Théophile ROMIEU
Daniel PINSON
VanTai NGUYEN
Yann MOURELON
Dylan FAUSSOT
Kyllian JOLIVET

L3 Informatique groupe 3 TP6

RAPPORT PROJET DAW 2022

L.Abrouk

Architecture MVC :

Scrum Master : Théophile ROMIEU
Product Owner : Daniel PINSON
Partie Vue : VanTai NGUYEN, Yann MOURELON
Partie Modèle : Théophile ROMIEU, Daniel PINSON
Partie contrôleur : Dylan FAUSSOT, Kyllian JOLIVET

INTRODUCTION :

Bienvenue dans notre projet de développement d'application web (DAW). Le but de ce projet consiste à faire une application web permettant à des étudiants/apprenants d' accéder à du contenus pédagogique mis en ligne par des administrateurs qui pourront gérer différents aspects du site. Un étudiant peut alors tester ses capacités via un QCM qui lui est proposé, pour ensuite lui donner des recommandations sur les cours qu'il peut vouloir suivre, accéder à différents cours proposés par notre plateforme, ainsi que interagir avec d'autres étudiants via un forum de discussion où différents canaux peuvent être créer pour satisfaire la curiosité d'un maximum de personnes. L'administrateur quant à lui, peut faire les mêmes choses qu'un étudiant, mais aussi mettre en ligne son propre cours, et le manipuler comme bon lui semble avec la mise en place CRUD (Create, Read, Update, Delete), la gestion des utilisateurs fait aussi partie de ses occupations.

Cependant, nous devons suivre certaines consignes techniques pour mener à bien notre mission. Tout d'abord, nous devons utiliser 4 langages de programmations imposés qui sont : HTML, CSS, JS, PHP. Bien évidemment, nous ne devons pas nous servir d'outil externe qui nous faciliterait la mise en forme du site ou bien la mise en place de notre architecture comme le ferait par exemple Bootstrap. Il nous est demandé d'utiliser une architecture MVC pour Model Vue Controller afin de mieux organiser notre site et faciliter la création, la compréhension et la modification de celui-ci. L'utilisation des cookies est exigée ainsi que la mise en place de QCM au format XML.

Le projet nécessite une configuration particulière qui est précisée dans le fichier readme à la racine du projet

Idée et organisation :

Nous n'avons pas eu beaucoup d'informations concernant la mise en place de ce qui nous est demandé, ce qui laisse beaucoup de flou et de champ libre. Nous avons cependant décidé de partir sur un site un peu à la manière de *plubel*, qui nous est proposé par l'université de bourgogne, tout en s'inspirant de *openclassroom* (anciennement *le site du zéro*) le site de formation en développement informatique par excellence. Nous avons décidé d'utiliser le modèle orienté objet pour créer notre site.

La répartition fut un peu plus compliquée à mettre en place. Nous sommes six dans le projet avec 3 rôles différents donc le plus simple pour commencer est de mettre 2 personnes par rôles. Cependant, nous avons conscience que le rôle du contrôleur est un peu plus long et pénible que la partie vue ou même modèle, nous restons donc disponible pour changer de rôle quand on en a un besoin.

La mise en place de réunions de projets était nécessaire pour la communication dans celui-ci, donc en moyenne une réunion par semaine nous semblait nécessaire et suffisante pour se tenir au courant les uns, les autres.

Outils pratiques dans l'organisation :

Pour nous tenir au courant des modifications apportées au sujet et des tâches à faire, nous avons mis en place un github modifiable uniquement par les membres de notre groupe mais visible par tout le monde. Un discord a aussi été créé dans le but de pouvoir communiquer plus aisément et pouvoir nous passer certaines ressources nécessaires au projet et nous indiquer quand un push discord à été fait pour nous tenir au courant de la moindre modification et se mettre à jour sur le projet pour avancer ensemble. Pour connaître les tâches en cours, faites ou à faire, nous avons aussi créé un Trello pour différencier du mieux possible les implémentations importantes des implémentations annexes.

Contenu du projet :

Après vous avoir présenté le projet à travers de grandes lignes et vous avoir expliqué notre organisation et comment on s'y est pris pour travailler ensemble, nous vous présentons les différents aspects de notre projet, notre réflexion et comment nous avons mis en œuvre le tout.

