03/04/2017

# Énoncé

Projet Bases de données : Application de gestion d’un établissement scolaire

David HA, Charles ECOLLAN et Quentin LECAILLE  
Groupe A – Promo 2019 L’3



Sous la direction de M. RAMDANE-CHERIF

Ce programme de gestion d’une école devra être réalisé entièrement en Java ou tout autre langage en accord avec votre chargé de TP.

Vous êtes totalement libre des fonctions à implémenter.

# Résumé

Ce projet consiste à réaliser un programme de gestion d’un établissement scolaire. L’utilisateur se connecte en précisant s’il est étudiant, professeur ou administrateur (membre de l’administration). Lorsque l’utilisateur se connecte, en fonction de son rôle dans l’établissement (les trois catégories données précédemment), il voit un panel avec toutes les fonctionnalités disponibles. Pour donner un exemple, l’administrateur peut créer un élève, un enseignant et un administrateur tandis que les autres non. Le projet est une application web faites en HTML/CSS pour la forme et en PHP/MySQL pour le fond et l’accès à la base de données (phpMyAdmin).

Toutes les fonctions du cahier des charges ont été implémentées. On peut, par exemple, créer une matière et lui assigner un coefficient, mais aussi assigner des notes à des élèves par rapport à une épreuve de passée.

L’application se trouve à l’adresse suivante : http://www.charles-dev.com/ . Pour entrer dans l’application avec un identifiant administrateur :

* Identifiant : [charles.ecollan@outlook.com](mailto:charles.ecollan@outlook.com)
* Mot de passe : azerty (par défaut)

Elle peut être utilisée en local en important la base de données vide (il faut juste créer un identifiant admin par phpMyAdmin : *INSERT INTO `bdd`.`administrateur` (`nom\_admin`, `prenom\_admin`, `id\_admin`, `email\_admin`, `password`) VALUES ('Ecollan', 'Charles', '1', 'charles.ecollan@outlook.com', 'azerty');*

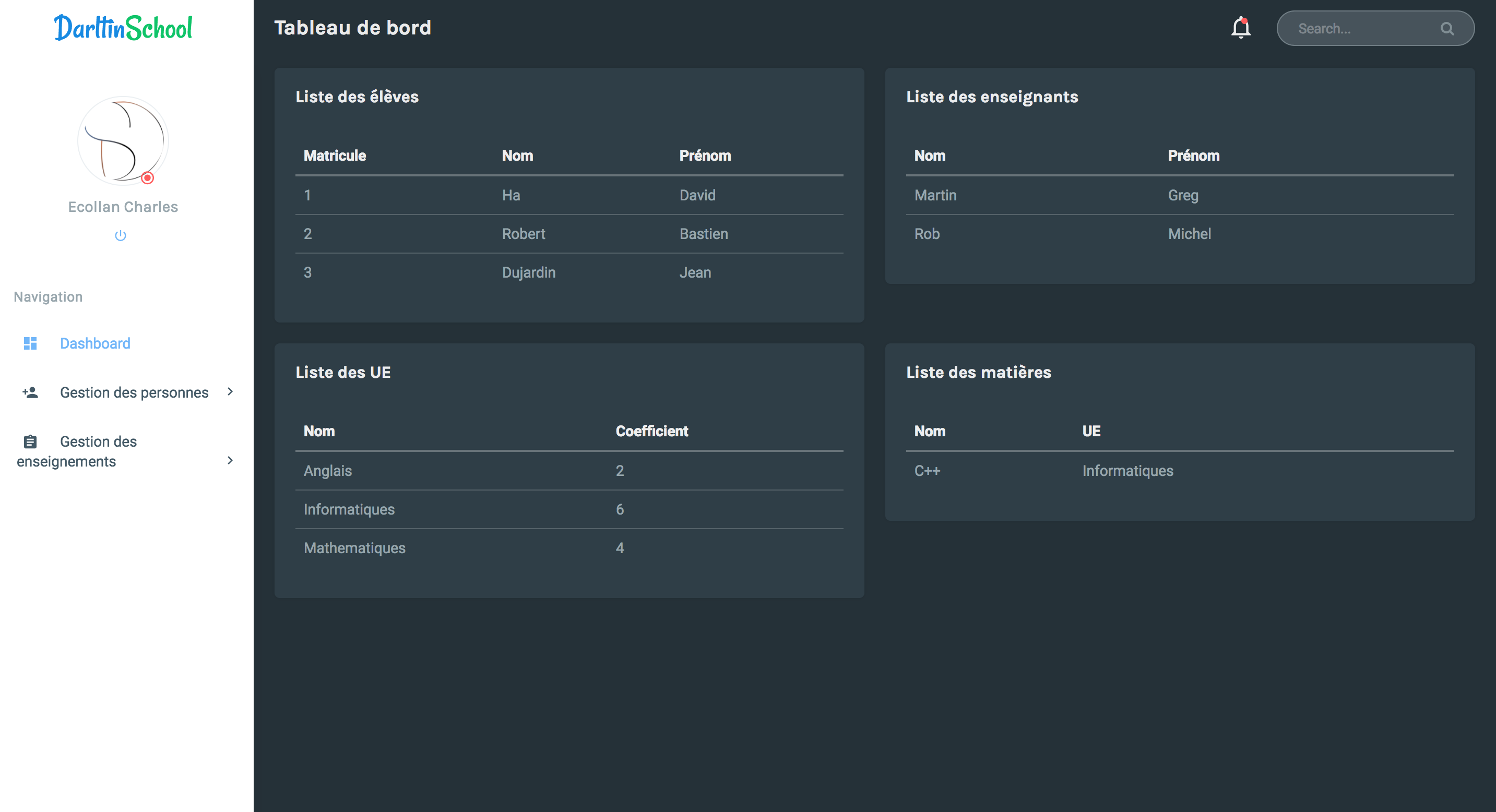
Table de matière

Table des matières

[Énoncé 0](#_Toc478993178)

[Résumé 2](#_Toc478993179)

[Introduction 4](#_Toc478993180)

[Analyse 5](#_Toc478993181)

[Cahier des charges 5](#_Toc478993182)

[Analyse de la base de données 5](#_Toc478993183)

[Dictionnaire de données 5](#_Toc478993184)

[MCD 7](#_Toc478993185)

[MLD 7](#_Toc478993186)

[Script SQL 7](#_Toc478993187)

[Langages 8](#_Toc478993188)

[HTML 8](#_Toc478993189)

[PHP 8](#_Toc478993190)

[MySQL 8](#_Toc478993191)

[CSS 9](#_Toc478993192)

[JavaScript 9](#_Toc478993193)

[Développement 10](#_Toc478993194)

[Plan de l’application 10](#_Toc478993195)

[Connexion 12](#_Toc478993196)

[Back-End 12](#_Toc478993197)

[Administrateur 13](#_Toc478993198)

[Gestion des personnes 13](#_Toc478993199)

[Gérer les enseignements 16](#_Toc478993200)

[Élèves 18](#_Toc478993201)

[Conclusion 20](#_Toc478993202)

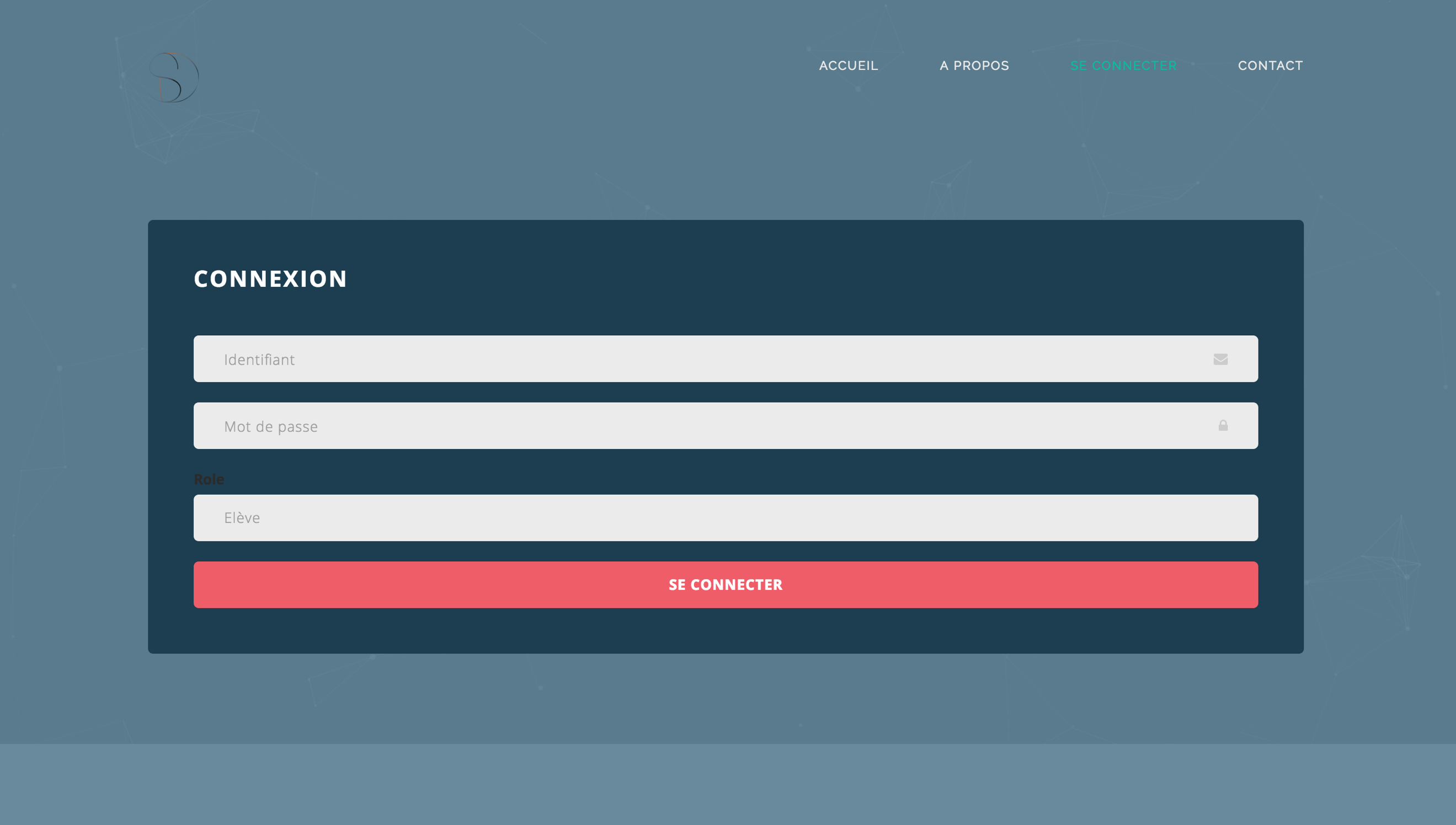
[Annexe 1 : Script SQL 21](#_Toc478993203)

# Introduction

Ce projet a pour but de gérer un établissement scolaire (étudiant, enseignant et administrateur). Il a permis de faire appel à tout ce que l’on a appris en Bases de données (1), c’est-à-dire les requêtes SQL, et en Bases de données (2), c’est-à-dire les MCD et MLD.

