

# Rapport de développement d'applications Web

## Projet 2022 : Création d'un site de formation

Groupe de projet: TD 1 Groupe 1  
Bertoux Hugo, Hamidou Nazim, Verstracte Valentin  
Petit Evan, Perion Maxence, Pinon Alexandre

### Sommaire

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>La modélisation</b>	<b>2</b>
2.1	Diagramme Use-Case . . . . .	3
2.2	Diagramme Relationnel (et base de données) . . . . .	5
<b>3</b>	<b>Notre installation</b>	<b>7</b>
<b>4</b>	<b>La charte graphique</b>	<b>7</b>
<b>5</b>	<b>Gestion du projet</b>	<b>7</b>
<b>6</b>	<b>L'architecture MVC</b>	<b>8</b>
6.1	Modèle . . . . .	8
6.2	Vue . . . . .	9
6.3	Contrôleur . . . . .	9
<b>7</b>	<b>L'inscription/Connexion</b>	<b>10</b>
<b>8</b>	<b>Les cours</b>	<b>10</b>
8.1	Quizz . . . . .	10
8.2	Forum . . . . .	11
<b>9</b>	<b>Les options</b>	<b>11</b>
9.1	Multilingue . . . . .	11
9.2	Thèmes . . . . .	12
9.3	Décompte des visiteurs . . . . .	13
9.4	Contacteur le support du site . . . . .	13
9.5	Gestion des ressources . . . . .	14
<b>10</b>	<b>Conclusion</b>	<b>16</b>

## 1 Introduction

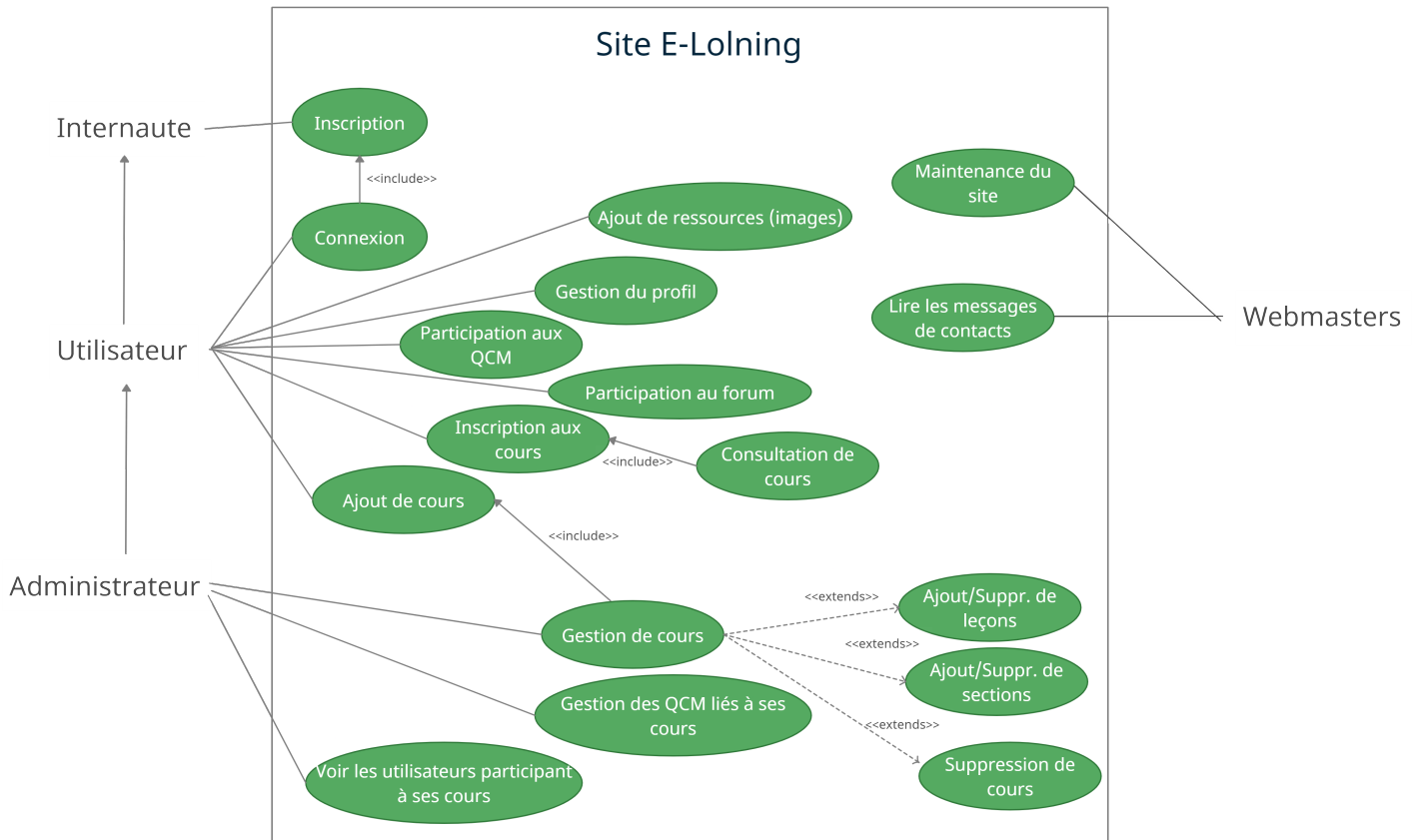
L'objectif de ce projet a été de réaliser un site de formation destiné à des apprenants/étudiants. Nous avons basé la construction de notre site autour d'un jeu vidéo nommé "League Of Legends" et c'est pour cette raison que nous avons décidé de l'appeler "E-lolning". Il se calque sur le fonctionnement de plusieurs sites de coaching en esport comme <https://www.skillcapped.com/lol/>. Dans ce rapport, nous allons aborder tous les aspects de la conception de notre site, c'est-à-dire que nous allons aussi bien expliquer la modélisation, le modèle MVC utilisé et tout ce qui peut-être fait sur notre site. Il est important de rappeler qu'un site de formation en ligne ou site d'e-learning dans notre cas pour un jeu vidéo, permet aux utilisateurs de celui-ci de développer leurs compétences en participant à différents cours et discuter à propos de ce même cours dans un forum entre apprenants. Dans notre cas, nous donnons également la possibilité de créer un cours à qui le souhaite afin de partager son savoir et ses connaissances.