Partie vue :

Au début de la production de notre application web, la partie vue était déconnectée des autres parties, nous avons imaginé le visuel du site en prenant en compte les méthodes de bases qui devront être installées et appelées pour effectuer différentes actions mais en négligeant les actions spécifiques à chaque utilisateur comme ajouter une matière ou encore la création de bouton permettant de suivre un cours. Nous sommes donc parti pour utiliser uniquement HTML CSS et JS pour la vue et faire le visuel du site (cf. Annexe1). Notre premier rendus étant satisfaisant, nous avons décidé de le garder et par la suite, nous avons ajouté les items nécessaire au bon fonctionnement de notre application et nous sommes passé du format .html au format .php nous permettant d'inclure plus facilement les items communs comme l'entête et le pied du fichier, le menu déroulant et la banderole d'utilisateur qui sont présents sur chaque page.

Une fois la base de données paramétrée et les parties contrôleur et modèle avancé, nous avons mis à jour les chemins d'accès aux différentes pages de notre site via un Router. Le lien entre la vue et le contrôleur est fait, il ne reste plus qu'à parfaire l'interface utilisateur.

Toutes les routes disponibles sont 127.0.0.1/

/login, /register, /home, /addCanal/{id}, /canal/{id}, /delCanal/{id}, /addForum, /forum, /delForum, /home, /login, /logout, addDocument/{id}, /delDocument/{id}, /addMlatiere, /matieres, /matiere/{id}, /delMatiere, /addQcm, /qcm, /qcm/{id}, /delQCM, /test, /delUser/{id}, /user

Les routes qui se terminent avec {id} représentent une ligne de la base de données. Les boutons permettent de ne pas avoir à s'en soucier.

Partie modèle :

La partie model va nous permettre d'extraire des données de la base. La connexion à la base de données est assez simple à faire en revanche pour pouvoir nous connecter à cette dernière, il faut que celle-ci existe. Nous avons utilisé le patron de conception du singleton qui nous évite d'avoir plusieurs connection à la base de donnée. Nous somme toujours connecté en tant qu'administrateur, sachant que les étudiants n'auront pas accès aux objets de cette partie

Nous avons dans un premier temps chercher à structurer les différentes informations dont un utilisateur aurait besoin afin de différencier un utilisateur lambda d'un étudiant et d'un administrateur. Plusieurs tables ont été créées pour structurer les données de notre site comme il se doit (voir Structure BDD). Nous avons donc commencé avec une base de données très simple nous donnant accès aux informations d'un utilisateur sans chercher à tout créer d'un coup. Différentes méthodes ont été créées pour interagir avec notre BDD afin de récupérer les informations d'un utilisateur, les mettre à jour lors d'une modification, ou bien les supprimer.

Par la suite, nous avons créé plus de tables pour mieux structurer nos données et sauvegarder les fichiers nécessaires à un cours, on a aussi modifié certaines tables comme celle de l'étudiant par exemple qui sauvegarde les cours que l'utilisateur veut suivre, une autre pour la création du forum, etc… Nous avons aussi créé toutes les méthodes permettant de récupérer ces données, les mettre à jour et les supprimer pour celles qui en ont besoin.

Pour pouvoir garantir que cette partie marche, il existe une suite de test qui permet de savoir s' il y a un problème. Il est possible d'y accéder depuis la route /test

Partie contrôleur:

Cette partie est pour nous la plus compliquée à mettre en place. Lorsque nous avons commencé à mettre les contrôleurs en place, la partie vue et la partie model n'étaient pas finies. Nous nous sommes donc cantonnés à faire uniquement la jonction entre ce qui était accessible côté vue et les méthodes existantes partie model, donc les objets métier qui nous permettent d'encapsuler de l'information pour mieux la manipuler. Ces objets ont été créés selon le Diagramme métier.

Lorsque la partie modèle et vues ont avancées pour laisser apparaître de nouvelles possibilités comme créer une matière et y ajouter des fichiers, nous avons repris la partie contrôleur en main et pouvoir, côté client, appeler les méthodes du contrôleur correspondant à l'action que l'utilisateur souhaite exécuter.

Le site contient un routeur qui nous permet de naviguer d'une page html à l'autre selon des URL que nous avons définies au préalable. Aller sur un lien du type: 'localhost/home' renvoie sur la page d'accueil du site mais 'localhost/maison' n'est pas une route assignée et renvoie sur la page 404. Aller sur une page revient à effectuer la méthode GET du routeur qui appelle le contrôleur correspondant à la page affichant la page avec toutes les informations nécessaires. La méthode POST est utilisée lorsqu'il est nécessaire de recevoir des informations de l'utilisateur, ces informations seront ensuite traitées par le contrôleur en conséquence.

