République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abderrahmane Mira de Béjaïa

Faculté des Sciences Exactes
Département Informatique

Mémoire de Master
Option Génie Logiciel

Thème

Conception et réalisation d'une application Web pour la numérisation d'électrocardiogrammes

Présenté par
BOUNECER Mohand et TITOUN Rami

Soutenu devant le jury composé de :

Président	M. KHAMMARI	Mohammed	U. A./MIRA Béjaïa
Examinateur	M. BOUCHEBBAH	Fatah	U. A./MIRA Béjaïa
Examinateur	M. MIHOUBI	Mohammed	U. A./MIRA Béjaïa
Encadrant	M. SIDER	Abderrahmane	U. A./MIRA Béjaïa

Année Universitaire 2023/2024

Remerciments

Tout d'abord, nos premiers remerciements vont à Dieu, Tout-Puissant et Miséricordieux, à qui nous devons tout.

Nous exprimons notre sincère gratitude envers nos familles, parents, frères et sœurs, qui ont été une source constante d'inspiration et de soutien. Leurs encouragements indéfectibles ont été essentiels à notre réussite.

Nous remercions Monsieur SIDER Abderahmane, notre encadrant, pour ses conseils et son accompagnement tout au long de ce projet. Son expertise a contribué de manière significative à la réalisation de ce mémoire.

Nous tenons également à exprimer notre gratitude envers les membres du jury pour leur patience et pour avoir accepté l'évaluation de notre travail avec rigueur et bienveillance. Leurs commentaires constructifs ont grandement enrichi notre approche académique.

Enfin, à tous nos amis qui nous ont soutenus tout au long de cette aventure, nous exprimons notre gratitude sincère pour leur encouragement constant et soutien amical. Leur présence a été une source de motivation et de réconfort tout au long de ce parcours.

Nous sommes profondément reconnaissants envers chacune de ces personnes et institutions qui ont contribué à notre succès, et nous leur adressons nos plus vifs remerciements.

Résumé

Ce projet vise à présenter le processus de conception et réalisation d'une application Web de scan et numérisation et gestion d'ECGs en détaillant l'approche déterministe basée sur le traitement d'image que nous avons entreprise afin d'aboutir à ce résultat.

This project aims to present the process of designing and developing a web application for scanning, digitizing, and managing ECGs. It details the deterministic approach based on image processing that we employed to achieve this outcome.

Mots-clés: Numérisation d'ECG, Traitement d'image, Application Web, Méthodologie Agile, Prisma, React, ExpressJS

Keywords: ECG Digitization, Image Processing, Web Application, Agile Methodology, Prisma, React, ExpressJS

Table des matières

1 Généralités **3**
- 1.1 Application informatique . . . 3
  - 1.1.1 Architecture d'une application . . . 3
  - 1.1.2 Application multiplateforme . . . 4
  - 1.1.3 Application Web . . . 4
  - 1.1.4 Application Ajax . . . 4
- 1.2 L'électrocardiographie . . . 5
  - 1.2.1 Définition . . . 5
  - 1.2.2 L'électrocardiogramme . . . 5
  - 1.2.3 Dérivations de l'ECG . . . 6
- 1.3 Signal . . . 7
  - 1.3.1 Types de signaux . . . 7
- 1.4 Image numérique . . . 7
- 1.5 Étude du cas existant . . . 7
  - 1.5.1 PMcardio . . . 8
  - 1.5.2 GARMIN . . . 9

2 Traitement d'images **11**
- 2.1 Définition . . . 11
  - 2.1.1 Origine du traitement de l'image . . . 11
  - 2.1.2 Domaines d'utilisations du traitement d'images . . . 12
- 2.2 Échantillonage et quantification . . . 13
  - 2.2.1 Échantillonage . . . 13
  - 2.2.2 Quantification . . . 13
- 2.3 Représentation des images . . . 15
- 2.4 Représentation des couleurs . . . 15
  - 2.4.1 Image binaire . . . 16
  - 2.4.2 Niveaux de gris . . . 16
  - 2.4.3 Espace RGB . . . 16
  - 2.4.4 Espace luminance/chrominance . . . 17
  - 2.4.5 Espace HSL . . . 17
  - 2.4.6 Espace HSV . . . 18
- 2.5 Séparation par composantes . . . 18
- 2.6 Histogramme . . . 19
  - 2.6.1 Normalisation d'histogramme . . . 20
  - 2.6.2 Interprétation d'histogrammes . . . 20

2.6.3 Égalisation d'histogramme . . . 20
2.7 Transformée de Fourier et Spectre . . . 21
2.7.1 La Transformée de Fourier . . . 21
2.7.2 La Transformée de Fourier Discrète . . . 22
2.8 Filtrage des images . . . 22
2.8.1 Convolution . . . 22
2.8.2 Filtrage fréquentiel . . . 23
2.8.3 Filtrage morphologique . . . 23
2.9 Segmentation . . . 25
2.9.1 Segmentation par seuillage . . . 25
2.9.2 Segmentation basée sur la détection des contours . . . 26
2.9.3 Segmentation par le Filtre Canny . . . 27

3 Conception et Réalisation **29**
3.1 Cahier de charge . . . 30
3.2 Analyse des besoins . . . 31
3.2.1 Besoins fonctionnels . . . 31
3.2.2 Besoins non fonctionnels . . . 32
3.3 Modélisation . . . 32
3.3.1 UML . . . 33
3.3.2 Modèle relationnel . . . 34
3.3.3 Identification des acteurs . . . 34
3.4 Méthode de conception . . . 35
3.4.1 Méthode AGILE . . . 36
3.4.2 Principes de la méthode AGILE . . . 36
3.5 Processus de développement Scrum . . . 37
3.5.1 Équipe Scrum . . . 37
3.5.2 Étapes clés de Scrum . . . 38
3.6 Rédaction du Product Backlog initial . . . 40
3.6.1 Identification des User Stories . . . 40
3.6.2 Identification des scénarios d'utilisation . . . 40
3.6.3 Product Backlog initial . . . 48
3.7 Mise en place de l'environnement de développement . . . 49
3.7.1 Côté Client (Frontend) . . . 49
3.7.2 Côté Serveur (Backend) . . . 50
3.7.3 Base de données . . . 50
3.7.4 Communication Client-Serveur . . . 50
3.7.5 Outils de test . . . 51
3.8 Réalisation des Landing Pages . . . 51
3.9 Itérations Scrum . . . 53
3.9.1 Sprint 01 . . . 53
3.9.2 Sprint 02 . . . 65
3.9.3 Sprint 03 . . . 74
3.9.4 Sprint 04 . . . 80
3.9.5 Sprint 05 . . . 93
3.9.6 Sprint 06 . . . 98

4 Scan et numérisation d'ECGs **106**
4.1 Caractéristiques d'un ECG . . . 106
4.1.1 Disposition des signaux de dérivations . . . 106
4.1.2 Encre et papier . . . 107
4.1.3 Étalonnage . . . 109
4.2 Scan de l'ECG . . . 109
4.2.1 Implémentation de la méthode . . . 110
4.2.2 Fiabilité de la fonctionnalité de scan . . . 117
4.2.3 Algorithme de scan d'un ECG . . . 118
4.3 Binarisation . . . 120
4.3.1 Approche par couleurs . . . 120
4.3.2 Approche par la transformée de Fourier . . . 125
4.4 Segmentation . . . 137
4.4.1 Détection de la disposition des signaux de dérivations . . . 139
4.4.2 Résolution du problème de chevauchement horizontal . . . 140
4.4.3 Segmentation approximative . . . 141
4.4.4 Segmentation précise . . . 144
4.5 Numérisation . . . 148
4.5.1 Extraction des points clés . . . 151
4.5.2 Interpolation . . . 153
4.5.3 Interpolation de l'ECG . . . 154
4.6 Résultats et discussions . . . 154
4.6.1 Précision . . . 154
4.6.2 Complexité algorithmique . . . 155
4.6.3 Portabilité . . . 157

Table des figures

2.1 Les longueurs d'onde associées aux couleurs.[28]. 16
2.2 Comparaison entre l'espace HSL et HSV. 18

3.1 Landing Page principale. 52
3.2 Page informative. 52
3.3 Landing Page des entités de confiance. 53
3.4 Diagramme de cas d'utilisation du Sprint 01. 54
3.5 Diagramme de séquence du cas d'utilisation "Inscription". 55
3.6 Diagramme de séquence du cas d'utilisation "Connexion". 56
3.7 Diagramme de séquence du cas d'utilisation "Réinitialisation du mot de passe". 57
3.8 Diagramme de séquence du cas d'utilisation "Modification du compte". 58
3.9 Diagramme de séquence du cas d'utilisation "Suppression du compte". 58
3.10 Diagramme de classe du Sprint 01. 59
3.11 Interface d'inscription. 60
3.12 Interface de connexion. 61
3.13 Interface de demande de réinitialisation du mot de passe. 62
3.14 Mail de récupération de mot de passe. 62
3.15 Erreur de validité du token de réinitialisation du mot de passe. 63
3.16 Interface de réinitialisation du mot de passe. 63
3.17 Interface des paramètres du compte. 64
3.18 Interface des modification du compte. 64
3.19 Interface des suppression du compte. 65
3.20 Diagramme de cas d'utilisation du Sprint 02. 66
3.21 Diagramme de séquence "Consultation de la liste des profils". 66
3.22 Diagramme de séquence "Création d'un profil". 67
3.23 Diagramme de séquence "Modification d'un profil". 68
3.24 Diagramme de séquence "Suppression d'un profil". 68
3.25 Diagramme de classe du Sprint 02. 69
3.26 Interface de la liste des profils. 70
3.27 Formulaire de création d'un profil. 70
3.28 Ajout d'un profil. 71
3.29 Interface de sélection de photo de profil. 71
3.30 Personnalisation des profils avec des images. 72
3.31 Formulaire de modification d'un profil. 73
3.32 Confirmation de suppression d'un profil. 73
3.33 Diagramme de cas d'utilisation du Sprint 03. 74
3.34 Diagramme de séquence du cas d'utilisation "Scan d'un ECG". 75

3.35 Diagramme de séquence du cas d'utilisation "Téléchargement d'un scan". . . . 76
3.36 Diagramme de classe du Sprint 03. . . . 76
3.37 Interface récapitulative du compte (Tableau de bord). . . . 77
3.38 Interface de soumission d'un ECG. . . . 78
3.39 Téléversement de la photo d'un ECG. . . . 78
3.40 Chargement du module de scan d'un ECG. . . . 79
3.41 Interface de scan d'un ECG. . . . 80
3.42 Téléchargement du scan obtenu. . . . 80
3.43 Diagramme de cas d'utilisation du Sprint 04. . . . 81
3.44 Diagramme de séquence du cas d'utilisation "Numérisation d'un ECG". . . . 82
3.45 Diagramme de séquence du cas d'utilisation "Consultation de la liste des ECGs". 83
3.46 Diagramme de séquence du cas d'utilisation "Suppression d'un ECG". . . . 83
3.47 Diagramme de séquence du cas d'utilisation "Import d'un ECG". . . . 84
3.48 Diagramme de classe du Sprint 04. . . . 84
3.49 Formulaire de soumission d'un ECG à la numérisation. . . . 85
3.50 Lancement du numérisation d'un ECG. . . . 86
3.51 Interface de consultation de la liste des ECGs. . . . 87
3.52 Visualisation de la note écrite de l'ECG. . . . 88
3.53 Filtrage d'ECGs par profil. . . . 88
3.54 Sélection du profil auquel attribuer un ECG. . . . 89
3.55 Soumission d'un ECG d'un profil particulier. . . . 90
3.56 Interface de consultation d'un ECG. . . . 91
3.57 Import d'un ECG. . . . 92
3.58 Bouton de suppression d'un ECG. . . . 92
3.59 Diagramme de cas d'utilisation du Sprint 05. . . . 93
3.60 Diagramme de séquence du cas d'utilisation "Signalement d'un dysfonctionnement". . . . 94
3.61 Diagramme de classe du Sprint 05. . . . 95
3.62 Image de test de l'application. . . . 96
3.63 Scan sur l'image de test. . . . 96
3.64 Notification de la réception d'un signalement de scan erroné. . . . 97
3.65 Message retourné lors de l'échec d'une numérisation. . . . 97
3.66 Affichage d'un ECG dont la numérisation est erronée. . . . 98
3.67 Statut d'un ECG dont la numérisation est erronée. . . . 98
3.68 Diagramme de cas d'utilisation du Sprint 06. . . . 99
3.69 Diagramme de séquence du cas d'utilisation "Partage des données ECGs". . . . 100
3.70 Diagramme de séquence du cas d'utilisation "Acquisition des ECGs partagés". 101
3.71 Diagramme de classe du Sprint 06. . . . 102
3.72 Paramètre de la participation à l'enrichissement de la base de données. . . . 103
3.73 Statut de la participation. . . . 103
3.74 Arrêt de la participation. . . . 104
3.75 Refus de l'obtention du Dataset. . . . 105

4.1 Exemples de disposition des dérivations dans un ECG. . . . 107
4.2 Encre bleu foncé d'un ECG. [37]. . . . 108
4.3 Inconsistance de l'encre dans un ECG. . . . 108
4.4 Schéma du rectangle d'étalonnage d'un ECG. . . . 109

4.5 Photo d'un ECG prise par un Google Pixel 8. . . . 110
4.6 Image redimensionnée avant le scan. . . . 111
4.7 Transformation en niveaux de gris de la photo de l'ECG. . . . 111
4.8 Filtre passe-bas sur l'image. . . . 112
4.9 Filtre Canny appliqué sur l'image redimensionnée. . . . 113
4.10 Détection de la région d'intérêt. . . . 114
4.11 Points extrêmes ($ABCD$)du polygone obtenu. . . . 115
4.12 Schéma des coins d'une région d'intérêt à quatre coins. . . . 116
4.13 Scan de l'ECG contenu dans l'image I_{input}. . . . 117
4.14 Test de la fonctionnalité de scan. . . . 118
4.15 Seuillage global sur un scan numérique d'ECG. . . . 120
4.16 Seuillage local sur un scan numérique d'ECG. . . . 122
4.17 Éventail de couleurs de la composante de teinte (H). . . . 123
4.18 Seuillage par HSV. . . . 124
4.19 Seuillage par HSV sur une photo à luminosité irrégulière. . . . 125
4.20 Transformées de Fourier sur des images binaires présentant des motifs de bandes blanches droites. . . . 126
4.21 Spectre de magnitude d'un ECG scanné par machine. . . . 127
4.22 Spectre de magnitude de l'inverse de I_{scan}. . . . 127
4.23 Inverse de I_{scan}. . . . 128
4.24 Suppression de basses fréquences. . . . 128
4.25 Suppression de hautes fréquences. . . . 128
4.26 Obstacles à la binarisation d'une image. . . . 130
4.27 Ajustement de l'image scannée. . . . 131
4.28 Inverse de l'image traitée. . . . 132
4.29 Traitement du spectre de magnitude. . . . 133
4.30 Test du traitement de magnitude sur une image scannée. . . . 134
4.31 Après rehaussement. . . . 135
4.32 Augmentation de la luminosité. . . . 135
4.33 Seuillage global. . . . 136
4.34 Dilatation correctrice de coupures. . . . 136
4.35 Suppression des petits objets. . . . 137
4.36 Segmentation par détection de contours. . . . 138
4.37 Tentative de segmentation par contours. . . . 139
4.38 Enveloppe convexe du cas de chevauchement. . . . 140
4.39 Solution au chevauchement horizontal. . . . 141
4.40 Visualisation des douze p_{start} (en vert) et des douze p_{end} (en rouge) reliés par une ligne (en bleu). . . . 142
4.41 Visualisation de la segmentation approximative en rouge sur trois dérivations. 143
4.42 Segmentation approximative. . . . 144
4.43 Algorithme de parcours en largeur sur un signal de dérivation binaire. . . . 145
4.44 Algorithme de parcours en largeur inverse sur un signal de dérivation binaire. . 146
4.45 ET logique entre les parcours de gauche à droite et de droite à gauche. . . . 146
4.46 Élimination des restes. . . . 147
4.47 Bornes maximale $y = y_{\max}$ en vert et minimale $y = y_{\min}$ en rouge. . . . 147
4.48 Segmentation précise sans chevauchement vertical. . . . 148

4.49 Rectangles d'étalonnage et petits objets isolés. 149
4.50 Détection des rectangles d'étalonnage. 149
4.51 Segmentation d'un rectangle d'étalonnage. 150
4.52 Amincissement du signal et division en deux parties verticalement (I_{new}). . . . 151
4.53 Détection des p_{hi} en jaune et des p_{lo} en bleu. 152
4.54 Sélection des ordonnées. 152
4.55 Numérisation par interpolation cubique. 154
4.56 Précision sur l'axe de temps de la numérisation. 155
4.57 Calcul de la taille de l'image obtenue après le scan. 157

Liste des tableaux

3.1 Identification des User Stories. 40
3.2 Scénario de ”Inscription”. 41
3.3 Scénario de ”Connexion”. 42
3.4 Scénario de ”Réinitialisation du mot de passe”. 42
3.5 Scénario de ”Modification du compte”. 43
3.6 Scénario de ”Suppression du compte”. 43
3.7 Scénario de ”Consultation de la liste des profils”. 43
3.8 Scénario de ”Création d’un profil”. 44
3.9 Scénario de ”Modification d’un profil”. 44
3.10 Scénario de ”Suppression d’un profil”. 44
3.11 Scénario de ”Scan d’un ECG”. 45
3.12 Scénario de ”Numérisation d’un ECG”. 45
3.13 Scénario de ”Consultation des ECGs”. 46
3.14 Scénario de ”Suppression d’un ECG”. 46
3.15 Scénario de ”Import d’un ECG”. 46
3.16 Scénario de ”Participation à l’enrichissement de la base de données”. 47
3.17 Scénario de ”Acquisition des ECGs partagés”. 47
3.18 Scénario de ”Signalisation d’un scan erroné”. 47
3.19 Scénario de ”Signalisation d’une numérisation échouée”. 48
3.20 Scénario de ”Signalisation d’une numérisation erronée”. 48
3.21 Product Backlog initial. 48
3.22 Product Backlog du Sprint 01. 53
3.23 Product Backlog du Sprint 02. 65
3.24 Product Backlog du Sprint 03. 74
3.25 Product Backlog du Sprint 04. 81
3.26 Product Backlog du Sprint 05. 93
3.27 Product Backlog du Sprint 06. 98

4.1 Complexité temporelle du scan et de la numérisation de l’ECG. 156

Introduction générale

Dans un monde de plus en plus tourné vers le numérique, les applications informatiques occupent une place centrale dans le développement de nouveaux outils. Que ce soit dans le cadre des communications, des loisirs, ou encore de la santé, leur rôle ne cesse de croître. L'un des domaines où cette avancée est particulièrement notable est celui de l'électrocardiographie (ECG), où la numérisation des signaux et l'analyse via l'intelligence artificielle offre de nouvelles possibilités en matière de diagnostic et de suivi médical. Dans cette étude, nous nous intéressons à la conception et au développement d'une application multiplateforme dédiée au scan et à la numérisation de photos d'ECGs, basée sur les dernières technologies Web et les standards du traitement d'images.

L'électrocardiographie reste un examen de référence pour le diagnostic des maladies cardiovasculaires. Cependant, de nombreux ECG sont encore enregistrés sous forme papier, ce qui limite leur accessibilité, leur partage et leur analyse approfondie. La numérisation de ces documents peut constituer donc un enjeu majeur pour améliorer la qualité des soins et la recherche médicale.

Nous introduisons de nouvelles techniques qui se sont avérées efficaces même sur des images de faible qualité ou présentant du bruit, grâce à une approche innovante se passant de l'intelligence artificielle atteignant ainsi des temps d'exécution très bas.

Une méthodologie de conception robuste et l'utilisation des dernières technologies nous ont permis la création d'une application accessible, efficace et moderne.

Ce mémoire est structuré en plusieurs chapitres, abordant chacun un aspect crucial du développement de l'application :

1. **Généralités** : Nous introduisons dans ce chapitre les concepts clés liés aux applications informatiques, leur architecture, les notions de base de l'électrocardiographie ainsi qu'une étude des solutions existantes ;

2. **Traitement d'images** : Nous y détaillons les techniques de traitement d'images utilisées en informatique ;

3. **Conception et Réalisation** : Nous démontrons la méthodologie entreprise à la conception de l'application et son importance dans l'implémentation des fonctionnalités couvrant les exigences des utilisateurs ;

4. **Scan et numérisation d'ECGs** : Nous y décrivons les étapes du processus de scan

et numérisation des signaux ECG, tout en détaillant les limites de l'approche que nous avons choisie.

À travers cette étude, nous visons à démontrer comment les technologies modernes peuvent être appliquées au domaine médical pour offrir des solutions à la fois performantes et accessibles.

Chapitre 1

Généralités

Introduction

Dans un premier temps, nous explorerons les notions essentielles de l'électrocardiographie, en commençant par sa définition et l'explication de l'électrocardiogramme (ECG), suivi des dérivations utilisées pour l'enregistrement des signaux cardiaques.

Ensuite, nous aborderons les caractéristiques des signaux en général, avec une distinction entre les différents types de signaux, ainsi que les propriétés de l'image numérique, un élément central dans le processus de numérisation des ECG. Cela fournira les bases nécessaires pour comprendre les techniques de traitement et de numérisation des données médicales.

Enfin, une étude des solutions existantes sera menée pour analyser les dispositifs et applications déjà disponibles sur le marché. Cette étude comparative mettra en évidence les forces et limites des solutions actuelles, avant de présenter notre propre solution.

1.1 Application informatique

Une application est, dans le domaine informatique, un programme directement utilisé pour réaliser une tâche, ou un ensemble de tâches élémentaires d'un même domaine ou formant un tout.

1.1.1 Architecture d'une application

Toute application est constituée de trois parties [1] :

1. Interface utilisateur : elle constitue le seul point d'entrée entre l'application et l'utilisateur, une action sur un élément de l'interface utilisateur déclenche dans l'application un processus.

2. Processus : une activité de l'application qui vise à réaliser une tâche demandée par l'utilisateur, elle nécessite un processeur et une mémoire afin d'être exécutée.

3. Données : les informations manipulées par l'utilisateur à travers les processus, ce sont des données d'un phénomène de la vie réelle telles qu'un prénom ou une liste de produits. Elles peuvent être stockées dans des fichiers ou dans une base de données.

Architecture Client-Serveur

Il s'agit d'un découpage d'une application en deux composants [1] :

1. Le Client, qui est le composant qui se trouve du côté de l'utilisateur ;
2. Le Serveur, qui est celui qui se trouve plus proche des données.

Cette architecture fonctionne sous la forme d'un dialogue, le Client contacte le Serveur toujours en premier via une requête. Le Serveur reçoit ensuite la requête, exécute le processus et envoie une réponse au client.

1.1.2 Application multiplateforme

Une application multiplateforme est une application informatique qui peut être utilisée indépendamment du système d'exploitation, de l'appareil ou de la plateforme sur laquelle elle est exécutée.

1.1.3 Application Web

Une application Web est une application Client-Serveur où la partie Client s'exécute sur un navigateur Web d'où elle envoie des requêtes vers un Serveur Web qui est une machine comportant les processus, données et généralement un Serveur de Gestion de Bases de Données (SGBD) [1].

Elle constitue aussi une application multiplateforme vu qu'elle nécessite qu'un navigateur Web pour être exploitée par l'utilisateur.

1.1.4 Application Ajax

Ajax (pour Asynchronous JavaScript and XML) désigne un ensemble de technologies permettant de développer des applications Web plus interactives et réactives. Grâce à Ajax, les échanges de données avec le serveur se font de manière asynchrone, sans recharger entièrement la page web, ce qui améliore l'expérience utilisateur. Les principales technologies utilisées dans Ajax incluent [2] :

- **XHTML et CSS** : Pour la présentation de l'information.
- **Document Object Model (DOM)** : Pour l'interaction dynamique avec les éléments de la page.
- **Objet XMLHttpRequest** : Pour échanger des données de manière asynchrone avec le serveur.
- **XML, HTML, et XSLT** : Pour l'échange et la manipulation des données.
- **JavaScript** : Pour lier les requêtes de données et l'affichage de l'information.

Contrairement aux applications web traditionnelles où chaque interaction utilisateur déclenche une requête HTTP qui recharge toute la page, Ajax permet de traiter les données en arrière-plan tout en maintenant l'interactivité offrant ainsi une expérience plus fluide, similaire à celle d'une application de bureau.

Afin de réaliser notre application, nous avons opté pour le développement d'une application Ajax dont nous détaillerons l'implémentation dans la partie Réalisation.

1.2 L'électrocardiographie

En 1902, le physiologiste néerlandais Willem Eintoven a enregistré le premier électrocardiogramme (ECG) chez l'homme. Depuis, le nombre de voies d'enregistrement est passé de 3 à 12, et les instruments d'enregistrement sont devenus des enregistreurs sophistiqués, numériques automatisés, capables d'enregistrer, de mesurer et d'interpréter les signaux électrocardiographiques. Toutefois, les principes de base qui sous-tendent l'ECG sont inchangés. Il enregistre, à partir de la surface du corps, les gradients de potentiel créés lors des séquences de dépolarisation et repolarisation des cellules du myocarde [3].

1.2.1 Définition

L'électrocardiographie est un examen qui vise à mesurer et à enregistrer l'activité électrique du cœur d'un patient.

L'ECG est la technique la plus couramment utilisée pour détecter et diagnostiquer les maladies du cœur et suivre l'effet de certains traitements sur l'activité électrique du cœur. Il n'est pas invasif, pratiquement sans risque et relativement peu coûteux. Depuis son introduction, une grande base de données a été assemblée, corrélant le signal ECG enregistré à partir de la surface du corps à l'activité électrique sous-jacente de cellules individuelles cardiaques, d'une part, et à la présentation clinique du patient, de l'autre, offrant ainsi un aperçu du comportement électrique du cœur et de sa modification à la suite d'événements physiologiques, pharmacologiques et pathologiques [3].

1.2.2 L'électrocardiogramme

L'électrocardiogramme, ou ECG, est la représentation graphique sur papier millimétré des variations électriques de l'activité cardiaque dans le temps et l'espace.

Il est réalisé sur prescription médicale, sur protocole ou sur rôle propre infirmier lorsque celui-ci juge que la situation clinique du patient le nécessite [4].

Il est couramment noté sur l'ECG :

- L'identité du patient et sa date de naissance ;
- La date du jour et l'heure de réalisation de l'ECG ;
- Les observations éventuelles : douleur thoracique, contrôle après prise de traitement, réduction d'un trouble du rythme...

1.2.3 Dérivations de l'ECG

L'ECG conventionnel comporte 12 dérivations et est réalisé au moyen de 10 électrodes. L'activité électrique instantanée du cœur peut se résumer à un vecteur résultant principal (dipôle). Ce dernier est enregistré par les dérivations électrocardiographiques qui sont des lignes de tension reliant 2 points distincts [5].

Une dérivation correspond donc à la mesure d'une différence de potentiel via un galvanomètre entre deux électrodes placées au niveau de deux points différents du corps.

Les électrodes des membres sont appelées périphériques (éloignées du cœur) et explorent le cœur dans le plan frontal. On a 3 dérivations bipolaires ou standards obtenues grâce à 3 électrodes : une est posée au niveau du bras droit, une sur le bras gauche et la dernière sur jambe gauche. L'électrode sur la jambe droite est indifférente. Elles sont annotées :

- DI : connexion bras gauche-bras droit.
- DII : bras droit-jambe gauche.
- $DIII$: bras gauche-jambe gauche.

La lettre D pour dérivation n'est pas en usage dans les pays anglo-saxons qui les appellent tout simplement I, II et III. Selon l'hypothèse avancée par Einthoven, on suppose que ces dérivations décrivent un triangle équilatéral dont le centre est occupé par le cœur. Des 3 électrodes sur les membres, on obtient également 3 dérivations unipolaires amplifiées et désignées comme suit :

- aVR : pour le bras droit.
- aVL : pour le bras gauche.
- aVF : pour la jambe gauche.

Ces dérivations unipolaires sont obtenues par l'enregistrement de différences de potentiels entre l'électrode exploratrice (positive) qui détecte les différences de potentiel là où elle se trouve et une électrode neutre obtenue par l'artifice du Central Terminal de Wilson. Dérivations uni et bipolaires dans le plan frontal déterminent ensemble un double triaxe (6 axes au total) au centre duquel se trouve le cœur.

Les électrodes sur le thorax sont les précordiales et sont notées de V_1 à V_6. Ces électrodes unipolaires analysent l'activité électrique dans le plan horizontal. La disposition des précordiales est la suivante :

- V_1 : 4ième espace intercostal, côté droit du sternum.
- V_2 : 4ième espace intercostal, côté gauche du sternum.
- V_3 : équidistance de V_2 et V_4.
- V_4 : 5ième espace intercostal, ligne médio-claviculaire.
- V_5 : idem, ligne axillaire antérieure.
- V_6 : idem, ligne axillaire moyenne.

1.3 Signal

Un signal est toute grandeur physique susceptible de contenir de l'information. La représentation temporelle d'un signal est définie par une fonction $x(t)$, réelle ou complexe, de la variable réelle du temps t qui doit approcher "au mieux" les informations contenues dans le signal [6].

La dénomination fonction est prise au sens le plus général du terme, i.e. $x(t)$ peut être une suite discrète, une fonction de la variable continue ou une distribution. Cette fonction peut être une fonction au sens classique et dans ce cas, on dira qu'elle représente un signal déterministe. Elle peut aussi être une "fonction aléatoire" et dans ce cas elle représente un signal dit aléatoire ou stochastique.

Tout signal physique est forcément perturbé par un bruit plus ou moins important (bruit de mesure, influence des appareils utilisés, etc). Un signal physique est donc un signal aléatoire.

1.3.1 Types de signaux

Il existe 03 types de signaux :

1. Signal (à temps) continu : c'est un signal modélisable, la plupart du temps, par une fonction $x(t)$ continue par intervalles au sens mathématique du terme, ou par une distribution.
2. Signal (à temps) discret : c'est un signal dont un modèle peut être une fonction définie par l'ensemble des valeurs x,. qu'elle prend sur un ensemble dénombrable t_0, t_1, ... , t_n, ... de valeurs de la variable (la différence $t_{i+1} - t_i$ étant en général constante).
3. Signal numérique : c'est un signal ne pouvant prendre qu'un nombre fini ou dénombrable de valeurs distinctes.

Un signal d'ECG est un signal temporel continu stochastique électrique exprimé en mV dépendant du temps t que nous visons à transformer en signal numérique.

1.4 Image numérique

La création d'une image numérique est faite par un appareil de mesure (scanner, appareil photo numérique, webcam, barrette CCD, ...). Une image numérique est une fonction à support discret et borné, et à valeurs discrètes. Le support est multidimensionnel, en général 2D ou 3D. Les valeurs peuvent être scalaires (images en niveaux de gris), ou bien vectorielles (imagerie multi composante, imagerie couleur) [7].

Nous aurons à faire à des images 2D prises par un appareil photo numérique.

1.5 Étude du cas existant

Il est rare de trouver des applications dans le domaine de l'électrocardiographie, nous pouvons cependant jeter un œil aux deux plus populaires d'entre elles : PMcardio et GARMIN.

1.5.1 PMcardio

Sur leur site officiel, PMcardio présente sa réalisation à travers ces quelques paragraphes [8] :

> PMcardio est une application d'interprétation d'ECG alimentée par l'intelligence artificielle, conçue pour améliorer la précision et l'efficacité des diagnostics cardiaques. Contrairement aux algorithmes traditionnels basés sur des conditions (if-then-else), PMcardio utilise l'apprentissage automatique et les réseaux neuronaux profonds pour analyser des ECGs et identifier des anomalies subtiles souvent invisibles à l'œil humain.
>
> L'application a été entraînée sur un ensemble de données contenant plus de 900 000 ECGs provenant de 172 750 patients, permettant la détection de 39 pathologies cardiovasculaires distinctes en quelques secondes. PMcardio se distingue par sa rapidité, son efficacité et sa capacité à fournir des recommandations de traitement conformes aux dernières directives médicales.
>
> Elle est reconnue comme un dispositif médical de classe IIb dans l'Union européenne, avec une précision qui rivalise avec celle des cardiologues experts. Cette solution représente une avancée significative dans le domaine des soins d'urgence cardiaque, en particulier pour la détection précoce des infarctus du myocarde.

Cette application est payante et à but lucratif, elle partage cependant l'un de nos objectifs qui est le scan et la numérisation d'un ECG à travers un média numérique (photo ou vidéo). Elle a le mérite de s'appuyer sur les principes modernes de l'apprentissage automatique et les réseaux neuronaux, mais nous estimons que cette approche est risquée, car :

- Les approches "traditionnelle" de l'algorithmique procédurale possèdent un code source et sont faciles à l'amélioration et la maintenance ;
- Concernant le diagnostique fourni par l'application, si la numérisation est extrêmement précise, une approche rigoureuse mathématique ou statistique du signal de l'ECG semble bien plus fiable qu'un entraînement sur des milliers d'échantillons. En effet, l'entraînement d'un réseau de neurones se base sur des résultats déjà existants issus de travaux d'humains et donc susceptibles d'être erronés ;
- Les développeurs de l'application prétendent être capables de recommander la prise de traitement sans avis médical d'un professionnel. Cela constitue un très grand danger car pour commencer certains traitements, il est primordial de connaître la santé actuelle du patient, ses antécédents et ses autres maladies. Ces données ne sont pas forcément accessibles à l'application.
- Selon le ministère de la santé de France : [9] :

  > Les dispositifs médicaux sont classés en 4 catégories, en fonction de leur risque potentiel pour la santé. A chaque catégorie sont associées des règles d'évaluation et de contrôle spécifiques :
  > ...
  > Classe IIb (risque potentiel élevé/important) : ...

> ...
> La classification d'un dispositif médical est de la responsabilité du fabricant. Pour ce faire, le fabricant s'appuie sur des règles de classification établies par la directive DM, en fonction de la finalité médicale que ce dernier revendique pour son produit.

La classe IIb est la troisième classe de dispositif la plus risquée sur les quatre classes définies par l'Union Européenne ;

- Peu importe combien un programme de numérisation d'ECGs peut être efficace, il possèdera toujours des limites comme nous allons le voir plus tard, car la validité des résultats dépendent de la qualité des données d'entrées et donc de l'image fournie par l'utilisateur. Une image de mauvaise qualité est presque impossible à numériser avec une précision allant jusqu'à détecter des maladies ou prescrire des traitements. ;
- Rivaliser avec un professionnel de la santé n'est pas un argument convaincant car même un professionnel de la santé reste un être humain et est donc susceptible de faire des erreurs dues à la fatigue, la déconcentration ou un oubli. L'un des principaux buts de l'informatique est l'automatisation, et celle-ci doit être fiable et sans erreurs.

L'intelligence artificielle est un domaine encore en développement et loin d'être parfait. Il est vrai que son implication dans la vie quotidienne a révolutionné notre manière d'apprendre, travailler et passer son temps, mais la santé reste un domaine où une erreur peut avoir de lourdes conséquences et ne peut être confiée qu'à une automatisation (ou programme, en l'occurrence) extrêmement fiable. On en déduit qu'une approche déterministe (if - else - then) semble être plus adaptée dans un domaine aussi sensible que la santé.

1.5.2 GARMIN

Leur produit est présenté de manière concise sur leur site [10] :

> Garmin est une entreprise de technologie spécialisée dans les dispositifs GPS, les montres connectées et les équipements pour le sport et l'aviation. Elle se distingue par son innovation constante, notamment avec des fonctionnalités avancées comme l'application ECG (ECG app). Cette application permet de surveiller la santé cardiaque des utilisateurs en enregistrant les signaux électriques du cœur via une montre connectée compatible. L'application peut détecter des rythmes cardiaques irréguliers tels que la fibrillation auriculaire (AFib). Grâce à une étude clinique, l'application a démontré une grande précision dans la détection des rythmes cardiaques, en comparant les résultats à ceux d'un ECG standard.

Ce produit sort du cadre de l'ingénierie logicielle et presque de l'informatique en général. Cependant, sa fiabilité ne dépend pas de l'utilisateur car elle numérise les données d'entrée directement de la source qui est les battements du cœur, un signal analogique.

Les seuls obstacles à la précision de leurs résultats sont la présence de bruit lors de la capture des signaux analogiques ou les limites technique de leur dispositif : la montre connectée.

Leur atout réside aussi dans la polyvalence de leur produit. En effet, s'il s'agissait d'une montre connectée qui ne faisait que des ECGs, elle aurait eu du mal à se vendre car les gens

se diraient qu'ils préféreraient directement faire un ECG sur une machine spécialisée à haute précision. Par contre, étant livrée avec tout ses autres avantages, la capture de l'ECG devient un argument de vente comme étant un plus.

Conclusion

À travers ce projet, nous allons essayer de concevoir et réaliser une application Web gratuite de gestion d'ECGs, incluant le scan d'images d'ECGs et numérisation de ces images. Une approche purement algorithmique (déterministe), basée sur les techniques de traitement d'images sera utilisée.

Chapitre 2

Traitement d'images

Introduction

Les images, définies comme des fonctions bidimensionnelles, sont souvent traitées numériquement par des ordinateurs, constituant ainsi des images numériques. Le traitement d'images numériques englobe un large éventail d'applications, allant des images visibles aux ultrasons, en passant par la microscopie électronique.

Les frontières entre le traitement d'images, l'analyse d'images et la vision par ordinateur sont parfois floues. Cependant, on peut distinguer trois niveaux de traitement : basique, intermédiaire et avancé. Les processus de bas niveau impliquent des opérations primitives telles que la réduction du bruit et l'amélioration du contraste. Les processus intermédiaires incluent la segmentation, la description et la classification des objets dans une image. Les processus avancés impliquent la reconnaissance d'objets et des fonctions cognitives associées à la vision humaine.

Le traitement d'images inclut à la fois les processus où les entrées et les sorties sont des images, ainsi que ceux qui extraient des attributs des images, y compris la reconnaissance d'objets individuels. Ceci est illustré par l'analyse automatisée du texte, où le traitement d'images va de l'acquisition et la prétraitement de l'image à la reconnaissance des caractères individuels.

En résumé, le traitement d'images numériques est une discipline fondamentale utilisée dans de nombreux domaines d'une grande valeur sociale et économique [25].

2.1 Définition

Le traitement d'images est une discipline de l'informatique et des mathématiques appliquées qui étudie les images numériques et leurs transformations, dans le but d'améliorer leur qualité ou d'en extraire de l'information.

2.1.1 Origine du traitement de l'image

L'origine du traitement d'images remonte à l'industrie des journaux, où des images étaient envoyées par câble sous-marin entre Londres et New York dès les années 1920. L'introduction du système de transmission d'images par câble Bartlane a considérablement réduit le temps nécessaire pour transporter une image à travers l'Atlantique. Les premiers systèmes Bartlane

étaient capables de coder des images en cinq niveaux de gris, ce qui a été augmenté à 15 niveaux en 1929.

Le traitement d'images numériques a véritablement pris son essor dans les années 1960 avec le début du programme spatial. Les premières machines capables d'effectuer des tâches significatives de traitement d'images sont apparues à cette époque, et les leçons apprises lors du traitement d'images des missions spatiales ont jeté les bases des méthodes améliorées utilisées dans des domaines tels que la médecine, l'observation de la Terre et l'astronomie.

Depuis lors, le domaine du traitement d'images numériques a connu une croissance vigoureuse, avec des applications étendues dans des domaines tels que la médecine, l'industrie, les sciences biologiques, la géographie, l'archéologie, la physique, l'astronomie, la défense et l'application de la loi. En parallèle, les techniques de traitement d'images ont été appliquées à la perception artificielle, où l'intérêt se porte sur l'extraction d'informations d'une image, souvent dans une forme non visuelle utilisée pour le traitement informatique.

Aujourd'hui, les avancées continues dans le domaine du traitement d'images numériques sont rendues possibles par la baisse des prix des ordinateurs et l'expansion des réseaux de communication via Internet, offrant ainsi des opportunités sans précédent pour la croissance continue de ce domaine [25].

2.1.2 Domaines d'utilisations du traitement d'images

Le traitement numérique des images trouve une application étendue dans divers domaines, en exploitant différentes sources d'énergie pour générer des images. Voici un résumé condensé des différentes modalités d'imagerie discutées [25]:

- **Imagerie par les Rayons Gamma :** Utilisée en médecine nucléaire pour les scintigraphies osseuses, la tomographie par émission de positons (TEP), et les observations astronomiques telles que la Boucle du Cygne.
- **Imagerie par les Rayons X :** Largement utilisée dans les diagnostics médicaux (par exemple, radiographies thoraciques, angiographie, tomodensitométrie (TDM)), l'inspection industrielle et l'astronomie.
- **Imagerie dans la Bande Ultraviolette :** Appliquée en microscopie de fluorescence pour l'imagerie biologique et l'astronomie (par exemple, imagerie de la Boucle du Cygne).
- **Imagerie dans les Bandes Visible et Infrarouge :** Dominante en microscopie optique, télédétection (par exemple, imagerie LANDSAT pour la surveillance environnementale), observation et prédiction météorologiques (par exemple, image de l'ouragan Katrina).
- **Imagerie dans la Bande des Micro-ondes :** Principalement utilisée en radar pour l'exploration terrestre, fournissant des images détaillées non affectées par les conditions météorologiques.
- **Imagerie dans la Bande Radio :** Présente dans l'imagerie par résonance magnétique (IRM) pour les diagnostics médicaux et l'astronomie (par exemple, imagerie du Pulsar du Crabe).

- **Autres Modalités :** Imagerie acoustique, telle que l'exploration sismique en géologie, et la microscopie électronique pour l'imagerie détaillée des spécimens. De plus, les images générées par ordinateur, y compris les fractales et la modélisation 3D, offrent des techniques de visualisation polyvalentes.

Chaque modalité sert des objectifs spécifiques dans un large éventail de domaines, démontrant l'impact et les applications étendues du traitement numérique des images. Dans notre cas, le traitement d'images s'est avéré être utile dans le scan et la numérisation des ECG.

2.2 Échantillonage et quantification

La numérisation des images est une étape cruciale pour diverses applications telles que la visualisation sur un moniteur, l'impression, le traitement informatique, le stockage sur des supports numériques et la transmission sur des réseaux informatiques.

L'échantillonnage et la quantification sont les deux opérations essentielles au processus de numérisation d'images [27].

2.2.1 Échantillonage

L'échantillonnage est une étape permettant de convertir un signal continu en un signal discret, indispensable pour le traitement numérique. Cette conversion est réalisée à l'aide de dispositifs électroniques tels que les capteurs photographiques (comme les capteurs CCD ou CMOS) ou des scanners capables de numériser des images analogiques [27].

Les capteurs CCD (Charge-Coupled Device) sont largement utilisés dans les appareils photo et les scanners en raison de leur capacité à transformer la lumière reçue en un signal électrique. Ce signal est ensuite amplifié et numérisé, produisant une matrice numérique I, qui représente les valeurs de l'intensité lumineuse capturée pour chaque élément de l'image.

La théorie de l'échantillonnage cherche à retrouver les valeurs de la matrice I représentative de l'image :

- **Résolution horizontale (N ou x)** : nombre de lignes dans l'image.
- **Résolution verticale (M ou y)** : nombre de colonnes dans l'image.
- **Résolution spatiale** : donné par $N \times M$.
- **Densité de résolution** : nombre de pixels par unité de longueur. S'exprime en ppi (pixels per inch), dpi (dots per inch) ou en français ppp (pixels par pouce).

2.2.2 Quantification

La quantification est une étape cruciale dans la conversion d'une image analogique en une image numérique. Elle consiste à discrétiser les niveaux de gris ou de couleurs de l'image afin de les représenter par des valeurs numériques comprises dans un ensemble fini. Ce processus permet de stocker et de traiter l'image de manière informatique.

Principe de la quantification : la quantification d'une image se déroule en plusieurs étapes :

- **Choix du nombre de niveaux de quantification** : Le nombre de niveaux de quantification détermine la précision de la représentation numérique de l'image. Plus le nombre de niveaux est élevé, plus la précision est grande, mais plus la taille du fichier numérique est importante.
- **Détermination des niveaux de décision** : Les niveaux de décision représentent les valeurs numériques qui seront attribuées aux différents niveaux de gris ou de couleurs de l'image. La répartition de ces niveaux peut être uniforme ou non uniforme, en fonction de la distribution des valeurs dans l'image.
- **Remplacement des valeurs originales** : Pour chaque pixel de l'image, la valeur originale est remplacée par le niveau de décision le plus proche.

Types de quantification : on distingue deux principaux types de quantification :

- **Quantification uniforme** : Les niveaux de décision sont répartis de manière uniforme sur l'intervalle de valeurs possibles. Cette méthode est simple à implémenter mais peut ne pas être optimale pour toutes les images.
- **Quantification non uniforme** : Les niveaux de décision sont répartis en fonction de la distribution des valeurs dans l'image. Cette méthode permet d'obtenir une meilleure précision pour les images avec une distribution non uniforme des niveaux de gris ou de couleurs.

Conséquences de la quantification : la quantification introduit des erreurs dans la représentation numérique de l'image originale. Ces erreurs peuvent se manifester sous forme d'artefacts, tels que la trame et l'effet de postérisation.

- **Trame** : Effet de "escalier" visible sur les lignes diagonales ou courbes, dû à la discrétisation des coordonnées spatiales.
- **Effet de postérisation** : Réduction du nombre de couleurs apparentes dans l'image, ce qui donne un aspect "pixellisé".

Le choix de la technique de quantification vise à minimiser ces artefacts tout en préservant la qualité de l'image et en optimisant la taille du fichier numérique.

Applications de la quantification : la quantification d'images est utilisée dans de nombreux domaines, notamment :

- **Compression d'images** : La quantification est une étape essentielle dans la compression d'images, permettant de réduire la taille du fichier numérique sans trop sacrifier la qualité de l'image.
- **Traitement d'images** : La quantification est utilisée dans diverses techniques de traitement d'images, telles que l'amélioration du contraste, le filtrage et la segmentation.
- **Graphisme numérique** : La quantification est utilisée pour créer des images numériques à partir de modèles 3D ou de descriptions vectorielles.

2.3 Représentation des images

Une image numérique est représentée par un tableau ou matrice I de n lignes et m colonnes. n et m sont respectivement la largeur et la hauteur de l'image I [7].

Le pixel est un point de l'image désigné par un couple de coordonnées (i, j) où i est l'indice de ligne et j l'indice de colonne. Par convention, le pixel origine $(0, 0)$ est en général en haut à gauche. Ainsi, le nombre $I(i, j)$ est la valeur du pixel (i, j) [7].

Il est tout autant possible de définir une image comme une fonction $f(x, y)$ qui associe à un couple (x, y) la valeur d'un pixel v. Le cardinal de l'ensemble des entrées x étant la largeur de l'image N et le cardinal de l'ensemble des entrées y étant la hauteur de l'image M.

2.4 Représentation des couleurs

Lors de la capture d'une scène, la quantité de lumière convertie en signal électrique analogique et ensuite transformée en signal numérique par un convertisseur analogique-numérique (CAN). Puis, elle est représentée sous forme d'une matrice de valeurs comme expliqué plus haut [7].

La lumière

La lumière couvre une partie du spectre d'énergie électromagnétique. Un rayonnement électro-magnétique est en général constitué d'un certain nombre de longueurs d'onde (ou fréquences) que les dispositifs dispersifs de séparer en un spectre. Le spectre est soit discret, soit continu. Les longueurs d'onde du spectre visible s'étendent approximativement de 380 à $720nm$. Une source est caractérisée par :

1. Son rayonnement, mesurable dans un système de grandeur correspondant à l'action proprement visuelle ;
2. Par le mélange des longueurs d'onde de cette énergie, mélange qui produit une sensation de couleur.

La lumière est donc une distribution d'énergie émise à certaines fréquences ayant une certaine intensité. Pour caractériser une couleur monochromatique, il suffit de connaître sa longueur d'onde γ et la luminance L, expression qualitative de la brillance énergétique [28].

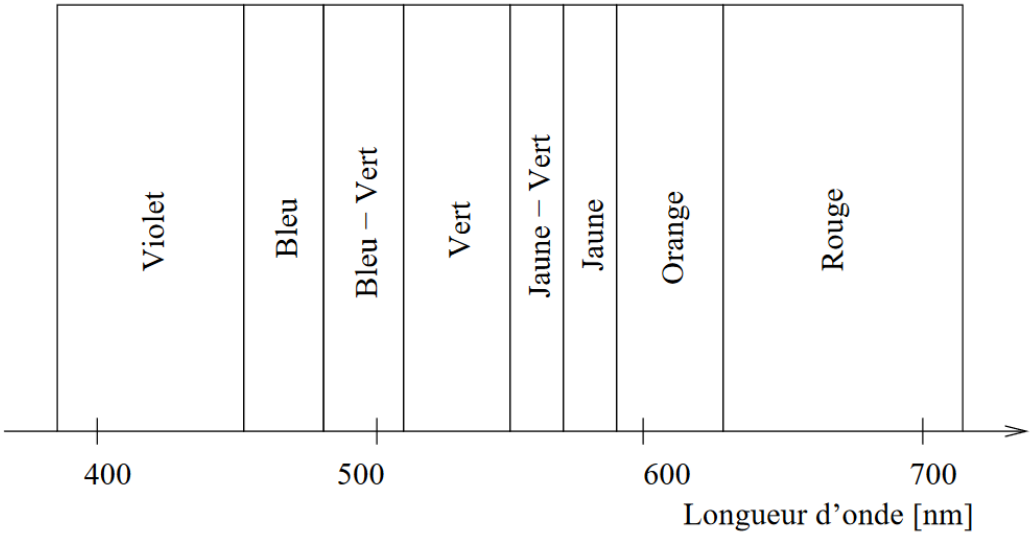


Figure 2.1: Les longueurs d'onde associées aux couleurs.[28].

Il existe plusieurs types de représentation des couleurs, on peut en citer [7] :

2.4.1 Image binaire

Une image binaire I_b est une image dont les valeurs des pixels sont un scalaire b possédant seulement deux valeurs possibles : 0 ou 1. En programmation, le 0 au "fond" ou "l'arrière-plan" tandis que le 1 est équivalent à "l'objet" ou "la forme".

$$I_b(i,j) = v \quad \text{avec} \quad v \in \{0,1\} \tag{2.1}$$

2.4.2 Niveaux de gris

Chaque pixel $I(i,j)$ est représente un nombre (scalaire) compris entre $[0, N_{\max}]$ avec $N_{\max}$ le nombre de nivaux de gris possibles.

$$I(i,j) \in \{0,1,\ldots,N_{\max}\} \tag{2.2}$$

Le noir est représente par la valeur 0 et le blanc par $N_{\max}$.

2.4.3 Espace RGB

C'est une synthèse additive qui utilise trois couleurs qui sont le rouge, le vert et le bleu (abrégé RVB en Français, équivalent à RGB pour red, green, blue en Anglais).

Pour chaque pixel $I(i,j)$, sa représentation est sous la forme de trois nombres (r,g,b) compris entre $[0, N_{\max}]$ avec $N_{\max}$ la quantité maximale de la composante r ou g ou b.

$$I(i,j) = (r,g,b)/ \quad r,g,b \in \{0,1,\ldots,N_{\max}\} \tag{2.3}$$

2.4.4 Espace luminance/chrominance

Les espaces luminance/chrominance possèdent 3 composantes déductible à partir du système RGB qui sont :

1. La luminance (Y) : qui est définie par l'intensité de la radiation lumineuse par unité de surface émettrice ; elle dépend des trois couleurs RGB.
2. La chrominance bleue (Cb).
3. La chrominance rouge (Cr).

Avec :

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{2.4}$$

2.4.5 Espace HSL

L'espace HSL (Hue, Saturation, Luminance, ou en français TSL pour teinte, saturation et luminance) est un modèle de représentation dit "naturel", c'est-à-dire proche de la perception physiologique de la couleur par l'œil humain, il consiste à décomposer la couleur en trois variables :

1. La teinte (H) : permet de déterminer la couleur souhaitée (rouge, vert, jaune, etc).
2. La saturation (S) : également appelée intensité; mesure la pureté des couleurs. Elle permet notamment de distinguer les couleurs "vives" des couleurs "pastels" ou "délavées".
3. La luminance (L) : également appelée brillance, indiquant la quantité de lumière de la couleur, c'est-à-dire son aspect clair ou sombre.

L'espace HSL est une représentation cylindrique de l'espace YCbCr décrit mathématiquement comme suit :

$$H = \begin{cases} \arctan\left(\frac{Cr}{Cb}\right) & \text{si } Cr \geq 0 \\ \pi + \arctan\left(\frac{Cr}{Cb}\right) & \text{si } Cr < 0 \end{cases} \tag{2.5}$$

$$S = \sqrt{Cr^2 + Cb^2} \tag{2.6}$$

$$L = Y \tag{2.7}$$

Il est important de noter qu'une luminance maximale (100%) est representée par la couleur blanche.

2.4.6 Espace HSV

Même que l'espace HSL, mais diffère dans la composante Valeur V (de l'anglais value) et la composante de la saturation S. En prenant un espace HSL avec les composantes H, S et L, nous obtenons un espace HSV avec les composantes h, s et v comme suit [29] :

$$h = H \tag{2.8}$$

$$s = \begin{cases} 0 & \text{si } V = 0 \\ 2(1 - \frac{L}{V}) & \text{sinon} \end{cases} \tag{2.9}$$

$$v = L + S \cdot \min(L, 1 - L) \tag{2.10}$$

(a) Spectre H.

(b) Spectre S_{HSL}, avec $H = 0$ et $L = 50\%$.

(c) Spectre S_{HSV}, avec $H = 0$.

(d) Spectre L_{HSL}, avec $H = 0$.

(e) Spectre V_{HSV}, avec $H = 0$.

Figure 2.2: Comparaison entre l'espace HSL et HSV.

2.5 Séparation par composantes

Pour les images possédant un ensemble de composantes (channels en anglais) comme valeur $v = \{c_1, c_2, c_3, \ldots, c_n\}$ telle que $c_i \in \{0, 1, \ldots, N_{\max}\}$, il est possible de séparer ces composantes en n images à valeur scalaires ainsi :

En partant d'une image I, pour chaque composante c_i nous définissons

$$I_{c_i}(i, j) = v(c_i) \quad \text{pour tout } v = p(i, j) \in I \tag{2.11}$$

Obtenant ainsi un ensemble de n images $I_{c_1}, I_{c_2}, I_{c_3}, \ldots, I_{c_n}$ qui ont pour valeur de pixel un scalaire $c \in \{0, 1, \ldots, N_{\max}\}$.

Exemple Prenons maintenant un exemple avec une image en couleurs au format RGB. Dans ce cas, chaque pixel de l'image I a une valeur $v = \{R, G, B\}$, où $R, G, B \in \{0, 1, \ldots, 255\}$.
L'image I peut être décomposée en trois images I_R, I_G, I_B correspondant aux composantes Rouge, Vert et Bleu respectivement. Mathématiquement, cela se traduit par :

$$I_R(p) = R \quad \text{pour tout } p \in \mathcal{P}$$
$$I_G(p) = G \quad \text{pour tout } p \in \mathcal{P}$$
$$I_B(p) = B \quad \text{pour tout } p \in \mathcal{P}$$

où R, G, B sont les valeurs numériques des composantes rouge, verte et bleue respectivement pour le pixel p et $\mathcal{P}$ étant l'ensemble des pixels de l'image I.

Transformation en niveaux de gris

Une transformation en niveaux de gris est tout simplement une extraction de la composante Y (ou L de l'espace HSL) d'une image YCrCb ou HSL.

En partant d'une image I_{YCrCb} (en YcrCb) dont on a extrait la composante Y vers une image I_Y (ou de la même image en HSL I_{HSL} dont on a extrait la composante I_L) l'image en niveau de gris I_g vaut :

$$I_g = I_Y = I_L \tag{2.12}$$

Il est facile d'obtenir un niveau de gris d'une image de n'importe quel espace de couleur car il suffit de la converir en YCrCb ou en HSL. Ainsi, une image en niveau de gris I_g d'une image I_{RGB} (en RGB) séparé en trois composantes I_R, I_G et I_B est représentée ainsi :

$$I_g = 0.299 \cdot I_R(i,j) + 0.587 \cdot I_G(i,j) + 0.114 \cdot I_B(i,j) \tag{2.13}$$

2.6 Histogramme

L'histogramme d'une image est la courbe représentant la fréquence des occurrences de valeurs scalaires présentes dans une image [28].

Pour obtenir un histogramme, la valeur des pixels de l'image doit être scalaire et non un ensemble de composantes (vecteur) comme dans le cas de l'espace RGB.

Soit I une image représentée par une matrice de taille $M \times N$ avec des valeurs de pixel $I(i,j) \in \{0, 1, \ldots, N_{\max}\}$. L'histogramme h de l'image I est une fonction qui retourne le nombre d'occurrences de chaque valeur de pixel v dans l'image.

$$h(v) = \sum_{i=1}^{M} \sum_{j=1}^{N} \delta(I(i,j) - v) \quad \text{pour} \quad v \in \{0, 1, \ldots, N_{\max}\} \tag{2.14}$$

où δ est la fonction delta de Kronecker définie par :

$$\delta(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{sinon} \end{cases} \tag{2.15}$$

En résumé, $h(v)$ représente le nombre de pixels dans l'image I ayant la valeur v.

2.6.1 Normalisation d'histogramme

L'histogramme normalisé d'une image I de taille $N \times M$ retourne la probabilité d'apparition d'une valeur v dans un histogramme, ceci est représente comme suit par $p(v)$:

$$p(v) = \frac{h(v)}{N \cdot M} \tag{2.16}$$

On en déduit que $p(v) \in [0, 1]$.

2.6.2 Interprétation d'histogrammes

La forme de l'histogramme est liée à l'apparence de l'image (à valeur de pixels salaire). Ceci s'illustre notamment dans certains cas comme [25]:

- **Image Sombre** : Dans l'image sombre, les valeurs les plus peuplés de l'histogramme sont concentrés à l'extrémité inférieure (sombre) de l'échelle des intensités.
- **Image Claire** : De même, les valeurs les plus peuplées de l'image claire sont biaisées vers l'extrémité supérieure de l'échelle.
- **Faible contraste** : Une image avec un faible contraste a un histogramme étroit situé typiquement vers le milieu de l'échelle des intensités, comme le montre la figure 3.16(c). Pour une image monochrome, cela implique un aspect gris terne et délavé.
- **Fort contraste** : Enfin, les composantes de l'histogramme de l'image à fort contraste couvrent une large gamme de l'échelle des intensités, et la distribution des pixels n'est pas trop éloignée de l'uniformité, avec peu de valeurs étant beaucoup plus hautes que les autres.

Intuitivement, il est raisonnable de conclure qu'une image dont les pixels tendent à occuper toute la gamme des niveaux d'intensité possibles et, en plus, tendent à être distribués uniformément, aura une apparence de fort contraste et présentera une grande variété de tons de gris. L'effet net sera une image montrant un grand nombre de détails en niveaux de gris et possédant une large gamme dynamique. Il est possible de développer une fonction de transformation qui peut atteindre cet effet automatiquement, en utilisant seulement l'histogramme d'une image d'entrée.

2.6.3 Égalisation d'histogramme

L'égalisation d'un histogramme se fait via des opérations mathématiques comme suit [30] :

Considérons une image en niveaux de gris $\{x\}$ discrète et notons $h(v)$ le nombre d'occurrences du niveau de gris $i \in [0, L[$. La probabilité d'occurrence d'une valeur i est donnée par $p(i)$. Définissons la fonction de distribution cumulative (CDF) correspondant à i comme :

$$\mathrm{cdf}_x(i) = \sum_{j=0}^{i} p(x = j) \tag{2.17}$$

qui est aussi l'histogramme accumulé et normalisé de l'image.

Nous souhaitons créer une transformation de la forme $y = T(x)$ pour produire une nouvelle image I_{eq} avec un histogramme plat. Une telle image aurait une fonction de distribution cumulative (CDF) linéarisée sur la plage de valeurs, c'est-à-dire :

$$\text{cdf}_y(i) = (i+1) \times K \quad \text{pour} \quad 0 \leq i < L \tag{2.18}$$

pour une constante K. Les propriétés de la CDF nous permettent d'effectuer une telle transformation (fonction de répartition inverse); elle est définie comme :

$$y = T(k) = \text{cdf}_x(k) \tag{2.19}$$

où k est dans l'intervalle $[0, L-1]$. Remarquez que T attribue les niveaux sur l'intervalle $[0, 1]$, puisque nous avons utilisé un histogramme normalisé de $\{x\}$. Pour attribuer les valeurs à leur plage originale, la transformation suivante doit être appliquée au résultat :

$$y' = y \cdot (\max\{x\} - \min\{x\}) + \min\{x\} = y \cdot (L-1). \tag{2.20}$$

Un y est une valeur réelle tandis que y' doit être un entier. Une méthode intuitive et populaire consiste à appliquer l'opération d'arrondissement :

$$y' = [y \cdot (L-1)]\,. \tag{2.21}$$

Cependant, une analyse détaillée aboutit à une formulation légèrement différente. La valeur attribuée y' devrait être 0 pour l'intervalle $0 < y \leq \frac{1}{L}$, et $y' = 1$ pour $\frac{1}{L} < y \leq \frac{2}{L}$, $y' = 2$ pour $\frac{2}{L} < y \leq \frac{3}{L}$, ..., et enfin $y' = L-1$ pour $\frac{(L-1)}{L} < y \leq 1$. La formule de quantification de y à y' devrait donc être :

$$y' = \lfloor L \cdot y - 1 \rfloor. \tag{2.22}$$

N.B. : $y' = -1$ lorsque $y = 0$, cependant, cela ne se produit pas car $y = 0$ signifie qu'il n'y a aucun pixel correspondant à cette valeur. Ainsi, nous obtenons $I_{\text{eq}} = y'$.