## 2 La modélisation

La problématique de la modélisation s'est centrée autour de deux points : Le diagramme use-case afin de pouvoir discerner toutes les fonctionnalités à implémenter, ainsi que le diagramme relationnel afin de s'assurer d'avoir un schéma de données correctement normalisé et logique avant même d'y stocker des données.

Nous pensons que l'étape du diagramme de classe n'est pas nécessaire puisque nous n'utilisons pas la programmation objet.

## 2.1 Diagramme Use-Case



Nous considérons le besoin de spécifier 4 acteurs chez les utilisateurs de notre site, qui correspondent à leurs "droits". Cela est fait avec la notion spécification/généralisation d'acteurs :

**L'internaute** : Il s'agit d'un simple visiteur du site. L'internaute n'a pas accès aux cours, mais seulement à la page d'accueil et la possibilité de s'inscrire.

**L'utilisateur** : Correspond aux **Apprenants** dans l'énoncé du sujet. L'utilisateur est un internaute qui s'est connecté. Il est bon de noter que la connexion n'est possible qu'après l'inscription, ce qui fait apparaître la notion de `<< include >>` de UML. Nous considérons qu'un utilisateur a accès à tous les cours du site, et il peut créer les siens : Dans ce cas il devient l'Administrateur de son propre cours (voir plus bas). L'utilisateur a accès à un panneau d'administration qui lui permet de gérer son profil, ajouter des cours, et aussi ajouter des ressources liées à son compte : Ce sont des