A chaque route correspond un contrôleur qui a pour rôle d'appeler les DAO afin de récupérer les informations nécessaires à l'affichage de la page, et qui va aussi traiter les

modifications, ajout et suppression en fonction de ce qui a été demandé par le professeur. La seule exception est authContrôleur, qui gère tout ce qui est en rapport avec création de compte, login et logout.

Structure BDD :

Lorsque nous avons imaginé notre projet, la base de données ne nous paraissait pas compliquée à mettre en place. La base de données que nous utilisons est PHPMyAdmin, qui utilise le langage SQL pour faire ses requêtes. Pour définir le reste de notre base de données nous nous sommes basés sur notre UML Diagramme métier, que nous avons converti en tables SQL.

Types d'utilisateur :

Nous avons choisi de définir deux types d'utilisateurs sur le site. Il y a les professeurs, qui ont le rôle d'administrateur, et les élèves, qui ont un rôle de visiteur. Leur type est défini lors de la création du compte, en fonction de ce qui est choisi par l'utilisateur… voir le readme pour avoir les identifiants afin de se connecter en tant que professeur ou étudiant

Professeur :

C'est une classe qui est une extension de la classe Utilisateur. Un professeur est un utilisateur dont la variable TYPE est initiée à Professeur. Il possède une (ou plusieurs?) matières qu'il enseigne et pour laquelle il peut créer des cours qui seront suivis par les étudiants qui suivent cette matière. Il peut également créer un nouveau forum et y envoyer des messages. Étant administrateur, il peut également supprimer les forums et/ou messages qui posent problème. Il peut aussi consulter la page "userpage" qui récapitule ses informations personnelles tel que son nom/prénom etc, et également son type d'utilisateur (s'il est professeur ou étudiant) où il peut modifier ses informations s'il le souhaite. Il peut aussi voir tous les autres utilisateurs et peut les modifier ou les supprimer.

Etudiant :

C'est une classe qui est une extension de la classe Utilisateur. Un étudiant est un utilisateur dont la variable TYPE est initialisée à Étudiant. Il suit une ou plusieurs matières de cours sur lesquelles il peut consulter le cours et ses diapositives, ainsi que répondre à des QCM pour définir son niveau. Il peut également créer un nouveau forum et y envoyer des messages. Il peut aussi consulter la page "userpage" qui récapitule ses informations

personnelles tel que son nom/prénom etc, et également son type d'utilisateur (s'il est professeur ou étudiant) où il peut modifier ses informations s'il le souhaite.

Cheminement application web :

Lorsqu'on lance notre application web, on arrive sur la page de login qui nous permet aussi de créer un compte pour se connecter sur le site. Une fois le compte créé, nous arrivons directement sur un QCM nous demandant notre niveau d'étude, et les matières que l'on aime/ que l'on souhaite suivre. à la validation de celui-ci, des matières recommandées ont été ajoutées sur la page des matières suivies.

Partie apprenant :

Les apprenants n'ont pas autant de liberté que les administrateurs / professeurs. Ceux-ci ne peuvent créer leurs cours, ou un forum, en revanche ils peuvent créer des canaux de discussions dans un forum et envoyer des messages sur les forums déjà existants. En revanche, ils peuvent mettre des cours en favori, répondre aux différents QCM proposés par les professeurs et avoir une note déterminant le nombre de bonnes réponses sur le nombre de questions posées.

Partie administrateur:

L'interface d'un administrateur est la même que pour un étudiant à la différence près qu'il a accès à des boutons permettant de gérer différents objets tel la création d'une matière ou bien la suppression de celle-ci, la création d'un forum, la mise en place d'un QCM dans une de ses matière, Supprimer un utilisateur etc… Pour les QCM, le professeur doit importer un fichier xml contenant une structure très spécifique, puis une fois fait, il doit répondre aux questions de son QCM ce qui génère un fichier réponse en xml qui servira pour la vérification des réponses apportées par les étudiants.

