1-Introduction :

LUnified Modeling Language (UML) est un langage de modélisation standard utilisé pour représenter visuellement et spécifier le comportement, la structure et les interactions d'un système logiciel. Il offre une notation graphique permettant de décrire les différents aspects d'un système de manière claire et compréhensible.

L'UML se compose de plusieurs types de diagrammes qui servent à représenter différents aspects du système. Voici quelques-uns des principaux diagrammes utilises en UML :

Diagramme de cas d'utilisation : Il montre les interactions entre les acteurs (utilisateurs) et le système, en identifiant les fonctionnalités fournies par le système.

Diagramme de classes : Il représente les classes du système, leurs attributs, leurs relations.

Diagramme de séquence : Il montre l'ordre chronologique des messages échangés entre les objets lors de l'exécution d'un scénario.

L'UML facilite la communication entre les différentes parties prenantes d'un projet logiciel, comme les développeurs, les architectes et les clients. Il permet de visualiser et de documenter les concepts, les fonctionnalités et les interactions du système de manière claire et structurée.

Il convient de noter que l'UML est un langage de modélisation et non une méthode de développement à part entière. Il peut être utilisé en conjonction avec différentes méthodologies de développement, telles que le développement agile ou le processus unifié rationnel (RUP), pour décrire et concevoir le système de manière efficace.

2-Identification des cas d'utilisations :

1- Administrateur :

UC1 : S'identifier

UC2 : Gérer les comptes .

UC3 : Générer des emploi du temps.

2- Enseignant :

UC1 : Saisir les notes.

UC2 : Consulter leur emploi du temps.

3- Étudiant :

UC1 : Consulter leur emploi du temps.

UC2 : Consulter leur résultats.

3-Les acteurs :

- **Administrateur :** Gère les comptes et génère les emplois du temps.

- **Enseignant :** Peut saisir les notes et consulter son emploi du temps.
- **Étudiant :** Peut consulter ses résultats et son emploi du temps.

4-Classement des cas d'utilisation :

Cas d'utilisation	Acteur	Priorité	Risque
UC1 : S'identifier	Administrateur	Haute	Faible
UC2 : Gérer les comptes	Administrateur	Haute	Haute
UC3 : Générer des emplois du temps	Administrateur	Moyenne	Haute
UC1 : Saisir les notes	Enseignant	Haute	Moyenne
UC2 : Consulter leur emploi du temps	Enseignant	Moyenne	Faible
UC1 : Consulter leur emploi du temps	Étudiant	Moyenne	Faible
UC2 : Consulter leurs résultats	Étudiant	Haute	Moyenne

5- Planification des itérations :

Itération	Cas d'utilisation	Acteur	Priorité	Risque
Itération 1	UC1 : S'identifier	Administrateur	Haute	Faible
Itération 1	UC2 : Gérer les comptes	Administrateur	Haute	Haute
Itération 3	UC3 : Générer des emplois du temps	Administrateur	Moyenne	Haute
Itération 2	UC1 : Saisir les notes	Enseignant	Haute	Moyenne
Itération 3	UC2 : Consulter leur emploi du temps	Enseignant	Moyenne	Faible
Itération 3	UC1 : Consulter leur emploi du temps	Étudiant	Moyenne	Faible
Itération 2	UC2 : Consulter leurs résultats	Étudiant	Haute	Moyenne

6- Diagramme de cas d’utilisation :

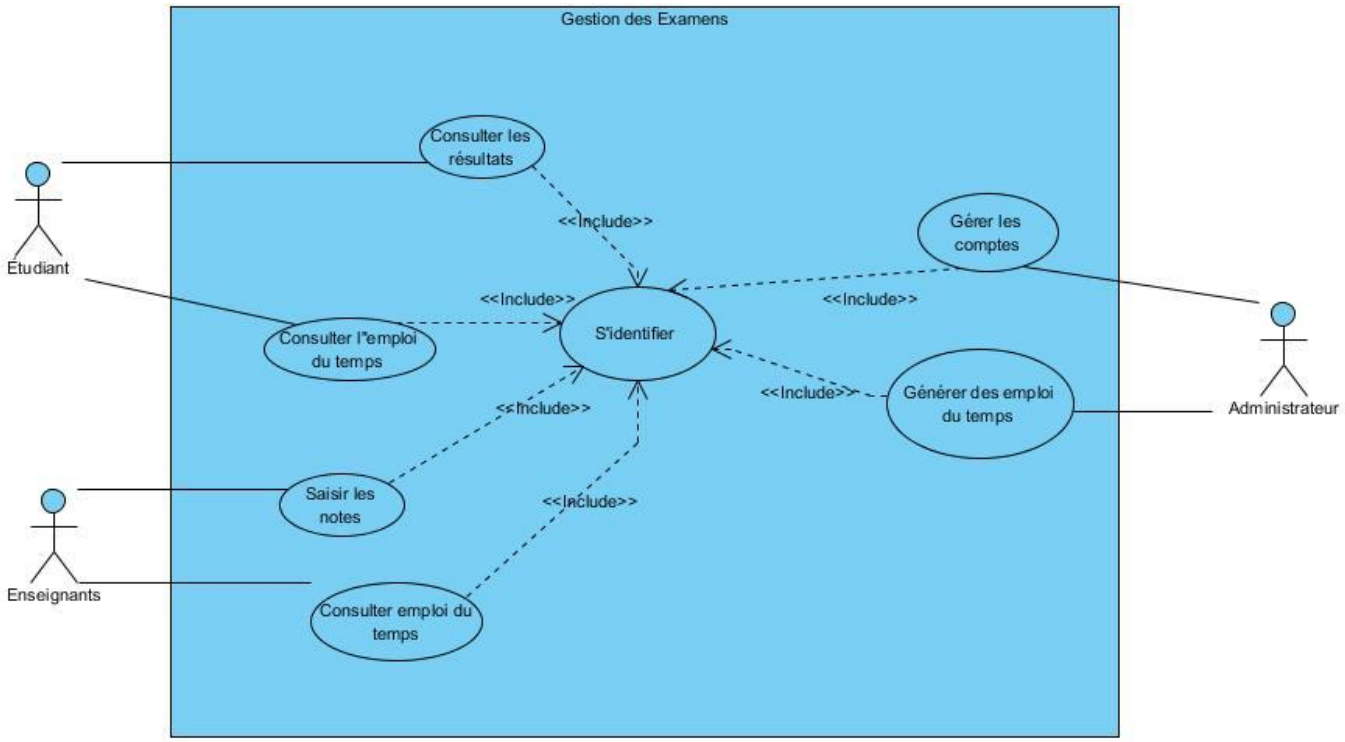
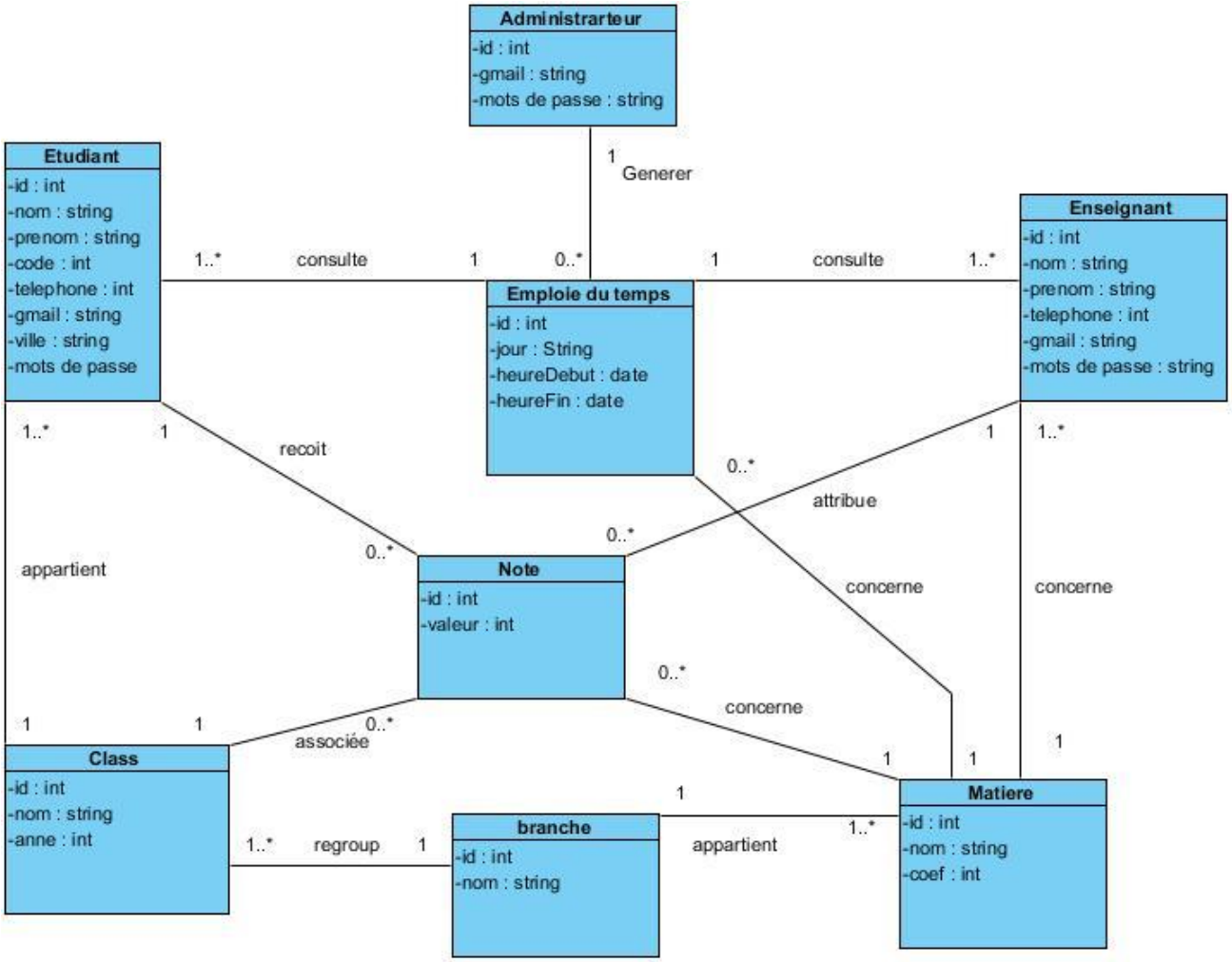


Figure 1 :diagramme de cas d’utilisation générale

Ce diagramme met en évidence les cas d'utilisation du système de gestion des examens, en définissant clairement les actions possibles pour chaque type d'utilisateur (étudiant, enseignant, administrateur) et les relations entre les différentes fonctionnalités, notamment l'accès aux résultats, la gestion des emplois du temps et la saisie des notes.

7.Diagramme de classe :



Ce diagramme de classe définit clairement la structure des entités du Systèmes de Gestion des Examens et les relations entre elles.

8.conception des cas d'utilisations :

Administrateur

1 : Cas d'utilisation (S'identifier) :

-Cas d'utilisation format textuelle :

Titre	S'identifier
Objectif	Permettre à l'utilisateur (étudiant, enseignant ou administrateur) d'accéder à son espace personnel sur la plateforme Parcours.
Acteur principal	Utilisateur (Étudiant / Enseignant / Administrateur)

Préconditions	L'utilisateur dispose d'un compte avec un email (gmail) et un mot de passe valides.
Scénario nominal	1. L'utilisateur accède à l'interface de connexion. 2. Il saisit son adresse gmail dans le champ prévu. 3. Il saisit son mot de passe. 4. Il clique sur le bouton **"connexion"**. 5. Le système vérifie les identifiants dans la base de données. 6. Le système redirige l'utilisateur vers son espace en fonction de son rôle (étudiant, enseignant ou administrateur).
Scénarios alternatifs	-L'utilisateur saisit un gmail ou mot de passe incorrect → Un message d'erreur s'affiche. - La base de données est indisponible → Un message d'erreur système s'affiche.
Actions possibles	-Saisir ses identifiants -Se connecter au système -Être redirigé vers l'interface correspondante
Post-conditions	- L'utilisateur est identifié et connecté à son espace personnel. -Les informations de session sont chargées.

-Diagramme de Séquence (S'identifier) :

Flux

Scénario nominal :

1. **Utilisateur** :
    - 1.1 : Ouvre l'application.
2. **Système** :
    - 1.2 : Affiche l'interface de login (champs *email* et *mot de passe*).
3. **Utilisateur** :
    - 2 : Saisit son email et son mot de passe.
4. **Système** :
    - 2.1 : Vérifie les identifiants dans la base de données.

- 2.2 : Si valides → Charge l'interface correspondant au type d'utilisateur (étudiant, enseignant, admin).

Scénario alternatif :

- **Étape 4** :
  - Si identifiants invalides →
    - 2.4 : Affiche un message d'erreur ("*Identifiants incorrects*").
    - Retour à l'interface de login (étape 1.2).