Le but était donc de réaliser l’interface graphique ainsi qu’implémenter les différentes fonctionnalités définies dans le cahier des charges.

Il a fallu réaliser ce projet dans le cadre des séances de TP encadrés mais aussi à l’extérieur. Afin de réaliser ce travail à trois, nous avons utilisé *git* pour permettre de travailler chacun sur une partie et de l’envoyer directement par l’intermédiaire de ce logiciel. De plus, étant un logiciel de gestion de versions, si un problème apparaissait, on pouvait revenir sur une phase précédente sans soucis.



# Analyse

## Cahier des charges

L’application doit permettre de gérer un établissement scolaire. Nous avons donc pris le modèle de l’EFREI où l’administration à accès à la totalité des fonctionnalités et entre les notes des élèves. L’application doit pouvoir gérer :

* Les étudiants

Consulter et modifier son dossier.

Consulter ses notes.

* Les administrateurs

Créer des élèves et des enseignants.

Créer des matières, des UE et des promotions.

Ajouter des notes à des élèves.  
 Créer des épreuves.

Assigner un UE à une promotion.

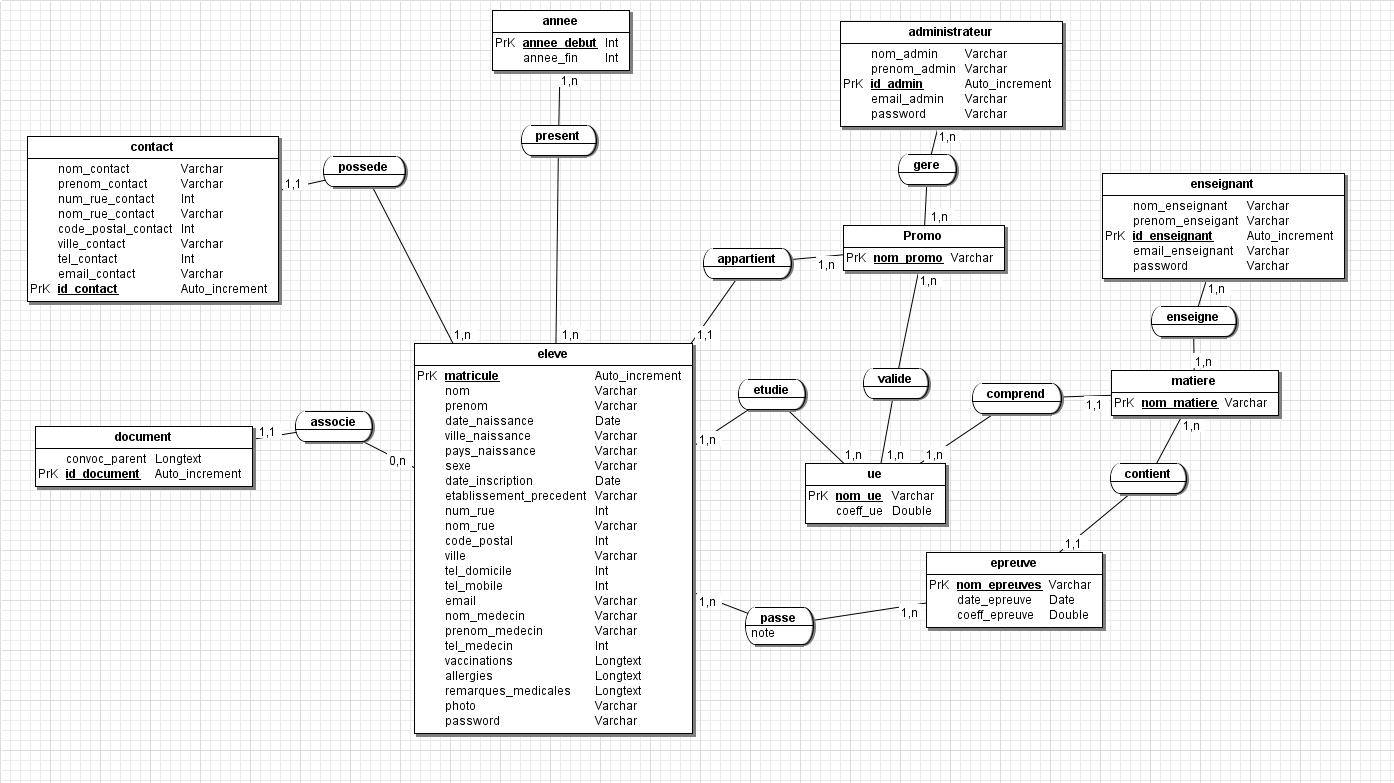
Assigner une matière à un professeur.

## Analyse de la base de données

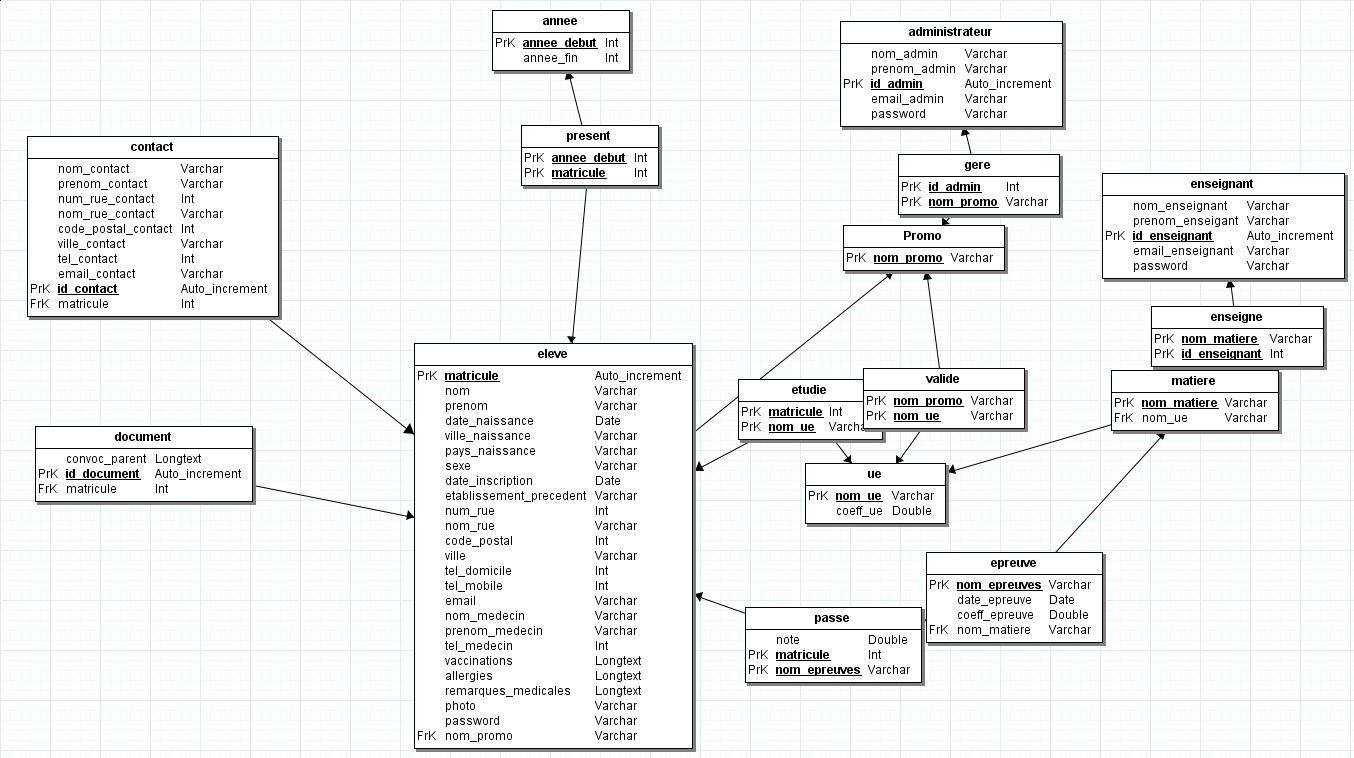
### Dictionnaire de données

|  |  |  |
| --- | --- | --- |
| **Sigle** | **Libellé** | **Type** |
| Nom\_contact | Nom | Varchar |
| Prenom\_contact |  | Varchar |
| Num\_rue\_contact |  | Int |
| Nom\_rue\_contact |  | Varchar |
| Code\_postal\_contact |  | Int |
| Ville\_contact |  | Varchar |
| Tel\_contact |  | Int |
| Email\_contact |  | Varchar |
| Id\_contact |  | Int |
| Convoc\_parent |  | Longtext |
| Id\_document |  | Int |
| Annee\_debut |  | Int |
| Annee\_fin |  | Int |
| Matricule |  | Int |
| Nom |  | Varchar |
| Prenom |  | Varchar |
| Date\_naissance |  | Date |
| Ville\_naissance |  | Varchar |
| Pays\_naissance |  | Varchar |
| Sexe |  | Varchar |
| Date\_inscription |  | Date |
| Etablissement\_precedent |  | Varchar |
| Num\_rue |  | Int |
| Nom\_rue |  | Varchar |
| Code\_postal |  | Int |
| Ville |  | Varchar |
| Tel\_domicile |  | Int |
| Tel\_mobile |  | Int |
| Email |  | Varchar |
| Nom\_medecin |  | Varchar |
| Prenom\_medecin |  | Varchar |
| Tel\_medecin |  | Int |
| Vaccinations |  | Longtext |
| Allergies |  | Longtext |
| Remarques\_medicales |  | Longtext |
| Photo |  | Varchar |
| Password |  | Varchar |
| Note |  | Int |
| Nom\_admin |  | Varchar |
| Prenom\_admin |  | Varchar |
| Id\_admin |  | Int |
| Email\_admin |  | Varchar |
| Password |  | Varchar |
| Nom\_promo |  | Varchar |
| Nom\_ue |  | Varchar |
| Coeff\_ue |  | Double |
| Nom\_epreuves |  | Varchar |
| Date\_epreuve |  | Date |
| Coeff\_epreuve |  | Double |
| Nom\_matiere |  | Varchar |
| Nom\_enseignant |  | Varchar |
| Prenom\_enseignant |  | Varchar |
| Id\_enseignant |  | Int |
| Email\_enseignant |  | Varchar |
| Password |  | Varchar |

