

DOSSIER DE SOUTENANCE POUR LE TITRE DE CONCEPTEUR D'APPLICATION

Clément WILFART



MyDigitalSchool
2021

Table des matières

INTRODUCTION AU PROJET	4
Remerciements	5
Présentation	6
Project Summary	8
Présentation de l'entreprise	10
Liste des compétences	11
DESCRIPTION ET CAHIER DES CHARGES DU PROJET	12
Analyse du besoin	13
Les utilisateurs du projet	17
Fonctionnalités attendues	18
Fonctionnalités principales	18
Fonctionnalités secondaires	19
ANALYSE FONCTIONNELLE	20
Etats successifs d'une adresse email	23
Etats successifs d'une demande d'indu	24
Les bases de données	25
Structure	25
Design préliminaire de l'application interne	28
Page d'accueil	28

Gestion des adresses	28
REALISATION	29
Choix de développement.....	30
Langages et frameworks	30
Logiciels et autres outils.....	30
Mise à jour et ajouts dans la base de données	31
Modifications du Gestionnaire de données.....	33
Les entités.....	33
L'Objet d'Accès aux Données.....	34
Ajout dans la procédure d'annulation	38
Le frontend	38
Le traitement backend.....	44
Adaptation du programme Annul.....	45
Développement de l'application interne	46
L'interface.....	46
Le traitement	48
Fonctionnement normal	50
Améliorations notables.....	52
Interface utilisateur sur le site internet	54
Frontend	54

Backend	56
Tests	58
Déploiement.....	59
CONCLUSION	61
L'année passée	62
En entreprise	62
En centre de formation	62
Le projet	63

INTRODUCTION AU PROJET

Remerciements

Je souhaite remercier vivement toutes les personnes qui ont contribué au bon déroulement de mon année scolaire en alternance.

Tout d'abord l'équipe pédagogique de **MyDigitalSchool – Lille** ; Mesdames **Katie-May Hopton** et **Anne Gaelle Chevalier**, Monsieur **Dazzan** pour leur aide et leurs conseils apportés à l'élaboration de ce rapport, ainsi que tous les intervenants venus nous apporter durant cette année leur compétences et connaissances des métiers de l'informatique.

Ensuite, je remercie particulièrement mon tuteur Monsieur **Jean-François WARLOP**, Monsieur **Jean-Claude WARLOP** et toute la société **AGETIP – Ronchin**, pour m'avoir fait confiance, m'avoir accueilli dans l'entreprise et formé au métier de développeur.

Enfin je remercie mes proches de m'avoir soutenu dans mon projet, les personnes qui m'ont conseillé et relu lors de la rédaction de ce rapport, et vous, pour juger ma soutenance de ce projet professionnel.

Présentation

Bonjour, je m'appelle Clément Wilfart, j'ai 25 ans depuis mai dernier.

Passionné par la technologie en général depuis tout petit, j'ai effectué un BTS CRSA (Conception Réalisation de Systèmes automatiques) au Campus OZANAM – Lille obtenu en 2016.

Ensuite, j'ai travaillé 3 ans comme ouvrier polyvalent dans une entreprise de fabrication de matériel agricole ; mon activité principale consistait à assembler des pièces mécaniques et à du câblage électrique – je n'ai réalisé que très peu de programmation et pas de CAO. Souhaitant me réorienter, j'ai appris en autodidacte du C++, de l'HTML et du CSS, via les livres d'OpenClassroom.

A l'approche de la rentrée, j'ai prospecté à la recherche d'une école proposant une formation en alternance, et j'ai donc postulé à MyDigitalSchool – Lille, pour un Bachelor Développeur Web et Mobile. Cette opportunité m'a permis de décrocher un contrat d'alternance au sein de l'entreprise AGETIP à Ronchin, ce qui m'a permis ainsi de développer tout au long de l'année mes compétences en HTML, C#, SQL, Javascript, Visual Basic, Sécurité web, etc...

L'entreprise AGETIP est spécialisée dans la gestion du tiers payant pour les pharmacies et les professionnels de santé. Elle assure la transmission des demandes de remboursement de ce tiers payant

auprès des organismes obligatoires (Caisses...) et complémentaires (Mutuelles), et avance le tiers payant aux professionnels de santé.

J'y ai appris et développé des fonctionnalités frontend et backend, des bases de données et leur entretien, des applications internes, à partir d'un cahier des charges ou à partir d'applications existantes.

Le projet le plus enrichissant et le plus vaste que j'ai réalisé concerne les demandes d'indus formulées par les pharmaciens auprès d'AGETIP, suite à l'annulation d'une facture client.

Ce projet m'a permis :

- d'en réaliser les schémas fonctionnels,
- de générer de nouvelles bases de données, d'en prévoir le dispositif de maintenance,
- d'inclure à l'application Web existante une nouvelle interface client frontend et sa logique backend
- de concevoir une application interne permettant :
 - d'une part, aux collaborateurs d'AGETIP de traiter les données
 - et d'autre part, aux autres applications d'utiliser ces données,
- et enfin de réaliser toute une série de tests de fiabilité, ainsi que les corrections nécessaires dans le but de délivrer le produit abouti.

Project Summary

Hello, my name is Clément Wilfart, I'm 25 years old since may.

Passionate about technology in general since I was little, I completed an advanced technician's certificate in design and realization of automatic systems at the OZANAM Campus - Lille.

I then worked for 3 years as a general-purpose worker in an agricultural equipment manufacturing company where I did a lot of mechanics and electrical wiring, some programming, and no Computer Aided Design. Wishing to reorient myself, I started by learning C++, HTML and CSS on my own, through OpenClassroom books.

As the start of the school year approached, I looked for a school offering work-study training, and I applied to MyDigitalSchool - Lille, for a Bachelor in Web and Mobile Developer. This opportunity allowed me to get a work-study contract within the company AGETIP - Ronchin, and thus allowed me to develop throughout the year my skills in HTML, C#, SQL, Javascript, Visual Basic, Web security, etc ...

The AGETIP company specializes in third-party payment management for pharmacies and healthcare professionals. It ensures the transmission of reimbursement requests from this third-party payer to compulsory and complementary organizations.

There I learned and developed frontend and backend functionalities, databases and their maintenance, internal applications, from scratch or from existing applications.

The most enriching and the largest project I have carried out concerns inquiries made by pharmacists to AGETIP, following the cancellation of a customer invoice.

This project allowed me:

- to draw up the functional diagrams,
- to generate new databases and plan their maintenance,
- to include in the existing web application a new frontend client interface and its backend logic
- to design an internal application allowing:
 - on the one hand, AGETIP agents to process the data
 - and on the other hand, other applications to use this data,
- and finally, to carry out a whole series of reliability tests, as well as the necessary corrections in order to deliver the successful product.

Présentation de l'entreprise



AGETIP est un organisme privé et indépendant fondé par des pharmaciens en 1990 à Ronchin. Sa mission principale est d'assurer la gestion des flux télétransmis par les professionnels de santé, leur comptabilité, l'envoi des demandes de remboursement aux organismes obligatoires et complémentaires, etc... Ce service est proposé à tous les professionnels de santé en contrepartie d'une adhésion.

L'entreprise travaille en collaboration avec des grands groupes pharmaceutiques, comme la SOPROPHAR, la COGEPHAR, la DPGS, etc... auxquels sont affiliés les professionnels de santé.

Sur le plan technique AGETIP possède un site public « vitrine » et un site privé réservé aux professionnels pour leurs opérations en lien avec l'entreprise, des bases de données SQL, sachant que la majorité des traitements s'effectuent grâce à des fichiers texte et .dbf, et des programmes Visual FoxPro, l'entreprise étant adepte des produits Microsoft.

Liste des compétences

N° Fiche AT	Activités types	N° Fiche CP	Compétences professionnelles
1	Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité	1	Maquetter une application
		2	Développer une interface utilisateur de type desktop
		3	Développer des composants d'accès aux données
		4	Développer la partie frontend d'une interface utilisateur web
		5	Développer la partie backend d'une interface utilisateur web
2	Concevoir et développer la persistance des données en intégrant les recommandations de sécurité	6	Concevoir une base de données
		7	Mettre en place une base de données
		8	Développer des composants dans le langage d'une base de données
3	Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité	9	Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement
		10	Concevoir une application
		11	Développer des composants métier
		12	Construire une application organisée en couches
		13	Développer une application mobile
		14	Préparer et exécuter les plans de tests d'une application
		15	Préparer et exécuter le déploiement d'une application

DESCRIPTION ET CAHIER DES CHARGES DU PROJET

Lorsqu'un professionnel de santé établit une facture, qui est télétransmise à AGETIP ; ce dernier la contrôle avant de la transmettre aux régimes obligatoires et complémentaires concernés. AGETIP avance les frais auprès du professionnel de santé et établit des appels de fonds auprès des caisses.

Si pour une raison X ou Y, le professionnel de santé souhaite annuler une facture établie et payée, une demande d'indu est à établir et à envoyer auprès de la mutuelle qui a réglé la facture, afin qu'elle se fasse rembourser.

Jusque-là, le professionnel de santé était dans l'obligation d'établir lui-même sa demande d'indu (soit par email, par lettre). AGETIP a proposé à ses adhérents de développer une procédure permettant d'envoyer cette demande d'indu de manière automatisée, et d'en suivre l'avancement. Cette mission m'a été confiée.

Analyse du besoin

Objectif du projet

L'objectif du projet est de simplifier la vie du professionnel de santé, en s'occupant d'une démarche administrative (demande d'indu), dont AGETIP peut se charger puisqu'en possession de presque toutes les informations nécessaires à cette démarche au moment de la télétransmission initiale de la facture.

Analyse de l'existant

Le site internet AGETIP permet aux professionnels de santé, grâce à leurs identifiants personnels, d'accéder à leurs données traitées par AGETIP, notamment leurs factures en cours de traitement. Ils ont la possibilité d'annuler, selon les cas, la part obligatoire, la part complémentaire, ou la totalité d'une facture.

