

**MINISTERE DE L'ENSEIGNEMENT SUPERIEUR,  
DE LA RECHERCHE ET DE L'INNOVATION  
ECOLE PRIVEE SUP'INFO  
INTERNATIONAL IT  
DEPARTEMENT INFORMATIQUE**

**Rapport de projet de gestion d'une application de gestion des inscriptions en JAVA**

---

**CONCEPTION ET REALISATION D'UNE APPLICATION  
GRAPHIQUE DE GESTION DES INSCRIPTIONS**

---

*Réalisé et présenté par*  
**MOUHAMADOU MOUSTAPHA DIOP**

**ENCADREUR : M. Mouhamed Bah**

**Année Académique : 2018/2019**

# **TABLE DES MATIERES**

|   |    |
|---|----|
| LISTE DES FIGURES.....                          | 4  |
| LISTE DES TABLEAUX.....                         | 4  |
| INTRODUCTION GENERALE.....                      | 5  |
| CHAPITRE 1.....                                 | 6  |
| 1.1 INTRODUCTION.....                           | 7  |
| 1.2 ETUDE DE L'EXISTANT.....                    | 7  |
| 1.2.1 ANALYSE DE L'EXISTANT.....                | 7  |
| 1.3 PROPOSITION DE SOLUTION.....                | 7  |
| 1.4 CONCLUSION.....                             | 8  |
| CHAPITRE 2.....                                 | 9  |
| 2.1 INTRODUCTION.....                           | 9  |
| Partie 1 : Modélisation.....                    | 10 |
| 1. Choix de la méthodologie de conception.....  | 10 |
| 1.1. Présentation du langage UML.....           | 10 |
| 2. Diagramme des cas d'utilisation.....         | 11 |
| 2.1. Identification des acteurs.....            | 11 |
| 2.2. Identification de cas d'utilisation.....   | 11 |
| 3. Modélisation conceptuelle des données.....   | 16 |
| 3.1. Dictionnaire des données.....              | 16 |
| 3.2. Représentation des classes.....            | 17 |
| 3.3. Diagramme de classe.....                   | 18 |
| Partie 2 : Modélisation logique.....            | 19 |
| 2. Modélisation logique de données.....         | 19 |
| 2.1. Modèle logique des données optimisées..... | 19 |
| CHAPITRE 3.....                                 | 20 |
| Partie 1 : Réalisation.....                     | 21 |
| 3.1. Introduction.....                          | 21 |
| 3.2. Matériel de base.....                      | 22 |

|   |    |
|---|----|
| 3.3. Choix du langage de développement.....   | 22 |
| Partie 2 : Architecture de l'application..... | 22 |
| CONCLUSION.....                               | 24 |
| BIBLIOGRAPHIE.....                            | 24 |

## **LISTE DES FIGURES**

|  |    |
|--|----|
| Figure 1 : Diagramme de cas d'utilisation de l'Administrateur..... | 11 |
| Figure 2 : Diagramme de cas d'utilisation du Coordonnateur .....   | 12 |
| Figure 3 : Diagramme de cas d'utilisation de l'Assistant.....      | 13 |
| Figure 4 : Diagramme de cas d'utilisation de l'application.....    | 14 |
| Figure 5 : Diagramme de classe .....                               | 17 |
| Figure 6 : Architecture de l'application.....                      | 21 |

## **LISTE DES TABLEAUX**

|  |    |
|--|----|
| Tableau 1 : Dictionnaire de données.....                     | 16 |
| Tableau 2 : Représentation des classes.....                  | 17 |
| Tableau 3 : Caractéristiques de l'ordinateur de travail..... | 20 |

## **Introduction Générale**

**De coutume la procédure d'inscription des étudiants dans nos écoles connaisse une lenteur administrative et la gestion de l'ensemble des dossiers de ces Apprenants nécessite une nouvelle réorganisation, une certaine automatisation de la part de l'administration de l'école ou de l'université en question. Et on constate que la période d'inscription des étudiants d'énormes problèmes rencontrés du fait d'une paperasse de dossiers dont la gestion est à revoir.**

**C'est dans cette perspective que nous avons jugé nécessaire mettre en place une application de gestion d'inscription qui permettra une meilleure organisation et une gestion des tâches. L'application met en place un système centralisé dont l'accès reste sécurisé.**

# **CHAPITRE 1**

---

## **PHASE D'ANALYSE**

---

## **1.1 Introduction :**

La phase d'analyse constitue une étape préliminaire pour la réalisation de l'application. En effet, elle permet d'analyser, d'évaluer et de critiquer le fonctionnement habituel. L'analyse et le critique de l'existant nous ont permis de cerner nos objectifs afin de développer un système de qualité dans le futur.

