

Une image contenant texte

Description générée automatiquement

**Remerciements**

Je voudrais tout d’abord remercier Monsieur Godart Christophe, Directeur du département ICT (Technologies de l'information et de la communication) de Solidaris, pour nous avoir offert l’opportunité de faire ce stage au sein de Solidaris.

Je voudrais également remercier mon maître de stage, Monsieur Palmieri Benoit, directeur de projet du département ICT, pour son aide et son soutien tout du long du stage. Il a su nous encadrer et s’est fortement impliqué dans la préparation de la présentation orale de notre TFE.

Pour finir, je remercie Madame Dumont Anne, notre professeure-conseil, qui nous a énormément aidé dans la rédaction de notre TFE.

Table des matières

[Introduction 1](#_Toc72435984)

[1 PRÉSENTATION DE L’ENTREPRISE 2](#_Toc72435985)

[1.1 Évolution de Solidaris 2](#_Toc72435986)

[1.2 Organigramme de Solidaris 3](#_Toc72435987)

[2 Position du problème 4](#_Toc72435988)

[2.1 Une mutualité c’est quoi ? 4](#_Toc72435989)

[2.2 Comparateur d’assurance complémentaire 5](#_Toc72435990)

[2.3 Environnement de travail informatique 7](#_Toc72435991)

[3 Outils 8](#_Toc72435992)

[3.1 Langages utilisés 8](#_Toc72435993)

[3.2 Webstorm 9](#_Toc72435994)

[3.3 NodeJS 9](#_Toc72435995)

[3.4 NPM 10](#_Toc72435996)

[3.5 NestJS 10](#_Toc72435997)

[10](#_Toc72435998)

[3.6 Angular 10](#_Toc72435999)

[3.7 TypeOrm 11](#_Toc72436000)

[3.8 PrimesFacesNG 11](#_Toc72436001)

[3.9 Wamp 11](#_Toc72436002)

[3.10 Swagger 12](#_Toc72436003)

[3.11 Postman 12](#_Toc72436004)

[3.12 Trello 13](#_Toc72436005)

[4 Analyse et conception 14](#_Toc72436006)

[4.1 Maquettes 14](#_Toc72436007)

[4.2 Cas d’utilisation 17](#_Toc72436008)

[4.3 Analyse de la base de données 20](#_Toc72436009)

[5 Implémentation 25](#_Toc72436010)

[5.1 Page de connexion 25](#_Toc72436011)

[5.2 Import-export fichier Excel (Administrateur) 26](#_Toc72436012)

[5.3 Tableau de gestion des avantages (Administrateur) 27](#_Toc72436013)

[5.4 Filtres du comparateur d’assurance complémentaire (Conseiller) 34](#_Toc72436014)

[5.5 Le comparateur d’assurance complémentaire 40](#_Toc72436015)

[6 Conclusion 42](#_Toc72436016)

[7 Bibliographie 43](#_Toc72436017)

# Introduction

Ce travail de fin d’études va détailler le processus de développement du comparateur d’assurance complémentaire développé pour le service marketing de Solidaris Mons Wallonie Picarde.

Cette application a pour objectif de comparer les offres d’assurances complémentaires des différentes mutualités wallonnes. Elle a été demandée par le service marketing de Solidaris pour aider leurs conseillers sur le terrain lors d’entrevue avec un éventuel affilié.

Ce travail sera divisé en plusieurs parties.

Premièrement une présentation de Solidaris et du système d’assurance maladie en Belgique sera nécessaire pour bien comprendre les besoins et utilités du comparateur d’assurance complémentaire.

Ensuite les besoins énoncés par le service marketing seront présentés. Cela permettra de bien comprendre leurs besoins et l’utilité de l’application.

L’étape suivante détaillera les outils utilisés pour réaliser ce projet.

La quatrième partie portera sur l’analyse du problème et les différentes solutions envisagées. Différentes maquettes réalisées pour mettre en place l’application seront présentées.

La dernière partie portera sur l’implémentation. Il y sera détaillé le fonctionnement de l’application accompagné de code commenté. Ceci permettra de comprendre les techniques employées et les solutions apportées par l’application.

Enfin, la conclusion détaillera mon évolution personnelle tout au long de ce projet.

# PRÉSENTATION DE L’ENTREPRISE

# 

« Solidaris » est une mutualité belge qui fait partie de l’Union nationale des mutualités socialistes.  
Actuellement, elle compte environ 3 millions d’affiliés à travers la Belgique.

L’union nationale des mutualités socialistes est séparée en plusieurs fédérations dont certaines sont regroupées sous le label Solidaris.

Voici les différentes fédérations présentes en Wallonie et a Bruxelles :

– Solidaris Mutualité – Brabant wallon

– Fédération des mutualités socialistes du Brabant

– Solidaris Mons – Wallonie picarde

– Solidaris, Mutualité socialiste du Centre – Charleroi et Soignies

– Mutualité Solidaris – mutualité socialiste et syndicale de la province de Liège

– Mutualité socialiste du Luxembourg

– Mutualité Solidaris – province de Namur

– Mutualis

Le présent travail concerne la fédération Solidaris Wallonie-Picarde. Son siège social se situe à la Rue du Fort numéro 48, à Ath.   
Le comparatif d’assurance complémentaire sera développé dans leur service IT situé Avenue des Nouvelles Technologies 24 à Frameries.

## Évolution de Solidaris

La première « société de secours mutuels socialistes », « La Solidarité », est créée en 1869, à Fayt-lez-Manage, dans le Hainaut.

Elle consiste en une organisation composée de membres qui versent une cotisation.   
Ces cotisations sont rassemblées dans un pot commun pour pouvoir s’assurer mutuellement en cas de maladie ou de blessure.

Dans la foulée, d’autres sociétés du même type ont été créées et gérées directement par les travailleurs qui voulaient assurer leurs arrières dans un contexte de vie difficile.

Il faudra attendre la fin du 19e siècle pour que les différentes « sociétés de secours mutuels socialistes » se regroupent en entités régionales, appelées « fédérations ». Elles s’officialisent donc et commencent à prendre de plus en plus d’importance.

C’est en 1894 que la première loi sur les mutualités est votée. Elle élargit leur champ d’action, mais ne garantit pas encore le financement des caisses maladie.

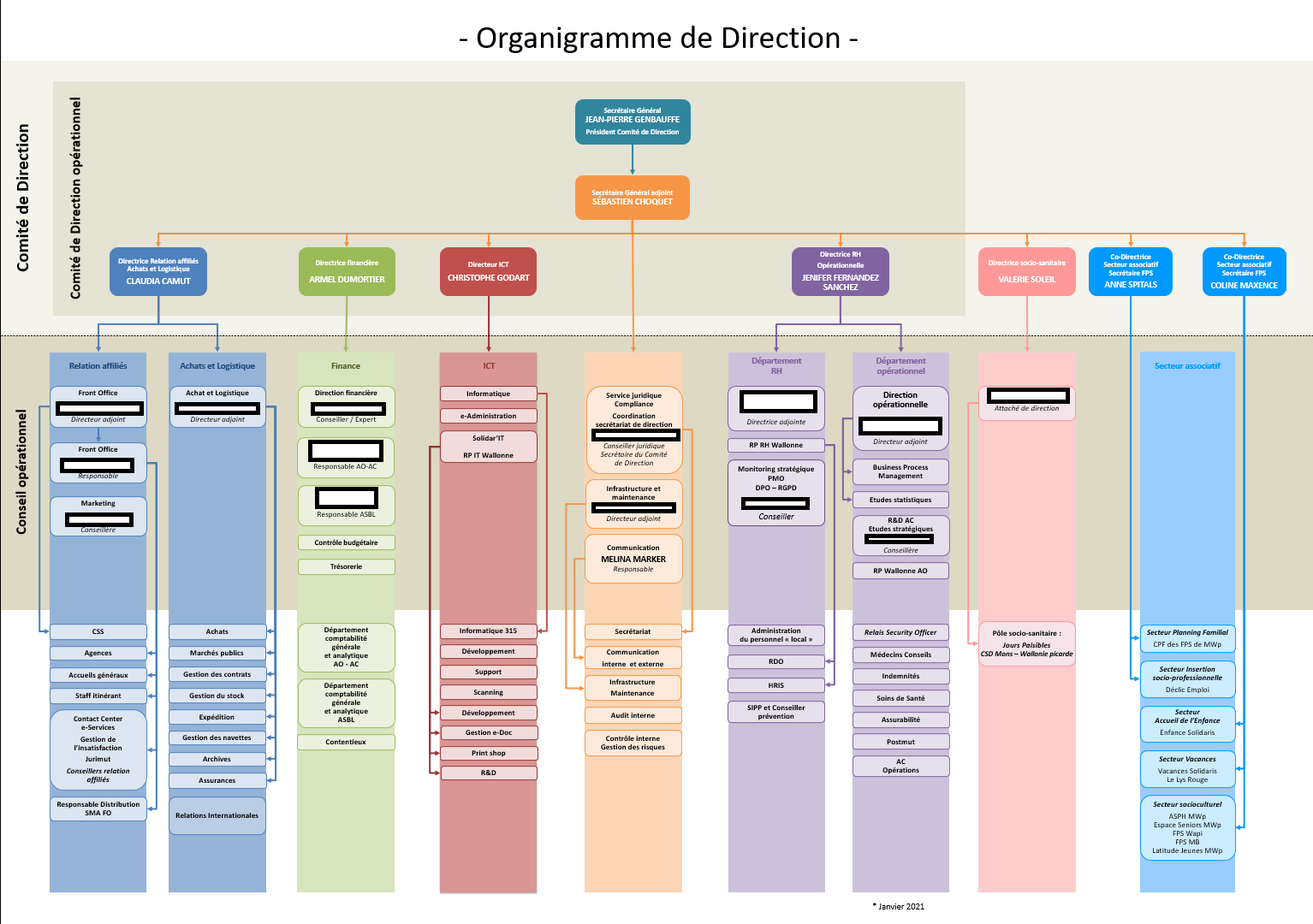
En 1913, pour permettre un meilleur contrôle des mutualités et une organisation plus efficace, les « Union Nationale des fédérations des Mutualités socialistes de Belgique » sont créées et chaque fédération prend une couleur politique.  
« La solidarité » rejoint l’union Socialiste.

