|  |
| --- |
| TEAM ABC |
| DOSSIER DES EXIGENCES SYSTEME |
| GESTION DU CENTRE DE FORMATION |

|  |
| --- |
| NOUH BAKHLAKH  SALMA EL MAIMOUNI  NADA EL KHAMLICHI |

Table des matières

[Section 1 : Introduction 5](#_Toc105608412)

[1.1 Objectif 5](#_Toc105608413)

[1.2 Portée et caractéristiques du projet 5](#_Toc105608414)

[1.3 public visé 6](#_Toc105608415)

[1.4 Définitions, Acronymes et Abréviations 6](#_Toc105608416)

[1.4.1 Définitions 6](#_Toc105608417)

[1.4.2 Les Acronymes et Abréviations 7](#_Toc105608418)

[1.5 References 8](#_Toc105608419)

[Section 2: Description général 8](#_Toc105608420)

[2.1 Perspective du produit 8](#_Toc105608421)

[2.2 Catégories des utilisateurs et ses caractéristiques 8](#_Toc105608422)

[2.3 Contraintes 9](#_Toc105608423)

[Section 3 : Description détaillé 9](#_Toc105608424)

[3.1 Utilisateurs 9](#_Toc105608425)

[3.2 Authentification 10](#_Toc105608426)

[3.2.1 Mécanisme d’authentification 10](#_Toc105608427)

[3.2.2 Interface d’authentification 12](#_Toc105608428)

[3.3 Inscrirons un nouvel utilisateur (Formateur/Administrateur) 15](#_Toc105608429)

[3.4 Les donnes gérer par le system 15](#_Toc105608430)

[3.4.1 flux du travail de system 16](#_Toc105608431)

[3.5 Structure des donnes et DB 18](#_Toc105608432)

[3.6 WEB API 18](#_Toc105608433)

[3.7 Frontend 19](#_Toc105608434)

[3.8 Informations Complémentaires 19](#_Toc105608435)

[4 Cas d’utilisation 20](#_Toc105608436)

[4.1 Connexion : 20](#_Toc105608437)

[4.2 Consultation d’aperçu de system 21](#_Toc105608438)

[4.3 Consulter les filières 22](#_Toc105608439)

[4.3.1 Modifier les informations d’une filière 23](#_Toc105608440)

[4.3.2 Modifier les modules d’une filière 24](#_Toc105608441)

[4.3.3 Modifier les informations des modules d’une filière 25](#_Toc105608442)

[4.3.4 Supprimer un module d’une filière sélectionner 26](#_Toc105608443)

[4.3.5 Ajouter une nouvelle filière 27](#_Toc105608444)

[4.3.6 Supprimer une filière 28](#_Toc105608445)

[4.4 Gestion des modules 29](#_Toc105608446)

[4.4.1 Ajout d’un module 29](#_Toc105608447)

[4.4.2 Supprimer un module 30](#_Toc105608448)

[4.4.3 Modifier un module 31](#_Toc105608449)

[4.4.4 Exporter les modules 33](#_Toc105608450)

[4.4.5 Rechercher un module 33](#_Toc105608451)

[4.5 Gestion des stagiaires 33](#_Toc105608452)

[4.5.1 Ajouter un stagiaire 33](#_Toc105608453)

[4.5.2 modifier les informations d’un stagiaire 33](#_Toc105608454)

[4.5.3 supprimer un stagiaire 33](#_Toc105608455)

[4.5.4 Exporter les stagiaires 33](#_Toc105608456)

[4.5.5 Rechercher un stagiaire 33](#_Toc105608457)

[4.6 Gestion des groupes 33](#_Toc105608458)

[4.6.1 Créer un groupe 33](#_Toc105608459)

[4.6.2 Supprimer un groupe 33](#_Toc105608460)

[4.6.3 Affecter des stagiaires un groupe 33](#_Toc105608461)

[4.6.4 Supprimer des stagiaires d’un groupe 33](#_Toc105608462)

[4.7 Gestion des salles 33](#_Toc105608463)

[4.7.1 Ajouter une salle 33](#_Toc105608464)

[4.7.2 modifier les informations d’une salle 33](#_Toc105608465)

[4.7.3 supprimer une salle 33](#_Toc105608466)

[4.7.4 Exporter une liste des salles 33](#_Toc105608467)

[4.8 Gestion des séances 33](#_Toc105608468)

[4.8.1 Ajouter une séance 33](#_Toc105608469)

[4.8.2 modifier les informations d’une séance 33](#_Toc105608470)

[4.8.3 supprimer une séance 33](#_Toc105608471)

[4.8.4 Exporter les séances 33](#_Toc105608472)

[4.8.5 Rechercher les séances 33](#_Toc105608473)

[4.9 Gestion des absences 33](#_Toc105608474)

[4.9.1 Gérons les absences par séance 33](#_Toc105608475)

[5 Les interfaces 34](#_Toc105608476)

[5.1 Gestion des filières 34](#_Toc105608477)

[34](#_Toc105608478)

[5.2 Gestion des modules 39](#_Toc105608479)

[5.3 Gestion des stagiaires 42](#_Toc105608480)

[5.4 Gestion des groupes 49](#_Toc105608481)

[5.5 Gestion des salles 49](#_Toc105608482)

[5.6 Gestion des séances 50](#_Toc105608483)

[5.7 Page d’accueil 53](#_Toc105608484)

[5.8 Backend API Endpoint visualiser 54](#_Toc105608485)

# Section 1 : Introduction

Le titre du projet est TRANING CENTER MANAGEMENT SYSTEM (TCMS). TCMS est une application basée sur le web visant à fournir des services de gestion pour l’administration de l’organisation. Ce system peut être utiliser pour la gestion des informations dans un centre de formation.

Pour un certain utilisateur, l’administrateur du system va créer un nom d’utilisateur et un mot de passe, utilisant cette combination, l’utilisateur peut accéder au système pour faire des opérations de mise à jour ou de consultation en interaction avec la base des données.

Le Frontend va être réaliser utilisant la librairie du javascript REACT, en autre part la logique des affaires va être en C# avec .NET CORE 6 WEB API résident dans la couche intermédiaire. Ces couches vont interagir avec une troisième couche de base des données, qu’il va être une base des données SQL-SERVER.

## 1.1 Objectif

L’objectif de ce document est de présenter une description détaillée de TRANING CENTER MANAGEMENT SYSTEM. Il va présenter la finalité et les caractéristiques du système, les interfaces du système, que fera le système et les contraintes sous laquelle il doit fonctionner. Ce document est destiné à la fois au client et au développeurs du system et va être proposer au comité de jugement.

## 1.2 Portée et caractéristiques du projet

Ce système va être un système de la gestion de centre de formation pour les membres de l’administration d’une organisation. Ce système va être designer pour maximiser la productivité administrative en fournissant des outils assistant à l’automatisation des certain procédures et processus qui autrement devrait être effectuer manuellement, ainsi que facilitent l’accès et la modification des information en relation avec l’organisation, en maximisant la production et l’efficacité des effort de l’administration le système va répondre aux besoins des utilisateurs en restant facile à comprendre et utiliser. C’est un system convivial pour consulter, gérer et accéder à l’information.

## 1.3 public visé

Ce document est visé principalement aux utilisateurs, nous avons divisé le reste du document en des sous-sections différentes. Nous suggestions que vous commencez par comprenant les définitions, les Acronymes et les abréviations, puis parcourez séquentiellement le contenu, la section de présentation et passez par les sections de description détaillée les plus pertinentes.

## 1.4 Définitions, Acronymes et Abréviations

### 1.4.1 Définitions

* **HTML:** Hypertext Markup Language is a specification for graphical layout of a document. The specification calls for the document to be stored as text containing a series of tags that contain formatting information
* **JavaScript:** is a high level, dynamic, untyped, and interpreted programming language.
* **SQL-SERVER:** SQL-SERVER is a Microsoft developed management system.
* **standalone application:** A Software that is not a part of some bundled software. A program that is run as a separate computer process, not an add-on of an existing process. Standalone program, a program that does not require operating system’s services to run. A portable application, which can be run without the need for installation procedure.
* **Web Portal:** A web portal is most often one specially designed web page that brings information together from diverse sources in a uniform way.
* **Data Integrity:** Data integrity refers to maintaining and assuring the accuracy and consistency of data over its entire life cycle, and is a critical aspect to the design, implementation and usage of any system which stores, processes, or retrieves data.
* **Browser :**  Logiciel utiliser pour afficher les documents HyperText, Google Chrome and Firefox sont des exemples de Browser.

### 1.4.2 Les Acronymes et Abréviations

|  |  |
| --- | --- |
| Acronymes | Signification |
| JWT | JSON Web token |
| API | Application Programming Interface |
| JSON | JavaScript Object Notation |
| DBMS | Database Management System |
| MUI | Material UI |
| HTML | Hypertext Markup Language |
| CSS | Cascading Style Sheets |
| GUI | Graphical User Interface |
| MBPS | Mega-Byte Per Second |
| OS | Operating System |
| JS | JAVASCRIPT |
| URL | Universal Resource Locator |

## 1.5 References

# Section 2: Description général

## 2.1 Perspective du produit

Le produit va être une application autonome and peuvent être exécuter on différents systèmes via le réseau internet. Le produit nécessitera une machine qui peuvent exécuter un navigateur et connecter à internet.

## 2.2 Catégories des utilisateurs et ses caractéristiques

Le public ciblé pour TCMS est les membres d’administration du centre de formation ainsi que les formateurs. Les Utilisateurs du système sont :

* Directeur (Administrateur) : le Super-User de système.
* Formateur : Utilisateur avec un accès limite.

## 2.3 Contraintes

Les contraintes actuelles sur le projet sont liées à la provision des ressources matérielle et logicielle.

* La connexion internet est une contrainte pour l’application, puisque l’application récupère les donnes de la base de donnes en internet,   
  il est important qu'ils aient une connexion internet pour l’application à fonctionner.
* Un navigateur web modern (IE6+)

# Section 3 : Description détaillé

## 3.1 Utilisateurs

Puisque nous n'allons avoir que 2 types d'utilisateurs, les données que nous allons gérer pour cette partie ne seront pas compliquées.

Une image contenant texte

Description générée automatiquement

Ce modèle enregistrerait toutes les informations nécessaires sur nos utilisateurs et en même temps limiterait leur accès via l'attribut « Level ».

Pour des raisons de sécurité on a décidé d’encrypter les mots de passe en binaire utilisant la méthode SHA252, enregistrent le « Hash » et le « Salt ». Utilisant cette méthode l’Authentification et l’inscription des nouveaux utilisateurs va être plus sécurise. Pour qu’on authentifie un utilisateur on doit d’abord encrypter le mot de passe donne par lui et le comparer avec le mot de passe encrypter qu’on a dans la base de donnes. Les méthodes d’encryptions va être discuter dans le chapitre d’Authentification.

## 3.2 Authentification

### 3.2.1 Mécanisme d’authentification

JWT pour JSON Web Token est une méthode sécurisée d’échange d’informations, décrite par la [RFC 7519](https://tools.ietf.org/html/rfc7519). L’information est échangée sous la forme d’un jeton signé afin de pouvoir en vérifier la légitimité. Ce jeton est compact et peut être inclus dans une URL sans poser de problème.

JWT est couramment utilisé pour implémenter des mécanismes d’authentification stateless pour des SPA (Single Page Application) ou pour des applications mobiles.

Un JWT est composé de trois parties, chacune contenant des informations différentes :

* Un header,
* Un payload (les “claims”),
* La signature.

Le header et le payload sont structurés en JSON. Ces trois parties sont chacune encodées en base64url, puis concaténées en utilisant des points (“.”).

* Le header identifie quel algorithme a été utilisé pour générer la signature, ainsi que le type de token dont il s’agit (souvent JWT, mais le champ a été prévu dans le cas où l’application traite d’autres types d’objet qu’un JWT).
* Le payload est la partie du token qui contient les informations que l’on souhaite transmettre. Ces informations sont appelées “claims”. Il est possible d’ajouter au token les claims que l’on souhaite, mais un certain nombre de claims sont déjà prévus dans les spécifications de JWT. Dans ce payload en peut inclue une identification à l’utilisateur qui en train d’authentifier ‘sub’, avec l’émetteur du token qui s’appelle ‘iss (issuer)’ ainsi que la date d’expiration du token. Cette dernière est très importante afin de limiter la durée de vie du token. Si la date d’expiration est dépassée, le token sera rejeté.

