



HAUTE ÉCOLE DE LA COMMUNAUTÉ FRANÇAISE EN HAINAUT  
Catégorie technique  
8A Avenue Victor Maistriau 7000 Mons

---

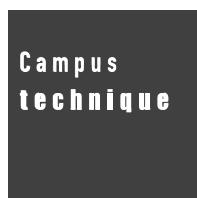
# Développement d'un site web permettant de suivre un cours et de tester les connaissances acquises

---

Travail de fin d'études réalisé en vue de l'obtention du titre de bachelier en  
Informatique et systèmes, orientation réseaux et télécommunications

*Étudiant :*  
Alexandre DUCOBU

*Promoteur :*  
Erwin DESMET



Année académique 2017 - 2018





HAUTE ECOLE  
EN HAINAUT

HAUTE ÉCOLE DE LA COMMUNAUTÉ FRANÇAISE EN HAINAUT  
Catégorie technique  
8A Avenue Victor Maistriau 7000 Mons

---

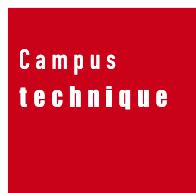
# Développement d'un site web permettant de suivre un cours et de tester les connaissances acquises

---

Travail de fin d'études réalisé en vue de l'obtention du titre de bachelier en  
Informatique et systèmes, orientation réseaux et télécommunications

*Étudiant :*  
Alexandre DUCOBU

*Promoteur :*  
Erwin DESMET



Janvier 2018  
Année académique 2017 - 2018



*En prologue de ce travail de fin d'études,  
je tiens à remercier les personnes qui m'ont  
aidé et soutenu dans la réalisation de celui-ci.*

*Tout d'abord, je remercie Monsieur Desmet,  
mon promoteur, pour ses conseils ainsi que  
pour son aide lors de la réalisation de mon  
pré-TFE et de mon TFE.*

*Je remercie également les professeurs de la  
HeH pour leur accompagnement tout au  
long de mon cursus.*

*Enfin, je remercie ma famille et mes amis  
pour leur soutien durant toute la  
conception de mon travail de fin d'études.*



# Table des matières

<b>Table des figures</b>	<b>3</b>
Abstract . . . . .	4
1 Présentation . . . . .	6
1.1 Introduction . . . . .	6
1.2 Objectif d'un travail de fin d'études . . . . .	6
1.3 Le sujet . . . . .	7
1.4 Choix du TFE . . . . .	8
2 Choix personnels . . . . .	9
2.1 Le projet . . . . .	9
2.2 Technologies . . . . .	9
2.3 Outils . . . . .	13
2.4 Architecture . . . . .	14
3 Fonctionnement de Laravel . . . . .	16
3.1 Authentification . . . . .	16
3.2 Migration . . . . .	16
3.3 Sécurité . . . . .	17
4 Analyse . . . . .	18
4.1 La phase d'analyse . . . . .	18
4.2 Maquettes . . . . .	18
4.3 UML . . . . .	20
4.4 Découverte de Laravel . . . . .	22
4.5 Cas d'utilisation . . . . .	22
4.6 Exigences fonctionnelles et non-fonctionnelles . . . . .	23
4.7 Seconde phase d'analyse . . . . .	24
5 Conception . . . . .	25
5.1 Organisation de la phase de développement . . . . .	25
5.2 Premier phase . . . . .	25
5.3 Seconde phase . . . . .	28
5.4 Résultat . . . . .	33
6 Conclusion . . . . .	34
6.1 Conclusion du projet . . . . .	34
6.2 Améliorations possibles . . . . .	38
6.3 Apports du TFE . . . . .	39
6.4 Conclusion . . . . .	40
Références . . . . .	41
<b>Annexes</b>	<b>44</b>

# Table des figures

Figure	Titre	Page
1	Bootstrap . . . . .	10
2	Font Awesome . . . . .	10
3	Quill.js . . . . .	10
4	Comparaison des recherches pour Laravel, Symfony et CakePHP . . .	11
5	Laravel . . . . .	12
6	MailHog . . . . .	12
7	Laravel Valet . . . . .	13
8	Atom . . . . .	13
9	Logo de git . . . . .	13
10	Gitmoji . . . . .	13
11	Architecture de Laravel . . . . .	15
12	Évolution de la barre de navigation . . . . .	18
13	La vue des cours . . . . .	19
14	Schéma conceptuel de la base de données . . . . .	20
15	Tables créées par Laravel . . . . .	21
16	Icône d'étudiant . . . . .	22
17	Icône de professeur . . . . .	22
18	Schéma conceptuel de la base de données mis à jour . . . . .	24
19	Structure d'un projet Laravel, avec le dossier public à supprimer . . .	26
20	Barre de navigation entre les différentes parties . . . . .	29
21	Boutons d'ajout et de publication . . . . .	30
22	Bogue des cases à cochées ( <i>cochée - non cochée</i> ) . . . . .	30
23	Compte rendu de l'importation de neuf utilisateurs . . . . .	32
24	Aperçu d'un cours terminé, à ajouter aux favoris . . . . .	35

# Abstract

En cette troisième et dernière année de bachelier en Informatique & systèmes, le temps est venu d'effectuer le fameux TFE.

Le TFE, ou *Travail de Fin d'Études*, est l'aboutissement de trois années d'apprentissage, tant théorique que pratique, qui permet à l'étudiant de révéler les connaissances et le savoir-faire qu'il a acquis dans le domaine de son cursus.

Celui-ci prendra la forme d'un site permettant d'apprendre et de créer des cours de tous niveaux, et cela, sur n'importe quel sujet (langue, informatique, sciences,).

Il proposera, en tant qu'exemples, différents cours composés de théorie et d'exercices.

Il fournira aussi un outil de création de cours afin que les utilisateurs puissent proposer de nouveaux sujets.

L'apprentissage se fera en trois étapes :

1. L'utilisateur découvrira le nouveau sujet par de la théorie ainsi que par un ou plusieurs exemples. Il en apprendra alors l'utilité et le fonctionnement.
2. Entre deux parties théoriques, l'utilisateur mettra en pratique ce qu'il aura appris au travers de petits QCM.
3. Une fois le chapitre terminé, un questionnaire (QCM, ordonnancement du code,) sera proposé à l'utilisateur. Celui-ci sera noté sur 10 afin que l'utilisateur puisse se juger et s'améliorer.

Le passage au chapitre suivant requerra une cote minimale de 7/10.

Lors de la conception du site, j'ai utilisé plusieurs technologies telles que les frameworks Bootstrap et Laravel, et le trio standard que forment HTML, CSS et JS (*épaulé par jQuery*).

Pour ce qui est de la base de données, j'ai encore utilisé un standard : MySQL.

Bootstrap permet de faciliter la création du design de sites et d'applications web, ainsi que la conception de sites web adaptatifs (*responsive design*).

C'est l'un des projets les plus populaires sur GitHub.

Laravel est un framework PHP fournissant des fonctionnalités en termes de routage de requête, d'authentification, de gestion d'exceptions, de vue, de migration de base de données, etc.

Ce travail m'a permis d'apprendre à utiliser de nouvelles technologies et à travailler sur un même projet pendant une longue période : une année scolaire.

Bien sûr, le développement n'a pas été continu. Entre les différents projets, les examens et le stage, j'ai dû le mettre de côté à plusieurs reprises.

Après la phase d'analyse du stage (*les trois premières semaines*), je n'ai plus eu de temps pour travailler sur le TFE. Par contre, pendant mes recherches liées au stage, j'ai engrangé des informations et outils utiles au développement de mon travail de fin d'études.

Grâce au stage, j'ai pu améliorer l'analyse que j'avais effectuée auparavant.

# 1 Présentation

## 1.1 Introduction

En cette troisième et dernière année de bachelier en Informatique & systèmes, le temps est venu d'effectuer le fameux TFE.

Le TFE, ou *Travail de Fin d'Études*, est l'aboutissement de trois années d'apprentissage, tant théorique que pratique, qui permet à l'étudiant de révéler les connaissances et le savoir-faire qu'il a acquis dans le domaine de son cursus.

## 1.2 Objectif d'un travail de fin d'études

Le TFE est un travail personnel que l'étudiant réalise pour approfondir un sujet qui l'intéresse particulièrement.

Il doit permettre à l'étudiant de démontrer ses connaissances et son savoir-faire dans un domaine professionnel en rapport direct avec son cursus. En pratique, la réalisation du TFE se traduit principalement par :

- La recherche d'un projet à caractère professionnel et la rédaction d'un cahier des charges.
- La recherche de documentations, l'analyse et la réalisation du travail.
- La rédaction d'un rapport réunissant et organisant les informations collectées, exposant les méthodes utilisées, communiquant les analyses et résultats obtenus, en vue de répondre au cahier des charges.
- La présentation et la défense orale du travail devant un jury.

### **1.3 Le sujet**

Dans ce but, j'ai choisi de développer un site éducatif d'enseignement en ligne sur lequel tout le monde peut s'inscrire et écrire des cours.

L'idée m'est venue l'année dernière lors d'une discussion avec deux de mes sœurs (*l'une est institutrice primaire, l'autre est professeure de français*).

Grâce à ce site, elles pourraient enseigner de nouvelles matières et tester rapidement les connaissances acquises.

De plus, leurs élèves pourraient avancer à leur vitesse tout en étant testés périodiquement.

Ils auront alors un rôle actif plutôt que passif, ce qui les impliquera d'avantage et aura tendance à améliorer leur mémorisation des leçons.

De même, si l'un d'entre-eux décroche quelques fois lors d'un cours, cela n'arrivera pas avec le site vu que l'apprentissage sera individuel.

Lors du second quadrimestre de l'année dernière, j'ai effectué un pré-TFE lié à ce projet dans le cadre du cours de « *Gestion de projets* ».

Celui-ci était développé en PHP pur et axé sur un seul cours (*Apprendre le Python*).

Le but était double.

Premièrement, j'ai pu avoir une première idée du fonctionnement et de la difficulté du projet final.

Deuxièmement, j'ai vu, avec monsieur Desmet, que le projet me convenait et qu'il avait du potentiel en tant que travail de fin d'études.

## 1.4 Choix du TFE

Le site permettra d'apprendre et de créer des cours de tous niveaux, et cela, sur n'importe quel sujet (langue, informatique, sciences...).

Le site proposera, en tant qu'exemples, différents cours composés de théorie et d'exercices.

Il fournira aussi un outil de création de cours afin que les utilisateurs puissent proposer de nouveaux sujets.

L'apprentissage se fera en trois étapes :

1. L'utilisateur découvrira le nouveau sujet par de la théorie ainsi que par un ou plusieurs exemples. Il en apprendra alors l'utilité et le fonctionnement.
2. Entre deux parties théoriques, l'utilisateur mettra en pratique ce qu'il aura appris au travers de petits QCM.
3. Une fois le chapitre terminé, un questionnaire (QCM, ordonnancement du code,...) sera proposé à l'utilisateur. Celui-ci sera noté sur 10 afin que l'utilisateur puisse se juger et s'améliorer.

Le passage au chapitre suivant requerra une cote minimale de 7/10.

Il pourra être utilisé aussi bien par les écoles que par « *les particuliers* ».

## 2 Choix personnels

### 2.1 Le projet

Le TFE consiste à développer un site d'enseignement en ligne.

Il permettra d'écrire facilement un cours, et de s'y inscrire en un clic.

Chaque utilisateur aura accès aux statistiques liées aux cours auxquels il sera inscrit, ainsi qu'à ceux qu'il aura, *potentiellement*, écrits.

Celles-ci porteront sur le nombre de cours de l'utilisateur, les résultats pour chaque chapitre et cours, ainsi que sur la moyenne de ces résultats.

De plus, les « *professeurs* » auront la possibilité d'importer des listes d'« *étudiants* » (*format .csv*) et des questionnaires (*format .json*).

Les étudiants inscrits depuis une liste recevront automatiquement un mail les prévenant de l'inscription et leur demandant de créer un mot de passe pour le site.