## **1.2 Etude de l'existant :**

Cette section a pour objectif de faire le point sur l'existant les solutions qui existent. Elle permet de démontrer les limites de ces solutions.

### **1.2.1 Analyse de l'existant :**

La procédure d'Inscription des étudiants dans les écoles ou universités se fait la plus part du temps de manière manuelle traditionnelle avec une lenteur ou lourdeur administrative qui est notée. Ce type de gestion des inscriptions présente certaines contraintes :

- Contrainte de temps
- Lenteur sur le traitement du dossier des étudiants
- Perte de dossiers parfois
- Système qui n'est pas centralisé ni sécurisé ni organisé.

## **1.3 Proposition de Solution :**

Nous mettons en place une application de gestion des inscriptions en Java qui facilitera et simplifiera la procédure d'inscription. Elle permet aussi une meilleure organisation du travail et l'automatisation du traitement des dossiers.

L'application permettra :

- D'inscrire de nouveaux étudiants
- De consulter les détails d'un étudiant
- Mettre à jour les informations d'un étudiant
- Désactiver un étudiant (Pas le supprimer)
- Ajouter un ou plusieurs contacts pour un étudiant et donner la possibilité à l'utilisateur de consulter ou modifier ou de supprimer un contact.
- Permettre de faire la gestion des diplômes d'un étudiant
- Permettre de faire la gestion des utilisateurs et des droits d'accès.

Les profils des utilisateurs de cette application sont les suivants :

- L'Administrateur (il a tous les droits d'accès et il définit aussi les différents rôles pour les autres utilisateurs de l'application).
- Le Coordonnateur (il a la possibilité d'ajouter de modifier de supprimer des étudiants ainsi que tout ce qui va avec)
- L'Assistant (il a la possibilité de faire de consultation seulement)

L'accès à l'application nécessite une authentification pour pouvoir l'utiliser ce qui explique son aspect sécurité. Ce qui l'explique, avec trois tentatives de connexion échouées, l'application se referme immédiatement.

## **1.4 Conclusion :**

Dans ce chapitre, nous avons fait un diagnostic sur l'existant qui nous a permis de préparer une bonne conception pour les améliorations que nous allons ajouter dans la solution proposée afin de répondre à nos besoins.

Dans le chapitre qui suit nous présenterons les démarches de développement et de conception de notre application.



# **CHAPITRE 2**

---

## **MODÉLISATION CONCEPTUELLE ET ORGANISATIONNELLE**

---

## **2.1 Introduction :**

La modélisation conceptuelle et organisationnelle constitue une étape importante dans la convergence des notations utilisées dans le domaine de l'analyse de conception objet puisqu'elle représente une synthèse de notre système.

Dans ce chapitre, nous exposons le modèle logique de données, les différents diagrammes de cas d'utilisation et notre diagramme de classe en question.

## **Partie 1 : Modélisation conceptuelle**

### **INTRODUCTION**

Le modèle conceptuel de données est une représentation statique du système d'information. Il a comme objectif de constituer une représentation claire et cohérente des données manipulées dans le système d'information.

Cette section, sera représentée comme suit : nous commençons par le choix de la méthodologie de conception et justification. Enfin nous identifions les différents diagrammes de classe d'utilisation et nous présentons le diagramme de classe.

### **1. Choix de la méthodologie de conception :**

Dans le cadre de notre projet, nous avons choisi le langage UML comme une approche de conception. Nous présentons ci-dessous ce langage puis nous justifions notre choix.

### **1.1 Présentation du langage UML :**

UML (Unified Modeling Language) est un langage formel et normalisé en termes de modélisation objet. Son indépendance par rapport aux langages de programmation, aux domaines de l'application et aux processus, son caractère polyvalent et sa souplesse font de lui un langage universel. En plus UML est essentiellement un support de communication, qui facilite la représentation et la compréhension de solution objet. Sa notation graphique permet d'exprimer visuellement une solution objet, ce qui facilite la comparaison et l'évaluation des solutions. L'aspect de sa notation, limite l'ambiguïté et les incompréhensions.

UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux vues.

## **2. Diagramme des cas d'utilisation :**

Les cas d'utilisation décrivent un ensemble d'actions réalisées par le système, en réponse à une action d'un acteur.

### **2.1 Identification des acteurs :**

L'administrateur, le coordonnateur et l'assistant sont les acteurs qui interagissent avec notre système :

- Administrateur : il a tous les droits d'accès.
- Coordonnateur : Il a la possibilité d'ajouter, de modifier et supprimer des étudiants ainsi que tout ce qui avec.
- Assistant : possibilité de consulter seulement.