Exemple

Requête :

Une image contenant texte, capture d’écran, moniteur, écran

Description générée automatiquement

Réponse :

Une image contenant texte

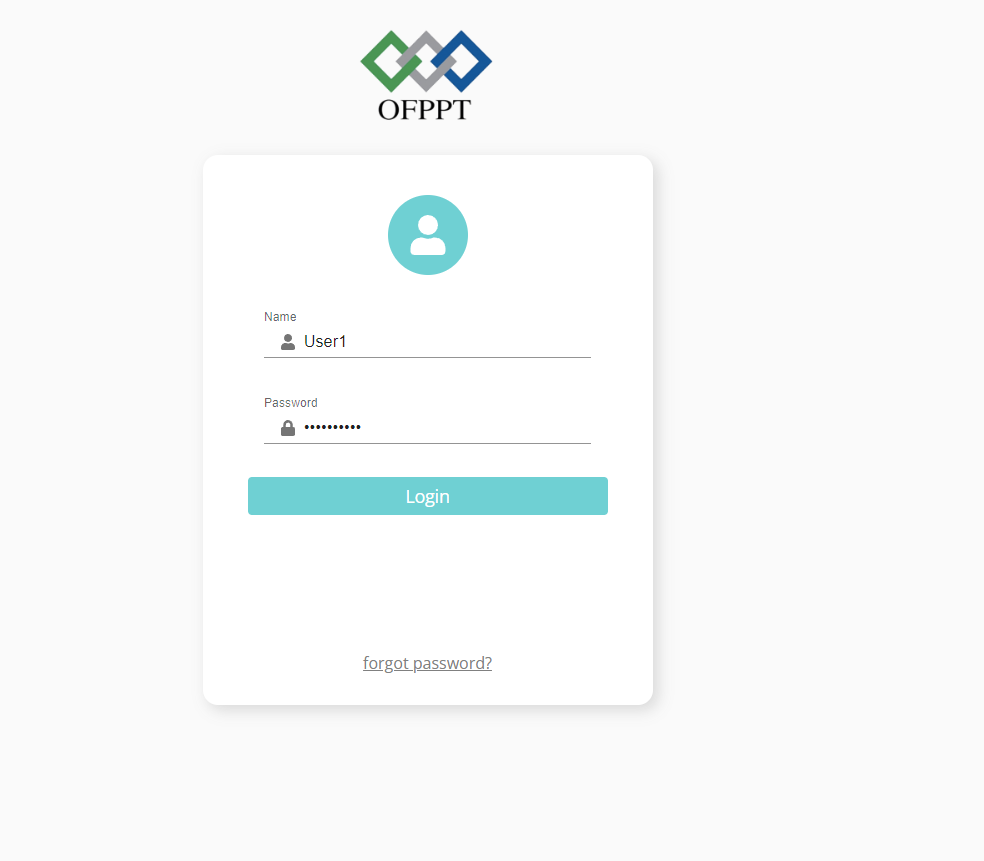
Description générée automatiquement

Token : Une image contenant texte

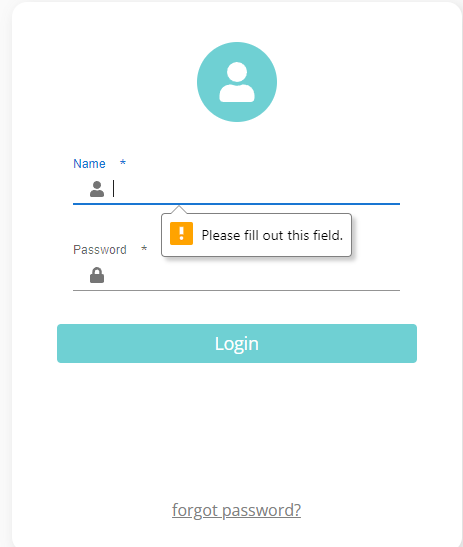
Description générée automatiquement

### 3.2.2 Interface d’authentification

On a développé un UI simple avec une forme d’authentification avec deux champs nécessaires et un Button de soumission.



En cliquons sur le Button Login, l’application va vérifier le saisie de tous les champs et afficher un message si la condition n’a pas été effectuer.



Si l’utilisateur a saisie tous les champs, l’application va envoyer une requête de type POST au ‘API Endpoint’ qui correspond à l’authentification, dans ce cas ‘/api/Login’. Cette Endpoint va recevez les informations que l’utilisateur a saisie puis les vérifier. Si les donnes sont valide le serveur va répondre avec un code http 201 et les informations liées à cet utilisateur encore avec la JWT, sinon le serveur va répondre avec un code 401 (UNAUTHORIZED) pas autorise.

Console de navigateur :

Une image contenant texte

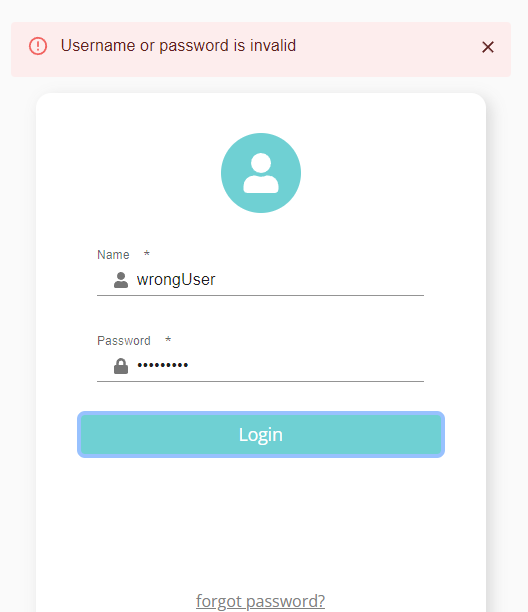
Description générée automatiquement

Réponse JSON du serveur :

Une image contenant texte

Description générée automatiquement

Message en UI :



Cas des Information correcte :

Une image contenant texte

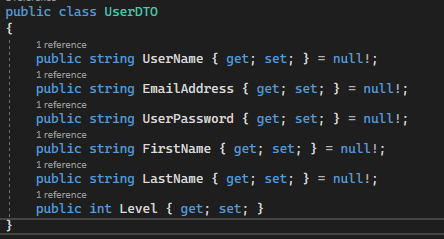
Description générée automatiquement

Ces Informations va être utiliser dans l’application pour faire des requête CRUD au serveur plus tard (aussi pour des raisons d’affichage).

L’utilisateur va être rediriger directement vers la page d’accueil après succès en authentification.

## 3.3 Inscrirons un nouvel utilisateur (Formateur/Administrateur)

Pour que l’administrateur peut inscrire un nouvel utilisateur au system soit être un autre administrateur ou un utilisateur avec accès limite, Il doit écrire une requête POST avec la structure ci-dessous en format JSON. On n’a pas développé une UI pour l’inscription, seulement pour l’Authentification.



## 3.4 Les donnes gérer par le system

Le system gèrent tous les donnes en relation avec un centre de formation en générale, c’est à dire le system permettre l’administrateur de faire :

* Gestion des modules
* Gestion des stagiaires
* Gestion des groupes
* Gestion des séances
* Gestion des formateurs
* Gestion d’absence
* Gestion des filières

### 3.4.1 flux du travail de system

Comme expliquer dans Section 1 : Introduction, Le system va avoir 3 couches, Une base de donnes SQL-SERVER qui servira a stocker tous les donnes de l’application, un « WEB API » .NET 6 qui servira comme une couche intermédiaire entre la base des données et l’interface utilisateur, relève généralement de la définition de l'API REST, une API est un ensemble de définitions et de protocoles qui facilite la création et l'intégration de logiciels d'applications, une solution pour partager des ressources et des informations, tout en maintenant un certain niveau de sécurité, de contrôle et d'authentification, en déterminant qui est autorisé à accéder à quoi. Ce dernier va recevez des demandes des donnes sous forme des requêtes HTTP, l’émetteur de ces demandes va être la dernière couche, une application web frontend développer en javascript-JSX avec la librairie REACT intégrons des librairies des composants (MUI, Bootstrap, FORMIK, easy-peasy), cette application va faire des requêtes http à l’aide du librairie js AXIOS pour recevez les donnes et les structurer pour afficher à l’utilisateur. Enfin pour achever cette dynamique I faut d’abord commencer par la création d’une DB à la base d’un schéma relationnelle bien détaille puis passer à la création de notre WEBAPI configurons les modèles des donnes, l’authentification, l’Autorisation, la sécurité (CORS), les contrôleurs et les librairies qu’on va utiliser ...etc. après tout ça on peut accéder au développement de l’application REACT en frontend. Il faut aussi organiser toutes les ressources utiliser d’une manière professionnelle, séparant les services des composants et les pages etc.., dans la partie frontend, et séparant la logique d’application des services et les modèles dans le backend.

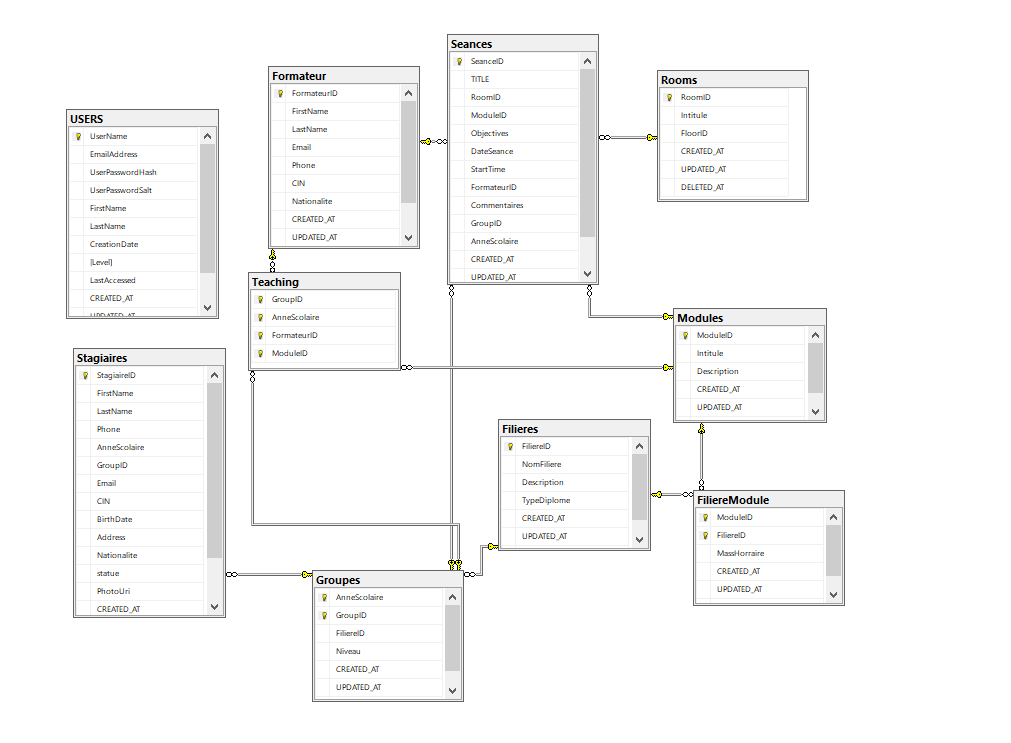
Exemple de structure :

Une image contenant texte

Description générée automatiquement Une image contenant texte

Description générée automatiquement

## 3.5 Structure des donnes et DB

Pour assurer que toutes les fonctionnalités du system sont achevés, on a designer un schéma relationnelle qui va stocker toutes les donnes nécessaires. 

## 3.6 WEB API

Pour assurer une communication sécurise et au même temps rapide et efficace on a choisi WEB API .NET 6 comme un Framework de travail. La gestion d’un projet web api est relativement la même qu’un projet MVC, la seule différence est que le web api n’inclus pas les vues, autre que ça la structure et le flux de travail est le même.