### MCD



### MLD



### Script SQL

Voir annexe 1

## Langages

Ci-dessous, la liste des langages utilisés dans un tableau avec les principales utilisations afin d’avoir une vue plus générale.

|  |  |
| --- | --- |
| Langage | Utilisation(s) |
| HTML | L’affichage des différentes pages du site web. |
| CSS | La présentation des différentes pages du site. |
| JavaScript | Contrôle des champs des formulaires. |
| PHP | Interaction avec le serveur. Permet le développement des pages dynamiques (relation avec la base de données) |
| MySQL | Interroger la base de données |

### 

### HTML

Le HTML, qui signifie Hypertext Markup Language, est un langage de balisage pour l’écriture de l’hypertexte. Il permet de créer un site dit « statique ». On entend par page statique, non pas une page sans mouvement ou sans animations, mais une page visible telle qu’elle a été conçue. Ces pages peuvent présenter toute forme de contenu, animations flash, images, musiques ou vidéos, mais elles sont toujours présentées de la même façon. Elles ne changent pas et c’est en ce sens qu’elles sont statiques. Il faut différencier le statique du dynamique. Effectivement, une page dynamique est une page qui s’affiche en fonction des requêtes des utilisateurs, telles que les zones de saisies et les listes déroulantes.

### PHP

Le PHP, Hypertext Preprocessor, est un langage de programmation permettant de faire un site dynamique, c’est-à-dire en affichant des données, par exemple, venant d’une base de données. Cela permet de mettre un lien entre le serveur et le client. Par exemple, la page de connexion pour sécuriser l’accès à l’application est faite en PHP.

### MySQL

MySQL, qui signifie My Structured Query Language, est un serveur de bases de données relationnelles SQL. Ici, on l’a couplé au PHP afin d’interroger, d’insérer et de mettre à jour la base de données.

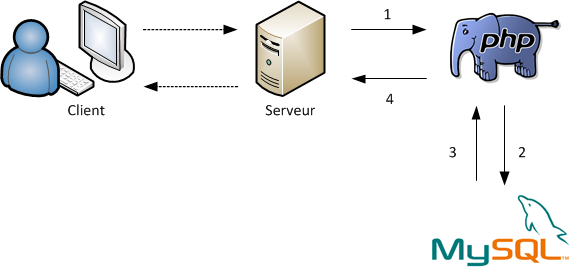


Figure 2 : Schéma récapitulatif HTML-PHP-MySQL

### CSS

Le CSS, signifiant Cascading Style Sheets ou en français Feuilles de Style en Cascade, permet tout simplement de mettre en page les pages de l’application.

### JavaScript

Le JavaScript est un langage de programmation permettant de faire des événements du côté du client.

# Développement

## Plan de l’application

Est établi un plan des différentes pages du site ci-dessous à créer pour se rendre compte du placement de telles et telles fonctions, mais aussi pour adopter une stratégie de conception claire et nette. Ce plan concerne la vue de l’administrateur.

Les grands points sont présentés ci-dessus. Au travers de ce diagramme, l’utilisateur va devoir s’authentifier avant de pouvoir accéder à l’application. Il se retrouve alors sur l’accueil et a le choix avec deux grandes possibilités distinctes :

* Gérer les personnes

Gérer les personnes, c’est-à-dire créer des étudiants et des enseignants. Mais aussi créer une promotion et ajouter une note à un élèves en fonction d’une épreuve donnée.

* Gérer les enseignements

Gérer les enseignements, c’est-à-dire créer un UE et une matière. Les assigner entre eux. Mais aussi assigner une matière à un professeur et ajouter une épreuve afin de pouvoir mettre des notes aux élèves.

* Se déconnecter

Quitter l’application.

Voyons maintenant un plan concernant la vue de l’étudiant.

Les grands points sont présentés ci-dessus. Au travers de ce diagramme, l’utilisateur va devoir s’authentifier avant de pouvoir accéder à l’application. Il se retrouve alors sur l’accueil et a le choix avec quatre grandes possibilités distinctes :

* Vos matières

Accéder à toutes les matières que l’étudiant a.

* Votre relevé de notes

Permet de connaître les notes que l’étudiant a.

* Modifier vos informations personnelles

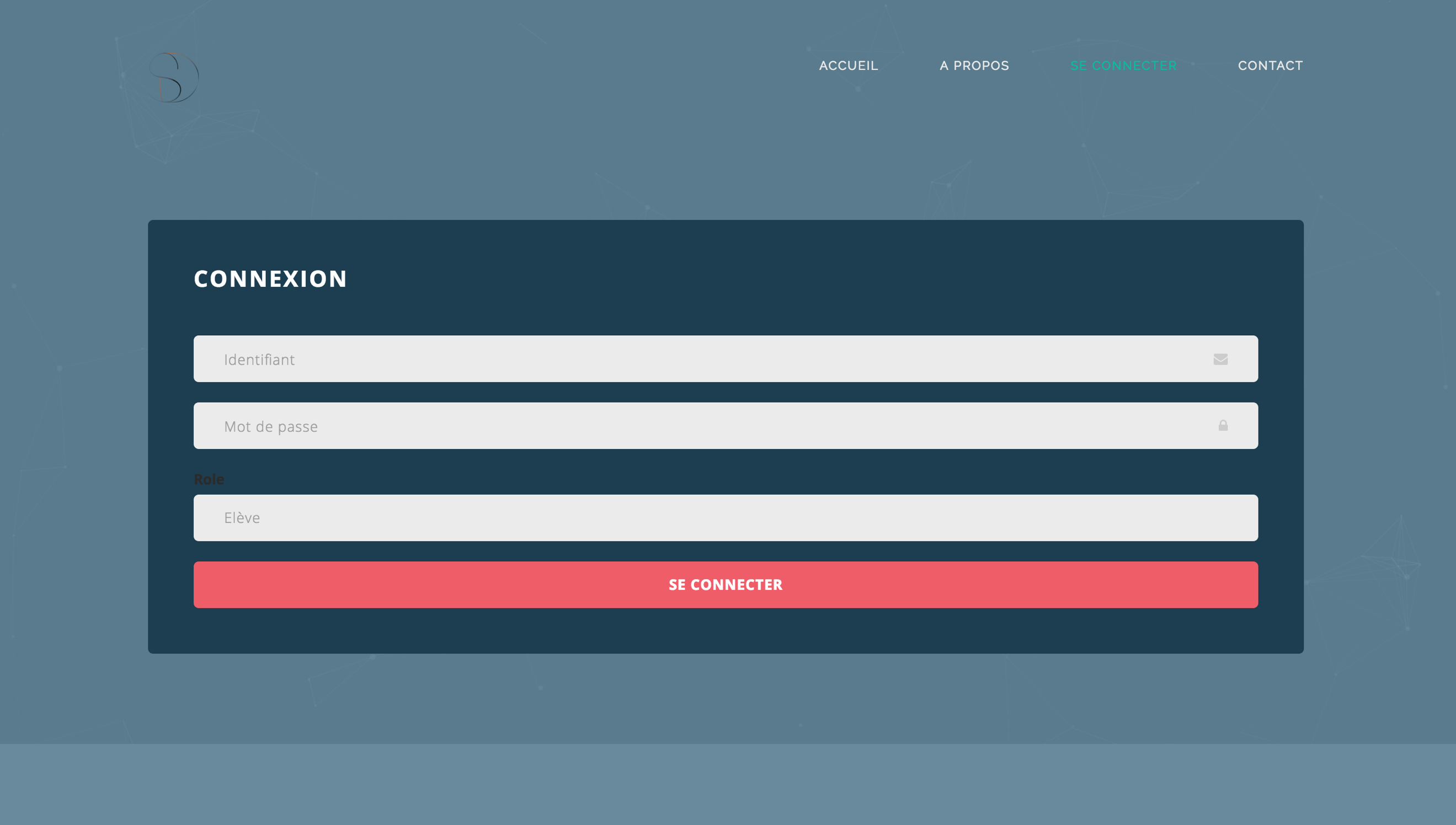
Modifier le dossier de l’élèves (son dossier du coup, comme les informations personnelles afin de les mettre à jour).

* Se déconnecter

Quitter l’application.

## Connexion

La page de connexion est à part En effet, c’est elle qui va permettre à l’utilisateur de rentrer dans l’application. Il faut d’abord créer la vue de la page ainsi que le formulaire de connexion afin de pouvoir envoyer les données au contrôleur qui va lui-même appeler le modèle pour interroger la base de données. Si l’utilisateur existe et que le mot de passe rentré est correct, alors on redirige l’utilisateur vers la vue de l’accueil de l’application correspondant à son profil : étudiant, enseignant ou administrateur. Sinon, on affiche un message d’erreur.