2.7 Transformée de Fourier et Spectre

Une image numérique est définie comme une fonction discrète à deux variables (x, y) ou encore un signal bidimensionnel. Ce signal peut être représenté soit dans le domaine spatial, soit dans le domaine fréquentiel. La représentation fréquentielle ou le spectre d'une image est obtenue à l'aide de la transformée de Fourier bidimensionnelle de l'image.

2.7.1 La Transformée de Fourier

La transformée de Fourier d'un signal sert à le décomposer en une somme de fonctions exponentielles complexes de fréquences variables. Dans le cas des signaux ou fonctions à deux variables, les transformées de Fourier directe et inverse (TF-2D) sont définies comme suit [7] :

$$\text{TF}[f(x,y)] = F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}dxdy \tag{2.23}$$

$$\mathrm{TF}^{-1}[F(u,v)] = f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{j2\pi(ux+vy)}dudv \tag{2.24}$$

Ici, x et y sont des coordonnées spatiales, tandis que u et v sont des coordonnées spectrales. En utilisant la transformée de Fourier d'une fonction bidimensionnelle, on peut définir deux spectres (amplitude et phase) par les expressions suivantes [7] :

Le spectre d'amplitude est :

$$F(u,v) = \sqrt{R(u,v)^2 + I(u,v)^2} \tag{2.25}$$

Le spectre de phase est :

$$\theta(u,v) = \arctan\left(\frac{I(u,v)}{R(u,v)}\right) \tag{2.26}$$

où $R(u,v) = \text{Réelle}[F(u,v)]$ et $I(u,v) = \text{Imaginaire}[F(u,v)]$.

2.7.2 La Transformée de Fourier Discrète

Pour les images numériques, qui sont des fonctions discrètes bidimensionnelles, la transformée de Fourier discrète bidimensionnelle (TFD-2D) doit être utilisée pour obtenir la représentation spectrale de l'image. La TFD-2D d'une image $f(x,y)$ est obtenue après discrétisation de la transformée de Fourier définie précédemment [7] :

$$F(u,v) = \frac{1}{N \cdot M}\sum_{x=0}^{N-1}\sum_{y=0}^{M-1} f(x,y)e^{-j2\pi\left(\frac{ux}{N}+\frac{vy}{M}\right)} \tag{2.27}$$

La TFD inverse est donnée par :

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{M-1} F(u,v)e^{j2\pi\left(\frac{ux}{N}+\frac{vy}{M}\right)} \tag{2.28}$$

où $u = 0, 1, \ldots, N-1$ et $v = 0, 1, \ldots, M-1$.

2.8 Filtrage des images

Le filtrage d'images est une technique de traitement d'images qui permet de manipuler ou améliorer les caractéristiques d'une image. Nous distinguons trois types de filtrages, le filtrage spatial (fait sur les images en tant que matrices, généralement par convolutions), le filtrage fréquentiel (fait sur les signaux, c'est-à-dire sur les transformations d'images en signaux) et le filtrage morphologique (fait généralement sur les images binaires).

2.8.1 Convolution

L'opération de la convolution (notée $*$) est une opération mathématique binaire. Une convolution d'une fonction continue f par une fonction continue g est donnée par [27] :

$$f * g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)g(u-x, v-y)\,du\,dv \tag{2.29}$$

Tandis que pour les fonctions discrètes :

$$f * g(x,y) = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} f(x,y)g(u-x, v-y) \tag{2.30}$$

Dans le cas du traitement d'images, une convolution d'une image I_{src} de dimensions finies s'effectue par une matrice (généralement carrée) à dimensions toutes deux impaires $p \times q$ appelée noyau de convolution notée K (pour Kernel en anglais) donne une image I_{dst} calculée de la manière suivante :

$$I_{\text{dst}}(i,j) = \sum_{x=0}^{p} \sum_{y=0}^{q} I_{\text{src}}(i+x-1, j+y-1) \cdot K(x,y) \tag{2.31}$$

Dans les images numériques, les valeurs de de la matrice I (notées $p(x,y)$) sont soit scalaires et donc appartient à un intervalle $\{0, 1, \ldots, N_{\text{max}}\}$ ou un vecteur (de composantes scalaires respectant la même contrainte) avec N_{max} étant généralement 255 (codé sur 08 bits). Ainsi, un résultat r négatif sera ramené à 0 et un résultat supérieur à N_{max} sera ramené à N_{max}.

$$p(x,y) = \begin{cases} 0 & \text{si } r < 0 \\ N_{\text{max}} & \text{si } r > N_{\text{max}} \\ r & \text{sinon} \end{cases} \tag{2.32}$$

Ou plutôt :

$$p(x,y) = max(0, min(r, N_{\text{max}})) \tag{2.33}$$

2.8.2 Filtrage fréquentiel

Les filtres fréquentiels peuvent être classés en différents types, tels que [31] :

- **Filtre passe-bas** : supprime les fréquences hautes et laisse passer les basses. La bande-passante est définie par la fréquence de coupure f_c, qui est la fréquence à partir de laquelle les fréquences sont atténuées.
- **Filtre passe-haut** : supprime les fréquences basses et laisse passer les hautes. La bande-passante est définie par la fréquence de coupure f_c, qui est la fréquence à partir de laquelle les fréquences sont atténuées.
- **Filtre passe-bande** : laisse passer les fréquences comprises entre deux fréquences de coupure f_{c1} et f_{c2}.

2.8.3 Filtrage morphologique

Le filtrage morphologique repose sur la morphologie mathématique, qui se base sur une description ensembliste des images. Il se concentre sur la forme des objets dans l'image plutôt que sur les valeurs d'amplitude des pixels.

Opérateurs binaires classiques

Les opérateurs binaires de base utilisés dans les images binaires sont :

- **ET** (AND) : La sortie est à 1 uniquement si les deux entrées sont à 1.
- **OU** (OR) : La sortie est à 1 si au moins une des deux entrées est à 1.
- **NON** (NOT) : La sortie est l'inverse de l'entrée.

Opérateurs morphologiques de base

Deux opérateurs morphologiques de base sont utilisés :

- **Érosion** : Notée $X \ominus B$, l'érosion d'une forme X par un élément structurant B est l'ensemble des pixels où l'élément structurant B est complètement inclus dans X. Elle tend à réduire les formes en supprimant les pixels isolés et en érodant les contours.
- **Dilatation** : Notée $X \oplus B$, la dilatation d'une forme X par un élément structurant B est l'ensemble des pixels où l'élément structurant B a une intersection non nulle avec X. Elle tend à augmenter les formes en comblant les trous et en dilatant les contours.

Opérateurs morphologiques complémentaires

En combinant les opérateurs de base, on obtient deux autres opérateurs :

- **Ouverture** : Notée $X \circ B$, l'ouverture est une érosion suivie d'une dilatation. Elle permet de lisser les formes, de supprimer les petits détails et de découper les isthmes étroits.
- **Fermeture** : Notée $X \bullet B$, la fermeture est une dilatation suivie d'une érosion. Elle permet de lisser les formes, de combler les petits trous et de connecter les composants proches.

Élément structurant

L'élément structurant B est une forme de référence utilisée pour comparer localement le signal d'image. Il peut être symétrique ou non, et sa forme influence le résultat des opérations morphologiques.

Exemples d'application

Sur une image binaire de référence :

- **Érosion** : Réduit les formes en supprimant les pixels isolés et en érodant les contours.
- **Dilatation** : Augmente les formes en comblant les trous et en dilatant les contours.
- **Ouverture** : Equivalent à une érosion suivie d'une dilatation, lisse les contours en supprimant les petits détails.
- **Fermeture** : Equivalent à une dilatation suivie d'une érosion, lisse les contours en comblant les petits trous et en fusionnant les objets proches.

2.9 Segmentation

La segmentation d'image est une technique qui divise une image numérique en groupes de pixels distincts appelés segments afin de faciliter la détection d'objets.

2.9.1 Segmentation par seuillage

Le seuillage est une fonction qui associe à une image I une image I_B binaire suivant une ou plusieurs conditions.

Seuillage global

Le seuillage global d'une image I est paramétré par une valeur T (de threshold qui signifie seuil en anglais) donne une nouvelle image I_B binarisée (c'est-à-dire possédant seulement deux couleurs) obtenue comme suit :

$$I_B(i,j) = \begin{cases} 1 & \text{si } I(i,j) > T \\ 0 & \text{sinon} \end{cases} \tag{2.34}$$

Seuillage par la méthode d'Otsu

La méthode d'Otsu est une technique de suillage introduite par Nobuyoki Otsu en 1979 dans son papier officiel [32].

Cette méthode cherche à déterminer automatiquement un seuil de segmentation en maximisant la variance inter-classes d'un histogramme de niveaux de gris. La procédure se déroule comme suit :

1. Calculer l'histogramme et les probabilités de chaque niveau de gris.
2. Initialiser les paramètres pour les classes en dessous et au-dessus du seuil.
3. Parcourir tous les niveaux de gris possibles comme seuil potentiel.
4. Pour chaque seuil potentiel, calculer les probabilités de classes et les moyennes des niveaux de gris pour les deux classes.
5. Calculer la variance inter-classes pour chaque seuil potentiel.
6. Le seuil optimal est celui qui maximise cette variance inter-classes.

La formule pour la variance inter-classes $\sigma_B^2(t)$ est donnée par :

$$\sigma_B^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2 \tag{2.35}$$

où $w_1(t)$ et $w_2(t)$ sont les probabilités des deux classes séparées par le seuil T, et $\mu_1(t)$ et $\mu_2(t)$ sont les moyennes des niveaux de gris des deux classes.

Seuillage adaptatif

Le seuillage adaptatif (Adaptive thresholding en anglais), ou seuillage dynamique, est une technique de seuillage d'image qui adapte le seuil de manière dynamique sur toute l'image, contrairement au seuillage global qui utilise un seuil fixe pour tous les pixels. Cette méthode est particulièrement utile pour les images présentant des variations d'éclairage, telles que des gradients d'illumination ou des ombres.

Le seuillage adaptatif prend en entrée une image en niveaux de gris ou en couleur et produit une image binaire représentant la segmentation. Pour chaque pixel i, j, un seuil $T(i, j)$ est calculé. Si la valeur du pixel $I(i, j)$ est inférieure au seuil $T(i, j)$, il est défini comme arrière-plan (0), sinon il est défini comme premier plan (1).

Il existe deux approches principales pour déterminer ce seuil [33]:

- **Approche de Chow et Kaneko** : Cette méthode divise l'image en sous-images chevauchantes et trouve le seuil optimal pour chaque sous-image en analysant son histogramme. Le seuil pour chaque pixel est interpolé à partir des résultats des sous-images. Cette méthode est précise mais coûteuse en calcul et inadaptée aux applications en temps réel.
- **Seuillage Local** : Cette approche examine statistiquement les valeurs d'intensité du voisinage local de chaque pixel. Les statistiques couramment utilisées incluent la moyenne des intensités locales (μ), la valeur médiane (med) ou la moyenne des valeurs minimales et maximales ($\frac{\min+\max}{2}$). Le seuil est alors défini comme suit :

$$T(i, j) = \mu(i, j) - C \tag{2.36}$$

où C est une constante ajustable.

Calcul du Seuil Local Pour un pixel (i, j), le seuil $T(i, j)$ est calculé en utilisant une fenêtre de voisinage de taille $N \times N$. Par exemple, pour une fenêtre de 7×7, on considère les pixels dans un voisinage de 3 pixels autour de (i, j) :

$$\text{Voisinage}(i, j) = \{I(i + k, j + l) \mid -3 \leq k, l \leq 3\}$$

La moyenne des valeurs d'intensité dans ce voisinage est :

$$\mu(i, j) = \frac{1}{N^2} \sum_{k=-3}^{3} \sum_{l=-3}^{3} I(i + k, j + l) \tag{2.37}$$

Le seuil local est alors ajusté par une constante C :

$$T(i, j) = \mu(i, j) - C \tag{2.38}$$

2.9.2 Segmentation basée sur la détection des contours

L'algorithme de détection des contours le plus largement utilisé est celui de Satoshi Suzuki and Keiichi Abe dans leur papier officiel [34] (datant de 1985) et est résumé comme suit :

1. L'algorithme prend en entrée une image binaire ;
2. L'objectif est de détecter et de suivre les contours non encore explorés de l'image ;
3. Pour chaque pixel objet de l'image, s'il ne fait pas partie d'un contour :
   (a) Créer un contour ;
   (b) Rechercher le pixel suivant appartenant au contour ;
   (c) Ajouter le pixel au contour ;
   (d) Marquer le pixel comme visité ;
   (e) Suivre le contour tant qu'il n'est pas fermé.
4. Une fois le contour complètement suivi, le stocker dans la liste des contours.

Il est souvent parallélisé malgré sa nature séquentielle par division de l'image en grilles égales et ensuite reliée une par une en reliant les bordures qui se touchent.

2.9.3 Segmentation par le Filtre Canny

Le filtre de Canny, développé par John Canny en 1986, est un algorithme utilisé pour la détection des contours en traitement d'images [35]. Il est conçu pour être optimal selon trois critères principaux : une bonne détection (faible taux d'erreur dans la signalisation des contours), une bonne localisation (minimisation des écarts entre les contours détectés et les contours réels) et une réponse claire (une seule réponse par contour et absence de faux positifs).

La mise en œuvre du filtre de Canny se déroule en plusieurs étapes :

- **Réduction du bruit :** Avant de détecter les contours, un flou Gaussien est appliqué pour éliminer les pixels isolés qui pourraient générer de faux positifs lors du calcul du gradient.
- **Calcul du gradient d'intensité :** Après le filtrage, l'intensité des contours est calculée en utilisant un gradient, qui détermine les changements d'intensité dans les directions X et Y de l'image.
- **Direction des contours :** La direction des contours est ensuite déterminée par l'angle formé entre les gradients selon les axes X et Y.
- **Suppression des non-maxima :** Seuls les points correspondant à des maxima locaux de l'intensité du gradient sont conservés, les autres sont supprimés pour éviter les faux contours.
- **Seuillage à hystérésis :** Cette étape consiste à appliquer deux seuils (haut et bas) pour différencier les véritables contours des autres éléments. Les points avec une intensité de gradient supérieure au seuil haut sont retenus comme contours, ceux en dessous du seuil bas sont rejetés, et ceux entre les deux sont retenus s'ils sont connectés à un point déjà accepté comme contour.

Les performances de l'algorithme dépendent des paramètres choisis, notamment la taille du filtre Gaussien et les seuils utilisés pour le seuillage. Un filtre plus grand réduit la sensibilité au bruit mais peut compromettre la précision de la localisation des contours.

Conclusion

Ce chapitre a posé les fondations théoriques et pratiques du traitement d'images, indispensables à la numérisation des signaux ECG à partir d'enregistrements papier.

En explorant les concepts clés de l'acquisition d'images, de la représentation des couleurs et des techniques de filtrage et de segmentation, nous avons jeté les bases d'un pipeline de traitement robuste.

Le prochains chapitres approfondira l'application de ces concepts à des cas concrets, en présentant des algorithmes spécifiques pour la segmentation des signaux et leur numérisation. L'objectif ultime étant de développer un outil capable de transformer des ECG papier en données numériques exploitables pour une analyse approfondie.

Chapitre 3

Conception et Réalisation

Introduction

Dans ce chapitre, nous allons nous intéresser à la conception et à la réalisation concrète d'une application informatique capable de réaliser les opérations de scan et numérisation d'un ECG à partir d'une photo de manière automatisée.

Nous aborderons les différentes étapes de développement, depuis l'analyse des besoins jusqu'à la mise en œuvre technique. Nous présenterons les choix technologiques effectués, la modélisation du système et la méthodologie de développement adoptée.

L'objectif est de fournir une vue d'ensemble du processus de création d'une application complexe, en mettant l'accent sur les aspects spécifiques à notre problématique.

Conception

La conception est une étape cruciale du cycle de développement, où les spécifications fonctionnelles et techniques d'un système sont transformées en une architecture détaillée. Cette phase permet de définir la structure du logiciel, ses composants, ainsi que leurs interactions.

3.1 Cahier de charge

Afin de concevoir notre application, que nous allons appeler myECG, nous avons imaginé le cahier de charge suivant :

> L'application myECG est une application Web multiplatforme dont le but est de scanner et numériser des ECGs.
>
> Le scan et la numérisation d'un ECG nécessitent la création d'un compte au sein de l'application en fournissant un nom d'utilisateur, une adresse e-mail, une date de naissance ainsi que le sexe de la personne inscrite.
>
> À partir de son adresse e-mail, un utilisateur peut réinitialiser son mot de passe au cas où il l'oublie.
>
> En s'authentifiant, l'utilisateur de l'application doit être en mesure de soumettre des photos d'ECGs à partir de son appareil et obtenir les versions scannées de ces derniers. S'il le souhaite, il peut poursuivre la procédure et numériser cet ECG scanné après avoir fourni la date de l'ECG et une courte note facultative personnalisée concernant son examen médical ou son état de santé.
>
> Après le processus de scan ou de la numérisation, l'utilisateur est en capacité de signaler l'échec de l'application à accomplir l'une de ces tâches.
>
> L'utilisateur pourra consulter ses ECGs numérisés et visualiser les dérivations sur son appareil. S'il le souhaite, il peut importer cette version numérisée de l'ECG vers son appareil.
>
> Des profils au sein d'un même compte peuvent être créés, chacun pouvant contenir un ensemble d'ECGs. Les mêmes informations demandées lors de l'inscription sont attendues d'un profil, en plus d'un pseudonyme facultatif.
>
> Lors de la création d'un compte, un profil est automatiquement créé à l'utilisateur ayant terminé son inscription. Ce profil se nomme "profil de base" et est rattaché aux informations du possesseur du compte.

Le profil de base ne peut être modifié, ni supprimé, mais il reflète plutôt les modifications apportées au compte dont il fait partie. Cependant, tous les autres profils peuvent être modifiés ou supprimés.

L'utilisateur peut choisir de partager ses ECGs anonymement à des entités de confiance ayant besoin d'une banque de données pour la recherche scientifique.

Ce cahier de charge nous servira de base à l'analyse des besoins utilisateur.

3.2 Analyse des besoins

L'analyse des besoins comprend trois étapes principales. L'analyse détaillée du besoin vise à définir précisément et exhaustivement le besoin, en utilisant divers outils d'analyse pour le modéliser et le rendre compréhensible. Le résultat est un document rédigé dans un langage accessible à celui qui a exprimé le besoin pour validation. Ensuite, l'analyse fonctionnelle définit les fonctions que le produit ou service doit remplir, avec des niveaux de performance associés, et hiérarchise ces fonctions pour permettre des arbitrages en cas de contraintes. Ce travail est réalisé conjointement par les futurs utilisateurs et les concepteurs. Enfin, l'analyse technique consiste à choisir ou concevoir des solutions techniques détaillées pour remplir les fonctions définies, en assurant leur niveau de performance. Cette étape est réalisée par des techniciens spécialisés. Chaque étape produit un document spécifique : un document d'étude détaillé du besoin, un document d'analyse fonctionnelle, et un document d'analyse technique [11].

3.2.1 Besoins fonctionnels

Selon le cahier de charge précédemment établi, nous déduisons les besoins fonctionnels suivant :

- Inscription ;
- Connexion ;
- Réinitialisation du mot de passe ;
- Gestion du compte :
  - Modification du compte ;
  - Suppression du compte.
- Gestion des profils :
  - Création d'un profil ;
  - Modification d'un profil ;
  - Suppression d'un profil.
- Scan d'ECGs ;
- Numérisation d'ECGs ;

- Consultation des ECGs numérisés, avec :
  - Suppression d'un ECG ;
  - Import d'un ECG ;
- Participation à l'enrichissement de la base de données partagée ;
- Acquisition des ECGs partagés ;
- Signalisation d'un dysfonctionnement, avec :
  - Signalisation d'un scan erroné ;
  - Signalisation d'une numérisation échouée ;
  - Signalisation d'une numérisation erronée ;

3.2.2 Besoins non fonctionnels

Nous cherchons à satisfaire au maximum les besoins non fonctionnels de notre application, qui sont :

- Performance : traitement efficace de données et temps de réponse réduit au maximum.
- Sécurité : assurance de la confidentialité et la sécurité des données des utilisateurs, notamment les informations médicales.
- Accessibilité : outils d'accessibilités aux personnes malvoyantes ou présentant d'autres handicaps.
- Ergonomie : via des interfaces simples et intuitives, ainsi que des messages d'erreurs ou succès concis.
- Multiplateforme : accessibilité et utilisation cohérente de l'application indépendamment de l'appareil (ordinateur, smartphone, tablette, etc) et du système d'exploitation.
- Transparence : possibilité d'importer ses données.

3.3 Modélisation

La modélisation est le processus de représentation abstraite des systèmes logiciels afin de comprendre, concevoir et communiquer leur structure et leur comportement. Elle permet de simplifier la complexité d'un système en fournissant des schémas visuels qui décrivent ses différents aspects.

3.3.1 UML

De l'Anglais "Unified Modeling Language", est un langage visuel formel dédié à la spécification, la construction et la documentation des artefacts d'un système logiciel [12].

L'UML est considéré comme étant une norme qui définit les diagrammes et les conventions à utiliser lors de la construction de modèles décrivant la structure (diagrammes structurels) et le comportement (diagrammes comportementaux) d'un logiciel. Nous pouvons citer parmi ses diagrammes :

Diagramme de cas d'utilisation

C'est un diagramme de comportement, c'est-à-dire qu'il modélise l'aspect dynamique d'un logiciel. Il sert à identifier les acteurs ainsi que leurs cas d'utilisation du logiciel dans le cadre d'un système donné. Il est basé sur des descriptions textuelles.

Diagramme de séquence

Il s'agit aussi un diagramme de comportement dans UML qui met en évidence les interactions entre différents objets d'un système au fil du temps. Chaque objet est représenté par une ligne de vie, et les messages échangés entre ces objets sont représentés par des flèches orientées. Les messages peuvent être synchrones (réponse attendue) ou asynchrones (pas de réponse immédiate).

Les diagrammes de séquence sont particulièrement utiles pour modéliser des scénarios dynamiques où l'ordre des interactions est crucial.

Diagramme de classe

Un diagramme structurel en UML qui décrit la structure statique d'un système. Il permet de modéliser les classes du système, leurs attributs, méthodes, ainsi que les relations (associations, généralisations, dépendances) entre elles.

Ce diagramme est essentiel lors de la phase de conception, car il permet de définir la structure interne d'un système logiciel en spécifiant comment les objets interagiront à travers leurs relations et comportements respectifs. Il fournit une vue d'ensemble du modèle de données utilisé par le système, facilitant ainsi la compréhension et la maintenance du code source par la suite.

La combinaison de ces trois diagrammes renforcent la conception et la compréhension globale du système et est essentielle afin :

- D'avoir une vue structurelle et comportementale complémentaire ;
- De définir les liens entre acteurs, objets et interactions ;
- De gérer la complexité du projet ;
- De vérifier la cohérence du système.

Afin d'exploiter les outils de modélisation UML, il est important d'identifier les acteurs du système ainsi que les scénarios d'utilisation de ces derniers.

3.3.2 Modèle relationnel

Le modèle relationnel est un concept fondamental dans la gestion des bases de données relationnelles. Il repose sur la théorie des ensembles et représente les données sous forme de tables, aussi appelées relations et présentent les aspects suivants [13] :

- Tables (Relations) ;
- Clés primaires ;
- Clés étrangères ;
- Contraintes d'intégrité.

L'obtention du modèle relationnel se fait à partir du diagramme de classe suivant les principales règles de conversion suivantes [14] :

- **Classes** : Une classe est convertie en une relation (table). Le nom de la relation peut être identique à celui de la classe.
- **Attributs** : Les attributs simples d'une classe sont convertis en colonnes de la relation correspondante. Les attributs multi-valués nécessitent la création d'une nouvelle relation, où chaque tuple inclut la clé primaire de la classe originale.
- **Attributs composés** : Les attributs composés peuvent être convertis en utilisant un type de données composite si disponible, sinon ils sont traités comme des attributs multi-valués.
- **Clés et Contraintes** : Les clés candidates et les contraintes d'intégrité définies dans la modélisation par l'UML doivent être converties en clés primaires et étrangères dans le modèle relationnel.
- **Associations** : Les associations entre classes sont converties en relations avec des clés étrangères pour représenter les liens entre les entités, plus précisément :
  - **Association un-à-plusieurs** : Une clé étrangère est déclarée, dans la relation correspondante à la classe ayant une multiplicité n dans l'association, avec une référence vers la classe ayant la multiplicité de 1 ;
  - **Association plusieurs-à-plusieurs** : Une nouvelle relation est crée contenant deux clés étrangères référençant les relations des classes associées ;
  - **Association un-à-un** : Les deux relations sont fusionnées, ceci peut engendrer une factorisation des attributs.

3.3.3 Identification des acteurs

Il s'agit d'une étape cruciale à la modélisation, nous identifions dans notre système les deux acteurs suivants :

1. Utilisateur : qui peut être un médecin souhaitant organiser les ECGs de ses patients ou le patient lui-même ;

2. Entité de confiance : qui est une entité permise d'accéder aux données partagées volontairement par les utilisateurs, il peut s'agir d'universités, centres de recherches, etc. Leur validation se fera par l'équipe de développement.

3.4 Méthode de conception

La méthode de conception de logiciel englobe un ensemble d'activités destinées à répondre à une demande d'informatisation d'un processus. Cette demande peut varier d'une simple question orale à un cahier des charges complet. Le processus permet de concevoir, d'écrire et de mettre au point un logiciel, jusqu'à sa livraison au client. En général, la création d'un logiciel suit trois grandes phases [15] :

- **Phase d'analyse (fonctionnelle) ou de conception** : Cette phase consiste à étudier simultanément les données et les traitements à effectuer. C'est ici que les techniques de modélisation sont appliquées. Le résultat inclut la description des bases de données à créer, les programmes à écrire et la manière dont tout cela sera intégré.
- **Phase de réalisation ou de programmation** :
  - Programmation
  - Gestion des versions
  - Factorisation
  - Tests unitaires
  - Optimisation du code
- **Phase de livraison** :
  - Intégration
  - Validation
  - Documentation du logiciel

Nous avons choisi la méthode Agile pour son approche flexible et adaptative qui permet de répondre rapidement aux changements et aux exigences évolutives du projet. Contrairement aux méthodologies traditionnelles, Agile favorise une livraison itérative et incrémentale, ce qui permet de produire des livrables fonctionnels à intervalles réguliers et d'apporter des améliorations continues basées sur les retours des utilisateurs.

De plus, en adoptant la méthode Agile, nous nous alignons sur les pratiques modernes largement reconnues et utilisées dans l'industrie, ce qui facilite une transition plus fluide vers le milieu professionnel.

Enfin, c'est une méthode adaptée à notre contexte de travail actuel qui consiste à régulièrement s'entretenir avec notre encadrant pour des rétrospectives et des échanges d'idées ou simplement la prise de décisions importantes dans le processus de conception ou réalisation.

3.4.1 Méthode AGILE

Le terme ”agile” est utilisé dans divers contextes avec des significations variées. Selon des sources telles que le dictionnaire Oxford ou Cambridge, l'adjectif ”agile” est souvent associé à la capacité de se déplacer rapidement, avec légèreté et facilité. Depuis le début du siècle en cours, l'intérêt pour les méthodes agiles a connu une augmentation significative, principalement dans l'industrie du logiciel. Cette évolution a été influencée par le Manifeste pour le développement logiciel Agile, émergé en 2001 grâce à l'initiative d'un collectif de développeurs de logiciels à la recherche d'une alternative aux processus de développement de logiciels axés sur la documentation et considérés comme lourds [16].

Agile n'est pas une méthodologie prescriptive, mais plutôt un ensemble de principes et de valeurs énoncés dans le Manifeste Agile (Baxter & Turner, 2021). Ce Manifeste Agile présente une approche novatrice du développement qui valorise fortement la collaboration, l'adaptabilité, la livraison continue de produits de valeur et la capacité à réagir rapidement aux changements. Il encourage l'auto-organisation des équipes, favorise une communication régulière avec les parties prenantes et encourage l'adoption de solutions évolutives plutôt que de se conformer à des processus rigides et prédictifs. Le Manifeste Agile est aujourd'hui une référence incontournable dans de nombreux domaines de développement, allant au-delà du seul secteur du logiciel.

Le développement agile est caractérisé par son approche itérative et incrémentale, ainsi que par sa flexibilité. Cette approche ouvre la voie à gérer efficacement l'incertitude inhérente aux projets de développement tout en maintenant un flux de travail efficace. Au fil du temps, le concept agile a évolué pour englober diverses méthodologies telles que Scrum, Kanban, Lean, ainsi que des variantes hybrides qui combinent différentes approches (Baxter & Turner, 2021). Bien que les différentes méthodes partagent l'idée de fournir en continu de la valeur et de s'adapter aux changements, elles se distinguent par le niveau d'encadrement et l'étendue des cycles de vie impliqués.

3.4.2 Principes de la méthode AGILE

Les 12 principes Agile selon ”Agile Alliance” sont [17] :

1. Notre plus haute priorité est de satisfaire le client par la livraison rapide et continue de logiciels à valeur ajoutée.
2. Accueillez favorablement les exigences changeantes, même tard dans le développement. Les processus Agile exploitent le changement pour l'avantage compétitif du client.
3. Livrez des logiciels fonctionnels fréquemment, de quelques semaines à quelques mois, avec une préférence pour les périodes les plus courtes.
4. Les gens de métier et les développeurs doivent travailler ensemble quotidiennement tout au long du projet.
5. Construisez les projets autour de personnes motivées. Donnez-leur l'environnement et le soutien dont ils ont besoin, et faites-leur confiance pour accomplir le travail.

6. La méthode la plus efficace et efficiente pour transmettre l'information à l'intérieur d'une équipe de développement est la conversation en face à face.
7. Le logiciel fonctionnel est la principale mesure de progression.
8. Les processus Agile promeuvent un développement durable. Les commanditaires, les développeurs et les utilisateurs devraient être capables de maintenir un rythme constant indéfiniment.
9. Une attention continue à l'excellence technique et à une bonne conception améliore l'agilité.
10. La simplicité – l'art de maximiser la quantité de travail non fait – est essentielle.
11. Les meilleures architectures, exigences et conceptions émergent d'équipes auto-organisées.
12. À intervalles réguliers, l'équipe réfléchit à la manière de devenir plus efficace, puis ajuste et modifie son comportement en conséquence.

Parmi les processus de développements suivant la méthode Agile, nous avons opté pour le Scrum en raison de ses nombreux avantages pour la gestion de projets complexes et évolutifs.

Scrum favorise une approche itérative et incrémentale héritée de la méthode Agile, permettant une livraison fréquente de produits fonctionnels. Cela garantit une adaptation rapide aux changements et une amélioration continue basée sur les retours des utilisateurs.

La flexibilité de Scrum, combinée à sa structure bien définie de rôles, événements et artefacts, en fait un choix idéal pour minimiser les risques, maximiser la productivité et assurer la qualité des livrables.

3.5 Processus de développement Scrum

Scrum est un cadre de travail (framework) permettant de gérer des projets complexes en livrant des produits de haute valeur de manière efficace et créative. Utilisé depuis les années 1990, Scrum n'est pas une méthode ou un processus en soi, mais plutôt un cadre dans lequel divers processus et techniques peuvent être appliqués. Il met en lumière l'efficacité de la gestion des produits et des techniques de travail, favorisant ainsi l'amélioration continue du produit, de l'équipe et de l'environnement de travail [18].

Le cadre Scrum se compose d'équipes Scrum, de rôles, d'événements, d'artefacts et de règles associées, chacun ayant un objectif précis essentiel au succès de Scrum. Les règles de Scrum, qui relient les rôles, les événements et les artefacts, sont décrites en détail dans le guide. Les tactiques d'utilisation de Scrum peuvent varier et ne font pas partie de ce guide.

3.5.1 Équipe Scrum

Une équipe Scrum comprend un Product Owner, une équipe de développement (Development Team) et un Scrum Master. Les équipes Scrum sont auto-organisées et pluridisciplinaires, ce qui signifie qu'elles choisissent la meilleure façon d'accomplir leur travail et

possèdent toutes les compétences nécessaires pour effectuer le travail sans dépendre d'autres personnes extérieures. Ce modèle optimise la flexibilité, la créativité et la productivité, permettant de livrer des produits de manière itérative et incrémentale. Elle est composée ainsi [18] :

- **Product Owner** : responsable de maximiser la valeur du produit issu du travail de l'équipe de développement. Il gère le Backlog Produit, en s'assurant que ses éléments sont clairs, ordonnés et compris par l'équipe. Le Product Owner est le seul à pouvoir modifier les priorités des éléments du Backlog Produit.
- **Équipe de développement** se compose de professionnels auto-organisés, responsables de créer des incréments "finis" à la fin de chaque Sprint. Ces équipes sont pluridisciplinaires et possèdent toutes les compétences nécessaires pour livrer un produit fonctionnel. La taille optimale de l'équipe de développement varie de trois à neuf membres pour maintenir l'efficacité et minimiser la complexité.
- **Scrum Master** est chargé de promouvoir et supporter Scrum, en aidant tout le monde à comprendre ses principes, pratiques, règles et valeurs. En tant que leader-serviteur, le Scrum Master aide l'équipe à maximiser sa valeur en adaptant les interactions internes et externes.

3.5.2 Étapes clés de Scrum

Les étapes clés de Scrum se présentent en quatre étapes comme suit :

Étape 01 : Le Product Backlog

Le Backlog Produit (Product backlog en anglais) est une liste ordonnée de tous les éléments nécessaires au produit, servant de source unique d'exigences pour les modifications à apporter. Le Product Owner est responsable de son contenu, de sa disponibilité et de son ordonnancement. Un Backlog Produit est toujours en évolution, car il s'adapte continuellement aux changements du produit et de son contexte d'utilisation. Il inclut toutes les fonctionnalités, fonctions, exigences, améliorations et corrections prévues pour les versions futures. Les éléments du Backlog Produit contiennent une description, un ordre, une estimation et une valeur, souvent accompagnés de descriptions des tests prouvant leur complétude. À mesure que le produit est utilisé et que des retours sont reçus, le Backlog Produit s'enrichit et s'ajuste en permanence aux nouveaux besoins de l'organisation, des conditions de marché ou des technologies [18].

Étape 02 : Le Sprint

Le cœur de Scrum est le Sprint, une période de temps limitée à un mois ou moins, au cours de laquelle un incrément de produit fonctionnel et potentiellement publiable est créé. Chaque Sprint a une durée cohérente tout au long de la phase de développement et commence immédiatement après la conclusion du Sprint précédent. Un Sprint inclut plusieurs événements essentiels : la planification du Sprint (Sprint Planning), les mêlées quotidiennes (Daily Scrums), les activités de développement, la revue de Sprint (Sprint Review) et la rétrospective de Sprint (Sprint Retrospective).

Pendant le Sprint :

- L'objectif du Sprint est fixe et ne peut pas être modifié, garantissant un cadre de travail stable pour l'équipe.
- Les objectifs de qualité sont maintenus, assurant que les standards de qualité ne sont jamais revus à la baisse.
- Le périmètre du travail peut être clarifié et renégocié entre le Product Owner et l'équipe de développement en fonction des nouvelles informations et apprentissages.

Chaque Sprint peut être considéré comme un projet en miniature, avec un horizon d'un mois. Comme tout projet, un Sprint a un objectif clair de ce qui doit être construit, une conception (design) et un plan flexible qui guidera la construction du produit, le travail lui-même et l'incrément de produit résultant. Les Sprints sont délibérément limités à un mois calendaire pour plusieurs raisons :

- Lorsque la durée d'un Sprint est trop longue, les définitions de ce qui est en cours de construction peuvent changer, augmentant la complexité et le risque.
- Les Sprints permettent une prédictibilité en assurant l'inspection et l'adaptation de la progression vers un objectif du Sprint (Sprint Goal) au moins tous les mois.
- Les Sprints limitent également le risque en assurant que le coût d'un échec potentiel ne dépasse jamais un mois calendaire.

Ainsi, les Sprints permettent une adaptation rapide aux changements tout en maintenant une cadence stable de livraison et d'amélioration continue du produit [18].

Étape 03 : Le Daily Scrum

La mêlée quotidienne (Daily Scrum) est une réunion de 15 minutes tenue chaque jour du Sprint pour l'équipe de développement. Elle vise à planifier le travail des prochaines 24 heures, optimiser la collaboration et inspecter l'avancement vers l'objectif du Sprint. La réunion se tient toujours au même endroit et à la même heure pour réduire la complexité. L'équipe discute des progrès réalisés, des tâches à venir et des obstacles potentiels. Le Scrum Master veille à ce que la réunion ait lieu et respecte la durée de 15 minutes, mais c'est l'équipe de développement qui en est responsable. La mêlée quotidienne améliore la communication, élimine les autres réunions, identifie les obstacles, encourage la prise de décision rapide et augmente le niveau de connaissance au sein de l'équipe [18].

Étape 04 : Le Sprint Review

La revue de Sprint (Sprint Review) est une réunion tenue à la fin de chaque Sprint pour inspecter l'incrément réalisé et adapter le Backlog Produit si nécessaire. Elle réunit l'équipe Scrum et les parties prenantes pour discuter des travaux effectués, des réussites, des problèmes rencontrés et des solutions apportées. Le Product Owner présente les éléments du Backlog Produit complétés et non complétés. L'équipe de développement démontre le travail achevé et répond aux questions. Cette réunion informelle, d'une durée maximale de quatre heures pour un Sprint d'un mois, vise à susciter des réactions et à favoriser la collaboration. Le résultat est un Backlog Produit révisé, définissant les éléments probables pour le prochain Sprint et tenant compte des nouvelles opportunités d'affaires [18].

3.6 Rédaction du Product Backlog initial

Avant de rédiger le Product Backlog, il faut d'abord identifier les User Stories :

3.6.1 Identification des User Stories

Une User Story (ou récit utilisateur) est la plus petite unité de travail dans un cadre Agile. Il s'agit d'un objectif final, et non d'une fonctionnalité, exprimé du point de vue de l'utilisateur du logiciel [18].

Les User Stories nous permettront de diviser nos itérations Scrum. Nous avons imaginé les User Stories suivantes :

Identifiant	User Story
US_ACCOUNT	En tant qu'utilisateur, je souhaite pouvoir créer et gérer un compte personnel
US_PROFILE	En tant qu'utilisateur, je souhaite pouvoir créer et personnaliser des profils au sein de mon compte
US_SCAN	En tant qu'utilisateur, je souhaite pouvoir scanner un ECG à partir d'une photo
US_DIGITAL	En tant qu'utilisateur, je souhaite pouvoir numériser mes ECGs et les consulter
US_REPORT	En tant qu'utilisateur, je souhaite pouvoir signaler les erreurs commises par l'application
US_SHARE	En tant qu'utilisateur, j'ai le choix de partager ou non mes ECGs dans le but de faire avancer les recherches scientifiques
US_DATASET	En tant qu'entité de confiance, j'aimerais avoir accès à une banque de données d'ECGs afin d'avancer dans mes recherches

Table 3.1: Identification des User Stories.

Ensuite, il faut identifier les scénarios d'utilisation de l'application de manière plus formelle, afin de détailler les fonctionnalités des User Stories déduites précédemment.

3.6.2 Identification des scénarios d'utilisation

Les scénarios d'utilisation est un outil qui interroge étape par étape, l'utilisation d'un service par ses utilisateurs.

L'analyse des besoins fonctionnels et l'identification des acteurs du système nous ont permis d'obtenir une liste de scénarios d'utilisation présentée comme suit :

Authentification

Se divise en deux scénarios, d'abord l'inscription :

Scénario	Inscription
Acteur	Utilisateur
Description	Création d'un nouveau compte dans l'application
Pré-condition	Aucune
Scénario principal	L'utilisateur saisit les données requises à la création d'un compte, c'est-à-dire : • Un nom d'utilisateur ; • Une adresse-mail ; • Une date de naissance ; • Son genre ; • Un mot de passe ainsi que la confirmation du mot de passe.
Scénario alternatif	Si le nom d'utilisateur ou l'adresse-mail est déjà prise, un message d'erreur est envoyé à l'utilisateur. Aussi, l'inscription de l'utilisateur est refusée s'il ne satisfait pas les conditions suivantes : • Fournir un nom d'utilisateur faisant entre 04 et 24 caractères, commençant par une lettre et contenant que des tirets, points et chiffres ; • Fournir une adresse e-mail valide ; • Avoir entre 12 et 150 ans ; • Fournir un mot de passe d'au moins 08 caractères ; • Faillir à la confirmation du mot de passe.
Post-condition	L'utilisateur possède un compte au sein de l'application et peut s'y connecter. Un profil (profil de base) a automatiquement été créé avec les informations fournies et a été attribué à son compte

Table 3.2: Scénario de "Inscription".

Ensuite la connexion :

Scénario	Connexion
Acteur	Utilisateur
Description	Connexion au compte
Pré-condition	Être inscrit
Scénario principal	L'utilisateur choisit entre saisir son nom d'utilisateur ou son adresse e-mail dans un premier temps, puis son mot de passe
Scénario alternatif	Un message d'erreur adéquat est affiché si : • Aucun compte ne correspond à l'identifiant fourni (que ce soit par adresse e-mail ou par nom d'utilisateur) ; • Mot de passe erroné ; • L'une des entrées est invalide (comme détaillée plus tôt).
Post-condition	L'utilisateur pourra désormais scanner et numériser des ECGs et gérer ses profils

Table 3.3: Scénario de "Connexion".

Réinitialisation du mot de passe

Représenté ainsi :

Scénario	Réinitialisation du mot de passe
Acteur	Utilisateur
Description	Obtention d'un nouveau mot de passe
Pré-condition	Être inscrit
Scénario principal	L'utilisateur entre son adresse e-mail et demande la réinitialisation de son mot de passe, un e-mail lui est envoyé contenant un lien vers un formulaire où il sera en mesure de soumettre un nouveau mot de passe
Scénario alternatif	Un message d'erreur est affiché si l'adresse e-mail fournie ne correspond à aucun compte
Post-condition	L'utilisateur possède un nouveau mot de passe

Table 3.4: Scénario de "Réinitialisation du mot de passe".

Gestion du compte

Décomposé en deux scénarios qui sont la modification et suppression du compte :

Scénario	Modification du compte
Acteur	Utilisateur
Description	Mise à jour des informations liées au compte de l'utilisateur
Pré-condition	Se connecter
Scénario principal	L'utilisateur édite les informations liées à son compte en respectant les mêmes conditions sur les entrées que celles de l'inscription
Scénario alternatif	Un message d'erreur est renvoyé si une condition n'est pas satisfaite
Post-condition	Le profil de base est automatiquement mis à jour suivant les modifications

Table 3.5: Scénario de "Modification du compte".

Scénario	Suppression du compte
Acteur	Utilisateur
Description	Suppression définitive du compte
Pré-condition	Se connecter
Scénario principal	L'utilisateur choisit de supprimer son compte en prenant soin de confirmer son choix
Post-condition	Les profils et ECGs liés au compte supprimé le seront aussi

Table 3.6: Scénario de "Suppression du compte".

Gestion des profils

Décomposé en quatre scénarios, d'abord la consultation de la liste des profils :

Scénario	Consultation de la liste des profils
Acteur	Utilisateur
Description	Affichage de la liste des profils
Pré-condition	Se connecter
Scénario principal	L'utilisateur voit l'ensemble des profils associés à son compte
Post-condition	L'utilisateur pourra créer, modifier ou supprimer un profil

Table 3.7: Scénario de "Consultation de la liste des profils".

La création d'un profil :

Scénario	Création d'un profil
Acteur	Utilisateur
Description	Ajout d'un nouveau profil
Pré-condition	Se connecter
Scénario principal	L'utilisateur saisit, pour le profil, les informations suivantes : • Un nom d'utilisateur respectant les mêmes conditions que celui d'un compte ; • Un pseudonyme (facultatif) ; • Une date de naissance ; • Le genre.
Scénario alternatif	Un message d'erreur adéquat est affiché si : • Le nom d'utilisateur du profil n'est pas unique au sein du compte ; • Le pseudonyme dépasse les 24 caractères ; • Le détenteur du profil est âgé de moins d'un mois ou plus de 150 ans.
Post-condition	L'utilisateur pourra désormais attribuer des ECGs à ce profil.

Table 3.8: Scénario de "Création d'un profil".

La modification des informations liées à un profil :

Scénario	Modification d'un profil
Acteur	Utilisateur
Description	Modification des informations liées à un profil existant
Pré-condition	Avoir créé un profil
Scénario principal	L'utilisateur saisit les nouvelles informations qu'il souhaite mettre à jour au sein d'un profil
Scénario alternatif	Un message d'erreur adéquat est affiché si les mêmes conditions sur les entrées que celles de la création d'un profil ne sont pas satisfaites
Post-condition	Les informations du profil sont mises à jour

Table 3.9: Scénario de "Modification d'un profil".

Et enfin la suppression d'un profil :

Scénario	Suppression d'un profil
Acteur	Utilisateur
Description	Suppression d'un profil existant
Pré-condition	Avoir créé un profil
Scénario principal	L'utilisateur supprime le profil en prenant soin de confirmer son choix
Post-condition	Les ECGs attribués à ce profil sont supprimés aussi

Table 3.10: Scénario de "Suppression d'un profil".

Scan d'un ECG

Se déroule comme suit :

Scénario	Scan d'un ECG
Acteur	Utilisateur
Description	Obtention d'un scan de l'ECG présent sur la photo envoyée
Pré-condition	Se connecter
Scénario principal	L'utilisateur envoie un fichier image contenant un ECG, un scan est ensuite renvoyé par l'application
Scénario alternatif	Un message d'erreur est envoyé indiquant qu'aucun ECG n'a été détecté dans la photo fournie ou si le fichier n'est pas de format image (JPG, JPEG ou PNG)
Scénario optionnel	L'utilisateur télécharge le scan renvoyé par l'application vers son appareil
Post-condition	L'utilisateur pourra numériser l'ECG scanné

Table 3.11: Scénario de "Scan d'un ECG".

Numérisation d'un ECG scanné

Se déroule de la manière suivante :

Scénario	Numérisation d'un ECG
Acteur	Utilisateur
Description	Numérisation d'un ECG en un ensemble de points
Pré-condition	Scanner un ECG
Scénario principal	L'utilisateur reçoit le scan et décide de numériser l'ECG contenu dans le scan après avoir rempli un formulaire contenant : • Le profil auquel il veut attribuer l'ECG ; • La date de l'ECG ; • Une note écrite (facultative).
Scénario alternatif	Un message d'erreur est envoyé si la numérisation a échoué ou si le détenteur du profil propriétaire de l'ECG avait moins d'un mois ou plus de 150 ans durant l'examen médical
Post-condition	L'utilisateur pourra consulter les ECGs numérisés

Table 3.12: Scénario de "Numérisation d'un ECG".

Gestion des ECGs

Se divise en deux scénarios, d'abord la consultation des ECGs :

Scénario	Consultation des ECGs
Acteur	Utilisateur
Description	Affichage de la liste de tous les ECGs liés au compte
Pré-condition	Se connecter
Scénario principal	L'utilisateur voit tous les ECGs numérisés avec succès
Scénario alternatif	L'utilisateur filtre les ECGs par profils.
Post-condition	L'utilisateur pourra supprimer un ECG ou avoir une vue détaillée d'un ECG

Table 3.13: Scénario de "Consultation des ECGs".

Ensuite la suppression :

Scénario	Suppression d'un ECG
Acteur	Utilisateur
Description	Suppression définitive d'un ECG
Pré-condition	Numériser un ECG
Scénario principal	L'utilisateur sélectionne un ECG et choisit de le supprimer.
Post-condition	L'ECG n'appartiendra plus à la base de données publique et n'apparaitra plus dans la liste des ECGs

Table 3.14: Scénario de "Suppression d'un ECG".

Import d'un ECG

Représenté selon le tableau suivant :

Scénario	Import d'un ECG
Acteur	Utilisateur
Description	Téléchargement d'un ECG sur l'appareil
Pré-condition	Numériser un ECG
Scénario principal	L'utilisateur sélectionne l'ECG qu'il souhaite importer parmi la liste de ses ECGs
Post-condition	Téléchargement d'un fichier archive contenant le scan et la version numérique de l'ECG sur l'appareil

Table 3.15: Scénario de "Import d'un ECG".

Participation à l'enrichissement de la base de données

Ou "partager ses données" pour participer aux programmes de recherches :

Scénario	Participation à l'enrichissement de la base de données
Acteur	Utilisateur
Description	Rendre ses données ECGs accéssibles à des entités permises
Pré-condition	Se connecter
Scénario principal	L'utilisateur active ou désactive cette option dans les paramètres de son compte
Post-condition	Tous les ECGs de l'utilisateur sont accessibles aux entités de confiance ainsi que son âge et son genre durant chacun de ces ECGs

Table 3.16: Scénario de "Participation à l'enrichissement de la base de données".

Acquisition des ECGs partagés

Ou "exploitation de la base de données publique" pour les programmes de recherche :

Scénario	Acquisition des ECGs partagés
Acteur	Entité de confiance
Description	Acquérir l'ensemble des ECGs dont les propriétaires ont accepté leur partage
Pré-condition	Posséder une adresse e-mail validée par l'application
Scénario principal	L'entité de confiance entre son adresse e-mail et télécharge un fichier archive contenant tous les scans accompagnés des numérisations des ECGs issus de la base de données publique
Scénario alternatif	L'adresse e-mail fournie n'est pas une adresse e-mail valide ou ne figure pas dans la liste des adresses e-mail des entités permises d'accès, un message d'erreur adéquat est ensuite affiché

Table 3.17: Scénario de "Acquisition des ECGs partagés".

Signalisation d'un dysfonctionnement

Peut se dérouler sous différentes manières, d'abord lorsqu'un scan ne donne pas les résultats attendus :

Scénario	Signalisation d'un scan erroné
Acteur	Utilisateur
Description	Signalisation d'un scan erroné
Pré-condition	Scanner un ECG
Scénario principal	L'utilisateur reçoit le scan et décide de le signaler s'il estime qu'il a été mal effectué
Post-condition	L'image d'entrée de l'ECG qui n'a pas pu être scannée est sauvegardée dans la partie Serveur

Table 3.18: Scénario de "Signalisation d'un scan erroné".

Ou lorsqu'une numérisation échoue :

Scénario	Signalisation d'une numérisation échouée
Acteur	Utilisateur
Description	L'utilisateur signale l'échec de la numérisation s'il estime que le scan de l'ECG est de bonne qualité
Pré-condition	Numériser un ECG
Scénario principal	L'utilisateur reçoit le message d'erreur de la numérisation et décide de signaler cet échec
Post-condition	Le scan de l'ECG qui n'a pas pu être numérisé est sauvegardé dans la partie Serveur

Table 3.19: Scénario de "Signalisation d'une numérisation échouée".

Ou encore lorsque l'utilisateur estime que la numérisation présente des imprécisions :

Scénario	Signalisation d'une numérisation erronée
Acteur	Utilisateur
Description	L'utilisateur signale une numérisation erronée s'il estime que les dérivations obtenues ne sont pas fidèles à l'ECG original
Pré-condition	Numériser un ECG
Scénario principal	L'utilisateur signale la numérisation comme étant erronée et sera prévenu de ce signalement à chaque consultation de cet ECG en question
Post-condition	L'ECG signalé est libellé comme étant erroné et l'utilisateur en est prévenu à chaque consultation de ce dernier

Table 3.20: Scénario de "Signalisation d'une numérisation erronée".

3.6.3 Product Backlog initial

Ainsi, nous obtenons un Product Backlog défini comme suit :

Sprint	User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Priorité
Sprint 01	US_ACCOUNT	Utilisateur	Authentification	Inscription	Maximale
				Connexion	
				Modification du compte	
				Suppression du compte	
				Réinitialisation du mot de passe	Haute
Sprint 02	US_PROFILE	Utilisateur	Gestion des profils	Consultation de la liste des profils	Haute
				Création d'un profil	
				Modification d'un profil	
				Suppression d'un profil	
Sprint 03	US_SCAN	Utilisateur	Scan d'un ECG	Scan d'un ECG à partir d'une photo	Maximale
				Téléchargement du scan	
Sprint 04	US_DIGITAL	Utilisateur	Numérisation d'un ECG	Gestion et numérisation des ECGs	Maximale
				Consultation des ECGs	
				Suppression d'un ECG	
				Import d'un ECG	Haute
Sprint 05	US_REPORT	Utilisateur	Signalisation d'un dysfonctionnement	Signalisation d'un scan erroné	Modérée
				Signalisation d'une numérisation échouée	
				Signalisation d'une numérisation erronée	
Sprint 06	US_SHARE	Utilisateur	Partage des données d'ECGs	Partage des données d'ECGs	Faible
	US_DATASET	Entité de confiance	Acquisition des données d'ECGs	Acquisition des données d'ECGs	

Table 3.21: Product Backlog initial.

Réalisation

3.7 Mise en place de l'environnement de développement

Pour la réalisation de l'application, nous avons opté pour des technologies adaptées au patron Modèle-Vue-Contrôleur.

Le patron Modèle-Vue-Contrôleur, ou MVC, décompose une application en trois sous parties [19] :

1. **Modèle** : qui regroupe la logique métier (business logic) ainsi que l'accès aux données. Il peut s'agir d'un ensemble de fonctions (Modèle procédural) ou de classes (Modèle orienté objet) ;

2. **Vue** : qui s'occupe des interactions avec l'utilisateur : présentation, saisie et validation des données ;

3. **Contrôleur** : qui gère la dynamique de l'application. Elle fait le lien entre les deux autres parties.

Le patron MVC a été choisi pour sa capacité à maintenir une cohérence entre la modélisation du système logiciel et son implémentation, tout en facilitant la réutilisation du code et la gestion des interactions utilisateur-système. Il garantit une architecture solide, répondant aux exigences fonctionnelles identifiées lors de la phase de conception. De plus, ce patron s'intègre parfaitement à l'architecture Client-Serveur propre aux applications web, assurant ainsi une séparation claire des responsabilités entre le Frontend (processus côté Client), qui sont satisfaites par les composants représentatifs des vues, le Backend (processus côté Serveur), qui sont satisfaites par les modules représentatifs des contrôleurs, et la base de données, qui sont satisfaites par les tables représentatives des modèles.

3.7.1 Côté Client (Frontend)

La principale technologie Web utilisée dans le Frontend est React (également connu sous le nom de React.js ou ReactJS), qui est une bibliothèque JavaScript frontale à code source ouvert permettant de créer des interfaces utilisateur ou des composants d'interface utilisateur. Elle est maintenue par Facebook et une communauté de développeurs individuels et d'entreprises.

En complément, nous avons utilisé Shadcn, une bibliothèque d'interface utilisateur qui permet de construire des composants robustes et accessibles dans les projets basés sur React. Elle propose des composants modifiables et des styles personnalisables tout en respectant les bonnes pratiques en matière d'accessibilité. Shadcn offre également une intégration fluide

avec Tailwind CSS, facilitant la gestion des styles au sein des applications React. Cela nous a permis de concevoir rapidement des interfaces utilisateur réactives et conviviales tout en réduisant le temps de développement.

3.7.2 Côté Serveur (Backend)

Pour la gestion du côté Serveur, nous avons opté pour Express.js, un framework Web minimaliste pour Node.js. Express.js facilite la création de serveurs web légers et performants en fournissant une architecture simplifiée et flexible. Grâce à ses middlewares et à sa gestion des routes, il nous permet de développer rapidement des API RESTful robustes et scalables. De plus, son intégration transparente avec d'autres modules et bibliothèques Node.js a grandement simplifié la gestion des requêtes HTTP, de l'authentification, et des interactions avec la base de données.

3.7.3 Base de données

Pour la persistance des données, nous avons utilisé SQLite, une base de données relationnelle légère et embarquée. Elle stocke les données dans un fichier unique, ce qui la rend idéale pour les petites applications ou celles qui n'exigent pas une infrastructure de base de données complexe.

Nous avons aussi intégré Prisma, un ORM (Object-Relational Mapping) moderne et performant. Il facilite la gestion des données dans la base SQLite en offrant une abstraction de haut niveau qui permet d'interagir avec la base de données en utilisant des modèles d'objets. Prisma nous a permis d'automatiser les migrations de schéma, de générer des requêtes optimisées, et de maintenir une synchronisation fluide entre les données et les modèles de l'application. Grâce à son intégration avec TypeScript, Prisma améliore également la sécurité et la productivité en offrant un typage strict dans les opérations de base de données.

3.7.4 Communication Client-Serveur

Pour la liaison Frontend-Backend, nous avons employé une API REST (Representational State Transfer), qui est une interface de programmation d'applications (API) qui respecte les principes architecturaux définis par Roy Fielding en 2000 [20]. Les API REST sont flexibles et légères, ce qui les rend idéales pour intégrer des applications et connecter des composants dans des Client-Serveur.

Les API REST sont basées sur six principes de conception :

- **Interface uniforme** : Toutes les demandes pour une même ressource doivent avoir la même apparence, avec un seul identifiant de ressource uniforme (URI).
- **Découplage client-serveur** : Les applications client et serveur doivent être indépendantes, ne communiquant que via l'URI de la ressource demandée.
- **Apatridie** : Chaque requête doit contenir toutes les informations nécessaires à son traitement, sans nécessiter de session côté serveur.

- **Mise en cache** : Les ressources doivent pouvoir être mises en cache côté client ou serveur, avec des indications de cache dans les réponses du serveur.
- **Architecture en couches** : Les appels et réponses peuvent passer par plusieurs intermédiaires sans que le client ou le serveur ne sachent s'ils communiquent directement.

Les API REST utilisent les méthodes HTTP pour exécuter des opérations CRUD (Create, Read, Update, Delete) sur des ressources. Les données sont souvent fournies dans des formats comme JSON, ce qui les rend lisibles et indépendantes du langage de programmation. Enfin, la sécurisation des API REST repose sur des bonnes pratiques comme l'utilisation de HTTPS, d'algorithmes de hachage et de protocoles d'autorisation.

Afin de permettre la communication Client-Serveur, il a fallu sauvegarder le nom de domaine d'où proviendraient les requêtes du côté Client dans une variable d'environnement `CLIENT`. Ensuite, grâce au package `cors`, la politique Cross-Origin Resource Sharing (CORS) a été implémentée afin de permettre le traitement de toutes les requêtes en provenance du Client.

3.7.5 Outils de test

Pour le Frontend, nous avons utilisé React Developer Tools (React DevTools), qui est une extension de navigateur qui permet de déboguer et d'inspecter les applications React directement dans les outils de développement des navigateurs. Elle fournit une interface pour explorer l'arborescence des composants React, voir l'état et les propriétés (props) de chaque composant, ainsi que l'ensemble des hooks utilisés dans l'application.

Quant à notre REST API, nous avons employé Postman API, qui est un outil qui sert à tester les APIs. Il permet aux développeurs de créer, envoyer, et documenter des requêtes API de manière simple et interactive.

3.8 Réalisation des Landing Pages

Une Landing Page (ou page d'atterrissage en français) est une page de destination sur laquelle arrive un internaute après avoir cliqué sur un lien ou entré une URL [21].

La Landing Page s'impose comme un levier essentiel pour transformer des visiteurs, qu'ils soient familiers ou non avec l'application, en des utilisateurs ; Son objectif principal est donc de maximiser le taux de conversion.