Pour configurer la connexion entre l’application web api et la base de donne on a utilisé « Entity Framework ». En utilisant le PMC on peut exécuter des commandes pour nous aidons a créé les modèles de base de donne avec le contexte automatiquement. Cela est possible grâce aux package additionnelle du « Entity Framework » référencer dans la section 3.7 Informations Complémentaires .

## 3.7 Frontend

L'idée principale derrière **React** est que vous construisiez votre application à partir de **composants**. Un **composant** regroupe à la fois le HTML, le JS et le CSS, créés sur mesure pour vos besoins, et que vous pouvez réutiliser pour construire des interfaces utilisateurs.

On a divisé le projet en plusieurs parties pour mettre le code plus efficace et lisible :

* Assets : contenant les images et les dessins utiliser dans le projet.
* Pages : contentant les pages réaliser en des fichier JS et JSX.
* Forms : contenants les formulaires utiliser dans l’application.
* Components : contenants les composants utiliser dans tout l’application.
* Constants
* Services : contenants les services utiliser dans le system (Authentification, backend calls)
* Styles : contenants les fichier CSS utiliser pour styler les composants, les formulaires et les pages.

Tous ça va être enregistrer dans le dossier SRC qui contient tout le code source.

## 3.8 Informations Complémentaires

Les packages additionnelle utiliser pour implémenter le mécanisme d’authentification sont :

Microsoft.IdentityModel.Tokens;

System.IdentityModel.Tokens.Jwt.

System.Security.Claims;

Les package encryption utiliser sont :

System.Security.Cryptography;

Avec des fonctions :

Une image contenant texte, moniteur, capture d’écran, écran

Description générée automatiquement

L’envoie des informations au saisie par l’utilisateur va être réaliser utilisant la librairie du JS Axios qui offre tous les types de requêtes HTTP :

« Axios » : "^0.26.0"

Exemple d’implémentation :



Une image contenant texte

Description générée automatiquement

# 4 Cas d’utilisation

## 4.1 Connexion :

|  |  |
| --- | --- |
| UCID : | 1 |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur a connecté. |
| Acteurs | Gestionnaire de compte |
| Précondition : | L’utilisateur doit avoir un compte pour connecter. |
| Post Condition : | 1. System va enregistrer la date de connexion et rediriger l’utilisateur vers un page menu. |
| Flux normal | 1. Utilisateur : Entre les donnes d’authentification et cliquer sur un Button connecter. 2. System : Vérifie que tous les champs sont saisis, Envoi une requête au serveur backend pour vérifier les donnes saisies par l’utilisateur, puis le rediriger vers un page menu. |
| Flux alternative | 1. Utilisateur : Entre les donnes d’authentification et cliquer sur un Button 2. System : Vérifie que tous les champs sont saisis, affiche un message sinon. |
| Flux alternative | 1. Utilisateur : Entre les donnes d’authentification et cliquer sur un Button 2. System : Vérifie que tous les champs sont saisis, Envoi une requête au serveur backend pour vérifier les donnes saisies par l’utilisateur, affiche un message d’erreur d’authentification |
| Références | N/A |

## 4.2 Consultation d’aperçu de system

|  |  |
| --- | --- |
| UCID : | O1 |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur accèdes a la page Home qui présente un résume de system. |
| Acteurs | Gestionnaire de compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. Le gestionnaire du compte peut consulter un résume de system dans une page. |
| Flux normal | 1. Utilisateur : Accédé a la page Home 2. System : Affiche un résume de system qui inclue les filières, stagiaires, absences, groupes etc.… avec un menu pour accéder aux autres fonctionnalités system, et des buttons d’utilisations varies. |
| Flux alternative | N/A |
| Références | N/A |

## 4.3 Consulter les filières

|  |  |
| --- | --- |
| UCID : | F1/C |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur accèdes au page de gestion des filières. |
| Acteurs | Le gestionnaire du compte. |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. Utilisateur peut consulter, ajouter modifier et supprimer toutes les informations relatives à une filière. |
| Flux normal | 1. Utilisateur : Accède au page gestion des filières 2. System : Affiche toutes les informations relatives à une filière, associer avec des buttons de défilement et des zones de recherche etc… 3. Utilisateur : Choisis une filière et appuy sur le Button modules pour consulter les modules de cette filière. 4. System : Affiche une modal qui Inclus les informations de la filière avec ses module avec la possibilité d’ajouter un module ou supprimer ou modifier les heures assigner a cette module dans la filière sélectionner. |
| Flux alternative | N/A |
| Références | N/A |

## 4.3.1 Modifier les informations d’une filière

|  |  |
| --- | --- |
| UCID : | F1/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur choisi a modifié une filière. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications faites par l’utilisateur. |
| Flux normal | 1. Utilisateur : choisie une filière et appuie sur le Button modifier. 2. System : affiche une modal avec des champs de saisie qui contient les informations précèdent de la filière, avec un Button de confirmation et d’annulation. 3. Utilisateur : Modifie les champs désirer et confirme la modification. 4. System : Valide tous les champs et envoie une requête au backend pour sauvegarder les changements dans la base de donnes. |
| Flux alternative | 1. Utilisateur : choisie une filière et appuie sur le Button modifier. 2. System : affiche une modal avec des champs de saisie qui contient les informations précèdent de la filière, avec un Button de confirmation et d’annulation. 3. Utilisateur : Annuler la modification. 4. System : Cacher le modal et affiche la page précèdent. |
| Exception | 1. Utilisateur : choisie une filière et appuie sur le Button modifier. 2. System : affiche une modal avec des champs de saisie qui contient les informations précèdent de la filière, avec un Button de confirmation et d’annulation. 3. Utilisateur : Modifie les champs désirer et confirme la modification. 4. System : Valide tous les champs et affiche un message d’erreur sur les champs qui ne valide pas certaine condition. 5. Utilisateur : Modifier les champs pour suivre les conditions puis clique sur le Button modifier. 6. FN-4 |
| Références | N/A |

## 4.3.2 Modifier les modules d’une filière

|  |  |
| --- | --- |
| UCID : | F2/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur choisi a modifié les modules d’une filière. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. 2. L’utilisateur peut consulter les modules d’une filière, aussi qu’ajouter un module existant à la filière sélectionner, le supprimer, ou modifier les heures lie à ce module dans cette filière. |
| Flux normal | 1. Utilisateur : Choisie une filière et appuie sur le Button modules associer 2. System : Affiche un modal qui inclue : tous les modules de la filière sélectionner associer avec deux buttons de suppression et de modification pour modifier les heures de module dans cette filière, et avec une liste déroulante et une zone de texte pour ajouter un model existant à la filière sélectionner. 3. Utilisateur : choisie un module de la liste déroulante et saisie le nombre des heures puis appuie sur le Button associer pour sauvegarder les changements. 4. System : Vérifie que les donnes son saisis et qu’ils suivent les condition fixe précédemment. Puis faire une requête pour sauvegarder les changements dans la base de donnes |
| Flux alternative | 1. Utilisateur : FN-1 2. System : FN-2 3. Utilisateur : Appuie sur le Button annuler 4. System : Cache le modal et retourne à la page précèdent. |
| Exception | 1. Utilisateur : FN-1 2. System : FN-2 3. FN-3 4. System : vérifie que tous les donnes son saisis et qu’ils suivent les conditions. Affiche un message d’erreur et annule l’opération de sauvegarde. |
| Références | N/A |

## 4.3.3 Modifier les informations des modules d’une filière

|  |  |
| --- | --- |
| UCID : | F3/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur choisi a modifié les modules d’une filière. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. 2. L’utilisateur peut consulter les modules d’une filière, aussi qu’ajouter un module existant à la filière sélectionner, le supprimer, ou modifier les heures lie à ce module dans cette filière. |
| Flux normal | F2/MAJ \_F-N1, FN-2   1. Utilisateur : choisie un module à modifier 2. System : affiche une zone de texte qui contient les informations précèdent associe avec un Button de sauvegarde. 3. Utilisateur : modifie les informations et choisie à sauvegarder les nouvelles donnes. 4. System : Vérifie que les donnes son saisis et qu’ils suivent les condition fixe précédemment. Puis faire une requête pour sauvegarder les changements dans la base de donnes. |
| Flux alternative | F2/MAJ \_F-N1, FN-2   1. Utilisateur : choisie un module à modifier 2. System : affiche une zone de texte qui contient les informations précèdent associe avec un Button de sauvegarde. 3. Utilisateur : Choisie à annuler l’opération. 4. System : Cache le modal et retourne à la page précèdent. |
| Exception | F2/MAJ \_F-N1, FN-2   1. Utilisateur : choisie un module a modifier 2. System : affiche une zone de texte qui contient les informations précèdent associe avec un Button de sauvegarde. 3. Utilisateur : modifie les informations et choisie à sauvegarder les nouvelles donnes. 4. System : Vérifie que les donnes son saisis et qu’ils suivent les condition fixe précédemment. Affiche un message d’erreur et annule l’opération de sauvegarde |
| Références | N/A |

## 4.3.4 Supprimer un module d’une filière sélectionner

|  |  |
| --- | --- |
| UCID : | F4/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur choisi à supprimer un module d’une filière sélectionner. |
| Acteurs | Le gestionnaire du compte. |
| Précondition : | 1. Avoir un compte administrateur. 2. Être connecter au system. 3. Être autoriser. |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. 2. L’utilisateur peut consulter les modules d’une filière, aussi qu’ajouter un module existant à la filière sélectionner, le supprimer, ou modifier les heures lie à ce module dans cette filière. |
| Flux normal | F2/MAJ \_F-N1, FN-2   1. Utilisateur : Choisie un module à supprimer d’après une liste. 2. System : Affiche un message de confirmation de suppression. 3. Utilisateur : Confirme la suppression. 4. System : Faire une requête au backend pour supprimer le module de cette filière avec toutes les informations relatives. Puis affiche la nouvelle liste des modules rafraichie. |
| Flux alternative | F2/MAJ \_F-N1, FN-2   1. Utilisateur : Choisie un module à supprimer d’après une liste. 2. System : Affiche un message de confirmation de suppression. 3. Utilisateur : Annule la suppression. 4. System : cacher le message de confirmation et retourne a l’interface précédant. |
| Exception | N/A |
| Références | N/A |

## 4.3.5 Ajouter une nouvelle filière

|  |  |
| --- | --- |
| UCID : | F5/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur veut ajouter une nouvelle filière. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. |
| Flux normal | 1. Utilisateur : Appuie sur Button d’ajout 2. System : affiche une modal avec une forme d’ajout 3. Utilisateur : entre les donnes et confirme l’ajout 4. System : vérifie les champs et les conditions, puis envoie une requête pour sauvegarder les donnes. |
| Flux alternative | FN-1, FN-2  3. Utilisateur : Annuler l’ajout  4. System : Annule l’opération, puis cacher le modal afficher l’interface précédant |
| Exception | FN-1, FN-2, FN-3   1. System : vérifie les champs et les conditions. Affiche un message d’erreur. |
| Références | N/A |

## 4.3.6 Supprimer une filière

|  |  |
| --- | --- |
| UCID : | F6/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur supprime une filière. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. |
| Flux normal | 1. Utilisateur : Choisie une filière à supprimer. 2. System : Affiche un message de confirmation de suppression. 3. Utilisateur : Confirme la suppression. 4. System : Faire une requête au backend pour supprimer la filière avec toutes les informations relatives. Puis affiche la nouvelle liste des filières. |
| Flux alternative | 1. Utilisateur : Choisie une filière à supprimer. 2. System : Affiche un message de confirmation de suppression. 3. Utilisateur : Annule la suppression. 4. System : Annule l’opération, puis afficher l’interface précédant |
| Exception | N/A |
| Références | N/A |

## 4.4 Gestion des modules

|  |  |
| --- | --- |
| UCID : | M1/C |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur veut consulter les modules. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. L’utilisateur peut consulter les modules aussi qu’ajouter un nouveau module, le supprimer, ou le modifier. |
| Flux normal | 1. Utilisateur : Accède au page gestion des modules. 2. System : Affiche toutes les informations relatives à un module dans une grille, associer avec des buttons de défilement et des zones de recherche etc… |
| Flux alternative | N/A |
| Exception | N/A |
| Références | N/A |