## Back-End

Nous avons maintenant un site dit statique, c’est-à-dire qui n’affiche que ce que l’on a écrit dans les différentes vues. Afin de rendre le site dynamique et de mettre un lien avec la base de données, nous allons passer à un site dynamique en utilisant trois choses :

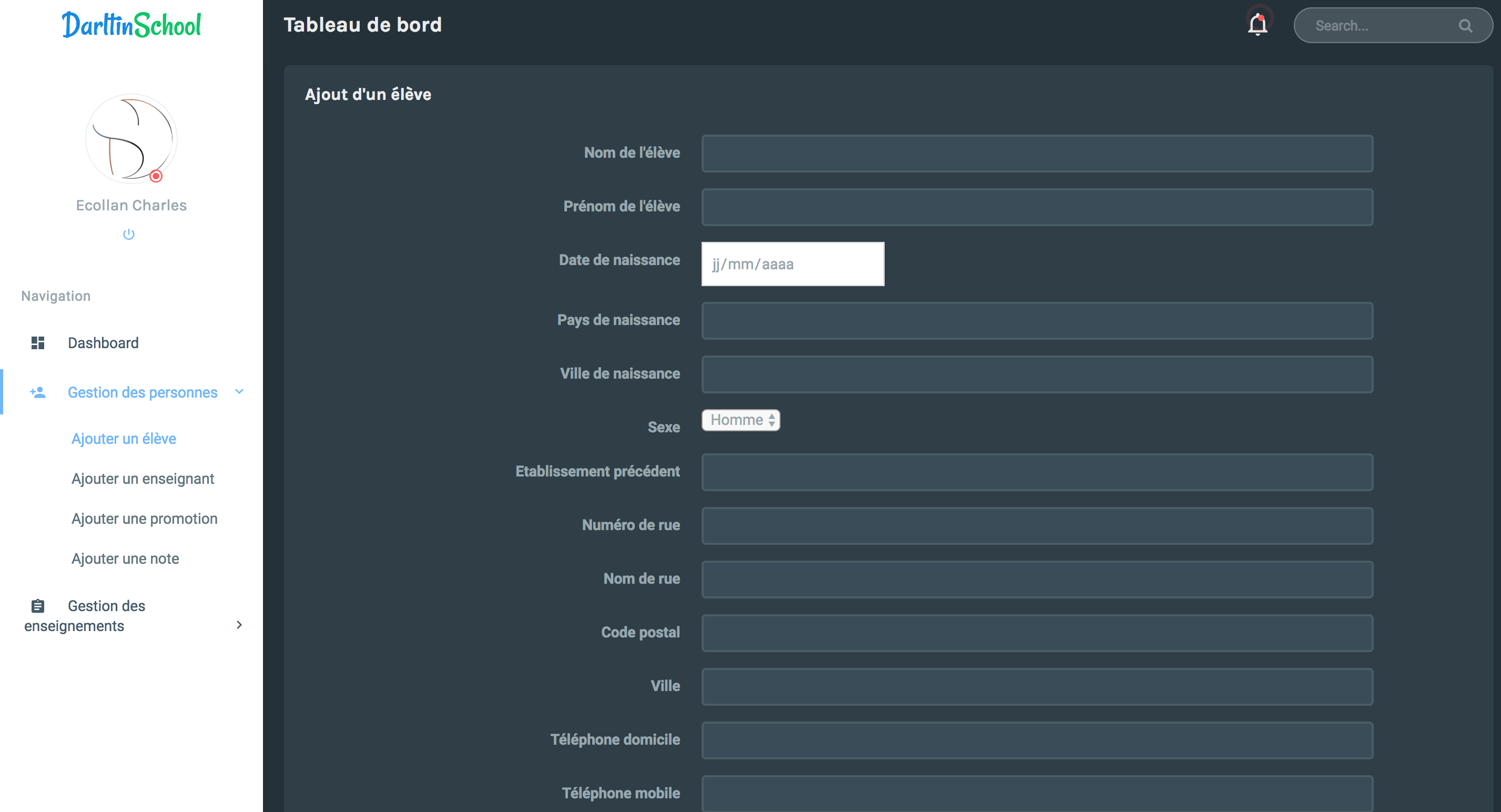
* Le serveur. Dans notre cas, l’application est hébergée sur un serveur chez OVH. Toutes les pages y sont stockées ainsi que la base de données.
* Le site web. On parle ici de la partie cachée (requêtes…)
* La base de données. Elle stocke toutes les personnes (administrateurs, enseignants et élèves), les notes, les matières…

Les langages utilisés sont donc le PHP et le MySQL.

## Administrateur

### Gestion des personnes

#### Ajouter une élève

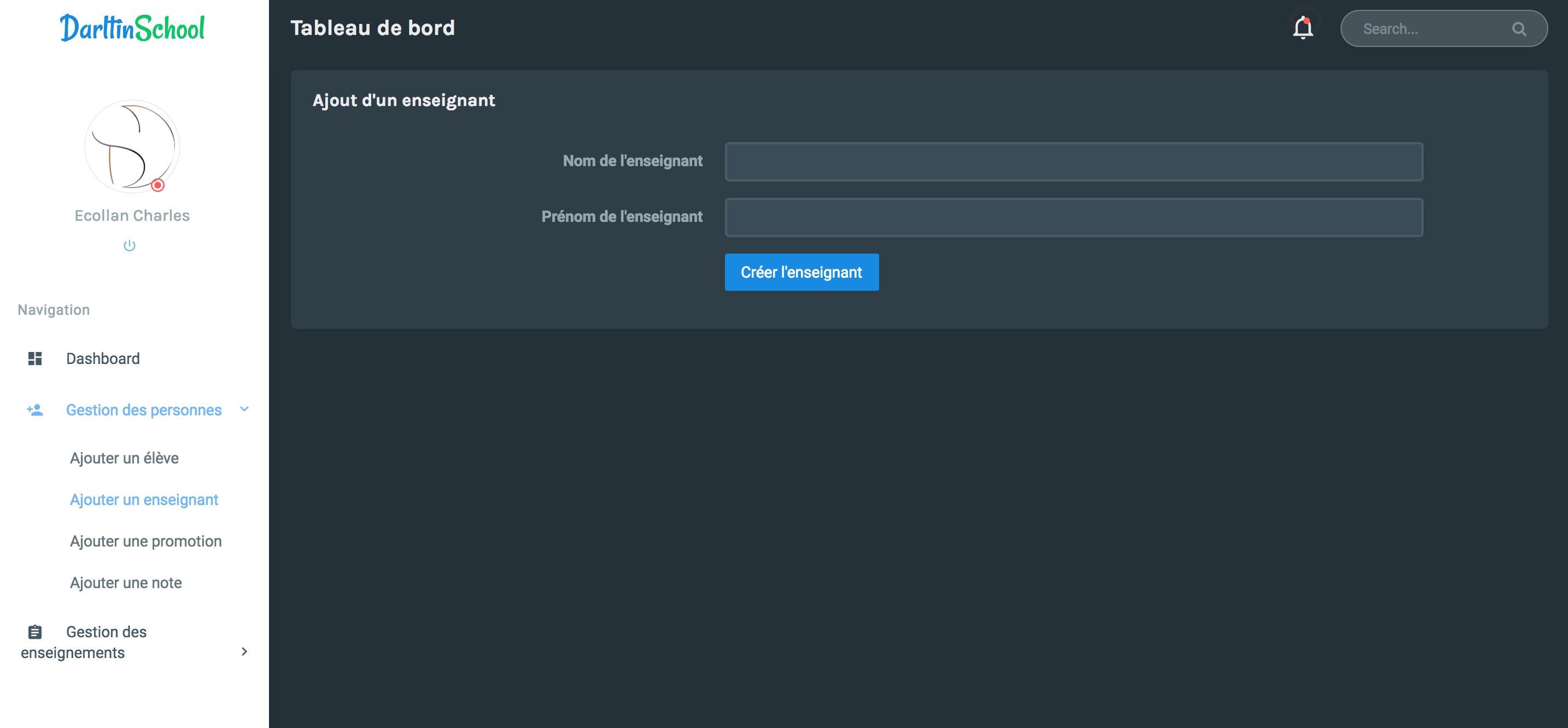


Pour ajouter un élève, il faut remplir les informations suivantes :

* Nom de l’élève
* Prénom de l’élève
* Date de naissance
* Pays de naissance
* Ville de naissance
* Sexe
* Établissement précèdent
* Numéro de rue
* Nom de rue
* Code postal
* Ville
* Téléphone domicile
* Téléphone mobile
* Nom du médecin
* Prénom du médecin
* Téléphone du médecin
* Allergie(s)
* Vaccination(s)
* Remarque(s) médicale(s)
* Promotion (choix menu si promotion créée)

L’email est automatiquement créé à la création de l’élèves : [prenom.nom@darltin.com](mailto:prenom.nom@darltin.com) ainsi que le mot de passe : azerty.

#### Ajouter un enseignant

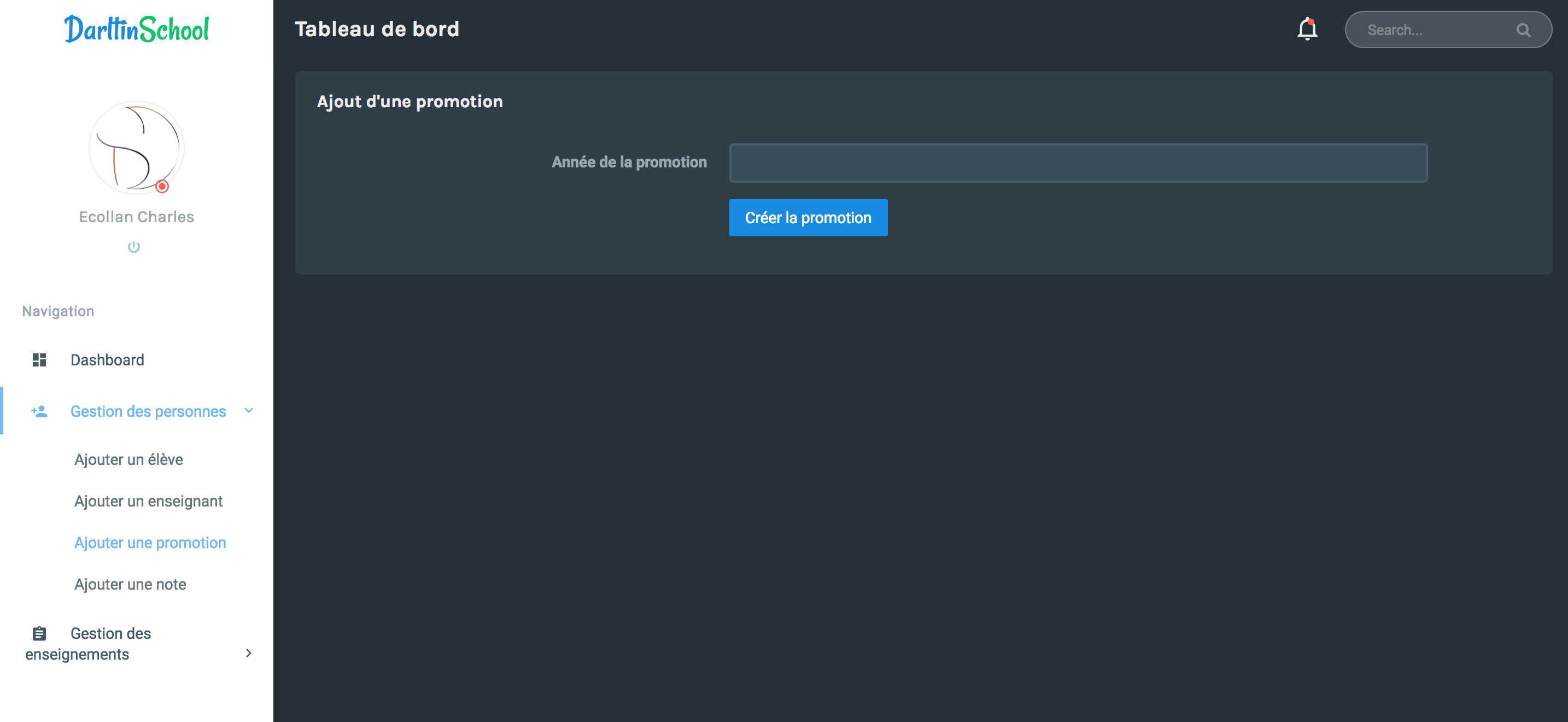


Pour ajouter un enseignant, il faut remplir les informations suivantes :

* Nom de l’enseignant
* Prénom de l’enseignant

L’email est automatiquement créé : [prenom.nom@darltin.com](mailto:prenom.nom@darltin.com) ainsi que le mot de passe : azerty.

