République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohammed Seddik Ben Yahia de Jijel
Faculté des Sciences Exactes et Informatique
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de master en informatique

Option : ***Réseaux et sécurité***

Thème

Application mobile pour l'enseignement à distance au sein de l'université de Jijel

Présenté par :
CHELLOUCHE Bachira
MECHETER Rahima

Soutenu devant le jury composé de :

Mr BENKINIOUAR Mouad	**Président**
Mme BRIGHEN Assia	**Examinatrice**
Mr BOUBAKIR Mohammed	**Encadreur**

Soutenance à huis clos
Promotion : 2020

Résumé

Les systèmes d'e-learning visent à offrir un accès facile et permanent aux ressources pédagogiques mises en ligne. L'objectif de ce mémoire est de développer une application mobile visant à faciliter l'enseignement au sein de l'université. L'application développée permet aux enseignants de rendre disponible en ligne des contenus pédagogiques aux étudiants ainsi qu'offrir un moyen d'interaction entre les enseignants et les étudiants à travers les commentaires et le forum. L'application comporte deux parties, l'une mobile offrant les fonctionnalités principales d'e-learning et l'autre web offrant les mêmes fonctionnalités de la partie mobile et aussi permettant d'effectuer les tâches administratives. Pour la développer, nous avons suivi une approche nommée processus simplifié en utilisant le langage UML, et pour sa réalisation, nous avons employé un ensemble de plateformes et d'outils tels que JEE, Android Studio, Eclipse, etc.

Mots clés : E-learning, Application mobile, Application web, Processus simplifié, UML.

Abstract

E-learning systems aim to offer an easy and permanent access to pedagogic resources posted online.The objective of this thesis is to develop a mobile application that aims to facilitate teaching inside the university. The developed application allows teachers to provide pedagogic contents online for students as well as a mean of interaction between teachers and students throughout comments and forums. The app consists of two parts,one mobile that offers main e-learning features and the other web offering the same functionalities of the mobile part and also allowing to perform administrative features. To develop it, we followed an approach called simplified process using the UML language, and for its realization, we used a set of platforms and tools such as JEE, Android Studio, Eclipse, etc.

Key words : E-learning, Mobile application, Web application, Simplified process, UML.

⁕ *Remerciements* ⁕

Avant toute personne, nous tenons à remercier ALLAH le tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce modeste travail.

Nous tenons à exprimer notre sincère gratitude à notre encadreur Mr.BOUBAKIR Mohamed pour ses conseils et ses encouragements tout au long de ce projet.

Nous tenons à exprimer toute notre gratitude à tous les membres de jury, pour avoir bien voulu juger notre travail.

Nous voulons aussi adresser nos sincères remerciements à tous les enseignants de département de l'informatique qui ont contribué à notre formation.

Aussi nos parents qui nous ont toujours soutenus et encouragés au cours de ce mémoire.

Par crainte d'avoir oublié quelqu'un, que tous ceux et toutes celles dont nous sommes redevables se voie ici vivement remerciés.

⁂ Ddicaces ⁂

Je dédie ce modeste travail à :

⁂ Mes chers parents, que nulle dédicace ne puisse exprimer mes sincères sentiments, ourleur aide, leur encouragement et leur patience illimitée ;

⁂ A mes chers frères et sœurs Hanane, Warda, Badreddine, Soufiane,Yasser, Youssra ;

⁂ Mon binôme Rahima Mecheter avec qui j'ai réalisé ce travail ;

⁂ Toute la promotion 2020 spécialement "Réseaux et sécurité" ;

⁂ Et une dédicace très spéciale a Mes amis de groupe « DZ Community » et surtout Housseyn Belhadja ;

Bachira

⁕ Ddicaces ⁕

Je dédie ce modeste travail à :

⁕ ***Mes chers parents qui ont contribué à ma réussite et m'ont encouragé ;***

⁕ ***Ma seule sœur Leila et tous mes frères ;***

⁕ ***Mon binôme Bachira Chellouche avec qui j'ai réalisé ce travail ;***

⁕ ***Toute la promotion 2020 spécialement "Réseaux et sécurité" ;***

⁕ ***Tous mes amis qui me connaissent de près ou de loin ;***

Rahima

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

OS Operating System
IOS iPhone Operating System
UP Unified process
PS Processus simplifié
XP eXtreme Programming
UML Unified Modeling Language
MVC Modèle-vue-contrôleur
DSS Diagramme de Séquence Système
FT Fiches types
UC Use Case
IHM Interface Homme-Machine
CD-ROM Compact Disc - Read Only Memory
HTTP Hypertext Transfer Protocol
API Application Programming Interface
HTML Hyper Text Markup Language
CSS Cascading Style Sheets
JS JavaScript
PHP Hypertext Preprocessor
JSP Java Server Page
MYSQL My Structured Query Language
JEE Java Entreprise Edition
AJAX Asynchronous JavaScript And XML
Adobe XD Adobe Experience Design
FCM Firebase Cloud Messaging

Introduction Générale

L'enseignement est un mode d'éducation permettant de transmettre des connaissances à un élève, un étudiant ou tout autre public dans le cadre d'un établissement d'enseignement. Au début, l'enseignement se limitait aux méthodes traditionnelles, où la présence de l'étudiant et du l'enseignant était une condition nécessaire à l'apprentissage, mais avec l'arrivée d'internet et l'évolution de la technologie, un nouveau mode d'enseignement est apparu appelé e-learning.

L'e-learning ou l'enseignement à distance est une technique qui utilise les supports électroniques, les technologies éducatives ainsi que les technologies d'information et de communication pour faciliter l'apprentissage à distance. Cette technique permet aux étudiants de se former en ligne, quand ils souhaitent, et d'accéder à divers contenus pédagogiques à tout moment, quel que soit leur emplacement. Depuis un certain temps, l'utilisation des supports CD-ROM a été réduite en e-learning, ouvrant le champ au web et à diverses applications. C'est une solution qui permet d'avoir accès aux formations plus facilement avec seulement une connexion internet. Cet accès permet d'acquérir les connaissances et de développer les compétences des étudiants.

Actuellement, le monde connaît une avance technologique considérable dans tous les secteurs, il est donc devenu nécessaire de suivre le rythme de ces évolutions dans divers domaines, y compris le domaine de l'éducation, qui à son tour a connu un grand changement avec l'apparition de nombreuses plates-formes éducatives, qui ont été largement utilisées, en particulier par les grandes universités, et avec l'émergence de Covid-19 l'utilisation de ces plateformes est devenue nécessaire.

Dans le cadre du projet de fin d'études, nous proposons de développer une application d'e-learning baptisée *UniverPack*, qui a pour objectif de faciliter l'enseignement à distance au sein de l'université de Jijel et d'offrir un moyen d'interaction et de communication entre les enseignants et les étudiants à travers les commentaires ou le forum, l'application comporte deux parties, l'une mobile offrant les fonctionnalités principales d'e-learning et l'autre web offrant les mêmes fonctionnalités de la partie mobile et aussi permettant d'effectuer les tâches administratives. Nous avons développé notre application suivant une approche nommée processus simplifié en utilisant le langage UML, et pour sa réalisation, nous avons employé un ensemble de plateformes et d'outils tels que JEE, Android Studio, Eclipse, etc.

Organisation du mémoire

Le mémoire est organisé en quatre chapitres :

- Dans le premier chapitre, nous présentons les différentes notions de base qui concernent notre travail ainsi que le processus de développement.
- Le deuxième chapitre, présente la partie spécification des besoins de notre application. Ces besoins sont spécifiés à l'aide des différents diagrammes (diagramme de cas d'utilisation, diagramme de séquences) et les maquettes ainsi que la description textuelle.
- Le troisième chapitre est consacré à la présentation de l'analyse et la conception de cette application en utilisant des diagrammes de classes participantes, et des diagrammes de navigations.
- Le quatrième chapitre est réservé pour la présentation des outils de développement utilisés ainsi que l'architecture de notre application.
- Enfin nous clôturons ce mémoire par une conclusion générale et perspective.

Chapitre 1

Généralités

1.1 Introduction

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. Donc dans ce chapitre, nous présentons l'e-learning, ses différents types et ses objectifs ainsi que quelques exemples de plateformes d'apprentissages en ligne. Ensuite, nous présentons les applications mobiles et les applications web. Enfin, nous présentons le langage de modélisation UML, le processus de développement suivi ainsi que le modèle MVC.

1.2 E-Learning

1.2.1 Définition

E-learning est une méthode d'apprentissage qui fournit des contenus pédagogiques via un support électronique (intranet, CD-ROM, ou internet) dans le but de faciliter l'apprentissage à distance. Il comprend aussi bien des outils et des applications pédagogiques que des contenus pédagogiques [1].

Différentes définitions sont proposées pour l'e-Learning parmi elles, nous citons : Celle proposée par la Commission de Bruxelles est : "E-Learning est l'utilisation des nouvelles technologies multimédias de l'internet pour améliorer la qualité de l'apprentissage en facilitant d'une part l'accès à des ressources et à des services, d'autre part les échanges et la collaboration à distance" [2].

1.2.2 Types d'E-learning

L'apprentissage en ligne prend plusieurs formes et souvent une combinaison des éléments suivants [3] :

1. **Asynchrone :** une méthode de formation qui ne nécessite pas la présence des gens en ligne en même temps, ce qui permet aux gens de participer à leur horaire.

2. **Synchrone :** apprendre où les gens sont en ligne en même temps comme une classe virtuelle et une conférence audio et vidéo.
3. **Mixte :** c'est une méthode qui combine les deux méthodes précédentes synchrone et asynchrone.

1.2.3 Objectifs d'E-learning

L'e-learning a de nombreux objectifs, notamment :

- Aider les apprenants à acquérir les compétences dont ils ont besoins.
- Offrir la possibilité de varier son apprentissage par l'utilisation des éléments médiatiques tels que des mots et des images pour fournir le contenu et les méthodes.
- Développer de nouvelles connaissances et compétences liées aux objectifs d'apprentissage individuels.

1.2.4 Exemples des plateformes E-Learning

Il existe de nombreuses plateformes d'apprentissage en ligne sur le marché, nous citons certains des exemples les plus connus :

1. **Moodle :** l'origine du terme Moodle était l'acronyme "Modular Object Oriented Dynamic Learning Environnement", c'est une plateforme d'apprentissage destinée à fournir aux enseignants, administrateurs et apprenants un système unique, robuste, sûr et intégré pour créer des environnements d'apprentissage personnalisés. Moodle est élaboré par le projet Moodle qui est conduit et coordonné par Moodle HQ, une entreprise australienne de 30 développeurs, soutenue financièrement par un réseau d'environ 60 entreprises de service du monde entier, les partenaires Moodle [4].
2. **Dokeos :** est un système d'enseignement et de gestion de cours en réseau, open source développé par le langage PHP pour permettre aux instructeurs de gérer et de créer des sites web de cours dans les navigateurs, prenant en charge 34 interfaces de langue grand public du monde. Via l'interface de gestion, vous pouvez créer de nouveaux cours et y ajouter des textes, des documents, des liens, un agenda, une description du parcours pédagogique, etc. Le logiciel est librement téléchargeable et peut être installé sur tout type de serveur [5].
3. **Claroline :** est une plate-forme open source, distribuée sous licence GPL, qui permet à des centaines d'institutions issues de plus de 80 pays de créer gratuitement des espaces de cours en ligne. Pour chaque cours, le formateur dispose d'une série d'outils lui permettant de rédiger une description du cours, publier des documents dans tous les formats (texte, PDF, HTML, vidéo, etc), d'administrer des forums de discussion publics ou privés, élaborer des parcours pédagogiques, créer des groupes de participants, de composer des exercices, structurer un agenda avec des tâches et des échéances, publier des annonces par e-mail, proposer des travaux à rendre en ligne, consulter les statistiques de fréquentation et de réussite aux exercices et d'utiliser le wiki pour rédiger des documents collaboratifs. Adaptable à différents contextes de formation, Claroline est utilisée non seulement dans les écoles et les universités, mais également dans les centres de formation, les associations et les entreprises [6].

1.3 Applications mobiles

1.3.1 Définition

Une application mobile est un programme conçu pour fonctionner sur smartphone, tablette et d'autres appareils mobiles, elle est téléchargeable de façon gratuite ou payante depuis des boutiques d'applications. Les boutiques les plus populaires pour les applications sont *App store*, *Google Play* et *Windows Store* [7].

1.3.2 Avantages des applications mobiles

Les applications mobiles détient de nombreux atouts telles que :

- Les applications de chat et de partage des données en temps réel prennent en charge la communication, la collaboration et la construction de connaissances. Cela permet aux étudiants de consommer et de créer des informations collectivement et individuellement [8].
- Accès direct aux contenus de l'application mobile via l'icône présente sur le dashboard du téléphone ou du tablette.
- Un fonctionnement en mode déconnecté pour certaines applications.

1.3.3 Inconvénients des applications mobiles

L'architecture de ces applications doit prendre en compte un certain nombre de contraintes de conception, telles que les ressources limitées, les problèmes de connectivité, les modèles de saisie des données et les différentes résolutions d'affichage des appareils mobiles [9].

1.3.4 Différents types d'applications mobiles

Il existe trois types d'applications mobiles [10] :

1. **Les applications natives ou embarquées :** une application native est une application mobile qui a été développée pour être utilisée sur une plate-forme ou un appareil particulier. Elle est installée sur l'appareil et répond plus rapidement qu'une application web parce que l'interface est plus directe. Une application native est téléchargée depuis un magasin d'application et installée sur l'appareil.
2. **Les applications web :** c'est un programme d'application qui est stocké sur un serveur distant comme une application web normale. L'application web est accessible et exécutable sur tous les smartphones via leur navigateur web.
3. **Les applications hybrides :** les applications hybrides sont des applications qui combinent les éléments d'une application web et les éléments d'une application native. Elle doit être installée dans les appareils.

1.3.5 Systèmes d'exploitation mobiles les plus populaires

Un système d'exploitation mobile, généralement connu sous le nom d'OS mobile est une plate-forme logicielle qui contrôle toutes les fonctionnalités des appareils mobiles.
Nous représentons ci-dessous les systèmes d'exploitation mobiles les plus connus :

- **Android :** est un système d'exploitation mobile qui est basé sur une version modifiée de Linux. Il a été initialement développé par une startup du même nom, Android, Inc. En 2005, dans le cadre de sa stratégie pour entrer dans l'espace mobile, Google a acheté Android et a repris son travail de développement (ainsi que son équipe de développement). Android est un OS gratuit et complètement ouvert, ce qui signifie que le code source et les API sont ouverts. Ainsi les fabricants de matériel peuvent ajouter leurs propres extensions propriétaires à Android et de le personnaliser pour différencier leurs produits des autres [11].
- **Windows Phone :** est le nom du système d'exploitation pour smartphones que Microsoft a publié en octobre 2010, comme remplacent du Windows Mobile en introduisant une interface utilisateur totalement redessinée et pensée pour les terminaux à écran tactile. De 2010 à 2015, Windows Phone a connu plusieurs mises à jour majeures avec notamment le passage du noyau Windows CE (Windows Phone 7) au noyau de Windows NT (Windows Phone 8) et l'introduction de l'assistant vocal Cortana (Windows Phone 8.1). À partir de novembre 2015, Windows Phone disparaît progressivement et il est remplacé par Windows 10 Mobile [12].
- **iOS :** (anciennement appelé «iPhone OS») est un système d'exploitation mobile développé par Apple Inc. Pour divers appareils Apple tels que l'iPhone, l'iPod Touch, l'iPad et même pour l'Apple TV de deuxième génération. Contrairement à Android de Google et au Windows Phone de Microsoft, l'utilisation d'iOS est limitée aux appareils Apple uniquement [13] .
- **BlackBerry OS :** est un système d'exploitation mobile propriétaire développé par la société canadienne Research In Motion (RIM) pour sa gamme de smartphones BlackBerry [14].

1.3.6 Part du marché

Dans cette section nous allons présenter une statistique publiée par Statista [15] sur les principaux systèmes d'exploitation mobile, et les résultats d'une questionnaire que nous avons envoyé aux étudiants universitaires algériens sur les systèmes d'exploitation mobiles les plus utilisés par eux, les résultats sont présentés dans les figures 1.1 et 1.2 respectivement.

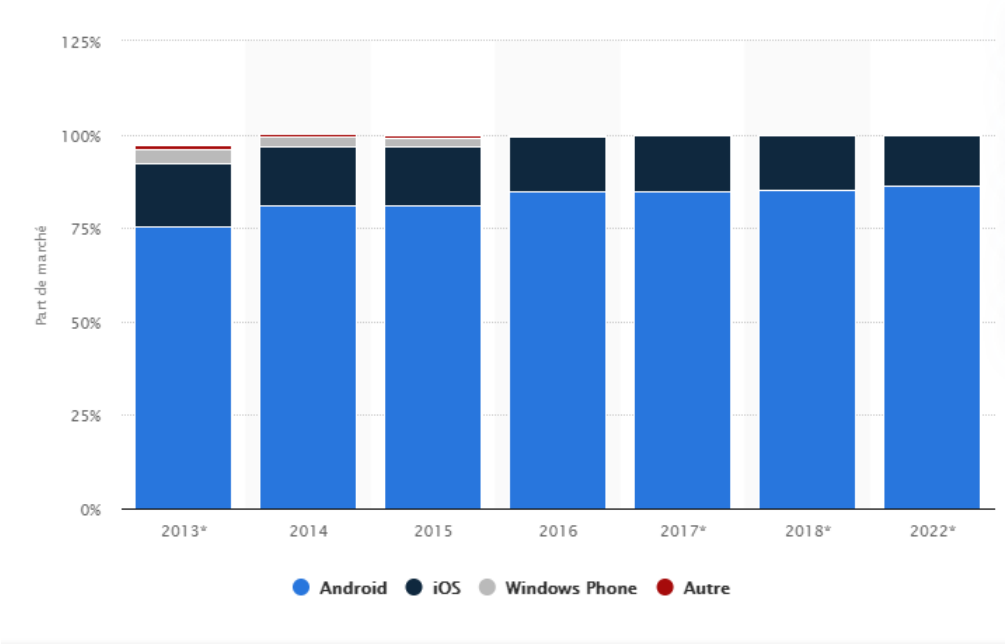


FIGURE 1.1 – Répartition des expéditions de smartphones dans le monde par système d'exploitation entre 2013 et 2022

[15]

FIGURE 1.2 – Répartition d'utilisation des OS par les étudiants des universités algériennes

La figure 1.1 présente la part de marché détenue par les systèmes d'exploitation pour smartphones en expéditions d'unités entre 2013 et 2022, on remarque qu'Android occupe une grande

place sur le marché mondial. En 2022, le système d'exploitation Android devrait représenter environ 86% des expéditions mondiales de smartphones, il est devenu leader du marché.

La figure 1.2 montre clairement que parmi 272 étudiants 91.9% utilisent le système d'exploitation Android. Pour cette raison, nous avons choisi Android comme OS pour notre application.

1.4 Applications web

1.4.1 Définition

Une application web est une application client-serveur qui utilise, généralement, le navigateur web comme client. Les navigateurs envoient des requêtes aux serveurs, et ces derniers génèrent des réponses et les renvoient aux navigateurs. Ils utilisent un programme client commun, à savoir le navigateur web [16].

1.4.2 Avantages d'application web

L'utilisation des navigateurs web en tant que clients présente des avantages importants :

- Aucune installation pour les utilisateurs.
- Un seul code permet d'être visible sur toutes les plateformes.
- Une application web fonctionne sur n'importe quel système d'exploitation. Elle s'adapte facilement à iOS, Android, Windows Phone ou autres.
- Accès depuis n'importe quel type d'appareil : pc, téléphone mobile, tablette, etc.

1.4.3 Architecture client/serveur

L'architecture client/serveur est une nouvelle technologie qui apporte des solutions à de nombreux problèmes de gestion des données auxquels sont confrontées les organisations modernes. Le terme client/serveur est utilisé pour décrire un modèle informatique pour le développement des systèmes informatisés. Ce modèle est basé sur la répartition des fonctions entre deux types de processus indépendants et autonomes : serveur et client, ce dernier est tout processus qui demande des services spécifiques au processus serveur, alors qu'un serveur est un processus qui fournit les services demandés au client. Les processus client et serveur peuvent résider sur le même ordinateur ou sur différents ordinateurs reliés par un réseau [17].
L'architecture client/serveur est présentée dans la figure 1.3.

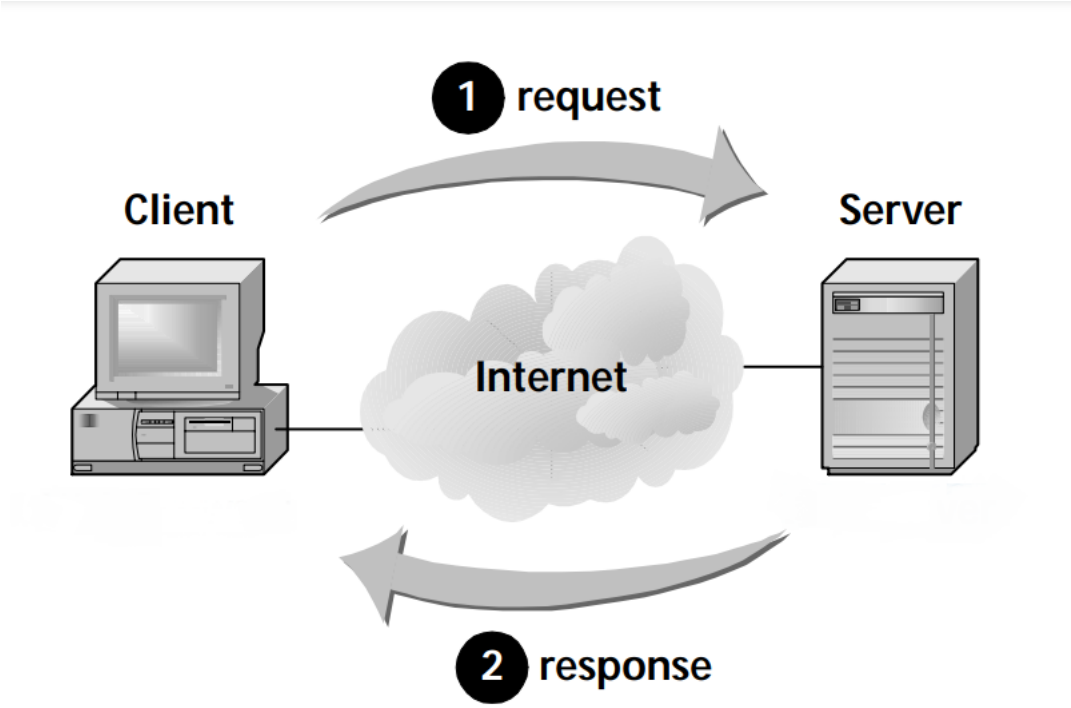


FIGURE 1.3 – Architecture client/serveur
[18]

1.4.3.1 Avantages du modèle client-serveur

Le modèle client-serveur possède plusieurs avantages dont nous pouvons citer [19] :

- Il répartit le traitement des demandes entre plusieurs machines.
- Il permet un partage plus facile des ressources du client vers les serveurs.
- Il réduit la réplication des données en les stockant sur chaque serveur au lieu du client.

1.4.4 Langages, protocoles et technologies web

Dans cette partie nous citons et définissons les différents protocoles, langages et technologies qui permet d'implémenter l'architecture client/serveur dans le web :

- **HTTP (HyperText Transfer Protocol) :** est un protocole au niveau de l'application, qui définit la communication entre un client (exemple : navigateur) et un serveur sur le World Wide Web, il est fonctionné sur le principe "requête-réponse" [16].
- **URL (Uniform Resource Locator) :** est un format de nommage universel pour désigner d'une ressource donnée, unique sur le web. Son rôle est de localiser une page dans l'océan des milliards des pages internet existantes.
- **CSS (Cascading Style Sheets) :** aussi appelées feuilles de styles, le rôle du CSS est de gérer l'apparence de la page web (agencement, positionnement, décoration, couleurs, taille du texte, etc)[20].

