Rapport de projet : DAW

Camille DARDOIZE, Wassim DJELLAT, Gwendal LOTTIN
Matthieu JOULAIN, Jonathan MARTIN–MAESTRE, Yann TROU

MAI 2021

Table des matières

Chapitre 1

Introduction

Dans le cadre d'un projet pour notre L3 Informatique, à l'Université de Bourgogne Franche-Comté, nous avons dû créer de A à Z un site internet. Celui-ci est composé de deux parties : une partie administrateur et une partie apprenant. Depuis ces deux parties, nous avons accès à différentes pages : page d'accueil, page des cours, page du forum.

Pour chaque partie, une page de profil est dédiée (une pour les étudiants, une pour les admins/enseignants) avec des informations uniques dedans, correspondants au compte connecté.
Depuis leur profil, les étudiants peuvent modifier leurs informations, ont accès aux cours qu'ils suivent et également à leurs recommandations personnalisées suite aux résultats des QCM.
Les admins/enseignants ont accès, depuis leur profil, aux cours qu'ils ont postés, mais également en ajouter (cours et QCM). Ils peuvent aussi modifier leurs informations et gérer les étudiants.

Tous ensemble, nous avons fait une première réunion pour mettre les idées du site en place.
Nous avons partagé nos idées, que ce soit par rapport aux visuels (header, accueil) ou aux fonctionnalités, et également fait un point sur la méthodologie Scrum afin de bien pouvoir l'appliquer.
Nous avons décidé de nous séparer en deux groupes distincts, afin d'avancer de manière rapide et optimisée : une partie FRONT-END et une partie BACK-END.
Il est à noter que les groupes s'aident également entre eux.

Cela nous a permis d'avancer simplement, et donc d'éviter de mettre quelqu'un sur des tâches ne lui correspondant pas.

Tous ensemble, nous avons réalisé le Logo. Le logo a été réalisé sur Photoshop, reprenant alors le nom de notre site internet "Web School", ainsi qu'une illustration d'un gorille car cet animal représente la sagesse et la détermination, mais c'est également un hommage à Harambe, un gorille qui a été tué dans sa cage car un enfant y avait pénétré.

Ci-dessous, une présentation des différentes fonctionnalités et contraintes imposées :

1. Partie Administrateur :
   - Charger les cours sous forme de diapos, vidéos ..
   - Gérer les utilisateurs (création, modification, suppression)
   - Gérer les QCM.
2. Partie apprenant : espace personnel, gestion des cours
   - Construction du profil de l'apprenant : des QCM sont proposés afin définir le niveau de l'apprenant afin de lui proposer des cours
   - Recommandation de cours
   - Un forum de discussion entre les apprenants
3. Consignes techniques
   - Utiliser une architecture MVC
   - Utilisation des sessions / cookies
   - Les QCM devront être en XML
   - Le site devra comporter 2 thèmes CSS

Chapitre 2

Gestion du projet

2.1 Méthode SCRUM

Cette partie est dédiée à l'explication de l'organisation et de la répartition du travail entre les différents membres du groupe de projet réalisée avec la méthodologie SCRUM.
SCRUM est une méthodologie faisant partie des méthodes Agiles.
Les principes de la méthodologie Agile correspondent à l'implication et à la participation active du client tout au long du projet.
N'ayant pas de client véritable, le Product Owner jouera ce rôle, sachant qu'il est normalement censé répondre aux attentes des utilisateurs et aux besoins du client.
Scrum se compose de divers éléments, comme des rôles, des événements, des artefacts mais également des règles.

Scrum Master : cette personne est responsable de la compréhension ainsi que de la mise en œuvre de la méthode Scrum. C'est une sorte de coach qui va organiser la communication et la productivité au sein de l'équipe.

Product Owner : cette personne donne les instructions aux développeurs, va établir les priorités des fonctionnalités...

Équipe de développeurs : ce sont les développeurs qui vont travailler sur le projet.

Nous sommes tous des développeurs, mais certains ont un rôle supplémentaire : ***Yann TROU*** est Product Owner et ***Camille DARDOIZE*** est Scrum Master.

L'application de nos rôles :

— Product Owner :

Définition des contraintes du sujet dans un cahier des charges et une documentation technique compréhensible.
Attribution des tâches aux développeurs.
Séparation de l'équipe en deux afin d'optimiser le temps de travail.
Réunions régulières afin de préciser les détails qui n'étaient pas clairs.
Vérification de la conformité du produit par rapport aux consignes demandées.

— Scrum Master :

Mise en place d'un sprint durant 2 semaines (du 12 avril au 26 avril, période de vacances). Pendant ce sprint, diverses fonctionnalités à réaliser en un temps donné ont été décrites.

A la fin de ce sprint, a été fait une revue du sprint. Cela a permis de valider ce qui a été accompli pendant le sprint. Également, une rétrospective du sprint a été faite. Nous avons vu ce que nous pouvions améliorer dans nos fonctionnalités faites pendant le sprint, concernant le Scrum Master, cela correspondait au design du site internet.

Une fois par semaine, une mêlée quotidienne, adaptée par semaine, a été réalisée. Pendant cette réunion, nous faisions le point entre les deux groupes, les "back-ends" et les "front-ends" : qu'avons-nous réalisé cette semaine, qu'avons-nous à faire la semaine prochaine, ce qui nous bloque, les choses à adapter entre les deux groupes...

Cependant, plusieurs fois par semaine, une mêlée quotidienne a été établie entre les membres du "sous-groupe". Pour le front-end, cela était le Lundi, Mardi, Vendredi et le Samedi, entre 4h et 8h à chaque fois. Pour le back-end, cela était le Dimanche, Mercredi, Jeudi et parfois le Lundi entre 3h et 5h. A chaque fin de mêlée, une liste de fonctionnalité/pages à faire lors de la séance suivante été écrite afin de ne pas perdre de temps. Les visuels ont été réalisés au préalable afin d'être le plus productif possible sans perdre de temps à réfléchir dessus, même si certaines modifications et améliorations ont été faites au cours du temps.

Pendant les temps libres, chacun avançaient tout de même seuls, en prévenant les autres de ses avancées.

2.2 Front-end

Les front-end se sont occupés de la partie : PHP, HTML, CSS, JS (et JQUERY).

Ce groupe est composé de :

— Camille DARDOIZE
— Jonathan MARTIN–MAESTRE
— Gwendal LOTTIN

Afin de partir sur de bonnes bases, nous avons tout d'abord créé tous les visuels à l'aide d'un site d'infographie (Piktochart). Au fur et à mesure de la construction de notre site, pour améliorer certaines pages, nous avons légèrement modifié les visuels.

Nous avons décidé de partir sur un site sobre, épuré mais également avec des illustrations personnalisées (montage) afin d'apporter des touches de couleurs.

Light mode

Pour le thème de base, le "Light mode", nous sommes parti sur du bleu et du orange. Le bleu est une couleur très utilisée dans le domaine de l'informatique et des nouvelles technologies, c'était donc une évidence pour nous de l'utiliser, de plus, elle symbolise l'intelligence.
La couleur orange symbolise l'ambition, elle correspond donc très bien aux valeurs que nous voulons transmettre aux étudiants.

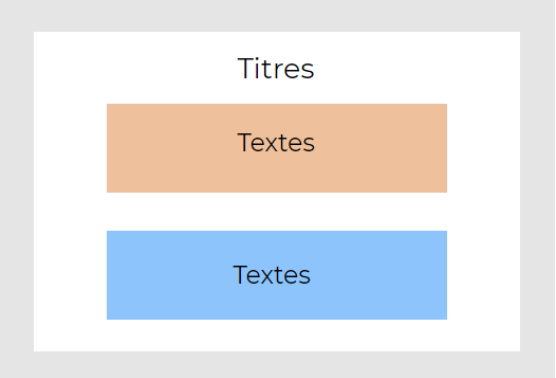

FIGURE 2.1 – Thème "Light"

Dark mode

Concernant le deuxième mode, nous avons voulu partir sur un mode sombre, le "Dark mode", avec du gris/noir en fond et des touches de jaune et de vert, apportant un touche de lumière, tout en restant dans le thème sombre grâce au vert foncé.