### **2.2 Identification des cas d'utilisation :**

Nous décrivons pour chaque acteur les cas d'utilisations et le cas d'utilisation du système global de notre application. Nous distinguons les cas d'utilisations suivantes :

#### **Administrateur :**

- S'authentifier
- Gérer le système CRUD (Create – Update – Delete) des Etudiants
- Gérer le système CRUD des Contacts d'un étudiant donné
- Gérer le système CRUD des Diplômes d'un étudiant donné
- Gérer les utilisateurs de l'application et définir les rôles.

#### **Coordonnateur:**

- S'authentifier
- Gérer le système CRUD (Create – Update – Delete) des Etudiants
- Gérer le système CRUD des Diplômes d'un étudiant donné
- Gérer le système CRUD des Contacts d'un étudiant donné.

#### **Assistant:**

- S'authentifier
- Consultation de la liste des Etudiants
- Consultation de la liste des Diplômes
- Consultation de la liste des Contacts.

Pour simplifier notre diagramme de cas d'utilisation, nous procédons à sa décomposition en trois diagrammes :

## Diagramme qui concerne les activités de l'administrateur

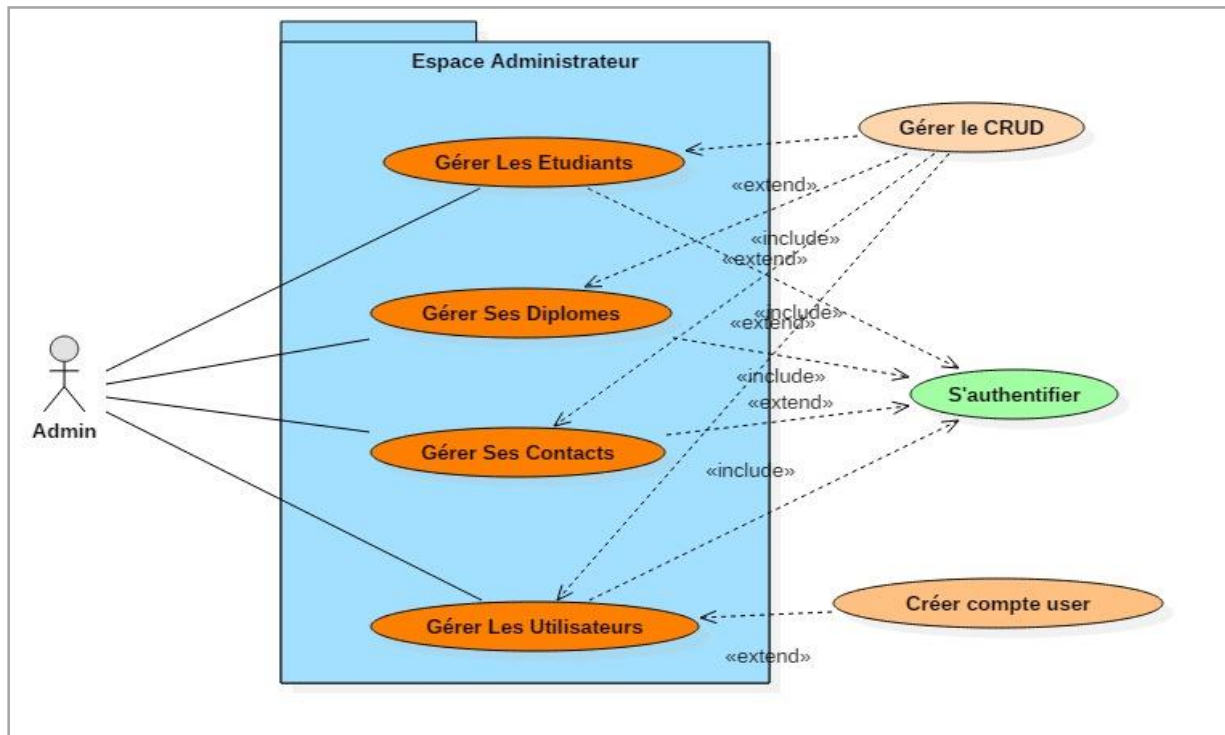


Figure 1 : Diagramme de cas d'utilisation de l'Administrateur

## Diagramme de cas d'utilisation du Coordonnateur

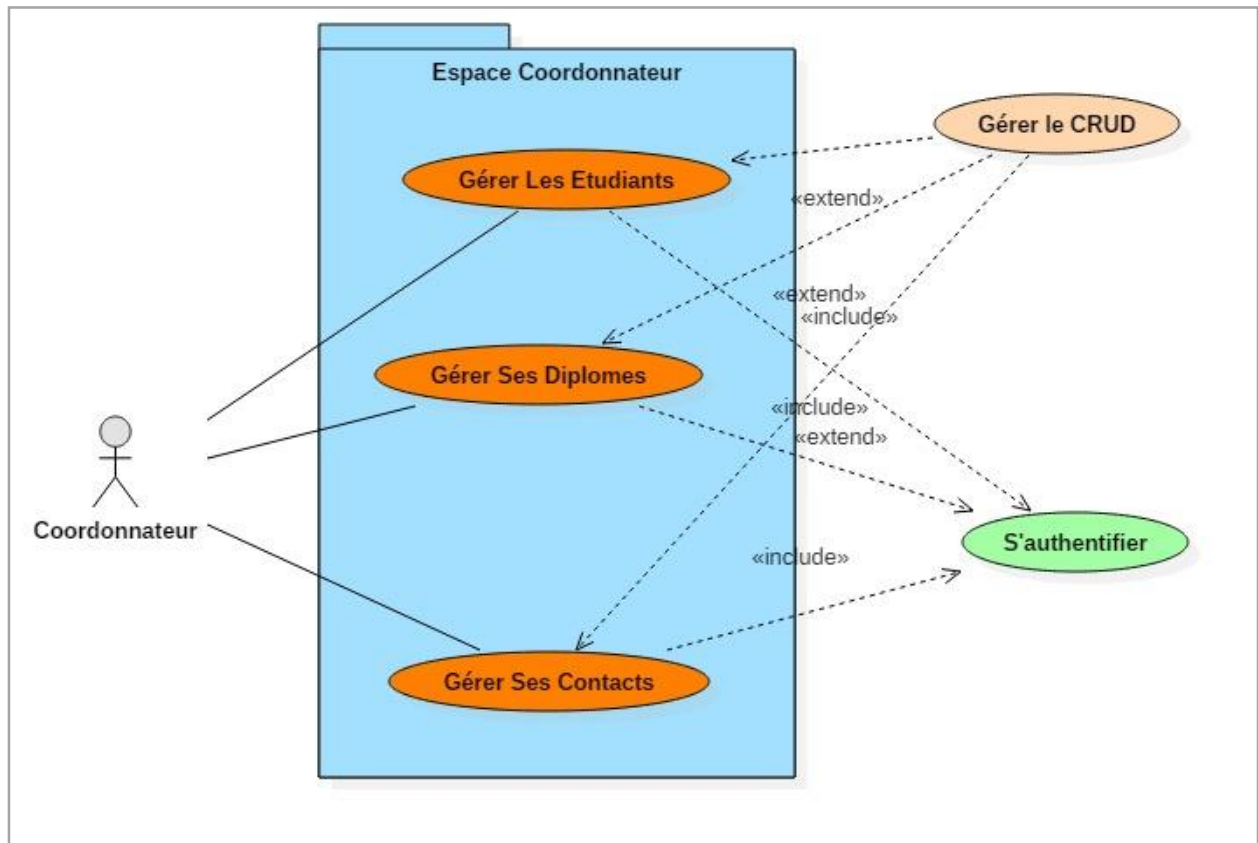