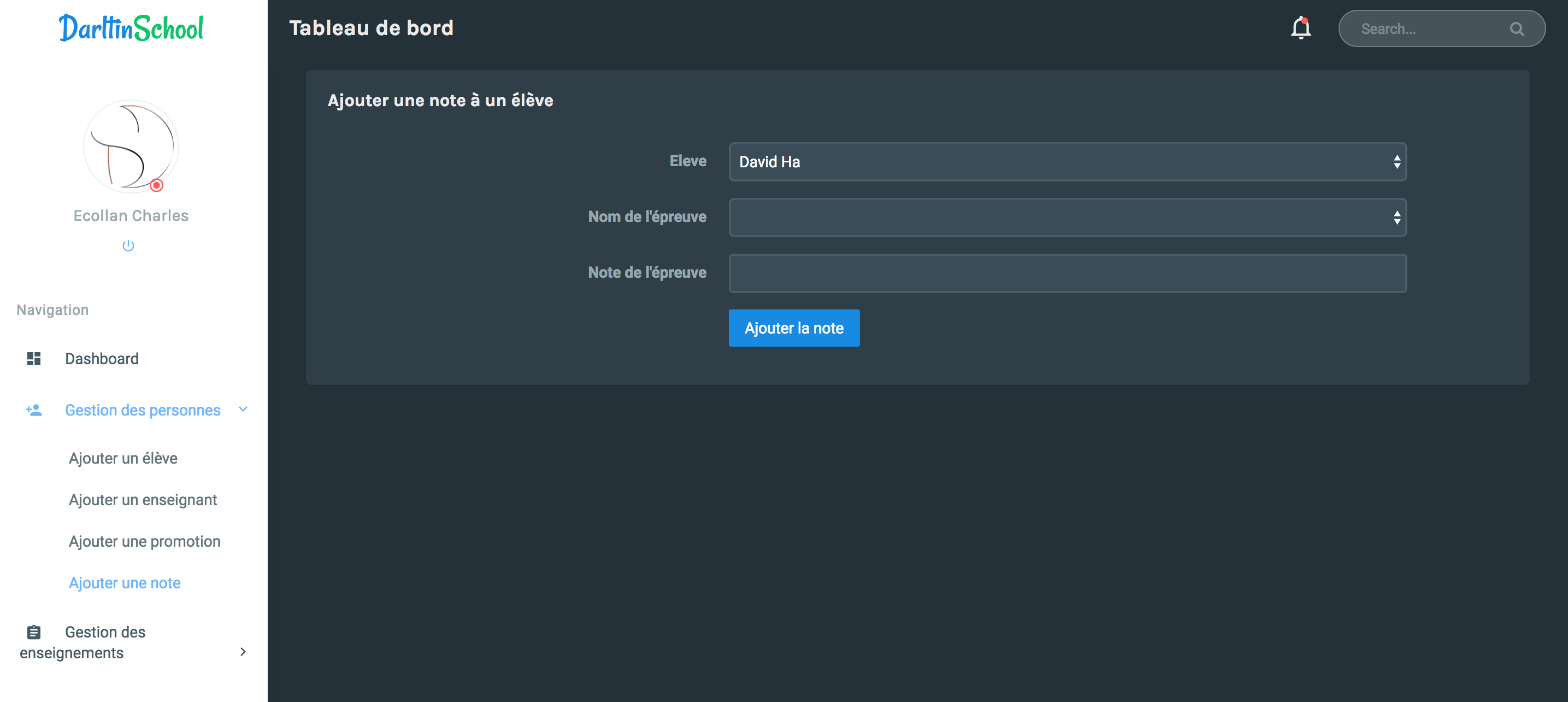
#### Ajouter une promotion



Pour ajouter une promotion, il faut remplir les informations suivantes :

* Année de la promotion

#### Ajouter une note



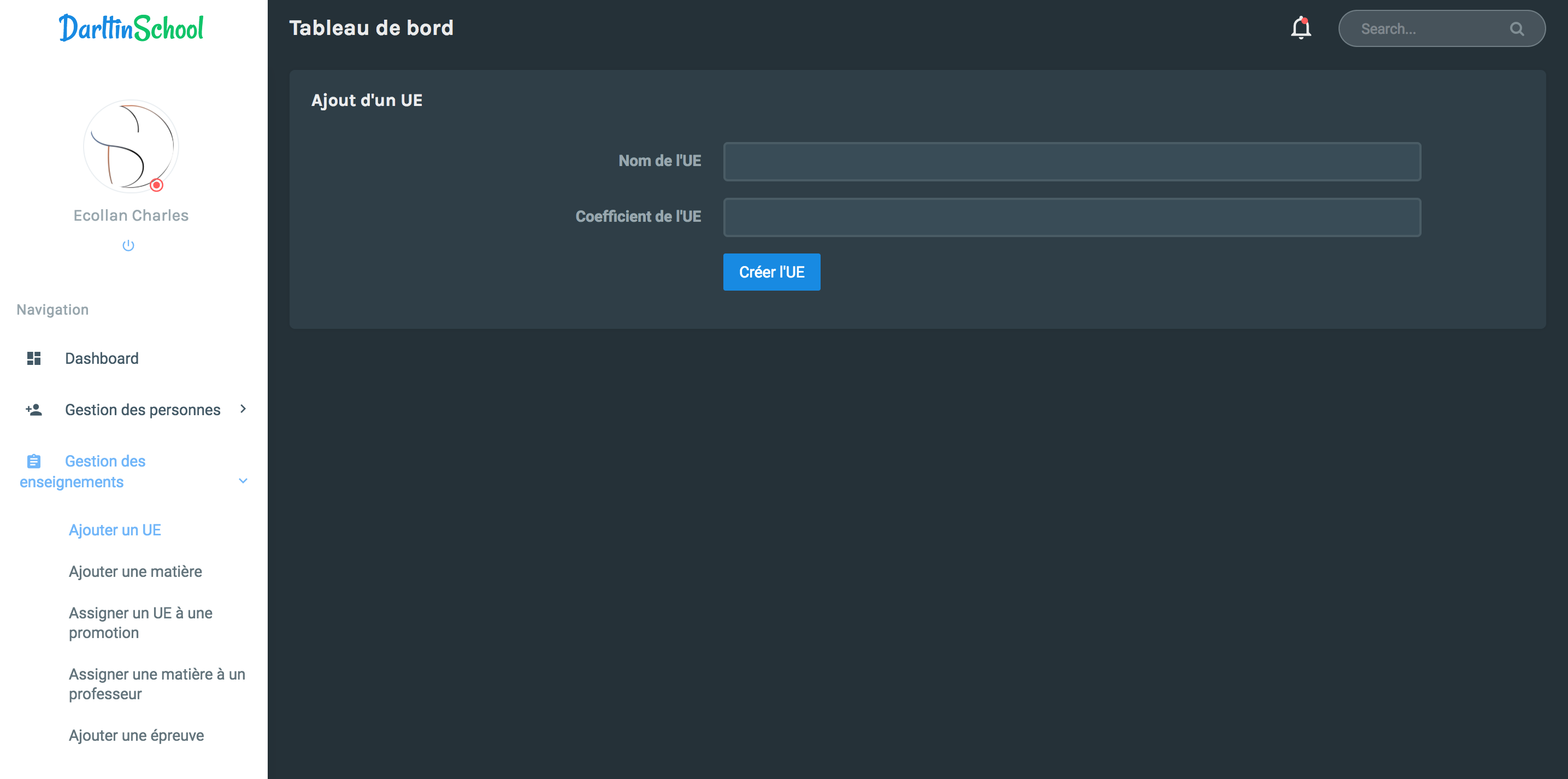
Pour ajouter une note, il faut remplir les informations suivantes :

* Nom de l’élève
* Nom de l’épreuve
* Note de l’épreuve

Pour ajouter une note à un élève, il faut auparavant ajouter une épreuve (expliqué un peu plus loin).

