

Enseignement supérieur économique de promotion sociale de type court

Application de gestion des étudiants avec un handicap.



Chairi Bounekoub Mohammed Anouar

Directeur du suivi : **Mr. D. Van Oudenhove**

*Travail de fin d'études présenté en vue de
L'obtention du titre de bachelier en
Informatique de gestion*

Année scolaire : 2020 – 2021

Remerciements



J'adresse mes plus vifs remerciements et ma profonde reconnaissance à toutes les personnes qui ont contribué et encouragé à la réalisation de ce travail.

Je tiens d'abord à remercier mon directeur de suivi, monsieur Didier Van Oudenhove pour son implication, sa bienveillance et son adaptation particulière à mon égard durant toute la période du suivi de mon TFE.

Je tiens également à remercier toute l'équipe de l'I.S.F.C. E, pour ces trois années passées au sein de leur établissement, de leur patience et de leur aide au bon suivi de ma formation.

Je suis reconnaissant envers toutes les personnes avec qui j'ai eu l'opportunité d'échanger durant cette période. Grâce à ces derniers et à leurs remarques constructives, cela m'a permis de m'améliorer et de trouver des solutions aux problèmes rencontrés.

Je voudrais également remercier ma famille et particulièrement ma mère pour son soutien durant toute cette formation.

Table des matières

Remerciements	- 2 -
Introduction.....	- 5 -
Cahier des charges	- 6 -
1. Contexte et définition du problème	- 6 -
2. Objectif du projet	- 6 -
3. Périmètre du projet	- 6 -
4. Description fonctionnelle des besoins	- 6 -
5. Choix du sujet.....	- 7 -
6. Outils utilisés.....	- 7 -
7. Délais (date de réalisation attendue)	- 7 -
Analyse :	- 8 -
1. Acteur de l'application web.	- 8 -
2. Fonctionnalité par acteur.....	- 8 -
2.1 Visiteur non-authentifié.....	- 8 -
2.2 Administrateur	- 9 -
2.3 Secrétariat (personnel administratif)	- 15 -
2.4 Référent inclusif.	- 20 -
2.4 Professeur	- 24 -
2.5 Étudiant.....	- 27 -
3. Analyse des données et DB	- 31 -
3.1 Données de chaque table (administrateur).....	- 34 -
3.2 Données de chaque table (référent inclusif)	- 36 -
3.3 Données de chaque table (professeur).....	- 36 -
3.4 Données de chaque table (secrétariat).....	- 37 -
3.5 Données de chaque table (étudiant)	- 38 -
4. Conception	- 39 -
1. Programmation.....	- 39 -
2. Gérer les données.....	- 39 -
5. Cheminement de l'application.....	- 40 -
6. Implémentation	- 43 -

6.1	Connexion à l'application.	- 43 -
6.2	Création de comptes	- 44 -
6.3	Création d'un étudiant	- 45 -
6.4	Sélectionner le matériel de l'étudiant.....	- 47 -
6.5	Liste des étudiants	- 50 -
6.6	La fiche signalétique	- 53 -
6.7	Système de Chat (utilisation de Chatify)	- 56 -
6.8	Cookie consentement.....	- 57 -
7.	Sécurité et règlement RGPD	- 59 -
7.1	Qu'est-ce que le RGPD ?	- 59 -
7.2	Les implications du RGPD pour les entreprises :	- 59 -
7.3	Définir les besoins :	- 59 -
7.4	Faire preuve de transparence :	- 60 -
7.5	Consentement des cookies.....	- 60 -
7.6	La sécurité.....	- 60 -
8.	Problèmes rencontrés et solutions	- 61 -
9.	Futur de l'application	- 61 -
10.	Conclusion	- 62 -
11.	Bibliographie	- 63 -

Introduction

Lors de ma dernière année d'informatique de gestion à l'institut supérieur de formation continue d'Etterbeek, j'ai pu participer à un dernier projet pour mon travail de fin d'études. Le jour où l'on m'a présenté la proposition qui était de créer une application pour l'établissement, cela m'a semblé une opportunité à ne pas manquer. J'ai donc saisi cette opportunité pour apporter mon expertise en programmation à un projet concret, qui pourrait être utilisé par la suite pour améliorer le quotidien des personnes ayant un handicap au sein de l'établissement. Cela me permettra aussi de m'améliorer et d'acquérir une expérience professionnelle sur un projet concret, qui pourra être amélioré dans les années à venir. Mon objectif était de réaliser ce projet en mettant à profit les connaissances que j'ai acquises durant ces trois dernières années, en y ajoutant l'expérience que j'ai acquise lors de mes stages avec le Framework Laravel. Ce sera pour moi une belle opportunité pour découvrir de nouvelles technologies qui ont été utilisées durant cette année dans le domaine du développement web. Ainsi, je serai mieux préparé pour affronter le monde professionnel.

Actuellement, beaucoup de gens sont affectés par le décrochage scolaire, à cause de divers événements inattendus (crise sanitaire, problèmes de famille, problèmes de santé, etc.), et qu'ils ne peuvent pas anticiper. Suivre un cursus scolaire n'est pas toujours évident pour tout le monde et chaque individu peut faire face à des problèmes dont l'établissement ne s'aperçoit pas directement ou difficilement. C'est à ce moment-là que l'application web va intervenir. Elle agira comme intermédiaire entre les responsables de l'établissement, le corps enseignant et les étudiants, pour permettre le bon suivi du cursus d'un étudiant se retrouvant avec un handicap, dès le jour de sa première inscription au sein de l'établissement.

Ainsi, les responsables administratifs pourront encoder toutes les informations et besoins de l'étudiant, depuis un seul endroit. Ensuite, l'information pourra être partagée avec tous les enseignants qui ont cours avec cet étudiant.

Les professeurs auront ainsi une meilleure communication en ce qui concerne les changements et les remarques au sujet de l'étudiant. Ils pourront également s'assurer du bon suivi de son parcours, en partageant des documents concernant son handicap et en le rapportant sur l'application pour le partager avec leurs collègues. Ils pourront partager directement les remarques par rapport à la mise en place du matériel si l'étudiant en a besoin. Ainsi, les responsables de la mise en place du matériel pourront mettre le matériel requis à disposition et en place.

Ce sera plus avantageux pour l'étudiant, car il pourra constater par lui-même que l'établissement le soutient afin qu'il puisse suivre ses études dans les meilleures conditions et que son expérience soit la plus agréable possible. L'étudiant pourra aussi mettre à disposition les documents administratifs qui seront directement envoyés aux utilisateurs administratifs pour qu'ils puissent faire le nécessaire.

Cahier des charges

1. Contexte et définition du problème

Notre équipe part du constat que les étudiants concernés par l'inclusif rencontrent des difficultés d'apprentissage très variées en fonction de leur(s) handicap(s). Par ailleurs, il est très difficile de coordonner toute l'équipe administrative et pédagogique autour d'une même problématique liée à un étudiant en demande d'aménagements spécifiques. Or, cette coordination est indispensable pour soutenir l'étudiant dans sa scolarité, d'autant plus dans le contexte de la Covid-19, où ce type d'étudiant est davantage sujet au décrochage scolaire. Dans certains cas, l'équipe enseignante et administrative souhaiterait également obtenir des conseils du médecin responsable de l'étudiant concerné pour pouvoir assurer une meilleure prévention à l'école. Certains handicaps, tels que l'épilepsie ou la schizophrénie, requièrent par exemple une attention toute particulière et demandent une préparation de toute l'équipe en cas de crise.

2. Objectif du projet

Le but de l'application serait ainsi de centraliser toutes les informations à distance afin qu'elles soient facilement accessibles à l'étudiant d'une part, et à l'équipe pédagogique et administrative d'autre part. Elle favoriserait ainsi une cohérence pédagogique personnalisée, une meilleure communication et une intervention adaptée en cas de problème. L'objectif est de créer une application web utilisable sur ordinateur qui rassemblera toutes les informations concernant l'étudiant et la mise en place du matériel adaptée pour ce dernier.

3. Périmètre du projet

Pour ce travail de fin d'études, je me consacre au développement de cette application au sein de l'ISFCE.

4. Description fonctionnelle des besoins

Fonction principale : enregistrer un étudiant dans l'application.

Sous-fonctions :

- Création d'un nouvel étudiant ;
- Créer une fiche signalétique pour l'étudiant ;
- Permettre de mettre à jour les informations qui se trouvent sur la fiche existante.

Fonction principale : enregistrement des aménagements spécifiques.

Sous-fonctions :

- Ajout des aménagements spécifiques (matériel mis à disposition par l'établissement) ;
- Système de check-list pour la mise en place du matériel pour chaque enseignant concerné.

Fonction principale : forum pour le personnel administratif et enseignant.

Sous-fonctions :

- Possibilité d'ajouter des fichiers, liens, etc. ;
- Permettre aux personnels administratifs et enseignants de discuter des problèmes rencontrés avec l'étudiant inscrit.

Fonction principale : espace pour les enseignants afin de poster des documents pédagogiques.

Sous-fonction :

- Importer et exporter des fichiers depuis cet espace.

5. Choix du sujet

Pour la réalisation de ce projet, l'établissement de l'ISFCE m'a proposé un sujet. J'ai trouvé leur proposition intéressante, car elle me permettra tout d'abord de concrétiser un projet tangible, qui pourra être utilisé au sein de l'établissement.

Le second avantage réside dans le fait que ce projet aidera certains étudiants souffrant d'un handicap à pouvoir s'adapter plus facilement et à pouvoir s'intégrer au système de l'établissement. Ils auront un moyen technologique pour combler leurs besoins spécifiques et pour échanger avec les membres de l'ISFCE (professeur, administration). Ainsi, ils auront les mêmes chances que n'importe quel autre étudiant(e) de suivre leur cursus.

Finalement, cela aidera aussi l'établissement, en permettant à cet élève de bénéficier d'un suivi plus facile sur le plan administratif et en suivant l'adaptation de l'étudiant concerné, tout au long de l'année.

6. Outils utilisés

Partie *backend* :

Je vais développer cette application dans le langage PHP et utiliser le Framework Laravel et, pour rendre le site plus dynamique, je vais utiliser *Livewire*. *Livewire* est un Framework complet pour Laravel, qui simplifie la création d'interfaces dynamiques, sans quitter le confort de Laravel.

Pour la partie *frontend*, j'utiliserai le Framework Tailwind css.

7. Délais (date de réalisation attendue)

La réalisation de cette application se déroulera durant le semestre janvier-juin 2021 et pourra être expérimentée en septembre 2021.

Analyse :

1. Acteur de l'application web.

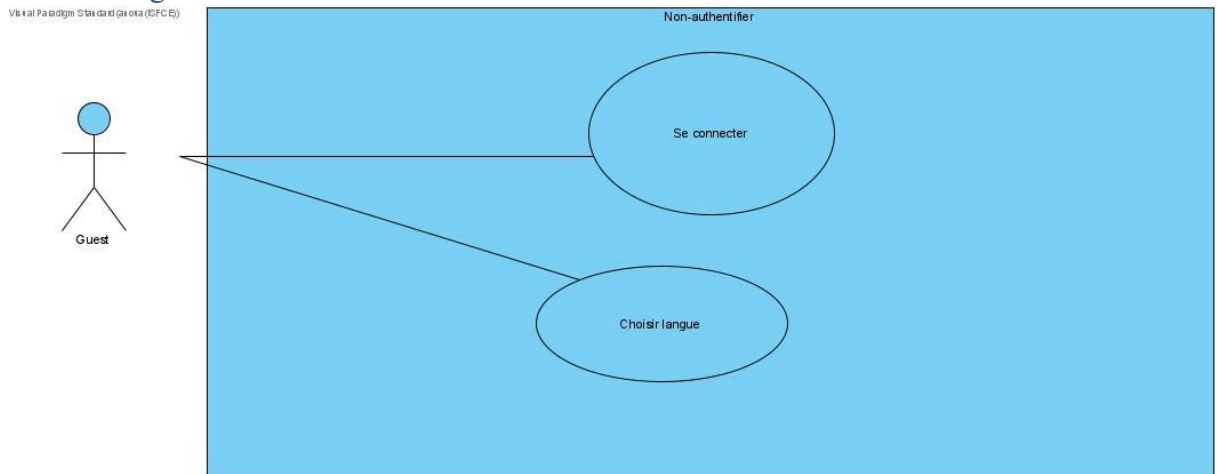
Les acteurs principaux de l'application seront le personnel administratif, les étudiants, un administrateur, un référent inclusif et le personnel pédagogique (professeurs).

2. Fonctionnalité par acteur.

2.1 Visiteur non-authentifié.

- Se connecter à un compte : n'importe quel type de visiteur qui se rend sur le site devra se connecter à l'application web pour pouvoir effectuer des actions. Un étudiant recevra ses accès lors de son inscription à l'école pour pouvoir se connecter sur l'application. Les comptes professeur et les comptes personnel administratif pourront être créés par l'administrateur de l'application web.

2.1.1 Diagramme de cas d'utilisation d'un visiteur



2.1.2 Description textuelle du cas d'utilisation

2.1.1 Cas d'utilisation : se connecter sur l'application

Cas d'utilisation	Se connecter
Niveau	Visiteur (ex. : secrétaire, étudiant, etc.)
Partie prenante	Le visiteur veut se connecter à l'application pour avoir accès au site
État après réussite	Le visiteur pourra avoir accès au site grâce aux fonctionnalités de l'application et grâce à ses identifiants, fournis par l'école
État après échec	Le visiteur n'aura pas accès au site web. Il sera redirigé vers la page de connexion
Trigger	Le visiteur ne va pas pouvoir continuer à visiter le site

2.1.2 Scénario nominal

Acteur	Système
1. Le visiteur (admin, référent inclusif, étudiant) se rend sur le lien du site pour se connecter à l'application grâce aux comptes qu'il a reçu de la part de l'établissement	
	2. Le système recherche dans la base de données si le compte existe. Le système le redirige vers la page d'accueil du site web où il aura accès au menu pour effectuer les actions autorisées

2.1.3 Scénario alternatif

Étapes	Actions de branchement
2.a	Le système ne trouve pas l'utilisateur dans la base de données
2.a.1	Le système redirige l'utilisateur vers le formulaire d'identification et affiche les erreurs au visiteur

2.2 Administrateur

L'administrateur pourra :

- **Créer les différents types de comptes (étudiant, professeur, personnel administratif) :**

L'administrateur aura accès au formulaire de création d'un nouveau compte pour ajouter des professeurs et de nouveaux personnels administratifs sur l'application. Les nouveaux utilisateurs recevront leur accès par e-mail.

- **Inscrire les étudiants :**

L'administrateur pourra aussi encoder un étudiant et ajouter toutes les informations le concernant : handicap, année de naissance, etc.

- **Ajouter les aménagements spécifiques à la base de données :**

L'administrateur aura la possibilité de créer du nouveau matériel. Lors de la sélection, celui-ci servira au personnel administratif pour évaluer les besoins de l'étudiant.