Le 28 décembre 1944, un arrêté-loi met en place la Sécurité sociale belge.   
Les Unions Nationales deviennent légalement chargées de la gestion de l’Assurance Maladie obligatoire.   
Les mutualités sont donc officiellement associées à l’état belge. Plus de pouvoir leur est accordé.

C’est donc réellement à partir de 1944 que Solidaris prend la forme d’une mutualité telle qu’on la connaît aujourd’hui.

## Organigramme de Solidaris

L’organigramme suivant représente les différents départements de Solidaris.



*Fig 1 : Organigramme de direction des mutualités socialistes*

# Position du problème

Avant de poser le problème relatif à ce travail, il est essentiel de comprendre le rôle d’une mutualité.

## Une mutualité c’est quoi ?

Le rôle principal d’une mutualité est d’assurer un remboursement partiel des frais médicaux et de santé ainsi que d’octroyer un complément financier en cas d’incapacité de travail.   
Ces remboursements sont effectués dans le cadre de l’Assurance obligatoire (AO), régie par la loi.

En Belgique, les mutualités font partie de la Sécurité sociale belge gérée par l’ONSS (Office National de la Sécurité Sociale).  
C’est l’ONSS qui perçoit l’ensemble des cotisations et les redistribue entre les différents organismes chargés du paiement de l’AO (organismes assureurs).   
Le financement de la Sécurité sociale est assuré par trois sources : les cotisations des travailleurs (salariés, indépendants, fonctionnaires), les cotisations des employeurs et les subventions de l’État.   
C’est ce financement qui permet à l’Assurance obligatoire d’être gratuite et identique pour tous.

En Belgique, toute personne ayant un revenu ou étant âgée de plus de 25 ans doit s’inscrire auprès d’une mutualité comme titulaire.

Pour cela, elle doit obligatoirement s’inscrire auprès d’un organisme assureur.

L’AO est assuré par les organismes assureurs belges :

* L’Union nationale des mutualités neutres,
* L’Union nationale des mutualités chrétiennes,
* L’Union nationale des mutualistes socialistes,
* L’Union nationale des mutualités libérales,
* L’Union nationale des mutualités libres,
* CAAMI,
* Caisse de soins de santé de HR Rail.

En plus de l’AO, une mutualité fournit également des interventions supplémentaires dans le cadre de l’Assurance complémentaire (AC). Cette dernière couvre des services et avantages non compris dans l’Assurance obligatoire comme le remboursement de lunettes ou la possibilité de partir en vacances sportives à moindre coût.

C’est la cotisation payée par un affilié à sa mutualité qui lui donne accès à ces services supplémentaires et qui est utilisée pour financer l’assurance complémentaire.

Dans le cadre de l’assurance complémentaire, les Unions nationales sont chargées de superviser le bon fonctionnement de ses différentes fédérations.

L’OCM, elle, s’occupe de vérifier le respect des règles comptables, administratives et financières dans les mutualités.

Ce qu’il est important de retirer de ce qui précède, c’est que l’assurance maladie obligatoire   
(AO) est la même, peu importe l’organisme assureur choisi.   
Pour se démarquer, les mutualités se font concurrence entre elles à l’aide de l’assurance complémentaire.

## Comparateur d’assurance complémentaire

Comme ce sont les assurances complémentaires qui influencent le choix de mutualité des éventuels affiliés, il a donc été demandé de créer un comparateur d’assurance complémentaire facilement utilisable sur le terrain par les conseillers Solidaris.

Ce comparateur est séparé en deux parties distinctes, la partie utilisée par les conseillers Solidaris et la partie administrateur pour le service marketing. Les conseillers veulent pouvoir utiliser le comparateur sur le terrain lors d’entrevue avec d’éventuels affiliés. Le comparateur doit leur permettre de facilement avoir accès aux informations des avantages qui pourraient intéresser l’éventuel affilié.

De leur côté, le service marketing doit pouvoir facilement encoder les différents avantages tous les ans lorsque la liste des avantages pour l’année à venir a été approuvée.

Il y a donc deux parties distinctes au comparateur d’assurance complémentaire.

La partie pour le conseiller :

* Formulaire pour filtrer les avantages à comparer
* Un tableau comparatif pour comparer les avantages sélectionnés grâce aux filtres.
* Un tableau reprenant uniquement les avantages de Solidaris sous format PDF pour le transmettre à l’éventuel affilié.

Une image contenant table

Description générée automatiquement

*Fig 2 : Exemple de comparateur fourni par Solidaris*

La partie administrateur pour le service marketing :

* Téléchargement d’un fichier Excel vierge à remplir une fois par an avec tous les avantages pour cette année.
* Lecture d’un fichier Excel rempli pour enregistrer tous les avantages de l’année et les modifications importantes.
* Un tableau de gestion des avantages pour des modifications simples.

Pour ces deux parties, un serveur communiquant avec la base de données des avantages sera nécessaire. Une partie visuelle et une interface graphique à destination de l’utilisateur sous forme de site internet. Cette partie visuelle sera connectée au serveur pour pouvoir afficher et utiliser les avantages récupérés en base de données.

## Environnement de travail informatique

Solidaris utilise un Intranet sur lequel les employés se connectent et ont accès à divers outils les aidant dans leur travail. Cet Intranet est disponible dans les locaux à Frameries en se connectant au réseau ou grâce à des cartes 4G connectées à ce même réseau.

# Outils

Pour la réalisation du comparateur d’assurance complémentaire, plusieurs outils ont été utilisés.

Certains de ses outils ont été imposés par Solidaris : Nest Js, Angular, Swagger, Postman, GitHub, Gitfork, Mysql.

## Langages utilisés



### HTML

Le langage HTML est un langage de programmation utilisant des balises. Il permet la création de pages web personnalisées.



### CSS

Le langage CSS (Cascading Style Sheets) sert à insérer du style dans les pages web (HTML ou XML) d’un site.

Il permet à l’aide de classes de modifier la mise en forme des balises HTML. Il permet par exemple de modifier l’emplacement des éléments dans une balise, ou encore de modifier la taille du texte, sa couleur ou même sa police.

De plus il permet tout cela depuis un unique fichier. Il épure les balises HTML de tous ses styles et rend ainsi le code beaucoup plus lisible et agréable à modifier.

### Tendencias de JavaScript en 2019 - Agencia de Diseño y Desarrollo Web CloudStudioJavaScript

JavaScript est un langage de développement web. Il permet d’interagir avec un site en le rendant réactif ou en permettant à l’utilisateur d’interagir avec la page web.

Le code est exécuté du côté client et est souvent utilisé de pair avec le HTML et le CSS.

JavaScript possède de nombreuses libraires permettant d’implémenter facilement de nouvelles fonctionnalités.



### Typescript

TypeScript est un langage open source construit à partir du JavaScript lui ajoutant la possibilité de créer et de définir des types statiques.

Dans ce cas-ci il nous permettra par exemple de définir la structure d’un avantage et toutes les données qu’il doit contenir.

Les types fournissent un moyen de décrire la forme d’un objet, permettent une meilleure documentation et permettent à TypeScript de valider si notre code fonctionne correctement.

L’écriture de type est cependant optionnelle en TypeScript permettant d’unir la flexibilité des variables JavaScript avec la rigueur de la déclaration de types selon nos besoins.



### SQL

Le langage SQL est un langage utilisé en base de données. Il permet la création de bases de données et se base sur un principe de requêtes pour interagir avec cette dernière.



## Webstorm

Webstorm est un IDE pour les langages web de la suite Jetbrains. L’éditeur offre plusieurs possibilités telles que l’autocomplétion intelligente, l’inspection du code, la refactorisation du code. Il est aussi muni d’un débogué et facilite la création de tests unitaires.

Il possède aussi un terminal permettant la création rapide de nouveaux fichiers et composants précomplétés. Ce même terminal permet en plus l’installation rapide et efficace de nouvelles librairies.



## NodeJS

Node.js est une plateforme logicielle en JavaScript. Orientée vers les réseaux évènementiels elle permet le développement de serveur HTTP. NodeJS est un environnement bas niveau permettant l’exécution JavaScript sur un serveur.

## npm — WikipédiaNPM

NPM est le gestionnaire de paquets officiels de Node.js, il permet l’installation de libraires et gère les dépendances pour une application.

Il fonctionne à l’aide d’un terminal et est installé par défaut sur Node.js.



## NestJS

La partie serveur de l’application est codée en NestJS. NestJS est une infrastructure logicielle utilisant le langage de programmation TypeScript. Cet outil permet un développement de projet par modules. Ces modules sont facilement transférables entre projets et l’ajout de fonctionnalités est très simple, car chaque module correspond à une fonctionnalité ou à un concept.

Par exemple les avantages auront leurs propres modules qui permettront de créer, supprimer, modifier ou récupérer des avantages en base de données.

Cela permet que toutes les fonctionnalités ne soient pas entremêlées entre elles et que l’application puisse donc facilement être modifiée.

## Getting started with Angular 5. Hi, are you a beginner in Web… | by Onejohi | Medium

## Angular

Angular est une infrastructure logicielle côté client basé sur le TypeScript. Il permet la création d’applications web accessible via une unique page web ce qui permet de fluidifier l’expérience de l’utilisateur.

Il possède une architecture permettant de séparer les données, le visuel et les actions ce qui permet une meilleure gestion des responsabilités et une meilleure organisation de plus gros projets.

## Add custom operators to typeorm - DEV CommunityTypeOrm

TypeORM est une librairie nestJS permettant l’écriture et la création de bases de données directement à partir du code.

Comme c’est le code qui va créer les tables de notre base de données, il ne faudra pas créer au préalable la base de données.



## PrimesFacesNG

PrimefaceNG est un ensemble de composants graphiques pour Angular permettant la création de formulaires et de tables de données.