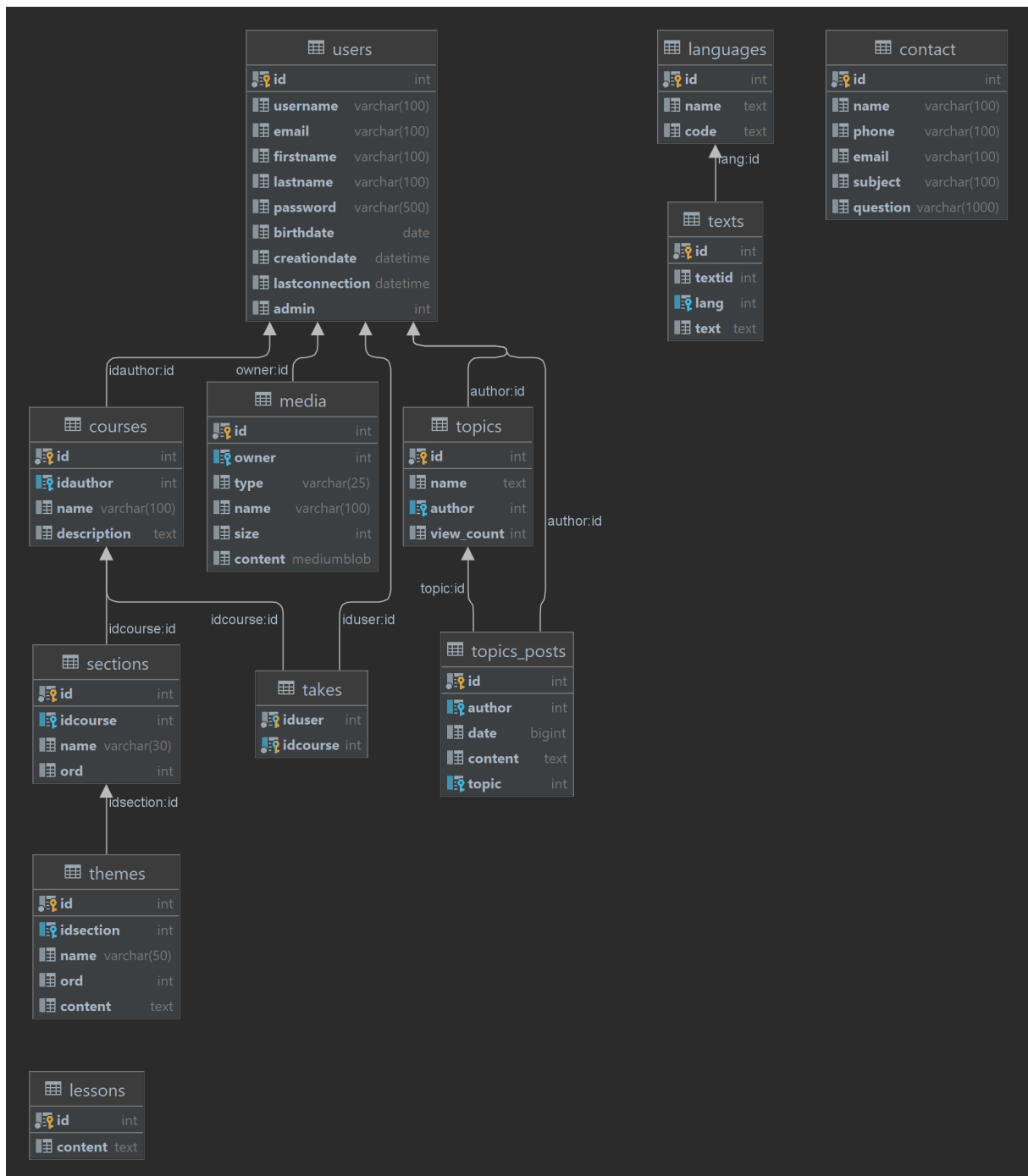
images qu'il peut utiliser dans ses cours, et qui sont hébergées dans notre base de donnée (via des BLOBs). Enfin, il peut participer au forum (créer des topics, répondre à ceux des autres utilisateurs, etc...). Pour consulter un cours, un utilisateur doit s'y être inscrit. Cela permet à l'administrateur de savoir si son cours a du succès ou non.

**L'administrateur :** Correspond au titre du même nom dans l'énoncé du sujet. Il s'agit du statut d'un utilisateur dans son propre cours. Il peut rajouter des leçons, les supprimer, gérer ses QCM, etc... : Afin de modéliser ces choix liés à la gestion d'un cours, encore une fois, nous pouvons profiter d'une notion d'UML. Cette fois-ci des << *extends* >>

**Les Webmasters :** Un comité très réduit d'utilisateurs possédant tous les droits sur le site. Ils ont un accès total au code source, à la base de données, et ont accès à une page leur permettant de voir des messages que les utilisateur leur ont envoyé via un formulaire de contact.

## 2.2 Diagramme Relationnel (et base de données)

Les données sont stockées sur une base de données MySQL dont voici le schéma relationnel :



Nous ne montrons que les tables liées à la modélisation du site, puisqu'une multitude d'autres tables sont nécessaires à l'administration de la base de données, créée ou non par MySQL.

Les images sont stockées directement dans la base de données au format MediumBlob (voir table media)

La table est faite de sorte à ce que la suppression d'un utilisateur supprime en cascade toutes entrées dans la table lui étant liées, comme ses cours, ses topics, etc...

Voici un cours descriptif de chaque table. Premièrement, toutes les tables liées aux utilisateurs :

**users** Stocke toutes les informations liées à un utilisateur à son inscription. Le boolean "admin" est mis à true s'il s'agit d'un webmaster.

**courses** Répertoire de tous les cours du site. Ils sont liés par clé étrangère aux utilisateurs les ayant créés.

**sections** Répertoire de chaque section (chapitres) de chaque cours. Ils sont liés par clé étrangère aux utilisateurs cours dont ils sont les chapitres

**themes** Répertoire de chaque leçon de chaque section. Ils sont liés par clé étrangère aux sections dont ils sont les leçons.

**takes** Associe un utilisateur à un cours, et signifie que l'utilisateur est inscrit à ce cours.

**topics** Stocke les topics du forum. Un topic est un fil de discussion basique, auquel chaque utilisateur peut répondre. Il est lié par clé étrangère à l'utilisateur qui l'a créé.