- **JavaScript :** JavaScript a été introduit en 1995 comme un moyen d'ajout des programmes aux pages web dans le navigateur Netscape. La langue a depuis été adoptée par tous les autres navigateurs web graphiques. Il a rendu possible les applications web modernes, c'est-à-dire les applications avec lesquelles vous pouvez interagir directement sans recharger une page pour chaque action. JavaScript est également utilisé dans des sites plus traditionnels pour fournir des formes d'interactivité et d'intelligence [21].
- **PHP :** est un langage de script côté serveur qui a été conçu spécifiquement pour le web. Le code PHP est inclus dans une page HTML et sera exécuté à chaque fois qu'un visiteur affichera la page, il est interprété au niveau du serveur web et génère du code HTML ou toute autre donnée affichable dans le navigateur de l'utilisateur [22].
- **JEE (Java Enterprise Edition) :** une plate-forme logicielle d'application d'Oracle basée sur le langage de programmation Java. Les services Java EE sont exécutés au niveau intermédiaire entre la machine de l'utilisateur et les bases de données et systèmes d'informations hérités de l'entreprise. Son composant principal est Enterprise JavaBeans (EJBs), suivi par JavaServer Pages (JSPs) et Java servlets et une variété d'interfaces pour la liaison aux ressources d'informations de l'entreprise [23].
- **JSP :** c'est l'acronyme de Java Server Page. C'est une technologie java qui permet la génération des pages web dynamiques, cette technique permet au code Java et à certaines actions prédéfinies d'être ajoutés dans un contenu statique [23].
- **Servlet :** une servlet est une classe de langage de programmation Java qui est utilisée pour étendre les capacités des serveurs qui hébergent des applications accessibles au moyen d'un modèle de programmation demande-réponse. Les servlets effectuent des traitements côté serveur en réponse aux requêtes des clients distants [23].

1.5 UML

1.5.1 Définition

UML se définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue [24].

1.5.2 Types de diagrammes UML

UML propose treize types de diagrammes pour modéliser un système. Ils se répartissent en deux grands groupes [24] :

1. **Diagrammes structurels ou diagrammes statiques (UML structure)**
   - **Diagramme de classes (Class diagram) :** un diagramme de classes est une collection d'éléments de modélisation statique qui montre la structure d'un modèle (classes, associations, interfaces, attributs, opérations, etc).
   - **Diagramme d'objets (Object diagram) :** permet de modéliser les instances des éléments structurels et leurs liens à l'exécution.

- **Diagramme de paquetages (Package diagram) :** représente les paquetages composant un système, ainsi que les relations entre packages.
- **Diagramme de structure composite :** il montre l'organisation interne d'un élément statique complexe.
- **Diagramme de composants (Component diagram) :** ces diagrammes décrivent les composants et leurs dépendances dans l'environnement de réalisation.
- **Diagramme de déploiement (Deployment diagram) :** les diagrammes de déploiement montrent la disposition physique des différents matériels.

2. **Diagrammes comportementaux ou diagrammes dynamiques (UML behavior)**
   - **Diagramme de cas d'utilisation (Use case diagram) :** les use cases décrivent les besoins des utilisateurs et les objectifs correspondants d'un système.
   - **Diagramme de séquence (Sequence diagram) :** dont le but est de décrire comment les éléments du système interagissent entre eux et avec les acteurs.
   - **Diagramme de communication (Communication diagram) :** anciennement appelé un diagramme de collaboration est un schéma d'interaction. Son objectif principal est de décrire les interactions entre les différents objets du système.
   - **Diagramme d'activités (Activity diagram) :** il montre l'enchaînement des actions et décisions au sein d'une activité.
   - **Diagramme de vue d'ensemble des interactions :** il fusionne les diagrammes d'activité et de séquence pour combiner des fragments d'interaction avec des décisions et des flots.
   - **Diagramme d'états-transitions (State machine diagram) :** il montre les différents états et transitions possibles des objets d'une classe.
   - **Diagramme de temps :** il fusionne les diagrammes d'états et de séquence pour montrer l'évolution de l'état d'un objet au cours du temps.

1.6 Processus de développement

Un processus définit une séquence d'étapes, partiellement ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. L'objet d'un processus de développement est de produire des logiciels de qualité qui répondent aux besoins de leurs utilisateurs dans des coûts prévisibles [24].

Généralement, il existe plusieurs processus de développement d'applications, nous avons opté pour un processus simplifié qui se situe entre UP (Unified Process) et les méthodes agiles, telles que XP (eXtreme Programming), et Scrum.

1.6.1 Processus unifié UP (Unified Process)

Le processus unifié fournit un cadre au développement logiciel pour la construction des systèmes orientés objet qui doit être associé à UML. Il est itératif et incrémental, centré sur l'architecture, conduit par les cas d'utilisation et piloté par les risques [24].

- **conduit par les cas d'utilisation :** le projet est mené en tenant compte des besoins et des exigences des utilisateurs. Les cas d'utilisation du futur système sont identifiés, décrits avec précision et priorisés.
- **itératif et incrémental :** le projet est découpé en itérations de courte durée qui aident à mieux suivre l'avancement global. À la fin de chaque itération, une partie exécutable du système final est produite, de façon incrémentale.
- **centré sur l'architecture :** tout système complexe doit être décomposé en parties modulaires afin de garantir une maintenance et une évolution facilitées. Cette architecture (fonctionnelle, logique, matérielle, etc.) doit être modélisée en UML et pas seulement documentée en texte.
- **piloté par les risques :** les risques majeurs du projet doivent être identifiés au plus tôt, mais surtout levés le plus rapidement possible. Les mesures à prendre dans ce cadre déterminent l'ordre des itérations.

1.6.1.1 Cycle de vie d'un processus unifié

Le processus unifié se décompose en quatre phases [25] :

1. **Inception :** la phase d'inception a pour objectif d'obtenir un consensus de l'ensemble des parties prenantes sur la vision ou la portée du projet, le périmètre et l'estimation globale.
2. **Elaboration :** la phase d'élaboration a pour objectifs d'établir et de valider l'architecture de référence, de lever les principaux risques ainsi que de spécifier en détail les exigences, c'est une phase exploratoire.
3. **Construction :** la phase de construction a pour objectif de livrer une première version opérationnelle du système, accompagnée de sa documentation.
4. **Transition :** la phase de transition a pour objectif de déployer et mettre en production la version finale de l'application, testée par les utilisateurs.

1.6.2 Méthode agile

Une méthode agile est une approche itérative et incrémentale, qui est menée dans un esprit collaboratif avec juste ce qu'il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins des clients [25].
Les quatre valeurs applicables dans toute démarche agile sont :

1. Les individus et leurs interactions, plus que les processus et les outils.
2. Des logiciels opérationnels, plus qu'une documentation exhaustive.
3. La collaboration avec les clients, plus que la négociation contractuelle.
4. L'adaptation au changement, plus que le suivi d'un plan.

1.6.3 Pratiques d'extrême Programming et les bases de Scrum

- **Scrum :** le Scrum désigne la réunion quotidienne (daily scrum meeting) qui réunit l'ensemble des acteurs du projet pour dresser, en un temps limité à quinze minutes, le bilan de la journée passée, planifier celle qui commence et repérer les éventuels obstacles rencontrés

par chacun. Le Scrum symbolise la solidarité et la force qui lient les membres de l'équipe pour le succès de l'itération [25].

- **XP (eXtreme Programming) :** est un cadre de développement logiciel agile qui vise à produire des logiciels de meilleure qualité. XP est le plus spécifique des cadres agiles concernant les pratiques d'ingénierie appropriées pour les logiciels [25].

  XP repose sur quatre valeurs [25] :

  1. **Communication :** l'effort de communication entre les différents intervenants est indispensable pour atteindre l'objectif commun.
  2. **Simplicité :** la solution la plus simple est la meilleure pour atteindre les objectifs, grâce à cette simplicité, l'application pourra évoluer facilement, si nécessaire. La simplicité est applicable au client dans la définition de ces besoins, dans le choix des outils et du processus.
  3. **Feedback :** le retour d'information est essentiel pour valider le fait que le projet est sur la bonne voie.
  4. **Courage :** le courage est absolument nécessaire pour adopter une démarche XP, démarrer un projet sans avoir toutes les spécifications, pour modifier le code existant sans en être l'auteur, et vice versa.

1.6.4 Démarche suivie

Dans un premier temps, les besoins vont être modélisés au moyen des cas d'utilisation UML. Ils seront représentés de façon plus concrète par une maquette d'IHM (Interface Homme-Machine) destinée à faire réagir les futurs utilisateurs. Chaque cas d'utilisation est décrit textuellement de façon détaillée, mais donne également lieu à un diagramme de séquence simple représentant graphiquement la chronologie des interactions entre les acteurs et le système vu comme une boîte noire. Par la suite, on modélise les diagrammes de classes participantes, qui sont des diagrammes de classes UML qui décrivent, cas d'utilisation par cas d'utilisation. Après, en remplaçant le système par un ensemble choisi d'objets de conception, on obtient les diagrammes d'interaction, et ensuite les diagrammes de navigation qui sont des diagrammes dynamiques représentant de manière formelle l'ensemble des chemins possibles entre les principaux écrans proposés à l'utilisateur et qui détaillent une exploitation supplémentaire de la maquette. Puis le diagramme de classe de conception qui représente bien la structure statique du code, par le biais des attributs et des relations entre classes [24].

Les différentes étapes du processus simplifié sont présentées dans la Figure 1.4.

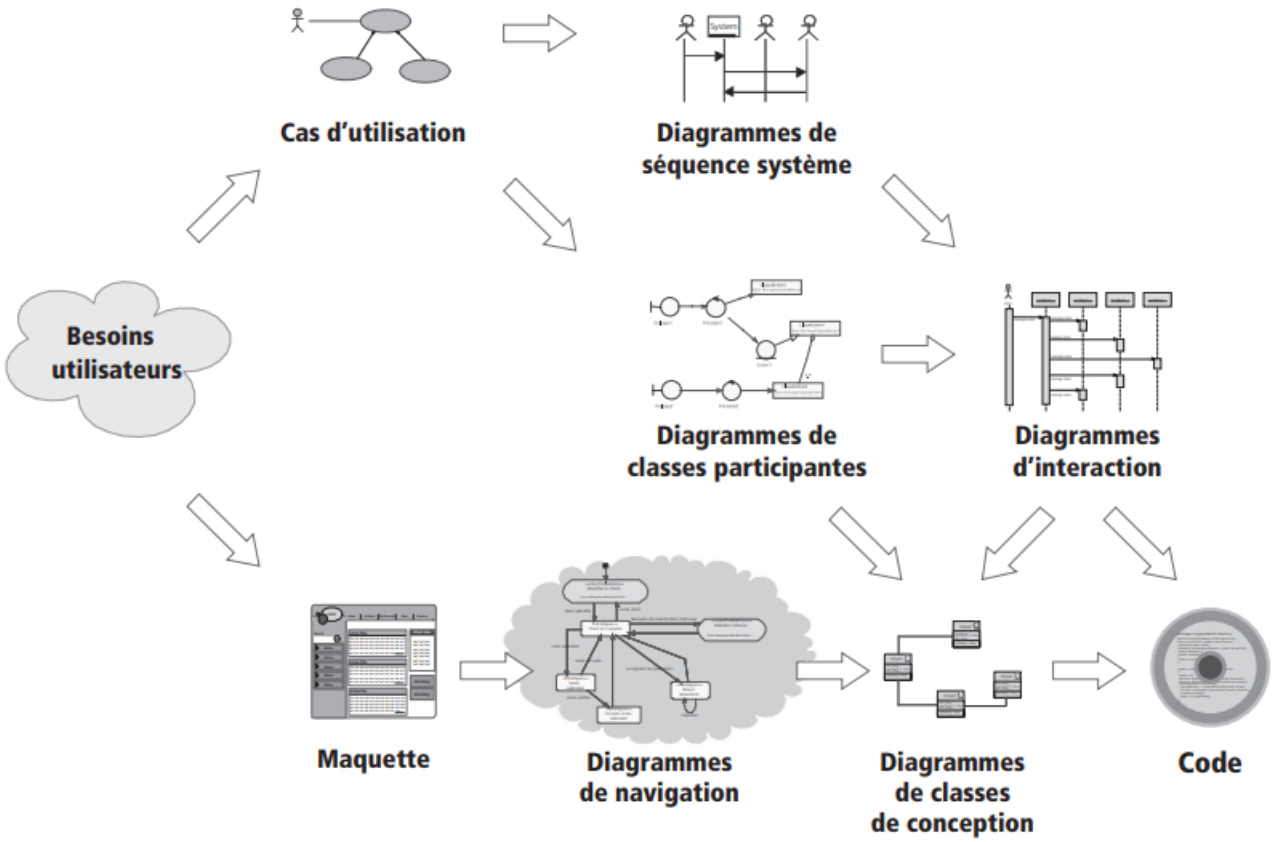


FIGURE 1.4 – Schéma complet du processus simplifié pour la modélisation d'une application web [24]

1.7 Modèle MVC

Le modèle de conception MVC (Model-View-Controller) est très utile pour l'architecture de systèmes logiciels interactifs, c'est une architecture de développement visant à séparer les différentes couches constituant une application interactive, de manière à simplifier la gestion de chacune (le modèle, la vue, le contrôleur). Le point essentiel consiste à séparer les objets graphiques des objets métiers, afin de pouvoir les faire évoluer indépendamment et les réutiliser. Le modèle MVC est représenté dans la figure 1.5 [26].

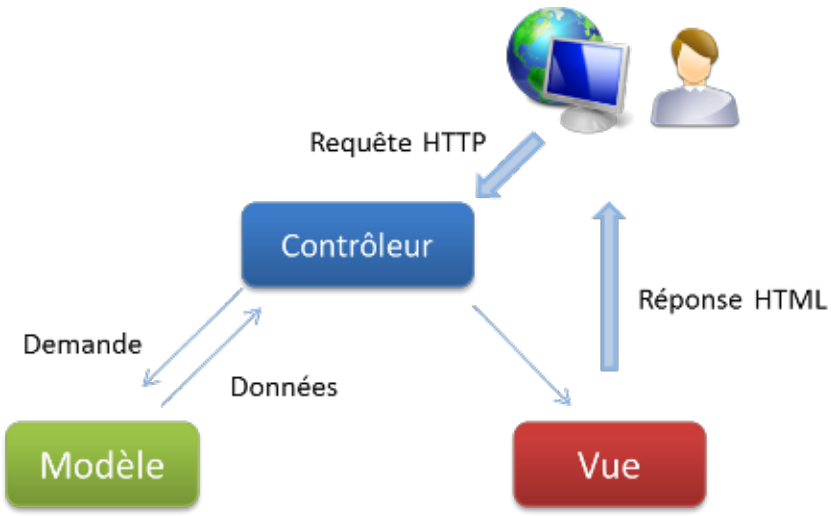


FIGURE 1.5 – Le paradigme MVC de base
[27]

- **Le modèle :** cette partie gère les données de l'application, son rôle se base sur la récupération des informations à partir de la base de données pour qu'elles puissent être traitées par le contrôleur.
- **La vue :** représente les données du modèle à l'écran d'utilisateur, il se maintient à jour lorsque le modèle est modifié.
- **Le contrôleur :** cette partie gère la synchronisation entre la vue et le modèle. Le contrôleur va demander au modèle les données puis les analyser et à la fin prendre des décisions et renvoyer le texte à afficher à la vue. C'est notamment lui qui détermine si le visiteur a le droit de voir la page ou non (gestion des droits d'accès). Aussi il est responsable de la sélection des types de modèle à utiliser et de la vue à afficher.

1.7.1 Avantages du MVC

Il y a plusieurs avantages de MVC, nous citons :

- Le conception MVC permet de séparer la logique métier, l'interface utilisateur et l'entrée du système.
- Le code est modulaire, et facilement réutilisable, donc il y a un gros gain de temps.
- Permet à plusieurs développeurs de travailler sur le même projet en parallèle.
- Maintenance facile niveau design et code.

1.7.2 Inconvénients du MVC

Le modèle MVC souffre de plusieurs inconvénients, parmi eux nous citons :

- Difficile à comprendre pour une personne extérieure.
- Si le code de contrôleur est modifié, il faut recompiler class et redéployer l'application.
- Le MVC est trop complexe pour des petites applications.

1.8 Conclusion

Dans ce chapitre, nous avons donné un petit aperçu sur le mode d'apprentissage qui est l'enseignement à distance ou bien l'e-learning, puis nous avons exploré le domaine des applications mobiles et web. Nous nous sommes intéressés au processus simplifié et le modèle architectural (MVC), la plateforme mobile Android et la plateforme de développement web JEE.

Dans le chapitre suivant, nous entamerons l'étape de spécification des besoins dans laquelle nous décrirons d'une manière détaillée les besoins de notre application.

Chapitre 2

Spécification des besoins

2.1 Introduction

La spécification des besoins est la première phase du cycle de vie d'une application, cette phase donne une image plus claire de logiciel à développer. Dans ce chapitre nous allons identifier les différentes fonctionnalités de notre application ainsi que les différents acteurs, puis en suivant le processus simplifié nous modélisons ces fonctionnalités en utilisant les diagrammes de cas d'utilisation UML (UCs). Ensuite, nous représentons de façon plus concrète ces UCs par des maquettes d'interface homme-machine (IHM). Enfin, nous décrivons chaque cas d'utilisation de façon détaillée par des fiches types et des diagrammes de séquences (DSS) qui représentent graphiquement les interactions entre les acteurs et le système vu comme une boîte noire.

2.2 Description des fonctionnalités

2.2.1 Objectif

L'objectif de l'application est de faciliter l'enseignement au sein de l'université de Jijel. L'application permet principalement aux enseignants de rendre disponibles en ligne des contenus pédagogiques aux étudiants, et ces derniers s'occupent de télécharger les cours disponibles sur la plateforme, ainsi qu'elle offre un moyen interactif qui facilite la communication entre les enseignants et les étudiants, aussi bien que la possibilité d'informer les étudiants lors d'une publication par un enseignant.

2.2.2 Fonctionnement

L'application permet aux enseignants de rendre en ligne des contenus pédagogiques, ces contenus sont classés dans trois sections qui est cours, travaux demandés et affichages. Dans la première section l'enseignant peut ajouter des cours (des pièces jointes associées à des titres et des descriptions). Comme il peut ajouter dans la section travaux demandés des exercices de travaux dirigés (td), des exposés, des rapports etc. Dans la section affichage, l'enseignant peut ajouter toute information

d'intérêt pour les étudiants comme les dates des contrôles, les notes, les consultations etc. Ainsi, elle permet aux étudiants d'accéder aux contenus pédagogiques disponibles sur la plate-forme.

Les enseignants et les étudiants peuvent communiquer entre eux en ajoutant des commentaires sur les publications ou bien en ouvrant des discussions sur n'importe quels sujets dans une section qui est nommée forum. Une fois qu'un contenu pédagogique est disponible, l'étudiant lié à cette publication reçoit une notification sur son téléphone portable.

l'application comporte deux partie : une partie mobile destinée uniquement aux enseignants et aux étudiants et une partie web qui est pour l'administrateur, les enseignants et les étudiants.

1. **Partie mobile :**

   L'application mobile permet aux enseignants de :
   - *Ajouter cours* : pour que l'enseignant ajoute un cours il faut d'abord choisit un fichier, puis spécifie les groupes et le publie sur l'application comme cours. Avant la publication, il a la possibilité d'ajouter un titre et une description sur ce cours et de spécifier si ce dernier sera visible ou invisible pour les étudiants. Ceci est fait après que l'enseignant ait sélectionné la spécialité et le module qui s'y rapporte.
   - *Supprimer ou modifier le cours* : l'enseignant a la possibilité de supprimer ou modifier un cours s'il pense qu'il est devenue inutile ou présente une lacune.
   - *Ajouter affichages* : l'enseignent publie des informations dans la section affichage telles qu'une date de consultation, une absence d'un enseignant, une correction ou une liste des notes. Avant que l'enseignant publie ces informations, il doit d'abord spécifier le type d'affichage qui ce soit une correction (interrogation, contrôle ou rattrapage), liste des notes, ou autres informations.
   - *Ajouter des travaux à faire* : l'enseignant a la possibilité d'ajouter des travaux à faire en poursuivant la même procédure suivi dans la publication des cours. Ces travaux seront des rapports TP, des exercices TD, des exposés etc.

   En plus, elle permet aux étudiants de :
   - *Consulter les publications* : l'étudiant peut consulter les publications disponibles affichées par les enseignants (cours, affichage, exercice etc.), et télécharger les fichiers liés à ces publications s'ils existent.
   - *Ajouter des solutions* : l'étudiant a la possibilité d'envoyer les solutions des travaux qui sont demandés par l'enseignant (TP, rapport TP, exposé etc).

   En plus les fonctionnalités citées précédemment l'étudiant et l'enseignant peuvent :
   - Modifier leurs profiles (photo de profile, mot de passe, numéro de téléphone, email).
   - Ajouter des nouveaux sujets de discussion ou des commentaires sur des sujets posés dans la section forum.

2. **Partie web :**

   L'application web offre à l'administrateur les fonctionnalités suivantes :
   - Gérer les comptes des étudiants et des enseignants (ajouter, supprimer ou modifier).
   - Consulter la liste des étudiants et celle des enseignants.
   - Consulter les listes des facultés, des départements, des spécialités et des groupes ou bien les ajouter, les modifier ou les supprimer.

Les fonctionnalités fournies par l'application web aux enseignants et aux étudiants sont les mêmes fonctionnalités fournies par l'application mobile. De plus, les enseignants peuvent ajouter des tests qui contiennent des paires de questions-réponses et qui aident les étudiants à tester leur compréhension d'un module particulier, et les étudiants peuvent répondre à ces tests.

2.3 Démarche

Dans un premier temps, les besoins vont être modélisés au moyen des cas d'utilisation (UCs). Ils seront représentés par une maquette d'IHM. Chaque cas d'utilisation est décrit textuellement de façon détaillée, puis représenté sous forme de DSS.
La figure 2.1 représente les étapes suivis par cette démarche.

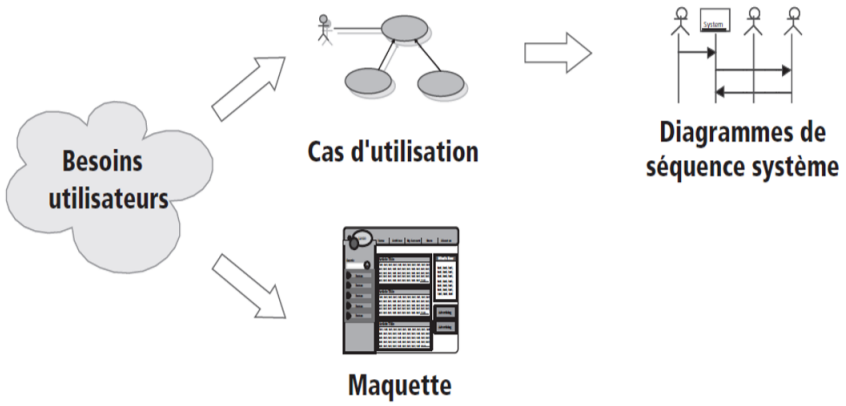


FIGURE 2.1 – Les cas d'utilisation et leurs prolongements dans la démarche [24]

2.4 Spécification des besoins d'après les cas d'utilisations et des IHMs

Dans cette partie nous allons détailler les deux premières étapes de la démarche, en les résumant dans les étapes suivantes :

- Identification des différents acteurs, un acteur est un utilisateur type qui a toujours le même comportement vis-à-vis d'un cas d'utilisation. Il peut se représenter symboliquement par un « bonhomme » et être identifié par son nom [28].

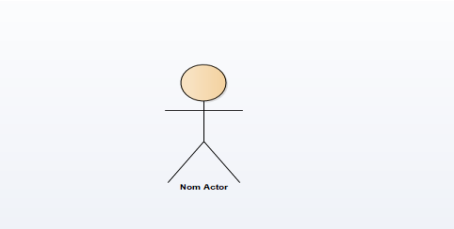


FIGURE 2.2 – Représentation de l'acteur

- Identification des cas d'utilisation, un cas d'utilisation correspond à un certain nombre d'actions que le système devra exécuter en réponse à un besoin d'un acteur. Il se représente par un ovale dans lequel figure son intitulé [28].