La première Landing Page est la page d'accueil, qui présente brièvement les principales fonctionnalités de l'application et son but :

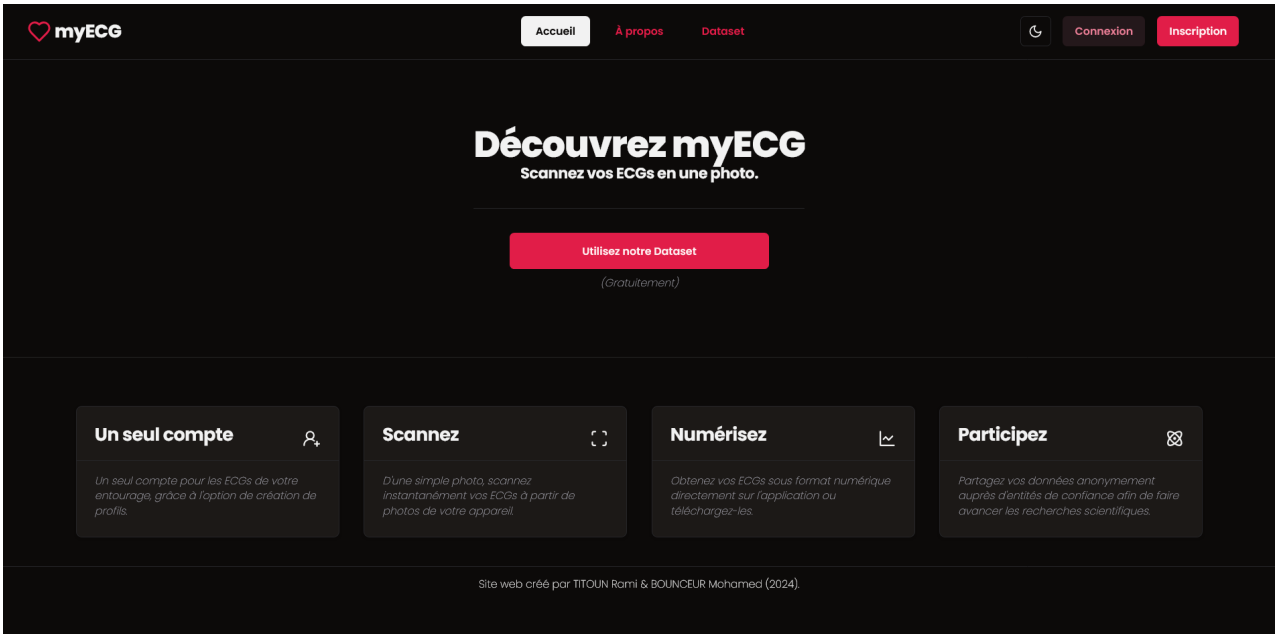


Figure 3.1: Landing Page principale.

Une autre page informative décrit le contexte de création de l'application et présente l'équipe qui a réalisé le projet :

Figure 3.2: Page informative.

La Landing Page "Dataset" est spécifique aux entités de confiance souhaitant exploiter la

base de données de l'application, elle présente aussi une adresse e-mail de contact des créateurs de l'application :

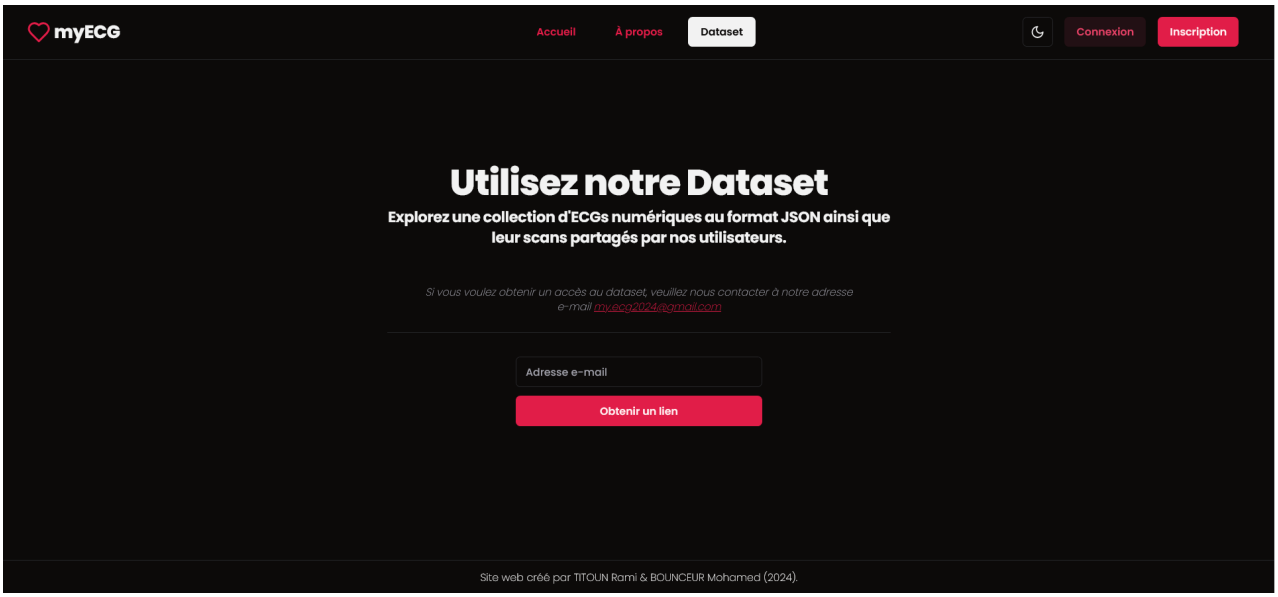


Figure 3.3: Landing Page des entités de confiance.

Nous allons explorer dans la prochaine section les processus de conception et réalisation de chaque fonctionnalité et l'interface graphique associée.

3.9 Itérations Scrum

Cette section présente l'implémentation des fonctionnalités de l'application par la méthode Scrum suivant le Product Backlog, en détaillant l'exécution des Sprints.

3.9.1 Sprint 01

Le Sprint initial traitera de l'implémentation des fonctionnalités relatives à l'authentification de l'utilisateur, sa gestion du compte et sa récupération du mot de passe :

Product Backlog

User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Durée estimée	Date de début	Date de fin
US_ACCOUNT	Utilisateur	Authentification	Inscription	01 semaine	01/03/2024	08/03/2024
			Connexion	02 semaines	09/03/2024	27/03/2024
			Réinitialisation du mot de passe	01 semaine	28/03/2024	04/04/2024
			Modification du compte	03 jours	04/04/2024	04/07/2024
			Suppression du compte	02 jours	07/04/2024	09/04/2024

Table 3.22: Product Backlog du Sprint 01.

Diagramme de cas d'utilisation

Composé de six cas d'utilisation, dont deux sont une spécialisation d'un même cas d'utilisation :

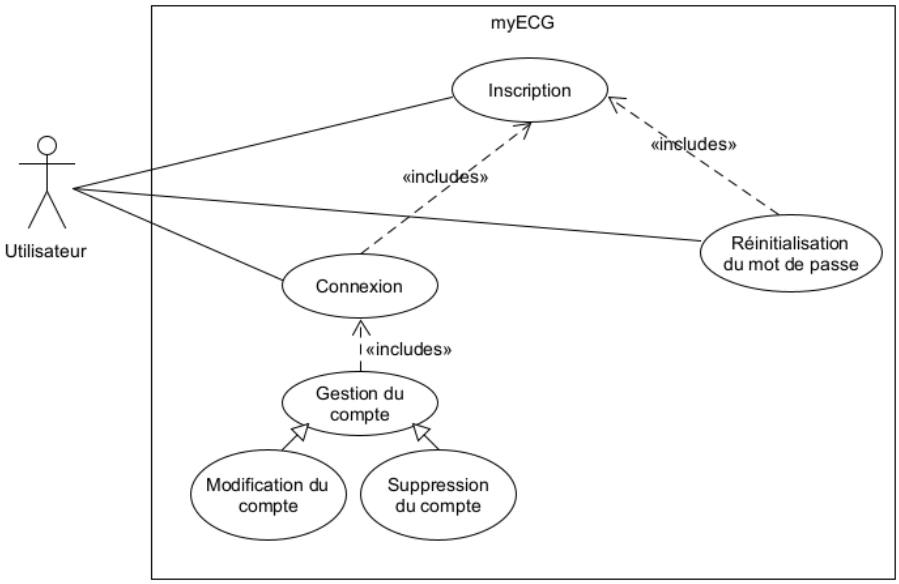


Figure 3.4: Diagramme de cas d'utilisation du Sprint 01.

Diagrammes de séquence

Composé de cinq diagrammes :

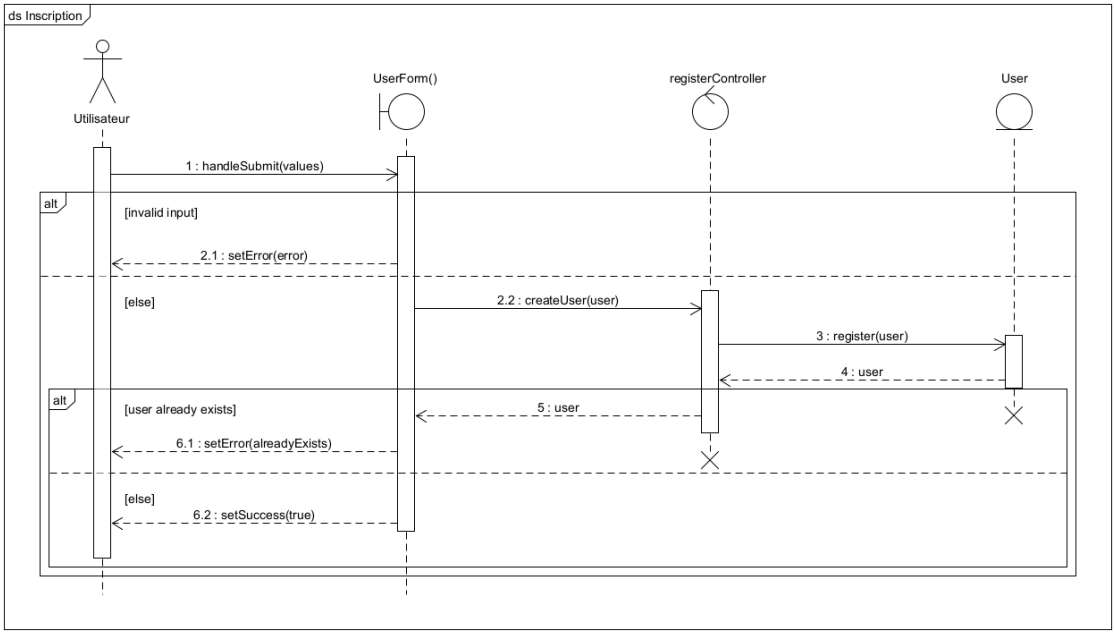


Figure 3.5: Diagramme de séquence du cas d'utilisation "Inscription".

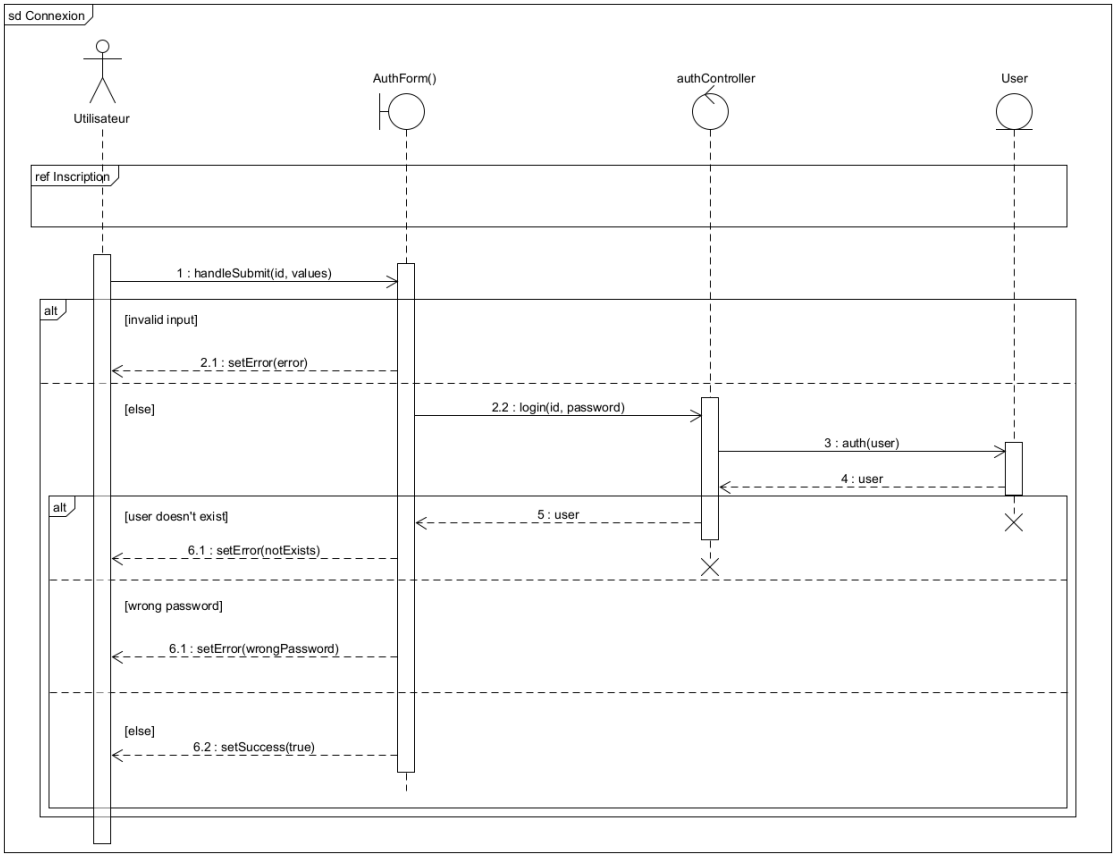


Figure 3.6: Diagramme de séquence du cas d'utilisation "Connexion".

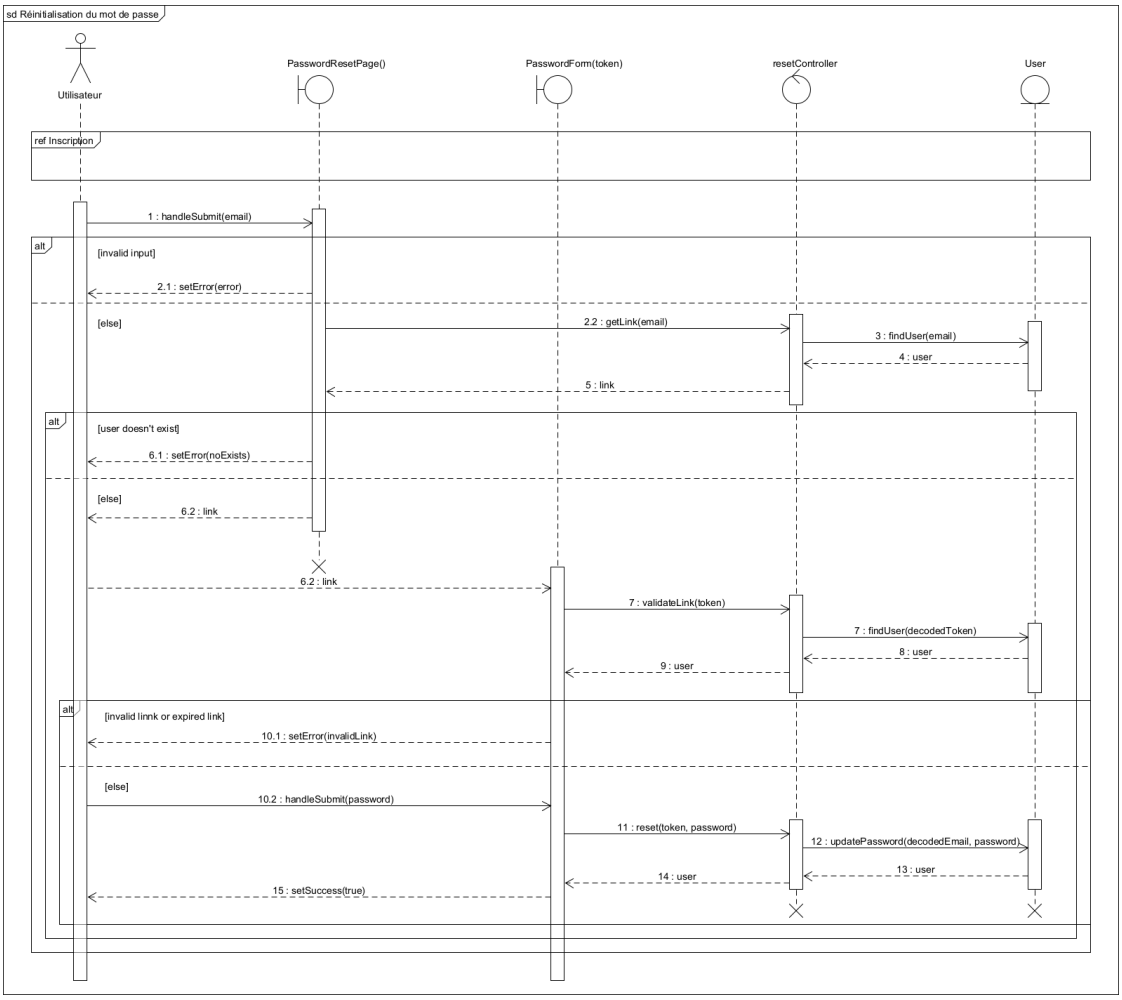


Figure 3.7: Diagramme de séquence du cas d'utilisation "Réinitialisation du mot de passe".

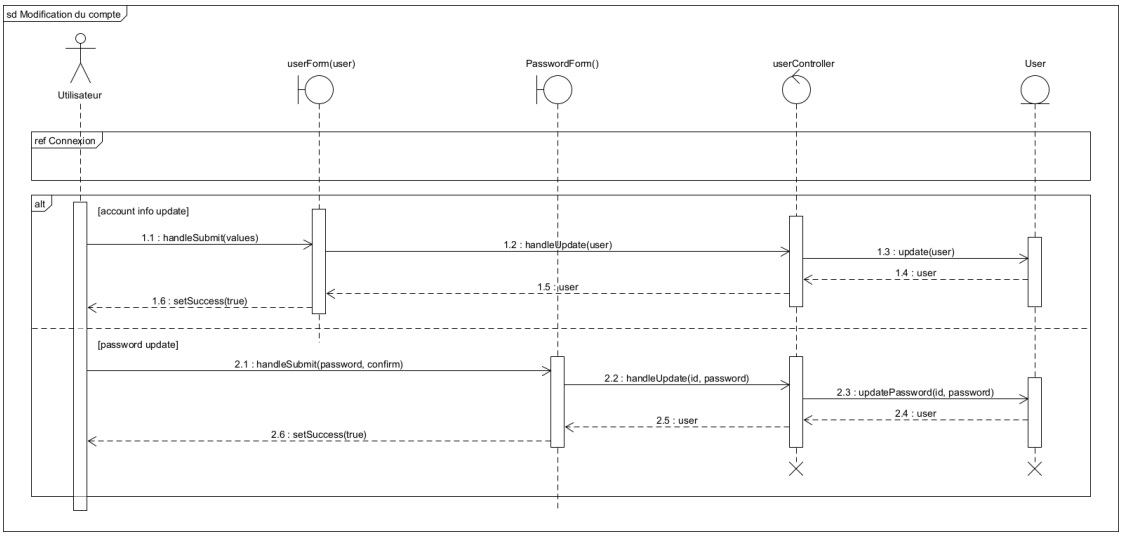


Figure 3.8: Diagramme de séquence du cas d'utilisation "Modification du compte".

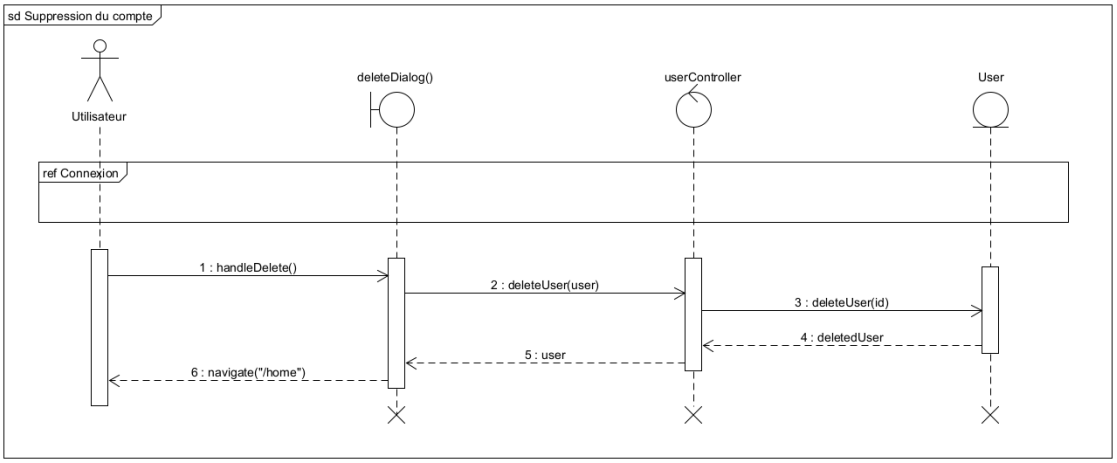


Figure 3.9: Diagramme de séquence du cas d'utilisation "Suppression du compte".

Diagramme de classe

Composé d'une seule classe, "User" :

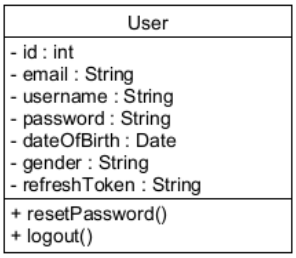


Figure 3.10: Diagramme de classe du Sprint 01.

Modèle relationnel

Composé, pour l'instant, d'une unique table "User" :

User(id, email, username, password, dateOfBirth, gender, refreshToken)

Contraintes :

- L'attribut `id` est la clé primaire de la table "User" ;
- Les attributs `username` et `email` sont uniques dans la table "User".

Implémentation

L'inscription se fait simplement en insérant un nouvel utilisateur dans la base de données en s'assurant que les champs du nom d'utilisateur et de l'adresse e-mail soient uniques.

La validation des entrées se fait grâce au package `zod`, qui est un module qui offre un ensemble de fonctions à combiner pour vérifier le respect des contraintes sur les entrées d'un formulaire. Il est plus flexible que les expressions régulières car il est intégré au formulaire HTML `form` et permet d'afficher des messages d'erreur personnalisés suivant chaque entrée du formulaire.

Le mot de passe fourni est traité avec les fonctions fournies par le package `bcrypt` : il est haché après génération d'un Salt sur 10 rondes avant d'être stocké dans la base de données.

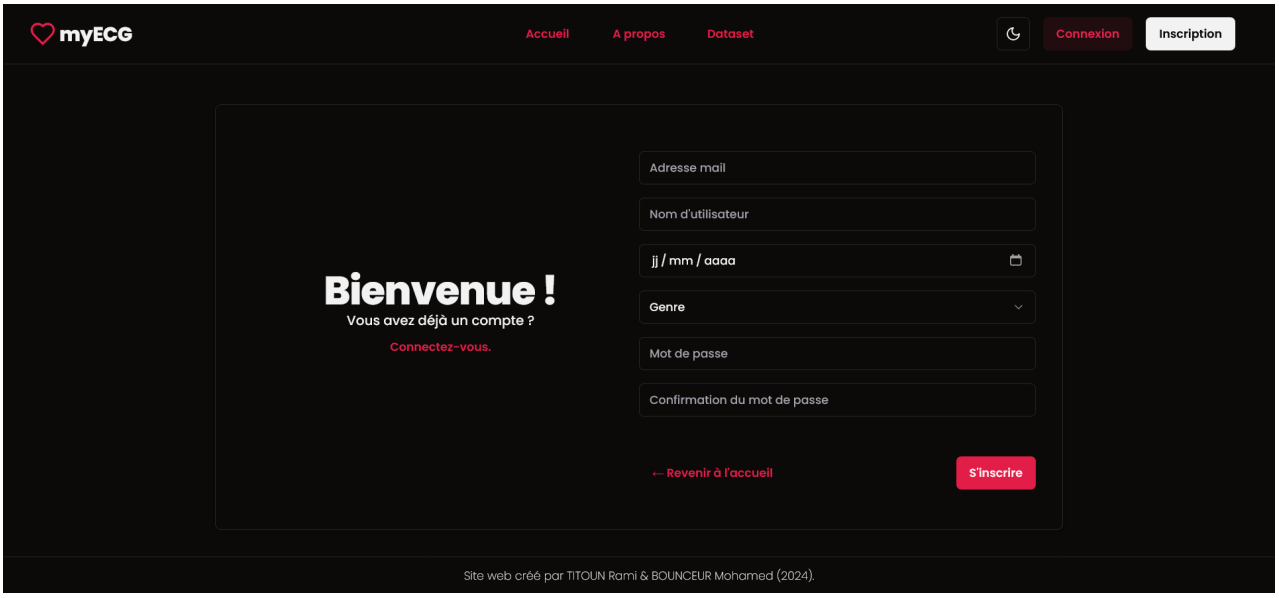


Figure 3.11: Interface d'inscription.

Après inscription, l'utilisateur est invité à se rendre à la page de connexion grâce à une notification indiquant que son inscription a été effectuée avec succès. La connexion ne requiert qu'un identifiant (qui est soit un nom d'utilisateur ou une adresse e-mail) ainsi qu'un mot de passe.

Il est possible de rechercher un utilisateur précis par son nom d'utilisateur ou par adresse e-mail car tous les noms d'utilisateur et adresses e-mail sont uniques, et un nom d'utilisateur valide ne peut être une adresse e-mail valide et vice-versa, à cause des contraintes sur les entrées du formulaire d'inscription. Par exemple, un nom d'utilisateur ne peut contenir le caractère de l'arobase, tandis que l'adresse e-mail doit contenir ce symbole exactement une fois.

Après que l'utilisateur ait entré ses coordonnées, un utilisateur ayant un nom d'utilisateur ou une adresse e-mail correspondante à l'identifiant fourni est recherché dans la base de données. Si aucun utilisateur n'est retrouvé, un message d'erreur est retourné indiquant que le compte n'existe pas. Sinon, le mot de passe fourni est haché sur 10 rondes et est comparé au mot de passe de l'utilisateur trouvé.

Deux clés secrètes ont été déclarées dans les variables d'environnement qui sont :

- La clé `ACCESS_TOKEN_SECRET` ;
- La clé `REFRESH_TOKEN_SECRET`.

Il s'agit de chaînes de caractères hexadécimaux aléatoirement générées toutes deux d'une longueur de 128 caractères. Lors de l'authentification d'un utilisateur, deux tokens JWT [22] (JSON Web Token, RFC 7519) sont générés à partir de son `id` avec les deux clés secrètes qui sont, respectivement, un `accessToken` et un `refreshToken`. Il servent de preuve d'identité

numérique et permettent d'autoriser l'accès à des ressources protégées.

Le token d'accès, stocké dans la mémoire vive du côté Client, possède une durée d'expiration de 30 minutes tandis que celui de rafraîchissement, stocké dans la base de données, une durée de 10 jours.

Pour chaque requête vers une ressource protégée, l'utilisateur enverra automatiquement son token d'accès et devra passer une phase de validation de ce token via un middleware `verifyJWT` implémenté sur le côté Serveur. Si le token d'accès est expiré, un nouveau token d'accès est généré après validation du token de rafraîchissement.

Cette méthode présente trois avantages qui sont l'invulnérabilité aux attaques par vol de Cookies (car les tokens d'accès sont stockés dans la mémoire vive plutôt que dans le Local Storage ou Session Storage), le renouvellement des tokens et leur rotation régulière ainsi que le stockage sécurisé du token de rafraîchissement dans la base de données et le fait que son utilisation se limite seulement au côté Serveur.

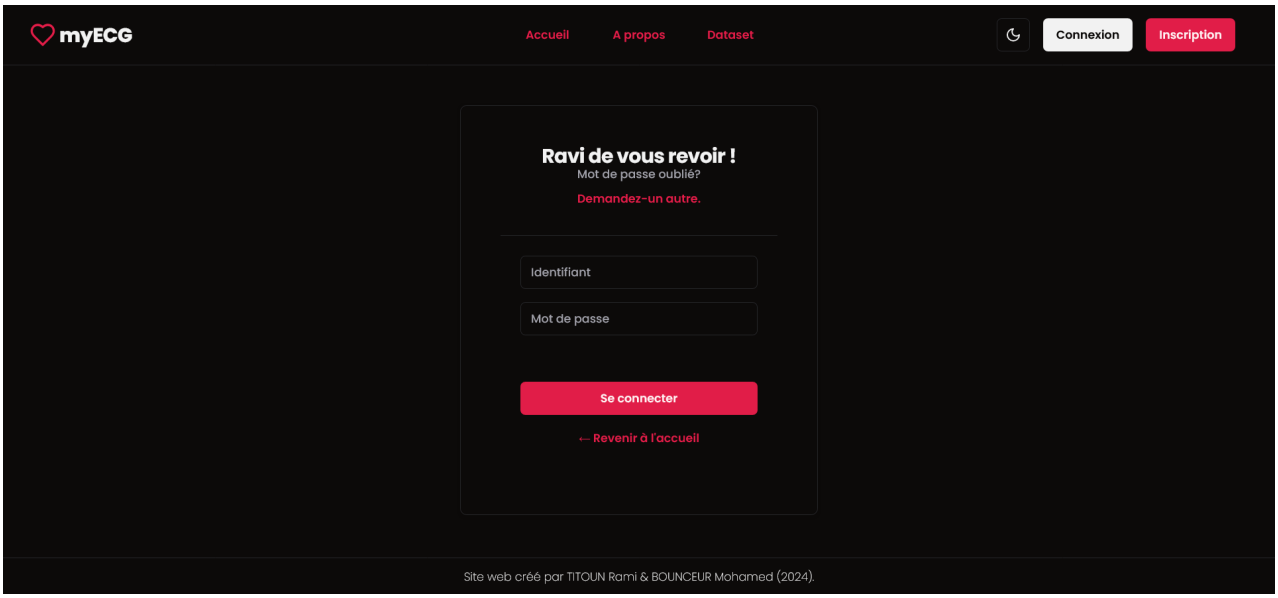


Figure 3.12: Interface de connexion.

En cas d'oubli du mot de passe, l'utilisateur peut réinitialiser ce dernier en fournissant son adresse e-mail :

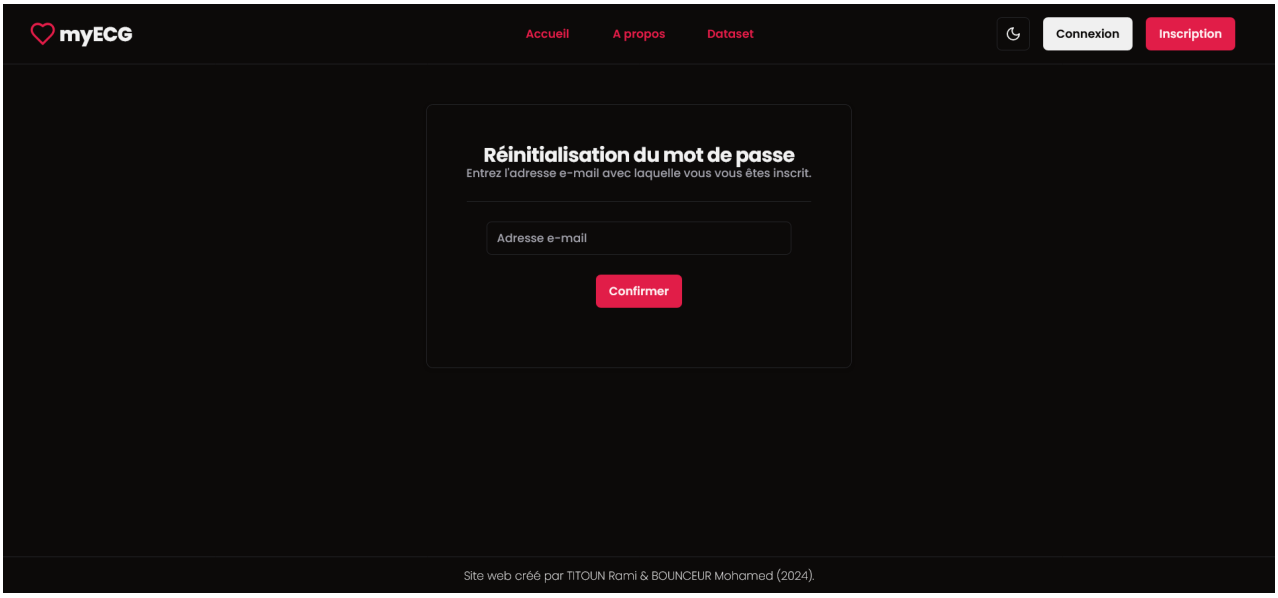


Figure 3.13: Interface de demande de réinitialisation du mot de passe.

Après avoir vérifié que son compte existe, l'utilisateur est notifié qu'un mail a été envoyé de la part de `my.ecg2024@gmail.com` vers son adresse e-mail. Cette opération est rendue possible grâce au package `nodemailer`.

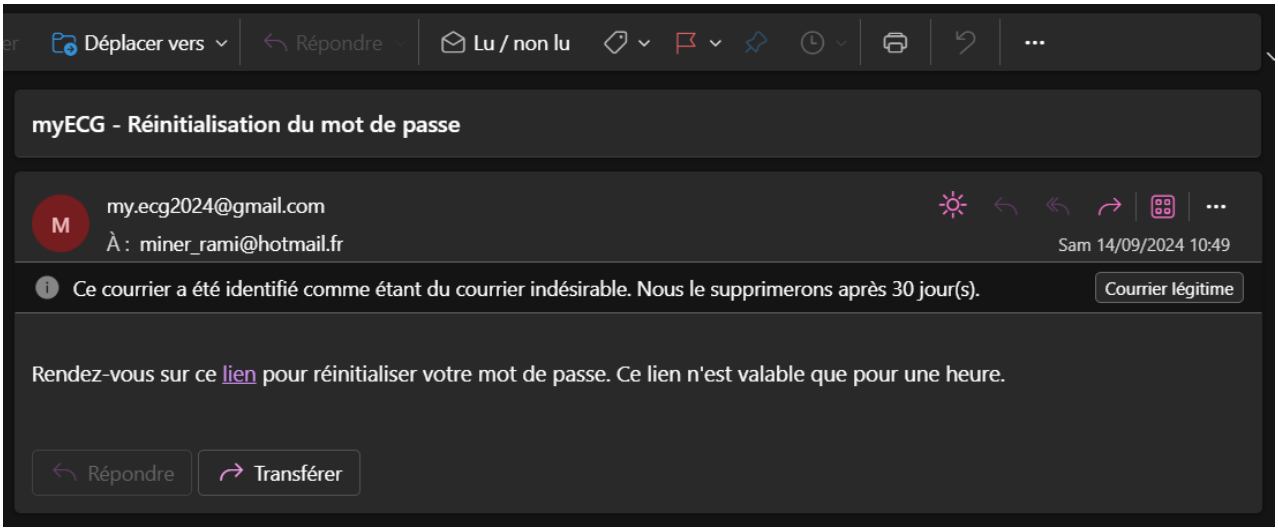


Figure 3.14: Mail de récupération de mot de passe.

Ce mail contient un lien qui a été généré à partir d'un token JWT contenant l'adresse e-mail en question crypté par une clé `EMAIL_TOKEN_SECRET` en caractères hexadécimaux de taille de 128 caractères ayant pour durée d'expiration 1 heure.

Lorsque l'utilisateur se rend vers ce lien, qui est une route dynamique (`reset/:token`) implémentée avec la librairie `react-router-dom`, il envoie une requête vers le Backend qui vérifie la validité de son token en retrouvant un compte possédant l'adresse e-mail équivalente au token décrypté. Si ce n'est pas le cas, un message d'erreur est affiché :

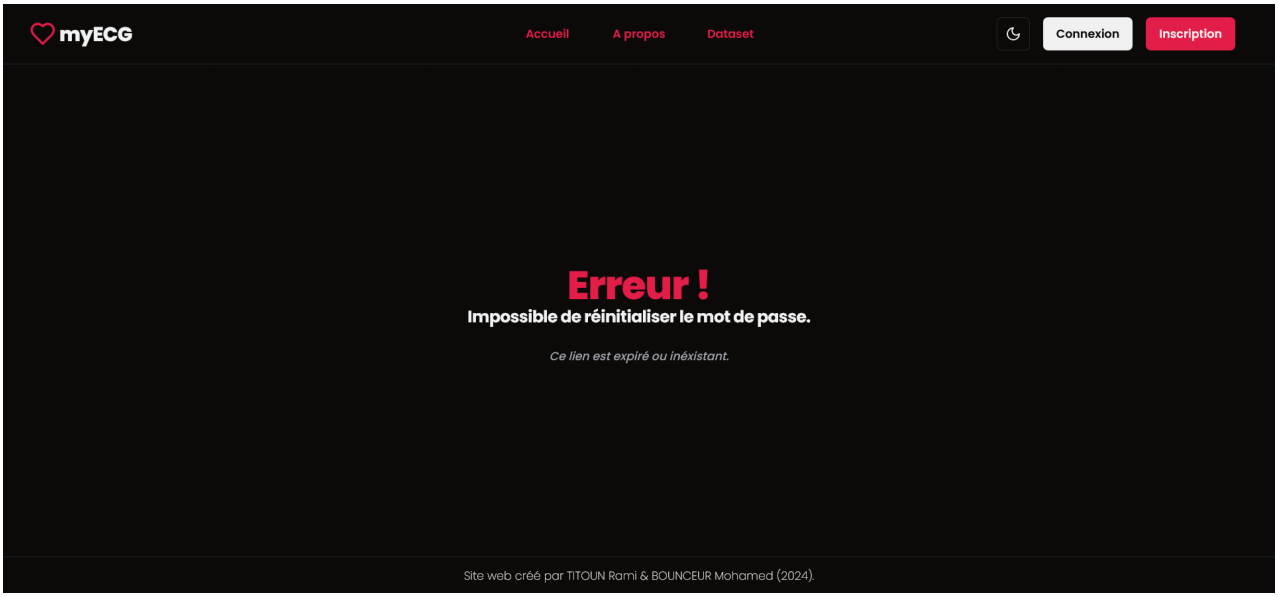


Figure 3.15: Erreur de validité du token de réinitialisation du mot de passe.

Sinon, il retrouve un court formulaire demandant un nouveau mot de passe ainsi que la confirmation de ce dernier :

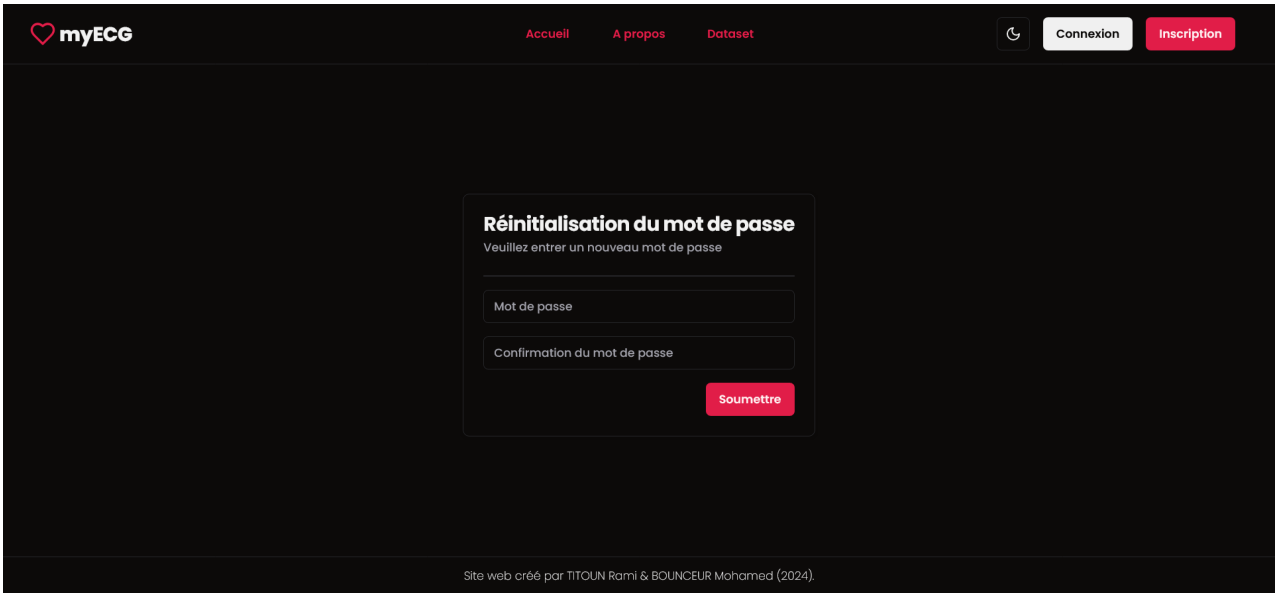


Figure 3.16: Interface de réinitialisation du mot de passe.

Le mot de passe sera mis à jour dans la base de données après avoir été haché. Quant à la gestion du compte, elle se fait via la page des paramètres :

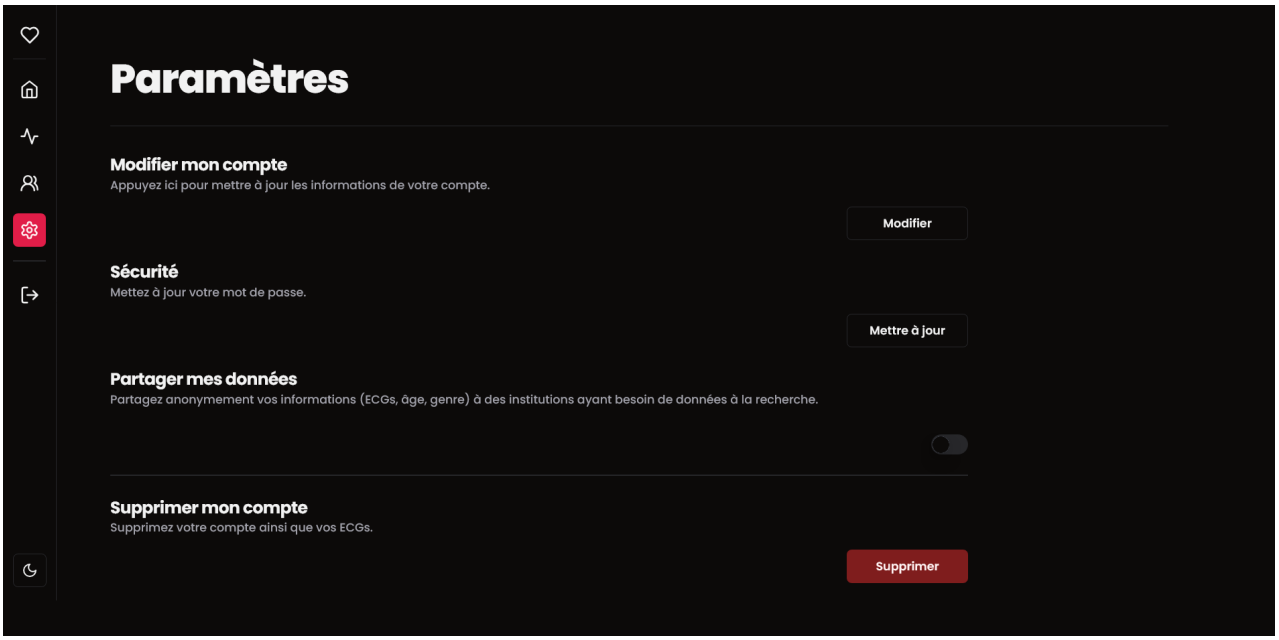


Figure 3.17: Interface des paramètres du compte.

Où l'on y retrouve l'interface de modification du compte :

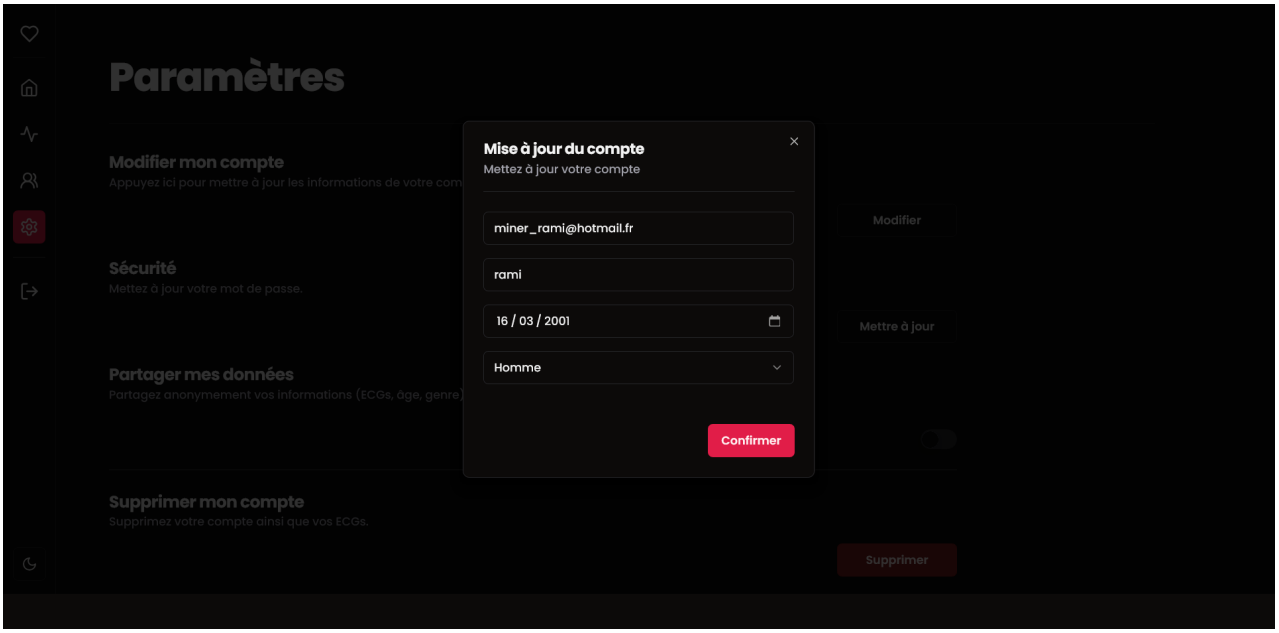


Figure 3.18: Interface des modification du compte.

Ainsi que celle de la suppression du compte, après confirmation :

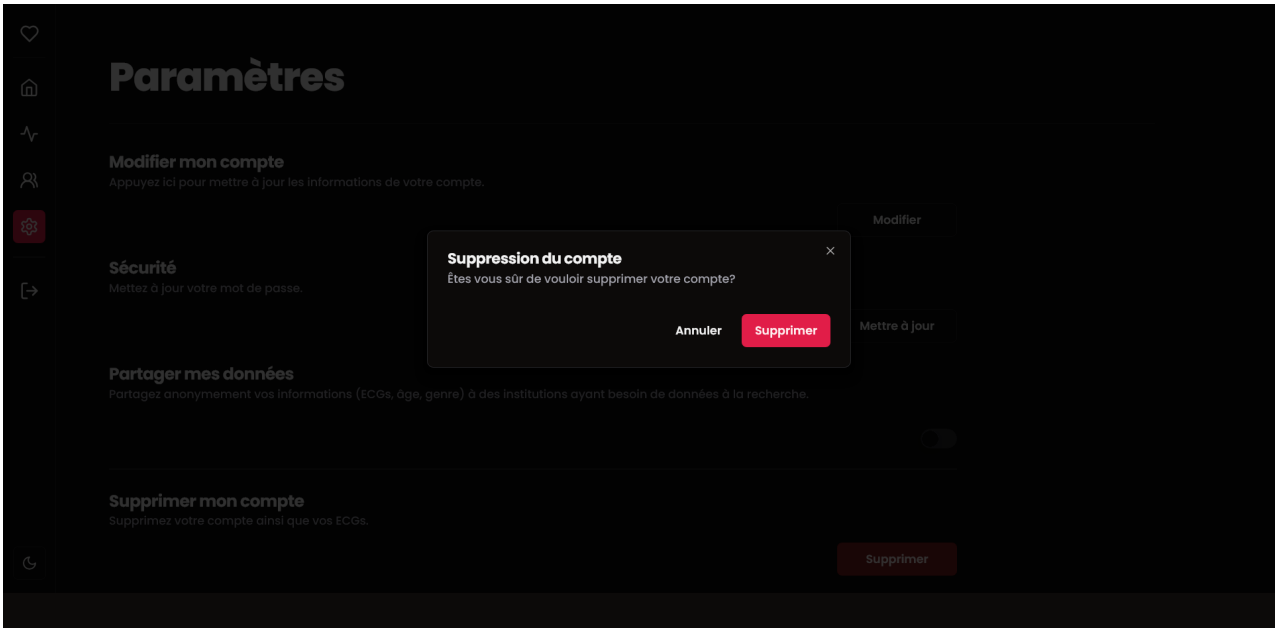


Figure 3.19: Interface des suppression du compte.

Une méthode a été implémentée dans le contrôleur utilisateur qui est la déconnexion, via la fonction `logout()`, qui supprime son token d'accès de la mémoire et de ses requêtes HTTP. Cette méthode déclenche aussi une navigation vers la route de la page d'accueil de l'application.

3.9.2 Sprint 02

Ce second Sprint traitera des fonctionnalités relatives à la gestion des profils :

Product Backlog

User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Durée estimée	Date de début	Date de fin
US_PROFILE	Utilisateur	Gestion des profils	Consultation de la liste des profils	01 semaine	09/04/2024	16/04/2024
			Création d'un profil	01 semaine	20/04/2024	27/04/2024
			Modification d'un profil	03 jours	30/04/2024	02/05/2024
			Suppression d'un profil	03 jours	02/05/2024	05/05/2024
			Gestion des images de profils	01 semaine	05/05/2024	12/05/2024

Table 3.23: Product Backlog du Sprint 02.

Diagramme de cas d'utilisation

Contenant quatre nouveaux cas d'utilisation :

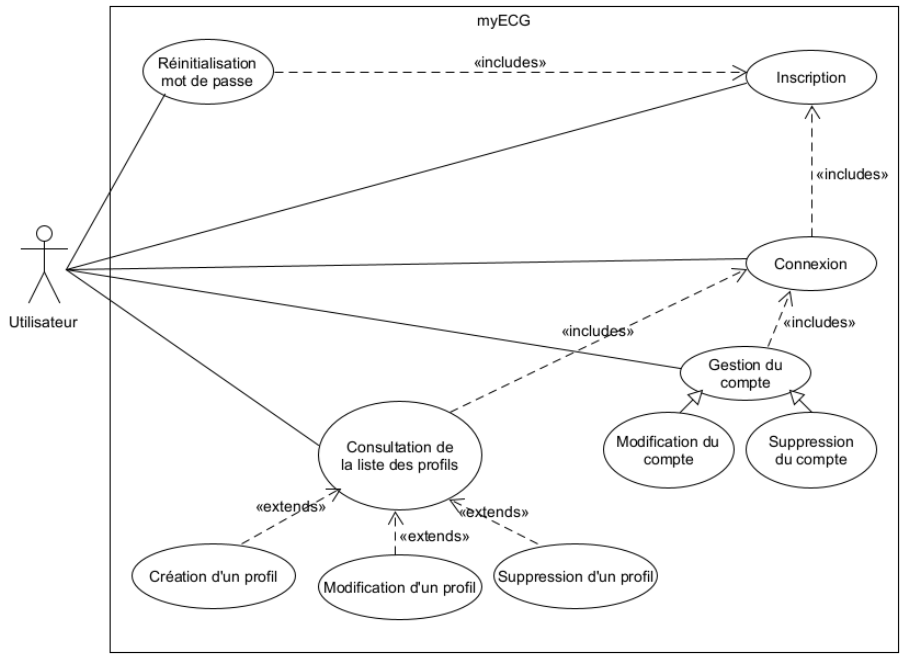


Figure 3.20: Diagramme de cas d'utilisation du Sprint 02.

Diagrammes de séquence

Quatre nouveaux diagrammes de séquence sont donc déduits :

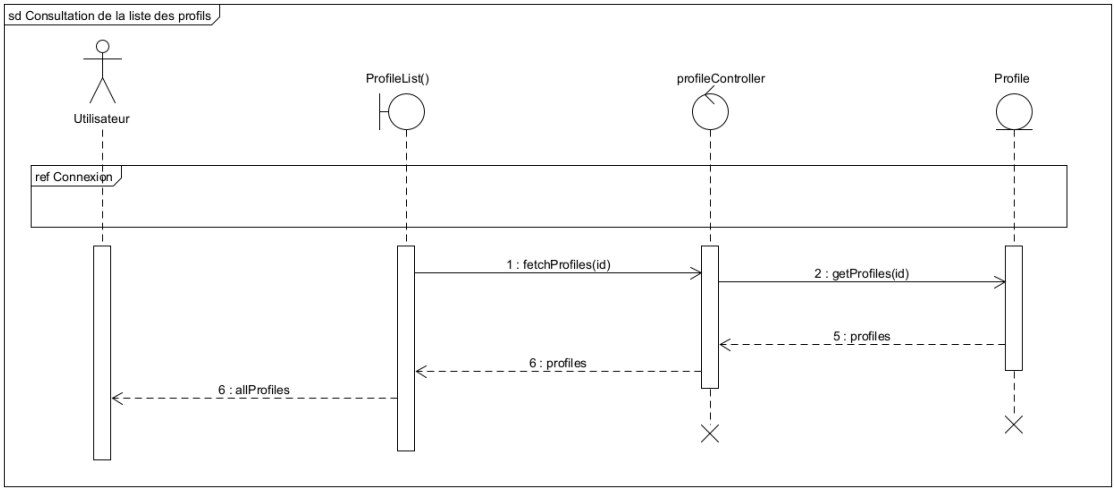


Figure 3.21: Diagramme de séquence "Consultation de la liste des profils".

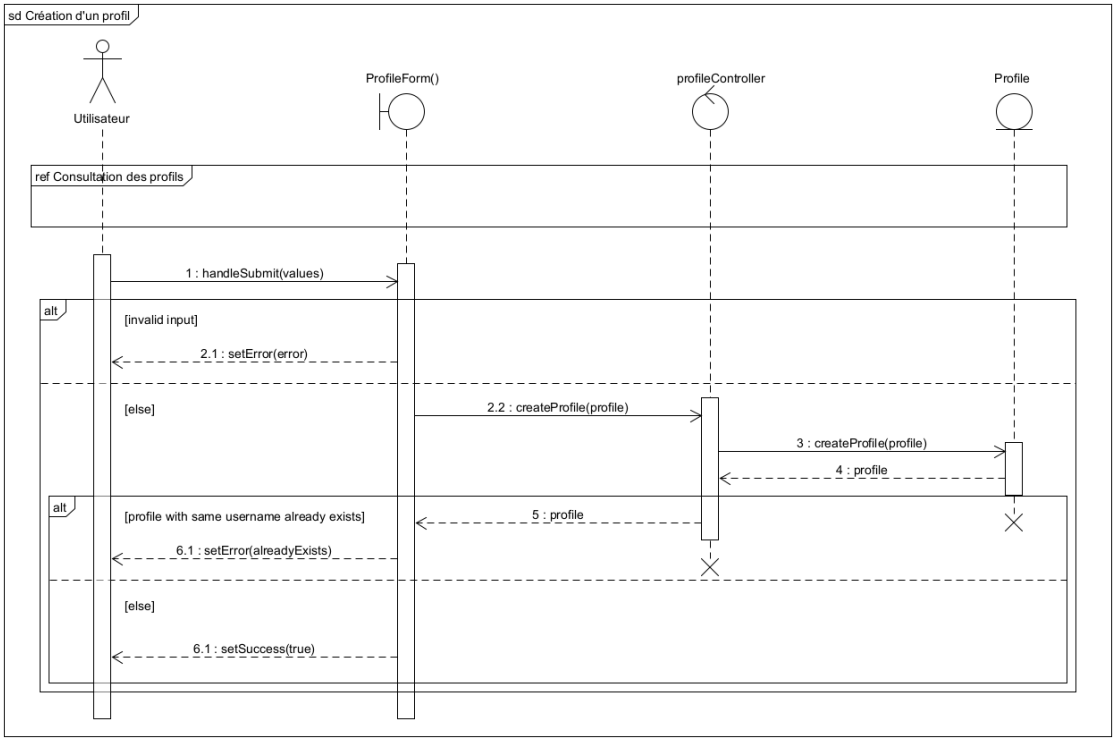


Figure 3.22: Diagramme de séquence ”Création d’un profil”.

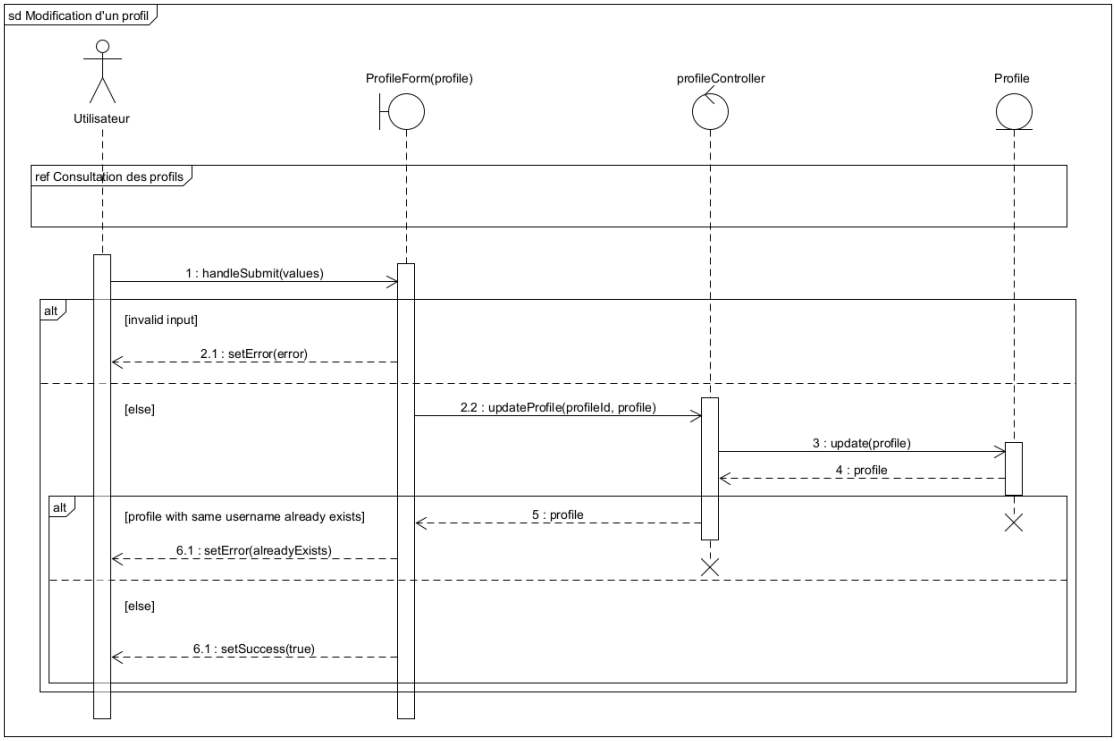


Figure 3.23: Diagramme de séquence "Modification d'un profil".

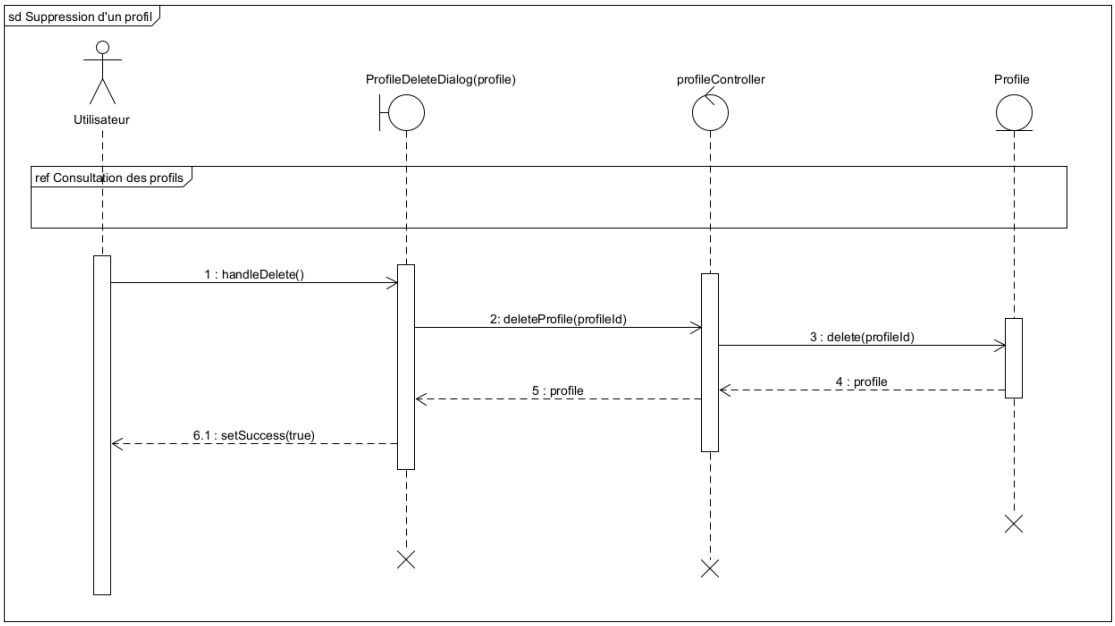


Figure 3.24: Diagramme de séquence "Suppression d'un profil".

Diagramme de classe

La classe profil a été introduite au diagramme de classe :

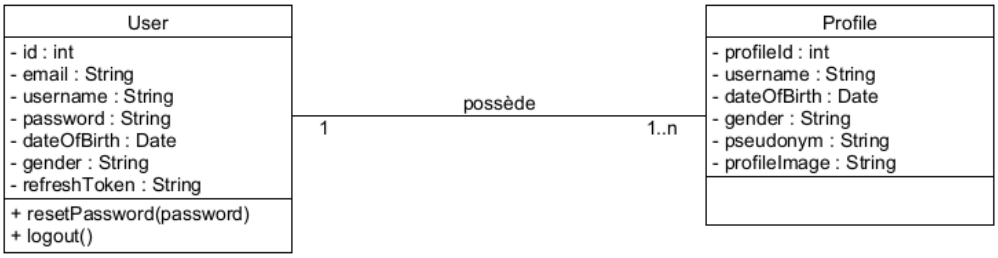


Figure 3.25: Diagramme de classe du Sprint 02.

