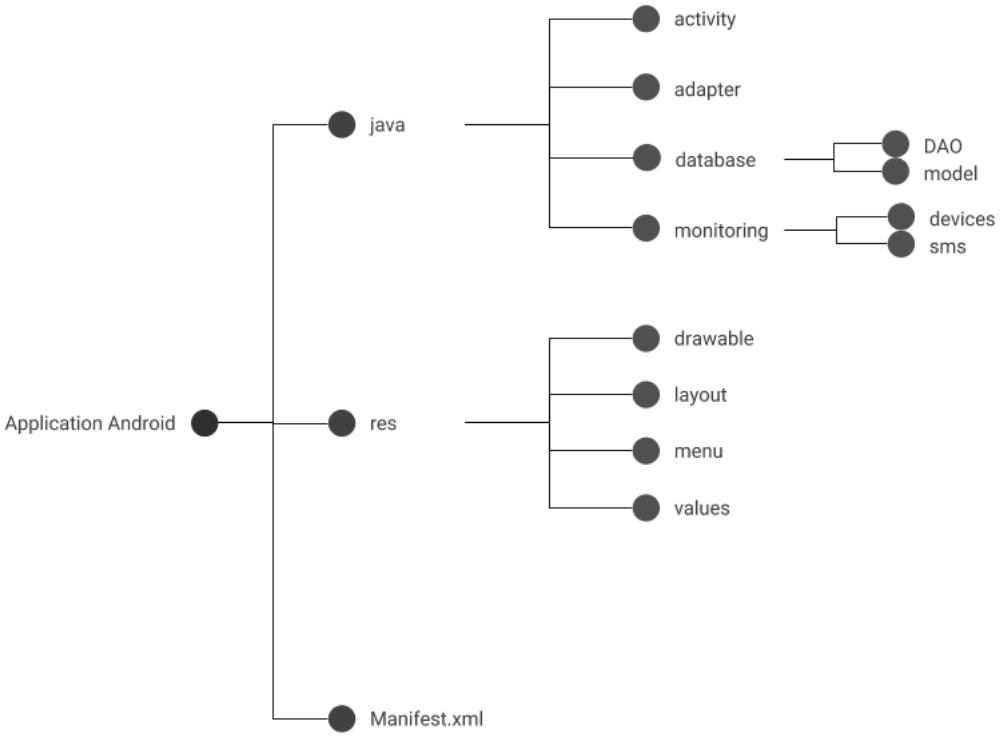


FIGURE 5.5 – Structure de l'application Android

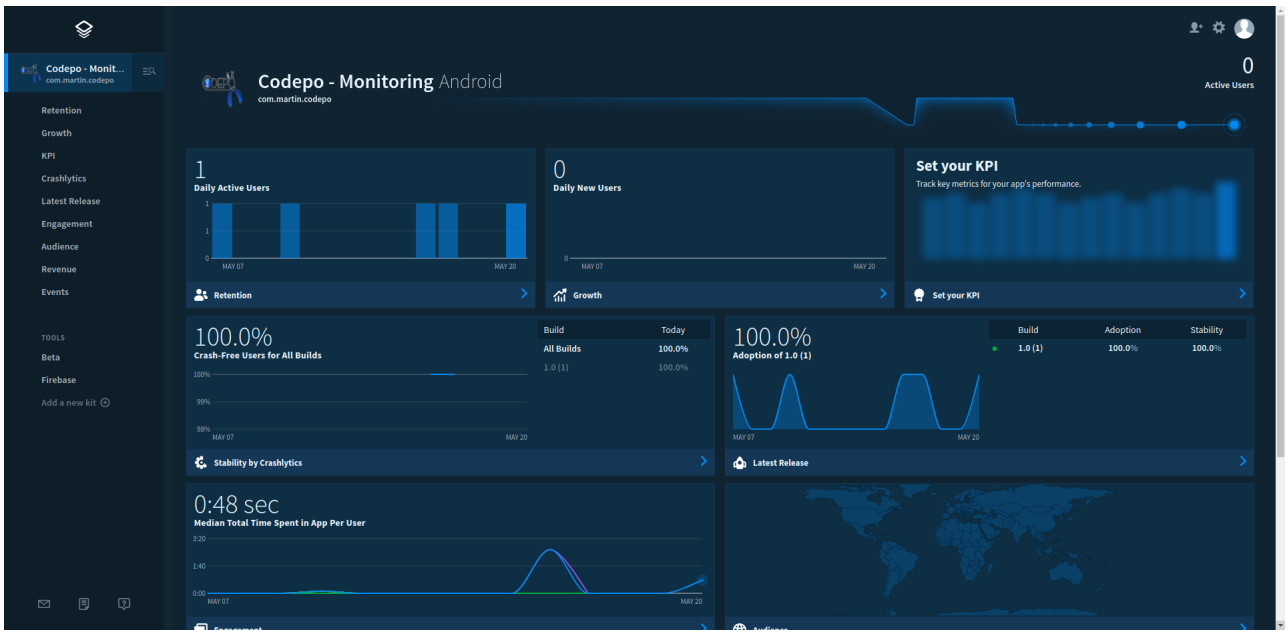


FIGURE 5.6 – Capture d'écran du dashboard de Fabric

Chapitre 6

Implémentation sur le terrain

La composante coopération au développement de ce mémoire impliquait la réalisation d'une mission à Kinshasa, en République Démocratique du Congo. Cette mission s'est déroulée du 2 avril au 17 avril 2018 et comptait plusieurs objectifs :

- Installation des systèmes de monitoring de l'année précédente
- Formation des partenaires locaux à l'installation des systèmes de monitoring
- Récolte des feedbacks nécessaires à la poursuite du mémoire
- Formation des partenaires à l'utilisation de l'application web

Les différents objectifs ont été répartis sur la durée du séjour et ont été séparés en fonction des deux grosses composantes du projet à savoir l'application de monitoring et l'application web. Des constats ont pu être tirés et ces derniers sont explicités dans les sections suivantes.

6.1 Organisation

L'organisation de la mission s'est fait sur base de l'ordre de mission repris dans les annexes C. Partant de ce document, un Gantt a été mis en place pour suivre l'avancée de la mission. Le Gantt de début de mission est représenté par la figure C.1 tandis que le Gantt de fin de mission est représenté par la figure C.2. Les retards de début de mission sont expliqués par le temps qu'il a fallu pour que les valises et le matériel qu'elles contenaient arrivent, l'installation des systèmes de monitoring dans les centres de santé étant impossible sans. L'application de monitoring ayant demandé beaucoup de développement supplémentaire pour tenter de pallier à certains problèmes, cette dernière a pris du retard et les améliorations sur cette application ont pris plus de temps que prévu (voir section 6.2 pour plus d'informations à ce sujet). La mise en production de l'application web a également pris plus de temps que prévu, car certaines modifications ont dû être apportées pour satisfaire les besoins des partenaires locaux.

6.2 Application de monitoring

Le système de monitoring devait être installé dans trois centres de santé/hôpitaux différents. Cet objectif a été atteint, mais il a fallu noter que la version des cartes électroniques fournies par l'ULB était obsolète et les pin de shield ethernet emboîtés sur les arduino ne permettaient pas de connecter lesdites cartes électroniques, l'espace pour les pin étant trop étroit. De ce fait, seule la partie logicielle du système de monitoring a pu être installée et prise en main par les partenaires. Cette limitation a eu comme impact que seuls les éléments suivants pouvaient être surveillés au lieu des sept prévus à l'origine dans l'application : le wifi - la présence des éléments sur le réseau - l'état du serveur.

Lors de la mission sur place, un employé de la société MaisOrdi, partenaire d'AEDES, a été désigné pour prendre en charge l'installation des systèmes de monitoring. C'est ainsi qu'en complément au wiki proposé sur la plateforme ci-dessous, l'employé a rédigé un carnet de notes reprenant les différentes étapes de l'installation du système. Le mode d'emploi du wiki est disponible en annexe A.1.

http://cerhis.info/doku.php/start

Certaines limitations du système ont été relevées lors de l'implémentation sur le terrain et ces dernières sont reprises dans la section 7.2.1.

6.3 Application web

L'application web devait pouvoir être prise en main par les partenaires locaux et mise en production lors de la mission. La pris en main de l'application s'est réalisée en deux temps.

Dans un premier temps, une présentation des fonctionnalités implémentées a été présentée lors de l'arrivée à Kinshasa. Partant de cette réunion, des feedbacks ont pu être collectés concernant l'utilisation et l'extension de certaines features. Ces feedbacks ont permis d'identifier de nouveaux rôles à ajouter dans l'application web ainsi qu'une fonctionnalité permettant de visualiser les centres de santé sur une carte. Une fois les ajouts terminés, une nouvelle version de l'application a été présentée en fin de mission et cette dernière a été validée par les membres de MaisOrdi.

Dans un second temps, la mise en production de l'application web était prévue pour la fin de parcours. Cependant, afin de minimiser les déploiements et au vu des changements encore en cours, il était préférable d'utiliser un tunnel proposé par l'outil ngrok. Faire tourner ngrok en local permet de se connecter aux services cloud qui acceptent le trafic sur une adresse publique et relaye ce trafic vers le processus de la machine qui elle-même redirige ce trafic à l'adresse spécifiée. Ainsi, les tests avec l'API de Twilio ont pu être effectués et il a pu être montré que la réception tout comme l'envoi de SMS transitant par la plateforme web étaient bien fonctionnels.