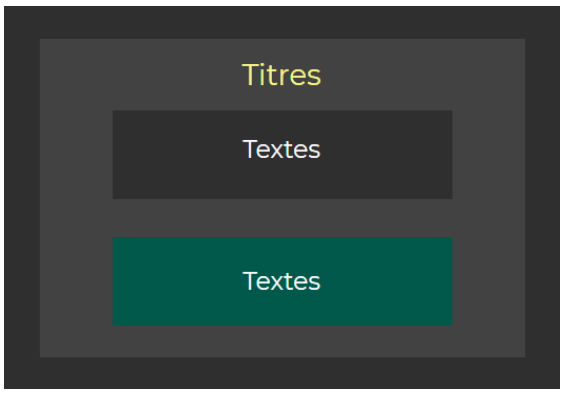

FIGURE 2.2 – Thème "Dark"

Gwendal LOTTIN ayant une grande expérience dans le domaine et ayant rencontré la plupart des problèmes auxquels nous avons finalement fait face, nous avons décidé de séparer notre groupe en deux sous-groupe pour être plus efficace, un trinôme allant ralentir notre progression. Nous avons opté pour l'organisation suivante :

1. Un binôme composé de Camille DARDOIZE & Jonathan MARTIN–MAESTRE.
2. Un monôme composé de Gwendal LOTTIN.

Remarque 2.2.1 *Il est également à noter que Gwendal a aussi aidé au niveau du Back-end (aide pour l'affichage des messages et diverses retouches pour faciliter l'intégration Front-end).*

Nous nous sommes ainsi organisés de cette manière :

Nous avons utilisé Trello afin de lister toutes nos tâches et savoir qui devait faire quoi. Également, sur un word partagé nous écrivions les tâches plus en détail, les choses à corriger, le rappel des couleurs utilisées, les TO-DO-LIST des prochaines réunions... Cela nous a donc donné cette organisation-là :

FIGURE 2.3 – Organisation et répartition Trello de Jonathan MARTIN–MAESTRE

FIGURE 2.4 – Organisation et répartition Trello de Camille DARDOIZE

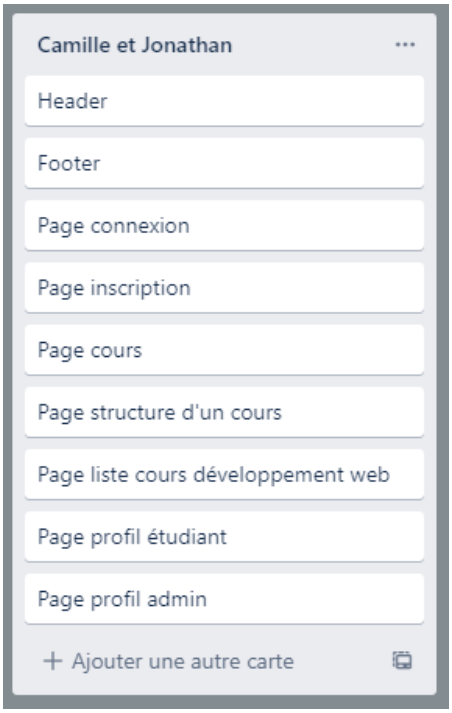

FIGURE 2.5 – Organisation et répartition Trello de Camille DARDOIZE et Jonathan MARTIN–MAESTRE

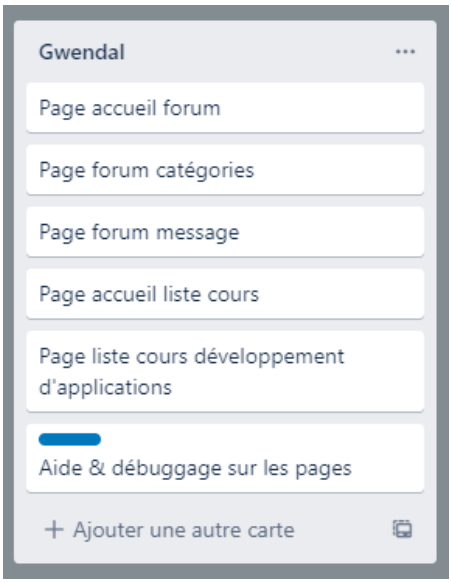

FIGURE 2.6 – Organisation et répartition Trello de Gwendal LOTTIN

2.3 Back-end

Tout d'abord pour le Back-end nous étions un groupe de 3 composés de :
— Matthieu JOULAIN
— Yann TROU
— Wassim DJELLAT
Comme mentionné précédemment, Gwendal LOTTIN aidait aussi à faire le lien entre le back-end et le front-end à certains moments du projet. Nous avons commencé la base de données qui était

le coeur de notre projet puis nous nous sommes occupés du reste en nous divisant les charges. L'utilisation d'un Trello nous a permis de lister les fonctionnalités à faire pour chacun. Nous nous sommes ainsi divisés les tâches en différentes parties disponibles ci-dessous

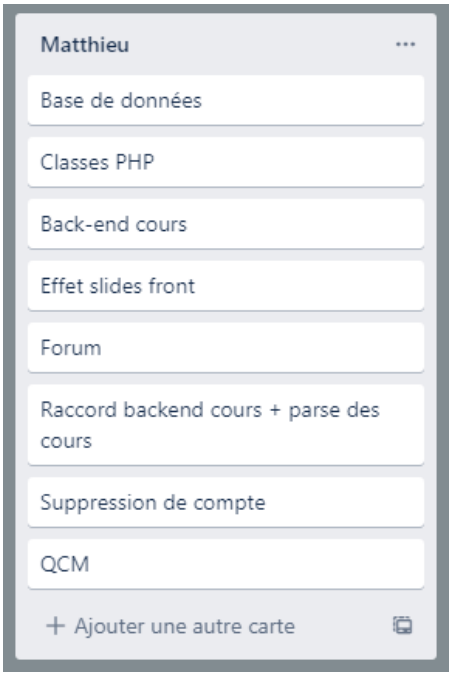

FIGURE 2.7 – Organisation et répartition Trello de Matthieu JOULAIN

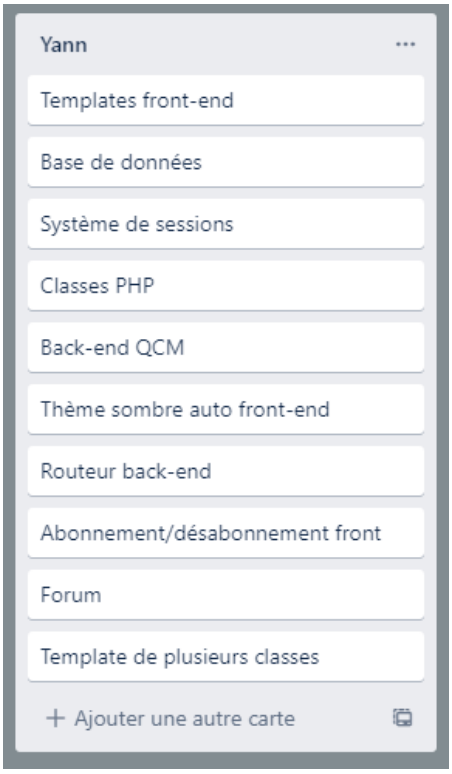

FIGURE 2.8 – Organisation et répartition Trello de Yann TROU

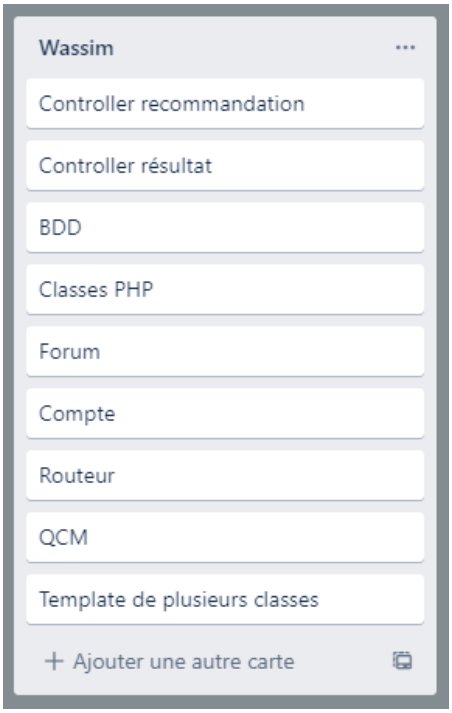

FIGURE 2.9 – Organisation et répartition Trello de Wassim DJELLAT

Chapitre 3

Modélisation

Il est important de noter que certaines fonctionnalités et diagrammes qui ont été dans la première partie du développement de l'application peuvent différer du rendu final des fonctionnalités implémentées, cependant, tout au long du projet nous avons au plus possible fait en sorte de nous tenir à notre modélisation initiale.

3.1 Diagramme Use-Case

Ci-dessous, notre diagramme use-case de premier niveau, nous avons cependant détaillé plus en détail certaines fonctionnalités ;

- Admin :
  - Gestion des cours
  - Gestion du forum
  - Gestion des évaluations
  - Suppression d'un utilisateur
- Etudiant :
  - Accès au forum
  - Gestion de l'abonnement aux cours
  - Visualisation des cours / évalutions / recommandations
  - Création d'un compte
  - Modification du compte

FIGURE 3.1 – Diagramme Use Case de niveau 1

3.2 Base de données

Nous avons décidé de représenter la base de données à l'aide du diagramme ci-dessous qui est découlé fortement de notre diagramme Use Case de niveau 1 de ci-dessus afin de connaître les différentes fonctionnalités à implémenter.