Les questionnaires seront composés de deux à dix questions à choix multiples. Seul un maximum de cinq questions sera affiché de manière aléatoire afin que chaque questionnaire soit différent.

Bien entendu, ils peuvent être créés et modifiés en un clic depuis le site.

Pour finir, l'avantage de la technologie web est que, quel que soit le matériel de l'utilisateur (*ordinateur, tablette ou smartphone*), celui-ci aura accès à l'application, à ses données et aux mêmes fonctionnalités.

### 2.2 Technologies

J'ai choisi de travailler avec des frameworks lors de ce projet.

Ceux-ci évitent de perdre du temps à réinventer la roue en réutilisant ce qui a déjà été fait par d'autres, souvent plus compétents, et qui a, en plus, été utilisé et validé par de nombreux utilisateurs.

Ils servent surtout à assister le développeur dans son travail plutôt qu'à combler son manque de connaissances.

Utiliser un framework ne présente pratiquement que des avantages.

En effet, l'utilisation d'un composant, déjà tout prêt et qui a fait ses preuves, est la promesse d'un gain de temps, de fiabilité, de mises à jour simples,...

Il existe des frameworks pour de nombreuses technologies : Java, PHP, CSS, etc.

Pour le côté client (*le front-end*), j'ai choisi d'utiliser le trio HTML5, CSS et JavaScript (*épaulé par jQuery*).

Côté design et responsive design, j'ai utilisé **Bootstrap**, un framework web open-source utile à la création du design de sites et d'applications web. Nous l'avons abordé à l'école l'année dernière.

Il est très simple à utiliser et à modifier, et permet également de créer, facilement, un site adaptatif.

C'est le framework front-end le plus populaire de nos jours avec plus de 125 000 étoiles sur GitHub.

En effet, parmi ces quatre frameworks : Bootstrap, Foundation, Semantic UI et UIkit, c'est Bootstrap qui est le plus populaire, et de loin. En effet, l'un des facteurs de cette popularité est que le plus répandu des trois autres ne possède *que* 41 500 étoiles sur GitHub.

Ceux-ci proposent un ensemble cohérent de composants d'interface : boutons, libellés, icônes, miniatures, formulaires...

Ils permettent de rendre plus rapide et de simplifier le développement de l'interface de sites web.

Le but du site étant éducatif, je l'ai imaginé comme s'affichant sur un tableau d'école. J'ai alors découvert, au cours de mes pérégrinations, **Bootswatch**.

Ce site propose des thèmes gratuits pour Bootstrap dont un nommé *Sketchy*. Celui-ci propose, comme son nom l'indique, un look « *dessiné à la main* » qui se lie parfaitement à l'idée du tableau.

Pour ce qui est des icônes, j'utilise **Font Awesome** depuis quelques temps.

C'est une police d'écriture qui permet d'afficher des icônes gratuites, personnalisables et redimensionnables, car sous format vectoriel.

Dans sa dernière version, la version 5, 1 109 icônes gratuites sont disponibles.



FIGURE 1 – Bootstrap



FIGURE 2 – Font Awesome

En ce qui concerne l'éditeur de cours, j'ai décidé d'utiliser **Quill**, un éditeur de texte WYSIWYG<sup>1</sup> léger, modulaire et personnalisable.



FIGURE 3 – Quill.js

---

1. C'est l'acronyme de la locution anglaise « *what you see is what you get* », signifiant littéralement en français « *ce que vous voyez est ce que vous obtenez* ».

Quant au côté serveur (*le back-end*), j'ai choisi le PHP qui est considéré comme une des bases de la création de sites web dits dynamiques, mais également des applications web.

Pour ce faire, j'ai utilisé Laravel, le framework PHP le plus populaire. Il est open-source, écrit en PHP, respecte le principe *modèle-vue-contrôleur*, et est entièrement développé en programmation orientée objet.

Deux de ses points forts sont sa documentation et sa large communauté.

Il regroupe aussi de nombreuses bibliothèques qui facilitent la gestion de sessions, l'authentification, la validation d'entrées « *utilisateurs* », la création de requêtes SQL...

Par exemple, le système de gestion des requêtes HTTP est celui de Symfony auquel un système de routage a été rajouté.

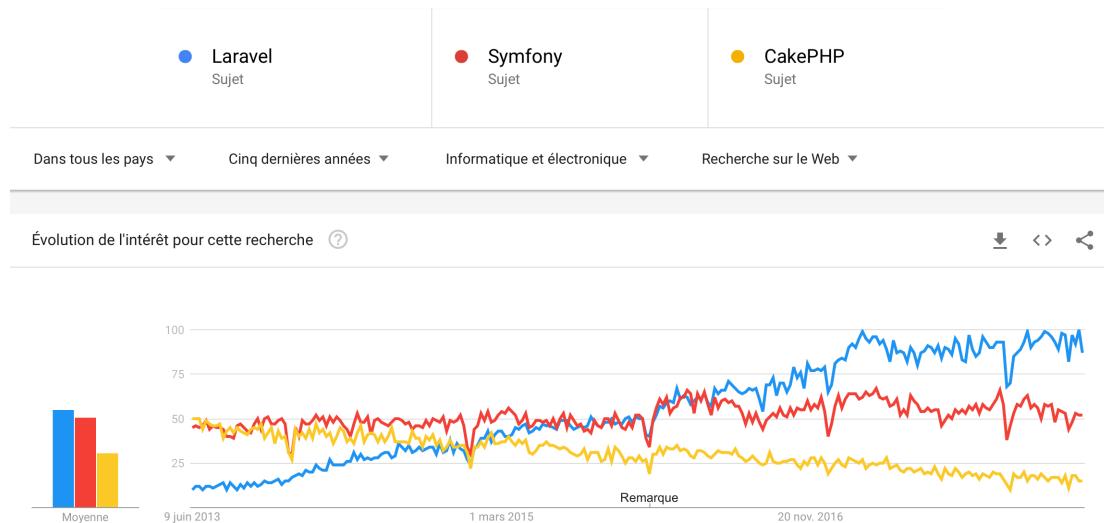


FIGURE 4 – Comparaison des recherches pour Laravel, Symfony et CakePHP<sup>2</sup>

Comme on peut le voir sur ce graphique, la croissance de Laravel sur ces cinq dernières années est impressionnante et il semble inévitable.

Stable, robuste, il est ainsi l'un des frameworks PHP les plus utilisés.

L'application étant dynamique (*les données proviennent d'une base de données*), j'ai choisi d'utiliser MySQL qui forme le couple PHP/MySQL utilisé et proposé par la majorité des hébergeurs web.

2. Lien vers Google Trends : <https://trends.google.fr/trends/>

Finalement, toujours du côté serveur, j'ai dû choisir un service mail local afin de pouvoir tester l'envoi de mail depuis l'application web.

De nombreux services mails sont pris en charge nativement par Laravel, ce qui facilite leur utilisation.

J'ai utilisé MailHog<sup>3</sup> pour sa simplicité d'utilisation. Je l'ai découvert et utilisé l'année passée dans le cadre du projet du cours de *Programmation web* et il ne m'a causé aucun soucis.

Il suffit de le lancer, et il exécute un simple serveur SMTP qui capture tous les messages qui lui sont envoyés pour les afficher dans une interface web.

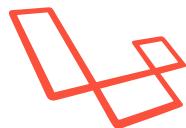


FIGURE 5 – Laravel

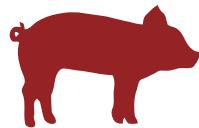


FIGURE 6 – MailHog

---

3. Lien vers GitHub : <https://github.com/mailhog/MailHog>

## 2.3 Outils

Étant donné que mon système d'exploitation est macOS, l'utilisation de **Laravel Valet** était une évidence.

C'est un environnement de développement minimaliste exclusif à macOS qui configure le Mac afin qu'il exécute en permanence Nginx et PHP préalablement installés. En outre, il utilise la technologie DnsMasq pour renvoyer toutes les requêtes depuis **\*.dev** aux sites installés sur la machine locale.

Ainsi, il me suffit de taper **learnit.dev** dans mon navigateur pour avoir accès au site.

En ce qui concerne l'éditeur de texte, c'est **Atom** que j'ai retenu pour sa simplicité, sa modularité ainsi que pour la familiarité que je ressens envers lui.



**Laravel Valet**

FIGURE 7 – Laravel Valet



FIGURE 8 – Atom

Ensuite, j'ai utilisé **git**, en passant par **GitHub**, afin de synchroniser mon code en ligne. De plus, il propose un gestion des versions, ce qui est très intéressant pour retourner en arrière ou voir comment un bogue a été créé ou corrigé.

J'en ai profité pour travailler avec des branches pour différentes fonctionnalités telles que l'ajout des statistiques, l'ajout des questionnaires, des cours,...

En outre, j'ai décidé d'utiliser **Gitmoji**, une initiative pour standardiser et expliquer l'utilisation des émojis dans les messages de commit. L'utilisation d'émojis sur les messages de commit permet d'identifier facilement le but ou l'intention d'un commit d'après les émojis utilisés.

Par exemple, Le premier émoji permet d'identifier la correction d'un bogue, le second désigne une mise à jour au niveau du « *responsive design* », et le dernier identifie la fusion de deux branches.



FIGURE 9 – Logo de git



FIGURE 10 – Gitmoji

## 2.4 Architecture

Laravel, comme de nombreux autres frameworks, utilise le design pattern MVC.

Celui-ci permet de séparer une application web en différentes couches : le Modèle, la Vue et le Contrôleur.

Ce design pattern, ou patron de conception, est destiné aux interfaces graphiques et est très populaire pour les applications web.

L'un des avantages apporté par ce modèles est la clarté de l'architecture qu'il apporte, et impose. Cela simplifie la tâche du développeur qui tenterait d'effectuer une maintenance ou une amélioration sur le projet.

En effet, étant donné que la modification des traitements n'impacte pas les vues, il est assez simple de changer, par exemple, le type de la base de données sans avoir à modifier les vues.

La **couche Modèle**, qui représente les données, est souvent appelée logique métier. Elle consiste en une série de classes qui, lorsque les données proviennent d'une base de données, représentent les différentes tables qui y sont stockées.

Un modèle définit les aspects fonctionnels : les états, les données et la logique applicative.

La **couche Vue** représente l'interface utilisateur et sert de représentation aux modèles. C'est elle qui constitue le font-end, ici développé en HTML et CSS.

En plus de cela, Laravel propose le moteur d'affichage *Blade*, qui simplifie l'utilisation du PHP dans les différentes vues. Celles-ci peuvent dès lors utiliser des instructions de Blade sur les données fournies par les modèles, afin de tester des conditions, effectuer des boucles, modifier ces données, etc.

La **couche Contrôleur** est en charge d'effectuer les traitements, c'est l'élément central du MVC.

Elle contient la logique concernant les actions effectuées par l'utilisateur (*dans la couche Vue*), ce qui permet de faire évoluer l'état du modèle.

De même, c'est cette couche qui récupère les données des modèles et qui les passe aux différentes vues pour les afficher.

En outre, les contrôleurs peuvent effectuer de nombreuses tâches telles que l'exécutions d'algorithmes, l'envoi d'emails,...

Laravel propose aussi ce qu'on appelle le **routage**.

Ce système permet de faire le lien entre une route créée par le développeur (p. ex. cela donnera `http://monsite.be/utilisateurs/liste`), et une méthode définie dans un contrôleur. Celle-ci pourra simplement afficher la vue listant les utilisateurs ou bien effectuer des actions telles la déconnexion et, ensuite, rediriger l'utilisateur vers une vue spécifique.

Ainsi, l'adresse ne se termine plus par le nom d'un fichier `.html` ou `.php`.

Ce qui augmente la sécurité en cachant l'architecture interne de l'application.