- **Ajouter des fichiers sur la plateforme.**

L'administrateur ajoutera des fichiers qui pourront servir à la compréhension et au fonctionnement du site web ; il tiendra les autres acteurs au courant des dernières mises à jour faites sur le site.

- **Ajouter de nouveaux cours :**

L'administrateur pourra ajouter de nouveaux cours si cela semble nécessaire dans le futur.

- **Recevoir et envoyer des messages depuis l'application de Chat.**

Il pourra se servir de l'application pour discuter avec le personnel administratif et éventuellement, le personnel pédagogique en cas de souci avec le fonctionnement de l'application.

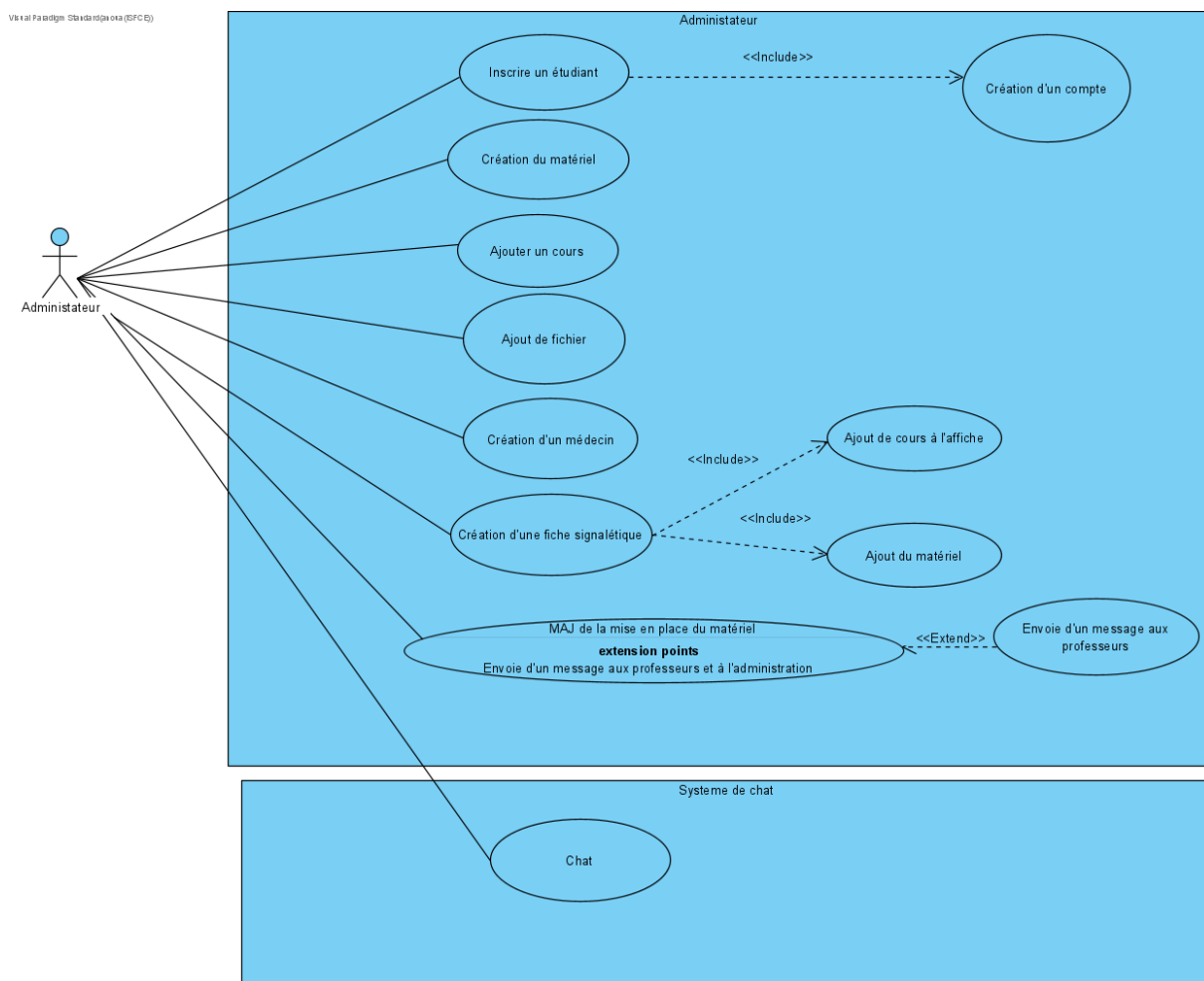
- **Mettre à jour les informations de la fiche signalétique :**

L'administrateur pourra mettre à jour les données de la fiche pour analyser la bonne marche de chaque fonctionnalité.

- **Ajouter des médecins à la base de données :**

L'administrateur pourra créer de nouveaux médecins, dans le but d'analyser le bon fonctionnement du site.

2.2.1 Diagramme de cas d'utilisation d'un administrateur



2.2.2 Description textuelle du cas d'utilisation

2.2.2.1 Cas d'utilisation : création de la fiche signalétique (besoin étudiant)

Cas d'utilisation	Créer la fiche signalétique de l'étudiant qui demande des aménagements spécifiques
Niveau	Administrateur, personnel administratif
Acteur principal	Administrateur, personnel administratif
Parties prenantes	L'administrateur peut créer une fiche signalétique pour que les autres acteurs puissent la consulter ensuite
État après réussite	La fiche signalétique sera créée et toutes les données concernant un étudiant seront enregistrées dans la base de données
État après échec	La fiche signalétique ne sera pas disponible
Trigger	L'administrateur clique sur le bouton de création d'un étudiant.

2.2.2.2 Scénario nominal

Acteurs	Système
1. L'administrateur clique sur le lien pour pouvoir créer un nouvel étudiant	
	2. Le système lui affiche le formulaire
3. L'utilisateur remplit le formulaire	
	4. Le système valide les données
	5. Le système ajoute les données dans la base de données
	6. Le système redirige l'utilisateur vers la liste des étudiants ayant une fiche signalétique
7. L'administrateur clique sur l'icône pour inscrire l'étudiant à des cours	
	8. Le système ajoute les cours à la fiche signalétique
	9. Le système redirige une dernière fois l'administrateur vers la liste des fiches signalétiques pour ajouter l'aménagement spécifique à l'étudiant
10. L'administrateur clique sur le bouton pour ajouter le matériel	
	11. Le système ajoute le matériel dans la base de données pour que celui-ci se rajoute à la fiche signalétique de l'étudiant concerné
12. L'administrateur peut consulter la fiche signalétique de l'étudiant	

2.2.2.3 Scénario alternatif

Étapes	Actions de branchement
3.a	L'utilisateur remplit mal ou pas du tout le formulaire
3.a.1	Le système ne valide pas les données
3.a.2	Le système renvoie vers le formulaire avec les messages d'erreur
7.a	L'administrateur ne remplit pas le formulaire
8.a	Le système n'ajoute pas les cours à la fiche signalétique
8.a.1	Le scénario reprend au point 10
10.a	L'utilisateur ne remplit pas les aménagements spécifiques
10.a.1	Le système n'ajoute pas le matériel dans la fiche signalétique
10.a.2	Le scénario reprend au point 12 sans que les cours et le matériel s'affichent dans la fiche signalétique

2.2.4.1 Cas d'utilisation : création des cours

Cas d'utilisation	Ajout d'un nouveau cours dans la base de données
Niveau	Administrateur
Acteur principal	Administrateur
Parties prenantes	L'administrateur peut ajouter de nouveaux cours
État après réussite	Ajout du cours dans la base de données
État après échec	Le cours ne sera pas ajouté sur le site
Trigger	L'administrateur clique sur le bouton de création d'un cours

2.2.4.2 Scénario nominal

Acteur	Système
1. L'administrateur clique sur le lien pour pouvoir créer un nouveau cours	
	2. Le système affiche le formulaire à remplir
3. L'administrateur remplit le formulaire	

	4. Le système envoie les données à la base de données
	5. Le système redirige le client à la liste des cours

2.2.4.3 Scénario alternatif

Étapes	Actions de branchement
3.a.1	L'utilisateur entre de mauvaises données
3.a.2	Le système ne valide pas les données
3.a.3	Le système renvoie vers le formulaire avec le message d'erreur

2.2.4.1 Cas d'utilisation : mise à jour de la mise en place du matériel

Cas d'utilisation	Mise à jour de la mise en place du matériel
Niveau	Administrateur, référent inclusif
Acteur principal	Administrateur, référent inclusif
Parties prenantes	L'administrateur met à jour et confirme si le matériel a bien été mis en place pour l'étudiant
État après réussite	Mise à jour de l'état du matériel et envoi d'un message sur l'application de messagerie aux professeurs qui ont cours avec l'étudiant
État après échec	La mise en place du matériel ne se fait pas et les professeurs ne reçoivent pas de message
Trigger	L'administrateur ou le référent clique sur le matériel qui doit être mis à jour pour la mise en place

2.2.4.1 Scénario nominal

Acteur	Système
1. L'administrateur clique sur le matériel dans la fiche signalétique de l'étudiant concerné là où il y a eu un changement.	
	2. Le système affiche le formulaire à remplir.
3. L'administrateur remplit le formulaire.	
	4. Le système met à jour les données et ajoute un message préparé à chaque professeur
	5. Le système redirige l'administrateur vers la fiche signalétique de l'étudiant.

2.2.4.2 Scénario alternatif

Étapes	Actions de branchement
3.a.1	L'utilisateur entre de mauvaises données
3.a.2	Le système ne valide pas les données
3.a.3	Le système renvoie vers le formulaire avec le message d'erreur et le message n'est pas envoyé au responsable des cours.

2.3 Secrétariat (personnel administratif)

Le secrétariat pourra :

- **Créer les comptes étudiants :**

Le secrétariat aura accès au formulaire de création d'un nouveau compte pour ajouter des professeurs et de nouveaux personnels administratifs sur l'application. Les nouveaux utilisateurs recevront leur accès par e-mail.

- **Inscrire les étudiants :**

Le secrétariat pourra aussi encoder un étudiant et ajouter toutes les informations le concernant : handicap, année de naissance, cours auxquels il est inscrit.

- **Ajouter les aménagements spécifiques à l'étudiant (fiche) :**

L'administrateur aura la possibilité de créer du nouveau matériel qui servira au personnel administratif, qui sélectionnera le matériel en fonction des besoins de l'étudiant.

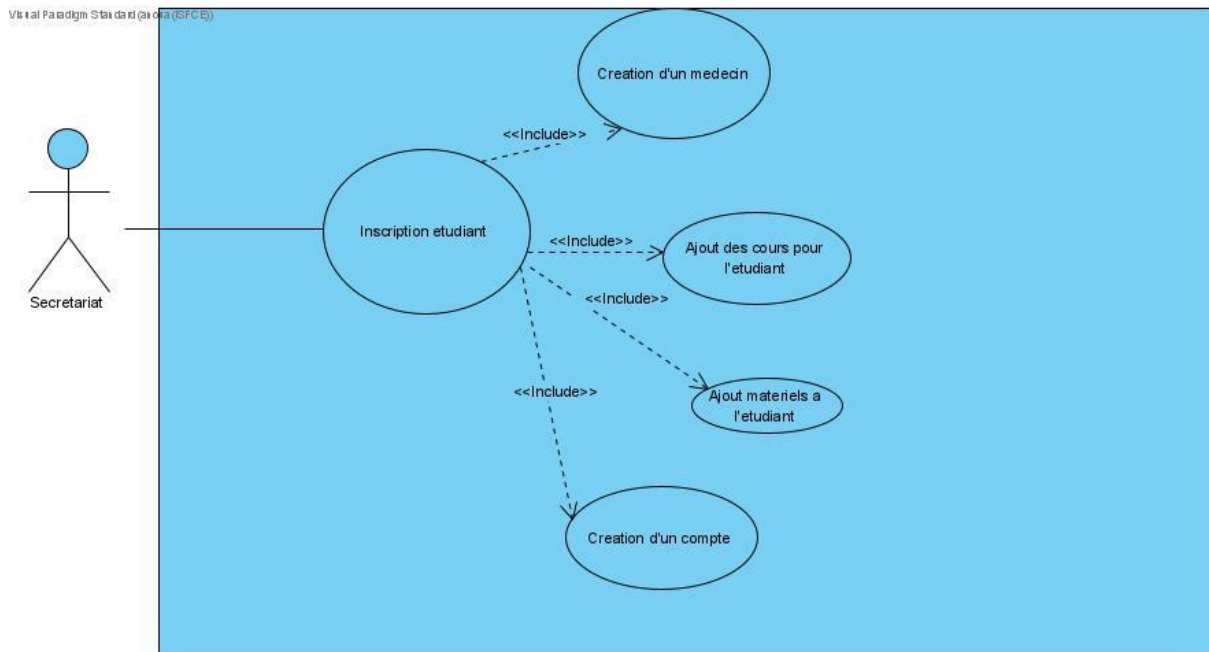
- **Ajouter/consulter des fichiers sur la plateforme.**

Le secrétariat ajoutera des fichiers qui pourront servir à la compréhension et au fonctionnement du site web, il tiendra les autres acteurs au courant des dernières mises à jour faites sur le site.

- **Recevoir et envoyer des messages depuis l'application de Chat.**

Le secrétariat pourra se servir de l'application pour discuter avec le personnel administratif et éventuellement le personnel pédagogique, afin de partager des conseils concernant la situation de l'étudiant.