Il contient des visuels préconçus et fonctionnels.

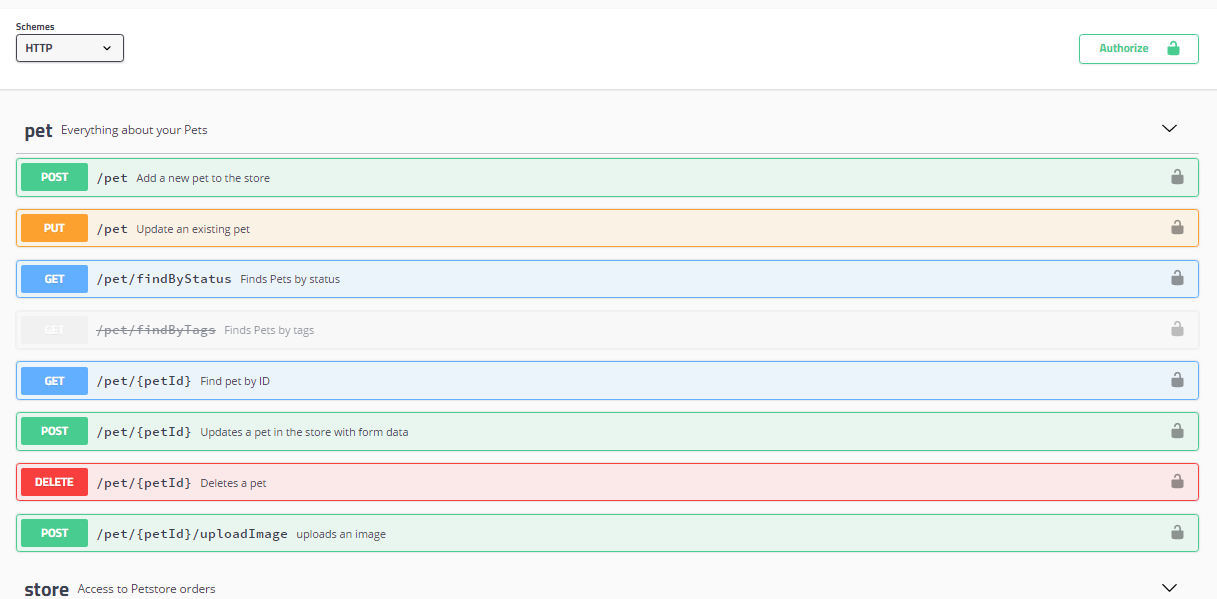
PrimeFacesNG a également ses propres classes css permettant de mettre en page tous ses composants.



## Wamp

Wamp est un environnement de travail permettant la simulation de serveur Mysql, Maria Db et Apache en local. Il permet aussi la connexion d’un serveur local avec des scripts PHP. C’est lui qui va nous permettre de créer et d’utiliser notre base de données.

## Swagger



Swagger est une API permettant le test de requêtes HTTP directement depuis un navigateur à travers une interface graphique.

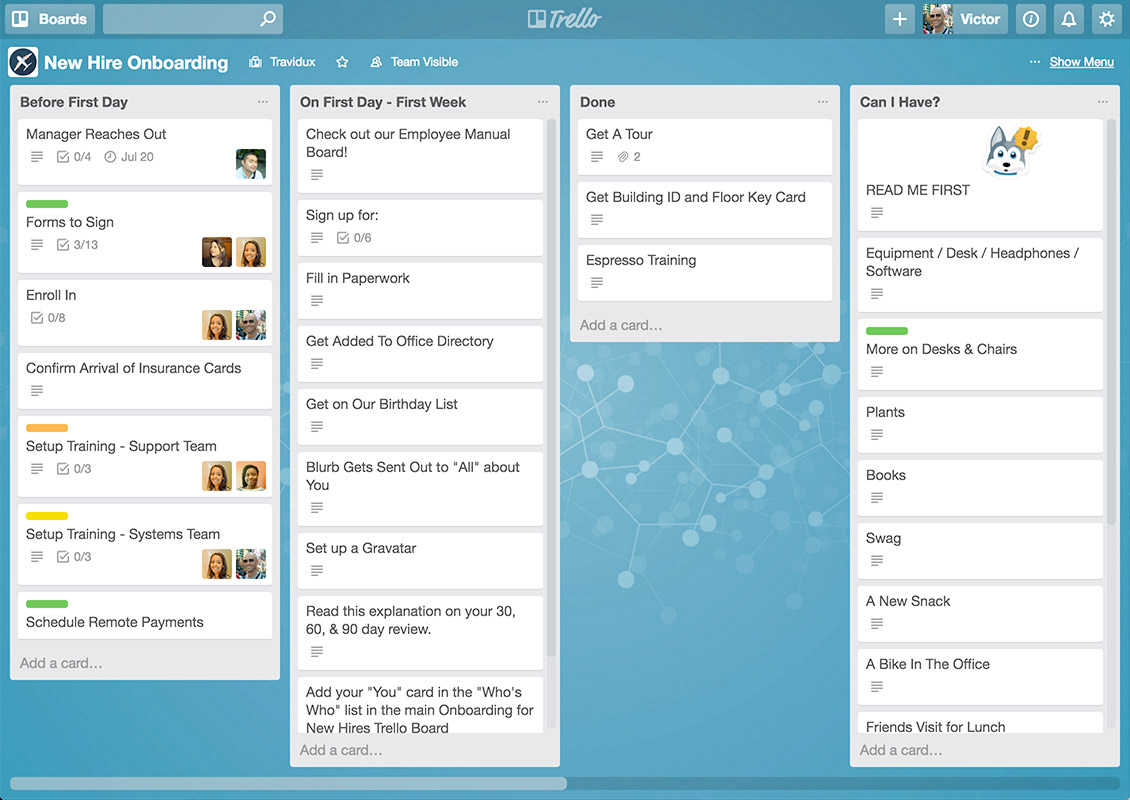
Swagger permet de tester le bon fonctionnement du serveur.



## Postman

Postman est un API permettant de tester les requêtes HTTP. Il permet de tester si la communication entre le serveur et notre base de données se passe bien. Il permet de tester des requêtes en y insérant un body de test et renvoie des détails supplémentaires sur les erreurs rencontrées.

## Trello



Trello permet d’organiser un travail ou projet sous forme de tableau. Chaque ligne du tableau représente une étape à réaliser et il est possible d’affecter des gens à chaque étape.

# Analyse et conception

## Maquettes

La première étape est de créer des maquettes. Ces maquettes ont été présentées au service marketing pour qu’ils puissent se faire une idée de l’application qu’ils utiliseront.

Ces maquettes ont aussi pour but de bien définir les différentes parties du projet, c’est une première ébauche du design général de l’application.

Une maquette pour chaque partie de l’application a été créée, une pour la partie utilisée par les conseillers et une pour la partie utilisée par le service marketing.

### Visuel du conseiller

Le conseiller aura accès à un tableau comparatif des avantages. Des filtres seront disponibles pour n’afficher que les avantages pouvant intéresser l’éventuel affilié.

Les avantages ne répondant pas au profil de l’éventuel affilié ne seront donc pas affichés.

Une image contenant table

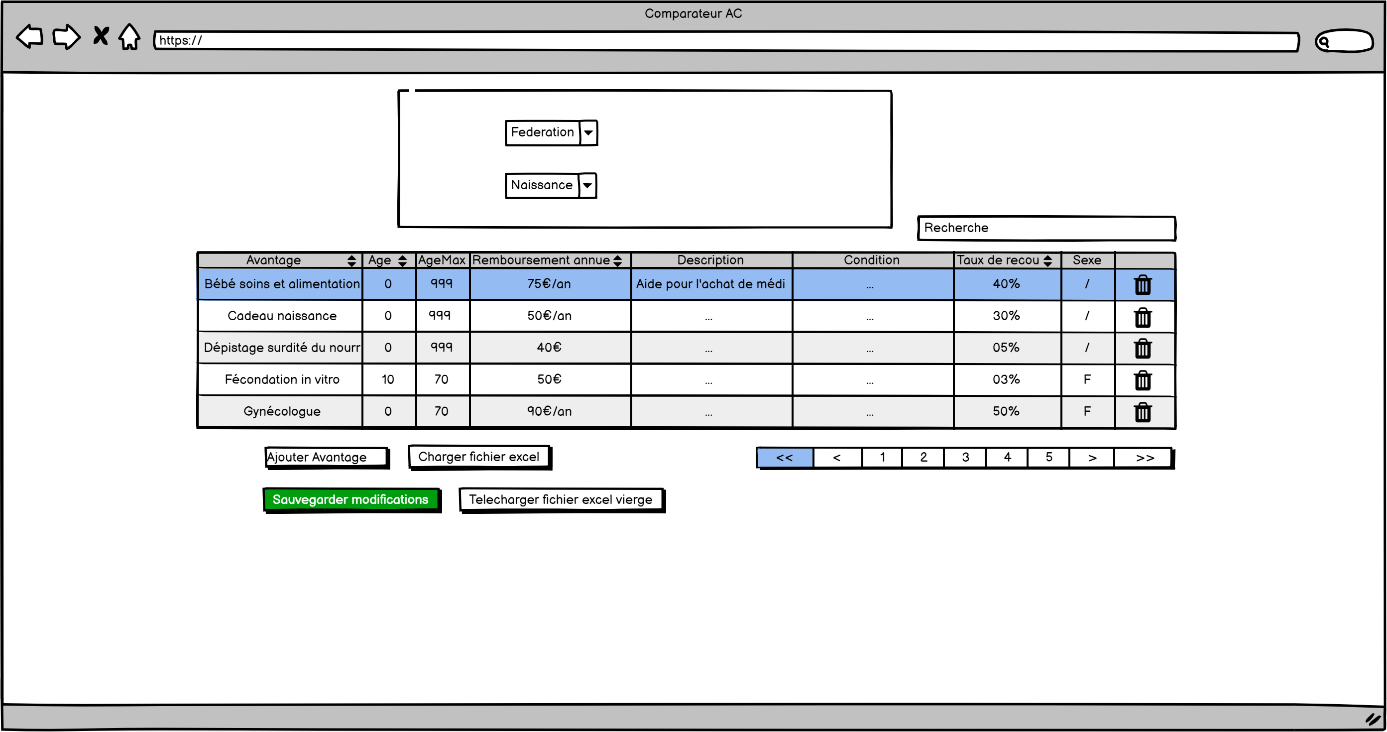
Description générée automatiquement

*Fig 3 : Interface du visuel du conseiller*

### Visuel de l’administrateur

Un administrateur devra pouvoir créer, modifier ou supprimer des avantages en base de données. Le tableau ci-dessous permet donc de modifier à la volée un avantage. Il permet aussi de créer un nouvel avantage.