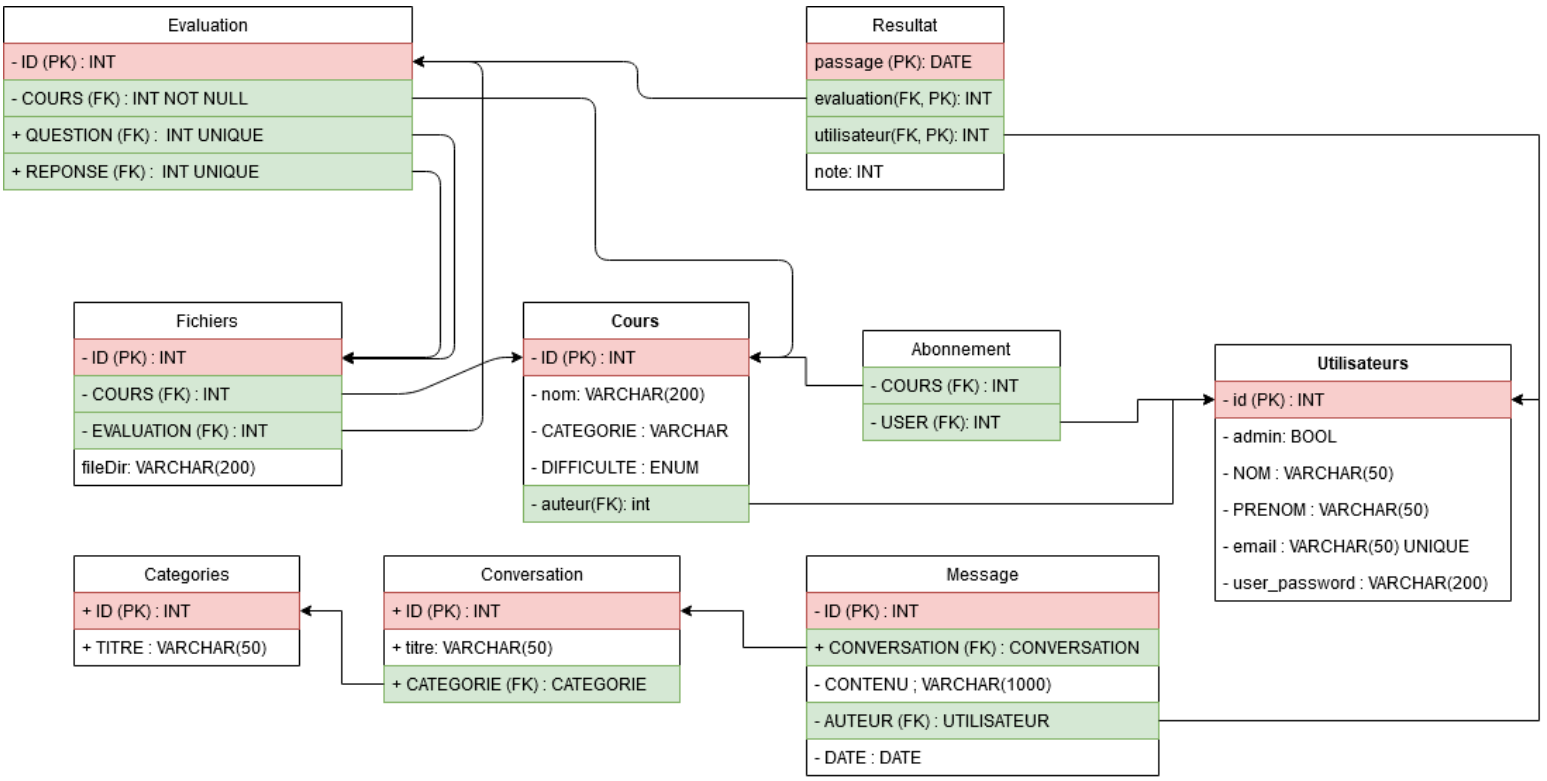

FIGURE 3.2 – Diagramme de la base de données

3.2.1 Relation entre les classes

On se retrouve alors avec 9 classes pouvant être modélisées ainsi, on souligne les clés primaire et les classes ayant un lien avec d'autres classes. ;

— Evaluation(**ID**, Cours, Question, Reponse)
— Resultat(**Passage**, utilisateur, evaluation, note)
— Fichiers(**ID**, Cours, Evaluation, fileDir)
— Cours(**ID**, Nom, Categorie, Difficulte, Auteur)
— Abonnement(**Cours**, utilisateur)
— Utilisateurs(**ID**, Admin, Nom, Prenom, Email, Password)
— Message(**ID**, Conversation, Auteur, Contenu, Date)
— Conversation(**ID**, Catégorie, Titre)
— Categories(**ID**, Titre)

3.3 MVC

Nous sommes parti sur une architecture "MVC".

"MVC" ou "Model-View-Controller" est un style d'architecture de logiciel très populaire, permettant d'être efficace et structuré lors du développement d'un projet.

Les trois composantes importantes du MVC sont : modèle, vue, contrôleur.

- Vue : c'est un moyen d'afficher des objets dans une application (affichage d'une fenêtre, des boutons, d'un texte...). En sommes, c'est tout ce que l'utilisateur peut voir, c'est donc l'interface utilisateur. La vue permet à l'utilisateur d'afficher les données à l'aide d'un modèle et lui permet également de modifier les données.

- Modèle : contient les données utilisées par un programme (base de données, fichier...). Par exemple, un objet Client récupérera les informations de la base de données, les manipulera et mettra à jour ses données dans la base de données.

- Contrôleur : ils agissent comme une interface entre le modèle et la vue, pour traiter toute la logique métier et les requêtes entrantes, manipuler les données à l'aide du composant Modèle et interagir avec les Vues pour rendre le résultat final. Par exemple, le contrôleur « Client » va traiter toutes les interactions et les entrées de la Vue « Client » et mettre à jour la base de données en utilisant le Modèle « Client ». Le même contrôleur sera utilisé pour visualiser les données du client.

Les trois parties du MVC sont inter-connectées. La vue affiche le modèle pour l'utilisateur. Le contrôleur accepte les entrées de l'utilisateur et met à jour le modèle et la vue.

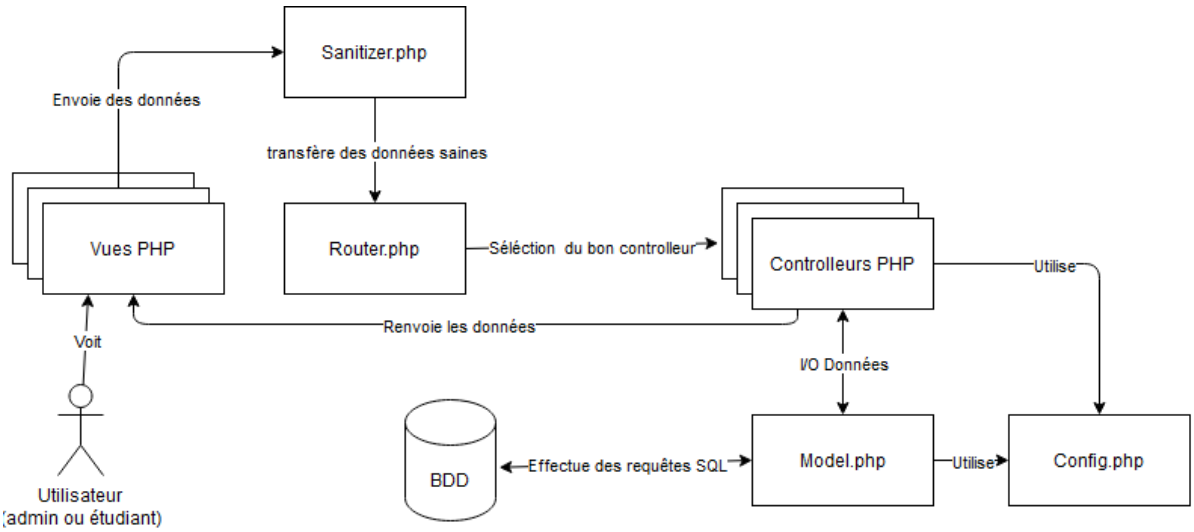

FIGURE 3.3 – Visuel de notre MVC

3.4 Diagramme de navigation

Il est à noter qu'il y a également deux pages internet supplémentaires, "Nous contacter" et "Qui sommes-nous", qui sont accessibles depuis le footer.