Figure 2 : Diagramme de cas d'utilisation du Coordonnateur

## Diagramme de cas d'utilisation de l'Assistant

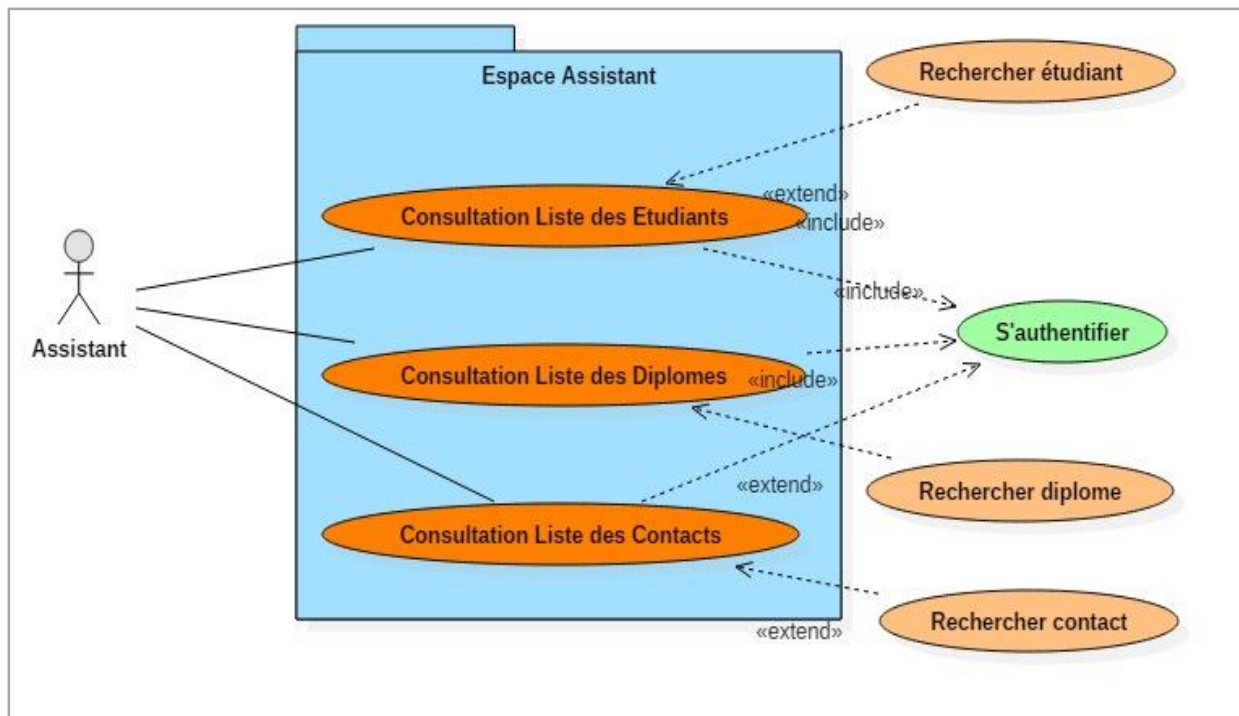


Figure 3 : Diagramme de cas d'utilisation de l'Assistant

## Diagramme de cas d'utilisation du système

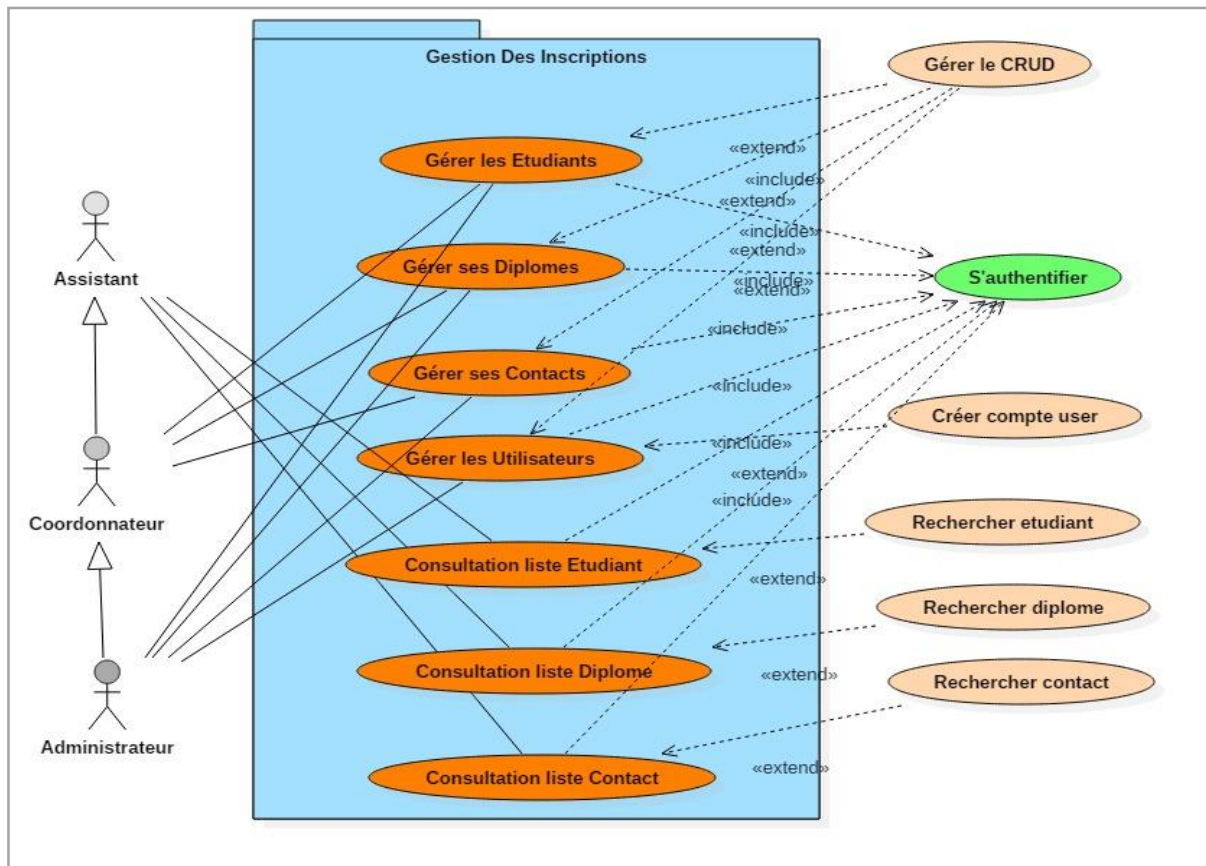


Figure 4 : Diagramme de cas d'utilisation de l'application

### **3. Modélisation conceptuelle des données :**

La modélisation conceptuelle des données permet de dégager l'ensemble des données manipulées en vue d'élaborer le diagramme de classe. En effet, ce dernier donne une vue statique du système.