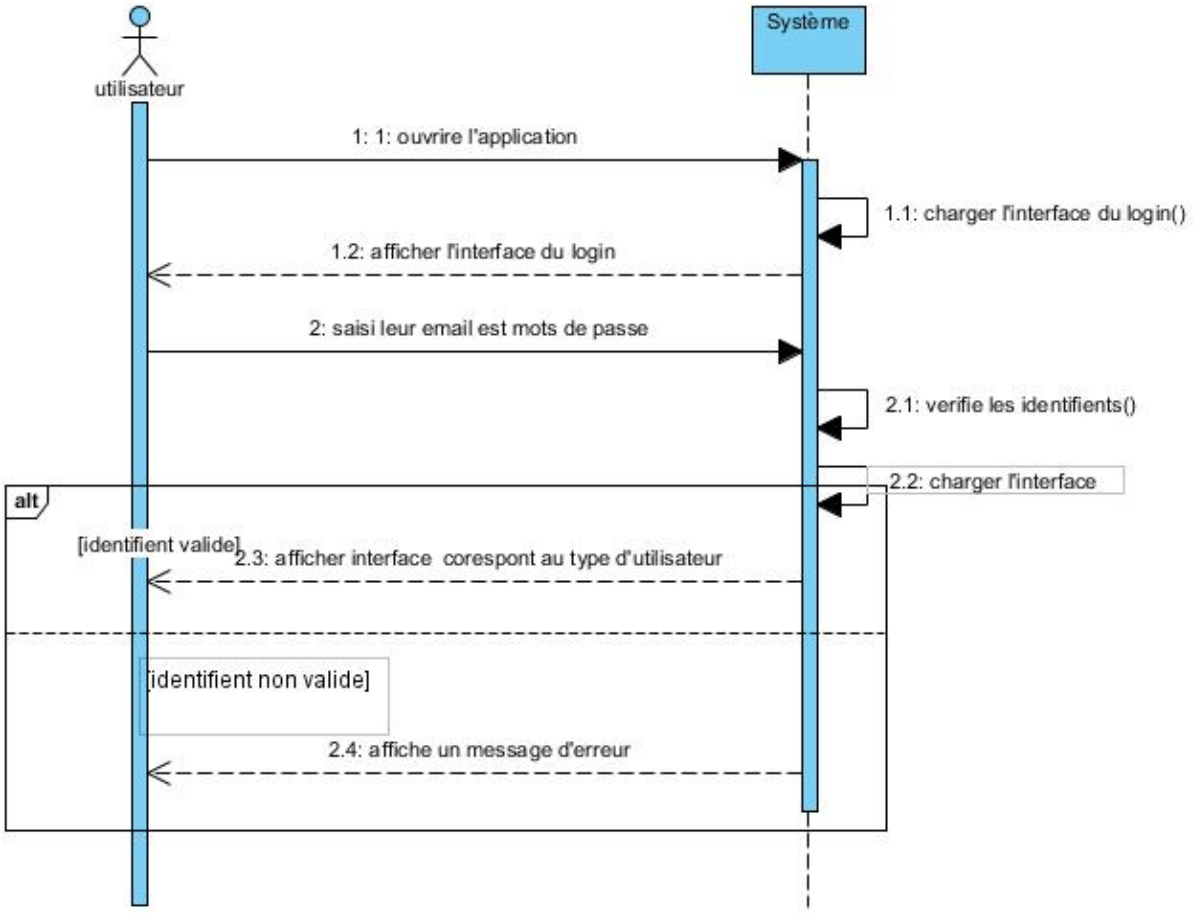


Figure :diagramme de Séquence (S'identifier)

-Diagramme d'activité (S'identifier) :

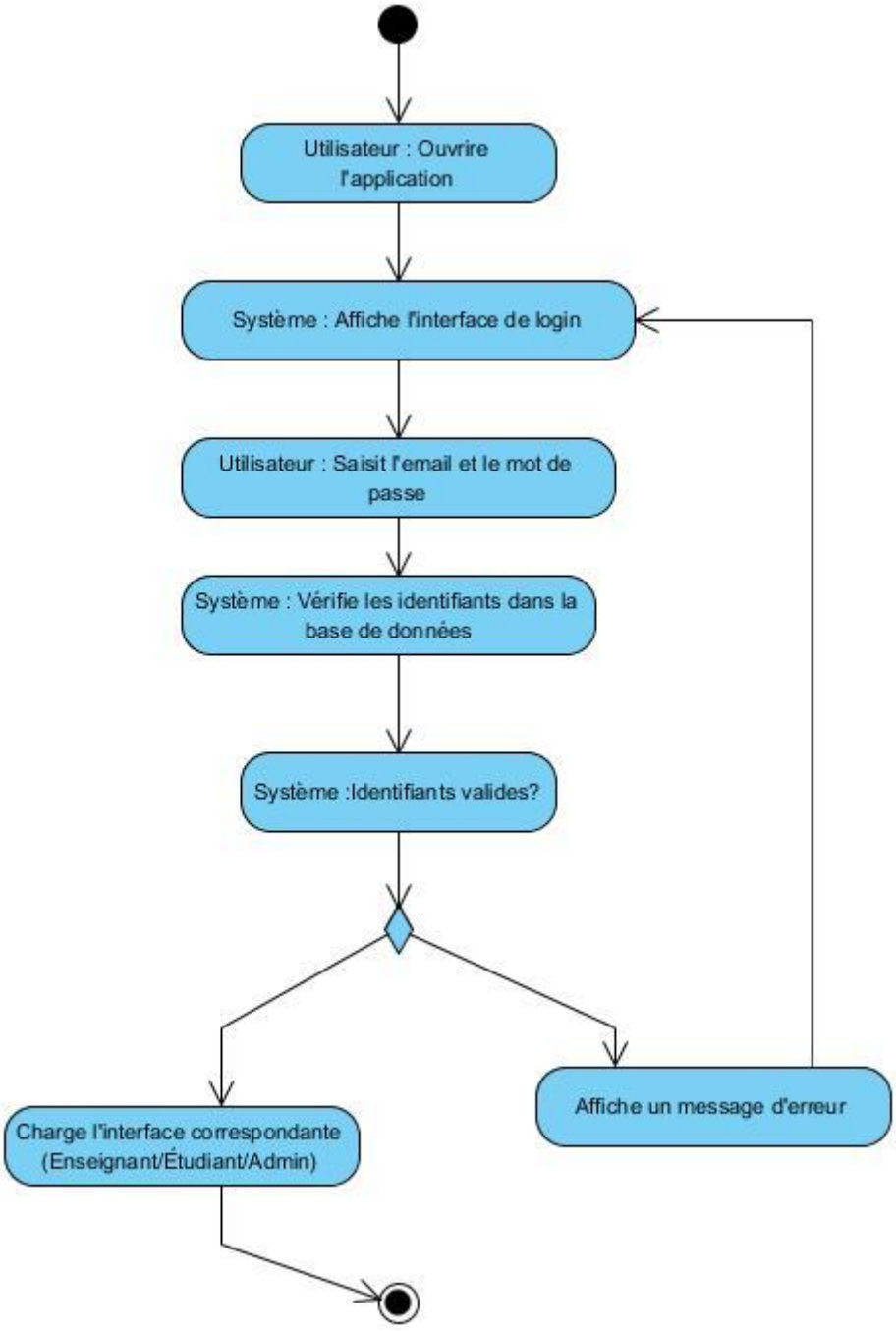


Figure : Diagramme d'activité (S'identifier)

-Diagramme de classes participantes (S'identifier) :

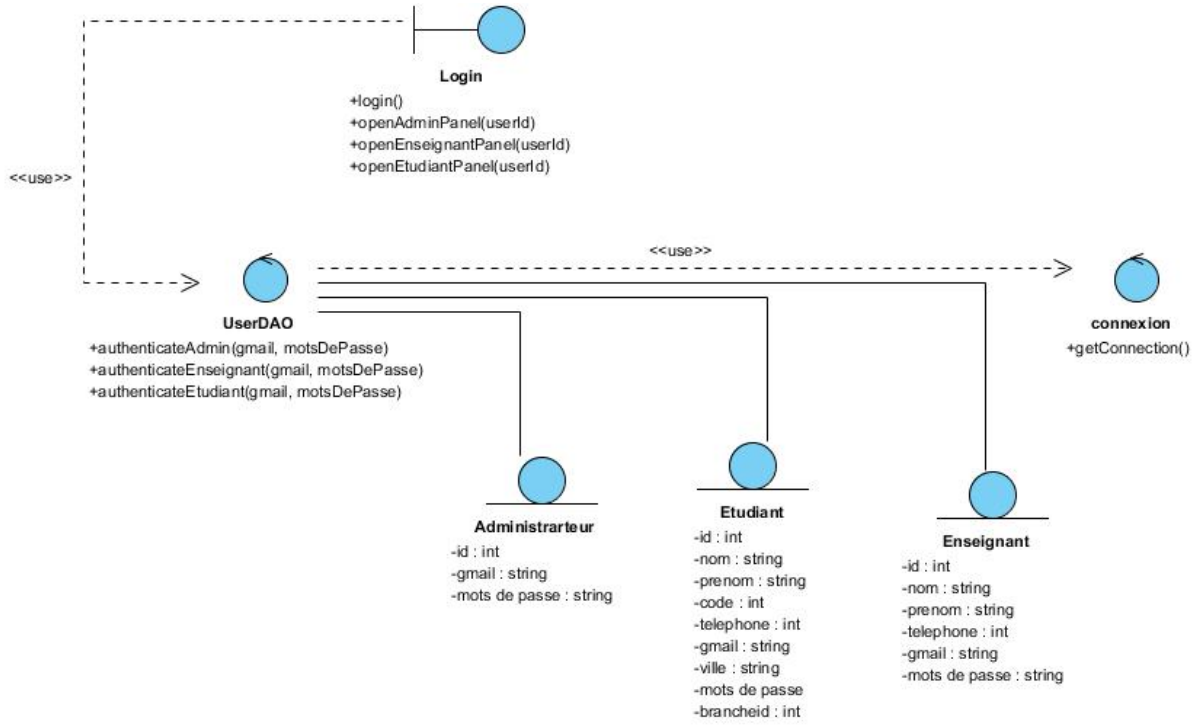


Figure : Diagramme de classes participantes (S'identifier)

-Réalisation (S'identifier) :

Description :

-Objectif : Permettre à l'utilisateur (enseignant, étudiant ou administrateur) de s'authentifier à l'aide de son adresse e-mail (Gmail) et de son mot de passe.

Composants :

- Deux champs de texte pour saisir :
  - le gmail
  - le mot de passe
- Un bouton "connexion" qui déclenche le processus d'authentification.
- Une zone d'accueil personnalisée avec le texte "Bonjour".
- Le logo PARCOURS pour l'identité visuelle de l'application.

Figure : image de l'interface de S'identifier

2 : Cas d'utilisation (Gérer les comptes) :

-Cas d'utilisation format textuelle :

Titre	Créer un compte utilisateur
Objectif	Permettre à l'administrateur de créer des comptes pour les enseignants ou les étudiants dans le système.
Acteur principal	Administrateur
Préconditions	-Être connecté au système en tant qu'administrateur.
Scénario nominal	1. L'administrateur accède à l'interface "Créer Compte". 2. Le système affiche les options "Enseignant" ou "Étudiant". 3. L'administrateur sélectionne le type de compte (Enseignant/Étudiant). 4. Le système affiche un formulaire adapté au type de compte. o Pour un enseignant : **Nom**, **Prénom**, **Gmail**, **Téléphone**, **Mot de passe**. o Pour un étudiant: **Nom**, **Prénom**, **Gmail**, **Téléphone**, **Ville**, **Code**, **Branche** , **matiere**, **Mot de passe**. 5. L'administrateur saisit les informations obligatoires. 6. L'administrateur valide l'enregistrement.
Scénarios alternatifs	-Champs obligatoires non remplis → Le système affiche un message d'erreur

Actions possibles	-Basculer entre les formulaires Enseignant/Étudiant.
Post-conditions	-Le compte est ajouté à la base de données. -L'interface est actualisée

-Diagramme de Séquence (Gérer les comptes) :

Flux

Scénario nominal :

1. **Administrateur** :
    - **1.1** : Clique sur le bouton "Créer un compte".
2. **Système** :
    - **1.1** : Affiche les options de type de compte (*Étudiant* ou *Enseignant*).
3. **Administrateur** :
    - Sélectionne le type de compte (ex: **Étudiant**).
    - **2** : Remplit les champs obligatoires pour l'étudiant (Nom, Prénom, Gmail, Téléphone, etc.).
4. **Système** :
    - **2.1** : Vérifie les données saisies.
    - Si valides → Enregistre le compte en base de données.
    - **2.2** : Affiche un message de confirmation (*"Compte étudiant créé avec succès"*).

Scénario alternatif :

- **Étape 4** :
    - Si erreur (ex: champ manquant, email invalide) →
        - **2.3** : Affiche un message d'erreur (*"Veuillez corriger les champs en rouge"*).
        - Retour à l'étape 2 pour modification.

Cas spécifique : Création d'un compte Enseignant

1. **Administrateur** :
    - Sélectionne *Enseignant* comme type de compte.

- **3** : Remplit les champs spécifiques (Nom, Prénom, Parcours, etc.).

2. **Système** :
    - **3.1** : Vérifie les données.
    - Si valides → **3.2** : Affiche confirmation (*"Compte enseignant créé"*).
    - Si erreur → **3.3** : Affiche message d’erreur.

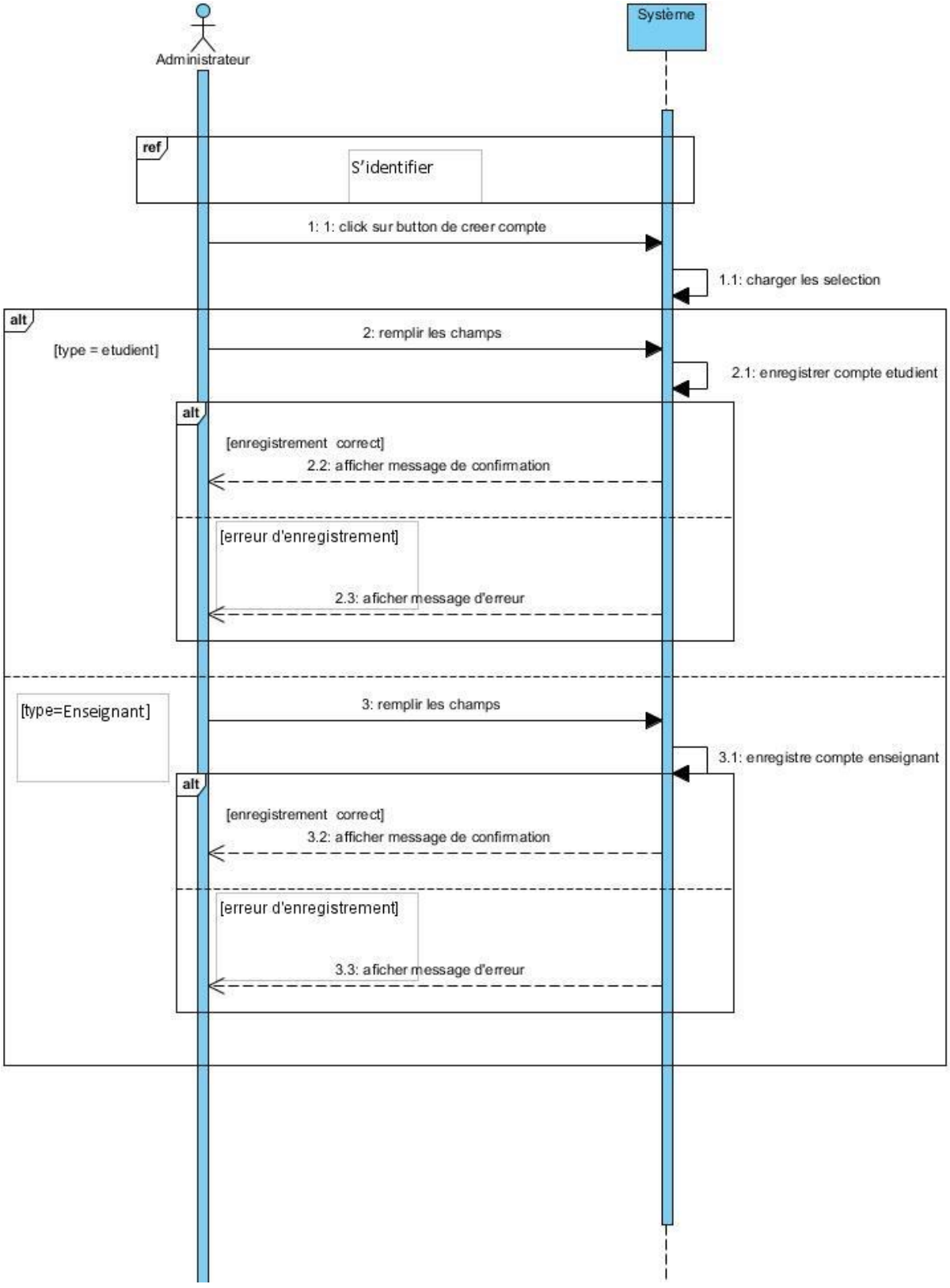


Figure : Diagramme de Séquence (Gérer les comptes)

