Analyse et conception

3.1 Introduction

L'analyse et conception sont les deux étapes fondamentales dans le processus de développement, donc dans ce chapitre nous commençons par établir les diagrammes de classes participantes qu'il s'agit de diagrammes de classes UML qui décrivent cas d'utilisation par cas d'utilisation les trois types de classes d'analyse, les dialogues, les contrôles et les entités ainsi que leurs relations, ces diagrammes sont regroupés pour obtenir le diagramme de classes global. Aussi nous présentons les diagrammes de navigation qui permettent de modéliser la navigation entre les différents liens des IHMs.

3.2 Diagramme de classes participantes

Les diagrammes de classes participantes (DCPs) sont particulièrement importants car ils font la jonction entre les cas d'utilisation, les maquettes et les diagrammes de conception logicielle (diagrammes d'interaction et diagrammes de classes).

3.2.1 Principe et démarche

Pour réaliser les diagrammes de classes participantes il faut premièrement identifier les concepts du domaine et les classes d'analyses puis ajouter les associations et les attributs.

1. **Identification des concepts du domaine et des classes d'analyses**
   L'étape typiquement orientée objet de l'analyse est la décomposition d'un domaine d'intérêt en classes conceptuelles représentant les entités significatives de ce domaine. Il s'agit simplement de créer une représentation visuelle des objets du monde réel dans un domaine donné [24].
   Si on emploie la notation UML, il s'agit d'un ensemble de diagrammes de classes dans lesquels on fait figurer les éléments suivants [24] :
   - Les classes conceptuelles ou les objets du domaine.
   - Les associations entre classes conceptuelles.
   - Les attributs des classes conceptuelles.

L'identification des concepts du domaine est élargie en utilisant une catégorisation des classes d'analyse qui a été proposée par I. Jacobson et popularisée ensuite par le RUP (Rational Unified Process).
Les classes d'analyse qu'ils préconisent se répartissent en trois catégories [24] :

(a) *Les dialogues* ce sont des écrans qui permettent les interactions entre les utilisateurs et l'application. Ils sont issus de la maquette. Ils possèdent des attributs et des opérations.

(b) *Les contrôles* correspondant à la logique interne de l'application. Elles font le lien entre les classes dialogue et les classes métier et vont seulement posséder des opérations.

(c) *Les entités* représentent les informations persistantes de l'application, permettent à des données et des relations d'être stockées dans des fichiers ou des bases de données. Les entités vont seulement posséder des attributs.

2. **Ajout des associations et des attributs**
Après l'identification des concepts du domaine on ajoute des associations entre les classes d'analyse et les attribues.

3.2.2 DCP des cas d'utilisation de notre système

Nous allons présenter les DCPs des principaux cas d'utilisation que nous avons identifiés pour notre système.

1. **DCP de l'UC *Ajouter cours***
L'*enseignant* peut ajouter un cours pour l'afficher aux étudiants. A ce moment-là l'*enseignant* passe par un seul écran géré par un contrôle unique. Le DCP de l'UC *ajouter cours* est présenté dans la figure 3.1.
   - L'écran formulaire d'ajout qui lui permet de saisir les informations de cours (titre, description, fichier, etc.) avec le choix de visibilité de cours (visible ou invisible).
   - Le contrôle *(CtrD'ajout)* contient les opérations suivantes : la vérification des champs de saisie, la vérification de l'existence de fichier, l'affichage de notification en cas d'échec, l'ajout en cas la validité des informations avec une notification de succès.

   Cet UC nous permet d'identifier trois classes entités : *Enseignant*, *Cours*, *Module*.

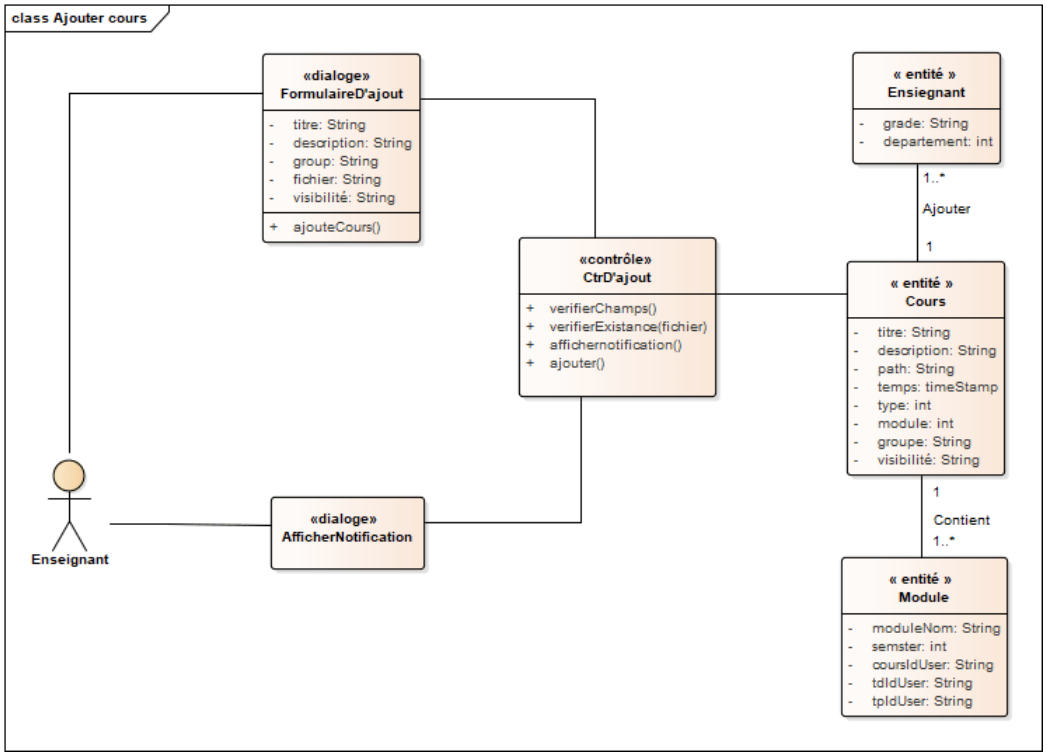


FIGURE 3.1 – DCP de l'UC *Ajouter cours*

2. **DCP de l'UC *Consulter liste étudiants***
   L'*enseignant* peut consulter la liste des étudiants. Le DPC de l'UC *consulter liste étudiants* est présenté dans la figure 3.2.
   Pour consulter la liste, l'*enseignant* passe par deux écrans gérés par un contrôle unique.
   - L'écran choix groupe qui lui permet de préciser le niveau et choisir un groupe.
   - L'écran liste étudiants affiche les noms des étudiants du groupe sélectionné par l'*enseignant*.
   - Le contrôle *(CtrDétailles)* qui contient l'opération d'affichage.

   Cet UC nous permet d'identifier une classe entité : *Étudiant*.

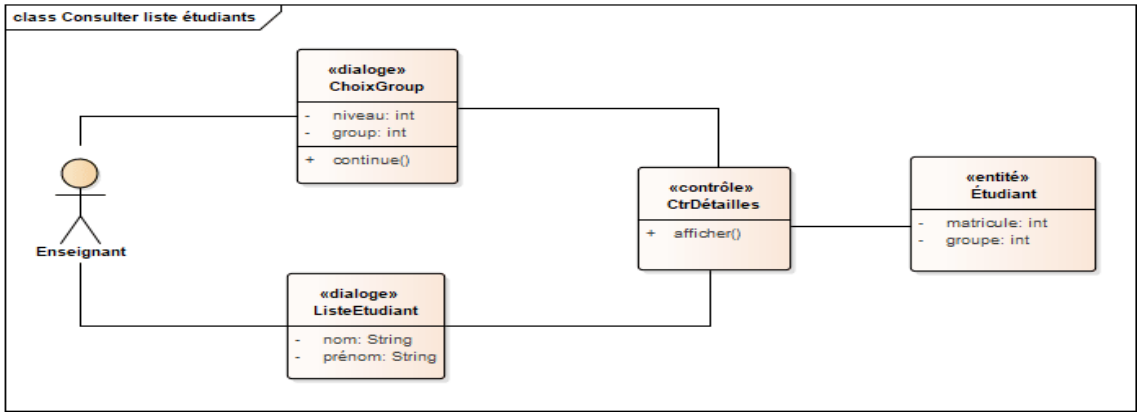


FIGURE 3.2 – DCP de l'UC *Consulter liste étudiants*

3. **DCP de l'UC *Ajouter compte***
   L'*administrateur* peut ajouter un compte d'un utilisateur, alors il passe par un seul écran géré par un contrôle unique. Le DCP de l'UC *ajouter compte* est présenté dans la figure ci-dessous.
   - L'écran formulaire d'ajout qui lui permet de saisir les informations de compte.
   - Le contrôle *(CtrD'ajout)* contient les opérations suivantes : la vérification des champs de saisie, la vérification de l'existence de compte, l'affichage de notification en cas d'échec, l'ajout de compte avec une notification de succès en cas la validité des informations.