6.4 Résumé

La gestion d'une mission sur le terrain est une expérience très formatrice et permet de se confronter à des problématiques auxquelles un ingénieur doit faire face dans la vie de tous les jours. Il est toujours important de tout prévoir et cela, peu importe que le projet se déroule dans un pays émergent ou non. De plus, une phase de tests avant le départ est un élément essentiel pour la réussite de la mission. L'implémentation d'une solution doit être efficace et des plans de secours doivent être prévus en cas problème. Typiquement, lors du transfert vers un lieu de travail, il est important de prendre en compte le temps de déplacement ainsi que le possible trafic et ce, tout particulièrement dans une ville telle que Kinshasa. Enfin, des éléments perturbateurs tels que l'oubli d'une clé d'un cadenas par un membre de l'équipe doivent soit pouvoir être évités par la planification soit résolu avec une certaine dose de débrouille.

Les feedbacks récoltés lors de la mission ont permis de finaliser le développement de l'application web, de former les techniciens à l'installation du système et à son utilisation et ont participé à consolider les liens existant entre AEDES et MaisOrdi. Le système est jusqu'à aujourd'hui toujours en fonctionnement sur place et des échanges continus sont en cours pour recueillir des rapports sur l'utilisation de la solution.

Chapitre 7

Résultats

7.1 Bilan

Si l'on se réfère à la section 3.5 du chapitre traitant du cahier des charges, l'objectif fonctionnel de ce mémoire était d'obtenir un MVB composé d'un système de centralisation et de visualisation de données de monitoring ainsi qu'une application de monitoring fonctionnelle et cet objectif est atteint. Les différentes fonctionnalités reprises dans les trois premiers niveaux de la pyramide des permissions, figure 3.3, ont été implémentées résultant dans l'architecture de la figure 7.1. Les SMS sont acheminés de la source à l'embouchure permettant de garder les données à jour sur l'application web (i.e : lorsqu'un centre est enregistré dans l'application Android, les données sont automatiquement envoyées sur la plateforme web) et de prévenir les personnes abonnées aux flux de SMS (chemin (**1**) (**2**) (**3**) (**4**)). Le système peut également être utilisé sans l'application web lors des phases de tests le chemin des SMS se réduisant à (**1**) (**4**).

7.2 Chiffres

7.2.1 Application Android

Les résultats des tests générés lors de l'implémentation de l'application Android ont été collectés sous deux formes :

- Les logs de Fabric.io
- Les données de monitoring

Les bugs de l'application Android ont été enregistrés dans Fabric.io et des alertes par mail permettaient prévenir lorsque l'application n'était plus en état de fonctionner. Une alerte de ce type est reprise dans la figure 7.2. Les logs enregistrés dans Fabric.io permettaient de retracer l'origine

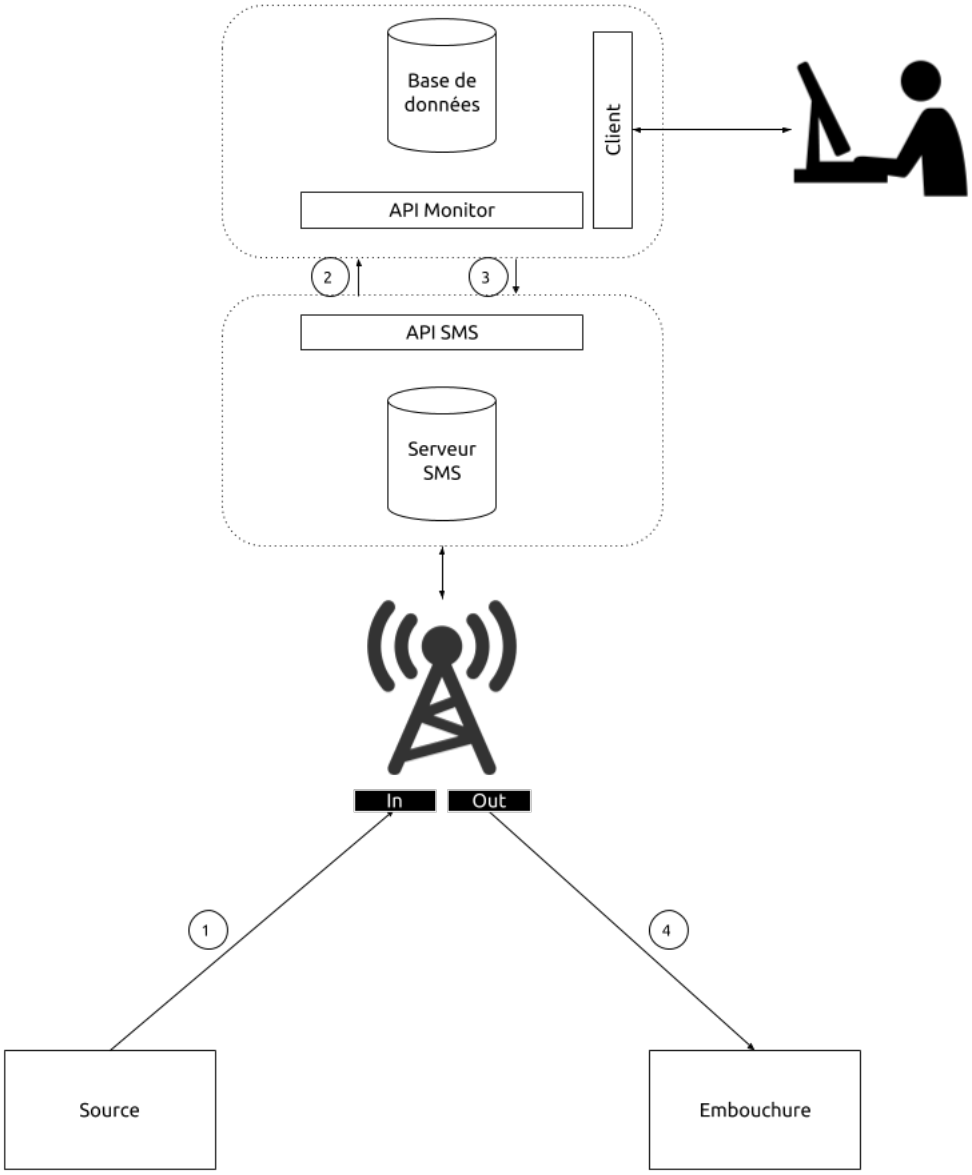


FIGURE 7.1 – Architecture du système développé

du problème et de gérer l'erreur en conséquence. La figure 7.3 reprend une description de l'état du système lorsque l'erreur est survenue dans l'application Android.

Les données de monitoring enregistrées dans l'application de monitoring ont permis de souligner plusieurs limitations du système Android. Premièrement, lorsque le smartphone se trouve verrouillé, débranché et non en mouvement depuis 1h, le système Android va entrer un mode particulier introduit à partir d'Android Marshmallow (API 23). Ce mode est le Doze mode qui ne va autoriser les applications à utiliser des services en arrière-plan que selon certaines fenêtres temporelles appelées fenêtres de maintenance (voir figure 7.4). Certains mécanismes permettent de contourner en partie l'intervalle entre les fenêtres de maintenance, mais il n'est pas possible de passer outre ce mode et l'intervalle minimum entre les fenêtres de maintenance est de 9 minutes. Ces données ont été obtenues de manière empirique et appuyées par de nombreux articles et discussions sur les forums professionnels.

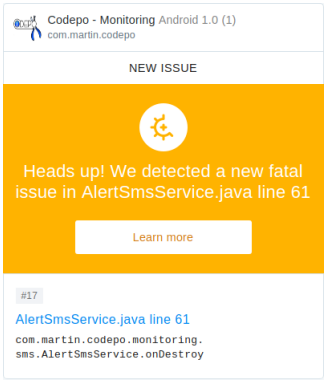


FIGURE 7.2 – Alerte par mail permettant de prévenir du crash de l'application

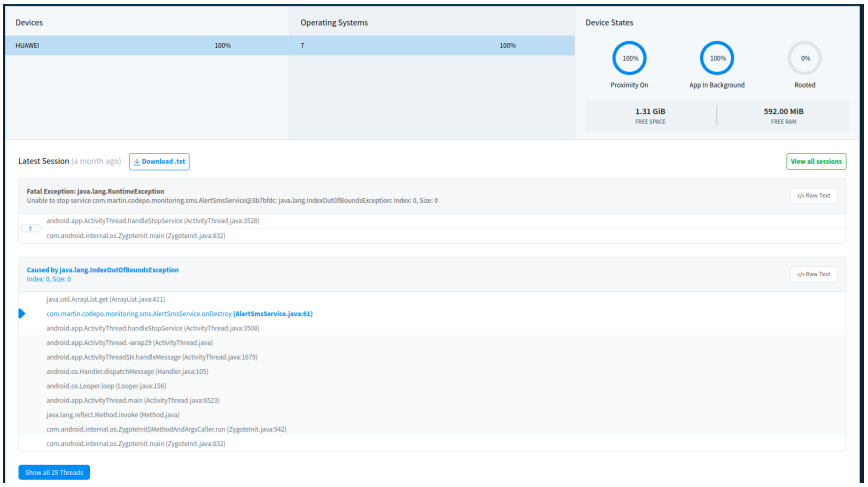


FIGURE 7.3 – Log de l'erreur découverte par Fabric.io

Afin de ne pas entrer dans le mode précédemment décrit, il est nécessaire de garder les smartphones de monitoring branchés à l'alimentation de CERHIS, l'optimisation de la batterie du smartphone devenant dès lors impossible. D'autres données de monitoring ont été récoltées dans cette configuration mais la régularité de l'échantillonnage laisse à désirer comme l'attestent les données du tableau 7.1 récoltées à Kinshasa lors de la surveillance du centre de santé Bolingo. L'échantillonnage était calibré pour avoir lieu toutes les 5 minutes. La raison du manque de régularité n'a pas été identifiée mais il est possible que cela soit lié à d'autres processus d'optimisation de la batterie propres à Android.

En fin de compte, il a été conseillé de n'utiliser l'application Android de monitoring que pour générer des rapports quotidiens dont la précision est moins critique que pour les alertes. A termes il est également conseillé de pivoter vers une solution basée sur un Raspberry couplé à un Arduino avec un module SMS pour pallier aux processus d'optimisation de la batterie d'Android.