-Diagramme d'activité (Gérer les comptes) :

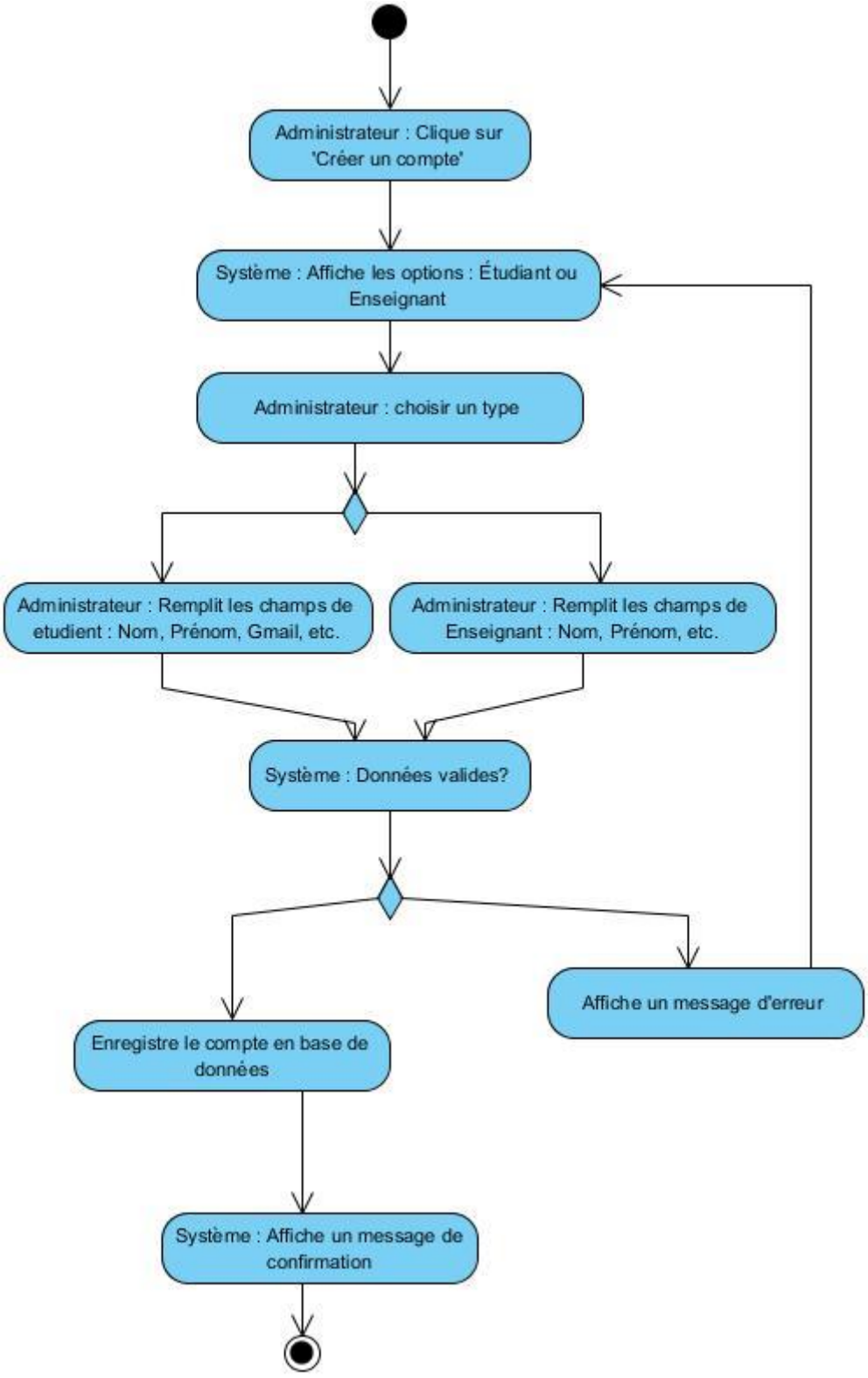


Figure : Diagramme d'activité (Gérer les comptes)

-Diagramme de classes participantes (Gérer les comptes) :

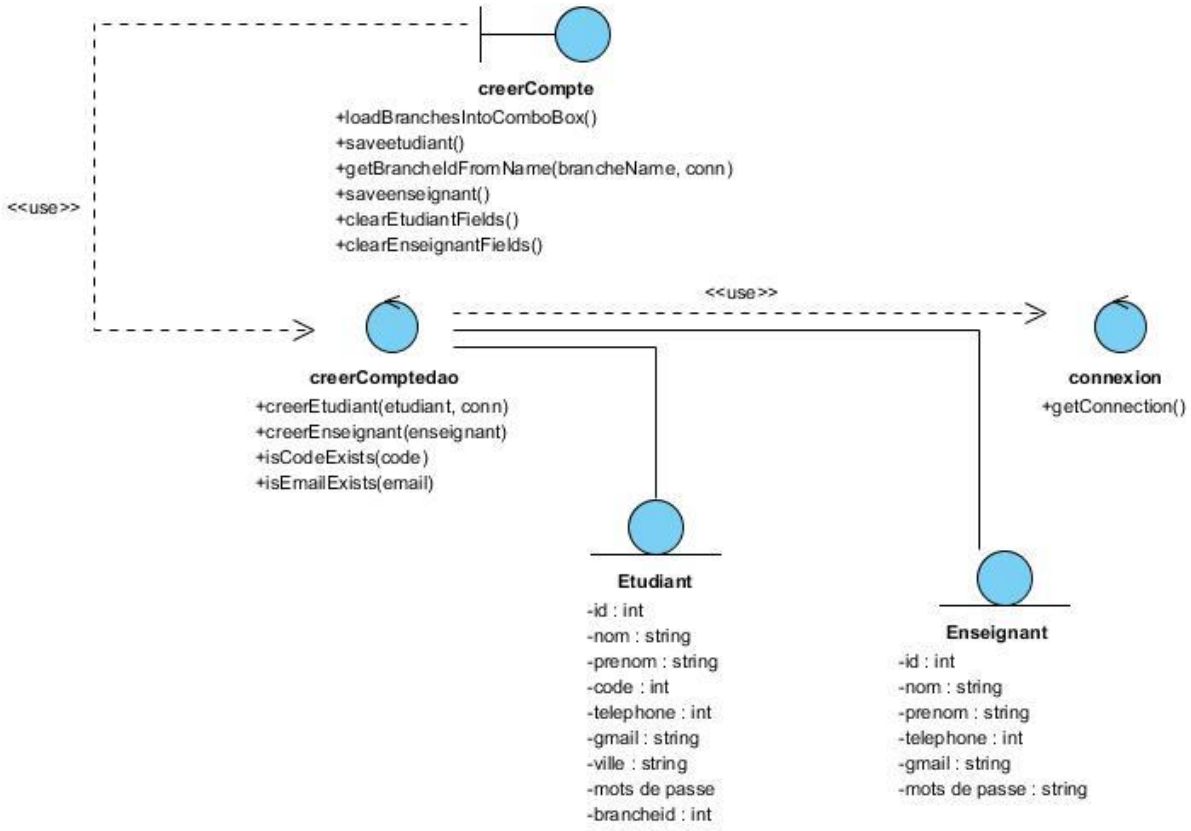


Figure : Diagramme de classes participantes (Gérer les comptes)

-Réalisation (Gérer les comptes) :

Description :

Objectif :
Permettre à un utilisateur (enseignant ou étudiant) de créer un compte personnalisé afin d'accéder aux fonctionnalités de l'application.

1. Interface : Créer un compte Enseignant

Composants :

- Cinq champs de texte permettant de saisir les informations suivantes :
    - Nom
    - Prénom
    - Gmail
    - Téléphone
    - Mot de passe
- Un bouton "enregistrer" pour soumettre les données.
- Un onglet supérieur permettant de basculer entre la création d'un compte enseignant et étudiant.

- Le logo PARCOURS pour renforcer l’identité visuelle de l’application.

Remarques :
Interface simple et claire avec un fond bleu-vert, destinée à faciliter l'inscription des enseignants.

2. Interface : Créer un compte Étudiant

Composants :

- Huit champs de saisie :
    - Nom
    - Prénom
    - Téléphone
    - Ville
    - Gmail
    - Code
    - Mot de passe
    - Branche
- Un bouton "enregistrer" pour valider l'inscription.
- Navigation entre les onglets (étudiant et enseignant).
- Le logo PARCOURS reste visible en haut à droite.

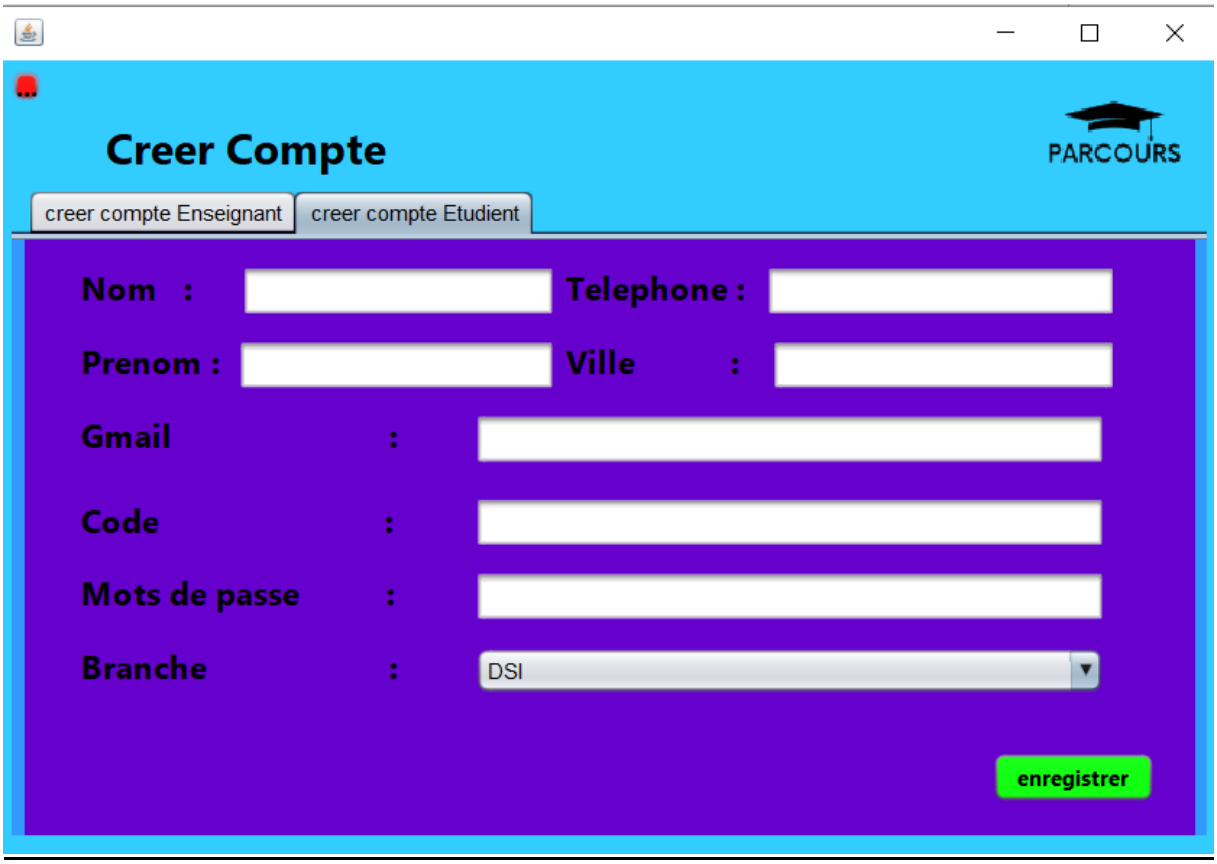


Figure : image de l'interface de creer compte Etudient

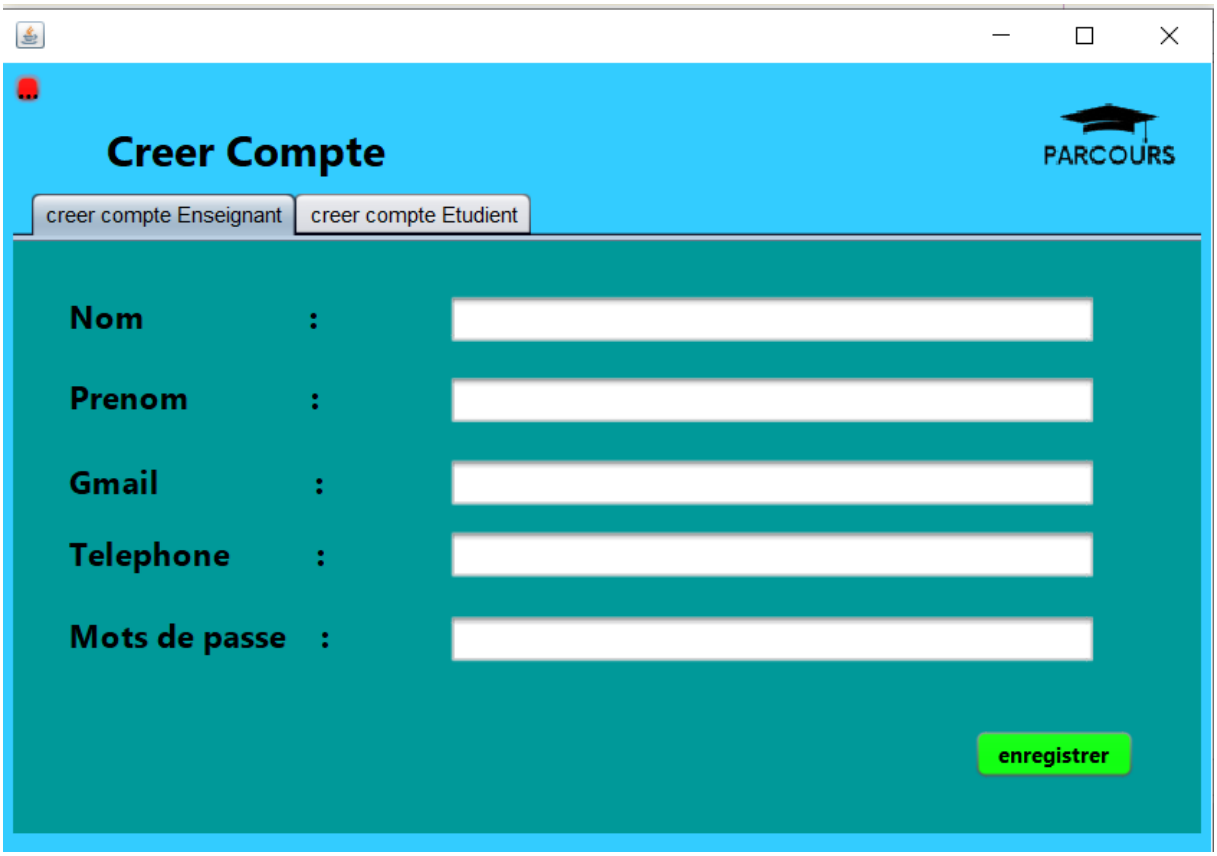


Figure : image de l’interface de creer compte Enseignant

3 : Cas d’utilisation (Générer des emploi du temps) :