Difficultés et améliorations :

La plus grande difficultée dans ce projet à été le nombre de personnes participant à l'élaboration du site. C'est la première fois que nous faisons un projet à plus de 2-3 personnes, pour pouvoir s'entendre sur ce qu'il fallait faire, nous devions organiser des meetings, et se tenir à jour sur l'avancée du projet pour ne pas se perdre dans les différentes modification et amélioration apportés au fur et à mesure du temps. La répartition des tâches aussi est plus ardue à mettre en place lorsqu'on est plusieurs à travailler en même temps sur le même projet, il ne faut pas que l'on se marche dessus.

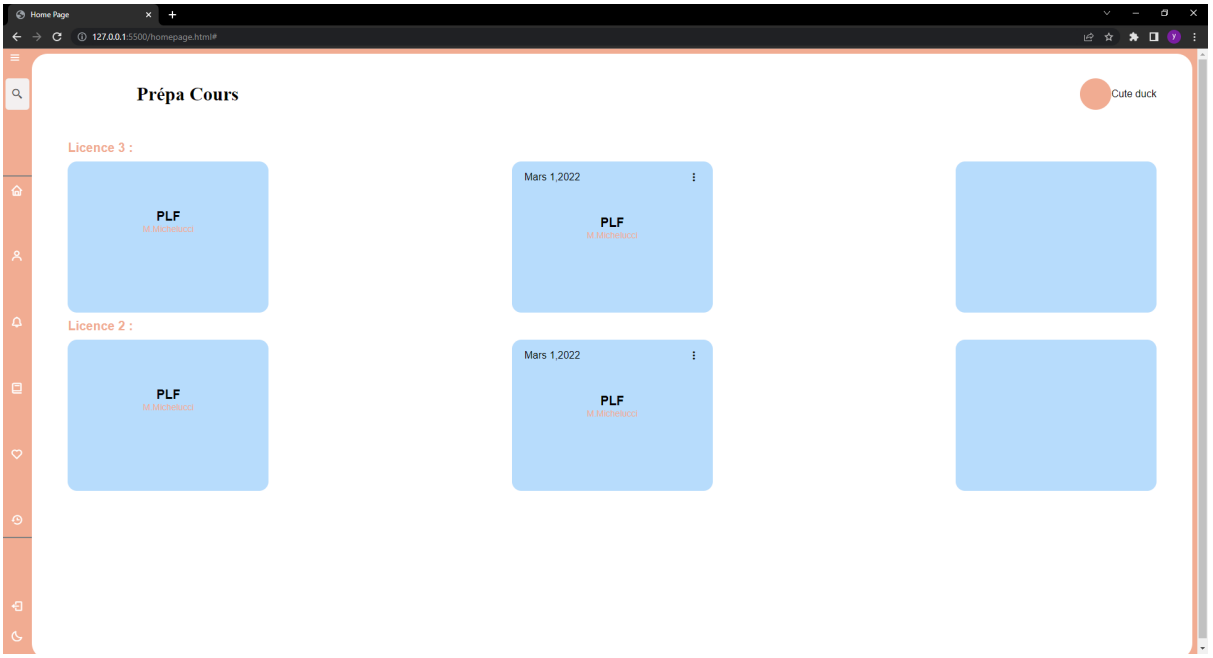
Le sujet est assez vague pour nous permettre de faire un peu ce que l'on souhaite mais c'est aussi un piège car nous ne savions pas exactement par où commencer et comment mettre en place l'architecture du site.

C’est aussi la première fois que nous touchions à du php et pour la plupart d’entre nous, nous n’avions pas touché au html/css/js depuis 2 ans minimum ce qui à été un gros inconvénient lorsqu’on a commencé l’interface utilisateur.