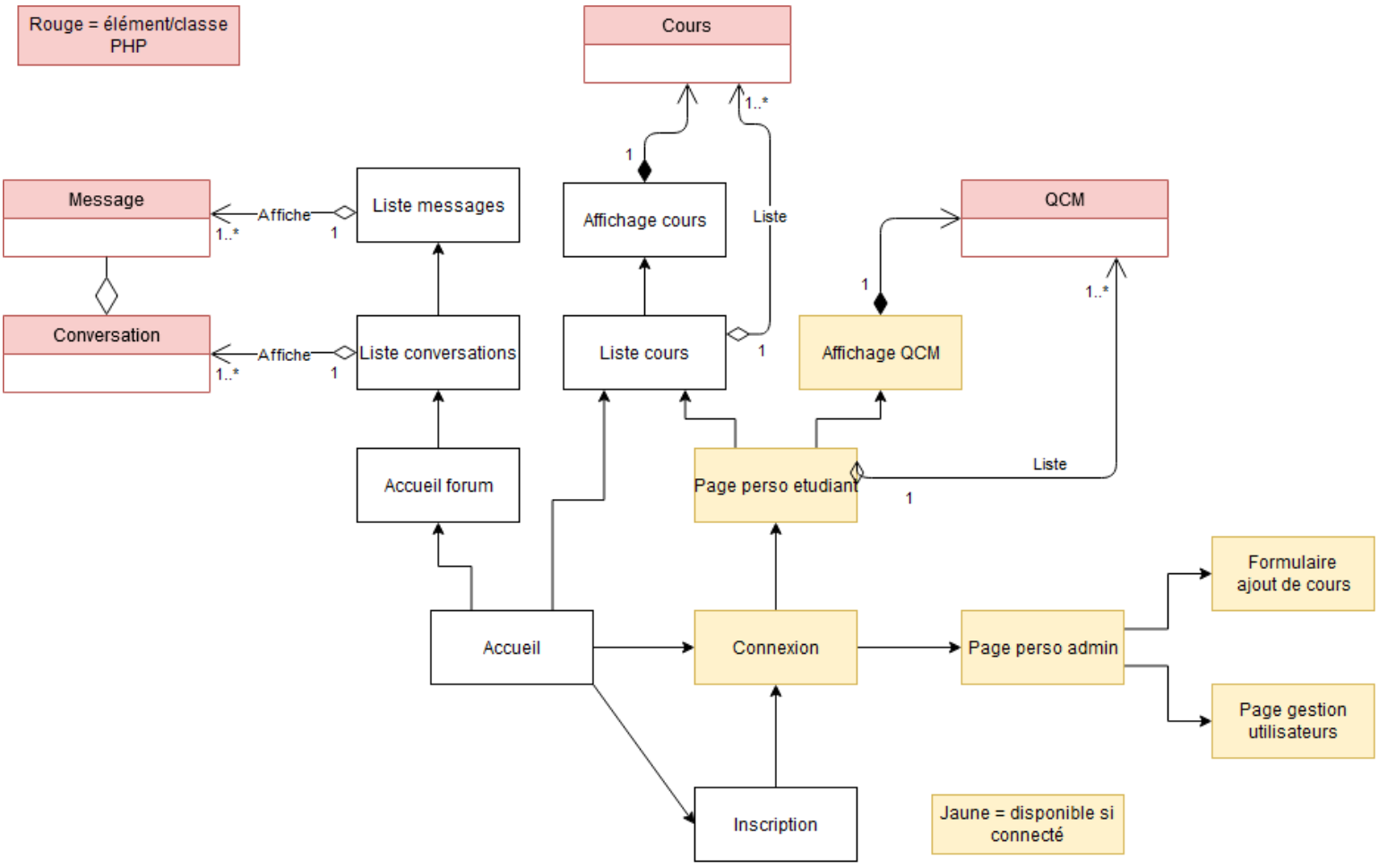

FIGURE 3.4 – Visuel de notre diagramme de navigation

Chapitre 4

Cours

4.1 Contrôleurs

La liste des cours :

- On doit prendre en compte un filtre pour toutes les catégories des cours.
- Il doit retourner les cours suivis par un utilisateur.
- Il doit retourner la liste des cours créés par un utilisateur.
- On peut suivre des cours.
- On peut se désabonner des cours.

Les cours :

- Pour cette catégorie, nous pouvons supprimer les cours dans la liste des cours publiés par un administrateur.
- Nous avons aussi une fonction qui permet de créer un cours.

L'affichage de cours :

- Pour afficher un cours, on cherche un fichier dans la base de données puis dans la data.
- On retourne la vue avec le cours.

4.2 Page liste des cours

4.2.1 Visuels & Structure

Visuels

Voici les visuels sur lesquels nous nous sommes appuyés pour réaliser nos pages de cours. Il est à noter que le design des pages actuelles ne correspond pas totalement à ces visuels.

FIGURE 4.1 – Visuel de la liste des cours

Structure de la page

Pour la structure HTML de la page, nous sommes partis du visuel et nous avons simplement essayé de s'y tenir. Comme à notre habitude, nous avons utilisé plusieurs divs pour faciliter l'intégration CSS.

Il y a donc un div pour le titre et le bloc de texte en dessous, un div pour le deuxième titre et les images **Développement Web** et **Développement Applications** qui vont changer lorsque l'on scrollera dessus. Pour finir, nous avons ensuite un site pour le bloc de texte en bleu et un div pour l'écran et le dernier texte.

4.2.2 Fonctions JavaScript

Sur cette page, l'intégration Javascript principale consiste à changer l'image **Développement Web** par une autre catégorie quand un scroll est détecté sur l'image. Ce changement se fera par le biais d'une animation.
Le fonctionnement est relativement simple : toutes les images sont chargées et les images en attente sont placées avec un z-index de -1 pour les cacher.

Lorsque l'événement de scroll est détecté sur l'image, on bloque l'action naturelle du scroll (pour éviter que la page de bouge) et si aucune animation n'est déjà en cours, on choisit quelle image afficher, puis sur laquelle on augmente le z-index de 1 en fonction de la direction du scroll.
Finalement, on applique l'animation et une fois celle-ci terminée, on met à jour les z-index pour

avoir la nouvelle image à 1 et toutes les autres à -1.

4.3 Page cours de Web - cours d'application

4.3.1 Visuels & Structure

Visuels

Voici les visuels sur lesquels nous nous sommes appuyés pour réaliser nos pages de cours. Il est à noter que le design des pages actuelles ne correspond pas totalement à ces visuels.

FIGURE 4.2 – Visuel de la page des cours d'une catégorie particulière (ici, web)

Structure de la page

De base la page est presque vide, elle ne contient que la barre de recherche, c'est après avec l'ajout des cours grâce à JS que l'on remplit les différentes fonctionnalités, nous avons décidé d'utiliser l'option grid pour la création de nos cases de cours dont voici l'architecture ci-dessous :

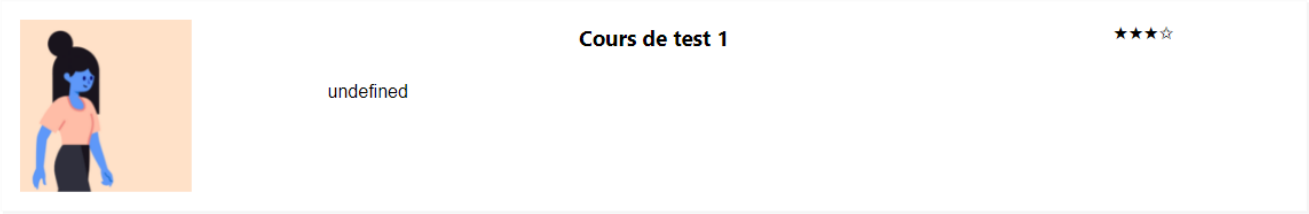

FIGURE 4.3 – Bouton amenant sur un cours lors d'un clic

Définition 4.3.1 *Les pistes peuvent être définies à l'aide de n'importe quelle unité de mesure. Les grilles proposent aussi une nouvelle unité de mesure pour aider à la création de pistes flexibles. Cette unité, fr, représente une fraction de l'espace disponible dans le conteneur de la grille.*

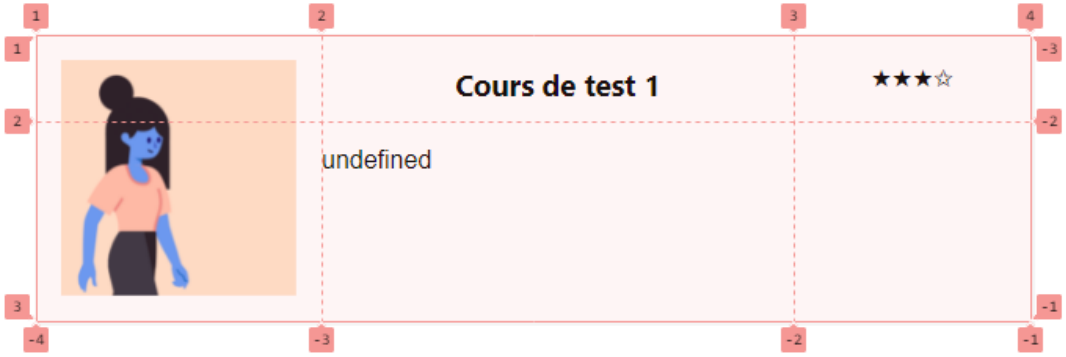

FIGURE 4.4 – Apperçu de la grille de l'élément cours

On a donc séparé la grille en 3 colonnes et 2 lignes, chacune des colonnes ayant respectivement 0.5fr, 1fr et 0.5fr de l'espace *width* de la div et chacune des lignes ayant respectivement 30% et 70% de l'espace *height* de la div.

La partie gauche sur l'ensemble de la grille contient la photo d'illustration du cours, sur la partie en haut à droite se trouve la difficulté, au centre en haut le titre et au centre bas une description.

Remarque 4.3.1 *N'ayant pas de description dans la base de données, l'emplacement au centre bas est libre de disposé d'autres informations comme le nom de celui qui a crée le cours.*

4.3.2 Fonctions JavaScript

Le chargement des différents cours, que ce soit web ou applications, fonctionne de la même manière. Les différents cours en rapport avec la catégorie choisie sont récupérés depuis PHP et transformés en objets javascript afin de simplifier leur traitement côté client.