### Gérer les enseignements

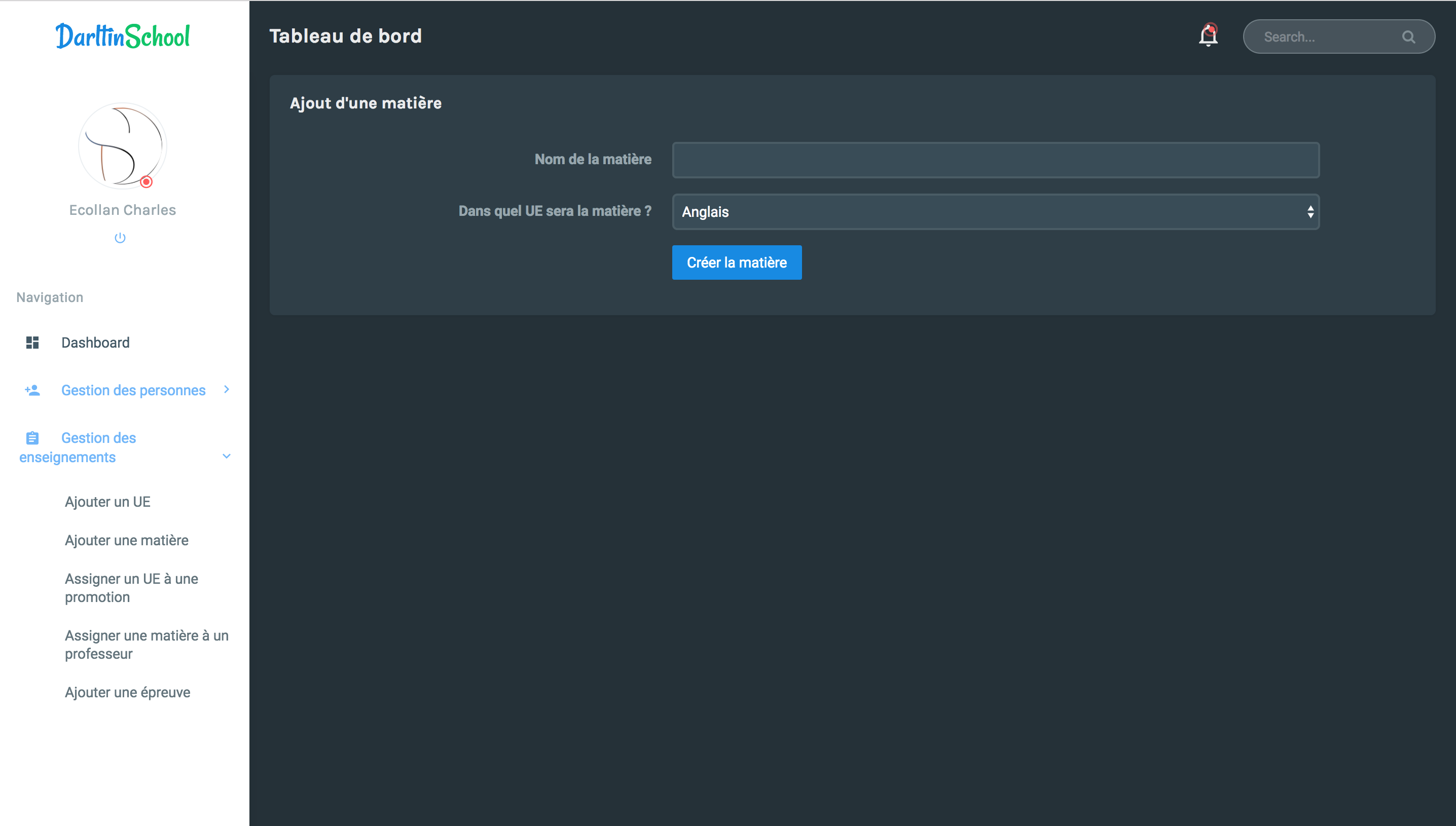
#### Ajouter un UE



Pour ajouter un UE, il faut remplir les informations suivantes :

* Nom de l’UE
* Coefficient de l’UE

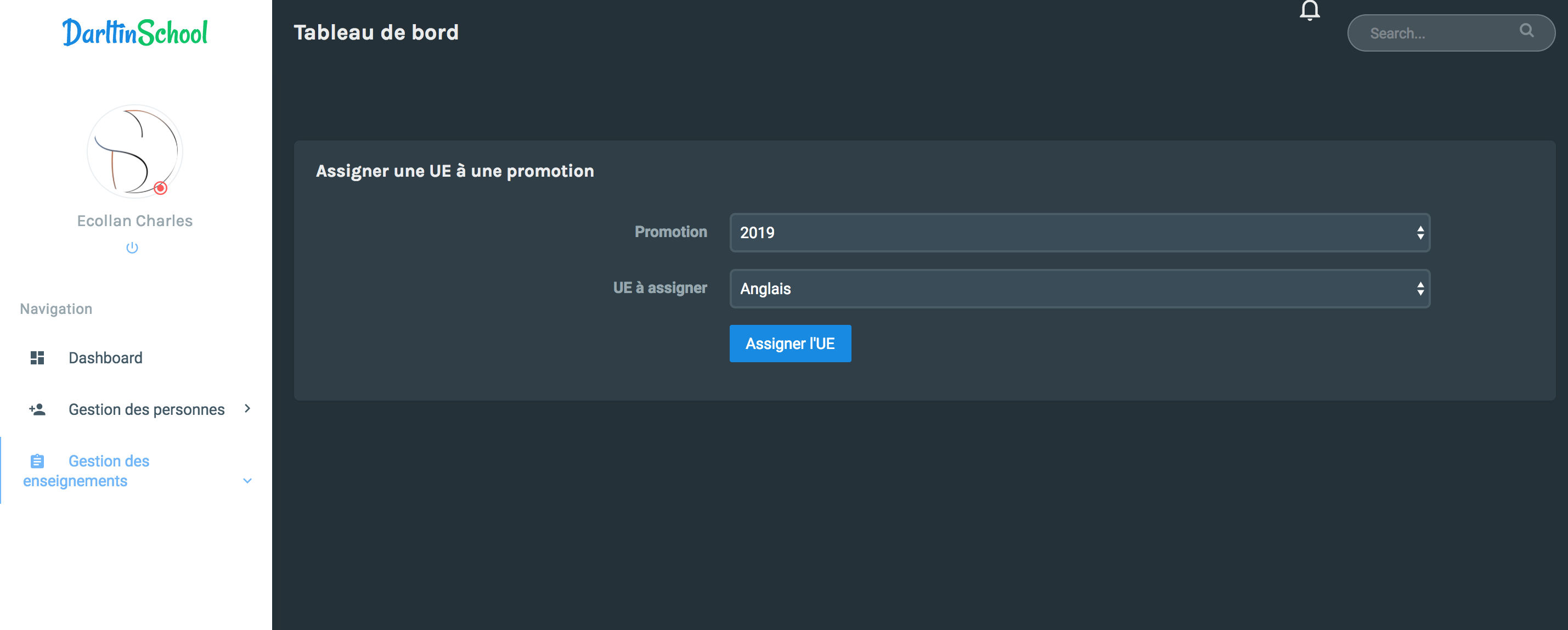
#### Ajouter une matière



Après avoir ajouté une UE, pour ajouter une matière, il faut remplir les informations suivantes :

* Nom de la matière
* Choix de l’UE

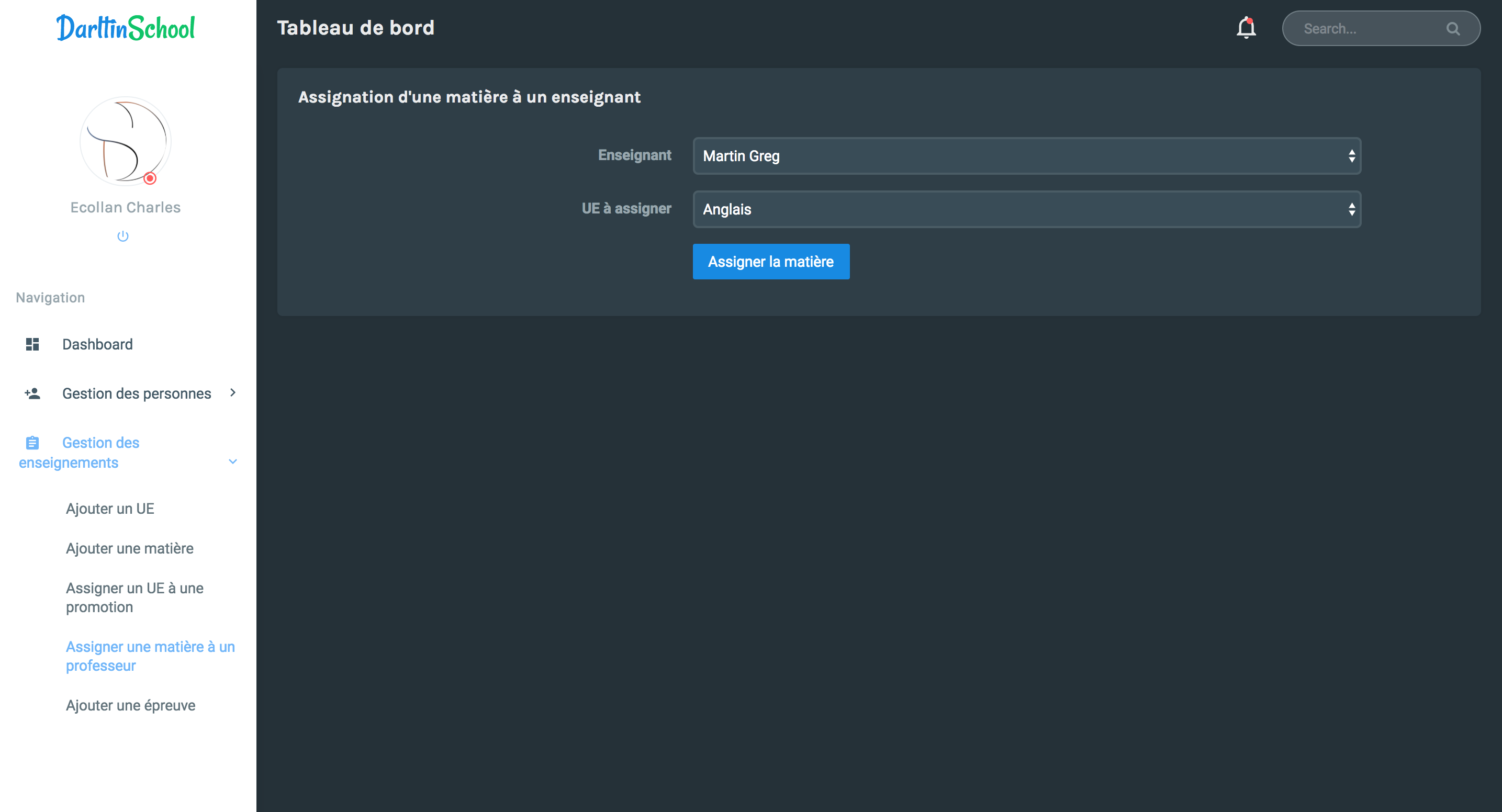
#### Assigner un UE à une promotion



Après avoir rentré une promotion et un UE, pour assigner un UE à une promotion, il faut remplir les informations suivantes :