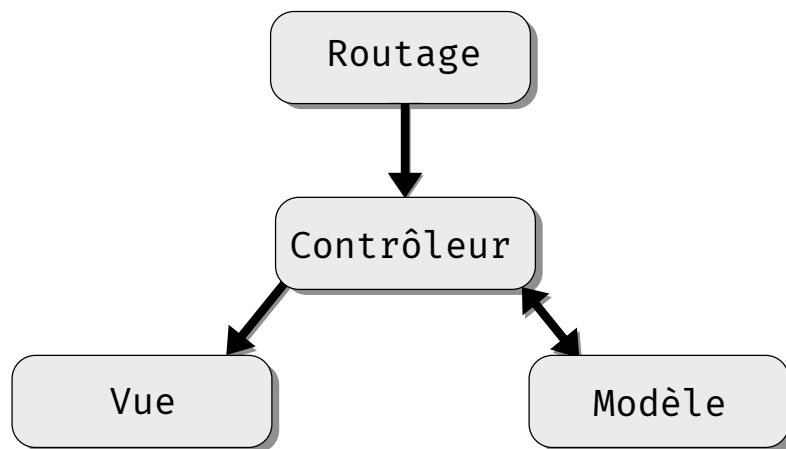


FIGURE 11 – Architecture de Laravel

Afin de déployer le site sur un serveur de test, j'ai été contraint de modifier l'architecture interne du projet Laravel.

Ainsi, j'ai dû déplacer le contenu du dossier `public` à la racine du projet, et le supprimer.

## 3 Fonctionnement de Laravel

### 3.1 Authentification

L'authentification est nativement gérée par Laravel (*au travers du module auth*).

Une fois activée, il faut migrer la base de données afin de prendre en compte l'authentification.

Les utilisateurs peuvent désormais s'inscrire, se (*dé*)connecter ou réinitialiser leur mot de passe (*une fois un serveur mail ajouté*).

#### Les middlewares

Les pages sont protégées par des middlewares (*Authenticate.php* et *RedirectIfAuthenticated.php*).

Les middlewares forment une couche de protection entre les requêtes et l'application : ils effectuent des traitements à l'arrivée ou au départ des requêtes.

Par exemple, la gestion des sessions, des cookies et de l'authentification se fait dans un middleware. On a plusieurs couches de middlewares (*comme des pelures d'ognon*). Chacun effectue son traitement et transmet la requête ou la réponse au suivant.

Le premier, *Authenticate.php*, vérifie qu'un utilisateur est authentifié.

Si ce n'est qu'un invité (*guest*), alors on renvoie *Unauthorized* si la requête est en Ajax. Si elle n'est pas en Ajax, on fait une redirection vers la page d'authentification (*route login*).

Si ce n'est pas un invité, on exécute la requête.

Le second, *RedirectIfAuthenticated.php*, fonctionne pratiquement à l'opposé du fichier *Authenticate.php*.

Si l'utilisateur est authentifié (*auth*), alors on le renvoie à la page de base du site. Sinon, on exécute la requête.

La majorité des pages du site ne doivent être accessibles qu'aux utilisateurs authentifiés. C'est là qu'entrent en jeu les middlewares : il suffit d'ajouter *middleware('auth')* à la route ou au contrôleur d'une page pour en interdire l'accès aux invités.

C'est donc *Authenticate.php* qui est utilisé.

### 3.2 Migration

Laravel propose cet outil dont le but est de s'occuper de la maintenance de la base de données de l'application.

Une migration permet donc de créer et de mettre à jour une base de données en créant et supprimant des tables, des colonnes dans ses tables, en créant des index...

### 3.3 Sécurité

Voici un aperçu de la sécurité mise en place dans l'application.

#### Attaques CSRF

Le CSRF, ou Cross Site Request Forgery, est un type de vulnérabilité des services d'authentification web.

Le fonctionnement de l'attaque est simple : l'attaquant transmet une requête HTTP falsifiée à un utilisateur authentifié. Celle-ci pointe sur une action interne au site et est exécutée par l'utilisateur authentifié *à son insu* et en utilisant *ses propres droits*.

#### Injections SQL

Laravel fournit une méthode de protection contre les injections SQL.

Celles-ci forment l'une des méthodes d'attaque la plus connue et les plus dangereuses en PHP.

Les risques que représente ce genre de faille sont énormes : comme son nom l'indique, cette faille se manifeste lorsqu'il est possible d'injecter ou de modifier une requête SQL.

Ceci en injectant des morceaux de code *non filtrés*, généralement par le biais d'un formulaire d'une page du site.

Le constructeur de requête (*Query builder*) de Laravel utilise des instructions préparées de SQL qui rendent les attaques par injection inimaginables.

Il n'y a donc pas besoin d'effectuer de traitements sur les données avant de les passer au constructeur de requête.

Par contre, les requêtes brutes, utilisées en passant par `DB ::raw()`, y sont vulnérables vu que non préparées...

#### Vérification des formulaires

Laravel permet d'ajouter une vérification aux différents champs des formulaires.

Grâce à cela, tous les champs ont été sécurisés : ajout de regex pour les mots de passe et autres champs texte, contrainte sur la longueur et le type du champ.

De plus, le routage permet de gérer facilement la méthode HTML à utiliser, soit GET, soit POST.

## 4 Analyse

### 4.1 La phase d'analyse

J'ai débuté l'analyse du projet dès la première semaine d'octobre, d'après les recommandations dont monsieur Desmet m'avait fait part lors de la signature du cahier des charges.

Lors de cette phase, il faudrait que je me forme à l'utilisation de Laravel, car je ne l'avais jamais utilisé.

### 4.2 Maquettes

Pendant cette phase, j'ai conçu les maquettes des différentes vues à la main dans un premier temps, puis à l'aide de **NinjaMock**<sup>4</sup> dans un second temps.

Au début, je me suis fortement inspiré du design que j'avais utilisé pour le pré-TFE, mais je me suis vite rendu compte que, vu les nouvelles fonctionnalités à implémenter, cela ne conviendrait pas.

Par exemple, la barre de navigation a gagné en choix : elle est passée de quatre à sept liens dont six cachés dans des menus déroulants.

J'ai quand même gardé, en partie, la vue théorique d'un chapitre qui me semblait assez claire.

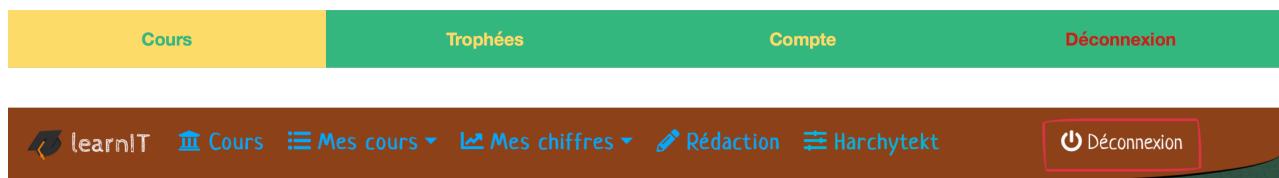


FIGURE 12 – Évolution de la barre de navigation

4. Lien vers NinjaMock : <https://nijamock.com>

J'ai choisi d'utiliser un affichage utilisant des *vignettes* pour exposer les cours. En plus du design agréable apporté, cela permet d'utiliser intelligemment le système de grille de Bootstrap qui adapte le contenu à la taille de l'écran.

De plus, cela m'a permis de développer un design lié à l'univers éducatif. La page est un tableau sur lequel sont accrochées les vignettes, tout comme des affiches le seraient en classe.

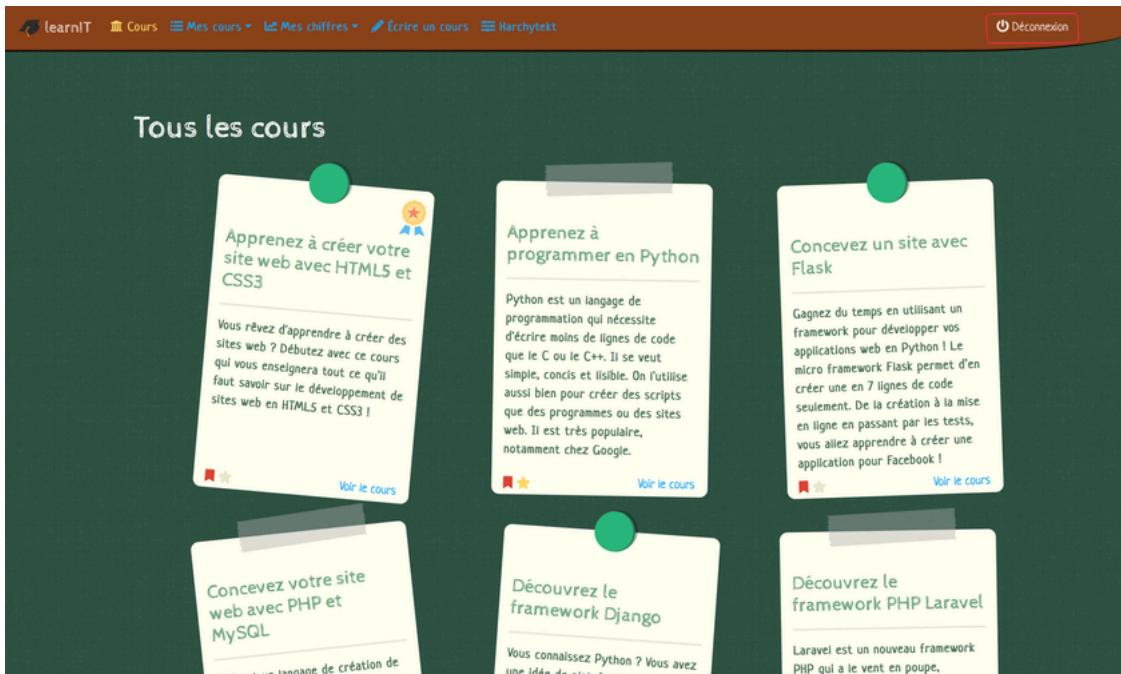


FIGURE 13 – La vue des cours

On peut voir que la barre de navigation prend la forme du cadre en bois du tableau d'une classe, et que les différentes vignettes sont attachées au tableau soit à l'aide d'un aimant, soit à l'aide de papier collant.

## 4.3 UML

Une fois cela fait, je me suis attaqué à la création du schéma de la base de données, d'après une liste des fonctionnalités à implémenter et des différentes vues prévues.

J'ai commencé la base de données avec la tables **users** qui contiendra tous les utilisateurs de l'application. En effet, il faudra être inscrit pour avoir accès à l'application, s'inscrire à des cours et passer les différents tests proposés.

Ensuite, je lui ai ajouté cinq tables.

La première, **comments**, contiendra les commentaires des étudiants à propos des différents cours.

Après, viennent les tables **courses**, **chapters** et **tests** qui contiennent les données des cours, des chapitres et des tests passés par les utilisateurs.

Le projet étant un site d'enseignement, ces tables sont centrales dans son fonctionnement. Un cours possèdera plusieurs chapitres qui se finiront par un test coté.

Finalement, la table **enrollments** fait le lien entre un utilisateur et son inscription à un cours. C'est à elle que sont liés les tests.

Cette table a été créée, car chaque utilisateur peut s'inscrire à plusieurs cours. On se retrouve alors avec une relation *Plusieurs à Plusieurs*, mais il est impossible de lier **directement** deux tables ayant ce type de relation. Il faut donc passer systématiquement par une troisième table.