Modèle relationnel

La table "Profile" a été introduite :

User(id, email, username, password, dateOfBirth, gender, refreshToken)
Profile(#userId, profileId, username, pseudonym, dateOfBirth, gender, profileImage)

Contraintes :

- L'attribut `id` est la clé primaire de la table "User" ;
- Les attributs `username` et `email` sont uniques dans la table "User" ;
- L'attribut `profileId` est la clé primaire de la table "Profile" ;
- L'attribut `userId` fait référence à l'attribut `id` de la table "User" ;
- Le couple (`userId`, `username`) est unique dans la table "Profile".

Implémentation

La contrainte d'unicité du couple (`userId`, `username`) garantit que deux profils au sein d'un même compte ne peuvent porter le même nom d'utilisateur.

Nous avons effectué des modifications au contrôleur de l'utilisateur (`userController`), plus précisément dans la fonction responsable de l'inscription (`register`) de sorte à ce qu'elle ne crée plus qu'un utilisateur seulement, mais crée un utilisateur et un profil de base dont les attributs sont déduits de ceux du compte.

Cette opération est effectuée sous la forme d'une transaction dans la base de données afin de garantir la création du compte et du profil de base. Le profil de base s'affiche automatiquement sur la liste des profils :

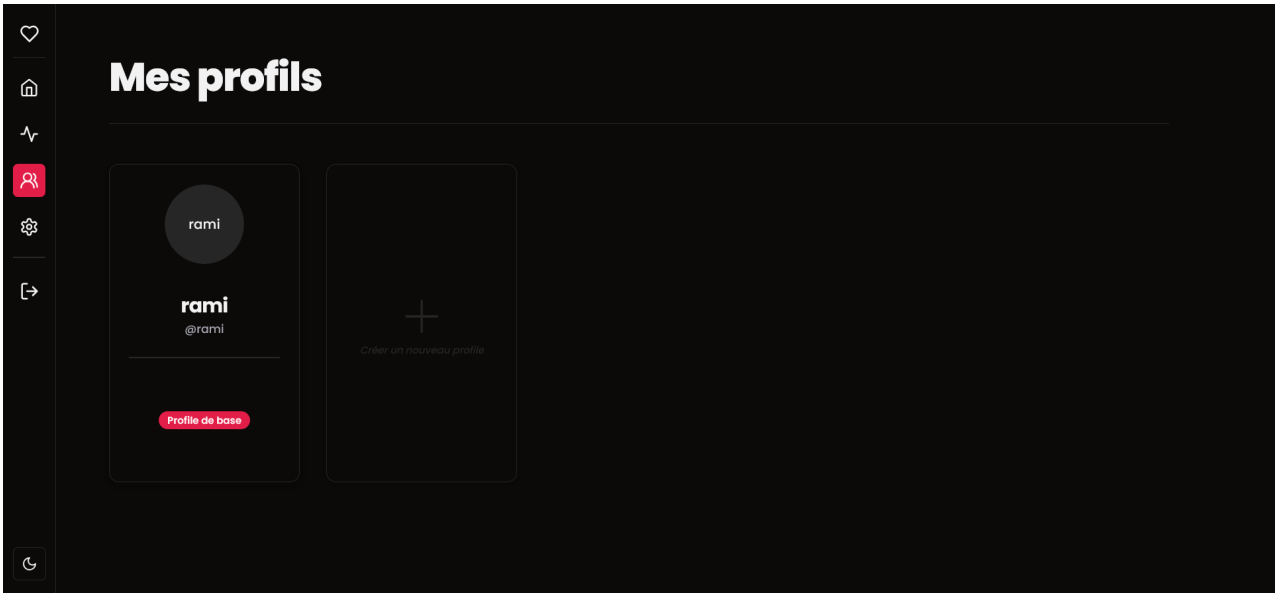


Figure 3.26: Interface de la liste des profils.

D'où l'on peut créer un profil :

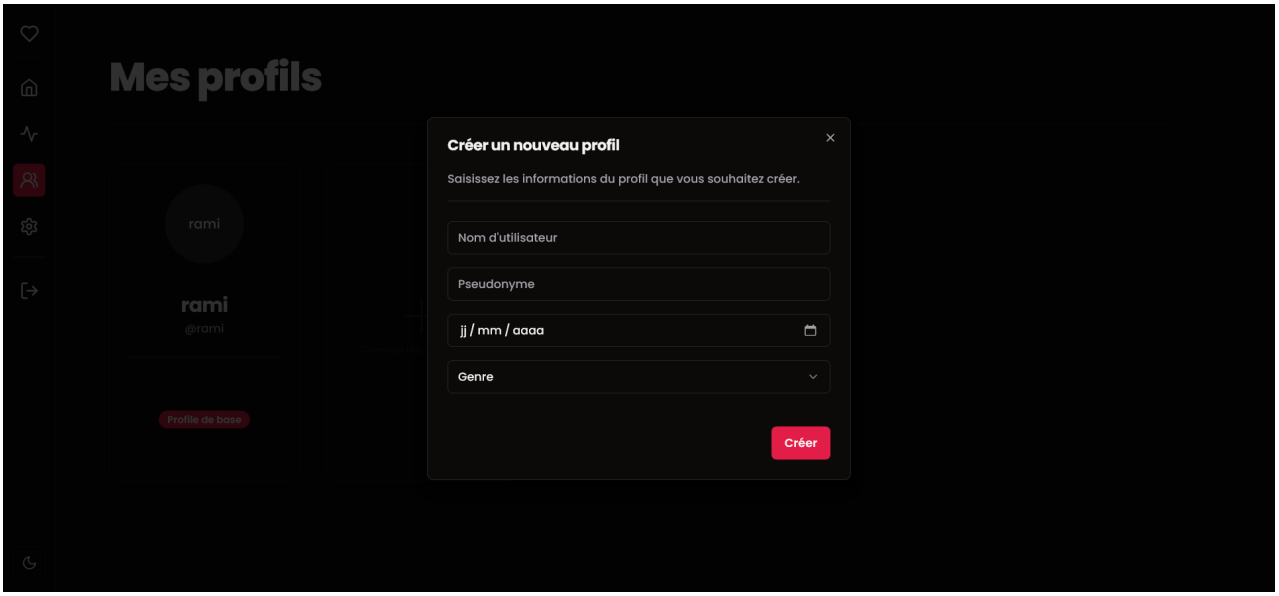


Figure 3.27: Formulaire de création d'un profil.

Qui est immédiatement ajouté à la liste des profils grâce au module de la gestion des états de la technologie React qui ne nécessite pas de rafraîchissements :

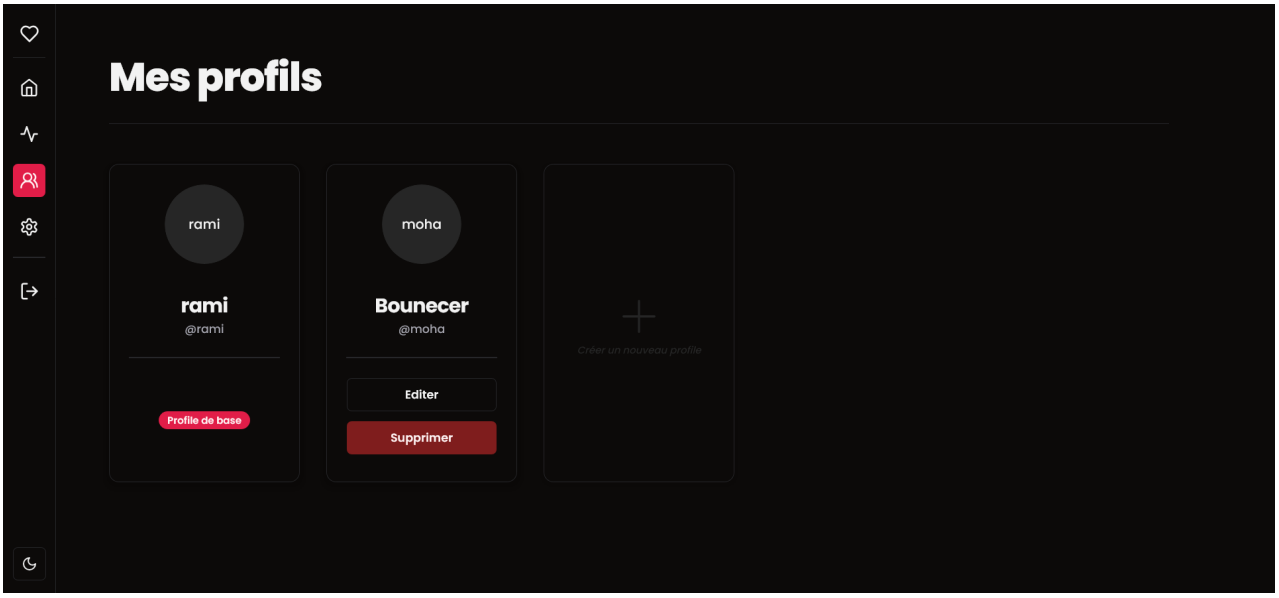


Figure 3.28: Ajout d'un profil.

Les profils peuvent être personnalisés avec des photos que l'on peut importer à partir de l'appareil. Ceci est rendu possible grâce au package `multer` qui se charge de recevoir les fichiers envoyées du côté Client vers le Serveur en facilitant le stockage des fichiers reçus ainsi que leur filtrage. Par exemple, seulement les images d'extension `jpg`, `jpeg` ou `png` sont acceptés :

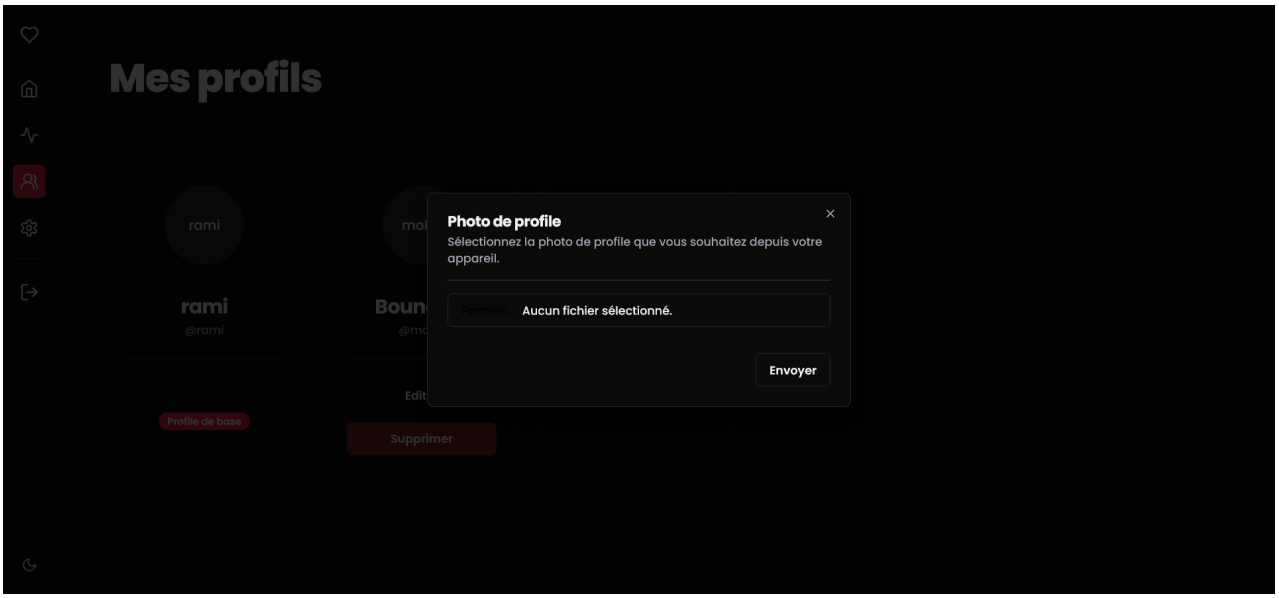


Figure 3.29: Interface de sélection de photo de profil.

Dans le serveur, un dossier `images` est localisé dans le même répertoire que le processus

principal `server.js` (point d'entrée du Serveur), il contient toutes les photos de profils d'un compte sous forme d'images avec pour nom :

`[id]_[profileId].[jpg|jpeg|png]`

Le chemin vers chaque photo de profil est stocké dans la base de données sous la forme d'un champ `profileImage` afin de faciliter la récupération de ces ressources :

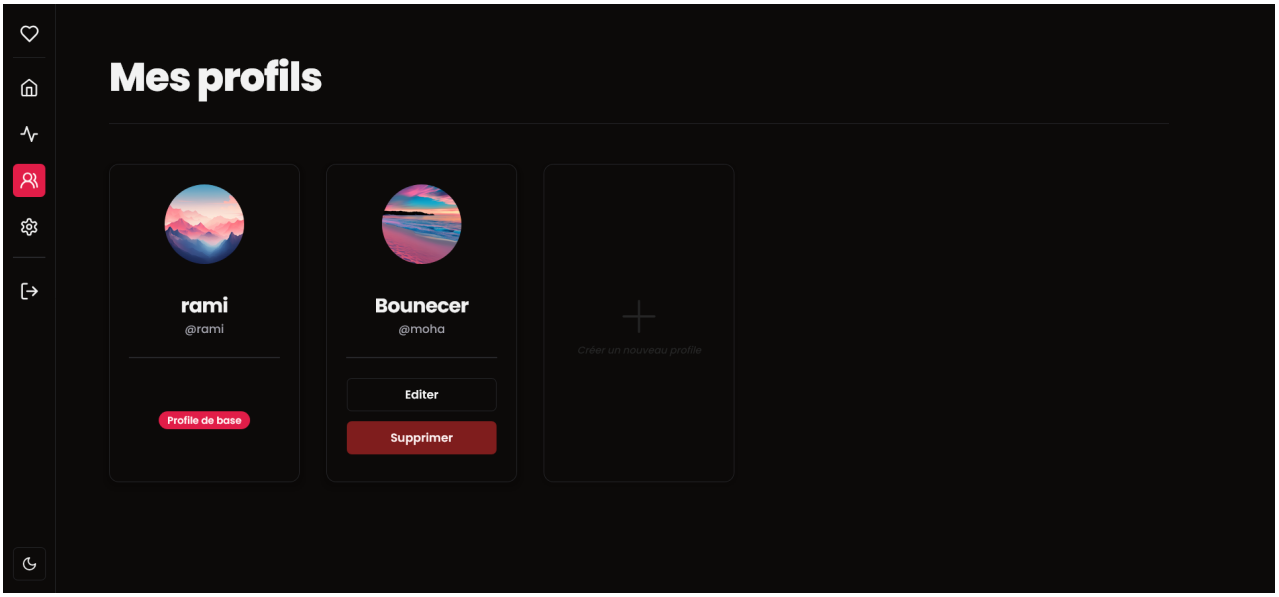


Figure 3.30: Personnalisation des profils avec des images.

Les interfaces de modification et suppression de profils sont similaires à celles de l'utilisateur :

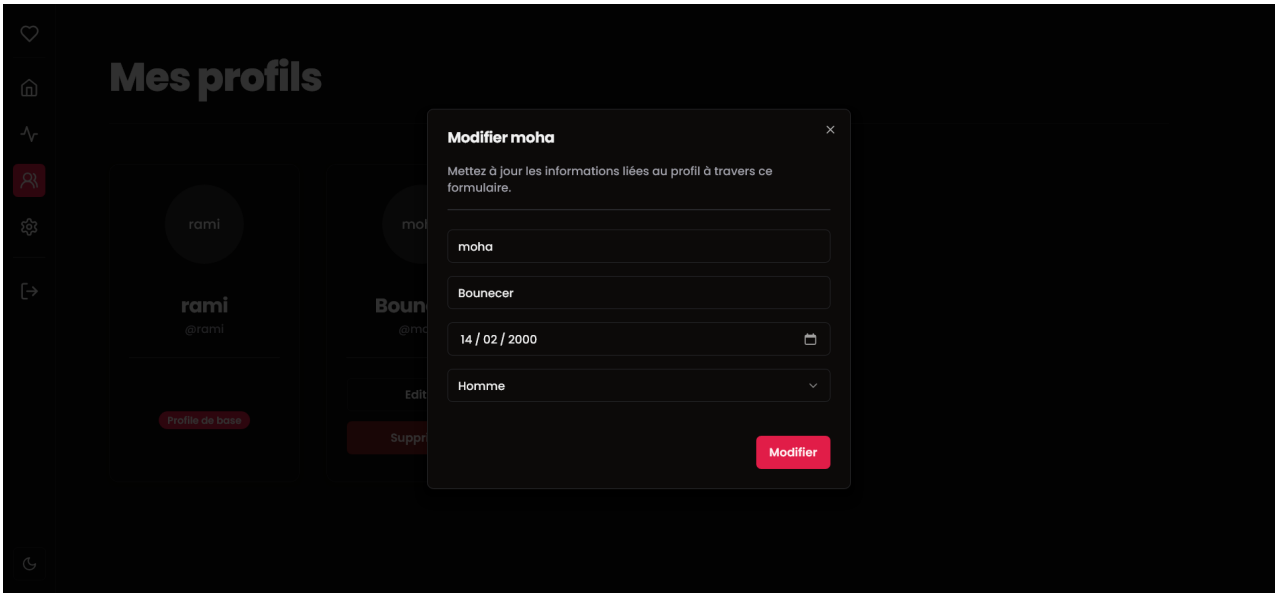


Figure 3.31: Formulaire de modification d'un profil.

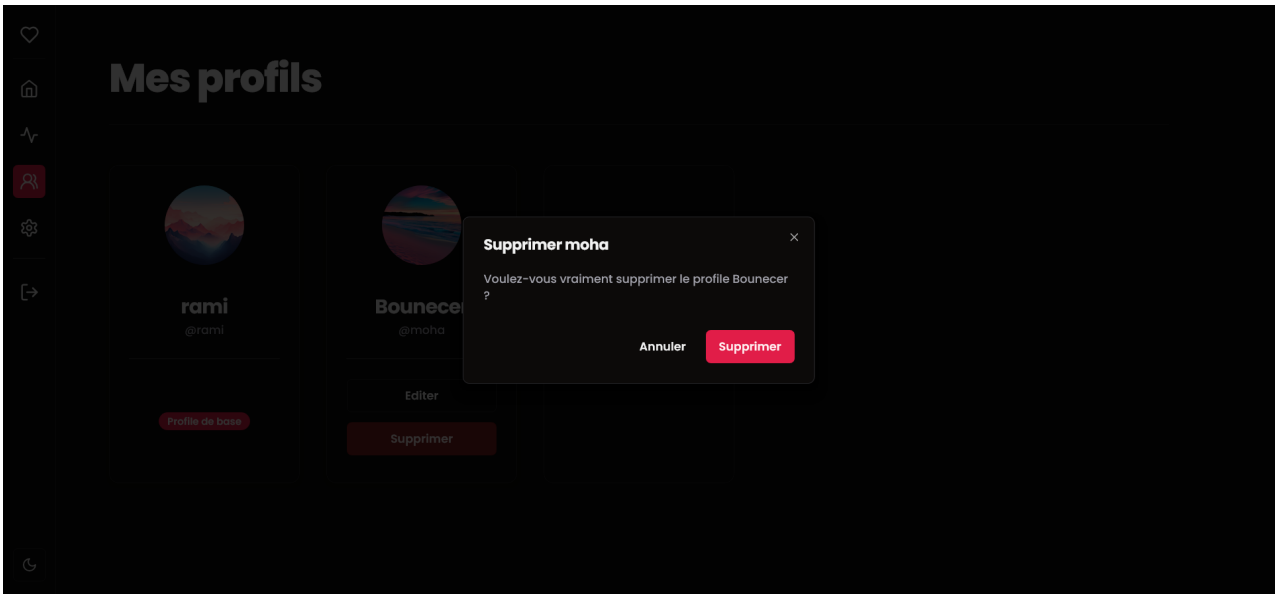


Figure 3.32: Confirmation de suppression d'un profil.

Le contrôleur du profil `profileController` a été modifié de sorte à ce que la suppression du profil entraîne aussi la suppression d'une photo de profil si ce dernier en avait une, cela se réalise en supprimant le fichier localisé dans le chemin donné par le champ `profileImage` de la table "Profile". De la même manière, la suppression d'un compte utilisateur entraîne la suppression de toutes les photos de profil associées au compte, c'est à dire tous les fichiers dans le dossier `images` commençant par "`[id]_`".

3.9.3 Sprint 03

Ce troisième Sprint traitera des fonctionnalités liées au scan des ECGs :

Product Backlog

User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Durée estimée	Date de début	Date de fin
US_SCAN	Utilisateur	Scan d'un ECG	Scan d'un ECG à partir d'une photo	01 mois	13/05/2024	13/06/2024
			Téléchargement d'un scan	03 jours	13/06/2024	16/06/2024

Table 3.24: Product Backlog du Sprint 03.

Diagramme de cas d'utilisation

Deux nouveaux cas d'utilisation ont été ajoutés :

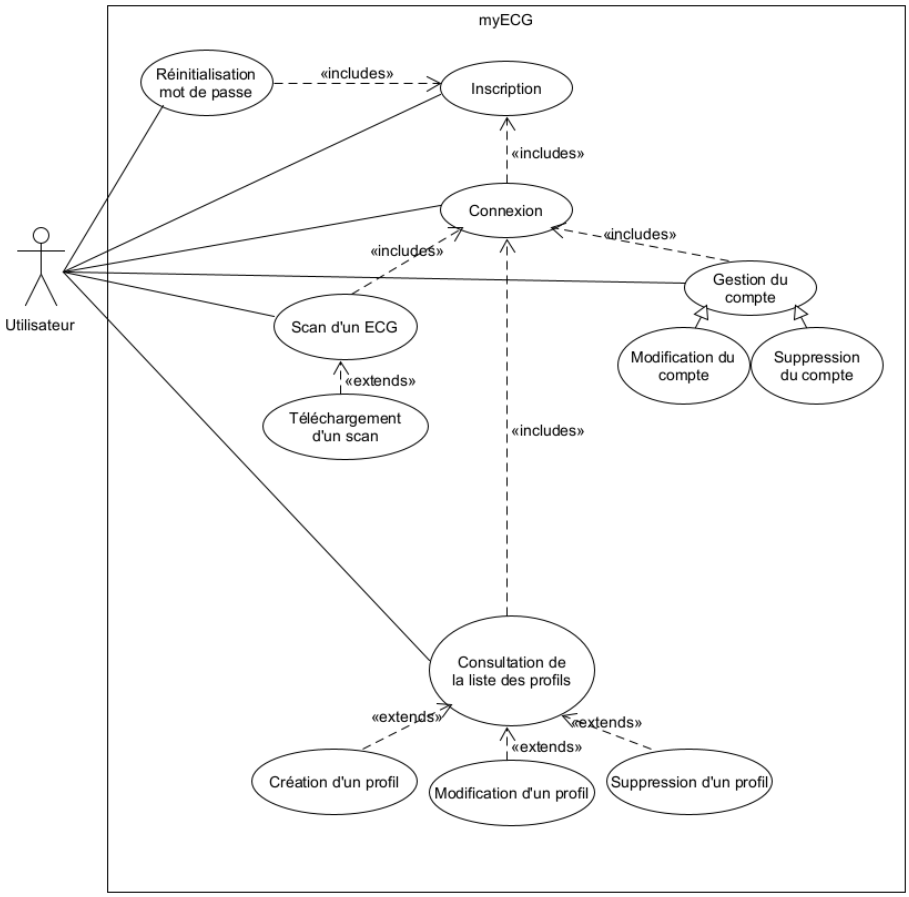


Figure 3.33: Diagramme de cas d'utilisation du Sprint 03.

Diagrammes de séquence

On en déduit deux diagrammes de séquence :

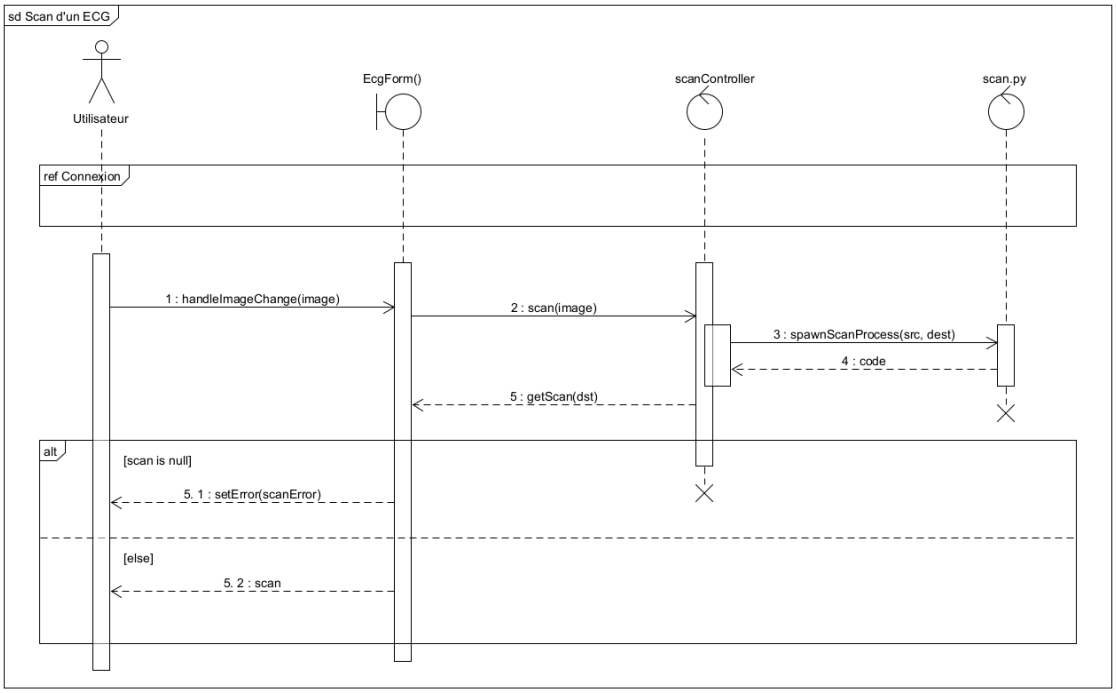


Figure 3.34: Diagramme de séquence du cas d'utilisation "Scan d'un ECG".

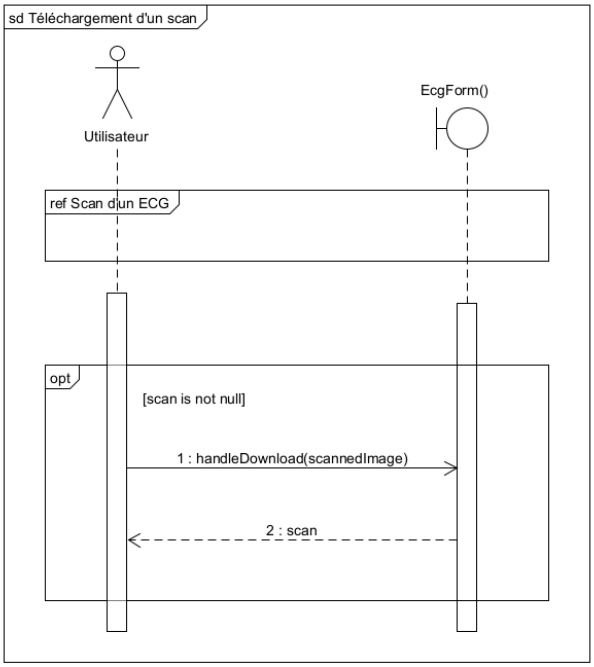


Figure 3.35: Diagramme de séquence du cas d'utilisation "Téléchargement d'un scan".

Diagramme de classe

Deux nouvelles méthodes sont ajoutées à la classe "User" :

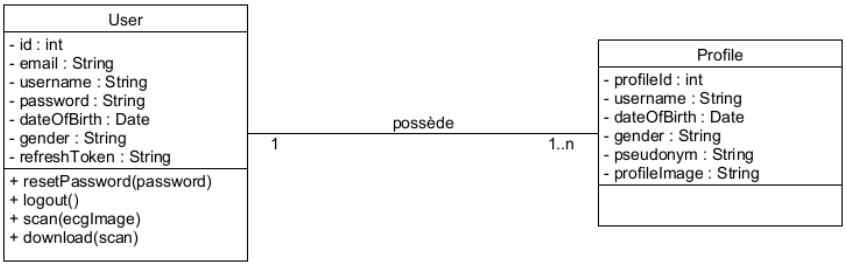


Figure 3.36: Diagramme de classe du Sprint 03.

Modèle relationnel

Le modèle relationnel reste inchangé.

Implémentation

Après la connexion, l'utilisateur est envoyé vers une page qui représente son tableau de bord :

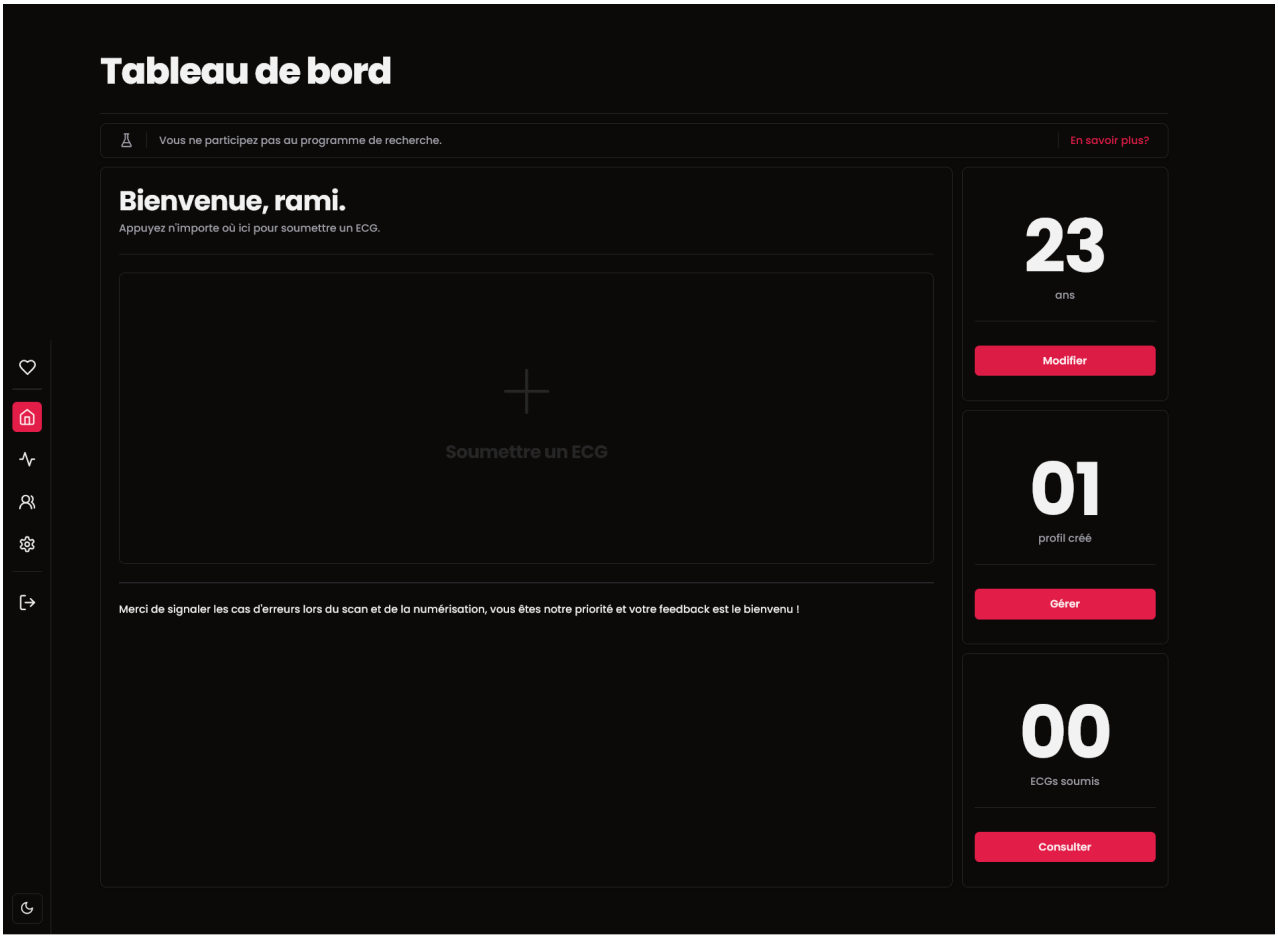


Figure 3.37: Interface récapitulative du compte (Tableau de bord).

Cette interface contient des informations sur l'utilisateur et son compte, telles que son âge et le nombre de profils créés. Elle contient une grande zone cliquable qui déclenche l'apparition du formulaire de soumission d'un ECG :

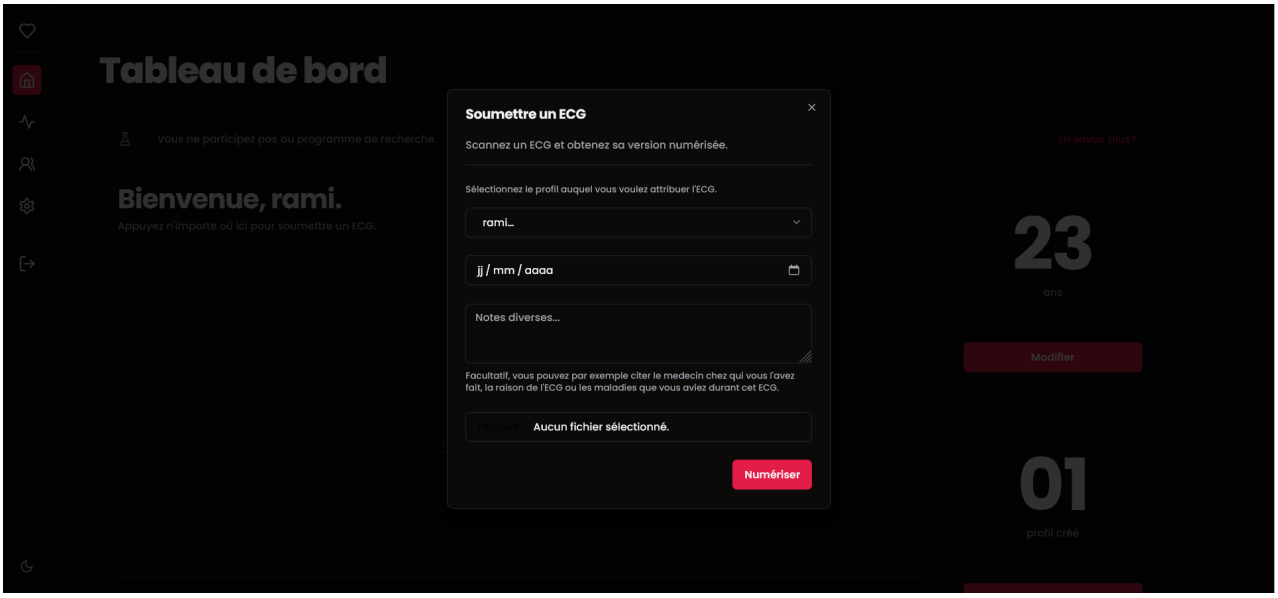


Figure 3.38: Interface de soumission d'un ECG.

Le dernier champ du formulaire sert à téléverser une photo d'un ECG :

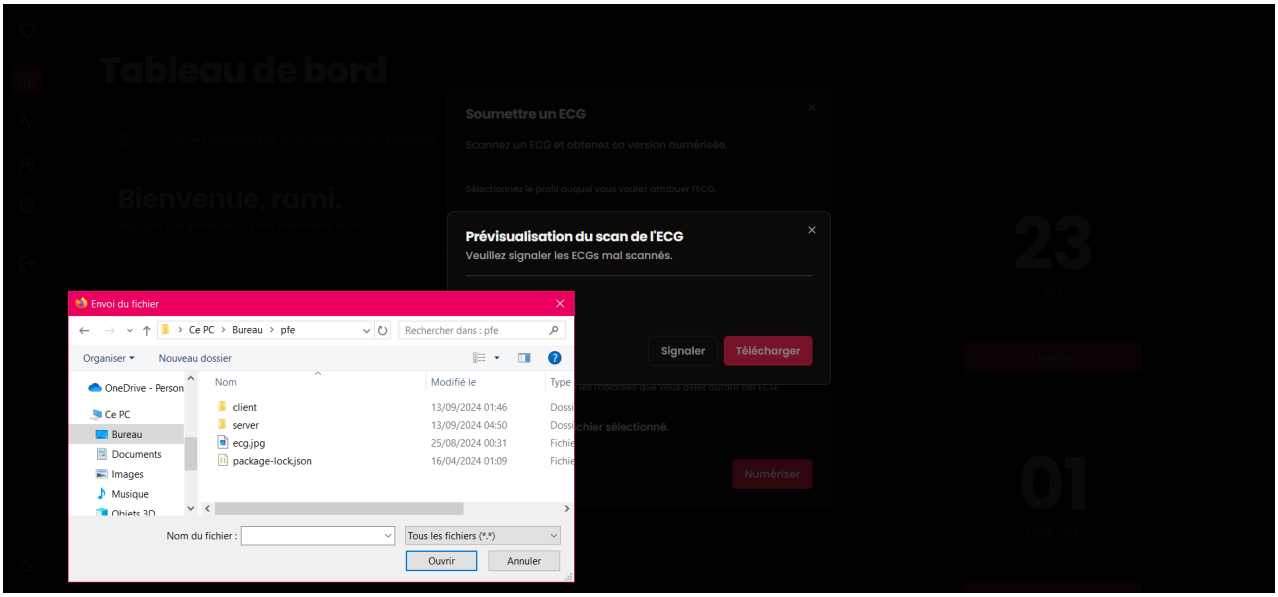


Figure 3.39: Téléversement de la photo d'un ECG.

Aussitôt que l'application détecte un changement de l'entrée de ce champ (l'utilisateur téléverse une image), elle déclenche le module du scan de l'ECG en notifiant l'utilisateur :

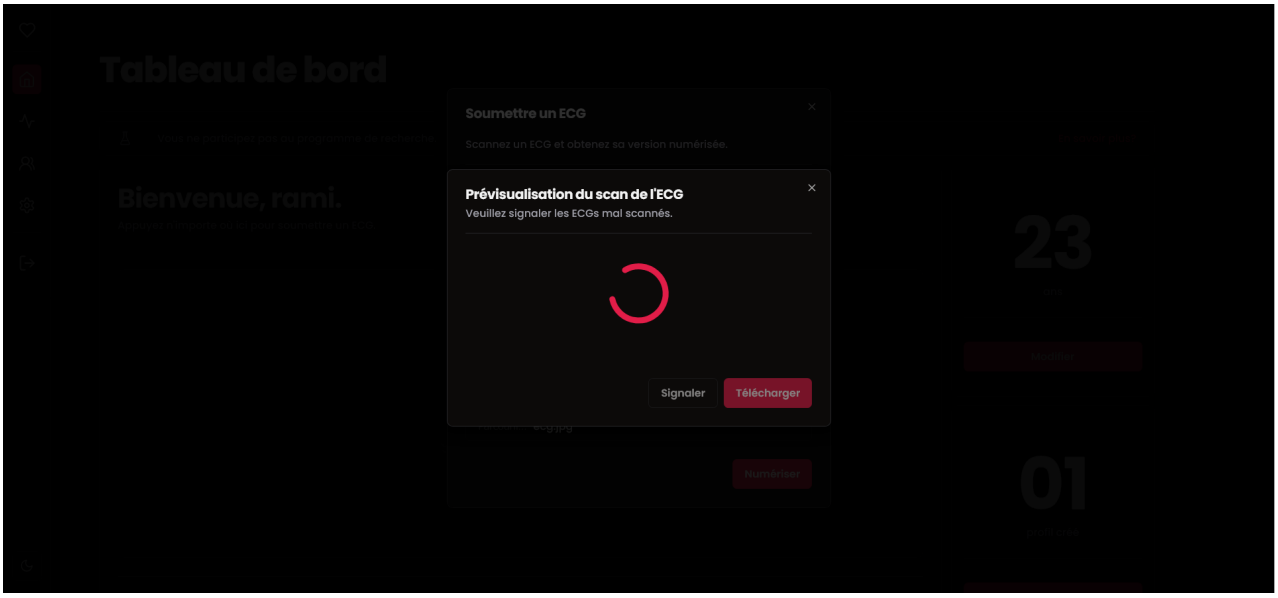


Figure 3.40: Chargement du module de scan d'un ECG.

Durant ce chargement, le Frontend envoie le fichier d'entrée au Backend et ce dernier vérifie la validité de l'entrée, c'est-à-dire si le fichier envoyé est un fichier image (comme précédemment réalisé pour les photos de profils) et renomme le fichier d'entrée en `input.png` si c'est le cas.

Ensuite, le Serveur enregistre l'entrée dans un dossier nommé `temp_[id]` (`id` étant l'identifiant de l'utilisateur) localisé dans le répertoire `images`. Les entrées candidates aux scans sont séparées par utilisateur dans le but de garantir l'exclusion mutuelle entre les processus de scan concurrents sur les ressources exploitées (photos d'ECGs) au cas où plusieurs utilisateurs seraient en train d'effectuer cette opération sur l'application.

Comme nous pouvons le voir dans le diagramme de séquence du cas d'utilisation "Scan d'un ECG", le contrôleur responsable du scan est un script Python 3 `scan.py`, c'est un module parmi les modules de scan et numérisation des ECGs dont nous allons détailler la réalisation dans le dernier chapitre. Ce module prend comme entrée le chemin vers l'image cible du scan et le chemin de destination du scan.

Le fichier de sortie est localisé dans le même répertoire que l'entrée enregistrée, mais se nomme `scan.png`.

L'invocation du script se fait en utilisant la fonction `spawn` du package `child_process` avec les deux paramètres cités auparavant, cela permet d'exécuter un processus enfant directement sur la machine Serveur ayant reçu la requête.

Les paramètres sont récupérées dans le script (sous forme d'arguments) et son exécution retourne un code `code` via la fonction `exit` de la librairie `sys` de Python. Ce code est ensuite capturé par le Serveur et indique si l'opération a été un succès (`code` est égal à 0) ou un échec

(`code` est égal à 1).

Enfin, le Serveur renvoie l'image `temp_[id]/scan.png` si le code retourné par le processus enfant vaut 0, sinon il renvoie une erreur qui sera affichée sur le côté Client :

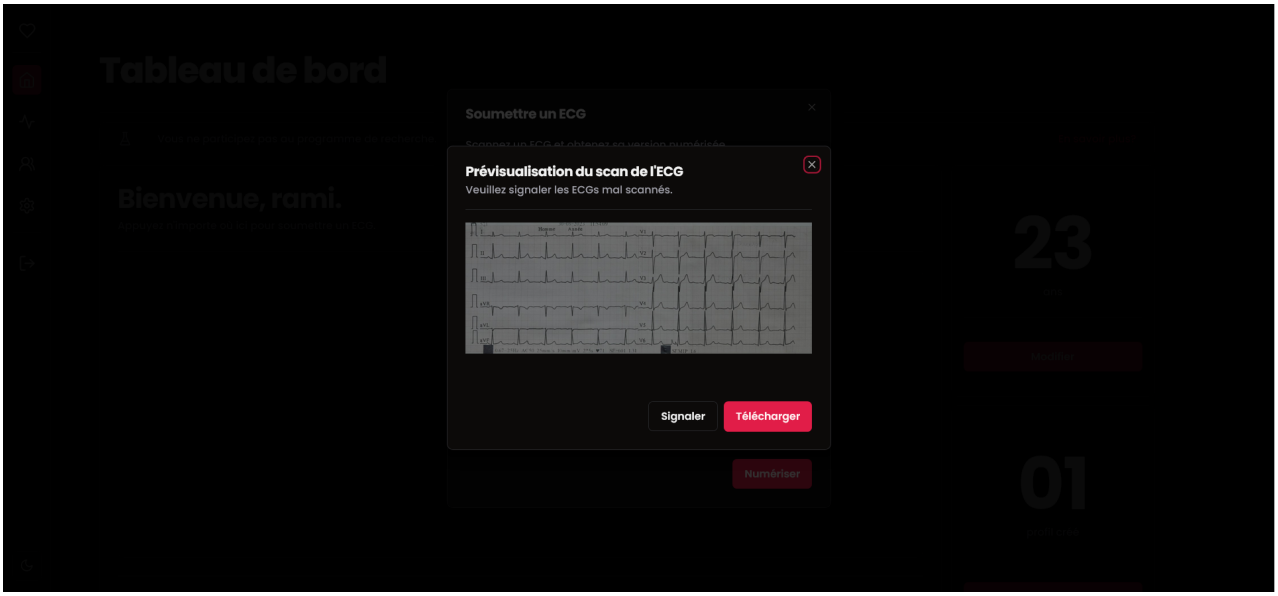


Figure 3.41: Interface de scan d'un ECG.

L'image du scan est stocké dans la mémoire vive côté Client, ainsi, l'utilisateur peut choisir de la télécharger sur son appareil :

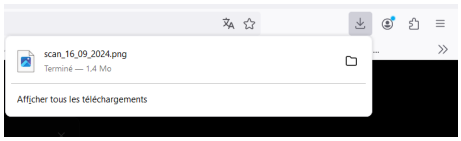


Figure 3.42: Téléchargement du scan obtenu.

Une limitation d'exécution de 20 processus de scan en concurrence a été implémentée dans le Serveur afin d'éviter la surcharge de ce dernier, grâce au package `p-queue` qui permet d'enfiler les processus déclenchés et de défiler les processus terminés. Si la file est pleine, la requête reste suspendue jusqu'à ce qu'un processus termine son exécution.

La fonction responsable de la suppression d'un compte utilisateur a été mise à jour de sorte à déclencher la suppression du dossier `images/temp_[id]` s'il existe.

3.9.4 Sprint 04

Ce quatrième Sprint traitera de la fonctionnalité de numérisation des ECGs :

Product Backlog

User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Durée estimée	Date de début	Date de fin
US_DIGITAL	Utilisateur	Gestion et numérisation des ECGs	Numérisation d'un ECG scanné	02 mois	17/06/2024	17/08/2024
			Consultation des ECGs	04 jours	18/08/2024	22/08/2024
			Suppression d'un ECG	02 jours	23/08/2024	25/08/2024
			Import d'un ECG	02 jours	26/08/2024	27/08/2024

Table 3.25: Product Backlog du Sprint 04.

Diagramme de cas d'utilisation

Inclut quatre nouveaux cas d'utilisation :

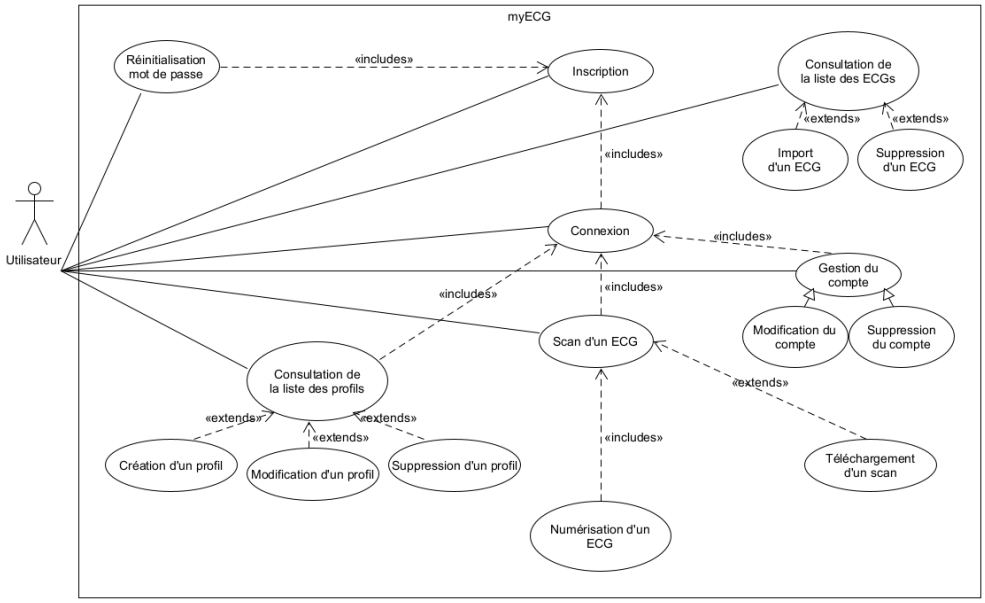


Figure 3.43: Diagramme de cas d'utilisation du Sprint 04.

Diagrammes de séquence

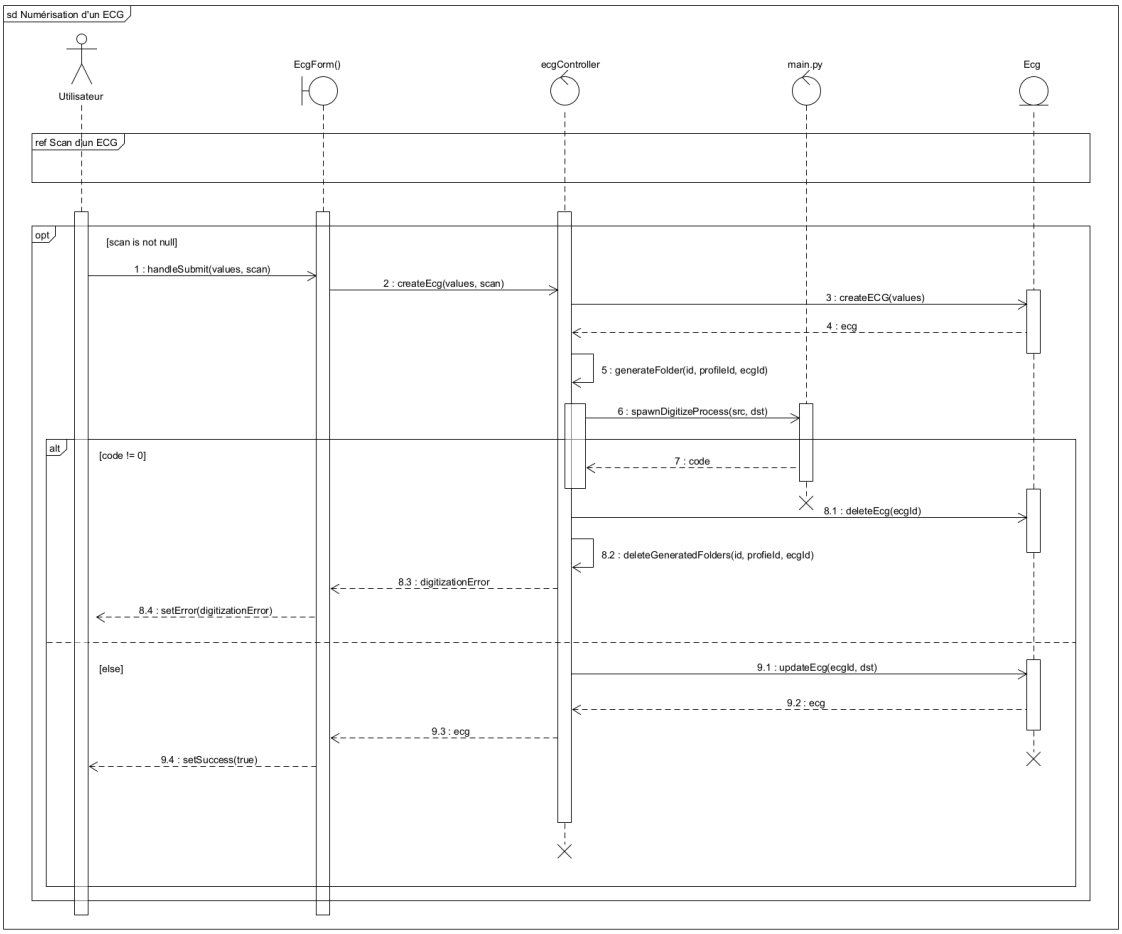


Figure 3.44: Diagramme de séquence du cas d'utilisation "Numérisation d'un ECG".

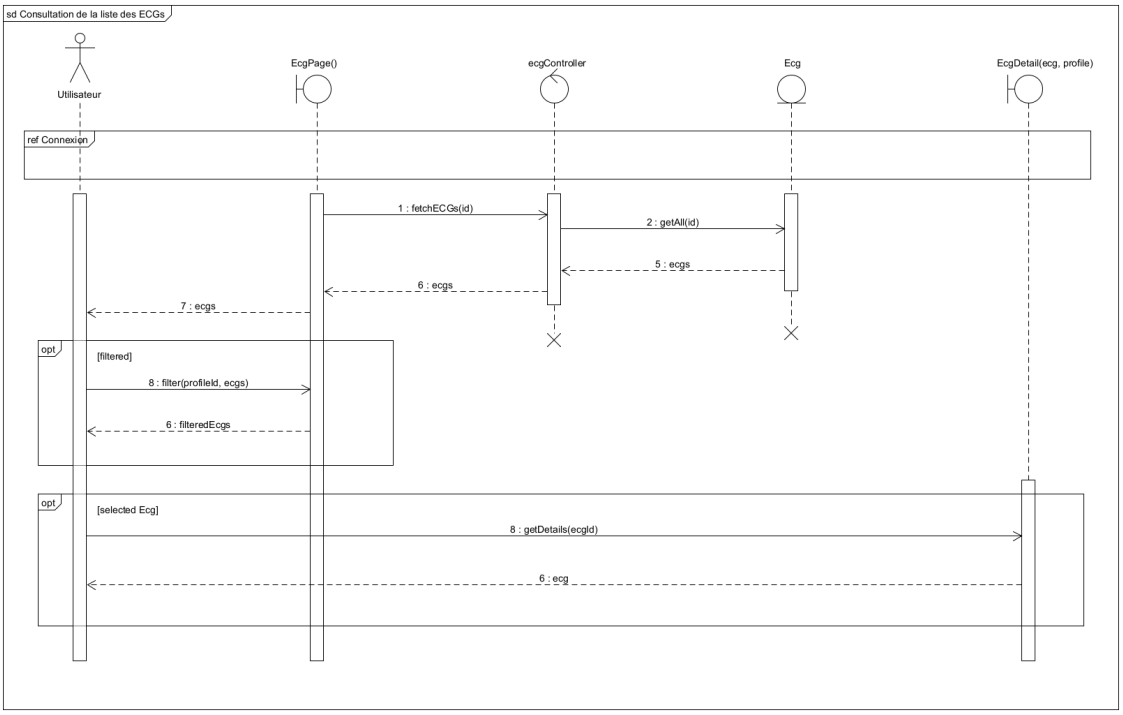


Figure 3.45: Diagramme de séquence du cas d'utilisation "Consultation de la liste des ECGs".

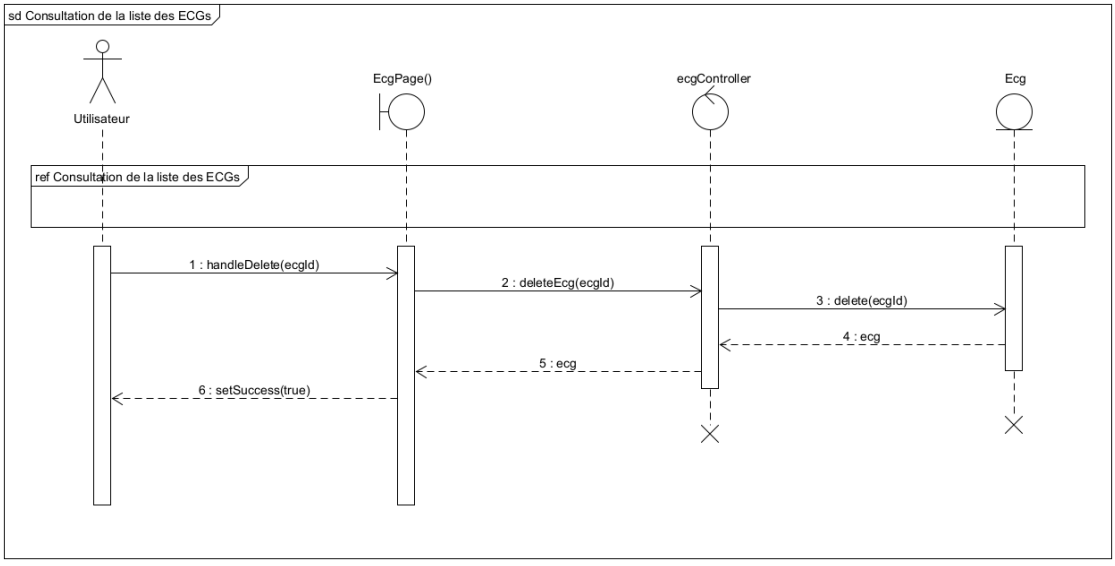


Figure 3.46: Diagramme de séquence du cas d'utilisation "Suppression d'un ECG".

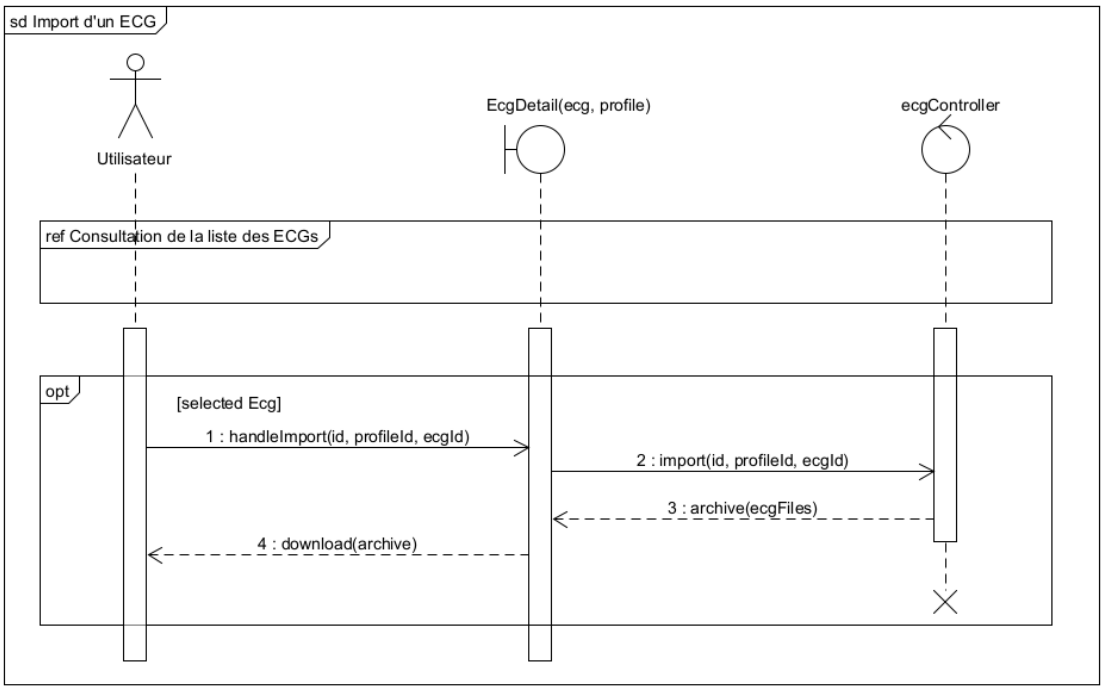


Figure 3.47: Diagramme de séquence du cas d'utilisation "Import d'un ECG".

Diagramme de classe

Une nouvelle classe "ECG" a été ajoutée :

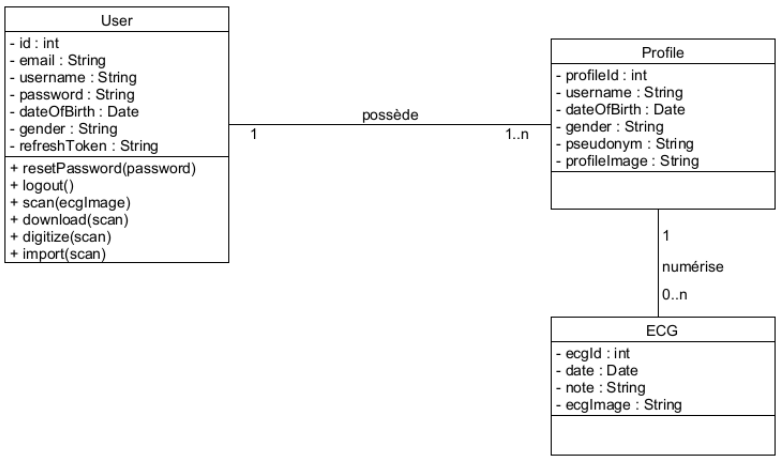


Figure 3.48: Diagramme de classe du Sprint 04.