* Nom de la promotion
* UE à assigner

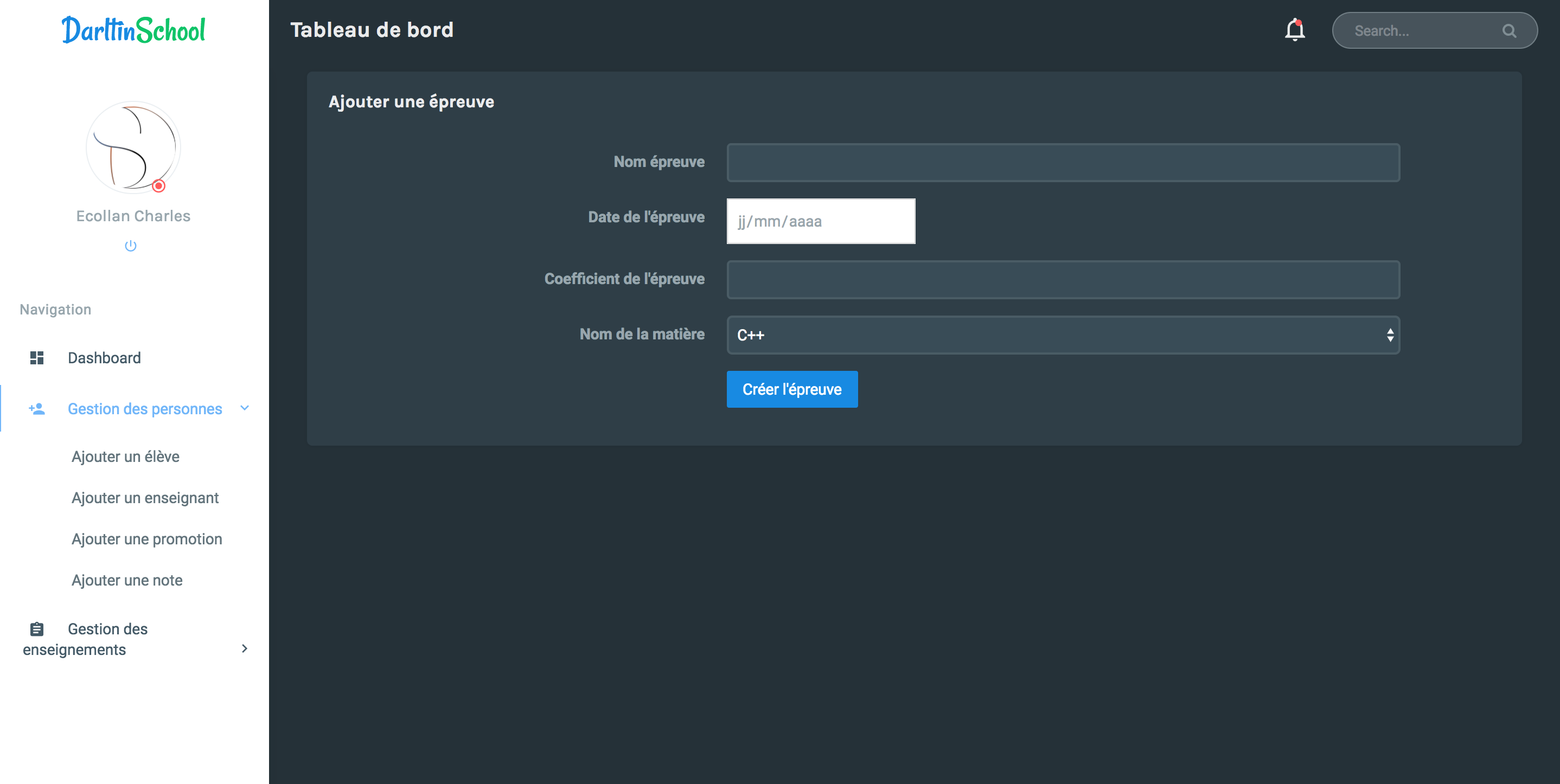
#### Assigner une matière à un enseignant



Après avoir rentré un enseignant et un UE, pour assigner une matière à un enseignant, il faut remplir les informations suivantes :

* Nom de l’enseignant
* UE à assigner

#### Ajouter une épreuve



Pour ajouter une épreuve, il faut remplir les informations suivantes :

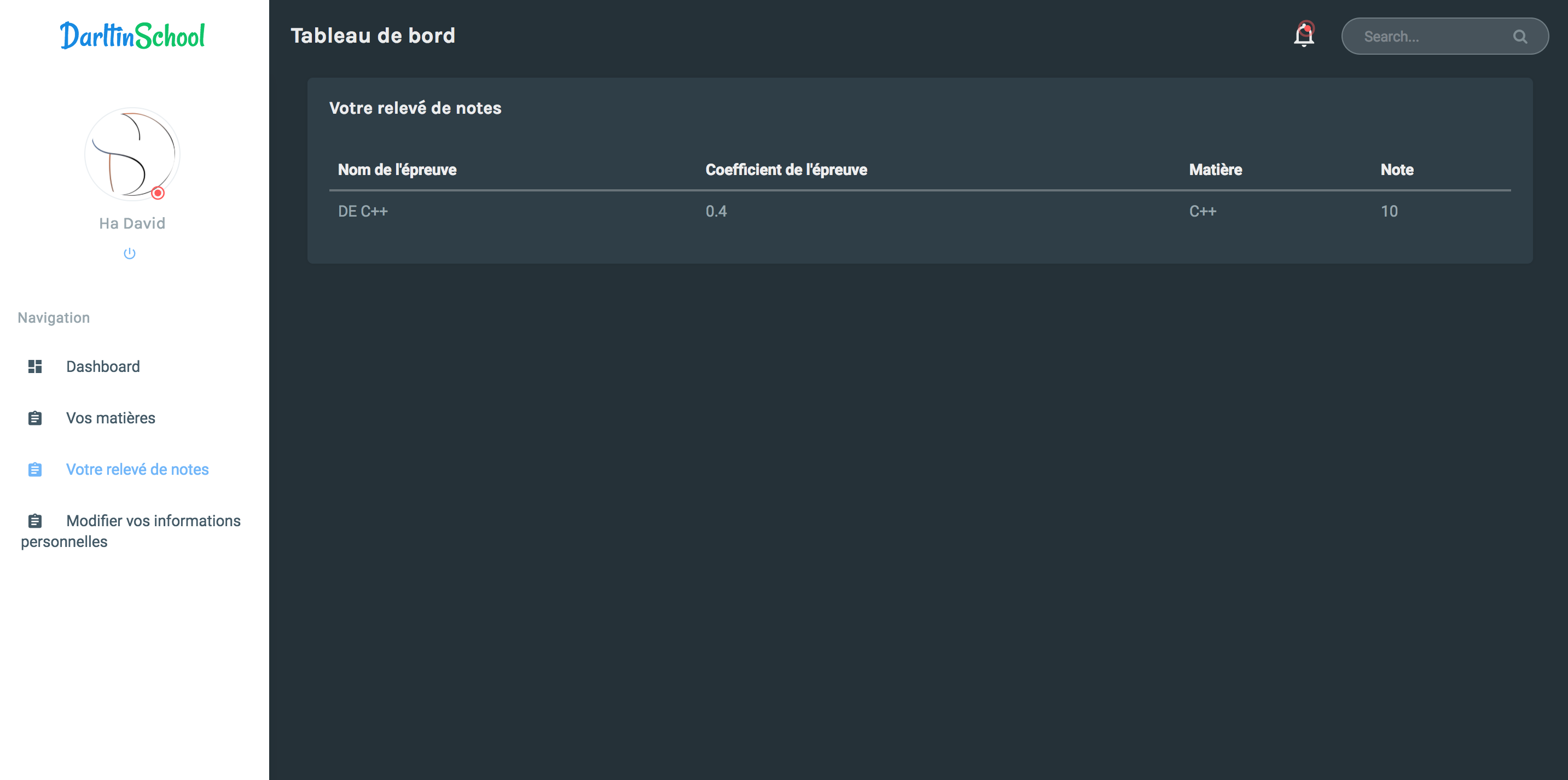
* Nom de l’épreuve (DE, TP…)
* Date
* Coefficient de l’épreuve
* Nom de la matière (préalablement créée)

## Élèves

#### Vos matières

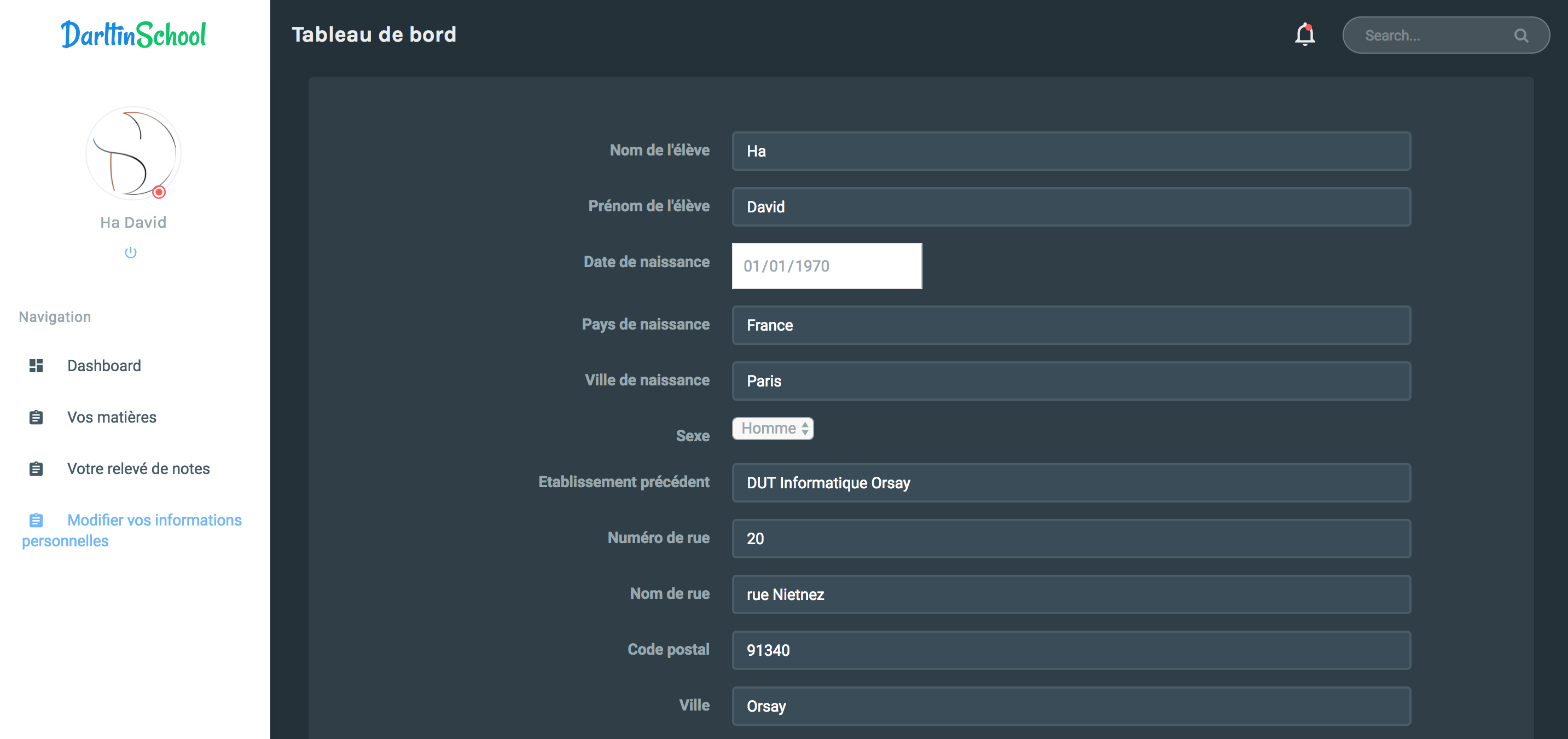
L’élève peut consulter toutes ses matières via cette vue.