Ce tableau affichera tous les avantages de cette année pour toutes les mutualités wallonnes. C’est également ici qu’il sera possible de télécharger le fichier Excel vierge et de lire un fichier Excel rempli avec tous les avantages pour l’année à venir.



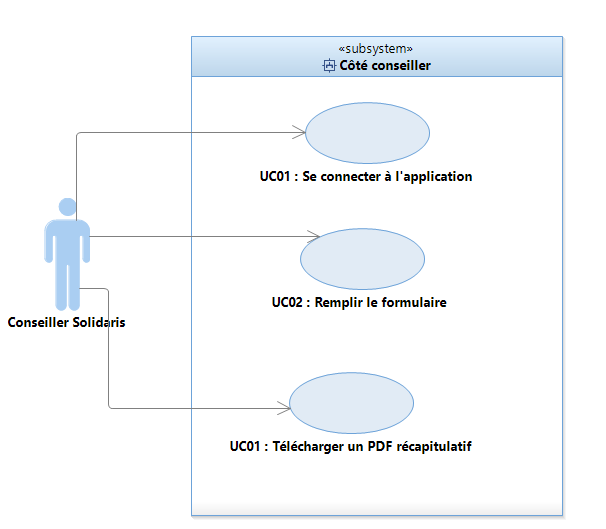
*Fig 4 : Visuel de l’administrateur*

## Cas d’utilisation

Il y a deux cas d’utilisation possible.

* Un conseiller lors d’une entrevue avec un éventuel affilié.
* Un administrateur lorsqu’il faut mettre à jour la liste des avantages.

### Cas d’utilisation du conseiller Solidaris



*Fig 5: Cas d’utilisation d’un conseiller*

UC01 : Se connecter à l’application

Le conseiller devra se connecter à l’application à l’aide de son compte professionnel Solidaris. Une fois connecté, il aura accès à la partie conseiller du comparateur.

UC02 : Remplir le formulaire

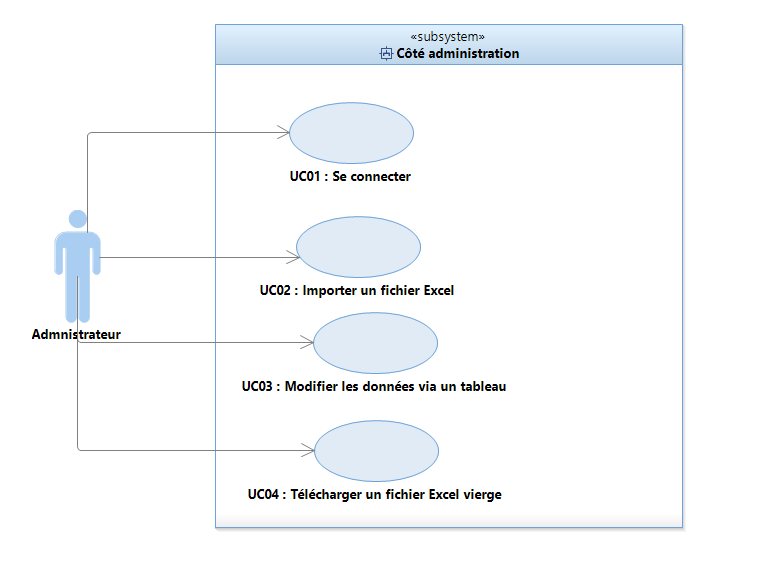
Une fois connecté, le conseiller devra remplir le formulaire de filtres. Dès que le formulaire sera rempli, le tableau comparatif des avantages s’affichera pour permettre au conseiller de comparer les avantages des différentes mutualités belges.

Le conseiller pourra donc s’aider des informations obtenues pour essayer de convaincre l’éventuel affilié de s’affilier chez Solidaris.

UC03 : Télécharger un PDF récapitulatif

Une fois l’entrevue avec l’éventuel affilié terminée, le conseiller aura la possibilité de télécharger un fichier PDF reprenant uniquement les avantages proposés par Solidaris. Il pourra aussi imprimer ce PDF ou l’envoyer par mail à l’éventuel affilié.

### Cas d’utilisation de l’administrateur



*Fig 6: Cas d’utilisation de l’administrateur*

UC 01 : se connecter

Comme pour le conseiller, l’administrateur pourra se connecter à l’application à l’aide de son compte professionnel Solidaris.

UC 02 : Importer un fichier Excel

L’administrateur pourra importer un fichier Excel, comprenant les différents avantages pour les fédérations de Wallonie. Les avantages déjà présents en base de données seront marqués comme supprimé. De ce fait, seuls les avantages qui viennent d’être importés seront affichés dans le comparateur d’assurance complémentaire.

L’import d’un fichier Excel sera utilisé lorsque de grosses mises à jour sont à apporter à la base de données.

UC03 : Modifier les données via un tableau

Ce tableau de gestion de données permettra à l’administrateur de modifier, créer ou encore supprimer un avantage de la base de données. Ce tableau de gestion des avantages sera utilisé pour de petites modifications ou l’ajout de quelques avantages en cours d’année.

UC 04 : Télécharger un fichier Excel vierge

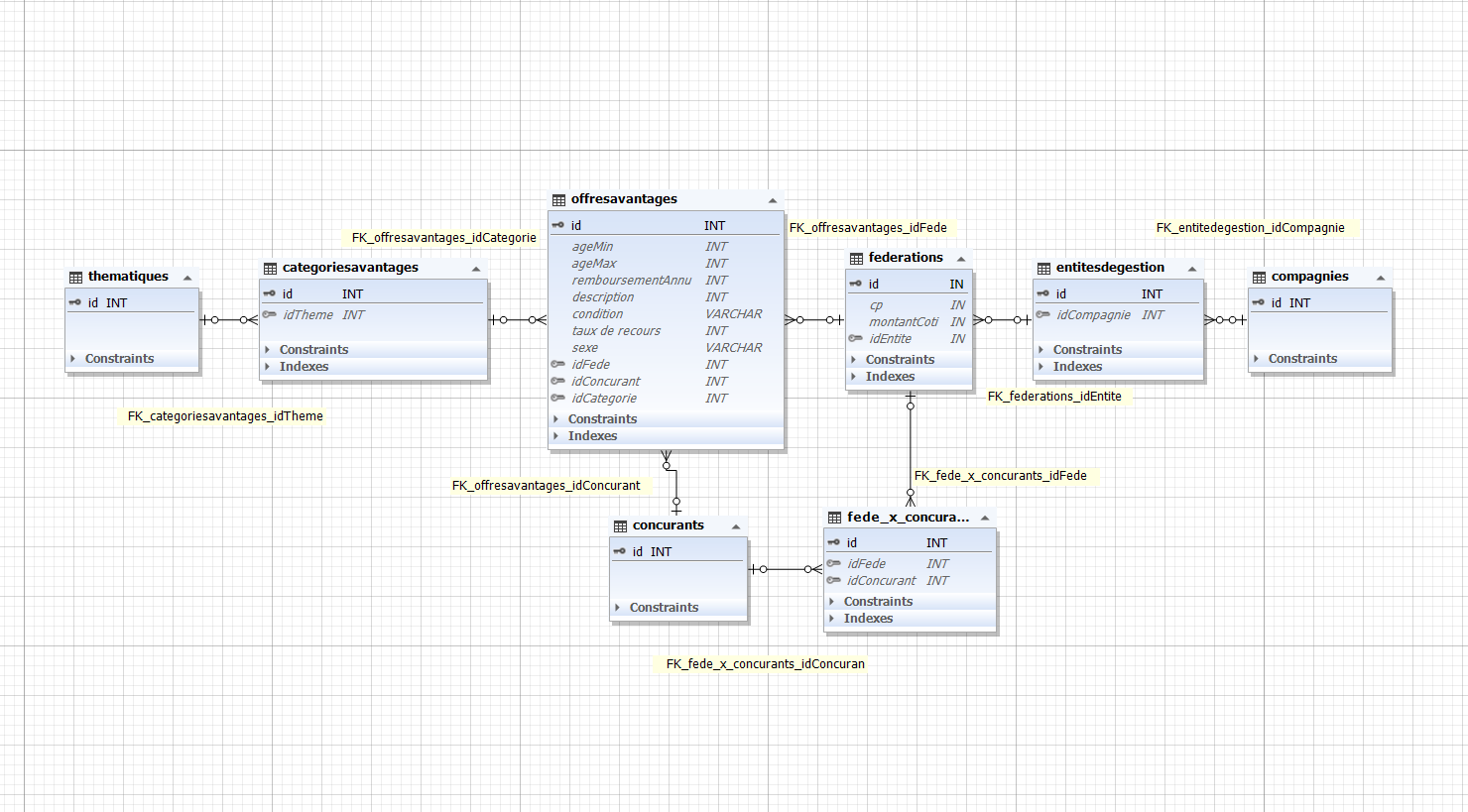
Lorsque l’administrateur devra récolter les avantages d’une nouvelle année. Il pourra télécharger un fichier Excel vierge afin de le remplir avec les avantages de toutes les mutualités wallonnes de cette année. Ce fichier sera ensuite utilisé pour importer les avantages en base de données à l’aide du cas d’utilisation UC 02.