   Cet UC nous permet d'identifier quatre classes entités : *Utilisateur*, *Étudiant*, *Enseignant*, *Compte*.

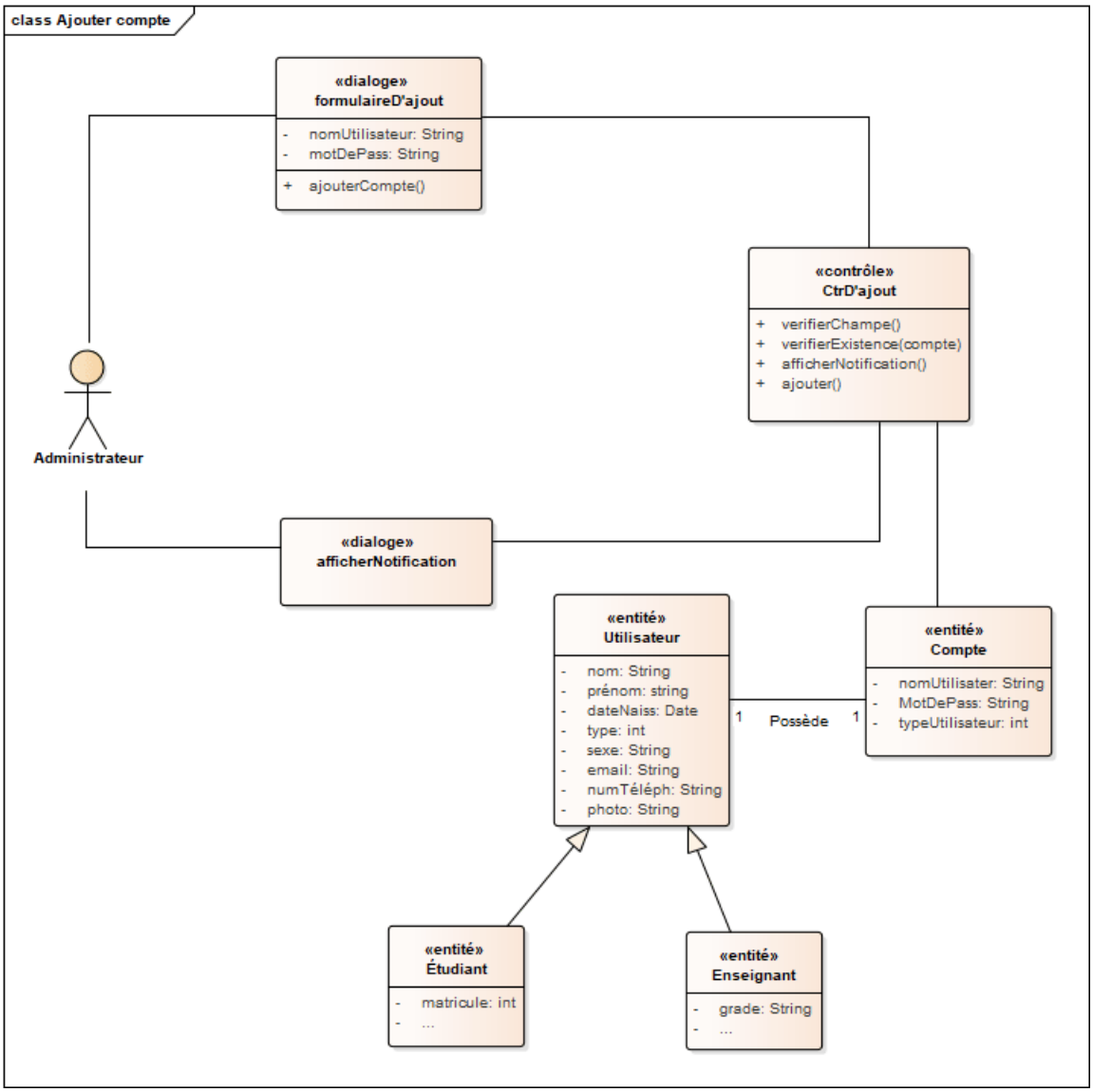


FIGURE 3.3 – DCP de l'UC *Ajouter compte*

4. **DCP de l'UC *Ajouter sujet de discussion***
   Chaque *utilisateur* à la possibilité d'ajouter un sujet de discussion. Pour que l'utilisateur ajoute un sujet il passe par trois écrans gérés par un seul contrôle. Le DCP de l'UC *ajouter sujet discussion* est présenté dans la figure 3.4.
   - L'écran choix type qui lui permet de choisir le type de sujet.
   - L'écran liste sujet qui présente les sujets existants.
   - L'écran formulaire d'ajout qui lui permet de saisir le titre et le contenu du sujet.
   - Le contrôle *(CtrD'ajout)* contient les opérations suivantes : la vérification des champs, l'affichage de notification cas d'échec, l'ajout en cas la validité des informations avec une notification de succès.

   Cet UC nous permet d'identifier quatre classes entités : *Utilisateur*, *Étudiant*, *Enseignant*, *Sujet*.

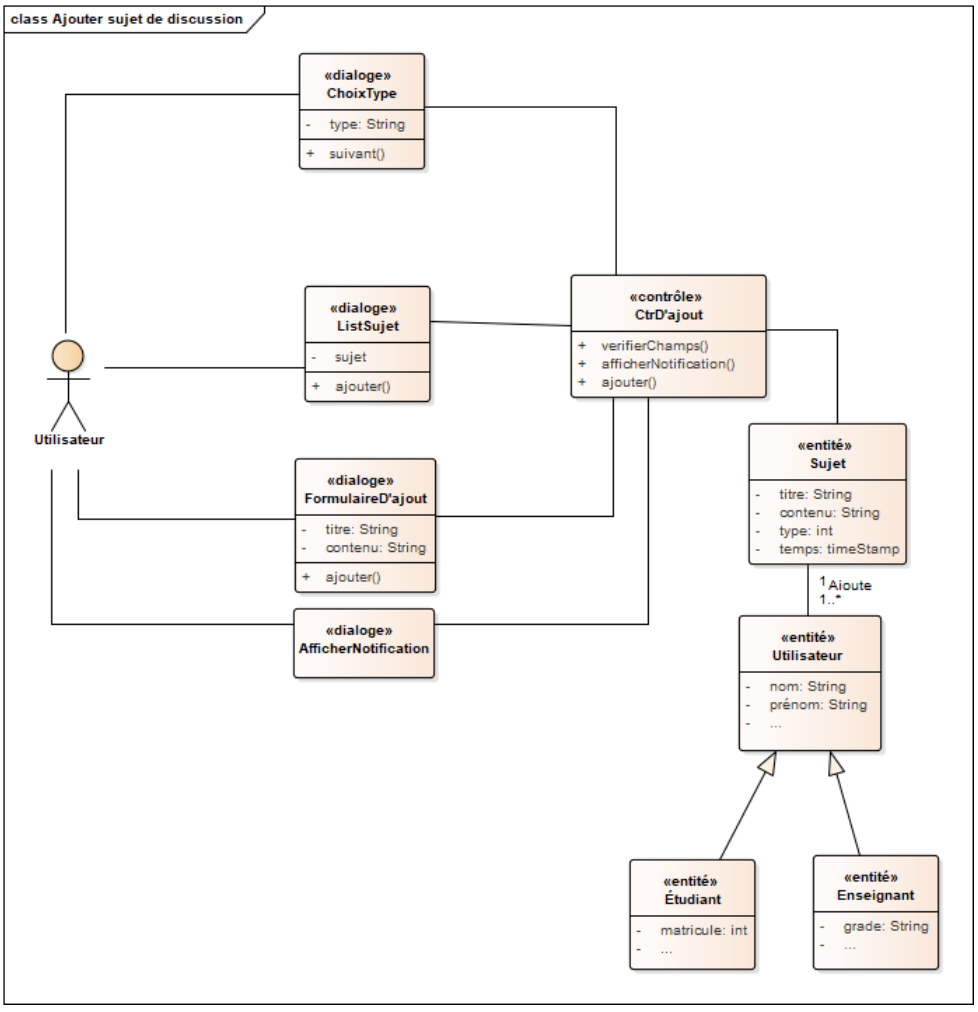


FIGURE 3.4 – DCP de l'UC *Ajouter sujet de discussion*

5. **DCP de l'UC *Modifier profile***
   l'utilisateur peut modifier son profile. Lors de la modification, l'utilisateur passe par un seul

écran géré par un contrôle unique. Le DCP d'UC *modifier profile* est présenté dans la figure 3.5.

- L'écran modifier Profile qui lui permet de saisir les nouvelles informations (photo profile, email, numéro de téléphone).
- le contrôle *(CtrModifier)* contient les opérations suivantes : la vérification des champs, l'affichage de notification si la modification est fait avec succès, l'affichage de notification en cas d'échec.