-Cas d’utilisation format textuelle :

Titre	Gérer l’emploi du temps
Objectif	Permettre à l’administrateur de planifier les exames pour une classe.
Acteur principal	Administrateur
Préconditions	- Être connecté au système -Avoir des enseignants et matières enregistrés.
Scénario nominal	1. L’administrateur accède à l’interface "Créer Emploi". 2. Le système affiche les champs : **Classe**, **Prof 1**, **Prof 2**, **Matière**, **Jour**, **Heure Début**, **Heure Fin**, matières. 3. L’administrateur saisit les informations. 4. L’administrateur valide l’enregistrement.
Scénarios alternatifs	-Champs obligatoires non remplis → Le système affiche un message d’erreur.
Actions possibles	- Aucune Actions possibles
Post-conditions	- L’emploi du temps est mis à jour dans la base de données.

-Diagramme de Séquence (Générer des emploi du temps) :

Flux

1. **Administrateur** :
    - 1.1 : Clique sur le bouton "Créer emploi du temps".
2. **Système** :
    - 3.1 : Charge les champs pré-remplis (combo-box : classes, professeurs, matières).
3. **Administrateur** :
    - 2 : Remplit les champs (classe, professeurs, matière, jour, horaires).
    - 4.1 : Valide l’enregistrement.

4. **Système** :
    - Vérifie les données.
    - Si valides → Enregistre l'emploi du temps en base de données.
    - Affiche un message de confirmation (2.2).
5. **Scénario alternatif** :
    - Si erreur (ex: conflit horaire) → Affiche un message d'erreur (2.3).

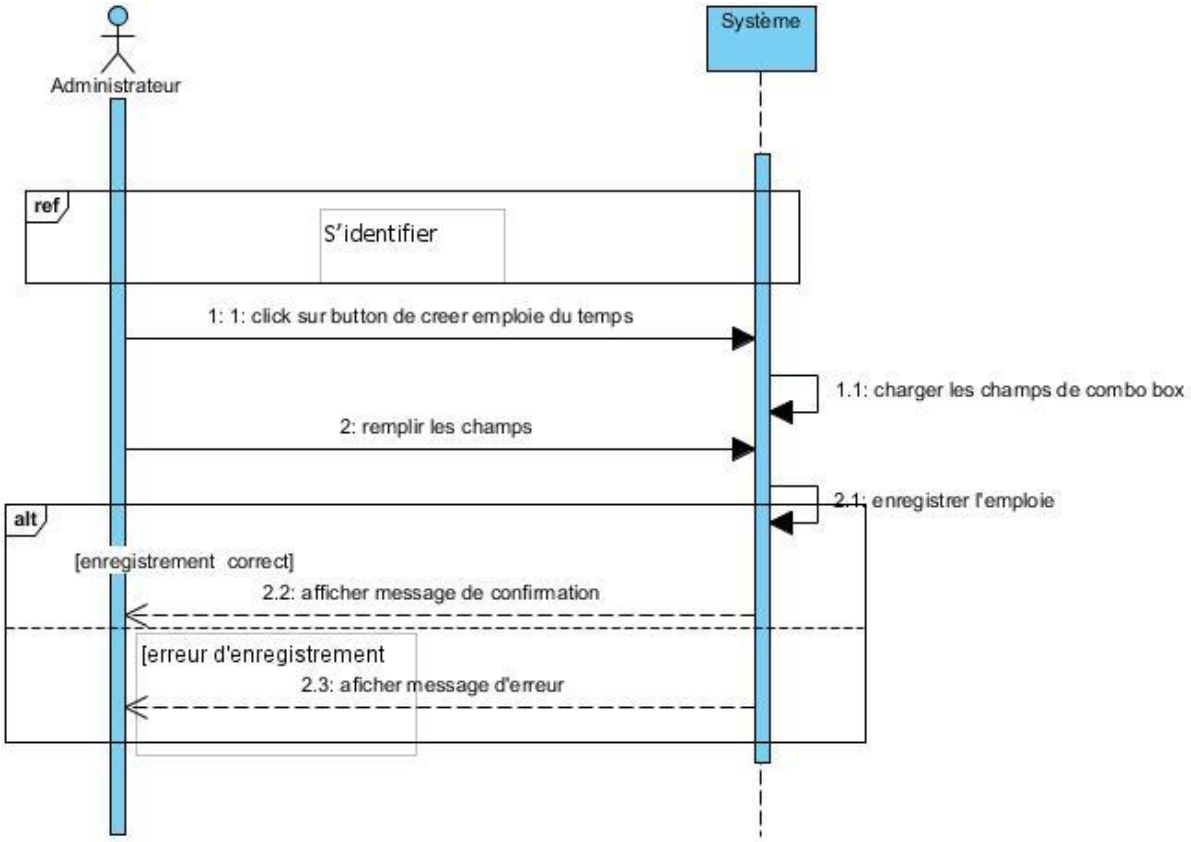


Figure : Diagramme de Séquence (Générer des emploi du temps)

-Diagramme d'activité (Générer des emploi du temps) :

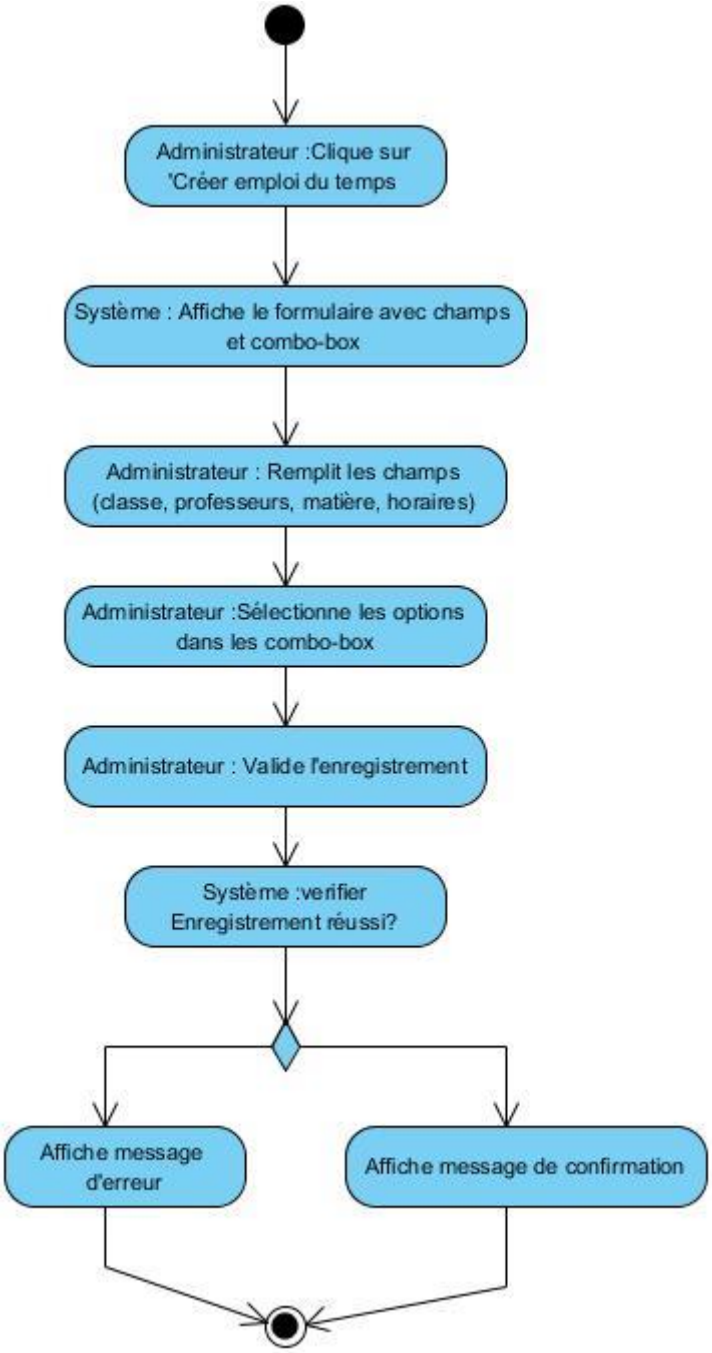


Figure : Diagramme d'activité (Générer des emploi du temps)

-Diagramme de classes participantes (Générer des emploi du temps) :

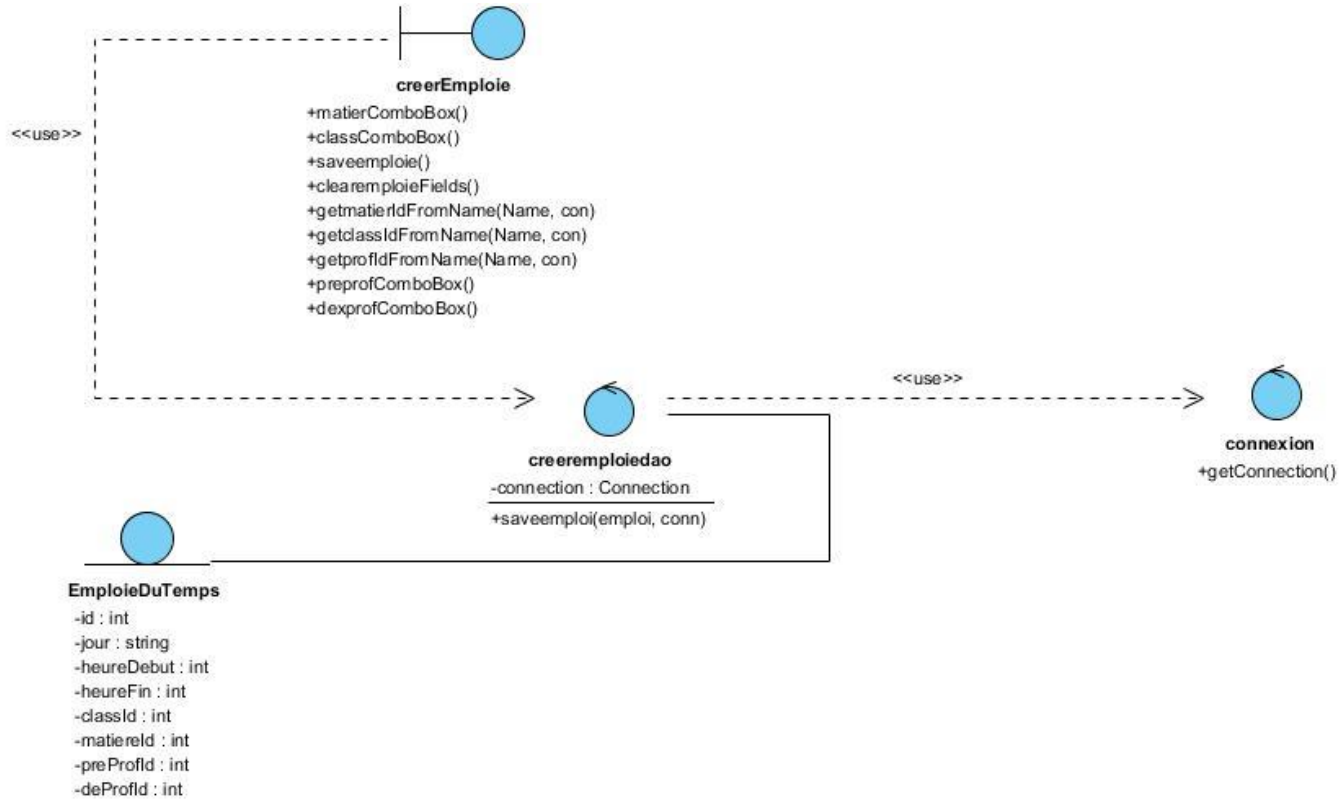


Figure : Diagramme de classes participantes (Générer des emploi du temps)

-Réalisation (Générer des emploi du temps) :

Description :

Objectif :
Permettre à l'administrateur de créer un emploi du temps pour une classe spécifique en associant des enseignants et des matières à des plages horaires.

Composants :

- Menu déroulant pour sélectionner :
    - La classe concernée
    - Le premier professeur
    - Le second professeur
    - La matière
    - Le jour
- Deux champs de texte :
    - Temps de début
    - Temps de fin
- Un bouton "enregistrer" pour enregistrer l'horaire.

- Le logo PARCOURS et une interface bleue épurée.

