

Ecole supérieur de technologie Meknès
Université Moulay Ismail

Filière : Génie Informatique (GL)
Rapport de Projet de fin
d'étude

Mise en œuvre d'une plateforme de contrôle de tricherie dans un examen en ligne



2016/2017

Filière : Encadrés par : Réaliser par :

Option :

Remerciements

Nous tenons à remercier dans un premier temps, toute l'équipe pédagogique de l'Ecole Supérieur de Technologie de Meknès (ESTM) et les intervenants professionnels responsables de la formation informatique.

Nos remerciements s'adressent à Monsieur LAHMER notre encadrant de projet fin d'études, pour sa confiance et ses conseils qui nous ont permis de progresser sans cesse durant la période de préparation du projet.

Enfin, un grand remerciement à l'ensemble des personnes qui ont participé de près ou de loin à l'élaboration de ce travail.

Abréviations

Désignation

HTML

Hypertext Markup Language

JSP

Java Server Page

RMI

Remote Methode Invocation

API

Application Programming Interface

JMX

Java Management Extension

XML

Extensible Markup Language

SGBDR

**Système de Gestion de Base de Données
Relationel**

CSS

Cascading Style Sheets

Liste des abréviations

Sommaire

| | |
|---|-----------|
| LISTE DES ABREVIATIONS | 2 |
| INTRODUCTION..... | 5 |
| CHAPITRE I : ANALYSE ET SPECIFICATION DES BESOINS | 7 |
| 1. CAHIER DE CHARGES | 7 |
| 1. PRESENTATION DE PROJET : | 7 |
| 2. LES OBJECTIFS DE PROJET : | 8 |
| 3. SPECIFICATION DES BESOINS : | 8 |
| 4. PRESENTATION DES ACTEURS : | 9 |
| 5. PROCESSUS DE DEVELOPPEMENT : | 9 |
| 1.5.1 LE MODELE DE CYCLE DE VIE | 9 |
| 5. APRES AVOIR ACCEPTE L'ETUDIANT LES CONTRAINTES LE SYSTEM VERS LA PAGE D'EXAMEN. | 15 |
| CHAPITRE III : ETUDE TECHNIQUE | 31 |

| | | |
|-----------|--|-----------|
| 2. | <u>SERVEUR UTILISE :</u> | 31 |
| 1. | MYSQL : | 31 |
| 2. | APACHE TOMCAT : | 31 |
| 3. | <u>OUTILS DE CONCEPTIONS :</u> | 32 |
| 1. | ENTREPRISE ARCHITECT : | 32 |
| 2. | XAMPP : | 32 |
| 4. | <u>LANGAGES :</u> | 33 |
| 1. | JAVA : | 33 |
| 2. | JAVA EE : | 33 |
| 3. | CSS : | 34 |
| 4. | HTML : | 34 |
| 5. | JAVA SCRIPT: | 35 |
| 5. | <u>OUTILS DE DEVELOPPEMENT:</u> | 35 |
| 1. | ECLIPSE IDE FOR JAVA EE DEVELOPERS: | 35 |
| 6. | <u>OUTILS DE DESIGN:</u> | 36 |
| 1. | ADOBE PHOTOSHOP CC : | 36 |
| 7. | <u>API:</u> | 36 |
| 1. | JMX : | 36 |
| 2. | WEBSOCKET : | 37 |
| | <u>CHAPITRE IV : PRESENTATION D'APPLICATION :</u> | 38 |
| 1. | INDEX : | 38 |
| 2. | L'AUTHENTIFICATION : | 39 |
| 3. | LE COMPTE ETUDIANT : | 39 |
| 4. | PAGE D'ERREUR : | 40 |
| 5. | CONSTRAINT : | 41 |
| 6. | PAGE D'EXAMEN : | 41 |
| 7. | PAGE DE NOTE : | 42 |

| | | |
|-----|-------------------------------------|----|
| 8. | LE COMPTE PROFESSEUR : | 42 |
| 9. | L'AJOUT D'UN EXAMEN..... | 43 |
| 10. | LA MODIFICATION D'UN EXAMEN..... | 44 |
| 11. | LA LISTE DES QUESTIONS : | 44 |
| 12. | L'AJOUT D'UNE QUESTION..... | 45 |
| 13. | LA MODIFICATION D'UNE QUESTION..... | 45 |

Table des figures

| | |
|--|-----------------------------|
| Figure 1 Méthode Projet du cycle en vie | 9 |
| Figure 2 MCD | 11 |
| Figure 3 : diagramme uml | 12 |
| Figure 4 diagramme etudiant | 14 |
| Figure 5 diagramme professeur | 16 |
| Figure 6 diagramme de classe..... | Erreur ! Signet non défini. |
| Figure 7 diagramme classe dao..... | 18 |
| Figure 8 diagramme classe controleur..... | 20 |
| Figure 9 diagramme authentification | 22 |
| Figure 10 diagramme d'examen | 23 |
| Figure 11Design pattern MVC..... | 27 |
| Figure 12 l'architecture de l'API JMX | 28 |
| Figure 13 l'architecture de projet contrôle de tricherie | 29 |
| Figure 14 Le fonctionnement de RMI | 30 |
| Figure 15 mysql..... | 31 |
| Figure 16 tomcat..... | 31 |
| Figure 17 UML..... | 32 |
| Figure 18 XAMP..... | 32 |
| Figure 19 JAVA | 33 |
| Figure 20 JEE | 33 |

| | |
|--|----|
| Figure 21 CSS..... | 34 |
| Figure 22 HTML..... | 34 |
| Figure 23 JAVA SCRIPT | 35 |
| Figure 24 Eclipse JEE | 35 |
| Figure 25 Photoshop cc..... | 36 |
| Figure 26 JMX..... | 36 |
| Figure 27 WebSocket | 37 |
| Figure 28 page accuei | 38 |
| Figure 29 authentification..... | 39 |
| Figure 30 liste de mtieres..... | 39 |
| Figure 31 fenêtre de d'erreur l'orsque l'étudiant triche..... | 40 |
| Figure 32 fenêtre de d'erreur l'orsque la date d'examen n'est pas compatible. | 40 |
| Figure 33 fenêtre de contraint..... | 41 |
| Figure 34 page d'examen | 41 |
| Figure 35 note etudiant | 42 |
| Figure 36 fenêtre de contrôle examen. | 42 |
| Figure 37 fenêtre de gestion d'examen. | 43 |
| Figure 38 Fenêtre pour ajouter un examen | 43 |
| Figure 39 modifier un examen | 44 |
| Figure 40 Fenêtre lister des questions..... | 44 |
| Figure 41 ajoute question | 45 |
| Figure 42 Fenêtre modifier question | 45 |

Introduction

La formation en DUT se complète par un projet de fin d'études. Celui-ci constitue une étape obligatoire pour l'obtention du diplôme.

Dans le cursus de formation de deux années en DUT, le projet fin d'études est conçu comme un processus d'immersion réelle dans une fonction opérationnelle tant dans ses dimensions gestionnaires que managériales.

Notre principale mission été donc de créer une application web de gestion de tricherie dans un examen en ligne en JEE.

Au cours de la réalisation de ce projet, nous avons pu surmonter quelques obstacles rencontrés au début dont nous citons le choix de la Template à utiliser aussi l'apprentissage de certaines technologies.

Et en fin nous avons pu acquérir des expériences que nous avons développées, cette période nous a réellement permis de comprendre la difficulté d'effectuer un travail qui a de la valeur ainsi elle nous a enrichi l'esprit de groupe et le travail collectif.

Chapitre I : Analyse et spécification des besoins

La réalisation de tous les projets doit commencer par une phase très important c'est l'analyse du projet et la spécification de besoins, qui permet d'avoir les charges nécessaire à la réalisation du projet et c'est la première chose qui va faire le développeur pour connaître les objectifs à atteindre, et bien comprendre la nature du problème à résoudre pour passer à réfléchir aux solutions du problème.

1. Cahier de charges

1. Présentation de projet :

Pour appliquer les connaissances acquises pendant le cours de programmation avancée en Java et pour aborder les concepts des cours objets, nous sommes amenés à réaliser un projet

informatique sous forme d'une application en ligne « contrôle de tricherie d'un Examen en ligne » basé sur la technologie Java et sur la connexion client/serveur.

Pour répondre à cet objectif, un examen en ligne désire mettre en place un outil de contrôle de triche l'heure de l'examen qui est géré par le professeur. Cet outil doit intégrer les fonctionnalités suivantes :

- module d'identification : identification d'un agent client (étudiants).
- module d'examen : passer des examens par les étudiants.
- module de contrôle de triche : lorsqu'un agent utilise au temps de l'examen l'un des paramètres non autorisés (Word, Reader, etc.) Ou bien dépasser le temps de l'examen, le serveur reçoit un message contenant les informations d'étudiant (adresse IP, login), dans ce cas l'étudiant sera redirigé vers une page d'erreur. Pour réaliser cette partie nous avons choisi comme solution l'API JMX qui permet de construire et de mettre en œuvre une solution de gestion de ressources sous une forme modulaire grâce à des composants. Son but est de proposer un standard pour faciliter le développement de systèmes de contrôle, d'administration et de supervision des applications et des ressources.
- module de note : Donner la note de chaque étudiant.

2. Les objectifs de projet :

Parmi l'objectif de cette application « contrôle de tricherie d'un examen en ligne » :

- Passer des tests d'une manière fiable plus l'égalité des opportunités.
- Le choix de logiciel autorisé à utiliser au temps de l'examen par le professeur.
- la gestion des examens par le professeur.

3. Spécification des besoins :

La partie analyse nous a permis d'identifier de manière exhaustive les besoins auxquels le système doit répondre :

1. Gérer un examen en ligne respectant les règles de l'examen.
2. Permettre le contrôle de triche.

4. Présentation des acteurs :

La plateforme de gestion de contrôle de tricherie d'un examen en ligne comprendra deux acteurs :

- Le professeur :
 - Créer, Modifier, Supprimer, Valider les examens.
 - Contrôler l'examen.
- L'étudiant :
 - Consulter et passer des examens.

5. Processus de développement :

1.5.1 Le modèle de cycle de vie

Le « cycle de vie d'un logiciel », désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition. L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement logiciel, c'est-à-dire la conformité du logiciel avec les besoins exprimés, et la vérification du processus de développement, c'est-à-dire l'adéquation des méthodes mises en œuvre.

Le modèle du cycle en V est un modèle conceptuel de gestion de projet imaginé suite au problème de réactivité du modèle en cascade. Il permet, en cas d'anomalie, de limiter un retour aux étapes précédentes. Les phases de la partie montante doivent renvoyer de 14 L'information sur les phases en vis-à-vis lorsque des défauts sont détectés, afin d'améliorer le logiciel.