Cet UC nous permet d'identifier trois classes entités : *Utilisateur*, *Étudiant*, *Enseignant*.

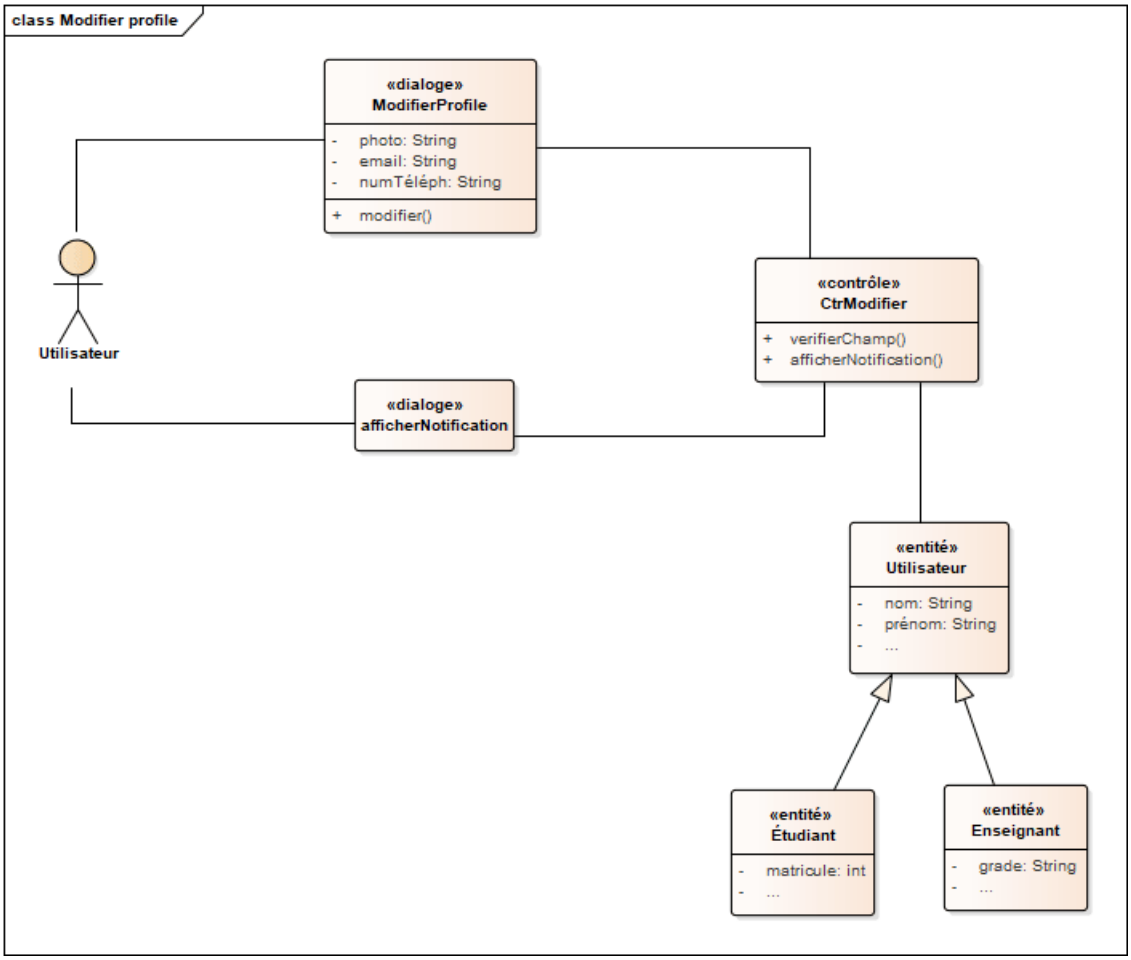


FIGURE 3.5 – DCP de l'UC *Modifier profile*

3.3 Diagramme de classe global

Nous présentons maintenant le diagramme de classe globale qui appartient à la phase de conception. Ce diagramme a été élaboré en regroupant et en raffinant les diagrammes de classes participants (DCPs) identifiés dans la section 3.2.2. Il est représenté dans la figure 3.6.

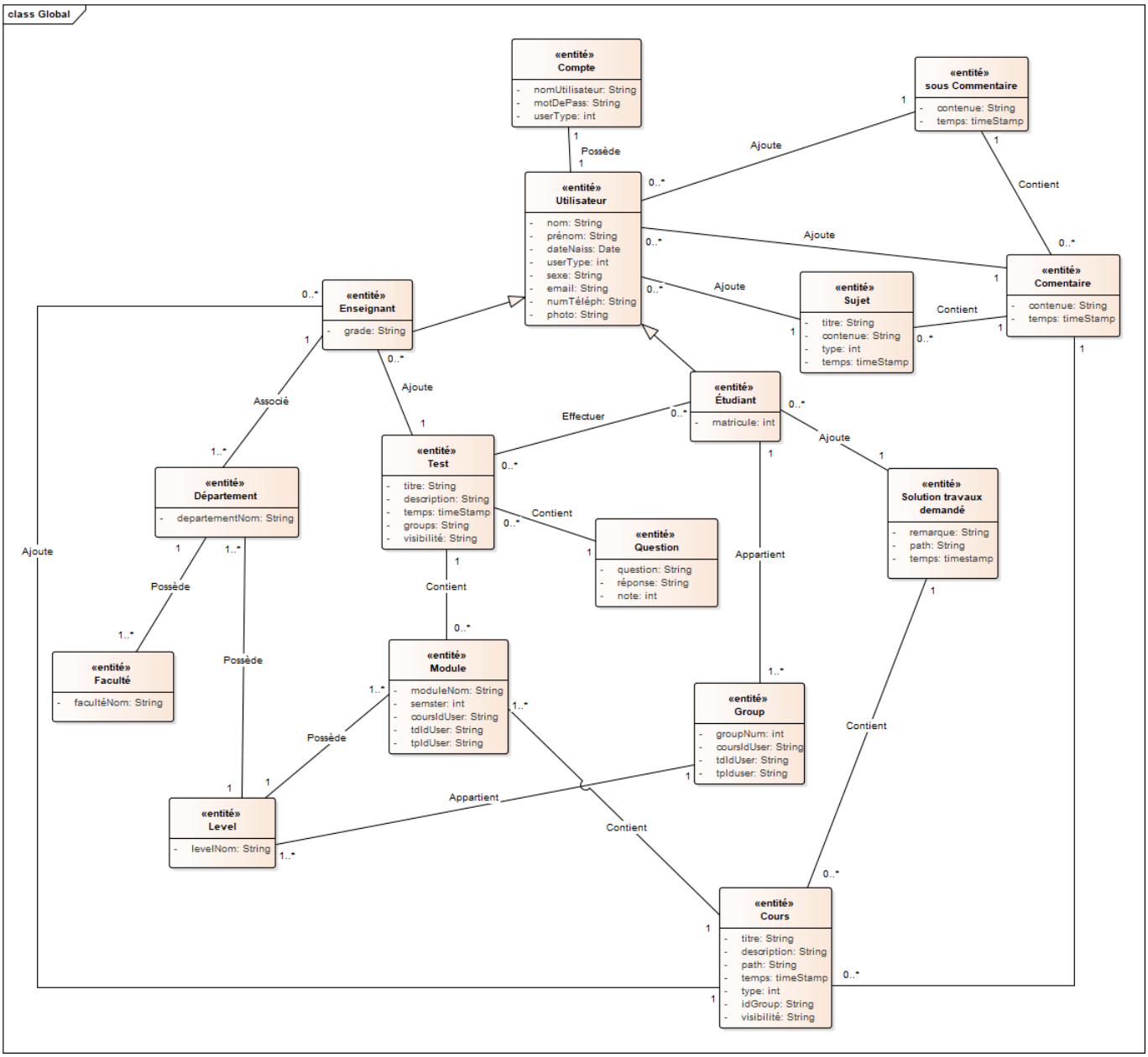


FIGURE 3.6 – Diagramme de classe global

3.4 Diagramme de navigation

Les diagrammes de navigation sont des diagrammes dynamiques représentant de manière formelle l'ensemble des chemins possibles entre les principaux écrans proposés à l'utilisateur [24]. Le diagramme de navigation de la démarche est représenté dans la figure 3.7. Par la suite nous présentons le diagramme de navigation de notre système.