7.2.2 Application Web

Les fonctionnalités de l'application web étant très liées à l'expérience utilisateur, l'application a été soumise à 25 personnes de profils différents (voir figure 7.5) qui ont utilisé toutes les fonctionnalités

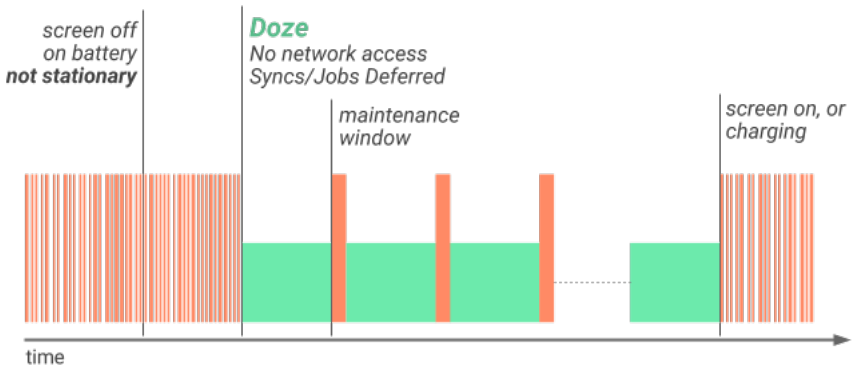


FIGURE 7.4 – Fenêtres de maintenance du Doze mode

Valeur du CPU du serveur	Moment de l'échantillonnage
20%	00 :59
18%	01 :06
17%	01 :13
19%	01 :18
17%	01 :21
18%	01 :30
20%	01 :42
23%	01 :45
17%	01 :50
16%	01 :56

TABLE 7.1 – Échantillonnage des valeurs du CPU du serveur au centre Bolingo

implémentées et les recommandations les plus récurrentes ont été ajoutées à l'application. Cette analyse a été réalisée afin de rendre l'application la plus facilement utilisable et pour permettre une plus grande pérennité du projet.

"Déconnexion en haut à droite" représente un bouton de déconnexion en haut à droite de l'application web, la déconnexion ne pouvant précédemment se faire qu'à partir de l'url de l'app. "Ajout d'un nouveau rôle" provient des demandes explicites des partenaires locaux lors de l'implémentation sur place. Un nouveau rôle pour les responsables d'hôpitaux a donc été imaginé. "Cartes" est une fonctionnalité permettant de visualiser graphiquement l'emplacement des différents centres hospitaliers. Cette fonctionnalité a été mise en place avec l'API de Google Maps. "Abonnement à partir des utilisateurs" représente la possibilité de générer les abonnements aux services et aux centres à partir des utilisateurs, cette fonctionnalité n'étant au début possible qu'à partir des centres. "Explications abonnements" était une demande des utilisateurs d'avoir une explication sur la manière de générer les abonnements aux centres et aux services. Cette fonctionnalité a été implémentée à l'aide d'une barre de progression décrivant l'étape suivante avant d'enregistrer les

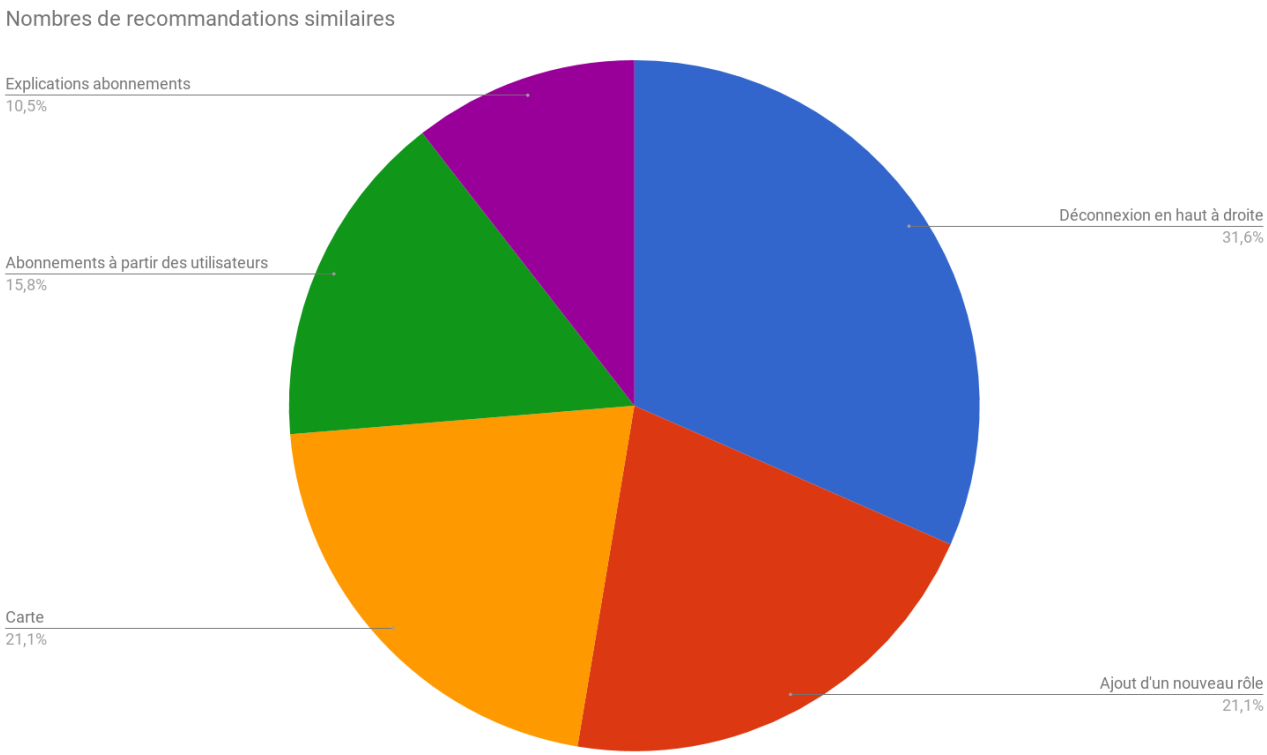


FIGURE 7.5 – Nombres de requêtes similaires sur un échantillon de 25 personnes

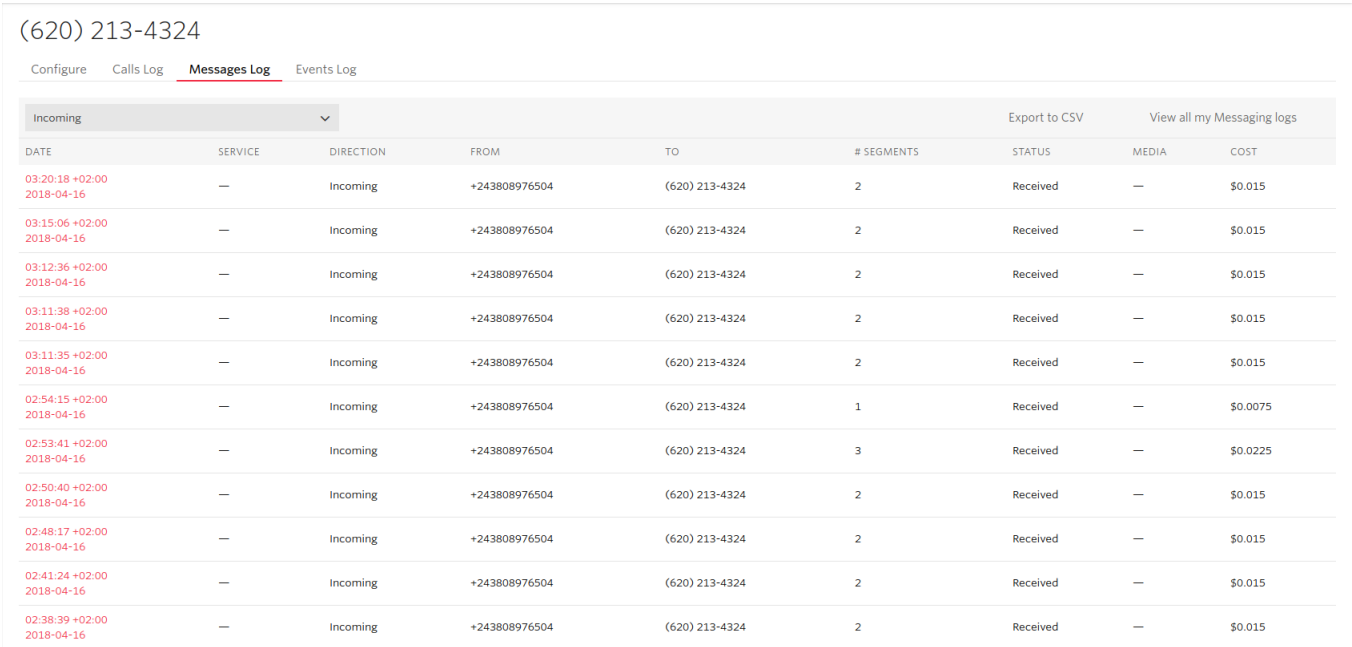
données.

En plus de l'expérience utilisateur, la gestion des abonnements impliquant l'intégration d'une API tierce, il a été très utile d'avoir accès à la console fournie par la compagnie Twilio pour avoir un aperçu des différents logs provenant des SMS reçus et envoyés (voir figure 7.6). Ainsi, les logs étaient centralisés et des alertes en cas de mauvais acheminement des SMS étaient envoyées par mail comme représenté dans la figure 7.7.

Au vu du bon fonctionnement du système de centralisation et de visualisation de données, il a été conseillé de migrer l'application web sur un serveur propre à AEDES et de chercher un partenariat à Kinshasa avec des fournisseurs de serveur SMS pour minimiser les coûts d'envoi de SMS. Les SMS de rapport étant les seuls SMS envoyés pour le moment afin de minimiser les coûts de cette solution.