Modèle relationnel

La table ”Ecg” a été introduite :

User(id, email, username, password, dateOfBirth, gender, refreshToken)
Profile(#userId, profileId, username, pseudonym, dateOfBirth, gender, profileImage)
Ecg(#profileId, ecgId, date, note, ecgImage)

Contraintes :

- L’attribut `id` est la clé primaire de la table ”User” ;
- Les attributs `username` et `email` sont uniques dans la table ”User” ;
- L’attribut `profileId` est la clé primaire de la table ”Profile” ;
- L’attribut `userId` fait référence à l’attribut `id` de la table ”User” ;
- Le couple (`userId`, `username`) est unique dans la table ”Profile” ;
- L’attribut `ecgId` est la clé primaire de la table ”Ecg” ;
- L’attribut `profileId` de la table ”Ecg” fait référence à l’attribut `profileId` de la table ”Profile”.

Implémentation

La numérisation d’un ECG peut se faire directement à partir du tableau de bord :

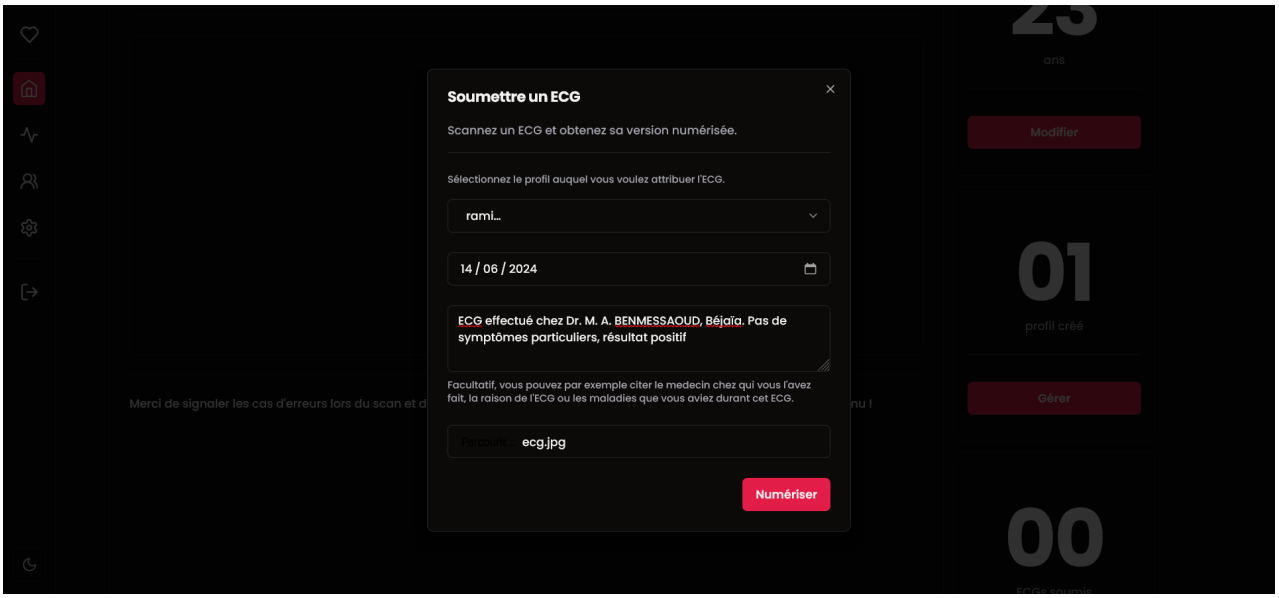


Figure 3.49: Formulaire de soumission d’un ECG à la numérisation.

La numérisation nécessite d'abord de scanner l'ECG comme il a été montré précédemment. Ensuite, l'utilisateur remplit les champs restants si ce n'est pas déjà fait et soumet son ECG à la numérisation.

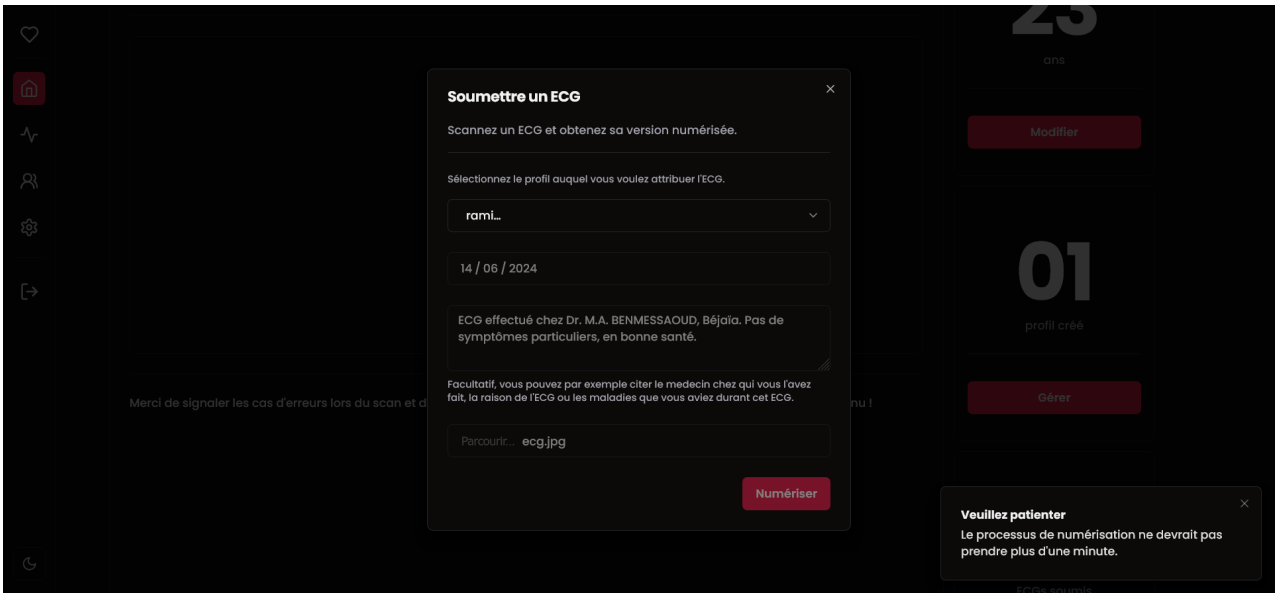


Figure 3.50: Lancement du numérisation d'un ECG.

Le scan, auparavant stocké dans la mémoire vive, est directement envoyé au Backend en plus des informations relatives à l'ECG.

D'abord, le Backend prépare le répertoire des données de la numérisation en créant un nouvel ECG dans la base de données, puis récupère son identifiant `ecgId`. Ensuite, il crée une suite de dossiers qui est :

`images/[id]/[profileId]/[ecgId]`

Le répertoire obtenu servira de destination du scan (`scan.png`) envoyé par le Frontend.

Le contrôleur responsable de la numérisation est aussi un module Python `main.py` qui prend comme paramètres un chemin de l'image source à numériser et le chemin de destination des données retournées (qui sont le dossier `ecgId`). L'implémentation de ce module sera aussi détaillée dans le dernier chapitre.

De la même manière que pour le scan, ce processus retourne un `code` capturé par le serveur. Cependant, si ce processus échoue, l'ECG auparavant créé dans la base de données l'ensemble des répertoires préparés sont supprimés.

Si le processus de numérisation s'exécute avec succès, un nouveau fichier JSON [23] (RFC 8259) nommé `ecg.json` est créé, contenant pour clés les douze signaux de dérivation et pour

chaque clé un vecteur de coordonnées (x, y) représentatifs des points composants les signaux de dérivation.

De la même manière que pour le scan, le nombre de processus de numérisation concurrents est limité à 5, car les ressources demandées par le processus de la numérisation sont bien plus conséquentes que celles du processus de scan.

Les ECGs numérisés peuvent être consultés sur l'interface de consultation de la liste des ECGs :

Figure 3.51: Interface de consultation de la liste des ECGs.

La note fournie peut être visualisée en survolant le bouton "note" ou en cliquant dessus :

Figure 3.52: Visualisation de la note écrite de l'ECG.

Les ECGs peuvent être filtrés par profil :

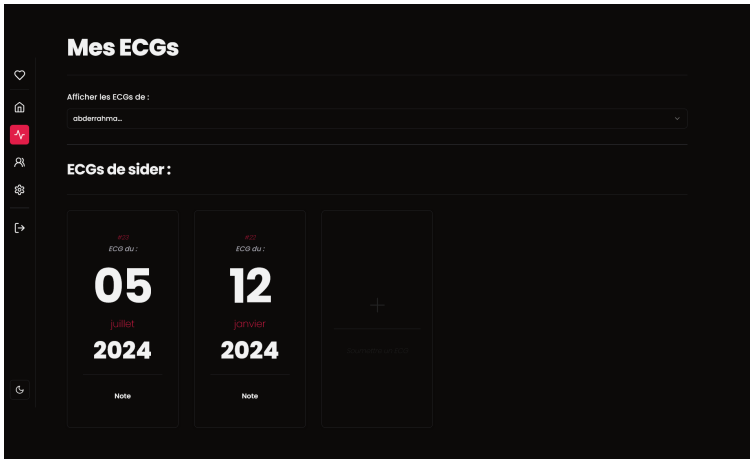


Figure 3.53: Filtrage d'ECGs par profil.

Dans le cas où l'utilisateur veuille numériser un ECG à partir du tableau de bord, le profil proposé par défaut est le profil de base. Par contre, s'il veut numériser un ECG à partir de la liste des ECGs, aucun profil n'est proposé et l'utilisateur doit donc en sélectionner un :

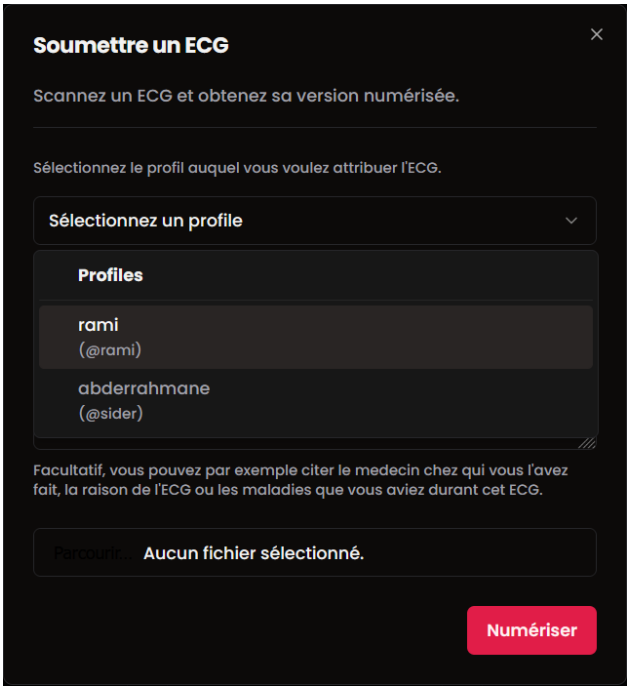


Figure 3.54: Sélection du profil auquel attribuer un ECG.

Ceci s'applique dans le cas où l'utilisateur n'a pas décidé de filtrer les ECGs. Au contraire, si le profil sélectionné comme filtre est mis comme valeur par défaut et ne peut être changé, pour attirer l'attention de l'utilisateur sur l'état du profil actuel :

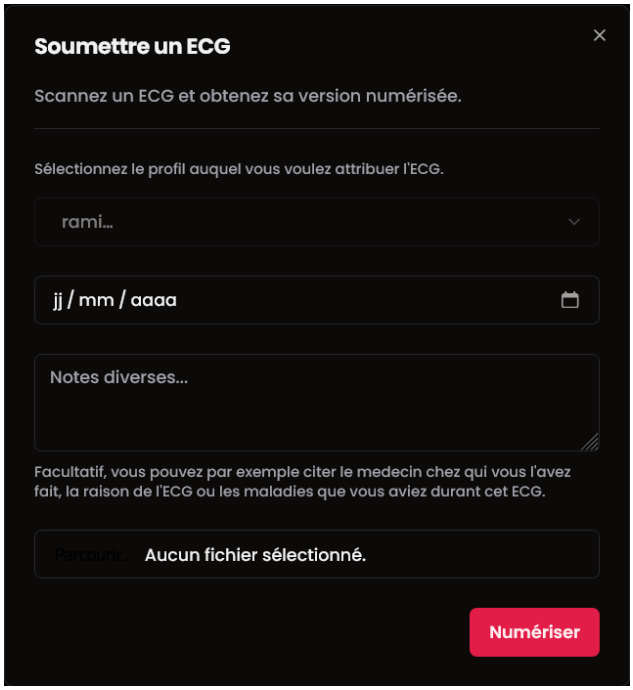


Figure 3.55: Soumission d'un ECG d'un profil particulier.

Pour plus d'informations, l'utilisateur peut cliquer sur l'ECG qu'il souhaite consulter et sera renvoyé vers la route `ecg/:ecgId` qui est une route dynamique. Une vue détaillée de l'ECG est ainsi présentée comme suit :

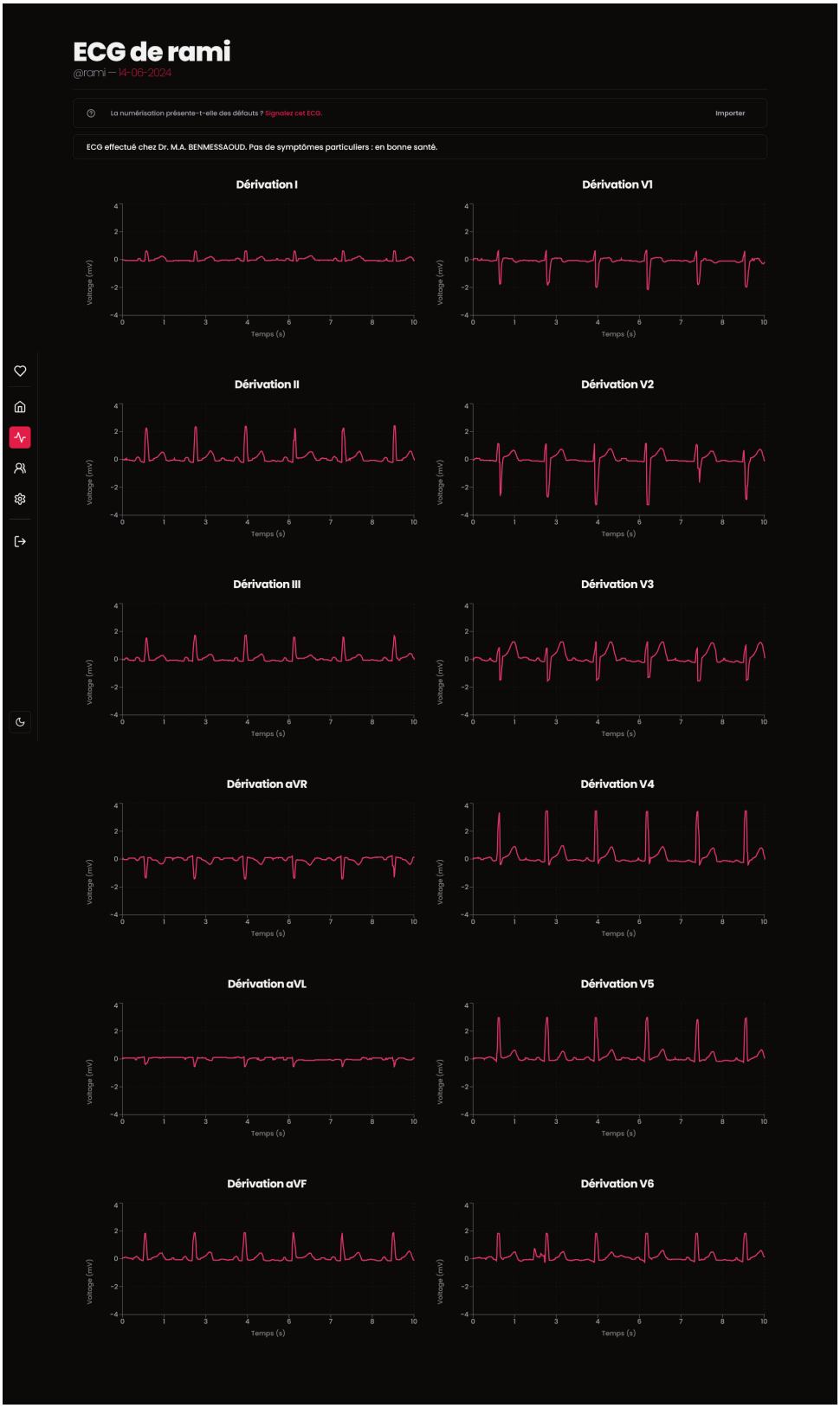


Figure 3.56: Interface de consultation d'un ECG.

Une page d'erreur sera affichée si l'utilisateur tente d'entrer un lien vers un ECG dont il n'est pas le propriétaire, car l'identifiant de l'ECG est comparé avec le token d'accès décrypté de la requête.

En cliquant ou en survolant un point de l'ECG, les coordonnées de ce point sont affichés. Nous pouvons remarquer la présence du bouton intitulé "importer" en haut à droite de l'interface qui, lorsque cliqué, déclenche le téléchargement du fichier `ecg.json` et le scan ayant servi à sa numérisation (`scan.png`) dans un fichier archive. Le package `archiver` a été utilisé pour effectuer la compression des données à retourner, une compression `Zlib` [24] (RFC 1950) de niveau 9 a été utilisée.

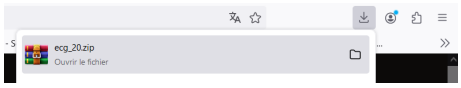


Figure 3.57: Import d'un ECG.

Finalement, un ECG peut être supprimé directement à partir de la liste des ECGs :

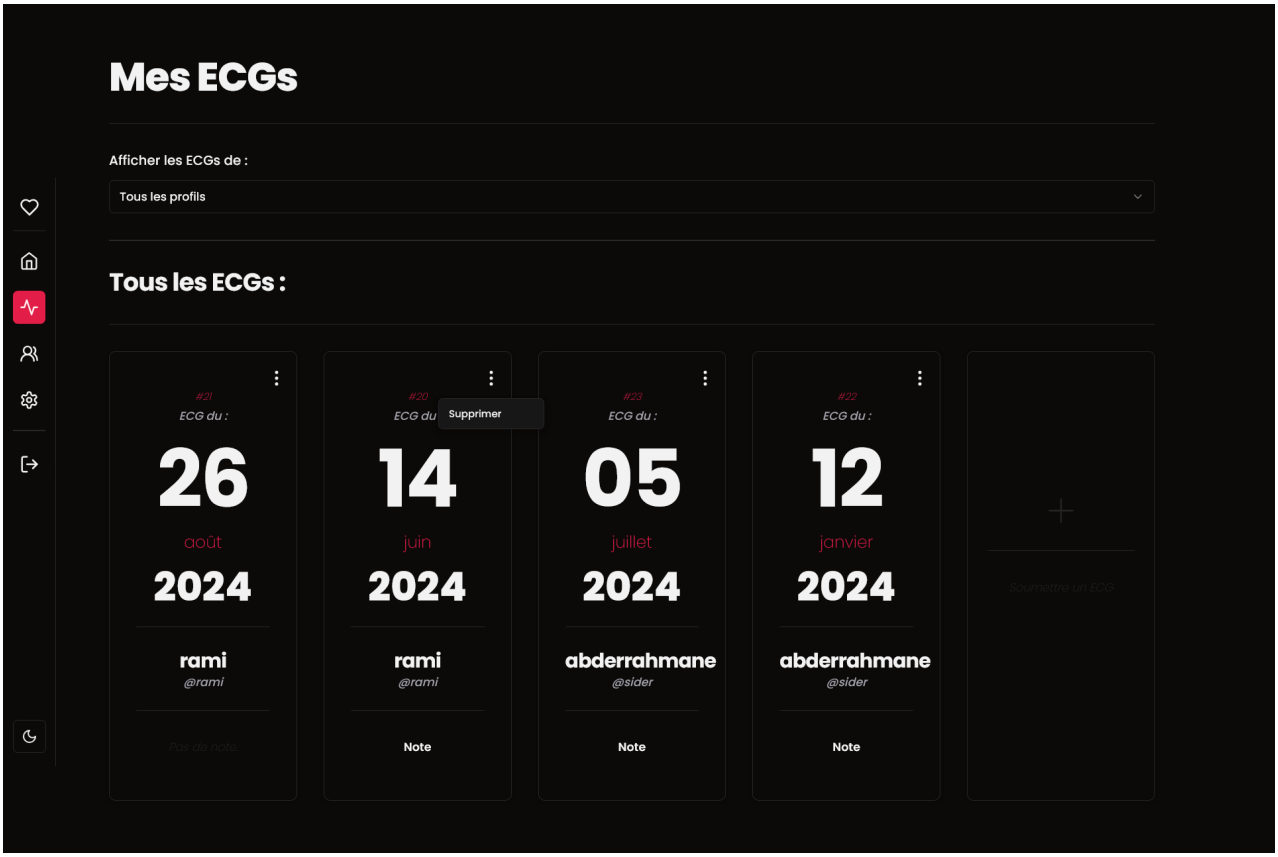


Figure 3.58: Bouton de suppression d'un ECG.

La suppression d'un compte utilisateur ou d'un profil entraine la suppression des répertoires contenant les données ECGs du compte ou profil associé.

3.9.5 Sprint 05

Ce cinquième Sprint traitera de la fonctionnalité de signalement de dysfonctionnements ou d'erreurs dans l'application.

Product Backlog

User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Durée estimée	Date de début	Date de fin
US_REPORT	Utilisateur	Signalisation d'un dysfonctionnement	Signalisation d'un scan erroné	02 jours	28/08/2024	29/08/2024
			Signalisation d'une numérisation échouée	02 jours	30/08/2024	31/09/2024
			Signalisation d'une numérisation erronée	04 jours	01/09/2024	04/09/2024

Table 3.26: Product Backlog du Sprint 05.

Diagramme de cas d'utilisation

Un nouveau cas d'utilisation est défini :

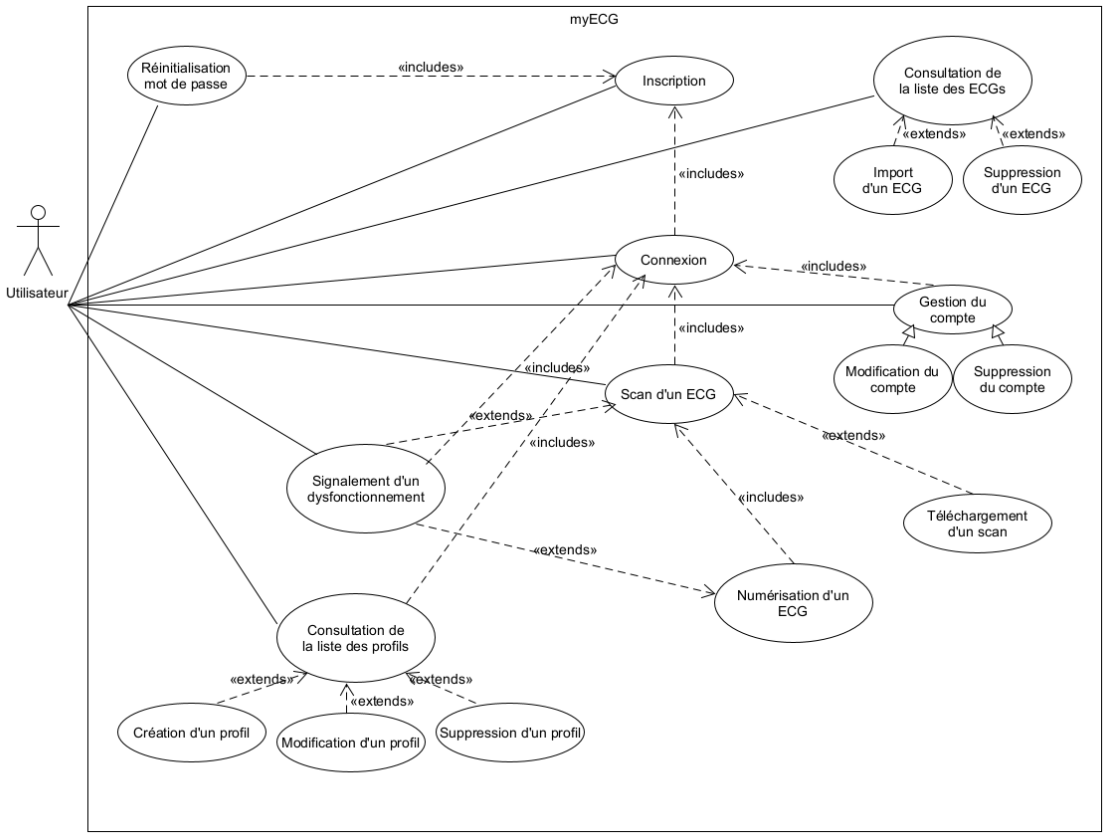


Figure 3.59: Diagramme de cas d'utilisation du Sprint 05.

Diagramme de séquence

Ce cas d'utilisation est représenté en trois scénarios :

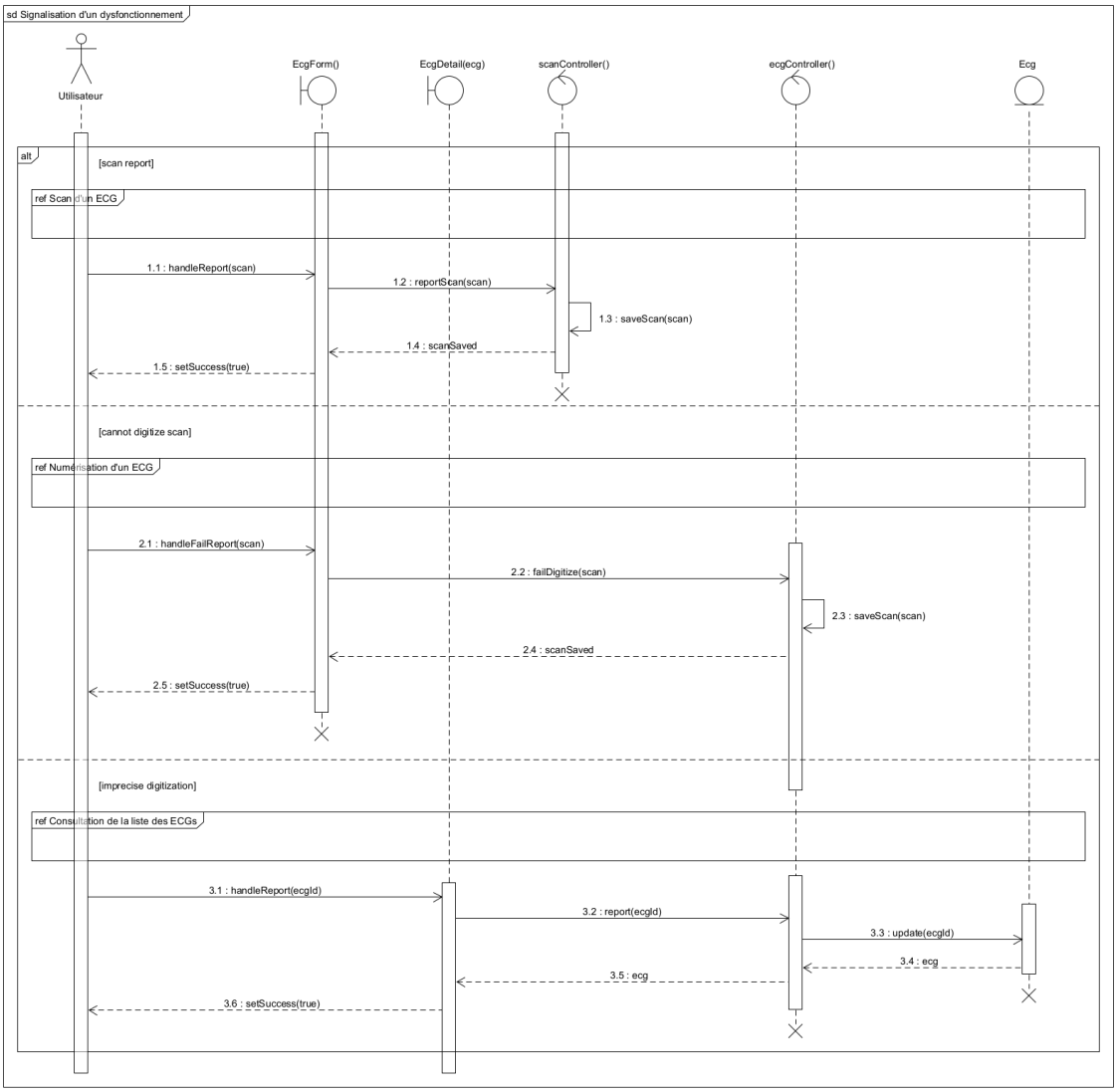


Figure 3.60: Diagramme de séquence du cas d'utilisation "Signalement d'un dysfonctionnement".

Diagramme de classe

Deux méthodes à la classe "User" ont été ajoutées, ainsi qu'un attribut à la classe "ECG" :

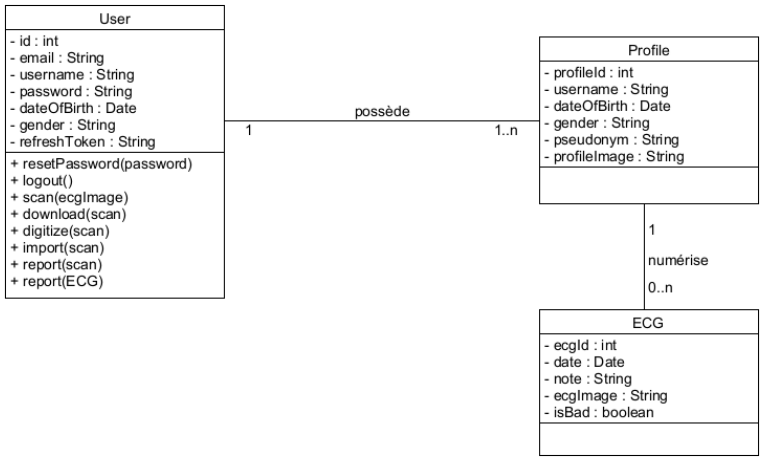


Figure 3.61: Diagramme de classe du Sprint 05.

Modèle relationnel

La table "Ecg" a été altérée :

User(id, email, username, password, dateOfBirth, gender, refreshToken)
Profile(#userId, profileId, username, pseudonym, dateOfBirth, gender, profileImage)
Ecg(#profileId, ecgId, date, note, ecgImage, isBad)

Contraintes : **Contraintes** :

- L'attribut `id` est la clé primaire de la table "User" ;
- Les attributs `username` et `email` sont uniques dans la table "User" ;
- L'attribut `profileId` est la clé primaire de la table "Profile" ;
- L'attribut `userId` fait référence à l'attribut `id` de la table "User" ;
- Le couple (`userId`, `username`) est unique dans la table "Profile" ;
- L'attribut `ecgId` est la clé primaire de la table "Ecg" ;
- L'attribut `profileId` de la table "Ecg" fait référence à l'attribut `profileId` de la table "Profile".

Implémentation

Nous allons visualiser les cas d'utilisation de la fonctionnalité de signalement de dysfonctionnements en partant de cette image :

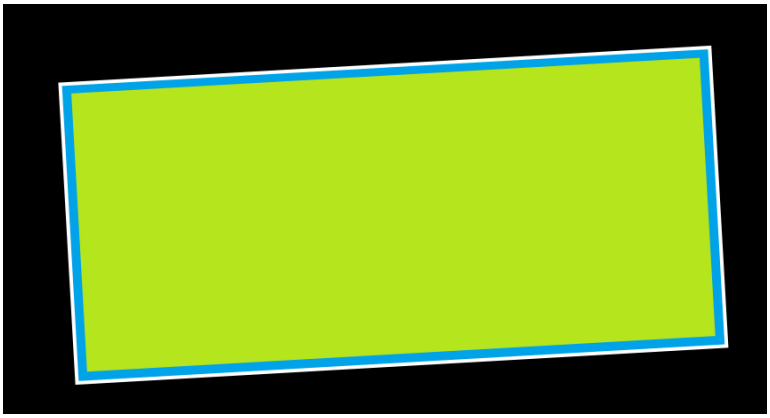


Figure 3.62: Image de test de l'application.

Cette image ne représente évidemment pas un ECG. Nous allons maintenant la scanner et voir le résultat obtenu :

Figure 3.63: Scan sur l'image de test.

Nous allons maintenant supposer que l'utilisateur estime que le scan est erroné. Dans ce cas, il n'a qu'à appuyer sur le bouton du signalement et sera notifié de sa soumission du signalement :

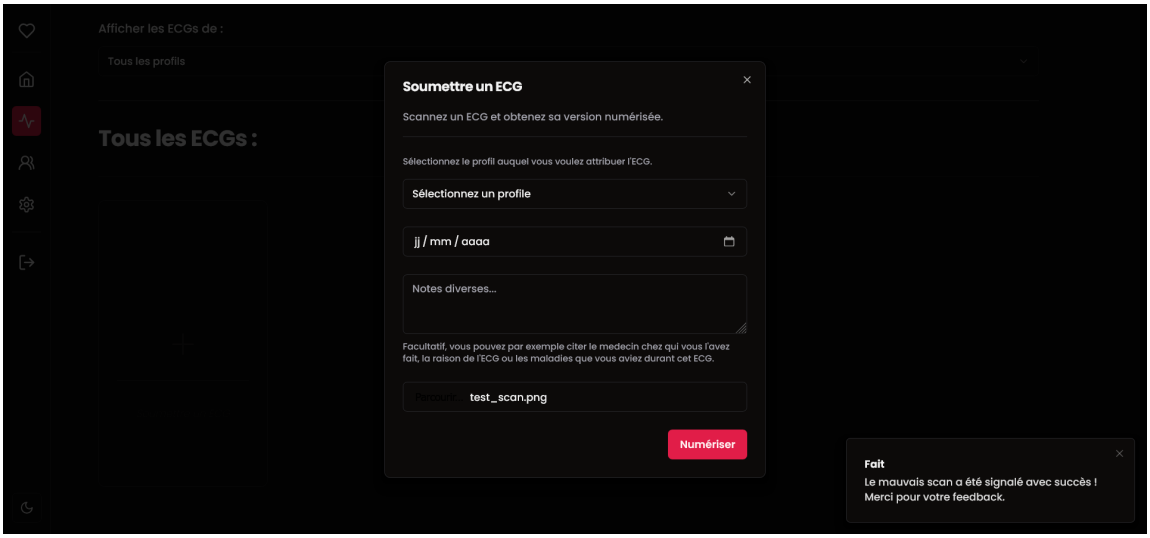


Figure 3.64: Notification de la réception d'un signalement de scan erroné.

Les scans signalés comme étant erronés sont directement envoyés à partir du Frontend vers le Backend et sont simplement enregistrés dans le dossier `images/bad_scans` avec un nom de fichier unique qui est la date et l'heure de la soumission du signalement par l'utilisateur.

Si une numérisation échoue, un message d'erreur est affiché en bas du formulaire de soumission d'un ECG :

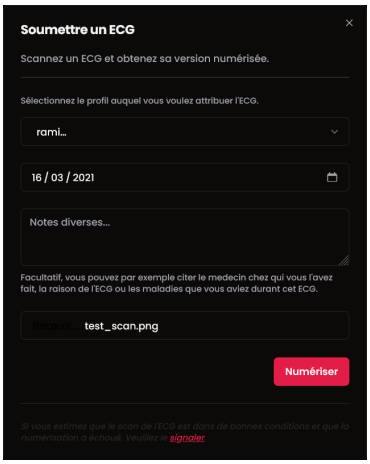


Figure 3.65: Message retourné lors de l'échec d'une numérisation.

Ce message possède une partie cliquable qui, lorsque en interaction, enregistre le scan ayant causé l'échec de la numérisation dans un dossier `bad_digitization` qui lui aussi est dans le répertoire `images`.

Enfin, un ECG numérisé peut-être déclaré comme étant erroné aussi, si l'utilisateur estime qu'il n'est pas assez fidèle à l'ECG réel. Dans ce cas, seulement le champ `isBad` de l'ECG est mis à vrai. Une annulation de ce signalement remet ce champ à faux.

Si un ECG est considéré comme étant mal numérisé, il sera affiché en rouge sur la liste des ECGs et labellisé comme étant "erroné" :

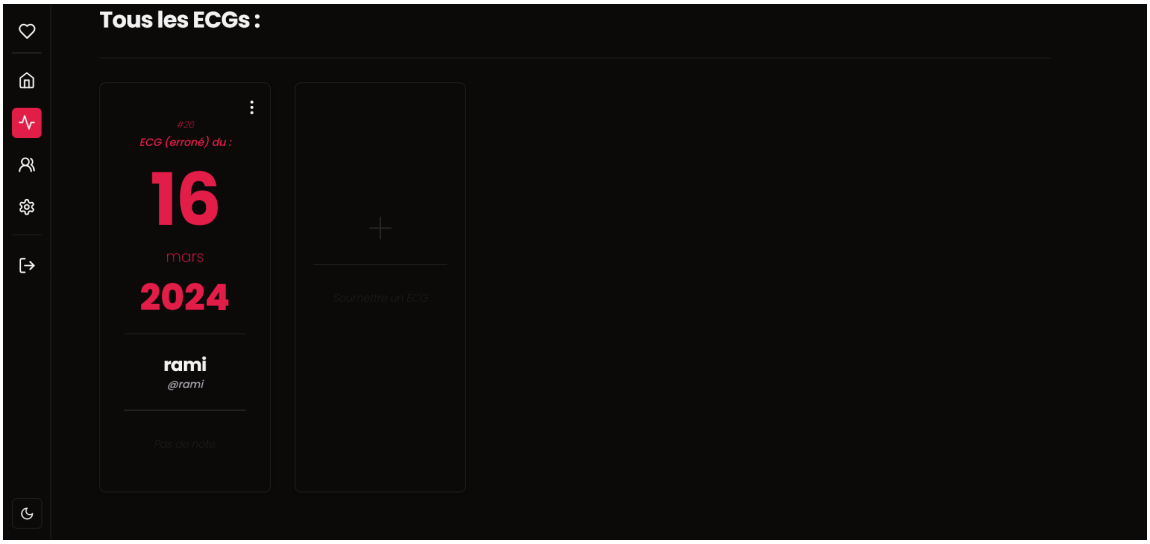


Figure 3.66: Affichage d'un ECG dont la numérisation est erronée.

Et lorsque visualisé, son statut sera aussi affiché en rouge :

Figure 3.67: Statut d'un ECG dont la numérisation est erronée.

3.9.6 Sprint 06

Ce sixième Sprint traitera des fonctionnalités liées au partage et l'acquisition des données d'ECGs.

Product Backlog

User Story	Acteur	Fonctionnalité principale	Fonctionnalités déduites	Durée estimée	Date de début	Date de fin
US_SHARE	Utilisateur	Partipication à l'enrichissement de la base de données	Aucune	02 jours	05/09/2024	07/09/2024
US_DATASET	Entité de confiance	Acquisition des ECGs partagés	Aucune	03 jours	06/09/2024	08/09/2024

Table 3.27: Product Backlog du Sprint 06.

Diagramme de cas d'utilisation

L'acteur externe "Entité de confiance" a été ajouté, ainsi que deux cas d'utilisation :

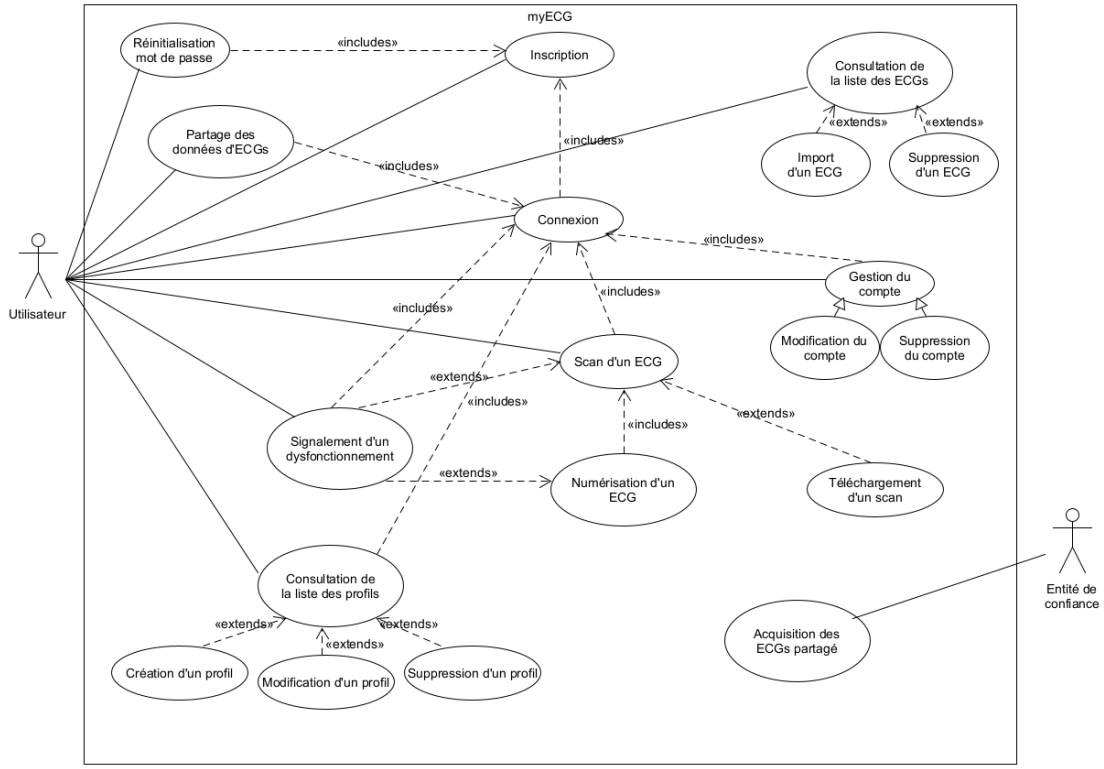


Figure 3.68: Diagramme de cas d'utilisation du Sprint 06.

Diagrammes de séquence

Deux nouveaux diagrammes de séquence :

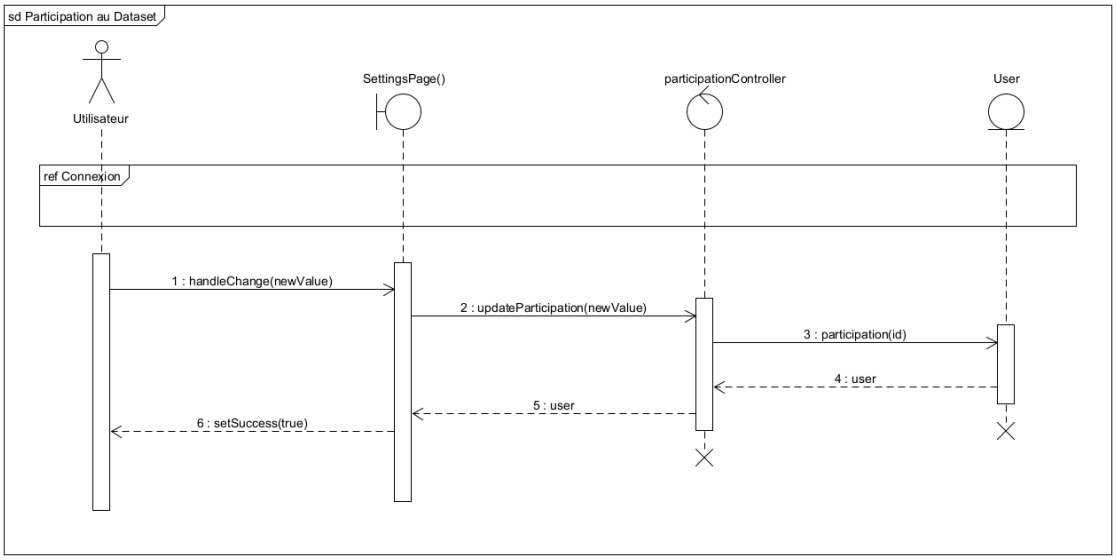


Figure 3.69: Diagramme de séquence du cas d'utilisation "Partage des données ECGs".

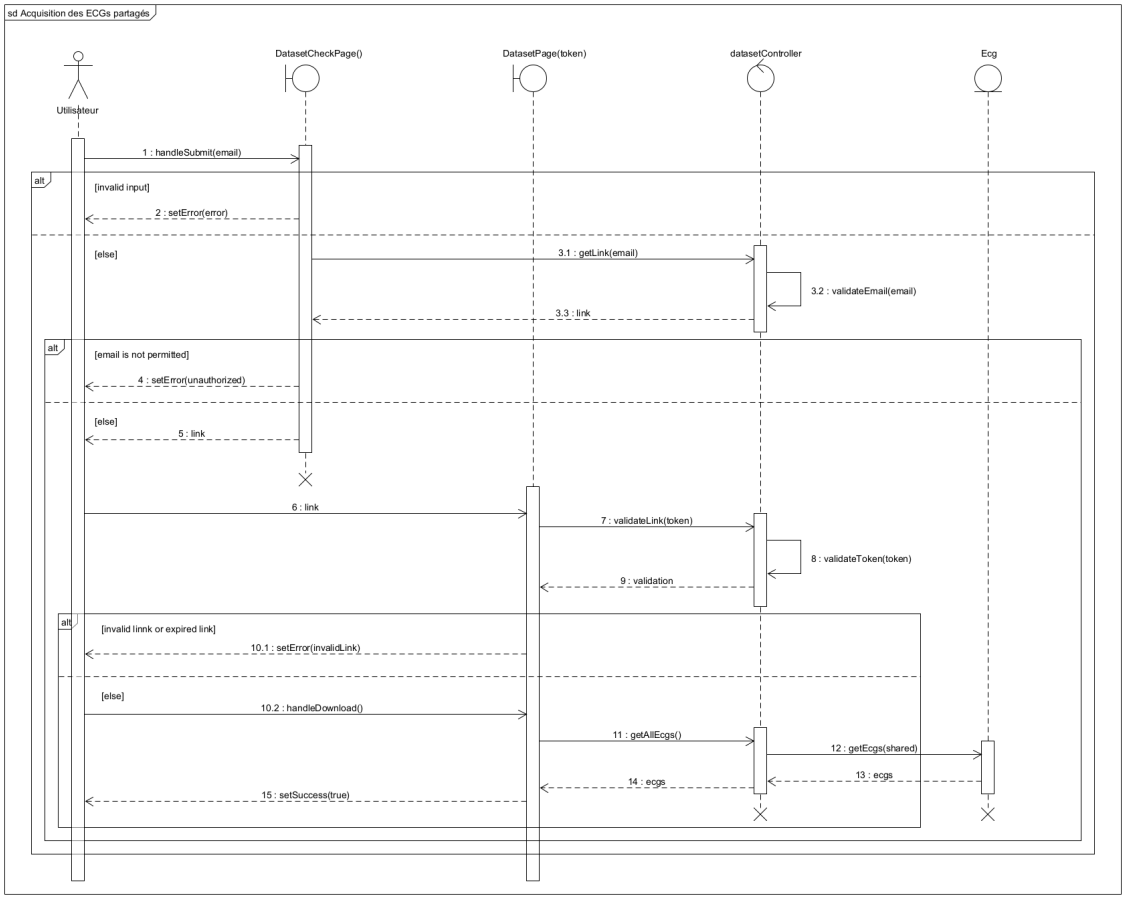


Figure 3.70: Diagramme de séquence du cas d'utilisation "Acquisition des ECGs partagés".

Diagramme de classe

Un nouvel attribut a été ajouté à la classe "User" :

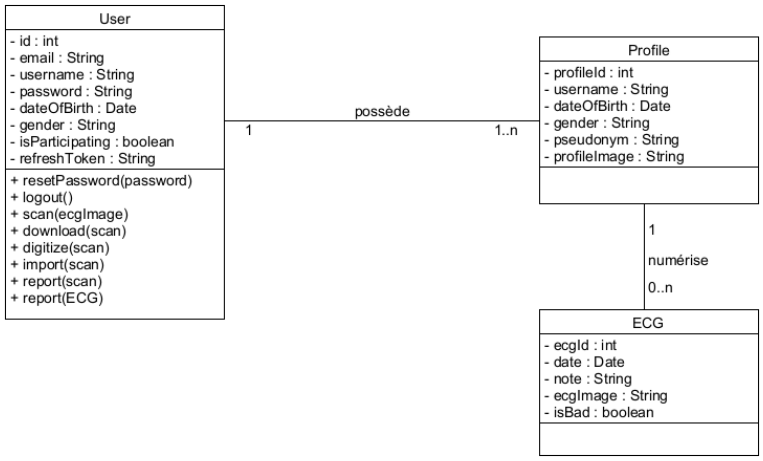


Figure 3.71: Diagramme de classe du Sprint 06.

Modèle relationnel

Suivant le diagramme de classe, le champ `isParticipating` a aussi été ajouté aussi :
User(id, email, username, password, dateOfBirth, gender, isParticipating, refreshToken)
Profile(#userId, profileId, username, pseudonym, dateOfBirth, gender, profileImage)
Ecg(#profileId, ecgId, date, note, ecgImage, isBad)

Contraintes :

- L'attribut `id` est la clé primaire de la table "User" ;
- Les attributs `username` et `email` sont uniques dans la table "User" ;
- L'attribut `profileId` est la clé primaire de la table "Profile" ;
- L'attribut `userId` fait référence à l'attribut `id` de la table "User" ;
- Le couple (`userId`, `username`) est unique dans la table "Profile" ;
- L'attribut `ecgId` est la clé primaire de la table "Ecg" ;
- L'attribut `profileId` de la table "Ecg" fait référence à l'attribut `profileId` de la table "Profile".

Implémentation

La participation au Dataset ne dépend que d'un champ de la table "User" qui est `isParticipating`. Sa modification se fait dans la page des paramètres et reflète simplement une opération `update` sur la base de données :

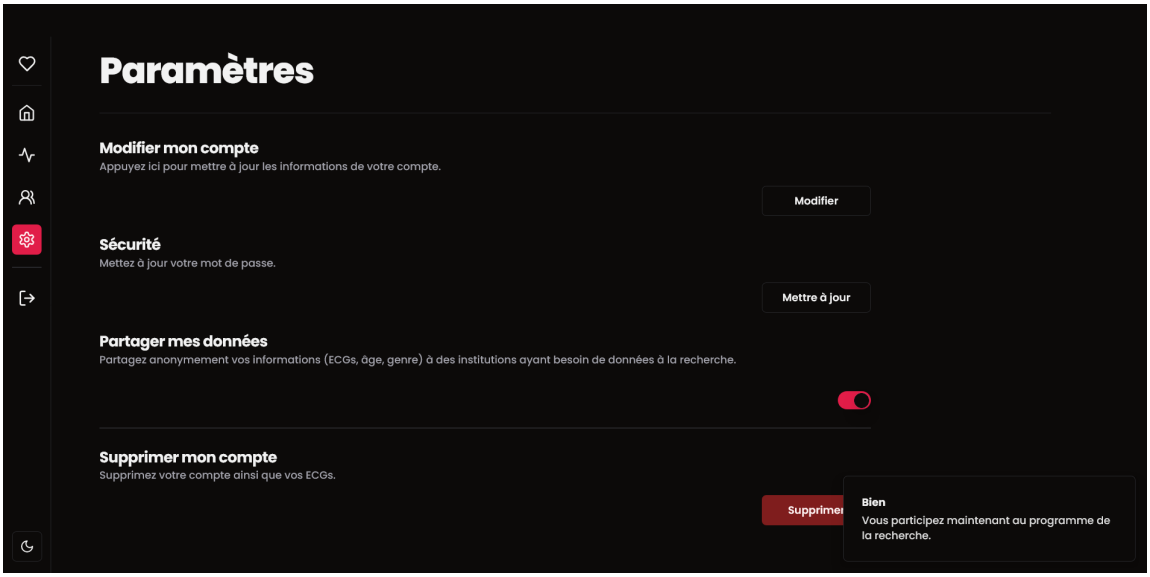


Figure 3.72: Paramètre de la participation à l'enrichissement de la base de données.

Le statut indiqué dans le tableau de bord change aussi en fonction de si l'utilisateur participe ou non au programme :

Figure 3.73: Statut de la participation.

L'utilisateur peut à tout moment cesser sa participation :

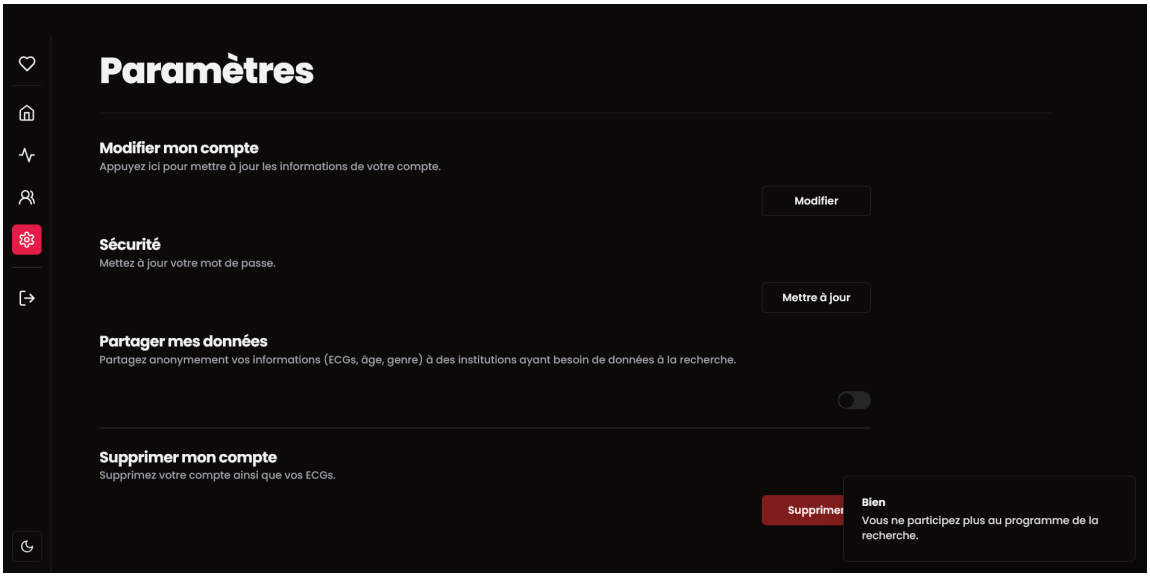


Figure 3.74: Arrêt de la participation.

Quant à l'entité de confiance, elle doit se rendre sur la Landing Page de l'obtention du Dataset et entrer son adresse e-mail.

La liste des adresses e-mail permises à l'exploitation du Dataset est contenue dans un tableau JSON dans les variables d'environnement sous le nom de :

```
ALLOWED_ENTITIES = [...]
```

Après la saisie, l'adresse e-mail est transmise au Backend et ce dernier vérifie si elle appartient à la liste des entités légitimes à l'exploitation du Dataset, si ce n'est pas le cas, un message d'erreur est renvoyé :

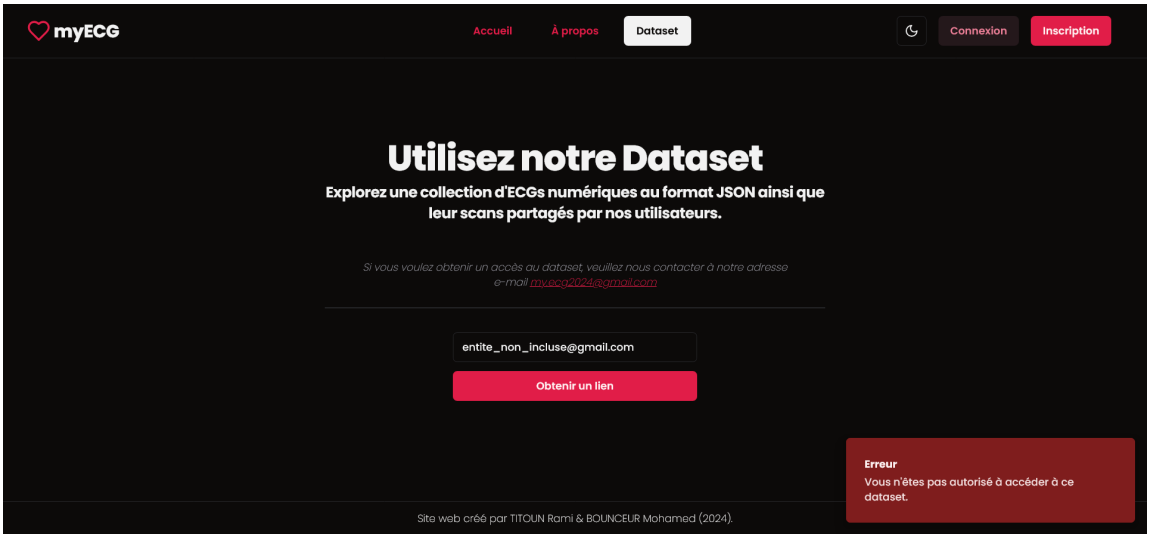


Figure 3.75: Refus de l'obtention du Dataset.

Sinon, un e-mail est envoyé contenant un lien vers le Dataset valable 15 minutes. De même que pour la réinitialisation du mot de passe, une clé de 128 caractères `DATASET_TOKEN_SECRET` a aussi été générée pour la création des tokens JWT composant les liens.

Lorsque l'entité de confiance consulte le lien `dataset/:token`, le token est renvoyé au Back-end pour sa validation avant de lancer le téléchargement de l'intégralité des ECGs partagés (`scan.png` et `ecg.json`) dans une archive `ecgs.zip`.

Le module de la numérisation a été modifié de sorte à inclure deux nouveaux clés à l'objet `ecg.json` qui sont `gender` et `age` afin de fournir plus d'informations à analyser pour les recherches.

Conclusion

Ce chapitre a présenté le processus de conception et de développement d'une application de numérisation d'électrocardiogrammes, depuis la phase d'analyse des besoins jusqu'à la mise en œuvre technique. Grâce à l'utilisation d'outils de modélisation tels que UML, nous avons pu formaliser les exigences fonctionnelles et non fonctionnelles de l'application. La méthode agile, et plus particulièrement Scrum, nous a permis d'organiser le développement de manière itérative et incrémentale, en nous adaptant aux contraintes et aux évolutions du projet.

Les choix technologiques effectués, notamment en termes de langage de programmation et de framework, ont été déterminants pour la réussite du projet. En combinant ces outils et méthodes, nous avons pu concevoir une application intuitive, robuste, évolutive et facile à maintenir.

Chapitre 4

Scan et numérisation d'ECGs

Introduction

Dans ce chapitre, nous allons appliquer les concepts du traitement d'image à un cas concret qui est la numérisation d'ECGs.

Les ECGs, traditionnellement enregistrés sur papier, contiennent des informations cruciales pour le diagnostic de maladies cardiaques. Leur conversion en format numérique permet une analyse plus approfondie et une meilleure conservation des données.

Nous allons donc explorer les différentes étapes de ce processus, depuis l'acquisition de l'image numérisée jusqu'à l'obtention d'un signal ECG numérique exploitable. Nous commencerons par étudier les caractéristiques spécifiques des ECG papier, puis nous présenterons les méthodes de traitement d'images adaptées à leur scan et numérisation.

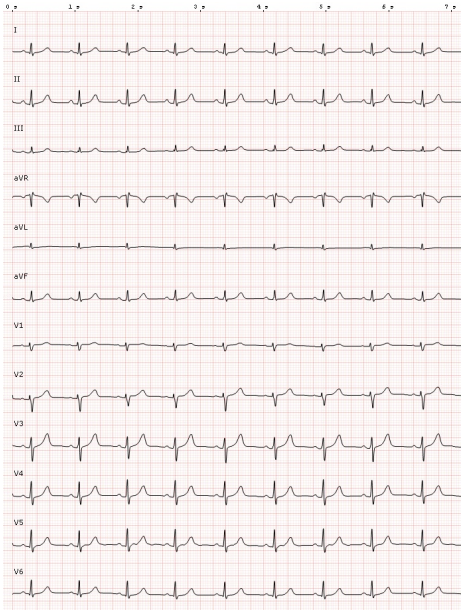
4.1 Caractéristiques d'un ECG

Il est crucial d'identifier les caractéristiques visuelles d'un ECG qui peuvent être déduite d'une image. L'étude de ces caractéristiques vont s'avérer être les lignes directrices de notre approche.

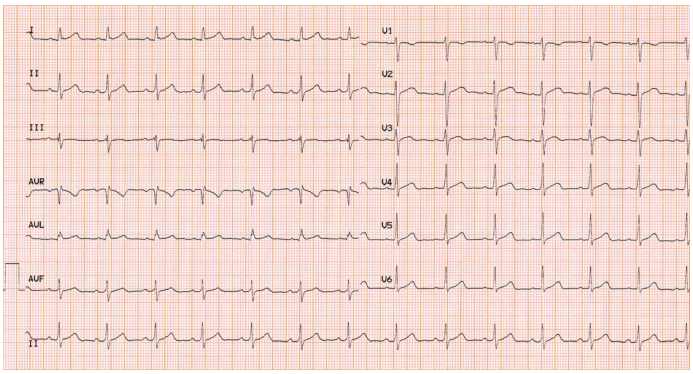
L'ECG est présenté sous la forme d'un papier millimétré rectangulaire comportant un ensemble de 12 signaux de dérivations suivant un ordre de haut en bas ensuite de gauche à droite (dans le cas d'une disposition à plusieurs colonnes) qui est : $D1$ (ou I), $D2$ (ou II), $D3$ (ou III), aVR, aVL, aVF et enfin de la V_1 à la V_6.