**topics-posts** Stocke les messages postés sur les topics. Liés par clé étrangère aux topics dont ils sont les réponses, et aux utilisateurs les ayant postés.

Il existe aussi un groupe de deux tables permettant la traduction en anglais et en français des contenus du site :

**languages** Liste les langages disponibles sur notre site

**texts** Table permettant d'inscrire la traduction d'un même texte en français et en anglais

Enfin, deux tables solitaires :

**contact** Liste tous les messages adressés aux créateurs du site via le formulaire de contact.

**lessons** Version obsolète de la table themes

### 3 Notre installation

Pour l'installation nous avons utilisé un serveur LAMP. Celui-ci tourne sous une VM avec Debian 11, 8 GO de RAM et 4 cœurs. Cette même VM est contenu dans un hyperviseur de type 1 qui est installé dans un HP DL380P G8 physiquement présent chez l'un de nos développeurs.

Un tel choix a été fait pour ajouter des plus values qui aurait été indisponible avec une installation à l'université. Nous avons un contrôle total et nous avons par exemple notre propre adresse web : <https://daw.privatedns.org/> Sur cet VM nous avons également installé MySQL, un outil que nous maîtrisons et connaissons.

Pour gérer notre code nous utilisons github. Une "pipeline" a été mise en place pour déployer automatiquement le code source sur le serveur LAMP.

### 4 La charte graphique

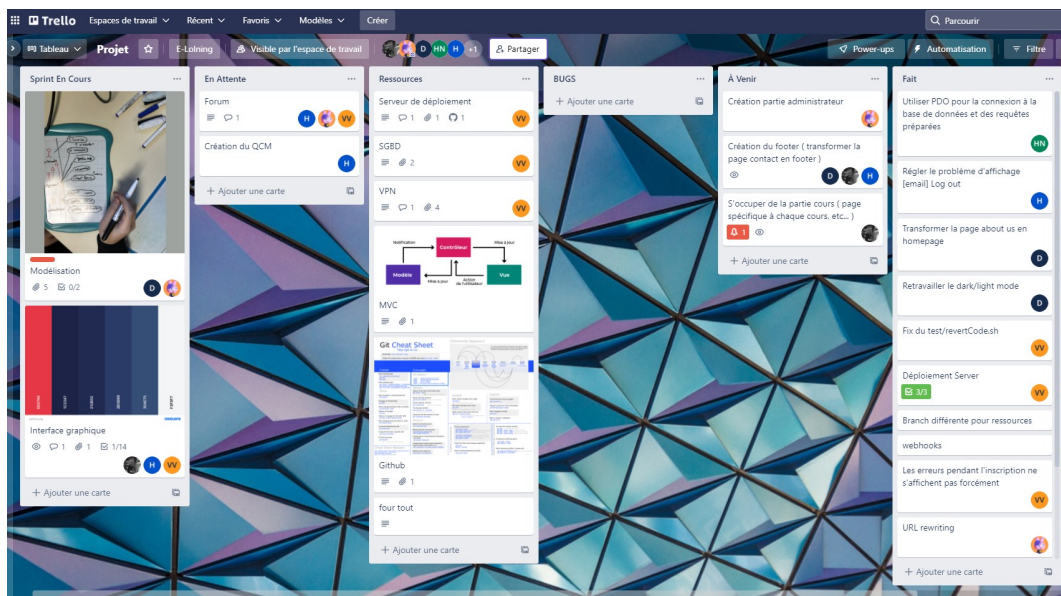
Une charte graphique a été établie, comportant police d'écritures, tailles de police, couleurs pour le CSS, etc... Les couleurs ont été choisies à l'aide de : <https://colors.co/>

### 5 Gestion du projet

Afin de mener à bien le projet, nous nous sommes répartis plusieurs rôles au sein du groupe. Déjà, les deux imposés dans l'énoncé du projet :

**Product Owner : Hugo Bertoux.** Le rôle du Product Owner a été constitué de deux points. Déjà, se mettre à la place du client et donner son avis sur le travail déjà effectué. Ensuite, d'être en relation (au besoin) avec l'équipe pédagogique en cas de concise peu claire, ou de demandes particulières pour le site.

**Scrum Master : Evan Petit.** Son rôle a été de répartir tout le travail à faire entre les participants au développement du site. Cela se faisait au travers de réunion quotidiennes via messagerie instantanée et appels, tous les mercredis de chaque semaine. Nous n'avons pas adopté le système de points d'efforts, et avons préféré se répartir simplement une liste de tâches, qui, toute complétées permettent d'avoir un produit terminé. Chaque développeur qui est en retard peut déposer des "Todo" en commentaire dans son code (c'est à dire, du code à faire) - Code sur lequel chaque autre développeur en avance peut se pencher.