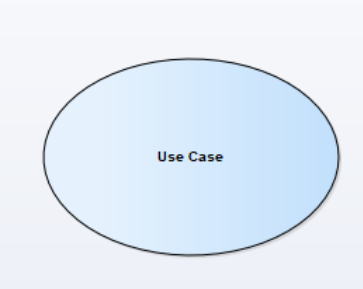


FIGURE 2.3 – Représentation d'un cas d'utilisation

- Ajouter des relations entre les acteurs et les UCs qui permettent de décrire les échanges entre eux et finaliser le diagramme de cas d'utilisation.
- Représenter les maquettes d'IHM.

2.4.1 Identification des acteurs

Les acteurs de notre système sont les suivants :

- **Acteur *administrateur :*** c'est une personne qui est chargée d'effectuer des tâches administratives telles que : gérer les comptes des utilisateurs, gérer les facultés, les modules, les spécialités et les groupes.
- **Acteur *enseignant :*** c'est toute enseignant appartenant à l'université et qui possède déjà un compte et peut gérer des cours, ajouter des travaux à faire, des listes des notes, et aussi répondre aux questions posées.
- **Acteur *étudiant :*** représente tout étudiant inscrit à l'université et possède un compte sur l'application, qui lui accorde la possibilité de suivre des cours en ligne, transmettre des travaux demandés, consulter les listes des notes et poser des questions.

2.4.2 Identification des cas d'utilisations

Pour chaque acteur identifié précédemment, nous allons spécifier ses cas d'utilisations.

1. **UCs de l'acteur *administrateur***

La figure 2.4 présente les UCs de l'acteur *administrateur*.

- *Gérer les comptes* : lui permet de gérer les comptes des enseignants et des étudiants (Ajouter, supprimer, modifier et consulter la liste des comptes).
- *Gérer les facultés* : lui permet d'ajouter, modifier, supprimer et de consulter la liste des facultés.
- *Gérer les départements :* lui permet d'ajouter, modifier, supprimer et de consulter la liste des départements.
- *Gérer les niveaux* : lui permet d'ajouter, modifier, supprimer et de consulter la liste des niveaux tel que 1ère année MI, master 1 RS, master 1 SIAD, 2éme année informatique.
- *Gérer les groupes* : lui permet d'ajouter, modifier, supprimer et de consulter la liste des groupes.
- *Gérer les modules* : lui permet d'ajouter, modifier, supprimer et de consulter la liste des modules.

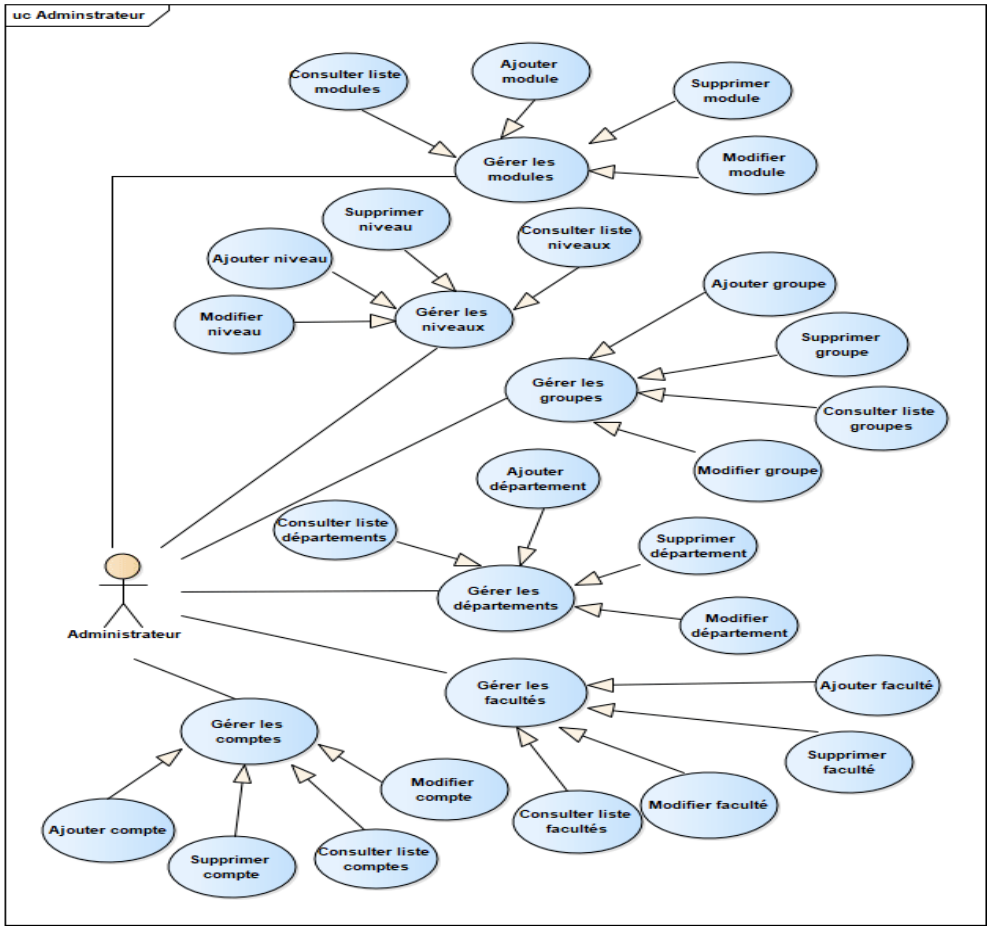


FIGURE 2.4 – UCs de l'acteur *administrateur*

2. **UCs de l'acteur *enseignant***

   La figure 2.5 présente les UCs de l'acteur *enseignant*.

   - *Consulter liste étudiants* : l'*enseignant* peut consulter la liste des étudiants après avoir spécifié le niveau et le groupe.
   - *Gérer cours* : lui permet d'ajouter, modifier ou supprimer des cours.
   - *Gérer affichage* : lui permet d'ajouter, modifier ou supprimer des affichages.
   - *Gérer travaux* : lui permet d'ajouter, modifier ou supprimer des travaux.
   - *Gérer tests* : lui permet d'ajouter ou supprimer des tests.
   - *Participer au forum* : l'*enseignant* a la possibilité d'ajouter un sujet de discussion ou bien répondre à un sujet posé.
   - *Modifier profile* : l'*enseignant* a la possibilité de modifier certaines informations de son espace personnel telles que la photo de profile, l'email, le numéro de téléphone ou le mot de passe.
   - *Sélectionner cours* : une fois que l'*enseignant* sélectionne un cours il a la possibilité de le télécharger ou y ajouter un commentaire.
   - *Sélectionner travail* : l'*enseignant* a la possibilité de sélectionner un travail puis de le télécharger ou y ajouter un commentaire.
   - *Sélectionner affichage* : une fois que l'*enseignant* sélectionne un affichage il a la possibilité de le télécharger ou y ajouter un commentaire.

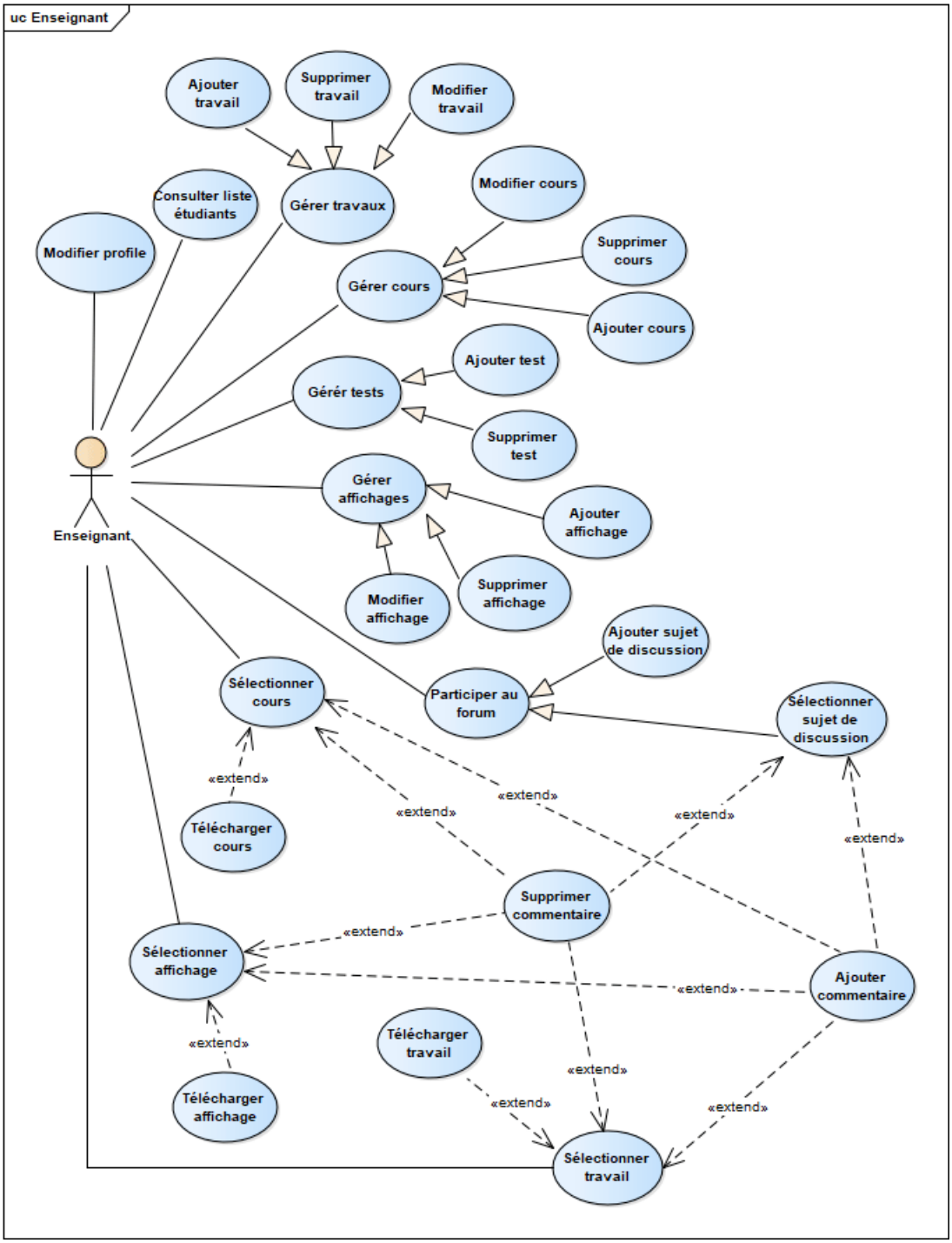


FIGURE 2.5 – UCs de l'acteur *enseignant*

3. **UCs de l'acteur *étudiant***

La figure 2.6 présente les UCs de l'acteur *étudiant*.

- *Sélectionner cours* : une fois que l'*étudiant* sélectionne un cours particulier, il peut le télécharger ou y ajouter un commentaire.
- *Sélectionner travail demandé* : l'*étudiant* a la possibilité de sélectionner un travail demandé puis de le télécharger ou y ajouter un commentaire.
- *Sélectionner affichage* : une fois que l'*étudiant* sélectionne un affichage il a la possibilité de le télécharger ou y ajouter un commentaire.
- *Effectuer test* : lorsque l'*enseignant* ajoute un test, l'*étudiant* peut répondre à ses questions, où à la fin il obtient un score montrant l'étendue de sa compréhension d'un module particulier.
- *Participer au forum* et *Modifier profile* : c'est les mêmes cas d'utilisation de l'*enseignant*.

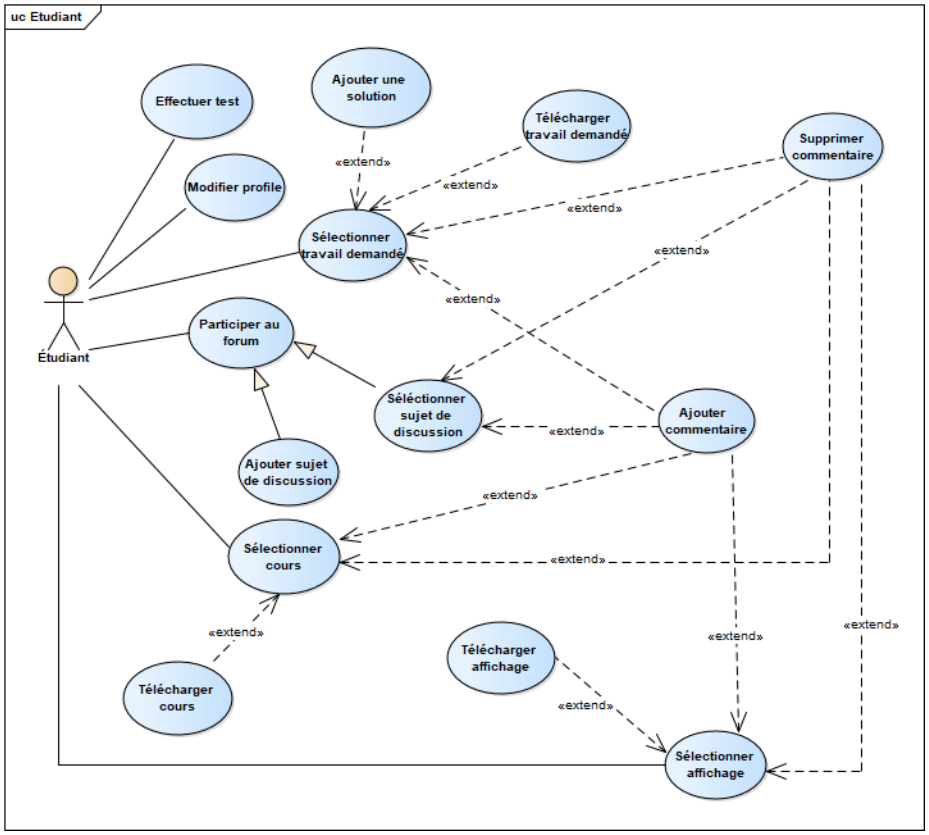


FIGURE 2.6 – UCs de l'acteur *étudiant*

2.4.3 Diagramme de cas d'utilisation global

Nous allons maintenant organiser les cas d'utilisation cités précédemment et les regrouper dans les trois diagrammes de cas d'utilisation suivants : UCs des utilisateurs, UCs d'administrateur, UCs de second rang.

1. **UCs des utilisateurs**
   Nous avons créé un acteur généralisé nommé *utilisateur*, dont l'*enseignant* et l'*étudiant* seront des spécialisations. Nous obtiendrons le diagramme dans la figure 2.7.

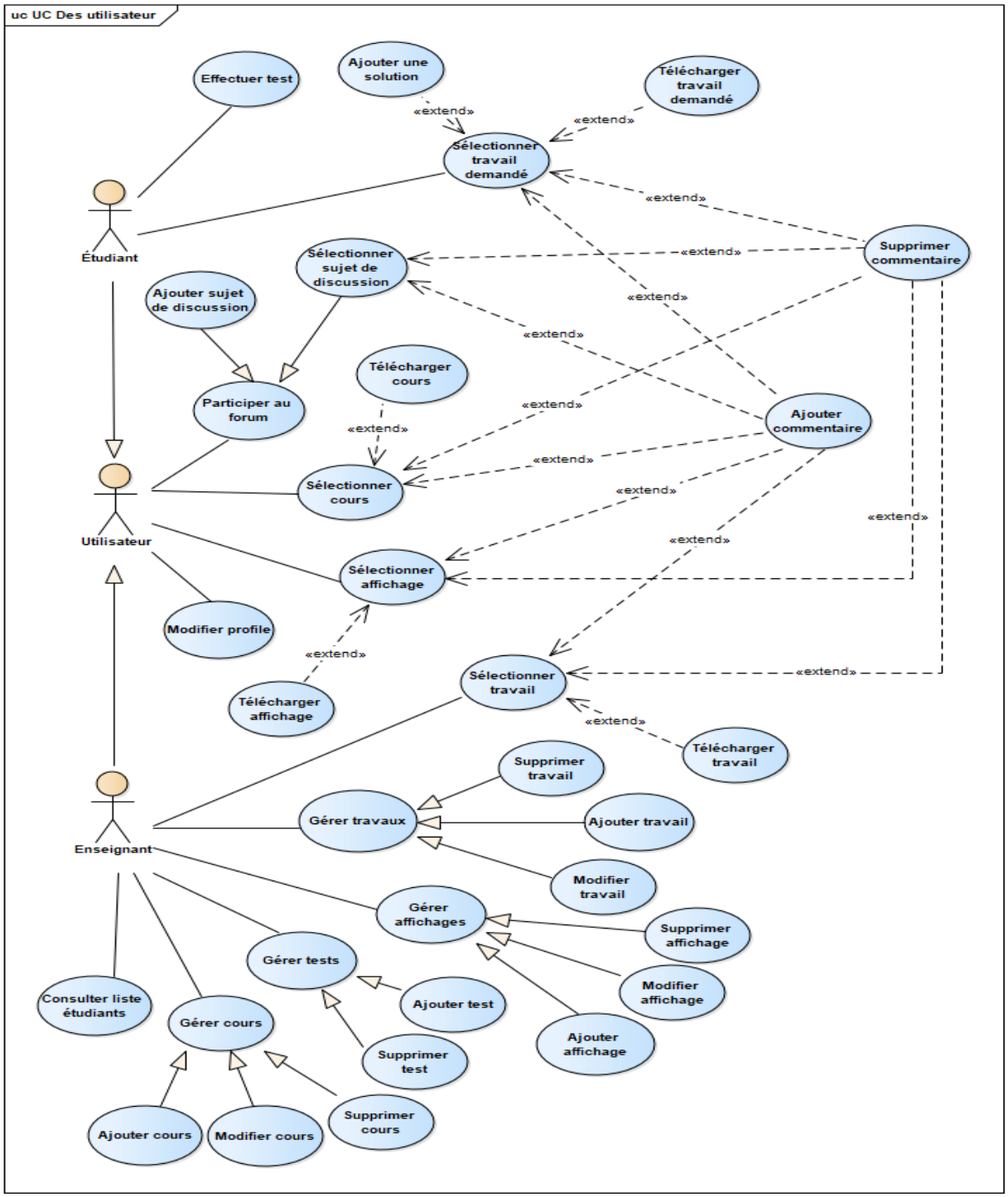


FIGURE 2.7 – UCs de l'utilisateur

2. **UCs de l'*administrateur***
   Le digramme de cas d'utilisation global de l'*administrateur* est le même que les UCs de l'*administrateur* spécifié précédemment (la figure 2.4) parce qu'on a un seul *administrateur*.

3. **UCs de second rang**
   Le diagramme ci-dessous présente les UCs de second rang. Un UC de second rang est un UC qui ne représente pas l'objectif principal d'un acteur. Le cas d'utilisation *authentification* devra être réalisé afin de permettre à l'utilisateur et à l'*administrateur* d'exécuter ses propres cas d'utilisation majeurs. Nous qualifierons donc ce cas d'utilisation par le stéréotype « Fragment » : il ne représente pas un objectif à part entière d'utilisateur et d'*administrateur*.

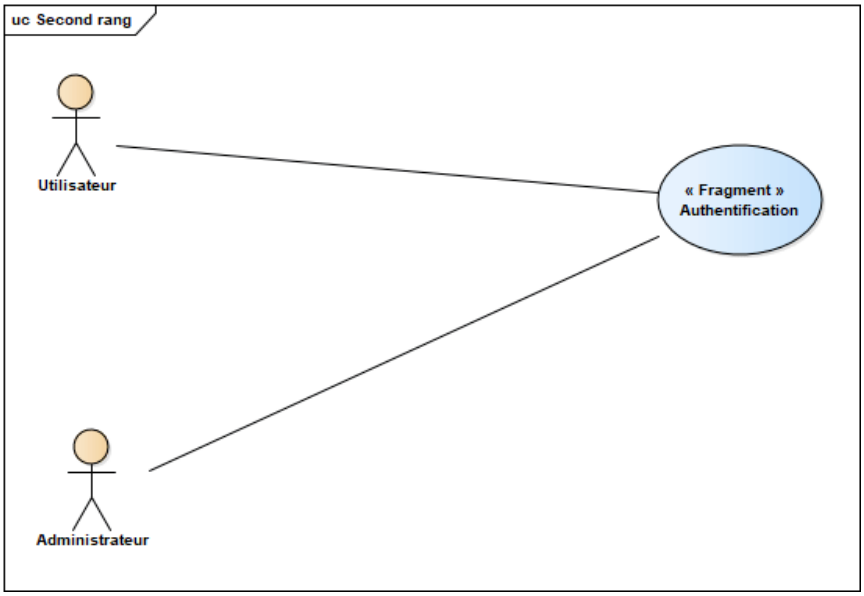


FIGURE 2.8 – UCs de second rang

2.4.4 Structuration des cas d'utilisation en package

Nous allons regrouper les cas d'utilisation en trois packages présentés dans la figure 2.9 :

- **Package des utilisateurs :** contient les utilisateurs et leurs UCs.
- **Package d'administrateur :** contient l'*administrateur* et son UCs.
- **Package de second rang :** contient l'use case authentification.

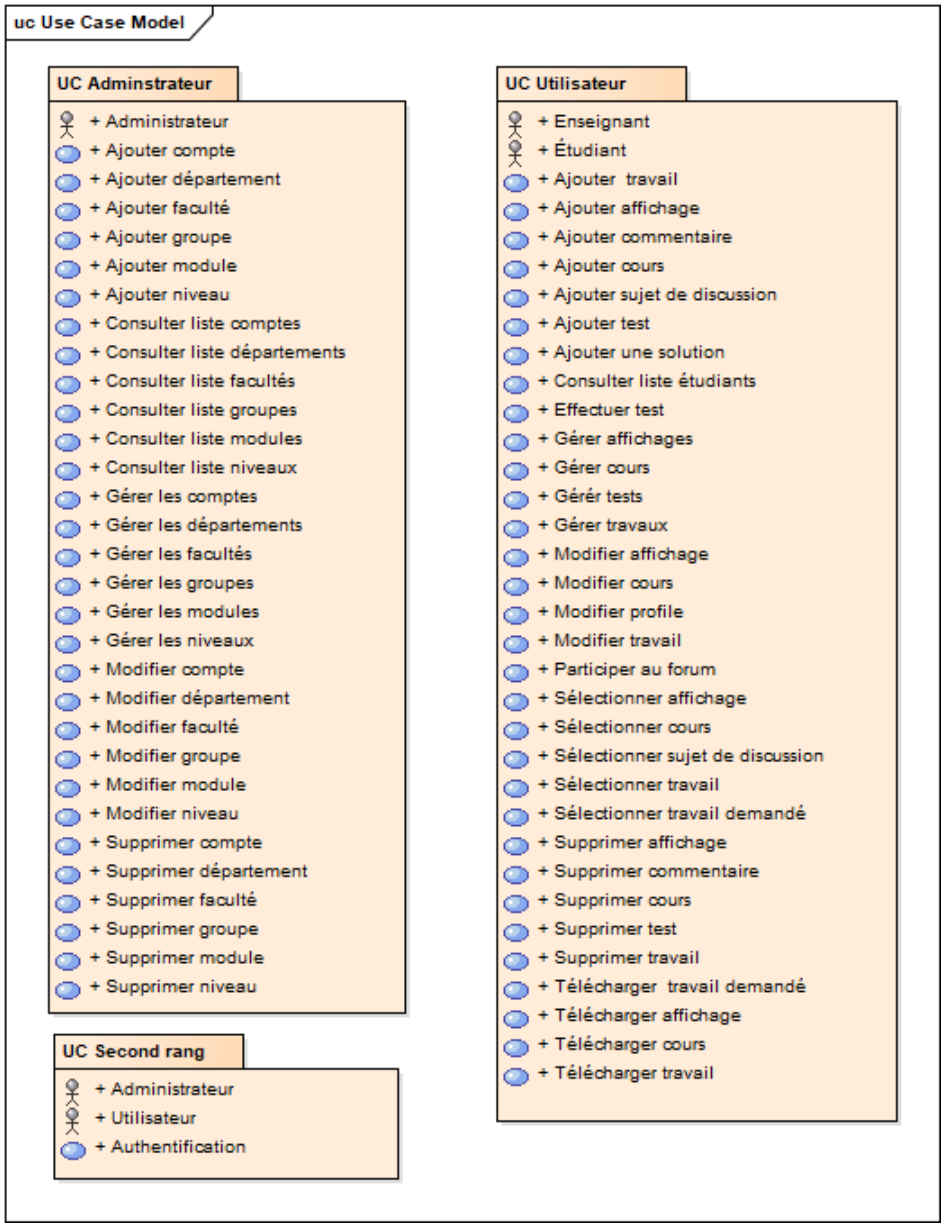


FIGURE 2.9 – Organisation des cas d'utilisation et des acteurs en packages (avec l'outil Enterprise Architect)

2.4.5 Présentation des IHMs

Notre application est composée de trois IHMs principal dont chacune représente l'espace personnel de l'*enseignant*, l'*étudiant* et de l'*administrateur*. Dans cette partie, nous allons présenter les IHMs les plus importantes de chaque acteur.