7.3 Erreurs commises

Lors de l'implémentation sur le terrain, la partie électronique n'était pas fonctionnelle. La mission consistait en partie à former les techniciens locaux et ces problèmes ont empêché d'atteindre pleinement l'objectif. Ce mémoire englobant un grand nombre de compétences, il est important que tous les éléments soient fonctionnels lors de leur mise en production d'où l'importance de générer des environnements de tests pour vérifier le bon fonctionnement matériel et logiciel.



(620) 213-4324

Configure | Calls Log | Messages Log | Events Log

Incoming — Export to CSV — View all my Messaging logs

DATE	SERVICE	DIRECTION	FROM	TO	# SEGMENTS	STATUS	MEDIA	COST
03:20:18 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
03:15:06 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
03:12:36 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
03:11:38 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
03:11:35 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
02:54:15 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	1	Received	—	$0.0075
02:53:41 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	3	Received	—	$0.0225
02:50:40 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
02:48:17 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
02:41:24 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015
02:38:39 +02:00 2018-04-16	—	Incoming	+243808976504	(620) 213-4324	2	Received	—	$0.015

FIGURE 7.6 – Logs des SMS reçus/envoyés sur Twilio

Lors de la mise en production de l'application web sur les serveurs d'AWS, 2 erreurs ont été commises. La première a été d'installer la version de Couchbase Community sur une instance EC2 avec une mémoire trop faible. Cette erreur a ralenti la mise en production, car les connexions SSH avec l'instance étaient instables et se bloquaient souvent. La solution a consisté à délocaliser la base de données directement sur les serveurs d'AEDES où un bucket Couchbase a spécifiquement été créé pour le projet. La deuxième erreur est apparue lors des tentatives de connexion avec la base de données distante. Lors du développement de l'application web, la base de données utilisée était locale et tous les ports étaient ouverts, les query effectuées n'étant pas cryptées. Cela ne posait aucun problème et utiliser le package npm couchbase suffisait pour réaliser toutes les requêtes voulues. Cela n'était plus le cas avec le bucket d'AEDES. Il a donc fallu générer des requêtes personnalisées avec un client http tiers. Le client http utilisé est curl (package npm curlrequest) et grâce à l'abstraction qui avait déjà été faite sur les fonctions du package npm couchbase, les modifications ont été minimes puisqu'il n'a fallu changer le fonctionnement que d'une seule fonction.

7.4 Résumé

Les résultats obtenus lors du développement et de l'implémentation du système ont démontré que l'architecture imaginée permet son utilisation sous deux formes :

- **Hors ligne** où les SMS sont directement envoyés des centres de santé vers les personnes enregistrées dans l'application Android.
- **En ligne** où les SMS transitent tout d'abord par le serveur SMS avant d'être redistribués

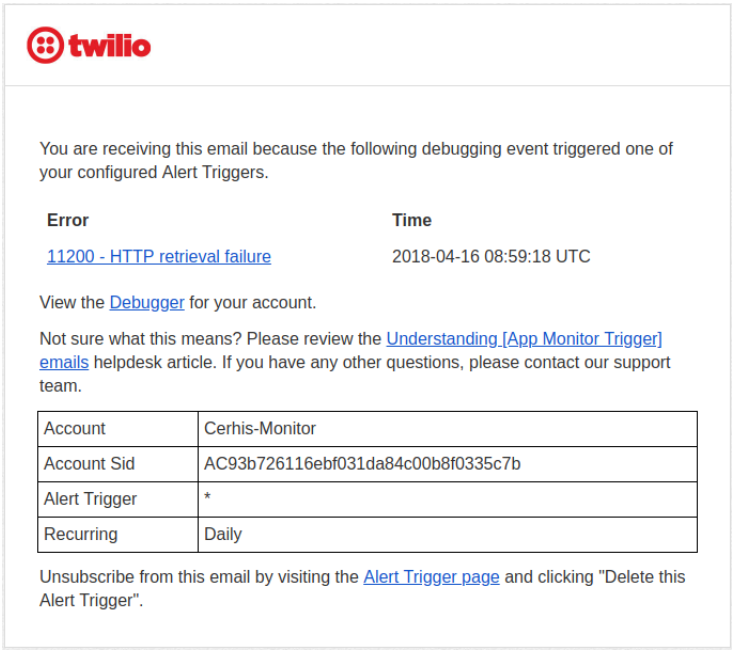


FIGURE 7.7 – Mail envoyé lors de l'apparition d'une erreur d'acheminement d'un SMS

aux personnes abonnées depuis l'application web.

Ces deux formes sont fonctionnelles et le système sous sa forme "hors ligne" est actuellement utilisé à Kinshasa, principalement pour le monitoring journalier des centres hospitaliers. De plus, toutes les fonctionnalités qu'il avait été prévu d'implémenter ont été mises en place (voir Résumé du chapitre 3). Enfin, une attention particulière a été portée à la facilité d'utilisation du système rendant l'expérience utilisateur de ce dernier intuitive et interactive.

Chapitre 8

Conclusion

Ce mémoire a permis de mettre en place une solution end-to-end permettant de surveiller les éléments d'un système informatique complexe, d'envoyer des SMS en cas de détection de panne ou de dysfonctionnement, d'enregistrer les données à la fois localement sur l'application et en ligne sur les serveurs de l'application web, de visualiser ces données sur une plateforme prévue à cet effet comprenant des fonctionnalités d'authentification, de gestion d'utilisateurs et de gestion d'abonnements aux flux de SMS et de redistribuer ces mêmes SMS aux personnes abonnées. Le système est fonctionnel bien qu'il doive être repris en main par AEDES et que certaines décisions et partenariats doivent encore être discutés et il respecte toutes les spécifications qui avaient été énoncées (voir chapitre 3).

Le système de centralisation et de visualisation de données de monitoring a été implémenté en tant qu'application web. Cette application est basée sur le stack React.js - Node (Express) - Couchbase et est actuellement hébergée sur les serveurs d'Amazon Web Services. L'API de Twilio a été utilisée pour gérer le transit des SMS et les redistribuer vers les personnes abonnées aux services et aux centres. Le numéro de téléphone par lequel les SMS transitent est un numéro aux USA, Kansas. Les SMS sont ensuite acheminés vers le serveur situé en Irelande puis enregistrés dans la base de données en Belgique (voir section 4.3.3 pour la justification des différents endroits).

Le système de monitoring correspond au projet Codepo de l'année précédente et ce mémoire avait également pour objectif de mettre à jour la partie logicielle du système. Les limitations du système Android ont montré que cette solution n'était pas pérenne et qu'un pivot vers une solution sur base d'un Raspberry couplé à un Arduino avec un module SMS serait plus robuste. Néanmoins le système reste fonctionnel et tant qu'un partenariat n'est pas trouvé avec un fournisseur de serveur SMS en République Démocratique du Congo, il serait trop onéreux d'envoyer plus de SMS que les SMS de rapport quotidien des centres de santé ce pour quoi l'application Android reste utilisable pour le moment. Lorsque l'application est en mode offline, les SMS sont directement expédiés vers les personnes enregistrées dans l'application. Par contre lorsque l'application est en mode online, les SMS transitent depuis la RDC vers le serveur SMS aux USA.

Le figure 8.1 reprend le flux des SMS à travers le monde avant d'être enregistrés dans la base de données.

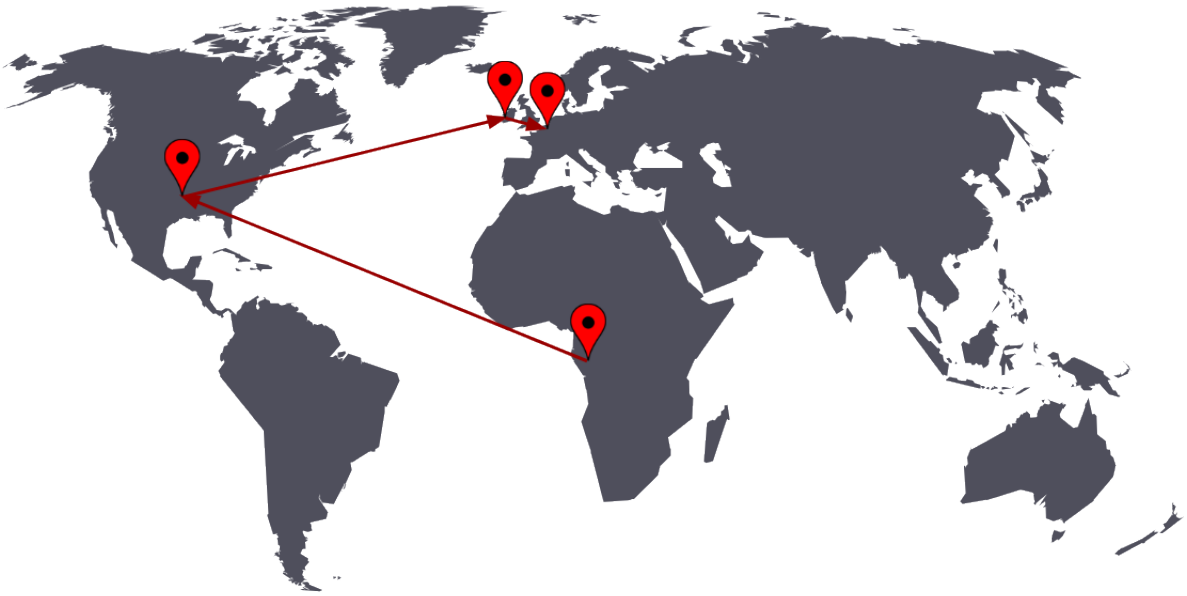


FIGURE 8.1 – Flux des SMS à travers le monde

8.1 Améliorations possibles et perspectives

Au vu des différentes fonctionnalités proposées dans le cahier des charges, les améliorations possibles ont déjà été listées. Ainsi, pour l'application web, le flux de données dans l'autre sens, de l'embouchure à la source ou de l'application web à la source permettrait de configurer les systèmes de monitoring à distance. La création d'une application pour les techniciens permettant la gestion des SMS. Cette application permettrait de traiter les SMS de monitoring et de visualiser ces derniers sur une carte en fonction du centre qui aurait envoyé le SMS. Un système d'authentification pour cette application serait également bienvenu. Enfin comme expliqué plus haut dans la section 7.2.1, il est conseillé de pivoter vers une solution sur base d'un Raspberry couplé à un Arduino muni d'un module SMS pour ce qui est du système de monitoring.