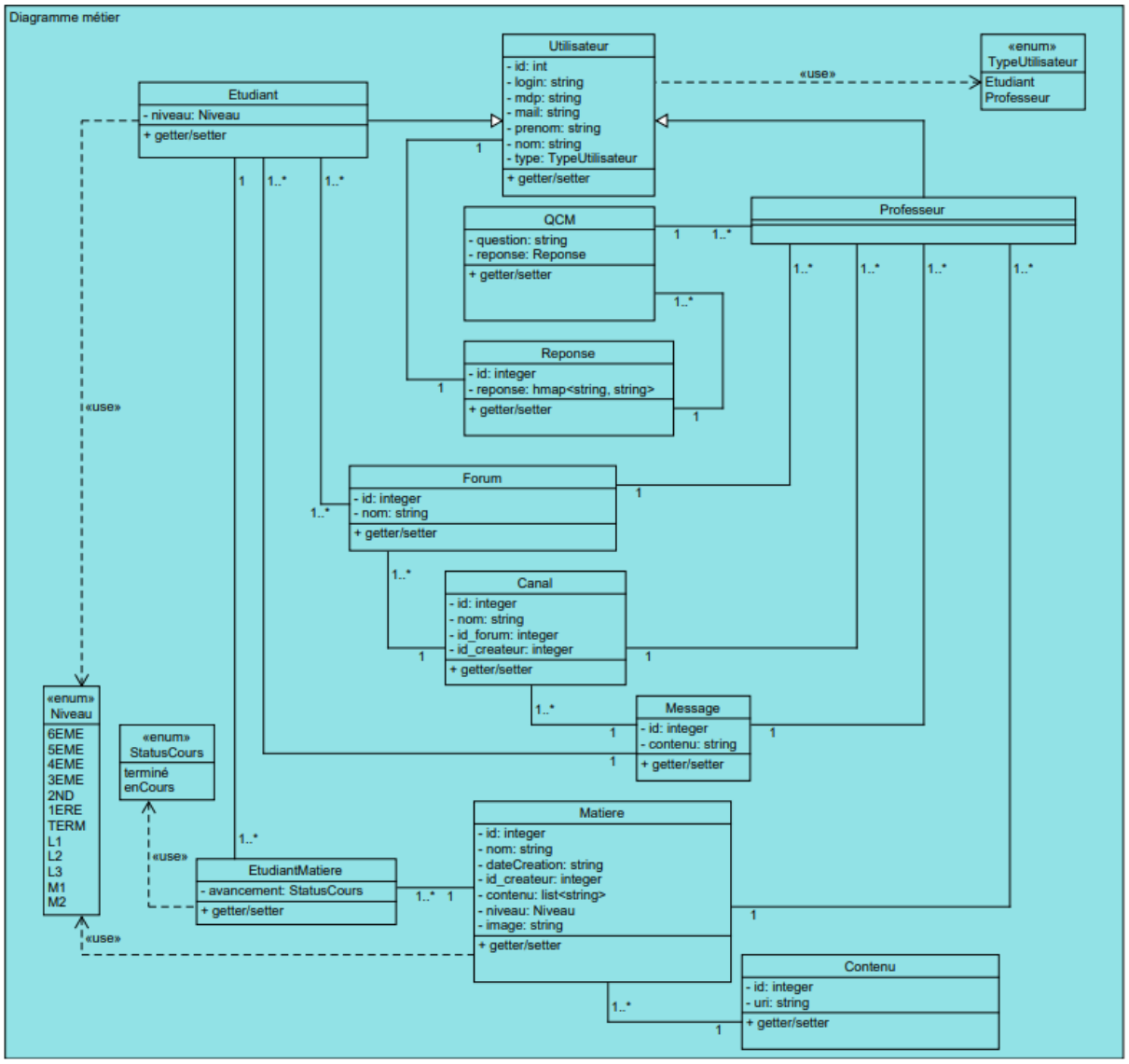
Pour les améliorations, notre site est loin d’être parfait. Quelques pages n’ont pas ce CSS. On ne peut pas ajouter de vidéos à notre site pour la partie cours et les QCM ne sont pas modulables : La structure d’un fichier xml pour le css est stricte. La sécurité de notre site est à revoir, par exemple un utilisateur est considéré comme un administrateur dans notre base de données. Il n’y a pas non plus de modération au niveau de notre forum (pas de suppression de message possible, pas d’avertissement et tout autre chose permettant le bon déroulement d’un forum). Un étudiant ne peut pas finir une matière et ne peut pas non plus arrêter de suivre une matière. Il n’est pas non plus possible de se désinscrire du site

Annexe :