4.1.1 Disposition des signaux de dérivations

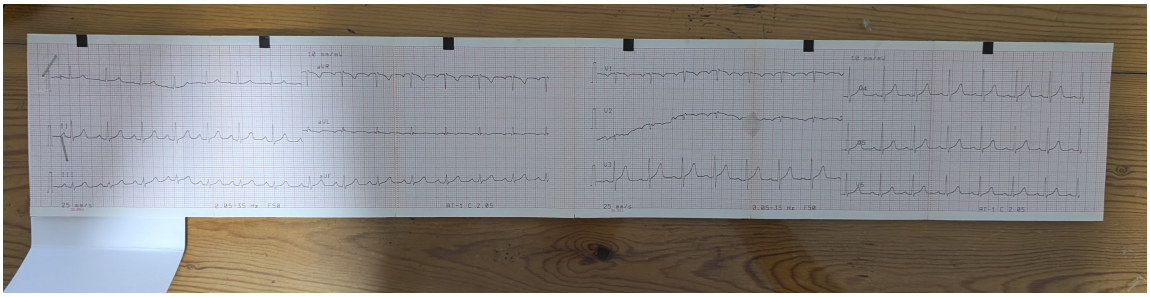
Les signaux de dérivation sont disposés soit en une colonne et douze lignes, ou en deux colonnes et six lignes ou encore en quatre colonnes et trois lignes :



(a) Disposition à une colonne [36].



(b) Disposition à deux colonnes [37].



(c) Disposition à quatre colonnes.

Figure 4.1: Exemples de disposition des dérivations dans un ECG.

Toutes les dérivations sont enregistrées sur la même période de temps t et font donc toutes la même longueur.

4.1.2 Encre et papier

L'encre qui sert à enregistrer les signaux d'un ECG est généralement noire ou, plus rarement, bleu foncé. Un détail commun à tous les ECG est que la grille est d'une couleur plus claire que l'encre.

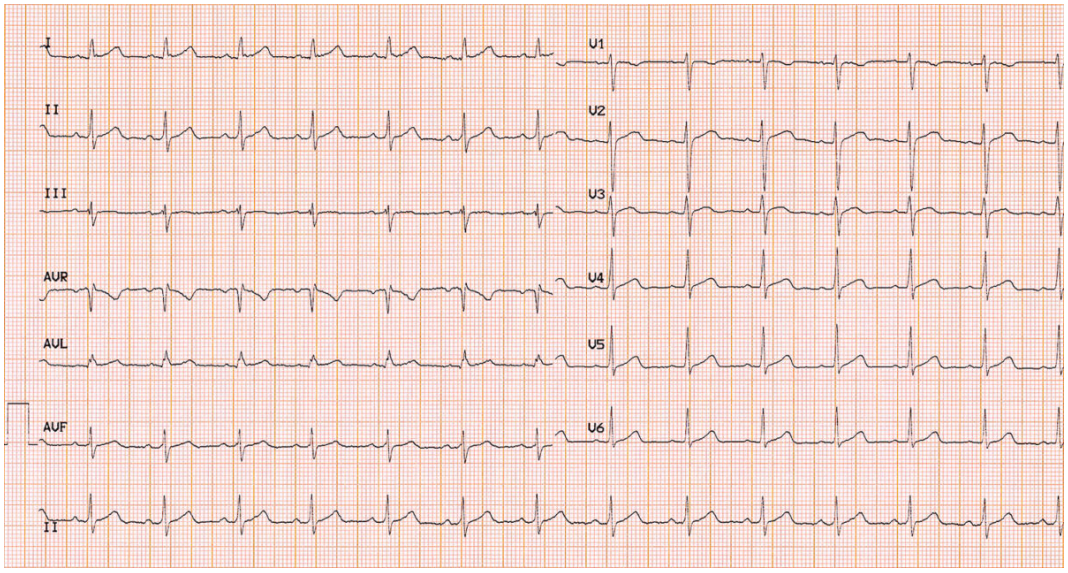


Figure 4.2: Encre bleu foncé d'un ECG. [37].

La couleur de l'encre de l'ECG n'est pas si importante car, en niveaux de gris, elle reste quand-même similaire au noir ou à un gris très sombre et se distingue du reste des couleurs.

Parfois, l'encre paraît plus claire ou plus fine à certains endroits (par exemple aux environs des gros mouvements) comme dans la descente du dernier pic de ce signal :

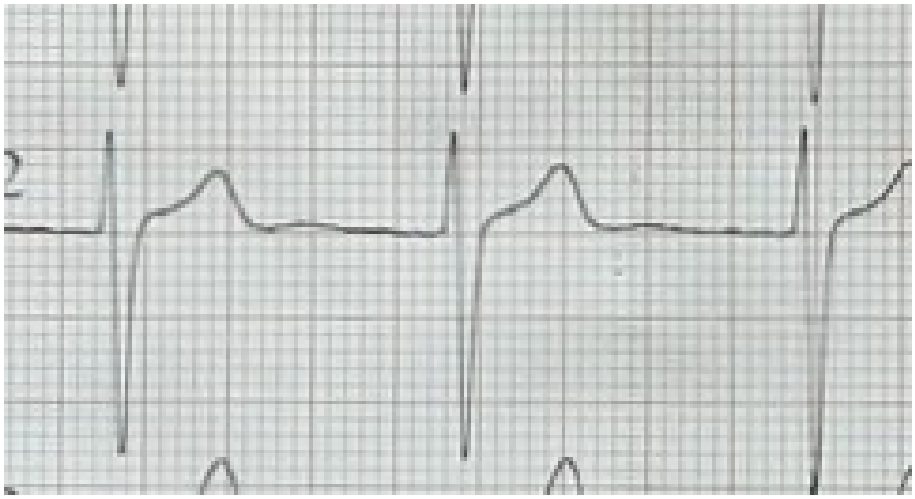


Figure 4.3: Inconsistance de l'encre dans un ECG.

Quant au papier, il est blanc et recouvert d'une grille de couleur qui s'apparente au rouge ou au marron, voire à l'orange ; ce qui se traduit par une teinte chaude. Il est possible de retrouver des grilles d'une couleur qui s'apparente à un gris clair.

4.1.3 Étalonnage

L'étalonnage usuel du signal électrique imprimé sur un ECG est $1mV = 10mm$ et la vitesse de déroulement du papier millimétré de $25mm/s$ [38]. On en déduit donc que $1cm = 0,4s$.

L'étalonnage est obligatoirement affiché sur un ECG imprimé ou numérique. Il est généralement symbolisé par un rectangle de $10mm$ de haut sur $5mm$ de large ou, plus rarement, $20mm$ de haut pour $10mm$ de large.

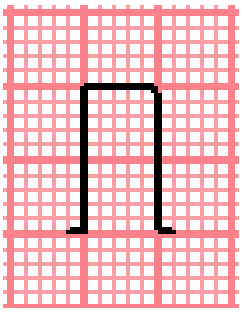


Figure 4.4: Schéma du rectangle d'étalonnage d'un ECG.

Le rectangle d'étalonnage est crucial à l'étape de la numérisation et à l'interprétation d'un ECG, et c'est pour cela que tout ECG dispose d'au moins un rectangle d'étalonnage.

On peut y retrouver dans certains cas un rectangle d'étalonnage pour chaque chaque dérivation ou une ligne entière de dérivations.

4.2 Scan de l'ECG

Le scan d'un ECG consiste à délimiter la région d'intérêt (qui est l'ECG lui-même) à partir d'une image contenant un ECG et retourner ce dernier aplati, droit et occupant toute la résolution spatiale de l'image retournée.

Sachant que le papier ECG possède une forme rectangulaire, les étapes que nous avons entreprises pour le scanner à partir d'une image sont :

1. Redimensionner l'image à une petite échelle ;
2. Transformer en niveaux de gris ;
3. Flouter l'image obtenue ;
4. Détecter les bords ;
5. Détecter les contours et récupérer les plus grands ;
6. Appliquer une approximation par polygone pour chaque contour pris jusqu'à trouver un polygone à quatre coins ;

7. Calculer la matrice de transformation à partir des quatre points du polygone auparavant obtenu ;

8. Appliquer une transformation de perspective pour obtenir une vue de face de l'ECG.

Cette procédure nous permet de redresser l'ECG en corrigeant la distorsion due à l'angle de la prise de vue.

La validité de cette méthode ne se limite pas seulement aux images contenant un ECG mais plutôt à tout document possédant une forme semblable à un rectangle présent dans l'image d'entrée.

4.2.1 Implémentation de la méthode

En partant d'une photo d'un ECG prise par un smartphone dont la marque est Google et la référence est "Pixel 8", de qualité volontairement dégradée par compression, nous obtenons une image de format JPEG, que nous allons appeler I_{input}.

Cette image possède une résolution horizontale de 1536 pixels pour une résolution verticale de 1156 pixels :

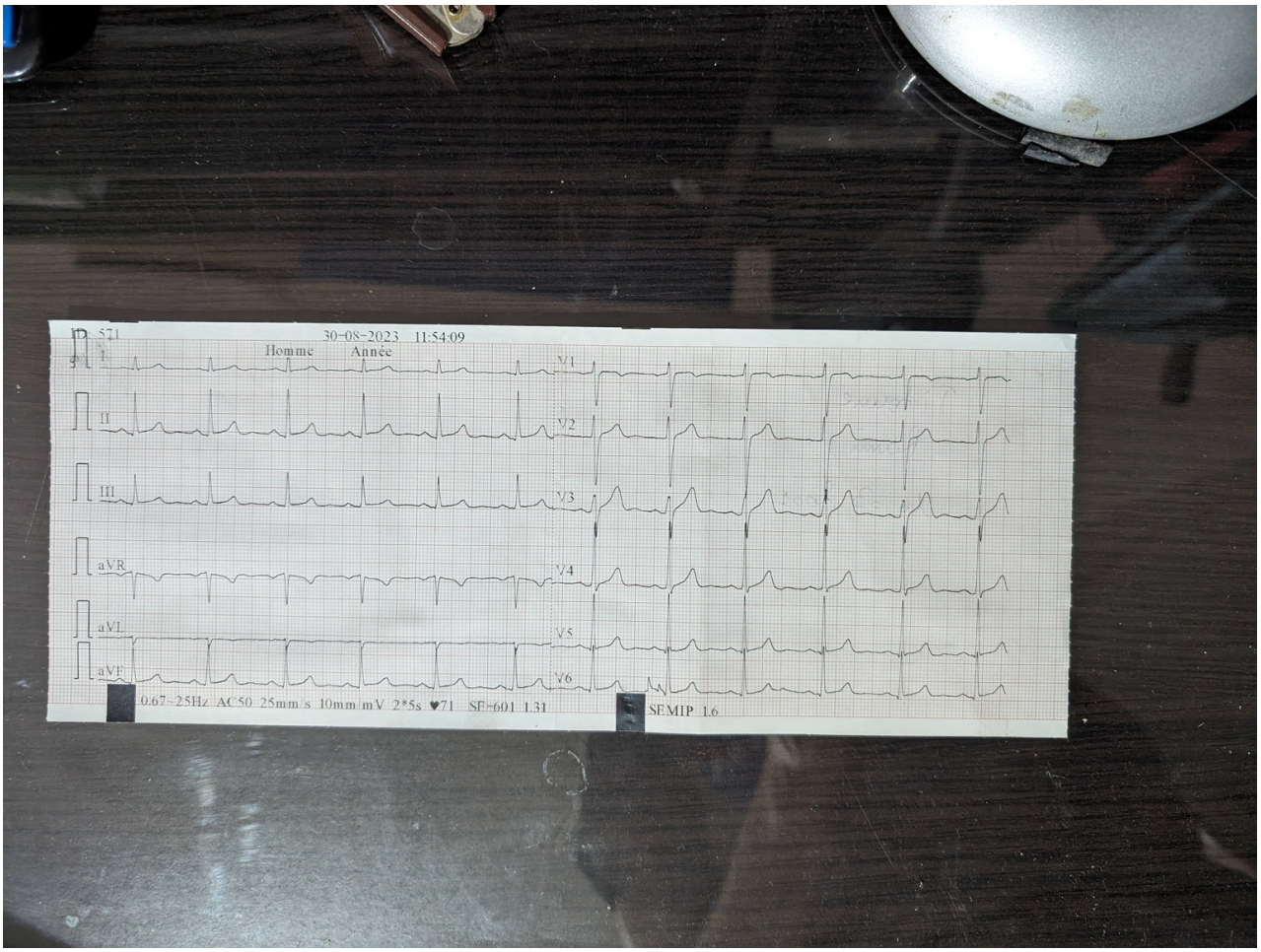


Figure 4.5: Photo d'un ECG prise par un Google Pixel 8.

L'intérêt de redimensionner l'image réside dans la volonté de perde de petits détails qui pourraient être considérés comme une région d'intérêt par l'algorithme, gardant ainsi que la forme générale des gros objets de l'image.

Le côté le plus long $L = 1536$px parmi la largeur et la hauteur sera rapporté à une taille fixe prédéfinie l qui vaut 320 pixels. Après redimensionnement par interpolation linéaire, nous obtenons une nouvelle image de dimensions 320×240 :

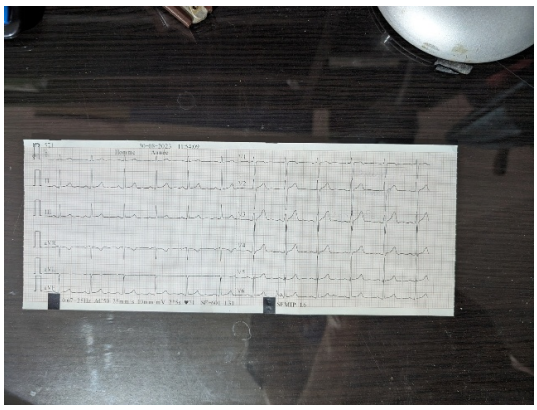


Figure 4.6: Image redimensionnée avant le scan.

Une transformation en niveaux de gris est ensuite appliquée pour continuer les opérations suivantes :

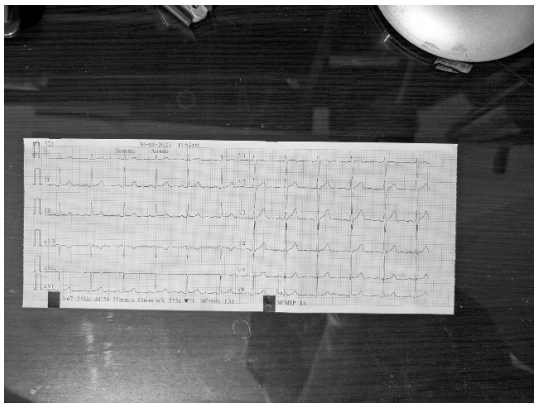


Figure 4.7: Transformation en niveaux de gris de la photo de l'ECG.

Le filtre passe-bas (ou floutage, appelé blur en Anglais) atténue les faibles différences de luminosité ainsi que les détails négligeables qui risquent d'être considérés comme région d'intérêt, permettant ainsi une plus grande homogénéité des couleurs et une élimination de candidats à la région d'intérêt. D'autre part, cela représente une légère optimisation par rapport au nombre d'objets à traiter.

Celui-ci se fait par un filtre Gaussien dont le noyaux est de taille 3×3 :

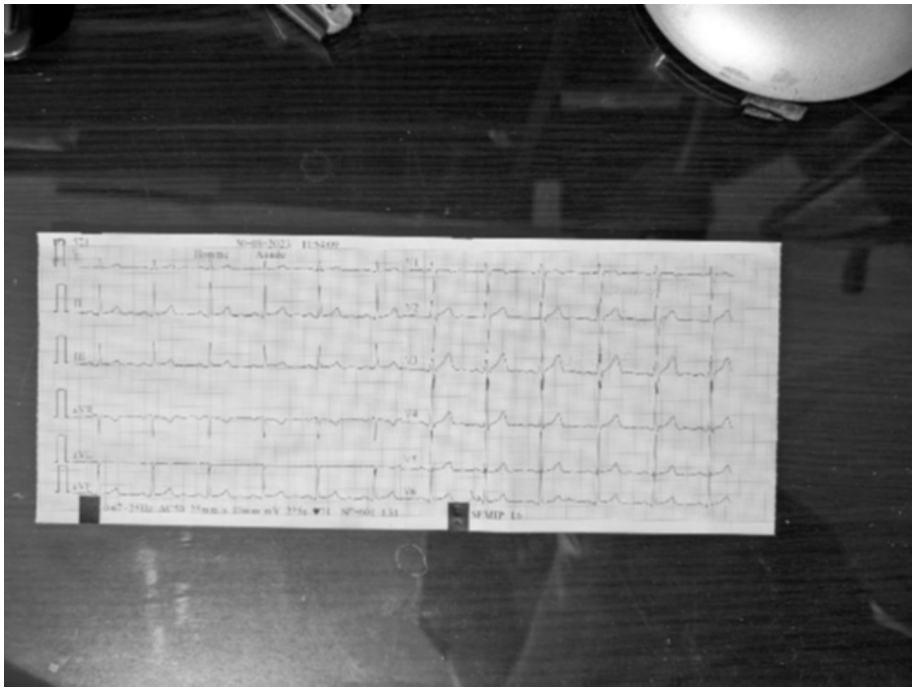


Figure 4.8: Filtre passe-bas sur l'image.

Il est fort probable que l'ECG pris en photo soit le plus gros objet dans l'image et y soit totalement inclus, il est tout aussi probable que l'ECG soit pris en photo sur une surface de couleurs assez sombres ou d'une couleur unie sans différences d'intensité de luminosité plus grandes que la différence entre la couleur des bords de l'ECG et la dite surface. Autrement dit, la couleur de l'ECG formera, sur l'image, un bord suffisamment apparent par rapport à la surface sur laquelle il est posé.

L'algorithme de J. Canny [35] (appelé filtre Canny) est utilisé avec pour seuils 64 et 210. Ces valeurs ont été trouvées par tâtonnement et se sont avérées être fiables pour une multitude d'exemples :

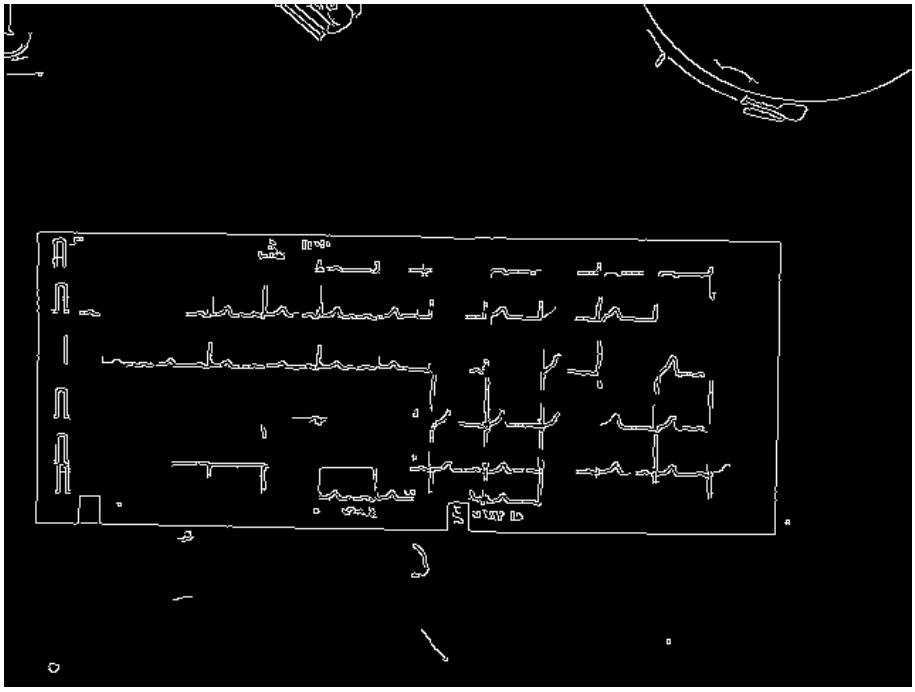


Figure 4.9: Filtre Canny appliqué sur l'image redimensionnée.

Un algorithme de détection des contours est appliqué afin de transformer les bords résultant du filtre Canny en structure de données dont les caractéristiques sont exploitables.

Les huit contours avec la plus grande aire sont sélectionnés et soumis à une approximation par polygone utilisant l'algorithme de Douglas-Peucker [39] avec comme paramètre $\epsilon = 0,03 \times$ longueur du contour.

Le premier polygone à quatre points obtenu est sélectionné afin de représenter les points extrêmes de l'ECG démontrant ainsi la délimitation de la région d'intérêt. Au cas où aucun polygone à quatre points n'est trouvé, le programme s'arrêtera et retournera une erreur.

Le premier contour aboutissant à un polygone à quatre points est entouré en vert :

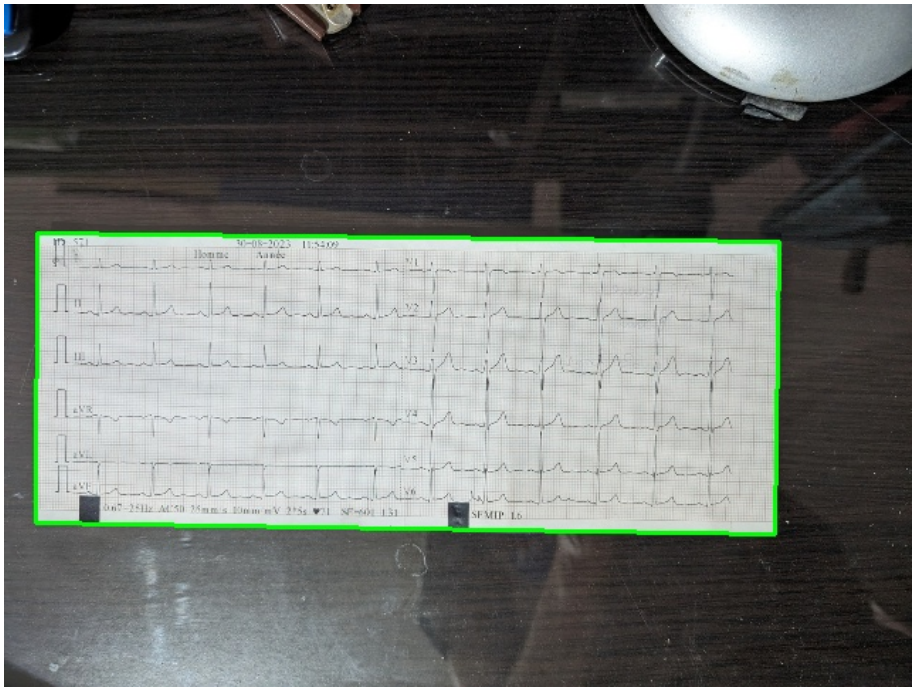


Figure 4.10: Détection de la région d'intérêt.

Les quatre points obtenus $A(x_A, y_A)$, $B(x_B, y_B)$, $C(x_C, y_C)$ et $D(x_D, y_D)$ nous permettront d'appliquer une transformation sur I_{input} par matrice de perspective :

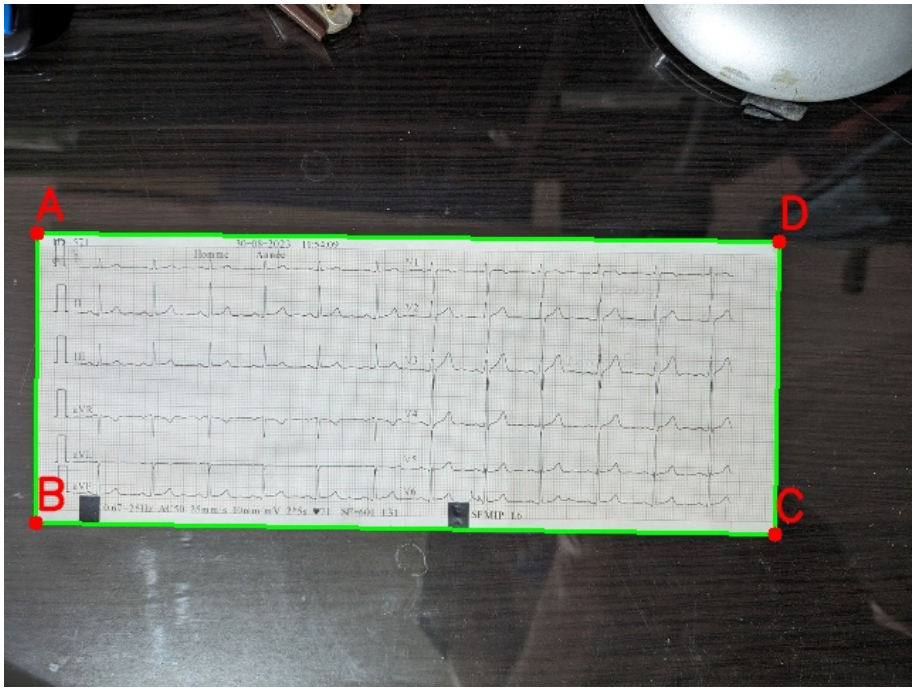


Figure 4.11: Points extrêmes ($ABCD$)du polygone obtenu.

Cependant, ils sont localisés selon l'image redimensionnée. Il est donc crucial de les translater vers l'image d'origine en multipliant leur coordonnées par le ratio $r = \frac{L}{l}$ avec L la dimension maximale de l'image d'origine, et $l = 320$. Obtenant ainsi de nouveaux points A', B', C' et D' :

- $A' = (r \cdot x_A, r \cdot y_A)$
- $B' = (r \cdot x_B, r \cdot y_B)$
- $C' = (r \cdot x_C, r \cdot y_C)$
- $D' = (r \cdot x_D, r \cdot y_D)$

Il est nécessaire de faire certains calculs afin d'identifier les coins de supérieur gauche (ou haut-gauche) HG, inférieur gauche (ou bas-gauche) BG, inférieur droit (ou bas-droit) BD et le coin supérieur droit (ou haut-droit) HD du polygone obtenu.

Pour visualiser les calculs, nous allons schématiser une région d'intérêt R ayant pour axe des abscisses x, axe des ordonnées y et point d'origine O :

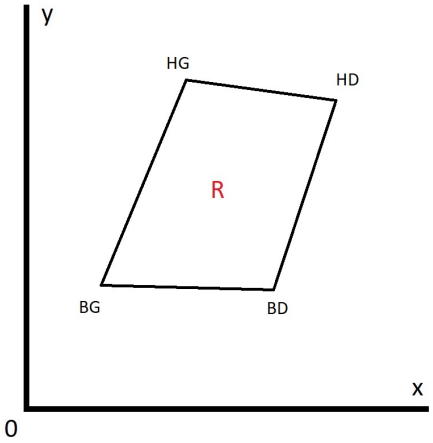


Figure 4.12: Schéma des coins d'une région d'intérêt à quatre coins.

Initialement, deux ensembles S et D sont déduits de chaque point A', B', C' et D' dont la formule est $S_i = x_i + y_i$ et $D_i = x_i - y_i$, représentant respectivement la somme et différence des coordonnées de chaque point. Nous remarquons ainsi que :

- Le point $HG(x_{\text{HG}}, y_{\text{HG}})$ est le point qui donne $\min(D)$;
- Le point $BG(x_{\text{BG}}, y_{\text{BG}})$ est le point qui donne $\min(S)$;
- Le point $BD(x_{\text{BD}}, y_{\text{BD}})$ est le point qui donne $\max(D)$;
- Le point $HD(x_{\text{HD}}, y_{\text{HD}})$ est le point qui donne $\max(S)$.

En effet, le coin supérieur droit HD est le point possédant la plus grande somme entre ses coordonnées (le plus loin de l'origine O), et le coin inférieur gauche BG est le point possédant la plus petite somme entre ses coordonnées (le plus proche de l'origine O).

Aussi, le coin supérieur gauche HG est le point dont l'ordonnée y est supérieure à son abscisse x (c'est-à-dire : $y > x$) et le coin inférieur droit BD est le point dont son abscisse x est supérieure à son ordonnée y (c'est-à-dire : $x > y$).

La largeur N de l'ECG scanné se déduit en prenant pour N le maximum entre B (représentant la distance entre BG et BD) et H (représentant la distance entre HG et HD). Tandis que la hauteur M, elle, se déduit en prenant le maximum entre G (représentant la distance entre HG et BG) et D (représentant la distance entre BD et HD). Voici une représentation mathématique de la déduction de N et M :

$$N = \max(B, H) \tag{4.1}$$

avec :

- $B = \sqrt{(x_{\text{BD}} - x_{\text{BG}})^2 + (y_{\text{BD}} - y_{\text{BG}})^2}$

- $H = \sqrt{(x_{\text{HD}} - x_{\text{HG}})^2 + (y_{\text{HD}} - y_{\text{HG}})^2}$

De la même manière, M vaut :

$$M = \max(G, D) \tag{4.2}$$

avec :

- $G = \sqrt{(x_{\text{HG}} - x_{\text{BG}})^2 + (y_{\text{HG}} - y_{\text{BG}})^2}$
- $D = \sqrt{(x_{\text{HD}} - x_{\text{BD}})^2 + (y_{\text{HD}} - y_{\text{BD}})^2}$

Enfin, il suffit de calculer la matrice de transformation de perspective avec comme source $s = (BG, BD, HD, HG)$ et destination $d = ((0, 0), (N, 0), (N, M), (0, M))$. Cette transformation appliquée à I_{input} retourne une nouvelle image I_{scan} :

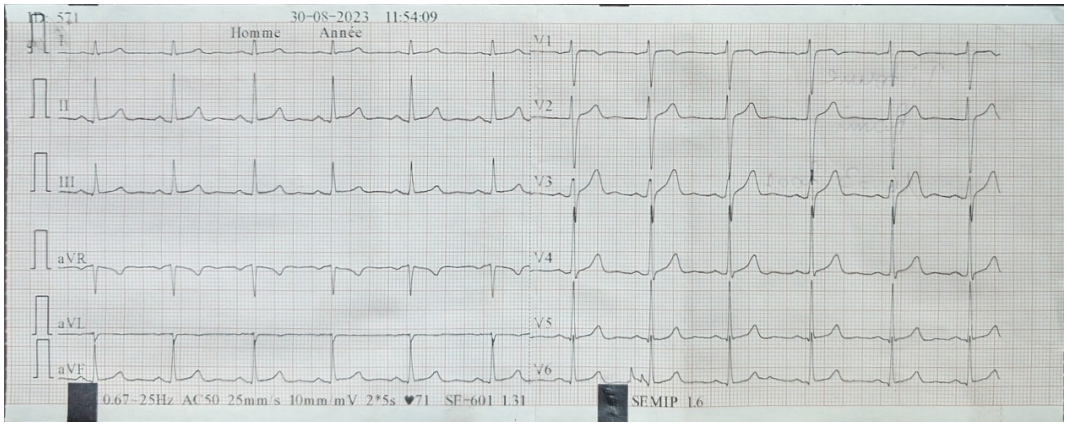


Figure 4.13: Scan de l'ECG contenu dans l'image I_{input}.

4.2.2 Fiabilité de la fonctionnalité de scan

Même dans un cas de luminosité très irrégulière et un angle extrême (dans le cadre d'une utilisation normale de la fonctionnalité de scan), l'algorithme reste fiable.

Il est montré ci-dessous un test de l'algorithme d'un cas supposé difficile avec une table en verre qui reflète la lumière et une ombre couvrant partiellement l'ECG :

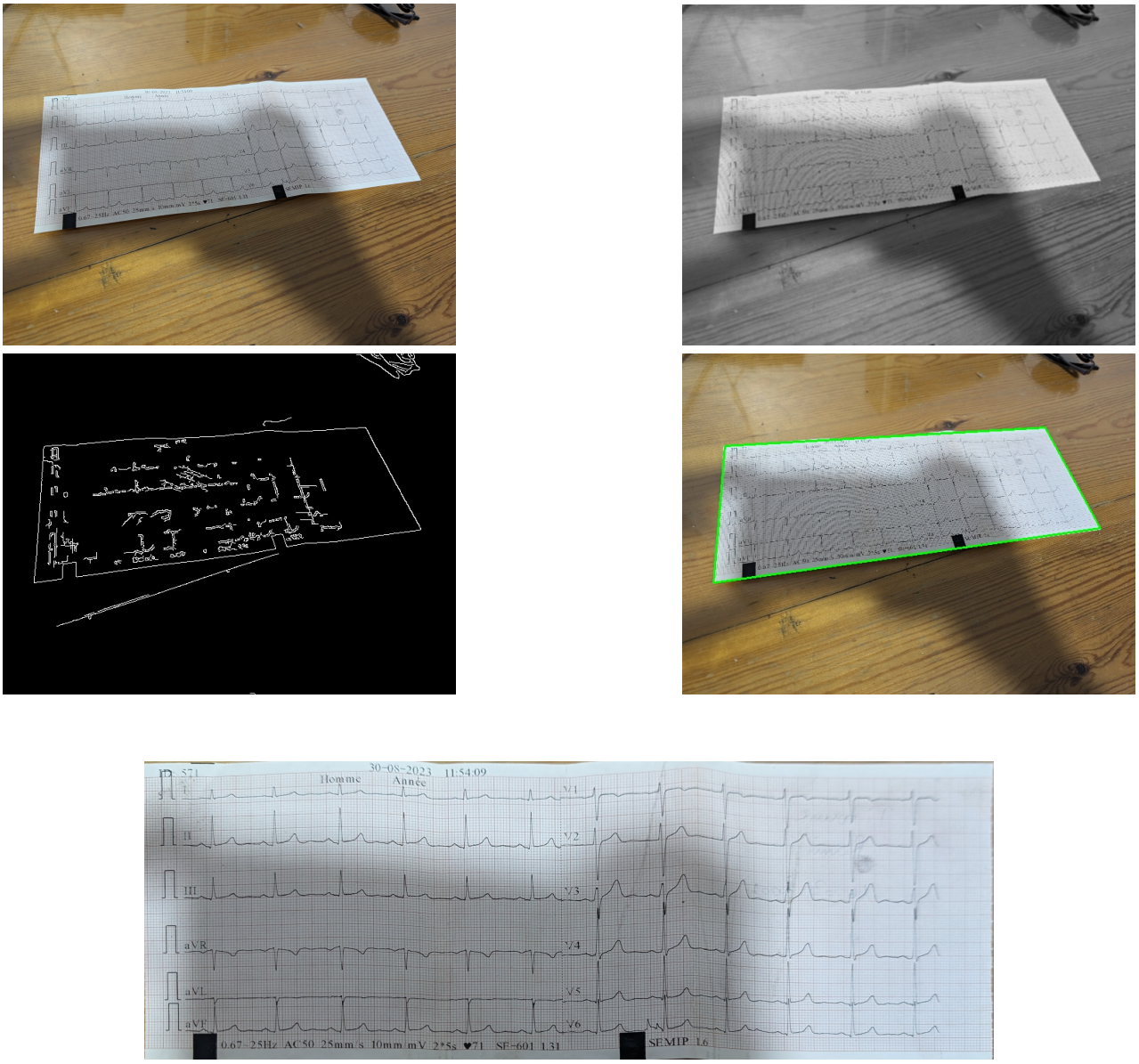


Figure 4.14: Test de la fonctionnalité de scan.

Il est important de noter que l'orientation initiale de l'ECG n'est pas prise en compte dans cette procédure. Cela peut, par exemple, se résoudre par l'utilisation d'un outil de reconnaissance optique de caractères (OCR en Anglais, pour optical character recognition) en détectant toute chaîne de caractères présente sur une image. De ce fait, il est possible d'obtenir des informations sur chaque chaîne de caractères présente sur l'image et donc connaître son orientation afin de corriger l'orientation de l'image d'entrée.

4.2.3 Algorithme de scan d'un ECG

La procédure du scan d'un ECG à partir d'une photo, ou d'un document en général, se résume à cet algorithme :

Algorithm 1 Algorithme de scan d'un ECG à partir d'une photo

Require: I: Image de dimensions $L \times H$
Ensure: I_{scan}: Scan

$C \leftarrow \max(L, H)$
$l \leftarrow 320$
$r \leftarrow \frac{l}{C}$
$(L', H') \leftarrow r \cdot (L, H)$
$I' \leftarrow \text{redimensionner}(I, (L', H'))$

$I_{\text{gris}} \leftarrow \text{niveauxDeGris}(I')$
$I_{\text{floue}} \leftarrow \text{filtreGaussien}(I_{\text{gris}}, (3 \times 3))$
$I_{\text{Canny}} \leftarrow \text{Canny}(I_{\text{floue}}, 64, 210)$

$\text{contours} \leftarrow \text{trouverContours}(I_{\text{Canny}})$
$\text{contours} \leftarrow \text{triParSurface}(\text{contours}, \text{croissant} = \text{vrai})[0..7]$
$\text{contours} \leftarrow \text{contoursFermés}(\text{contours})$

for $c \in \text{contours}$ **do**
    $l_c \leftarrow \text{longueur}(c)$
    $\epsilon \leftarrow l_c \cdot 0,05$
    $\text{approx} \leftarrow \text{approximationParPolygone}(c, \epsilon)$
    if $\text{coins}(\text{approx}) = 4$ **then**
        break
    end if
     return Erreur("Impossible de trouver l'ECG")
end for

$(A, B, C, D) \leftarrow \text{points}(\text{approx})$
$(A', B', C', D') \leftarrow (\frac{A}{r}, \frac{B}{r}, \frac{C}{r}, \frac{D}{r})$

for $p(x, y) \in (A', B', C', D')$ **do**
    $S_i \leftarrow x + y$
    $D_i \leftarrow x - y$
end for

$P \leftarrow (p_0, p_1, p_2, p_3) \leftarrow (\min(S), \min(D), \max(S), \max(D))$

$N_1 \leftarrow \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2}$
$N_2 \leftarrow \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$
$N \leftarrow \max(N_1, N_2)$

$M_1 \leftarrow \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
$M_2 \leftarrow \sqrt{(x_0 - x_3)^2 + (y_0 - y_3)^2}$
$M \leftarrow \max(M_1, M_2)$

$\text{dest} \leftarrow ((0, 0), (L, 0), (L, H), (0, H))$
$\text{matrice} \leftarrow \text{matriceTransformation}(P, \text{dest})$

$I_{\text{scan}} \leftarrow \text{perspective}(I, \text{matrice}, \text{dimensions} = (N, M))$

4.3 Binarisation

Le but de la binarisation est d'obtenir deux ensembles de pixels :

1. P_{signal} : qui représente, le plus fidèlement possible, l'ensemble des pixels appartenant au signal. Ils feront office de pixels objets (ou forme) et auront pour valeur 1.

2. $P_{\text{background}}$: qui représente, le plus fidèlement possible, l'ensemble des pixels n'appartenant pas au signal. Ils feront office de pixels d'arrière-plan (ou fond) et auront pour valeur 0.

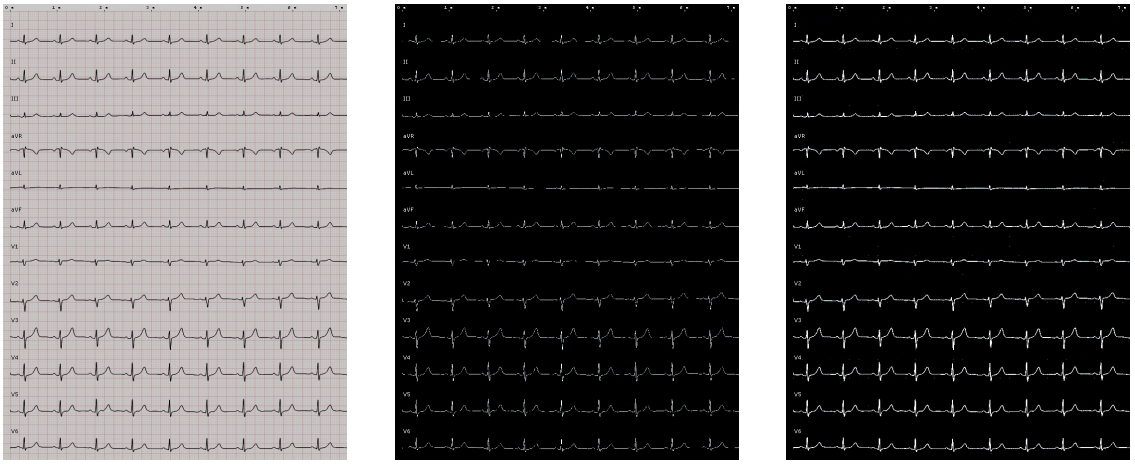
Pour se faire, il faudra trouver un moyen de filtrer les pixels du signal de ceux de la grille et du fond. Nous avons envisagé deux approches pour résoudre ce problème.

4.3.1 Approche par couleurs

Ces algorithmes se basent sur la couleur des pixels afin de déterminer P_{signal} et $P_{\text{background}}$:

Seuillage global

Comme nous l'avons supposé plus tôt, le signal est plus sombre que le papier. Il serait donc légitime d'essayer un seuillage global avec un seuil T assez bas. Prenons par exemple $T_1 = 25$ (10% de la valeur maximale) et $T_2 = 127$ (environ 50% de la valeur maximale) sur une image scannée :



(a) Scan. (b) Seuillage à $T_1 = 25$. (c) Seuillage à $T_2 = 127$.

Figure 4.15: Seuillage global sur un scan numérique d'ECG.

Nous remarquons que le seuillage avec T_1 altère les signaux de l'ECG en y créant des coupures. Ce résultat est à éviter afin de minimiser la perte d'informations durant la numérisation, en plus de rendre la segmentation bien plus difficile. Obtenant ainsi des pixels $p(x, y) = 0 \in P_{\text{signal}}$.

Tant que ces coupures ne sont pas trop grandes (moins de dix pixels), une opération morphologique de fermeture avec pour dilatation un élément structurant en diamant (pour les pics des signaux) ou en barre horizontale (voire en croix) pour les parties plates des signaux

boucherait ces trous facilement ; quant à l'érosion, elle peut se faire avec un élément structurant de la dilatation ou carré.

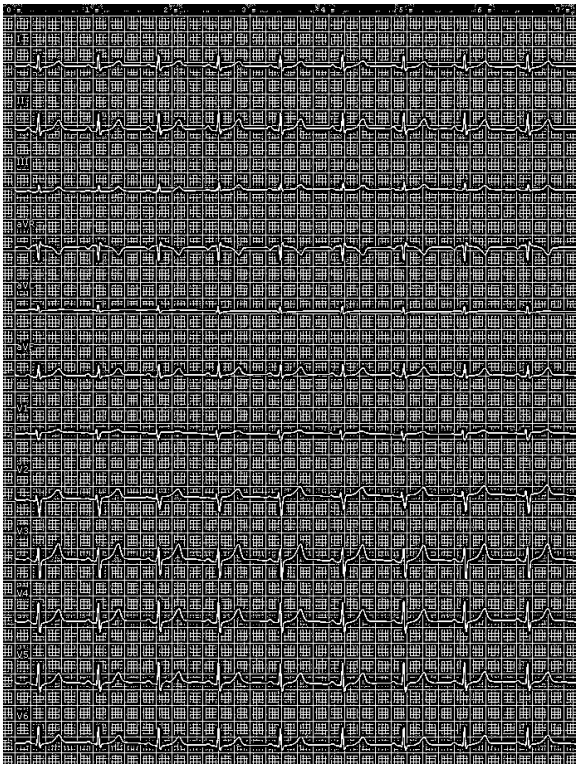
Il est important de noter que les opérations morphologiques sont à utiliser seulement en dernier recours car, dans la plupart des cas, elles altèrent les pixels objet du signal (surtout les pics) et donc menacent la fiabilité de la numérisation. Si elles doivent être utilisées, elles doivent avoir un impact minimal sur la forme initiale des signaux et doivent seulement servir à sa restitution.

Le seuillage avec T_2 est légèrement trop souple et prend des pixels appartenant à la grille ou même au fond. Nous les remarquons sous forme de petits amas de pixels blancs. Cette approche est aussi à éviter car ces amas de pixels peuvent créer des ponts entre les pics des signaux et donc une ambiguïté sur la forme du signal. Obtenant ainsi des pixels $p(x, y) = 1 \notin P_{\text{signal}}$.

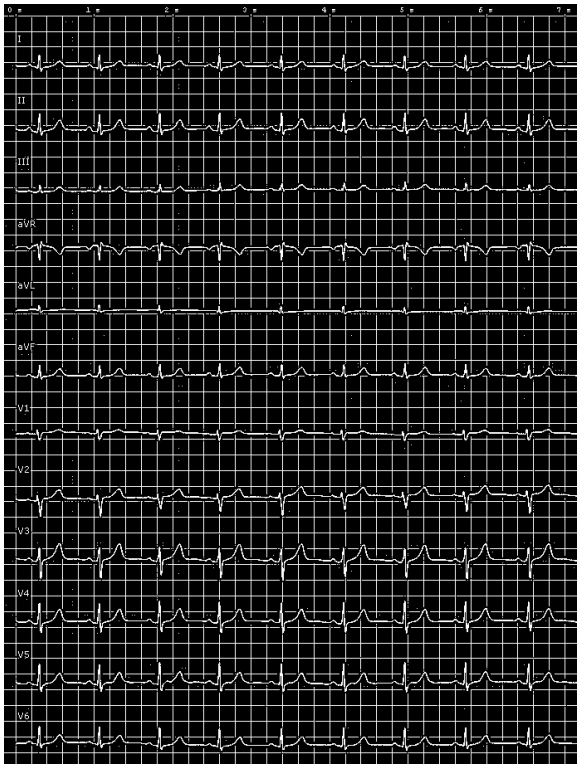
Seuillage adaptatif

Une autre méthode qui semble plus polyvalente serait d'utiliser un seuillage adaptatif (local) avec la méthode suivante :

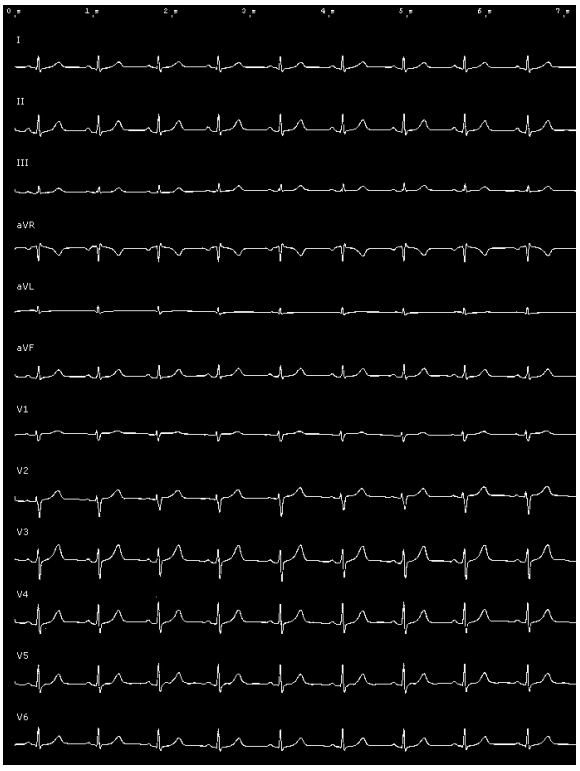
- La valeur de seuil est une somme pondérée par une Gaussienne des valeurs du voisinage, à laquelle on soustrait une constante C ;
- La taille du bloc est $B = (11 \times 11)$.



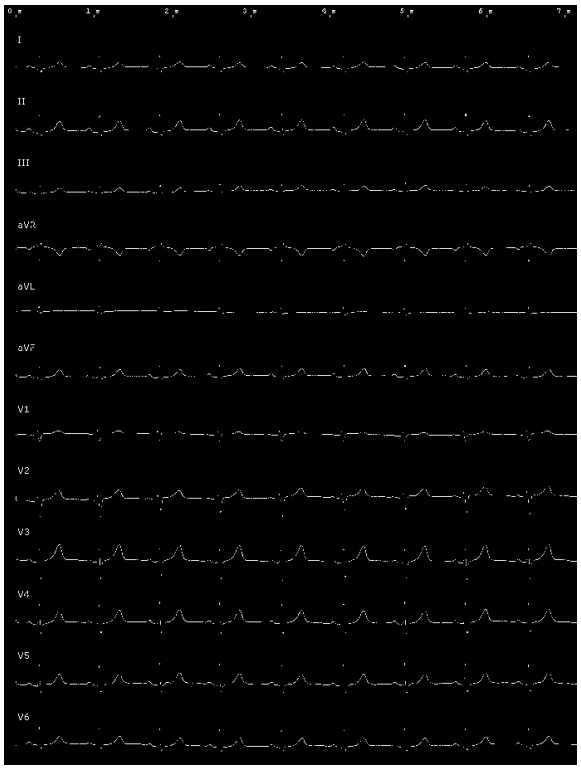
(a) Seuillage à $C = 2$.



(b) Seuillage à $C = 16$.



(c) Seuillage à $C = 64$.



(d) Seuillage à $C = 128$.

Figure 4.16: Seuillage local sur un scan numérique d'ECG.

Nous remarquons selon la valeur de C les résultats suivants :

- $C = 2$: Présence de la grille en entier ;
- $C = 16$: Présence des grandes lignes de la grille (lignes des centimètres) ;
- $C = 64$: Légère perte des signaux (courtes coupures) ;
- $C = 128$: Forte perte des signaux (longues coupures).

Cette méthode est bien trop sensible aux paramètres C et B et n'est donc pas flexible. De plus, elle a été testée sur un ECG scanné numériquement sans irrégularité de luminosité, elle sera donc bien moins fiable dans le cas d'une photo prise par un smartphone comme nous l'attendons.

Seuillage par HSV

Cette approche se base sur la couleur de la grille (supposément rougeâtre) en filtrant les couleurs claires, saturées et de teinte apparentée à celle de la grille.

Figure 4.17: Éventail de couleurs de la composante de teinte (H).

Les valeurs possibles d'une teinte sont quantifiées sur l'ensemble des entiers entre 0 et 179 (car 180 est le plus grand diviseur de 360 pouvant être représente sur un octet). Tandis que ceux de la saturation sont quantifiés de 0 à 255.

En partant d'une image I, nous la transformons dans un premier temps en une image I_{HSV} vers l'espace HSV. Ensuite nous effectuons une séparation par composantes obtenant ainsi I_H, I_S, I_V. Le but étant d'appliquer une sorte de seuillage sur chacune de ces composantes suivant ces conditions :

- $I'_H(x, y) = 1$ si $(0,11 \times 180) \leq I_H(x, y) \leq (0,89 \times 180)$, sinon 0 ;
- $I'_S(x, y) = 1$ si $0 \leq I_S(x, y) \leq (0,8 \times 255)$, sinon 0 ;
- $I'_V(x, y) = 1$ si $0 \leq I_V(x, y) \leq \frac{255}{2}$, sinon 0.

L'image que l'on cherche à obtenir est un ET logique entre I'_H et I'_S et I'_V. Voici le résultat :

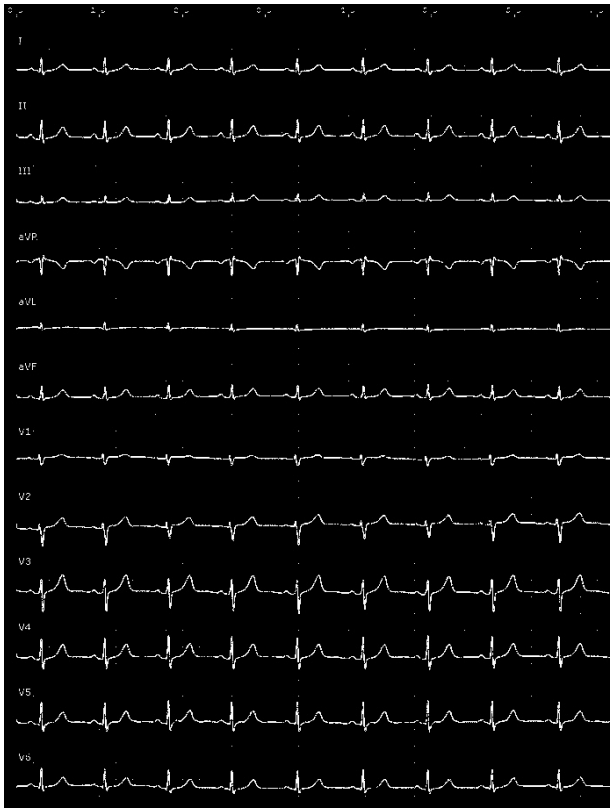
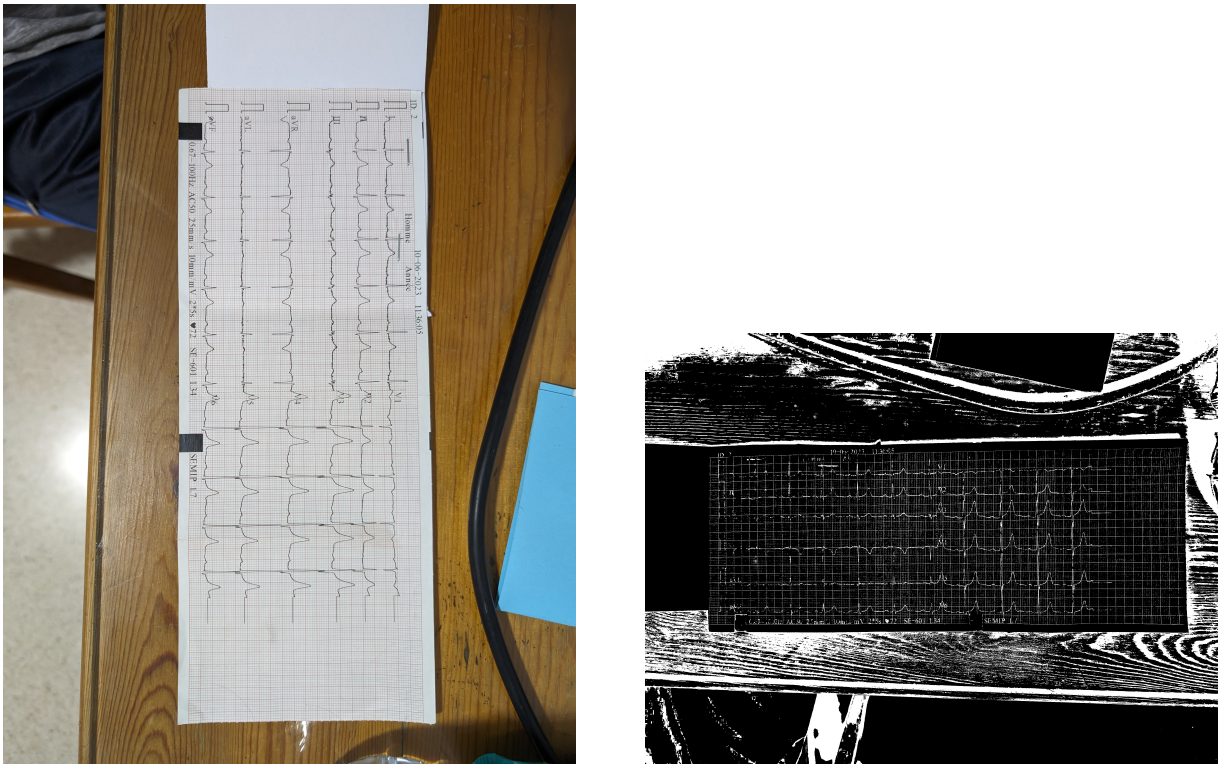


Figure 4.18: Seuillage par HSV.

Cette méthode semble donner de bons résultats car elle élimine un grand nombre de pixels de la grille. Cependant, elle n'est pas adaptée à une image à luminosité irrégulière ou présentant du bruit comme nous pouvons l'observer dans ce cas :



(a) Entrée. (b) Sortie.

Figure 4.19: Seuillage par HSV sur une photo à luminosité irrégulière.

Nous remarquons que dans plusieurs zones de l'image, les pixels de la grille sont pris pour pixels objet et ceux du signal ne le sont pas, faisant ainsi l'exact opposé de ce que l'on cherche à faire. Tandis que dans certaines zones, la grille et le signal sont pris comme objet ou seulement le signal l'est.

En plus d'être sensible à la luminosité, cette méthode n'est pas adaptée aux images de faible qualité ou qui contiennent un bruit modéré ou fort.

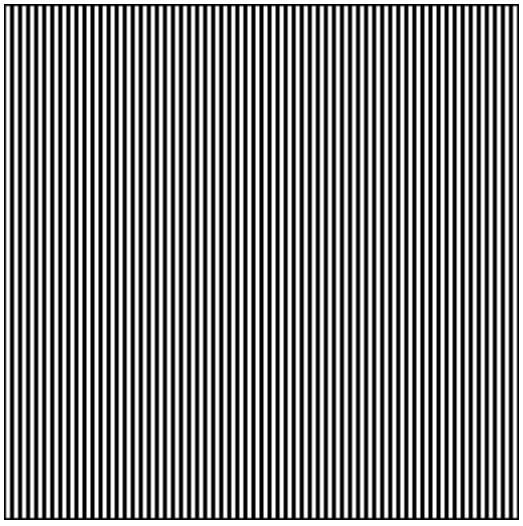
Problématique Ces approches par couleurs s'appuient trop sur des constantes programmées en dur (c'est-à-dire que leur valeur est connue avant l'exécution du programme) et ne sont donc pas assez flexibles et surtout trop sensibles aux irrégularités de la luminosité, ce qui est un problème plus au moins commun sur une photo prise par un utilisateur.

Il est possible d'améliorer les résultats obtenus au cas par cas en jouant sur le contraste, les transformations morphologiques et l'histogramme de l'image d'entrée. Ces améliorations reposent aussi sur des constantes programmées en dur vu qu'elles sont spécifiques à chaque cas.

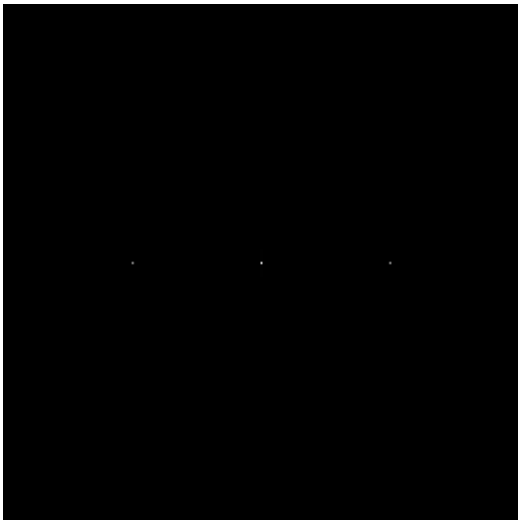
Cela n'est pas envisageable dans notre application car la numérisation est censée être un processus automatisé.

4.3.2 Approche par la transformée de Fourier

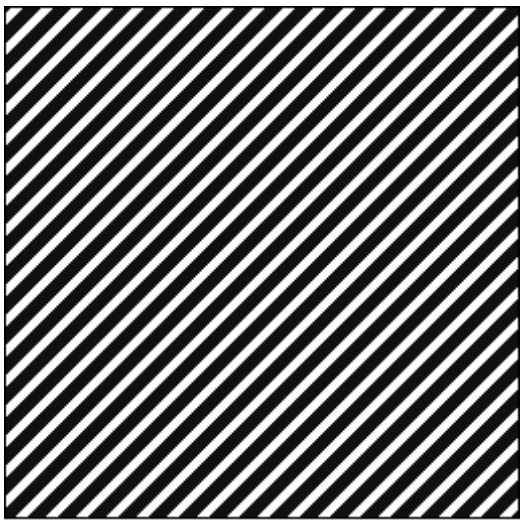
Pour comprendre cette approche, nous allons comparer des images d'entrée à leur spectre de magnitude suivant une transformée de Fourier en deux dimensions :



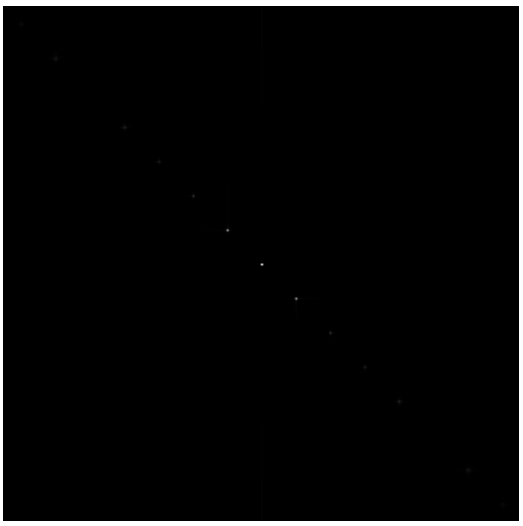
(a) Image à bandes blanches verticales sur fond noir.



(b) Spectre de magnitude de la transformée de Fourier.



(c) Image à bandes blanches en diagonale sur fond noir.



(d) Spectre de magnitude de la transformée de Fourier.

Figure 4.20: Transformées de Fourier sur des images binaires présentant des motifs de bandes blanches droites.

Nous remarquons d'abord que la transformée de Fourier est symétrique au centre, le point $O = (0, 0)$.

Ensuite, nous pouvons distinguer deux points blancs espacés d'un troisième point blanc au centre, tous alignés selon la direction du motif répété. Enfin, les distances entre ces points est inversement proportionnelle à la largeur des bandes.

La visualisation du spectre de magnitude de la transformée de Fourier est en niveaux de gris, partant du noir pour 0 vers le blanc pour la valeur maximale

.