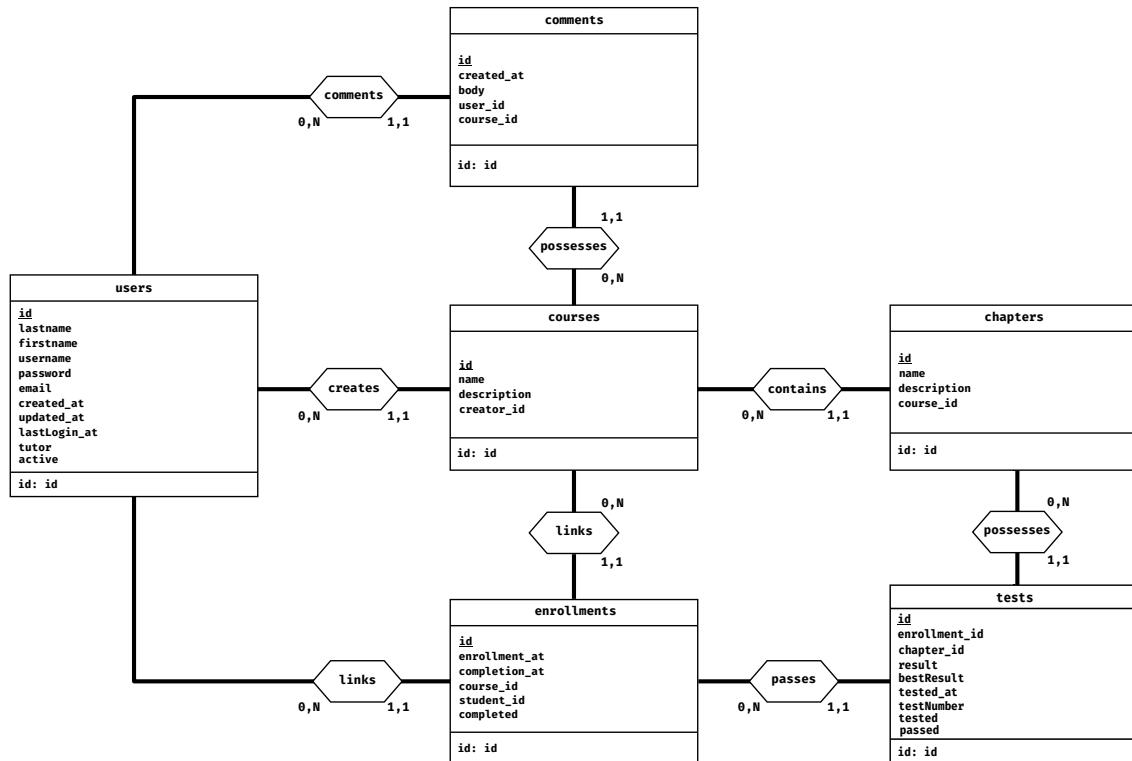


FIGURE 14 – Schéma conceptuel de la base de données

Par défaut, le framework Laravel crée trois tables dans la base de données : **users**, **migrations** et **password\_resets**.

La première, la table **users**, a été créée par Laravel pour gérer les utilisateurs, mais je l'ai modifiée afin qu'elle réponde aux besoins du site.

La deuxième gère les migrations (*p. ex. la création des différentes tables*), et la troisième gère la réinitialisation des mots de passe.

<b>migrations</b>
<b><u>id</u></b>
<b><u>migration</u></b>
<b><u>batch</u></b>
<b><u>id: id</u></b>

<b>password_resets</b>
<b><u>email</u></b>
<b><u>token</u></b>
<b><u>created_at</u></b>
<b><u>id: email</u></b>

FIGURE 15 – Tables créées par Laravel

La table **password\_resets** permet de gérer la réinitialisation des mots de passe en toute sécurité et possède trois champs.

Le premier contient l'adresse email de l'utilisateur ayant demandé la réinitialisation de son mot de passe.

Le deuxième contient le jeton de réinitialisation unique lié à l'adresse email.

Le troisième et dernier champ contient la date et l'heure de création du jeton de réinitialisation.

Grâce à cette table, un utilisateur peut facilement réinitialiser son mot de passe perdu de manière sécurisée.

## 4.4 Découverte de Laravel

Ensuite, je suis passé à la phase de découverte et de formation à Laravel.

Après m'être documenté sur Laravel et son fonctionnement, je me suis mis en quête d'une formation. Heureusement, la documentation officielle de Laravel propose une formation d'introduction *gratuite et complète* au moyen de vidéos disponibles sur **Laracasts**.

La série « **Laravel From Scratch<sup>5</sup>** » se base sur la version 5.4 de Laravel. Elle accompagne le développeur de l'installation à la mise en place d'un site de blogging. Grâce à cette formation, j'ai appris à utiliser les routes, les modèles et les contrôleurs de manière correcte.

J'ai ainsi commencé par faire mes armes sur un projet à part, ce qui m'a permis de bien comprendre comment les différentes couches travaillaient entre-elles.

De plus, j'ai pu effectuer quelques tests sur ce projet avant de passer au développement du TFE.

## 4.5 Cas d'utilisation

L'application sera utilisée par deux types d'utilisateurs : les « *étudiants* » et les « *professeurs* ».

Un étudiant est un utilisateur qui suit des cours, alors qu'un professeur en a écrit au moins un. C'est ainsi que les profils sont définis.

Par défaut, à la création d'un compte, l'utilisateur est défini comme *n'étant pas* un professeur (*le champ **tutor** est à 0*).

L'autre manière de s'inscrire, c'est par l'entremise d'un professeur. Il peut alors ajouter une liste d'étudiants depuis un fichier **.csv**.

L'utilisateur peut facilement voir qu'il est étudiant dans la vue compte, en dessous de son nom. De plus, l'icône présente dans cette vue évoque un étudiant.

Tout étudiant peut devenir professeur : il lui suffit pour cela de publier un cours. Son statut passe alors à celui de professeur, et son icône est mise à jour afin d'évoquer un enseignant.



FIGURE 16 – Icône d'étudiant



FIGURE 17 – Icône de professeur

5. Lien vers la formation : <https://laracasts.com/series/laravel-from-scratch-2017>

## 4.6 Exigences fonctionnelles et non-fonctionnelles

Pendant les trois premières semaines de mon stage en entreprise, j'ai encore eu du temps pour travailler sur le TFE.

J'en ai profité pour lister les exigences fonctionnelles, celles demandées explicitement, et les exigences non-fonctionnelles, celles qui sont explicites.

Certaines des exigences fonctionnelles sont donc :

- Visualiser les différents cours.
- Créer/éditer des chapitres et questionnaires.
- Importer une liste d'étudiants, un questionnaire.
- Gérer les commentaires par cours.
- Gérer les questionnaires cotés et le déblocage des chapitres.
- Afficher des statistiques à propos des cours auxquels l'utilisateur est inscrits et à propos des cours qu'il a écrits.

Les exigences non-fonctionnelles sont, elles :

- L'ergonomie :
  - L'application doit être intuitive pour permettre une prise en main facile et efficace.
  - L'application doit respecter une design clair et rester ergonomique.
- La sécurité :
  - L'application doit sécuriser les différentes vues internes.
  - L'application doit sécuriser les différentes formulaires.

## 4.7 Seconde phase d'analyse

Lors de la reprise du développement après le stage en entreprise, je me suis vite rendu compte que mon analyse n'était pas complète.

J'ai alors décidé de prendre le temps d'analyser les besoins de l'application afin d'améliorer ma précédente analyse.

J'ai ainsi ajouté de nouvelles tables et retiré certains champs de la base de données. Les changements les plus importants sont l'ajout des deux tables **favorites** et **parts**.

La première permet aux utilisateurs de placer des cours dans leurs favoris sans pour autant les obliger de s'y inscrire.

La seconde est plus importante. Désormais, le contenu du chapitre n'est plus stocké dans la table **chapters**, mais dans cette table-ci.

Une partie est donc soit une partie théorique d'un chapitre, soit un questionnaire (*coté ou non*).

Désormais, les chapitres et les parties possèdent un champ **order\_id** permettant de les sélectionner d'après leur ordre.

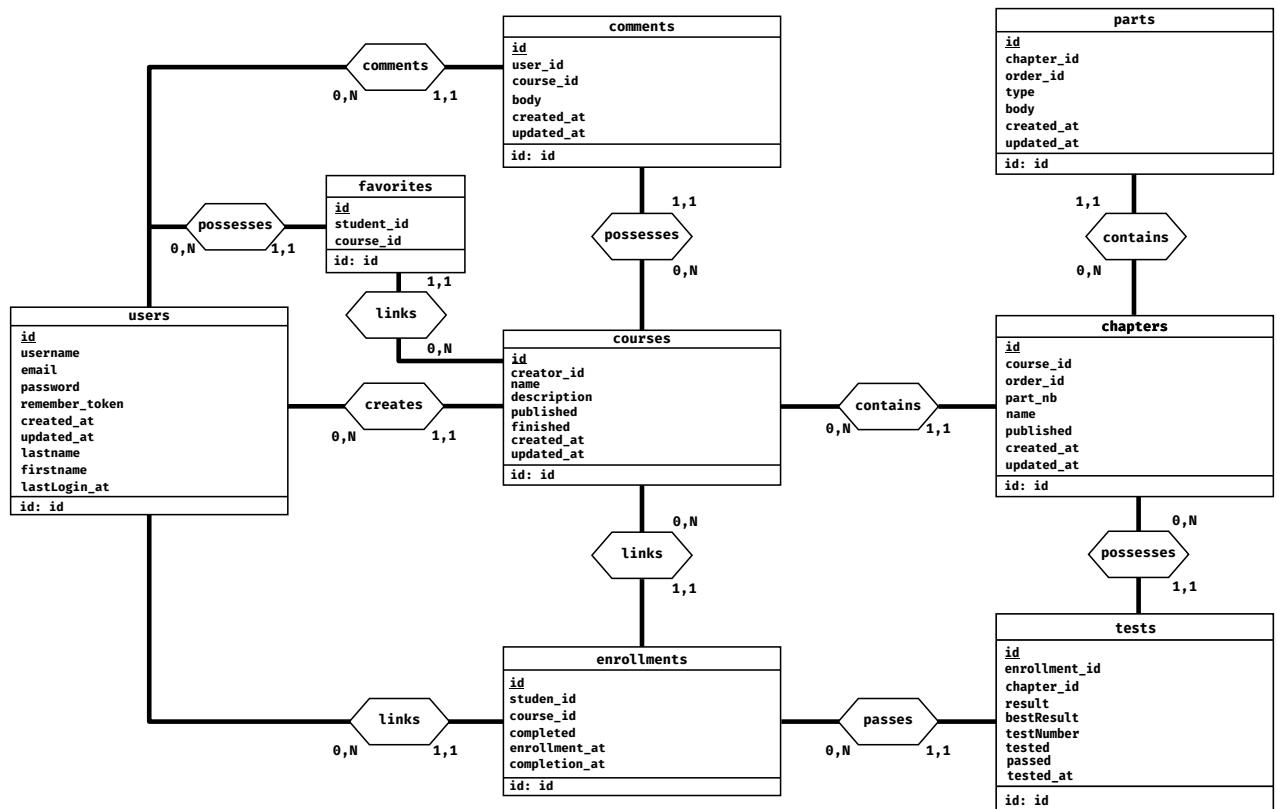


FIGURE 18 – Schéma conceptuel de la base de données mis à jour

## 5 Conception

### 5.1 Organisation de la phase de développement

la phase de développement a été divisée en deux parties.

La première a débuté une semaine après la semaine de vacances de la Toussaint, et a pris fin vers le vingt février, à la fin de la phase d'analyse de mon stage.

Pendant cette phase, j'ai commencé la création des principales vues, de la base de données et j'ai aussi mis la sécurité en place.

La seconde a pris place du quatorze mai au trente mai, après le stage et la rédaction de son rapport.

C'est à ce moment que j'ai mis à jour la base de données, j'ai aussi ajouté la création et la modification de cours et questionnaires.

### 5.2 Premier phase

J'ai donc commencé cette phase le douze novembre.

La première chose que j'ai faite, a été d'initialiser le projet Laravel et de personnaliser sa page d'accueil.

La semaine qui a suivi, j'ai créé la majorité des pages :

- **Cours** : permet à l'utilisateur d'afficher tous les cours disponibles.
- **Mes cours** (*quatre pages*) : comprend les pages permettant à l'utilisateur d'accéder aux cours auxquels il est inscrits, à ses favoris et aux cours qu'il a écrits.
- **Mes chiffres** (*deux pages*) : comprend les pages permettant à l'utilisateur d'accéder aux chiffres liés aux cours auxquels il est inscrits ainsi qu'à ceux qu'il a écrits.
- **Écrire un cours** : permet à l'utilisateur de créer un cours.
- **Compte** : permet à l'utilisateur de modifier ses données, supprimer son compte ou encore d'ajouter une liste d'élèves au format `.csv`.