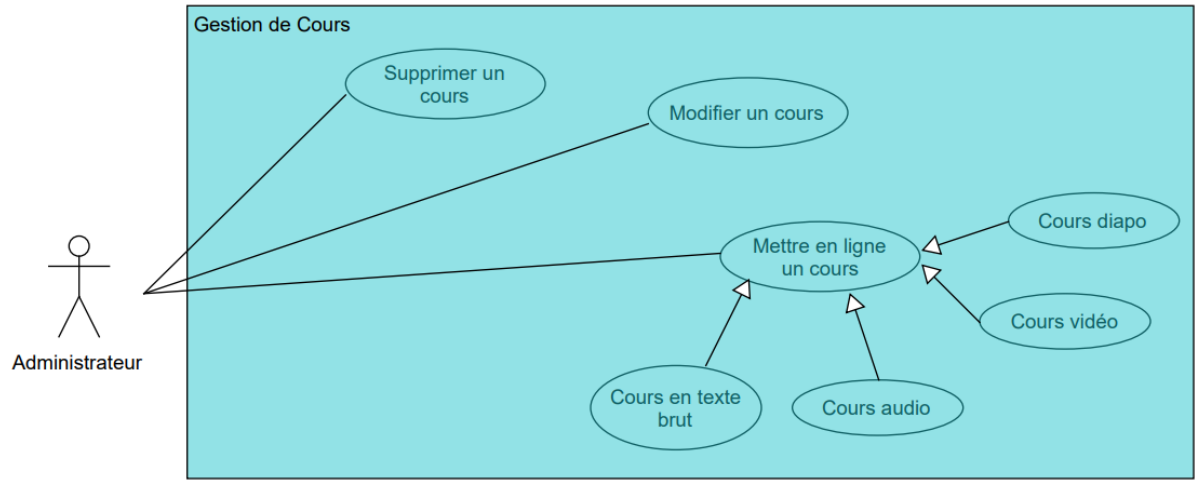
Annexe 1 : Homepage v1.



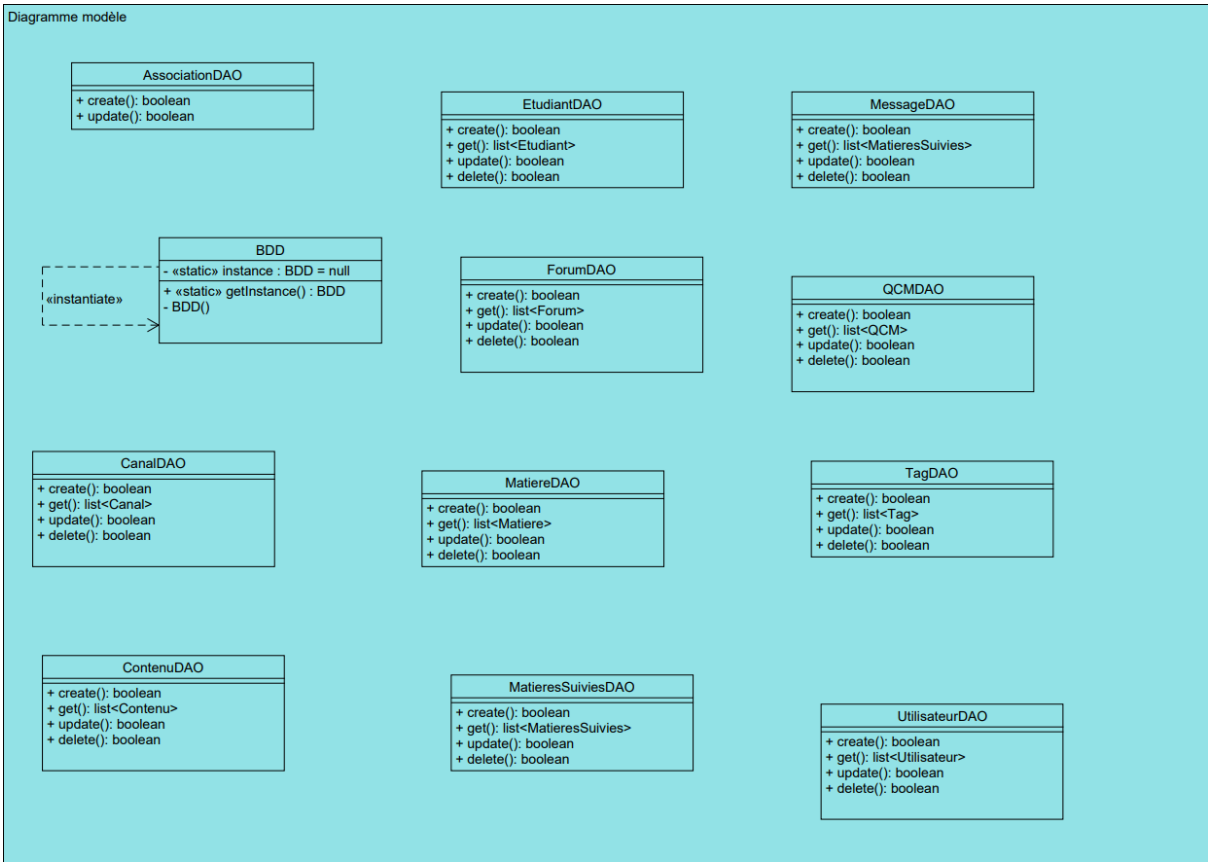
Annexe 2 : Diagramme métier.