2.4.1 Diagramme de cas d'utilisation d'un secrétariat



2.4.2 Description textuelle du cas d'utilisation

2.3.2.1 Cas d'utilisation : création d'un étudiant et de la fiche signalétique

Cas d'utilisation	Inscription d'un étudiant
Niveau	Secrétariat
Acteur principal	Secrétariat
Parties prenantes	Le secrétariat crée un nouveau compte étudiant et commence à encoder toutes les informations nécessaires pour compléter la fiche signalétique de l'étudiant
État après réussite	Création d'une fiche signalétique pour l'étudiant concerné et création de son compte
État après échec	La fiche signalétique ne sera pas créée

2.3.2.2 Scénario nominal

Acteurs	Système
1. Le secrétariat clique sur le formulaire de création d'un compte étudiant	
	2. Le système affiche le formulaire à remplir
3. L'administrateur remplit le formulaire	
	4. Le système ajoute le nouveau compte à la base de données
	5. Le système redirige vers un formulaire, le secrétariat remplit les

	informations personnelles de l'étudiant Ex : description de la maladie numéro de contact, naissance, lieu d'habitation
6. Le secrétaire remplit les données	
	7. Le système ajoute les informations dans la base de données
	8. Le système redirige vers la liste des étudiants inscrits pour que le secrétariat continue à encoder les matériel et les cours de l'étudiant
9. Le secrétariat clique sur le bouton pour encoder les cours de l'étudiant	
	10. Le système affiche le formulaire des cours pour inscrire l'étudiant
11. Le secrétariat remplit les données	12. Le système ajoute les données dans la base de données
	13. Le système redirige le secrétaire vers la liste des étudiants
14. Le secrétaire clique sur le bouton d'ajout de matériel pour l'étudiant	
	15. Le système redirige vers le formulaire de sélection du matériel disponible dans l'établissement
16. Le secrétaire remplit les informations	
	17. Le système ajoute le matériel à la fiche signalétique
	18. Le système redirige le secrétaire vers la liste des étudiants inscrits
19. Le secrétariat choisit l'étudiant dont il veut consulter la fiche signalétique	
	20. Le système affiche les données enregistrées pour l'étudiant concerné

2.3.2.3 Scénario alternatif

Étapes	Actions de branchement
9.a.1	L'utilisateur clique directement sur l'affichage de la fiche
9.a.2	Le système affiche la fiche mais signale qu'il faut inscrire les étudiant à des cours et que le besoin du matériel spécifique pour cet étudiant n'a pas encore été sélectionné
14.a.1	Le secrétaire clique sur la fiche de l'étudiant

14.a.2	Le système affiche toutes les données et signale qu'aucun matériel ne lui a été octroyé
--------	---

2.3.3.1 Cas d'utilisation : ajout de document par le secrétariat

Cas d'utilisation	Ajout d'un document
Niveau	Secrétariat
Acteur principal	Secrétariat
Parties prenantes	Le secrétariat ajoute des documents qui peuvent concerner un cas de handicap et le mettent à disposition sur l'application pour le personnel administratif et le personnel éducatif
État après réussite	Le fichier est ajouté à la base de données et les utilisateurs autorisés peuvent télécharger le fichier
État après échec	Le fichier n'est pas ajouté et les utilisateurs ne peuvent le télécharger

2.3.3.2 Scénario nominal

Acteurs	Système
1. Le secrétariat clique sur le bouton d'ajout d'un document	
	2. Le système affiche le formulaire à remplir
3. L'administrateur choisit le fichier ajouté sur l'application	
	4. Le système ajoute le document dans la base de données
	5. Le système redirige l'utilisateur vers Le formulaire et signale que le document a été ajouté
6. Le secrétariat clique sur le bouton pour afficher les documents qui sont disponibles sur la plateforme	
	7. Le système affiche tous les documents sur un tableau où sont indiqués le nom du fichier, la date d'ajout et le responsable de la mise en ligne du document

2.3.3.3 Scénario alternatif

Étapes	Actions de branchement
3.a.1	Le secrétaire ajoute un fichier où l'extension n'est pas autorisée
3.a.2	Le système redirige vers le formulaire avec le message d'erreur selon lequel le fichier n'a pas été ajouté à la base de données

2.3.4.1 Cas d'utilisation : système de Chat envoi et réception de messages

Cas d'utilisation	Envoi de message
Niveau	Secrétariat
Acteur principal	Secrétariat
Parties prenantes	Le secrétariat a la possibilité d'utiliser le Chat pour communiquer avec les personnels administratif et les enseignants
État après réussite	Envoyer le message à la personne concernée
État après échec	L'envoi du message ne se fait pas

2.3.4.4 Scénario nominal

Acteur	Système
1. Le secrétariat clique sur le bouton du Chat	
	2. Le système affiche la partie Chat
3. Le secrétaire recherche l'utilisateur avec qui il veut communiquer	
	4. Le système ouvre le Chat et affiche la conversation
5. Le secrétaire écrit son message ou envoie les fichiers dans une discussion privée	
	6. Le système ajoute le message à la base de données et l'envoie au destinataire
7. Fin du processus	

2.3.4.5 Scénario alternatif

Étapes	Action de branchement
3.a.1	Le secrétaire ne trouve pas l'utilisateur
3.a.2	Impossibilité d'envoi du message (secrétaire)
6.a.1	Impossibilité d'envoi du message (système ; perte de connexion)
6.a.2	Le processus reprend au point 1

2.4 Référent inclusif.

- **Check si toutes les données ont été bien encodées :**

Le référent doit vérifier que rien n'a été oublié concernant l'étudiant, que ce soit au niveau du matériel mis en place ou des éléments à savoir sur son handicap. Il s'assure de vérifier toutes les informations et de les communiquer aux chargés de cours.

- **Met à jour l'état du matériel par rapport à l'étudiant :**

Le référent met à jour la disponibilité du matériel et signale si l'élément est bien mis en place, puis ajoute un commentaire par rapport à cette mise à jour. Le système se charge d'envoyer un message aux chargés de cours auxquels l'étudiant est inscrit.

- **Ajouter/consulter des fichiers sur la plateforme.**

Le secrétariat ajoutera des fichiers qui pourront servir à la compréhension et au fonctionnement du site web ; il tiendra les acteurs au courant des dernières mises à jour faites sur le site.

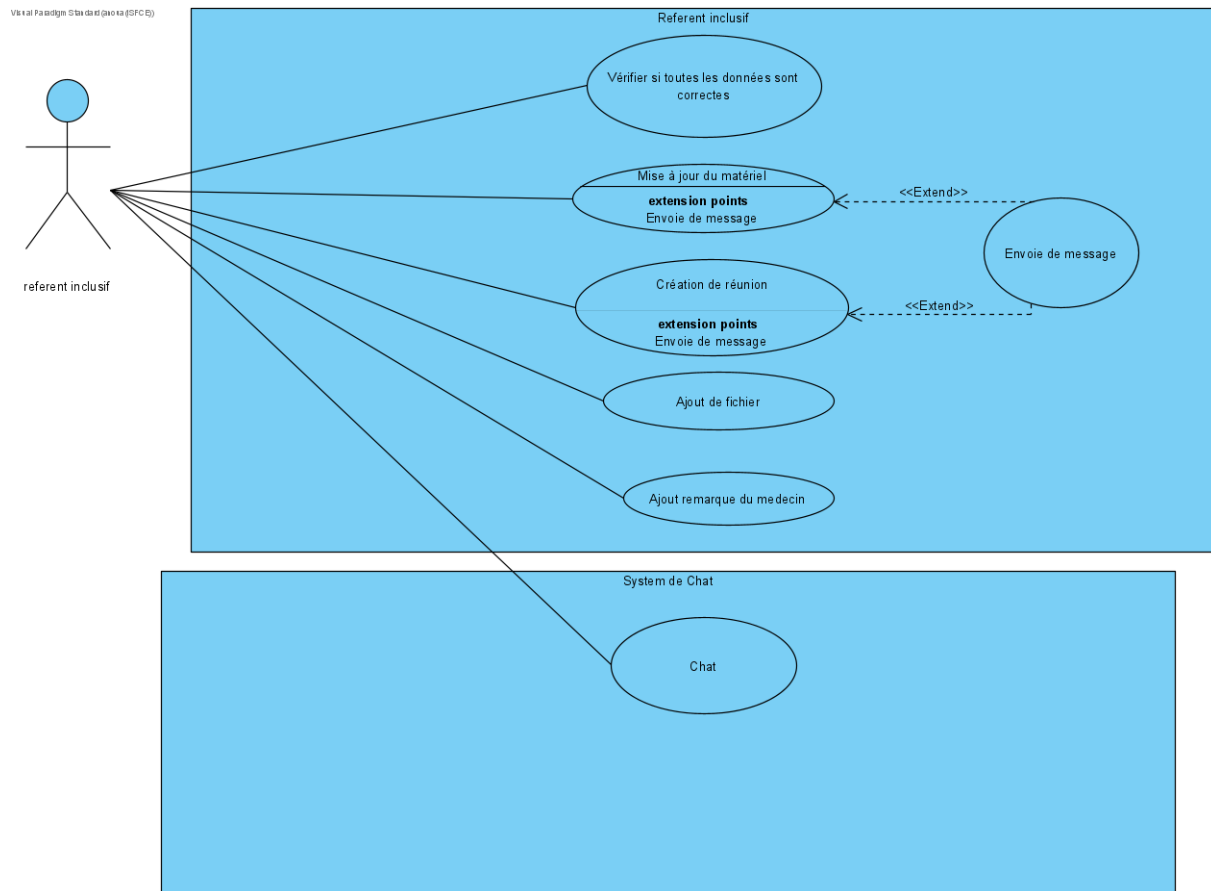
- **Recevoir et envoyer des messages depuis l'application de Chat.**

Le secrétariat pourra se servir de l'application pour discuter avec le personnel administratif et éventuellement le personnel pédagogique, afin de partager des conseils et de faire un point sur la situation de l'étudiant.

- **Créer des réunions**

Le référent aura la possibilité de créer des réunions si besoin avec le personnel administratif et le personnel enseignant. Le système enverra un message dans l'application Chat pour tenir les étudiants au courant de la date et de l'heure de la réunion, afin que dans le Chat, chaque personne puisse confirmer sa présence.

2.4.1 Diagramme de cas d'utilisation d'un référent inclusif



2.4.2 Description textuelle du cas d'utilisation

2.4.3.1 Cas d'utilisation : check information fiche signalétique

Cas d'utilisation	Vérification des données de la fiche signalétique
Niveau	Référent inclusif
Acteur principal	Référent inclusif
Parties prenantes	Le référent vérifiera et modifiera les éléments qui doivent être mis à jour
État après réussite	La fiche signalétique est mise à jour
État après échec	La modification ne se fait pas et l'affiche signalétique n'est pas mise à jour
Trigger	Le référent clique sur l'affiche signalétique de l'étudiant concerné

2.4.3.2 Scénario nominal

Acteur	Système
1. Le référent inclusif clique depuis le lien : liste des étudiants	
	2. Le système affiche la liste des étudiants inscrits sur l'application
3. Le référent recherche dans la barre de recherche l'étudiant concerné et clique sur la fiche signalétique	
	4. Le système affiche la fiche signalétique
5. Le référent vérifie les données et modifie ce qui doit être mis à jour	
	6. Le système met à jour les informations sélectionnées par le référent inclusif
7. Fin du processus	

2.4.2.3 Scénario alternatif

Étapes	Actions de branchement
3.a.1	Le système ne trouve pas l'étudiant qui doit être vérifié par le référent
3.a.2	L'étudiant n'a pas encore été enregistré sur le site. Fin du processus
6.a.1	Le système n'arrive pas à mettre à jour les informations
6.a.2	Le système envoie les messages d'erreur pour que le référent puisse ré-encoder

2.4.3.1 Cas d'utilisation : création d'une réunion pour les professeurs et le personnel administratif

Cas d'utilisation	Création d'une réunion
Niveau	Référent inclusif
Acteur principal	Référent inclusif
Parties prenantes	Une fois qu'il a contacté le médecin, le référent peut créer une réunion avec l'établissement pour communiquer les informations utiles qui peuvent se rajouter pour le bon suivi de l'étudiant
État après réussite	La réunion est créée et se rajoute à la liste des réunions prévues
État après échec	La réunion ne se crée pas, il sera redirigé vers la liste signalétique
Trigger	Le référent clique depuis la fiche signalétique pour demander à créer une future réunion

2.4.3.2 Scénario nominal

Acteur	Système
1. Le référent inclusif clique depuis la fiche signalétique de l'étudiant pour créer une réunion	
	2. Le système affiche le formulaire de création de réunions
3. Le référent remplit les données	
	4. Le système ajoute la réunion à la base de données
	5. Le système envoie un message au professeur et au personnel administratif de la date et de l'heure de la réunion
	6. Le système redirige le référent dans la liste des réunions créées
7. Le référent vérifie si toutes les réunions ont été créées et mise sur le site	
8. Fin du processus	

2.4.3.3 Scénario alternatif

Étapes	Action de branchement
3.a.1	Le système ne rajoute pas la réunion
3.a.2	Le système renvoie le référent vers le formulaire et affiche les messages d'erreur

2.4.4.1 Cas d'utilisation : ajout des informations reçues par le médecin

Cas d'utilisation	Ajout des informations reçues par le médecin chargé de l'étudiant
Niveau	Référent inclusif
Acteur principal	Référent inclusif
Parties prenantes	Une fois qu'il a contacté le médecin, le référent ajoute les conseils en matière de prévention pour le handicap de l'étudiant
État après réussite	La fiche signalétique est mise à jour et les directives s'ajoutent
État après échec	Les directives du médecin ne sont pas mises à jour
Trigger	Le référent clique depuis la fiche signalétique pour ajouter les directives du médecin

2.4.4.2 Scénario nominal

Acteur	Système
1. Le référent inclusif clique depuis la fiche signalétique pour ajouter les directives du médecin	
	2. Le système l'envoie vers le formulaire (qui est un éditeur de texte)
3. Le référent remplit les informations et conseils donnés par le médecin	
	4. Le système met à jour les conseils reçus par le médecin
	5. Le système envoie un message au professeur concerné pour être averti que les conseils du médecin sont disponibles
	6. Le système redirige le référent vers l'affichage de la fiche signalétique
7. Le référent peut voir la mise à jour apportée	
8. Fin du processus	

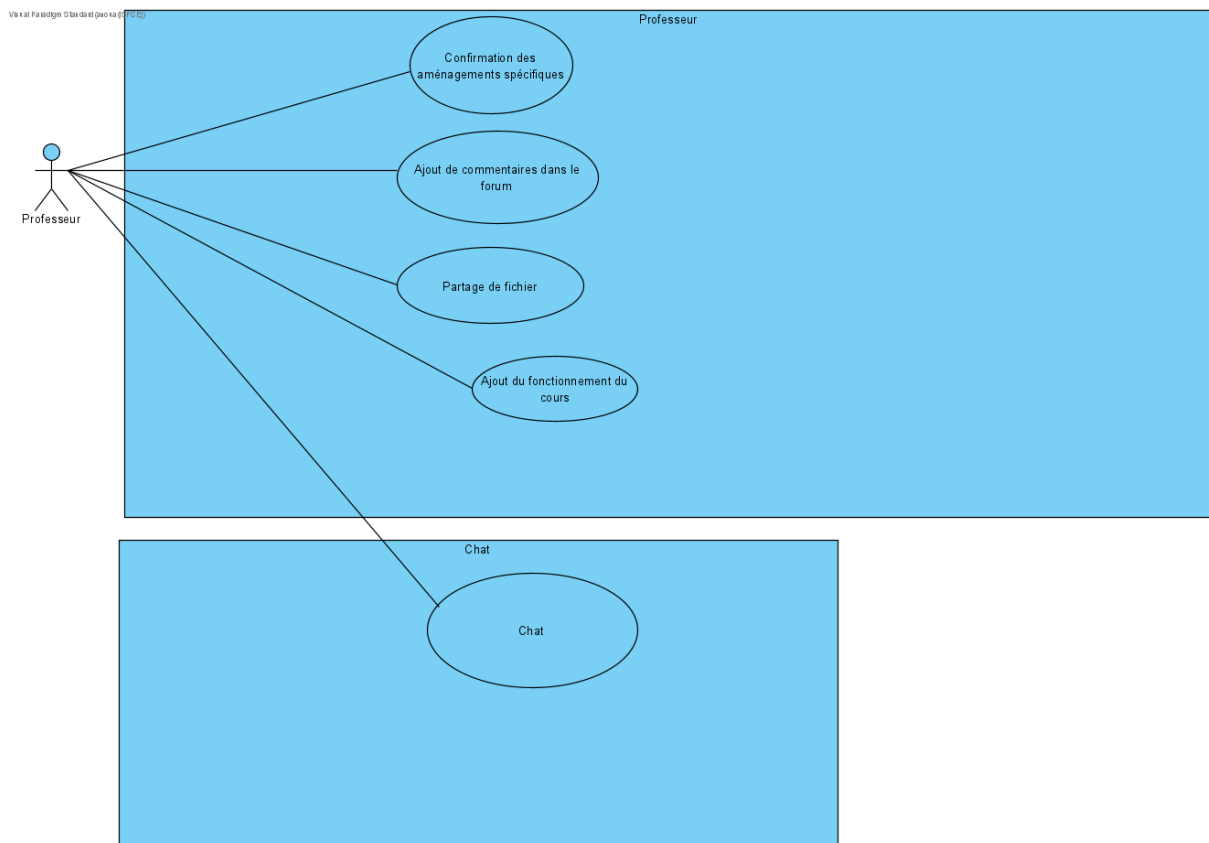
2.4.4.3 Scénario alternatif

Étapes	Actions de branchement
3.a.1	Le référent ne remplit pas le champ
3.a.2	Le système renvoie le référent vers le formulaire et affiche les messages d'erreur
5.a.1	Le système n'envoie pas de message au professeur concerné
5.a.2	Les professeurs ne seront pas mis au courant mais la mise à jour a bien été faite