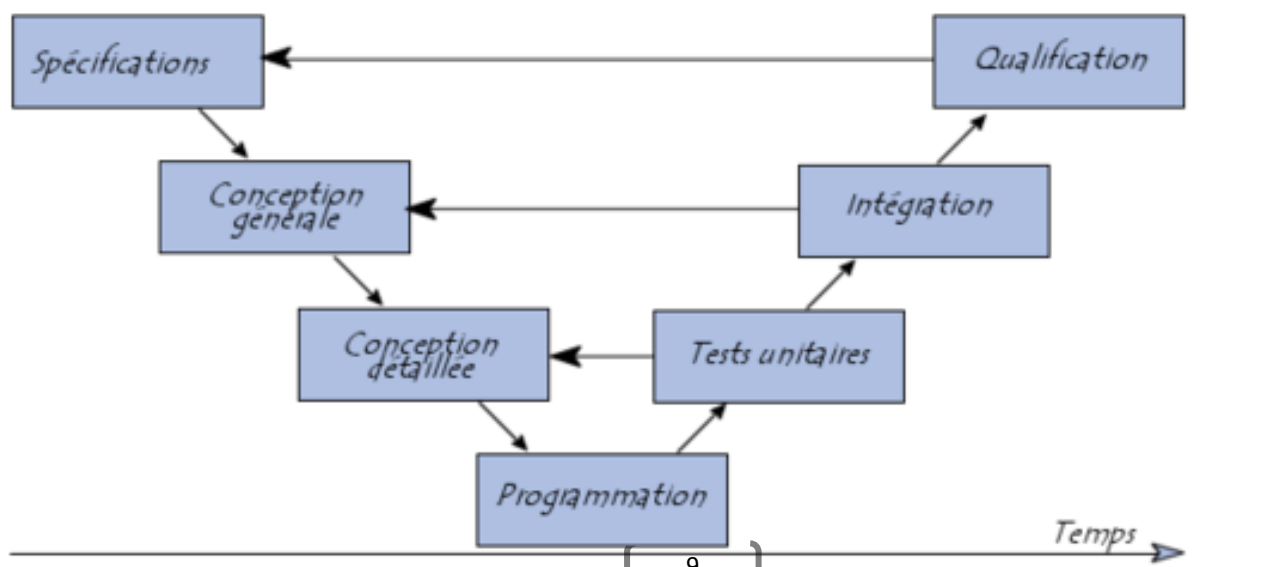


Figure 1 Méthode Projet du cycle en vie

Chapitre II : Modélisation et Conception de l'application

Ce chapitre aborde, dans un premier temps, la présentation des méthodes utilisées dans l'analyse et la conception ainsi l'architecture de gestion qui constitue l'application et la manière dont le système doit répondre.

1. La conception:

1. Modèle relationnel :

Le modèle relationnel est une manière de modéliser les informations contenues dans une base de données qui repose sur des principes mathématiques. Le modèle relationnel est basé sur la notion d'ensemble. Chaque ensemble possède un nom et aussi des attributs nommés qui appartiennent dans cet ensemble et il existe des restrictions intégrales qui limitent cet

ensemble. On peut présenter cet ensemble sous forme d'une table relationnelle qui porte son nom et les attributs sont représentés par les colonnes. Dans les cellules de la table peuvent être seulement les éléments atomiques. L'aménagement des dates dans la base des données est pour un utilisateur non essentiel et les dates peuvent exister indépendamment sur le dépôt physique. Un grand avantage de ce modèle est que l'utilisateur n'a pas à savoir où sont les dates physiquement déposées et il peut les utiliser sans problème. C'est un grand avantage par rapport au modèle hiérarchique ou au modèle réseau.

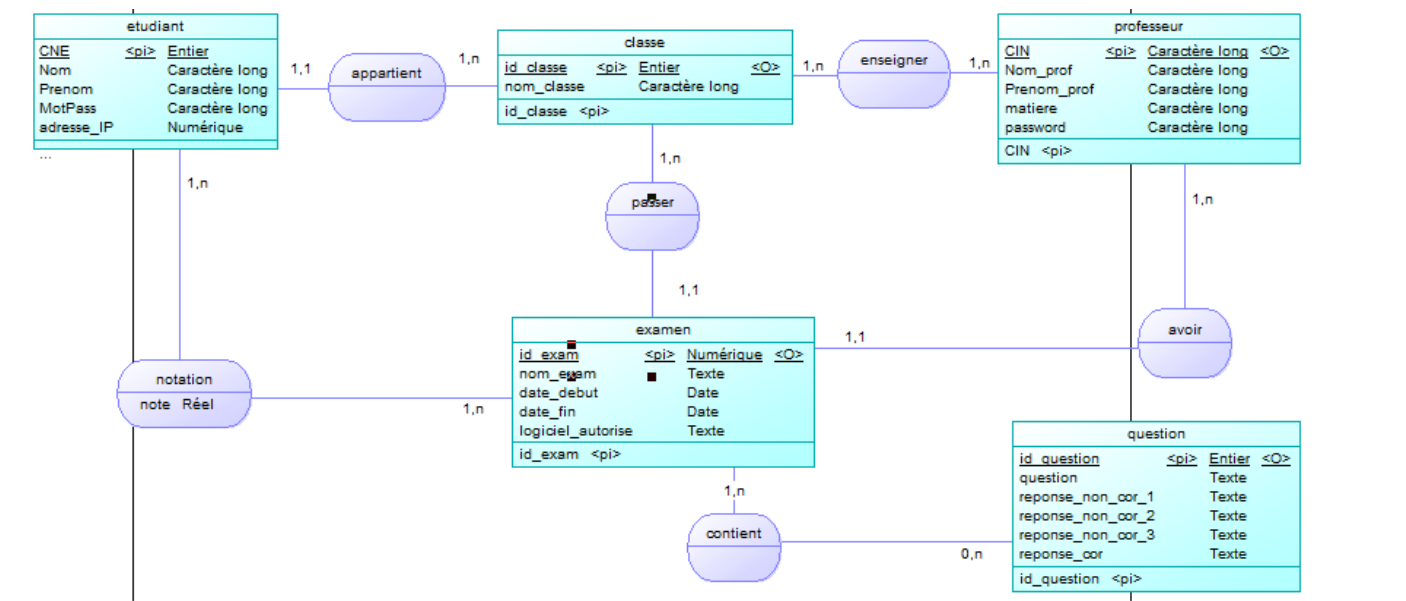


Figure 2 MCD

2. Environnement de modélisation UML :

1.1 Définition :

UML (Unified Modeling Language), un langage graphique de modélisation des données et des traitements. C'est une notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion de plusieurs méthodes existant auparavant, et est devenu désormais la référence en terme de modélisation objet, à un tel point que sa connaissance est souvent nécessaire pour obtenir un poste de développeur objet.

1.2 Le formalisme d'UML :

UML propose 13 types de diagrammes (9 en UML 1.3). UML n'étant pas une méthode, leur utilisation est laissée à l'appréciation de chacun, même si le diagramme de classes est généralement considéré comme l'élément central d'UML, des méthodologies, telles que

l'Unified Process, axent l'analyse en tout premier lieu sur les diagrammes de cas d'utilisation(Use Case.

Les 13 diagrammes UML sont dépendants hiérarchiquement et se complètent, de façon à permettre la modélisation d'un projet tout au long de son cycle de vie.

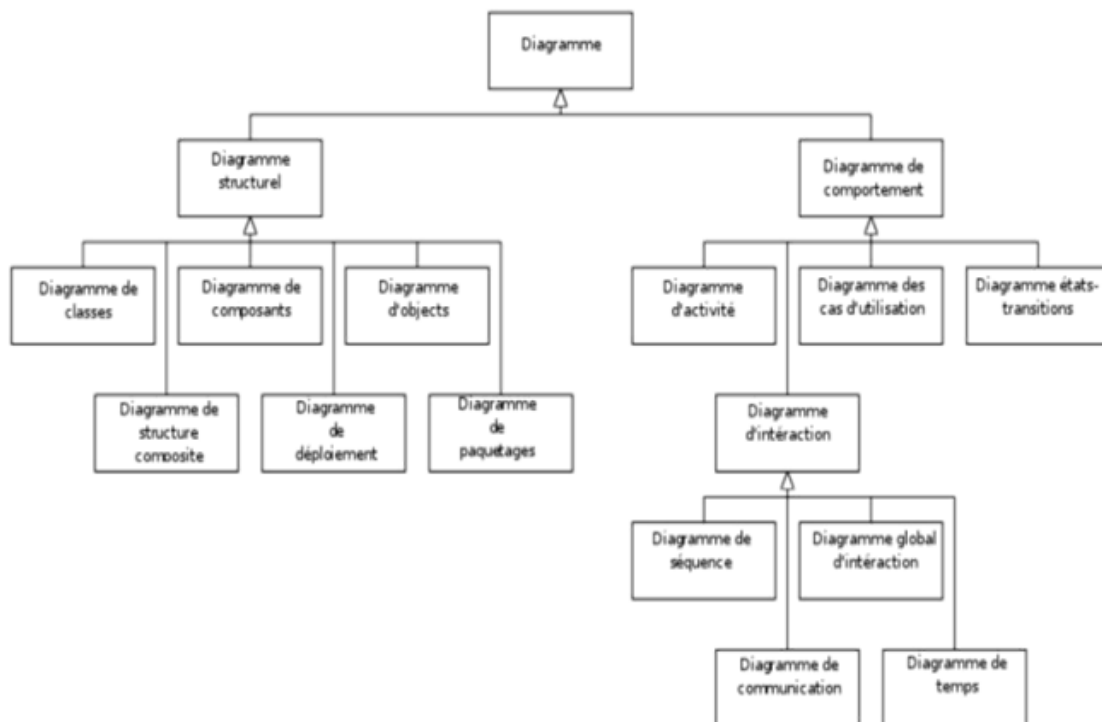


Figure 3 : diagramme uml

Diagrammes structurels ou statiques

Les diagrammes structurels ou statiques rassemblent :

- **Diagramme de classes** : C'est un ensemble d'éléments statiques qui montre la structure d'un modèle(les classes, leur type, leur contenu et leurs relations).
- **Diagramme d'objets** : il sert à représenter les instances de classes (objets) utilisées dans le système. □ **Diagramme de composants**: il permet de montrer les composants du système d'un point de vue physique, tels qu'ils sont mis en œuvre (fichiers, bibliothèques, bases de données...)

- **Diagramme de déploiement** : il sert à représenter les éléments matériels (ordinateurs, périphériques, réseaux, systèmes de stockage...) et la manière dont les composants du système sont répartis sur ces éléments matériels et interagissent entre eux.

- **Diagramme des paquetages** : un paquetage étant un conteneur logique permettant de regrouper et d'organiser les éléments dans le modèle UML, le diagramme de paquetage sert à représenter les dépendances entre paquetages, c'est-à-dire les dépendances entre ensembles de définitions.

- **Diagramme de structure composite** : depuis UML 2.x, permet de décrire sous forme de boîte blanche les relations entre composants d'une classe.