J'ai ensuite mis l'authentification en place, en passant par une branche git spécifique. J'ai alors ajouté les pages de connexion, d'inscription et de réinitialisation de mot de passe.

Fin décembre, j'ai ajouté une page d'accueil une fois la connexion faite afin que l'utilisateur puisse voir rapidement le nombre de cours auxquels il est inscrit, et qu'il a placé dans ses favoris.

Ensuite, j'ai déployé l'application sur un serveur de test afin de pouvoir tester l'application sur différents appareils tels que des ordinateurs de bureau Windows, des smartphones (*iOS* et *Android*), etc.

Le déploiement a pris du temps, principalement à cause de la connexion avec le serveur qui était très lente. De plus, j'ai été contraint de changer la structure interne de Laravel<sup>6</sup>.

En effet, sur un serveur mutualisé, le dossier `public` n'a aucune raison d'exister, et son contenu doit être placé à la racine du projet.



FIGURE 19 – Structure d'un projet Laravel, avec le dossier public à supprimer

Afin que l'utilisateur puisse réinitialiser son mot de passe après avoir reçu un email, j'ai dû mettre en place un serveur mail.

En local, il m'a suffit de lancer **MailHog** que j'avais déjà utilisé lors d'un projet de l'année dernière.

Sur le serveur, j'ai utilisé **Mailgun**, un service mail de test gratuit. Il a fallu deux semaines pour que les mails s'envoie correctement depuis Mailgun à cause d'un problème de domaine.

Par contre, la mise en place a été rapide grâce à sa prise en charge native par Laravel.

Pendant la majorité du mois de décembre, j'ai mis le développement du TFE en pause afin de développer les projets des différents cours et afin de préparer mes examens.

À la Noël, j'ai réglé un bogue lié aux alertes affichées lors de la déconnexion, de l'envoi d'un mail,...

Ensuite, j'ai rédigé le pré-rapport de TFE. J'en ai profité pour régler un problème d'affichage provenant de **Bootstrap**, ajouter des regex<sup>7</sup> dans les formulaires, et j'ai ajouté l'heure de dernière connexion dans la page du compte.

6. Lien vers la structure : <https://laravel.sillo.org/cours-laravel-5-3-plus-loin-le-deploiement/>

7. Expression régulière (en anglais : **Regular expression**)

J'ai repris le développement de la semaine blanche jusqu'à la fin de la phase d'analyse du stage, le vingt-trois février.

J'ai commencé par ajouter l'état de publication des cours et chapitres, et par régler un bogue d'affichage lors de l'utilisation d'écrans d'à peu près 1 000 pixels.

Ensuite, j'ai construit le reste de la vue *Compte*, et j'ai changé la couleur de son lien dans la barre de navigation.

Les ajouts permettent désormais de modifier son nom, son adresse mail ainsi que son mot de passe. Un étudiant peut supprimer son compte, mais pas un professeur, car il possède des cours susceptibles d'être suivis par des étudiants.

Afin d'améliorer l'expérience utilisateur, j'ai ajouté des pages d'erreurs personnalisées avec des messages explicitant l'erreur et permettant à l'utilisateur de retourner rapidement vers la page d'accueil.

Après cela, j'ai rempli les différentes vues avec des données « *en dur* » afin d'avancer sur le design du site.

Une fois cela fini, j'ai rempli la base de données à la main afin de commencer la création dynamique des vues, et de retirer les données ajoutées précédemment « *en dur* ».

Grâce à la phase d'analyse du stage, j'ai découvert le plug-in jQuery **DataTables** qui permet, dans un tableau, de trier les colonnes, de rechercher des données simplement et d'afficher plus ou moins de résultats.

Cela a été intéressant pour les vues reprenant les chiffres de statistiques liés aux cours.

Après le 25 février, je n'ai plus eu le temps de travailler sur le TFE, mais, lors de mes recherches pour le stage, j'en ai profité pour me constituer une liste de pages, d'idées et de technologies intéressantes pour la suite du développement.

### 5.3 Seconde phase

Lors de la reprise du développement après la fin du stage en entreprise, je me suis vite rendu compte que mon analyse n'était pas complète.

C'est à ce moment-là que j'ai décidé de prendre le temps d'analyser les besoins de l'application afin d'améliorer la base de données, et de lister les différentes exigences de l'application (*cf sections 4.6 et 4.7*).

À ce moment, il me restait trois semaines pour terminer le développement et pour rédiger ce rapport.

J'ai alors décidé de retirer certaines fonctionnalités afin de proposer une application fonctionnelle.

En tout, ce sont quatre fonctionnalités dont deux « *bonus* » qui ont été retirées.

Celles-ci sont :

- Différents niveaux de difficulté d'après l'âge ou le niveau de l'utilisateur.
- Générer un PDF depuis un/des chapitre(s) choisi(s).
- Bonus :
  - Choix de la difficulté des questions, ainsi que gestion de leur pondération.
  - Création d'une application mobile Android pour un accès hors-ligne.

Afin d'être le mieux organisé possible, j'ai décidé de travailler avec une méthode proche de la méthode agile : chaque semaine avait son objectif.

La première semaine, du quatorze au vingt mai, avait pour objectif d'ajouter diverses fonctionnalités liées aux cours ainsi qu'à leur aperçu, et en l'ajout et l'édition de chapitres.

J'ai alors commencé par régler plusieurs erreurs du code JavaScript.

Ensuite, j'ai ajouté la possibilité d'ajouter/retirer un cours à ses inscriptions et à ses favoris en un clic.

Dans la suite logique, les cours sont désormais listés dans toutes les vues de manière dynamique.

Que ce soit la liste des favoris, des cours écrits ou en cours d'écriture, celles-ci sont enfin récupérées depuis la base de données.

Logiquement, s'il y a une liste de cours en cours d'écriture, il faut pouvoir changer leur état comme état publiés. Dès lors, un bouton permet de publier rapidement les cours ainsi que les chapitres.

Les chapitres sont divisé en parties pouvant être de différents types : *théorique*, *questionnaire* ou *questionnaire coté*.

Chacune d'elles est accessible depuis une barre de navigation listant les différentes parties disponibles dans un chapitre.



FIGURE 20 – Barre de navigation entre les différentes parties

Afin d'écrire et d'éditer les parties théoriques, j'ai utilisé **Quill**, un éditeur WYSIWYG libre et open source construit pour le web moderne.

Avec son architecture modulaire et son API expressive, il est entièrement personnalisable pour répondre à tous les besoins.

Désormais, l'édition est accessible depuis un petit bouton, et c'est tout. L'utilisateur peut déjà éditer la partie du chapitre choisie.

Pour les cours de programmation, la coloration syntaxique est de la partie pour de nombreux langages (*tels que Python, Java, C++, PHP, etc.*).

L'objectif de la seconde semaine consistait en l'ajout des questionnaires et, si possible, au début de l'ajout des statistiques.

J'ai commencé cette semaine par l'ajout de la lecture des questionnaires.

Pour commencer, j'ai choisi le format de stockage des questionnaires ainsi que sa structure interne. Je me suis basé sur le JSON pour sa structure simple et efficace, constituée de dictionnaires et de tableaux.

Ensuite, j'ai structuré le fichier afin de représenter un questionnaire simplement. Celui-ci contient plusieurs contraintes telles que le nombre de choix possibles par question qui est de trois, et le nombre de questions par questionnaire qui est compris entre deux et dix.

Une fois cela fait, j'ai sauvegardé un exemple de questionnaire afin de développer la couche vue liée aux questionnaires.

Je passe ainsi par du JavaScript pour récupérer les données du fichier et afficher les différents choix.

Les questions sont affichées aléatoirement et seules cinq questions seront affichées. Le but de cette contrainte est de proposer des questionnaires différents à chaque fois que l'utilisateur le passe.

Deux cas sont possibles lorsqu'un utilisateur passe un questionnaire.

Soit le questionnaire est facultatif et permet de tester ses connaissances, soit le questionnaire est coté.

Dans le premier cas, la correction de la question actuelle est affichée dès qu'un choix est effectué.

Dans le second cas, ce n'est qu'une fois le questionnaire validé que les corrections sont affichées.

Les corrections prennent la forme de petits pop-ups positionnés à côté du choix effectué. L'utilisateur peut alors cacher et afficher les pop-ups en un clic.

Pour en finir avec les questionnaires, j'ai mis en place la vue d'édition d'un questionnaire qui tire parti des formulaires.

Par défaut, celui-ci propose la création de deux questions proposant trois choix chacune. L'utilisateur peut facilement ajouter et supprimer des questions à l'aide des boutons idoines, tout en suivant les règles minimum deux questions et maximum dix.

De plus, l'utilisateur peut aussi importer un questionnaire au format JSON depuis une fenêtre modale. Afin de l'aider à créer des questionnaires au bon format, l'utilisateur a accès à une page d'aide.

Avec l'ajout de ces fonctionnalités, je suis repassé sur tous les boutons de l'application afin de garder une cohérence tant au niveau des couleurs qu'au niveau des icônes les identifiant.

Ainsi, tous les boutons de publication sont de couleur jaune avec un petit nuage, et ceux utilisés pour ajouter un chapitre, une partie sont bleu avec une icône en forme de plus.



FIGURE 21 – Boutons d'ajout et de publication

J'ai mis à jour les différents boutons radio et cases à cocher du site afin de garder une cohérence graphique, mais aussi pour régler un bogue du thème Bootstrap sur iOS.



FIGURE 22 – Bogue des cases à cochées (*cochée - non cochée*)

La semaine s'est terminée avec l'ajout de la partie dynamique des vues statistiques pour les inscriptions de l'utilisateur.

Celui-ci peut désormais voir le nombre de cours auxquels il est inscrit, combien sont terminés, combien de chapitres il a terminé, et la moyenne de tout ses cours.

La vue lui propose un tableau listant tous ses cours avec leur statut (*réussi, en cours, ...*) et la moyenne des résultats par cours.

Il peut alors passer à la vue par chapitres pour les cours qu'il a commencés.

Cette vue est très proche de la précédente, mais elle affiche en plus le dernier résultat, le meilleur résultat ainsi que le nombre de fois que l'utilisateur a passé le test de fin de chapitre.

La dernière semaine était courte, car la défense du stage y prenait place, et qu'il m'a fallu préparer l'examen de télécommunication et réseaux de deuxième bloc.

J'ai terminé la mise en place des vues de statistiques en ajoutant celles liées aux cours écrits par les utilisateurs.

Celles-ci sont très proche des deux précédentes.

Désormais, l'utilisateur peut voir le nombre de cours qu'il a écrits et la moyenne générale pour tous ces cours.

La vue par cours gagne une colonne donnant le nombre de participants par cours, et la colonne statut affiche désormais si le cours est publié ou non.

Quant à la vue par chapitres, elle n'affiche plus que l'intitulé des chapitres, leur statut et le résultat moyen de tous les étudiants (*basé sur le meilleur résultat*).

J'ai terminé la phase de développement avec l'ajout de la fonctionnalité permettant aux professeurs d'inscrire une liste de personnes à l'application depuis un fichier **.csv**.

Tout comme pour l'importation d'un questionnaire, il faut passer par une fenêtre modale, et une page d'aide est disponible afin de suivre le format attendu.

Si le fichier est correct, l'importation peut s'exécuter.

Une fois cela fait, un compte rendu est affiché. Il reprend le nombre d'utilisateurs créés, ainsi que les utilisateurs non créés ainsi.



FIGURE 23 – Compte rendu de l'importation de neuf utilisateurs

La cause peut être que le nom d'utilisateur ou l'adresse email existe déjà dans la base de données.

Le professeur en est alors prévenu, et peut prendre des mesures afin que tous les utilisateurs soient inscrits.

On peut aussi remarquer que les doublons existants dans le fichier d'importation sont détectés (*l'utilisateur ArnaudUrbain n'est pas créé une seconde fois*).

Les nouveaux étudiants reçoivent alors un email les informant de l'inscription au site et leur demandant de créer un mot de passe pour leur compte.

Enfin, pour terminer le développement, je suis repassé sur le code afin d'améliorer sa structure ainsi que sa documentation.