Pour répartir efficacement les tâches, en garder une trace, cela et d'autres informations, un tableau sur Trello a été créé, voir image ci-dessus. On peut y apercevoir des catégories comme "à faire", "terminé", "ressources", etc... Nous sommes agréablement surpris que tous les participants aient apprécié s'en servir, et cela a permis d'accélérer significativement la vitesse de développement de notre site.

D'un format plus classique, le code a été rédigé en collaboration sur GitHub, qui est un élément absolument nécessaire à maîtriser pour qui que ce soit qui compte créer un site web.

Afin d'aller plus loin, nous avons considéré un autre rôle majeur :

**DevOps : Valentin Verstracte.** En tant que propriétaire d'un serveur physique, Valentin a été l'administrateur du serveur sur lequel nous hébergeons le site. C'est lui qui a mis en place l'accès distant au serveur, l'hébergement de la base de données...

En plus des front-ends, back-ends, administrateurs de base de données.

## 6 L'architecture MVC

### 6.1 Modèle

Dans notre projet, les Contrôleurs appellent les Modèles pour deux raisons: Aller chercher des données dans la base de données dans la plupart des cas, ou bien les fichiers XML stockés sur notre serveur.



Dans ces deux cas, nous utilisons les fonctions natives à PHP qui permettent de dialoguer avec la base de données MySQL et de modifier, créer ou supprimer des fichiers sur le serveur.

## 6.2 Vue

Les vues permettent l’affichage des pages HTML au client grâce à php. Ce sont des fichiers PHP qui vérifient les droits de l’utilisateur courant grâce aux différents cookies, et sessions (est-ce un administrateur? est-il connecté?), et appelle diverses fonctions vis-à-vis de ces droits pour afficher certains éléments de la page :

Un utilisateur connecté n’a pas besoin de voir le bouton ”se connecter”, un élève ne doit pas avoir accès au bouton ”modifier le cours” ou ”ajouter une leçon”.

Lorsque certaines données ont besoin d’être récupérées, par exemple pour afficher le contenu d’un cours, le pseudo de tous les utilisateurs, etc... Un appel est fait au contrôleur correspondant.

## 6.3 Contrôleur

Les contrôleurs sont des fichiers PHP qui permettent de faire appel aux modèles via les vues. Les données récupérées sont souvent brutes (les tables clé-valeur qui sont retournées par MySQL par exemple)

Avec les contrôleurs, nous nous assurons que les données récupérées sont traitées et affichables à l’utilisateur afin que la vue puisse l’afficher correctement.

## 7 L'inscription/Connexion

Le principe ici est assez simple et s'inspire de tout site web moderne. Des expressions régulières ont été ajoutées pour pouvoir contrôler les entrées, du genre le nom et prénom ne doivent pas contenir autre chose que des lettres, le nom d'utilisateur ne doit contenir que des lettres et des chiffres. Un contrôle de présence de chacune des entrées a été mis en place (on peut pas s'inscrire sans email par exemple).

Chaque erreur (si présente) est directement transmise à l'utilisateur pour qu'il puisse la corriger avant de pouvoir valider son inscription.

## 8 Les cours

### 8.1 Quizz

Les quiz vont être proposés pour les apprenants. Les quiz doivent être sous le format XML, on va donc partir sur un document XML par cours. Celui-ci sera créé à la création du cours. Voici la DTD de notre quiz :

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Questionnaire (question+)>
<!ATTLIST Questionnaire chapitre NMOKEN #REQUIRED>
<!ATTLIST Questionnaire Nom CDATA #REQUIRED>
<!ELEMENT question (difficulte , intitule , choix + , reponse +)>
<!ATTLIST question id NMOKEN #REQUIRED>
<!ELEMENT difficile (#PCDATA)>
<!ELEMENT intitule (#PCDATA)>
<!ELEMENT choix (#PCDATA)>
<!ATTLIST choix id NMOKEN #REQUIRED>
<!ELEMENT reponse (#PCDATA)>
```

On peut donc voir qu'une question aura un id associé, une difficulté, son intitulé ainsi que 4 choix et sa réponse. Et donc lors de la création du cours, un document XML de cette forme sera créé :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Questionnaire SYSTEM "quiz.dtd">
<Questionnaire chapitre="idducours" Nom="nomducours">
</Questionnaire>
```

Ainsi l'administrateur de son cours pourra par la suite venir ajouter ou supprimer des questions de son cours.