Côté JavaScript, nous avons donc une fonction updateCours() qui s'occupe d'effacer les cours et de recharger la liste affichée avec un tableau contenant ceux-ci (catégories et cours). fonctionnement de cette fonction repose sur la création d'éléments HTML simples depuis JavaScript avec toutes les informations du cours en question, et ce pour chaque cours retourné par PHP dans notre tableau JavaScript.

La recherche réutilise le concept de la fonction ci-dessus puisque lorsque du texte est tapé, le tableau JavaScript faisant référence à nos cours est filtré pour ne contenir que les cours répondant à la recherche. Ce tableau final est ensuite donné à la méthode updateCours qui recharge l'affichage des cours automatiquement, cette fois-ci uniquement avec les cours souhaités.

4.4 Page ajout cours

La page d'ajout de cours, disponible depuis le profil Admin permet l'ajout de cours à la Base de données. Écrit par l'admin lui-même, des fonctions sont implémentées permettant de créer des sections et chapitre à volonté.

4.4.1 Front-end

Pour des raisons de visibilité, il a été fait en sorte que la page d'ajout cours puisse avoir la même structure que la page d'un vrai cours, l'HTML étant pratiquement le même, la grande difficulté réside alors dans l'implémentation JavaScript des fonctionnalités d'ajout de section, d'ajout de chapitre, de remplissage automatique du sommaire, et de validation du formulaire.

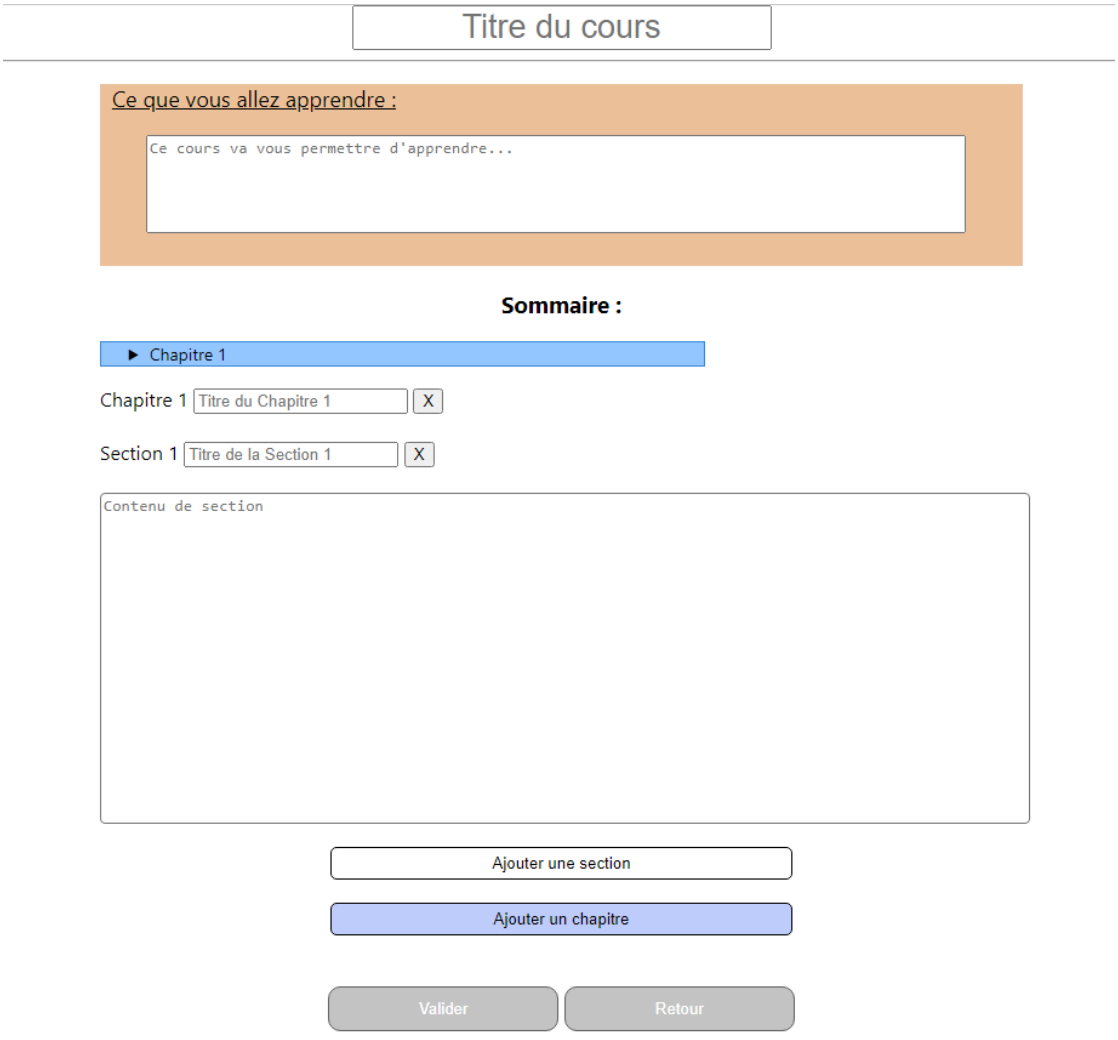

FIGURE 4.5 – Visuel de la page d'un ajout de cours

4.4.2 Différentes fonctionnalités JavaScript

Les fonctionnalités de cette page peuvent être décrites en 2 parties, la partie JavaScript pour permettre l'organisation de la récupération des informations (partie développer ici), et la partie d'envoi au PHP après avoir effectué un *parse*.

Organisation des informations

L'organisation des informations se fait alors en 3 parties.

1. Ajouter des sections et des chapitres.
2. Supprimer des sections et des chapitres.
3. Mettre à jour le sommaire.

Remarque 4.4.1 *Pour mettre à jour le sommaire, on crée une fonction qui va être appelée à chaque changement.*

On choisit alors de montrer en pseudo-code l'ajout d'une section, ainsi cela renforcera la compréhension de l'implémentation des autres fonctions.

Exemple 4.4.1

Algorithm 1: Ajouter une section après l'appuie sur le bouton d'ajout de section

```
chap : Représente le chapitre dans lequel on va rajouter une section;
Ajoute +1 a nb_section;
Ajoute +1 au nombre d'enfants de notre classe .LesSections;
Créer de l'élément html pour section;
Création d'un tableau de chapitre chap[];
for ( Parcourir tous les chapitres ) do
    if ( chap[i] === chap ) then
        break;
    end
end
Append a chap[i] les éléments html pour section;
UpdateSommaire();
```

Récupération du formulaire

Le formulaire envoyé est un formulaire caché, contenant un input avec comme valeur le cours que l'on vient de créer sous format HTML. Une fois submit, ce formulaire stock le code HTML converti en JSON dans POST, et appelle le router.

4.4.3 routeur.php et Cours.php

Le routeur va vérifier la page appelante et va appeler la fonction permettant la sauvegarde de Cours.php, celle-ci a pour fonction de récupérer le code HTML stocké dans POST, transforme le JSON en tableau, récupère le titre, récupère le dernier ID de la base de données, l'incrémente et créer un dossier ayant pour nom l'ID incrémenté, puis créer et écrit le fichier HTML.

4.4.4 Visuels & Structure

Visuels

Voici les visuels sur lesquels nous nous sommes appuyés pour réaliser nos pages de cours. Il est à noter que le design des pages actuelles ne correspond pas totalement à ces visuels.

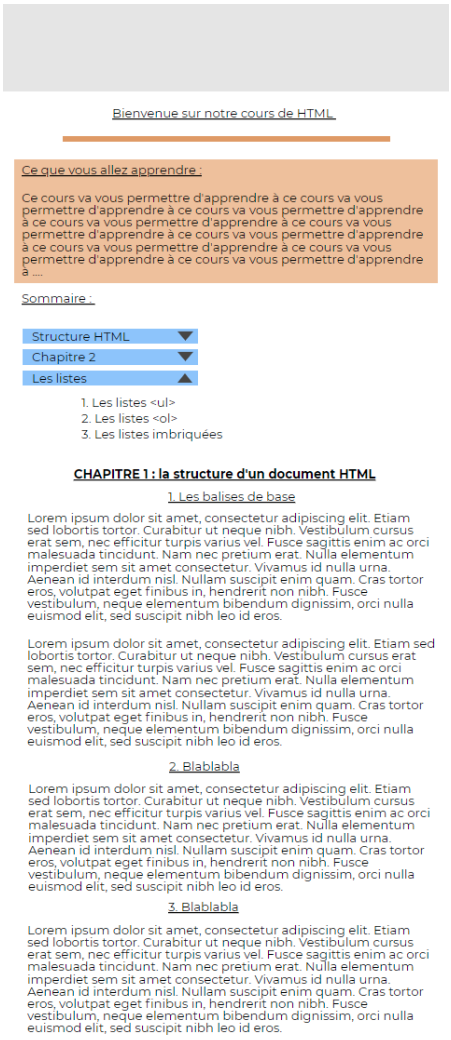

FIGURE 4.6 – Visuel de la page d'un cours

Structure de la page

Pour la structure de nos cours, nous avons décidé de faire une architecture plutôt simple de telle façon que lors de la création d'un cours par un enseignant/admin, il n'y ait pas d'élément de surcharge.