### Ajout d’un module

|  |  |
| --- | --- |
| UCID : | M1/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur veut ajouter un nouveau module. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. |
| Flux normal | 1. Utilisateur : Appuie sur Button d’ajout 2. System : affiche une modal avec une forme d’ajout 3. Utilisateur : entre les donnes et confirme l’ajout 4. System : vérifie les champs et les conditions, puis envoie une requête pour sauvegarder les donnes. |
| Flux alternative | FN-1, FN-2  3. Utilisateur : Annuler l’ajout  4. System : Annule l’opération, puis cacher le modal afficher l’interface précédant |
| Exception | FN-1, FN-2, FN-3   1. System : vérifie les champs et les conditions. Affiche un message d’erreur. |
| Références | N/A |

### Supprimer un module

|  |  |
| --- | --- |
| UCID : | M2/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur supprime un module. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications de l’utilisateur dans la base des données. |
| Flux normal | 1. Utilisateur : Choisie un module à supprimer. 2. System : Affiche un message de confirmation de suppression. 3. Utilisateur : Confirme la suppression. 4. System : supprimer le module avec toutes les informations relatives. Puis affiche la nouvelle liste des modules. |
| Flux alternative | 1. Utilisateur : Choisie un module à supprimer. 2. System : Affiche un message de confirmation de suppression. 3. Utilisateur : Annule la suppression. 4. System : Annule l’opération, puis afficher l’interface précédant |
| Exception | N/A |
| Références | N/A |

### Modifier un module

|  |  |
| --- | --- |
| UCID : | M3/MAJ |
| Description : | Ce cas d’utilisation d’écris le flux de system après qu’un utilisateur choisi a modifié un module. |
| Acteurs | Le gestionnaire du compte |
| Précondition : | 1. Avoir un compte administrateur 2. Être connecter au system 3. Être autoriser |
| Post Condition : | 1. System enregistre les modifications faites par l’utilisateur. |
| Flux normal | 1. Utilisateur : choisie un module et appuie sur le Button modifier. 2. System : affiche une modal avec des champs de saisie qui contient les informations précèdent, avec un Button de confirmation et d’annulation. 3. Utilisateur : Modifie les champs désirer et confirme la modification. 4. System : Valide tous les champs et sauvegarde les changements. |
| Flux alternative | 1. Utilisateur : choisie un module et appuie sur le Button modifier. 2. System : affiche une modal avec des champs de saisie qui contient les informations précèdent, avec un Button de confirmation et d’annulation. 3. Utilisateur : Annuler la modification. 4. System : Cacher le modal et affiche la page précèdent. |
| Exception | 1. Utilisateur : choisie un module et appuie sur le Button modifier. 2. System : affiche une modal avec des champs de saisie qui contient les informations précèdent, avec un Button de confirmation et d’annulation. 3. Utilisateur : Modifie les champs désirer et confirme la modification. 4. System : Valide tous les champs et affiche un message d’erreur sur les champs qui ne valide pas certaine condition. |

### Exporter les modules

### Rechercher un module

## 4.5 Gestion des stagiaires

### 4.5.1 Ajouter un stagiaire

### 4.5.2 modifier les informations d’un stagiaire

### 4.5.3 supprimer un stagiaire

### 4.5.4 Exporter les stagiaires

### 4.5.5 Rechercher un stagiaire

## 4.6 Gestion des groupes

### 4.6.1 Créer un groupe

### 4.6.2 Supprimer un groupe

### 4.6.3 Affecter des stagiaires un groupe

### 4.6.4 Supprimer des stagiaires d’un groupe

## 4.7 Gestion des salles

### 4.7.1 Ajouter une salle

### 4.7.2 modifier les informations d’une salle

### 4.7.3 supprimer une salle

### 4.7.4 Exporter une liste des salles

## 4.8 Gestion des séances

### 4.8.1 Ajouter une séance

### 4.8.2 modifier les informations d’une séance

### 4.8.3 supprimer une séance

### 4.8.4 Exporter les séances

### 4.8.5 Rechercher les séances

## 4.9 Gestion des absences

### 4.9.1 Gérons les absences par séance

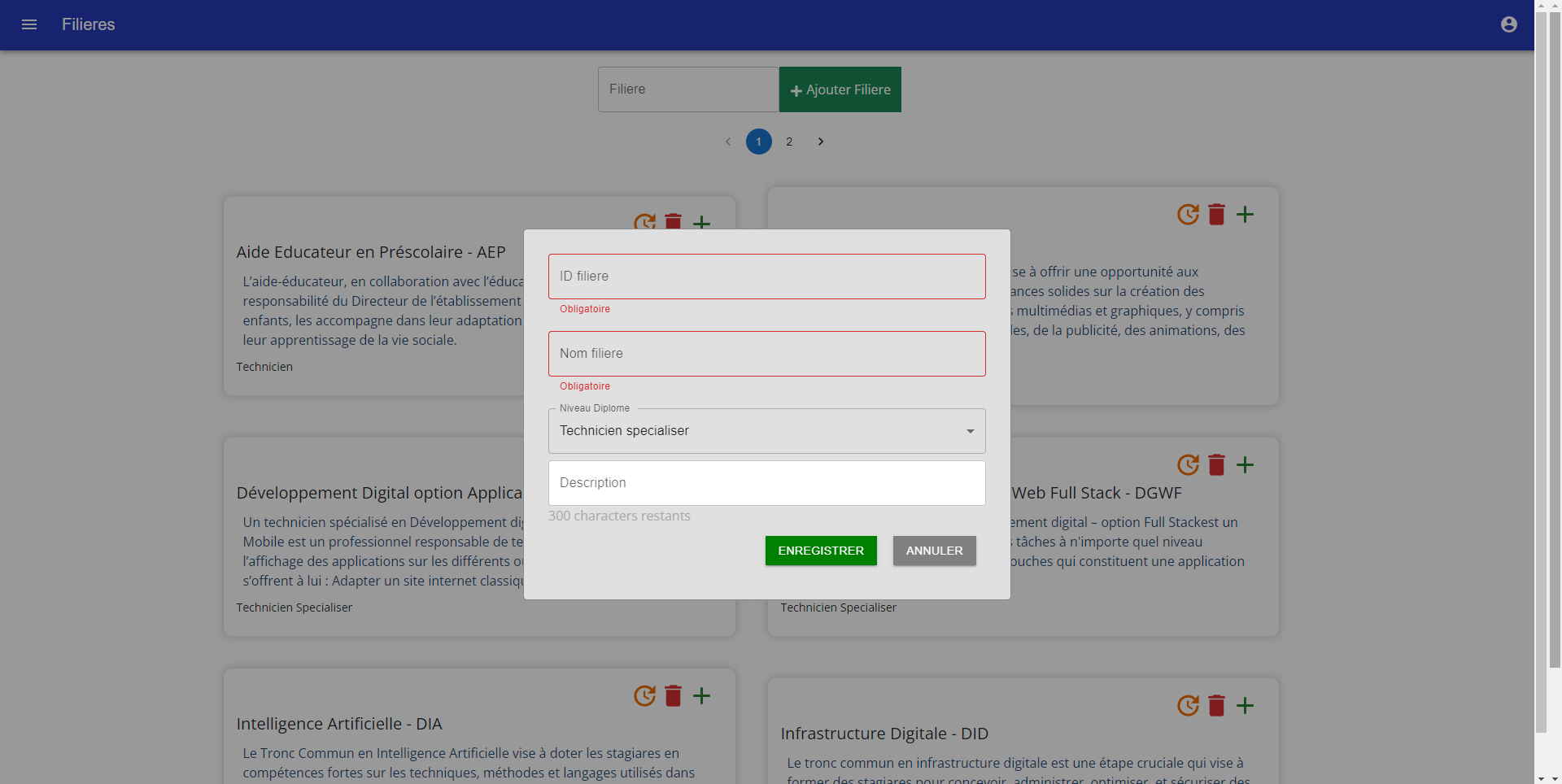
# 5 Les interfaces

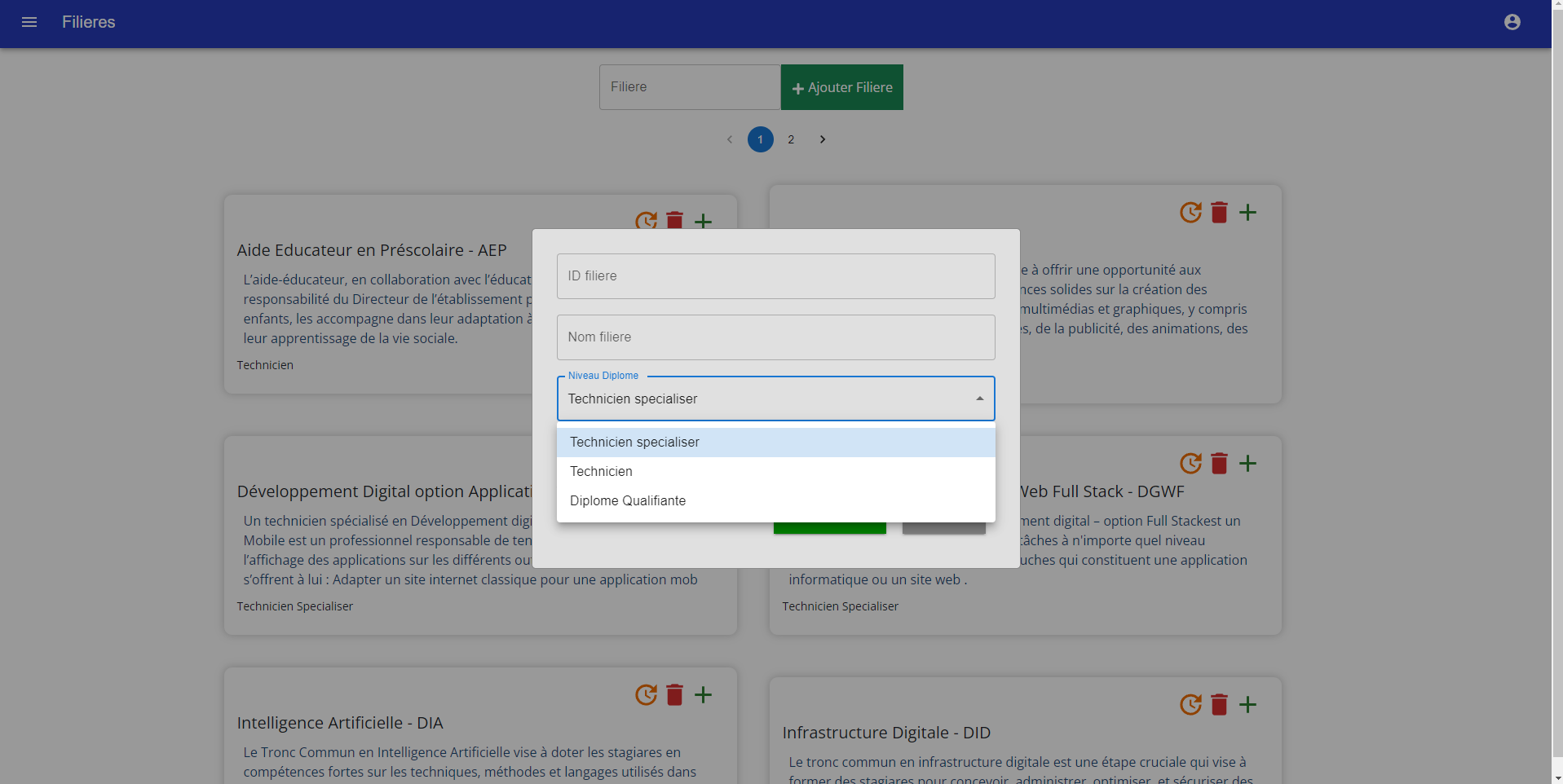
## Gestion des filières

Interface contient tous les opérations CRUD

## Une image contenant texte Description générée automatiquement

Ajouter une filière :





Modifier les informations d’une filière existant :