Du côté des apprenants, ils pourront passer le quiz dans le cours associé. La liste des questions s'affichera et ils pourront y répondre en cliquant sur le choix qui veulent faire et ensuite valider le questionnaire. A la fin, ils verront leurs notes et pourront repasser le test s'ils ont envie de faire mieux.

## 8.2 Forum

Le forum permet la discussion entre apprenants. Il est global et les utilisateurs peuvent créer un nouveau sujet ou répondre dans un déjà créé s'ils sont connectés. Ils peuvent également consulter la liste des sujets et les messages dans ceux-ci. Les utilisateurs connectés ont également la possibilité de modifier ou supprimer les sujets et les messages dont ils sont les auteurs.

## 9 Les options

### 9.1 Multilingue

La première option que nous avons implémentée est un support multilingue ou plurilingue. Les différentes langues implémentées sont le Français et l'Anglais pour tous les utilisateurs. Nous avons fait cet ajout dans un souci d'utilisation étrangère de notre site. En effet, nous nous sommes basés sur un jeu de renommée mondiale avec plusieurs milliers de millions d'utilisateurs annuels, et forcément parmi tous ces utilisateurs tous ne parlent pas couramment Français.

Afin de changer la langue il suffit de cliquer sur ce menu déroulant:

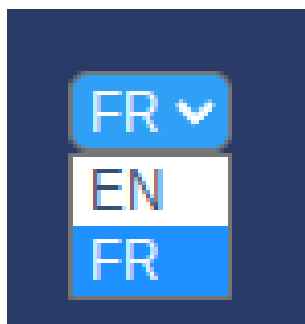


Figure 1: Les différentes langues proposées.

Pour ce qui est de la partie technique, nous avons dans notre base de données la table “texts” où sont inscrit tous les textes à afficher selon un identifiant de texte (qui est le même peu importe la langue) et une langue. Ensuite, il nous suffit d’afficher la traduction souhaitée dans le site après avoir récupéré la langue renseignée par l’utilisateur. Seules les traductions pour la langue anglaise et française sont actuellement disponibles, cependant le système a été conçu de façon scalable, c’est à dire que l’ajout d’une nouvelle langue dans la base de données rendra directement disponible cette langue pour les utilisateurs, sans que nous aillions à modifier quoi que ce soit dans nos scripts php. Cela est rendu possible grâce à une table “languages” qui renseigne les différentes langues (abréviation, code, clé, ...) et

la sélection de la langue depuis le site lit ces informations pour les afficher dans le sélecteur. La langue pour chaque traduction dans la table “texts” est une référence (clé étrangère) sur la clé de la table “languages”.

## 9.2 Thèmes

Notre site d’e-learning possède une option qui permet de changer de thèmes, c’est-à-dire passer du thème clair (thème par défaut) au thème sombre. Les différents thèmes sont utilisables par tous les utilisateurs à leur bon vouloir. L’intérêt d’utiliser des thèmes sont les suivants: le thème sombre permet de réduire la fatigue des yeux et les différents thèmes sont plus approprié à certaines conditions par exemple en pleine nuit le thème sombre sera plus apprécié.

Les utilisateurs peuvent changer de thèmes par l’intermédiaire de ce bouton:

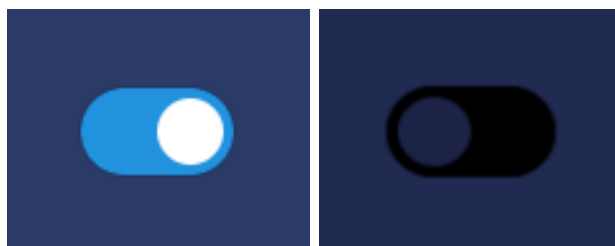


Figure 2: Bouton en mode thème clair

Figure 3: Bouton en mode thème sombre

Le fonctionnement des thèmes est assez simple, en effet il nous suffit de déclarer 2 fichiers css que nous avons nommée respectivement “light-Mode.css” et “darkMode.css”. Et dans ces fichiers déclarer des variables avec “-nomVariable”. Pour ces deux fichiers les variables doivent avoir le même nom mais avoir évidemment des valeurs différentes. Une fois ceci effectuée, nous allons utiliser seulement les variables contenus dans les fichiers “light-Mode.css” ou “darkMode.css” via différentes fonctions jQuery créé par nos propres moyens. Finalement pour avoir un thème permanent et ne pas avoir besoin de repasser dans le thème souhaité à chaque rafraîchissement de la page, nous avons utilisé un cookie. Ce cookie permet de stocker la valeur du thème choisie et avec le traitement de nos fonctions en jQuery chargé le bon thème.

Dernièrement dans les fichiers css qui vont organiser nos différentes pages nous allons utiliser cette syntaxe: “var(-nomVariable)”, qui aura une valeur différente en fonction du thème choisie.