## Analyse de la base de données

La base de données a dû être créée vu qu’elle n’existait pas avant le début de ce projet. Cette base de données permettra de contenir tous les avantages pour toutes les mutualités wallonnes.

### Schéma de la base de données

Un schéma décrivant à quoi devra ressembler la base de données a été communiqué par le service marketing de Solidaris.

Ce schéma sera utilisé comme base pour définir la structure et les relations des tables de la base de données.

*Fig 7 : Schéma de structure de la base de données*

### Détails des tables de la base de données

La base de données contient des compagnies qui seront rattachées à une entité de gestion. Chaque entité de gestion contiendra plusieurs fédérations. Par exemple, la compagnie « Solidaris » est rattachée à l’entité de gestion « Union nationale des mutualités socialiste ». Cette entité de gestion possède à son tour une liste de fédération telle que « Solidaris — Mons Wallonie picarde ».

Chaque fédération possédera une liste des avantages qu’il propose, ces avantages seront classés dans diverses catégories qui appartiennent à des thématiques plus générales.

Par exemple la fédération « Solidaris — Mons Wallonie picarde » possède des avantages sur la thématique de « l’Hospitalisation ». Cette thématique dispose davantage de différentes catégories telles que « Frais d’hospitalisation » ou encore « Chambre privée ».

**La table « avantages/advantages »**

Cette table contient les avantages.

Chaque avantage sera relié à une catégorie et a une fédération.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nom | Description | Type | Taille | Est relié à |
| id | Identifiant de l’avantage | Int | 11 |  |
| maxAge | L’âge maximum pour bénéficier de l’avantage | Int | 10 |  |
| minAge | L’âge minimum pour bénéficier de l’avantage | Int | 10 |  |
| annualRepayment | Rembourser annuellement | Int | 10 |  |
| Condition | Condition pour bénéficier de l’avantage | Varchar | 500 |  |
| rateOfAppeal | Taux de recours de l’avantage | Int | 10 |  |
| Gender | Le genre de la personne qui peut bénéficier de l’avantage (femme/homme) | Varchar | 5 |  |
| idFede | Identifiant de la fédération | Int | 11 | federations.id |
| idConcurrant | Identifiant du concurrent | Int | 11 | concurrants.id |
| Description | Description de l’avantage | Varchar | 1000 |  |
| idCategory | Identifiant de la catégorie dont elle fait partie | Int | 11 | categories.id |

**La table « fédération/fédérations»**

Contient les fédérations de Wallonie.

Chaque fédération est reliée à une entité de gestion.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nom | Description | Type | Taille | Est relié à |
| Id | Identifiant de la fédération | Int | 11 |  |
| federationName | Nom de la fédération | Varchar | 250 |  |
| CP | Code postal de la fédération | Int | 11 |  |
| contribution | La cotisation mensuelle de la fédération | Int | 11 |  |
| entitiesManagementId | Identifiant de l’entité de gestion | Int | 11 | Entitymanagement.id |
| federationNumber | Le numéro de la fédération | Int | 11 |  |

**La table « entitedegestion/entitymanagement»**

Contient toutes les unions des mutualités.

Chaque union sera reliée à une compagnie.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nom | Description | Type | Taille | Est relié à |
| Id | Identifiant de l’entité de gestion | Int | 11 |  |
| managementEntityName | Nom de l’entité de gestion | Varchar | 250 |  |
| companyId | Identifiant de la compagnie | Int | 11 | companies.id |

**La table « compagnies/companies»**

Cette table contient tous les organismes assureurs de Wallonie.

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Description | Type | Taille |
| Id | Identifiant de la compagnie | Int | 11 |
| companyName | Nom de la compagnie | Varchar | 250 |

**La table « Concurrents/concurrents»**

Contient les noms des concurrents des fédérations de Solidaris.

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Description | Type | Taille |
| Id | Identifiant du concurrent | Int | 11 |
| concurantName | Nom du concurrent | Varchar | 250 |

**La table « federation\_concurrant\_concurrants »**

Table de jointure permettant de savoir quel concurrent fait partie de quelle fédération. Une fédération peut avoir plusieurs concurrents et un concurrent peut appartenir à plusieurs fédérations.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nom | Description | Type | Taille | Est relié à |
| federationId | Identifiant de la fédération | Int | 11 | federation.id |
| concurrantId | Identifiant du concurrent | Int | 11 | concurrant.id |

**La table « thématiques/thèmes»**

Contient toutes les thématiques possibles pour les catégories.

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Description | Type | Taille |
| Id | Identifiant de la thématique | Int | 11 |
| name | Nom de la thématique | Varchar | 250 |

**La table « categoriesavantages / catégories»**

Contient toutes les catégories d’avantages.

Chaque catégorie est reliée à une thématique.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nom | Description | Type | Taille | Est relié à |
| Id | Identifiant de la catégorie | Int | 11 |  |
| idTheme | Identifiant de la thématique | Int | 11 | Themes.id |
| name | Nom de la catégorie | Varchar | 250 |  |

**La table « User/Utilisateur»**

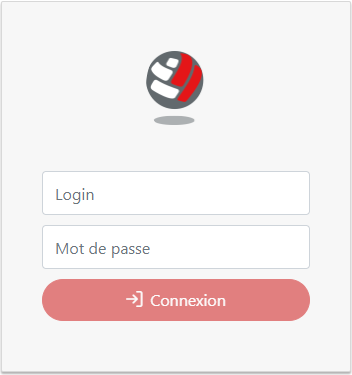
Contient tous les utilisateurs pouvant se connecter à l’application.

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Description | Type | Taille |
| Id | Identifiant de l’utilisateur | Int | 11 |
| username | Nom d’utilisateur | Varchar | 250 |
| Password | Mot de passe utilisateur | Varchar | 1000 |
| Permission | Permission d’administrateur | Bit | 1 |

# Implémentation

## Page de connexion

La page d’accueil du site est une fenêtre de connexion composée d’un champ pour le nom d’utilisateur et d’un champ pour rentrer son mot de passe.



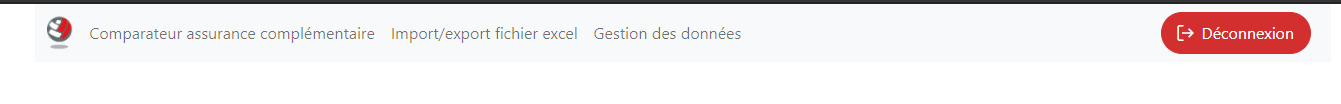
*Fig8 : fenêtre de connexion*

Il existe deux types de compte : les comptes administrateur et les comptes conseiller.

Une fois connecté à l’application, un menu de navigation s’affiche en haut de la fenêtre. Ce menu permet d’accéder aux différentes parties de l’application. De plus, ce menu s’adapte en fonction du type de compte connecté.

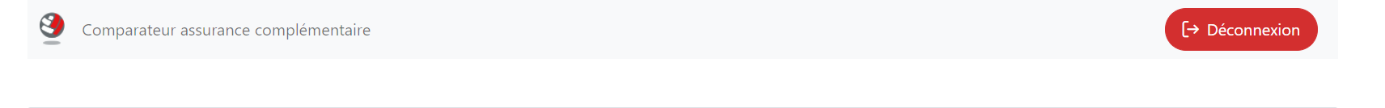
Les administrateurs peuvent accéder aux onglets import-export de fichier Excel et au tableau de gestion des avantages.

Le menu de navigation s’adapte donc pour afficher les parties de l’application auxquelles l’administrateur a accès.



*Fig9 : Menu Administrateur*

De la même façon, lorsqu’un conseiller se connecte, le menu de navigation s’adapte pour n’afficher que les parties auxquelles ont accès les conseillers, c’est-à-dire le comparateur d’assurance complémentaire.



*Fig10 : Menu Conseillers*

## Import-export fichier Excel (Administrateur)

Depuis cette page, il est possible pour un administrateur d’importer un fichier Excel contenant tous les avantages de Solidaris et de ses concurrents afin de les ajouter à la base de données.

Il est également possible de télécharger un fichier Excel vierge contenant la mise en page et les différents champs à remplir pour un avantage.

Le bouton « parcourir » permet de sélectionner le fichier Excel contenant les avantages pour cette année.

Le bouton « télécharger fichier Excel vierge », quant à lui, permet de télécharger un fichier Excel vierge contenant les noms des colonnes à remplir lorsque l’on veut importer une nouvelle liste d’avantages.

Une image contenant table

Description générée automatiquement

*Fig11 : Import/export fichier Excel*

## Tableau de gestion des avantages (Administrateur)

Le tableau de gestion des avantages permet à l’administrateur du site de modifier, supprimer, ajouter et visionner les différents avantages de Solidaris et de ses concurrents.

Une image contenant texte, capture d’écran, intérieur

Description générée automatiquement

2

6

7

5

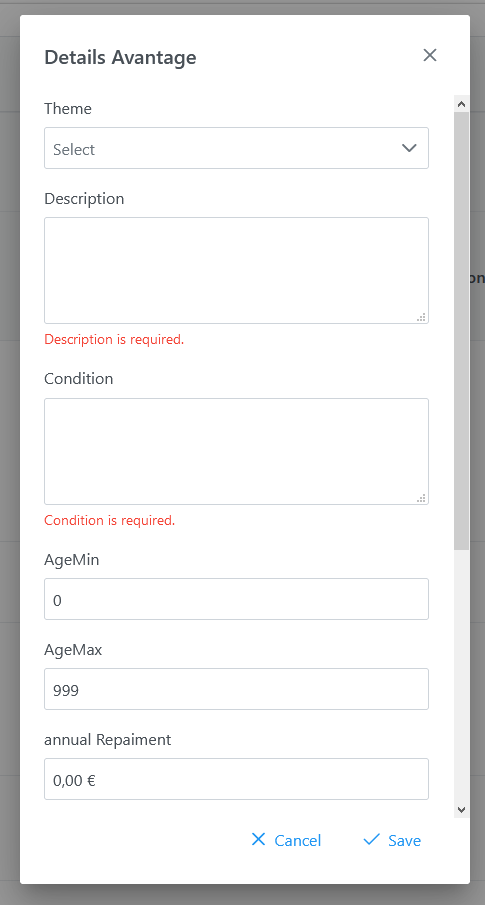
4

3

1

*Fig12 : Tableau de gestion des avantages*

(1) — Ce bouton permet d’afficher le formulaire d’ajout d’avantages et d’ajouter un nouvel avantage dans la base de données pour la catégorie sélectionnée.