Les perspectives de la solution proposée s'inscrivent également dans une optique moins technique. En effet, cette plateforme peut être vue comme une façade du projet CERHIS, permettant de présenter une fenêtre plus large du projet. Présenter des données collectées en temps réel de centres de santé répartis géographiquement et représentés sur une carte donne un aspect très concret et pourra faciliter les interactions avec les potentiels centres de santé désirant installer le système et les potentiels investisseurs désireux de s'impliquer dans le projet. Cette solution donne des perspectives de mise à l'échelle très intéressantes et pourra, à terme, je l'espère, s'étendre dans plusieurs pays d'Afrique.

Bibliographie

[1] Agence Européenne pour le Développement et la Santé, Présentation, http://www.aedes.be

[2] CERHIS, Présentation, http://cerhis.org

[3] Ecole polytechnique de Bruxelles, Cellule Codepo, https://www.ulb.ac.be/facs/polytech/cooperation-Mission.html

[4] Ben Townsend, Jemal Abawajy and Tai-Hoon Kim : SMS-Based Medical Diagnostic Telemetry Data Transmission Protocol for Medical Sensors, 2011

[5] Neelambike S, Sushmitha M.P, Uzma Afrin, Shruthi.N, Sindhu K.V : Android Based Health Monitoring Using an SMS Based Telemedicine System, 2015

[6] Phil O'Connell, Interactive simple telehealth for the management of blood pressure, Stoke-on-Trent Clinical Commissioning Group

[7] Centre de crise en Belgique, alerte par SMS https://crisiscentrum.be/fr/content/be-alert-soyez-alerte-en-situation-durgence

[8] Network monitor, Fingbox, https://www.fing.io/fingbox-network-security-appliance/

[9] Monitoring and controlling of EMS-SCADA via SMS gateway, Rino Andias Anugraha and Tatang Mulyana, 2015 3rd International Conference on Information and Communication Technology (ICoICT)

[10] Hillebrand, Trosby, Holley, Harris : SMS the creation of Personal Global Text Messaging, Wiley 2010

[11] Crystal, David (2008-07-05). "2b or not 2b ?". Guardian Unlimited. London, UK. Retrieved 2008-07-08.

[12] Alphabets and language-specific information, https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=745

[13] Fernandes JP. Integration of Information for Environmental Security : Environmental Security, Information Security, Disaster Forecast and Prevention, Water Resources Management. Springer ; New York, NY, USA : 2008. Emergency Warnings with Short Message Service.

[14] M. Khan, S.S. Khan, Data and Information Visualization Methods and Interactive Mechanisms : A Survey, International Journal of Computer Applications, 34(1), 2011, pp. 1-14.

[15] Lidong Wang, Guanghui Wang and Cheryl Ann Alexander, Big Data and Visualization : Methods, Challenges and Technology Progress, Accepted July 20, 2015.

[16] D3, Introduction https://d3js.org/#introduction

[17] "For Protovis Users", Mbostock.github.com, retrieved August 18, 2012

[18] Myatt, Glenn J. ; Johnson, Wayne P. (September 2011), Making Sense of Data III : A Practical Guide to Designing Interactive Data Visualizations, Hoboken, New Jersey : John Wiley & Sons, p. A–2, ISBN 978-0-470-53649-0, retrieved January 23, 2013

[19] Wappalyzer, D3 https://www.wappalyzer.com/technologies/d3

[20] Nations Unies, Sommet des Nations Unies consacré à l'adoption du programme de développement pour l'après-2015

[21] Site du ministère des affaires étrangères, Afrique, https://diplomatie.belgium.be/fr/politique/regions_mondiales/afrique

[22] Enable, domaines d'action, https://www.enabel.be/fr

[23] Enable, Que fait Enable en RD Congo, https://www.enabel.be/fr/content/que-fait-enabel-en-rd-congo

[24] Redux, Three Principles https://redux.js.org/introduction/three-principles

[25] Redux, Glossary, Reducer https://redux.js.org/glossary#reducer

[26] CloudWatch, Mesure du temps de latence pour les services d'AWS : http://www.cloudwatch.in/

[27] Orange, forfaits internationaux https://travel.orange.fr/syntheseZonesEtTarifs

Annexe A

Modes d'emploi

Ces modes d'emploi sont également disponibles sur le wiki de CERHIS à l'adresse suivante :

http://cerhis.info/doku.php

A.1 Application de monitoring

l'application possède un tutoriel reprenant en détails les différents types de monitoring qu'il est possible d'implémenter ainsi que la manière de le faire. Ce qui suit reprend un tutoriel plus général en l'appuyant à l'aide d'illustrations.

Lorsque vous lancez l'application pour la première fois vous tombez sur un écran bleu vous demandant d'inscrire le nom du centre de santé dans lequel se trouve le système de monitoring. Cette page ne s'affichera plus par la suite (sauf si vous supprimez les données relatives à l'application), prenez donc garde en encodant ce nom. Lorsque vous lancez l'application pour la première fois, l'application va automatiquement vous demander les autorisations nécessaires à son bon fonctionnement, à savoir l'envoi de SMS et la position géographique. Complétez les données comme demandé, allumez le wifi du téléphone ainsi que sa localisation et complétez les données propres au centre.

Le deuxième écran sur lequel vous arrivez est l'écran principal de l'application. Ce dernier reprend une liste de tous les éléments du réseau qui sont monitorés. La première fois que vous lancez l'application, cette liste sera bien évidemment vide, pas de panique ! Pour ajouter un élément à monitorer, il suffit d'appuyer sur le bouton "+" en bas de l'écran et de choisir le type de monitoring que vous désirez. L'écran suivant vous permet d'encoder le service. Certaines parties sont spécifiques au type de service à encoder mais voici ce que vous devez savoir.

Les valeurs de seuils déterminent les valeurs critiques à partir desquelles il faut envoyer un SMS. Dans certains cas la valeur de seuil est un maximum et dans d'autres, un minimum. Le nom et l'adresse IP sont des identifiants qui permettent de communiquer et/ou de surveiller les appareils

relatifs. Une option 'Scan du réseau' permet de lister les appareils connectés et de sélectionner celui qui vous intéresse en cliquant puis en confirmant. Dans certains cas, l'adresse IP à encoder est l'adresse de l'Arduino, dans d'autres l'adresse est celle de l'appareil à monitorer. Les personnes à contacter sont les numéros de GSM encodés dans l'application qui recevront les SMS de l'application. Lorsque vous supprimez un nom, ce nom est supprimé pour tous les services. A chaque fois que vous ajoutez un numéro, il faut ensuite cocher la personne pour qu'elle soit contactée en cas de problème. Une fois ces différentes étapes complétées, vous pouvez appuyer sur l'icône en haut à droite qui permet d'enregistrer le service dans la base de données. Vous êtes redirigé sur l'écran principal et vous pouvez voir le service repris dans la liste. Si vous cliquez sur ce service, vous apercevrez les données récoltées à son propos ainsi qu'une liste reprenant les changements critiques dans ces valeurs. Vous pouvez zoomer sur ce graphe pour mieux voir l'heure à laquelle la donnée a été collectée mais cela est surtout gadget. Il vous est possible de modifier le service en cliquant sur l'icône de modification en haut à droite. La modification est tout à fait similaire à l'encodage des données.

Si vous désirez changer la fréquence d'échantillonnage de certains services, rendez-vous dans le troisième onglet de l'application, onglet "Paramètre", et cliquez sur l'item "Notification". Un écran vous propose ensuite les fréquences d'échantillonnage de tous les types de service.

'Contrôle toutes les' : Permet de choisir la fréquence d'échantillonnage en minutes lorsqu' aucune anomalie n'est détectée. '1 ère notification après' : Permet de choisir le nombre de fois consécutives d'anomalies que le système doit détecter avant d'envoyer un SMS. 'Intervalle entre les notifications' : Permet de choisir l'intervalle de temps en heures entre chaque SMS en cas d'anomalie persistante et ininterrompue.