Figure : image de l'interface de creer des emploi du temps

Enseignant

1 : Cas d'utilisation (Saisir les notes) :

-Cas d'utilisation format textuelle :

Titre	Saisir les notes
Objectif	Permettre à un enseignant de saisir les notes des étudiants pour une matière donnée.
Acteur principal	Enseignant
Préconditions	-L'enseignant est connecté à son compte. -Les matières enseignées sont enregistrées dans le système.

	-Les étudiants sont inscrits avec un code valide.
Scénario nominal	1. L'enseignant accède à l'interface de saisie des notes. 2. Il saisit le **code** de l'étudiant. 3. Il saisit la **note** obtenue. 4. Il sélectionne la **matière** correspondante dans la liste déroulante. 5. Il clique sur le bouton **"SAVE"**. 6. Le système enregistre la note dans la base de données. 7. Un message de confirmation est affiché.
Scénarios alternatifs	-Le code de l'étudiant n'existe pas → un message d'erreur s'affiche. -La note saisie est invalide (ex. : supérieure à 20 ou non numérique) → le système bloque l'enregistrement. -La matière n'a pas été sélectionnée → un message d'avertissement s'affiche.
Actions possibles	-Saisir les notes -Sélectionner une matière -Enregistrer une note
Post-conditions	-La note est sauvegardée et associée à l'étudiant et à la matière correspondante dans la base de données.

-Diagramme de Séquence (Saisir les notes) :

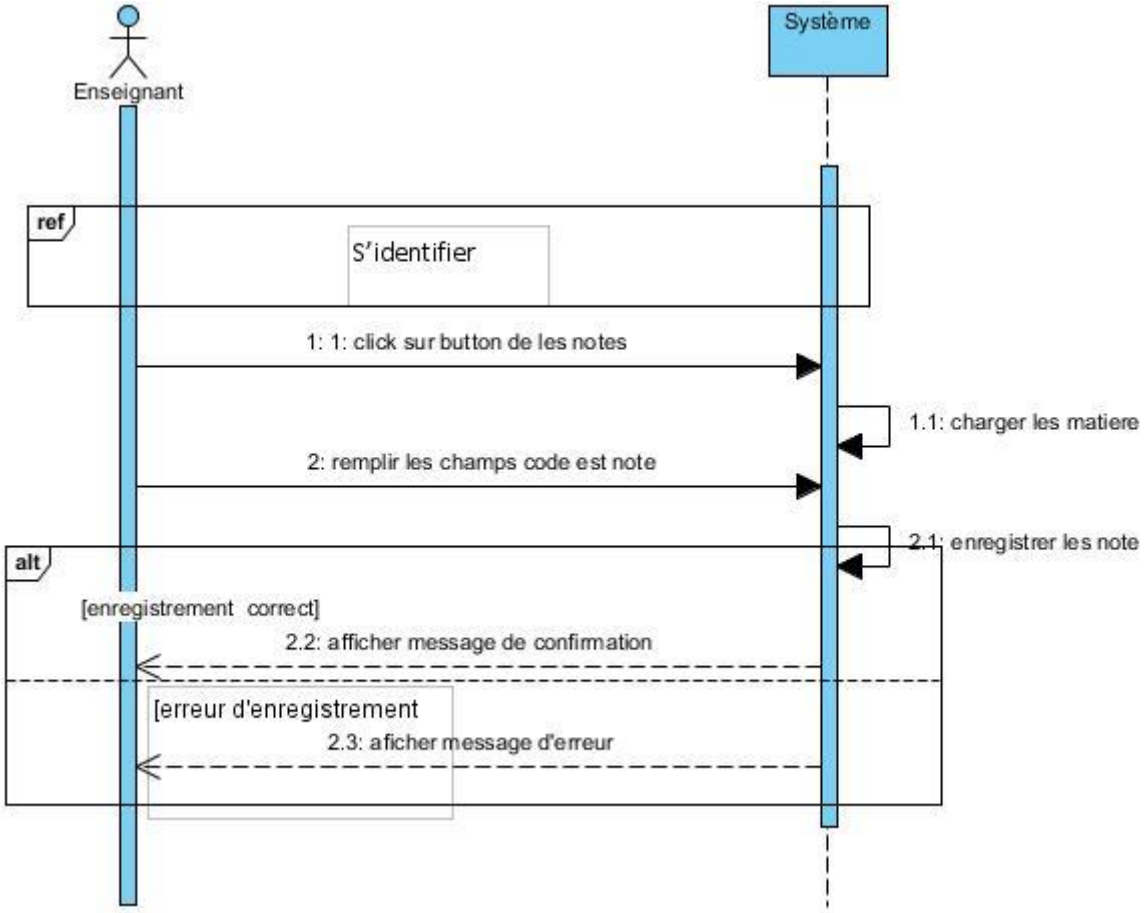


Figure : Diagramme de Séquence (Saisir les notes)

-Diagramme d'activité (Saisir les notes) :

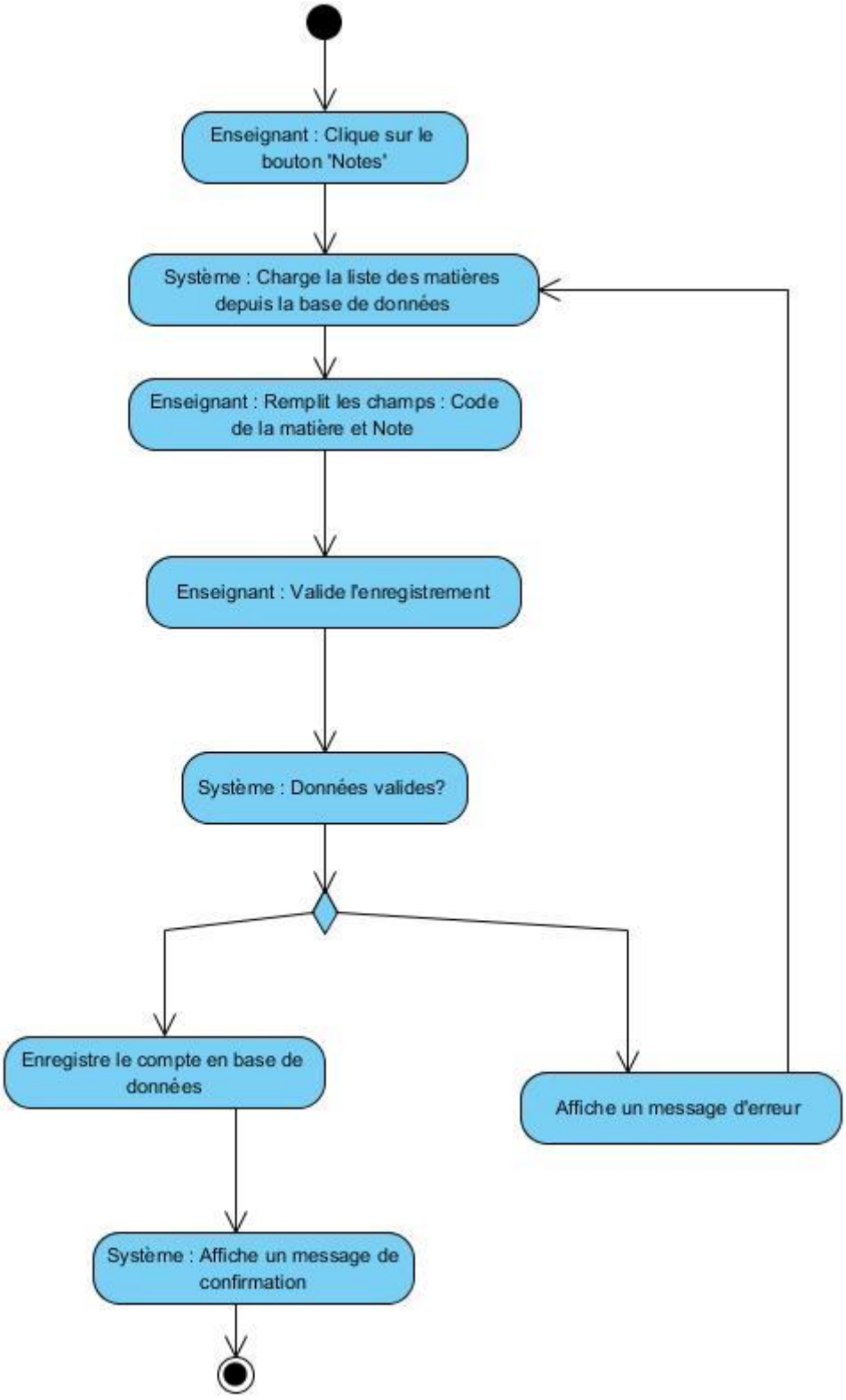


Figure : Diagramme d'activité (Saisir les notes)

-Diagramme de classes participantes (Saisir les notes) :

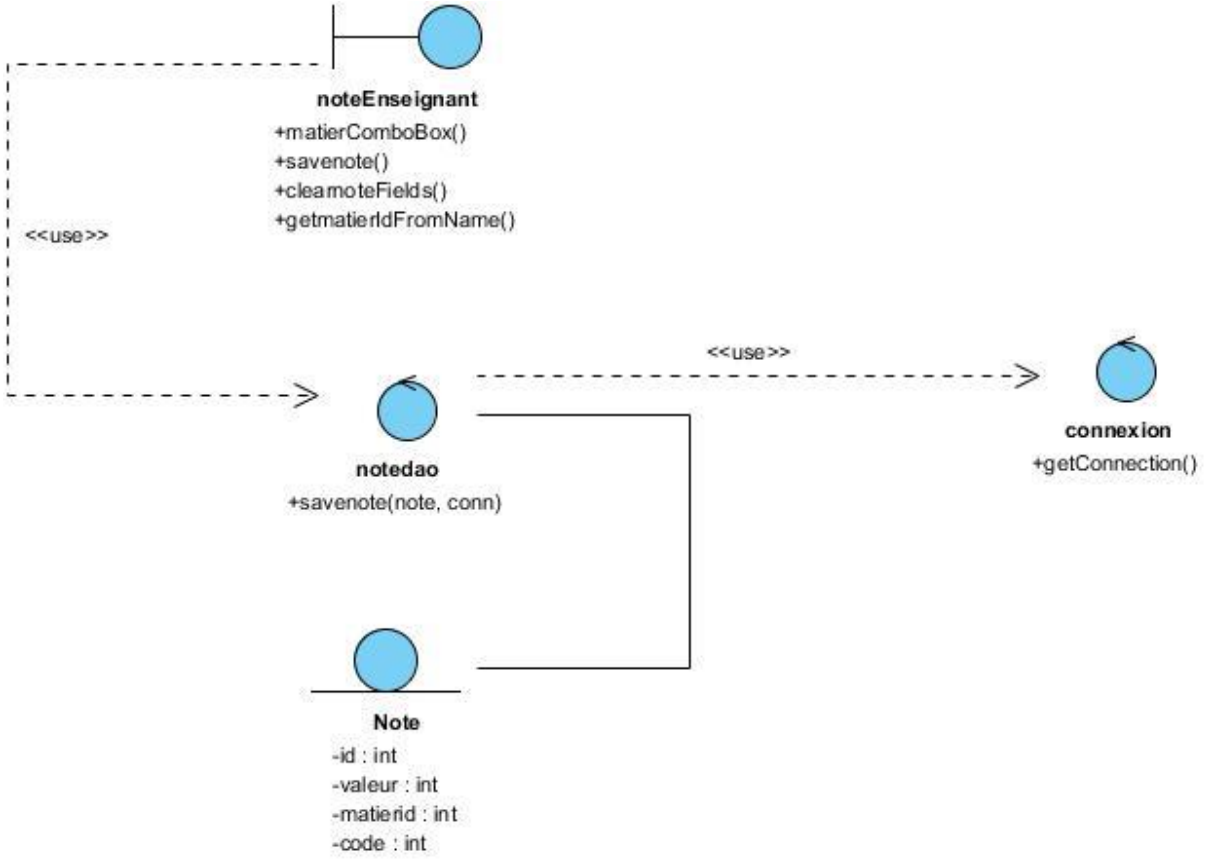


Figure : Diagramme de classes participantes (Saisir les notes)

-Réalisation (Saisir les notes) :

Description :

Objectif :

Permettre à un enseignant de saisir les notes des étudiants pour une matière donnée.

Composants :

- Champ texte pour saisir le code (identifiant de l'étudiant).
- Champ texte pour saisir la note.
- ComboBox pour choisir la matière.
- Bouton "SAVE" pour enregistrer la note.
- Logo PARCOURS présent dans le coin supérieur droit pour une cohérence visuelle.