2.4 Professeur

- **Signaler dans l'application si les aménagements ont bien été mis en place**
- **Commentaire dans le forum (s'il rencontre des difficultés avec l'étudiant)**
- **Partage de fichier avec les étudiants dans la rubrique dédiée**
- **Recevoir et envoyer des messages depuis l'application de Chat.**

2.5.1 Diagramme de cas d'utilisation d'un professeur



2.5.2 Description textuelle du cas d'utilisation

2.5.2.1 Cas d'utilisation : check matériel mis en place

Cas d'utilisation	Prévenir depuis la fiche signalétique que le matériel est bien mis à disposition de l'étudiant pour le cours concerné
Niveau	Professeur
Acteur principal	Professeur
Parties prenantes	Le professeur ayant donné son premier cours vérifie que l'étudiant a bien le matériel nécessaire pour suivre le cours durant toute la période. Ensuite, il prévient sur l'application si le matériel est disponible pour l'étudiant et met une remarque pour plus de détails
État après réussite	La mise à jour se fait et le référent peut vérifier que tout a été mis en place
État après échec	L'état du matériel pour le cours reste non-mis en place et le référent devra prendre contact avec le professeur pour voir où en est la progression dans la mise à disposition
Trigger	Le professeur clique sur le bouton pour mettre l'état de la mise en place du matériel à jour

2.5.2.2 Scénario nominal

Acteur	Système
1. Le professeur se rend sur la fiche de l'étudiant	
2. Dans la partie cours, il trouvera un bouton « mettre à jour la mise en place du matériel »	
3. Le professeur remplit les données	
	4. Le système l'envoie vers le formulaire. Il pourra activer ou pas la mise en place et ajouter un commentaire
	5. Le système met les données à jour
	6. Le système envoie un message au référent pour être averti qu'il y a eu un changement
	7. Le système redirige le professeur vers l'affichage de la fiche signalétique
8. Le professeur peut voir le commentaire et la mise à jour	
9. Fin du processus.	

2.5.2.3 Scénario alternatif

Étapes	Actions de branchement
2.a.1	Le professeur n'a pas accès au bouton.
2.a.2	Le système ne peut renvoyer le professeur vers le formulaire pour mettre à jour l'état du matériel
3.a.1	Le professeur laisse les champs vides et n'apporte aucune modification
3.a.2	Aucune mise jour n'est faite
6.a.1	Si aucune mise à jour n'a été faite, le référent ne reçoit aucun message

2.5.3.1 Cas d'utilisation : le professeur ajoute des documents au cours

Cas d'utilisation	Ajout d'un document explicatif sur la manière de bien suivre le cours
Niveau	Professeur
Acteur principal	Professeur
Parties prenantes	Le professeur pourra ajouter un document au cours qui permettra à l'étudiant de suivre le cours correctement
État après réussite	Le document s'ajoute à la partie fichier des informations des cours
État après échec	Le document ne s'ajoute pas au cours
Trigger	Le professeur clique sur le cours depuis la liste et ajoute le document nécessaire

2.5.3.2 Scénario nominal

Acteur	Système
1. Le professeur se rend sur la liste des cours	
2. Il choisit un de ses cours et appuie sur le bouton d'ajout de fichier	
	3. Le système enregistre le fichier dans la base de données
	4. Le système ajoute le fichier à côté du cours concerné
5. Le professeur peut télécharger son fichier ou le mettre à jour	
6. Fin du processus	

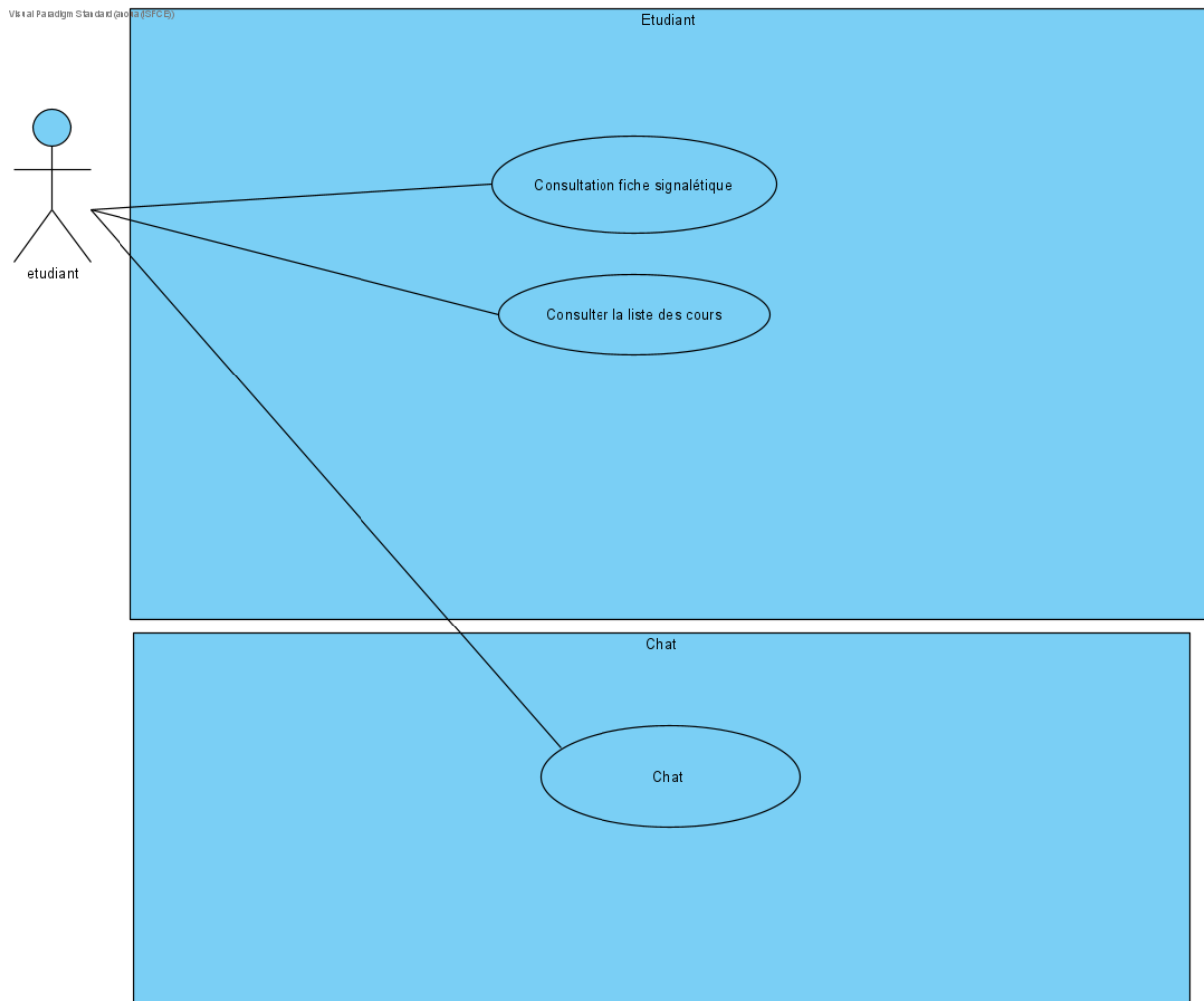
2.5.3.3 Scénario alternatif

Étapes	Actions de branchement
2.a.1	Le professeur n'a pas accès au bouton
2.a.2	Le système n'ouvre pas la sélection de fichier
2.a.3	Le professeur ne peut ajouter de fichier pour le cours
2.a.3	Fin du processus

2.5 Étudiant

- Consulter ses informations dans la fiche signalétique
- Consulter les informations du professeur
- Recevoir et envoyer des messages depuis l'application de Chat
- Consulter les cours

2.6.1 Diagramme de cas d'utilisation d'un étudiant



2.6.2 Description textuelle du cas d'utilisation

2.6.2.1 Cas d'utilisation : consulter sa propre fiche signalétique

Cas d'utilisation	Consultation de sa fiche signalétique
Niveau	Étudiant
Acteur principal	Étudiant
Parties prenantes	L'étudiant consultera sa propre fiche signalétique et vérifiera les données pour voir l'avancée par rapport au matériel dont il a besoin
État après réussite	Visualisation de la fiche signalétique
État après échec	L'étudiant ne peut pas voir ses informations et est redirigé vers la page home
Trigger	L'étudiant clique sur sa fiche signalétique depuis le menu

2.6.2.2 Scénario nominal

Acteur	Système
1. L'étudiant clique depuis le menu sur sa fiche signalétique	
	2. Le système affiche les données de l'étudiant connecté
3. Fin du processus	

2.6.2.3 Scénario alternatif

Étapes	Actions de branchement
2.a.1	Le système n'affiche pas la fiche de l'étudiant
2.a.2	Le système redirige l'étudiant vers la page home avec un message d'erreur
2.a.3	Reprise au point 1

2.6.3.1 Cas d'utilisation : consulter la liste des cours

Cas d'utilisation	Consultation de la liste des cours
Niveau	Étudiant
Acteur principal	Étudiant
Parties prenantes	L'étudiant pourra consulter les informations concernant un cours et consulter le document laissé par le responsable du cours
État après réussite	Visualisation des caractéristiques de chaque cours
État après échec	L'étudiant ne pourra pas voir les cours
Trigger	L'étudiant clique sur la liste des cours depuis le menu

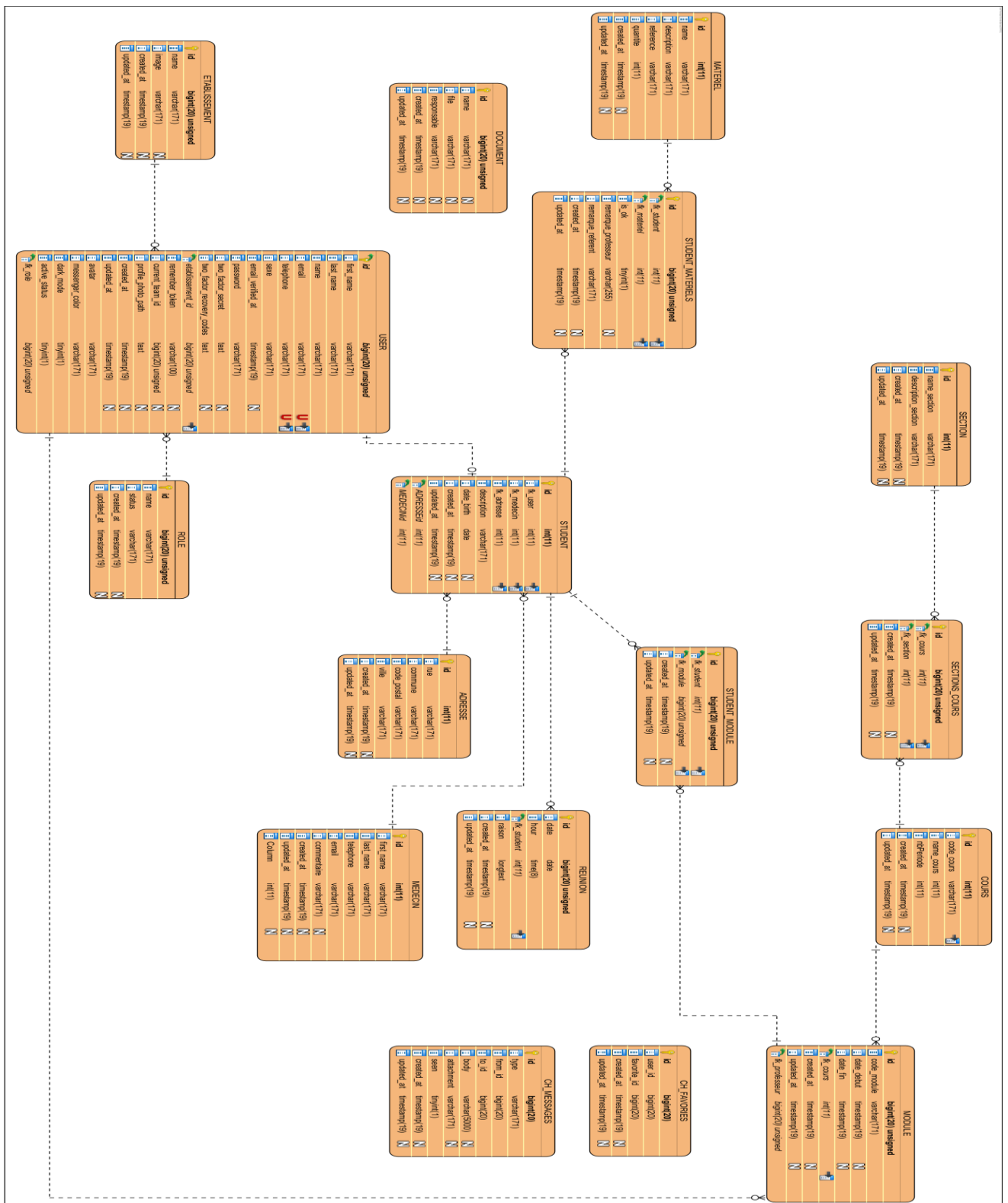
2.6.3.2 Scénario nominal

Acteur	Système
1. L'étudiant clique depuis le menu sur sa liste de cours	
	2. Le système affiche un bouton option pour choisir la section
3. L'étudiant choisit la section	
	4. Le système affiche les cours de la section
5. L'étudiant choisit un cours pour voir les informations	
	6. Le système affiche les informations du cours

2.6.3.3 Scénario alternatif

Étapes	Actions de branchement
2.a.1	Le système n'affiche pas la fiche de l'étudiant
2.a.2	Le système redirige l'étudiant vers la page home avec un message d'erreur
2.a.3	Reprise au point 1

3. Analyse des données et DB



Pour la réalisation de cette application web, j'ai utilisé 18 tables pour stocker les données. Les données contenues dans les tables seront affichées dans différents endroits du site. La plupart des fonctionnalités vont nécessiter l'ouverture d'un compte personnel et privé pour chaque utilisateur afin de pouvoir effectuer une action sur l'application. Les actions de chaque utilisateur dépendront du rôle qui leur a été assigné lors de la création de leur compte.