*Fig13 : Formulaire d’ajout d’avantages*

(2) — Cette barre de recherche permet de filtrer les avantages présentés dans le tableau grâce à des mots-clés. Les filtres s’effectuent sur toutes les colonnes et vont chercher des apparitions des mots-clés encodés dans chaque colonne du tableau. Par exemple, écrire « hôpital » dans la barre de recherche filtrera les avantages pour ne montrer que ceux qui possèdent le mot « hôpital » dans leur description, nom de catégorie ou condition.

(3) — Ce bouton permet de supprimer tous les avantages sélectionnés dans le tableau de la base de données. La suppression n’est cependant pas définitive. L’élément est alors marqué comme étant supprimé dans la base de données. Cependant, il est toujours possible de le récupérer en cas de fausse manipulation.

(4) — En cliquant sur le titre d’une colonne, il est possible de trier les avantages par ordre croissant ou décroissant pour les colonnes contenant des chiffres et par ordre alphabétique pour les colonnes contenant du texte.

(5) — Le bouton vert permet d’ouvrir un formulaire pour modifier les informations de l’avantage choisi. Lorsque le formulaire est envoyé, les données de l’avantage sont modifiées dans la base de données. Le bouton jaune permet de supprimer cet avantage de la base de données.

*Une image contenant texte

Description générée automatiquement*

*Fig14 : Formulaire de modification d’avantages*

(6) — Ces boites à cocher permettent de sélectionner les avantages du tableau.

(7) — Pour rendre le tableau plus lisible, seuls dix avantages peuvent être affichés en même temps. Le tableau est donc séparé en plusieurs pages navigables à l’aide de cette barre.

Pour pouvoir afficher tous les avantages, il faut les récupérer dans la base de données.   
Il est donc nécessaire d’avoir un moyen de communiquer avec le serveur qui pourra à son tour communiquer avec la base de données.

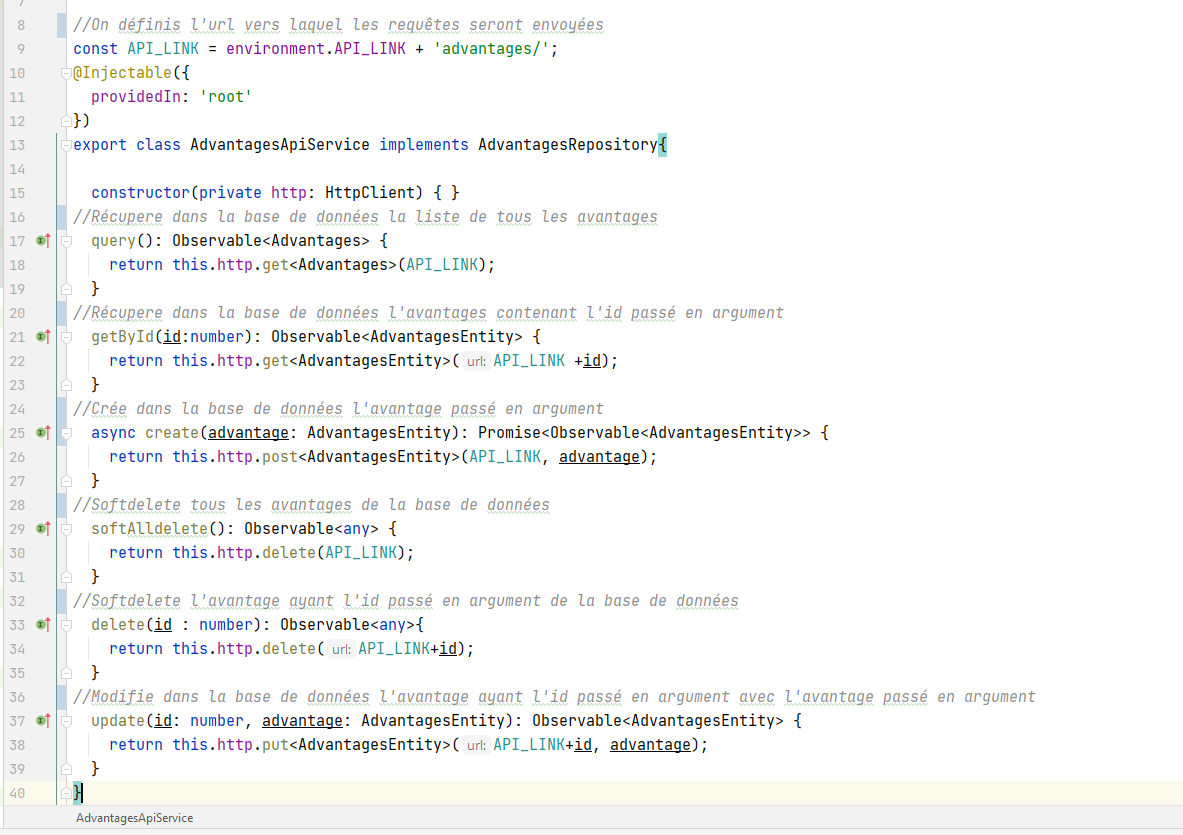


*Fig15 : Code de l’interface « Repository » des avantages*

Tout d’abord, pour ce faire, une interface « repository » est créée. Cette dernière a pour rôle de définir les différentes requêtes possibles pour un élément (dans ce cas-ci pour les avantages). Par son intermédiaire, il est possible, par exemple, de récupérer tous les avantages à l’aide de la méthode query (), ou de crée un avantage à l’aide de la méthode create ().

L’interface « repository » défini donc ce qu’il est possible de faire avec un avantage. Par exemple, ici getById (id : number) permet de récupérer l’avantage possédant l’id voulu.

Cette interface « repository » est implémentée à une classe « service ». Cela veut dire que la classe « service » doit obligatoirement définir le fonctionnement des méthodes de l’interface « repository ».



*Fig16 : Code de la classe « Service » des avantages*

La classe « Service » s’occupe de communiquer avec la base de données. Contrairement à l’interface « Repository », qui s’occupe uniquement de lister les actions possibles sur un avantage en base de données, la classe « Service » définit le fonctionnement de ces actions et contacte le serveur à l’aide d’une requête HTTP.

Par exemple, lorsqu’on fait appel à la méthode getById (id : number), une requête get HTTP est envoyée au serveur. Le serveur qui reçoit cette requête contacte à son tour la base de données pour récupérer l’avantage possédant l’id donné entre parenthèses. Une fois l’avantage récupéré, le serveur communique à la classe « Service » l’avantage à l’aide de cette même requête HTTP. Cet avantage peut alors être utilisé par le reste de l’application pour être, par exemple, affiché dans le tableau de gestion des avantages.

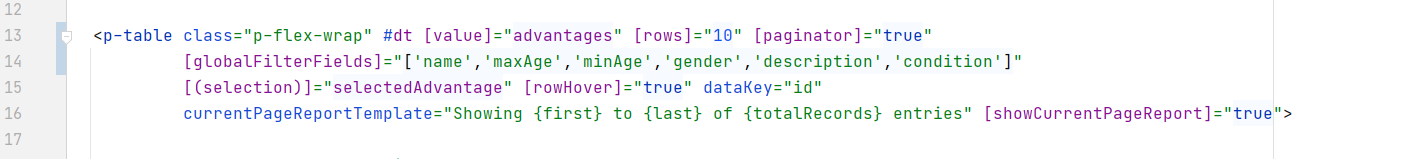


*Fig17 : Initialisation du tableau de gestion des avantages*

Le tableau de gestion des avantages est créé à partir de la liste de tous les avantages. Lorsqu’il faut afficher le tableau de gestion des avantages, la méthode ngOnInit () est exécutée. Cette méthode fait appel à la classe « Service » afin de récupérer la liste de tous les avantages contenus dans la base de données.

Lorsque le tableau de gestion des avantages est modifié, la classe « Service » est appelée pour enregistrer les modifications en base de données.

Il faut ensuite afficher le tableau contenant tous les avantages.Il faut ensuite afficher le tableau contenant tous les avantages.



*Fig18 : Affichage du tableau de gestion des avantages*

La liste des avantages est alors passée à la balise <p-table> qui contient le tableau.

Cela permet aux balises contenues dans le tableau d’avoir accès aux avantages récupérés en base de données à l’aide de la classe « service ».

On peut donc afficher les différents avantages directement depuis le code HTML.