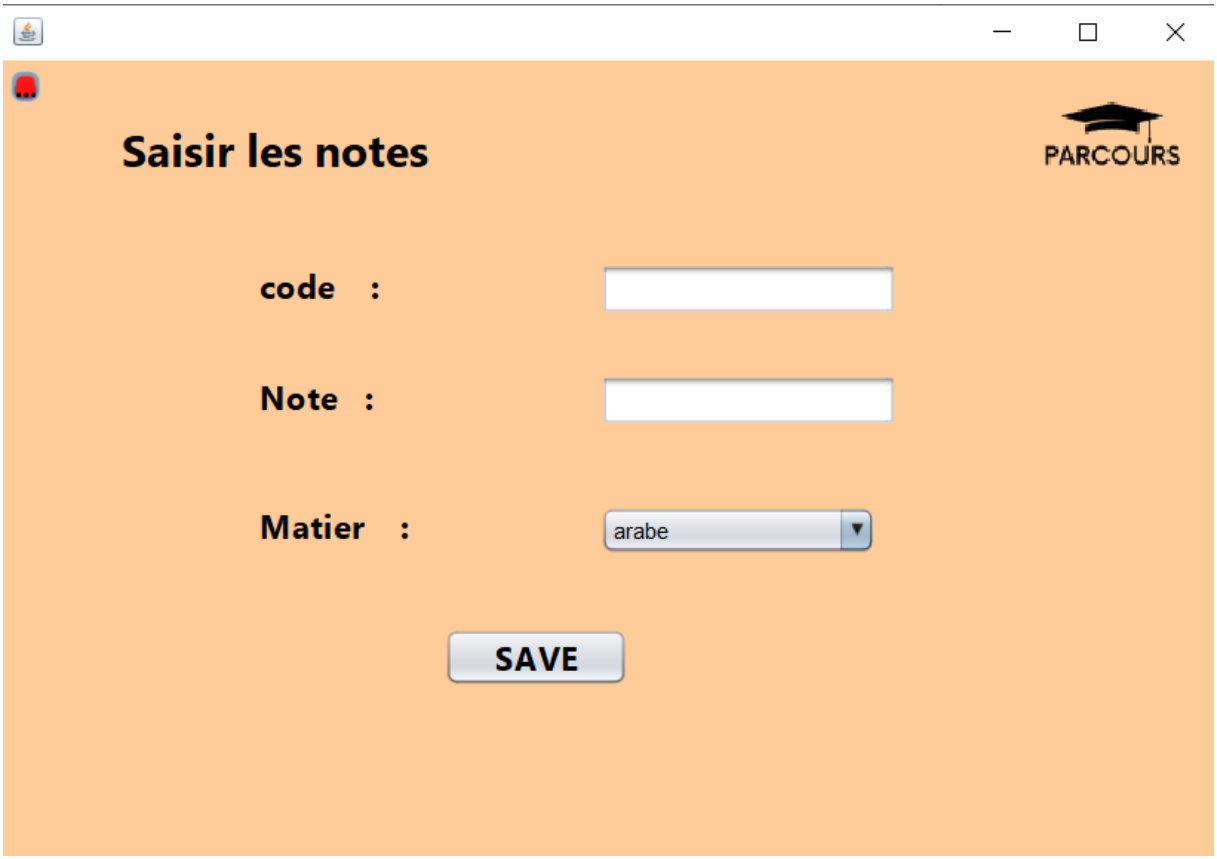


Figure : image de l'interface de Saisir les notes

2 : Cas d'utilisation (Consulter leur emploi du temps) :

-Cas d'utilisation format textuelle :

Titre	Consulter l'emploi du temps
Objectif	Permettre aux Enseignant de consulter leur planning.
Acteur principal	Enseignant
Préconditions	- Être connecté au système
Scénario nominal	1. L'Enseignant accède à l'interface "Emploi du Temps". 2. Le système affiche un tableau avec les cours (Jour, Heures, Matière,Classe).
Scénarios alternatifs	- Aucune Emploi du Temps dans la base de données
Actions possibles	- Aucune Actions possible

Post-conditions	-Aucune modification de la base de données.

-Diagramme de Séquence (Consulter leur emploi du temps) :

Flux

1. **Enseignant** :
    - 1.1 : Clique sur le bouton "Emploi du temps des examens".
2. **Système** :
    - 1.1 : Charge les données des examens (dates, salles, matières).
    - 1.2 : Affiche l'emploi du temps dans un tableau ou calendrier.
3. **Scénario alternatif** :
    - Si données manquantes → Affiche un message d'erreur.

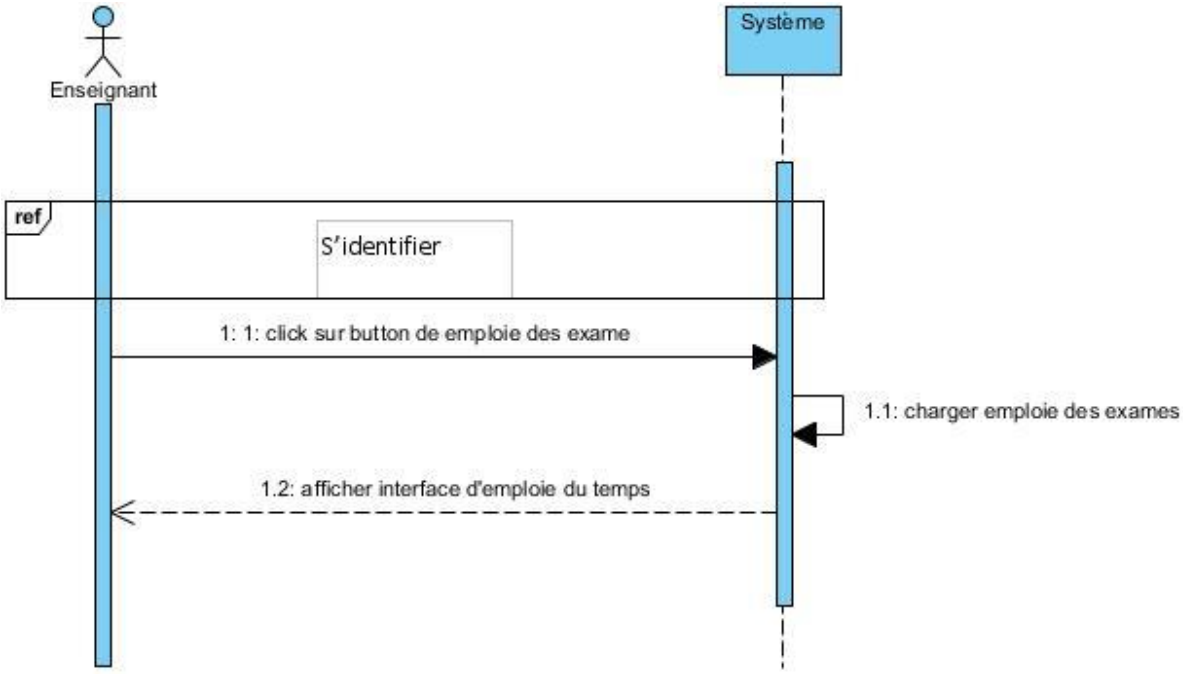


Figure : Diagramme de Séquence (Consulter leur emploi du temps)

-Diagramme d'activité (Consulter leur emploi du temps) :

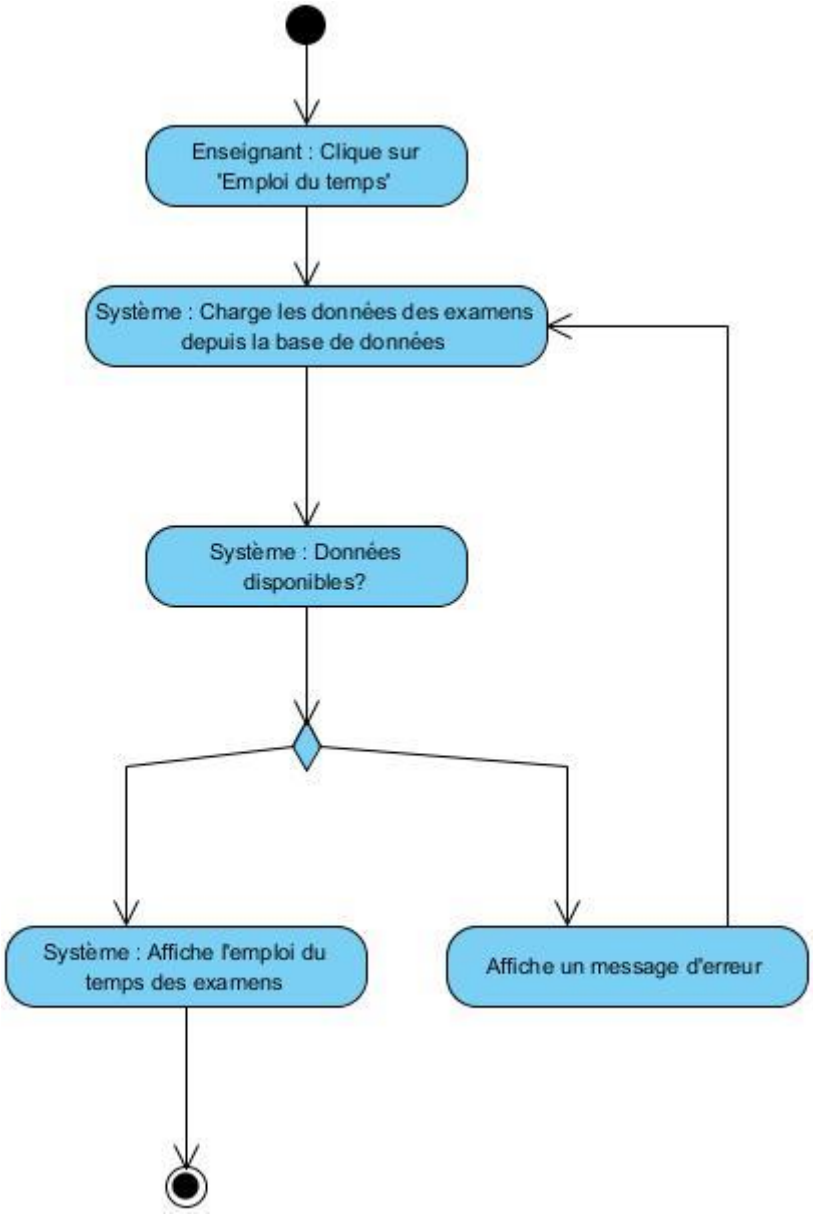


Figure : Diagramme d'activité (Consulter leur emploi du temps)

-Diagramme de classes participantes (Consulter leur emploi du temps) :

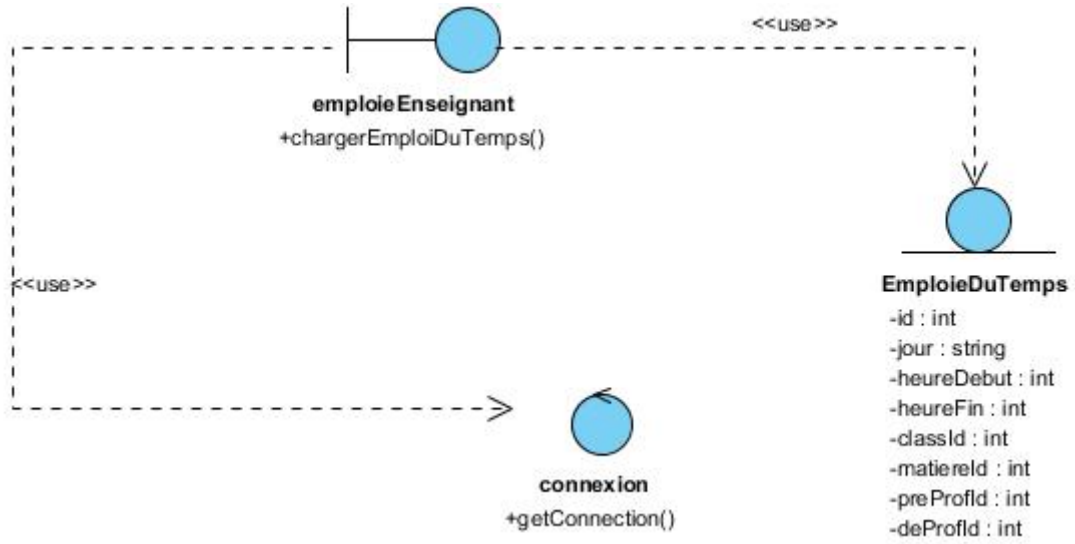


Figure : Diagramme de classes participantes (Consulter leur emploi du temps)

-Réalisation (Consulter leur emploi du temps) :

Figure : image de l'interface de emploi du temps

Étudiant

1 : Cas d’utilisation (Consulter leur résultats) :

-Cas d’utilisation format textuelle :

Titre	Consulter les notes
Objectif	Permettre à un étudiant de visualiser ses résultats obtenus dans chaque matière.
Acteur principal	Étudiant
Préconditions	-L’étudiant est connecté à son compte. -Des notes ont été préalablement enregistrées pour cet étudiant.
Scénario nominal	1-L’étudiant accède à l’interface de consultation des notes. 2-Le système récupère automatiquement les notes associées à son code depuis la base de données. 3-Les notes sont affichées sous forme de tableau (matière / note). 4-L’étudiant peut consulter librement ses résultats.
Scénarios alternatifs	-Aucune note n’a encore été saisie pour cet étudiant → un tableau vide - Problème de connexion à la base de données → une erreur s’affiche.
Actions possibles	- Visualiser la liste des notes
Post-conditions	-L’étudiant prend connaissance de ses résultats dans les différentes matières.

-Diagramme de Séquence (Consulter leur résultats) :

Flux

1. **Étudiant** :
    - 1.1 : Clique sur le bouton "Notes".
2. **Système** :
    - 1.1 : Récupère les notes depuis la base de données.
    - 1.2 : Affiche les notes dans un tableau (matière, note, date).
3. **Scénario alternatif** :
    - Si notes non disponibles → Affiche un message d’erreur.

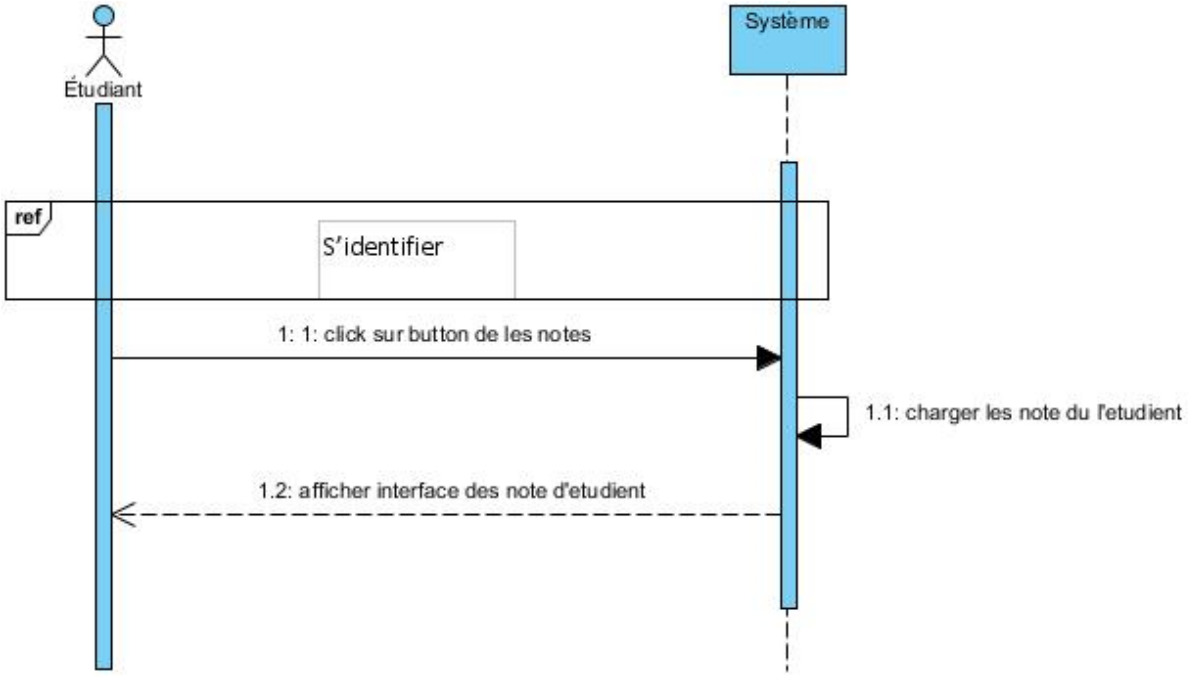


Figure : Diagramme de Séquence (Consulter leur résultats)