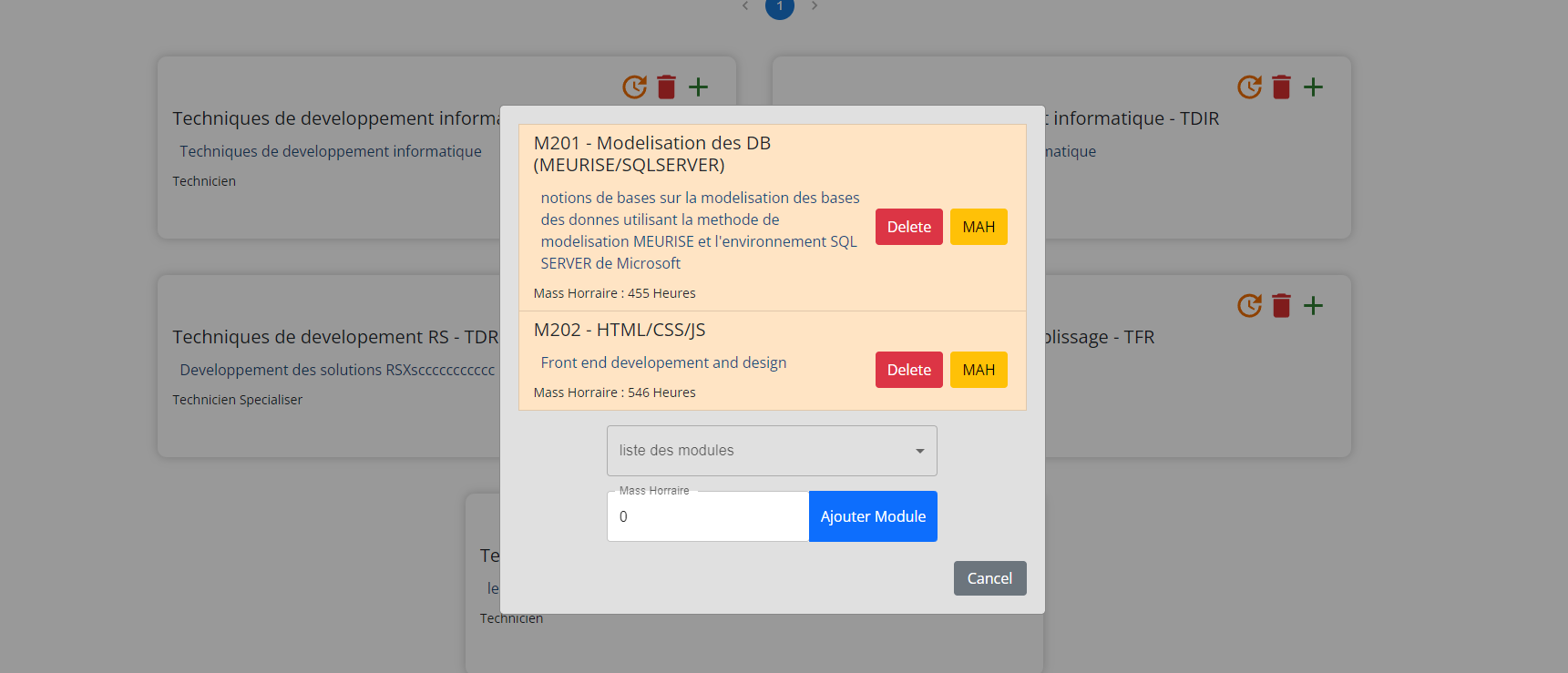
Une image contenant texte

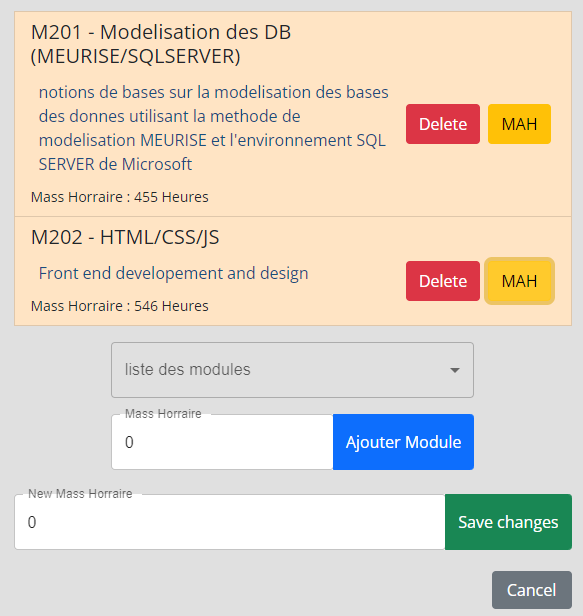
Description générée automatiquement

Consulter les modules d’une filière existant :

Une image contenant texte

Description générée automatiquement





Suppression :

Une image contenant texte

Description générée automatiquement

Message de succès :

Une image contenant texte

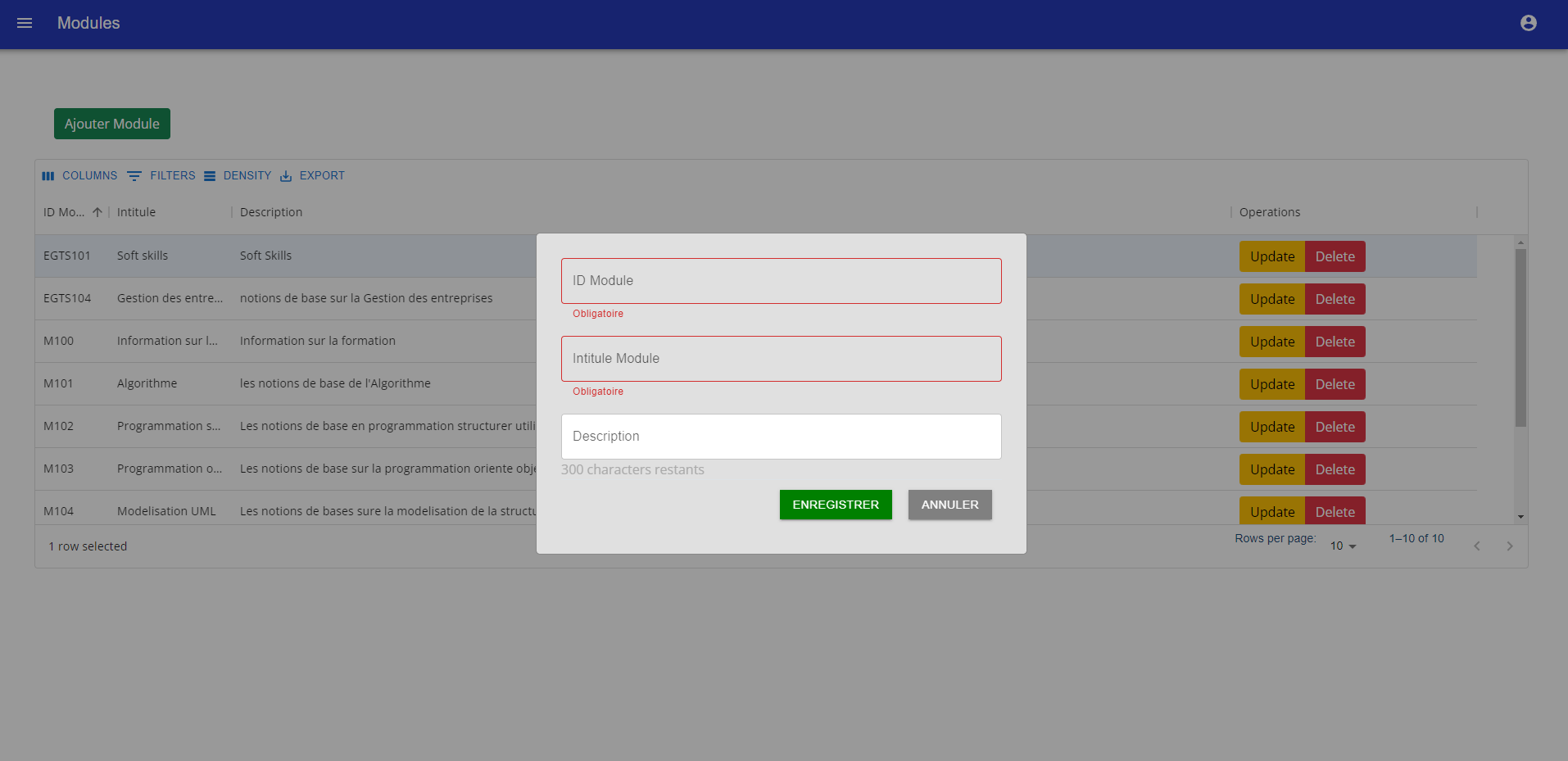
Description générée automatiquement

## Gestion des modules

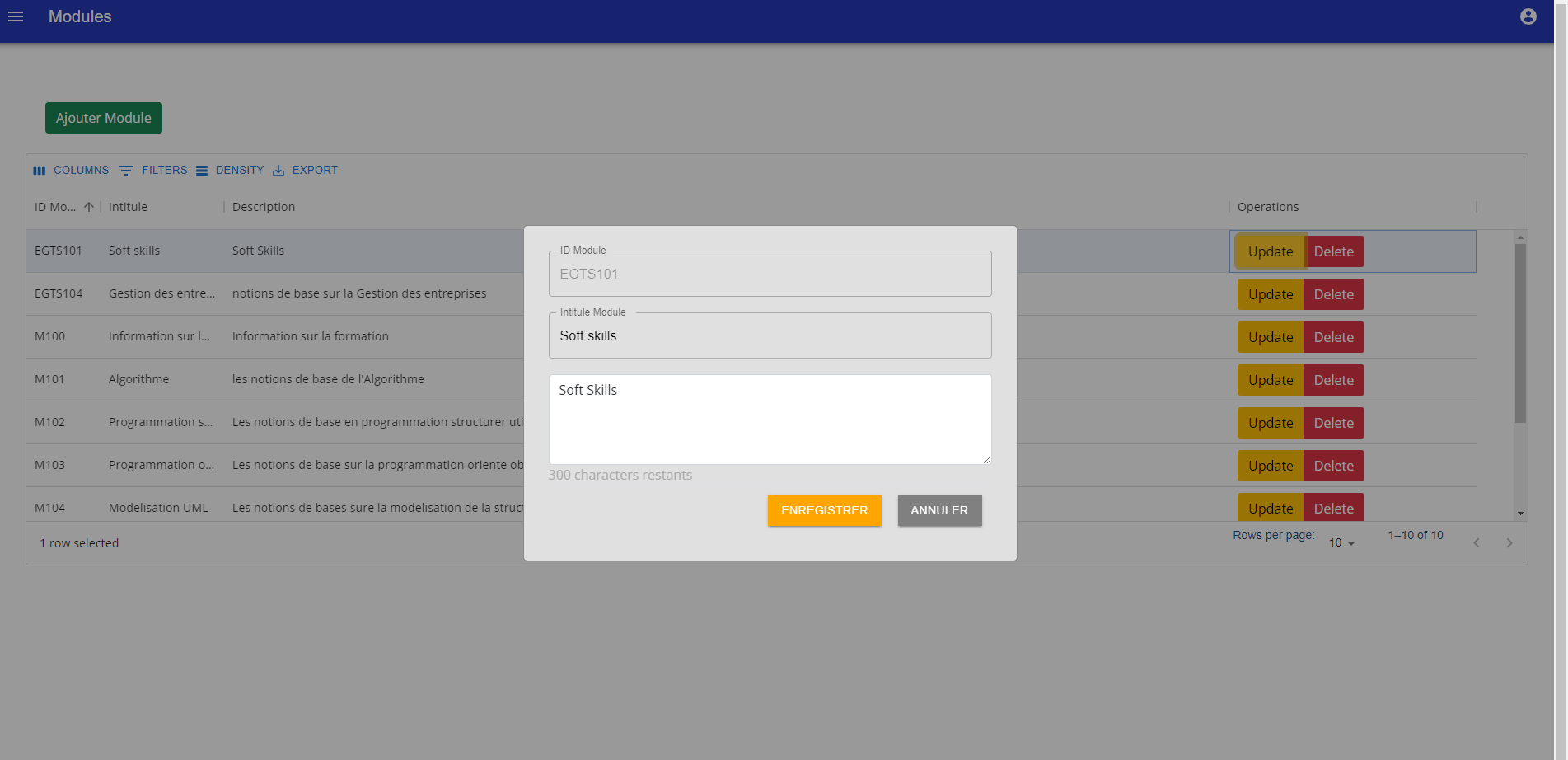
Cette page contient tous les opérations CRUD pour la gestion des modules :