User :

Cette table contiendra toutes les données principales d'un utilisateur. Lorsque l'administrateur créera les différents comptes, il devra remplir tous les champs du formulaire. Le propriétaire du compte pourra choisir la photo de profil qui sera affichée sur son compte. La variable *profil_photo_path* sera utilisée pour vérifier si l'utilisateur l'a téléchargée. Le système utilisera une image par défaut si le champ est à *NULL*, sinon il utilisera la photo choisie pour son profil.

Adresse :

La table adresse va servir à créer une adresse pour l'étudiant. Cette table va contenir les champs ville ; commune ; rue ; code postal.

Rôle :

La table des rôles est directement liée à la table des utilisateurs car elle sera utilisée afin que le système reconnaisse l'utilisateur en tant qu'étudiant, administrateur, professeur, référent inclusif ou personnel administratif. Dans l'application, chaque rôle aura accès à des fonctionnalités différentes. Par exemple, le rôle « administrateur » aura accès à la création, la modification et la suppression de nouveaux cours, fonctionnalités auxquelles les autres utilisateurs n'auront pas accès.

L'établissement :

La table « établissement » est liée à la table « user » car elle sera utilisée dans le futur pour plusieurs établissements. Cela permettra de différencier les utilisateurs d'une école et/ou d'une promotion d'une autre. Cela va permettre à un utilisateur de ne pas pouvoir voir les professeurs ou les étudiants d'une autre école.

Médecin :

Cette table va contenir toutes les informations du médecin qui s'occupe de l'étudiant, tel qu'un champ nom ; prénom ; téléphone ; adresse e-mail et commentaires du médecin. Cette table est complétée par le secrétariat lors de l'inscription d'un étudiant.

Student :

La table « student » est liée aux tables « user », « médecin » et « adresse », car elle sera utilisée pour créer la fiche signalétique de l'étudiant. Elle sera utilisée pour récupérer toutes les informations concernant l'étudiant, telles que les informations du médecin (nom, prénom, tel, e-mail, etc.) et les informations de l'étudiant. La *Fk_user* sera utilisée pour récupérer les informations de base d'un étudiant depuis son compte.

Module :

La table « module » est liée à la table « cours », car un module a un cours et cela permettra d'avoir les différents modules pour chaque cours. La table module est liée aussi à la table user car un module va posséder un utilisateur qui a comme rôle le rôle professeur.

Cours :

La table « cours » va contenir les informations principales du cours tel que le code du cours le nom du cours et nombre période.

Section :

La table « section » a été créée dans le but de diviser les cours selon l'option choisie. Par exemple, la section informatique de gestion, marketing, comptabilité, etc.

Section cours :

La section « cours » va être liée à la table « section » et « cours ». Cette table va nous permettre de récupérer une section qui va être liée à un cours ; ainsi, on pourra trier par exemple les cours de marketing et voir uniquement les cours concernant cette option.

Student module :

« *Student_module* » va être liée à la table « student » et à la table « module », car pour les besoins de la fiche signalétique, nous aurons besoin de récupérer les cours(module) auquel l'étudiant s'est inscrit pour l'année en cours.

Matériel :

La table « matériel » va permettre de créer du nouveau matériel sur le site. Chaque matériel aura un nom de marque, une description, une quantité, et une référence unique en plus de l'id.

Student materiel :

La table « *student_materiel* » est liée aux tables « student » et « matériel ». Cette table va permettre de récupérer les matériels spécifiques dont un étudiant aura besoin pour suivre son cursus correctement. La variable « *is_ok* » permettra au référent inclusif de confirmer que le matériel a bien été correctement installé pour l'étudiant. La variable « *remarque_referent* » va permettre au référent d'ajouter une remarque lors de la mise à jour pour tenir l'ensemble des responsables de l'étudiant au courant de ce qu'il reste à faire pour mettre correctement le matériel en place.

Réunion :

La table « réunion » est liée à la table « student », car une réunion sera créée par le référent inclusif pour un étudiant. Dans cette table, nous allons particulièrement retrouver un champ « *fk_student* » pour récupérer l'étudiant concerné. La table contiendra un champ « *date* » pour organiser la date, un champ « *hour* » pour indiquer l'heure de la réunion pour cet étudiant et un champ « *raison* » qui permettra au référent de mentionner l'objet de cette réunion.

Document :

La table « document » va permettre d'ajouter des documents à la base de données qui va pouvoir être chargée/téléchargée depuis le site web. La table contiendra un champ « *name* » pour le nom du fichier, « *file* » pour récupérer le nom complet du fichier avec l'extension et un champ « responsable » qui contiendra le nom du responsable qui a publié le fichier et un champ « date » pour savoir quand le fichier a été publié.

Ch favorite :

Cette table a été ajoutée lors de l'import du système de Chat Chatify.

Ch message :

Cette table a été importée du package Chatify. Cette table permet de contenir des messages privés envoyés depuis la partie Chat du site. Elle contient les champs :

Type : qui sera toujours de type user.

From_id : contiendra l'id de l'utilisateur qui envoie un message.

To_id : contiendra l'id du destinataire.

Body : contiendra le contenu du message.

Attachment : peut être null ou contenir un fichier ou autre.

3.1 Données de chaque table (administrateur)

Table	Consulter	Ajouter	Modifier	Supprimer
User	X			
Rôle	X	X	X	X
Établissement	X	X	X	X
Document	X	X		X
Adresse	X	X	X	X
Student	X	X	X	X
Médecin	X	X	X	X
Cours	X	X	X	X
Section	X	X	X	X
Module	X	X	X	X
Cours_section	X	X	X	X
Student_module	X	X	X	X
Réunion	X	X	X	

Student_materiel	X	X		
Check_professeur				
Matériel	X	X	X	X
Ch_message	X	X		
Ch_favorite	X	X		

3.2 Données de chaque table (réfèrent inclusif)

Table	Consulter	Ajouter	Modifier	Supprimer
User	X			
Rôle				
Établissement				
Document	X	X		X
Adresse		X	X	
Student	X	X	X	
Médecin	X	X	X	
Cours	X			
Section	X			
Module	X			
Cours_section	X			
Student_module	X	X		X
Réunion	X	X	X	X
Student_materiel	X	X	X	X
Check_professeur				
Matériel	X			
Ch_message	X	X		
Ch_favorite	X	X		

3.3 Données de chaque table (professeur)

Table	Consulter	Ajouter	Modifier	Supprimer
User	X			
Rôle				
Établissement				
Document	X	X		X
Adresse		X	X	

Student	X			
Médecin				
Cours	X		X	
Section	X			
Module	X			
Cours_section	X			
Student_module	X	X		
Réunion	X	X	X	X
Student_materiel	X			
Check_professeur	X	X	X	
Matériel	X			
Ch_message	X	X		
Ch_favorie	X	X		

3.4 Données de chaque table (secrétariat)

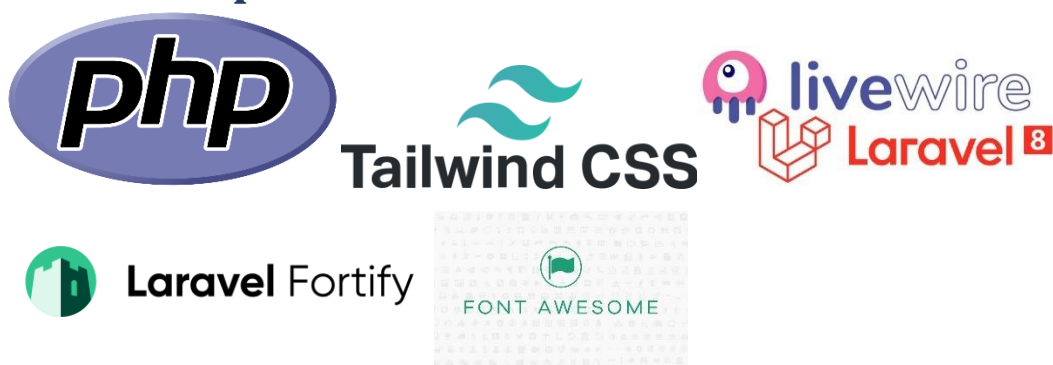
Table	Consulter	Ajouter	Modifier	Supprimer
User	X	X	X	X
Rôle	X			
Établissement	X			
Document	X			X
Adresse	X	X	X	
Student	X	X	X	X
Médecin	X	X	X	X
Cours	X			
Section	X			
Module	X			
Cours_section	X			
Student_module	X	X	X	
Réunion	X			

Student_materiel	X	X	X	X
Check_professeur	X			
Matériel	X			
Ch_message	X	X		
Ch_favorite	X	X		

3.5 Données de chaque table (étudiant)

Table	Consulter	Ajouter	Modifier	Supprimer
User				
Rôle				
Établissement				
Document				
Adresse				
Student				
Médecin				
Cours	X			
Section	X			
Module	X			
Cours_section	X			
Student_module	X			
Réunion				
Student_materiel	X			
Check_professeur				
Matériel	X			
Ch_message	X	X		
Ch_favorie	X	X		

4. Conception



L'application web sera réalisée en utilisant le Framework Laravel dans sa version 8, ainsi que la version 7.4.9 de PHP. J'utiliserai aussi Livewire à partir de la version 2.4 pour rendre le site web dynamique.

Pour la partie *backend* de l'application, j'utiliserai MySQL 5.7.31 en tant que système de base de données. Le langage que j'utiliserai pour communiquer avec la base de données est le SQL. Pour la partie « authentification », j'utiliserai Fortify, une implémentation backend d'authentification frontale pour Laravel.

Pour la partie *frontend*, j'utiliserai le Framework Tailwind css. Tailwind css est un Framework css utilitaire, qui est utilisé pour créer n'importe quel design à partir des balises html. Pour tout ce qui est icône sur l'application, j'utiliserai *font-awesome*, un kit de démarrage qui propose un large contenu de logos pour faire son site web.

1. Programmation

J'utiliserai IDE Visual studio code pour écrire le code de l'application web, qui est parfaitement adapté pour l'encodage avec du Laravel. J'ai choisi cet IDE car je l'utilise depuis le début avec Laravel et j'ai une assez bonne expérience avec ce logiciel depuis mes stages.

2. Gérer les données.

J'utiliserai PHPMyAdmin version 5.0.2 afin de stocker toutes les données de l'application dans une base de données. L'interface est simple d'utilisation et je pourrai tester mes requêtes SQL facilement. J'utilise phpMyAdmin depuis le début de mes études et je l'ai aussi utilisé lors de mon stage. Les données sont très clairement affichées et je pourrai créer des tables et insérer de nouveaux enregistrements très facilement pour tester mon application.

5. Cheminement de l'application

Dans cette partie, je vais vous expliquer le cheminement selon lequel j'ai pu mener ce projet à terme. Cela permettra au lecteur de pouvoir réaliser l'évolution de mon projet tout au long de sa réalisation.

En premier lieu, j'ai créé la structure de mon application. Pour cela, j'ai utilisé *Composer*, qui crée la principale structure de mon projet Laravel. *Composer* génère tous les dossiers et fichiers nécessaires au bon fonctionnement de l'application. Il contient plusieurs parties :

- Un dossier nommé *config* pour gérer la configuration du Framework. ;
- Un dossier *Database*, qui permet la gestion de la base de données. Ce dossier est composé de sous-dossiers, tels que les dossiers de migration, *seeds* et *factories*. Le sous-dossier migration contiendra des fichiers permettant de représenter la base de données. Le sous-dossier *seeds* permettra de préparer des enregistrements qui s'inséreront dans les tables de la base de données. Le dernier sous-dossier *factories* permettra de créer des enregistrements en masse, ce qui est pratique pour faire des tests ;
- Un dossier *public* : ce dossier contiendra tous les fichiers accessibles pour nos utilisateurs, tels que les images, les fichiers charger/télécharger ; nous retrouverons aussi les fichiers css, JavaScript ;
- Un dossier *resources*, qui contiendra :
 - o 2 sous-dossiers principaux :
 - Le sous dossier *Lang*, qui va nous permettre d'internaliser notre site avec diverses langues ; dans le cas du projet, nous aurons le français et l'anglais ;
 - Le sous-dossier *views*, qui contient les vues de l'application. Ces fichiers seront principalement composés de code HTML qui permet d'afficher notre application web.
- Un dossier *route* :
 - o Ce dossier va contenir un fichier "web.php". C'est ce fichier qui va permettre de configurer nos routes et de faire les liens entre notre contrôleur et la vue générale, car Laravel est un Framework PHP basé sur le modèle MVC.

Au niveau des fichiers principaux, nous avons plusieurs fichiers qui se sont ajoutés lors de la mise en place de l'application, tels que : le fichier « .env », git ...

Le fichier « .env » va contenir tous les mots de passe pour les services que j'ai utilisés. Par exemple : mot de passe de la base de données, pour la configuration du port de la base de données, pour le nom de l'application, etc.

J'ai utilisé Git comme contrôle de version. C'est ce qui m'a permis de travailler sur différentes tâches sans pour autant affecter les fonctionnalités principales ou déjà réalisées. Cela m'a permis aussi de travailler sur différentes branches et de fusionner les différentes tâches au fur et à mesure de l'avancement du projet.

En deuxième lieu, j'ai réfléchi à l'analyse de la base de données et aux tables qu'elle allait contenir, de sorte qu'elle réponde au mieux aux besoins de l'établissement. C'est au cours de cette étape que j'ai dû réfléchir aux tables nécessaires à la bonne cohérence de ce que l'on attendait de cette application. Les besoins de l'application étant déjà définis par l'établissement, je devais m'assurer que chaque table créée devait répondre à ces besoins.

En troisième lieu, j'ai dû choisir un moyen d'authentification pour les utilisateurs. Après recherches sur le web, et après avoir vu ce que l'on proposait comme technologies pour l'authentification, j'ai déterminé trois moyens pour gérer cette authentification.

La première est Laravel UI, celle que je connais depuis que j'ai commencé à utiliser Laravel. Elle a souvent été utilisée dans les versions antérieures de Laravel 8. À savoir que Laravel/ui risque de ne plus avoir de mise à jour et donc, de devenir obsolète à l'avenir.

La seconde est Laravel/breeze ; un nouveau package officiel de Laravel. Il est simple à utiliser et utilise tailwind css. C'est l'option la plus simple à utiliser pour un débutant, car le code est simple, lisible, accessible et modifiable facilement.