Pour illustrer ce qui a précédemment été vue, voyons avec un exemple concret, définissons une variable ”background-color: #FFFFFF;” pour le thème clair et ”background-color: #000000;” pour le thème sombre. Maintenant dans un fichier css quelconque on peut utiliser ceci:


```
body{  
    background-color: var(--background-color);  
}
```

Si le thème clair est utilisé alors la couleur de fond aura pour valeur FFFFFFFF c’est-à-dire blanc, et dans le cas contraire pour le thème sombre la couleur de fond aura pour valeur 000000 c’est-à-dire noir.

### 9.3 Décompte des visiteurs

Dans une dynamique de création d’options agréables pour tous, nous avons intégré un compteur du nombre de visiteur, ce compteur est affiché dès l’arrivée de l’utilisateur sur la page d’accueil. Afin de stocker le nombre de visiteur de notre site nous l’avons fait via un fichier et non base de données car PHP permet la gestion de ce genre de petit fichier. Le fichier se nomme tout simplement ”compteur.txt” et contient le nombre de visiteurs.

Voici un aperçu de l’affichage en question:



Nombre de visiteurs cette année : 21

Figure 4: L’affichage du nombre de visiteur.

### 9.4 Contacter le support du site

Finalement, pour tous utilisateurs ayant la moindre question que ce soit sur le site, un cours en particulier ou d’autres informations, alors l’utilisateur peut nous faire parvenir un message via la section “Contact” sur la page d’accueil. Pour envoyer à sa requête, il faut remplir les champs nécessaires sur cette page. Si le message est correctement traité alors un message de confirmation d’envoi est affiché. L’utilisateur peut ensuite attendre une réponse via l’un des moyens de contact qu’il a renseigné. Pour l’équipe d’administration, il existe une catégorie supplémentaire dans la partie profil. Cette catégorie se nomme “Manage contacts” et elle permet d’afficher l’ensemble des messages envoyés par les utilisateurs. Ces messages peuvent être supprimés par les Webmasters, mais ils sont également dans la possibilité de répondre à ces messages via un “mailto”.

Voici à quoi ressemble la page de gestion des messages:

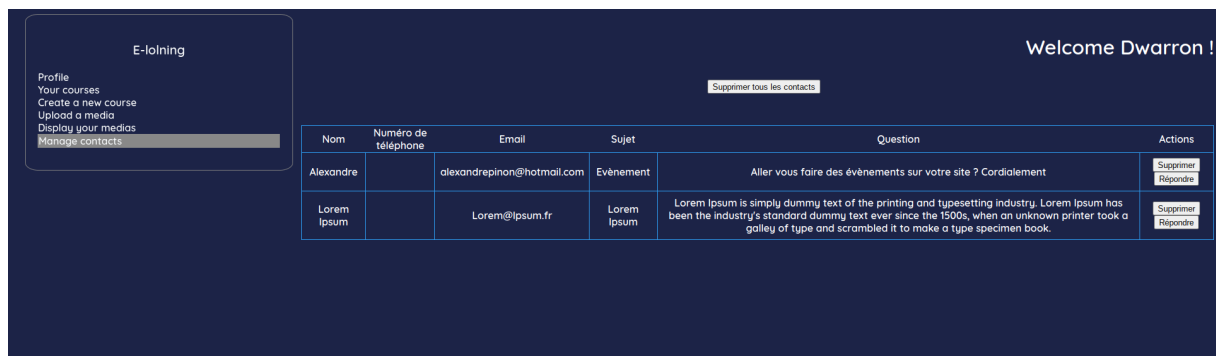


Figure 5: L’affichage des messages pour les Webmasters.

Dernièrement, les messages sont stockés dans la table “contact” de la base de données et les administrateurs sont définis par un booléen dans la table “users” que nous pouvons ajouter à la main.

## 9.5 Gestion des ressources

Un utilisateur peut utiliser deux types de ressources afin de décorer son cours : Les images, et les vidéos.