## 5.4 Résultat

Le résultat de la phase de développement a été positif.

L'application web est fonctionnelle et permet de créer facilement un cours composé de parties théoriques et de questionnaires à choix multiples.

Elle permet aussi à l'étudiant de voir rapidement son avancement dans les cours auxquels il est inscrit au travers des vues de statistiques.

Malheureusement, j'ai été contraint de revoir mes objectifs à la baisse en retirant certaines fonctionnalités.

J'ai pu m'inspirer du stage pour la deuxième phase de développement.

J'ai organisé ces trois semaines comme des sprints, ce qui m'a permis de me mettre dans un esprit proche de celui dans lequel j'étais pendant le stage.

J'ai ainsi pu évoluer rapidement, mettre en place de nouvelles fonctionnalités et régler les éventuelles erreurs qui sont apparues.

## 6 Conclusion

### 6.1 Conclusion du projet

Le but du projet était de développer un site éducatif permettant d'apprendre et de créer des cours de tous niveaux à propos de n'importe quel sujet (langue, informatique, sciences...).

Étant donné le type de site, j'ai décidé que son thème serait lié à l'univers de l'éducation.

La page représente un tableau d'école sur lequel les aperçus des cours, et les cours, sont accrochés. Ceux-ci sont d'ailleurs affichés comme des feuilles de papier collées ou aimantées au tableau.

De plus, le design est adaptatif (*ou responsive*).

Bien que personne n'aime voir une page d'erreur, j'ai créé et personnalisé les pages d'erreurs les plus susceptibles d'apparaître afin garder une cohérence et d'aider l'utilisateur.

Un utilisateur, identifié par son nom d'utilisateur, peut modifier ses données depuis la page *Compte*.

Il peut aussi s'inscrire à des cours ou en créer, commenter ceux auxquels il est inscrit et doit achever chaque chapitre par un test.

Son résultat doit être d'au moins 7 / 10 afin que la chapitre suivant soit débloqué. Lorsque le chapitre est le dernier d'un cours, le cours est marqué comme réussi, et un trophée est ajouté sur l'aperçu du cours.

L'acquisition de trophées donne une meilleure dynamique à l'apprentissage et permet d'être certain de la bonne compréhension de chaque chapitre avant de passer au suivant.

Lorsque le cours est terminé, l'utilisateur a appris la matière du cours. L'objectif de ce cours, ainsi que celui du projet, est alors atteint.

L'étudiant peut suivre son évolution dans la vue montrant les chiffres à propos des cours auxquels il est inscrit.

C'est à cet endroit que l'utilisateur peut voir :

- le nombre de cours auxquels il est inscrit,
- le nombre de cours et de chapitres réussis,
- l'état d'avancement dans ses cours et chapitres,
- la moyenne et les résultats de ses cours et chapitres.

Si l'utilisateur crée un cours, alors il devient un *professeur*.

Il a toujours accès aux vues montrant les chiffres à propos des cours auxquels il est inscrit, mais il a aussi accès à une vue similaire reprenant les différents cours qu'il a écrits.

La vue par cours affiche désormais le nombre de participants par cours et le statut du cours, c'est-à-dire s'il est publié ou non.

Désormais, l'utilisateur peut visualiser la moyenne générale de tous ses cours, et la moyenne par cours et par chapitre.

Il peut ainsi se faire une idée de la difficulté et de la compréhension de son cours.

Afin d'ajouter facilement un cours à ses favoris ou à sa liste d'inscriptions, l'utilisateur peut cliquer sur des petites icônes placées en bas à gauche dans l'aperçu des cours.

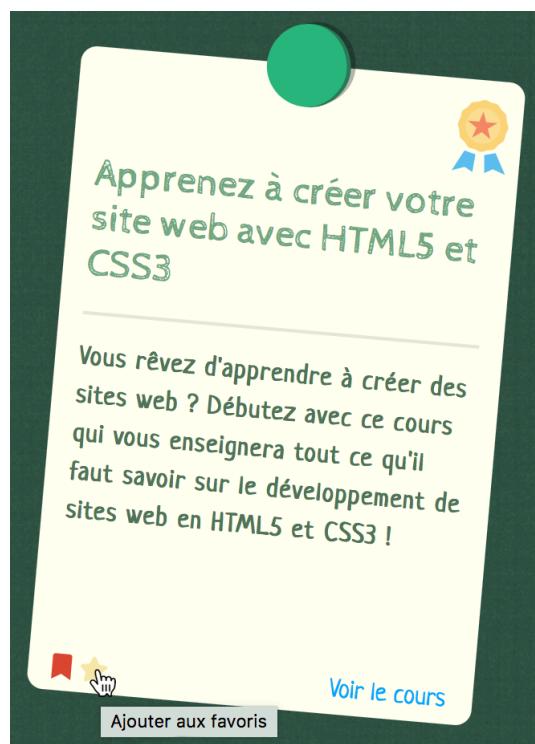


FIGURE 24 – Aperçu d'un cours terminé, à ajouter aux favoris

Afin de créer un cours, l'utilisateur doit aller sur la vue *Rédaction* à partir de laquelle il peut initialiser son cours avec un titre et une description.

Une fois cela fait, l'utilisateur peut ajouter autant de chapitre qu'il le souhaite. Un chapitre est divisé en différentes parties sous forme théorique ou de questionnaire. À la création d'un nouveau chapitre, la première partie est créée automatiquement en tant que partie théorique.

Afin de l'éditer, l'utilisateur a accès à l'éditeur de cours. Celui-ci est WYSIWYG ainsi, le professeur n'a pas à apprendre comment écrire le cours et il peut commencer tout de suite.

Pour ajouter un questionnaire, il suffit à l'utilisateur d'ajouter une nouvelle partie au chapitre, et de choisir le type : questionnaire ou test. La seule différence entre ces deux types est que, lorsque l'étudiant le complète, la correction n'est pas montrée d'office si c'est un test. De plus, l'étudiant doit pouvoir valider son questionnaire afin d'enregistrer le résultat du test, ce qui n'est pas utile dans un questionnaire simple.

Une fois cela fait, l'édition du questionnaire se fait au travers d'un formulaire composé de deux questions et de trois choix par question.

Le professeur peut facilement ajouter et supprimer des questions tant que leur nombre reste entre deux et dix.

Le questionnaire peut aussi être importé depuis un fichier .json afin de simplifier l'ajout de questionnaire.

Ensuite, il peut être édité comme tout autre questionnaire.

L'affichage du questionnaire tire parti du nombre de questions disponibles.

Ainsi, les questions sont affichées de manière aléatoire et le nombre maximal de questions à afficher est de cinq. Grâce à cela, le questionnaire sera différent à chaque fois que l'étudiant voudra le passer, ce qui évitera qu'il connaisse les réponses par cœur.

Pour finir, le professeur peut aussi importer une liste d'utilisateurs au format .csv, afin d'enregistrer rapidement des étudiants.

Une fois l'importation terminée, un rapport est affiché. Celui-ci affiche le nombre d'utilisateurs importés, et affiche ceux qui ne l'ont pas été avec une explication sur le pourquoi de l'erreur à l'import.

Les erreurs possibles sont un problème d'unicité sur le nom d'utilisateur ou sur l'adresse mail.

Les utilisateurs inscrits recevront un email d'information les prévenant, et leur demandera de créer un mot de passe pour leur compte.

Du côté du back-end, j'ai utilisé Laravel qui se base sur d'autres frameworks pour développer son propre système et être plus efficace.

Voici quelques un des outils qu'il propose :

- Un système de routage des vues
- Un créateur de requêtes SQL et une gestion de version de base de données
- Un ORM performant
- Un système d'authentification pour les connexions
- Un système de cache
- Une gestion de sessions

L'authentification intégrée de Laravel fait merveille : elle permet à un utilisateur de créer un compte, de s'y connecter, mais aussi de réinitialiser son compte à l'aide d'un token envoyé par email.

Il faut donc lier un serveur mail à Laravel.

En local, j'ai choisi d'utiliser **MailHog** et, sur le serveur de test, j'ai utilisé **Mailgun**. Il a suffit de quelques modifications dans le fichier d'environnement de Laravel pour que le serveur y soit lié.

La session d'un utilisateur est gérée nativement, mais la durée d'inactivité avant déconnexion peut être personnalisée. J'ai décidé de garder la durée par défaut qui est de deux heures.

Comme la majorité des frameworks web, Laravel se base sur le design pattern MVC qui permet de séparer l'application web en trois couches : le Modèle, la Vue et le Contrôleur.

Ce design pattern est destiné aux interfaces graphiques. L'un de ses avantages est la clarté de l'architecture qu'il apporte et qui simplifie la tâche du développeur qui tenterait d'effectuer une maintenance ou une amélioration sur le projet.

Laravel m'a permis de créer la base de données très rapidement grâce aux migrations. Celles-ci permettent de créer et de mettre à jour une base de données en créant et supprimant des tables, des colonnes dans ses tables... depuis des fichiers .php reprenant les différents champs d'une table à créer.

## 6.2 Améliorations possibles

Premièrement, les quatre fonctionnalités n'ayant pas été développées sont des améliorations à ajouter.

Celles-ci sont :

- *Générer un PDF depuis un/des chapitre(s) choisi(s)*, cela permettrait au professeur de réutiliser les QCM lors d'exercices ou d'examens sur papier.
- *Différents niveaux de difficulté d'après l'âge ou le niveau de l'utilisateur*.
- *Choix de la difficulté des questions, ainsi que gestion de leur pondération*, celle-ci est fortement liée à la précédente et ajoute la possibilité d'avoir différentes pondérations par question.
- *Créer une application mobile Android pour un accès hors-ligne*.

Ces fonctionnalités non pas été implémentée par manque de temps.

Lors de la reprise du développement après la fin du stage, j'ai travaillé avec une méthode proche de la méthode Agile, et j'ai dû faire un choix sur les fonctionnalités à développer.

Trois options se sont alors offertes à moi.

Premièrement, continuer le développement en essayant d'en faire le plus possible le plus rapidement possible.

Cette option aurait diminué la qualité globale de l'application et du code, et ne garantissait pas le développement complet des fonctionnalités de base.

Deuxièmement, développer toutes les fonctionnalités restantes dans le temps imparti (*3 semaines*).

En plus de faire baisser la qualité du code et de l'application, cette option était fortement impossible à tenir.

Troisièmement, développer les fonctionnalités les plus importantes, et celles pouvant être développées en plus dans le temps imparti.

Cette option permettait de garder un bon niveau de qualité, tout en garantissant que l'application finale serait complètement utilisable.

C'est donc la troisième option que j'ai choisie, car c'était la plus intéressante que ce soit du côté de l'utilisateur, ou du côté du développeur.

Trois autres améliorations me viennent ensuite.

La première étant d'utiliser **Sass** (*Syntactically Awesome Stylesheets*) au lieu du CSS. Il permet de créer des constantes et des variables pour, par exemple, n'avoir à définir qu'une seule fois des valeurs utilisée à plusieurs endroits.

L'avantage premier est de pouvoir changer rapidement le rendu du site en ne changeant, par exemple, les couleurs du thèmes qu'à un seul endroit.

En outre, le maintien du code est simplifié, car il peut être divisé en différents fichiers. Ils peuvent alors être rassemblés en un seul fichier lors de la compilation du code en CSS.

La deuxième serait d'utiliser une **API REST**.

Par exemple, certaines modifications des données du modèle, comme l'ajout d'un cours dans les favoris, demande un rafraîchissement de la page afin que la modification ait lieu.

L'utilisation de l'API, en passant par des appels ajax, permettrait d'effectuer ces modifications sans recharger les pages, leur fonctionnement étant asynchrone.

Un autre avantage, est que, grâce à l'API REST, une application mobile pourrait tirer parti de l'application web.

En effet, si l'application mobile doit récupérer un chapitre à afficher en lecture seule, l'API permettrait de le récupérer facilement en un seul appel.

On peut aussi imaginer la possibilité de donner l'accès à l'API à des applications tierces.