- **Diagramme de profils** : depuis UML 2.2, permet de spécialiser, de personnaliser pour un domaine particulier un méta-modèle de référence d'UML. Diagrammes comportementaux

Les diagrammes comportementaux rassemblent :

- **Diagramme des cas d'utilisation** : il permet d'identifier les possibilités d'interaction entre le système et les acteurs (intervenants extérieurs au système), c'est-à-dire toutes les fonctionnalités que doit fournir le système.

- **Diagramme états-transitions** : permet de décrire sous forme de machine à états finis le comportement du système ou de ses composants.

- **Diagramme d'activité** : permet de décrire sous forme de flux ou d'enchaînement d'activités le comportement du système ou de ses composants. **Diagrammes d'interaction ou dynamiques**

Les diagrammes d'interaction ou dynamiques rassemblent :

- **Diagramme de séquence**: représentation séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou de ses acteurs.

- **Diagramme de communication** : depuis UML 2.x, représentation simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets.

- **Diagramme global d'interaction** : depuis UML 2.x, permet de décrire les enchaînements possibles entre les scénarios préalablement identifiés sous forme de diagrammes de séquences (variante du diagramme d'activité).

• **Diagramme de temps** : depuis UML 2.3, permet de décrire les variations d'une donnée au cours du temps.

3. Les diagrammes :

1.1 Diagramme de cas d'utilisation :

Diagrammes d'étudiant:

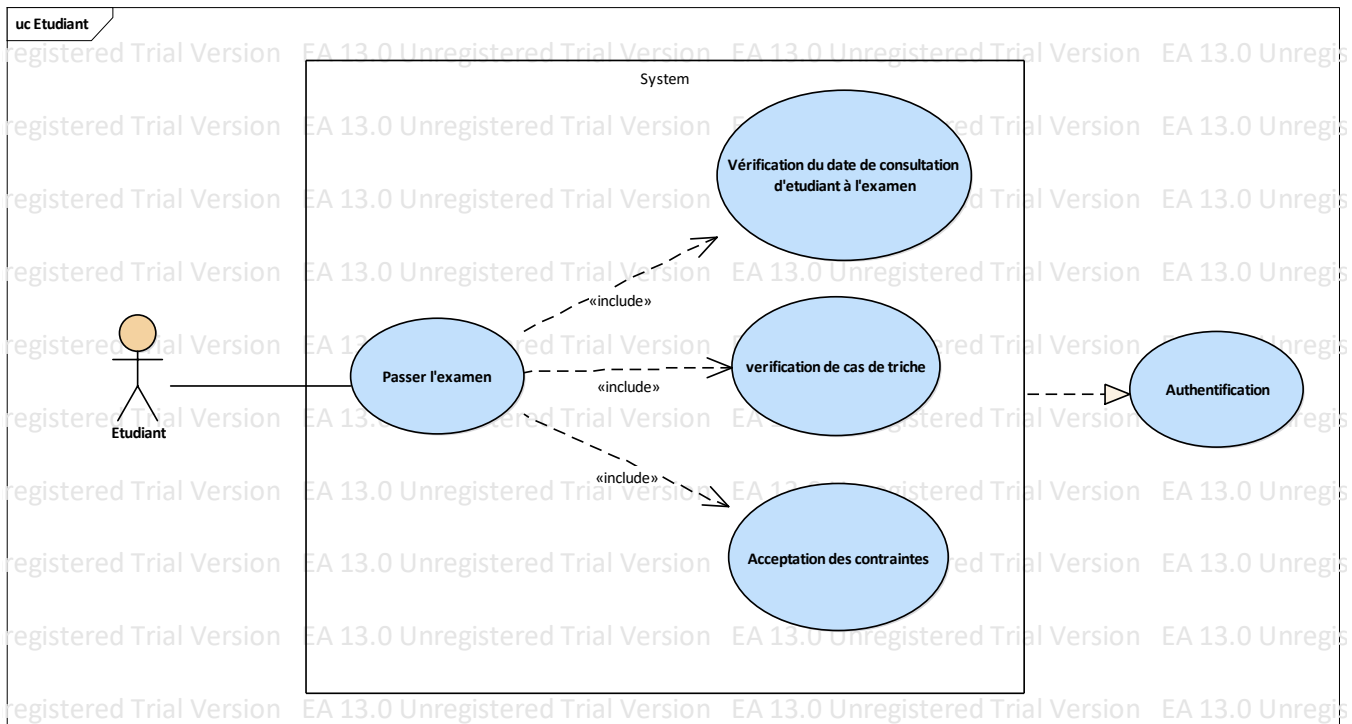


Figure 4 diagramme etudiant

Sommaire d'identification

Titre : Authentification.

Résumé : L'utilisateur saisie son nom utilisateur et mot de passe pour accéder à les fonctionnalités de l'application.

Acteurs : Utilisateur (principal), Système(Secondaire).

Description des scénarios

Scénario nominal

| | |
|----------------------------------|--|
| 1. Ce cas d'utilisation commence | |
|----------------------------------|--|

| | |
|--|-----------------------------------|
| quand l'utilisateur veut s'authentifier. | |
| 2. L'utilisateur saisie ces informations. | 3. le système vérifie les données |
| 4. Après avoir saisie tous les données et le système redirige l'utilisateur vers sa page de profile. | |

Sommaire d'identification

Titre : Passer l'examen.

Résumé : L'étudiant choisit l'examen et il répond aux questions avant d'accepter les contraintes concerné.

Acteurs : Etudiant (principal), Système(Secondaire).

Description des scénarios

Scénario nominal

| | |
|--|--|
| 1. Ce cas d'utilisation commence quand l'utilisateur veut passer l'examen. | |
| 2. L'utilisateur choisit l'examen. | 3. Le système vérifie la date de consultation d'étudiant à l'examen. |
| 4. Le système vérifie est ce que l'étudiant est déjà tricher dans cet examen | 4. Accepter les contraintes de l'examen |
| 5. Après avoir accepté l'étudiant les contraintes le system vers la page d'examen. | |

Diagrammes de professeur :

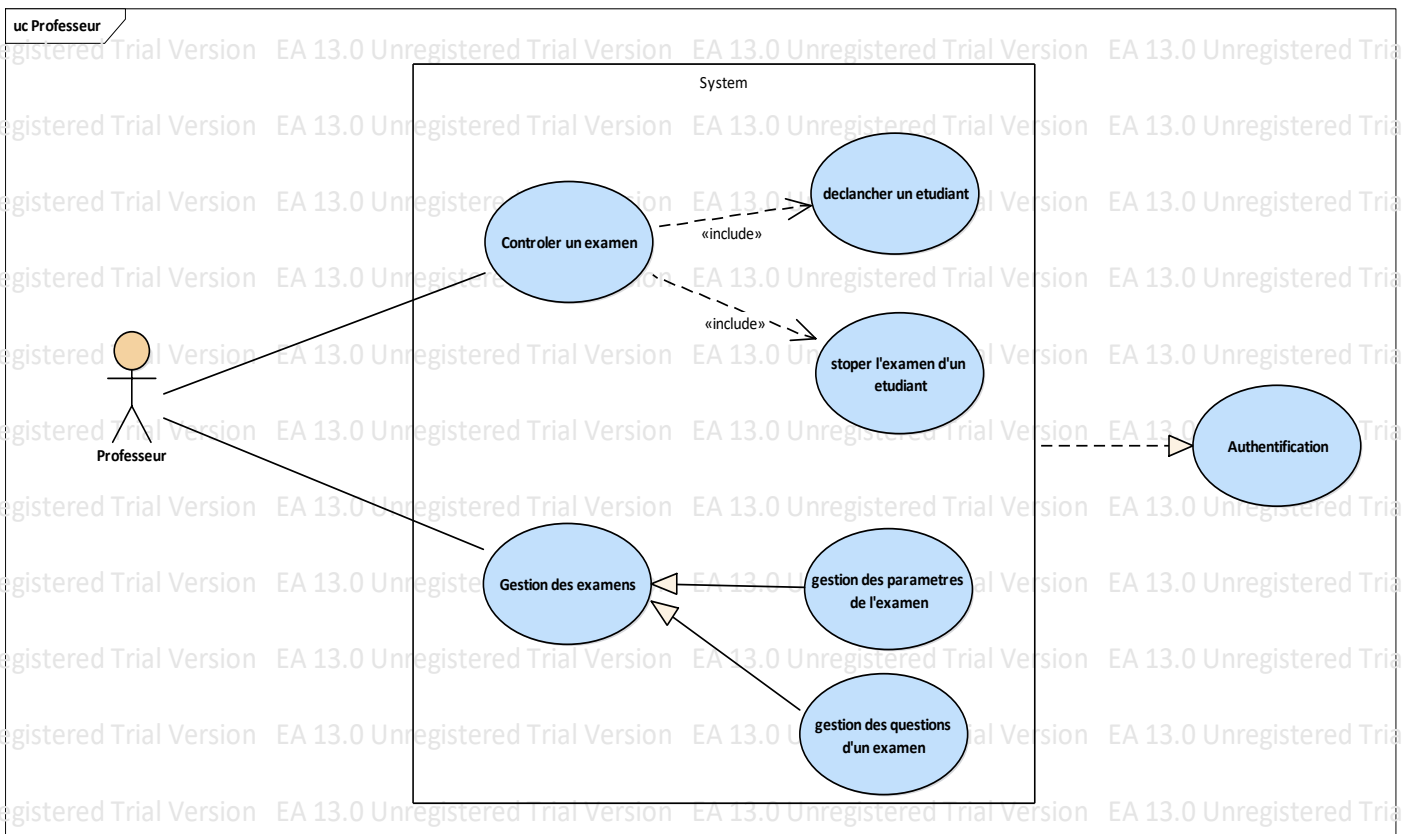


Figure 5 diagramme professeur

Sommaire d'identification

Titre : Contrôler examen.

Résumé : Le professeur redirige vers la page de tracking et il choisit l'étudiant puis il stoppe son examen.

Acteurs : Professeur (principal), Système(Secondaire).

Description des scénarios

Scénario nominal

| | |
|--|---|
| 1. Ce cas d'utilisation commence quand l'utilisateur veut contrôler les étudiants. | |
| 2. L'utilisateur choisit l'examen qui peut contrôler. | 3. Le système détecte l'étudiant connecté et les tricheurs à la fois. |

4. Le professeur choisit l'étudiant qui peut fermer sa session

4. le system stop l'examen de l'étudiant qu'il triche.