1. **IHMs de tous les acteurs**
   Les utilisateurs et l'*administrateur* possède une interface commune qui est l'IHM login.

(a) **IHM login**

Après avoir installé l'application ou connecté au site web, l'*administrateur* se trouve sur l'IHM représenté dans la figure 2.11, et l'utilisateur se trouve sur l'une des IHMs représentées dans les figures ci-dessous (les figures 2.10 et 2.11) à partir de lesquelles il peut accéder à son espace.

Si le nom ou le mot de passe sont erronés le système affiche une alerte, sinon il va ouvrir l'espace personnel.

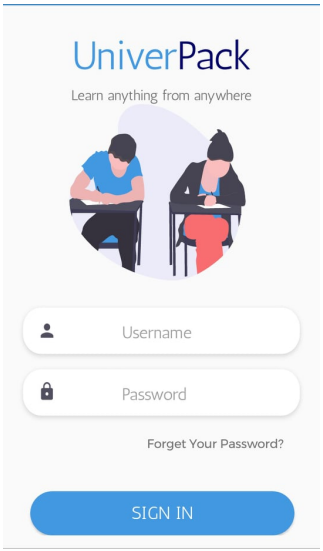


FIGURE 2.10 – IHM login de l'*enseignant* et l'*étudiant* sur l'application

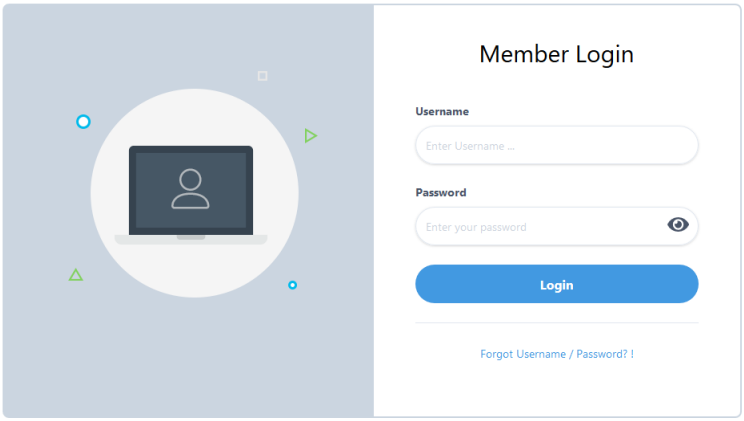


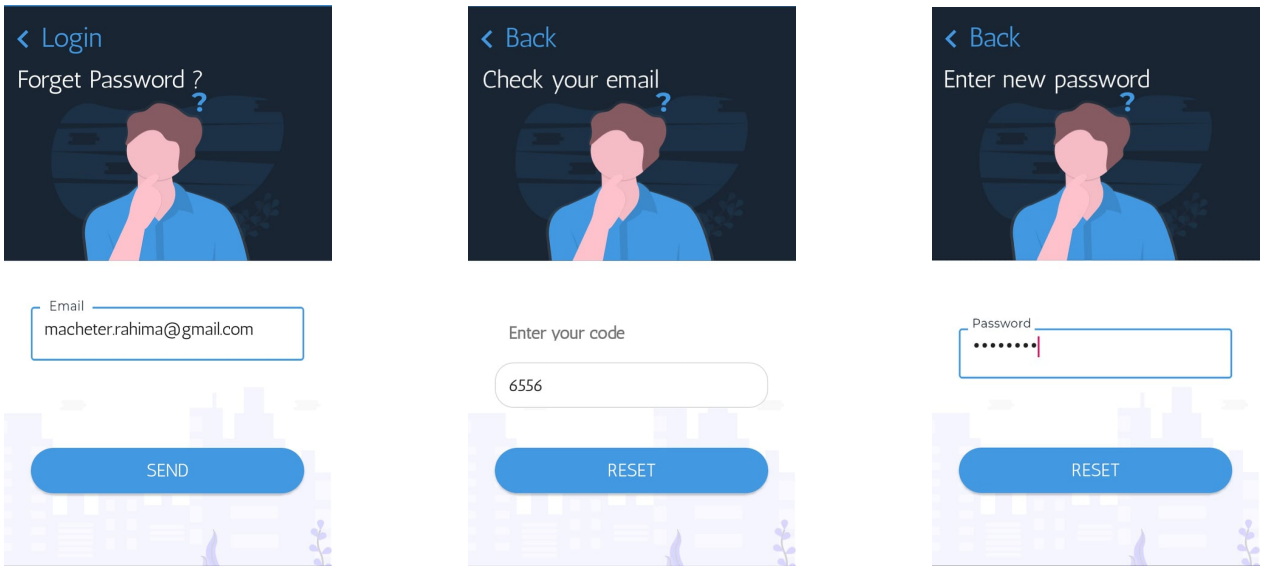
FIGURE 2.11 – IHM login sur le site web

2. **IHMs communes des utilisateurs**

Les utilisateurs possèdent des IHMs communes telles que : l'IHM mot de passe oublié et l'IHM forum.

(a) **IHM mot de passe oublié**

Si l'*étudiant* ou l'*enseignant* oublie le mot de passe de son compte, il peut accéder à son espace personnel en envoyant une demande de réinitialisation de mot de passe via son e-mail. Un code à quatre chiffres valide pendant 10 minutes sera envoyé à l'e-mail, où il le retapera et définira un nouveau mot de passe. Les étapes de cette procédure sont définies dans les figures ci-dessous.



(a) IHM Mot de passe oublié (b) IHM Vérifier email (c) IHM réinitialiser password

FIGURE 2.12 – IHM Mot de passe oublié sur l'application

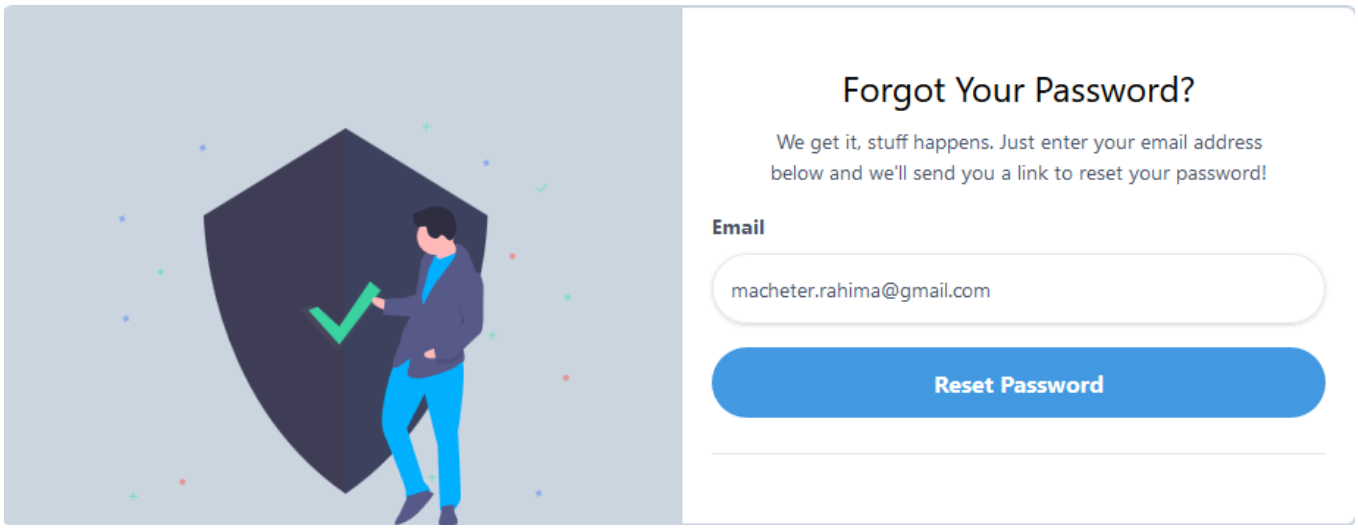


FIGURE 2.13 – IHM Mot de passe oublié sur le site

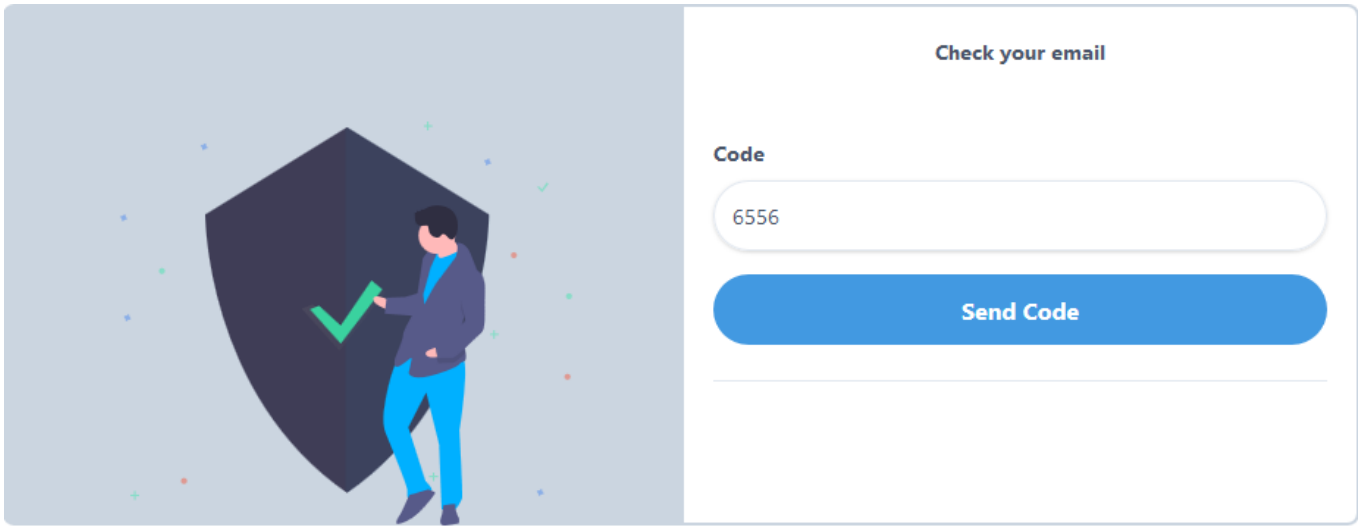


FIGURE 2.14 – IHM Vérifier email sur le site

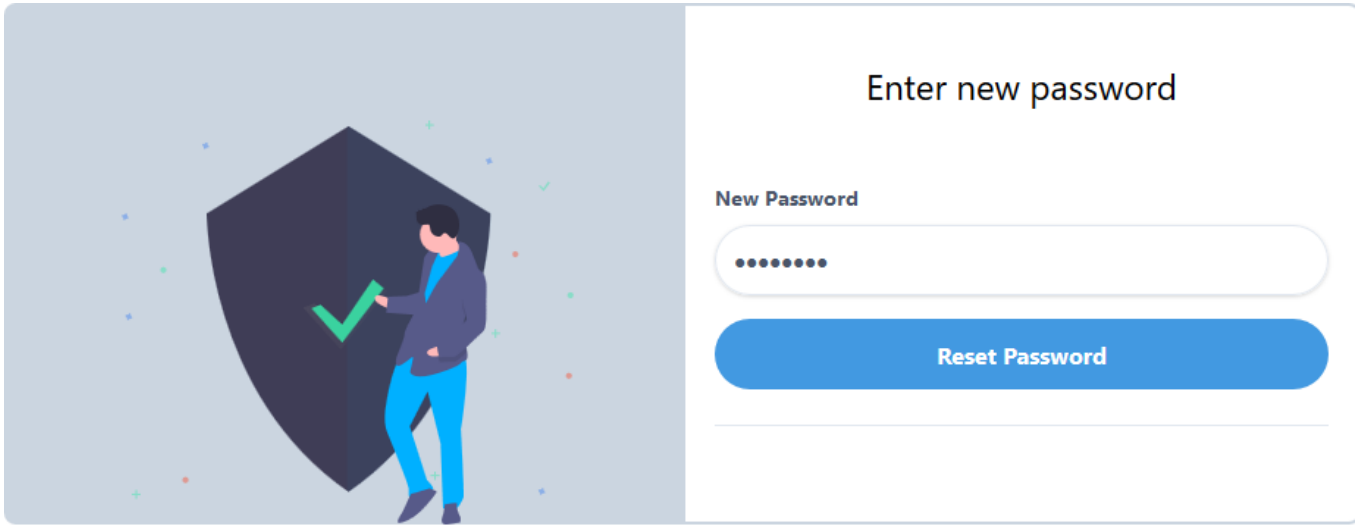
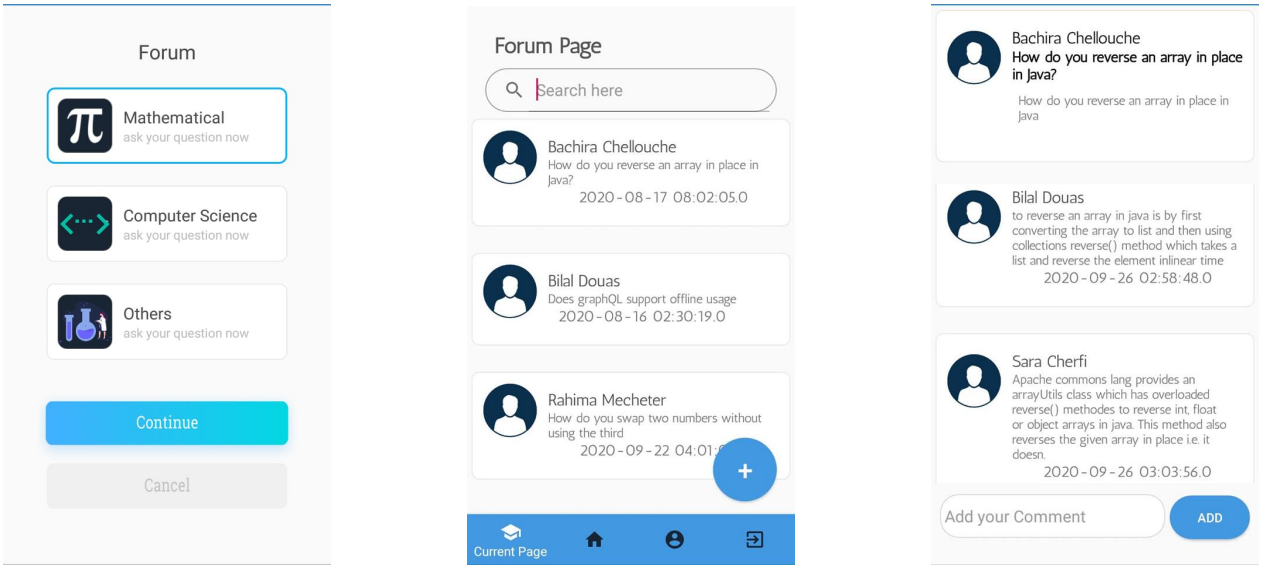


FIGURE 2.15 – IHM réinitialiser password sur le site

(b) **IHM forum**
Dans cette page l'*enseignant* et l'*étudiant* peuvent ajouter des sujets de discussions ou commenter sur des sujets qui existent déjà ou bien répondre aux commentaires des autres utilisateurs. Les figures 2.16 et 2.17 représentent respectivement l'IHM forum dans l'application mobile et le site web.



(a) IHM type des sujets de discussion (b) IHM de page questions (c) IHM de page commentaires

FIGURE 2.16 – IHM forum sur l'application

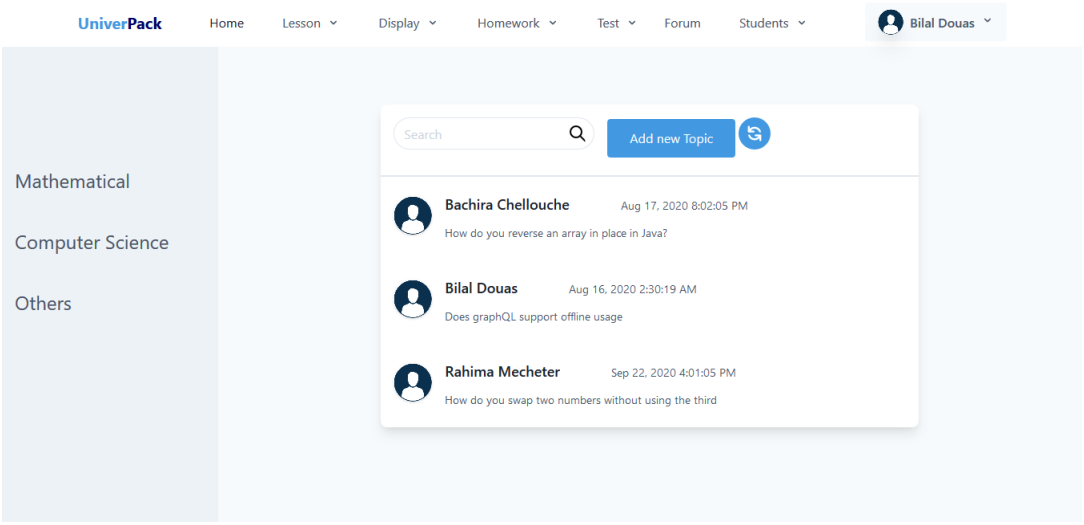


FIGURE 2.17 – IHM forum sur le site

3. **IHMs de l'acteur *enseignant***
   L'*enseignant* possède plusieurs IHMs telles que : IHM principale, IHM liste des cours, IHM ajouter un devoir à faire et IHM ajouter un test.

   (a) **IHM principale (Home Page)**
   Cette IHM contient des liens vers d'autres pages telles que : la page des cours, la page des devoirs, la page des affichages afin de faciliter l'accès de l'*enseignant* à ces pages.

Les figures 2.18 et 2.19 représentent respectivement l'IHM principale de l'*enseignant* sur l'application mobile et sur le site web.

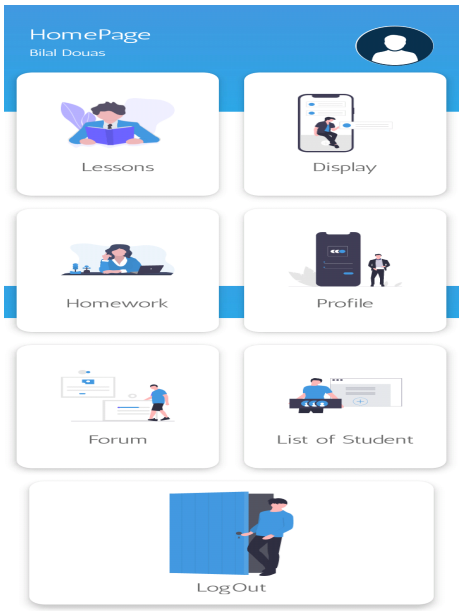


FIGURE 2.18 – IHM principale de l'*enseignant* sur l'application

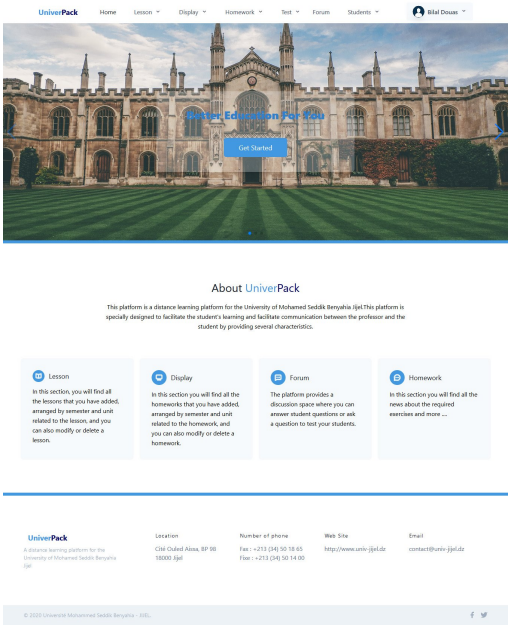
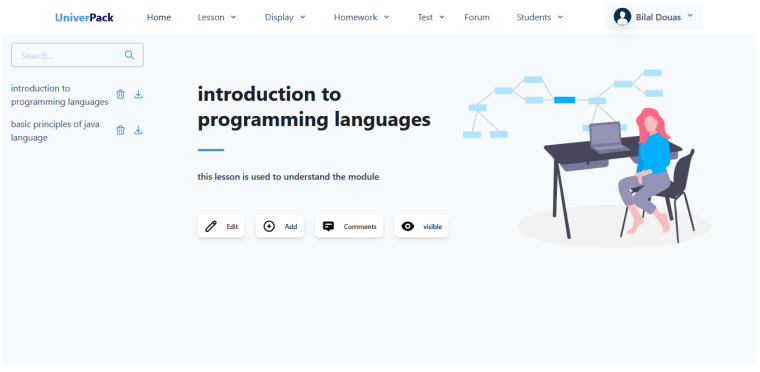


FIGURE 2.19 – IHM principale de l'*enseignant* sur le site

(b) **IHM liste des cours**

Dans cette interface l'*enseignant* peut ajouter, télécharger, supprimer ou bien modifier ses cours. Il a la possibilité également d'ajouter des commentaires sur un cours particulier. Cet IHM est représentée dans la figure 2.20.

(a) IHM liste des cours sur l'application



(b) IHM liste des cours sur le site

FIGURE 2.20 – IHM liste des cours

(c) **IHM Ajouter un devoir à faire**

Les figures 2.21 et 2.22 représentent respectivement l'interface ajouter un devoir dans l'application mobile et dans le site web, qui permet à l'*enseignant* de choisir un fichier à partir de son appareil et de remplir tous les champs liés à ce fichier, puis cliquer sur ajouter devoir.

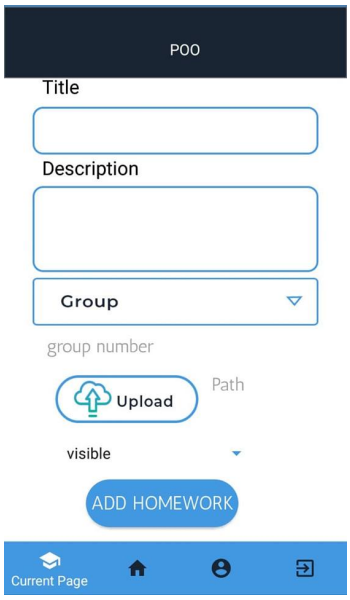


FIGURE 2.21 – IHM Ajouter un devoir à faire sur l'application

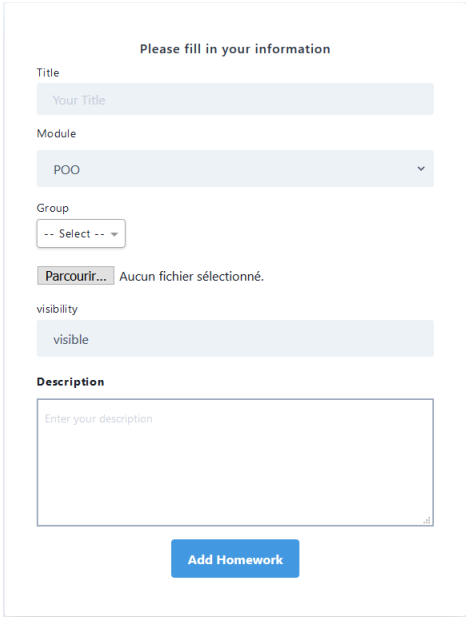


FIGURE 2.22 – IHM Ajouter un devoir à faire sur le site

(d) **IHM Ajouter un test**

À partir de cette interface (figure 2.23) l'*enseignant* peut ajouter des tests pour que les étudiants testent leur compréhension d'un module particulier. Donc il choisit un titre et une description puis clique sur ajouter puis ajoute quelques paires question/réponse à ce test.