#### Votre relevé de notes



L’élève peut consulter toutes ses notes des épreuves (DE, TP…).

#### Modifier vos informations personnelles



L’élève peut modifier ses informations personnelles.

# Conclusion

L’application actuelle respecte bien celle imaginée dès le début dans le cahier des charges. Ce projet nous a permis de mettre en avant nos connaissances sur les bases de données ainsi que son intégration dans une application telle qu’un site internet.

Ce fut également une expérience très enrichissante dans le sens où on a appris à travailler en équipe en utilisant un système de versionning comme git.

# Annexe 1 : Script SQL

#------------------------------------------------------------

# Script MySQL.

#------------------------------------------------------------

#------------------------------------------------------------

# Table: eleve

#------------------------------------------------------------

CREATE TABLE eleve(

matricule int (11) Auto\_increment NOT NULL ,

nom Varchar (255) NOT NULL ,

prenom Varchar (255) NOT NULL ,

date\_naissance Date NOT NULL ,

ville\_naissance Varchar (255) NOT NULL ,

pays\_naissance Varchar (255) NOT NULL ,

sexe Varchar (255) NOT NULL ,

date\_inscription Date NOT NULL ,

etablissement\_precedent Varchar (255) NOT NULL ,

num\_rue Int NOT NULL ,

nom\_rue Varchar (255) NOT NULL ,

code\_postal Int NOT NULL ,

ville Varchar (255) NOT NULL ,

tel\_domicile Int NOT NULL ,

tel\_mobile Int NOT NULL ,

email Varchar (25) NOT NULL ,

nom\_medecin Varchar (255) NOT NULL ,

prenom\_medecin Varchar (255) NOT NULL ,

tel\_medecin Int NOT NULL ,

vaccinations Longtext NOT NULL ,

allergies Longtext NOT NULL ,

remarques\_medicales Longtext NOT NULL ,

photo Varchar (255) NOT NULL ,

password Varchar (255) ,

nom\_promo Varchar (255) ,

PRIMARY KEY (matricule )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: annee

#------------------------------------------------------------

CREATE TABLE annee(

annee\_debut Int NOT NULL ,

annee\_fin Int NOT NULL ,

PRIMARY KEY (annee\_debut )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: contact

#------------------------------------------------------------

CREATE TABLE contact(

nom\_contact Varchar (255) NOT NULL ,

prenom\_contact Varchar (255) NOT NULL ,

num\_rue\_contact Int NOT NULL ,

nom\_rue\_contact Varchar (255) NOT NULL ,

code\_postal\_contact Int NOT NULL ,

ville\_contact Varchar (255) NOT NULL ,

tel\_contact Int NOT NULL ,

email\_contact Varchar (255) NOT NULL ,

id\_contact int (11) Auto\_increment NOT NULL ,

matricule Int NOT NULL ,

PRIMARY KEY (id\_contact )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: document

#------------------------------------------------------------

CREATE TABLE document(

convoc\_parent Longtext NOT NULL ,

id\_document int (11) Auto\_increment NOT NULL ,

matricule Int NOT NULL ,

PRIMARY KEY (id\_document )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Promo

#------------------------------------------------------------

CREATE TABLE Promo(

nom\_promo Varchar (255) NOT NULL ,

PRIMARY KEY (nom\_promo )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: matiere

#------------------------------------------------------------

CREATE TABLE matiere(

nom\_matiere Varchar (255) NOT NULL ,

nom\_ue Varchar (255) NOT NULL ,

PRIMARY KEY (nom\_matiere )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: ue

#------------------------------------------------------------

CREATE TABLE ue(

nom\_ue Varchar (255) NOT NULL ,

coeff\_ue Double ,

PRIMARY KEY (nom\_ue )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: enseignant

#------------------------------------------------------------

CREATE TABLE enseignant(

nom\_enseignant Varchar (255) NOT NULL ,

prenom\_enseigant Varchar (255) NOT NULL ,

id\_enseignant int (11) Auto\_increment NOT NULL ,

email\_enseignant Varchar (255) ,

password Varchar (255) ,

PRIMARY KEY (id\_enseignant )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: epreuve

#------------------------------------------------------------

CREATE TABLE epreuve(

nom\_epreuves Varchar (255) NOT NULL ,

date\_epreuve Date ,

coeff\_epreuve Double ,

nom\_matiere Varchar (255) NOT NULL ,

PRIMARY KEY (nom\_epreuves )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: administrateur

#------------------------------------------------------------

CREATE TABLE administrateur(

nom\_admin Varchar (255) ,

prenom\_admin Varchar (255) ,

id\_admin int (11) Auto\_increment NOT NULL ,

email\_admin Varchar (255) ,

password Varchar (255) ,

PRIMARY KEY (id\_admin )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: present

#------------------------------------------------------------

CREATE TABLE present(

annee\_debut Int NOT NULL ,

matricule Int NOT NULL ,

PRIMARY KEY (annee\_debut ,matricule )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: etudie

#------------------------------------------------------------

CREATE TABLE etudie(

matricule Int NOT NULL ,

nom\_ue Varchar (255) NOT NULL ,

PRIMARY KEY (matricule ,nom\_ue )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: passe

#------------------------------------------------------------

CREATE TABLE passe(

note Double ,

matricule Int NOT NULL ,

nom\_epreuves Varchar (255) NOT NULL ,

PRIMARY KEY (matricule ,nom\_epreuves )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: enseigne

#------------------------------------------------------------

CREATE TABLE enseigne(

nom\_matiere Varchar (255) NOT NULL ,

id\_enseignant Int NOT NULL ,

PRIMARY KEY (nom\_matiere ,id\_enseignant )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: valide

#------------------------------------------------------------

CREATE TABLE valide(

nom\_promo Varchar (255) NOT NULL ,

nom\_ue Varchar (255) NOT NULL ,

PRIMARY KEY (nom\_promo ,nom\_ue )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: gere

#------------------------------------------------------------

CREATE TABLE gere(

id\_admin Int NOT NULL ,

nom\_promo Varchar (255) NOT NULL ,

PRIMARY KEY (id\_admin ,nom\_promo )

)ENGINE=InnoDB;

ALTER TABLE eleve ADD CONSTRAINT FK\_eleve\_nom\_promo FOREIGN KEY (nom\_promo) REFERENCES Promo(nom\_promo);

ALTER TABLE contact ADD CONSTRAINT FK\_contact\_matricule FOREIGN KEY (matricule) REFERENCES eleve(matricule);

ALTER TABLE document ADD CONSTRAINT FK\_document\_matricule FOREIGN KEY (matricule) REFERENCES eleve(matricule);

ALTER TABLE matiere ADD CONSTRAINT FK\_matiere\_nom\_ue FOREIGN KEY (nom\_ue) REFERENCES ue(nom\_ue);

ALTER TABLE epreuve ADD CONSTRAINT FK\_epreuve\_nom\_matiere FOREIGN KEY (nom\_matiere) REFERENCES matiere(nom\_matiere);

ALTER TABLE present ADD CONSTRAINT FK\_present\_annee\_debut FOREIGN KEY (annee\_debut) REFERENCES annee(annee\_debut);

ALTER TABLE present ADD CONSTRAINT FK\_present\_matricule FOREIGN KEY (matricule) REFERENCES eleve(matricule);

ALTER TABLE etudie ADD CONSTRAINT FK\_etudie\_matricule FOREIGN KEY (matricule) REFERENCES eleve(matricule);

ALTER TABLE etudie ADD CONSTRAINT FK\_etudie\_nom\_ue FOREIGN KEY (nom\_ue) REFERENCES ue(nom\_ue);

ALTER TABLE passe ADD CONSTRAINT FK\_passe\_matricule FOREIGN KEY (matricule) REFERENCES eleve(matricule);

ALTER TABLE passe ADD CONSTRAINT FK\_passe\_nom\_epreuves FOREIGN KEY (nom\_epreuves) REFERENCES epreuve(nom\_epreuves);

ALTER TABLE enseigne ADD CONSTRAINT FK\_enseigne\_nom\_matiere FOREIGN KEY (nom\_matiere) REFERENCES matiere(nom\_matiere);

ALTER TABLE enseigne ADD CONSTRAINT FK\_enseigne\_id\_enseignant FOREIGN KEY (id\_enseignant) REFERENCES enseignant(id\_enseignant);

ALTER TABLE valide ADD CONSTRAINT FK\_valide\_nom\_promo FOREIGN KEY (nom\_promo) REFERENCES Promo(nom\_promo);

ALTER TABLE valide ADD CONSTRAINT FK\_valide\_nom\_ue FOREIGN KEY (nom\_ue) REFERENCES ue(nom\_ue);

ALTER TABLE gere ADD CONSTRAINT FK\_gere\_id\_admin FOREIGN KEY (id\_admin) REFERENCES administrateur(id\_admin);

ALTER TABLE gere ADD CONSTRAINT FK\_gere\_nom\_promo FOREIGN KEY (nom\_promo) REFERENCES Promo(nom\_promo);

# Annexe 2 : Code Projet

Voir dossier code source