1.2 Diagramme de classe :

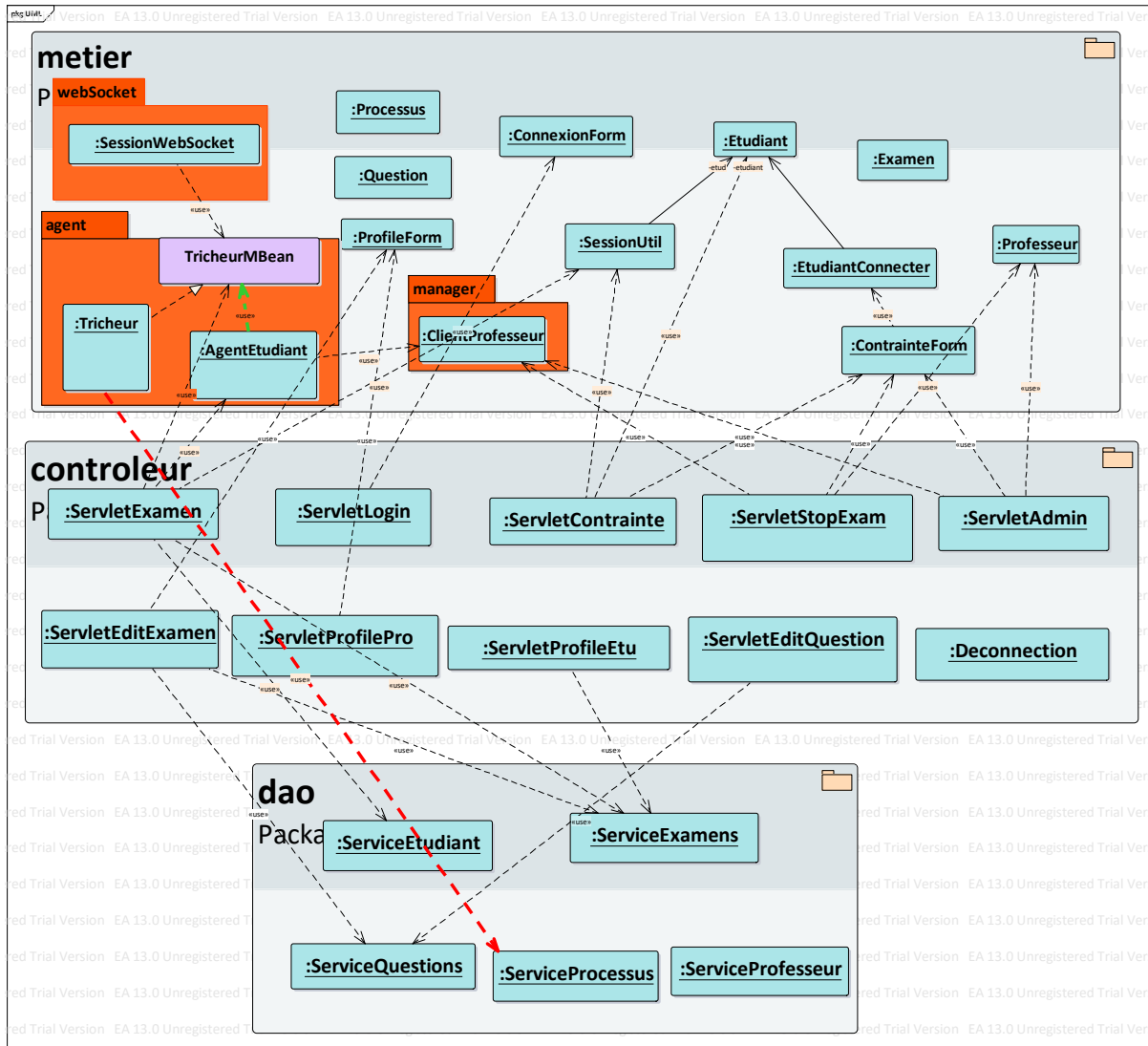


Figure 6 diagramme de classe

L'application contient trois paquetages métier, contrôleur et DAO. Le paquetage Beans contient lui-même des classe entités et deux paquetage comme agent contient les classes qui concerne le MBean (Etudiant) et paquetage client contient les classe qui concerne le manager (Professeur), le paquetage Contrôleur contient les Servlets et Dao contient les classes qui interagir avec la base de données.

Diagrammes de classe dao :

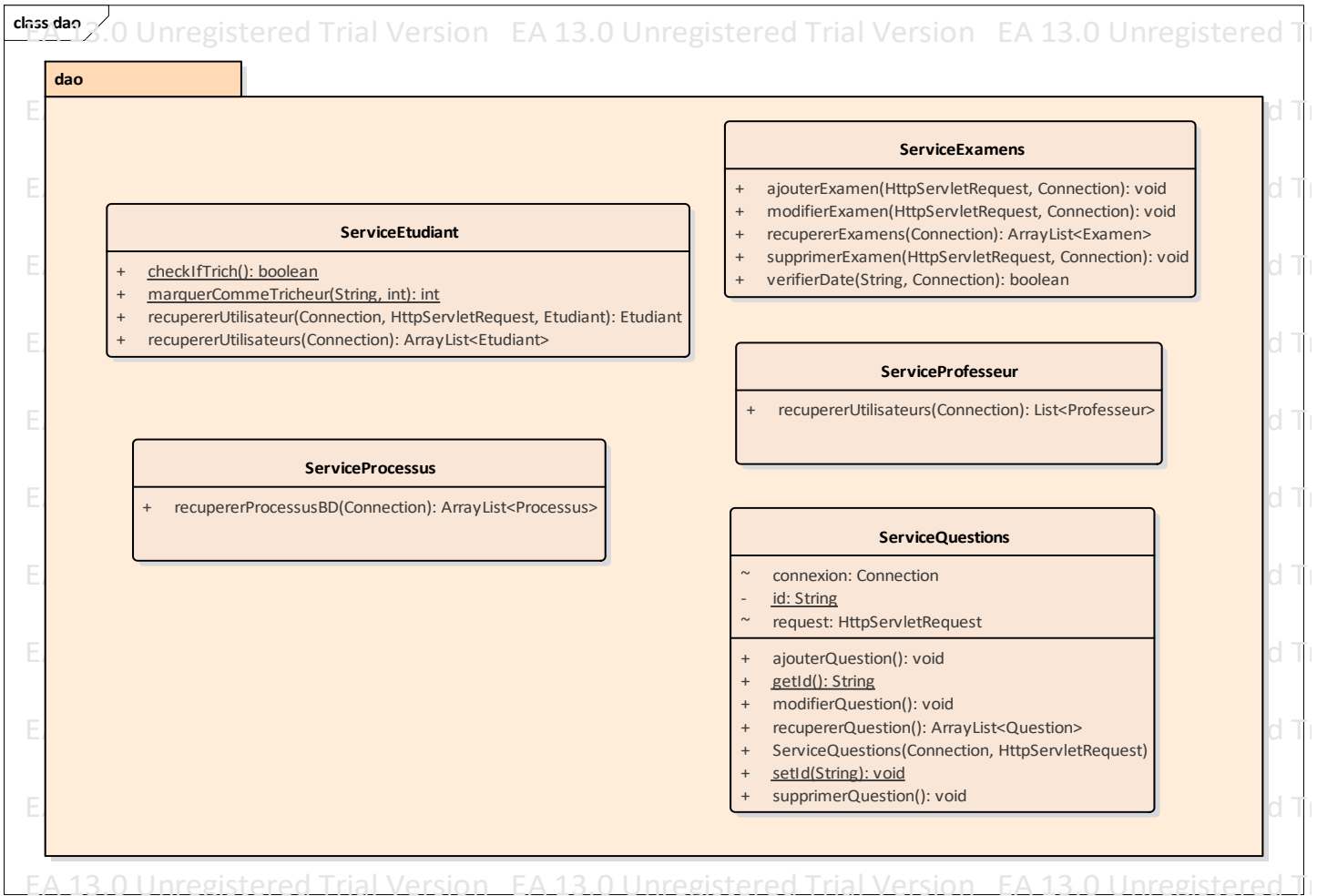


Figure 7 diagramme classe dao

Diagrammes de classe métier :