La troisième et dernière amélioration serait d'ajouter le support des images dans l'éditeur de cours.

L'ajout d'images entraîne leur stockage sur le serveur, et donc une réflexion de où et comment les stocker de manière sécurisée et optimale.

### 6.3 Apports du TFE

Grâce à mon stage, j'ai eu la possibilité d'améliorer l'analyse du projet, ce qui m'a permis de régler certaines erreurs et de bien préparer la seconde phase de développement.

En outre, j'ai appris à utiliser de nouvelles technologies et pratiques qui m'ont permis de mener le projet à bien.

Comme l'utilisation de Laravel qui m'a servi à mieux comprendre le fonctionnement du design pattern MVC, ainsi que le fonctionnement des frameworks PHP en général.

## 6.4 Conclusion

Ce travail m'a permis d'apprendre à utiliser de nouvelles technologies et à travailler sur un même projet pendant une longue période : une année scolaire.

Bien sûr, le développement n'a pas été continu. Entre les différents projets, les examens et le stage, j'ai dû le mettre de côté à plusieurs reprises.

C'est ainsi que la phase de développement a été divisée en deux parties : avant et après le stage.

Après la phase d'analyse du stage (*les trois premières semaines*), je n'ai plus eu de temps pour travailler sur le TFE. Par contre, pendant mes recherches liées au stage, j'ai engrangé des informations et outils utiles au développement de mon travail de fin d'études.

J'y ai donc pensé continuellement.

Une fois le stage et son rapport terminés, j'ai pu améliorer l'analyse que j'avais effectuée auparavant grâce aux connaissances que j'ai acquises lors du stage chez Realdolmen.

Cette période de travail non conventionnelle n'a pas été de tout repos, surtout lors de la phase finale du développement.

J'ai pu me baser sur mes acquis personnels ainsi que ceux assimilés pendant mon stage et mon cursus scolaire, et les faire passer au niveau supérieur afin de mener à bien mon travail de fin d'études.

Je suis bien content du choix de mon TFE, car il pourra servir tant dans un cadre professionnel que dans un cadre scolaire ou même personnel.

# Bibliographie

---

## Livres

- 
- [1] LAMPORT, L. A Document Preparation System.  
(consulté du 18 Septembre 2017 au 2 Janvier 2018).
  - [2] MANDOUX, D. Guide à la réalisation du TFE.  
(consulté du 18 Septembre 2017 au 2 Janvier 2018).
  - [3] MILLER, I. Développement front-end.  
(consulté du 15 Octobre au 25 Novembre 2017).
  - [4] MILLER, I. Internet et Multimédia.  
(consulté du 15 Octobre au 25 Novembre 2017).

---

## Internet

---

- [5] BESTMOMO. Laravel 5.3 plus loin le déploiement, *informatique*, Site, [en ligne].  
<https://laravel.sillo.org>, (consulté entre octobre et décembre 2017).
- [6] CHAVELLI, M. Découvrez le framework PHP Laravel, *informatique*, Site, [en ligne]. <https://openclassrooms.com>, (consulté le 28 Décembre 2017).
- [7] GERCHEV, I. The 5 Most Popular Front-end Frameworks Compared, *informatique*, Site, [en ligne].  
<https://www.sitepoint.com>, (consulté le 5 Juin 2018).
- [8] HACKER NOON. Top 5 Most Popular CSS Frameworks that You Should Pay Attention to in 2017, *informatique*, Site, [en ligne].  
<https://hackernoon.com>, (consulté le 5 Juin 2018).
- [9] LAGACÉ, C. Les routes avec Laravel, *informatique*, Site, [en ligne].  
<https://christianelagace.com>, (consulté le 5 Juin 2018).
- [10] LAGACÉ, C. Quest-ce que MVC ? un exemple concret avec Laravel, *informatique*, Site, [en ligne].  
<https://christianelagace.com>, (consulté le 5 Juin 2018).
- [11] THE ANDTHEDESIGN COMMUNITY. Le php oui, mais avec Laravel, *informatique*, Site, [en ligne].  
<https://www.anthedesign.fr>, (consulté le 5 Juin 2018).
- [12] THE LARACAST COMMUNITY. Laravel 5.4 From Scratch, *informatique*, Site, [en ligne].  
<https://laracasts.com>, (consulté entre octobre et décembre 2017).
- [13] THE WIKIPEDIA COMMUNITY. Cross-Site Request Forgery, *informatique*, Site, [en ligne].  
<https://fr.wikipedia.org>, (consulté le 2 Janvier 2018).
- [14] THE WIKIPEDIA COMMUNITY. Bootstrap (framework), *informatique*, Site, [en ligne]. <https://fr.wikipedia.org>, (consulté le 27 Décembre 2017).
- [15] THE WIKIPEDIA COMMUNITY. Fonction de hachage cryptographique, *informatique*, Site, [en ligne].  
<https://fr.wikipedia.org>, (consulté le 27 Décembre 2017).
- [16] THE WIKIPEDIA COMMUNITY. Laravel, *informatique*, Site, [en ligne].  
<https://fr.wikipedia.org>, (consulté le 27 Décembre 2017).
- [17] THE WIKIPEDIA COMMUNITY. PHP, *informatique*, Site, [en ligne].  
<https://fr.wikipedia.org>, (consulté le 4 Juin 2018).

Ducobu Alexandre  
BA3 Informatique & systèmes  
Orientation réseaux & télécommunications  
Développement

---

# Cahier des charges du TFE

## CRÉATION D'UN SITE ÉDUCATIF

- **Description brève :**

Découvrir de nouvelles matières comme une langue, un langage de programmation, etc.

- **Description détaillée :**

Le site permettra d'apprendre et de créer des cours de tous niveaux à propos de n'importe quel sujet (langue, informatique, sciences,...).

Le site proposera, en tant qu'exemples, différents cours composés de théorie et d'exercices.

Il fournira aussi un outils de création de cours afin que les utilisateurs puissent proposer de nouveaux sujets.

L'apprentissage se fera en trois étapes :

1. L'utilisateur découvrira le nouveau sujet par de la théorie ainsi que par un ou plusieurs exemples. Il en apprendra alors l'utilité et le fonctionnement.
2. Entre deux parties théoriques, l'utilisateur mettra en pratique ce qu'il aura appris au travers de petits QCM.
3. Une fois le chapitre terminé, un questionnaire (QCM, ordonnancement du code,...) sera proposé à l'utilisateur. Celui-ci sera noté sur 10 afin que l'utilisateur puisse se juger et s'améliorer.  
Le passage au chapitre suivant requerra une cote minimale de 7/10.

- **Fonctionnalités :**

- Apprentissage du cours
- Exercices pour fixer la matière
- Trophées de réussite de chapitres et du cours
- Statistiques pour chaque étudiant ainsi que pour le créateur du cours
- Enregistrement depuis un fichier .csv et envoi automatique de mails dans ce cas
- Différents QCM disponibles, dont certains seront choisis au hasard (*aussi par import d'un fichier*)
- Possibilité de cacher des questions et de générer un PDF depuis toutes les questions
- Possibilité de générer un questionnaire d'après un/des chapitre(s) choisi(s)
- Correction des QCM personnalisée d'après la réponse (*semi-automatique*)
- Choix de la difficulté des questions, ainsi que gestion de leur pondération
- Différents niveaux de difficulté d'après l'âge ou le niveau de l'utilisateur
- Création d'une application mobile *Android* pour un accès hors-ligne

---

Signature de l'étudiant

Signature de l'enseignant

# Table des annexes

<b>Annexes</b>	<b>44</b>
Les maquettes . . . . .	45
Le site . . . . .	49

# Les maquettes

Voici les différentes maquettes du site :



FIGURE 25 – La page d'accueil

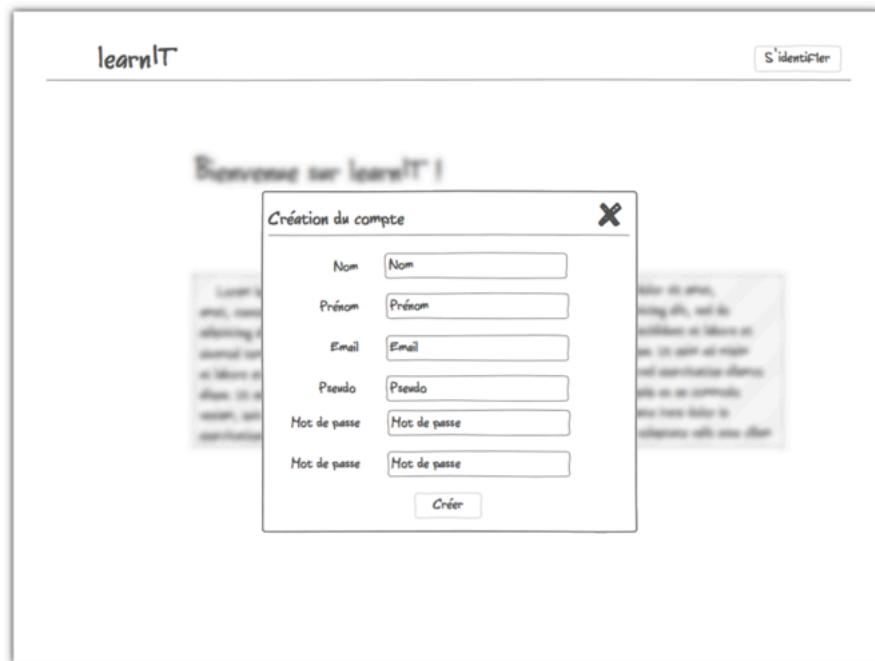


FIGURE 26 – La page d'inscription

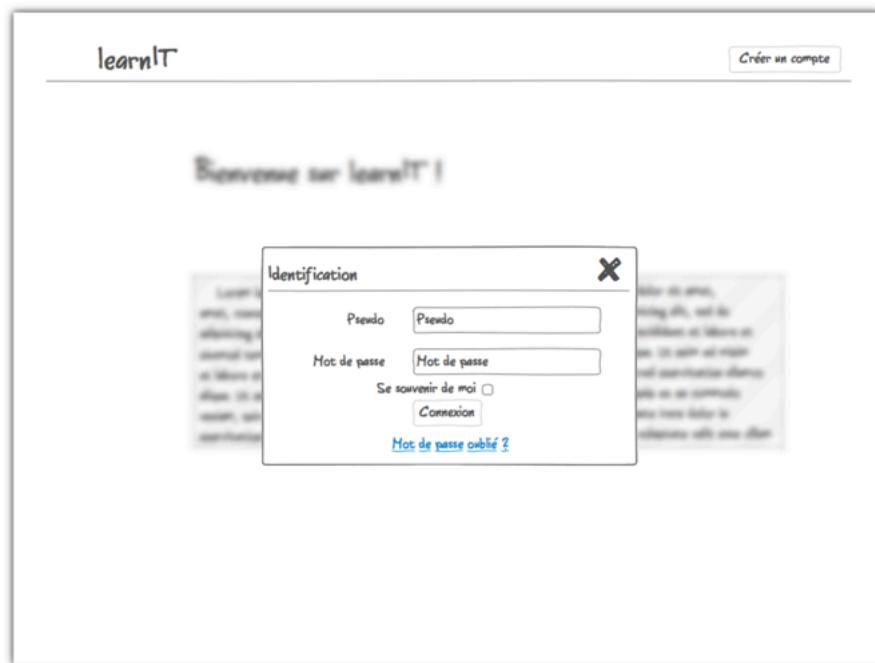


FIGURE 27 – La page de connexion

The screenshot shows the learnIT website's course list page. At the top, there is a navigation bar with links for "learnIT", "Cours", "Mes cours", "Mes chiffres", "Écrire un cours", "Chercher un cours" (with a search icon), "Compte", and "Déconnexion". The main content area has a large "Cours" title and a "Catégories" section. Under "Catégories", there is a checked checkbox for "Toutes" and three unchecked checkboxes for "Catégorie 1", "Catégorie 2", and "Catégorie 3". Below this, there are two sections: "Catégorie 1" and "Catégorie 2", each containing three course cards. Each card displays a small image of a landscape, some placeholder text ("Lorem ipsum dolor sit amet, consectetur adipiscing elit."), and a small blue ribbon icon with a yellow star on the card in the "Catégorie 1" section.