Une image contenant texte

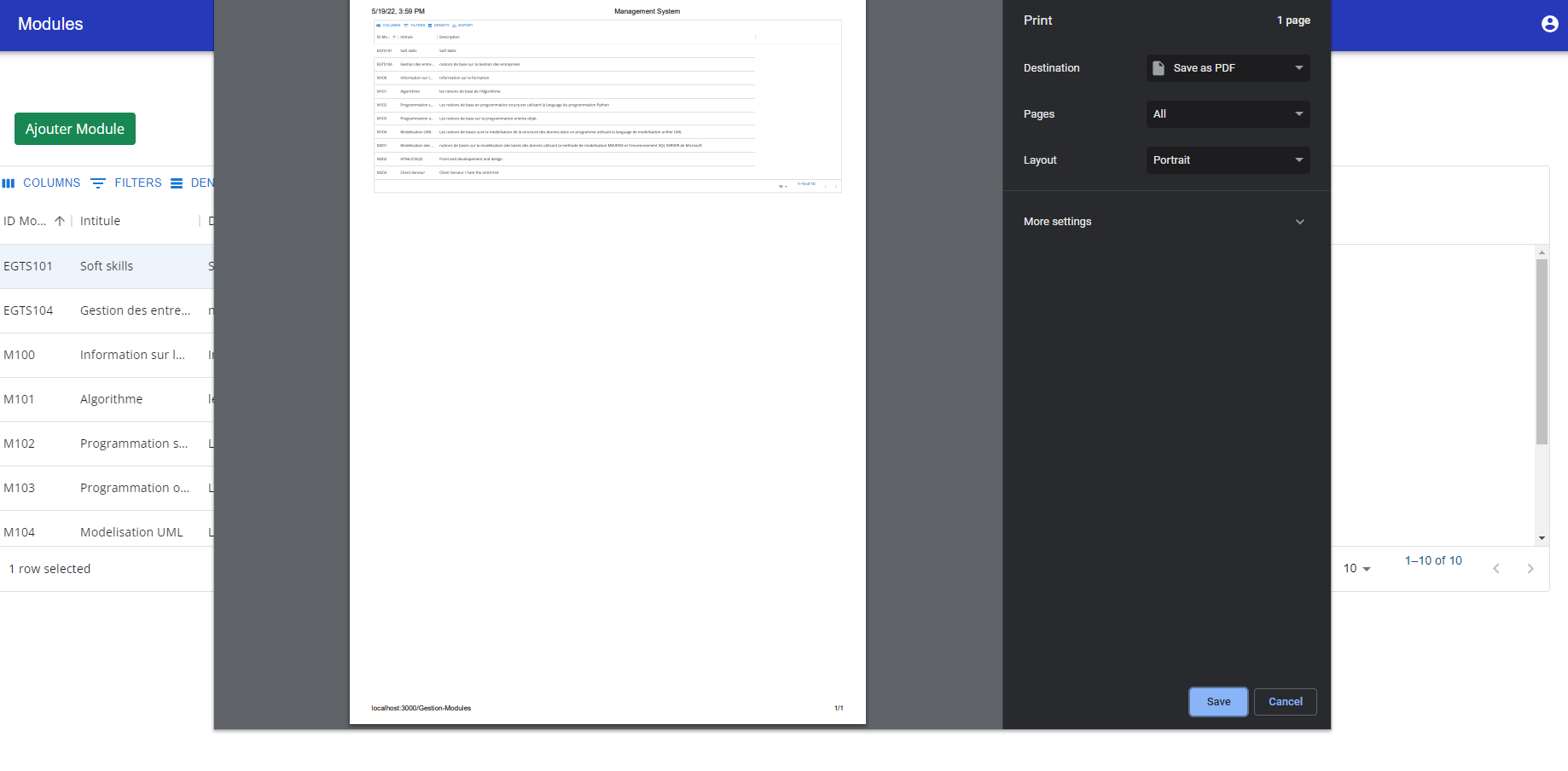
Description générée automatiquement

Ajouter : 

Modifier :



Imprimer :



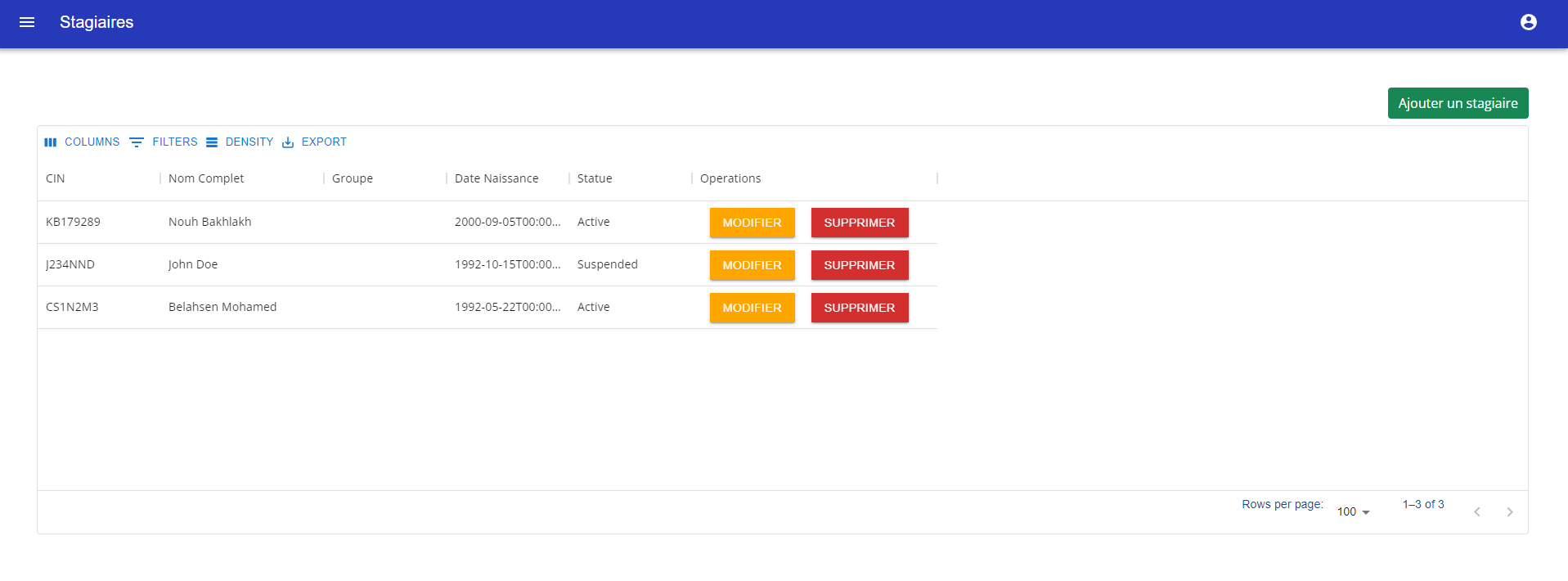
Rechercher :