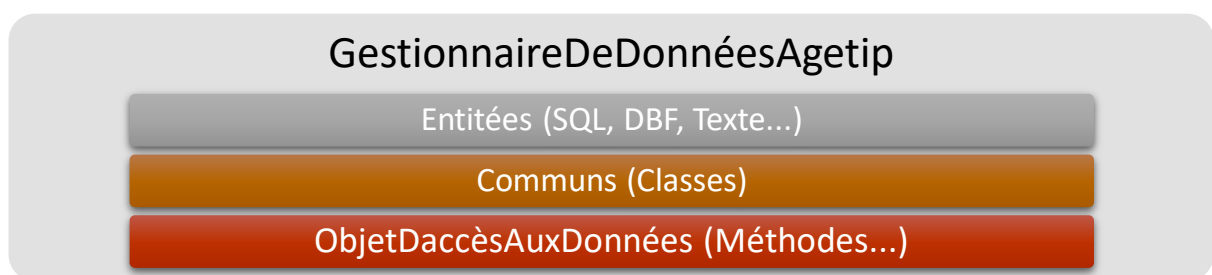
Les annulations effectuées sont stockées dans une base de données interne nommée « Annulation » créée en début d'année civile pour remplacer le stockage en fichier texte ; ce fut l'un de mes premiers projets dans l'entreprise.

Le professionnel de santé peut retirer son annulation, en cas de fausse manipulation par exemple, pour rétablir la facture à son état initial. Cependant, il doit le réaliser avant 8h30 du prochain jour ouvré qui suit l'annulation de la facture.

En effet, tous les jours ouvrés, à 8h30, un programme nommé « Annul » ferme partiellement le site Web AGETIP pour traitement des données et mises à jour avant de rouvrir le site Web aux professionnels de santé. Entre autres, les annulations de facture effectuées depuis la précédente exécution du programme sont définitives et seront traitées.

Le traitement backend du site internet, certaines fonctions du programme « Annul » et plus largement, une grande partie des traitements effectués par les programmes internes à AGETIP sont codés dans une bibliothèque de classes, divisée en 3 sous-projets, appelée « GestionnaireDeDonnéesAGETIP ». Cette bibliothèque contient les procédures d'accès aux bases de données SQL, aux fichiers textes, les structures DBF (Visual FoxPro) et leurs classes associées, les classes communes, et méthodes qui sont utilisées par plusieurs programmes.

Une application liée au compte mail de l'entreprise est capable de lire des fichiers texte ayant un format précis et de générer des emails sur cette base, et de les envoyer.



Contraintes

La procédure devra demander le moins de sujétions de la part du professionnel de santé, tout en le gardant informé de l'avancement de ses demandes d'indu.

Les mutuelles communiquent aux professionnels de santé les adresses email destinées au traitement des indus ; la procédure visera à récupérer ces adresses email, les stocker dans une base de données, et les mettre en œuvre pour envoyer les demandes d'indu pour le compte des professionnels de santé.

Un collaborateur d'AGETIP devra être capable de consulter, vérifier, agir :

- sur les demandes d'indu,
- sur les adresses email associées,

et ce par le biais d'une application interne, chargée également d'effectuer des vérifications automatiques (expliquées en détail plus loin).

Les utilisateurs du projet

Les utilisateurs du projet sont :

- Les développeurs (et chef de projet),
- Les collaborateurs AGETIP,
- Les professionnels de santé.

Les développeurs

Les développeurs disposeront de tous les droits sur l'application interne (notamment à des fins de debug), sur le code front et backend du site, et sur les bases de données.

Les collaborateurs d'AGETIP

Les collaborateurs disposeront de droits limités à l'utilisation de l'application interne dans le cadre de leur activité afin d'éviter les erreurs de saisie possibles.

Les professionnels de santé

Les professionnels de santé n'ont, bien entendu, pas accès à l'application interne ; ils interagissent avec le projet via la page d'annulation de facture, et une page récapitulative de l'avancement du traitement des demandes d'indu.

Fonctionnalités attendues

Fonctionnalités principales

Site Internet AGETIP

Le professionnel de santé une fois connecté doit pouvoir renseigner sa volonté de faire parvenir sa demande d'indu via AGETIP, et si oui, il doit pouvoir renseigner l'adresse email de la Mutuelle qui doit recevoir la demande.

Il doit aussi pouvoir consulter l'avancement de ses demandes d'indu sur une page du site.

Programme Annul

Le programme journalier doit enregistrer définitivement les demandes d'indu en même temps que les annulations, et doit mettre de côté les demandes d'indu relatives à des annulations retirées.

Application interne

L'application interne doit mettre à jour l'avancement des demandes d'indu en fonction d'autres données internes de AGETIP. Néanmoins, une intervention manuelle doit être possible pour changer l'état des indus, ou des adresses email (par exemple si l'adresse email n'existe pas).

L'application devra enfin générer un fichier contenant des emails formatés qu'une autre application existante sera chargée d'envoyer aux mutuelles.

Fonctionnalités secondaires

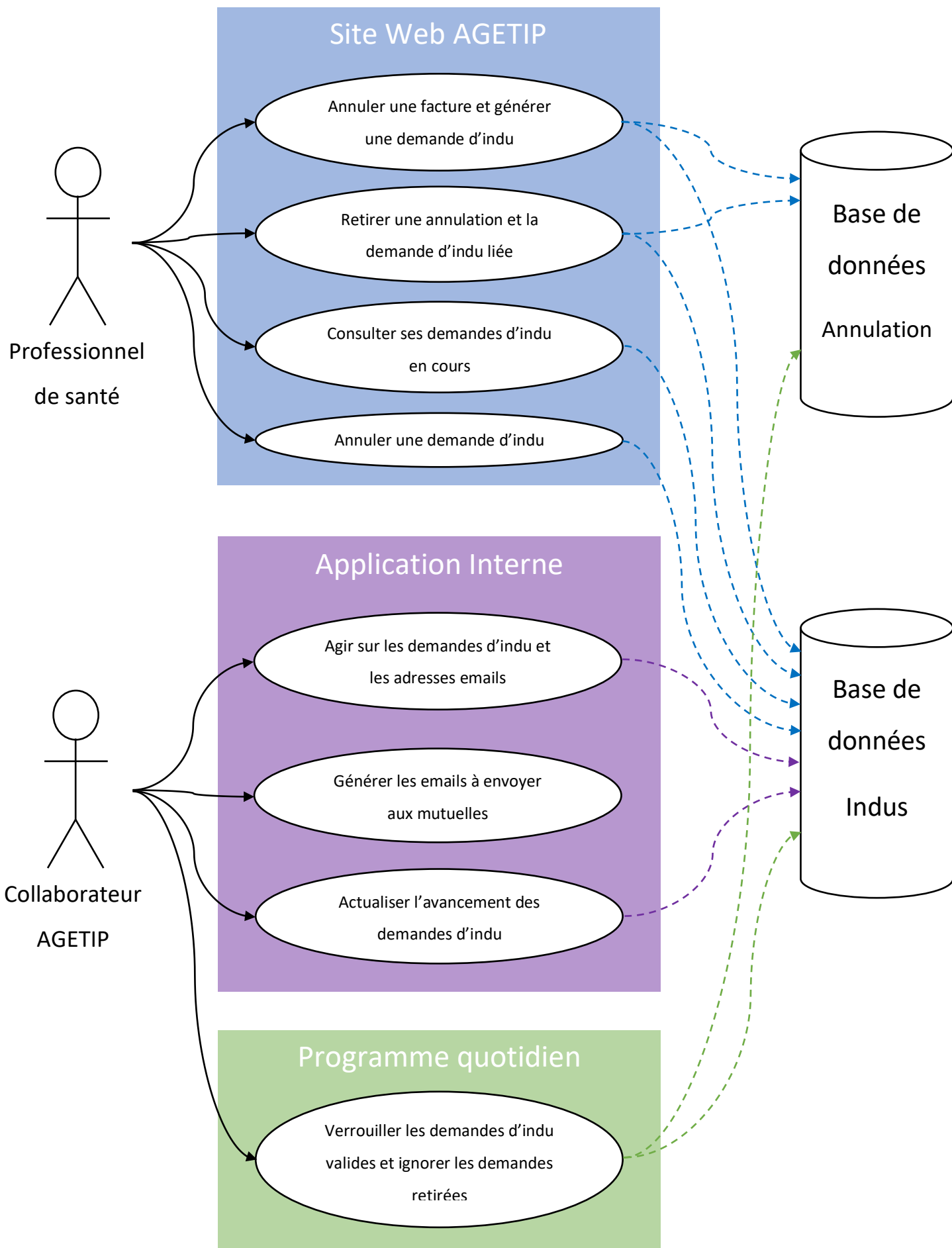
Site Internet AGETIP

Le professionnel pourra gérer ses demandes d'indu selon les cas, les annuler indépendamment de l'annulation de la facture, ou les supprimer de son affichage.

Application interne

L'application interne peut inclure un outil de recherche précis de demandes d'indus ou d'adresses, à des fins de modifications ou de consultations