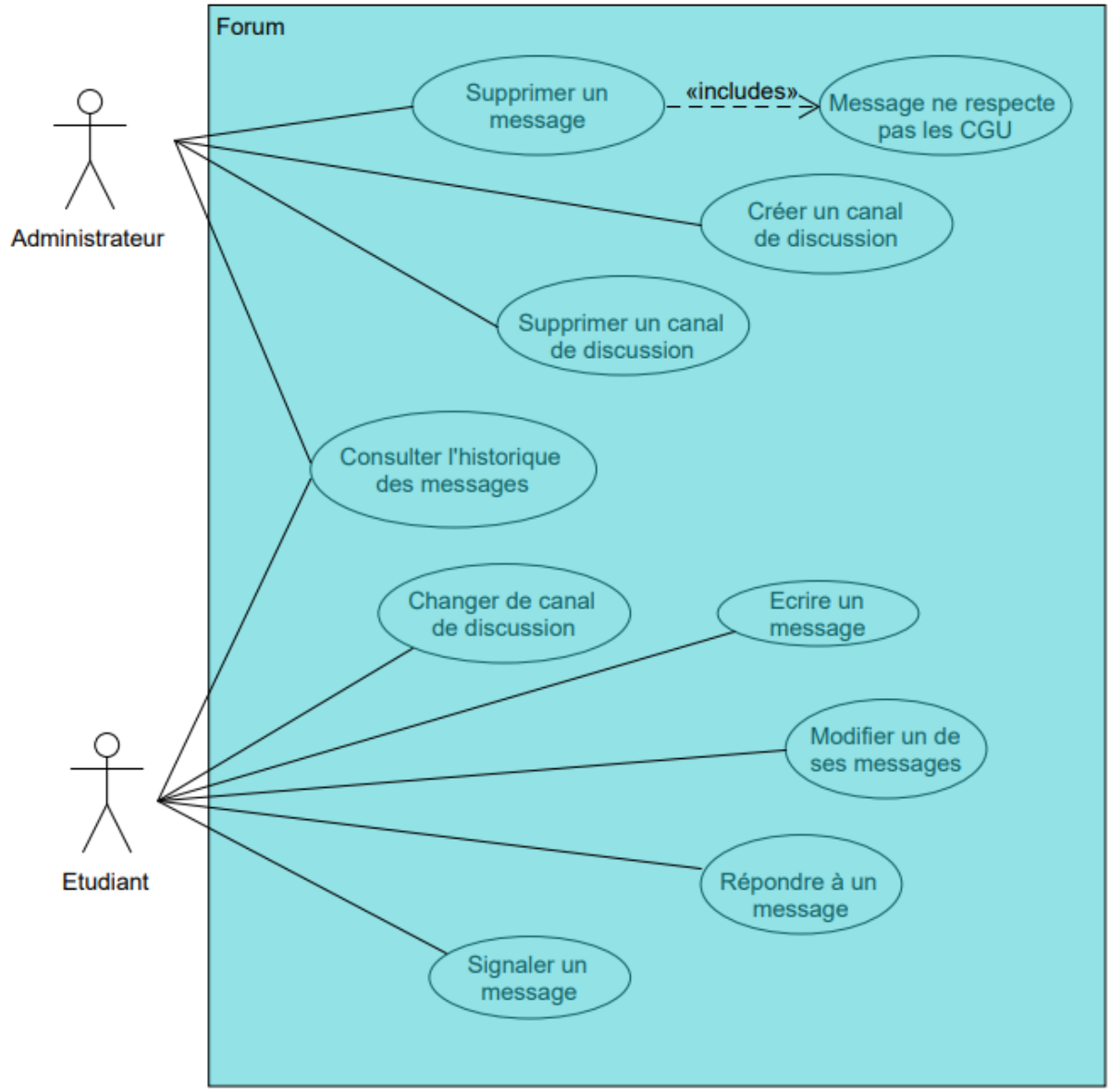
Annexe 3 : Gestion de Cours



Annexe 4 : Diagramme modèle



Annexe 5 : Forum



Annexe 6 : Use Case Site de formation