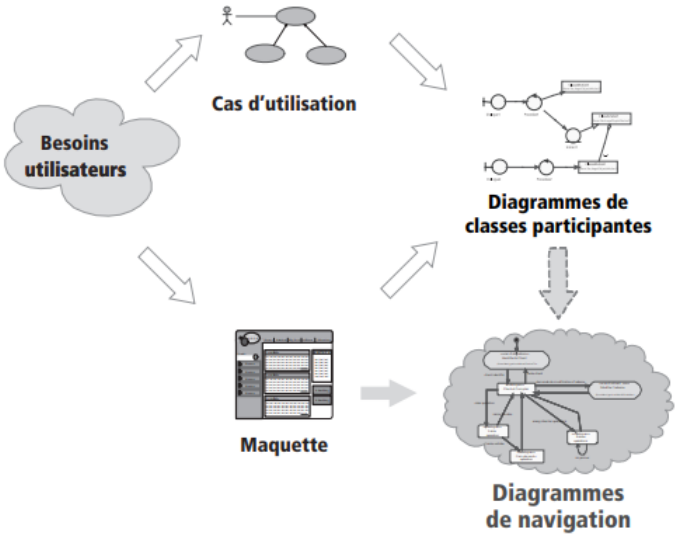
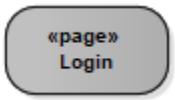


FIGURE 3.7 – Les diagrammes de navigation dans la démarche [24]

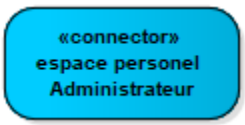
3.4.1 Diagrammes de navigation de notre système

Dans ce qui suit, nous représentons le diagramme de navigation de chaque acteur : *administrateur*, *enseignant* et *étudiant*. Ces diagrammes sont dynamiques, il a pour rôle de modéliser la navigation entre les différents liens des pages web proposés à l'utilisateur. Pour modéliser la navigation, nous allons utiliser un nombre restreint d'éléments standard, à savoir [24] :

- Les états représentent les classes dialogues. Pour modéliser les différents concepts des états on utilise les conventions graphiques suivantes :
  — Une page complète du site ou de l'application («page» avec la couleur gris).



  — Une liaison vers un autre diagramme d'activité, pour des raisons de structuration et de lisibilité («connector» avec la couleur bleu).



- Les transitions entre états déclenchées par des événements et pouvant porter des conditions, pour représenter les actions IHM.

1. **Diagramme de navigation de l'*administrateur***
   Selon la figure 3.8, l'*administrateur* saisit son nom d'utilisateur et son mot de passe dans la

page d'authentification. Si les informations d'authentification sont correctes l'*administrateur* sera dirigé vers son espace personnelle, sinon un message d'échec sera affiché.

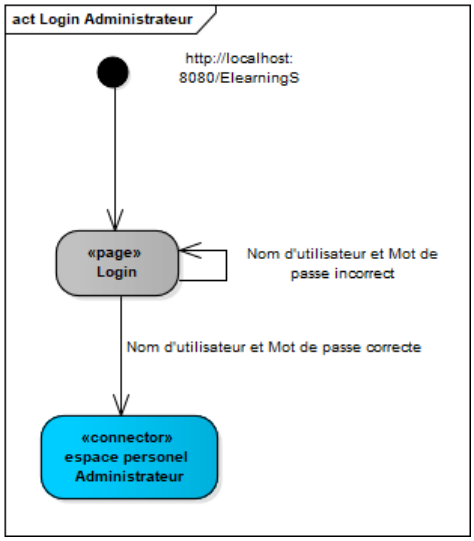


FIGURE 3.8 – Début de diagramme de navigation d'*administrateur*

À partir de la page d'accueil, l'*administrateur* peut flâner dans un ensemble de pages : faculté, département, niveau, module, compte, groupe et utilisateurs. Cette dernière est composée de deux pages dont chacune affiche les détails d'un type d'utilisateur (enseignant ou étudiant). L'*administrateur* a la possibilité d'ajouter des modules, des utilisateurs, des facultés etc, les modifier ou les supprimer. La figure suivante représente le diagramme de navigation de l'*administrateur* :

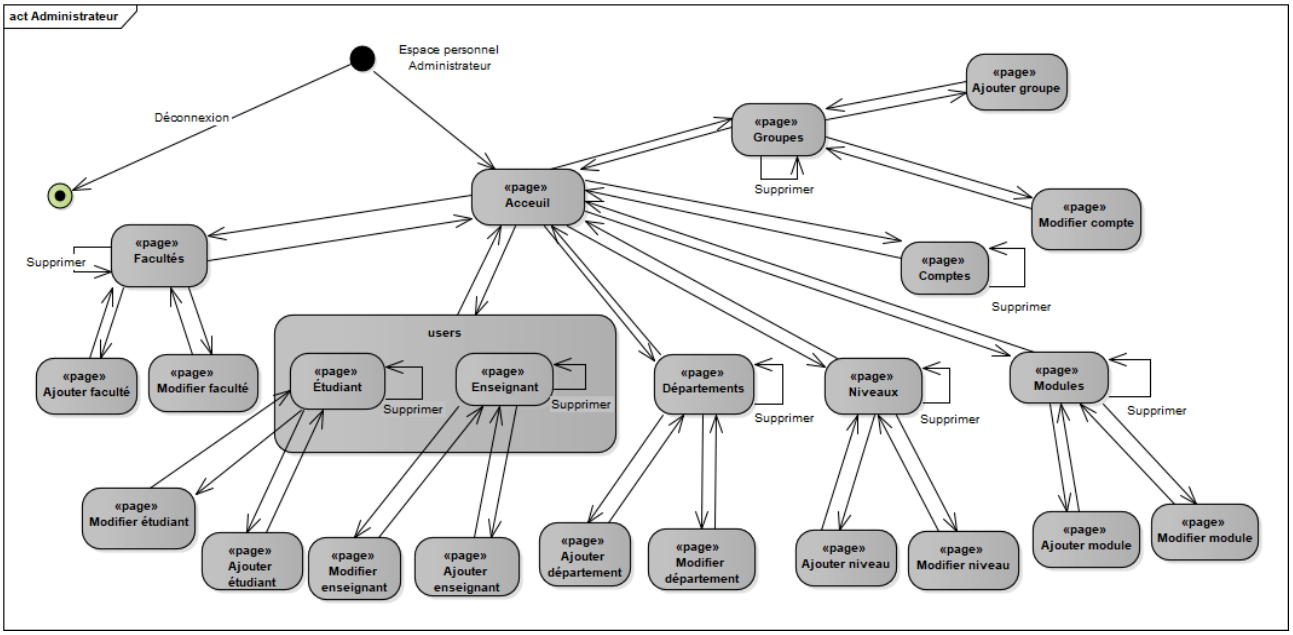


FIGURE 3.9 – Diagramme de navigation de l'*administrateur*

2. **Diagramme de navigation de l'utilisateur**
   Selon la Figure 3.10 l'utilisateur saisit son nom d'utilisateur et son mot de passe dans la page d'authentification de l'application. Si les informations d'authentification sont correctes l'utilisateur sera dirigé vers son espace personnel (soit en tant que *enseignant* ou bien un *étudiant*), sinon un message d'échec sera affiché.

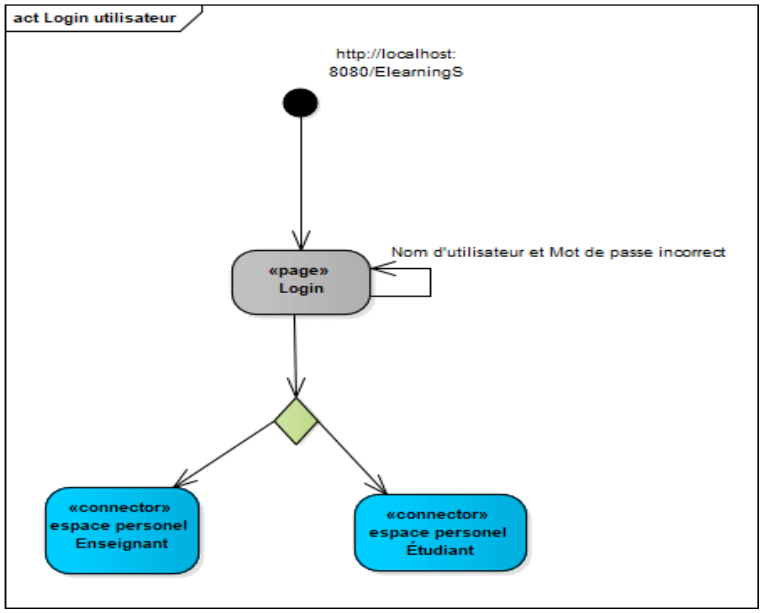


FIGURE 3.10 – Début de diagramme de navigation d'utilisateur

(a) ***Enseignant***
Comme le montre la figure 3.11 qui présente le diagramme de navigation de l'*enseignant*. Une fois l'*enseignant* est dirigé vers son espace personnel, il peut naviguer entre différentes pages, telles que la liste des sujets de discussions, la liste de ses étudiants, la liste des cours, etc.