La troisième consiste à utiliser Fortify en créant vos propres vues et en gérant des fonctions supplémentaires. C'est-à-dire que Fortify enregistrera des routes et contrôleurs nécessaires pour mettre en œuvre les fonctionnalités comme la connexion, l'enregistrement, la réinitialisation de mot de passe, etc. En revanche, Fortify ne fournit pas sa propre interface utilisateur par rapport aux moyens énumérés précédemment.

Finalement, pour mon projet, j'ai utilisé Fortify, car elle correspond mieux à l'attente de l'application et le fait que je puisse gérer moi-même l'interface me correspondait mieux. De plus, je souhaitais aussi me lancer dans une nouvelle technologie et découvrir d'autres manières de gérer l'authentification.

En quatrième lieu, je me suis occupé de la conception du site web pour les pages principales, en faisant des croquis sur papier. Ces pages représenteront le design que j'essaierai d'atteindre dans l'application. J'avais déjà en tête quelques idées de la disposition et du visuel (partie *frontend*). Ces croquis vont me permettre de mieux me structurer lors de l'implémentation du site web.

À ce moment, une fois les premiers croquis réalisés, je me suis décidé à me lancer dans l'implémentation des fonctionnalités de base comme l'authentification. Je devais choisir un Framework css pour gérer le visuel du site. Mon premier choix s'est porté sur Bootstrap, car c'est le Framework que j'ai utilisé tout au long de mes études. Toutefois, lors de la recherche d'un Framework css, j'ai découvert Tailwind css, un tout nouveau Framework que j'ai testé et que j'ai trouvé simple d'utilisation. De plus, il offre un large choix de possibilités. J'ai voulu l'intégrer à l'application pour donner un aspect différent au site et lui amener de la nouveauté.

Finalement, une fois le css mis en place (partie *frontend*), j'ai commencé à implémenter les fonctionnalités et à m'assurer de leur bon fonctionnement. Mon site devait aussi être dynamique ; pour ceci, je m'étais d'abord dirigé vers JQuery et Ajax. Mais en recherchant sur le web, j'ai découvert le Framework full-stack nommé Livewire. Ce qui m'a plu dans ce Framework, c'est que je pouvais rendre mon site dynamique sans quitter le confort de Laravel. J'ai donc commencé à suivre un tuto pour comprendre le fonctionnement de ce Framework.

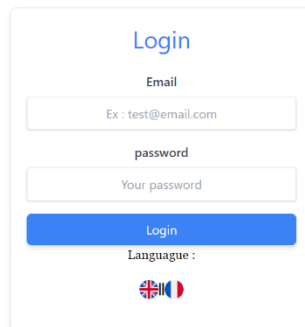
Une fois la gestion de l'étudiant terminée, j'ai voulu ajouter un système de messagerie entre les utilisateurs de l'application. Pour cela, j'ai utilisé un package nommé Chatify, créé par munafio. J'ai trouvé que ceux qui étaient proposés correspondaient bien au besoin formulé par l'établissement, et j'ai trouvé inutile de créer quelque chose de nouveau pour cela.

6. Implémentation

Dans cette partie, je vais vous mettre les parties du code et j'expliquerai leurs buts, ainsi que ma réflexion. J'expliquerai aussi le choix de certaines requêtes SQL.

6.1 Connexion à l'application.

Your experience on this site will be improved by allowing cookies. Allow cookies



The login form is centered on the page. It has a title 'Login' in blue. Below it are two input fields: 'Email' with a placeholder 'Ex : test@email.com' and 'password' with a placeholder 'Your password'. A blue 'Login' button is below the password field. At the bottom, there is a 'Language :' label followed by two flags: the UK flag and the French flag.

Tout d'abord, n'importe quel type de visiteur qui se rend sur l'application web devra se connecter avant de pouvoir faire quelque action que ce soit sur l'application (consulter, ajouter, éditer, supprimer, etc.).

```

Fortify::authenticateUsing(function (Request $request) {
    /**
     * Je vérifie que les données de connexion de l'utilisateur sont bien
     * validées. Donc je vérifie que l'adresse Email correspond bien à une
     * adresse email et je vérifie que le mot de passe a bien été rentré.
     */

    $request->validate([
        'email' => 'email',
        'password' => 'required'
    ],
    );
    /**
     * Strip_tags permet de supprimer tout le contenu HTML et PHP
     * d'une variable ensuite je recherche l'email que
     * l'utilisateur introduit dans la base de données.
     */
    $user = User::where('email', strip_tags($request->email))->first();
    /**
     * Je vérifie que l'user n'est pas nul et je check
     * si le mot de passe correspond bien au mot de passe
     * introduit par l'utilisateur. Lors de la création du compte
     * le mot de passe est crypté donc je dois utiliser le check
     * pour vérifier que le mot de passe introduit par l'utilisateur
     * correspond bien au mot de passe qui est crypté dans la base de
     * données.
     */
    if ($user &&
        Hash::check($request->password, $user->password)) {
        /**
         * Je stock l'email dans un cookie qui se nomme email.
         * Ce cookie va permettre à la prochaine connexion de l'utilisateur
         * à ne pas à ne pas avoir besoin de Ré entrer son adresse mail.
         */
        Cookie::queue(Cookie::make('email', $request->email, 60));
    }
    return $user;
});

```

Cette partie de code va permettre de gérer l'authentification sur l'application.

Je vérifie bien qu'il n'y ait pas d'injection de code HTML ou PHP depuis l'e-mail, à l'aide du "strip_tags" (). J'ai ajouté des validations sur les champs au moment où l'on récupère les données du formulaire pour être sûr que l'on rentre une adresse e-mail et pas autre chose.

6.2 Création de comptes

Comme mentionné précédemment dans l'analyse, seuls les administrateurs et responsables administratifs (secrétaire) peuvent créer des comptes. Les secrétaires peuvent juste créer le type de compte étudiant.

Sur l'image de droite, nous retrouvons un utilisateur avec un type de compte "administrateur".

Sur l'image de gauche, c'est un utilisateur avec un type de compte "secrétaire".

Creation d'un compte.

nom d'utilisateur

Nom d'utilisateur

prenom d'utilisateur

Nom d'utilisateur

Adresse email

Votre email

Téléphone

Votre numero

mot de passe

Votre mot de passe

Confirmation de mot de passe

Retapez votre mot de passe

Etablissement :

isfce

efp

Type de compte

etudiant

Créer mon compte

Creation d'un compte.

nom d'utilisateur

Nom d'utilisateur

prenom d'utilisateur

Nom d'utilisateur

Adresse email

Votre email

Téléphone

Votre numero

mot de passe

Votre mot de passe

Confirmation de mot de passe

Retapez votre mot de passe

Etablissement :

isfce

efp

Type de compte

administrateur

etudiant

secretariat

professeur

Referent_inclusif

Créer mon compte

Pour ceci, j'ai créé des méthodes dans mon modèle "User" qui vont vérifier quel est le type d'utilisateur connecté. J'ai créé deux méthodes : une "isSect"() et l'autre : "isAdmin". Ainsi, je peux les utiliser depuis ma vue pour accorder la visibilité des types de compte à créer.

```
public function isSect(){return (Auth::check() && $this->role_id === 3);}
public function isAdmin(){return (Auth::check() && $this->role_id === 1);}
```

```
{!-- dans cette partie je vérifie que l'utilisateur qui est connecté est
un(e)secrétaire et que dans ma liste des rôles le statut est bien
un statut étudiant. Alors je afficherai pour l'utilisateur authentifié
de type secrétaire le type de rôle étudiants à créer seulement.
--}}
@if($r->status === 'student' && Auth::user()->isSect())

<div class="flex justify-between items-center">
<label for="{{ $r->status }}">{{ $r->name }}
<input type="radio" value="{{ $r->id }}" id="{{ $r->statut }}" name="role_id">
</label>
</div>
{!-- Sinon si c'est un administrateur j'affiche tous les types de comptes. --}}
@elseif(Auth::user()->isAdmin())
<div class="flex justify-between items-center">
<label for="{{ $r->status }}">{{ $r->name }}
<input type="radio" value="{{ $r->id }}" id="{{ $r->statut }}" name="role_id">
</label>
</div>
@endif
```

6.3 Création d'un étudiant

La création d'un nouvel étudiant demande d'insérer le nom d'un médecin ; dans cette partie, j'ai permis à l'utilisateur de sélectionner des médecins depuis une liste déroulante. Si le médecin

ne se trouve pas dans les choix proposés, l'utilisateur pourra afficher des nouveaux champs qui lui demanderont d'encoder les informations du médecin choisi. Bien évidemment, tout ceci se réalise dynamiquement. J'ai utilisé "AlpineJs" pour faire une réalisation dynamique.

The image displays two screenshots of a web application interface. The top screenshot shows a form titled "Completez vos informations" with a dropdown menu for selecting a doctor. A red arrow points to the dropdown. The bottom screenshot shows the same form with a red arrow pointing to a blue profile icon, indicating a dynamic change in the form fields.

Completez vos informations

Selectionné un étudiant :
Mohammed Anouar Chairi Bounekoub

Selectionné un médecin :
Choisissez un médecin

Description de l'handicap :
Date de naissance :
jj-mm-aaaa
Rue :
commune :
Code postal :
ville :

Prenom du medecin :

Nom du medecin :

Telephone du medecin :

email du medecin :

Description de l'handicap :

Description de l'handicap

Une fois les données envoyées vers le contrôleur, je fais une vérification de la valeur du sélectionneur. Si la valeur est différente de "null", cela signifiera que l'utilisateur a trouvé le

médecin depuis la liste déroulante. Dans le cas contraire, le médecin a été encodé. À ce moment, on l'ajoute à la base de données.

```
/**
 * Je vérifie si la valeur médecin est bien a null. Si c'est le cas
 * cela signifiera que l'utilisateur a créé un nouveau médecin,
 * donc il faudra d'abord ajouter le nouveau médecin pour ensuite
 * ajouter le médecin qui vient d'être créé à la fiche signalétique
 * de l'étudiant.
 */
if($request->input('medecin') === null){
    $medecin= Medecin::create([
        'first_name' => $request->input('first_name'),
        'last_name' => $request->input('last_name'),
        'telephone' => $request->input('telephone'),
        'email' => $request->input('email'),
    ]);
}
$adresse = Adresse::create([
    'rue' => $request->input('rue'),
    'commune' => $request->input('commune'),
    'code_postal' => $request->input('code_postal'),
    'ville' => $request->input('ville'),
]);

Student::create([
    'fk_user' => $request->input('etudiant'),
    //Condition ternaire qui va permettre de savoir si le médecin
    //a été sélectionné dans la liste déroulante ou créez grâce aux inputs
    'fk_medecin' => empty($request->input('medecin')) ? $medecin->id : $request->input('me
    'fk_adresse' => $adresse->id,
    'description' => $request->input('description'),
    'date_birth' => $request->input('date_birth'),
    'fk_materiel' => $request->input('materiel')
]);
return redirect()->route('student.index');
```

6.4 Sélectionner le matériel de l'étudiant

Pour cette partie, j'ai utilisé le Framework Livewire. Pour cela, il faut déclarer la méthode "blade Livewire" dans la vue.

```

<div class="container">
  <div class=""></div>
  <div class="row mt-5">
    <div class="col-md_12">
      <div class="card">
        <div class="card-header">
          @livewire('dropdown-dynamics',[ 'id_student'=>$student->id ])
        </div>
      </div>
    </div>
  </div>
</div>

```

You, 3 months ago • table livewire

Ensuite, Je lance la commande Livewire qui va me créer la vue et le composant Livewire pour que je puisse faire les actions nécessaires.

Exemple : `php artisan make:livewire dropdown-dynamics`

Cela va me créer un fichier dans le dossier http->Livewire-> dropdown-dynamics.php et dans le dossier ressources->Livewire-> dropdown-dynamics.blade.php

L'instruction "blade Livewire" renvoie vers le composant qui lui, va renvoyer vers la vue.

J'envoie au composant l'id de l'étudiant. Cela va permettre de dire que les cours que l'on sélectionnés sont destinés à cet étudiant.

Dans le composant, nous allons retrouver 3 variables :

1. Id de l'étudiant ;
2. La valeur du premier select donc l'id de la section ;
3. La liste des cours sélectionnés.

Dans la fonction "render", on retourne vers la vue Livewire, qui envoie l'id de l'étudiant et la liste des section disponibles dans l'établissement.

Ci-dessous, une capture du code qui gère cela :


```

class DropdownDynamics extends Component
{
    //Permet de récupérer l'id de l'étudiant.
    public $id_student;
    //Permet de récupérer l'id de la section
    // que l'on récupère depuis la liste déroulante
    public $selectedSection = null;
    // permet de récupérer les cours de la section
    public $cours = null;

    public function render()
    {
        return view('livewire.dropdown-dynamics',[
            'student' => $this->id_student,
            'sections' => Section::all(),
        ]);
    }
}

```

Une fois dans la vue, nous retrouvons dans la balise “select” la directive “wire:model=selectedSection”, qui va permettre de renvoyer vers une méthode dans le composant.

```

<select name="section" id="section" wire:model="selectedSection" class="form-select block w-full focus:bg-white border" >

```

Dans le composant, j’ai créé une fonction “updatedSelectedSection” qui va permettre de mettre à jour les cours selon la section choisie. La fonction “recherche” dans la table “sectionCours” permet de rechercher tous les cours de la section choisie. Ensuite, grâce à la variable publique “selectedSection”, je récupère la section choisie par l’utilisateur. Enfin, je la stocke dans la variable “cours” que je récupère dans ma vue pour que la liste à choix multiple change dynamiquement.

```

// Fonction qui va permettre de sélectionner les cours
// selon la section choisie mais dynamiquement grâce à livewire

public function updatedSelectedSection($section_id){

    $this->cours = SectionCours::leftJoin("cours", 'cours.id' , '=', 'section_cours.fk_cours')
    ->where('fk_section', '=', $this->selectedSection)->get();

}

```

Voici le résultat en capture :

1

Add course

Selected a section :

Select Section ▼

submit

2

Add course

Selected a section :

Informatique de gestion ▼

Selected a course :

Gestion et exploitation de bases de données
INITIATION AUX BASES DE DONNEES

submit

6.5 Liste des étudiants

Une fois que l'étudiant est créé, l'utilisateur le verra s'afficher dans un tableau qui va contenir tous les étudiants. Selon le rôle de l'utilisateur, il pourra effectuer les actions suivantes : consulter, ajouter du matériel et les cours à l'étudiant.

Rechercher un étudiant			
NOM/PRENOM	DESCRIPTION	MEDecin	ACTION
Chairi Bounekoub Mohammed Anouar moroc3002@gmail.com	L'étudiant a un trouble de l'élocution.	NomMedecin2 PrenomMedecin2	<div>Ajout cours</div> <div>Afficher la fiche</div> <div>Ajout materiel</div> <div></div>

Ensuite, j'ai pensé que s'il y avait une grande quantité d'inscriptions pour les étudiants ayant un handicap, il faudrait prévoir un système de recherche d'étudiants pour le personnel administratif, professeur et référent inclusif. J'ai donc ajouté une barre de recherche qui va permettre de rechercher les étudiants selon leur prénom, nom et la description. Cela a été réalisé avec Livewire.