*Fig19 : Affichage d’un avantage dans le tableau de gestion des avantages*

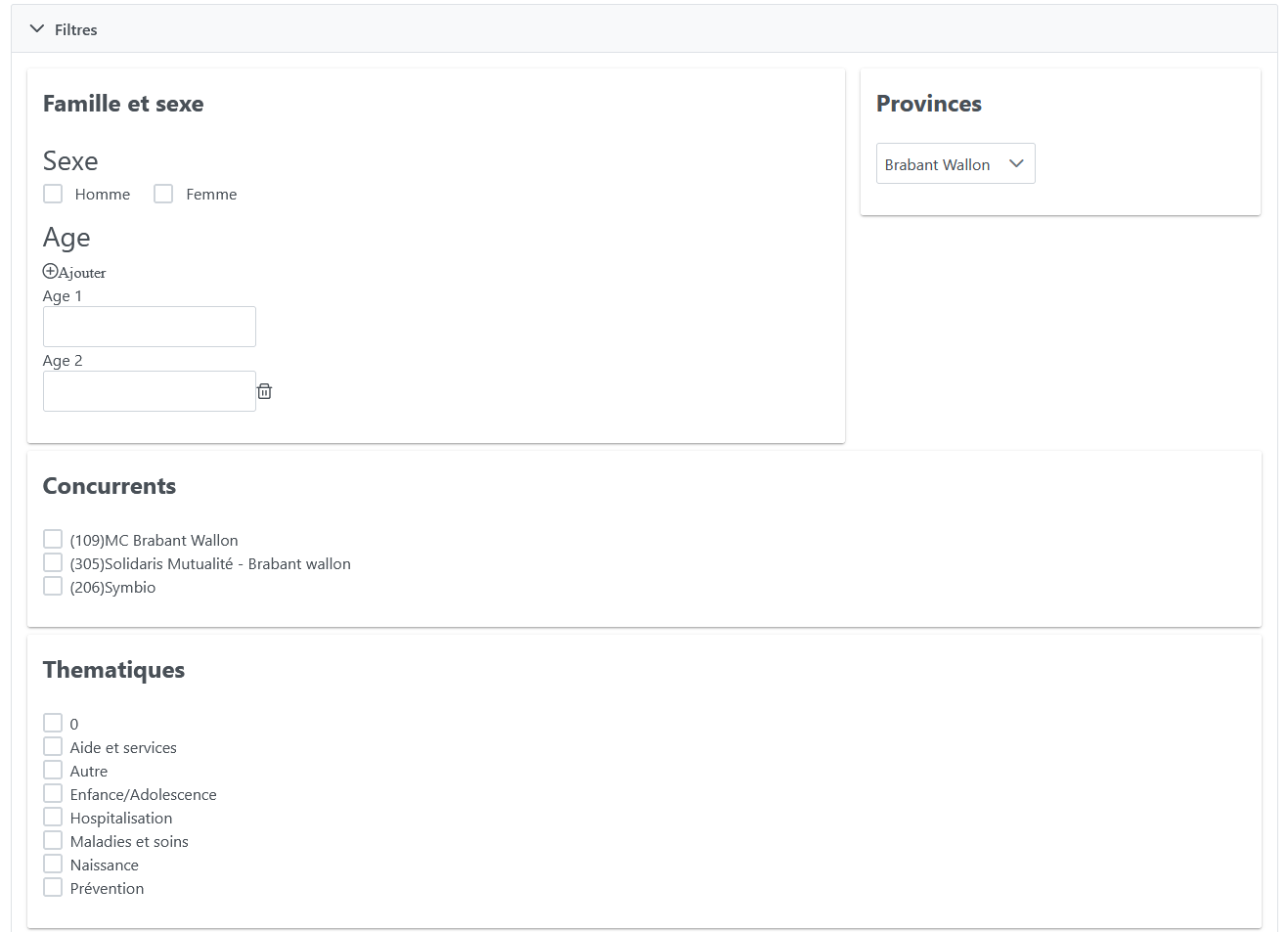
La liste de tous les avantages est donc parcourue et pour chaque avantage une ligne est rajoutée au tableau de gestion. Les informations de l’avantage sont alors affichées dans les colonnes de cette même ligne.

Pour résumer, lorsque le tableau de gestion des avantages doit être affiché par le biais du serveur, la liste de tous les avantages est récupérée. Cette liste est ensuite parcourue pour créer une ligne par avantages dans le tableau. Chaque ligne correspond à un avantage et les informations de cet avantage sont rentrées dans les colonnes pour chaque ligne.

## Filtres du comparateur d’assurance complémentaire (Conseiller)

Le conseiller Solidaris peut utiliser une série de filtres pour afficher uniquement les avantages pouvant intéresser l’éventuel affilié.

1



6

5

4

3

2

*Fig20 : Filtres du comparateur d’assurance complémentaire*

(1) — Les filtres sont contenus dans un accordéon permettant de les cacher une fois remplis.

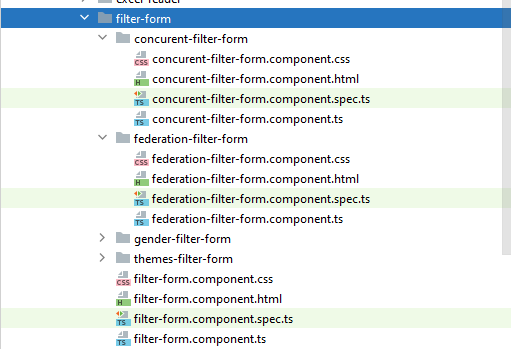
(2) — Deux cases à cocher permettent de sélectionner le genre du potentiel affilié.

(3) – Zone de texte permettant de rentrer l’âge du potentiel affilié. Il est possible d’ajouter plusieurs âges afin de voir les avantages qui pourraient intéresser le reste de la famille de l’éventuel affilié.

(4) — Liste déroulante pour sélectionner la province où se situe l’éventuel affilié. Il est possible de sélectionner plusieurs provinces.

(5) — Liste des concurrents pour les provinces sélectionnées.

(6) — Liste des thématiques disponibles pour les différents avantages.



*Fig21 : Architecture des Filtres*

Pour faciliter la création du formulaire de filtres, un composant Filter-form est créé. Ce composant contient d’autres composants tels que themes-filter-form. Ces derniers composants sont donc les sous-composants de filter-form.

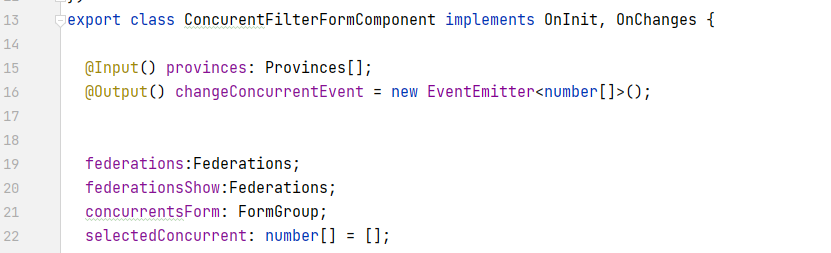
Dans le code HTLM de filter-form, chacun de ses sous-composants sont affichés dans des encadrés à l’aide de balises <p-card>.



*Fig22 : Composant Filter-form*

Lorsqu’un sous-composant de filter-form est modifié par l’utilisateur, un événement est créé pour prévenir filter-form de la modification. Ces évènements contiennent les valeurs rentrées dans les filtres du sous-composant.

Par exemple, lorsque la province « Liège » est sélectionnée dans le filtre « filter-form-province » un évènement « provinceEvent » est propagé. Cet évènement prévient alors le composant global « filter-form » que la province « Liège » a été sélectionnée. « Filter-form » prévient donc à son tour le composant contenant le filtre des concurrents que la province de Liège a été sélectionnée. De cette façon, le sous-composant sait qu’il doit afficher les concurrents se situant dans la province de Liège.



*Fig23 : Sous composant ConcurentFilterForm*

La province de Liège est alors ajoutée dans la variable provinces qui contient la liste de toutes les provinces sélectionnées. L’annotation @Input() précise que les provinces sélectionnées sont reçues par le composant global filter-form.

L’annotation @output() changeConcurrentEvent définit l’évènement qui a lieu chaque fois qu’un concurrent est sélectionné ou désélectionné. Cet évènement renvoie au composant filter-form une liste de nombres contenant les ids de tous les concurrents sélectionnés.

Par exemple, lorsque le concurrent Partenas — Liège est sélectionné, la liste des ids de tous les concurrents actuellement sélectionnés est communiquée au composant global filter-form.

Par exemple, lorsque le concurrent Partenas — Liège est sélectionné, la liste des ids de tous les concurrents actuellement sélectionnés est communiquée au composant global filter-form.

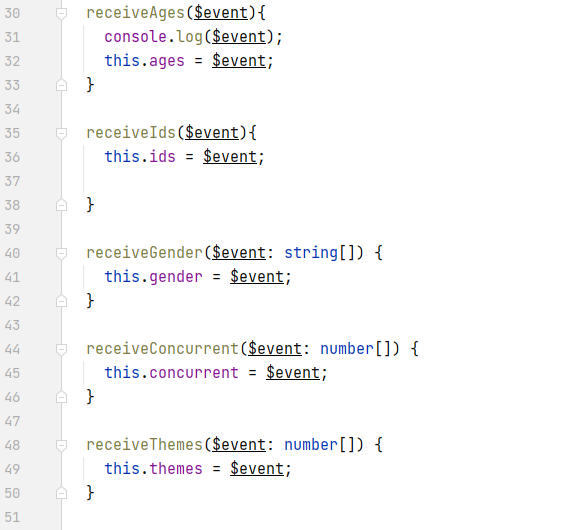


*Fig24 : Composant Filter-form.ts*

Le composant global filter-forme contient plusieurs variables qui vont accueillir les résultats des évènements émis par ses sous-composants.

Ces variables sont ensuite utilisées pour créer une instance de l’objet Filtres.

L’Élément @output(), quant à lui, définit l’évènement qui a lieu lorsque tous les filtres ont été remplis et qu’ils peuvent être envoyés au tableau du comparateur d’assurance complémentaire.

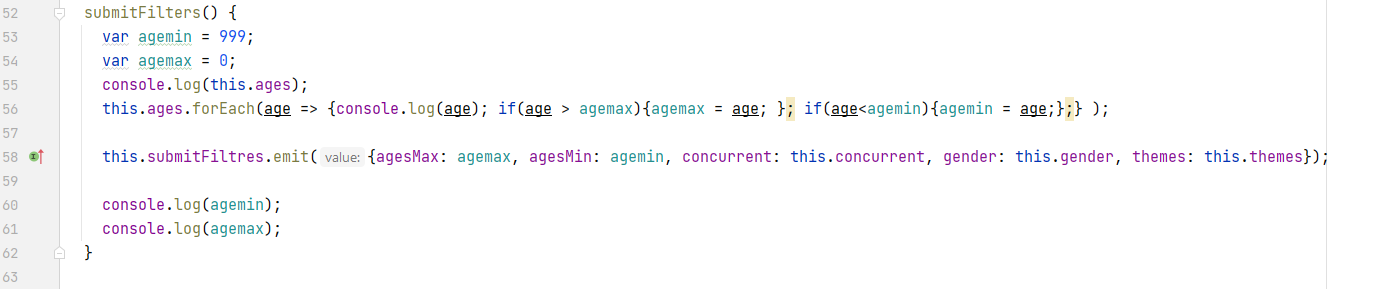


*Fig25 : Méthodes appelées lors de la réception des évènements*

Lorsque le composant filter-form reçoit des évènements de la part de ses sous-composants, on fait appel aux méthodes affichées ci-dessus.