ANALYSE FONCTIONNELLE

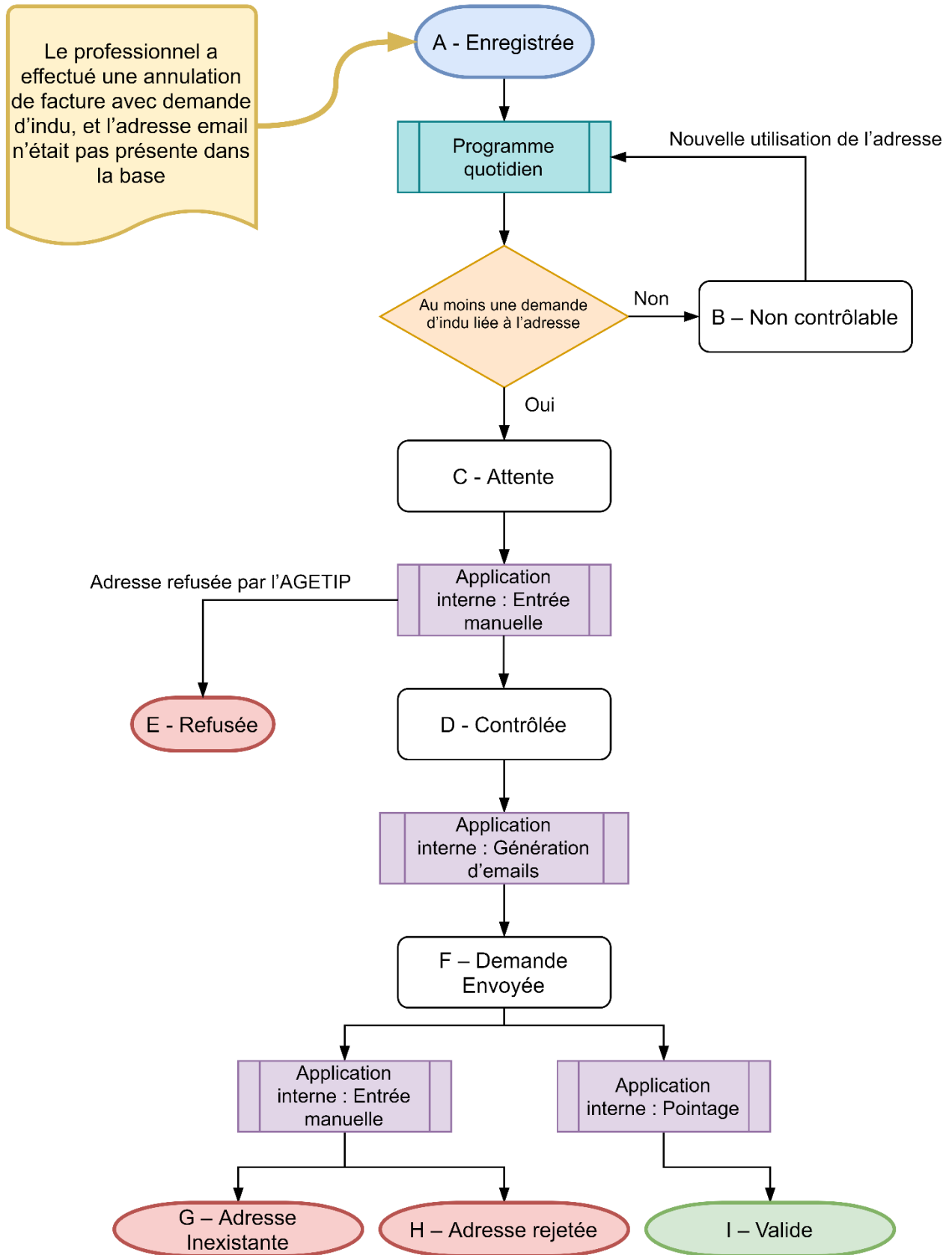


Dans le but de mieux comprendre et programmer l'avancement d'une demande d'indu ou d'une adresse, j'ai réalisé les diagrammes suivants. Les demandes d'indus et les adresses email sont caractérisées ici par un des états, stockés dans une base sous la forme d'une lettre identificatrice, une courte description, et une description plus longue. La description plus longue n'est pas montrée ici.

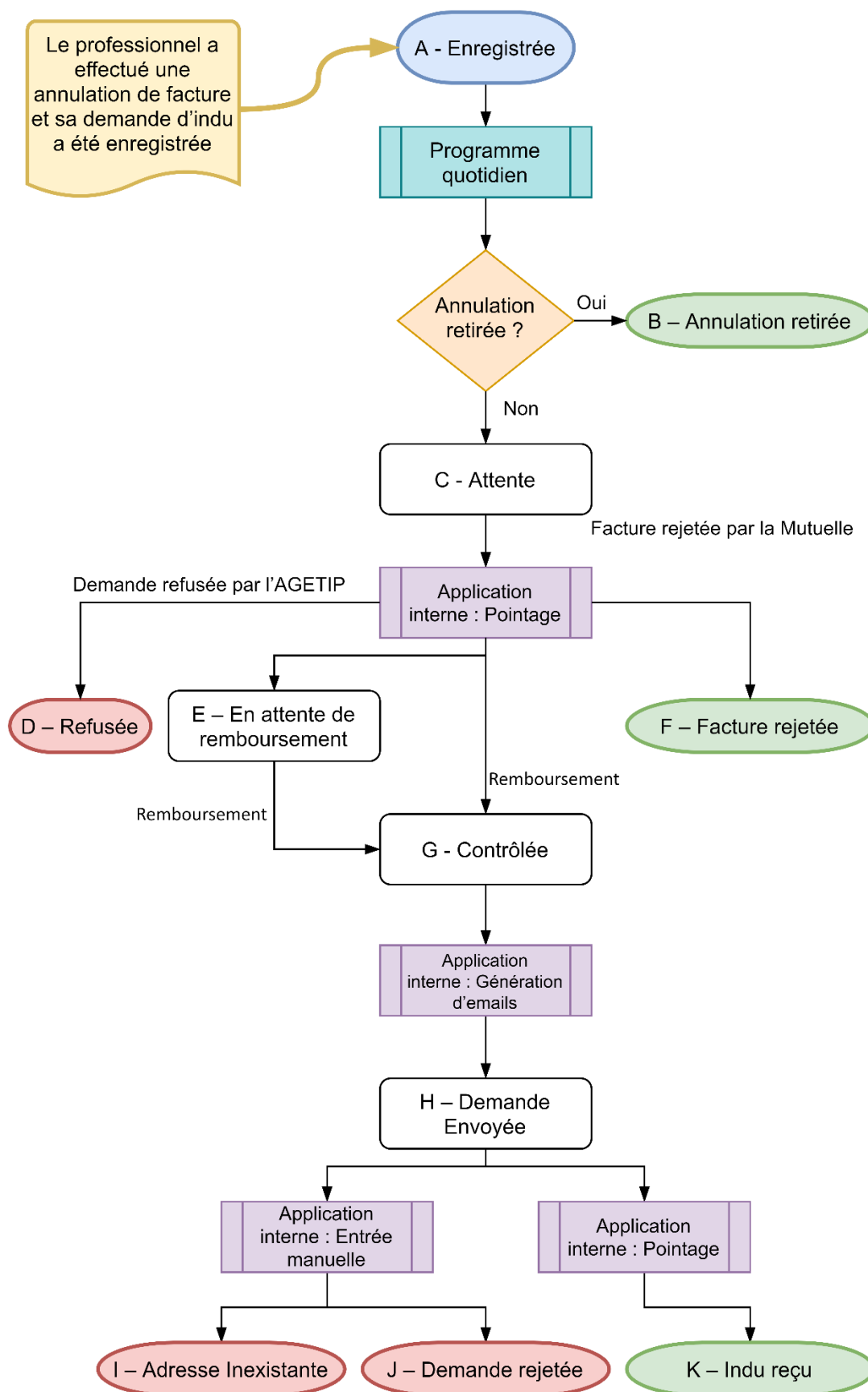
Une terminaison indique un état définitif pour l'adresse ou la demande d'indu. Celle-ci peut être due à un fonctionnement normal (Adresse valide, Indu reçu...), ou à une erreur ou un rejet (Adresse inexistante...). Dans tous les cas, une demande d'indu « terminée » restera affichée pendant 14 jours sur le tableau de bord du professionnel, sur le site web, à partir de son dernier changement d'état. Une adresse restera indéfiniment sur le site, sauf si le professionnel en décide autrement.

Dans les cas où la demande d'indu a échoué en raison d'une erreur ou d'un rejet, le professionnel sera prévenu par le biais d'un message personnel sur le site web, dont la notification est visible sur la page d'accueil, dès sa connexion.

Etats successifs d'une adresse email



Etats successifs d'une demande d'indu

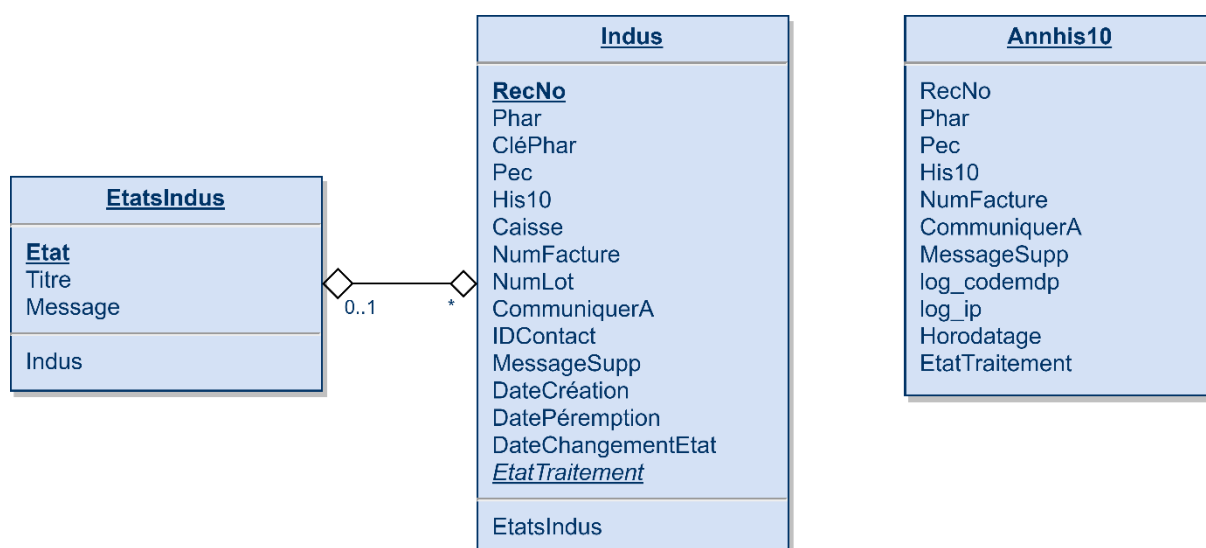


Les bases de données

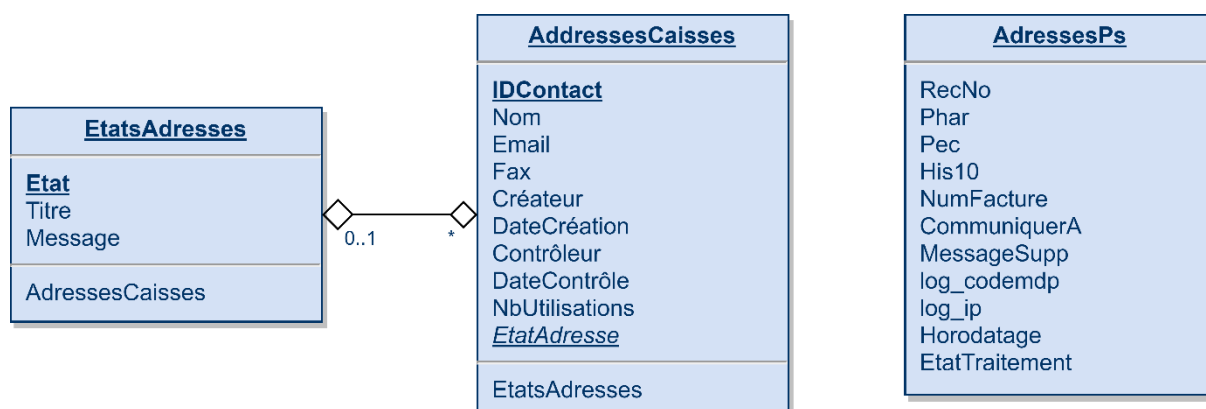
Structure

Voici la structure des bases de données et des tables créées pour ce projet¹.

Base Annulation



Base Contact



¹ Comme mentionné plus tôt, la base Annulation et la table Annhis10 existaient déjà

Voilà deux exemples plus détaillés de deux tables créées.