Il décrit les types et les objets du système.

Il s'agit donc d'une représentation des données du champ de l'étude ainsi que le lien sémantique entre ces données, facilement compréhensible, permettant de décrire le système d'information à l'aide des concepts proposés par le modèle UML.

#### **3.1. Dictionnaire de données**

Le tableau ci-dessous représente la liste des attributs composants toutes les classes formant notre système ainsi que leur type.

| Attribut       | Libellé                                | Type                 |
|----------------|--|----------------------|
| identifiant    | Identifiant de l'étudiant              | Entier               |
| prenom         | Prénom de l'étudiant                   | Chaine de caractères |
| Nom            | Nom de l'étudiant                      | Chaine de caractères |
| date_naissance | La date de naissance de l'étudiant     | Chaine de caractères |
| lieu_naissance | Son lieu de naissance                  | Chaine de caractères |
| Genre          | Le genre de l'étudiant                 | Chaine de caractères |
| nationalite    | Sa nationalité                         | Chaine de caractères |
| telephone      | Son téléphone                          | Entier               |
| Email          | Son adresse électronique               | Chaine de caractères |
| emploi         | Son emploi                             | Chaine de caractères |
| Statut         | Son statut                             | Chaine de caractères |
| Annee          | L'année universitaire                  | Chaine de caractères |
| Cycle          | Son cycle universitaire                | Chaine de caractères |
| departement    | Son département d'étude                | Chaine de caractères |
| id_diplome     | L'identifiant du diplôme               | Entier               |
| Annee          | L'année d'obtention du diplôme         | Entier               |
| Mention        | La mention obtenue                     | Chaine de caractères |
| etablissement  | L'établissement d'obtention du diplôme | Chaine de caractères |
| id_contact     | L'identifiant de son tuteur            | Entier               |
| etat_civil     | L'état civil de son tuteur             | Chaine de caractères |
| prenom_contact | Le prénom de son tuteur                | Chaine de caractères |
| nom_contact    | Le nom de son tuteur                   | Chaine de caractères |
| lien_parente   | Le lien parental de son tuteur         | Chaine de caractères |



|             |   |                      |
|-------------|---|----------------------|
| Adresse     | L'adresse de son tuteur   | Chaine de caractères |
| Ville       | La ville où il habite   | Chaine de caractères |
| tel_contact | Le numéro de téléphone du tuteur  | Entier               |
| Id          | L'identifiant automatique de l'utilisateur affecté par notre système de base de données | Entier               |
| id_user     | L'identifiant de l'utilisateur donné par l'administrateur                               | Entier               |
| Username    | Son pseudonyme  | Chaine de caractères |
| Password    | Son mot de passe  | Chaine de caractères |
| Type        | Son rôle  | Chaine de caractères |

**Tableau 1 : Dictionnaire de données**

### **3.2 Représentation des classes :**

La modélisation objet est utilisée dans le langage UML pour définir des objets-métiers et l'architecture de l'application. Ces objets sont créés en tant qu'instance de classe et s'interagissent dynamiquement pour offrir le comportement décrit par les cas d'utilisation.

Les objets constituent la base de l'architecture des applications, ils peuvent être réutilisés à travers des applications ou encore identifiés ou dérivés directement des cas d'utilisation ou des domaines d'application. Une classe est composée :

- **Attributs** : représentant des données dont les valeurs représentent l'état de l'objet.
- **La méthode** : il s'agit des opérations applicables aux objets

**Nous mettons en place les classes ainsi leurs méthodes et leurs attributs qui sont représentés dans le tableau suivant :**