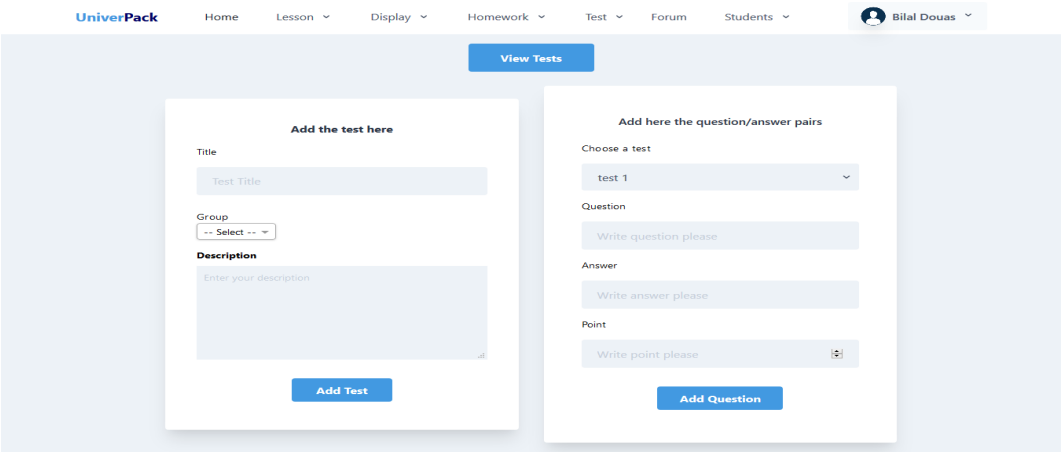


FIGURE 2.23 – IHM Ajouter un test

4. **IHM de l'acteur *étudiant***
   L'*étudiant* possède plusieurs IHMs telles que : IHM principale, IHM liste d'affichages, IHM ajouter une solution d'un travail demandé.

   (a) **IHM principale (Home Page)**
   Cette IHM (les figures 2.24 et 2.25) permet aux étudiants d'accéder aux plusieurs pages telles que la page des cours, la page des travaux demandés et la page d'affichage ainsi que la page de profile et la page de forum.

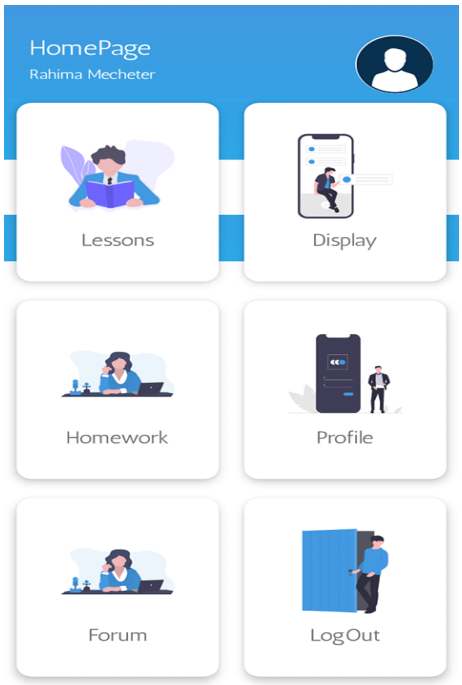


FIGURE 2.24 – IHM principale de l'*étudiant* sur l'application

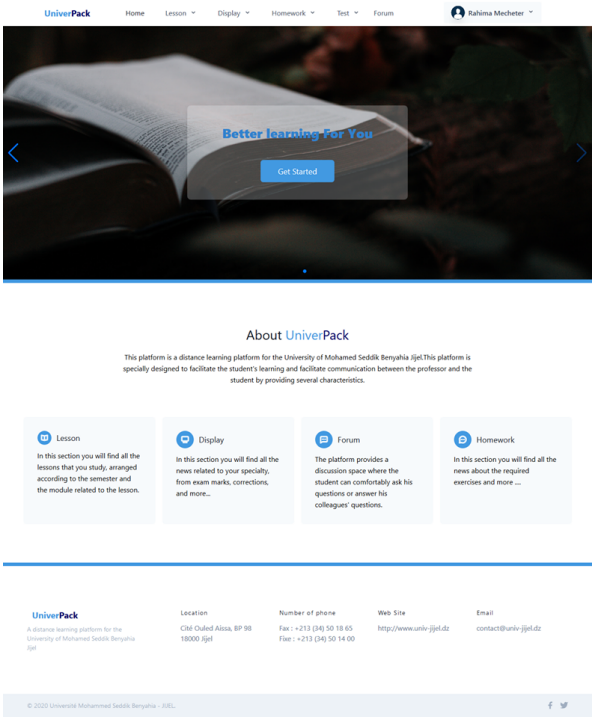
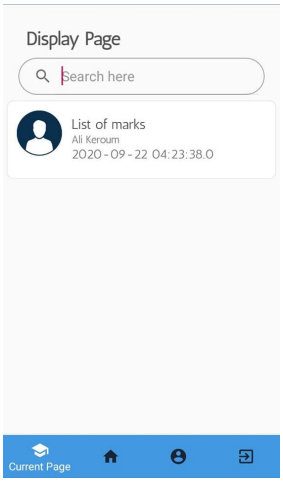
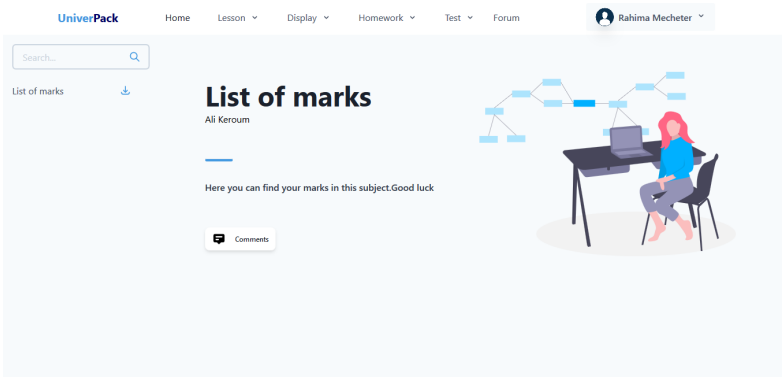


FIGURE 2.25 – IHM principale de l'*étudiant* sur le site

   (b) **IHM liste d'affichages**
   Chaque *étudiant* peut télécharger des affichages ou ajouter des commentaires sur un affichage particulier. L'interface liste d'affichages est présenté dans la figure 2.26.



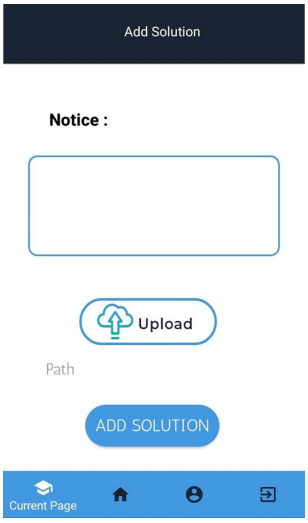
(a) IHM liste d'affichages sur l'application



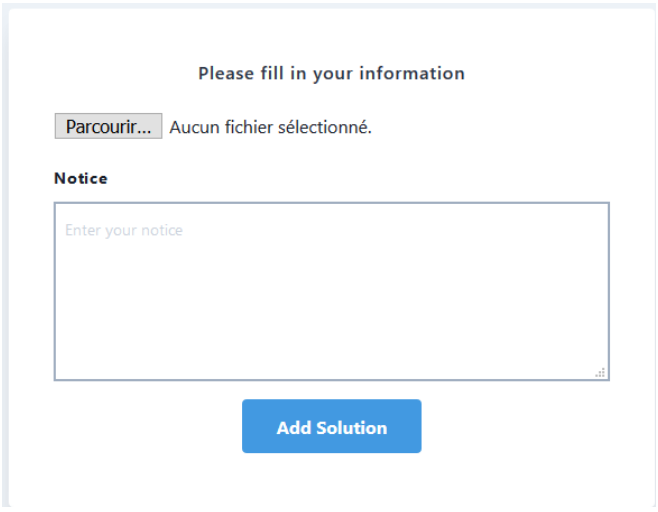
(b) IHM liste d'affichages sur le site

FIGURE 2.26 – IHM liste d'affichages

(c) **IHM ajouter une solution d'un travail demandé**
L'*étudiant* choisit un fichier et remplit le champ de notice puis clique sur le bouton ajouter.



(a) IHM d'ajouter une solution d'un travail demandé sur l'application



(b) IHM d'ajouter une solution d'un travail demandé sur le site

FIGURE 2.27 – IHM d'ajouter une solution d'un travail demandé

5. **IHM de l'acteur *administrateur***
L'*administrateur* possède des interfaces qui lui permettent de gérer les utilisateurs, les modules, les facultés, les départements, etc. Nous avons sélectionné quelques interfaces à montrer qui sont : IHM liste utilisateurs, IHM ajouter un niveau et l'IHM modifier un département.

(a) **IHM liste utilisateurs**
Dans cette interface, l'*administrateur* peut modifier ou supprimer des utilisateurs. Il peut également ajouter des comptes pour les utilisateurs qui ne l'ont pas encore.

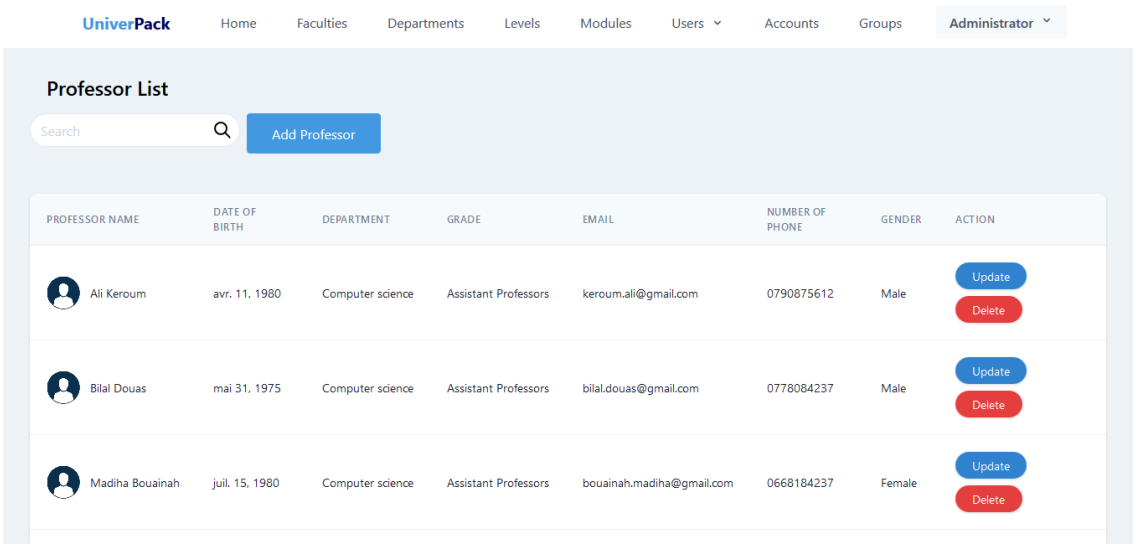


FIGURE 2.28 – IHM liste des enseignants

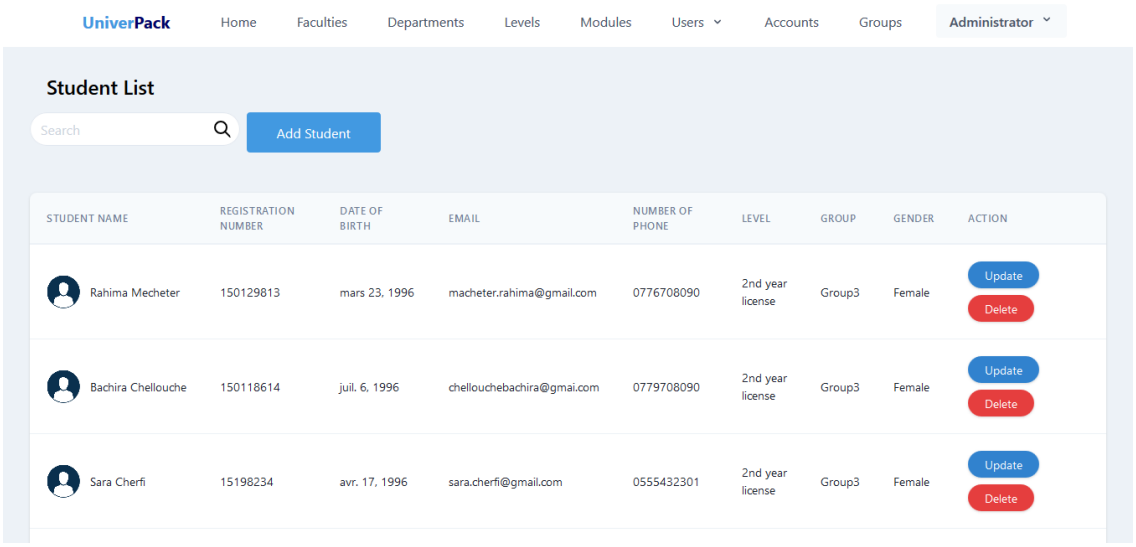


FIGURE 2.29 – IHM liste des étudiants

(b) **IHM Ajouter un niveau**
Pour ajouter un niveau, l'*administrateur* saisit les informations de ce niveau puis clique sur le bouton ajouter.

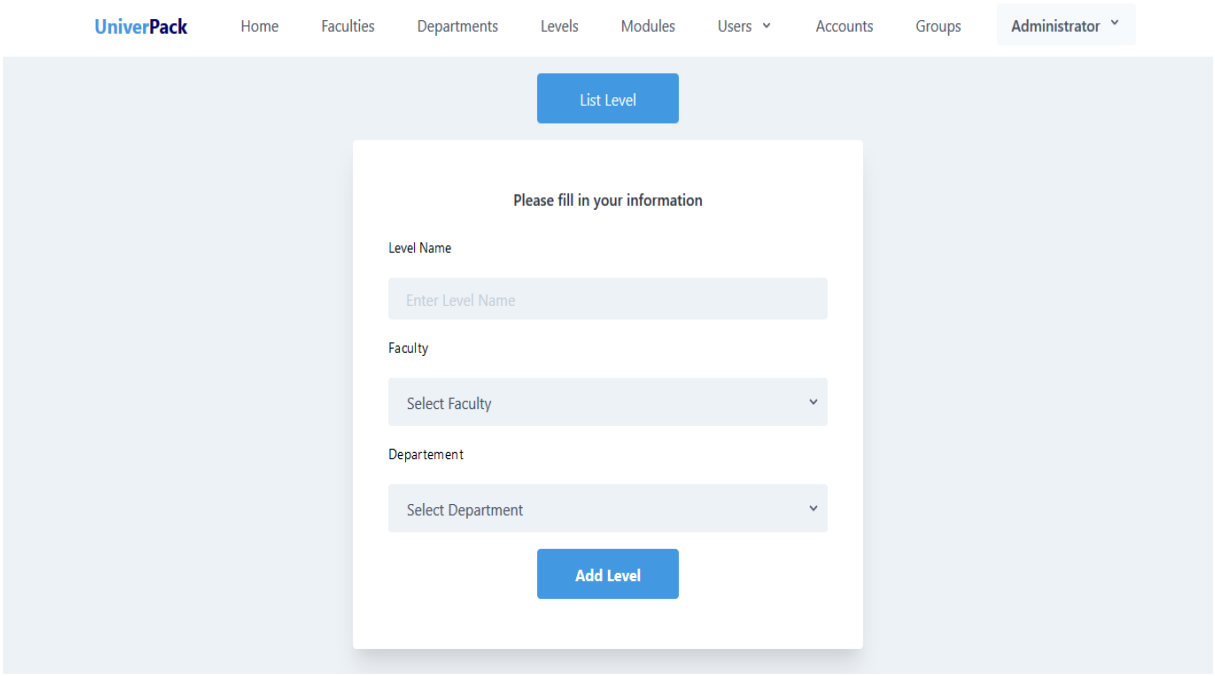


FIGURE 2.30 – IHM Ajouter un niveau

(c) **IHM modifier un département**
La figure 2.31 représente l'interface modifier département qui permet à l'*administrateur* de modifier les informations d'un département donné.

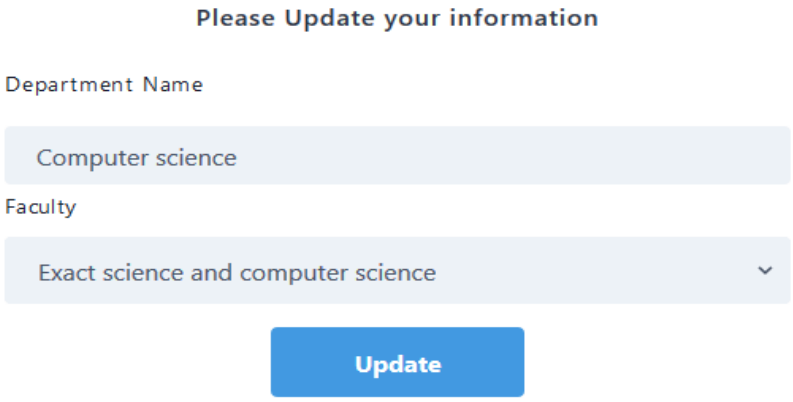


FIGURE 2.31 – IHM Modifier un département

2.5 Spécification détaillée

Nous allons maintenant décrire textuellement les UCs de façon détaillée par des fiches types (FT) et des diagrammes de séquence (DSS). L'objectif du diagramme de séquence est de représenter les interactions entre objets en indiquant la chronologie des échanges. Cette représentation peut se réaliser par cas d'utilisation en considérant les différents scénarios associés [28].

2.5.1 Fiches types et les diagrammes de séquence système

Nous choisissons les UCs les plus importants pour les détailler par des FT et les illustrer par les DSS.

1. Ft et DSS de l'UC *Authentification*

Cas d'utilisation	Authentification
Acteur principal	*Enseignant*, *Étudiant*, *Administrateur*
Objectifs	L'utilisateur et l'*administrateur* accèdent à leurs espace personnel.
Préconditions	
Postconditions	Ouverture de l'espace personnel
Scénario nominal	1 L'utilisateur demande de s'authentifier. 2 Le système affiche le formulaire d'authentification. 3 L'utilisateur saisit son nom d'utilisation et son mot de passe. 4 Le système vérifie la validité du compte. 5 Le système dirige l'utilisateur vers son espace personnel.
Alternative	4a. Les informations ne sont pas valides . 1 Le système affiche une notification d'échec. 2 L'utilisateur peut réessayer en revenant à l'étape 2 du scénario. nominal.

TABLE 2.1 – FT de l'UC *authentification*

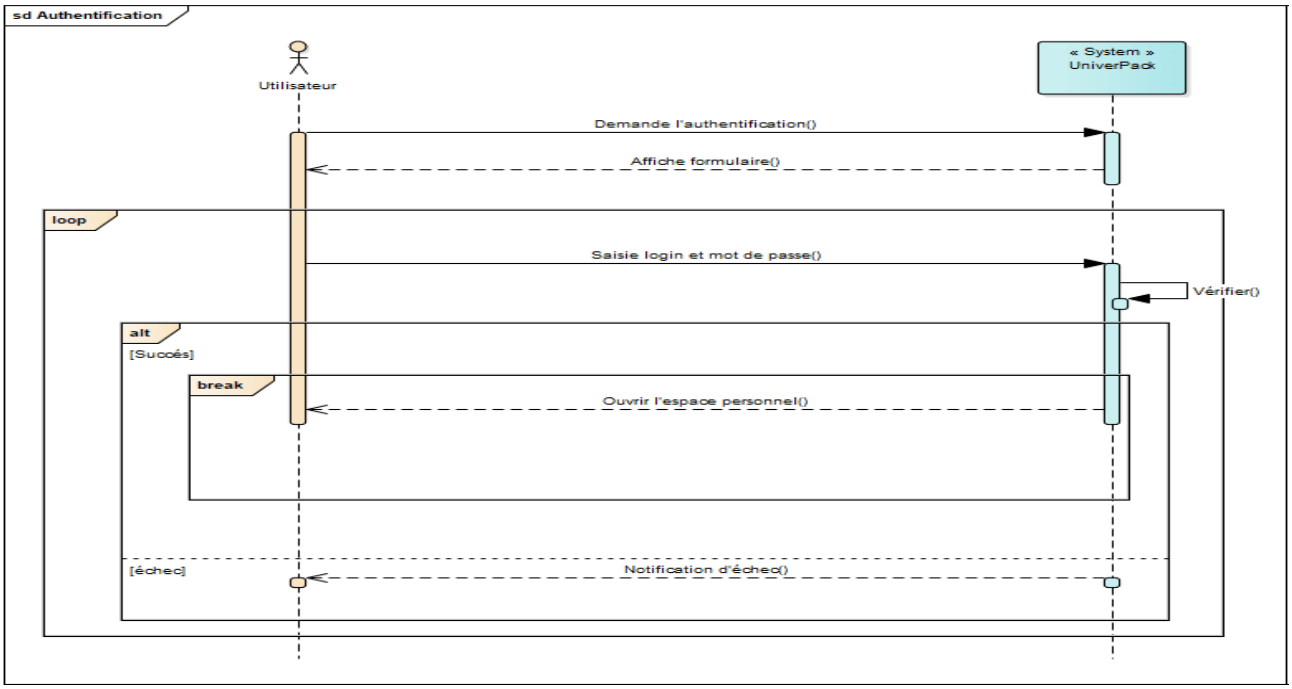


FIGURE 2.32 – DSS de l'UC *authentification*

2. FT et DSS de l'UC *Ajouter compte*

Cas d'utilisation	Ajouter compte
Acteur principal	*Administrateur*
Objectifs	L'*administrateur* ajoute un compte
Préconditions	L'*administrateur* est authentifié
Postconditions	Un nouveau compte est ajouté
Scénario nominal	1 L'*administrateur* demande l'ajout d'un compte. 2 Le système affiche le formulaire d'ajout. 3 L'*administrateur* remplit le formulaire et valide. 4 Le système vérifie les informations. 5 Le système enregistre les informations du compte. 6 Le système affiche une notification de sucées.
Alternative	4a. Les informations ne sont pas valides. 1 Le système affiche une notification d'échec. 2 L'administrateur reprend à partir de l'étape 2 du scénario nominal.