Remarque 4.4.2 *Cette page découlant des informations dans la création de cours, les éléments seront mis en place selon le choix de l'Admin.*

Cette page dispose de 3 principales fonctions JavaScript ;

— L'étudiant a un bouton qui lui permet de s'abonner ou de se désabonner du cours.
— Le sommaire correspond à différentes listes cliquables. Si l'utilisateur clique sur l'une d'entre elles, il est directement amené à la partie concernée.

— Afin de savoir où en est la progression de l'étudiant par rapport à la page, une barre de progression est implémentée. Plus l'étudiant avance dans le cours, plus la barre avance.

Le corps du cours est composé de titre et de paragraphes "p".

Chapitre 5

Connexion et Inscription

5.1 Page de connexion-inscription

5.2 Contrôleur

Ici, on peut lui lier le contrôleur du compte :

- Se connecter.
- Se déconnecter.
- S'inscrire.

5.2.1 Visuels & Structure

Visuels

Voici le visuel sur lequel nous nous sommes appuyés pour réaliser notre page de connexion. Il est à noter que le design de la page actuelle ne correspond pas totalement à ce visuel.

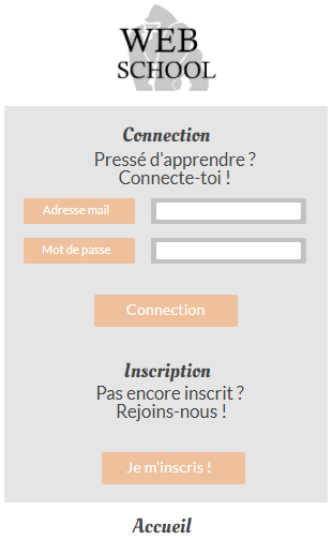

FIGURE 5.1 – Visuel de la page de connexion

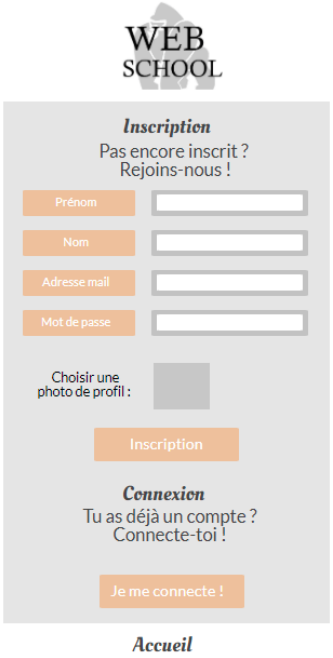

FIGURE 5.2 – Visuel de la page d'inscription

Structure de la page

Dans ces deux pages, nous avons utilisé un formulaire encadré par un fieldset.
En effet, un fieldset permet de regrouper visuellement les éléments logiquement liés au formulaire.

Pour la connexion, l'étudiant peut ainsi rentrer son adresse mail et son mot de passe et se connecter via le bouton "Connexion" qui l'emmènera alors à son profil. S'ils ne sont pas bons, un message le prévient.
Pour l'inscription, l'utilisateur doit rentrer son prénom, nom, email, mot de passe et sa photo de profil. Si des informations ne sont pas bonnes, le formulaire le prévient.

Dans ces deux pages, l'utilisateur peut accéder au formulaire d'inscription (depuis la page connexion grâce à un bouton), et au formulaire de connexion (depuis la page inscription, grâce à un bouton).

Chapitre 6

Profils

6.1 Page du profil étudiant

6.2 Contrôleur

Pour cette partie, nous avons le contrôleur du compte :

- En étant utilisateur
  - Nous pouvons supprimer notre propre compte.
  - On peut modifier ses données, son nom, prénom, mot de passe et on peut changer son mail.
  - On peut se connecter.
  - On peut se déconnecter.
  - On peut créer un compte dans la base de donnée.
- En étant administrateur
  - Même chose que les utilisateurs.
  - On peut gérer les utilisateurs en les supprimant.

6.2.1 Visuels & Structure

Visuels

Voici le visuel sur lequel nous nous sommes appuyés pour réaliser notre page de profil étudiant. Il est à noter que le design de la page actuelle ne correspond pas totalement à ce visuel.

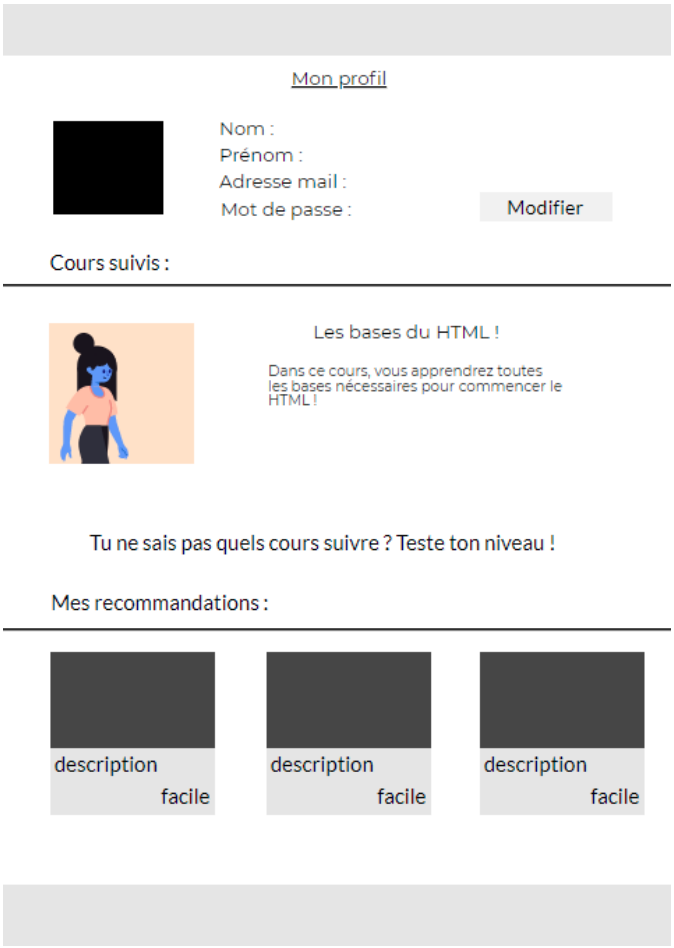

FIGURE 6.1 – Visuel de la page du profil étudiant

Structure de la page

Depuis la page profil de l'étudiant, celui-ci a accès aux cours qu'il suit, ses recommandations, aux QCMs et peut également modifier les informations de son profil.
Cette page est divisée en 3 parties, grâce à des div.

Nous avons une partie d'affichage des informations relatives à l'utilisateur. Ce dernier peut les modifier. S'il essaye de modifier les informations sans avoir cliqué sur le bouton "Modifier" cela ne marchera pas. Il doit cliquer sur ce bouton, puis ensuite il est libre de tout modifier (également de changer son mot de passe). Dès lors qu'il cliquera sur "Enregistrer", tout sera bien sauvegardé et modifié. Il peut également choisir d'annuler ces modifications.

Remarque 6.2.1 *La deuxième partie concerne les cours suivis de l'étudiant en question. L'affichage d'un cours est le même que celui utilisé dans un "Cours" contenu dans les catégories de cours.*

Enfin, la dernière partie permet à l'étudiant d'accéder à la liste des QCM via un élément d'ancre "a" (définit un hyperlien). Également, les recommandations de l'étudiant sont disponibles.
Le système de recommandation est expliqué dans un chapitre dédié "Recommandations". Nous y retrouvons la photo du cours, son titre ainsi que sa difficulté.

6.2.2 Fonctions JavaScript

Nous avons deux fonctions JavaScript. La première sert à empêcher l'étudiant de changer ses informations s'il n'a pas cliqué sur "Modifier". Dans le cas contraire, s'il a cliqué sur ce bouton, il aura le droit de les modifier ainsi que d'autres informations (qui vont alors apparaître, telles que son mot de passe). La deuxième fonction permet de savoir si les deux mots de passe sont différents ou non. La modification ne sera acceptée que si "Nouveau mot de passe" et "Confirmation nouveau mot de passe" sont les mêmes.

6.3 Page du profil admin

6.3.1 Visuels & Structure

Visuels

Voici les visuels sur lesquels nous nous sommes appuyés pour réaliser notre page de profil admin. Il est à noter que le design de la page actuelle ne correspond pas totalement à ce visuel.

FIGURE 6.2 – Visuel de la page du profil enseignant

Structure de la page

Depuis la page profil de l'admin/enseignant, celui-ci a accès à différentes fonctionnalités :
— Il peut voir les cours qu'il a postés.
— Il peut ajouter des cours et/ou des QCM.
— Gérer les utilisateurs.

Cette page est divisée en 3 parties, grâce à des div.

Nous avons une partie d'affichage des informations relatives à l'utilisateur. Ce dernier peut les modifier. S'il essaye de modifier les informations sans avoir cliqué sur le bouton "Modifier" cela ne marchera pas. Il doit cliquer sur ce bouton, puis ensuite il est libre de tout modifier (également de changer son mot de passe).
Dès lors qu'il cliquera sur "Enregistrer", tout sera bien sauvegardé et modifié. Il peut également choisir d'annuler ces modifications.