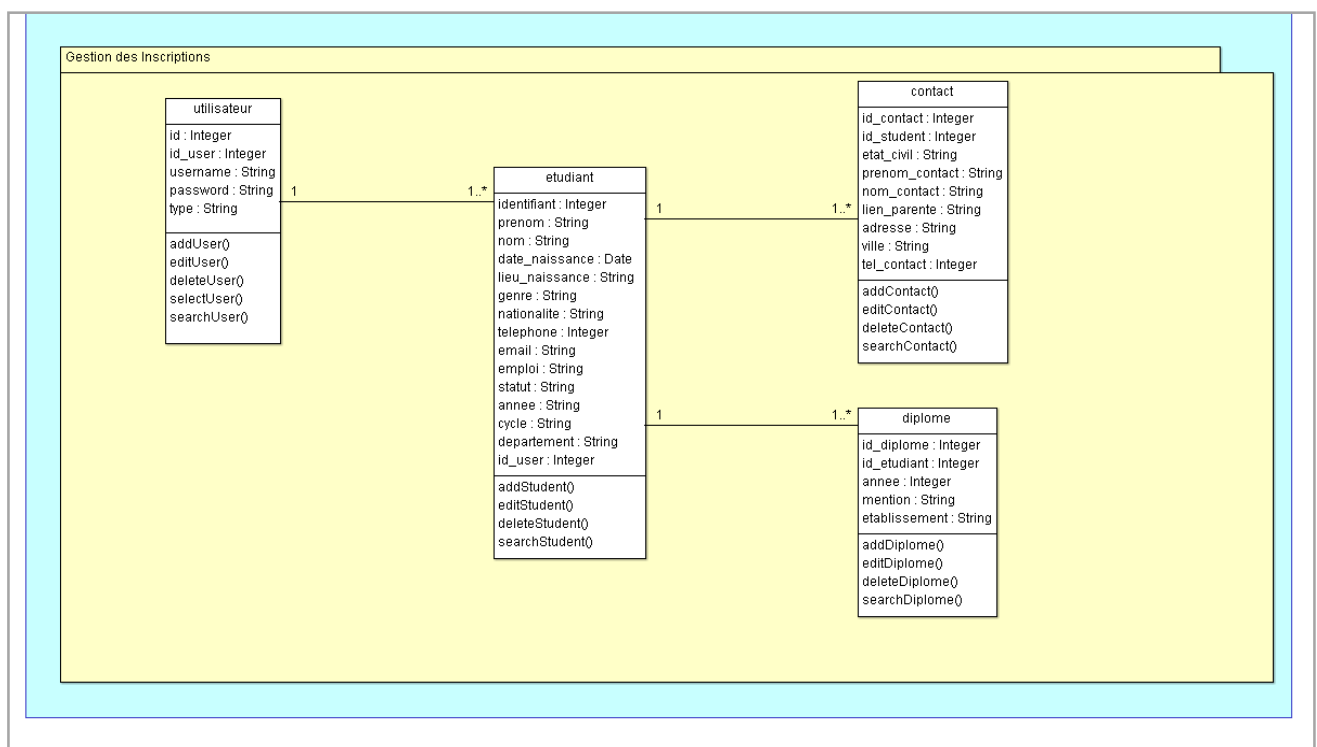
| N° | Nom classe  | Liste des attributs  | Méthodes  |
|----|-------------|--|---|
| 1  | Utilisateur | <ul style="list-style-type: none"> <li>➤ id</li> <li>➤ id_user</li> <li>➤ username</li> <li>➤ password</li> <li>➤ type</li> </ul>  | <ul style="list-style-type: none"> <li>➤ addUser()</li> <li>➤ editUser()</li> <li>➤ deleteUser()</li> <li>➤ selectUser()</li> <li>➤ searchUser()</li> </ul> |
| 2  | etudiant    | <ul style="list-style-type: none"> <li>➤ identifiant</li> <li>➤ prenom</li> <li>➤ nom</li> <li>➤ date_naissance</li> <li>➤ lieu_naissance</li> <li>➤ genre</li> <li>➤ nationalite</li> <li>➤ telephone</li> <li>➤ email</li> <li>➤ emploi</li> </ul> | <ul style="list-style-type: none"> <li>➤ addStudent()</li> <li>➤ editStudent</li> <li>➤ deleteStudent()</li> <li>➤ searchStudent()</li> </ul>               |

|   |         |  |   |
|---|---------|--|---|
|   |         | <ul style="list-style-type: none"> <li>➤ statut</li> <li>➤ annee</li> <li>➤ cycle</li> <li>➤ departement</li> <li>➤ id_user</li> </ul>   |   |
| 3 | Diplôme | <ul style="list-style-type: none"> <li>➤ id_diplome</li> <li>➤ id_etudiant</li> <li>➤ annee</li> <li>➤ mention</li> <li>➤ etablissement</li> </ul>   | <ul style="list-style-type: none"> <li>➤ addDiplome()</li> <li>➤ editDiplome()</li> <li>➤ deleteDiplome()</li> <li>➤ searchDiplome()</li> </ul> |
| 4 | Contact | <ul style="list-style-type: none"> <li>➤ id_contact</li> <li>➤ id_student</li> <li>➤ etat_civil</li> <li>➤ prenom_contact</li> <li>➤ nom_contact</li> <li>➤ lien_parente</li> <li>➤ adresse</li> <li>➤ ville</li> <li>➤ tel_contact</li> </ul> | <ul style="list-style-type: none"> <li>➤ addContact()</li> <li>➤ editContact()</li> <li>➤ deleteContact()</li> <li>➤ searchContact()</li> </ul> |

**Tableau 2 : Représentation des classes**

### 3.3 Diagramme de classe :

La figure ci-dessous récapitule le tableau précédent dans un diagramme de classes qui contient toutes les informations telles que les classes, les méthodes, les associations et les propriétés.