A.2 Application de visualisation

Application Web

Utilisation de l'application

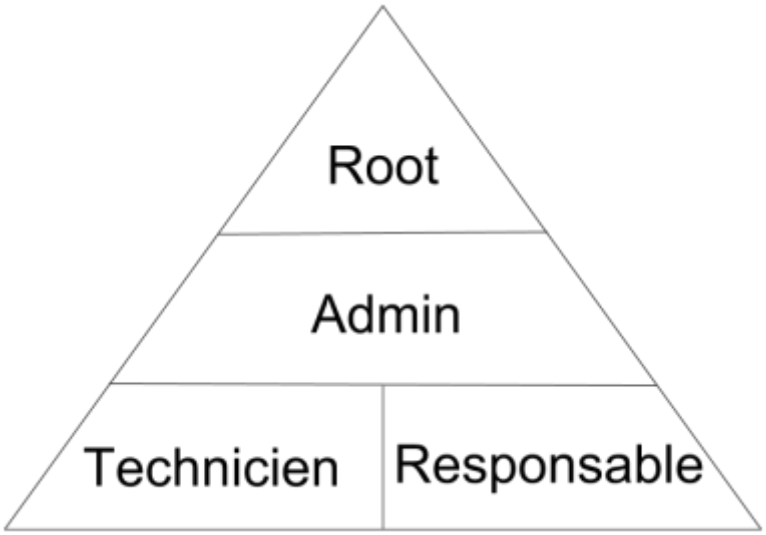
Pour vous rendre sur l'application web de centralisation et de visualisation des SMS de monitoring cliquez sur le lien suivant:

http://34.252.56.151/

Sur la page de login, rentrez vos identifiants pour vous connecter à l'application. Lors de votre connexion, vous êtes redirigé vers la page d'accueil qui reprend une explication des différentes permissions que vous possédez en fonction de votre rôle. Ces permissions sont explicitées ci dessous:

- Technicien, vous pouvez :
  - Consulter les données des centres auxquels vous êtes abonné
  - Modifier vos données de profil
  - Décider de recevoir ou non des SMS
- Responsable, vous pouvez :
  - Consulter les données de tous les centres
  - Modifier vos données de profil
- Admin, vous pouvez :
  - Consulter les données de tous les centres
  - Modifier vos données de profil
  - Décider de recevoir ou non des SMS d'alerte
  - Décider de recevoir ou non des SMS journaliers
  - Créer, mettre à jour ou même supprimer un utilisateur
  - Gérer les abonnements des différents utilisateurs
- Root, vous pouvez :
  - Consulter les données de tous les centres
  - Modifier vos données de profil
  - Décider de recevoir ou non des SMS d'alerte
  - Décider de recevoir ou non des SMS journaliers
  - Créer, mettre à jour ou même supprimer un utilisateur
  - Gérer les abonnements des différents utilisateurs
  - Il est impossible de vous supprimer

Ces permissions peuvent être représentées via la pyramide des permissions reprise ci-dessous. Plus vous êtes haut dans la pyramide plus vous avez de droits.



Les différents onglets

L'onglet “Dimensionnement” vous permet de consulter les données des centres encore en dimensionnement. L'onglet “Surveillance” vous donne un aperçu de tous les centres monitorés. Si un centre est en rouge, c'est qu'un problème n'est pas résolu. Autrement, il est précisé que tout est en ordre. Si vous cliquez sur un centre, vous pouvez avoir accès aux données qui lui sont relatives en fonction des services enregistrés dans le centre.

L'onglet “Paramètre” contient deux sous-onglets : Profil et Abonnements. Dans l'onglet “Profil”, vous pouvez modifier vos données personnelles. Dans l'onglet “Abonnements”, vous pouvez sélectionner les services auxquels vous désirez être abonné en fonction du centre. Si vous vous abonnez à un service en particulier, vous recevrez tous SMS d'alertes et de retour à la normale qui lui sont propre.

Enfin, dans l'onglet “Admin”, vous retrouvez à la fois une gestion des utilisateurs sous l'onglet “Utilisateurs” ainsi qu'une gestion des abonnements sous l'onglet “Abonnements”. L'onglet “Utilisateurs” vous permet de gérer les données des utilisateurs enregistrés dans l'application web. Vous pouvez également supprimer un utilisateur si ce dernier n'est pas “Root”. L'onglet “Abonnements” vous permet de gérer à la fois les abonnements des utilisateurs aux centres ainsi que les abonnements aux services. Si un utilisateur est abonné à un centre alors il recevra les SMS journaliers et s'il est abonné aux services, il recevra en plus les SMS d'alerte et de retour à la normale. Il n'est pas possible d'abonner un utilisateur aux services du centre s'il n'est pas abonné au centre. Les abonnements peuvent soit être générés à partir des centres soit à partir des utilisateurs.

Conditions de fonctionnement

Afin que les données de monitoring de l'application Android soient exportées sur la plateforme web, veillez à bien cocher l'option “Mise en ligne des données” lorsque vous complétez les données à propos du centre de santé à monitorer.

Annexe B

Mise en production

B.1 Serveur Amazon Web Service

La mise en production d'une application web peut se faire de différentes manières et via différents services ou fournisseurs de services mais ce tutoriel se concentrera sur la mise en production du stack React.js - Node.js - CouchbaseDB. Si vous possédez déjà un nom de domaine, certaines étapes notamment pour configurer les options SSL sont à considérer en plus.

B.1.1 Création d'une instance EC2

Connectez-vous à la console AWS et sélectionnez une région (figure B.1).

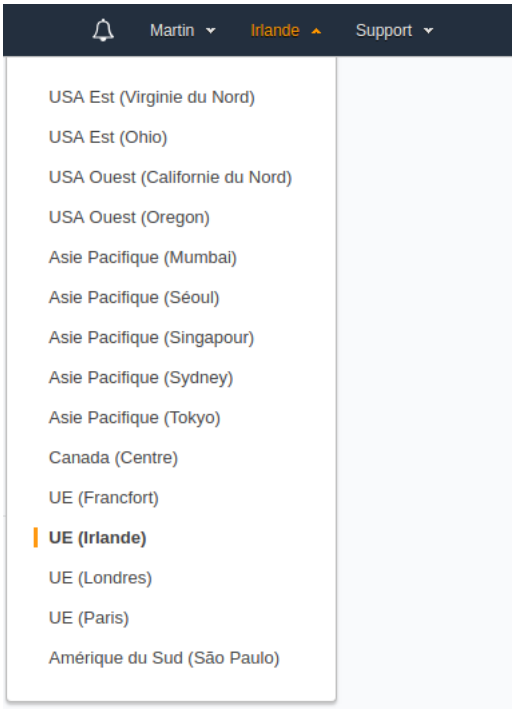


FIGURE B.1 – Choix d'une région AWS

Sélectionnez ensuite le service EC2 et cliquez sur "Créer une instance". Suivez ensuite la procédure qui est en fait composée de 4 grosses étapes :

1. Sélectionner le type de serveur (Ubuntu 16.04)
2. Nommer l'instance (optionnel)
3. Ajouter les groupes de sécurité pour les protocoles HTTP et HTTPS
4. Créer et télécharger une nouvelle paire de clés dont vous aurez besoin pour vous connecter au serveur (Conservez-les dans un endroit sûr car vous ne pourrez plus les re-télécharger)

B.1.2 Connexion et mise à jour

Connectez-vous maintenant en SSH au serveur fraîchement installé. Pour cela, utilisez un client SSH (putty pour windows ou directement la ligne de commande pour Unix avec la commande ssh). Pour vous connecter, vous aurez également besoin de l'adresse IP publique de votre instance. Cette adresse IP se trouve sur la page de vos instances, lorsque vous cliquez sur "Instances en exécution" depuis le tableau de bord ou sur "Instances" depuis le menu à gauche (voir figure B.2). Vous trouvez ensuite l'IP dans la colonne "IP publique (IPv4)".

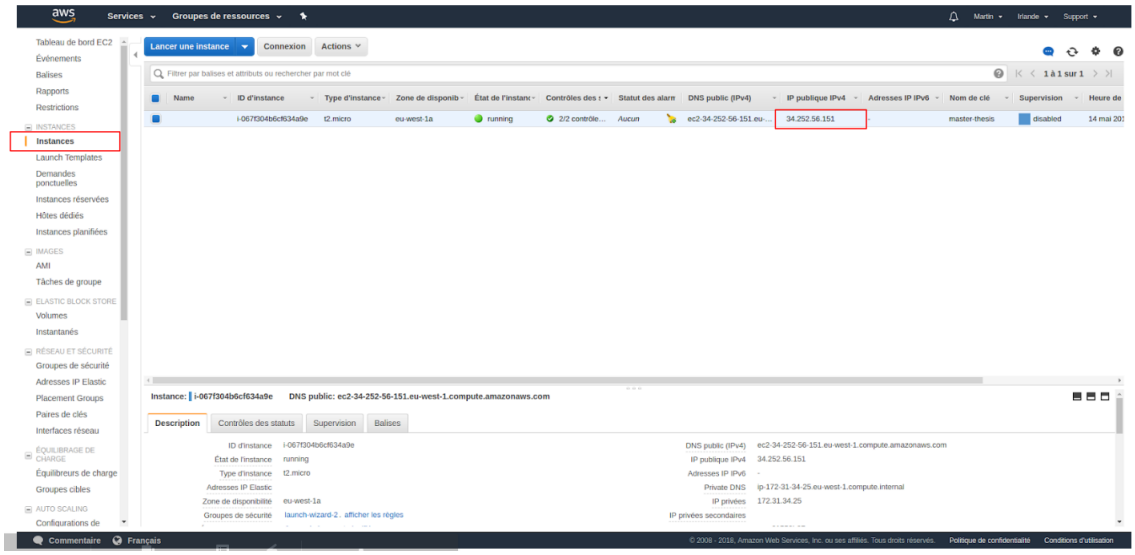


FIGURE B.2 – Ip publique de l'instance

Ouvrez maintenant votre terminal et rendez vous dans le dossier contenant votre paire de clés SSH. Utilisez votre client SSH (Putty pour windows) et tapez la commande suivante :

```
1        $ chmod 400 ./YOUR_KEY_PAIR_NAME.pem
```

Cela va vous permettre d'utiliser votre paire de clé pour votre client SSH. Tapez maintenant la commande suivante pour vous connecter à votre instance :

```
1        $ ssh -i "YOUR_KEY_PAIR_NAME.pem" ubuntu@YOUR_IPv4_LINK
```

Maintenant que vous êtes connecté à votre instance, mettez cette dernière à jour en tapant les commandes suivantes :

```
$ sudo apt-get update && sudo apt-get upgrade -y
```

B.1.3 Installation du serveur Nginx

Nginx est un logiciel libre de serveur Web ainsi qu'un proxy inverse écrit par Igor Sysoev, dont le développement a débuté en 2002 pour les besoins d'un site russe à très fort trafic. C'est une très bonne alternative à Apache. Installez Nginx avec les commandes suivantes :

```
$ sudo apt-get install nginx -y && sudo systemctl start nginx
```

Permettez ensuite que Nginx se lance lorsque le serveur s'allume :

```
$ sudo systemctl start nginx
```

B.1.4 Mise en place de Node.js

Installez maintenant Node.js via le gestionnaire de version NVM :

```
$ sudo apt-get update
$ sudo apt-get install build-essential libssl-dev
$ curl -sL https://raw.githubusercontent.com/creationix/nvm/
VERSION_NUMBER/install.sh -o install_nvm.sh
$ bash install_nvm.sh
$ source ~/.profile
$ nvm install 6.0.0
$ nvm use 6.0.0
$ node -v
$ npm -v
```

B.1.5 Configuration de Nginx

Maintenant que Node.js est installé sur votre serveur, clonez le git de votre projet sur votre serveur et lancez votre projet. Si vous désirez que le projet ne s'arrête pas lorsque vous quittez votre session SSH, lancez votre projet avec pm2.