Table Indus

Colonne	Clé	Type	Null ?	Description
RecNo	Primaire	bigint	NOT NULL	Identifiant
Phar		nvarchar(8)	NOT NULL	Identifiant du professionnel
CléPhar		nvarchar(1)	NOT NULL	Clé du professionnel
Pec		nvarchar(9)	NOT NULL	Numéro original de facture
His10		int	NOT NULL	Position de la facture dans le fichier interne
Caisse		nvarchar(3)	NOT NULL	Code de la Caisse du client
NumFacture		nvarchar(9)	NULL	Nouveau numéro de facture
NumLot		nvarchar(3)	NOT NULL	Numéro de lot de la facture
CommuniquerA		nvarchar(1)	NOT NULL	Part de la facture annulée
IDContact		bigint	NOT NULL	Identifiant de l'Adresse utilisée
MessageSupp		nvarchar(255)	NULL	Message supplémentaire fourni par le professionnel
DateCréation		DateTime	NOT NULL	Date de création de la demande d'indu
DatePéremption		DateTime	NULL	Date de disparition de la demande d'indu de l'affichage du professionnel
DateChangementEtat		DateTime	NOT NULL	Date de dernière modification de l'état de la demande d'indu
EtatTraitement		nvarchar(1)	NOT NULL	Etat actuel de la demande d'indu
Log_codempdp		nvarchar(2)	NOT NULL	Type de compte du professionnel

Table Adresses

Colonne	Clé	Type	Null ?	Description
IDContact	Primaire	bigint	NOT NULL	Identifiant
Nom		nvarchar(50)	NULL	Nom propre de la caisse
Email		nvarchar(255)	NULL	Adresse email de la caisse
Fax		nvarchar(12)	NULL	Adresse Fax de la caisse
Créateur		nvarchar(8)	NOT NULL	Professionnel à l'origine de l'adresse
DateCréation		DateTime	NOT NULL	Date de création de l'adresse
Contrôleur		nvarchar(50)	NULL	Initiales de l'collaborateur et nom de la machine ayant contrôlé l'adresse
DateContrôle		DateTime	NULL	Date du contrôle de l'adresse
NbUtilisations		int	NOT NULL	Nombre d'utilisations de l'adresse
EtatAdresse	Etrangère	nvarchar(1)	NOT NULL	Etat actuel de l'adresse

Design préliminaire de l'application interne

Page d'accueil

DATAGRID	Bouton des Adresses
	Bouton des carnets d'Adresses
	Champs de Recherche
	Liste d'états Valider
	Générer Emails

Gestion des adresses

DATAGRID	Champs de Recherche
	Liste d'états Valider

REALISATION

Choix de développement

J'ai utilisé les langages et outils utilisés par l'entreprise afin de garder une cohésion dans l'intégration de mon travail, à savoir :

Langages et frameworks

Site internet AGETIP

- HTML 5
- CSS 3
- Framework ASP.NET Webforms
- C#
- Visual Basic

Application Interne WPF

- C#
- Framework 4.7.2

Bases de données

- SQL

Logiciels et autres outils

- Visual Studio pour le développement, ce qui permet de faire des tests en local sur un site internet et du debug sur l'application interne
- SQL Server management Studio pour la gestion des bases de données de l'entreprise
- TFS (Team Foundation Server) Azure pour partager, sauvegarder et versionner le code de l'entreprise, sur un serveur interne

Mise à jour et ajouts dans la base de données

Pour entamer le développement de ce projet, j'ai procédé à des modifications sur les bases de données internes de l'entreprise. En utilisant SQL Server Management Studio, j'ai importé sur ma machine une copie de la base de données « Annulation » existante et sa table « Annhis10 ».

Toujours sur ma machine et grâce au créateur graphique de SQLSMS, j'ai généré conformément aux directives définies en réunion de travail, une table « Indus » et une table « EtatsIndus » dans la base « Annulation », puis j'ai créé une base « Contact » dans laquelle j'ai ajouté les tables « Adresses », « AdressesCaisses » et « EtatsAdresses ».

Les tables Etat contiennent tous les états possibles décrits dans les organigrammes vus plus haut. Ces tables servent de référence et sont amenées à ne pas changer ou peu changer. La clé primaire est une lettre, de A à K (voir plus si des états sont ajoutés). Le titre est une valeur suffisamment courte pour s'afficher dans un tableau, sur l'application interne ou sur le site internet. Le message est une description plus longue de l'état à l'intention du professionnel de santé, qui s'affichera donc sur le site, sous la forme d'un tooltip.

La table Indus sauvegarde les demandes d'indus générées à partir du site internet. Elle contient notamment le Phar, qui est l'identifiant du professionnel de santé, le numéro de facture Pec, et l'IDContact, qui est rattachée à la clé primaire du même nom de la table Adresses.

La table Adresses contient les adresses email ou Fax utilisées par tous les professionnels de santé. Chaque adresse est unique et est identifiée par la clé primaire IDContact. Les adresses ont aussi un état, dont la procédure vérifie la validité. Le cas échéant, le site Web bloque la génération de nouvelles demandes d'indus.

La table AdressesCaisses est en quelque sorte le carnet d'adresses de tous les professionnels de santé. Chaque entrée est un couple unique comprenant l'IDContact, le Phar, le numéro de mutuelle, et l'usage de l'adresse. Cette table permet donc de mémoriser qui a utilisé quelle adresse, pour quelle mutuelle.

Pour chaque table, le champ RecNo, Etat, ou IDContact, est défini comme clé primaire, non nul, et auto incrémental, ceci afin de laisser la base de données gérer l'ID de chaque entrée.

Modifications du Gestionnaire de données

La prochaine étape importante a été de modifier le Gestionnaire de données, notamment pour prendre en compte les modifications de la base de données.

Les entités

J'ai modifié la connectionString d'accès à la base de données de production pour utiliser uniquement la base interne de ma machine le temps du développement. Ensuite j'ai utilisé Visual Studio pour régénérer les tables créées précédemment en classes utilisables dans le code C#. Par exemple, la table Indus a été générée comme suit :

```
namespace GestionnaireDeDonnéesAGETIP.Entités.SQLServer
{
    using System;
    using System.Collections.Generic;

    public partial class Indus
    {
        public long RecNo { get; set; }
        public string Phar { get; set; }
        public string CléPhar { get; set; }
        public string Pec { get; set; }
        public int His10 { get; set; }
        public string Caisse { get; set; }
        public string NumFacture { get; set; }
        public string NumLot { get; set; }
        public string CommuniquerA { get; set; }
        public long IDContact { get; set; }
        public string MessageSupp { get; set; }
        public System.DateTime DateCréation { get; set; }
        public Nullable<System.DateTime> DatePéréemption { get; set; }
        public System.DateTime DateChangementEtat { get; set; }
        public string EtatTraitement { get; set; }
        public string log_codempd { get; set; }
    }
}
```

Pour accéder aux bases de données depuis le code C#, nous utilisons donc une classe DbContext, qui a généré les accès aux bases automatiquement. La base Contact a été générée, puis modifiée comme suit (la connectionString a été raccourcie) :

```
namespace GestionnaireDeDonnéesAGETIP.Entités.SQLServer
{
    using System;
    using System.Data.Entity;
    using System.Data.Entity.Infrastructure;

    public partial class ContactEntities : DbContext
    {
        public ContactEntities() : base("metadata=res://*/SQLServer.Contact.csdl|[...]")
        {
        }
    }
}
```

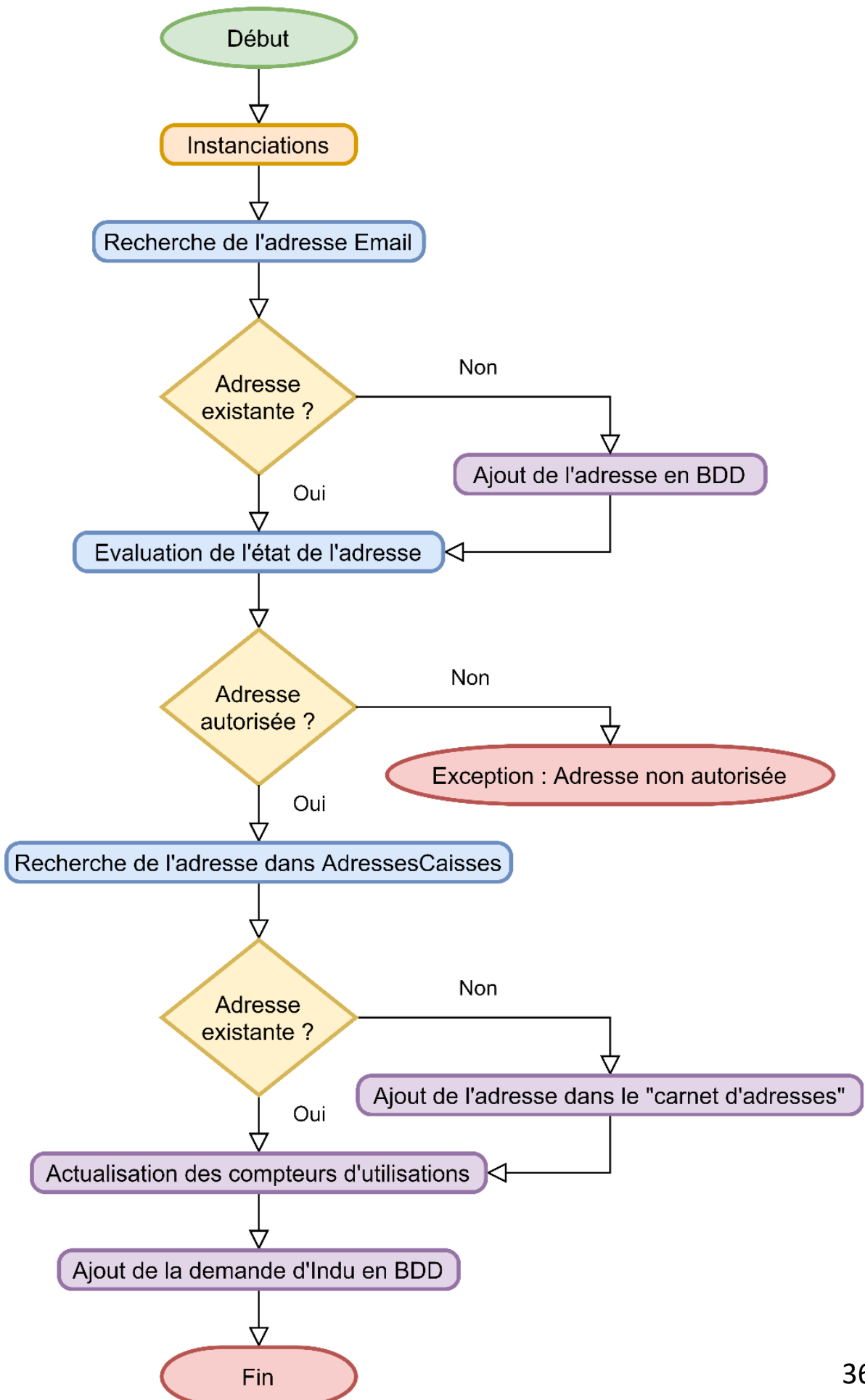
La classe DbContext permet d'accéder facilement aux bases de données internes juste en instanciant la classe, et nous permet d'effectuer toutes sortes d'actions sur la base sans devoir écrire de code SQL. Les tentatives d'injections SQL sont alors très réduites étant donné que l'ORM se base sur des objets et prépare lui-même les requêtes.