FIGURE 28 – La liste des cours

FIGURE 29 – La liste des cours auxquels l'utilisateur est inscrit

Cours	Status	Résultat
Cours 1	Réussis 🎓	97%
Cours 2	En cours	30%
Cours 3	En cours	42%
Cours 4	En cours	78%
Cours 5	En cours	90%

FIGURE 30 – La page des chiffres

**Titre :** Titre du cours

**Catégorie :** Choisir une catégorie

**Résumé :** Résumé du cours

**Aperçu du cours :** Sélectionner une image | Choisir...

Ajouter un chapitre

Sauvegarder | Annuler

FIGURE 31 – La page de création d'un cours

Votre email : prénom.nom@heb.be

Nouvel email :  Sauver

Changer de mot de passe

Ancien :  Nouveau :

Réécrire le nouveau :  Sauver

Supprimer votre compte

Adieu connaissances!

FIGURE 32 – La page du compte utilisateur

# Le site

Voici les différentes pages du site finalisé :

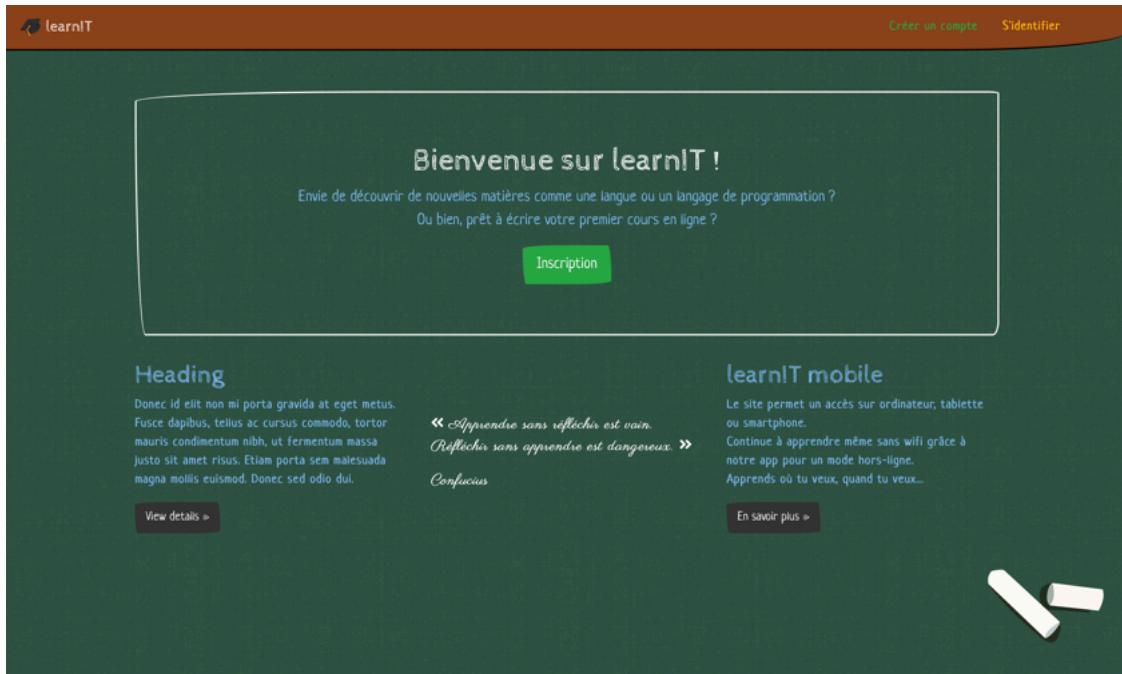


FIGURE 33 – La page d'accueil

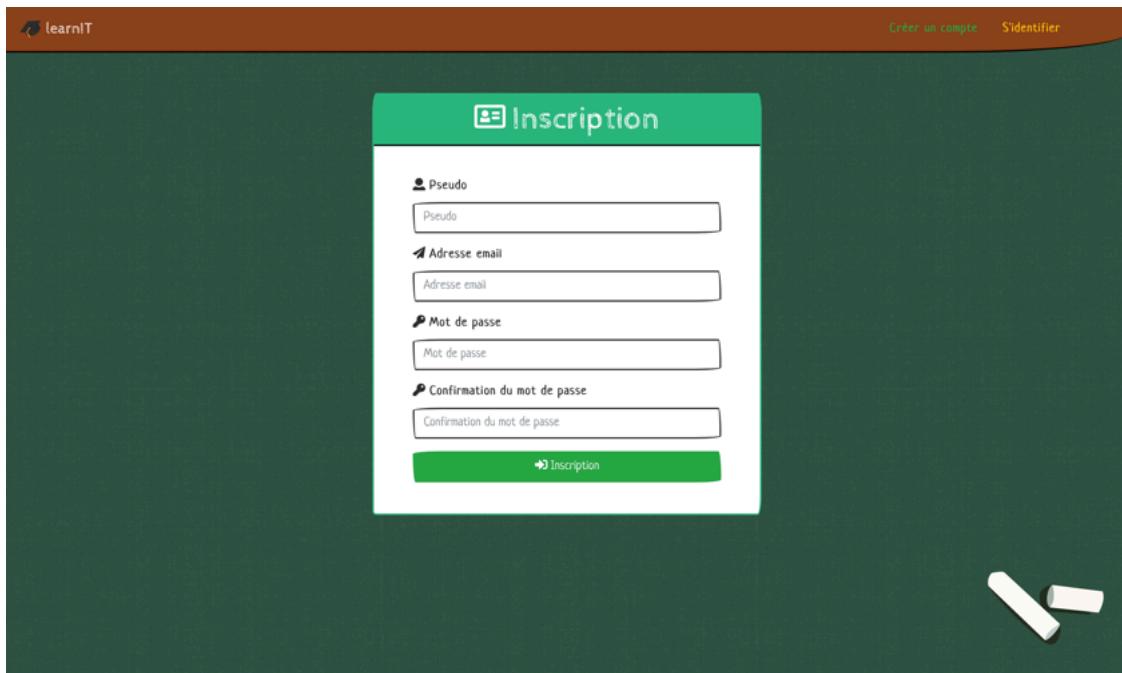


FIGURE 34 – La page d'inscription

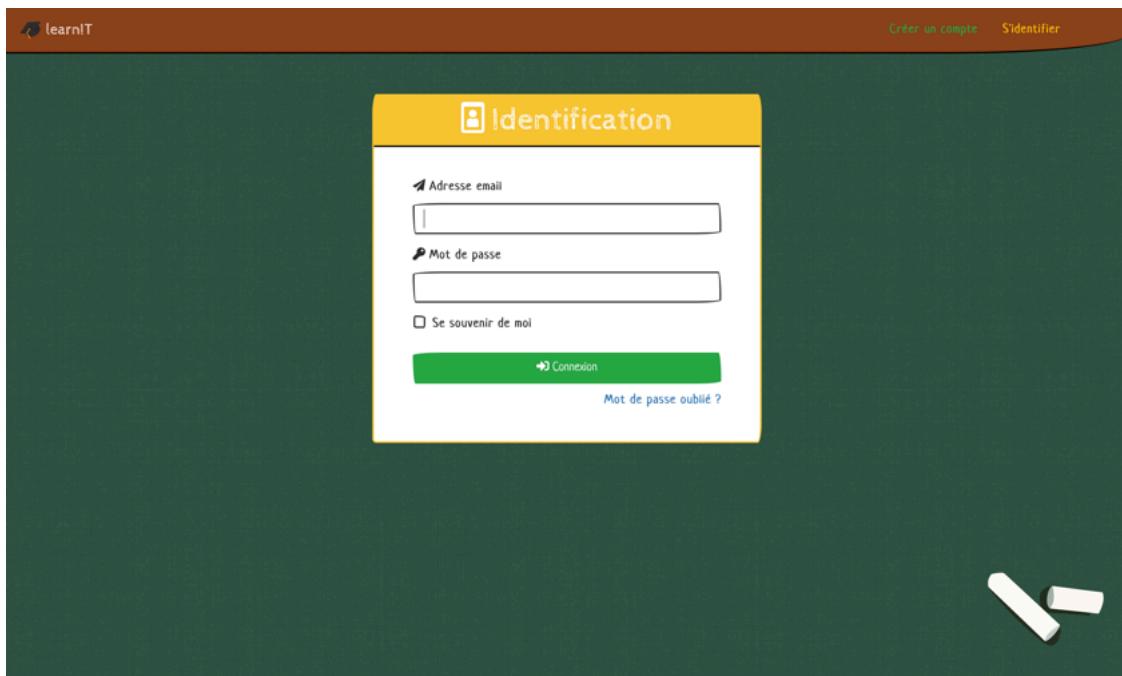


FIGURE 35 – La page de connexion

A screenshot of the learnIT platform's course list page. The header includes navigation links for 'Cours', 'Mes cours', 'Mes chiffres', 'Écrire un cours', 'Architect', and 'Déconnexion'. The main content is titled 'Tous les cours' and displays six course cards arranged in a grid. Each card has a title, a brief description, a star rating, and a 'Voir le cours' (View course) button. The courses listed are: 'Apprenez à créer votre site web avec HTML5 et CSS3', 'Apprenez à programmer en Python', 'Concevez un site avec Flask', 'Concevez votre site web avec PHP et MySQL', 'Découvrez le framework Django', and 'Découvrez le framework PHP Laravel'. The descriptions provide details about the programming languages and frameworks used in each course.

FIGURE 36 – La liste des cours

FIGURE 37 – La liste des cours auxquels l'utilisateur est inscrit

Cours	Statut	Résultat
Apprenez à créer votre site web avec HTML5 et CSS3	Réussi	-
Apprenez à programmer en Python	En cours	100 %
Concevez un site avec Flask	En attente	-
Découvrez le framework PHP Laravel	En attente	-

FIGURE 38 – La page des chiffres

The screenshot shows the 'Réécriture de cours' (Course Rewrite) section of the learnIT platform. At the top, there is a navigation bar with links: 'learnIT', 'Cours', 'Mes cours', 'Mes chiffres', 'Rédaction', 'Harchytek', and 'Déconnexion'. Below the navigation, the title 'Réécriture de cours' is displayed. There are three course cards visible:

- Apprenez à coder avec JavaScript**: A card with a large plus sign icon. Description: "Program or be programmed": avoir la capacité de comprendre, créer ou modifier des logiciels permet de devenir acteur du monde numérique qui nous entoure. Ce cours vous donnera les bases pour programmer avec le langage standard du Web : JavaScript." Action button: 'Éditer le cours'.
- Apprenez à programmer en Java**: Description: Java est un langage extrêmement populaire utilisé dans un grand nombre d'entreprises. Très structuré et mature, il permet de créer aussi bien des programmes que des applications web. Sa grande force ? "Write once, run everywhere": un même code fonctionnera partout ! Action button: 'Éditer le cours'.
- Découvrez la programmation orientée objet avec Python**: A card with a green circle icon. Description: [not fully visible]

FIGURE 39 – La page de création d'un cours

The screenshot shows the 'Votre compte' (Your Account) section of the learnIT platform. At the top, there is a navigation bar with links: 'learnIT', 'Cours', 'Mes cours', 'Mes chiffres', 'Rédaction', 'Harchytek', and 'Déconnexion'. Below the navigation, the title 'Votre compte' is displayed. The account information for 'Harchytek' is shown, including the last connection time ('Dernière connexion il y a 42 secondes') and the date of inscription ('Inscription le Jeudi 01 février 2018'). The profile picture is a placeholder icon. The account name is 'Alexandre Ducobu, professeur(e)'.

**Changer votre adresse email**  
Adresse email actuelle : [harchytekt@yahoo.fr](mailto:harchytekt@yahoo.fr)  
  
Enregistrer la nouvelle adresse email

**Changer votre mot de passe**  
  
  
  
Enregistrer le nouveau mot de passe

FIGURE 40 – La page du compte utilisateur