-Diagramme d'activité (Consulter leur résultats) :

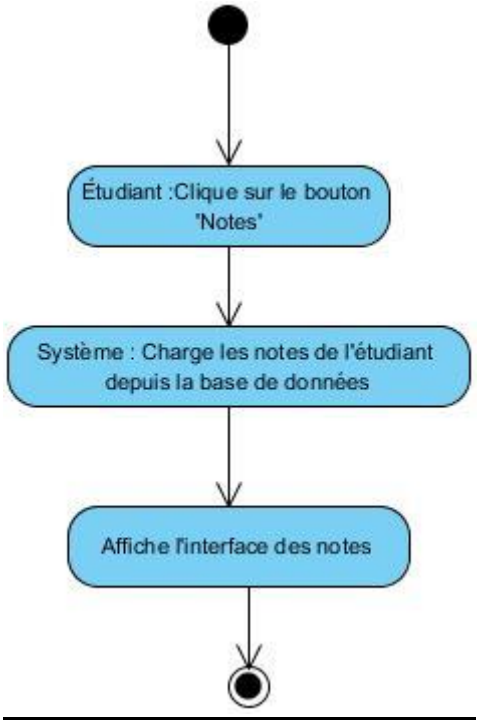


Figure : Diagramme d'activité (Consulter leur résultats)

-Diagramme de classes participantes (Consulter leur résultats) :

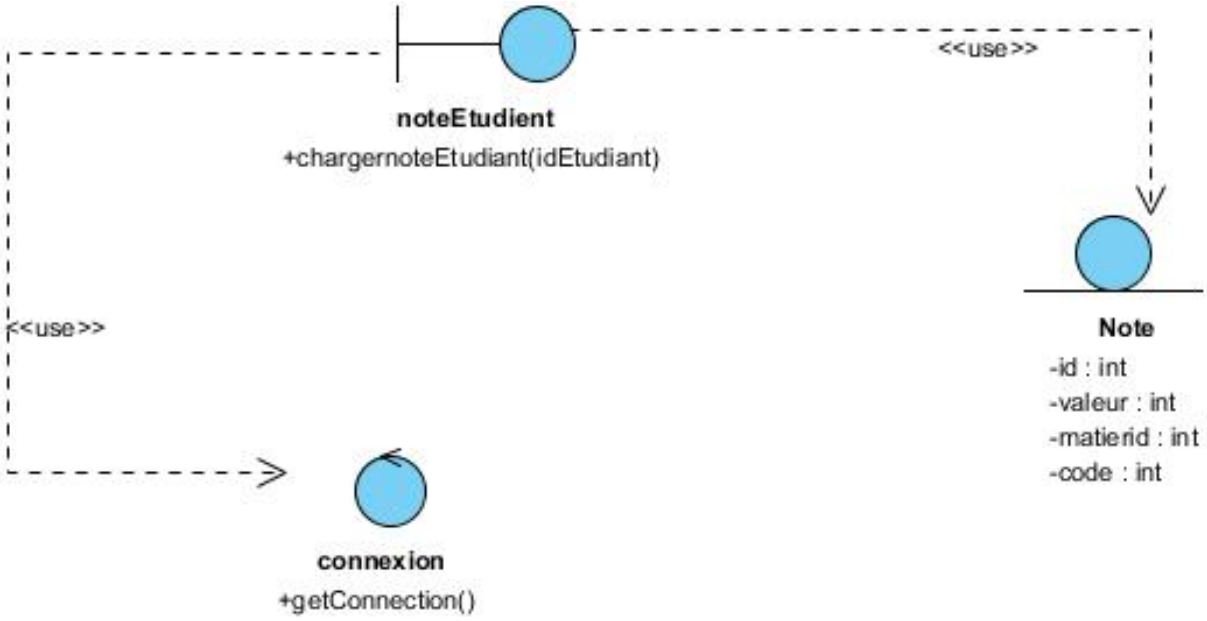


Figure : Diagramme de classes participantes (Consulter leur résultats)

-Réalisation (Consulter leur résultats) :

Description :

Objectif :

Permettre à un étudiant de consulter l'ensemble de ses notes par matière.

Composants :

- Tableau affichant :
    - les matières
    - les notes correspondantes
- Titre "Voir les notes" en gras.
- Logo PARCOURS visible en haut à droite pour la cohérence graphique.

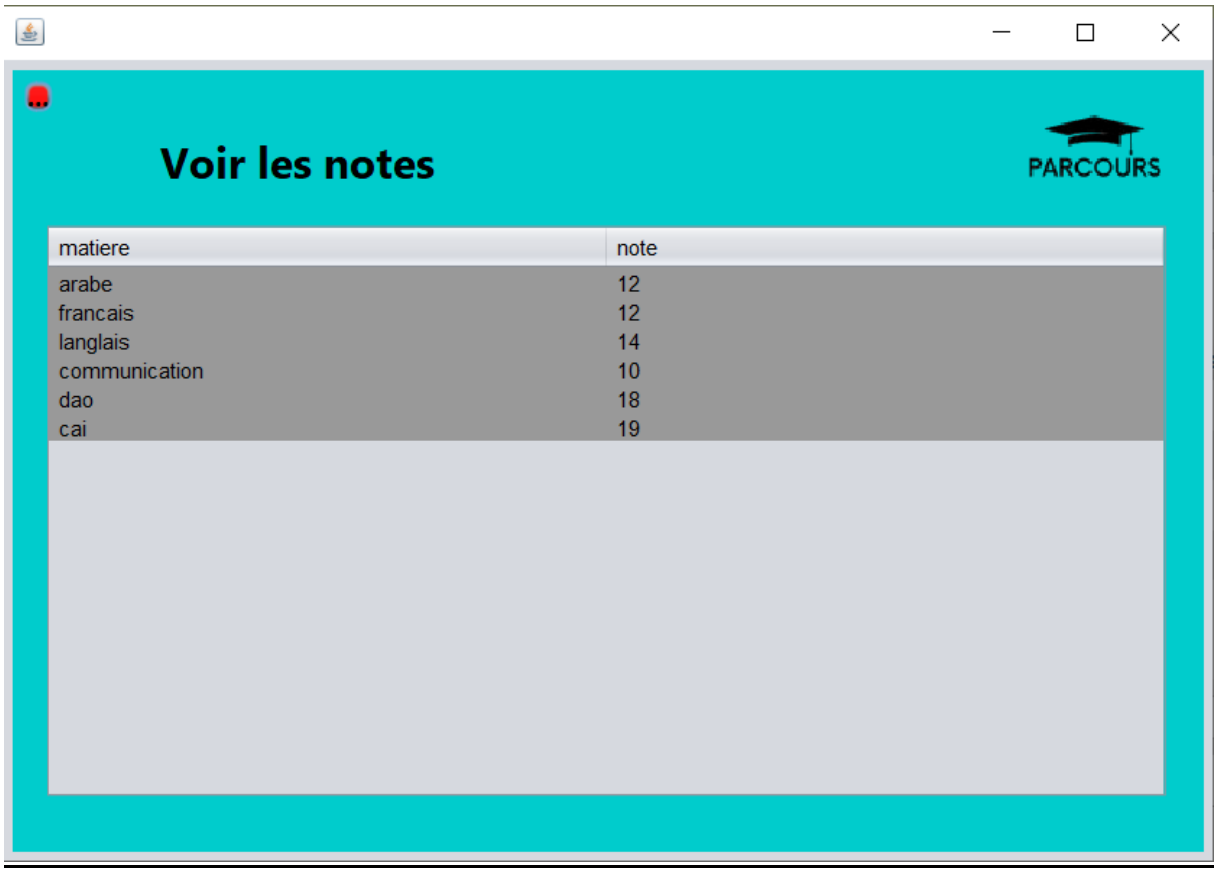


Figure : image de l'interface de Consulter les résultats

2 : Cas d'utilisation (Consulter leur emploi du temps) :

-Cas d'utilisation format textuelle :

Titre	Consulter l'emploi du temps
Objectif	Permettre aux étudiants de consulter leur planning.
Acteur principal	Étudiant
Préconditions	- Être connecté au système
Scénario nominal	3. L'étudiant accède à l'interface "Emploi du Temps". 4. Le système affiche un tableau avec les cours (Jour, Heures, Matière).
Scénarios alternatifs	- Aucune Emploi du Temps dans la base de données
Actions possibles	- Aucune Actions possible

Post-conditions	-Aucune modification de la base de données.

-Diagramme de Séquence (Consulter leur emploi du temps) :

Flux

1. Étudiant :
    - 1.1 : Clique sur le bouton "Emploi du temps des examens".
2. Système :
    - 1.1 : Charge les données des examens.
    - 1.2 : Affiche l'emploi du temps (simplifié, sans détails de classe).
3. Scénario alternatif :
    - Si données manquantes → Affiche un message d'erreur.

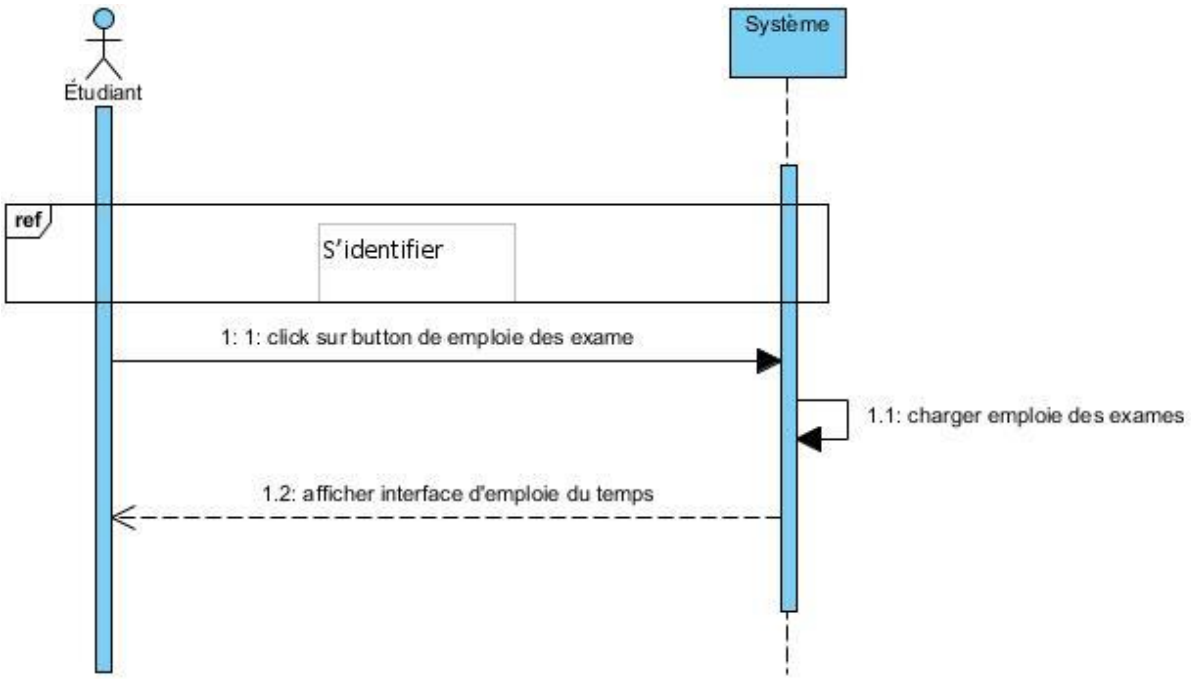


Figure : Diagramme de Séquence (Consulter leur emploi du temps)

-Diagramme d'activité (Consulter leur emploi du temps) :

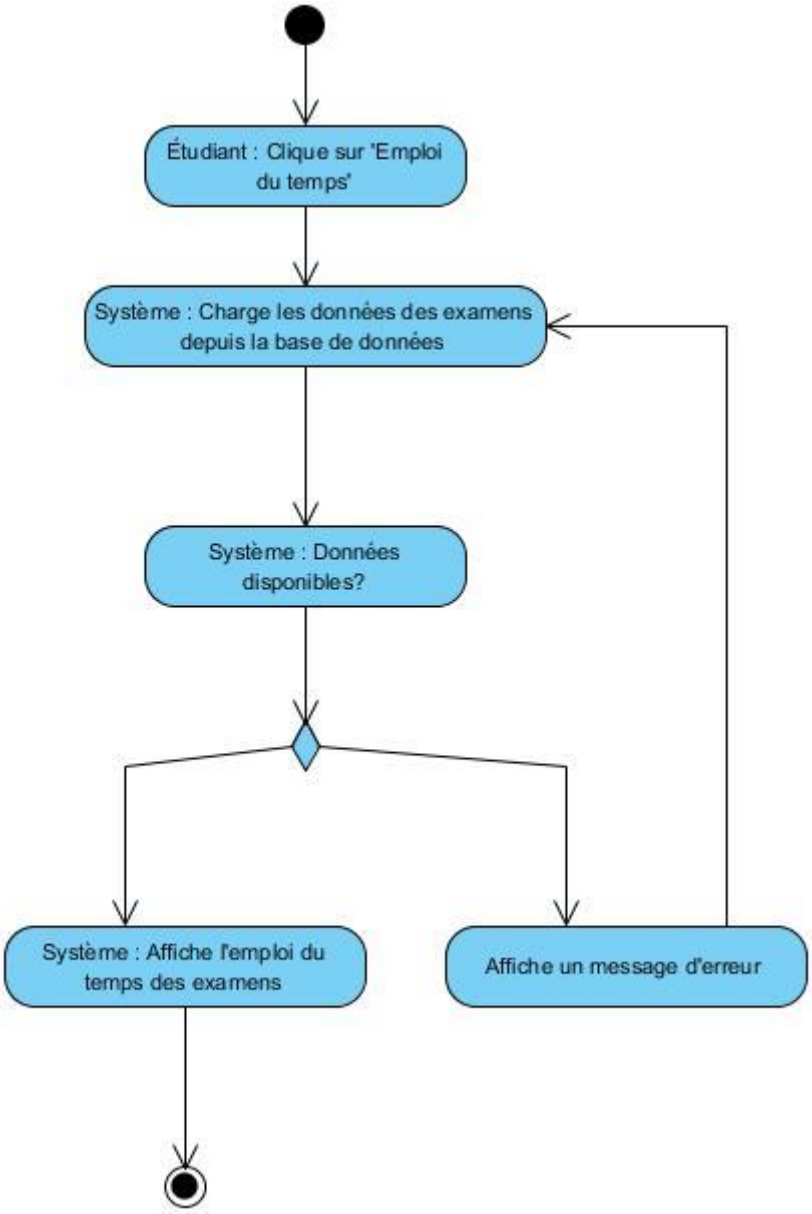


Figure : Diagramme d'activité (Consulter leur emploi du temps)