La transformée de Fourier est un outil utile à la détection de motifs réguliers dans une image. Nous pouvons essayer d'exploiter cette spécificité afin de d'effacer les pixels composant

la grille.

En effet, la grille est un motif régulier dans un ECG suivant l'axe horizontal et vertical. Nous nous attendrons donc à retrouver un spectre de magnitude composé de plusieurs points très proches du blanc suivant l'axe vertical et l'axe horizontal, tous espacés d'une même distance. Il suffira donc de les effacer en les rendant noirs (nuls) pour au final inverser la transformée de Fourier obtenant ainsi une image, en théorie, sans grille.

Spectre de magnitude d'un ECG

Nous allons commencer par observer les spectres de magnitude de deux images d'un ECG :

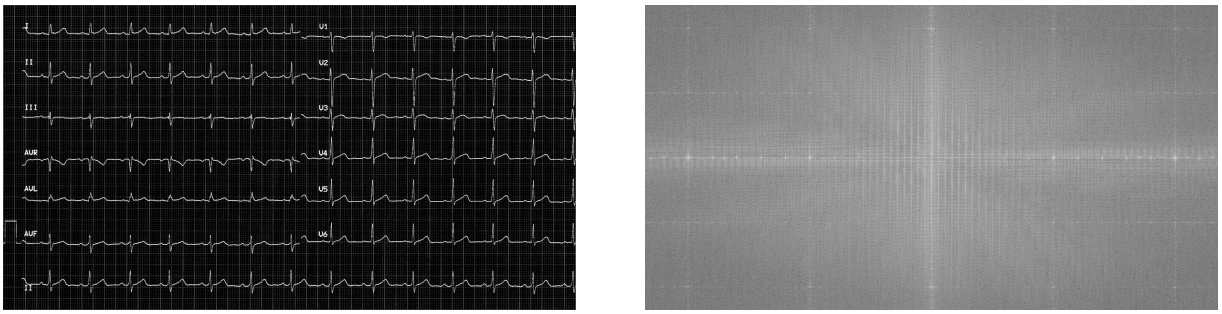


Figure 4.21: Spectre de magnitude d'un ECG scanné par machine.

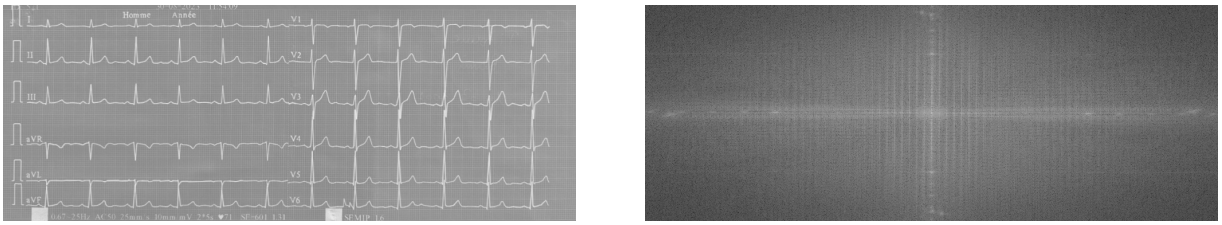


Figure 4.22: Spectre de magnitude de l'inverse de I_{scan}.

Nous remarquons, comme prévu, la présence de points blancs le long de l'axe horizontal et l'axe vertical. Nous allons maintenant observer les effets de l'altération du spectres de magnitude sur I_{scan}.

Altération du spectre de magnitude de l'image d'un ECG

Il est préférable d'inverser les couleurs de I_{scan} pour mieux voir les changements apportés :

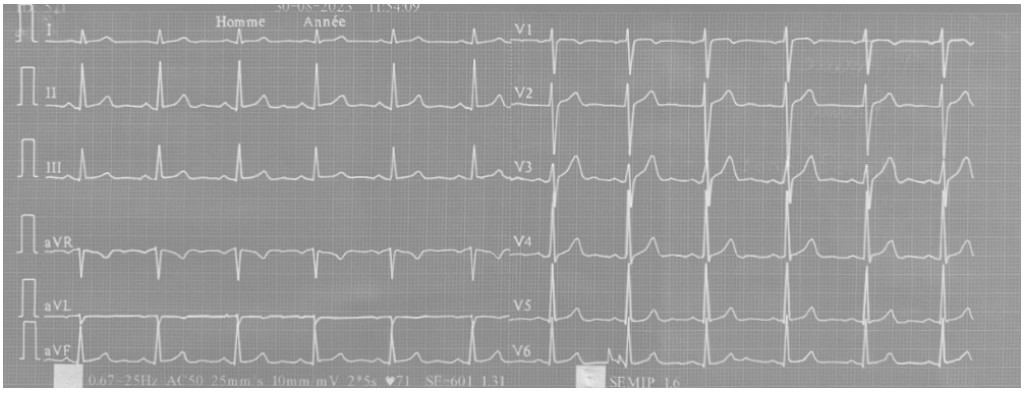
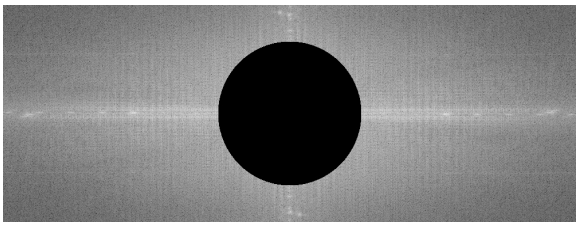
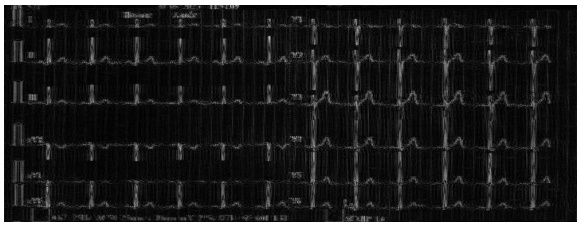


Figure 4.23: Inverse de I_{scan}.

Commençons par supprimer les basses fréquences du spectre de magnitude (les plus proches du centre O) en insérant un disque de zéros centré au point O de rayon $r = 125$ pixels :



(a) Spectre de magnitude altéré.



(b) Résultat.

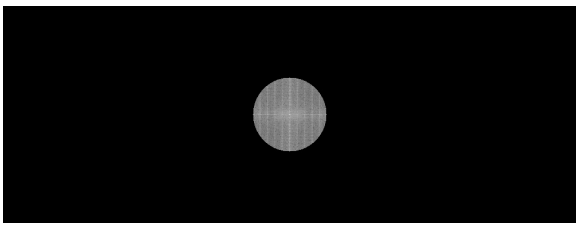
Figure 4.24: Suppression de basses fréquences.

Nous remarquons que seulement les "détails" de l'image sont préservés, nous venons d'effectuer un filtre passe-haut.

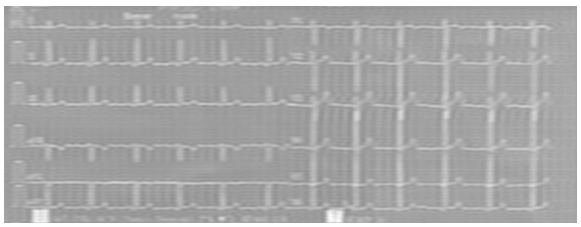
En effet, ce résultat est attendu. Car comme nous l'avons vu précédemment, plus un point est blanc, plus il représente un changement intense de valeur (ou intensité) suivant sa position dans l'image. D'autre part, les points loin du centre du spectre reflètent une différence d'intensité des points proches dans l'image d'entrée.

En résumé, cette opération préserve seulement l'ensemble des pixels montrant un changement de valeur avec ses pixels voisin les plus proches.

Maintenant, nous allons supprimer les hautes fréquences. Pour se faire, il suffit d'épargner les points loin du centre O d'un rayon $r = 64$ pixels et supprimer le reste des points :



(a) Spectre de magnitude altéré.



(b) Résultat.

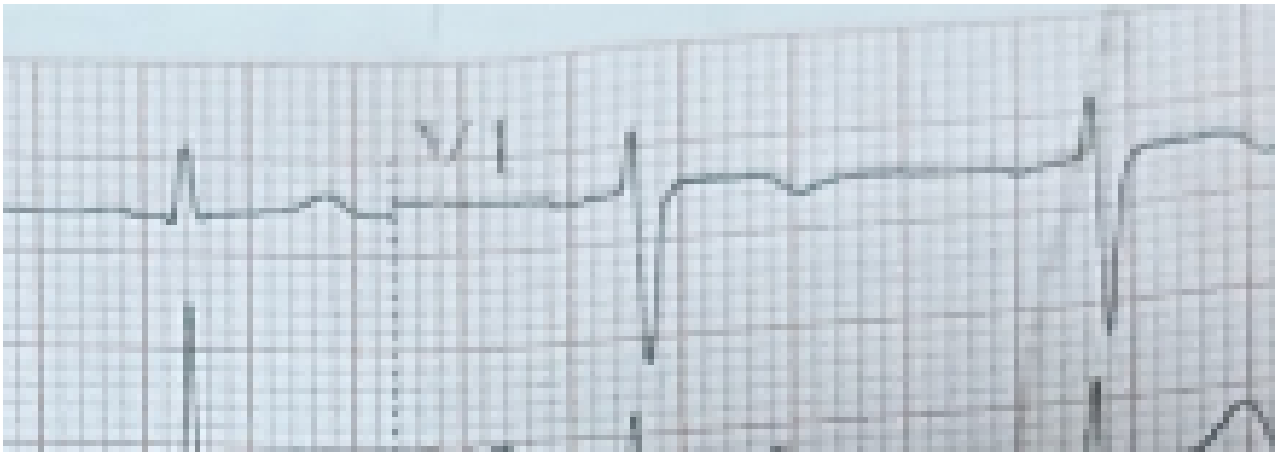
Figure 4.25: Suppression de hautes fréquences.

Cette fois-ci, le résultat est une image floue ; nous venons d'effectuer un filtre passe-bas.

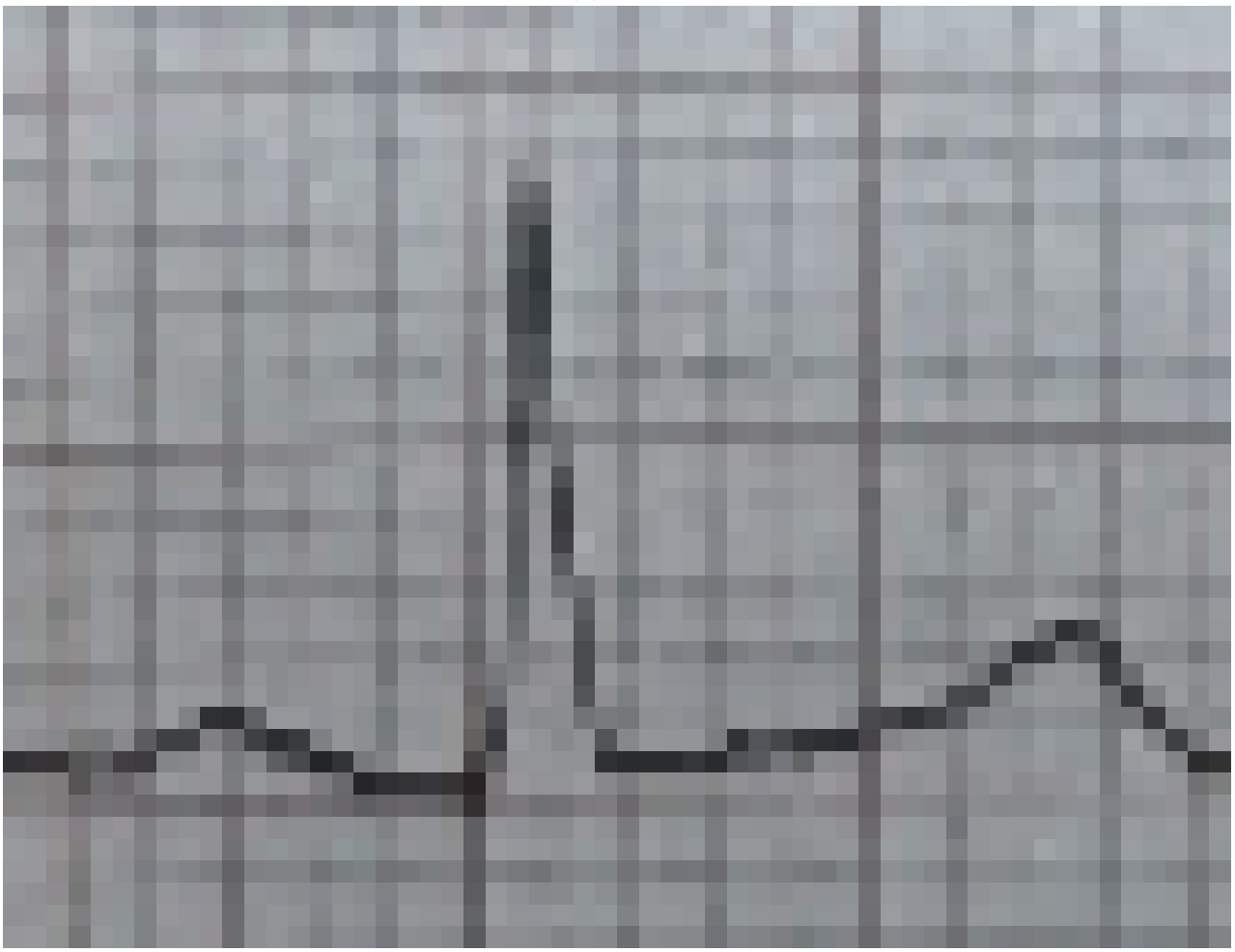
Problématique

L'idéal serait de détecter la distance entre les lignes de la grille des centimètres ainsi que celle des millimètres et de retrouver les fréquences correspondantes sur les axes du spectre de magnitude afin de les supprimer. Cependant, cette solution est bien trop complexe à implémenter en prenant en compte que :

- La majorité des ECGs présentent des plis pour deux raisons : le papier est stocké sous forme de rouleau, le médecin ou le patient plie souvent le papier pour le transporter facilement. Cela crée un une déformation des lignes de la grille et un effet d'escalier plus au moins prononcé ;
- Les imprécisions de la fonction de scan dues au redimensionnement, l'interpolation, les arrondissements dans les calculs et le filtre passe-bas qui crée une ambiguïté le long des bords de la région d'intérêt ;
- La qualité de l'image d'entrée, que ce soit de par la qualité de l'appareil photo ou d'éventuelles détériorations telles que celle causée par la compression ;
- La similarité entre la grille et le signal dans certaines zones ;
- La difficulté à trouver un seuillage fiable qui garantit la segmentation de la grille et de la sous-grille ;
- La présence de bruit sur l'image.



(a) Plis.



(b) Détérioration de la qualité et présence de bruit.

Figure 4.26: Obstacles à la binarisation d'une image.

Ces obstacles constituent des raisons additionnelles qui expliquent l'échec de l'approche par couleurs.

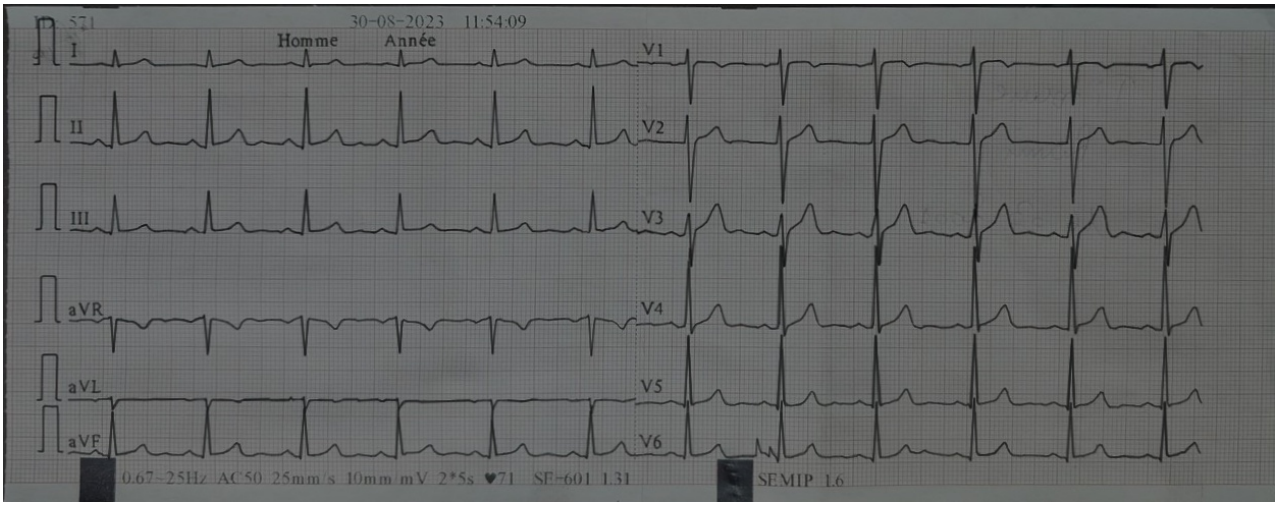
Nous remarquons même qu'en zoomant encore plus sur la seconde figure, il devient presque impossible de discerner les pixels de la grille de ceux du signal.

Solution

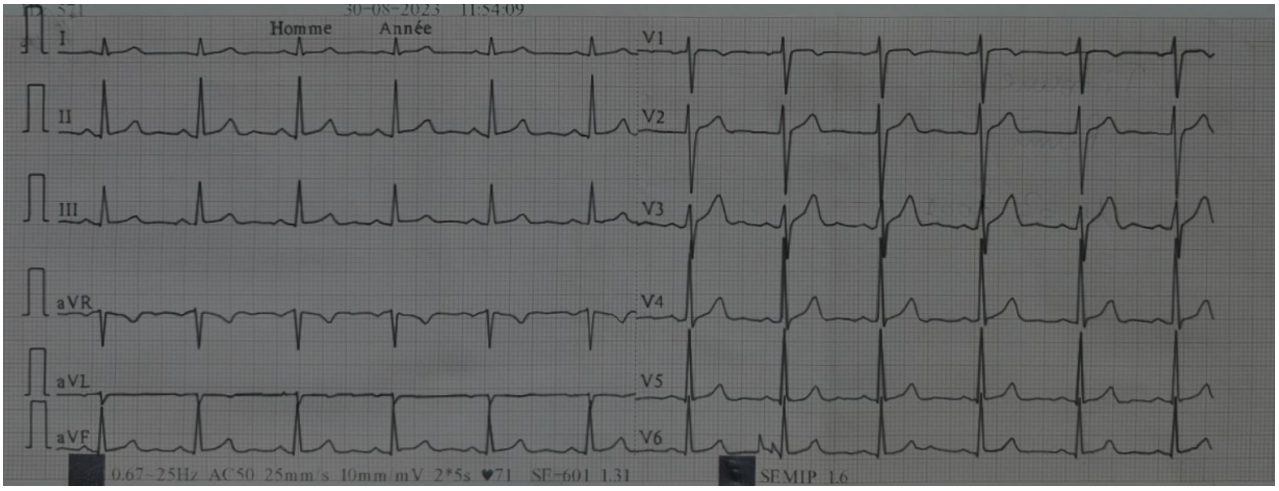
Étant donné que le signal est la partie la plus importante à préserver, le filtre passe-haut comme nous l'avons effectué est trop risqué car il est susceptible de créer des coupures conséquentes dans le signal à plusieurs endroits et de manière imprédictible.

Ceci est dû à la nature répétitive des signaux sur l'axe horizontal, il est donc important de préserver certaines basses fréquences.

Tout d'abord, nous allons redimensionner l'image de sorte à avoir une résolution spatiale équivalente à 1600×626 pixels et ensuite la rogner de 16 pixels de chaque côté afin de palier aux imprécisions de la fonctions de scan et aux vides laissés par les plis :



(a) Après l'algorithme de scan.



(b) Redimensionnée puis rognée.

Figure 4.27: Ajustement de l'image scannée.

Le rendimensionnement à cette résolution spatiale permet d'avoir une idée sur l'ordre de grandeur des signaux et des distances qui les séparent indépendamment de la disposition des éléments de l'ECG traité.

Car en soit, tant que le nombre de dérivations est le même, l'espace occupé par ces

éléments, ou plus précisément la surface "remplie" de l'ECG, sera approximativement la même, dépendant seulement des marges laissées entre les signaux de dérivation ou entre les bords et les signaux.

Ensuite, nous allons inverser les couleurs de l'image :

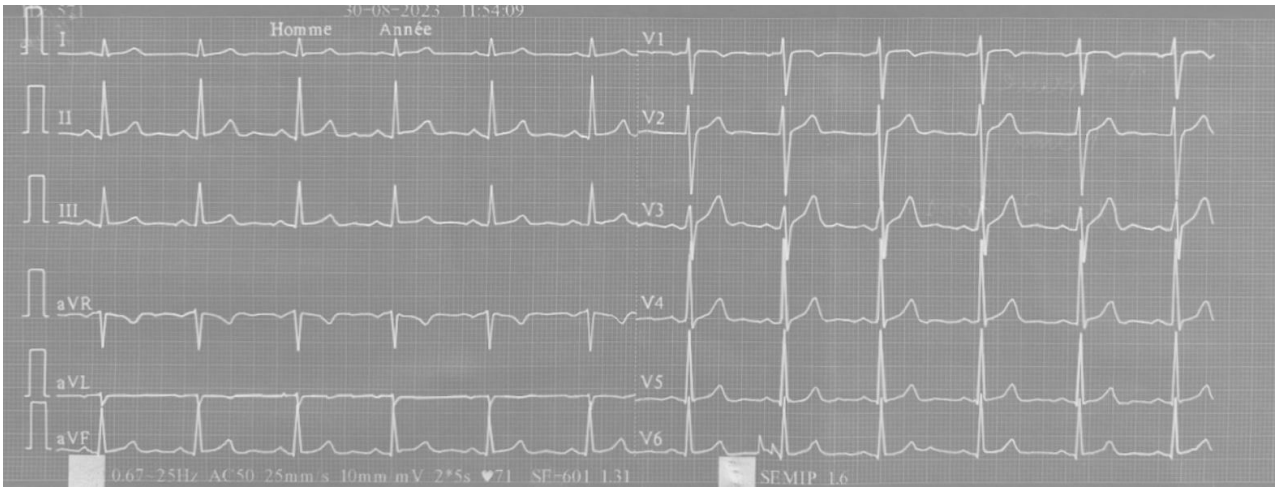
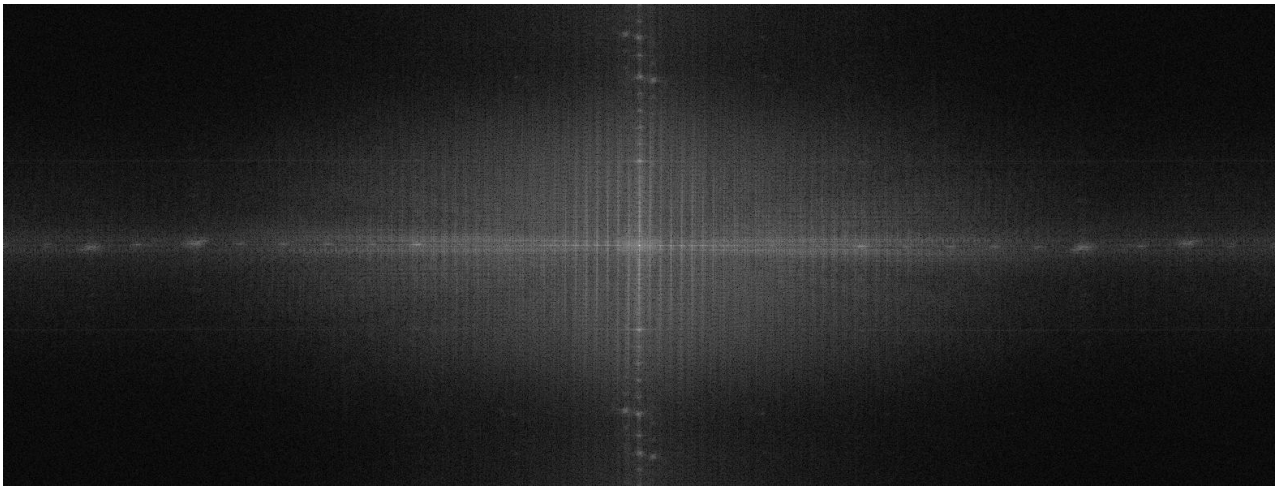
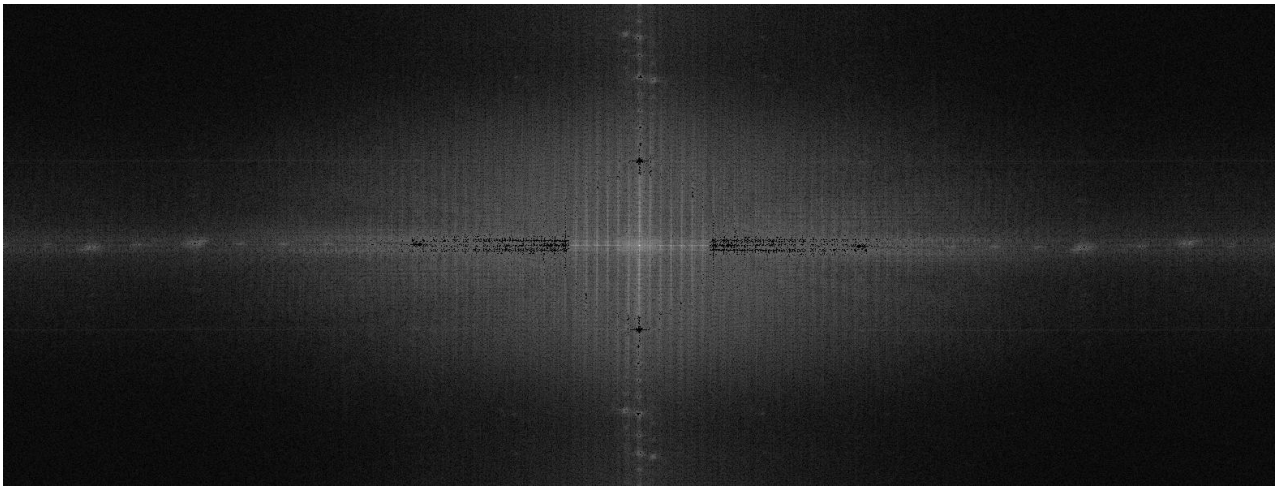


Figure 4.28: Inverse de l'image traitée.

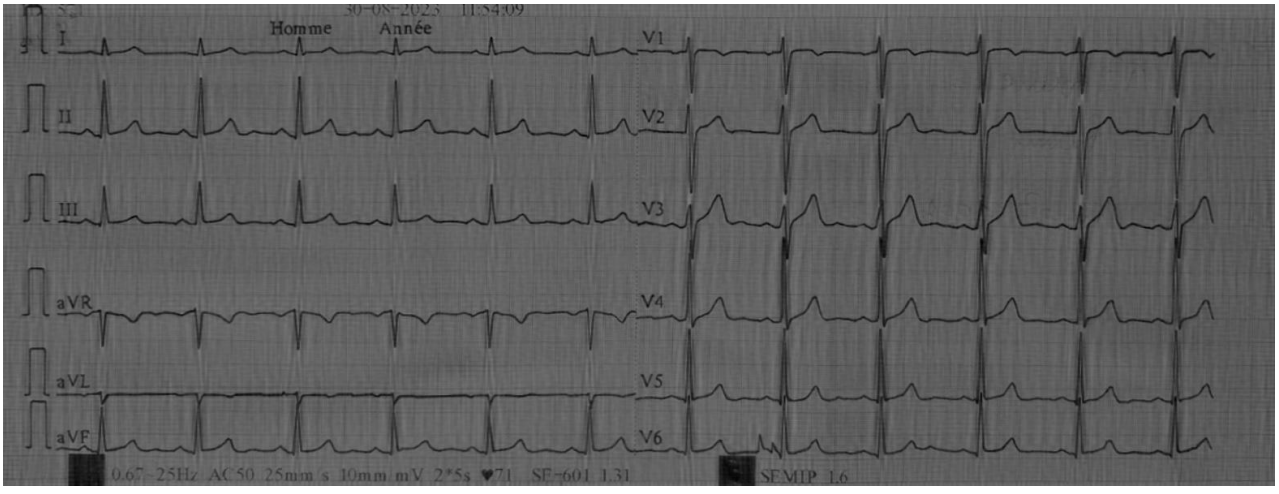
Un traitement du spectre de magnitude sera effectué. D'abord, la valeur maximale M du spectre de magnitude et les valeurs supérieures à $0,995M$ (les points blancs ou "presque" blancs) seront éliminés, sauf dans une zone sous forme de disque incluant les basses fréquences. Le rayon du disque est pris comme 5,5% de la résolution horizontale de l'image, donnant une nouvelle image I_{filtered} :



(a) Spectre de magnitude avant le traitement.



(b) Spectre de magnitude après le traitement.

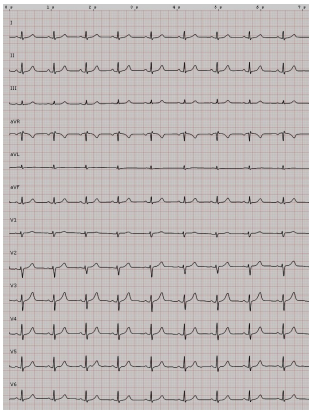


(c) Résultat.

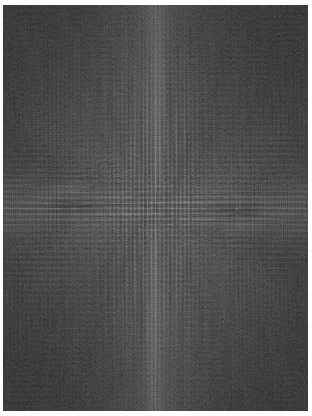
Figure 4.29: Traitement du spectre de magnitude.

Les pixels composants la grille ont été très fortement altérés tandis que ceux des signaux

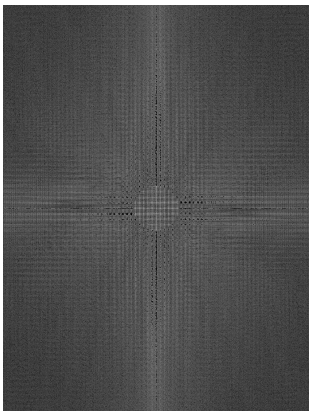
de dérivation sont presque intactes. Cette étape est efficace même sur les ECGs scannés par machines :



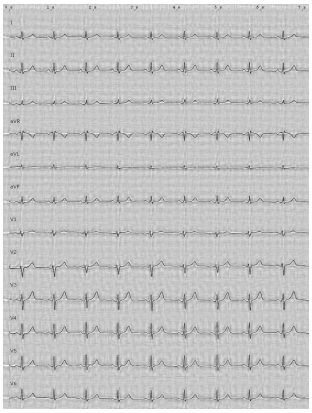
(a) Premier scan.



(b) Spectre de magnitude avant le traitement.



(c) Spectre de magnitude après le traitement.



(d) Résultat.

Figure 4.30: Test du traitement de magnitude sur une image scannée.

Un traitement par la transformer de Fourier a permis de cibler des pixels non pas par leur couleur, mais par leur disposition dans l'image ainsi que leur valeur relative à leurs voisins.

Trois opérations de traitement d'images sont efficaces à accroître la différence entre un signal et une grille qui sont :

1. Le rehaussement de l'image (filtres passe-haut) : fait ressortir le signal légèrement plus que la grille ;
2. Le floutage (filtres passe-bas) : efface le signal mais efface légèrement plus la grille ;
3. La correction Gamma (changement du niveau de luminosité) : efface la grille si elle n'est pas trop foncée mais efface le signal s'il est trop clair.

Après le traitement du spectre de magnitude, l'altération de la grille a été tellement forte que peu importe la technique de rehaussement utilisée, les pixels du signal en bénéficieront bien plus que ceux de la grille.

Nous avons utilisé une technique de rehaussement qui profite des avantages du filtre passe-bas puis de ceux du filtre passe-haut en calculant une nouvelle image I_{sharp} à partir d'une image I et son équivalent après un filtre Gaussien I_{Gaussian} par l'équation suivante :

$$I_{\text{sharp}} = I + q \cdot (I - I_{\text{Gaussian}}) \tag{4.3}$$

Un autre paramètre r (pour rayon) représente le paramètre σ du filtre Gaussien.

Cette transformation est intéressante car le flou affecte fortement ce qui reste de la grille après le traitement du spectre de magnitude. Voici le résultat après rehaussement sur l'image avec pour paramètres $r = 16$ et $q = 1,25$:

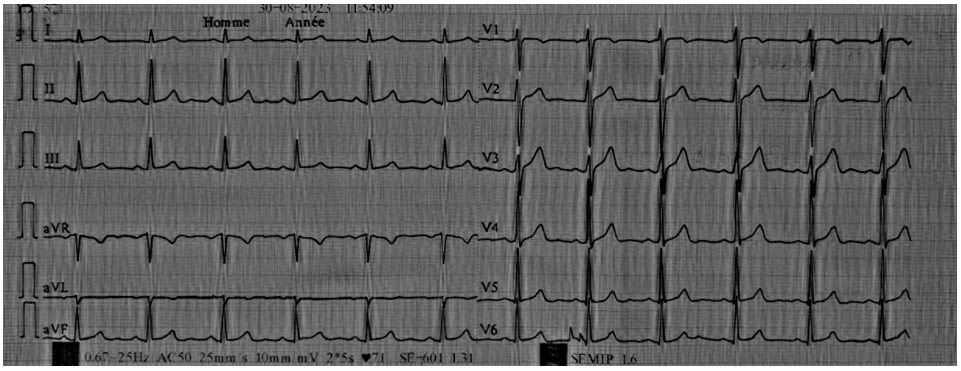


Figure 4.31: Après rehaussement.

Maintenant que le signal a été assombri et restitué à certains endroits, nous allons effectuer une correction Gamma en ramenant la valeur des pixels à un réel $x \in [0, 1]$ et en retournant de nouveaux pixels $V = x^{\gamma}$.

Il est important de remarquer que plus un pixel possède une valeur extrême (proche du noir ou du blanc), moins il est affecté par la transformation Gamma. Voici le résultat avec $\gamma = 0,8$:

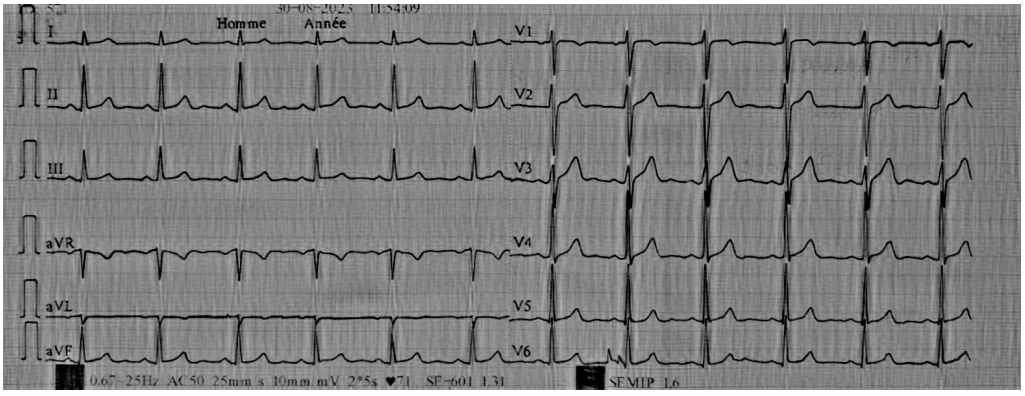


Figure 4.32: Augmentation de la luminosité.

La grille étant à ce point altérée et le signal renforcé, le seuillage devient "facile" et tolérant à un large ensemble de valeurs, et ceci même en utilisant un seuillage global.

Voici le résultat après un seuillage global avec pour seuil $T = 127$:

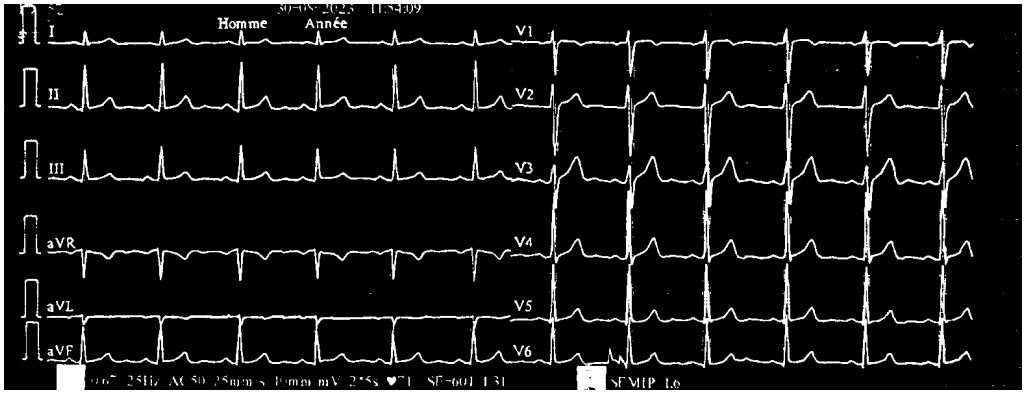
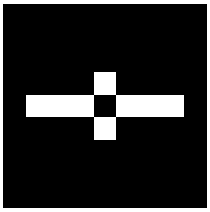


Figure 4.33: Seuillage global.

Malgré l'enchaînement de ces étapes qui ont visé à maximiser la préservation du signal et minimiser celle de la grille, il n'est pas impossible de se retrouver avec de petites coupures du signal (rarement plus d'un pixel) notamment dans les parties purement horizontales ou verticales. Afin de combler ces éventuelles coupures du signal, une opération morphologique de dilatation est effectuée suivant l'élément structurant ci-dessus (la valeur 1 est représentée par le blanc et la valeur 0 par le noir) :



(a) Élément structurant de taille (9 × 9) de la dilatation.

(b) Résultat de la dilatation.

Figure 4.34: Dilatation correctrice de coupures.

Cet élément structurant a été conçu spécialement pour remédier plus aux coupures horizontales que verticales pour minimiser l'altération de la forme des signaux de dérivations autour des pics.

Les éventuels restes de la grille peuvent facilement s'enlever en supprimant les surfaces blanches possédant une connectivité strict (c'est-à-dire qu'il éxiste toujours un chemin de n'importe quel pixel de cette surface vers n'importe quel autre en traversant seulement les pixels objets qui la composent) ayant une aire inférieure à deux tiers de la résolution horizontale de l'image :

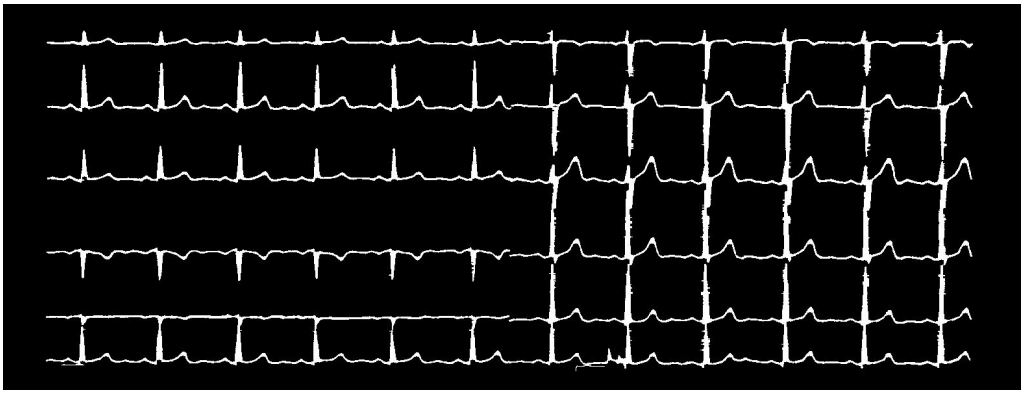
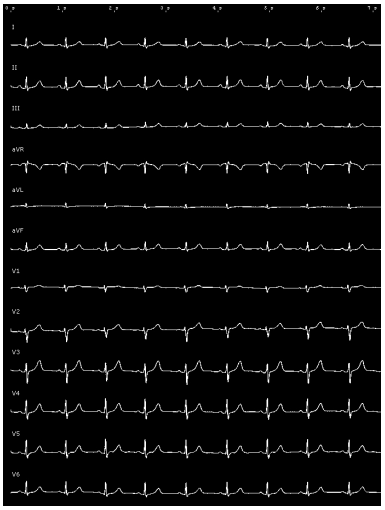


Figure 4.35: Suppression des petits objets.

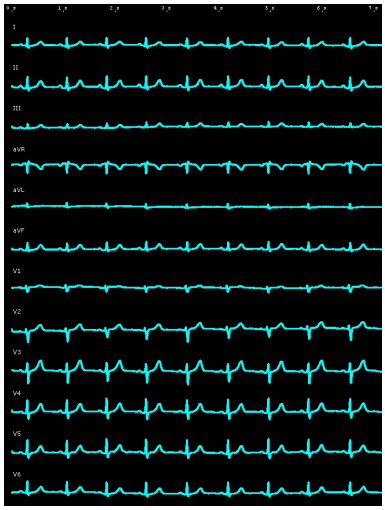
Nous arrivons enfin à notre but qui était d'obtenir une image binaire. Il est temps maintenant de séparer les signaux de dérivations et de les identifier.

4.4 Segmentation

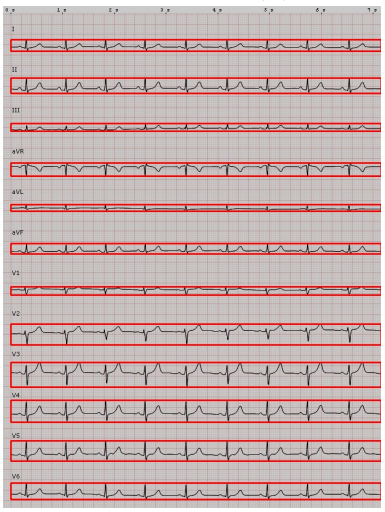
Une première idée qui vient à l'esprit en parlant de segmentation d'images binaires est l'utilisation de l'algorithme de détection des contours :



(a) Image binaire.



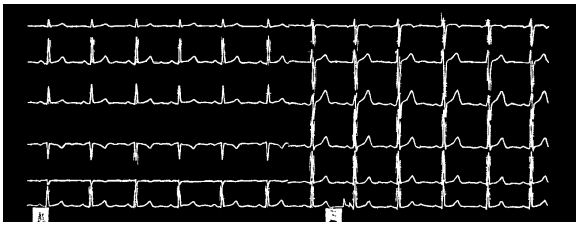
(b) Détection des contours.



(c) Rectangle minimal contenant chaque contour.

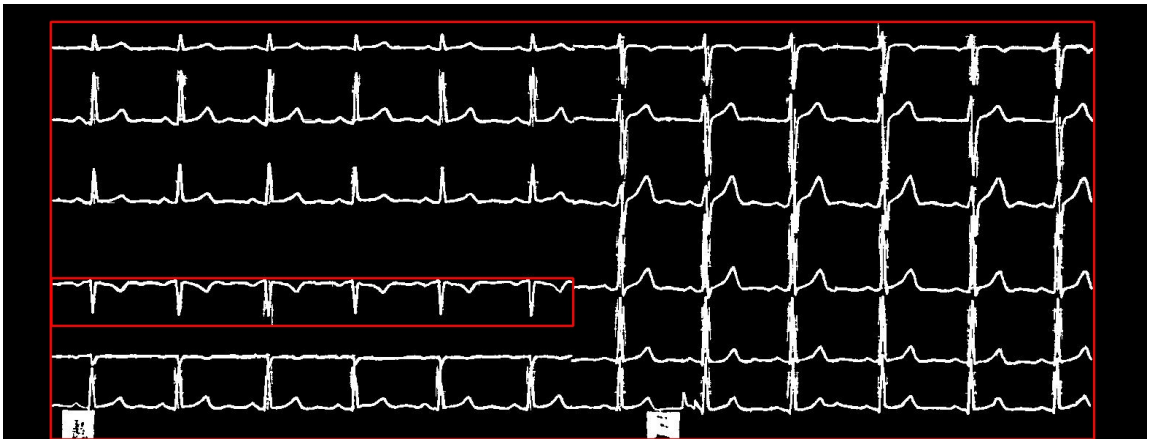
Figure 4.36: Segmentation par détection de contours.

Cette méthode ne fonctionne pas sur l'image que nous avons binarisée plus tôt à cause de chevauchements horizontaux entre les deux colonnes de dérivations et des chevauchements verticaux entre dérivations voisines :



(a) Image binaire.

(b) Détection des contours.



(c) Rectangle minimal contenant chaque contour.

Figure 4.37: Tentative de segmentation par contours.

Cette méthode vient de retourner 02 contours au lieu de 12.

4.4.1 Détection de la disposition des signaux de dérivations

Il serait légitime de d'abord identifier la disposition des signaux de dérivation au sein de l'ECG avant de tenter une segmentation. Pour cela, une solution simple et efficace consiste à se baser sur les échantillons à notre disposition et s'en servir comme références.

Le rapport de la résolution horizontale sur la résolution verticale de chaque type de disposition est très différent d'une disposition à l'autre.
En l'occurrence, on trouve, selon la disposition, les ratios suivants :

- Disposition à une colonne : environ 0,758 ;
- Disposition à deux colonnes : environ 2,639 ;
- Disposition à quatre colonnes : environ 5,930.

Et effectivement, ces rations possèdent des valeurs assez éloignées (d'au moins 1,881) et seront donc considérés comme étant des références fiables pour la détection de la disposition d'ECG.

Pour détecter la disposition des signaux de dérivation à partir d'une image scannée de dimensions (N, M), il suffit de calculer le rapport $\frac{N}{M}$ et de choisir la disposition dont le ratio est le plus proche de ce nombre.

4.4.2 Résolution du problème de chevauchement horizontal

Dans le cas précédent (disposition à deux colonnes), cinq dérivations se chevauchent horizontalement. Afin de résoudre ce problème, nous allons déduire l'enveloppe convexe (Convex hull en Anglais) de nos pixels objets.

L'enveloppe convexe d'un objet ou d'un regroupement d'objets géométriques est l'ensemble convexe le plus petit parmi ceux qui le contiennent. Dans le domaine du traitement d'images, l'algorithme de Steve Eddins [40] est utilisé afin de déterminer l'enveloppe convexe d'une image binaire :

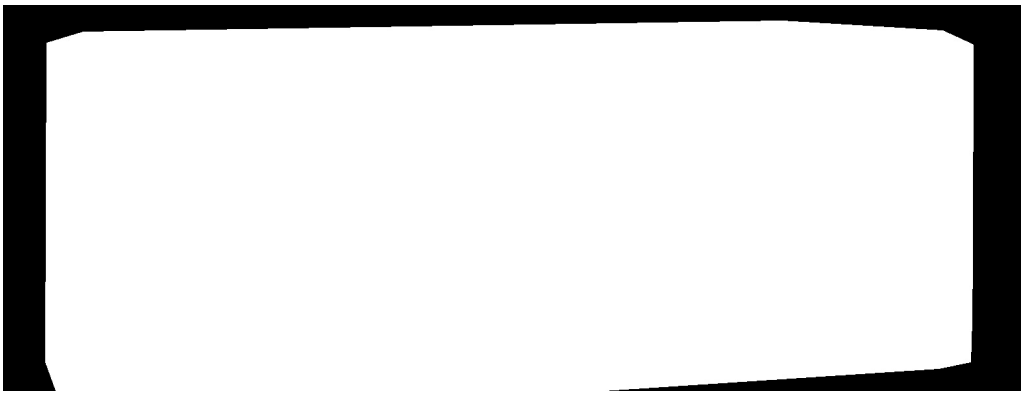
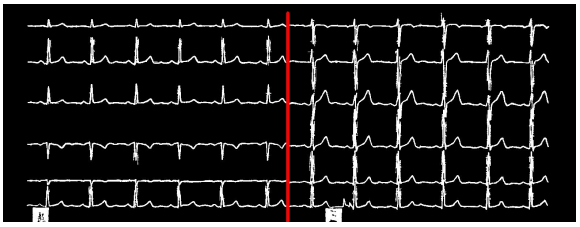
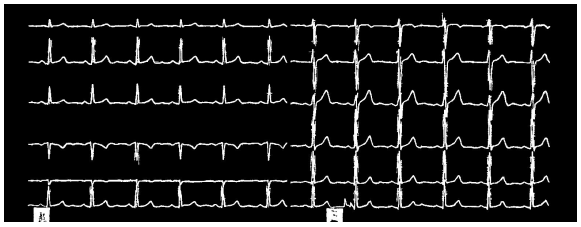


Figure 4.38: Enveloppe convexe du cas de chevauchement.

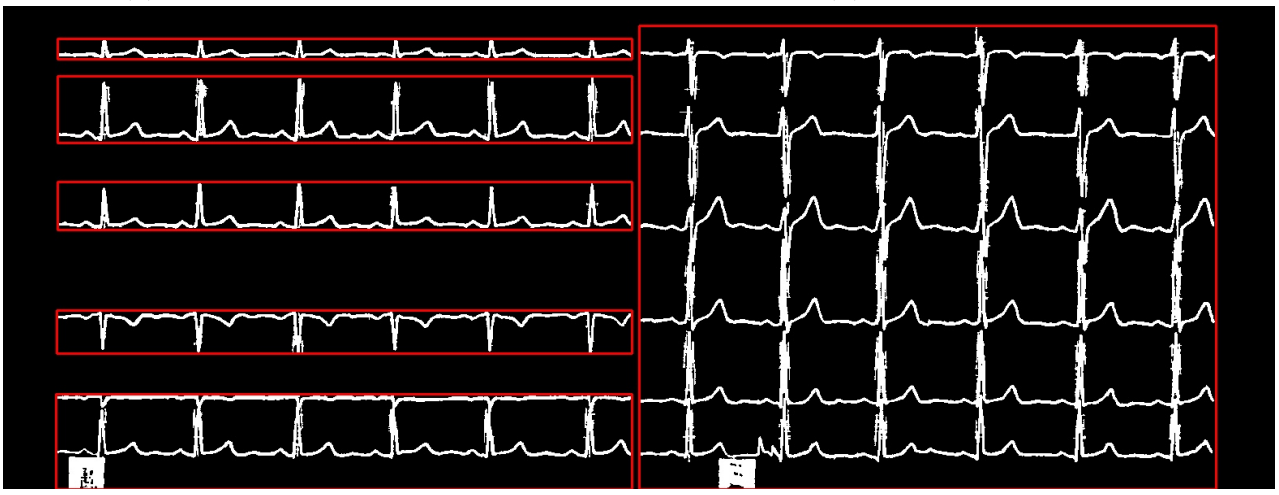
Il suffit de simuler des coupures verticales à parts égales au nombre de colonnes déduit par la détection de la disposition des signaux de dérivations. Dans le cas d'une disposition à quatre colonnes par exemple, il faudra couper l'enveloppe verticalement en quatre plutôt que deux. Cette coupure se fera par une ligne droite verticale noire (dont la valeur des pixels est 0) de huit pixels de large :



(a) Visualisation de la coupure.



(b) Coupure en noire.



(c) Segmentation par contours.

Figure 4.39: Solution au chevauchement horizontal.

Cette étape ne sert pas seulement à régler le problème du chevauchement horizontal, elle est primordiale à la segmentation et sera donc exécutée peu importe l'image binaire traitée.

Nous remarquons que les deux dernières dérivations de la première colonne se chevauchent verticalement et que toutes les dérivations de la seconde colonne se chevauchent de la même manière. Cela arrive très souvent dans les ECGs.

4.4.3 Segmentation approximative

Avant de régler le problème du chevauchement vertical, nous allons tenter de segmenter les signaux de dérivations de manière approximative. C'est à dire une segmentation qui garantit l'intégralité des pixels objet du signal de dérivation mais, peut-être en plus, des pixels objets des signaux de dérivations voisins verticalement.

En identifiant le pixel objet le plus à gauche p_{start} de chaque signal de dérivation (que l'on peut considérer comme étant un pixel de départ du signal) ainsi que le pixel objet le plus à droite p_{end} de chaque dérivation (de la même manière, un pixel d'arrivée), nous obtenons deux ensemble P_{start} et P_{end}.

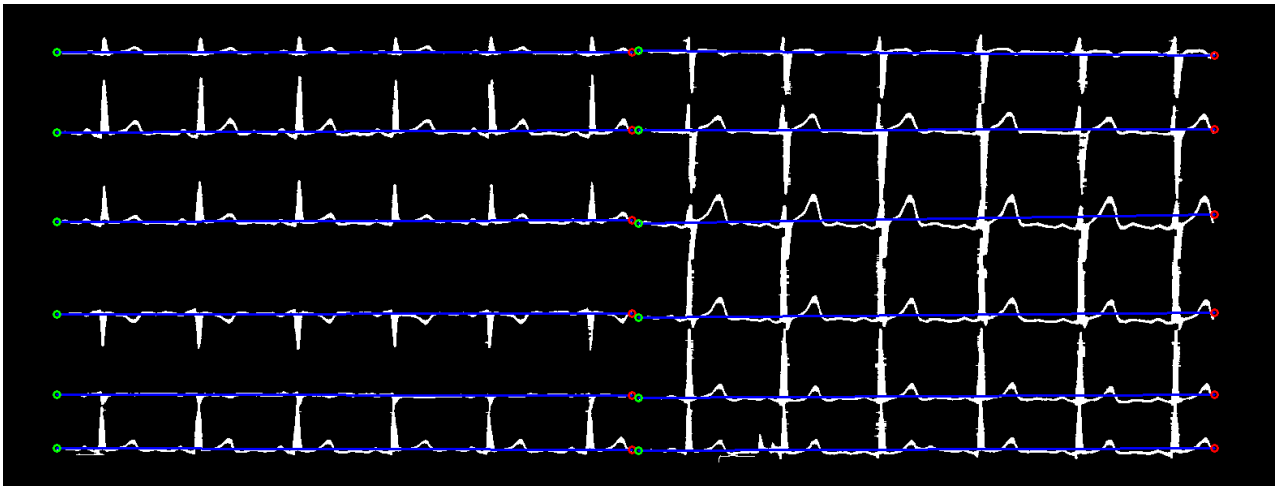


Figure 4.40: Visualisation des douze p_{start} (en vert) et des douze p_{end} (en rouge) reliés par une ligne (en bleu).

Il est intéressant de noter qu'un ECG valide et correctement binarisé est censé retourner douze p_{start} et douze p_{end}, constituant ainsi une condition que l'image d'entrée doit satisfaire avant d'être numérisée.

Pour effectuer la segmentation approximative, il va falloir suivre les étapes suivantes en partant de l'image binaire et de l'ensemble P_{start} et P_{end} :

- Parcourir l'ensemble P_{start} en prenant à chaque fois un élément p_{start} et P_{end} en prenant un élément p_{end} ;
- Déterminer une borne supérieure égale à la hauteur maximale de l'enveloppe convexe si la dérivation est la première dérivation dans sa colonne, sinon prendre le précédent p_{start} comme borne inférieure ;
- Déterminer une borne inférieure égale à la hauteur minimale de l'enveloppe convexe si la dérivation est la dernière dérivation dans sa colonne, sinon prendre le précédent p_{end} comme borne inférieure ;
- Tracer un rectangle borné verticalement par les abscisses de la borne supérieure et inférieure ; borné horizontalement par les coordonnées de p_{start} et p_{end} ;
- Découper ce rectangle de l'image binaire vers une nouvelle image.

Voici un exemple sur trois dérivations :

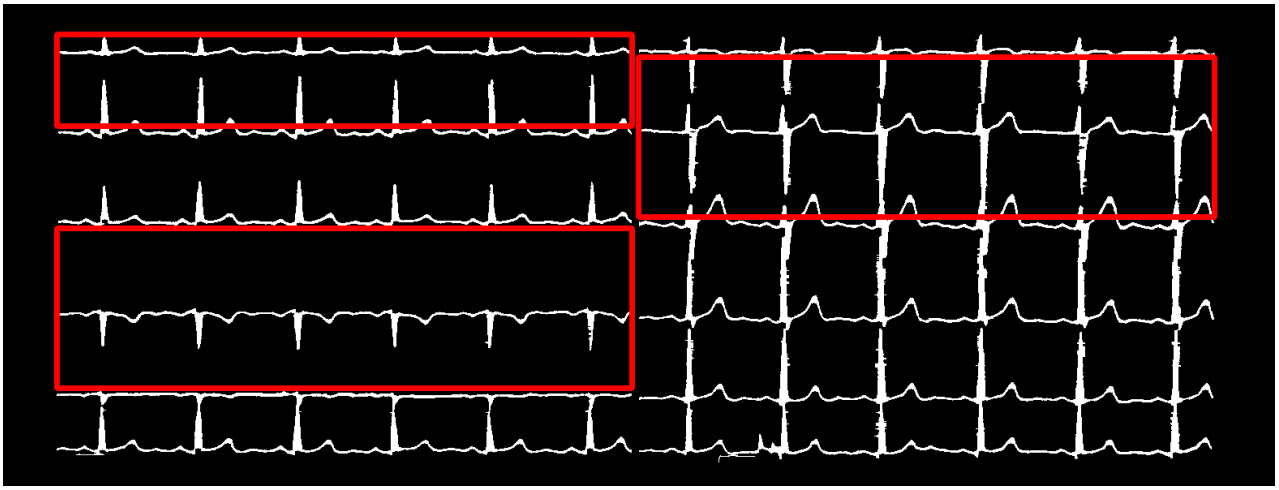
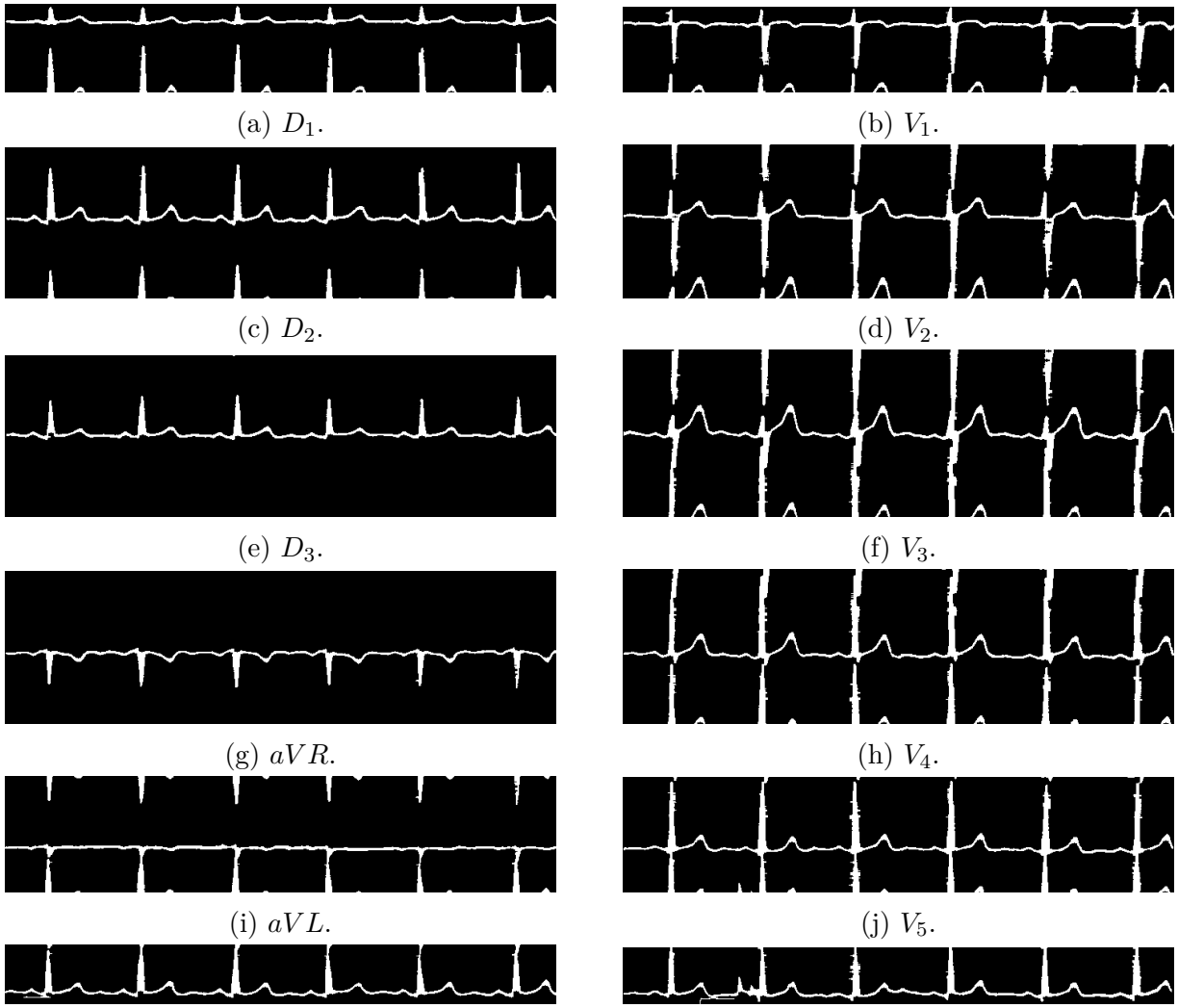


Figure 4.41: Visualisation de la segmentation approximative en rouge sur trois dérivations.