L'Objet d'Accès aux Données

J'ai ensuite créé une page de code dans le Gestionnaire de données Objet d'Accès aux Données, nommée ContactOad, afin d'y concentrer tout le code de la classe du même nom, qui consiste à l'acquisition, le traitement, et l'écriture des Adresses.

Le code s'est étoffé au fur et à mesure du développement, suivant les besoins en méthodes.

La méthode « principale » de cette classe, `ContactCaisse`, est une méthode appelée une seule fois, dans le traitement back end du site web, que nous verrons plus tard. Cette méthode est un enchainement de méthodes internes à la classe, procédant comme suit :



Le reste de la classe se compose d'une vingtaine de méthodes, d'accès aux données ou de traitement.

```
/// <summary>
/// Recherche l'IDContact dans la base et renvoie l'objet Adresses associé si il existe
/// </summary>
/// <param name="IDContact"></param>
/// <returns>Un objet AdressesCaisses correspondant à l'entrée de la base associée à l'IDContact,
ou null s'il n'existe pas</returns>
public static AdressesCaisses RechercheAdresseCaisse(long IDContact)
{
    ContactEntities DbContact = new ContactEntities();
    AdressesCaisses AdresseCaisse = new AdressesCaisses();
    if (DbContact.AdressesCaisses.Where(0 => 0.IDContact == IDContact).Any())
    {
        AdresseCaisse = DbContact.AdressesCaisses.DefaultIfEmpty(null)
            .FirstOrDefault(0 => 0.IDContact == IDContact);

        return AdresseCaisse;
    }
    else return null;
}
```

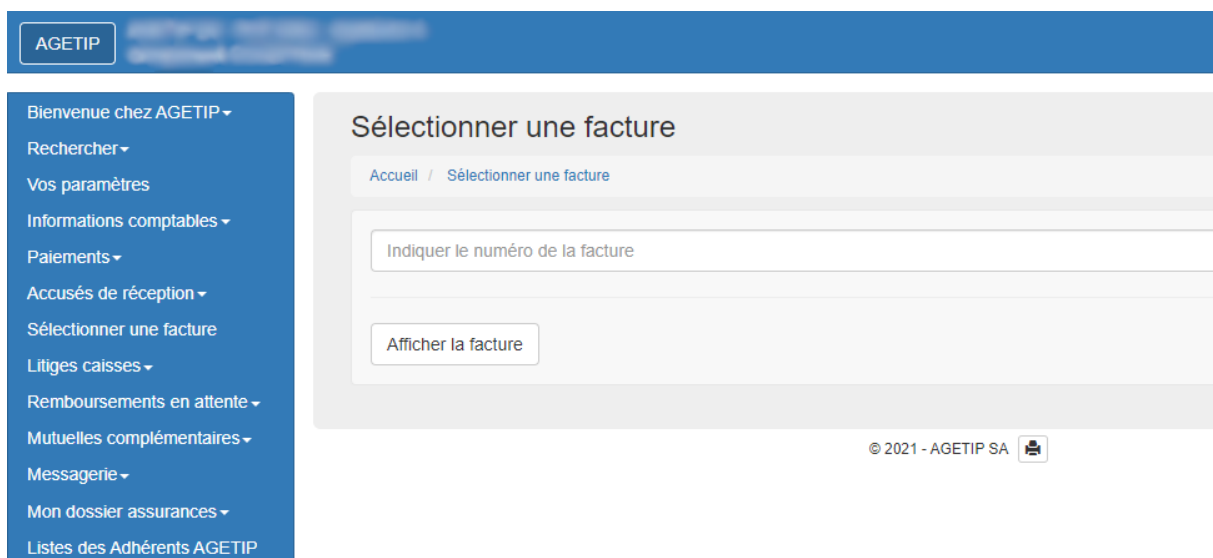
Pour procéder à des requêtes, j'utilise les commandes LINQ et les prédicats, deux outils puissants et polyvalents permettant des requêtes extrêmement rapides et efficaces. Dans un certain cas évoqué plus tard, j'ai été amené à utiliser un système un peu plus complexe appelé PredicateBuilder, inutile ici.

Ajout dans la procédure d'annulation

L'étape suivante consistait à ajouter un protocole de création de demande d'indu à la procédure d'annulation de facture habituelle.

Le frontend

Pour accéder à l'annulation de sa facture, le professionnel de santé possède plusieurs moyens, le plus simple étant d'utiliser le moteur de recherche « Rechercher une facture », et d'y entrer le numéro de facture recherché.



The screenshot displays the AGETIP web application interface. On the left is a blue sidebar menu with the following items: 'Bienvenue chez AGETIP', 'Rechercher', 'Vos paramètres', 'Informations comptables', 'Paielements', 'Accusés de réception', 'Sélectionner une facture', 'Litiges caisses', 'Remboursements en attente', 'Mutuelles complémentaires', 'Messagerie', 'Mon dossier assurances', and 'Listes des Adhérents AGETIP'. The main content area is titled 'Sélectionner une facture' and includes a breadcrumb trail 'Accueil / Sélectionner une facture'. Below this is a text input field labeled 'Indiquer le numéro de la facture' and a button labeled 'Afficher la facture'. At the bottom right of the main area, there is a copyright notice '© 2021 - AGETIP SA' and a small icon of a person.

Une fenêtre s'ouvre alors contenant les informations relatives à la facture.

The screenshot shows a web application interface for managing invoices. At the top, there is a blue header bar. Below it, a light gray box contains the following elements:

- A back arrow icon and the text "Facture n° 123456789".
- The text "Recherche effectuée le 15/04/2021 à 10:00:00".
- A section titled "Historiques de la facture" containing a table with the following data:

Document	Semaine	Matricule	Montant	Observation
Facture payée (DP10)	2021	123456789	1000	Facture payée
- A section titled "Situation comptable de la facture" with a light blue background and the text "Facture en Rejet Mutuelle".
- A section titled "Action possible sur la facture" containing two buttons: "Effectuer une annulation" and "Indiquer un recyclage".

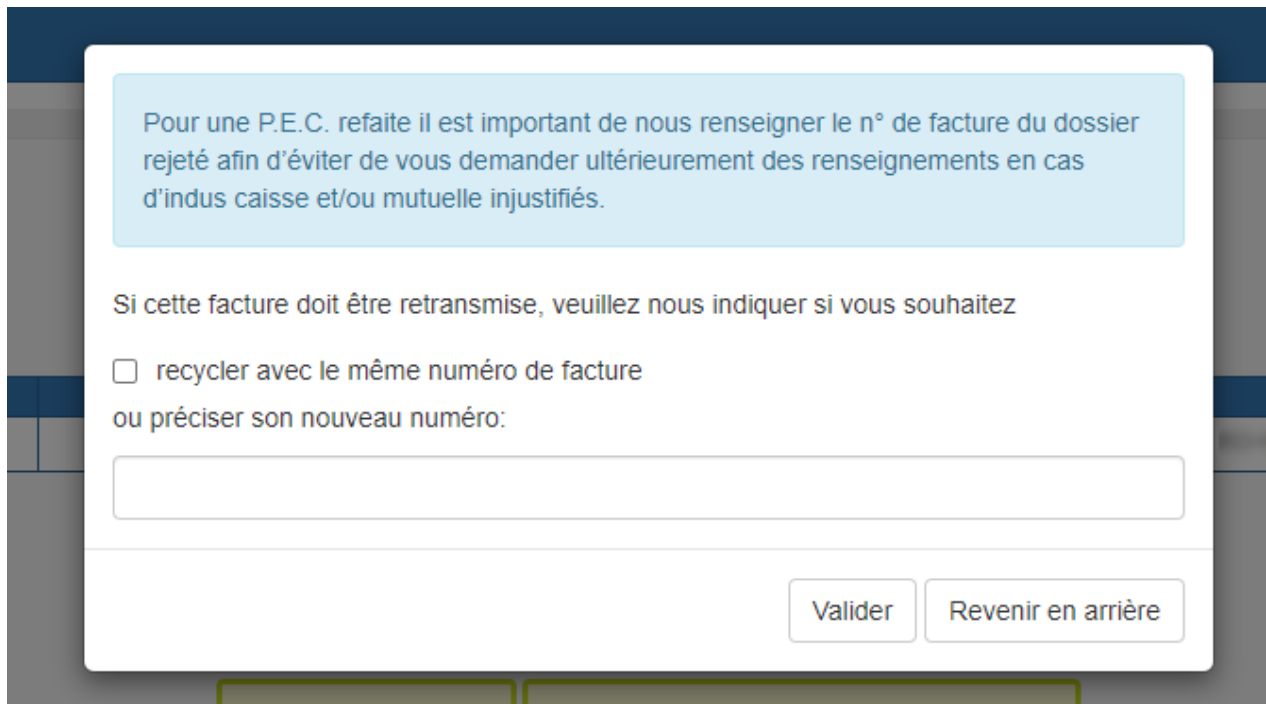
A ce stade, et selon les cas, le professionnel peut choisir d'annuler tout ou partie de sa facture, de retirer tout ou partie de son annulation, demander un paiement, etc...

Un clic sur un des boutons en bas fait apparaître une fenêtre modale relative au bouton cliqué et assombrit le fond pour centrer l'attention de l'utilisateur sur cette fenêtre. Par exemple, cliquer sur « Annuler la facture » ouvre une fenêtre modale qui propose d'annuler la part obligatoire, la part mutuelle, ou la totalité.

The screenshot shows a modal window with a white background and a dark gray border. It contains the following elements:

- The text "Effectuer une demande d'annulation sur :" in a light blue font.
- Three buttons: "la part caisse", "la part mutuelle", and "la totalité".
- A "Fermer" button in the bottom right corner.