Supprimez tout d'abord le fichier de configuration de Nginx puis créez-en un nouveau :

```
$ sudo rm /etc/nginx/sites-available/default
$ sudo nano /etc/nginx/sites-available/default
```

Configurez maintenant le proxy pour rediriger les requêtes vers votre application web en modifiant le port du fichier de configuration proposé ci-dessous :

```
        server {
            listen 80;
            server_name;
            location / {
                proxy_pass http://127.0.0.1:8080;
                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection 'upgrade';
                proxy_set_header Host $host;
                proxy_cache_bypass $http_upgrade;
                proxy_redirect off;
             }
        }
```

Rechargez maintenant Nginx pour appliquer le changement de configuration :

```
        $ sudo systemctl reload nginx
```

B.2 Configuration de Couchbase

Si vous avez utilisé une instance micro d'AWS, il est inutile d'essayer d'installer Couchbase Community sur ce même serveur, la RAM que demande Couchbase étant trop importante. Pour pouvoir utiliser Couchbase, lancez-la sur un autre serveur et changez vos données d'environnement dans votre projet pour satisfaire aux exigences (nom du bucket - nom d'utilisateur - mot de passe). ⚠Si vous bloquez certains ports du serveur avec votre base de données, assurez vous de modifier la bibliothèque que vous utilisez pour les requêtes ou créez vos propres requêtes.

B.3 Serveur SMS Twilio

Afin de pouvoir envoyer des SMS en utilisant l'API de Twilio, récupérez tout d'abord vos JSON Web Token et enregistrez-les dans vos variables d'environnement. Ensuite, pour rendre votre envoi de SMS générique, créez une fonction utilisant vos Token :

```
    require("dotenv").config()
    const client = require("twilio")(process.env.SMS_SID,
        process.env.SMS_TOKEN)

    const defaultCallBack = (err, msg) => {
        if(!err) console.log(msg.sid)
        else console.log(err)
    }

    export const sendSmsTwilio = (to : string, from : string,
```

```
11      body : string, callback = defaultCallBack) => {
12          client.messages.create(
13              {
14                  to,
15                  from,
16                  body
17              },
18              callback
19          )
20      }
```

Ensuite pour pouvoir récupérer les SMS envoyés vers votre numéro Twilio, configurez la route sur votre compte Twilio pour qu'elle corresponde à la route définie sur votre serveur comme montré sur la figure B.3.

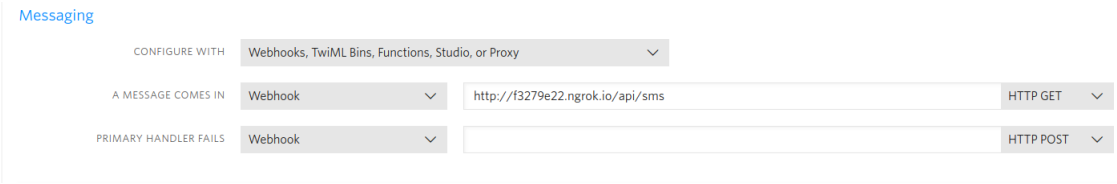


FIGURE B.3 – Configuration de la route d'acheminement du SMS sur Twilio

B.4 Plugin Fabric

Afin d'ajouter Fabric à vos applications Android, installez tout d'abord le plugin depuis l'IDE Android Studio. Pour cela, rendez-vous dans les paramètres de l'éditeur et recherchez le plugin avant de l'installer. Une fois le plugin installé, une icône bleue en haut à droite de votre éditeur va apparaître. Cliquez sur cette icône lorsque vous désirez rajouter Fabric à votre application (voir figure B.4). N'oubliez pas de vous créer un compte sur l'application !

Une fois le plugin lancé, sélectionnez le kit "Crashlytics" qui vous permettra d'avoir les logs de toutes les erreurs rencontrées lors de l'utilisation de l'application. Cliquez sur "Get code" pour que le code soit automatiquement ajouté dans votre application. Une fois le code généré, relancez un build de votre application pour que le plugin soit ajouté.

Le plugin est maintenant ajouté et vous pouvez partager votre application en déposant l'apk dans la fenêtre représentée à la figure B.5. Encodez les adresses mails des personnes auxquelles vous désirez envoyer l'application et confirmez ensuite l'envoi. Vous recevrez des notifications lorsque les personnes auront installé votre application.

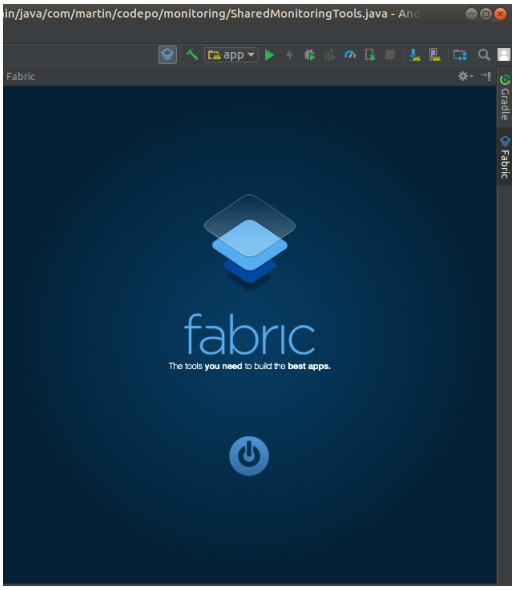


FIGURE B.4 – Plugin Fabric dans Android Studio

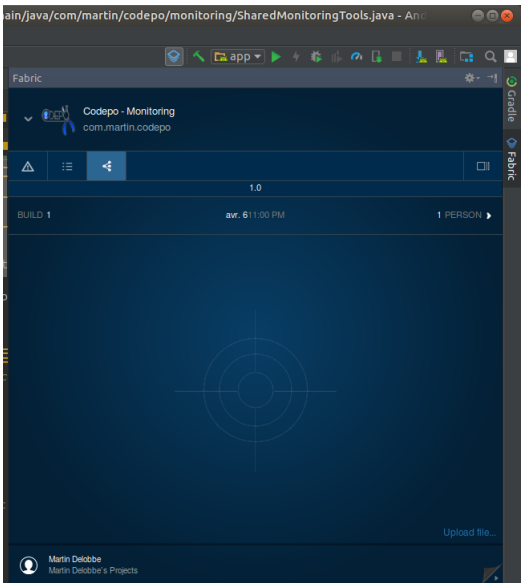


FIGURE B.5 – Partage de l'apk

Annexe C

Documents et programme de la mission

C.1 Termes et références

Termes de référence mission court terme

1. CONTEXTE

<Rappel projet 2016-2017, dont l'objet est nommé « système de monitoring » dans le document>

<Description de l'objet du mémoire, nommé « site internet de monitoring » dans le document>

2. OBJECTIFS ET RÉSULTATS ESCOMPTÉS

2.1. Objectifs généraux du projet

- Renforcer le bon fonctionnement de CERHIS dans les structures de santé
- Faciliter et optimiser le travail des techniciens chargés du suivi de CERHIS
- Collecter des données techniques utiles pour la poursuite du projet

2.2. Objectifs spécifiques de la mission

- Le système de monitoring et le site internet de monitoring permettent la surveillance à distance du fonctionnement de CERHIS dans trois structures de santé de Kinshasa
- Les partenaires locaux sont capables de mettre en œuvre le système de monitoring
- Les partenaires locaux sont capables d'utiliser le site internet de monitoring

2.3. Principaux résultats attendus et activités liées

- Le système de monitoring - partie projet 2016-2017 - est installé et fonctionne dans trois structures de santé à Kinshasa
    - Assemblage du système de monitoring (Arduino, smartphone…)
    - Raccord sur les appareils à surveiller (batteries, réseau…)
    - Configuration de l'application de monitoring sur le smartphone
    - Configuration des appareils à surveiller (SNMP sur serveur…)
    - Tests avec simulations de pannes
    - Au besoin, améliorations et/ou corrections du système de monitoring
    - NB : les parties électroniques et les smartphones sont fournis à l'étudiant
- Les partenaires locaux (MaisOrdi) sont capables d'installer et entretenir le système de monitoring
    - Les partenaires locaux participent à l'installation du système dans au moins une structure de santé