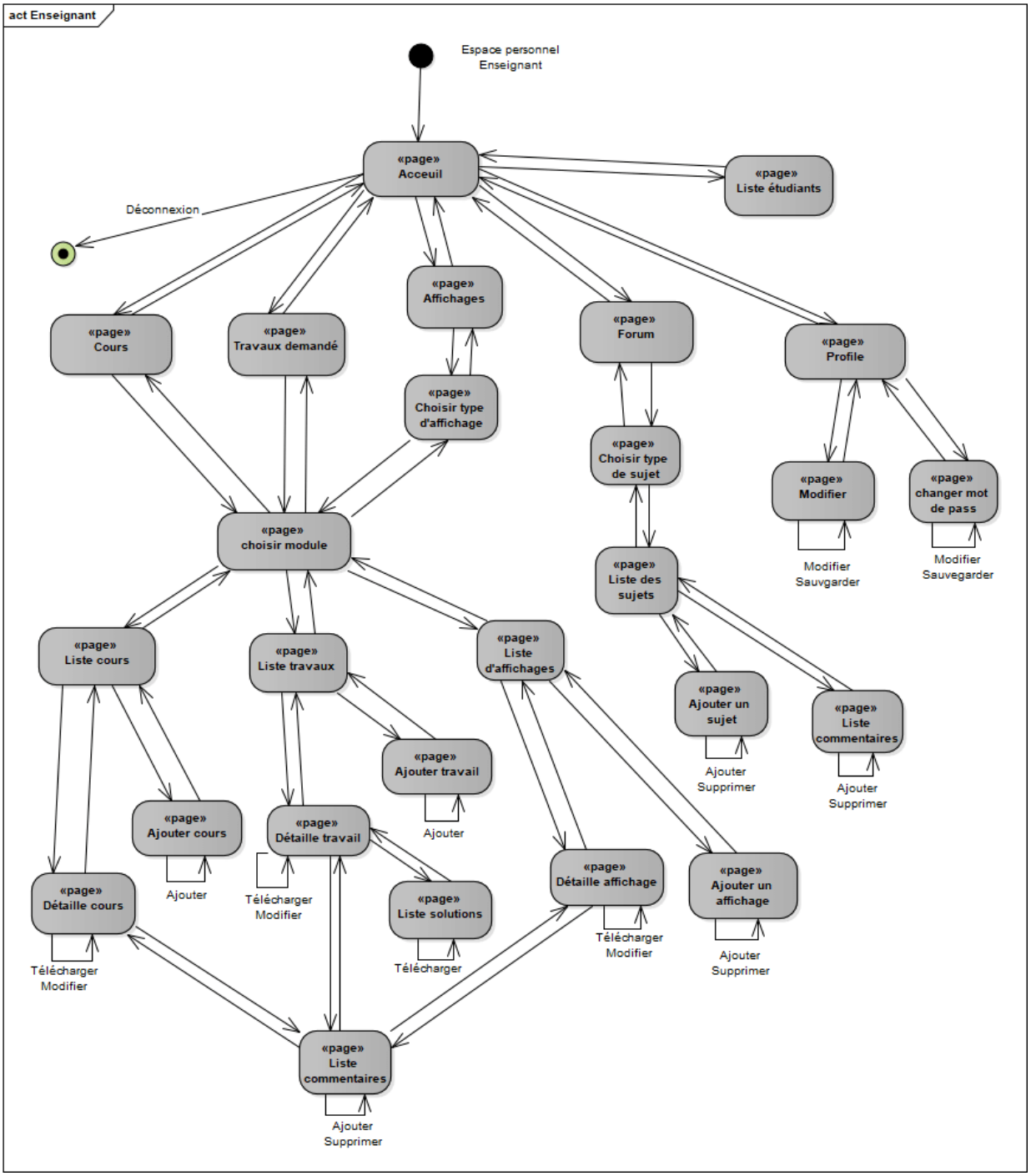


FIGURE 3.11 – Diagramme de navigation de l'*enseignant*

(b) ***Étudiant***

Comme le montre la Figure 3.12 qui présente le diagramme de navigation de l'*étudiant*. Une fois ce dernier est dirigé vers son espace personnel, il peut naviguer entre différentes pages, telles que la liste des sujets de discussions, la liste des cours, la liste d'affichage etc.

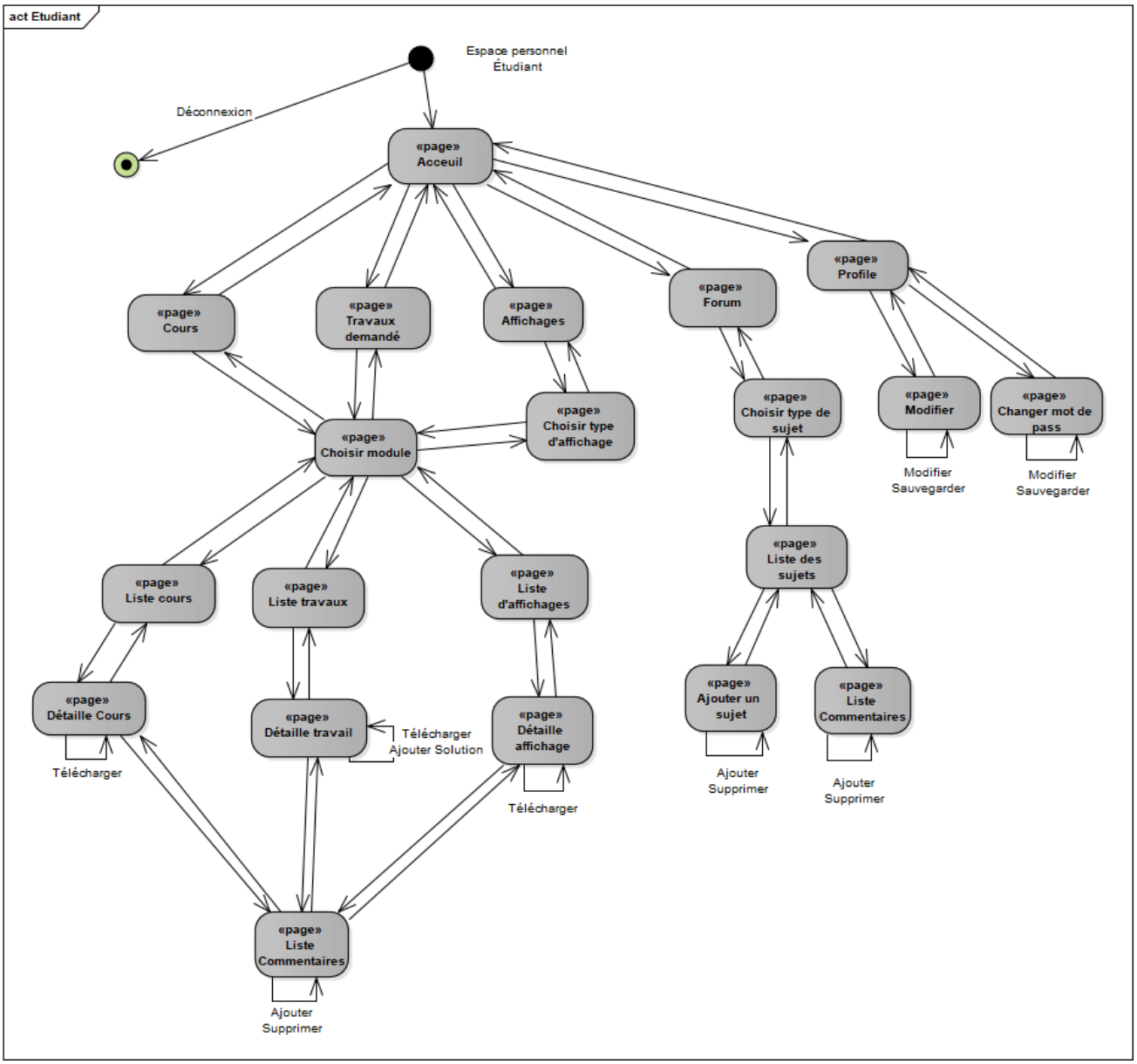


FIGURE 3.12 – Diagramme de navigation de l'*étudiant*

3.5 Conclusion

Dans ce chapitre, nous avons présenté les diagrammes de classes participantes des cas d'utilisation de notre système. Puis nous avons regroupé ces diagrammes pour construire le diagramme de classe global. Ensuite, et à l'aide de diagramme de navigation nous avons modélisé la partie dynamique du système.

Chapitre 4

Réalisation

4.1 Introduction

Ce dernier chapitre dédié à la réalisation de notre application. L'objectif de ce chapitre est d'exposer les différents choix techniques : langages, environnements et les outils de développement utilisés, ainsi que les plates-formes de développement choisies. Et à la fin, nous présentons l'architecture de notre application.