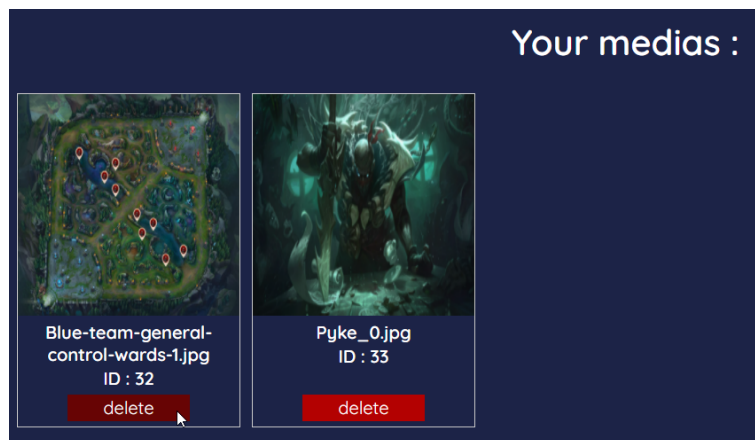
**Les images** - L'utilisateur a accès à un menu "Enregistrer un media" dans son panneau d'administration. Cela lui donne accès à un bouton lui permettant de choisir un fichier sur son ordinateur. Un clic sur le bouton "Envoyer" signale à PHP via un formulaire que la superglobale \$\_FILES['nomVariable'] de PHP est remplie avec toutes les informations concernant l'image. Incluant sa taille, son type, son nom et surtout le tableau de bits qui la compose.

Upload an image :

Choose File No file chosen
Envoyer

La vue envoie au contrôleur toutes les informations, et essaye de les insérer dans le modèle (notre base de données) au format BLOB. Cela se fait si le format correspond (png ou jpeg), que la taille est sous le seuil maximum, etc...

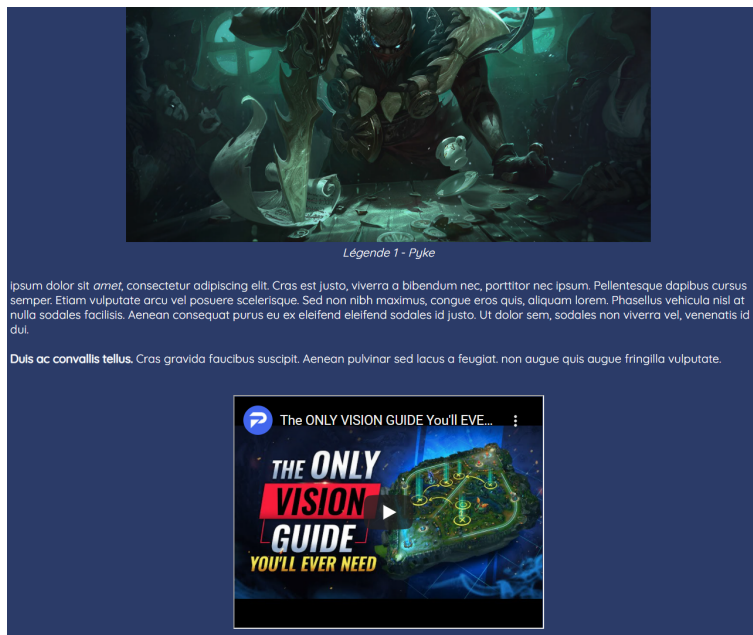
Cela redirige l'utilisateur vers la page "Voir ses medias" qui permet à l'utilisateur de consulter le nom et l'id de chacune de ses images. Il peut aussi les supprimer.



L'utilisateur peut utiliser les balises `[img =< id >] < legende > [/img]` dans son cours, en remplaçant `< id >` par l'ID de l'image qu'il souhaite afficher de son répertoire de medias. Cela ne marchera bien évidemment pas s'il essaye d'afficher une image en utilisant l'id d'un autre utilisateur. Avec PHP, le nécessaire est fait pour convertir ces balises en code HTML qui permet d'afficher des images à l'écran.

**Les vidéos -** Pour éviter de surcharger la base de données qui est hébergée sur le serveur personnel d'un des membres du groupe, nous avons décidé de traiter le cas des vidéos d'une manière différente : L'utilisateur peut, muni d'un lien Youtube, utiliser les balises `[video] < lien > [/video]` afin d'afficher une vidéo depuis youtube sur son cours, de la même manière qu'une image.

Voici l'exemple d'un cours contenant une image et une vidéo :



## 10 Conclusion

Pour rappel le but de ce projet était de réaliser un site de formation en ligne. Nous y sommes parvenues en respectant scrupuleusement les étapes de développement expliqué tout au long de ce rapport. En effet nous ne sommes pas partie dans tous les sens et commencer à coder sans but précis, la modélisation a été la première marche que nous avons gravie afin de poser pour tous les limites et les premières étapes à concevoir. Et finalement grâce au travail du Scrum Master nous sommes parvenue à rendre un travail complet et régulier. Il serait alors intéressant de regarder les améliorations envisageables pour notre site, comme par exemple un serveur SMTP pour la gestion des mails, mais encore un site avec un meilleur design mais qui aurait demandé l'utilisation d'un framework comme Bootstrap. Finalement nous avons utilisé AJAX uniquement dans la recherche d'un topic dans le forum, mais on aurait pu l'utiliser avec le système de multilangues, les messages des utilisateurs pour contacter le support et dans plusieurs autres cas.