Pour cela, dans ma vue, je déclare l'instruction "blade" qui va rediriger vers le composant Livewire.

```
@extends('layouts.app') You, 4 months ago • first commit

@section('content')

<h1 class="text-6xl font-normal text-center leading-normal mt-0 mb-2 text-blue-800">Liste des etudiants</h1>
<livewire:student-table model="App\Models\student" />

@endsection
```

Dans mon composant, nous allons déclarer en variable publique la variable "filtre" qui a la valeur "Null" de base. Une fois dans la fonction "render", je récupère les étudiants dans la table "Student".

Ensuite, je vérifie ma variable "filtre". Si elle est différente de "null", cela signifie que l'utilisateur a essayé de faire une recherche sur la table. Lorsque que l'utilisateur clique sur la barre de recherche, la valeur du filtre prend une valeur vide. Si c'est le cas, on continue simplement. Ensuite, je fais la recherche de l'étudiant dans la table "Student".

Une fois la recherche effectuée, j'ai ajouté un système de pagination pour que l'utilisateur n'ait pas une trop grande liste sur une seule page. Enfin, j'affiche la liste des étudiants.

```

public function render()
{
    /**
     * Je prépare ma requête sur la table student.
     */
    /**
     * $students = Student::query();
     */
    /**
     * Tout d'abord je vérifie cela est a Null,
     * si c'est le cas je renvoie directement vers ma vue liveire.
     * Sinon si la variable filtre contient un élément,
     * donc j'effectue une recherche dans ma base de données
     * sur les champs first name last name et description
     * ainsi je pourrais afficher l'étudiant recherché selon
     * l'un des 3 champs.
     */
    if(!empty($this->filters)){
        foreach($this->filters as $key => $value){
            if(empty($value)){
                continue;
            }
            $student = $students->leftjoin("users",'users.id' , '=' , 'students.fk_user')
            ->where('users.first_name','like',"%. $value. %")
            ->orWhere('users.last_name','like',"%. $value. %")
            ->orWhere('description','like',"%. $value. %");
        }
    }
    /**
     * J'ai ajouté un système de pagination après le 10e
     * étudiant affiché dans mon tableau ,
     * le 11e étudiant va s'afficher dans une nouvelle page.
     */
    $students = $students->paginate(10);

    return view('livewire.student-table',['students' => $students]);
}

```

À savoir, pour l'exemple, dans l'image ci-dessous, j'ai mis la pagination à 1.

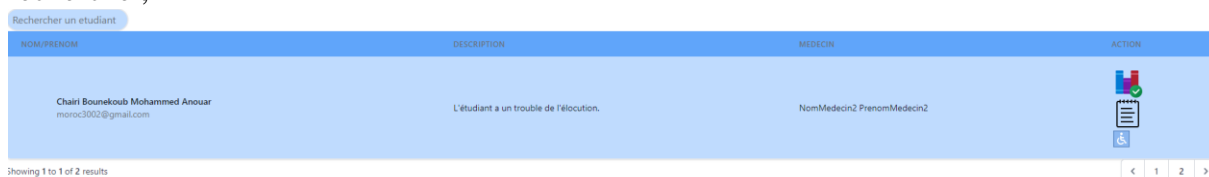
Donc, on est sur la première page et en bas à droite, nous avons la page 2 pour afficher les étudiant suivants sur une autre page.

Rechercher un étudiant			
NOM/PRENOM	DESCRIPTION	MEDECIN	ACTION
Chairi Bounekoub Mohammed Anouar moroc3002@gmail.com	L'étudiant a un trouble de l'élocution.	NomMedecin2 PrenomMedecin2	 
Showing 1 to 1 of 2 results			
<div> <div><</div> <div>1</div> <div>2</div> <div>></div> </div>			

Rechercher un étudiant			
NOM/PRENOM	DESCRIPTION	MEDECIN	ACTION
Chairir Halloum Maroine moroc3002@gmail.com	L'étudiant a un trouble de la concentration. L'étudiant a besoin de toujours être au premier rang à tous ses cours.	NomMedecin2 PrenomMedecin1	 
Showing 2 to 2 of 2 results			
<div> <div><</div> <div>1</div> <div>2</div> <div>></div> </div>			

Pour la partie de la recherche, je tiens à préciser qu'une fois que l'utilisateur a recherché son nom, le filtrage recherche sur l'ensemble des pages.

- 1) Me retrouvant sur la page où il y a le premier étudiant, je me place sur la barre de recherche ;

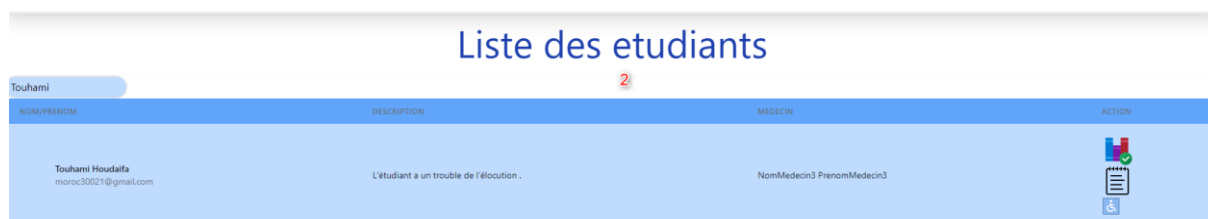


Rechercher un étudiant

NOM/PRENOM	DESCRIPTION	MEDECIN	ACTION
Chairi Bounekoub Mohammed Anouar moroc3002@gmail.com	L'étudiant a un trouble de l'élocution.	NomMedecin2 PrenomMedecin2	

Showing 1 to 1 of 2 results

- 2) Je tape le nom Touhami, le tableau se met à jour et m'affiche l'étudiant.



Liste des etudiants

Touhami

NOM/PRENOM	DESCRIPTION	MEDECIN	ACTION
Touhami Houdalla moroc30021@gmail.com	L'étudiant a un trouble de l'élocution .	NomMedecin3 PrenomMedecin3	

6.6 La fiche signalétique

Nous voici arrivés à la partie la plus importante : la fiche signalétique.

Le personnel administratif, personnel pédagogique (professeurs) et le référent inclusif auront accès à cette fiche depuis la liste des étudiants, comme montré dans la partie 5.5.

Dans la fiche signalétique, nous retrouverons toutes les informations de l'étudiant, telles que son nom, prénom, la description de son handicap et ses coordonnées téléphoniques et postales.

Dans la partie du milieu, nous retrouverons les informations concernant son médecin. La particularité de cette partie est que le référent inclusif aura la possibilité d'ajouter un commentaire. Ce commentaire contiendra toutes les remarques du médecin, car le référent doit s'assurer de pouvoir appeler le médecin juste après la création de la fiche signalétique. C'est pour cela que l'ajout du commentaire se fait à cette étape.

Ensuite, tout à droite, nous retrouverons les cours auxquels l'étudiant s'est inscrit pour cette année.

Il y aura comme information : l'intitulé du cours et le nom du chargé de cours.

Tout en bas de la fiche signalétique, nous retrouverons le matériel dont l'étudiant a besoin. J'ai fait le choix de mettre les informations du matériel dans un tableau, car le référent aura besoin de mettre à jour la partie "remarques". Dans la partie disponible, nous retrouvons un voyant lumineux rouge, qui signifie que le matériel n'a pas encore été vérifié par le référent. Une fois que le référent se sera assuré de la disponibilité du matériel, il pourra modifier cette valeur pour la changer en « vérifié » ou bien la laisser à « non vérifié ». Dans les deux cas, le référent pourra ajouter une remarque qui s'affichera sur la fiche signalétique et le système enverra un message de la remarque à tous les chargés de cours.

Management of materials
Social network
Student Management
Course
Meeting
Helpful document
Sign out

Fiche signalétique

Information personnelle de l'etudiant
Prenom : Mohammed Anouar
nom : Chairi Bounekoub
Né le : 06-01-1996
Email : moroc3002@gmail.com
Description :
L'étudiant a un trouble de l'élocution.
Coordonnée :
Rue : Avenue H.V Wolvens
Commune : WOLUWE-SAINT-LAMBERT
Code postale : 1200
Ville : Bruxelles

Information sur le medecin
Prenom : PrenomMedecin2
nom :NomMedecin2
Telephone : 0488627125
Email : test@gmail.com
Comment :
Not yet noticed by the doctor
add comment

Cours inscrit :
Gestion et exploitation de bases de données
INITIATION AUX BASES DE DONNEES
MATHÉMATIQUES APPLIQUÉES A L'INFORMATIQUE
PRINCIPES D'ANALYSE INFORMATIQUE

Besoin materiel:

Name materiel	remarque	Disponible	mise a jour
Place de parking	Aucune remarque actuellement		Modifier

Voici un exemple de ce que le référent peut indiquer comme remarque.

Remarque

File Edit View Format

↶ ↷ Paragraph B I

Place de parking

Disponibilité :

Problème de disponibilité de la place de parking entre 2 étudiants. Plage horaire 13h00-14h00.

Solution :

La solution que je peux proposer c'est l'ouverture de la porte du garage 13h-14h pour que tous les étudiants avec un handicap puissent trouver une place facilement.

P = SPAN
POWERED BY TINY

Mis a disposition
☒ Actif ☐ Non actif

Envoyer

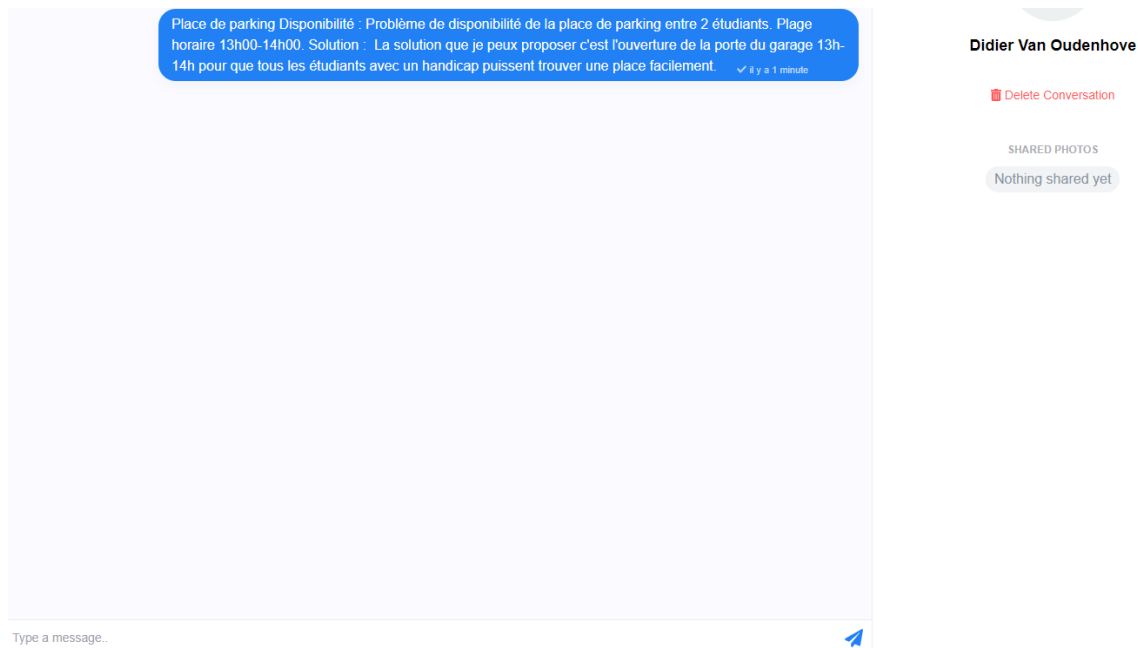
Après l'envoi des modifications des données, le système renvoie vers la fiche signalétique et envoie un message au professeur concerné depuis l'application Chat.

Modification des données du matériel depuis la fiche signalétique :

Besoin materiel:

Name materiel	remarque	Disponible	mise a jour
Place de parking	Disponibilité : Problème de disponibilité de la place de parking entre 2 étudiants. Plage horaire 13h00-14h00. Solution : La solution que je peux proposer c'est l'ouverture de la porte du garage 13h-14h pour que tous les étudiants avec un handicap puissent trouver une place facilement.		Modifier

Envoi automatique du message sur l'application au professeur :



Voici la partie du code qui gère cette modification :

```
/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id,$id_student)
{
    $validatedData = $request->validate([
        'remarque' => 'required',
        'is_ok' => 'required'
    ]);

    StudentMateriel::whereId($id)->update($validatedData);
    $studentCours = StudentCours::select('*')
    ->leftJoin('Cours','Cours.id','=', 'student_cours.fk_cours')
    ->where('student_cours.fk_student','=', $id_student)
    ->get();
    $test=[];
    $i=0;
    /**
     * Mohammed A.
     * commentaire :
     * Récupération des professeurs qui ont l'étudiant dans le cours
     * ils doivent apparaître dans le tableau test je ferai un arrêt
     * unique qui récupérera les professeurs une seule fois.
     * afin de leur envoyer un message
     */
    foreach($studentCours as $val){
        $test[$i] = $val->fk_user;
        $i++;
    }
    $unique = array_unique($test);
}
```

Explication :

Tout d'abord, je valide les données insérées grâce au système de validation. Ensuite, je modifie le matériel de l'étudiant concerné.

Une fois ceci fait, je récupère tous les cours de l'étudiant. Je crée un tableau qui va contenir les professeurs avec lesquels l'étudiant a cours. Il est possible qu'un professeur apparaisse deux fois, car il peut donner cours dans plusieurs matières.

Ensuite, à partir de cette table, je vérifie qu'il n'y ait pas de doublons, car les professeurs ne doivent pas recevoir le message plusieurs fois.

```
foreach($unique as $val){  
  
    $messageID = mt_rand(9, 999999999) + time();  
    Chatify::newMessage([  
        'id' => $messageID,  
        'type' => "user",  
        'from_id' => Auth::user()->id,  
        'to_id' => $val,  
        'body' => $request["remarque"],  
        'attachment' => null,  
    ]);  
    // send to user using pusher  
    Chatify::push('private-chatify', 'messaging', [  
        'from_id' => Auth::user()->id,  
        'to_id' => $val,  
        'message' => $request['remarque']  
    ]);  
};  
  
return redirect('/etudiant/'.$id.'/fiche');
```

6.7 Système de Chat (utilisation de Chatify)

Pour cette partie, j'ai utilisé un système de chat déjà existant qui a été développé par Munaf A. Mahdi. Il utilise la technologie des "web Socket". Le "web socket" permet de faire communiquer la partie serveur et la partie client dans les deux sens. Il permet à un client d'envoyer un message au serveur, et inversement. Cela va permettre à l'application Chat de recevoir des messages en temps réel, par exemple, de recevoir un message d'une personne pendant que l'on est occupé à lui écrire. Pour ceci, le système de Chat utilise l'API "pusher". "Pusher" permet d'ajouter en toute sécurité des fonctionnalités en temps réel via les "web socket". L'avantage à utiliser ce package est que le système de création de Channel a été déjà configuré par le créateur et l'auteur de Chatify. Cela m'a fait gagner du temps dans sa configuration.