4.2 Choix techniques

- **Langage JAVA :** est un langage de programmation orienté objet développé par Sun Microsystems, aujourd'hui racheté par Oracle. Il permet de développer des logiciels fonctionne sous Windows, Mac, Linux, etc. il permet aussi de développer des programmes pour téléphones portables [29].
- **HTML :** (HyperText Markup Language) il a fait son apparition dès 1991 lors du lancement du web. Son rôle est de gérer et organiser le contenu. C'est donc en HTML que vous écrirez ce qui doit être affiché sur la page : du texte, des liens, des images etc [20].
- **CSS :** signifie Cascading Style Sheets, soit « feuilles de style en cascade ». Il a été créé en 1996 et a pour rôle de mettre en forme du contenu en lui appliquant ce qu'on appelle des styles [30].
- **JavaScript :** un langage de programmation initialement introduit dans les navigateurs web afin de rendre les pages HTML plus dynamiques dans leurs interactions avec l'utilisateur. Ce langage a été développé par la société Netscape Corporation, qui l'a introduit, pour la première fois dans son navigator 2.0 en 1996 [21].
- **Java EE :** la plate-forme Java Enterprise Edition fournit une plate-forme basée sur des normes pour le développement d'applications web et d'entreprise. Ces applications sont généralement signées comme des applications de trois couches (selon le modèle MVC)[31]. Ces trois couches sont [32]
  1. **Le modèle :** contient les données manipulées et conservées par le programme. Son rôle se base sur la récupération des informations à partir de la base de données (via des requêtes SQL) pour qu'elles puissent être traitées par le contrôleur.

2. **Vue (JSP) :** est une technique permet aux développeurs de générer dynamiquement du code HTML, XML ou tout autre type de page web. Une page JSP aura un contenu pouvant être différent selon certains paramètres (des informations stockées dans une base de données etc) tandis qu'une page HTML statique affiche continuellement la même information.
3. **Contrôleur (Servlet) :** sont des programmes Java fonctionnant côté serveur elles permettent de recevoir des requêtes HTTP, de les traiter et de fournir une réponse dynamique au client.

- **JSON :** (JavaScript Object Notation) est un langage léger d'échange de données textuelles qui est complètement indépendant du langage mais utilise des conventions qui sont familières aux programmeurs de la famille C des langages, y compris C, C#, Java, JavaScript, Perl, Python, et bien d'autres. Ces propriétés font de JSON un langage idéal pour l'échange de données [33].
- **Apache Tomcat :** le logiciel Apache Tomcat est une implémentation open source des technologies Java Servlet, JavaServer Pages, Java Expression Language et Java WebSocket. Il est développé dans un environnement ouvert et participatif et publié sous la licence Apache version 2 [34].
- **Eclipse :** est un environnement intégré de développement (IDE) pour le langage Java(et d'autres langages), Il est développé par IBM, gratuit et open source publié sous les termes de la licence publique Eclipse 2.0 [35].
- **Volley :** Volley est une bibliothèque HTTP qui rend la mise en réseau très facile et rapide, pour les applications Android. Elle a été développée par Google et présentée au cours de Google I / O 2013. Elle a été développée parce qu'il y a une absence dans Android SDK, d'une classe de réseau capable de travailler sans interférer avec l'expérience utilisateur. Il offre les avantages suivants [36] :
  1. Programmation automatique des requêtes réseau.
  2. Plusieurs connexions réseau simultanées.
  3. Volley fournit un cache disque et mémoire transparent.
- **MYSQL :** est un système de gestion de bases de données relationnelles (SGBDR) robuste et rapide. Une base de données permet de manipuler les informations de manière efficace, de les enregistrer, de les trier, de les lire et d'y effectuer des recherches. MySQL est un serveur multi-utilisateur et multithread. Il utilise SQL (Structured Query Language), le langage standard des requêtes de bases de données [22].
- **AppServ :** est un logiciel complet d'Apache, MySQL, PHP, phpMyAdmin il permet d'accéder à un autre serveur web et serveur de base de données, qui est configuré localement. Cela permet d'effectuer tous les tests nécessaires sur le site web avant de le lancer en ligne [37].
- **Android Studio :** est l'environnement intégré (IDE) officiel pour le développement d'applications Android. En tant qu'environnement de développement intégré officiel de Google, Android Studio inclut tout ce dont le développeur besoin pour développer une application : éditeur de code et déboguer intelligents, outils d'analyse des performances, émulateurs, et bien plus [36].
- **SDK Android :** le SDK Android (kit de développement logiciel) est un ensemble d'outils de développement utilisés pour développer des applications pour la plate-forme Android.

- **AVD Android :** un appareil virtuel Android (AVD) est une configuration qui définit les caractéristiques d'un téléphone Android, d'une tablette, d'un système d'exploitation Wear, d'Android TV ou d'un système d'exploitation automobile que vous souhaitez simuler dans l'émulateur Android. AVD Manager est une interface que vous pouvez lancer à partir d'Android Studio qui vous aide à créer et à gérer des AVD [36].
- **AJAX :** Ajax n'est pas un langage de programmation ni même une technologie. En effet, la programmation Ajax est une combinaison de JavaScript côté client avec des transferts de données formatés en XML et une programmation côté serveur avec des langages comme PHP. Généralement, la programmation Ajax produit une interface utilisateur plus propre et plus rapide pour les applications interactives parce qu'elle permet aux utilisateurs d'effectuer de nombreuses tâches sans avoir besoin de recharger ou de réafficher des pages entières [22].
- **Entreprise Architect :** est un outil graphique multi-utilisateurs conçu pour aider les développeurs à construire des systèmes robustes et maintenables. Et en utilisant des rapports et une documentation intégrée de haute qualité, vous pouvez offrir une vision vraiment partagée facilement et avec précision [38].
- **Tailwind CSS :** est un framework CSS de bas niveau hautement personnalisable qui vous donne tous les éléments de base dont vous avez besoin pour créer des conceptions sur mesure sans styles d'opinion ennuyeux que vous devez vous battre pour remplacer [39].
- **Adobe XD :** est une solution d'UX/UI design complète pour la conception des sites web, d'applications mobiles etc. Alliant rapidité, précision et qualité, XD permet aux designers de modifier et partager facilement des prototypes interactifs avec collaborateurs et réviseurs sur l'ensemble des appareils et plates-formes, dont Windows, Mac, iOS et Android [40].
- **FCM :** Firebase Cloud Messaging (FCM) est une solution de messagerie multiplateforme qui permet d'envoyer des messages de manière fiable et sans frais. À l'aide de FCM, l'application cliente peut être notifiée d'un nouvel e-mail ou d'autres données disponibles pour la synchronisation. Pour les cas d'utilisation tels que la messagerie instantanée, un message peut transférer une charge utile allant jusqu'à 4 Ko vers une application cliente [41].

4.3 Architecture du système

Cette section constitue une présentation de l'architecture de notre système "*UniverPack*". Elle représente l'implémentation de l'architecture client/serveur en utilisant la technologie JEE web et en respectant le modèle MVC. Nous allons détailler l'architecture de ce système qui est présenté dans la figure suivante :

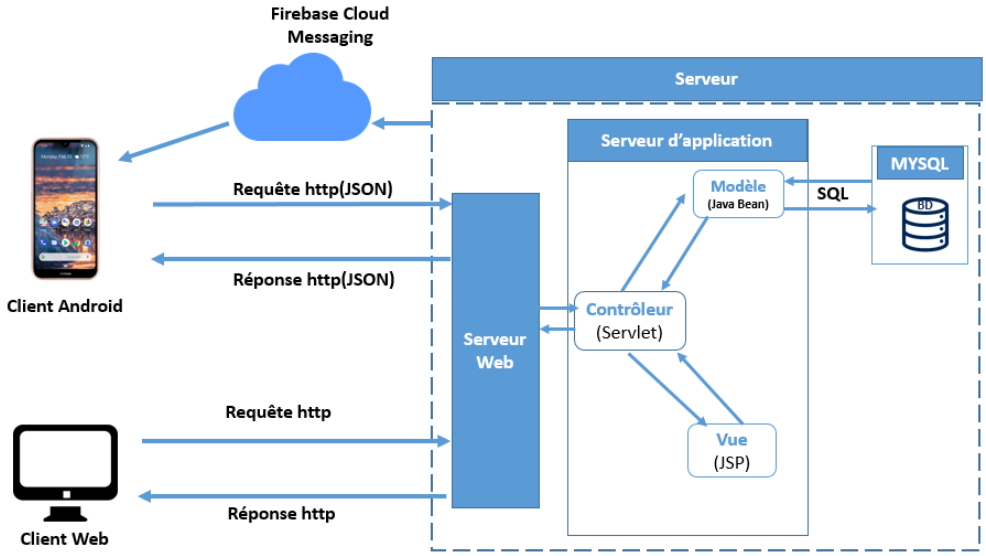


FIGURE 4.1 – Architecture global de notre application

1. **Client :** le client envoie une requête au serveur, le serveur reçoit la requête sur le contrôleur, ce dernier envoie les données au modèle pour le traitement, après le traitement le contrôleur renvoie la réponse au client, cette communication se fait à travers le protocole HTTP.
   Client android : une application mobile développée en java sous android studio. L'échange de données entre l'application et le serveur est en format léger d'échange de données JSON.
   Client web : application web développée en java.
2. **Serveur :** nous avons utilisé Apache Tomcat 9.0.
3. **Gestion des notifications :** nous avons utilisé FCM pour l'envoie de notification du serveur vers le client Android.
4. **Base de données :** la base de données (présentée dans la figure 4.2) est réalisée avec php-MyAdmin, nous proposons une architecture avec 17 tables qui répondent, selon nous, à tous les besoins de ce travail.