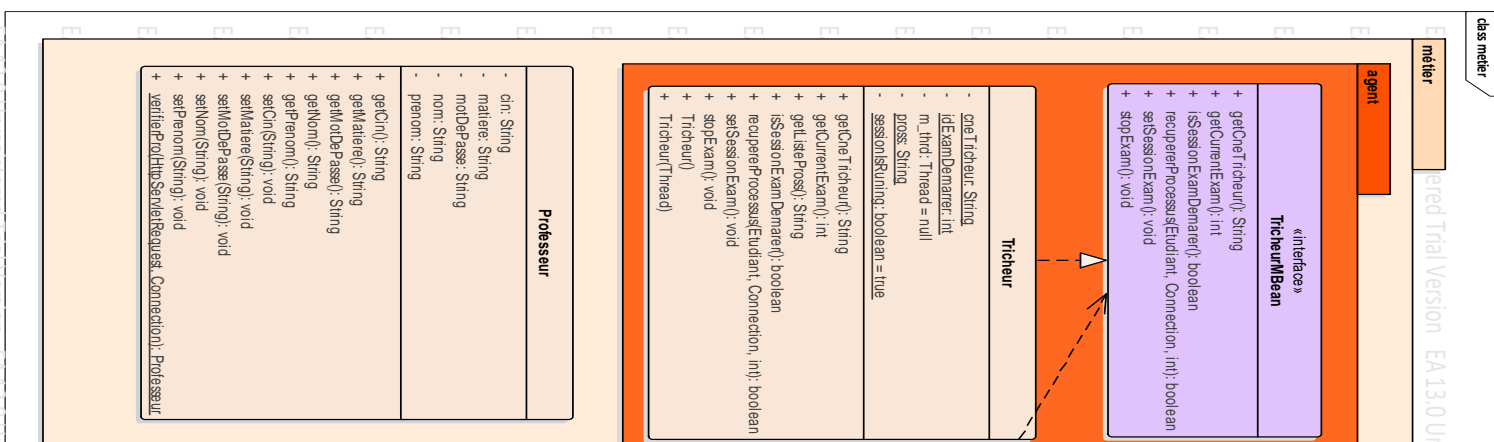


Figure 8 Diagrammes de classe métier

class controleur

controleur

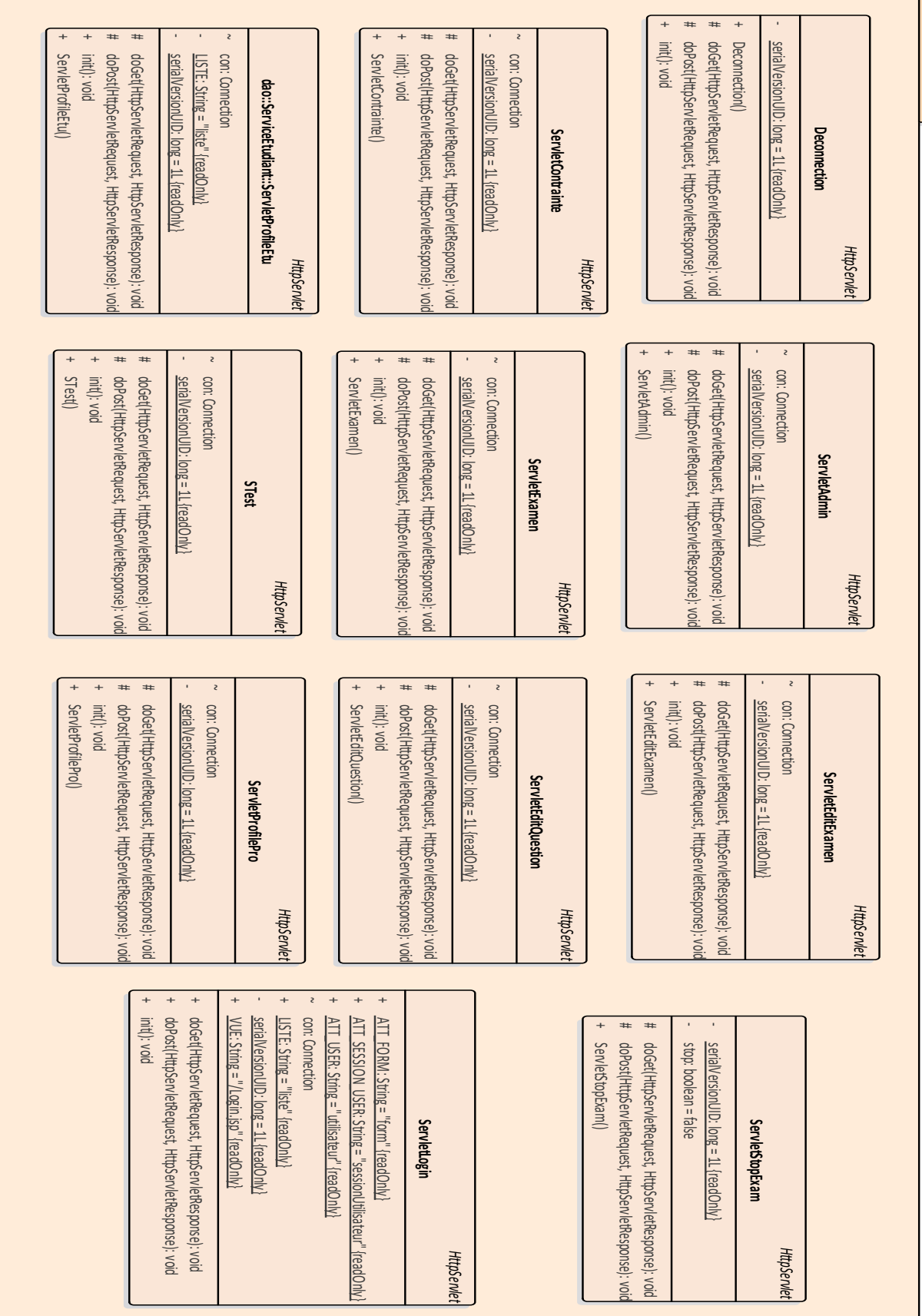


Figure 9 Diagrammes de classe contrôleur

1.3 Diagramme de séquence :

Authentification :

Les utilisateurs étudiant, professeur doivent s'authentifier pour accéder à l'application c'est pour cela ils doivent saisir leurs Nom utilisateur et mot de passe, ces derniers vont être vérifié et comparer avec la base de données.

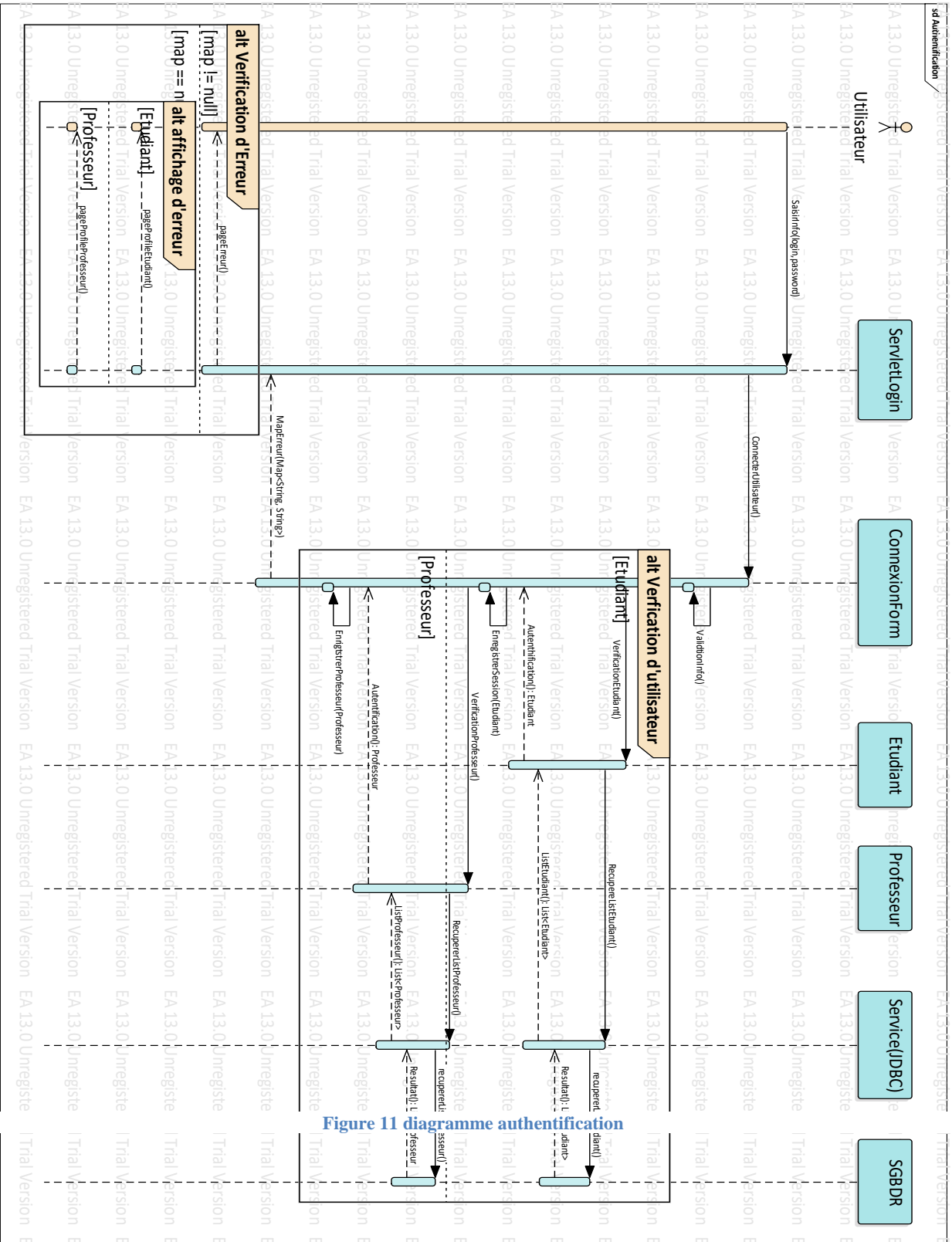


Figure 11 diagramme authentication

Contrôler un examen :

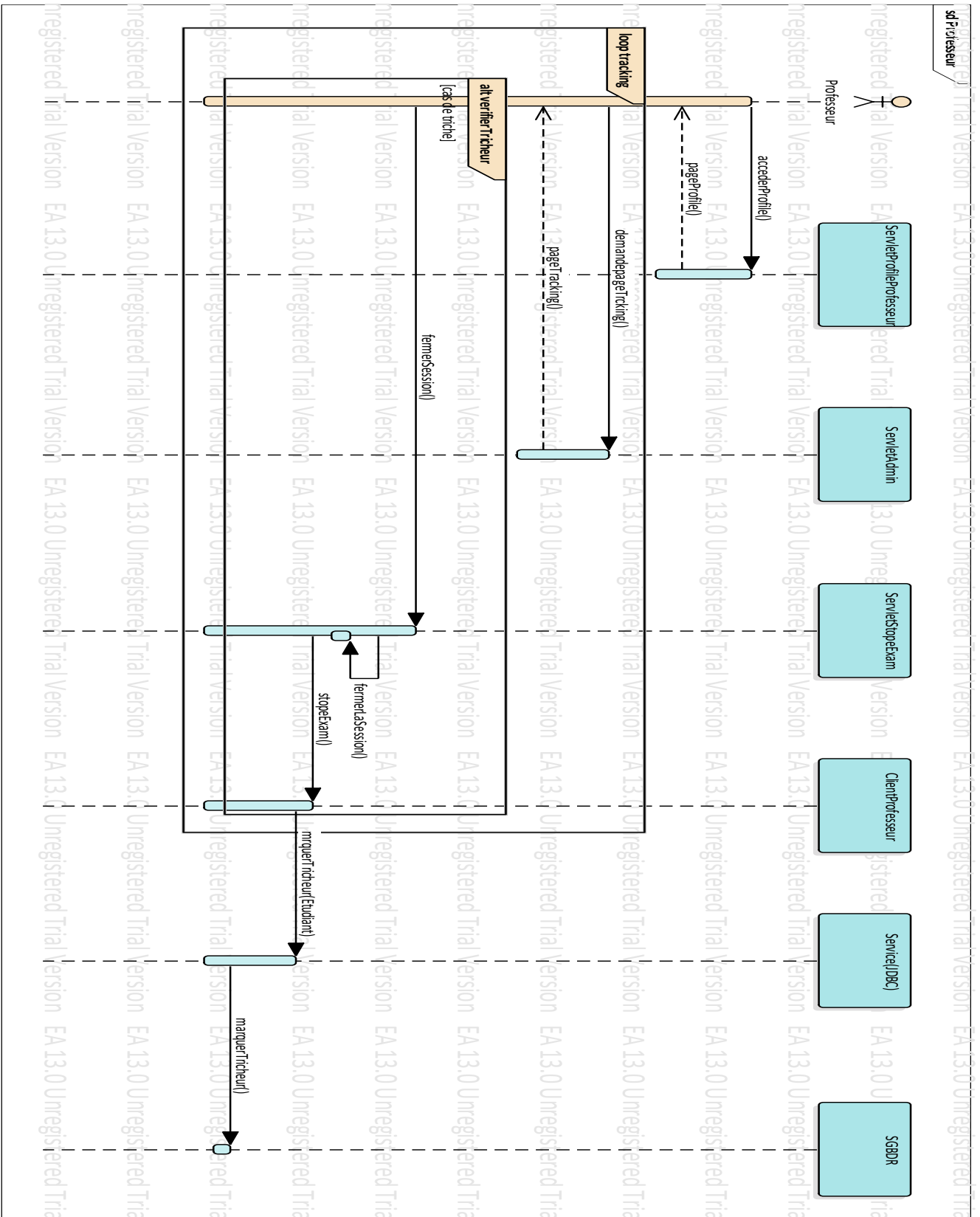


Figure 13 diagramme de controler un examen

Professeur peut contrôler les étudiants par ses adresses IP, de cas de triche il va stopper leur examen et va marquer les tricheurs dans la base de donner.