En résumé, voici ce qu'il faut savoir sur "pusher" pour pouvoir le configurer.

Canal : permet de différencier les flux de données auprès d'une application. Une application peut posséder plusieurs canaux et un canal peut avoir plusieurs clients.

Événement : permet que le serveur envoie les données au client pour qu'il puisse les voir en temps réel. Par exemple, dans un Chat style "WhatsApp", "Messenger", etc., l'événement est déclenché par une bibliothèque d'éditeur et les clients peuvent s'abonner à ces événements.

Il existe trois types de canaux :

Chaînes publiques : ce sont des chaînes auxquelles tout le monde peut avoir accès, les personnes doivent juste connaître le nom de la chaîne.

Chaînes privées : canal auquel seuls les utilisateurs authentifiés peuvent avoir accès.

Chaînes de présence : idem que pour les canaux privés, mais permettent en plus de notifier les autres clients connectés.

La partie qui gère tout ceci se trouve dans le fichier "code.js" dans le "public->js->chatify->code.js" :

Exemple d'un channel :

```
// subscribe to the channel
var channel = pusher.subscribe("private-chatify");

// Listen to messages, and append if data received
channel.bind("messaging", function(data) {
  if (data.from_id == messenger.split("_")[1] && data.to_id == auth_id) {
    $(".messages")
      .find(".message-hint")
      .remove();
    messagesContainer.find(".messages").append(data.message);
    scrollTopBottom(messagesContainer);
    makeSeen(true);
    // remove unseen counter for the user from the contacts list
    $(".messenger-list-item[data-contact=" + messenger.split("_")[1] + "]")
      .find("tr>td>b")
      .remove();
  }
});
```

6.8 Cookie consentement

Pour réaliser la bannière des cookies consentement, j'ai utilisé un package qui a été fait par la compagnie "Spatie". Donc, pour installer le package, j'ai utilisé la commande suivante :
`composer require spatie/laravel-cookie-consent`.

J'ai inclus dans ma vue « app.blade.php » cette bannière pour qu'elle s'affiche sur chaque page de mon site pour que l'utilisateur l'accepte.

```

</head>
@if (Auth::user())
<div class="">

    @include('incs.navbar')
    @include('cookieConsent::index')
</div>

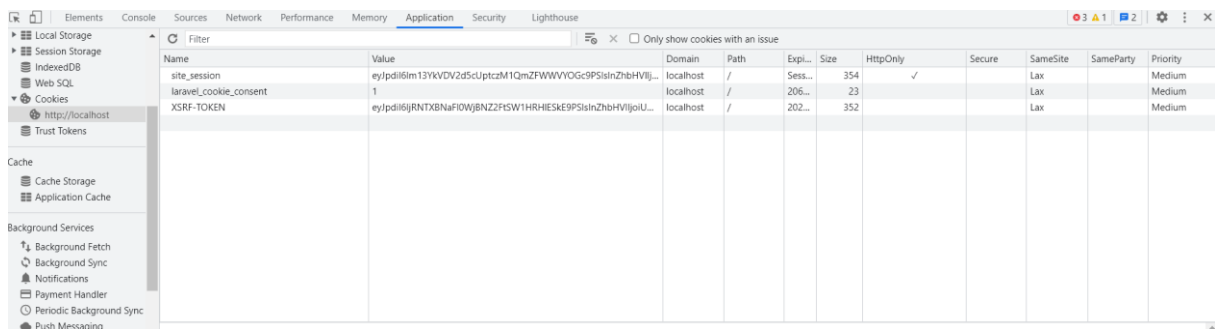
@else
<div>
    @include('cookieConsent::index')
</div>
@endif
<body class="">

```

Ensuite, je devais juste gérer le ccs de la bannière.

Ce site nécessite l'autorisation de cookies pour fonctionner correctement. Accepter

Le package gère déjà l'activation de la bannière si le cookie a été accepté. Si l'utilisateur appuie sur "accepter le cookie", cela équivaut à "1" et la bannière ne sera plus affichée.



Name	Value	Domain	Path	Expires	Size	HttpOnly	Secure	SameSite	SameParty	Priority
site_session	eyJpdjI6Im13YkVDV2d5cUptczM1QmZFWWVVOGc9PSIsInZhbHVlZj...	localhost	/	Sess...	354			Lax		Medium
laravel_cookie_consent	1	localhost	/	206...	23	✓		Lax		Medium
XSRF-TOKEN	eyJpdjI6jRNTXBnaF0WjBNZ2F5SW1HRHIESke9PSIsInZhbHVlZjoiU...	localhost	/	202...	352			Lax		Medium

7. Sécurité et règlement RGPD

Dans cette partie, je vais vous présenter les informations récoltées sur le RGPD. Ainsi, je vais vous montrer ce que j'ai pensé pour que l'utilisateur soit informé de ce que nous allons faire à propos des données que l'on va récolter.

7.1 Qu'est-ce que le RGPD ?

Le RGPD est l'acronyme de Règlement général sur la protection des données. Il est appliqué depuis le 25 mai 2018. Il permet de protéger l'individu et ses données à caractère personnelles. Par exemple son nom, numéro d'identification, données de localisation, identifiant en ligne, ou des éléments plus spécifiques propres à son identité physique, telles que les éléments physiques, génétiques, psychologiques, économiques, culturels ou sociaux.

7.2 Les implications du RGPD pour les entreprises :

- « Privacy by default and by design » : toute personne qui traite des données personnelles doit automatiquement garantir le plus haut niveau de protection des données, de la conception technique à la collecte.
- Enregistrement des activités de traitement : pour les entreprises de 250 personnes ou plus les entreprises qui traitent régulièrement des données personnelles ; les entreprises qui traitent des données sensibles et les entreprises susceptibles d'effectuer des traitements, toutes doivent tenir un registre, car il existe un risque pour les droits et libertés de la personne. Pour les autres, sous contrôle de l'autorité compétente, elles devraient également en conserver un pour leur propre protection, mais sans obligation.
- En cas de violation de la CPS (par exemple, si la base de données est piratée), l'organisation est obligée d'informer le personnel concerné de la situation.
- Ils doivent également revoir tous les processus contractuels.
- Enfin, nous leur recommandons fortement de désigner une personne qui sera spécifiquement chargée de la mise en œuvre de la surveillance RGPD. Elle s'assurera également que l'entreprise est conforme aux différentes normes en vigueur.

7.3 Définir les besoins :

Il est très difficile de déterminer si une donnée est personnelle ou non.

Les recommandations de la DPA pour l'application du RGPD au marketing constituent la base de sa définition. Nous pouvons traiter "toute information relative à un individu ou des données personnelles, quelle que soit la façon dont vous vous occupez de votre enfant en tant qu'individu". Si cet individu est associé à une ou plusieurs données, il

peut alors identifier et personnaliser les personnes physiques, appartenant aux données personnelles. Union, adhésion, données médicales, orientation sexuelle, etc.

7.4 Faire preuve de transparence :

Le concept est simple : fournir un maximum d'informations claires et compréhensibles par écrit. Cette information doit être facilement accessible, telle qu'elle est affichée. L'obligation de transparence s'applique à la fois au moment de la collecte des données et pendant le traitement des données. Si une personne vous contacte à leur sujet, vous êtes tenu d'informer cette personne des actions qu'elle a demandées ainsi que de celles qui s'y rapportent.

7.5 Consentement des cookies.

Par rapport aux données récupérées par les cookies en 2021, il y a eu quelques changements.

- La simple poursuite de navigation sur un site ne peut plus être considérée comme une validation de consentement ;
- Les utilisateurs doivent accepter de déposer leur consentement par une action claire et positive, comme cliquer sur un bouton “j’accepte” dans une bannière cookie.
- Les utilisateurs devront être en mesure de retirer leur consentement ;
- Refuser les cookies doit être aussi facile que de les accepter.

À savoir : dans le futur, il risque d’avoir un durcissement de la réglementation sur les cookies donc, cela devra bien être surveillé dans les prochaines années.

7.6 La sécurité

Laravel facilite la protection contre les attaques de falsification de requêtes intersites (CSRF).

Les falsifications de requêtes sont des commandes non autorisées qui sont lancées au nom d’un utilisateur authentifié. Laravel génère un jeton CSRF pour chaque session utilisateur. Ce jeton va permettre de vérifier que l’utilisateur connecté est celui qui effectue réellement les actions. Dans chaque formulaire de l’application, j’ai inclus le jeton CSRF qui sera masqué dans le formulaire. Ainsi, le middleware de protection CSRF pourra valider la demande.

Laravel permet de créer ses propres middlewares personnalisés. J’en ai créé par exemple pour bloquer le contenu de certaines pages à certains rôles. J’ai utilisé aussi le middleware “Auth” pour que les utilisateurs non-connectés aient accès aux pages des personnes identifiées.

Laravel permet aussi de sécuriser l’url. Le procédé est simple : on crée des autorisations grâce à la commande “php artisan make : policy UserPolicy –model=user”. Dans le fichier créé, nous allons retrouver des fonctions nous permettant de protéger par exemple la méthode “update”. Ensuite, il ne reste plus qu’à utiliser l’autorisation dans le contrôleur “\$this->authorize(‘update’, \$user)”. Cela permet qu’un utilisateur modifie, par exemple, uniquement les formulaires ou l’url correspondant à son id.

8. Problèmes rencontrés et solutions

Lors de la conception de cette application, j'ai rencontré certains problèmes auxquels j'ai dû remédier.

Le premier problème que j'ai rencontré a été le fait que je me suis lancé dans la nouvelle version de Laravel, qui était à ce moment-là Laravel 8. Le souci avec cette version, c'est qu'il y a eu des changements et que j'ai dû m'adapter au changement de certaines syntaxes au niveau des routes. Par rapport à l'authentification, Laravel 8 proposait surtout Fortify, fusionnée avec "jetStream" et cela ne me convenait pas vraiment. J'ai dû faire des recherches pour bien comprendre le principe de cette nouvelle technologie pour s'identifier. Au début de la version 8 de Laravel, j'ai rencontré des problèmes dans certains packages que j'utilisais dans la version précédente et qui n'avaient pas encore été mis à jour avec la version 8 de Laravel. Cela m'a permis de prendre conscience qu'il faut attendre un peu avant de se lancer dans une nouvelle version et bien se renseigner sur les changements avant de se lancer, car cela peut affecter l'avancement d'un projet.

Je me suis aussi rendu compte, au fur et à mesure de l'avancement du projet, que certaines fonctionnalités étaient un peu plus complexes à développer. Cela m'a demandé beaucoup de temps de codage et la date de remise du projet approchait. Le cas pour lequel j'ai eu le plus gros souci est le système de Chat, qui fonctionne avec le système de "web socket". J'ai pu réaliser le système de Chat, mais le souci est que le système de fonctionnement en temps réel ne voulait pas fonctionner. Après avoir passé plus ou moins dix jours à trouver une solution j'ai dû me rabattre sur le package "Chatify". Ce package utilise aussi le système de web socket, mais toute la configuration était déjà mise en place, cela m'a permis de pouvoir me concentrer sur d'autres aspects de l'application.

9. Futur de l'application

L'application sera sûrement utilisée pour l'établissement et il se peut que celui-ci veuille continuer à développer d'autres fonctionnalités dans les années à venir. Un système de forum pourrait bien s'ajouter pour que le personnel administratif et personnel pédagogique discute sur des cas de handicaps et que cela développe un accès à tous les utilisateurs de l'application.

L'application pourrait être aussi développée pour qu'elle puisse servir, non seulement à un seul établissement, mais à tous les établissements de promotion sociale se trouvant dans la région de la fédération Wallonie-Bruxelles.

10. Conclusion

Ce travail de fin d'études a été une excellente occasion pour moi d'utiliser toutes les connaissances que j'ai acquises tout au long de mes études à l'I.S.F.C.E. Cela m'a permis de prendre conscience des difficultés d'un projet et de me rendre compte de la raison pour laquelle les entreprises travaillent en équipe de trois à cinq personnes sur un projet. Ce n'est pas le fait qu'une personne ne serait pas capable de gérer un projet seul, mais cela demande des ressources et beaucoup de temps. Certaines fonctionnalités demandent beaucoup plus de temps à développer que d'autres et avoir une équipe aide certainement à gagner du temps. Cela rendra l'application plus performante et sécurisée, car il y aurait plusieurs personnes et non pas une seule à gérer chaque fonctionnalité de l'application.

Points forts :

- J'ai pu renforcer mes connaissances sur le Framework Laravel. Je sais maintenant sur quoi j'aimerais me concentrer dans le développement web. J'ai pu constater quels sont les aspects sur lesquels je me sentais le plus confortable. Cela me donne une voie sur les points que je devrai améliorer et sur lesquels me former.
- Ce TFE m'a permis d'être plus autonome, car au cours de ce projet, je n'ai pas eu d'aide dans le développement des fonctionnalités que j'ai eues à réaliser. Ce qui m'a permis de vraiment voir tous les aspects d'une application web. Cela me préparera à intégrer le monde professionnel où les entreprises attendent de l'autonomie de la part des personnes avec qui elles collaborent.

Points faibles :

- Le plus gros point faible que j'ai pu constater dans la réalisation de ce projet est la gestion de mon temps et l'organisation des tâches. J'ai eu du mal à organiser mon travail et cela m'a porté préjudice. Le côté positif est que cela m'a donné une expérience et je sais à présent comment faire pour que, dans mes futurs projets, je choisisse la meilleure façon d'être performant et de gagner du temps, tout en respectant les délais demandés.
- J'ai aussi constaté que dans le domaine du développement web, il faut que je m'améliore dans le domaine de la sécurité. Après mon diplôme, je compte bien suivre une formation dans la sécurité des données, car dans l'avenir, la protection des données sera plus importante que les fonctionnalités que propose une application.

11. Bibliographie

<https://laravel.sillo.org/laravel-8/>

<https://www.udemy.com/course/le-guide-de-laravel-8/>

<https://www.udemy.com/course/creez-une-messagerie-temps-reel-multiroom-laravel-livewire/>

<https://www.udemy.com/course/analyse-et-conception-de-systeme-dinformation-merise-co/>

<https://laravel.com/docs/8.x>

<https://laravel.sillo.org/cours-laravel-8-la-securite-on-se-protege/>

<https://www.webrankinfo.com/dossiers/droit-internet/consentement-cookies>

<https://www.poush.be/blog/pourquoi-et-comment-appliquer-le-rgpd-a-votre-site-internet/>

<https://github.com/munafio/chatify>

<https://github.com/spatie/laravel-cookie-consent>

<https://www.youtube.com/playlist?list=PLeeuvNW2FHVj4vHJRj9UDeDsXshHlnHJk>

<https://tailwindcss.com/>

<https://laravel-livewire.com/>