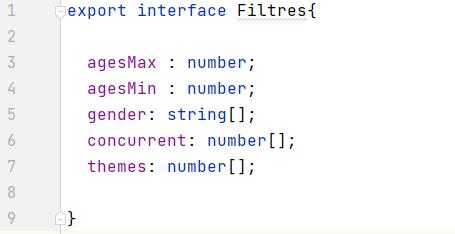
Le contenu des évènements se trouve dans la variable $event qui contient les valeurs rentrées dans chaque formulaire de filtre. Le contenu de $event est stocké dans les variables ages, gender, concurrent, thèmes et provinces de filter-form. Lorsque l’on clique sur le bouton d’application des filtres, ces différentes variables sont alors utilisées pour créer un objet Filtres qui est ensuite envoyé au tableau d’affichage des avantages à l’aide de l’évènement submitFiltres.

Dans le cas de notre exemple, le sous-composant concurrent-filter-form prévient le composant filter-form que le concurrent Partenas — Liège a été sélectionné. On fait alors appel à la méthode receiveConcurrent. Cette dernière a pour effet d’ajouter l’id de Partenas — Liège à la liste de concurrents de filter-form.



*Fig26 : Méthode SubmitFiltres()*

Lorsque tous les filtres ont été remplis et que le bouton « appliquer filtres » est pressé, la méthode submitFilters() s’exécute. Cette méthode envoie alors un objet de type Filtres au tableau du comparateur d’assurance complémentaire.

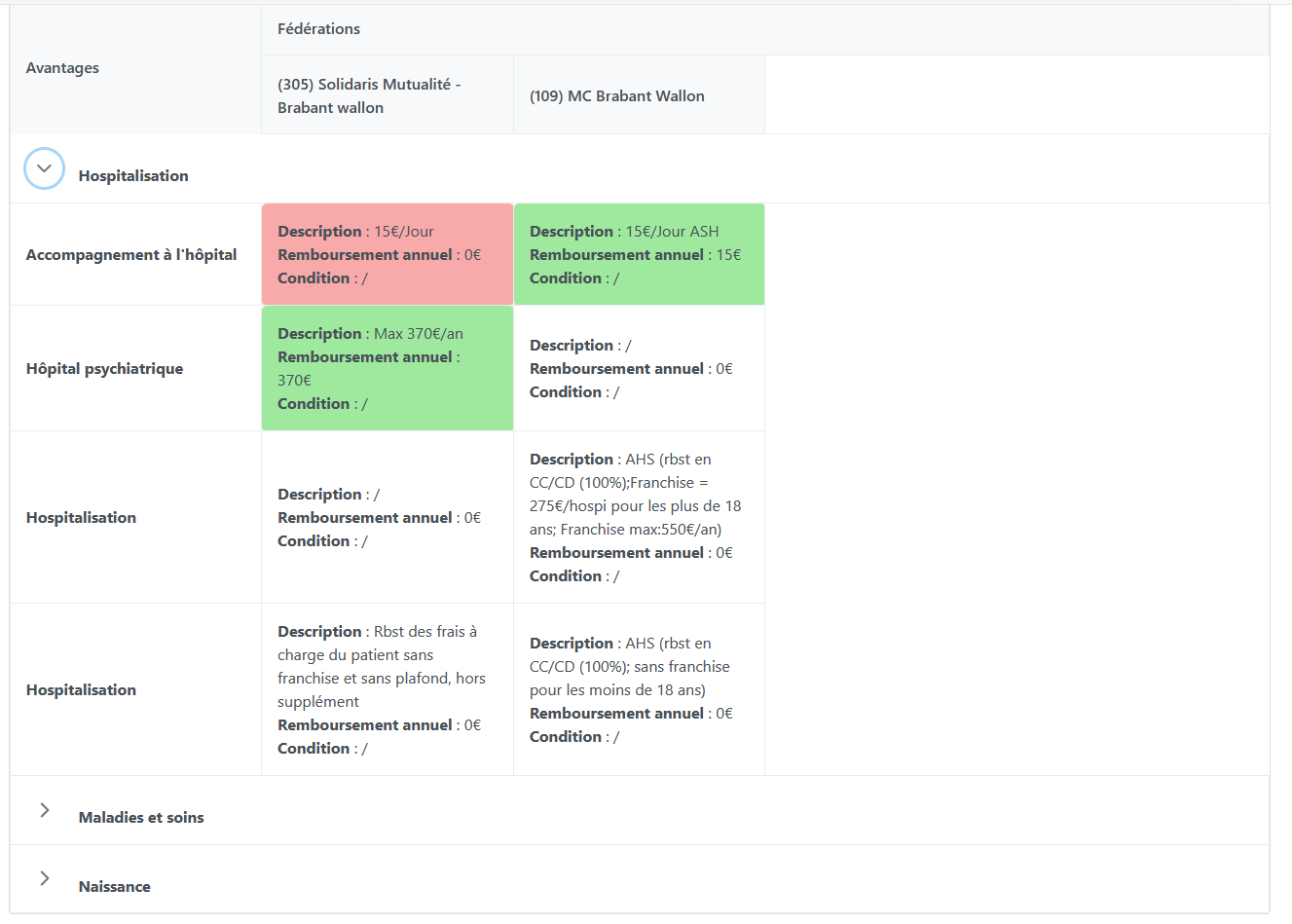


*Fig27 : Interface Filtres*

Pour ce faire, on utilise l’interface Filtres, qui permet de définir un objet capable de stocker les valeurs de tous les filtres, afin que l’événement submitFilter puisse transporter un objet Filtre contenant les valeurs de tous les filtres. Ainsi, submitFilter peut être envoyé au tableau du comparateur d’assurance complémentaire afin de filtrer les avantages à afficher.

## Le comparateur d’assurance complémentaire

Les filtres sélectionnés ont donc été communiqués au tableau d’assurance complémentaire qui affiche alors uniquement les avantages des concurrents et des thématiques sélectionnés. De plus, si certains avantages ne sont disponibles que pour certaines tranches d’âge, les âges sélectionnés dans les filtres, quant à eux, sont vérifiés pour savoir si cet avantage doit être affiché ou pas.



5

4

3

2

1

*Fig28 : Comparateur d’assurance complémentaire*

(1) — Les fédérations choisies dans les filtres sont affichées. Solidaris est toujours mise en première colonne.

(2) — Les avantages sont classés par thématique. Il est possible de cacher ou d’afficher les avantages de chaque thématique pour plus de lisibilité.

(3) — Chaque avantage dispose d’une ligne. Les offres pour chacune des fédérations choisies y sont affichées.

(4) — Lorsque Solidaris propose la pire offre, l’avantage est affiché en rouge.

(5) — Lorsque le remboursement annuel d’un avantage dépasse la moyenne de remboursement annuel de 20 % pour cet avantage, la case de l’avantage est colorée en vert.

Un bouton « télécharger le tableau des avantages » permet de télécharger un PDF reprenant tous les avantages de Solidaris.



*Fig29 : PDF des avantages de Solidaris*

Ce PDF reprend toutes les informations de Solidaris sur les avantages des thématiques sélectionnées.

# Conclusion

Pour conclure ce travail de fin d’études, nous aimerions partager notre ressenti par rapport à cette première expérience dans le monde du travail.

Cette entrée dans le monde du travail s’est déroulée dans un contexte bien particulier de pandémie qui a affecté le travail. Le projet a totalement été développé en télétravail, mais cela ne nous a pas empêché de trouver cette expérience enrichissante.

Être confronté à un réel projet ouvre les yeux sur la profondeur et l’importance de la communication dans le monde du travail. Contrairement à un projet d’école, il faut répondre aux attentes de vraies personnes pouvant changer d’avis, avoir des idées en cours de route ou même ne pas être sûres de ce dont elles ont besoin.

Des compétences orales ont donc pu être développées lors des nombreuses présentations de l’application avec le service marketing de Solidaris. Nous avons eu l’occasion de proposer nos propres idées lors de ces réunions. Ce stage a souligné que la mission d’un développeur ne se limite pas à l’écriture de codes.

Dans le futur, nous devrons en effet conseiller nos clients, les aiguiller et leur faire découvrir les possibilités que renferme la technologie. Il est de notre devoir d’être capable de présenter et d’expliquer clairement nos idées afin de trouver la meilleure solution main dans la main avec nos clients.

La formation d’un développeur n’est jamais finie. En effet les technologies évoluent sans cesse et il est primordial de se tenir au courant des dernières avancées.

Finalement, même si le développement de l’application a pris beaucoup plus de temps que prévu, une application fonctionnelle et répondant aux besoins du service marketing de Solidaris a pu être délivrée.

# Bibliographie

* Solidaris Mutualité (2021), <http://www.solidaris.be/MonsWP/Pages/accueil.aspx?choixregion=true>, consulté le 14/05/21.
* OCM (2021) <https://www.ocm-cdz.be/fr/documents/ocm>, consulté le 14/05/21.
* Stackoverflow (2021), https://stackoverflow.com/questions, consulté le 07/05/21.
* TypeORM documentation (2021), <https://typeorm.io>, consulté le 07/05/21.
* Angular documentation (2021), <https://angular.io/>, consulté le 07/05/21.
* Documentation NestJS (2021), <https://nestjs.com/>, consulté le 07/05/21.
* Documentation PrimeFacesNG (2021), <https://primefaces.org/primeng/showcase/#/setup>, consulté le 07/05/21.
* Udemy (2021), <https://www.udemy.com/>, consulté le 07/05/21.