Une fois la part choisie, une autre fenêtre modale prend la place de la précédente, proposant de conserver le numéro de facture actuel ou d'en changer.

A modal window with a light blue header containing a message about the importance of providing the invoice number for recycling. Below this, a question asks if the invoice should be retransmitted. There is a checkbox for 'recycler avec le même numéro de facture' and a text input field for 'ou préciser son nouveau numéro:'. At the bottom right, there are two buttons: 'Valider' and 'Revenir en arrière'.

Pour une P.E.C. refaite il est important de nous renseigner le n° de facture du dossier rejeté afin d'éviter de vous demander ultérieurement des renseignements en cas d'indus caisse et/ou mutuelle injustifiés.

Si cette facture doit être retransmise, veuillez nous indiquer si vous souhaitez

☐ recycler avec le même numéro de facture

ou préciser son nouveau numéro:

Valider Revenir en arrière

Un clic sur « Valider » fermera la fenêtre modale et appellera le code-behind pour effectuer l'action choisie sur la part choisie. La page se rafraichira à la fin de la procédure avec un message de succès ou d'erreur.

Pour s'intégrer parfaitement à la procédure actuelle, il a été décidé que la proposition de demande d'indu se présentera sous la forme d'une troisième fenêtre modale, après les deux premières, mais uniquement dans le cas d'une annulation de la part caisse ou de la totalité de la facture.

J'ai donc utilisé le code existant pour créer cette troisième fenêtre modale, et l'ai insérée dans le code, avec une génération conditionnelle cependant.

La facture se chargeant dans le code-behind en premier, j'ai utilisé ses valeurs pour décider ou non si la troisième fenêtre devait s'afficher.

The screenshot shows a web form with a light blue header box containing the text: "La facture a déjà été soldée ou est en attente de retour. Vous avez à envoyer à la Caisse une demande pour établir l'indu." Below this, the question "Souhaitez-vous qu'AGETIP transmette cette demande ?" is followed by two radio buttons: "Non, je l'envoie moi-même" and "Oui" (which is selected). Below the radio buttons, there is a label "Email du Service Indus de la Caisse destinataire" followed by a redacted email address. A note states: "Nous ne disposons pas de cette adresse, si elle existe, vous l'avez reçue directement de l'organisme (Flux-Tiers, courrier, ...)". Below this is an empty text input field. Further down, the label "Message supplémentaire à transmettre à la Caisse (facultatif):" is followed by another empty text input field. At the bottom right, there are two buttons: "Valider" and "Revenir en arrière".

Cette fenêtre est composée de deux radioboutons, et d'une textbox.

Le radiobouton est relatif à la question « AGETIP doit-elle envoyer une demande d'indu », et propose les réponses Oui et Non. Pour éviter toute confusion, le bouton Non est coché par défaut, et la textbox est désactivée. L'état du radiobouton et le contenu de la textbox sont envoyés au code-behind lors du clic sur « Valider » pour définir si la demande d'indu doit être effectuée.

Si l'utilisateur clique sur « Oui », la textbox se déverrouille et une adresse email peut être renseignée. J'ai codé une fonction en javascript, qui évalue l'état du radiobouton à chaque changement, et définit si oui ou non la textbox doit être active.

J'ai créé un regex d'adresse email, présent dans une fonction javascript et dans le code-behind. La fonction javascript permet d'empêcher l'utilisateur d'entrer une adresse email non autorisée, les adresses étant limitées au domaine de l'assurance-maladie. Le même regex est présent dans le code-behind comme redondance, au cas-où l'utilisateur désactiverait les scripts locaux.

De plus, la textbox est en réalité couplée à une DropDownList, dans laquelle sont stockées les adresses précédemment utilisées par ce professionnel, pour la même caisse destinataire. L'auto-complétion du navigateur est désactivée, ce qui permet de ne pas mélanger les adresses entre les différentes caisses, mais aussi et surtout de ne plus proposer d'adresses qui ont été déclarées comme inutilisables.

La facture a déjà été soldée ou est en attente de retour.
Vous avez à envoyer à la Caisse une demande pour établir l'indu.

Souhaitez-vous qu'AGETIP transmette cette demande ?

☐ Non, je l'envoie moi-même

☒ Oui

Email du Service Indus de la Caisse destinataire (01 30 30 30 30)
Nous ne disposons pas de cette adresse, si elle existe, vous l'avez reçue directement de l'organisme (Flux-Tiers, courrier, ...)

Valider

Revenir en arrière

Le traitement backend

Quand le professionnel charge la page de sa facture, celle-ci est agrégée des données internes de AGETIP par le code-behind. J'utilise donc certaines de ces données dans la génération de la page de la facture, notamment dans la génération de la troisième fenêtre modale, qui contient la question des demandes d'indus.

Si le professionnel choisi d'annuler une facture, la méthode « Annuler » est appelée. Celle-ci rassemble les données nécessaires et selon le cas, appelle une méthode de FactureOad avec des paramètres selon la part à annuler.

Parmi ces données, un paramètre a été ajouté aux méthodes existantes. Il s'agit d'un objet commun DemandeIndu, contenant les informations nécessaires pour déclarer la demande d'indu. La méthode ainsi appelée effectue ses contrôles et opérations habituelles ; puis, juste avant la fin de cette méthode, la méthode « ContactCaisse » créée plus tôt est appelée, et l'objet DemandeIndu lui est passé. La classe ContactOad prend alors temporairement le relais pour sauvegarder la demande d'indu selon le diagramme du point précédent.

Adaptation du programme Annul

Le programme Annul est un programme .exe autonome, codé en Visual Basic. Il est lancé chaque jour ouvré vers 8h30. Sa fonction est de vérifier la véracité des annulations effectuées depuis le précédent lancement du programme, et d'en dispatcher les informations dans des fichiers spécifiques selon le professionnel de santé.

J'ai récupéré le code du programme « Annul » et j'y ai inclus le nouveau Gestionnaire de données permettant l'accès aux méthodes et entités des nouvelles bases. Ensuite j'ai ajouté une petite portion de code après le code original, permettant de traiter les demandes d'indus, tout en les différenciant des demandes retirées. Le programme se charge donc de changer les états de toutes les demandes d'indu à l'état A – Enregistrée à C – En attente. Le programme procède ensuite de même avec les Adresses.

Développement de l'application interne

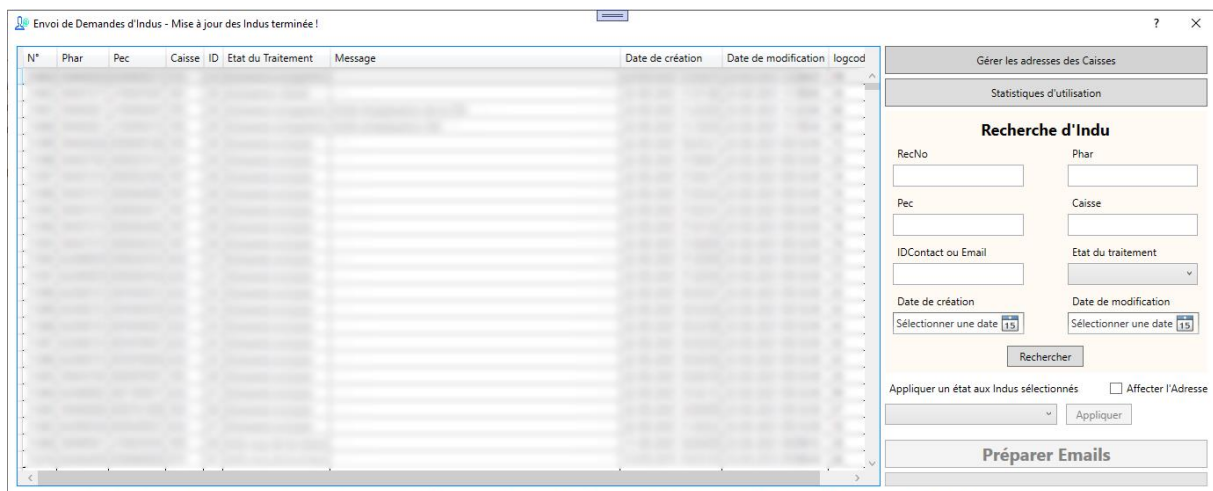
L'apport en demandes d'indus étant fonctionnel, il était temps de se pencher sur l'application interne de AGETIP, pour traiter les demandes d'indu. Pour ce faire, j'ai réalisé une application Windows WPF (Windows Presentation Foundation) et y ai inclus le Gestionnaire de données. D'autres packages ont été rajoutés au fur et à mesure du développement, selon les besoins.

L'interface

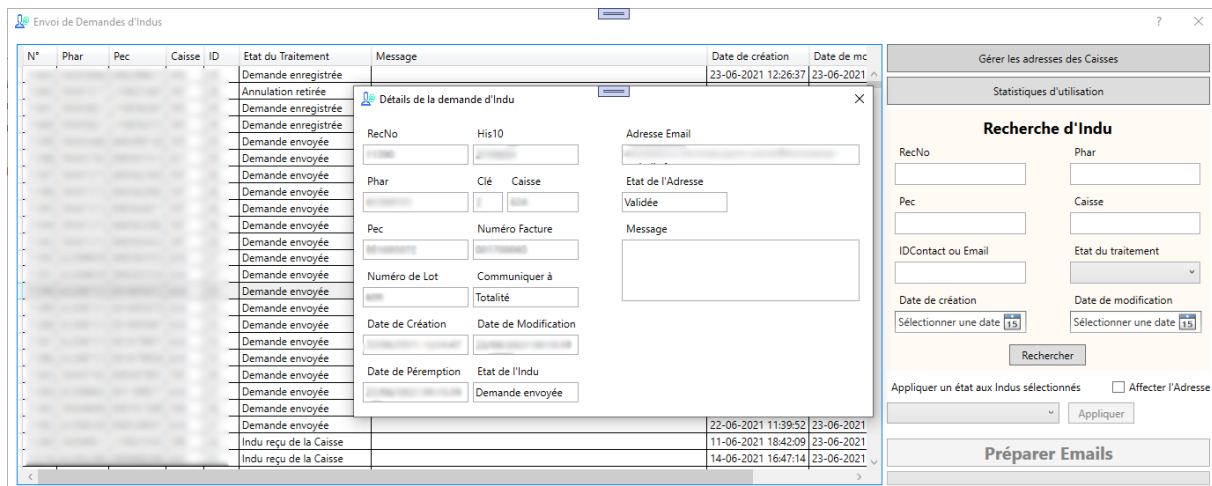
J'ai designé l'interface de l'application selon des croquis effectués en collaboration avec mon tuteur. L'application devait donc comprendre :

- Un grand Datagrid pour afficher les données
- Deux boutons qui ouvriraient des fenêtres similaires pour les Adresses et AdressesPs
- Une zone avec divers inputs utilisateurs pour procéder à des recherches dans la base
- Une liste déroulante pour choisir un état à appliquer à une ou plusieurs demandes d'Indu sélectionnées, et un bouton pour appliquer le choix
- Un bouton pour générer le fichier d'emails à envoyer aux mutuelles

Les éléments ont été placés et attachés entre eux de telle sorte à ce que si on redimensionne la fenêtre, seul le Datagrid change de taille, les autres éléments sont attachés aux bords de la fenêtre et gardent leurs tailles respectives. La fenêtre possède une taille minimum de sorte à ce qu'aucun élément ne se retrouve écrasé ou ne disparaisse. Elle n'a en revanche pas de limite maximale et peut prendre un plein écran sans trop se dénaturer.