TABLE 2.2 – FT de l'UC *ajouter compte*

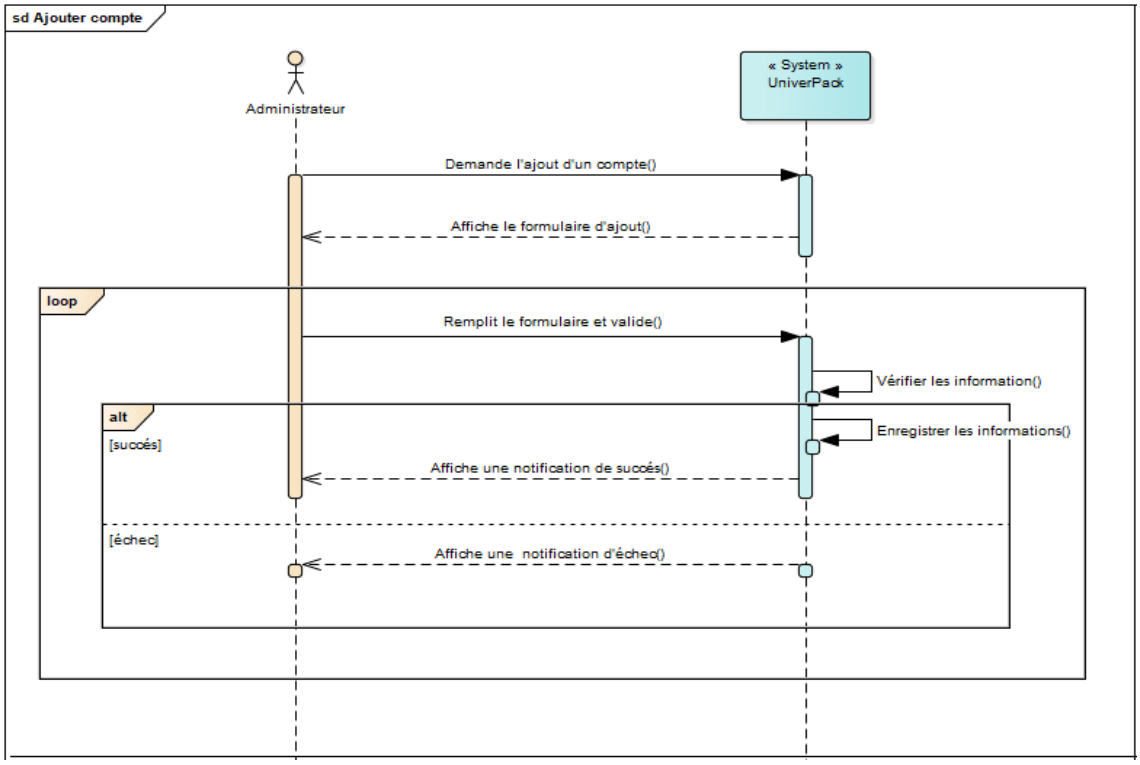


FIGURE 2.33 – DSS de l'UC *ajouter compte*

3. FT et DSS de l'UC *Supprimer compte*

Cas d'utilisation	Supprimer compte
Acteur principal	*Administrateur*
Objectifs	L'*administrateur* supprime un compte
Préconditions	L'*administrateur* est authentifié
Postconditions	Le compte est supprimé
Scénario nominal	1 L'*administrateur* demande la suppression d'un compte. 2 Le système affiche une barre de confirmation. 3 L'*administrateur* valide la suppression. 4 Le système confirme la suppression.
Alternative	

TABLE 2.3 – FT de l'UC *supprimer compte*

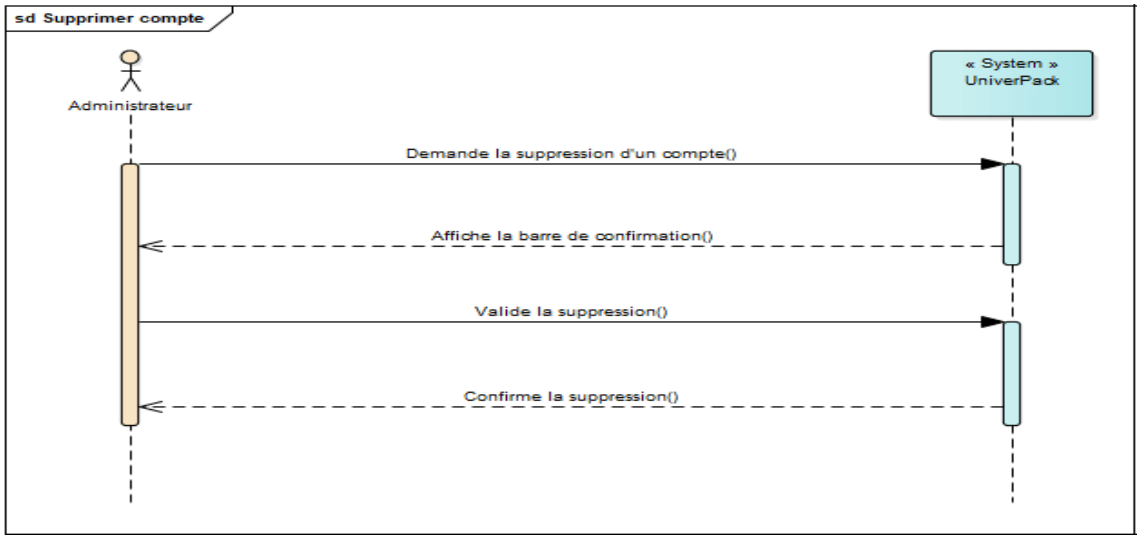


FIGURE 2.34 – DSS de l'UC *supprimer compte*

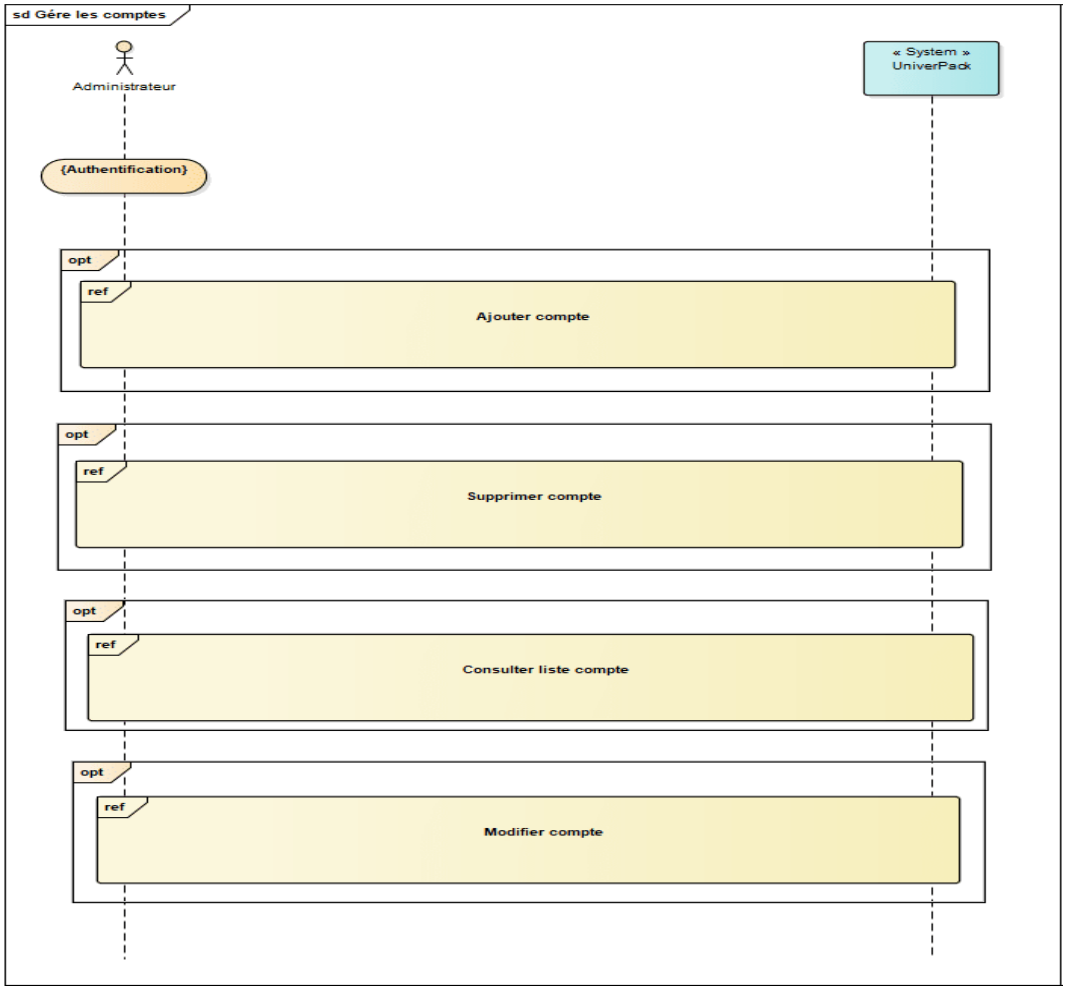


FIGURE 2.35 – DSS de l'UC *Gérer les comptes*

4. Ft et DSS de l'UC *Ajouter cours*

Cas d'utilisation	Ajouter cours
Acteur principal	*Enseignant*
Objectifs	L'*enseignant* ajoute un cours
Préconditions	L'*enseignant* est authentifié
Postconditions	un nouveau cour est ajouté
Scénario nominal	1 L'*enseignant* demande l'ajout d'un cours. 2 Le système affiche le formulaire d'ajout. 3 L'*enseignant* remplit le formulaire et valide. 4 Le système vérifie les informations. 5 Le système enregistre les informations. 6 Le système affiche une notification de succès.
Alternative	4a. Les informations ne sont pas valides. 1 Le système affiche une notification d'échec. 2 L'*enseignant* reprend à partir de l'étape 2 du scénario nominal.

TABLE 2.4 – FT de l'UC *ajouter cours*

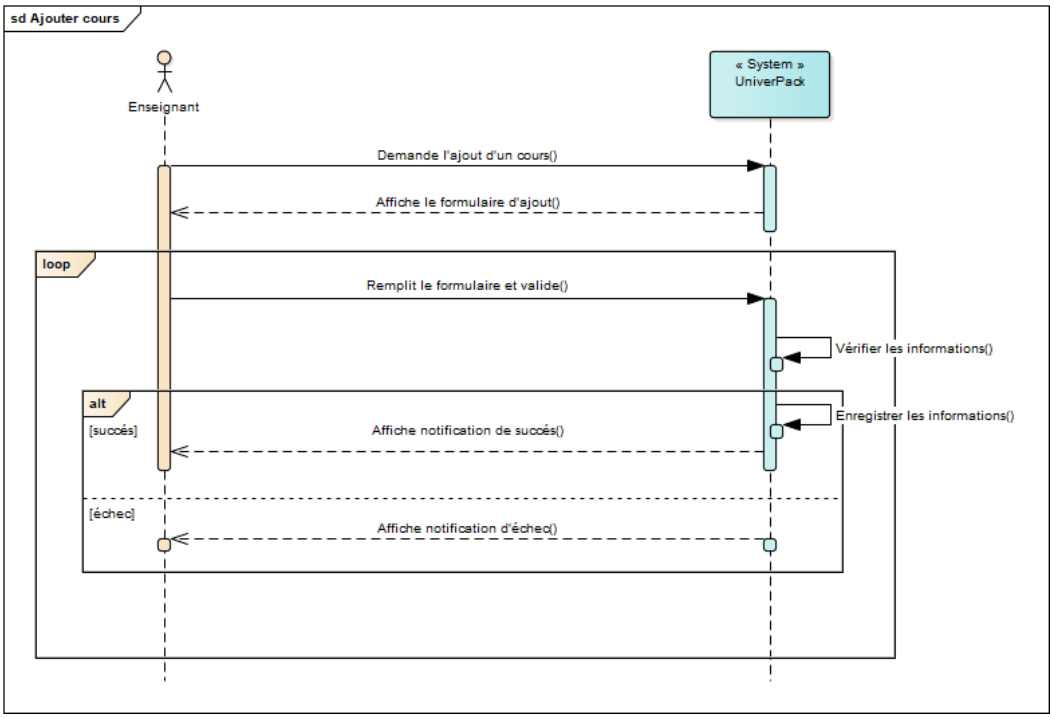


FIGURE 2.36 – DSS de l'UC *ajouter cours*

5. Ft et DSS de l'UC *Modifier cours*

Cas d'utilisation	Modifier cours
Acteur principal	*Enseignant*
Objectifs	L'*enseignant* effectue des modifications
Préconditions	L'*enseignant* est authentifié
Postconditions	
Scénario nominal	1 L'*enseignant* demande la modification d'un cours. 2 Le système affiche le formulaire des informations. 3 L'*enseignant* saisie les nouveaux informations et valide. 4 Le système vérifie les nouveaux informations. 5 Le système enregistre les nouveaux informations. 6 Le système affiche une notification de succès. 6 Le système ouvre la liste des cours.
Alternative	4a. Les informations ne sont pas valides . 1 Le système affiche une notification d'échec. 2 L'*enseignant* reprend à partir de l'étape 2 du scénario nominal.

TABLE 2.5 – FT de l'UC *modifier cours*

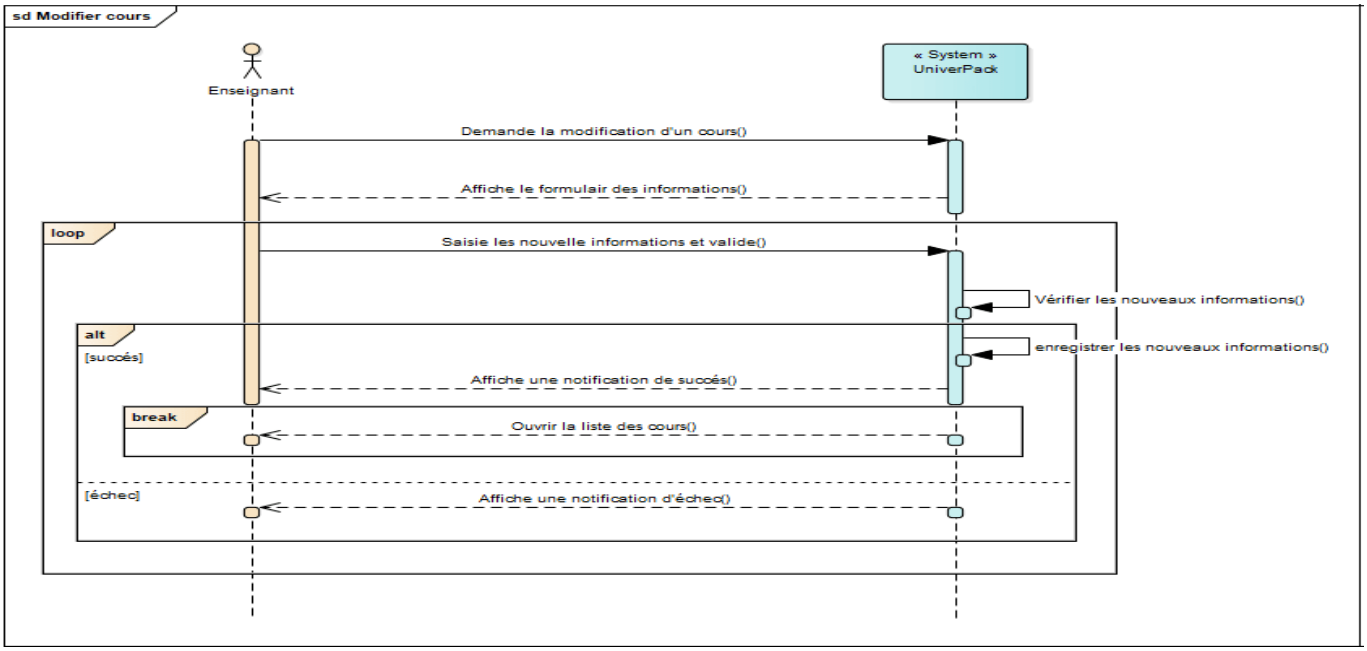


FIGURE 2.37 – DSS de l'UC *modifier cours*

6. Ft et DSS de l'UC *Supprimer cours*

Cas d'utilisation	Supprimer cours
Acteur principal	*Enseignant*
Objectifs	L'*enseignant* supprime un cours
Préconditions	L'*enseignant* est authentifié
Postconditions	Le cours est supprimé
Scénario nominal	1 L'*enseignant* demande la suppression d'un cours. 2 Le système affiche la barre de confirmation. 3 L'*enseignant* valide la suppression. 4 Le système confirme la suppression.
Alternative	

TABLE 2.6 – FT de l'UC *supprimer cours*

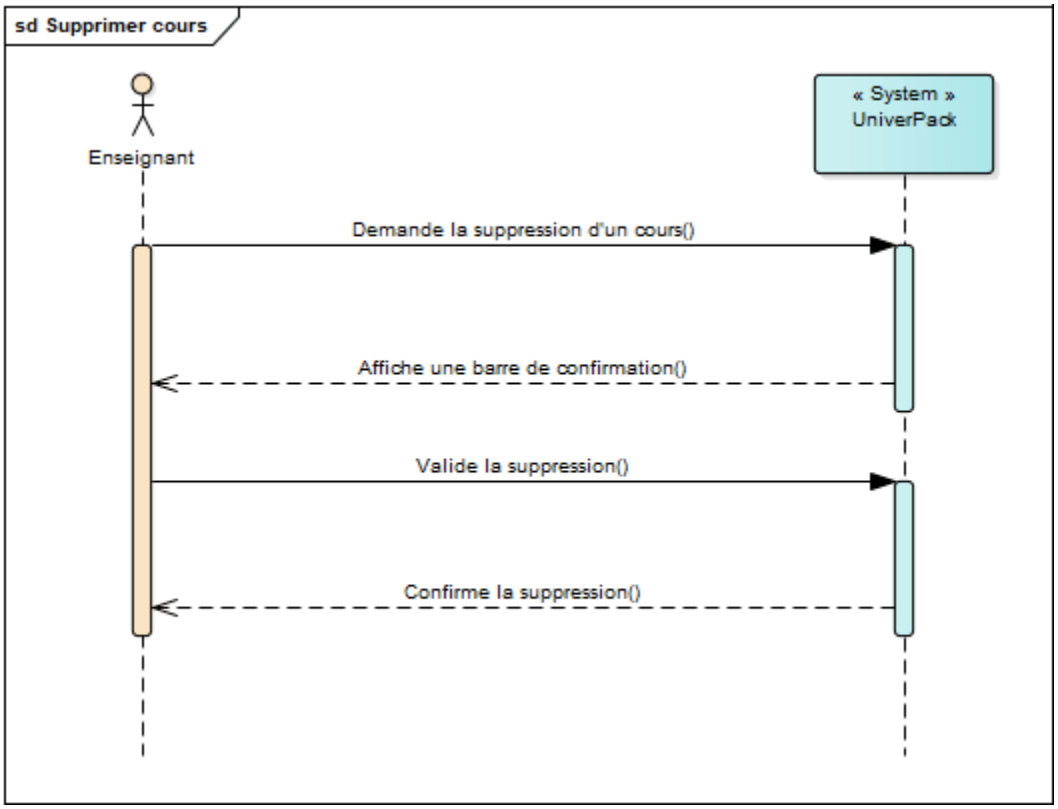


FIGURE 2.38 – DSS de l'UC *supprimer cours*

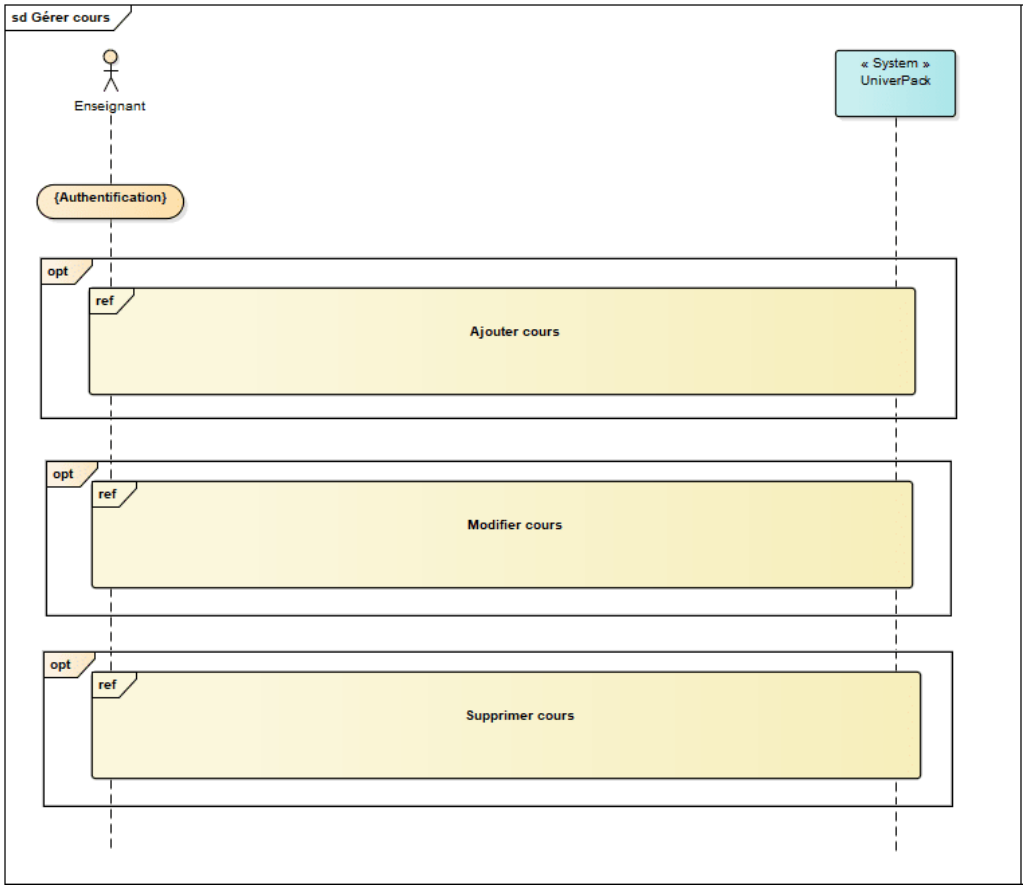


FIGURE 2.39 – DSS de l'UC *Gérer cours*

7. Ft et DSS de l'UC *Télécharger cours*

Cas d'utilisation	Télécharger cours
Acteur principal	*Étudiant*, *Enseignant*
Objectifs	L'utilisateur télécharge un cours
Préconditions	L'utilisateur est authentifié
Postconditions	Le cours est téléchargé
Scénario nominal	1 L'utilisateur sélectionne le cours à télécharger. 2 Le système affiche le titre et la description de cours. 3 L'utilisateur demande le téléchargement. 4 Le cours est téléchargé.
Alternative	

TABLE 2.7 – FT de l'UC *télécharger cours*

FIGURE 2.40 – DSS de l'UC *télécharger cours*

8. Ft et DSS de l'UC *Ajouter un sujet de discussion*

Cas d'utilisation	Ajouter un sujet de discussion
Acteur principal	*Enseignant*, *Étudiant*
Objectifs	L'utilisateur ajoute un sujet de discussion
Préconditions	L'utilisateur est authentifié
Postconditions	Le sujet est ajouté
Scénario nominal	1 L'utilisateur demande l'ajout d'un sujet. 2 Le système affiche le formulaire d'ajout. 3 L'utilisateur remplit le formulaire et valide. 4 Le système vérifie les informations. 5 Le système enregistre les informations. 6 Le système affiche la liste des sujets.
Alternative	4a. Les informations ne sont pas valides . 1 Le système affiche une notification d'échec. 2 L'utilisateur reprend à partir de l'étape 2 du scénario nominal.

TABLE 2.8 – FT de l'UC *ajouter un sujet de discussion*

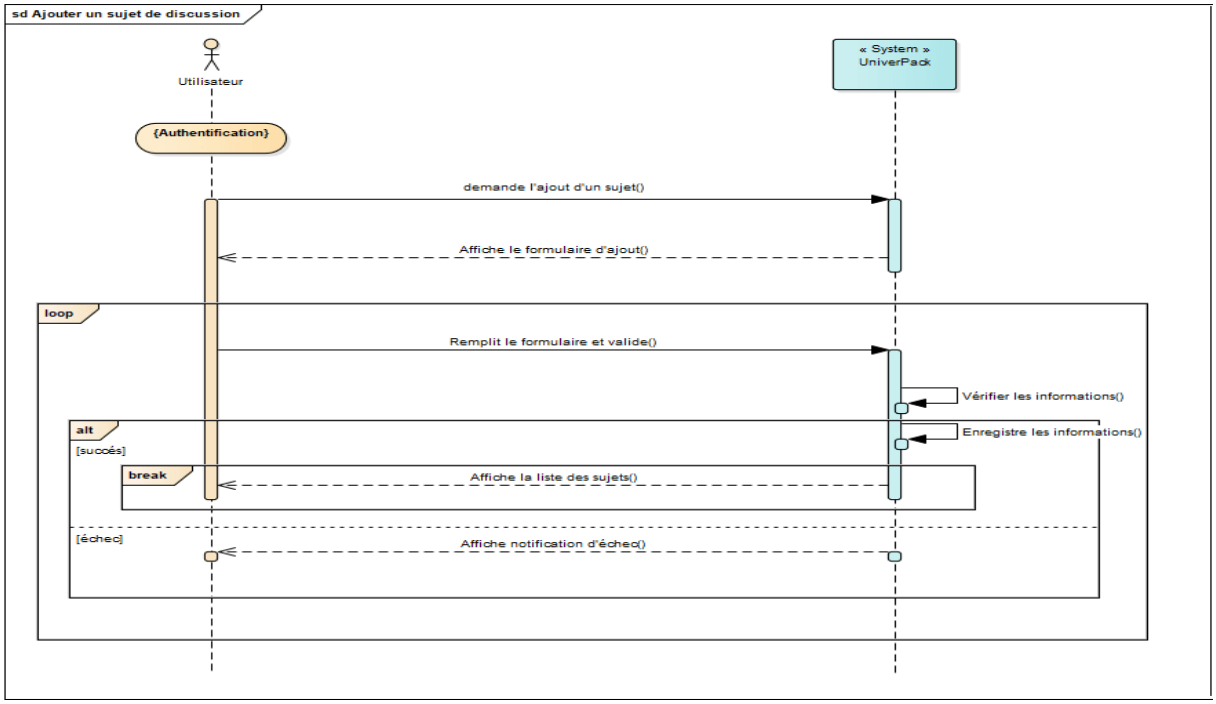


FIGURE 2.41 – DSS de l'UC *ajouter un sujet de discussion*

9. Ft et DSS de l'UC *Modifier profile*

Cas d'utilisation	Modifier profile
Acteur principal	*Enseignant*, *Étudiant*
Objectifs	L'utilisateur modifie son profile
Préconditions	L'utilisateur est authentifié
Postconditions	Le profile est modifié
Scénario nominal	1 L'utilisateur demande la modification de profile. 2 Le système affiche les informations. 3 L'utilisateur saisie les nouveaux informations et valide. 4 Le système vérifie les nouveaux informations. 5 Le système enregistre les nouveaux informations. 6 Le système affiche une notification de succès. 6 Le système affiche la page profile.
Alternative	4a. Les informations ne sont pas valides. 1 Le système affiche une notification d'échec. 2 L'utilisateur reprend à partir de l'étape 2 du scénario nominal.