2. Les architectures et les technologies utilisées:

Pour développer notre application nous avons utilisé Java Entreprise Edition (JEE) on implémente le modèle MVC (modèle-vue-contrôleur) et pour le contrôle d'étudiant on a utilisé l'API Java Management Extensions.

1.1. Le patron de conception « design pattern » MVC :

Le patron Modèle-vue-contrôleur(en abrégé MVC, de l'anglais Model-View-Controller), tout comme les patrons Modèle-vue-présentation ou Présentation, abstraction, contrôle, est un modèle destiné à répondre aux besoins des applications interactives en séparant les problématiques liées aux différents composants au sein de leur architecture respective.

Ce paradigme regroupe les fonctions nécessaires en trois catégories :

- un modèle (modèle de données),
- une vue (présentation, interface utilisateur)
- un contrôleur (logique de contrôle, gestion des événements, synchronisation)

1.1. Présentation de l'architecture MVC

L'organisation d'une interface graphique est délicate. L'architecture MVC ne prétend pas

en éliminer tous les problèmes, mais fournit une première approche pour le faire.

Offrant un cadre normalisé pour structurer une application, elle facilite aussi le dialogue entre les concepteurs.

L'idée est de bien séparer les données, la présentation et les traitements. Il en résulte les trois parties énumérées plus haut : le modèle, la vue et le contrôleur.

Le modèle :

Le modèle représente le cœur (algorithmique) de l'application : traitements des données, interactions avec la base de données, etc. Il décrit les données manipulées par l'application. Il regroupe la gestion de ces données et est responsable de leur intégrité. La base de données sera l'un de ses composants. Le modèle comporte des méthodes standards pour mettre à jour ces données (insertion, suppression, changement

de valeur). Il offre aussi des méthodes pour récupérer ces données. Les résultats renvoyés par le modèle ne s'occupent pas de la présentation. Le modèle ne contient aucun lien direct vers le contrôleur ou la vue. Sa communication avec la vue s'effectue au travers du patron Observateur.

La vue :

C'est avec quoi l'utilisateur interagit se nomme précisément la vue. Sa première tâche est de présenter les résultats renvoyés par le modèle. Sa seconde tâche est de recevoir toute action de l'utilisateur (Clic de souris, sélection d'un bouton radio, entrée de texte, de mouvements de voix, etc). Ces différents événements sont envoyés au contrôleur. La vue n'effectue pas de traitement, elle se contente d'afficher les résultats des traitements effectués par le modèle et d'interagir avec l'utilisateur.

Le contrôleur :

Le contrôleur prend en charge la gestion des événements de synchronisation pour mettre à jour la vue ou le modèle et les synchroniser. Il reçoit tous les événements de l'utilisateur et enclenche les actions à effectuer. Si une action nécessite un changement des données, le contrôleur demande la modification des données au modèle, et ce dernier notifie la vue que les données ont changé pour qu'elle se mette à jour, D'après le patron de conception observateur/observable, la vue est un <<observateur>> du modèle qui est lui <<observable>> Certains événements de l'utilisateur ne concernant pas les données mais la vue. Dans ce cas, le contrôleur demande à la vue de se modifier. Le contrôleur n'effectue aucun traitement, ne modifie aucune donnée. Il analyse la requête du client et se contente d'appeler le modèle adéquat et de renvoyer la vue correspondant à la demande.

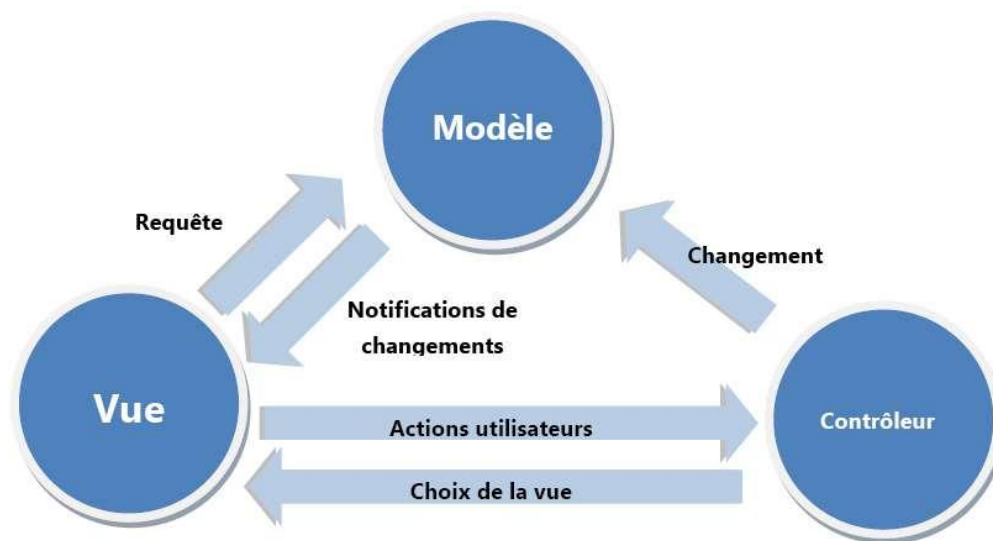


Figure 14 Design pattern MVC

1.2. Java Management Extensions :

1.1. La présentation JMX

Java Management Extensions ou bien JMX est une spécification qui définit une architecture, une API et des services pour permettre de surveiller et de gérer des ressources en Java. Il permet de mettre en place, en utilisant un standard, un système de surveillance et de gestion d'une application, d'un service ou d'une ressource sans avoir à fournir beaucoup d'efforts.

JMX permet de construire et de mettre en œuvre une solution de gestion de ressources sous une forme modulaire grâce à des composants. Son but est de proposer un standard pour faciliter le développement de systèmes de contrôle, d'administration et de supervision des applications et des ressources.

Ainsi les utilisations possibles de JMX sont nombreuses, par exemple :

- consulter et modifier les paramètres de la configuration
- calculer et diffuser des statistiques d'utilisation
- émettre des événements lors de changements d'état ou d'erreurs
- ...

1.2. L'architecture de JMX

L'architecture de JMX se compose de plusieurs niveaux :

Services distribués :

Cette couche définit la partie IHM. C'est généralement une application qui permet de consulter les données relatives à l'application et d'interagir avec elles. Cette couche utilise des connecteurs et des adaptateurs de protocoles pour permettre à des outils de gestion de se connecter à un agent.

Agent :

Cette couche définit un serveur de MBeans qui gère les MBeans. Elle propose des fonctionnalités sous la forme d'un agent JMX et assure la communication avec la couche services distribués grâce à des Connectors et des Adapters.

Instrumentation :

Cette couche définit des MBeans qui permettent l'instrumentation d'une ressource (application, service, composant, objet, appareil, ...) grâce à des attributs, des opérations et des événements. La ressource peut être écrite en Java. Une ressource peut être instrumentée par un ou plusieurs MBeans. Un dynamic MBean implémente une interface particulière qui permet plus de flexibilité à l'exécution. Les MBeans n'ont pas besoin de référence sur l'agent qui va les gérer.

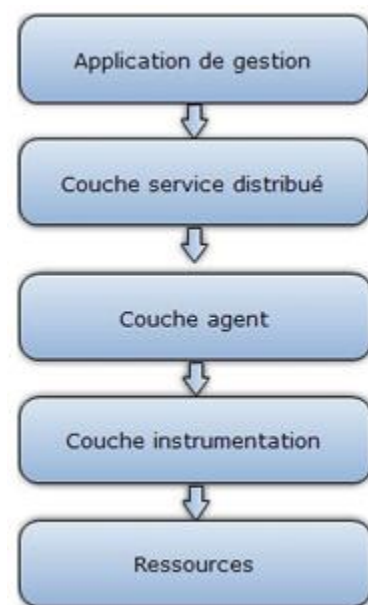


Figure 15 l'architecture de l'API JMX

Le découpage de l'architecture de l'API en trois couches permet une meilleure répartition des rôles et réduit la complexité des fonctionnalités des différentes couches. Chacune des trois couches propose des objets avec des interfaces bien définies. L'élément principal du niveau agent est un objet de type « MBean Server » : son rôle est de gérer et de mettre en œuvre les MBeans qui se sont enregistrés auprès de lui. L'élément principal du niveau instrumentation est un objet de type MBean.

Exemple d'architecture de notre projet :

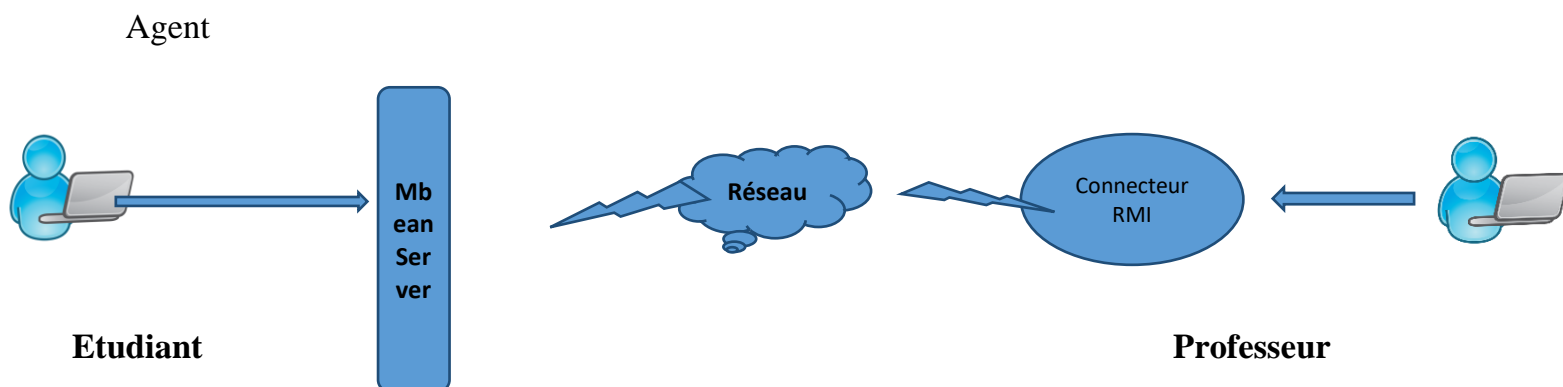


Figure 16 l'architecture de projet contrôle de tricherie

1.3. Connecteur RMI :

Un connecteur permet le dialogue entre l'agent et l'application de gestion distante grâce à un protocole dédié. Un connecteur est composé d'une partie cliente liée à l'application de gestion et d'une partie serveur liée à l'agent JMX. La partie serveur du connecteur attend les connexions de la partie cliente : c'est donc la partie cliente qui est responsable de l'initialisation de la connexion.

Il permet donc à un client JMX (serveur) distant de communiquer avec un agent JMX. L'agent JMX est chargé d'initialiser et de configurer le connecteur côté serveur. Le client JMX est chargé d'initialiser et de configurer le connecteur côté client.