Des événements ont été générés à partir de la plupart des éléments de l'interface (double-clic sur le Datagrid, clic sur les boutons, appui sur la touche Entrée dans les textbox, etc...) dans le but de faire une interface fonctionnelle et user-friendly.



Le traitement

Au lancement de l'application, diverses actions sont effectuées. Pour éviter de monopoliser le thread de l'UI et assurer un retour sur l'avancement de ces actions à l'utilisateur, ces traitements sont effectués par un `BackgroundWorker`, qui permet d'effectuer ces traitements sur un autre thread libre. Pour chaque étape du traitement automatique, la barre de titre de l'application indique à quelle étape elle se situe et indique le nombre d'opérations restantes à effectuer en temps réel.

Sauvegarde

En tout premier lieu, une sauvegarde complète des deux bases de données Indus et Adresses sont effectuées. En effet, pour être capable de restaurer les données à la suite d'un plantage ou d'une mauvaise manipulation, une copie des bases au format `.csv` est effectuée à chaque lancement et avant tout traitement. Pour ce faire, j'ai utilisé une bibliothèque nommée `CsvWriter`.

Les sauvegardes de plus d'un mois sont automatiquement supprimées par l'application avant la sauvegarde du jour.

Pointage des indus reçus

L'application appelle une méthode située dans ContactOad pour récupérer grâce à une requête avec prédicat, les demandes d'indus qui ont été envoyées. Ensuite, pour chaque demande d'indu correspondant à ce critère, l'application vérifie dans un fichier de pointage, toujours par l'intermédiaire du Gestionnaire de données, si AGETIP a reçu un indu depuis une de ses sources. Après quelques contrôles supplémentaires pour s'assurer du bien-fondé du lien entre la demande et l'indu, l'état de l'indu dans la base est changé en K – Indu reçu. Le fichier de pointage est lui aussi mis à jour.

Pointage des indus envoyés

Dans l'étape suivante, la procédure est presque identique, sauf que l'application vérifie les demandes d'indu en attente (donc basculées au matin par le programme Annul), grâce à un tableau .dbf, afin de vérifier si les factures relatives aux demandes d'indu ont été réglées par les mutuelles.

Ces trois grandes étapes de traitement sont protégées par des try-catch, qui intercepteront les erreurs potentielles et les feront apparaître clairement dans une MessageBox. En cas d'erreurs, un fichier texte logue la date et l'heure, ainsi que le descriptif de l'erreur pour une enquête ultérieure approfondie qui pourra amener à une réparation de bug.

Fonctionnement normal

Une fois les traitements effectués, l'application charge dans le Datagrid les demandes d'indus prêtes à être envoyées aux mutuelles. Un simple clic sur le bouton de génération d'email va générer un fichier texte, contenant toutes les lignes nécessaires à la mutuelle pour l'établissement d'un indu, précédées par des balises. Ces balises ([SUBJECT], [BODY]) seront interprétées par un programme d'envoi d'email situé sur une autre machine du réseau de l'entreprise.

L'application va alors produire un fichier prêt à l'emploi à l'endroit où le programme suivant pourra l'utiliser.

L'utilisateur de l'application peut rechercher, sélectionner et appliquer un état choisi à une demande d'indu. Les demandes d'indus correspondant aux critères renseignés lors de la recherche s'afficheront dans le DataGrid. Un clic sur une ligne sélectionnera la demande d'indu et permettra de changer son état grâce à la liste en bas à droite. La demande d'indu changera d'état, et l'adresse email changera en conséquence, et pourra se répercuter sur les autres indus liés à cette adresse.

	Demande envoyée	22-06-2021 16:40:19	23-06-2021	Appliquer un état aux Indus sélectionnés	<input type="checkbox"/> Affecter l'Adresse
	Demande envoyée	22-06-2021 15:52:15	23-06-2021	Remboursement non reçu	Appliquer
	Demande envoyée	22-06-2021 14:40:09	23-06-2021		
	Demande envoyée	22-06-2021 11:39:52	23-06-2021		
	Indu reçu de la Caisse	11-06-2021 18:42:09	23-06-2021		
	Indu reçu de la Caisse	14-06-2021 16:47:14	23-06-2021		

Préparer Emails

Par exemple, si AGETIP reçoit un avis de non délivrance d'un email, l'adresse utilisée est donc erronée. En appliquant l'état I – Adresse inexistante à cet indu, l'adresse qui lui est attachée changera pour le même état, et de fait, tous les indus liés à cette adresse changeront pour cet état.

:19	23-06-2021
:15	23-06-2021
:09	23-06-2021
:52	23-06-2021
:09	23-06-2021
:14	23-06-2021

Appliquer un état aux Indus sélectionnés ☐ Affecter l'Adresse

Appliquer

Emails

- Demande enregistrée
- Annulation retirée
- En attente de contrôle
- Demande refusée
- Remboursement non reçu**
- Facture rejetée
- Demande contrôlée
- Demande envoyée
- Adresse Caisse inexistante
- Indu rejeté par la Caisse
- Indu reçu de la Caisse
- Retirée par le professionnel
- Demande d'Indu de test/demo
- Email erroné pour la Caisse
- Indu préalablement reçu

Améliorations notables

Pour plus de lisibilité dans le code, j'ai proposé de créer une classe qui servirait de matrice de traduction d'états, c'est-à-dire qu'en renseignant dans le code un énumérable en texte clair, il sera traduit en lettre unique appropriée lors de l'exécution, ce qui permet donc de rendre le code plus clair à lire, sans risquer d'impacter la logique de l'exécution. Cette classe comporte donc des méthodes chargées de traduire une lettre d'état en énumérable clair, et une autre méthode fait l'inverse.

Pour le confort des utilisateurs, j'ai rédigé un fichier d'aide .chm, que j'ai inclus dans la barre de titre, sous la forme d'un point d'interrogation. Ainsi, un utilisateur ayant une question ou un doute pourra se référer à l'aide sous forme résumée ou détaillée.

De plus, j'ai utilisé un PredicateBuilder afin de construire le prédicat de recherche, selon les valeurs renseignées dans les textbox prévues à cet effet. Cette technique permet de ne procéder qu'à une seule requête pour avoir le résultat souhaité, là où une procédure classique consisterait à requêter la liste entière et à la filtrer selon les valeurs recherchées.

Suite aux premiers retours sur l'utilisation de l'application, j'ai apporté quelques retouches et améliorations de type « quality of life ». Par exemple une recherche peut être exécutée en écrivant ou en copiant/collant une valeur dans une des cases de recherche et en appuyant sur Entrée. De la même façon, sélectionner un état à rechercher dans la liste déroulante produira automatiquement une recherche. Les états que l'utilisateur ne devrait pas être capable d'appliquer à une demande d'indu ont été supprimés de la liste prévue à cet effet.

Gestion des adresses des Caisses

ID	Nom	Email	Etat de l'Adresse	Créateur	Date de création	Contrôleur	Date de
27			Validée		19-03-2021 17:07:11		02-04-20
28			Validée		19-03-2021 17:57:00		02-04-20
29			Validée		22-03-2021 10:06:01		02-04-20
30			Validée		22-03-2021 18:22:38		22-04-20
31			Validée		23-03-2021 15:32:18		22-04-20
32			Validée		24-03-2021 09:43:41		22-04-20
33			Validée		24-03-2021 16:21:13		22-04-20
34			Validée		24-03-2021 17:58:11		14-06-20
35			Inexistante		24-03-2021 18:14:40		30-03-20
36			Validée		25-03-2021 13:42:00		02-04-20
37			Inexistante		25-03-2021 14:51:36		02-04-20
38			Contrôlée		25-03-2021 17:29:08		26-03-20
39			Inexistante		26-03-2021 12:07:10		29-03-20
40			Demande Envoyée		26-03-2021 16:23:12		06-05-20
41			Inexistante		26-03-2021 17:29:01		29-04-20

En attente : 0 Refusées : 3 Non contrôlées : 0 Validées : 29 Inexistantes : 30

Indu reçu de la Caisse 11-06-2021 18:42:09 23-06-2021

Indu reçu de la Caisse 14-06-2021 16:47:14 23-06-2021

Recherche d'Adresse

IDContact: Nom:

Etat de l'Adresse: Email:

Créateur: Contrôleur:

Date de création: Date de contrôle:

Sélectionner une date [15] Sélectionner une date [15]

Rechercher

Appliquer un état aux Adresses sélectionnées

Préparer Emails

Interface utilisateur sur le site internet

Frontend

Il a été validé que les professionnels de santé doivent pouvoir consulter l'avancement de leurs demandes d'indu. A cet effet, une page a été créée sur le site web AGETIP. Cette page doit contenir les informations de toutes les demandes d'indu en cours pour ce professionnel, ainsi qu'un moyen de les annuler si elles n'ont pas encore été traitées.

J'ai créé une page `DemandesIndus.aspx`, ce qui a généré la page associée `DemandesIndus.aspx.cs` et `DemandesIndus.aspx.designer.cs`. La page est liée à la `MasterPage` du site web, et j'y ai tout de suite inclus les standards du site (les cases montrant les messages d'erreur et de succès, le fil d'Ariane, etc...). Une page similaire à celle-ci a été créée pour les Adresses.

J'y ai ajouté un `GridView`, avec des `asp:BoundField` comme colonnes, contenant chacune une information à propos d'une demande. Les colonnes sont dimensionnées en pourcentages afin de garder à peu près la même structure selon la largeur de la fenêtre. La dernière colonne est une `asp:TemplateField` contenant un bouton vierge. Un clic sur le bouton déclenche un script Javascript, déjà utilisé ailleurs dans le site, qui stocke l'ID du bouton dans une variable pour le code-behind, de façon à savoir quel bouton a été cliqué.