    - Si nécessaire, l'appropriation est corrigée ou renforcée grâce à un complément de formation et/ou un complément de documentation sur le wiki.
    - Les partenaires locaux font la démonstration des acquis en mettant en place de façon autonome le système de monitoring dans au moins une autre structure de santé, sous la supervision de l'étudiant
    - Les partenaires locaux réalisent l'entretien du système de monitoring pendant la durée de la mission
- Le site internet de monitoring fonctionne (à faire avant le début de la mission)
    - Le site internet est installé
    - Il reçoit et traite les messages SMS des trois systèmes de monitoring installés
    - Les données sont accessibles sur internet à partir de la RDC et de la Belgique
    - Les informations pertinentes sont envoyées sur les smartphones des techniciens
- Les partenaires locaux sont capables d'utiliser le site internet de monitoring
    - Formation des techniciens locaux à la consultation du site et à la configuration des notifications, des conditions d'alertes, etc.
- Les données monitorées correspondent aux besoins des partenaires locaux
    - Définition avec les partenaires locaux des paramètres nécessaires au bon suivi du système, éventuelles adaptations du système de monitoring
    - Liste de pannes ou problèmes auxquels les partenaires locaux sont confrontés et qui pourraient être traités de façon plus optimale à l'aide du système de monitoring et du site internet de monitoring
- Les partenaires locaux sont informés des enjeux du projet et de ses progrès
    - Présentation/rappel du projet en début de mission
    - Réunion de restitution et présentation des résultats en fin de mission
- Les résultats de la mission sont documentés pour la suite du mémoire et du projet
    - Relevé du feedback des utilisateurs (partenaires locaux), liste de bugs à corriger et adaptations à réaliser dans la suite du mémoire, etc.
    - Suggestions de fonctionnalités supplémentaires et améliorations pour le futur…
    - Présentation des résultats de la mission chez AEDES
- La visibilité du projet est renforcée
    - Photographies pour promotion du projet

3. CHAMPS D'INTERVENTION

- Travail sur le terrain à l'Hôpital Général de Référence Roi Baudouin, au Centre de Santé Bolingo et au Centre de Santé Kitoko, dans la commune de Masina à Kinshasa
- Développement et travail administratif soit au siège de la société MaisOrdi, soit au logement de l'étudiant.

4. CALENDRIER ET LOGISTIQUE

- La mission est prévue du 31/03/2018 au 15/04/2018.
- <modalités de logement et per diem à confirmer>
- Le transport est assuré par les partenaires locaux.

C.2 Gantt de la mission

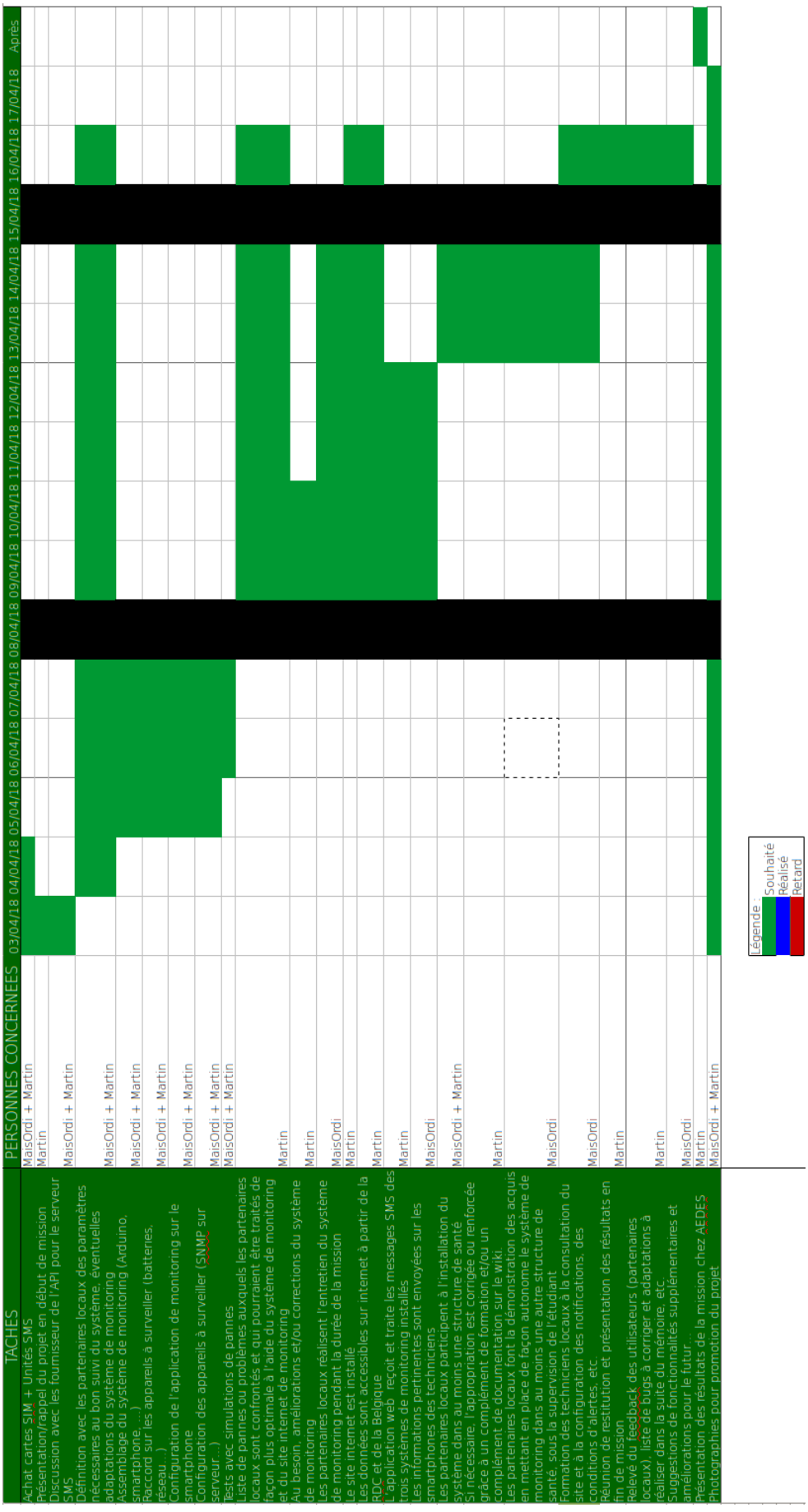


FIGURE C.1 – Diagramme de Gantt en début de mission

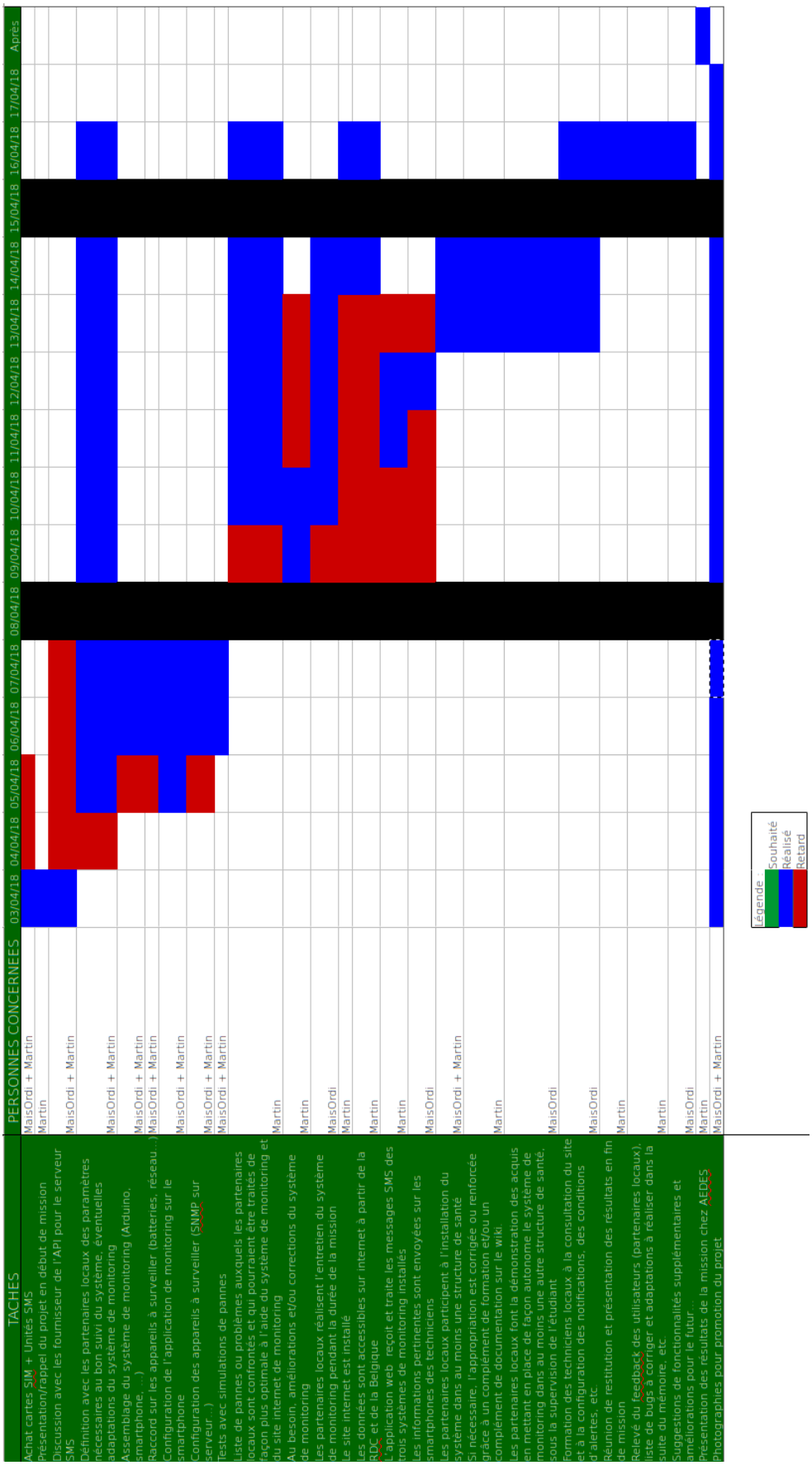


FIGURE C.2 – Diagramme de Gantt en fin de mission