Le but de RMI est de permettre l'appel, l'exécution et le renvoi du résultat d'une méthode exécutée dans une machine virtuelle différente de celle de l'objet l'appelant. Cette machine virtuelle peut être sur une machine différente pourvu qu'elle soit accessible par le réseau.

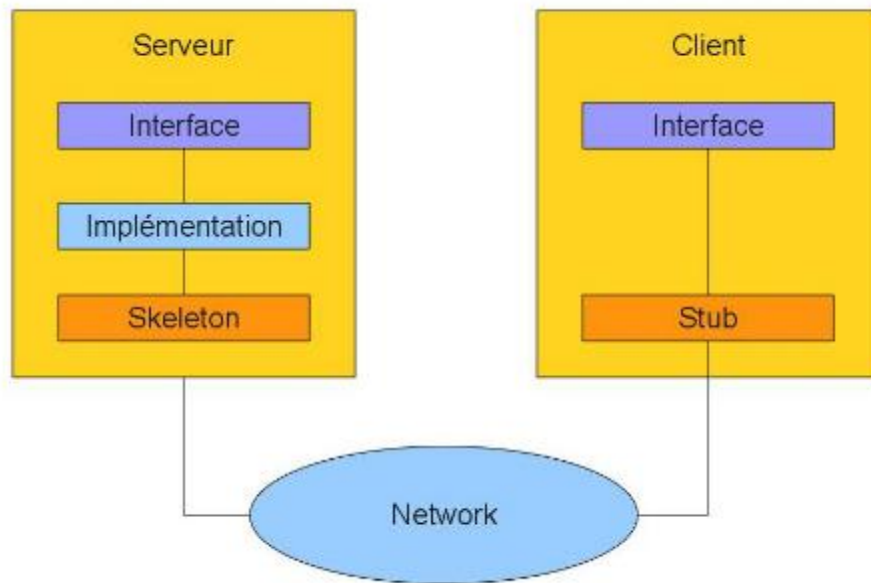


Figure 17 Le fonctionnement de RMI

- l'objet côté client qui va rediriger l'appel vers le serveur (le Stub) ;
- l'objet côté serveur qui sera atteint par l'Object dynamique côté client (le Skeleton).

Chapitre III : Etude technique

Après avoir parlé de l'analyse et la conception de l'application, nous allons proposer une série d'outils logiciels pour la réalisation des différents tiers de l'application.

Il existe plusieurs implémentations possibles des applications, chaque solution technique ayant ses avantages et ses inconvénients. On peut bien faire le choix des outils de développement tels JEE et un serveur Apache avec une base de données tels que MySQL et SQL serveur etc.

Serveur Utilisé :

6. MySQL :



Figure 18 mysql

MySQL est un système de gestion de base de données (SGBD). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, Informix et Microsoft SQL Server.

7. Apache Tomcat



Figure 19 tomcat

Apache Tomcat est un conteneur web libre de servlets et JSP Java EE. Issu du projet Jakarta, c'est un des nombreux projets de l'Apache Software Foundation. Il implémente les spécifications des servlets et des JSP du Java Community Process³, est paramétrable par des

fichiers XML et des propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.

2. Outils de conceptions :

1. Enterprise Architect



Figure 20 UML

Enterprise Architect est un outil d'analyse et de création UML, couvrant le développement de logiciels du rassemblement d'exigences, en passant par les étapes d'analyse, les modèles de conception et les étapes de test et d'entretien. Cet outil graphique basé sur Windows, peut être utilisé par plusieurs personnes et conçu pour vous aider à construire des logiciels faciles à mettre à jour. Il comprend un outil de production de documentation souple et de haute qualité.

2. XAMPP



Figure 21 XAMP

XAMPP est un ensemble de logiciels permettant de mettre en place facilement un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.

3. Langages :

1. Java :



Figure 22 JAVA

Le langage Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java. La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou GNU/Linux, avec peu ou pas de modifications. Pour cela, divers plateformes et frameworks associés visent à guider, sinon garantir, cette portabilité des applications développées en Java.

2. JAVA EE :



Figure 23 JEE

Java Enterprise Edition, ou Java EE est une spécification pour la technique Java d'Oracle plus particulièrement destinée aux applications d'entreprise. Ces applications sont considérées dans une approche multi-niveaux¹. Dans ce but, toute implémentation de cette spécification contient un ensemble d'extensions au framework Java standard (JSE, Java Standard Edition) afin de faciliter notamment la création d'applications réparties.

3. CSS :



Figure 24 CSS

CSS est l'acronyme de Cascading Style Sheets, est un langage de feuille de style utilisé pour décrire la mise en forme d'un document écrit avec un langage de balisage. Il permet aux concepteurs de contrôler l'apparence et la disposition de leurs pages web.

4. HTML :



Figure 25 HTML

HTML5 (HyperText Markup Language 5) est la dernière révision majeure d'HTML . Cette version a été finalisée le 28 octobre 2014. HTML5 spécifie deux syntaxes d'un modèle abstrait défini en termes de DOM : HTML5 et XHTML5. Le langage comprend également une couche application avec de nombreuses API, ainsi qu'un algorithme afin de pouvoir traiter les documents à la syntaxe non conforme. Le travail a été repris par le W3C en mars 2007 après avoir été lancé par le WHATWG. Les deux organisations travaillent en parallèle sur le même document afin de maintenir une version unique de la technologie. Le W3C vise la clôture des ajouts de fonctionnalités le 22 mai 2011 et une finalisation de la spécification en 2014, et encourage les développeurs Web à utiliser HTML 5 dès maintenant.

5. JAVA SCRIPT:



Figure 26 JAVA SCRIPT

C'est un langage de script incorporé dans un document HTML. Historiquement il s'agit du premier langage de script pour le Web. Ce langage est un langage de programmation qui permet d'apporter des améliorations au langage HTM pour exécuter des commandes du côté client, c'est-à-dire au niveau du navigateur et non du serveur web.

4. Outils de développement:

1. Eclipse IDE for Java EE Developers:



Figure 27 Eclipse JEE

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Son objectif est de produire et fournir des outils pour la réalisation de logiciels, englobant les activités de programmation mais aussi d'AGL recouvrant modélisation, conception, test, gestion de configuration, reporting... Son EDI, partie intégrante du projet, vise notamment à supporter tout langage de programmation à l'instar de Microsoft Visual Studio.

Bien qu'Eclipse ait d'abord été conçu uniquement pour produire des environnements de développement, les utilisateurs et contributeurs se sont rapidement mis à réutiliser ses

briques logicielles pour des applications clientes classiques. Cela a conduit à une extension du périmètre initial d'Eclipse à toute production de logiciel : c'est l'apparition du Framework Eclipse RCP en 2004.

Figurant parmi les grandes réussites de l'Open source, Eclipse est devenu un standard du marché des logiciels de développement, intégré par de grands éditeurs logiciels et sociétés de services. Les logiciels commerciaux Lotus Notes 8, IBM Lotus Symphony ou WebSphere Studio Application Développer sont notamment basés sur Eclipse.

5. Outils de désigne:

1. Adobe Photoshop CC :



Figure 28 Photoshop cc

Photoshop est un logiciel de retouche, de traitement et de dessin assisté par ordinateur édité par Adobe. Il est principalement utilisé pour le traitement de photographies numériques, mais sert également à la création d'images.

6. API

1. JMX



Figure 29 JMX

Java Management Extensions (JMX) est une API pour Java permettant de gérer le fonctionnement d'une application Java en cours d'exécution. JMX a été intégré dans J2SE à partir de la version 5.0.

2. WebSocket



Figure 30 WebSocket

WebSocket est un standard du Web désignant un protocole réseau de la couche application et une interface de programmation du World Wide Web visant à créer des canaux de communication full-duplex par-dessus une connexion TCP.

Chapitre IV : Présentation d'application :

La conception des interfaces de l'application est une étape très importante puisque toutes les interactions avec le coeur de l'application passent à travers ces interfaces, on doit alors guider l'utilisateur avec les messages d'erreurs et de notification si besoin, ainsi présenter un système complet.

Dans cette partie, nous allons présenter quelques interfaces de l'application, répondant aux recommandations de compatibilité, de guidage, de clarté, d'homogénéité et de souplesse.

1. Index :



Figure 31 page accueil

Cette page représente page d'accueil.

2. L'authentification



Figure 32 authentification

Un écran d'authentification illustré par la figure doit donc apparaître au moment de l'accès à de l'application, l'utilisateur doit identifier par un login et un mot de passe.

3. Le Compte étudiant :



Figure 33 liste de mtieres

L'onglet « Home » vous permet de retourner a la page d'index.

4. Page d'erreur :

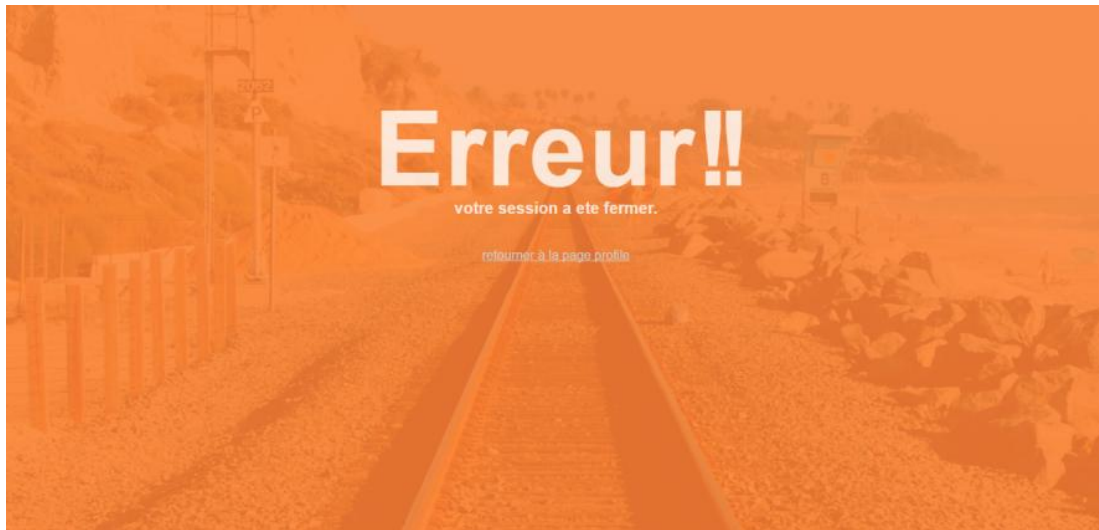


Figure 34 fenêtre de d'erreur l'orsque l'étudiant triche.