Une image contenant texte

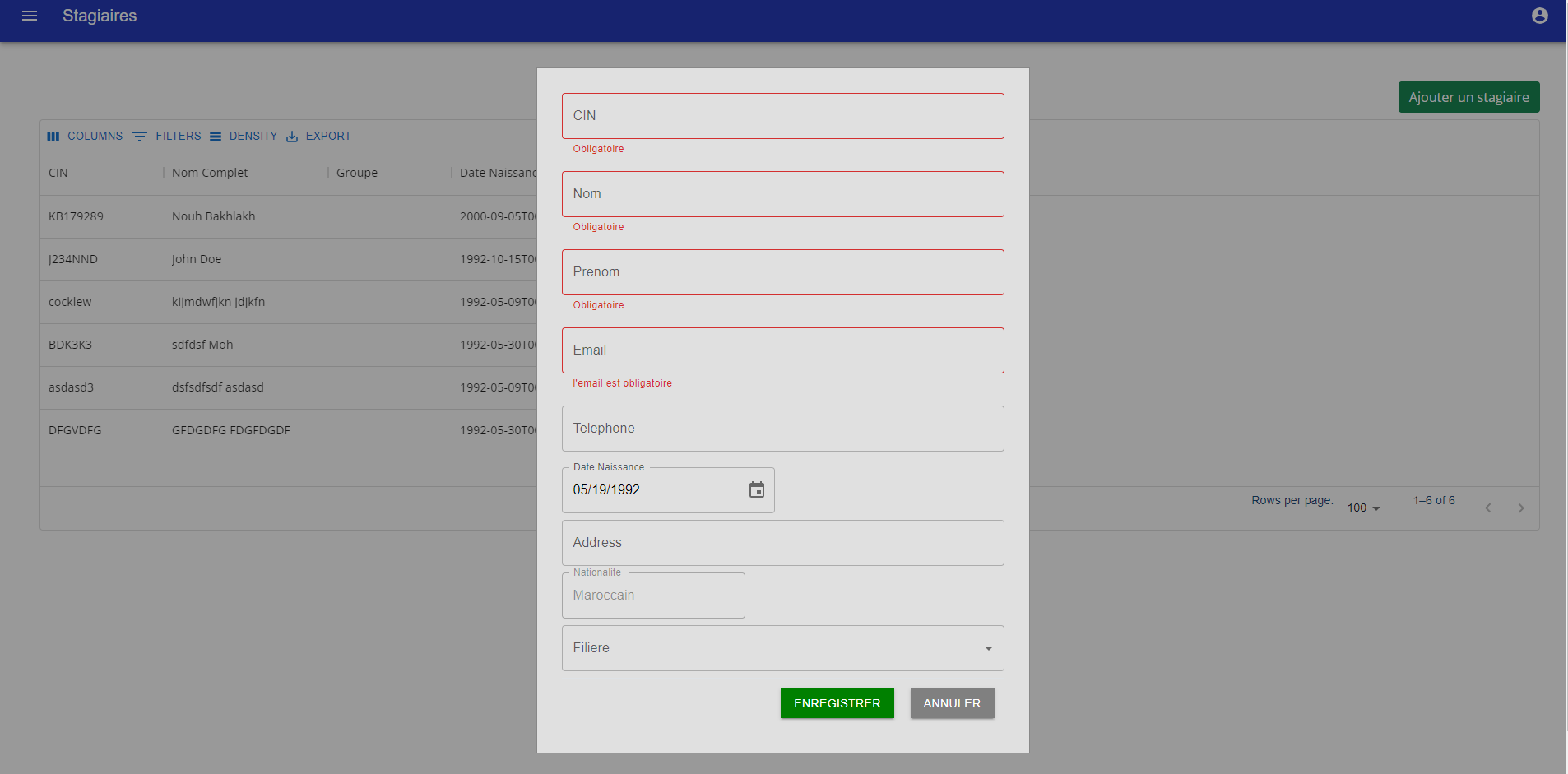
Description générée automatiquement

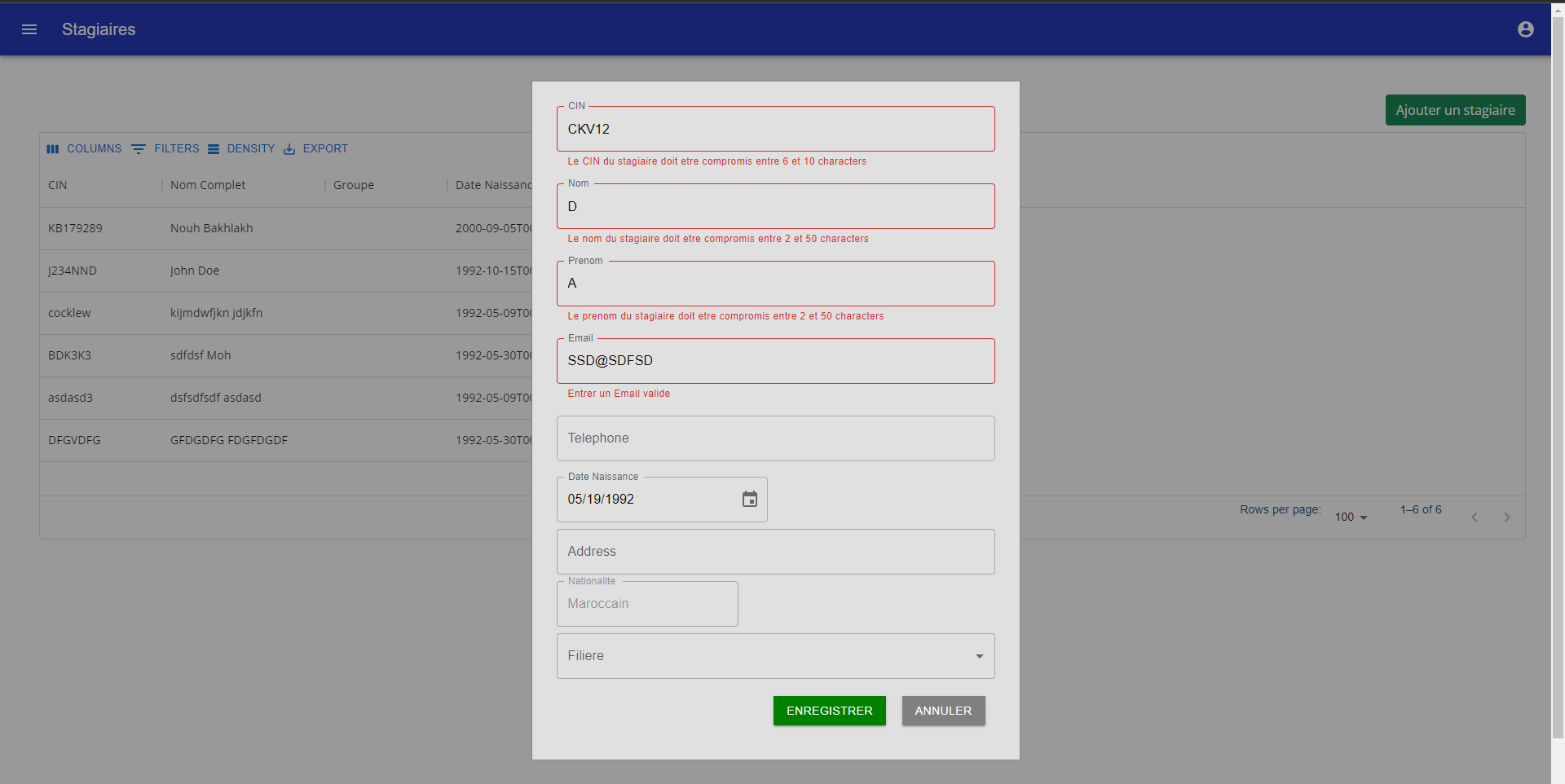
## Gestion des stagiaires

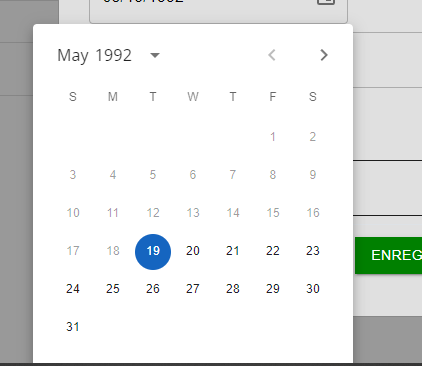
Cette page contient tous les opérations CRUD pour la gestion des stagiaires :



Ajouter :







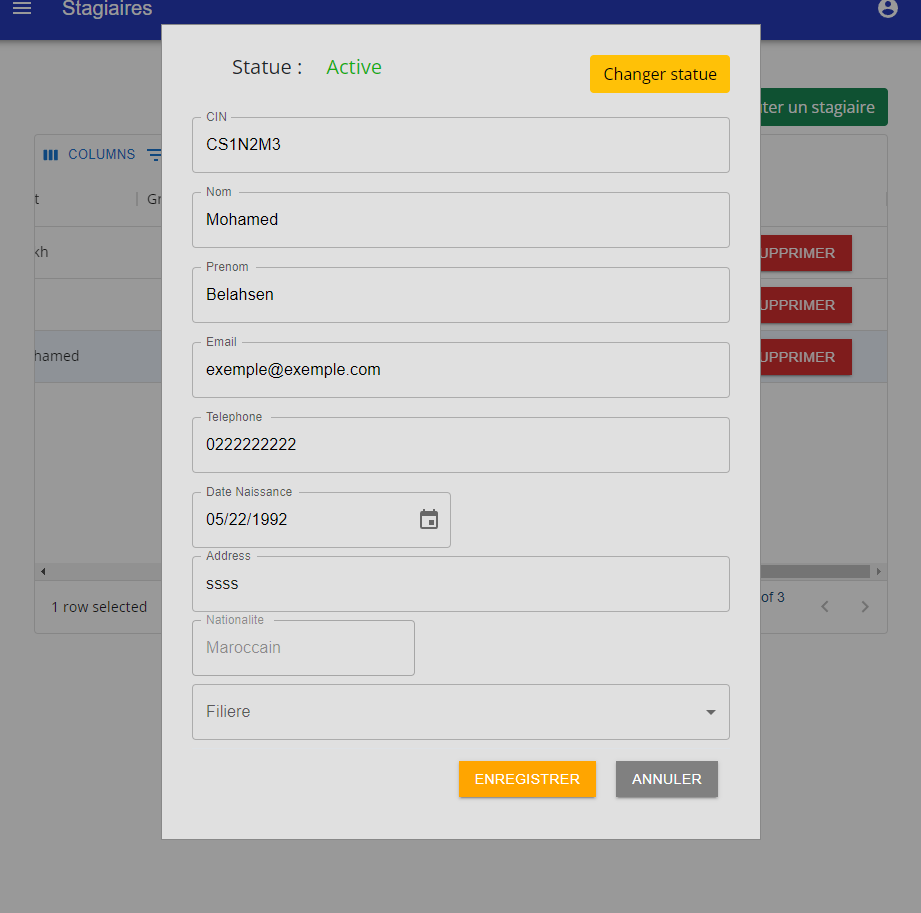
Une image contenant texte

Description générée automatiquement

Une image contenant texte

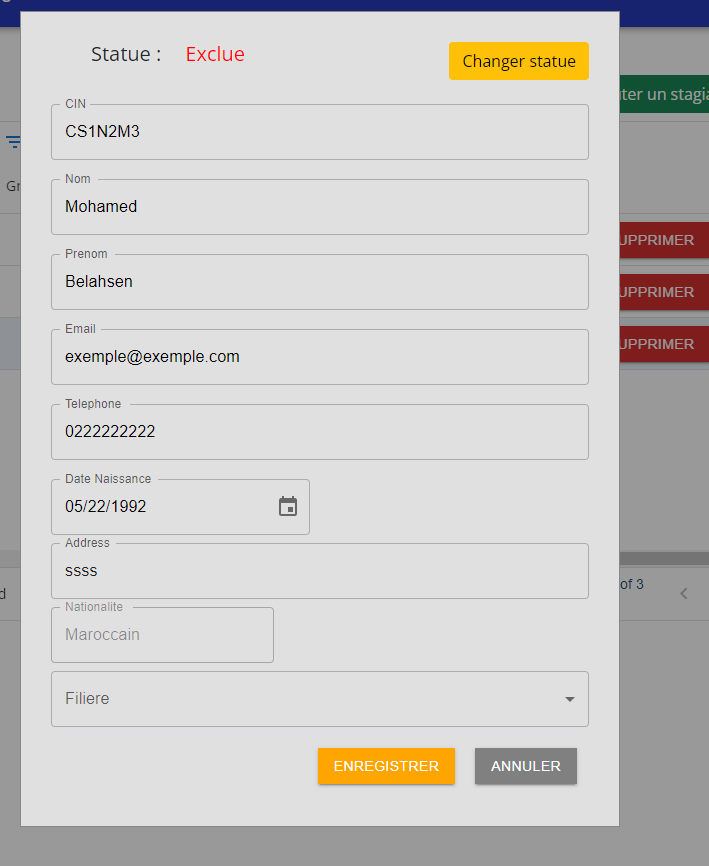
Description générée automatiquement

Modifier :



Une image contenant texte

Description générée automatiquement



Une image contenant texte

Description générée automatiquement

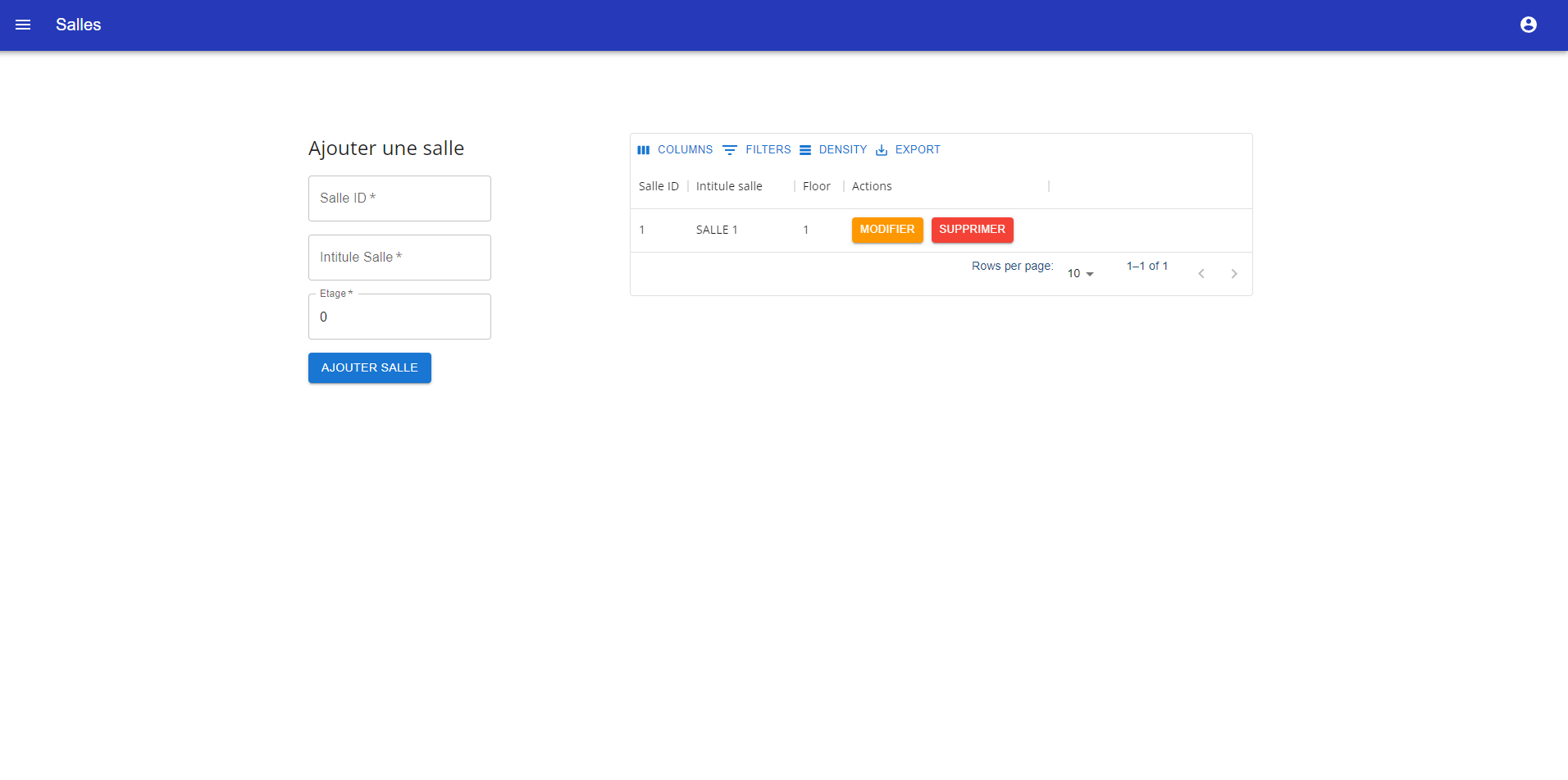
Les opérations du recherche, filtre et exportation on fait de la même manière que dans la gestion des modules