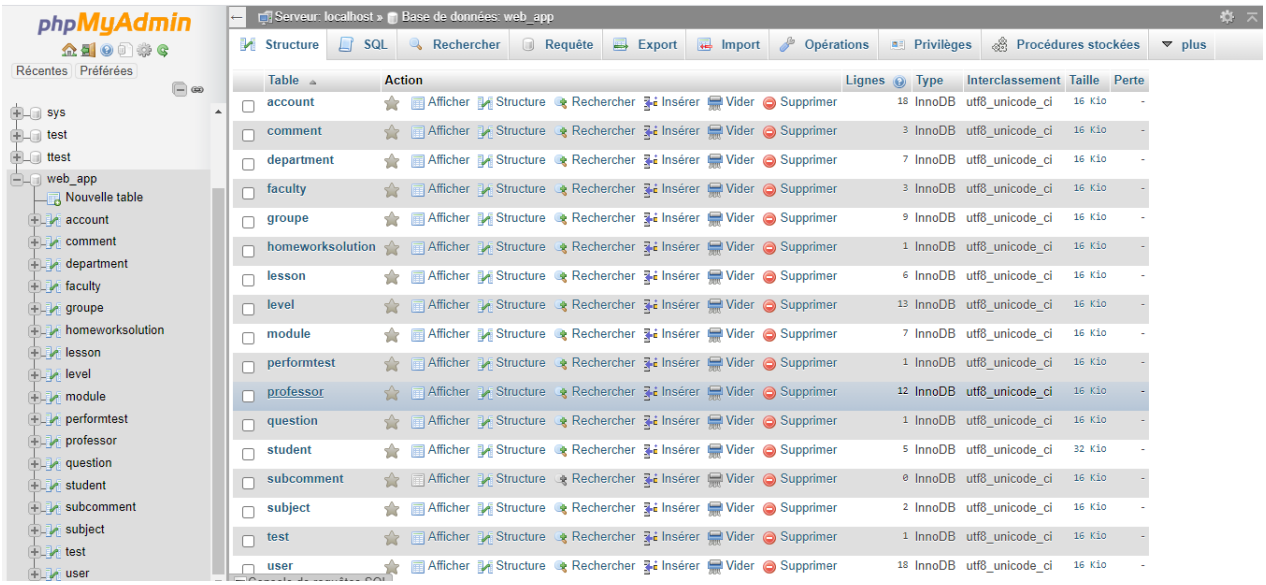


FIGURE 4.2 – Tables de la base de données

Nous présentons ci-dessous quelque exemples des tables de la base de données sous forme détaillée.

Compte : cette table nous permet de savoir si l'utilisateur a le droit d'accéder à l'application ou non, elle contient un identifiant unique, le nom, le mot de passe, le type et l'identifiant d'utilisateur.

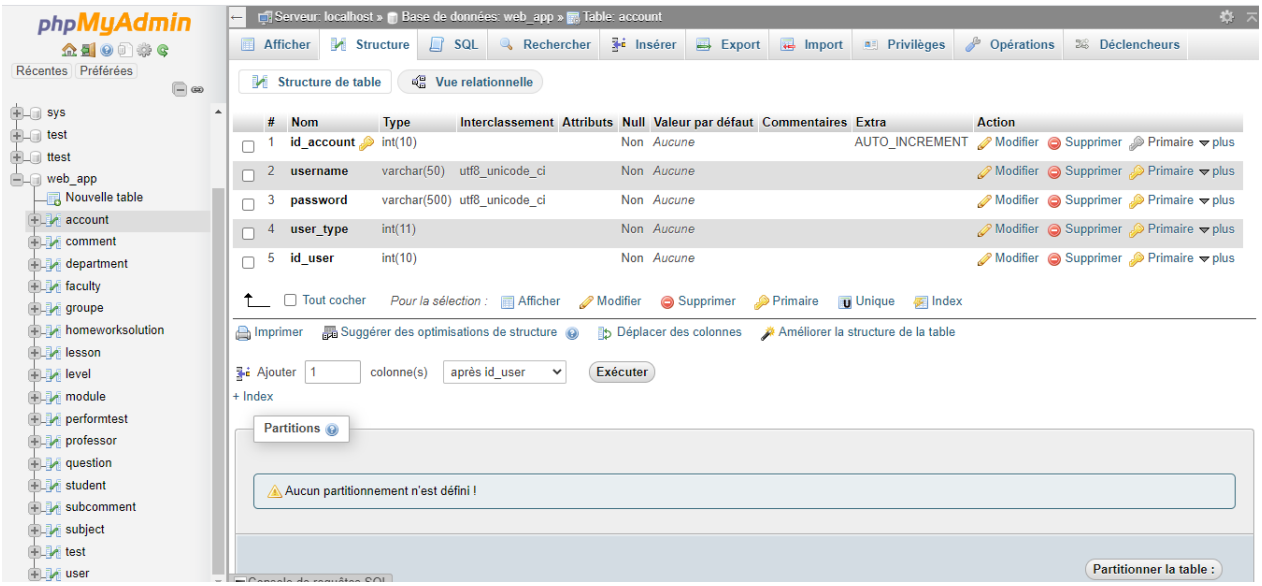


FIGURE 4.3 – Table compte

Utilisateur : cette table contient les information de l'utilisateur tel que le nom, le prénom, date de naissance,numéro de téléphone, etc.

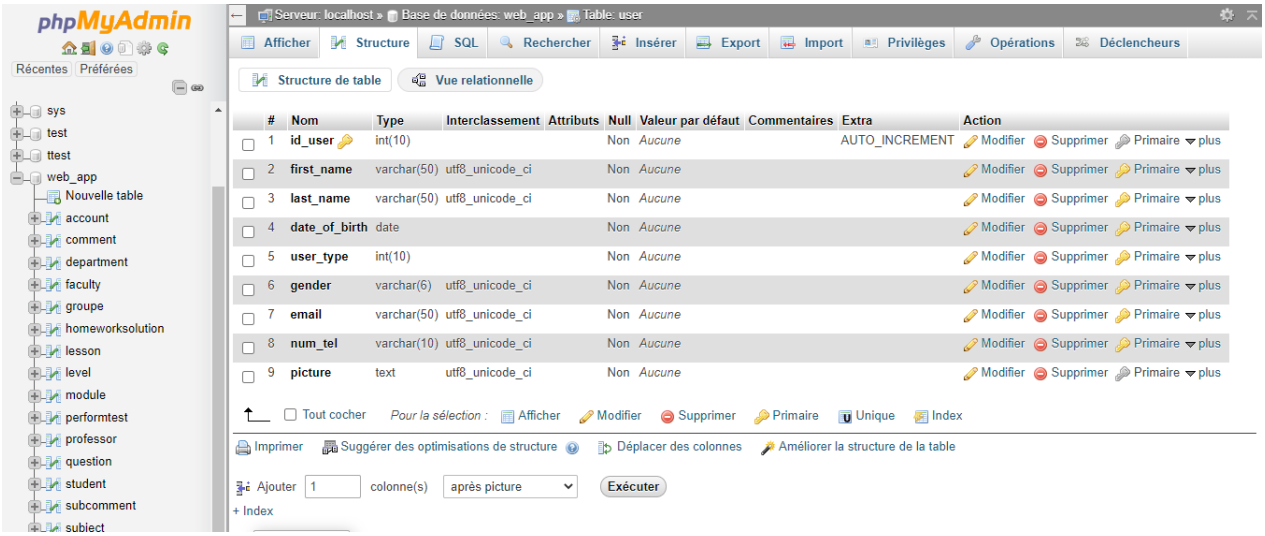


FIGURE 4.4 – Table utilisateur

Cours : chaque cours possède un identifiant unique (clé primaire), le titre, la description, le lien, le temps et le type etc.