Lorsque l'étudiant utilise l'un des logiciels non autoriser,le professeur fait un arrêt a la session de l'étudiant , un message d'erreur va apparaitre

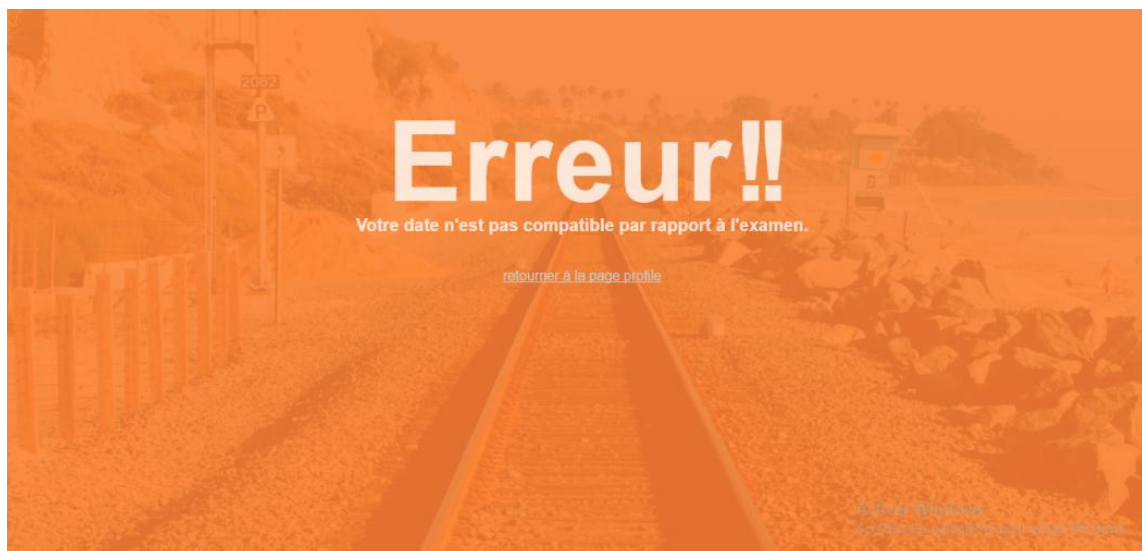


Figure 35 fenêtre de d'erreur l'orsque la date d'examen n'est pas compatible.

Lorsque étudiant veux passer un examen dans une date qui n'est pas compatible a la date d'examen l'étudiant va redirejet vers une page d'erreur,

5. Contraint :



Home

amchayd mariam

DÉCONNEXION

Bienvenu amchayd mariam, avant vous passez votre examen il faut accepter les contraintes suivant: il ne faut pas utiliser les logiciels suivant WORD, ADOBE PDF, CODE BLOC, l'examen peut accessible juste à partir sa date debut jusqu'a la date fin de l'examen.

☐ Accepter il faut accepter les contraintes

Passer l'examen

Copyright © 2017

Realiser par:AGGAGUI YASSINE & AMCHAYDE MARIAM & DAADI FARAH

Activer Windows

Figure 36 fenêtre de contraint.

Pour que l'étudiant passe l'examen il faut qu'il accepte les contraint de l'examen qui sont : il faut respecter le temps de l'examen et ne pas utiliser les logiciels non autoriser.

6. Page d'examen :



Home

amchayd mariam

DÉCONNEXION

Qu'est-ce qui est indispensable au bon fonctionnement d'un site web écrit avec PHP afin qu'il s'affiche correctement dans un navigateur ?

Un navigateur

Un serveur web (Apache, Nginx...)

Un IDE

Un navigateur

précédent

suivant

Enregistrer les repenses

Copyright © 2017


Realiser par:AGGAGUI YASSINE & AMCHAYDE MARIAM & DAADI FARAH

Activer Windows

Figure 37 page d'examen

L'examen sous forme de qcm, l'étudiant choisi leur réponse et clique sur le bouton suivant, a la fin il va cliquer sur enregistré les réponses pour qu'il obtient leur note.

7. Page de note :



Home amchayd mariam DÉCONNEXION

plus de chance la prochaine fois.
You got 2/4 questions correct.

Copyright © 2017
Realiser par: AGGAGUI YASSINE & AMCHAYDE MARIAM & DAADI FARAH

Figure 38 note etudiant

Cette page représente le note de l'étudiant.

8. Le Compte professeur :



Home Gestion d'examen Contrôler examen > LAHMAR Ahmed DÉCONNEXION

Les listes des examens

| MATIÈRE | DATE DEBUT | DATE FIN | |
|---------|------------------|------------------|---------|
| JAVA | 2017/04/05 13:00 | 2017/05/03 13:10 | Suivant |

Copyright © 2017
Realiser par: AGGAGUI YASSINE & AMCHAYDE MARIAM & DAADI FARAH

Figure 39 fenêtre de contrôle examen.

L'ongle « contrôle examen » vous permet de lister les examens disponibles de professeur connecté.

| CIN PROFESSEUR | DATE DÉBUT | DATE FIN | ACTION | Ajouter |
|----------------|------------------|------------------|-----------------------|---------|
| X123456 | 2017/04/05 13:00 | 2017/05/03 13:10 | show exam Edit Delete | |

Figure 40 fenêtre de gestion d'examen.

Cette fenêtre permet de lister et de gérer les examens.

9. L'ajout d'un examen

Ajouter Examen

Cin

Date debut

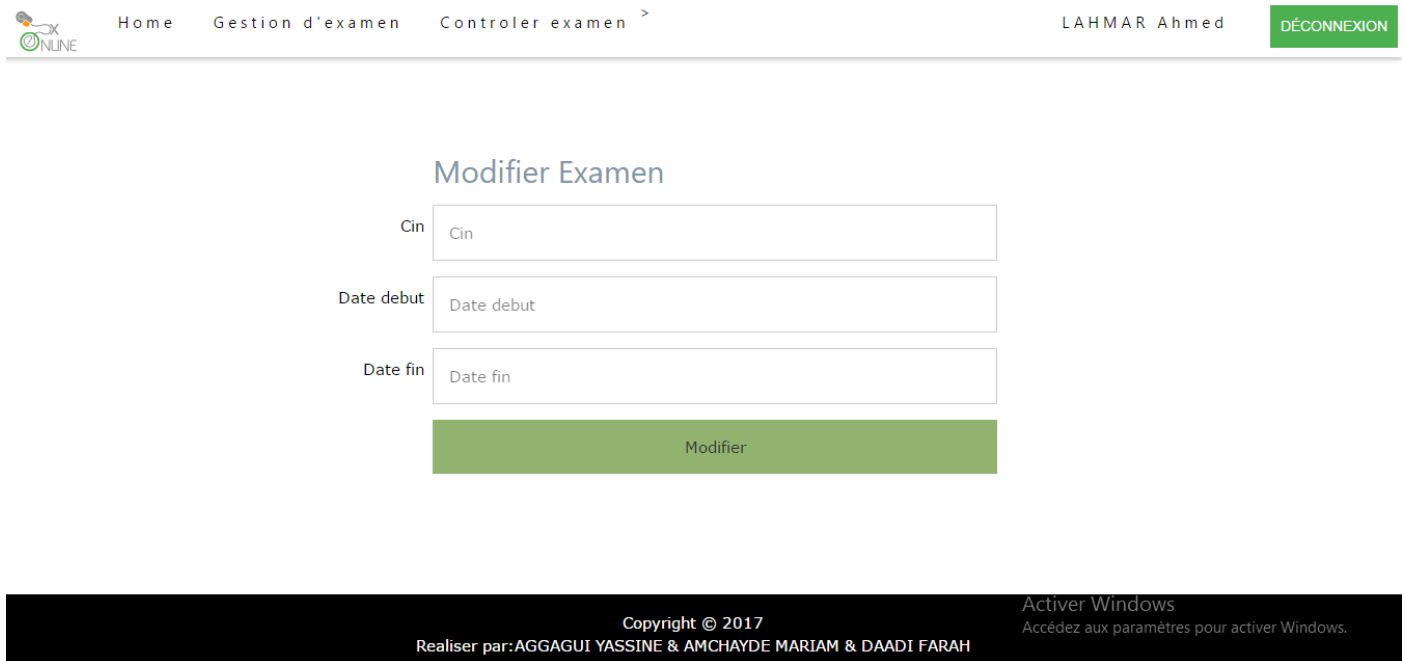
Date fin

Ajouter

Figure 41 Fenêtre pour ajouter un examen

Le professeur peut ajouter un examen en saisi son cin de professeur et les informations sur la date de l'examen.

10. La modification d'un examen



The screenshot shows the 'Modifier Examen' (Modify Exam) form. At the top, there is a navigation bar with 'Home', 'Gestion d'examen', and 'Controler examen >'. The user 'LAHMAR Ahmed' is logged in, with a 'DÉCONNEXION' button. The form contains three input fields: 'Cin' (with the value 'Cin'), 'Date debut' (with the value 'Date debut'), and 'Date fin' (with the value 'Date fin'). A green 'Modifier' button is at the bottom. A footer bar contains copyright information and a Windows activation notice.

Figure 42 modifier un examen

Seul le professeur a la main de modifier les informations des examens.

11. La liste des questions :



| QUESTION | REPENSE CORRECTE | PREMIERE REPENSE | DEUXIEME REPENSE | TROISIEME REPENSE | ACTION | Ajouter |
|------------------|------------------|-------------------|--------------------|-------------------|-------------|---------|
| Que fait die() ? | arrête le script | sort d'une boucle | bypasse une erreur | tue un processus | Edit Delete | |

Figure 43 Fenêtre lister des questions

Cette page représente la liste des questions et réponse d'un examen.

12. L'ajout d'une question


[Home](#)
[Gestion d'examen](#)
[Controler examen](#) >

LAHMAR Ahmed

DÉCONNEXION

Ajouter Question

Question

reponse correcte

premier repense

deuxième repense

troisième repense

Ajouter

Copyright © 2017

Realiser par: AGGAGUI YASSINE & AMCHAYDE MARIAM & DAADI FARAH


Activer Windows

Accédez aux paramètres pour activer Windows.

Figure 44 ajoute question

Le professeur peut ajouter des questions en saisirot leur informations.

13. La modification d'une question


[Home](#)
[Gestion d'examen](#)
[Controler examen](#) >

LAHMAR Ahmed

DÉCONNEXION

Modifier Question

Question

reponse correcte

premier repense

deuxième repense

troisième repense

Modifier

Figure 45 Fenêtre modifier question

Le professeur a la main de modifier les informations de questions d'examens.

Conclusion

Durant notre projet de fin d'études, nous avons pu réaliser une application de contrôle de tricherie qui consiste de contrôler les examens se basant sur la technologie Java EE et l'API JMX. Le projet nous a permis d'enrichir nos connaissances, et d'appliquer le savoir que nous avons acquis durant les deux ans. En effet, ce projet était une occasion pour se familiariser avec l'encadrant. Il a aussi constitué un moyen complémentaire pour notre formation. Nous pensons que cette expérience à l'Ecole Supérieure de Technologie de Meknès nous a offert une bonne préparation à notre insertion professionnelle car elle fut pour nous une expérience enrichissante et complète.

Bibliographies

Jean Michel DOUDOUX documentation (jmdoudoux)

- <https://www.jmdoudoux.fr/java/>

Bootstarp documentation

- <http://getbootstrap.com>