-Diagramme de classes participantes (Consulter leur emploi du temps) :

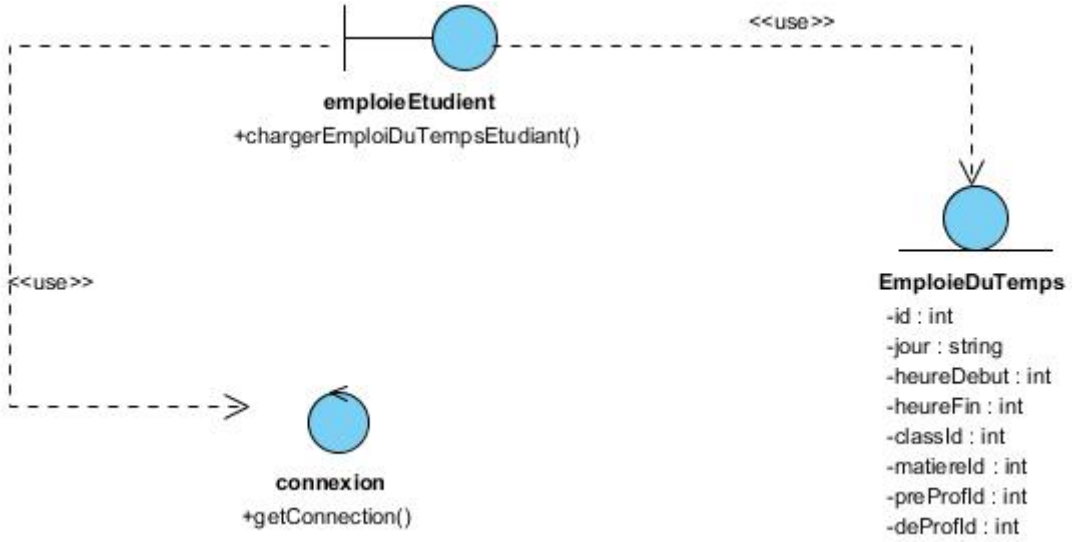


Figure : Diagramme de classes participantes (Consulter leur emploi du temps)

-Réalisation (Consulter leur emploi du temps) :

Description :

Objectif :

Permettre aux étudiants de visualiser leurs emploi du temps.

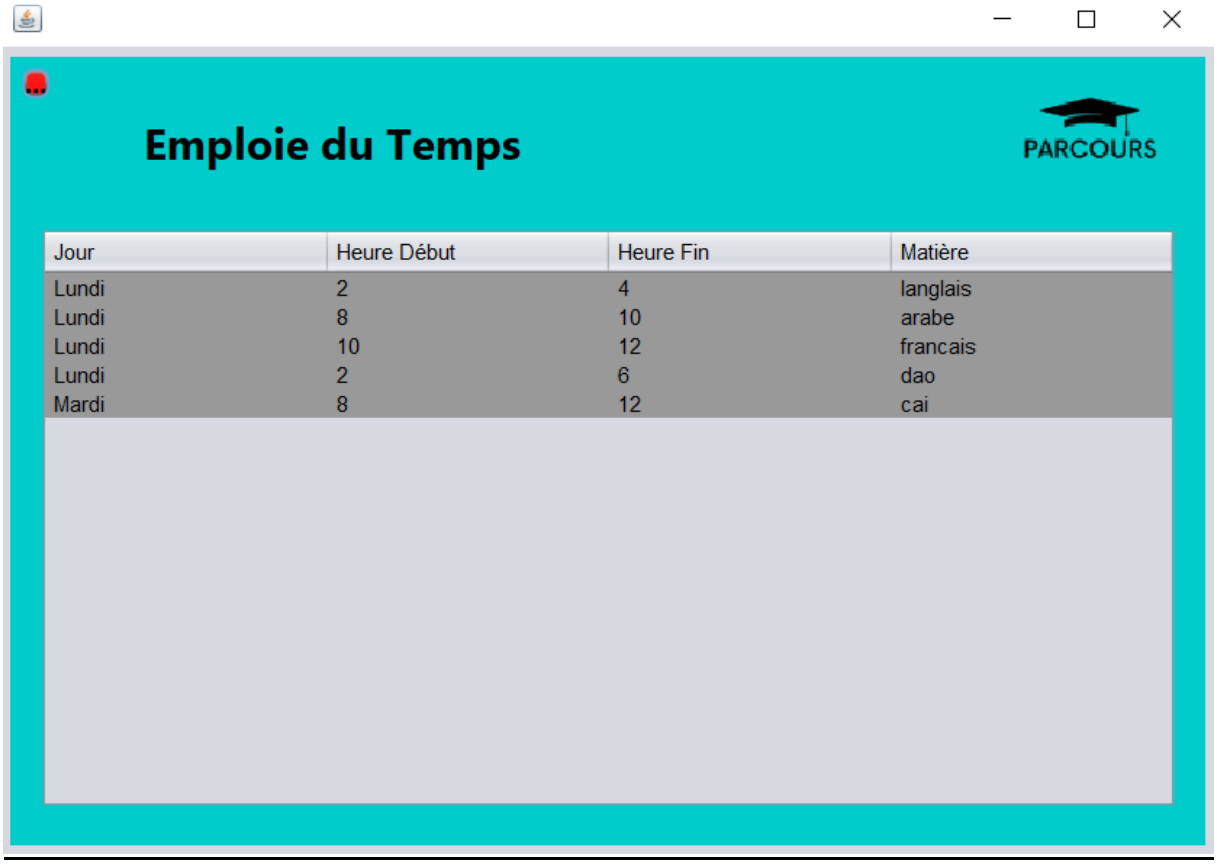


Figure : image de l'interface de emploi du temps

9-Tests et Validation :

1.Introduction :

Les tests constituent une étape cruciale dans tout projet logiciel, car ils permettent de garantir la fiabilité, la robustesse et la stabilité de l'application. nous avons mis en place des tests unitaires pour valider les différentes fonctionnalités offertes par l'application.

2.Outils utilisés :

Pour effectuer les tests, nous avons utilisé les outils suivants :

• JUnit 5 : Framework de test pour Java, intégré dans notre projet Maven.

• JaCoCo (Java Code Coverage) : Outil de génération de rapports de couverture de code, permettant de mesurer quelles portions du code sont effectivement exécutées lors des tests.

3.Configuration Maven :

4.Test du Package dao :

1. Classe de test : CreerCompteTest.java

Cette classe teste les fonctionnalités liées à la création de comptes pour les enseignants et les étudiants.

Méthode de test	Description
testCreerEnseignantSuccess()	Vérifie la création réussie d'un compte enseignant
testCreerEtudiantSuccess()	Vérifie la création réussie d'un compte étudiant
testIsEmailExistsReturnsTrue()	Teste si le système détecte correctement un email déjà existant
testIsEmailExistsReturnsFalse()	Vérifie que le système identifie correctement un email non-existant
testIsCodeExistsReturnsTrue()	Teste si le système détecte correctement un code étudiant déjà existant
testIsCodeExistsReturnsFalse()	Vérifie que le système identifie correctement un code étudiant non-existant

2. Classe de test : CreerEmploieEnsTest.java

Cette classe teste les fonctionnalités de création et de gestion des emplois du temps.

Méthode de test	Description
testSaveEmploi_Success()	Vérifie l'enregistrement réussi d'un emploi du temps
testSaveEmploi_Failure_NullConnection()	Teste la gestion d'erreur lors d'une tentative d'enregistrement avec une connexion null

3. Classe de test : NoteDAOTest.java

Cette classe teste les opérations liées à la gestion des notes des étudiants.

Méthode de test	Description
testSaveNote_Success()	Vérifie l'enregistrement réussi d'une note d'étudiant
testSaveNote_Failure_NullConnection()	Teste la gestion d'erreur lors d'une tentative d'enregistrement avec une connexion null

4. Classe de test : UserDAOTest.java

Cette classe teste les fonctionnalités d'authentification pour différents types d'utilisateurs.

Méthode de test	Description
testAuthenticateAdmin_Success()	Teste l'authentification réussie d'un administrateur

Méthode de test	Description
testAuthenticateEnseignant_Success()	Teste l'authentification réussie d'un enseignant
testAuthenticateEtudiant_Success()	Teste l'authentification réussie d'un étudiant
testAuthenticateAdmin_Fail()	Vérifie la gestion des échecs d'authentification

5. Classe de test : connexionTest.java

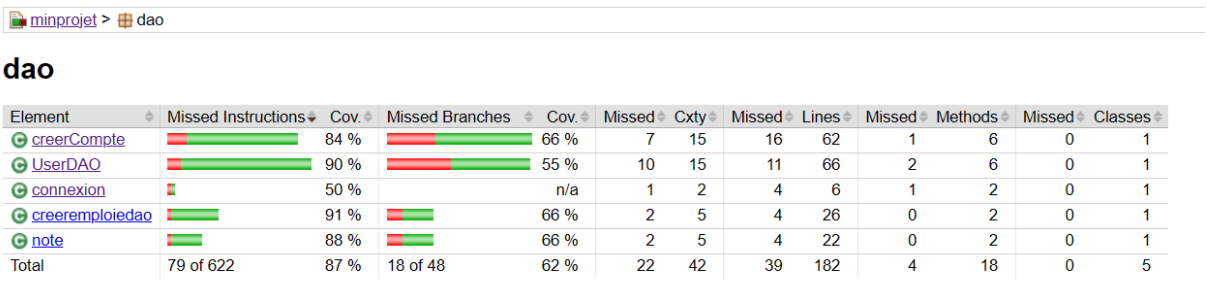
Cette classe teste la fonctionnalité de connexion à la base de données.

Méthode de test	Description
testGetConnection()	Vérifie que la connexion à la base de données est établie correctement et valide

6. Rapport de couverture de code

La couverture des classes principales est la suivante:

Classe	Couverture des lignes
creerCompte.java	84%
creeremploiedao.java	91%
note.java	88%
UserDAO.java	90%
connexion.java	50%



Ce tableau met en lumière à la fois les forces (couverture élevée des lignes/méthodes) et les faiblesses (branches partiellement couvertes) des composants DAO analysés

4.Test du Package entite :

1. Classe de test : AdministrateurTest.java

Cette classe teste les fonctionnalités de l'entité Administrateur.

Méthode de test	Description
testConstructeurParDefautEtSetters()	Vérifie le constructeur par défaut et les setters de la classe Administrateur
testConstructeurAvecParametres()	Teste le constructeur avec paramètres et vérifie l'initialisation correcte des attributs

2. Classe de test : BrancheTest.java

Cette classe teste les fonctionnalités de l'entité Branche.

Méthode de test	Description
testConstructeurEtGetters()	Vérifie le constructeur et les getters de la classe Branche
testSetters()	Teste les setters de la classe Branche et vérifie la modification correcte des attributs

3. Classe de test : ClasseTest.java

Cette classe teste les fonctionnalités de l'entité Classe.

Méthode de test	Description
testConstructeurEtGetters()	Vérifie le constructeur et les getters de la classe Classe
testSetters()	Teste les setters de la classe Classe et vérifie la modification correcte des attributs

4. Classe de test : EmploieDuTempsTest.java

Cette classe teste les fonctionnalités de l'entité EmploieDuTemps.

Méthode de test	Description
testConstructeurComplet()	Vérifie le constructeur avec tous les paramètres de la classe EmploieDuTemps
testSettersEtGetters()	Teste les setters et getters de la classe EmploieDuTemps et vérifie la modification correcte des attributs

5. Classe de test : EnseignantTest.java

Cette classe teste les fonctionnalités de l'entité Enseignant.

Méthode de test	Description
testConstructeurEtGetters()	Vérifie le constructeur avec paramètres et les getters de la classe Enseignant
testSetters()	Teste les setters de la classe Enseignant et vérifie la modification correcte des attributs

6. Classe de test : EtudiantTest.java

Cette classe teste les fonctionnalités de l'entité Etudiant.

Méthode de test	Description
testConstructeurEtGetters()	Vérifie le constructeur avec paramètres et les getters de la classe Etudiant
testSetters()	Teste les setters de la classe Etudiant et vérifie la modification correcte des attributs

7. Classe de test : NoteTest.java

Cette classe teste les fonctionnalités de l'entité Note.

Méthode de test	Description
testConstructeurEtGetters()	Vérifie le constructeur avec paramètres et les getters de la classe Note
testSetters()	Teste les setters de la classe Note et vérifie la modification correcte des attributs

8. Classe de test : MatiereTest.java

Cette classe teste les fonctionnalités de l'entité Matiere.

Méthode de test	Description
testConstructeurEtGetters()	Vérifie le constructeur avec paramètres et les getters de la classe Matiere
testSetters()	Teste les setters de la classe Matiere et vérifie la modification correcte des attributs

4. Rapport de couverture de code

La couverture des classes du package entite est la suivante:

Classe	Couverture des lignes
Etudiant.java	100%
Note.java	100%
Matiere.java	100%
Administrateur.java	100%
Branche.java	100%
Classe.java	100%
EmploieDuTemps.java	100%
Enseignant.java	100%

minprojet > entite

entite

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
Etudiant		100 %		n/a	0	20	0	40	0	20	0	1
EmploieDuTemps		100 %		n/a	0	18	0	36	0	18	0	1
Enseignant		100 %		n/a	0	14	0	28	0	14	0	1
Note		100 %		n/a	0	10	0	20	0	10	0	1
Administrateur		100 %		n/a	0	8	0	16	0	8	0	1
Classe		100 %		n/a	0	7	0	14	0	7	0	1
Matiere		100 %		n/a	0	7	0	14	0	7	0	1
Branche		100 %		n/a	0	5	0	10	0	5	0	1
Total	0 of 419	100 %	0 of 0	n/a	0	89	0	178	0	89	0	8

Ce tableau résume une couverture de code optimale (100%) pour toutes les entités, avec une complexité maîtrisée et aucun élément manquant.