TABLE 2.9 – FT de l'UC *modifier profile*

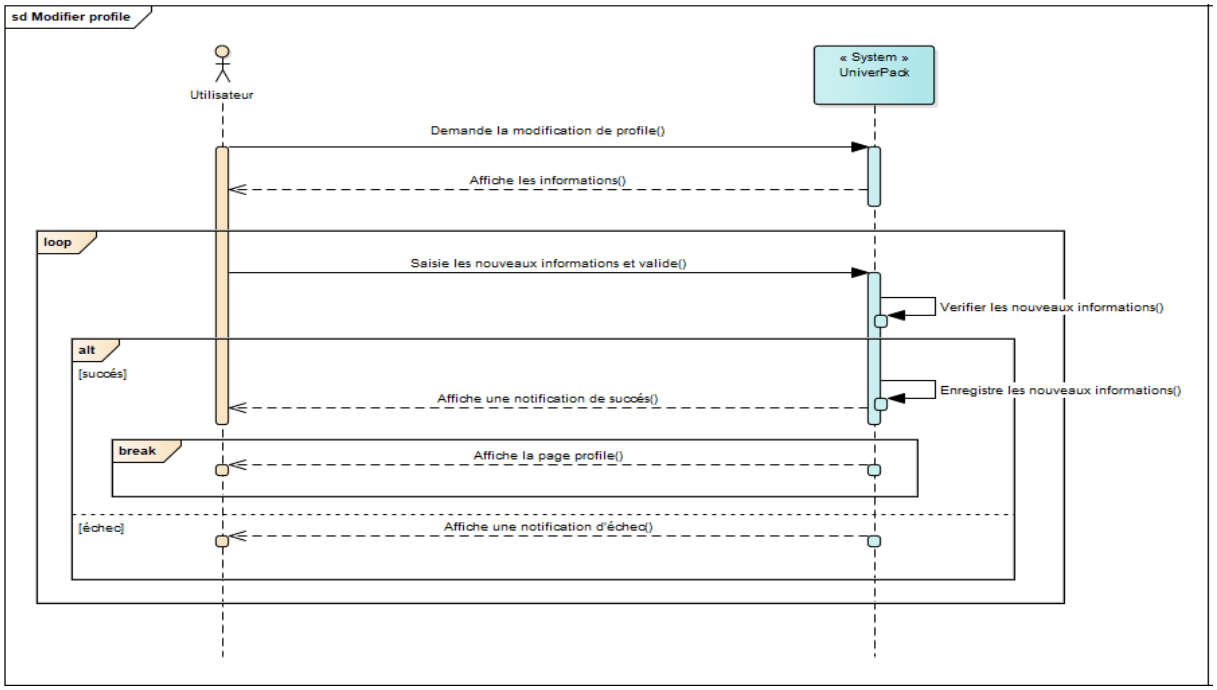


FIGURE 2.42 – DSS de l'UC *modifier profile*

2.6 Conclusion

La phase de spécification des besoins nous a permis de décrire les fonctionnalités de notre plateforme. Nous avons exprimé ces fonctionnalités sous forme des diagrammes de cas d'utilisation UML (UCs) avec leurs IHMs en suivant le processus simplifié, et enfin nous avons décrit ces cas d'utilisation de façon détaillée par les fiches types et les diagrammes de séquence (DSS).

Chapitre 3

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*.

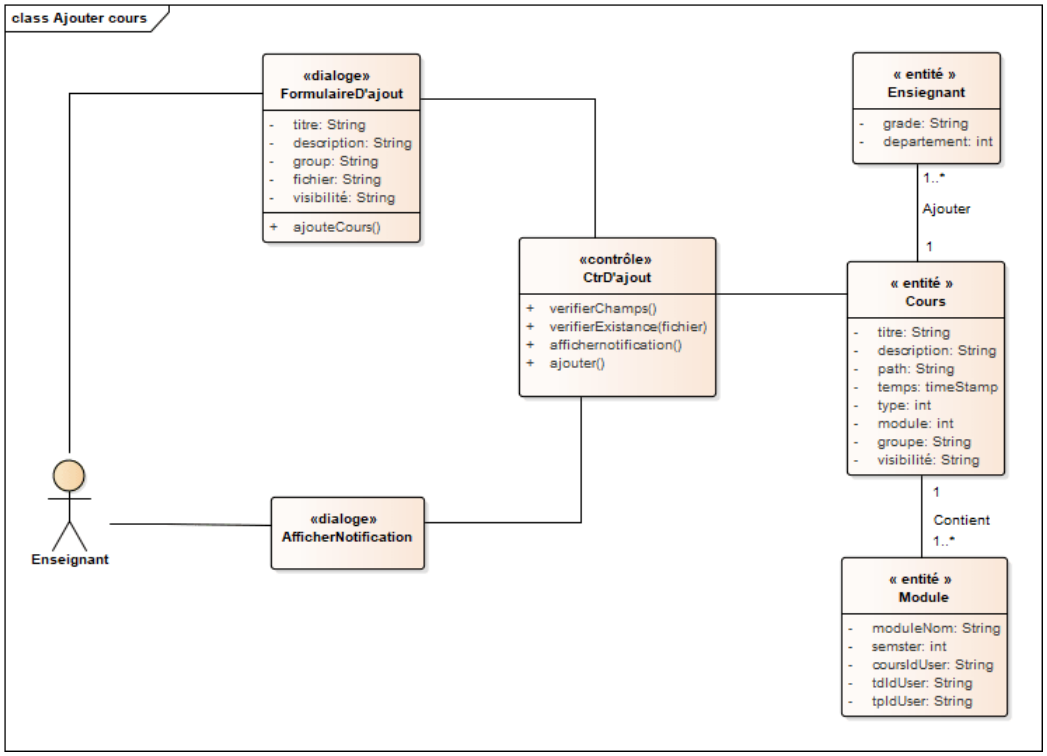


FIGURE 3.1 – DCP de l'UC *Ajouter cours*

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*.

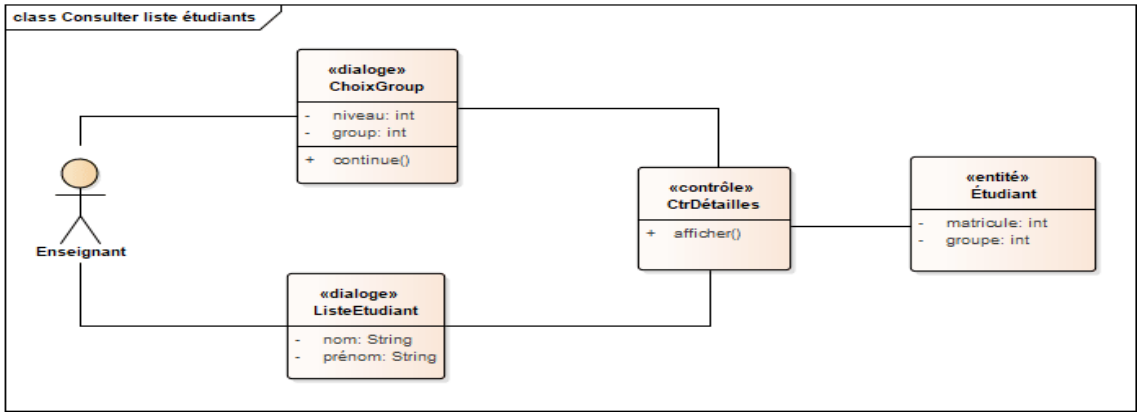


FIGURE 3.2 – DCP de l'UC *Consulter liste étudiants*

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*.

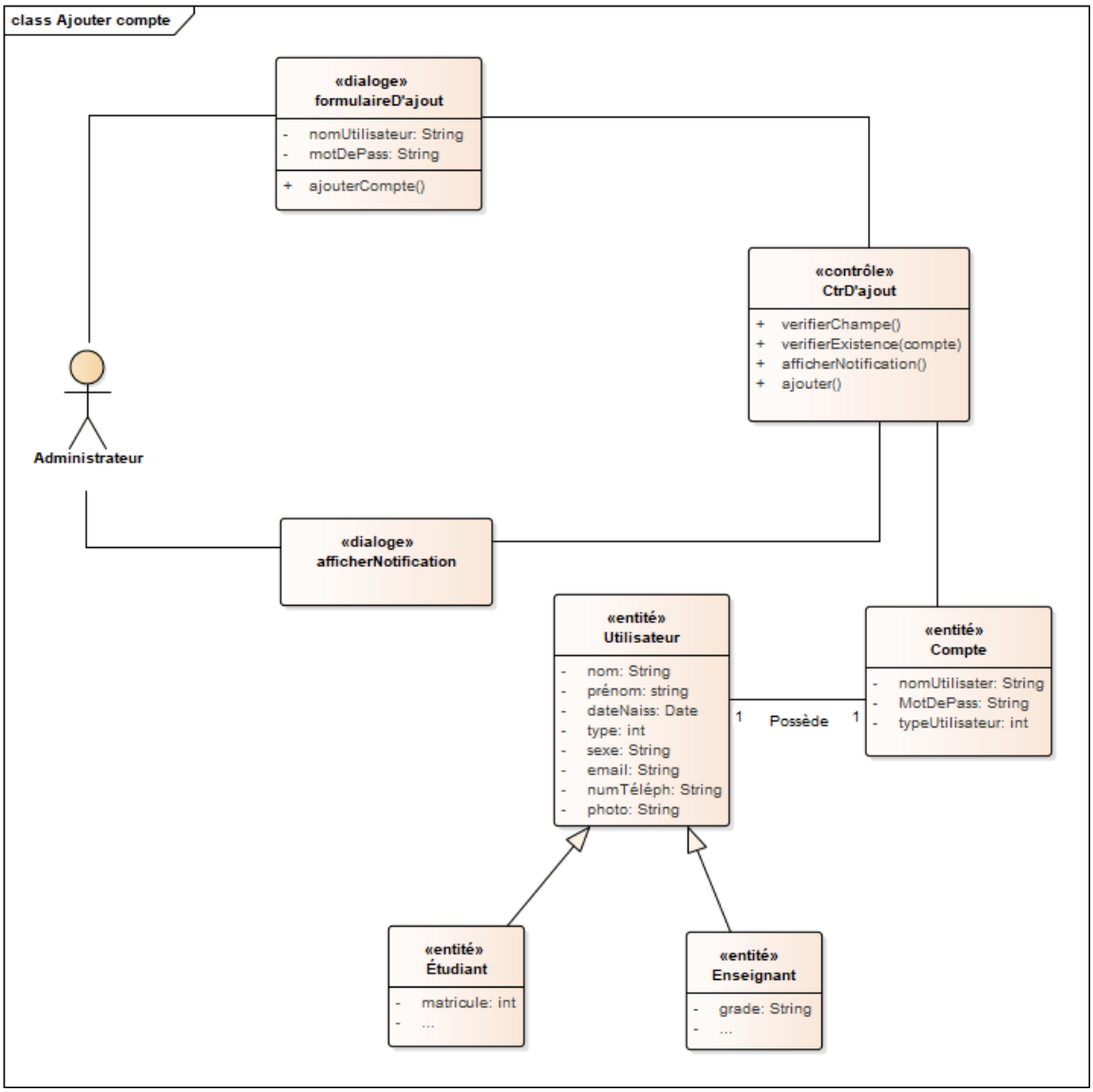


FIGURE 3.3 – DCP de l'UC *Ajouter compte*

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*.

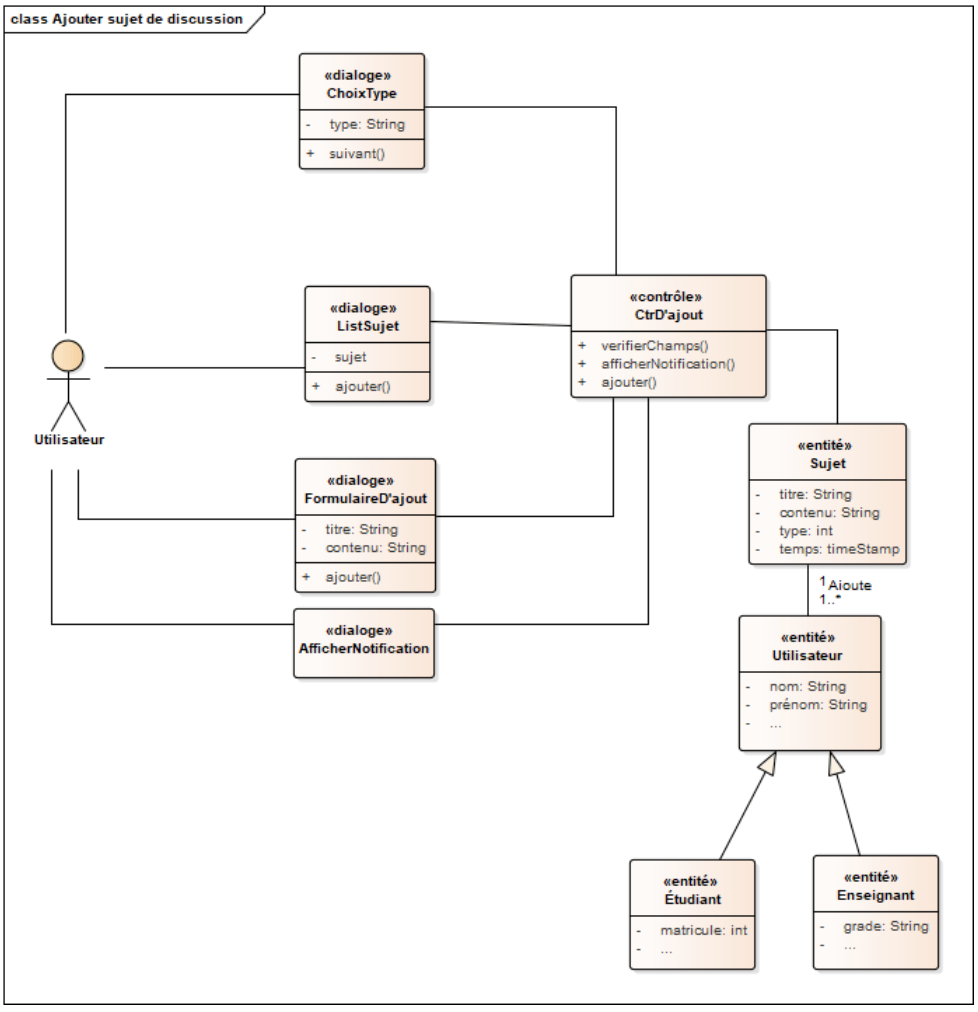


FIGURE 3.4 – DCP de l'UC *Ajouter sujet de discussion*

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*.

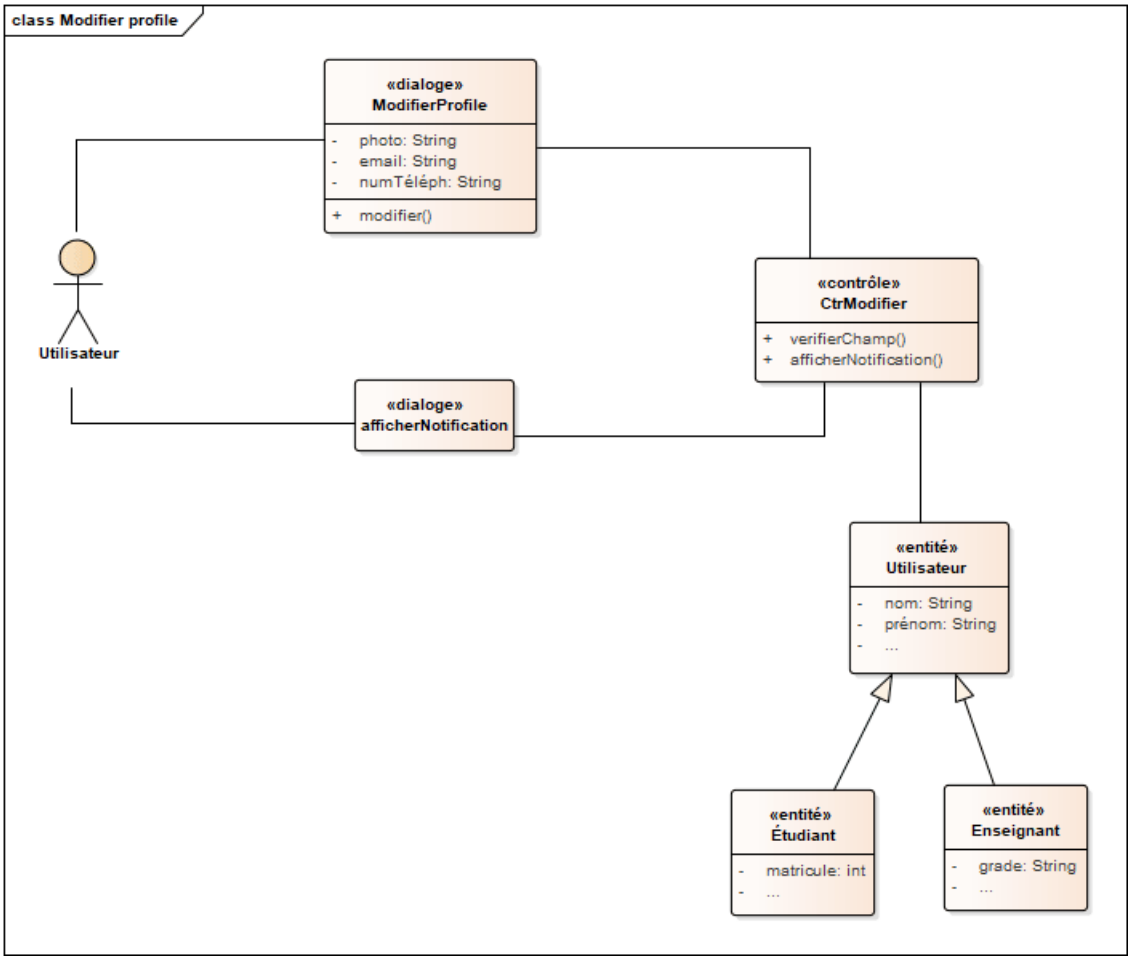


FIGURE 3.5 – DCP de l'UC *Modifier profile*

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.

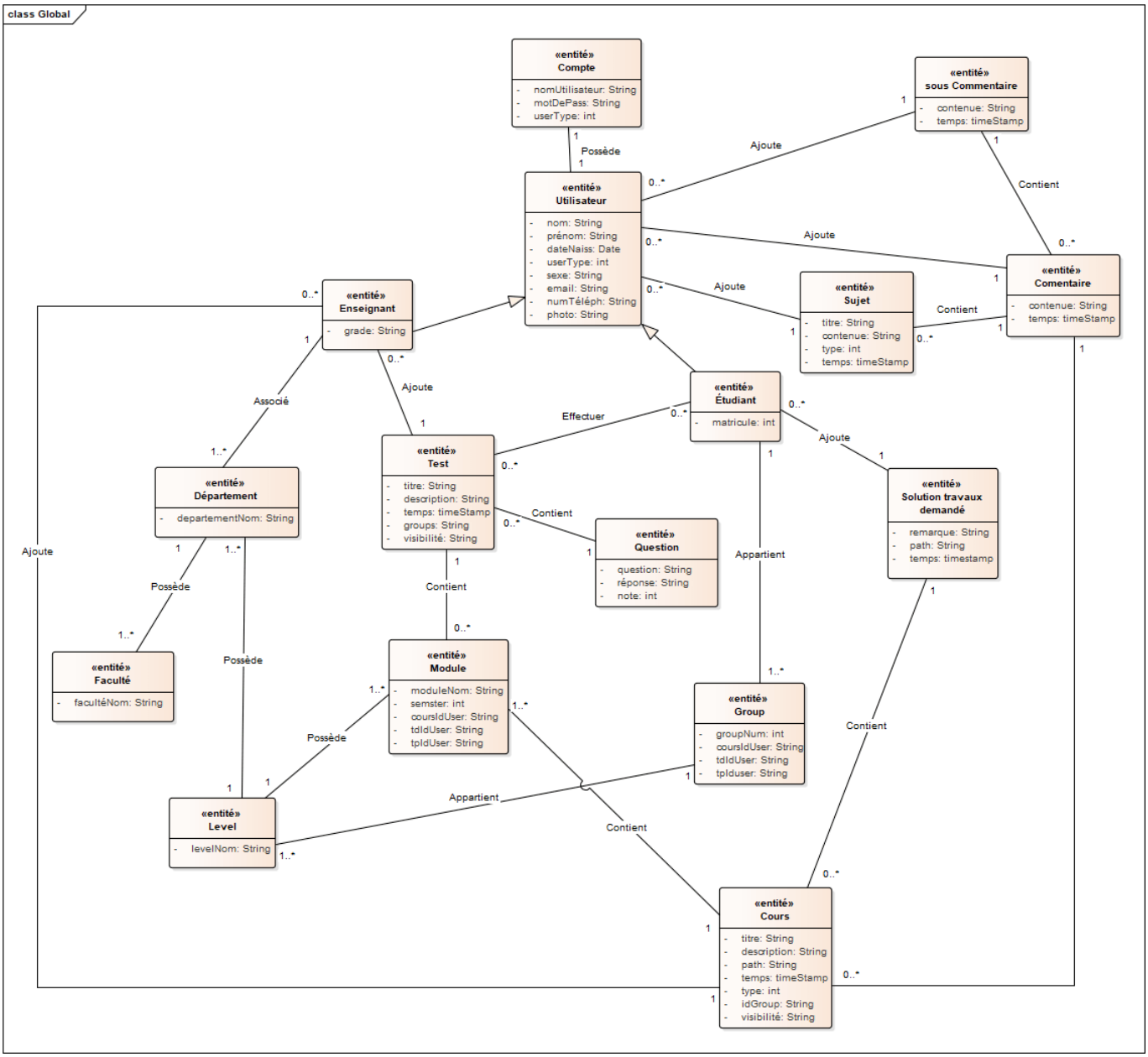


FIGURE 3.6 – Diagramme de classe global

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.

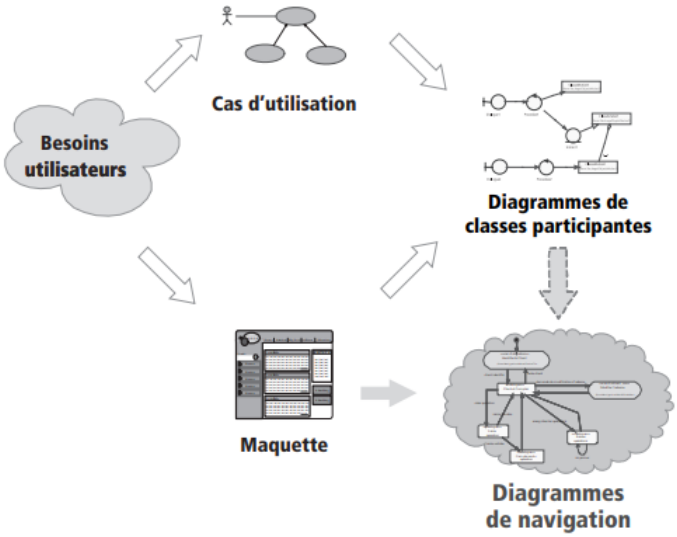
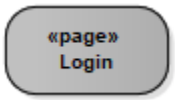


FIGURE 3.7 – Les diagrammes de navigation dans la démarche [24]

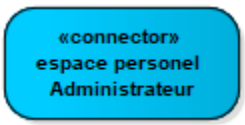
3.4.1 Diagrammes de navigation de notre système

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.