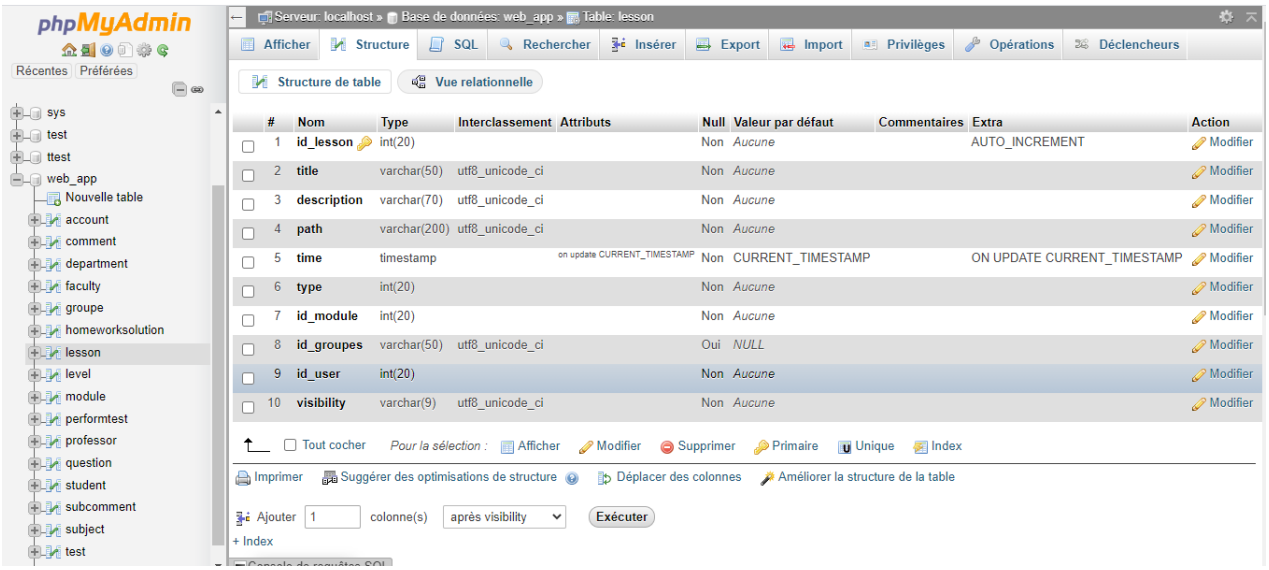


FIGURE 4.5 – Table cours

4.4 Difficultés rencontrées

Au cours de la réalisation de ce projet, nous avons rencontré de nombreuses difficultés, dont les suivantes :

1. **Le choix d'émulateur :** pour l'exécution de notre application nous avons utilisé l'appareil Virtuel Android, mais ça ne nous a jamais aidé parce que c'était très lourd et prend beaucoup

de temps pour l'exécution, et après nous avons essayé d'utiliser un autre émulateur qui était Memu mais malheureusement nous avons rencontré le même problème, donc nous avons utilisé le smart phone pour éviter ce problème.

2. **Les outils utilisés :** nous avons utilisé de nouveaux outils que nous n'avons pas utilisés auparavant. Au début, il y avait une certaine ambiguïté sur la façon dont cela fonctionnerait, mais avec le temps, nous nous y sommes habitués.

4.5 Conclusion

La phase de réalisation c'est la dernière étape dans le cycle de vie d'une application. Dans ce dernier chapitre, nous avons présenté les plateformes, les outils et les langages de programmation que nous avons utilisé pour réaliser notre application et nous avons représenté l'architecture global de l'application.

Conclusion générale

L'e-learning ou bien l'enseignement à distance est une technologie basée sur un enseignement formalisé mais à l'aide de ressource électronique dans le but d'améliorer la qualité d'apprentissage.
Notre travail consiste à la réalisation d'une application mobile pour l'enseignement à distance au sein de l'université, donc nous avons réalisé une application mobile pour l'université « mohamed seddik benyahia » de Jijel.

Cette application contient deux parties l'une mobile et l'autre web, elle permet principalement aux enseignants de mettre en ligne des contenus pédagogiques et d'interagir avec leurs étudiants. Ces derniers aussi peuvent télécharger des contenus publiés par leurs enseignants et communiquer avec eux et avec ses collègues à travers les commentaires ou le forum.

Le travail que nous avons réalisé nous a permis d'améliorer et d'enrichir nos compétences dans le développement mobile et web, nous avons donc acquis de l'expérience avec un ensemble de langages, méthodes et technologies comme Java EE, Java, JavaScript, Jquery, Ajax, CSS, HTML, TailwindCss, Adobe XD, etc.

Comme perspectives, nous avons l'intention de terminer l'implémentation de certaines fonctionnalités telles que la messagerie.

Bibliographie

[1] R. C. Clark and R. E. Mayer, *E-learning and the science of instruction : Proven guidelines for consumers and designers of multimedia learning*. John Wiley & Sons, 2016.

[2] "Plan d'action elearning." https ://eur-lex.europa.eu/.

[3] M. Abdellatief, A. B. M. Sultan, M. A. Jabar, R. Abdullah, *et al.*, "A technique for quality evaluation of e-learning from developers perspective," *American Journal of Economics and Business Administration*, vol. 3, no. 1, pp. 157–164, 2011.

[4] "À propos de moodle." https ://docs.moodle.org, consulté le 27/02/2020.

[5] Han, Wen, "A fundamentals of financial accounting course multimedia teaching system based on dokeos and bigbluebutton," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 13, no. 05, pp. 141–152, 2018.

[6] N. C. Benabdellah, *Pour une adaptation du contenu pédagogique en ligne au profil de l'apprenant*. PhD thesis, Université Mohammed 5 de Rabat, 27 Novembre 2015.

[7] M. Qiu, W. Dai, and K. Gai, *Mobile Applications Development with Android : Technologies and Algorithms*. Chapman and Hall/CRC, 2016.

[8] Park, Yeonjeong, "A pedagogical framework for mobile learning : Categorizing educational applications of mobile technologies into four types," *The international review of research in open and distributed learning*, vol. 12, no. 2, pp. 78–102, 2011.

[9] F. Nayebi, J.-M. Desharnais, and A. Abran, "The state of the art of mobile application usability evaluation," in *2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–4, IEEE, 2012.

[10] D. M. Mahmud and N. A. S. Abdullah, "Mobile application development feasibility studies : A case study in universiti teknologi mara," in *2014 IEEE Conference on Open Systems (ICOS)*, pp. 30–35, IEEE, 2014.

[11] Lee, Wei-Ming, *Beginning android 4 application Development*. John Wiley & Sons, 2012.

[12] "smartphone-windows-phone." https ://www.futura-sciences.com/tech/definitions/smartphone-windows-phone-15584/.

[13] Mahdi H. Miraz, "Development platforms : Android, ios, windows phone, blackberry and symbian," *Computer Barta*, vol. 17, pp. 110–112, 01 2014.

[14] "Blackberry os - definition." https ://www.gsmarena.com/glossary.php3 ?term=bb-os.

[15] "Répartition des expéditions de smartphones dans le monde par système d'exploitation entre 2013 et 2022." https ://fr.statista.com.

[16] R. R. Leon Shklar, *Web Application Architecture Principles, protocols and practices*. John Wiley & Sons Ltd, 2003.

[17] S. K. S. Subash Chandra Yadav, *an introduction to client/server computing*. New Age International, 2009.

[18] Stephen Thomas, *HTTP Essentials*. Wiley, 2001.

[19] H. S. Oluwatosin, "Client-server model," *IOSRJ Comput. Eng*, vol. 16, no. 1, pp. 2278–8727, 2014.

[20] M. N. Lien, *Réalisez votre site web avec HTML5 et CSS3*. EYROLLES, 20 décembre 2011.

[21] Marijn Haverbeke, *Eloquent JavaScript*. No Starch Press, 2011.

[22] L. T. Luke Welling, *PHP et MySQL*. Pearson Education France, 2009.

[23] Jérôme Lafosse, *Java EE guide de dévloppement d'applications web on java*. ENI, 9 février 2009.

[24] Pascal Roques, *UML2-Modéliser une application Web*. EYROLLES, 2 octobre 2008.

[25] J. T. Messager Rota V, *Gestion de projet vers les méthodes agiles*. EYROLLES, 7 mai 2009.

[26] O. G. e. D. F. Jérôme LARD, Frédéric LANDRAGIN, "Un cadre de conception pour réunir les modèles d'interaction et l'ingénierie des interfaces," *arXiv preprint arXiv :0807.2628*, 2008.

[27] "Mvc : Module vue controler." https ://openclassrooms.com.

[28] D. G. Joseph Gabay, *UML 2,Mise en œuvre guidée avec études de cas analyse et conception*. Dunod, 16 avril 2008.

[29] Cyrille Herby, *Apprenez à programmer en Java*. Eyrolles, 2018.

[30] Pierre Giraud, *Apprenez-a-coder-en-html5-et-en-css3*.

[31] Arun Gupta, *Java EE 7 Essentials*. O'Reilly Media, 2013.

[32] Antonio Goncalves, *Java EE 5*. EYROLLES, 2011.

[33] "Introducing json." https ://www.json.org/.

[34] Apache Software Foundation, "Apache tomcat." http ://tomcat.apache.org/.

[35] Eclipse foundation, "The eclipse ide'." https ://www.eclipse.org/.

[36] Google Developers, "Android studio." https ://developer.android.com.

[37] "Appserv." https ://www.appserv.org.

[38] Sparx systems, "Entreprise Architect." https ://sparxsystems.com/.

[39] "A utility-first css framework for rapidly building custom designs." https ://tailwindcss.com/.

[40] Adobe, "Adobe xd." https ://www.adobe.com/.

[41] Google Developers, "Firebase Cloud Messaging." https ://firebase.google.com/.