En appliquant cette méthode, la segmentation suivante en résulte :



(a) D_1. (b) V_1.

(c) D_2. (d) V_2.

(e) D_3. (f) V_3.

(g) aVR. (h) V_4.

(i) aVL. (j) V_5.

(k) aVF. (l) V_6.

Figure 4.42: Segmentation approximative.

Nous venons de rétrécir l'intervalle entre les bornes supérieure et inférieure verticales et horizontales de chaque rectangle contenant un signal de dérivation parmi les douze dérivation de l'image binaire, le but étant d'arriver à un rectangle minimal contenant chaque signal de dérivation.

4.4.4 Segmentation précise

La segmentation précise vise à isoler l'ensemble des pixels objets appartenant seulement au signal de dérivation segmenté.

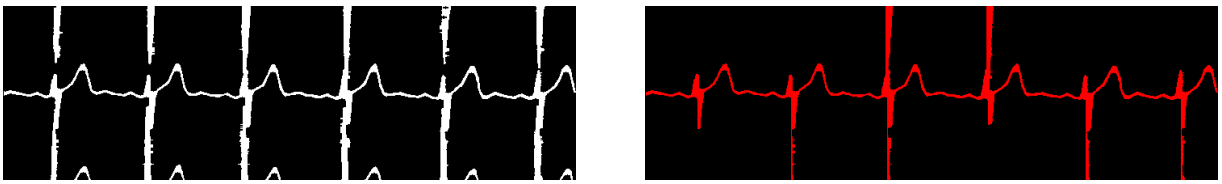
Pour se faire, nous allons démarrer du p_{start} et essayer de trouver un chemin vers p_{end} contenant exactement tous les pixels objet du signal de dérivation.

La première étape consiste à implémenter l'algorithme du parcours en largeur d'un graphe (BFS, pour Breadth-First Search en anglais) de la manière suivante :

1. Initialiser une file F possédant le pixel de départ ($F = p_{\text{start}}$) ;
2. Initialiser un ensemble $V =$ vide qui représente les pixels visités ;
3. Déclarer un pixel p dont la valeur vaut nil ;
4. Tant que F n'est pas vide ou que p est différent p_{end} :
   (a) Défiler de F un pixel p qui deviendra le pixel courant ;
   (b) Prendre par rapport à p :
      i. Le pixel au dessus ;
      ii. Le pixel au dessus à droite ;
      iii. Le pixel à droite ;
      iv. Le pixel en dessous à droite ;
      v. Le pixel en dessous.

      Ce qui revient à prendre les pixels immédiatement voisins de p de gauche à droite.
   (c) Vérifier la validité de ces pixels indiviuellement, c'est-à-dire s'ils ne sortent pas des bords de l'image, s'ils n'appartient pas à V et s'ils sont des pixels objets ;
   (d) Ajouter p à l'ensemble V (marquer p comme étant visité) ;
5. Si le dernier élément de l'ensemble V est égal à p_{end}, alors un chemin est retrouvé. Sinon, il n'existe aucun chemin vers p_{end}.

En appliquant cette méthode sur le signal de dérivation de la V_3 en coloriant les pixels visités en rouge, nous obtenant par exemple ce chemin :



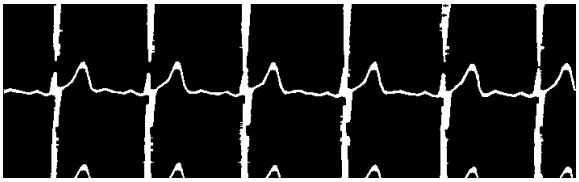
(a) Entrée. (b) Résultat.

Figure 4.43: Algorithme de parcours en largeur sur un signal de dérivation binaire.

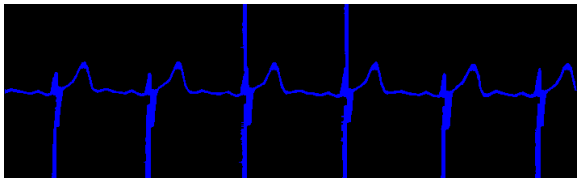
Nous remarquons déjà un avancement, les pixels objets déconnectées de la V_3 ont été supprimés.

Il est très important de noter que ce choix de déplacements implique que les chemins trouvés sont les ceux partant de gauche à droite sans reculer.

En se basant sur ce fait, voyons le résultat obtenu après avoir réitéré le parcours en autorisant cette fois-ci les déplacements partant de droite à gauche et en inversant les pixels de départ et d'arrivée (les chemins étant représentés en bleu) :



(a) Entrée.

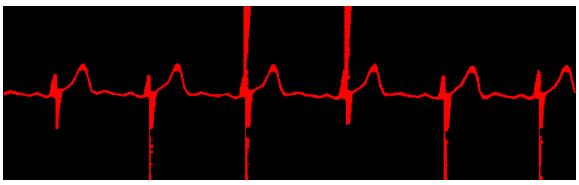


(b) Résultat.

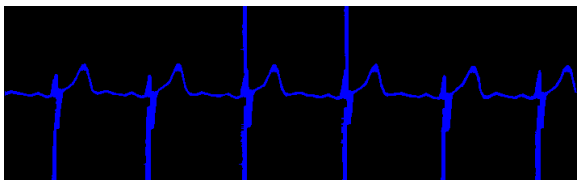
Figure 4.44: Algorithme de parcours en largeur inverse sur un signal de dérivation binaire.

Les chemins compris dans l'intersection entre les chemins de gauche à droite sans reculer, et de droite à gauche sans reculer sont les chemins qui permettent de se déplacer de p_{start} vers p_{end} et inversement. C'est à dire des chemins qui parcourent l'ensemble des pixels objets appartenant seulement au signal de dérivation segmenté.

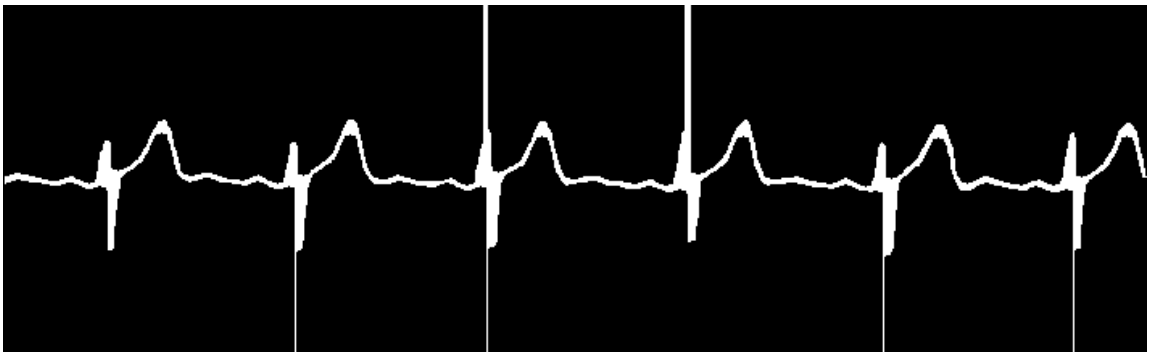
Ces chemins obtenus s'obtiennent à travers un ET logique entre les deux parcours comme il est montré ci-dessous :



(a) Parcours de gauche à droite.



(b) Parcours de droite à gauche.



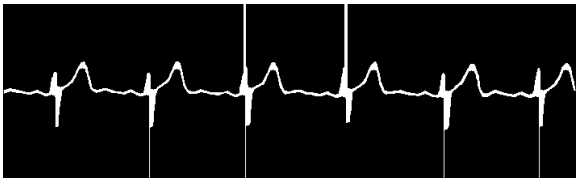
(c) ET logique entre les deux.

Figure 4.45: ET logique entre les parcours de gauche à droite et de droite à gauche.

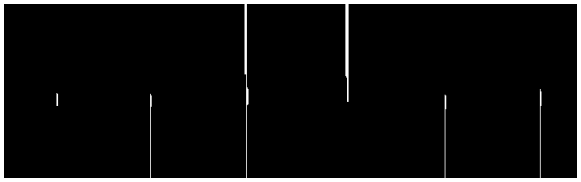
Il reste des pixels objet du chevauchement car les pics des parties de signaux externes à la derivation se sont refermés sur eux-même de gauche à droite à cause de la dilatation et sont donc considérés comme la continuité de la dérivation segmentée.

Ces restes sont les chemins qui, à partir d'un certain point p, n'atteignent pas le pixel objet d'arrivée et finissent sur un parcours purement vertical.

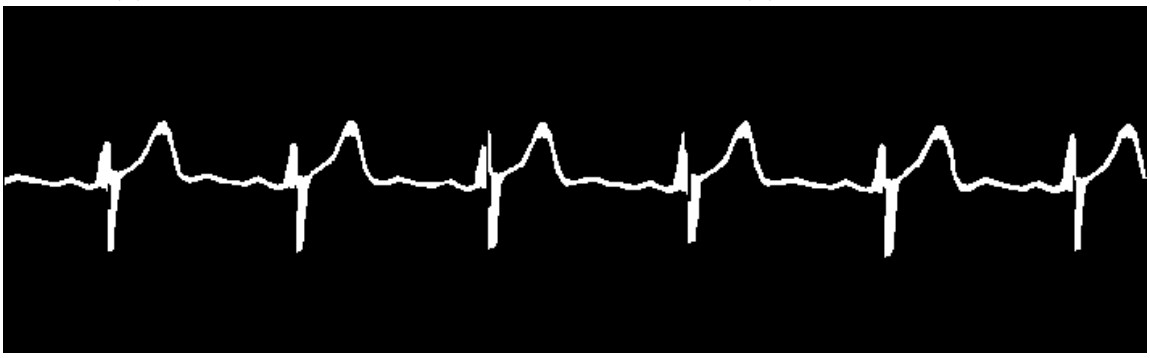
Il va falloir éliminer ces chemins inutiles en effectuant une opération morphologique d'érosion avec un élément structurant rectangulaire d'un pixel de large et d'un quart de la hauteur de la dérivation de haut, obtenant une nouvelle image binaire qui élimine les chemins purement verticaux :



(a) Résultat précédent.



(b) Résultat de l'érosion.



(c) Soustraction de l'érosion du résultat précédent.

Figure 4.46: Élimination des restes.

Cette étape nous a permis d'obtenir deux nouveaux nombres qui sont $y_{\max}$ et $y_{\min}$ qui représentent respectivement l'ordonnée maximale et minimale d'un pixel objet appartenant au signal de dérivation segmenté.

Autrement dit, l'ensemble des pixels objets $p(x, y)$ appartement seulement à ce signal possèdent tous une ordonnée $y_{\min} \leq y \leq y_{\max}$.

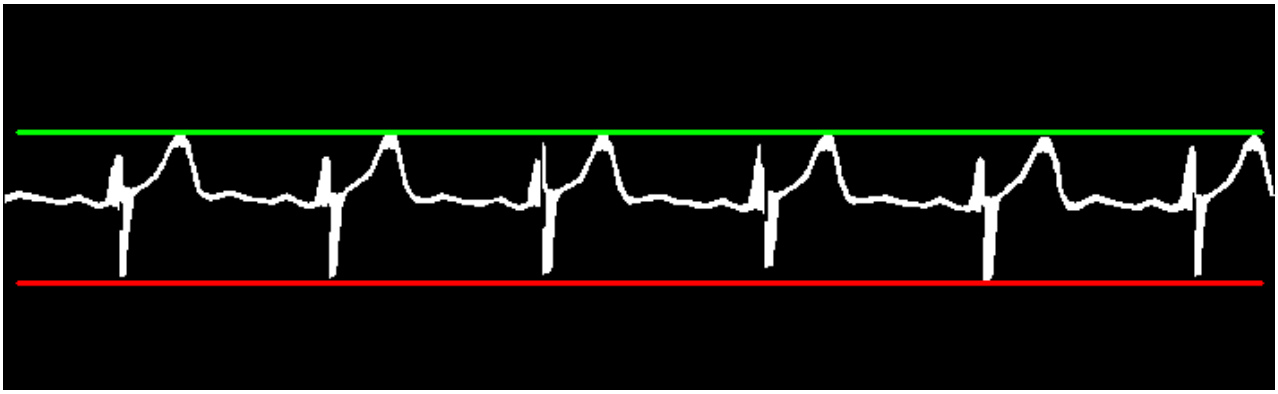
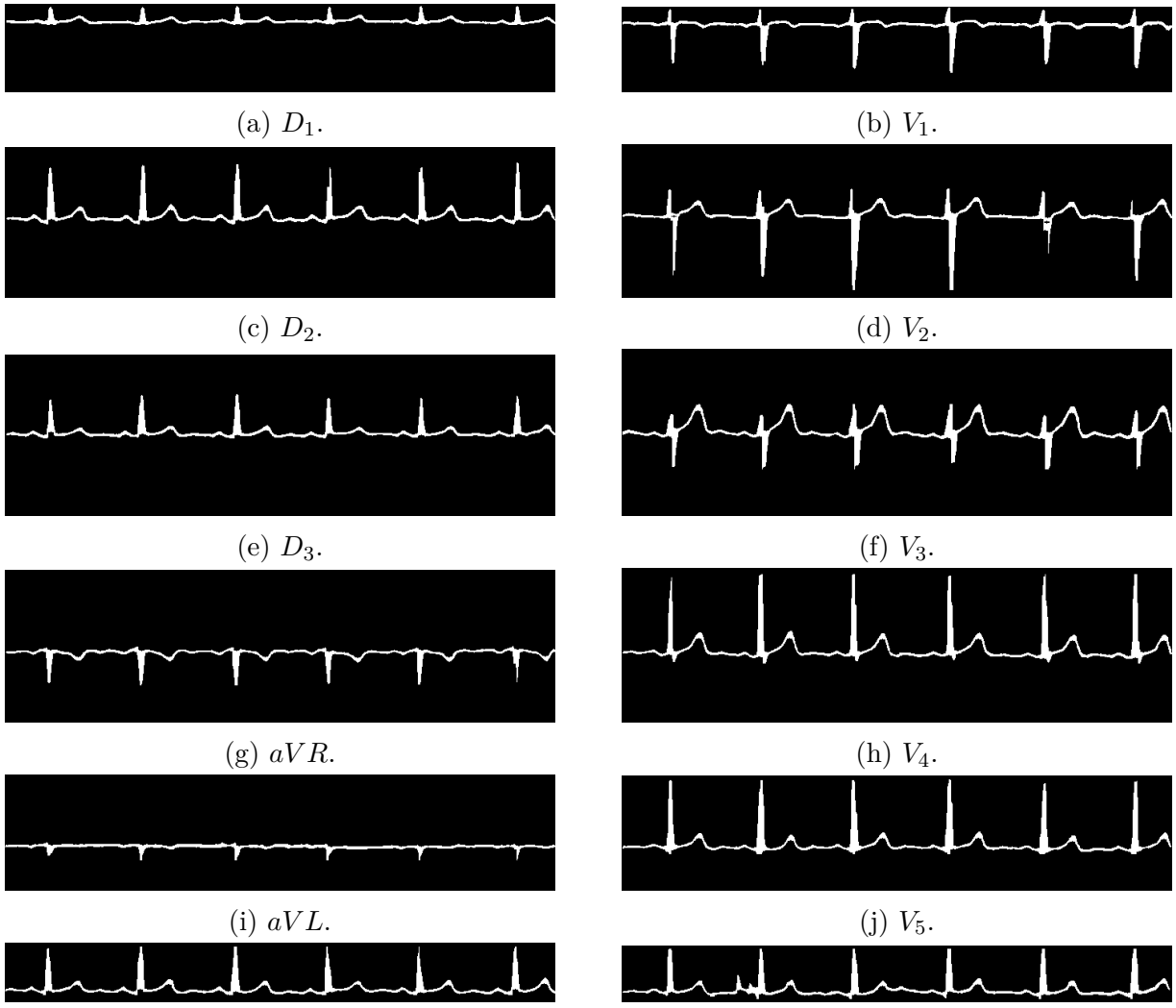


Figure 4.47: Bornes maximale $y = y_{\max}$ en vert et minimale $y = y_{\min}$ en rouge.

En reprenant l'ensemble des pixels objets de l'intersection du parcours en largeur de droite à gauche et de gauche à droite et en filtrant les pixels objets d'ordonnées en dehors des bornes supérieures et inférieures, nous arrivons au résultat suivant :



(a) D_1. (b) V_1.

(c) D_2. (d) V_2.

(e) D_3. (f) V_3.

(g) aVR. (h) V_4.

(i) aVL. (j) V_5.

(k) aVF. (l) V_6.

Figure 4.48: Segmentation précise sans chevauchement vertical.

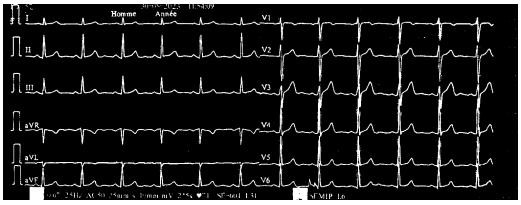
La segmentation précise a été effectuée avec succès. La prochaine étape consiste à la numérisation des signaux de dérivations obtenus.

4.5 Numérisation

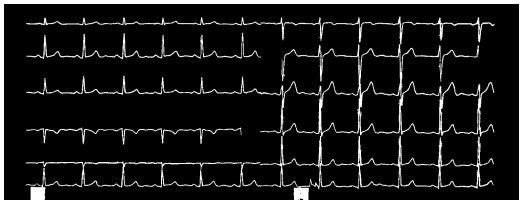
Avant de commencer la numérisation, il va falloir détecter au moins un rectangle d'étalonnage qui, une fois étudier, nous permettra de déduire par combien de pixels est représente un centimètre sur les images précédemment trouvées. Il est important de se rappeler que le rectangle d'étalonnage est deux fois plus haut que large.

En reprenant l'étape du seuillage global de la binarisation I_{raw}, nous allons calculer I_{raw} moins l'image "nettoyée" des petites surfaces, qui pour rappel, représentait seulement les pixels objets appartenant aux signaux de dérivation. Obtenant une nouvelle image qui est

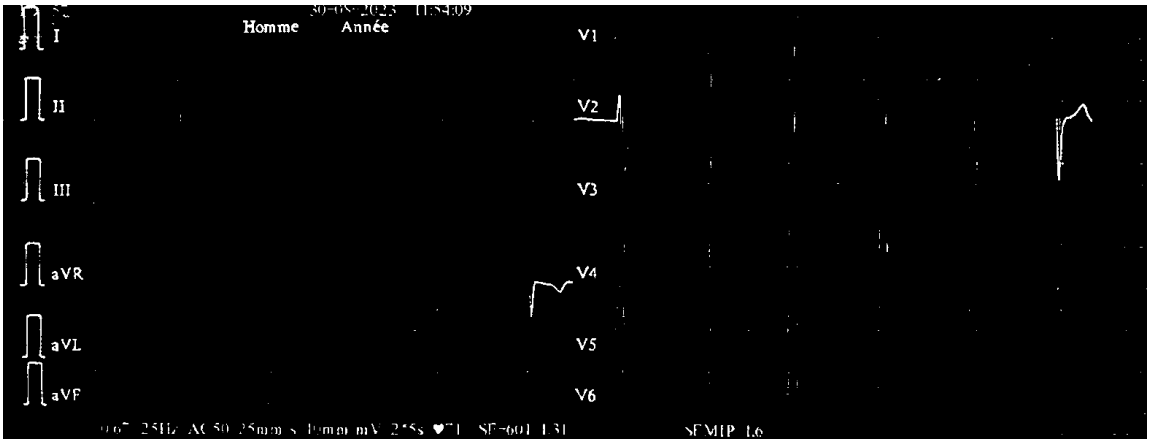
censée retourner tous les pixels objets de I_{raw} sauf la majorité des pixels objets des signaux de dérivation :



(a) Image binaire sans dilatation.



(b) Image binaire nettoyée.



(c) Différence.

Figure 4.49: Rectangles d'étalonnage et petits objets isolés.

En réutilisant la technique de détection des contours, nous allons sélectionner les rectangles minimaux de contours dont la hauteur est à peu près deux fois plus grande que la largeur. Plus exactement, pour un contour dont le rectangle minimal possède pour largeur l et hauteur h, nous prendrons seulement les contours qui vérifient $1,9 < \frac{h}{l} < 2,1$. Il faut veiller à prendre les douze plus grand contours pour éviter les petits amas de pixels qui sont des restes de signaux ou de grilles qui risquent de donner de faux positifs.

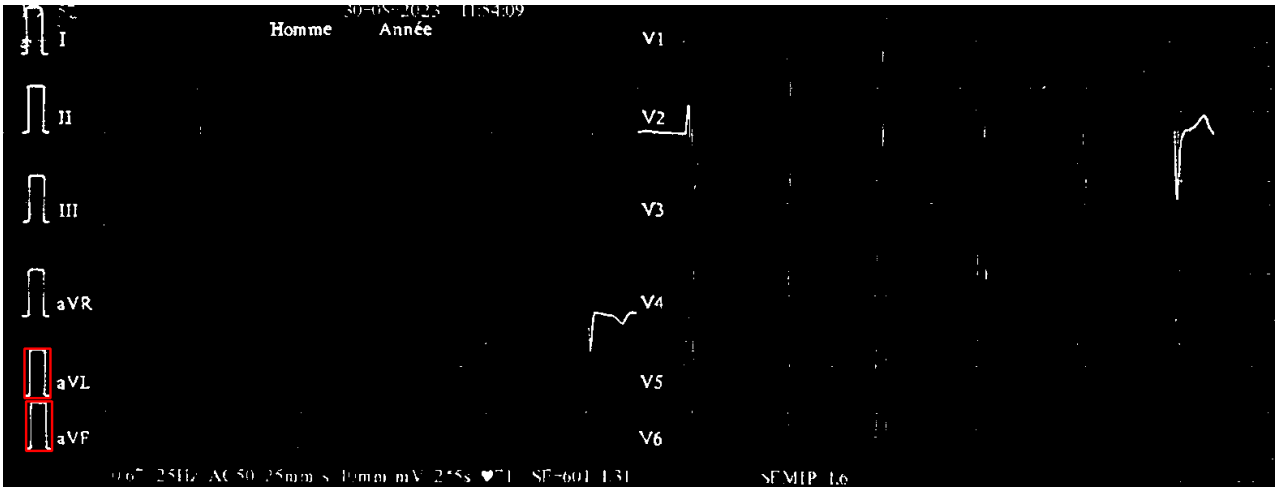


Figure 4.50: Détection des rectangles d'étalonnage.

Dans ce cas, les deux rectangles possèdent la même hauteur et la même largeur, ils sont donc équivalent. Mais idéalement, il serait préférable de prendre le contour dont le ratio $\frac{h}{l}$ est le plus proche de 2.

Prenons par exemple le premier retrouvé. Il possède une résolution spatiale de 32×61 pixels. Sa hauteur valant 61 pixels, on imagine donc que $2cm$ valent 61 pixels et donc $1cm = 30,5$ pixels.

Figure 4.51: Segmentation d'un rectangle d'étalonnage.

Ce résultat est faux, car il n'est pas assez précis. Déjà, nous ne pouvons obtenir que des valeurs entières positives ou des entiers positifs + $\frac{1}{2}$. Cela revient au fait qu'une résolution spatiale d'une image est un nombre entier positif et que l'on souhaite la diviser par deux.

Un changement de 0,5 affecte grandement la numérisation, que ce soit dans les valeurs de voltage ou de temps. Il faudra trouver un moyen d'améliorer la précision de cette procédure.

Cependant, un problème s'impose : un pixel est en quelques sortes un élément atomique d'une image, il n'est donc pas divisible en "sous-pixels" de coordonnées réelles.

L'idée que nous avons eu consiste à prendre un nombre positif f assez grand, $f = 100$ par exemple, et à agrandir le rectangle d'étalonnage par ce facteur f dans les deux résolutions gardant ainsi ses proportions exactes (et donc le même ratio). La nouvelle image obtenue est de résolution spatiale 3200×6100.

Il est obligatoire d'utiliser une interpolation de type "plus proche voisin" lors du redimensionnement afin d'obtenir une image de rectangle d'étalonnage binaire.

On remarque aussi qu'avant le redimensionnement, le rectangle d'étalonnage avait une certaine épaisseur (supérieure à un pixel) et donc était propice à l'ambiguïté concernant sa hauteur réelle.

Pour atténuer cette ambiguïté, nous allons prendre la colonne centrale de l'image binaire du rectangle agrandi et calculer la moyenne de la nouvelle épaisseur du rectangle e_{med} et soustraire la quantité $2e_{\text{med}}$ de sa résolution verticale. Il est impératif de multiplier l'épaisseur par deux car les rectangles d'étalonnage possèdent toujours des continuations sur les coins du bas vers l'extérieur afin de les différencier des lignes verticales de la grille.

Cette amélioration de la précision a généré, de la même manière qu'avant le redimensionnement, un nouveau nombre qui représente le nombre de pixels par centimètre $px/cm = 29,005$. Plus f est grand, plus px/cm devient précis, cependant, une valeur exagérée de f

risque de rendre le programme trop gourmand en ressources et lent en exécution.

Nous allons définir deux nouveaux nombres : s/px qui est le résultat de la division du nombre de secondes par centimètre sur l'ECG réel (s/cm) par px/cm et mV/px qui est le résultat d'une division du nombre de millivolts par centimètre sur l'ECG réel (mV/cm) par px/cm.

4.5.1 Extraction des points clés

Comme nous l'avons vu plus tôt, la dilatation lors de la binarisation a refermé les espaces horizontaux entre la partie ascendante et descendante d'un pic rendant la numérisation difficile.

Si le signal ne faisait qu'un pixel d'épaisseur, il aurait suffit de prendre les coordonnées de chaque chaque pixel objet et de les considérer comme des points appartenant au signal.

La numérisation devient difficile car elle se base sur les coordonnées des pixels objets, et lorsqu'il y en a autant sur une même colonne, sélectionner le bon pixel objet pour le voltage n'est pas évident. Prenons pour exemples des méthodes de sélection d'ordonnées de pixels objets et leur défaut :

- Prendre la moyenne des ordonnées : efficace sur les parties plates ou d'épaisseur minimale sur le signal, mais divise la hauteur des pics en deux et est, de ce fait, loin d'être précise (50% de précision sur tous les pics).
- Prendre l'ordonnée maximale : efficace sur les pics ascendants, mais prends une valeur proche de 0 sur les pics descendants et les ignore donc totalement.
- Prendre l'ordonnée minimale : même problème que la méthode de la sélection de l'ordonnée maximale.
- Squelettisation : une squelettisation (thinning en Anglais) par l'algorithme de Zhang [41] va tout simplement réduire le signal à un T inversé. C'est à dire un saut vers la valeur extrême du pic en partant verticalement de la ligne des $0mV$ sur un temps précis t.

La solution consiste à restituer les pics des signaux sans altérer leurs formes. À partir des images de la segmentation I_{segment}, nous allons déduire deux nouvelles images I_{hi} et I_{lo} comme suit :

- I_{hi} : translation d'un pixel vers le bas suivant l'axe vertical ;
- I_{lo} : translation d'un pixel vers le haut suivant l'axe vertical ;

Ensuite, nous allons calculer $I_{\text{new}} = I_{\text{segment}}$ - I_{hi} - I_{lo} obtenant l'image suivante sur la V_6 :

Figure 4.52: Amincissement du signal et division en deux parties verticalement (I_{new}).

Le signal se retrouve divisé en deux parties, la partie haute issue de I_{segment} - I_{lo}, et la partie basse issue I_{segment} - I_{hi}. Nous remarquons que cette méthode ne change rien à un signal d'épaisseur d'un pixel et est donc adaptée à toutes les images binaires de signaux.

En reprenant le p_{start} du signal, apporter les modifications à l'image suivant ces étapes :

1. Calculer la moyenne de la hauteur des pixels partageant la colonne de p_{start} pour obtenir une ordonnées y_{baseline} qui représentera la valeur 0 mV ;
2. Compléter l'image par une ligne droite horizontale allant de y_{baseline} jusqu'au bord gauche de l'image pour obtenir des signaux de même résolution horizontale (équivalent à avoir des signaux de même durée t) ;
3. Effectuer les même étapes pour p_{end} en complétant par une ligne allant, cette fois, jusqu'au bord droit de l'image.

Après avoir initialisé un ensemble de points P vide, un parcours de l'image colonne par colonne est effectué, en prenant toutes les ordonnées des pixels objets présents sur cette colonne P_y et calculer :

- $p_{\text{hi}} = \max(P_y) - 1$;
- $p_{\text{lo}} = \min(P_y) + 1$.

L'addition et la soustraction de 1 annule l'effet des translations précédentes.

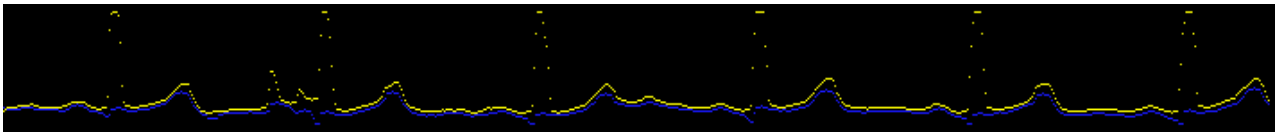


Figure 4.53: Détection des p_{hi} en jaune et des p_{lo} en bleu.

La prochaine étape consiste à calculer une sorte de dérivée, qui en réalité est un moyen de quantifier la pente entre un p_{hi} ou un p_{lo} et son antécédent dans l'ensemble P_y par la formule $y_{\text{slope}} = \mid p_i - p_{i-1} \mid$.

Calculer $y_{\text{selected}} = \max(y_{\text{slope}}(\text{hi}), y_{\text{slope}}(\text{lo}))$, qui représente l'ordonnée dont la dérivée en ce point est maximale. Voici une représentation des y_{selected} en vert s'ils sont issus d'un p_{hi}, sinon en vert s'ils sont issus d'un p_{lo} :

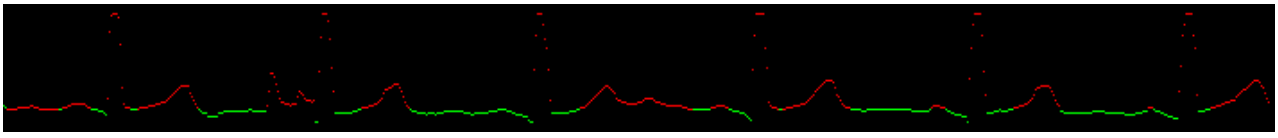


Figure 4.54: Sélection des ordonnées.

Il faudra soustraire de y_{baseline} cette valeur obtenue les ordonnées choisies afin de retrouver la hauteur relative à la valeur 0 mV, retrouvant ainsi $y_{\text{relative}} = y_{\text{baseline}} - y_{\text{selected}}$.

Le point $p(x, y)$ est ajouté à l'ensemble P tel que :

- $x = c \cdot s/px$, avec c étant la colonne courante ;
- $y = y_{\text{relative}} \cdot mV/px$.

Le signal a été reconstruit en un ensemble de points clés P qui sont less points réels du signal sur l'ECG.

4.5.2 Interpolation

L'interpolation est une méthode qui vise à estimer une fonction qui passe par un ensemble donné de points (x_i, y_i), souvent dérivés de données expérimentales. L'objectif est de trouver un modèle mathématique calculable, intégrable, et dérivable permettant de représenter ces données [42].

Il existe deux principales approches :

- **Interpolation :** Le modèle est exact pour tous les points de l'ensemble donné.
- **Régression (ou lissage) :** Le modèle est optimisé pour s'ajuster au mieux à l'ensemble des points, sans nécessairement passer par chacun d'eux.

Interpolation Linéaire

L'interpolation linéaire consiste à approcher une courbe par une application affine entre deux valeurs successives des abscisses x_i et x_{i+1}. Pour deux points (x_a, y_a) et (x_b, y_b), où $x_a < x_b$, la fonction d'interpolation linéaire est donnée par :

$$f(x) \approx y_a + \frac{y_b - y_a}{x_b - x_a} \times (x - x_a)$$

Remarque : Pour une valeur de x en dehors de l'intervalle $[x_a, x_b]$, on parle d'extrapolation.

Interpolation Polynômiale

Selon le théorème de Weierstrass, toute fonction continue peut être approximée uniformément par un polynôme. Si l'on dispose d'une suite de n couples (x_i, y_i), il existe un polynôme unique de degré $n - 1$ passant par tous ces points.

Définition : L'interpolation polynômiale consiste à approcher une fonction f dont on connaît n points par un polynôme de degré $n - 1$.

Erreur : L'erreur d'interpolation dépend de la fonction initiale, du nombre de points utilisés, et de leur répartition sur l'intervalle. Une interpolation par un polynôme de trop haut degré peut entraîner des oscillations indésirables, connues sous le nom de phénomène de Runge.

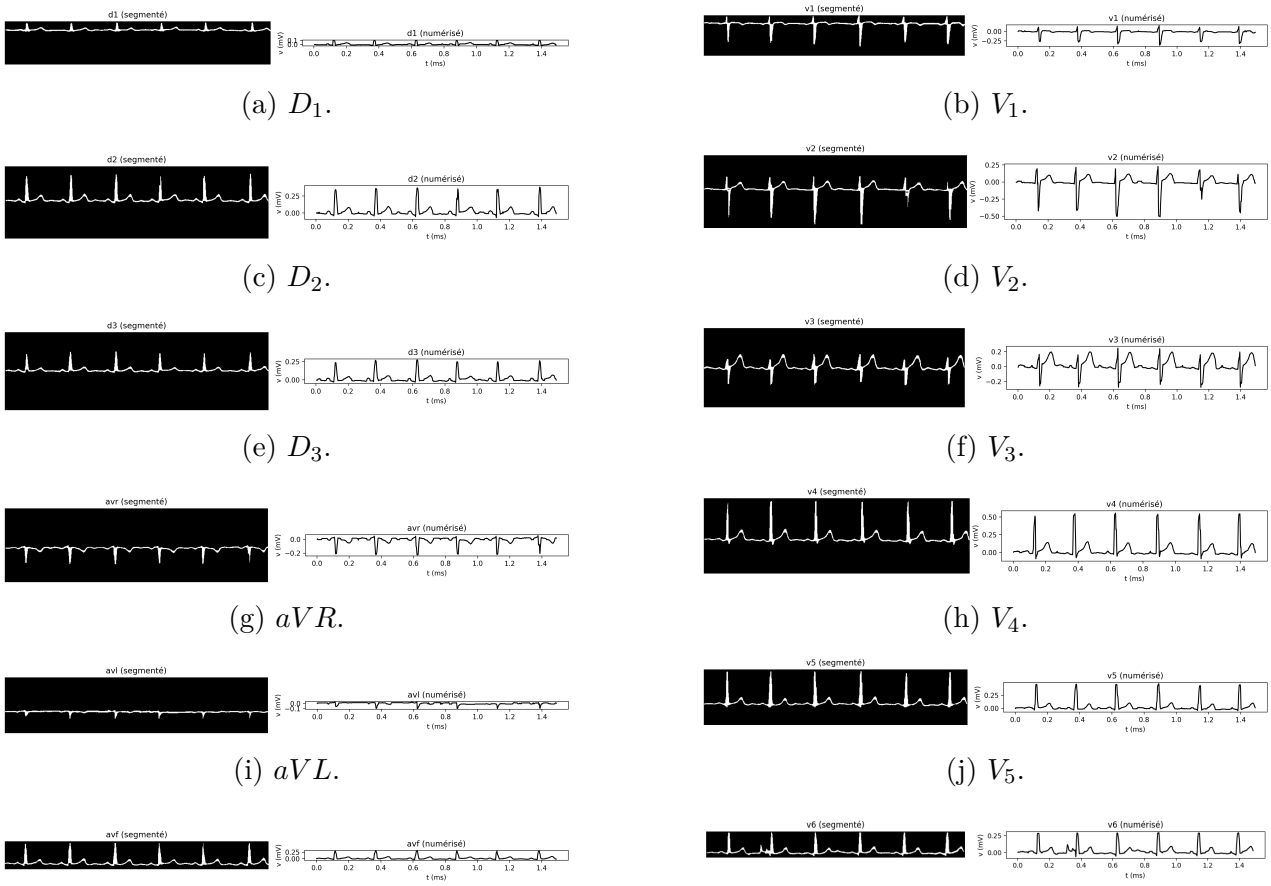
Splines

Il est souvent préférable de découper l'intervalle initial en plus petits intervalles et d'interpoler la fonction par des polynômes de faible degré sur chacun de ces intervalles. C'est le principe des fonctions splines, qui permettent de minimiser les erreurs d'interpolation tout en assurant une transition fluide entre les segments.

4.5.3 Interpolation de l'ECG

L'ensemble X, de cardinalité inférieure ou égale à la largeur de l'image, ainsi que l'ensemble Y valent respectivement les abscisses et ordonnées de l'ensemble des points P obtenus plus tôt.

En utilisant l'interpolation cubique avec un échantillonage de 2000 points sur les ensembles X et Y, nous obtenons deux nouveaux ensembles X_I et Y_I, tracés sur un repère orthonormé :



(a) D_1. (b) V_1.

(c) D_2. (d) V_2.

(e) D_3. (f) V_3.

(g) aVR. (h) V_4.

(i) aVL. (j) V_5.

(k) aVF. (l) V_6.

Figure 4.55: Numérisation par interpolation cubique.

Le type d'interpolation importe peu. Les mêmes résultats sont observés à cette échelle en utilisant l'interpolation linéaire avec le même échantillonage.

4.6 Résultats et discussions

Nous allons discuter des résultats obtenus suivant la précision du programme, les ressources utilisées et la portabilité des données obtenues :

4.6.1 Précision

L'ECG que nous avons pris pour exemple tout le long du processus de numérisation possède comme échelle de voltage $10mm/mV$ (équivalent à $1mV/cm$) et une échelle de temps de

$25mm/s$ (équivalent à $0,4s/cm$).

La longueur des dérivations est de $24,5cm$ soit $9,8s$.

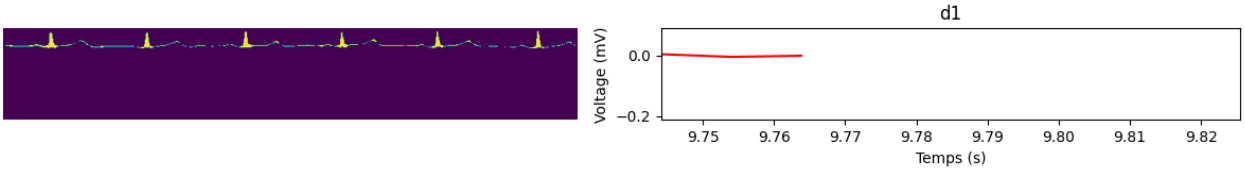


Figure 4.56: Précision sur l'axe de temps de la numérisation.

La valeur observée est entre $9,76$ et $9,77s$. La valeur réelle étant $9,76383382175487$. Cela démontre une erreur absolue de $0,036166178245130354cm \approx 0,036cm = 0,36mm$ et une précision de $99,63095736484561\% \approx 99,63\%$.

Quant au voltage, sa précision varie le long de l'ECG avec une erreur absolue allant jusqu'à $0,2mV$.

Nous remarquons cependant la perte presque totale du 5^{e} pic négatif de la V_2 et la perte partielle du 4^{e} pic positif de la D_2, ceci est dû à :

- L'algorithme de suppression du chevauchement vertical sur des signaux légèrement coupés ;
- La présence de trous (pixels d'arrière-plan entourés de pixels objets) inclus dans le signal lors de la binarisation ;
- Prise de trop de pixels lors de la binarisation (dû au rehaussement) ou pas assez (dû à au traitement par la transformée de Fourier d'un point maximal ou minimal d'un pic qui coïncide avec une ligne de la grille) ;
- Échantillonnage imprécis dans les images à petite résolution spatiale, autrement dit, un pixel objet change de beaucoup le voltage.

4.6.2 Complexité algorithmique

La complexité d'un algorithme est une prédiction ou une garantie que l'algorithme ne prendra jamais plus qu'un certain nombre d'étapes ou opérations (dans le cas de la complexité temporelle) ou un certain espace mémoire (dans le cas de la complexité spatiale), qui dépend souvent de la taille des données n qu'il manipule. On note en général n cette taille et on cherche $C(n)$ qui représente le nombre maximum d'opérations et dépend de l'algorithme [43].

Le paramètre n vaut tout simplement la résolution spatiale de l'image d'entrée. Par contre, vu les spécificités de chaque langage de programmation et le nombre d'instructions des programmes ayant servi au scan et à la numérisation de l'ECG, il est difficile de retrouver précisément ou exactement la valeur de $C(n)$. Cependant, nous allons effectuer des estimations sur chacune des étapes du scan et de la numérisation de l'ECG.

Complexité temporelle

La complexité temporelle d'un algorithme est une information sur son temps d'exécution liée au volume n de données à traiter. La complexité temporelle d'un programme est la complexité temporelle de l'algorithme associé [44].

Les programmes de scan et numérisation de l'ECG ont été implémentée en langage Python 3. Donc, afin de mesurer les performances de chaque procédure du programme, nous allons utiliser une fonction de la librairie `time` de Python 3 appelée `time.perf_counter()` qui, selon la documentation officielle [45], est définie comme suit :

> "Return the value (in fractional seconds) of a performance counter, i.e. a clock with the highest available resolution to measure a short duration. It does include time elapsed during sleep and is system-wide. The reference point of the returned value is undefined, so that only the difference between the results of two calls is valid."

Voici les temps d'exécution en millisecondes en ms (sur le processeur i5-6300U avec 8 Go de RAM) arrondis à deux chiffres après la virgule sur quatre exécutions consécutives des programmes du scan, de la binarisation, de l'étalonnage, de la segmentation, de la numérisation et enfin de l'interpolation au fil des itérations i_1, i_2, i_3 et i_4 :

Programme	i_1	i_2	i_3	i_4	Moyenne (en ms)	Pourcentage (%)
Scan	53.72	56.69	54.52	55.45	55.09	0.47
Binarisation	6479.91	5945.93	5953.00	6020.72	6099.89	52.03
Étalonnage	2925.56	2977.74	2692.06	2739.06	2833.60	24.19
Segmentation	1452.95	1301.19	1319.31	1378.03	1362.87	11.62
Numérisation	1335.77	1299.23	1227.74	1327.80	1300.14	11.08
Interpolation	10.35	8.52	8.64	8.58	8.77	0.07
Total	12258.26	11539.30	11375.27	11729.64	11725.62	-

Table 4.1: Complexité temporelle du scan et de la numérisation de l'ECG.

Le temps d'exécution total vaut en moyenne 11,7 secondes. Cependant, on remarque que le scan et l'interpolation à eux deux dépassent à peine 1% du temps d'exécution total. Nous pouvons les considérer comme des opérations peu gourmandes en ressources.

Complexité spatiale

La complexité spatiale d'un algorithme est une estimation de l'espace mémoire occupé au cours de l'exécution d'un programme en fonction du volume n de données à traiter. La complexité spatiale d'un programme est la complexité spatiale de l'algorithme associé [44].

Selon l'article "An Analysis of Memory Usage in Web Browser Software" [46], rien que le navigateur Web consommerait entre $134.67MB$ et $699.66MB$ de mémoire. Vu que nous ramenons la résolution spatiale de l'image traitée (après le scan) à une résolution spatiale prédéfinie, la taille $n = 1,35Mo$ de l'image de la binarisation jusqu'à l'interpolation :

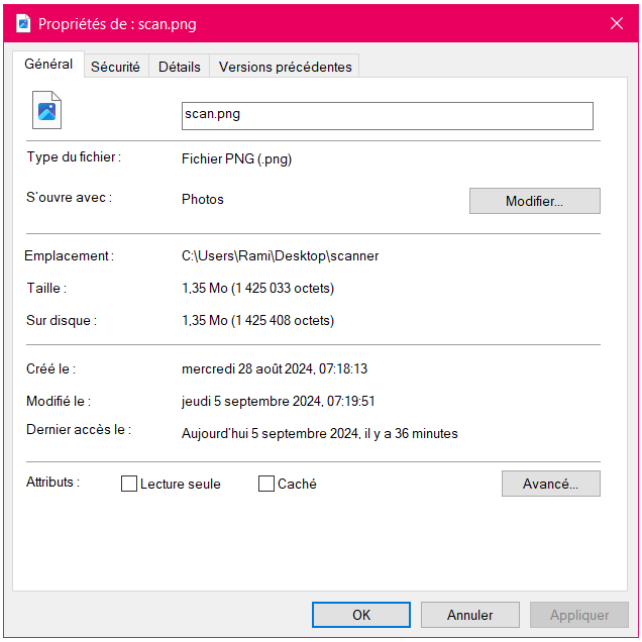


Figure 4.57: Calcul de la taille de l'image obtenue après le scan.

De ce fait, à moins que l'image ne soit clonée 100 fois pendant l'exécution d'une partie du programme, elle ne risque pas de consommer plus que le plus optimisé des navigateurs web. De plus, les opérations effectuées sur les images sont simples (convolutions, filtres linéaires, filtres non linéaires, etc) et nécessitent qu'une seule copie de l'image dans la RAM ainsi que quelques centaines d'octets dans le pire des cas pour effectuer les calculs requis.

L'étalonnage par redimensionnmenet du facteur $f = 100$ ne comporte pas de risque car même dans le pire des cas, le rectangle d'étalonnage est bornée verticalement d'un tiers de la hauteur de l'image (dans le cas d'une disposition à quatre colonnes et trois lignes) et est borné horizontalement d'un sixième de la largeur de l'image (la moitié de sa hauteur) et donc sa résolution spatiale, redimensionnée par le facteur f, maximale vaudrait $\frac{100 \cdot \text{taile de l'image}}{3 \times 6}$ soit environ $5,55 \times 1,35 Mo = 7,5 Mo$.

4.6.3 Portabilité

L'avantage de cette numérisation est que l'intégralité des tracés des signaux de dérivation peuvent être reconstruits seulement en partant de l'ensemble des points clés P.

Cela est dû au fait que l'interpolation ne dépend que de l'ensemble P et que c'est une opération largement supportée par presque toutes les librairies mathématiques de tous les langages de programmation. Et puis même si elle ne l'est pas, une implémentation de l'interpolation linéaire est facile à reproduire. Certes, elle prend comme paramètre le px/cm, s/cm et mV/cm mais ces valeurs ne sont que des scalaires servant à l'étalonnage des ensembles X et Y de P par une multiplication.

La simplicité de l'ensemble P fait sa force. En effet, il s'agit d'un vecteur (ou tableau) de tuples (x, y). Il peut aussi être exprimé sous forme de matrice à deux colonnes ou de manière récursive (par exemple, chaque y_i est exprimé en fonction de $y_{i-1} + c$, avec $c \in \mathbb{R}$).

Comme nous l'avons vu plus tôt, nous avons opté pour un enregistrement au format JSON. Cependant, cela n'empêche pas l'enregistrement sous format CSV [47] (RFC 4180) par exemple, qui est un type de fichier texte sans formatage particulier où chaque champ est séparé par une virgule et chaque tuple est séparé par une ligne. Considéré être comme une base de données recueillie, il est d'une excellente portabilité et est supporté par presque tous les langages de programmation que ce soit nativement ou à travers une librairie ainsi que les applications lourdes de tableaux et classeurs telles que l'Excel.

Cela facilite l'exploitation du Dataset en ne forçant aucun formatage particulier des données ECGs.

Conclusion

Ce chapitre a présenté une méthodologie complète pour la numérisation d'ECGs papier. Les résultats obtenus montrent que notre approche permet de convertir avec précision les signaux ECG analogiques en données numériques. Les différentes étapes du processus, du scan à la numérisation, ont été détaillées et évaluées.

Bien que les résultats soient encourageants, plusieurs pistes d'amélioration peuvent être envisagées. Notamment, l'étude de méthodes de réduction du bruit plus avancées et l'adaptation de l'algorithme à différents types de papier pourraient améliorer la qualité des résultats.

Conclusion générale

Ce mémoire a porté sur la conception et la réalisation d'une application dédiée à la numérisation des signaux ECG, en s'appuyant sur des technologies modernes telles que React pour le front-end, ExpressJS pour le back-end, et Prisma pour la gestion de la base de données. Nous avons adopté la méthodologie Agile Scrum, qui a permis une gestion itérative et collaborative du projet. De plus, une attention particulière a été portée à la conception à travers l'utilisation des diagrammes UML pour structurer l'architecture du système.

Tout au long de ce projet, nous avons mis en œuvre des techniques avancées de traitement d'images afin d'extraire, analyser et visualiser des signaux ECG à partir d'images. Ces étapes ont représenté un véritable défi technique, mais elles nous ont également permis d'approfondir nos compétences en traitement de signal et en algorithmie.

L'utilisation d'outils et de technologies modernes tels que le développement full-stack (React et ExpressJS) et la gestion de base de données avec Prisma a permis de garantir la modularité et l'évolutivité du projet. Le développement continu, accompagné de tests, a permis de corriger les anomalies au fur et à mesure, assurant ainsi un produit de qualité.

Cette expérience nous a permis de développer des compétences techniques solides et de mieux appréhender les défis liés au traitement d'images dans le cadre médical. Nous avons également acquis une compréhension approfondie des besoins des utilisateurs finaux et de l'importance de la précision et de la fiabilité dans ce type d'applications.

Perspectives

Bien que cette application soit déjà fonctionnelle et réponde aux besoins initiaux, plusieurs pistes d'amélioration et de développement peuvent être envisagées pour la rendre plus performante et plus complète :

- **Architecture distribuée** : Envisager une architecture serveur distribuée pour une meilleure scalabilité et pour supporter un grand nombre d'utilisateurs simultanément, répartissant ainsi la charge de traitement et augmentant la résilience du système.
- **Déploiement et CI/CD** : Intégrer un pipeline de déploiement continu (CI/CD) afin d'automatiser les mises à jour et les déploiements, garantissant une intégration fluide des nouvelles fonctionnalités et des correctifs sans interruption du service.
- **Analyse des signaux ECG numérisés** : Ajouter une fonctionnalité d'analyse automatique des signaux ECG numérisés, utilisant des algorithmes d'apprentissage automatique pour détecter automatiquement des anomalies ou des pathologies spécifiques, permettant ainsi un diagnostic préliminaire.

- **Support multilingue** : Étendre l'application avec un support multilingue, incluant des langues telles que l'anglais et l'arabe, pour répondre aux besoins d'une audience plus large et offrir une meilleure accessibilité.
- **Personnalisation du traitement d'image** : Permettre à l'utilisateur de personnaliser certaines étapes du traitement des images ECG en fonction de ses besoins spécifiques, notamment en ajustant les paramètres de filtrage et de segmentation.

En conclusion, cette expérience nous a non seulement permis de mettre en pratique nos connaissances théoriques, mais elle a également contribué de manière significative à notre développement professionnel. Nous restons motivés pour continuer à perfectionner cette application, en tenant compte des perspectives d'innovation et des évolutions technologiques à venir.

Réferences

[1] A. SIDER, *Introduction aux Architectures Informatiques*. Béjaia, Algérie: Université de Béjaia, 2020, Module : Technologies Web, Master 2 Génie Logiciel, Département Informatique.

[2] IBM Corporation, *Asynchronous javascript and xml (ajax) overview*, Last updated: 2021-03-04, IBM Corporation, 2021. [Online]. Available: https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=page-asynchronous-javascript-xml-ajax-overview.

[3] L. S. Gettes, "30 - électrocardiographie," in *Médecine interne de Netter*, M. S. Runge and M. A. Greganti, Eds., Paris: Elsevier Masson, 2011, pp. 221–233, ISBN: 978-2-294-70951-7. DOI: https://doi.org/10.1016/B978-2-294-70951-7.00030-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B978229470951700030X.

[4] I. Hssain, "Fiche 47 - électrocardiogramme4949.isabelle piedade.," in *Guide infirmier des urgences (deuxième édition)*, I. Hssain, Ed., deuxième édition, Paris: Elsevier Masson, 2015, pp. 278–281, ISBN: 978-2-294-73408-3. DOI: https://doi.org/10.1016/B978-2-294-73408-3.00047-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9782294734083000472.

[5] P. C. Scavée, "Rappels des principes fondamentaux en électrocardiographie," *Louvain Médical*, May 2018, Rubrique(s): Congrès UCL de Médecine Générale. [Online]. Available: https://www.louvainmedical.be/article/rappels-des-principes-fondamentaux-en-electrocardiographie.

[6] M. Benidir, *Théorie et traitement du signal, tome 1 : Représentation des signaux et des systèmes*. Paris: Dunod, 2002, ISBN: 978-2100059843.

[7] A. Boukaache, *Cours de traitement d'images et vision*, Niveau : Master 1, Automatique et Informatique Industrielle (AII), Université de 08 Mai 1945 Guelma, Faculté des Sciences et de la Technologie, Département d'Electrotechnique et Automatique, République Algérienne Démocratique et Populaire, Ministère de l'Enseignement Supérieur et de la Recherche Scientifique, 2016.

[8] P. Medical, "Ecg reader: Ai-powered expert interpretation app," *Powerful Medical Blog*, Aug. 2023. [Online]. Available: https://www.powerfulmedical.com/blog/ecg-reading-app-how-pmcardio-changes-cardiac-diagnosis-on-the-go/.

[9] Ministère du Travail, de la Santé et des Solidarités, *Les dispositifs médicaux (implants, prothèses...) : Mise sur le marché, surveillance, législation*, Mise à jour: 02.08.24, 2024. [Online]. Available: https://sante.gouv.fr.

[10] G. Ltd., *How to use the ecg app*, 2024. [Online]. Available: https://www.garmin.com/en-US/p/ECGApp.

[11] H. Marchat, *Analyse des besoins: La gestion des projets par étape - 1ère étape* (Les Livres outils. Gestion de projet). Paris: Eyrolles, 2008, 236 pp., ISBN: 2212541449.

[12] C. Solnon, *Modélisation uml*, INSA de Lyon - 3IF, 2014.

[13] IBM, *Création d'un modèle relationnel*, IBM, 2021. [Online]. Available: https://www.ibm.com/docs/en/cognos-analytics/11.1.0?topic=designer-cube-cognos.

[14] K.-B. Yue, *Mapping uml class diagrams to the relational model*, Course CSCI5333, University of Houston-Clear Lake, 2020. [Online]. Available: http://dcm.uhcl.edu/yue/courses/csci5333/Spring2020/notes/model/MappingUMLToRelationalModel.html.

[15] Wikipédia, l'encyclopédie libre. "Conception de logiciel. "[Online]. Available: https://fr.wikipedia.org/wiki/Conception_de_logiciel.

[16] S. Nacih, "L'application de la méthode scrum dans les projets de conception et de fabrication d'équipements industriels," p. 9, Jul. 2023.

[17] Agile Alliance. "The 12 principles behind the agile manifesto. "[Online]. Available: https://www.agilealliance.org/agile-essentials/#principles.

[18] K. Schwaber and J. Sutherland, *Le guide scrum™, Le guide de référence de scrum: Les règles de jeu*, Version française, Nov. 2017. [Online]. Available: https://scrumguides.org/.

[19] R. G. Yende, *Support de cours de genie logiciel*, HAL Id: cel-01988734, Jan. 2019. [Online]. Available: https://hal.science/cel-01988734.

[20] R. Fielding, "Architectural styles and the design of network-based software architectures," Chapters 5 and 6 focus on REST, including the derivation of REST, client-server constraints, statelessness, and the uniform interface., Ph.D. dissertation, University of California, Irvine, 2000, ch. 5 & 6.

[21] C. Desprez. "Sites vitrines : Exemples, objectifs et conseils. "[Online]. Available: https://www.shopify.com/fr/blog/sites-vitrines-exemples-objectifs-conseils.

[22] M. Jones, J. Bradley, and N. Sakimura, *JSON Web Token (JWT)*, Internet Engineering Task Force, May 2015. DOI: 10.17487/RFC7519. [Online]. Available: https://www.rfc-editor.org/info/rfc7519.

[23] E. T. Bray, *The javascript object notation (json) data interchange format*, Request for Comments 8259, Internet Engineering Task Force (IETF), Obsoletes RFC 7159, Standards Track, ISSN 2070-1721, Dec. 2017. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8259.

[24] L. P. Deutsch and J.-L. Gailly, *Zlib compressed data format specification version 3.3*, Request for Comments (RFC), Informational, May 1996. [Online]. Available: https://www.rfc-editor.org/rfc/rfc1950.

[25] R. C. Gonzalez and R. E. Woods, *Digital Image Processing, Global Edition*, English. PEARSON, 2018, ISBN: 1292223049.

[26] IBM. "Qu'est-ce que la computer vision ? "[Online]. Available: https://www.ibm.com/fr-fr/topics/computer-vision.

[27] D. Lingrand, "Projet rainbow: Rapport de recherche," Université Nice Sophia Antipolis, Centre national de la recherche scientifique, Laboratoire Informatique, Signaux et Systèmes de Sophia Antipolis, UMR 6070, Sophia-Antipolis Cedex, France, Tech. Rep. ISRN I3S/RR–2004-05–FR, Jan. 2004, LABORATOIRE I3S: Les Algorithmes / Euclide B – 2000 route des Lucioles – B.P. 121 – 06903 Sophia-Antipolis Cedex, France – Tél. (33) 492 942 701 – Télécopie : (33) 492 942 898. [Online]. Available: http://www.i3s.unice.fr/I3S/FR/.

[28] M. V. Droogenbroeck, *Traitement numérique des images*, Version 4.80, Institut MONTEFIORE, Service de Télécommunications et d'Imagerie, Sep. 2007.

[29] Wikipedia, the free encyclopedia. "Hsl and hsv. "[Online]. Available: https://en.wikipedia.org/wiki/HSL_and_HSV#HSL_to_HSV.

[30] Wikipedia contributors, *Histogram equalization – Wikipedia, The Free Encyclopedia*, 2023. [Online]. Available: https://en.wikipedia.org/wiki/Histogram_equalization.

[31] L. Oudre, *Traitement numérique du signal - cours 5 : Filtrage dans le domaine fréquentiel*, Université Paris 13, Institut Galilée, Ecole d'ingénieurs Sup Galilée, Parcours Informatique et Réseaux Alternance - 1ère année, 2017. [Online]. Available: laurent.oudre@univ-paris13.fr.

[32] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979. DOI: 10.1109/TSMC.1979.4310076.

[33] R. Fisher, S. Perkins, A. Walker, and E. Wolfart, *Adaptive thresholding*, 2003. [Online]. Available: http://home_url/adaptive_thresholding.

[34] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985, ISSN: 0734-189X. DOI: https://doi.org/10.1016/0734-189X(85)90016-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0734189X85900167.

[35] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986. DOI: 10.1109/TPAMI.1986.4767851.

[36] Jmarchn. "Électrocardiographie," Wikipédia, l'encyclopédie libre. [Online]. Available: https://fr.wikipedia.org/wiki/%C3%89lectrocardiographie.

[37] P. Taboulet. "Ecg : Normal. "[Online]. Available: https://www.e-cardiogram.com/ecg-normal.

[38] P. Kligfield, L. S. Gettes, J. J. Bailey, *et al.*, "Recommendations for the standardization and interpretation of the electrocardiogram: Part i: The electrocardiogram and its technology: A scientific statement from the american heart association electrocardiography and arrhythmias committee, council on clinical cardiology; the american college of cardiology foundation; and the heart rhythm society: Endorsed by the international society for computerized electrocardiology," *Circulation*, vol. 115, no. 10, pp. 1306–1324, 2007.

[39] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[40] S. Eddins, *Binary image convex hull – algorithm notes*, Steve on Image Processing with MATLAB, Oct. 2011. [Online]. Available: https://blogs.mathworks.com/steve/2011/10/04/binary-image-convex-hull-algorithm-notes/.

[41] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[42] P. Labatte, *Mathématiques analyse numérique : Interpolation et résolution d'équation*, 2009. [Online]. Available: https://math.univ-angers.fr/~labatte/INH/numerique2009.pdf.

[43] F. B. Tichadou, *Complexité algorithmique*, Mis à disposition sous licence Creative Commons Attribution-Partage dans les mêmes conditions 3.0 non transposé., Sep. 2021. [Online]. Available: https://bouchflo.gricad-pages.univ-grenoble-alpes.fr/L2-algo/.

[44] *Cours informatique: Complexité d'un algorithme*, Lycée Michelet, classes de PCSI, 2022. [Online]. Available: https://cahier-de-prepa.fr/psi-michelet.

[45] Python Software Foundation, *Python documentation: Time access and conversions*, 2024. [Online]. Available: https://docs.python.org/3/library/time.html.

[46] A. Oktavianita, H. Arifin, M. Fauzi, and A. Rifai, "An analysis of memory usage in web browser software," *IJID (International Journal on Informatics for Development)*, vol. 5, p. 21, Dec. 2016. DOI: 10.14421/ijid.2016.05204.

[47] Y. Shafranovich, *Common format and mime type for comma-separated values (csv) files*, Request for Comments 4180, Internet Engineering Task Force (IETF), Informational, Oct. 2005. [Online]. Available: https://www.ietf.org/rfc/rfc4180.txt.