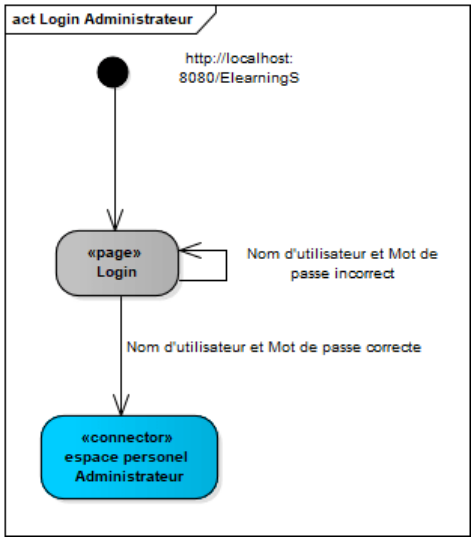


FIGURE 3.8 – Début de diagramme de navigation d'*administrateur*

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* :

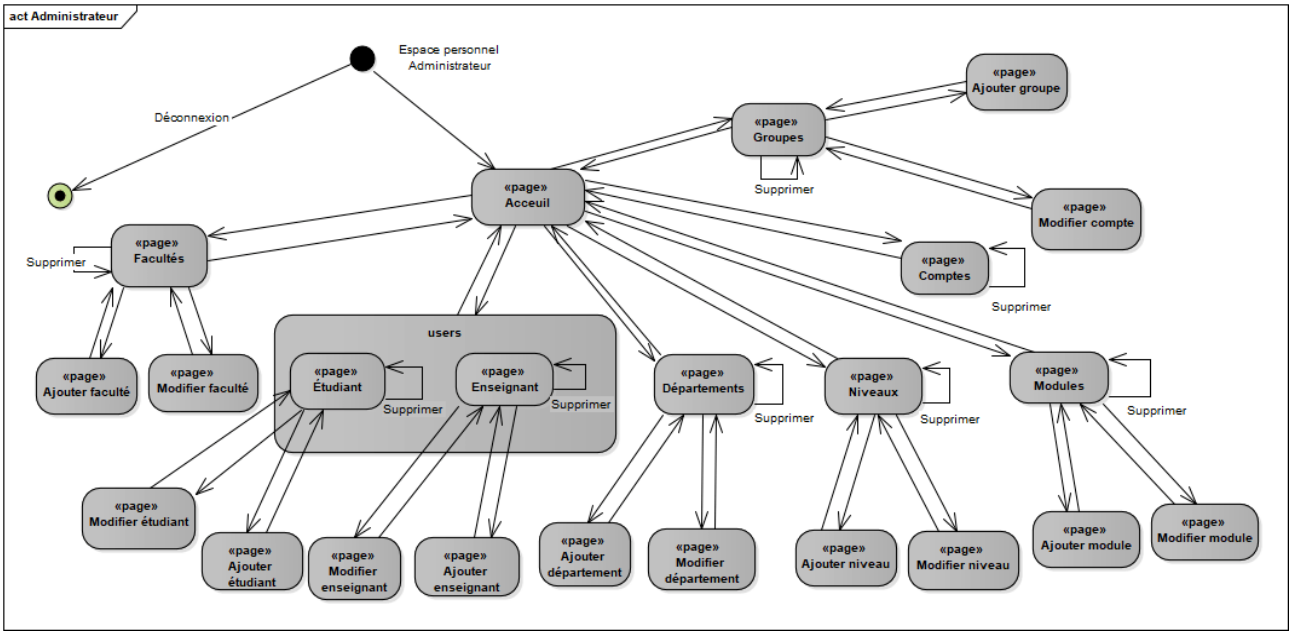


FIGURE 3.9 – Diagramme de navigation de l'*administrateur*

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.

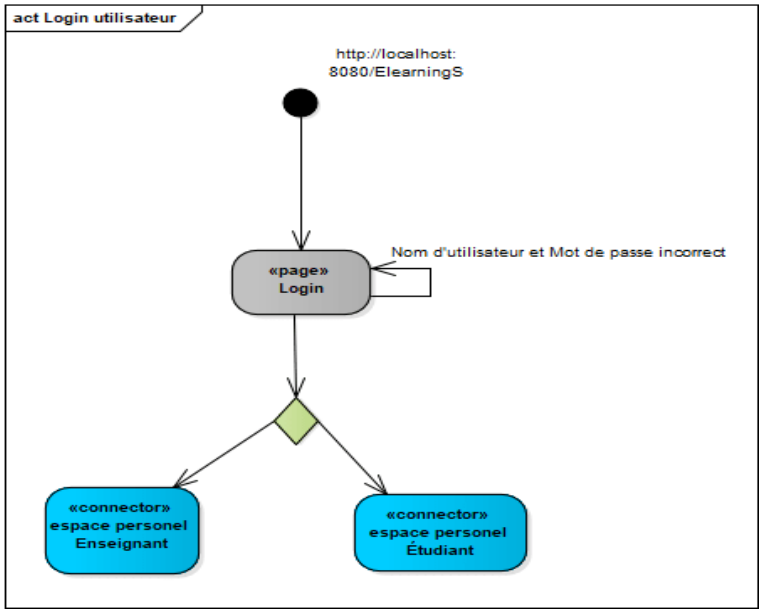


FIGURE 3.10 – Début de diagramme de navigation d'utilisateur

(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.

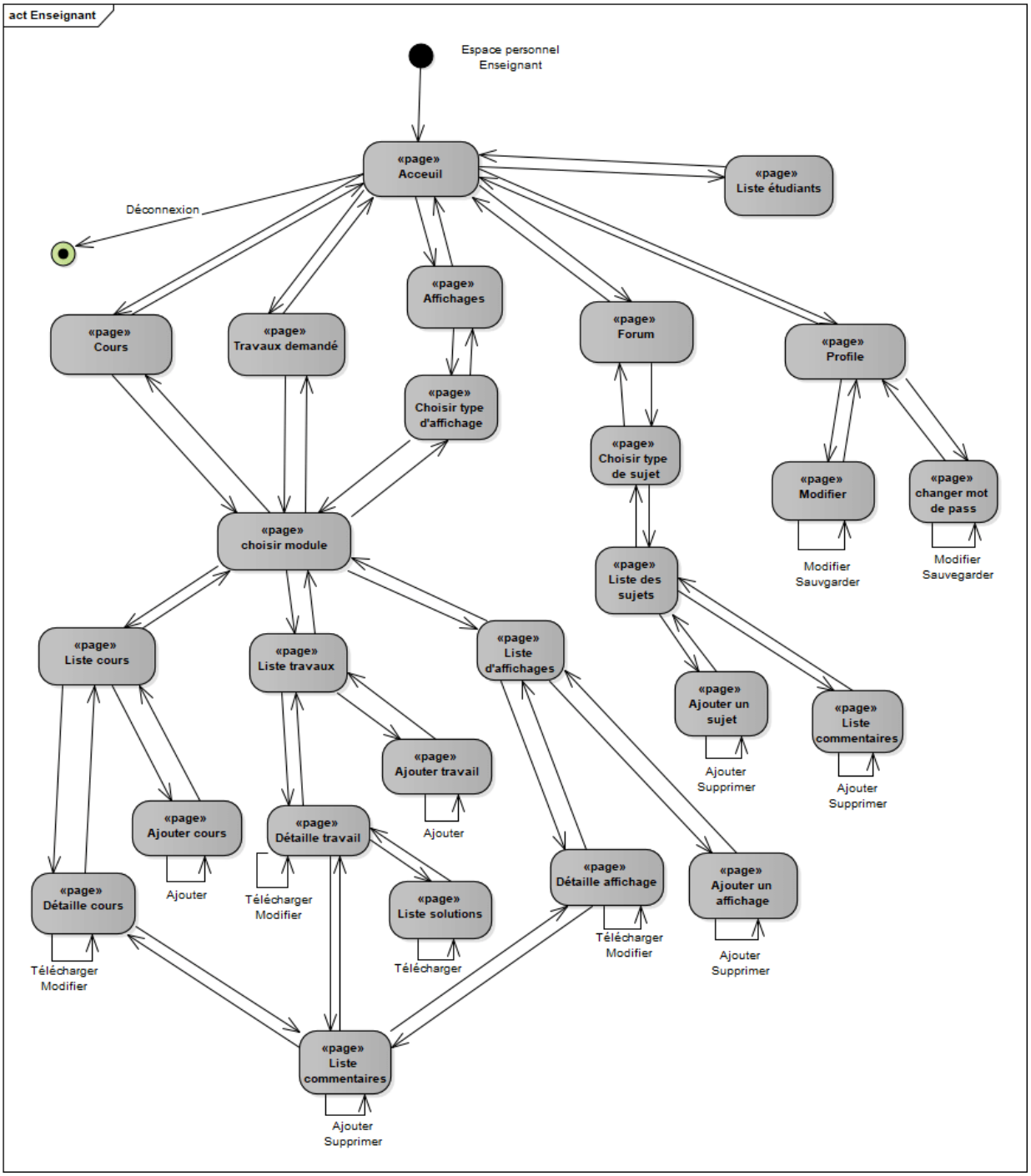


FIGURE 3.11 – Diagramme de navigation de l'*enseignant*

(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.

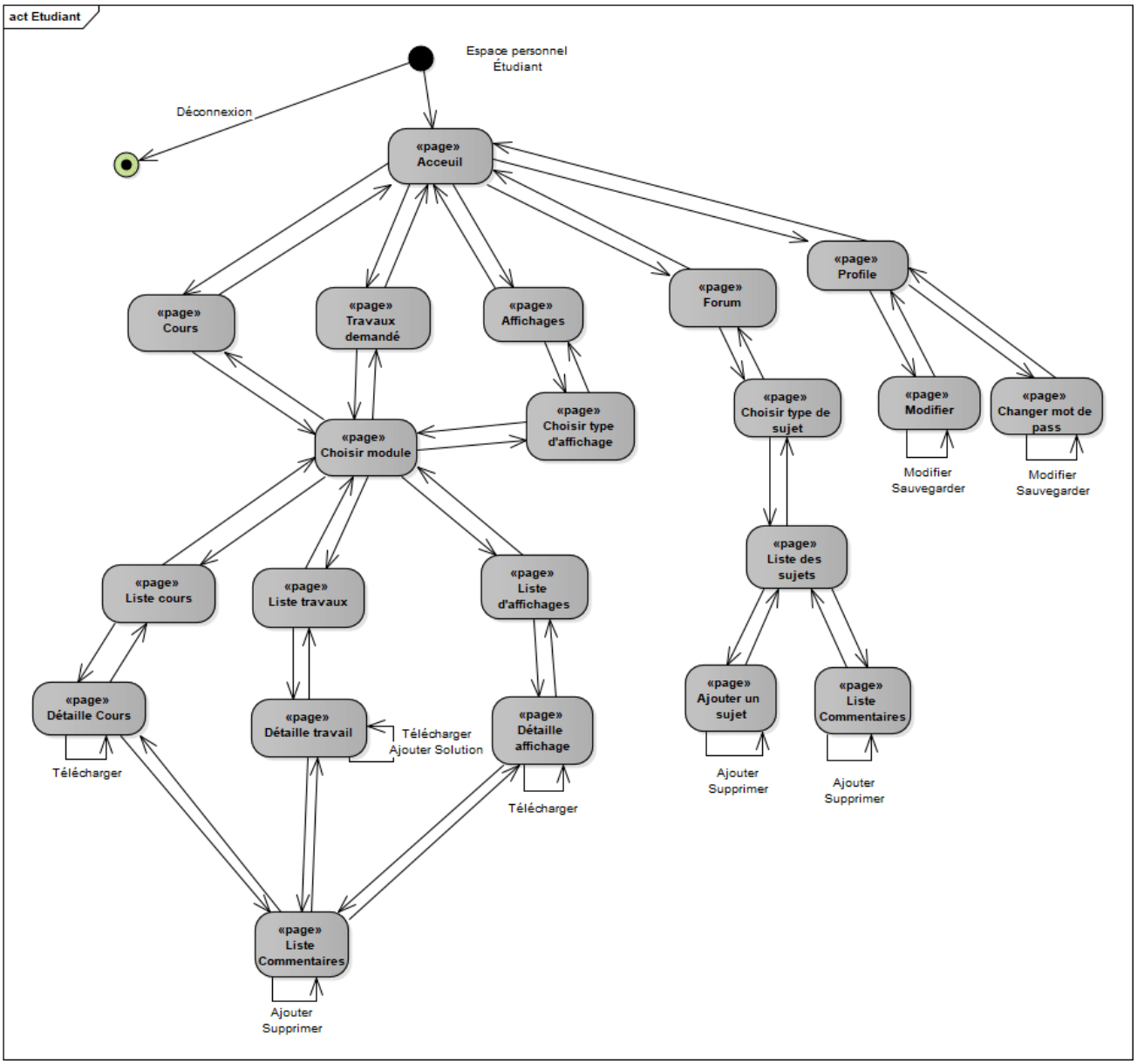


FIGURE 3.12 – Diagramme de navigation de l'*étudiant*

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 :

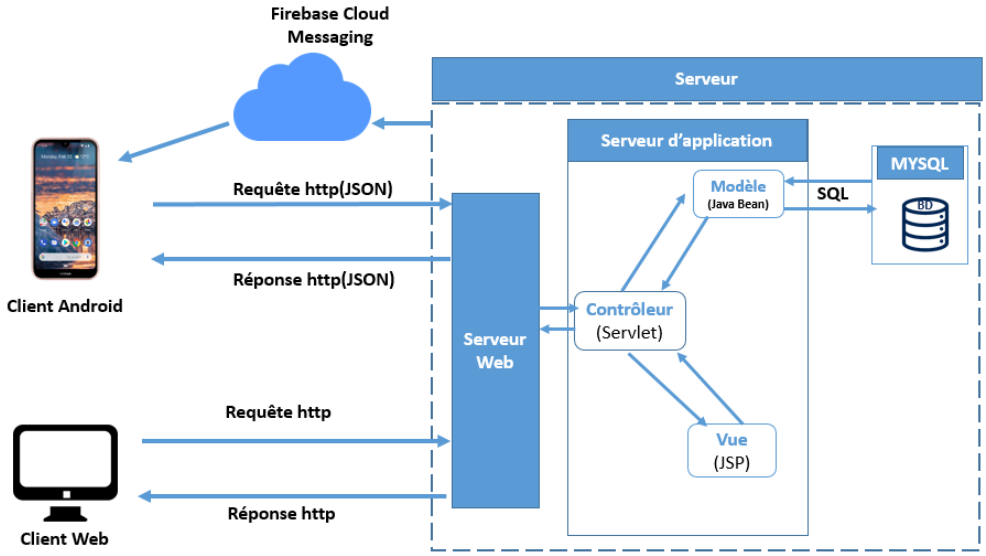


FIGURE 4.1 – Architecture global de notre application

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.

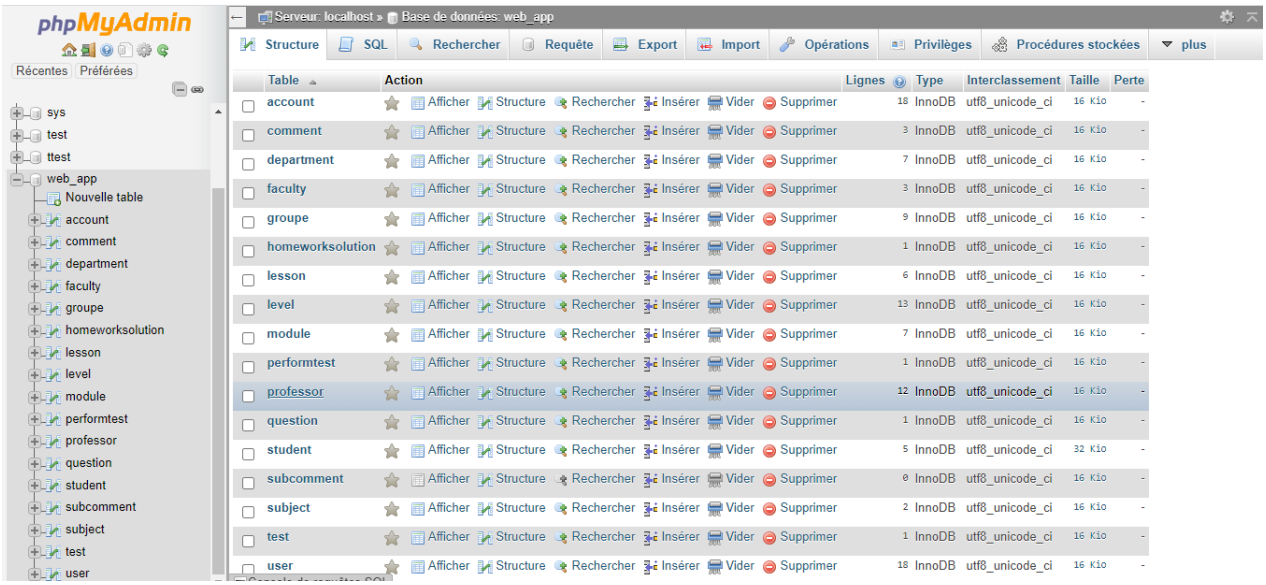


FIGURE 4.2 – Tables de la base de données

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.

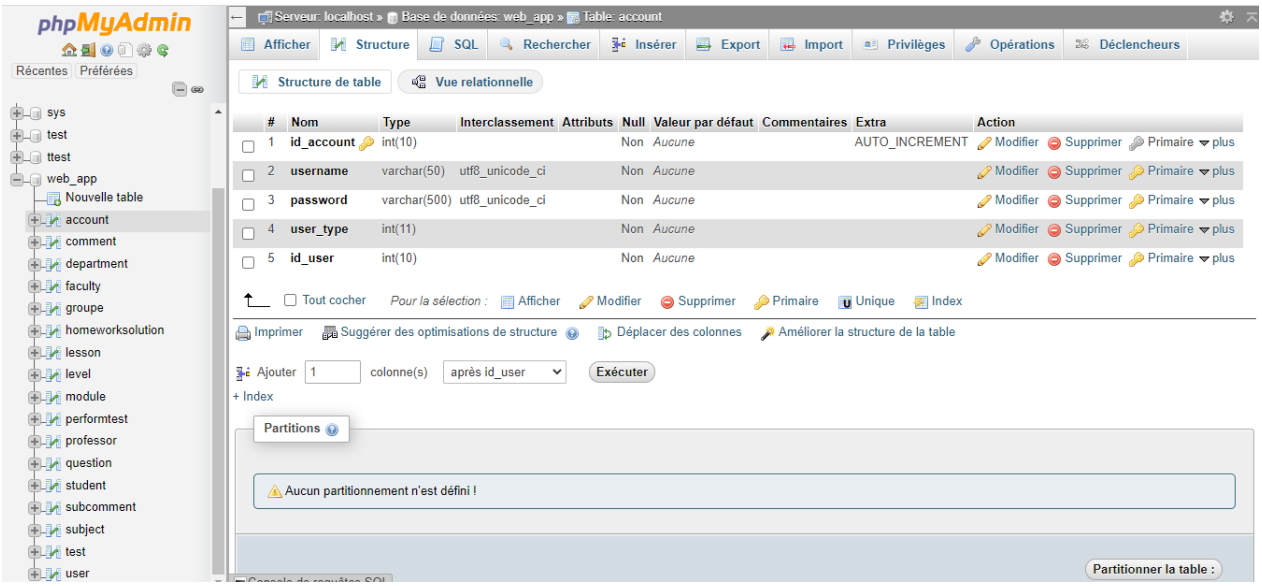


FIGURE 4.3 – Table compte

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

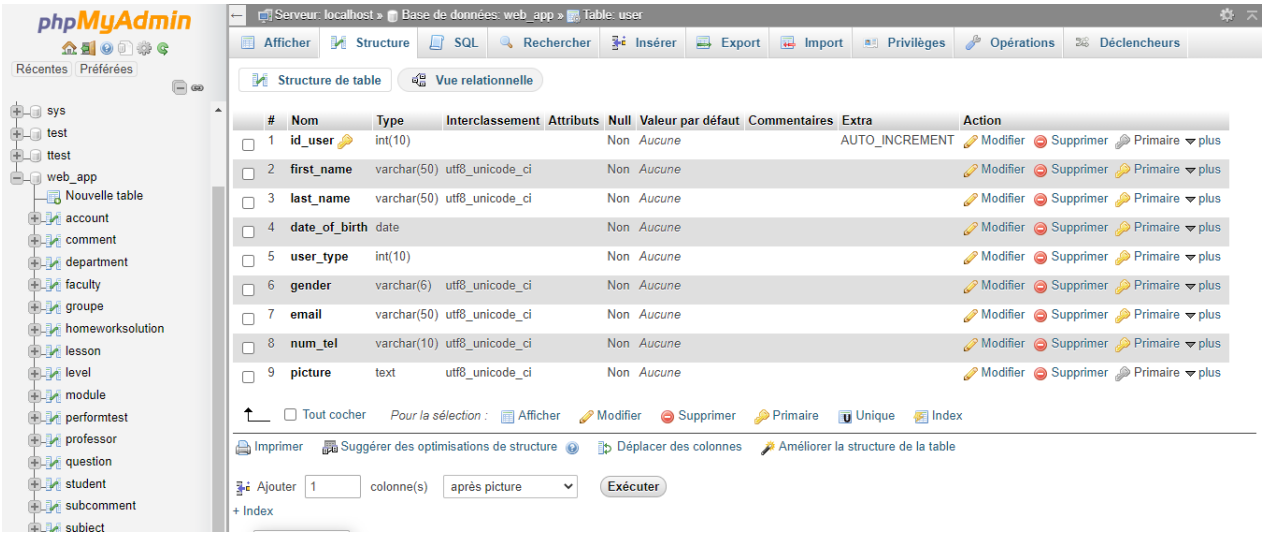


FIGURE 4.4 – Table utilisateur

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

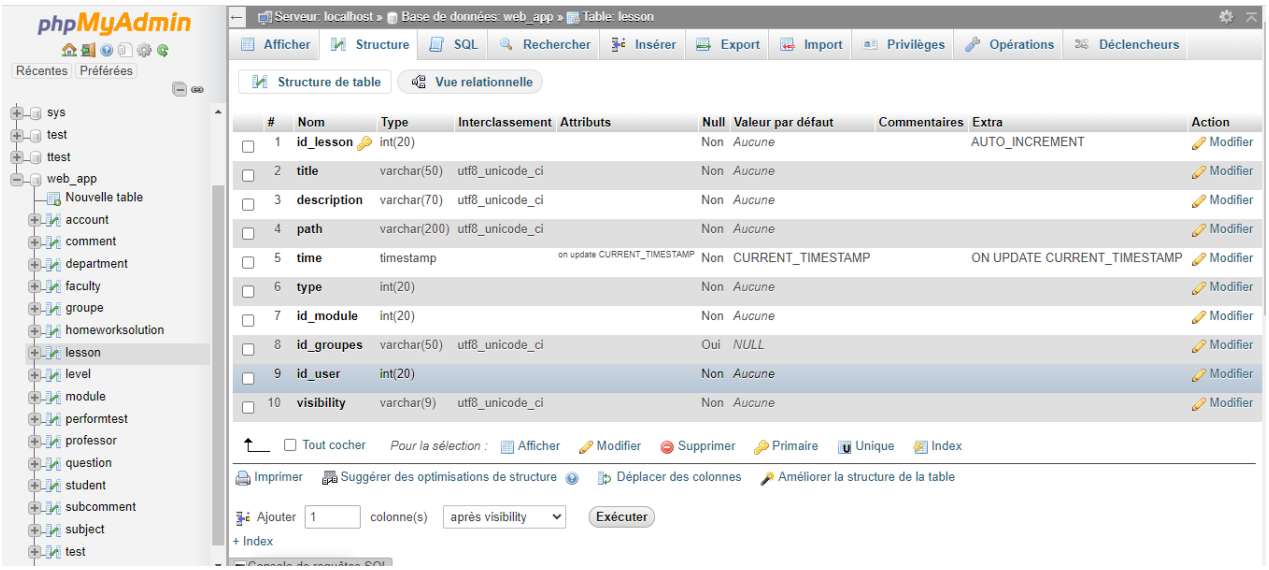


FIGURE 4.5 – Table cours

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/.