Une deuxième partie concerne la gestion de ses cours et des QCM. Grâce à deux boutons, il peut soit ajouter un nouveau cours (cela l'emmènera alors sur la page concernée) ou ajouter un nouveauQCM (cela l'emmènera alors sur la page concernée). Dans cette partie, il peut voir tous les cours qui sont à son actif.

Enfin, une dernière partie permet de modérer les étudiants (un bouton amenant à la page "Gérer les utilisateurs").

6.3.2 Fonctions JavaScript

Nous avons deux fonctions JavaScript.

1. La première sert à empêcher l'admin/enseignant de changer ses informations s'il n'a pas cliqué sur "Modifier". Dans le cas contraire, s'il a cliqué sur ce bouton, il aura le droit de les modifier ainsi que d'autres informations (qui vont alors apparaître, telles que son mot de passe).
2. La deuxième fonction permet de savoir si les deux mots de passe sont différents ou non. La modification ne sera acceptée que si "Nouveau mot de passe" et "Confirmation nouveau mot de passe" sont les même.

Chapitre 7

Forum

7.1 Contrôleur

Pour le contrôleur du Forum :
- Récupérer des messages.
- Envoyer des messages.
- Créer une conversation.
- Créer une catégorie.
- Verrouiller une conversation pour qu'elle ne puisse pas être supprimée par n'importe qui.
- Supprimer son message.
- Supprimer des messages d'utilisateurs si vous êtes administrateurs.

7.2 Page d'accueil et catégorie forum

Pour le forum, les classes liées sont celles de l'utilisateur, message et conversation. Le contrôleur lié est celui du QCM. Les utilisateurs vont pouvoir choisir ou créer une conversation, dans celle-ci les utilisateurs vont pouvoir envoyer des messages ou bien supprimer les leurs. Les administrateurs peuvent supprimer les messages des utilisateurs.

7.2.1 Visuels & Structure

Visuels

Voici le visuel sur lequel nous nous sommes appuyés pour réaliser notre page de forum. Il est à noter que le design de la page actuelle ne correspond pas totalement à ce visuel.

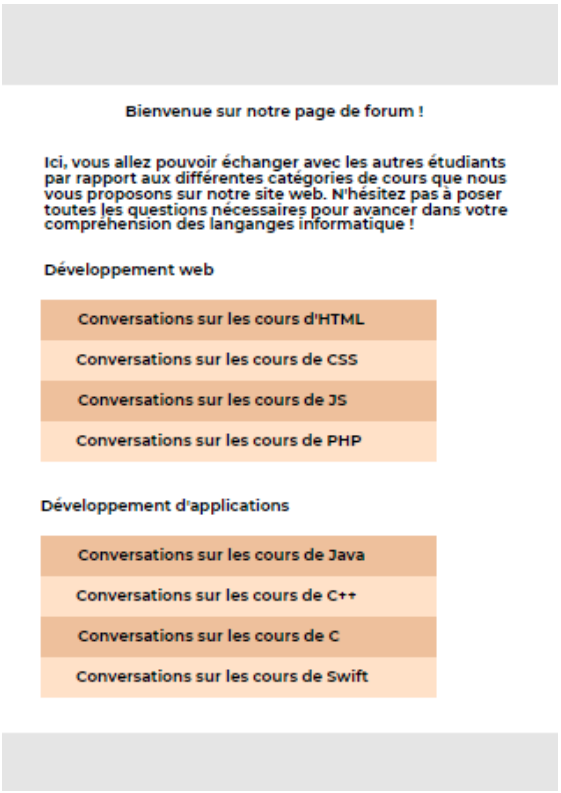

FIGURE 7.1 – Visuel de la page de l'accueil du forum

Structure de la page

La page d'accueil du forum fonctionne avec 4 blocs différents. Le premier contenant le titre centré ainsi qu'une petite description de la page sur laquelle l'utilisateur se trouve, le deuxième réservé à la création d'une catégorie pour les utilisateurs connectés, les deux autres représentent les deux catégories ainsi que les sous-catégories associées.

En terme de structure, notamment pour les catégories, nous utilisons un tableau et générons depuis PHP les lignes correspondant à chaque sous-catégorie.

Remarque : Lorsque l'on clique sur une sous-catégorie, nous sommes amenés sur la liste des conversations en rapport avec cette sous-catégorie, le fonctionnement d'affichage de la liste est le même que la page d'accueil : l'utilisation de tableaux et la structure globale reste la même avec le deuxième bloc réservé non plus pour la création d'une catégorie mais d'une conversation

7.3 Page de message forum

7.3.1 Structure de la page

La structure de la page de messages est décomposée de la manière suivante :

— Rappel de la sous-catégorie, titre de la conversation et message initial
— Autres messages dans triés par date d'envoi croissante
— Bouton de réponse

Chaque message est composé d'un en-tête avec l'auteur et la date d'envoi ainsi qu'un corps contenant le message en lui-même.

Chapitre 8

QCM

8.1 Base de données

Tout d'abord quand l'utilisateur va vouloir passer un QCM, il va devoir charger ses réponses en XML.
Le contrôleur du QCM va regarder si les réponses sont valides et envoyer la réponse à la vue. Après cela l'utilisateur sera redirigé vers la liste des QCMs avec le résultat obtenu en POP-UP.

8.2 Pages du QCM

8.2.1 Structure des pages

Il y a deux pages concernant les QCM (autre que la page "ajout QCM").
Il y a tout d'abord la page d'accueil des QCM qui permet de regrouper toutes les catégories et sous-catégories de QCM. Un fieldset est utilisé afin d'entourer chaque lien cliquable.
Et enfin, une page "QCM", utilisée comme template pour chaque nouvelle page de QCM. Des fieldsets entourent toutes les réponses possibles. Des légendes sont utilisées pour écrire les questions.

8.2.2 Fonctions JavaScript

La validation des réponses se fait en deux parties, une première en javascript du côté utilisateur. Dans celle-ci, on récupère les valeurs que l'utilisateur a coché, les identifiants des questions ainsi que l'identifiant de l'évaluation.
A partir de ces données, on construit un objet JSON dont la structure est la même que celle des fichiers XML de réponses. On envoie ensuite ce fichier au backend pour vérifier les réponses et attribuer la note.

8.3 Page Ajout QCM

Un admin/enseignant, depuis son profil, peut créer un QCM.
Pour cela, il a le bouton "ajouter QCM" de disponible dans sa page profil. Il arrivera alors sur une page, où il va devoir spécifier la difficulté, le type de cours et un fichier XML de questions puis de réponses.

Chapitre 9

Recommandations

Ayant comme contrainte le développement de recommandations de cours pour nos utilisateurs, nous avons donc décidé de développer deux algorithmes différents pour la création de cette fonctionnalité.

9.1 Cours

Pour celui-ci, nous envoyons des recommandations en fonctions des abonnements de l'utilisateur.

Exemple 9.1.1 *Si un utilisateur est abonné à plusieurs cours d'HTML, il recevra sur sa page de profil des recommandations en lien avec des cours HTML.*

S'il n'est pas abonné à un cours, on lui en propose aléatoirement parmi tous les cours disponibles, ainsi, la liste de cours recommandée n'est jamais vide, il y a toujours à apprendre !

9.2 QCM

Pour les recommandations du QCM nous avons procédé d'une manière différente. Dès que l'utilisateur aura participé un QCM, suivant sa note il aura différents cours proposés dans ses recommandations :

- Entre 0 et 24% : il aura des cours de difficulté facile.
- Entre 25 et 49% : il aura des cours de difficulté intermédiaire.
- Entre 50 et 74% : il aura des cours de difficulté avancé.
- Entre 75 et 100% : il aura des cours de difficulté expert.

Exemple 9.2.1 *Un utilisateur complète un QCM avec une note de 14/20, cela correspond donc à la tranche 50/74%, il lui sera donc proposé des cours de difficulté avancé.*

Remarque 9.2.1 *Il existe un délai de 3 jours avant de pouvoir refaire un même QCM afin d'éviter les abus.*

La liste des recommandations contient toujours 3 cours, ainsi à chaque arrivée sur le profil de l'utilisateur, 3 cours recommandés sont affichés en fonction de la liste générée à l'aide des algorithmes décrits ci-dessus.

Chapitre 10

Pages secondaires

Ce chapitre est dédié aux "pages secondaires", correspondant à des pages plus "simples", ne nécessitant pas de base de données.

10.1 Page d'accueil

La page d'accueil correspond à la page principale.
C'est la toute première page que l'utilisateur voit. Elle permet de présenter rapidement notre site internet, mais également d'amener directement l'utilisateur sur nos cours, selon la catégorie souhaitée, et également le forum afin d'avoir un accès direct, si l'utilisateur connaît déjà bien notre site. Utilisation de div pour diviser en 4 parties :

1. Une division comportant une image, un titre et un paragraphe
2. 3 illustrations, divisées à l'aide de div également. On retrouve alors, à gauche, au milieu et à droite : une illustration avec un titre et un paragraphe.
3. Une partie contenant un titre, deux images et deux liens cliquables
4. Une partie contenant une seule image et un lien cliquable

Pour ces deux dernières parties, que ce soit les photos ou les liens : les deux sont entourés d'un élément d'ancre "a" (définit un hyperlien), donc sont cliquables et amènent aux pages concernées (cours de web, cours d'appli, forum).