Vos demandes d'Indus

Accueil / Vos demandes d'Indus

Etat des demandes d'Indus à transmettre aux Caisses par AGETIP.

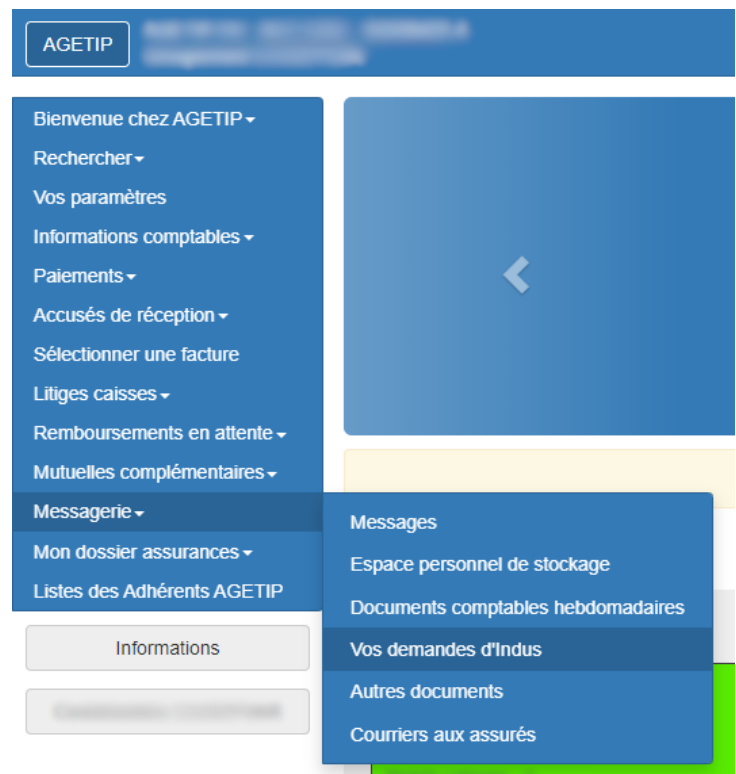
Liste des demandes d'Indus

Facture - ▾	Caisse - ▾	Numéro de lot - ▾	Adresse - ▾	Message - ▾	Date de création - ▾	Date de mise à jour - ▾	Etat actuel - ▾	Action - ▾
00100000	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		01 avril 2021	08 avril 2021	Demande envoyée	
00100001	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		19 mai 2021	20 mai 2021	Demande envoyée	
00100002	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		25 mai 2021	26 mai 2021	Demande envoyée	
00100003	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		03 juin 2021	16 juin 2021	Indu reçu de la Caisse	
00100004	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		05 juin 2021	16 juin 2021	Indu reçu de la Caisse	
00100005	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		07 juin 2021	16 juin 2021	Indu reçu de la Caisse	
00100006	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		10 juin 2021	17 juin 2021	Demande envoyée	
00100007	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		19 juin 2021	21 juin 2021	Facture rejetée	
00100008	CAISSE D'INDUS	001	01 rue des... (adresse fictive)		21 juin 2021	22 juin 2021	Demande envoyée	

J'ai ensuite rajouté quatre fenêtres modales, invisibles à première vue, et qui interviendraient après un clic sur un des boutons de la dernière colonne, dans le but de confirmer le choix de l'utilisateur.

Des boutons ont été rajoutés sur le côté gauche, permettant d'afficher une fenêtre modale informative, et de naviguer entre cette

page et la page des Adresses, construite donc sur le même modèle. L'accès à la page des demandes d'indus se fait via le navigateur à gauche du site (ou dans un menu burger si la fenêtre est trop étroite), contenu dans la MasterPage.



Backend

Dans la partie back se trouve d'abord le code que l'on trouve sur toutes les pages du site, c'est-à-dire celui permettant de récupérer les informations de l'utilisateur, et une redirection automatique si l'on essaie d'accéder à la page sans être connecté.

Une structure est définie pour contenir chaque information qui s'affichera dans le GridView :

```
private struct LigneGridViewDemandesIndus
{
    public string Facture { get; set; }
    public string Caisse { get; set; }
    public string NumLot { get; set; }
    public string Adresse { get; set; }
    public string Message { get; set; }
    public string DateCréation { get; set; }
    public string DateMiseAJour { get; set; }
    public string Etat { get; set; }
}
```

La fonction Page_PreRender appelle une seule méthode, qui consiste à remplir le Gridview. Cette méthode commence par récupérer les demandes d'indus effectuées par l'utilisateur connecté, toujours par le biais du Gestionnaire de données, puis pour chaque demande, charge un objet de la structure contenant toutes les informations.

De plus, le bouton de la dernière colonne est généré par le code-behind en définissant le RecNo de la demande comme l'ID du bouton, et, selon les cas, génère un glyphicon dans le bouton (un bouton OFF, une flèche « répéter », une poubelle, une flèche vers la droite, ou un sablier) dépendant de l'action effectuable sur la

Chaque fenêtre modale est assignée à un ou plusieurs boutons. Un script Javascript récupère le numéro de la facture du bouton cliqué et le valorise dans la fenêtre modale pour confirmer la demande de l'utilisateur. Cliquer sur Valider déclenchera une des 4 méthodes qui agissent sur les demandes d'indus. Pour respecter nos obligations de sécurité, le code-behind vérifie les valeurs rapportées par le script Javascript, avant lancer la procédure demandée. Le message d'erreur ou de succès est valorisé en conséquence du résultat de l'action choisie.

57

Tests

Tout au long du développement de ce projet, j'ai effectué des tests en parallèle afin de m'assurer du bon fonctionnement de chaque partie. J'ai été grandement aidé par deux éléments ; d'une part les bases de données étant locales je disposais d'un contrôle total, sans risque d'effacer des données de production, et d'autre part l'existence d'une version des données de test/développement du site web qui sont en réalité une copie de celles du site de production, effectuée chaque matin.

Le site de développement/debug contient des données réalistes et à jour, et se comporte exactement comme le site de production. Il est ainsi possible d'effectuer toutes les manipulations que feraient des professionnels de santé et d'en observer les conséquences, sans aucun risque de compromettre les données de production, les données de test étant écrasées par celles de production quotidiennement.

C'est en utilisant cette fonctionnalité que j'ai simulé des dizaines de demandes d'indus fictives, tenté diverses actions relevant de la sécurité, et testé non seulement le fonctionnement des procédures, toutes les possibilités envisageables, mais aussi les retours à l'utilisateur, les messages d'erreur et de succès, la facilité d'utilisation, etc...

Déploiement

Après la réalisation de nombreux tests, le déploiement en production pouvait commencer. Il a été décidé de procéder par étapes afin de s'assurer du bon fonctionnement des procédures et de la satisfaction des usagers finaux.

Après avoir recréé les bases de données sur le serveur de production et changé la cible du Gestionnaire de données, plusieurs tests ont été effectués depuis le site de debug avant de le mettre en ligne en production. Afin de recueillir des avis, suggestions et éventuels retours de bugs, une condition a été codée afin de permettre à 4 personnes (des chefs de groupements pharmaceutiques) d'accéder à la nouvelle procédure.

Ces personnes, prévenues par mail, ont formulé plusieurs retours et suggestions d'améliorations, notamment au niveau UI, pour rendre l'interface plus complète et compréhensible. Après quelques tests effectués en production, la condition codée plus tôt a été retirée et remplacée par une autre autorisant tout professionnel de santé (ayant souscrit à AGETIP) à essayer la nouvelle fonctionnalité, uniquement sur le site de démonstration.

Cette version du site est similaire à la version de test du site, mais contrairement au site de test accessible uniquement en local, le site de démonstration est accessible n'importe où, grâce à un mot de

passee fourni dans les paramètres du compte du professionnel de santé. Ce principe permet à un professionnel de santé de faire tester le site à un confrère, et, dans le cas présent, une fonctionnalité.

Les collaborateurs d'AGETIP ont ensuite été briefés sur le fonctionnement de l'application interne et sur les étapes à suivre ; les conditions d'accès à la fonctionnalité ont finalement été levées sur le site de production. Après s'être assurés que tout fonctionnait bien, grâce à un monitoring pendant une semaine, le projet a été validé comme fonctionnel et terminé. Je reste néanmoins à l'affût de potentiels bugs qui pourraient remonter dans l'application interne ou sur le site.

Le déploiement de l'application interne s'est fait très facilement, à l'aide de l'outil de publication de Visual Studio qui a généré un Click-Once dans un dossier prévu à cet effet sur le serveur. Il a suffi ensuite de lancer l'exécutable sur les machines devant être équipées de cette application. Cette méthode est d'ailleurs bien pratique car elle vérifie la présence de mises à jour au démarrage, ce qui permet aux collaborateurs de profiter sans problèmes de la dernière version à jour.

CONCLUSION

L'année passée

En entreprise

D'une manière générale, j'ai été très satisfait de cette année en alternance. L'entreprise m'a très bien accueilli, j'ai pu y développer très rapidement mes compétences, chaque projet a été source de nouvelles techniques de développement, de nouveaux apprentissages. Etant donné que mes interactions avec mes collègues, en dehors de mon tuteur, ont été très peu nombreuses, j'envisage de saisir chaque future occasion possible pour améliorer ce point.

Par ailleurs continuer dans la voie du développement, continuer à apprendre comment coder plus efficacement et développer de nouvelles capacités, et pourquoi pas dans de nouveaux langages. L'entreprise souhaitant dans le futur développer une application mobile, étudier sérieusement les langages d'Android et d'iOS me semblent indispensables.

En centre de formation

Les cours au centre de formation se sont bien déroulés. Les leçons dispensées par les intervenants en droit du numérique, en SQL, en Framework, en ergonomie, en application mobile, etc... m'ont été bénéfiques, peut-être pas dans la gestion du projet présenté ici mais le seront dans le futur.

Le projet

Dans l'ensemble, j'estime avoir atteint les objectifs fixés par le cahier des charges. Les procédures sont relativement simples, pour les professionnels de santé comme pour les collaborateurs d'AGETIP, le code est globalement suffisamment bien écrit pour être évolutif et j'ai respecté les standards d'AGETIP, autant au niveau du design utilisateur que dans la logique du code.

J'ai beaucoup apprécié m'occuper d'un projet de A à Z, qui de plus couvre une vaste étendue de technologies différentes. Le fait de pouvoir avancer à mon rythme a été plutôt bénéfique, dans le sens où j'ai pu pousser la technologie de l'application plus loin que prévu, la rendant plus évoluée que prévue.