## Gestion des groupes

Une image contenant table

Description générée automatiquement

## Gestion des salles



## Gestion des séances

Une image contenant texte, table

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

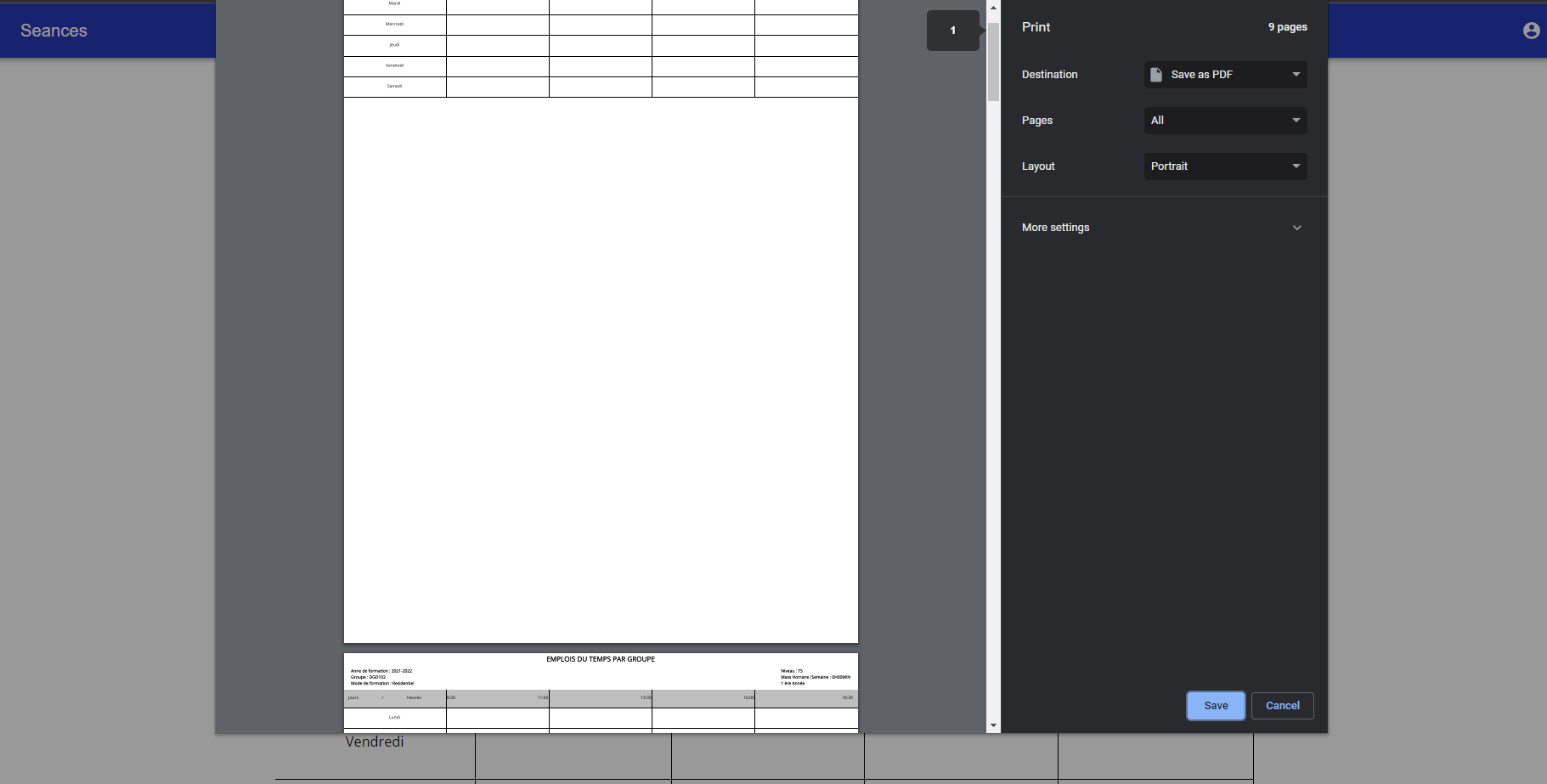


Une image contenant table

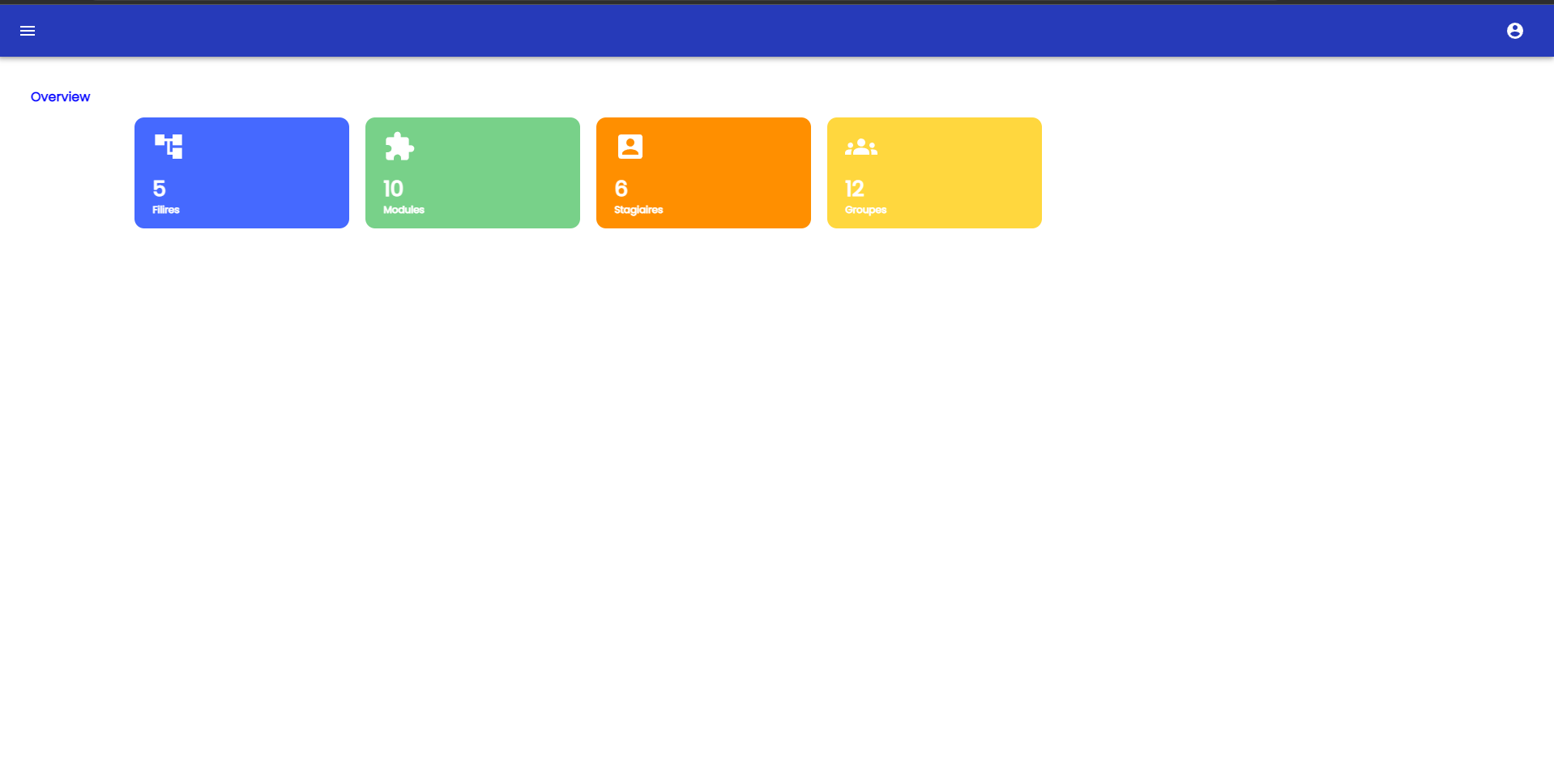
Description générée automatiquement

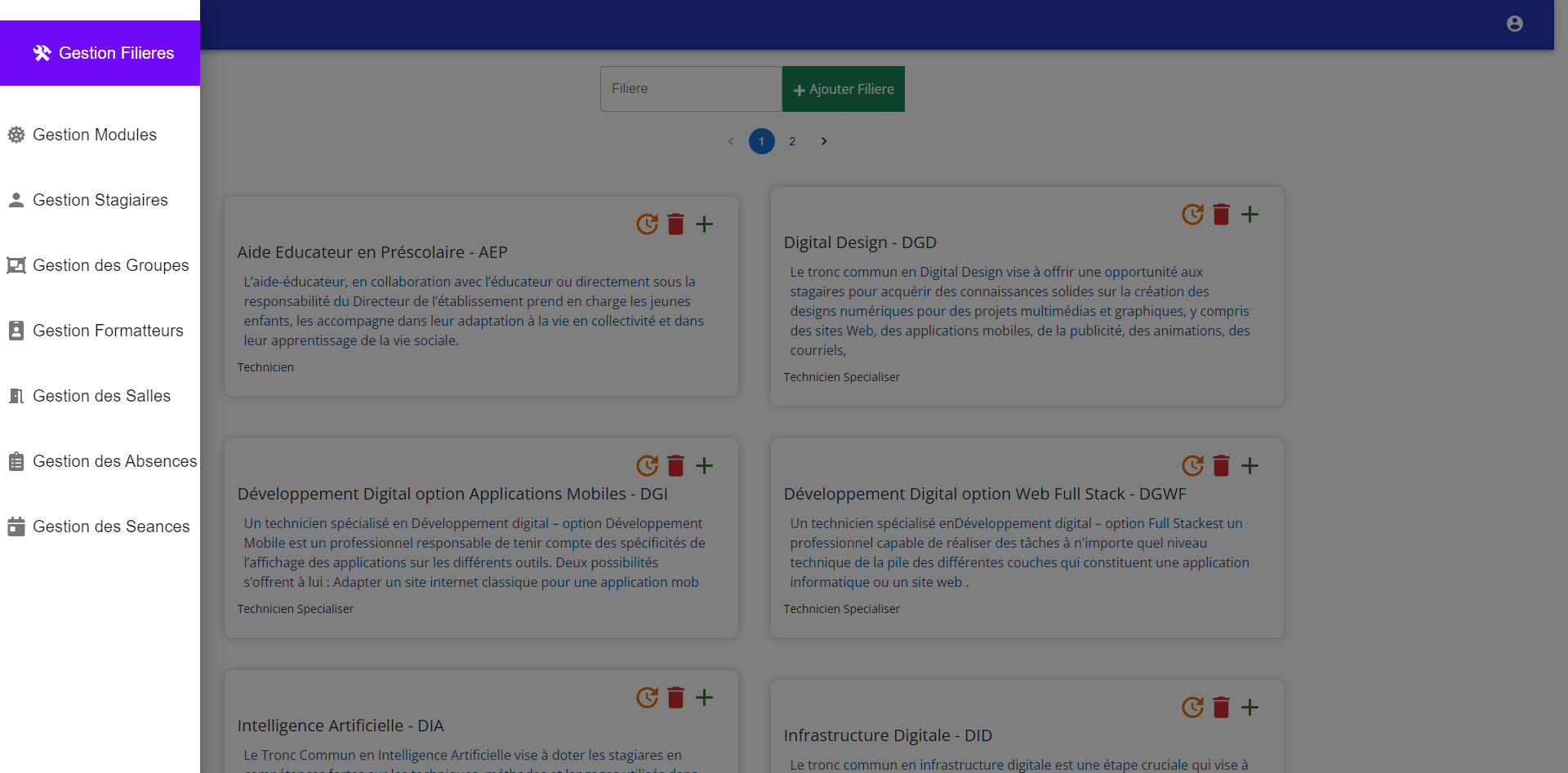
Une image contenant texte

Description générée automatiquement



## Page d’accueil





## Backend API Endpoint visualiser



Une image contenant texte

Description générée automatiquement