10.1.1 Visuels

Voici le visuel sur lequel nous nous sommes appuyé pour réaliser notre page d'accueil. Il est à noter que le design de la page actuelle ne correspond pas totalement à ce visuel.

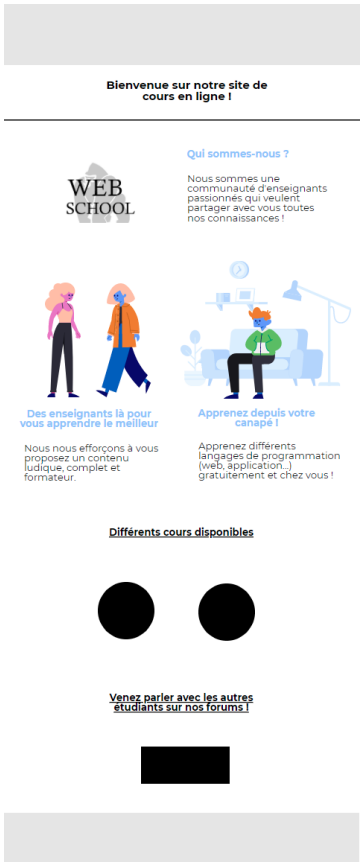

FIGURE 10.1 – Visuel de la page d'accueil

10.2 Footer

Afin de rendre notre site plus "réaliste" et "professionnel", nous avons rajouté deux pages au footer.
L'une permet de présenter l'équipe "Qui sommes-nous ?" composée de deux div, avec une image et un paragraphe. Une illustration permet de présenter en dessin l'équipe, permettant de rapprocher les étudiants et les enseignants/administrateurs.
L'autre permet de contacter l'équipe "Nous contacter" composée également de deux div, avec une image et un paragraphe.

10.3 Élément - Header

10.3.1 Front-end et visuels

Nos éléments de tête et bas de pages ont été réalisés de deux façons différentes décrites ci-dessous ; Voulant tester les limites et fonctionnalités de CSS, le Header est fait à l'aide d'une grille.

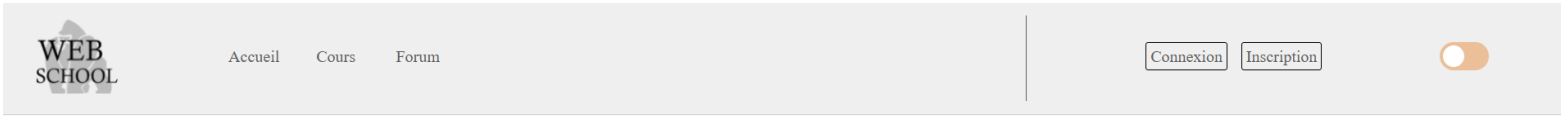

FIGURE 10.2 – Visuel du header

Définition 10.3.1 *Les pistes peuvent être définies à l'aide de n'importe quelle unité de mesure. Les grilles proposent aussi une nouvelle unité de mesure pour aider à la création de pistes*

flexibles. Cette unité, fr, représente une fraction de l'espace disponible dans le conteneur de la grille.

Ainsi le header est découpé en 3 colonnes :

— (0.7 fr) pour la colonne de gauche contenant le logo.
— (3 fr) pour la colonne du milieu contenant le trait de style et les différents liens pour nous amener aux différentes pages.
— (1.5 fr) pour la colonne de droite contenant les boutons de connexion / inscription et le mode dark.

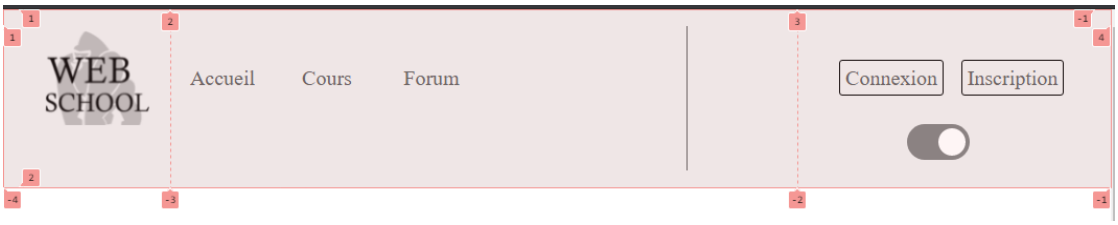

FIGURE 10.3 – Header décrit avec la grille

Remarque 10.3.1 *L'image ci-dessous est zoomée, on peut voir que le bouton s'est déplacé sur une "seconde ligne", cela est dû au fait que le site n'est pas parfaitement responsive*

Le même peut être fait pour les lignes mais cela n'a pas été jugé nécessaire, tout étant sur une même ligne. cependant, cela pourrait être utile dans d'autres cas, par exemple, si on veut créer une grille pour une page html entière.

10.4 Élément - Footer

Le footer lui lie toutes les autres pages qui servent plus d'immersion que de réel fonctionnalités, cependant elle représente bien l'identité graphique du site et mène vers des pages de contact et de présentation de notre petit groupe.

Cette fois-ci nous avons décidé de créer 3 div et d'utiliser la propriété float pour donner un taille horizontale à chaque div, de la même façon que pour le header, cela nous donne alors une sorte de "grille".
— **.gauche** est la colonne de gauche et contient le À propos.
— **.milieu** est la colonne du milieu et contient le Contact.
— **.droite** est la colonne de droite et contient les réseaux sociaux.
Ci-dessous le visuel :

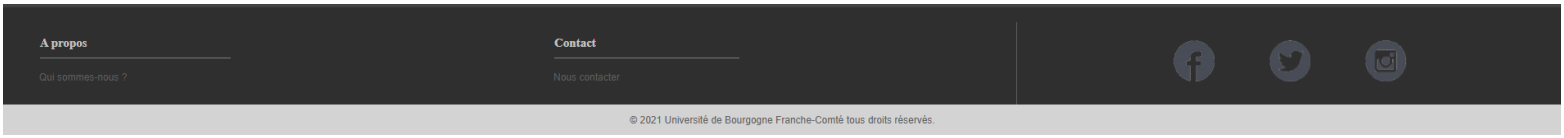

FIGURE 10.4 – Visuel du footer de notre site avec les crédits

Remarque 10.4.1 *Il est important de noter qu'on aurait pu étendre le footer en largeur afin de rajouter d'autres catégories en dessous des deux seules redirections qui existent si besoins futurs.*

Chaque div a une width de 33%, et les crédits se trouvent en bas de cette "grille".

10.5 Light & Dark mode

Afin que le mode sombre soit conservé entre les pages, on a utilisé les cookies pour sauvegarder cet état.
Par défaut le thème est défini en blanc, si le cookie n'existe pas.
Lorsque l'utilisateur clique sur le bouton, les transitions et les thèmes sont appliqués sur les éléments de la page et la nouvelle valeur est enregistrée.

Pour créer le bouton qui va permettre de passer du mode "Light" au "Dark" et inversement, nous avons utilisé différentes choses. Ce dernier est défini par un label, qui contient un input de type checkbox ainsi qu'une div.

Remarque 10.5.1 *Il est à noter que l'input n'est pas visible (il a été mis en "display none" dans le CSS) afin de rendre cela plus esthétique.*

Le label fait figure de conteneur principal.
La div est de couleur orange lorsque nous sommes en "Light" mode, et est de couleur grise quand nous sommes en "Dark" mode. Un rond blanc permet de montrer la transition.
Si l'input est "checked", le bouton change de couleur, tout en déplaçant le rond blanc du côté opposé (on "slide").

Chapitre 11

Conclusion

Ce projet a été très formateur, car il nous a permis de toucher à divers langages informatiques, que ce soit la partie design du site, mais aussi la partie gestion des données et des informations. Le fait de nous séparer en deux groupes distincts nous a réellement permis d'avancer plus rapidement. Chacun était à l'aise avec les tâches qu'il devait effectuer, et pouvait aider l'autre groupe quand cela était nécessaire.

Pour parler des différentes difficultés, la principale problématique fut la gestion du temps entre les différents projets et l'arrivée des révisions. Mais nous pouvons aussi parler de la liaison entre le back-end et le front-end qui fut quelque peu difficile, cependant, notre motivation et notre scrupuleuse application de la méthode AGILE liée avec une organisation cohérente et des personnes travailleuses, nous a permis la réalisation d'un projet abouti dont nous pouvons être fiers.

Plusieurs perspectives sont possibles, par rapport à notre site internet. Comme par exemple, l'ajout de nouveaux enseignants ou de nouvelles fonctionnalités JavaScript pour rendre le site réactif, plus dynamique et avec plus de possibilités. L'ajout d'un système payant aurait pu également être intéressant. Nous aurions alors proposé des cours gratuits mais également payants, avec un système d'abonnement mensuel. De ce fait, à chaque fin de cours, un diplôme aurait pu être délivré.