**Figure 5 : Diagramme de classe**

## **Partie 2 : Modélisation Logique**

Dans la section précédente nous avons proposé une modélisation conceptuelle des données et des traitements en se basant sur l'approche objet UML qui représente l'état de l'art des langages de modélisation objet, il permet de modéliser la structure et le comportement d'un système indépendamment de toute méthode ou langage de programmation.

La fiabilité de cette conception est notre porte vers un modèle logique de données efficace et confiant. Ainsi dans cette section nous allons présenter le modèle logique de notre application.

### **2. Modèle logique des données :**

La modélisation logique des données est une représentation des données, issue de la modélisation conceptuelle puis des données.

#### **2.1 Modèle logique des données optimisées :**

Nous présentons le modèle de données optimisées :

- + **utilisateur** (id, id\_user, username, password, type)
- + **etudiant** (identifiant, prenom, nom, date\_naissance, lieu\_naissance, genre, nationalite, telephone, email, emploi, statut, annee, cycle, departement)
- + **diplome** (id\_diplome, #id\_etudiant, annee, mention, etablissement)
- + **contact** (id\_contact, #id\_student, etat\_civil, prenom\_contact, nom\_contact, lien\_parnte, adresse, ville, tel\_contact)

### **Conclusion :**

Dans cette partie, nous avons réalisé la modélisation logique de notre application de gestion des inscriptions. Cette modélisation nous a permis de dégager le modèle logique des données qui sera exploité lors de l'implémentation. Ce modèle sera transformé en modèle physique de données qui fera l'objet du chapitre suivant.

# **CHAPITRE 3**

---

## **RÉALISATION ET ARCHITECTURE**

---

## **Partie 1 : Réalisation**

### **3.1 Introduction :**

Après l'élaboration de la conception de notre application, nous aborderons dans ce chapitre le dernier aspect de notre rapport, dont l'objectif est d'exposer la phase de réalisation.

La phase de réalisation est considérée comme étant la concrétisation finale de toute la méthode de conception.

Nous menons dans un premier temps une étude technique où nous décrivons les ressources logicielles utilisées dans le développement de notre projet. Nous présentons en premier lieu notre choix de l'environnement de travail, où nous spécifions l'environnement matériel et logiciel qu'on a utilisé pour réaliser notre application puis nous détaillons l'architecture, aussi nous présentons quelques interfaces réalisées pour illustrer le fonctionnement de quelques activités du système

### **3.2 Matériel de base :**

Le développement est réalisé par le billet d'un ordinateur ayant les caractéristiques suivantes :

| Caractéristique        | HP                                  |
|------------------------|-------------------------------------|
| Marque                 | HP                                  |
| Processeur             | Intel® Celeron® CPU N3060 @ 1.60GHZ |
| RAM                    | 4.00Go                              |
| Disque dur             | 500Go                               |
| Système d'exploitation | Windows 10 Professionnel            |

**Tableau 3 :** Caractéristique de l'ordinateur de travail

### 3.3 Choix du langage de développement et Système de Gestion de Base de Données(SGBD) :

- + **Java** : C'est un langage de programmation :
  - ✓ conçue pour produire du code de qualité, portable et facile à intégrer
  - ✓ de haut niveau, orienté objet et totalement libre
  - ✓ hautement productif et dynamique
  
- + **MySQL** : C'est un système de Gestion de Bases de Données Relationnelles (SGBDR). Il permet de gérer des bases de données, et donc de gérer de grosses quantités d'informations en utilisant le langage SQL. Il s'agit d'un des SGBDR les plus connus et les plus utilisés. MySQL peut donc s'utiliser seul, mais est la plupart du temps combiné à un autre langage de programmation : PHP pour les sites web, mais aussi Java pour notre cas notre, Python, C++, et beaucoup d'autres.
  
- + **Pilote JDBC** : Java DataBase Connectivity, communément appelé JDBC est un ensemble de classes Java permettant de se connecter et d'interagir avec des bases de données. Il permet à des programmes Java de communiquer avec des bases de données.

Le modèle orienté objet est utilisé pour la conception et la réalisation de notre application.

### Partie 1 : Architecture de l'application

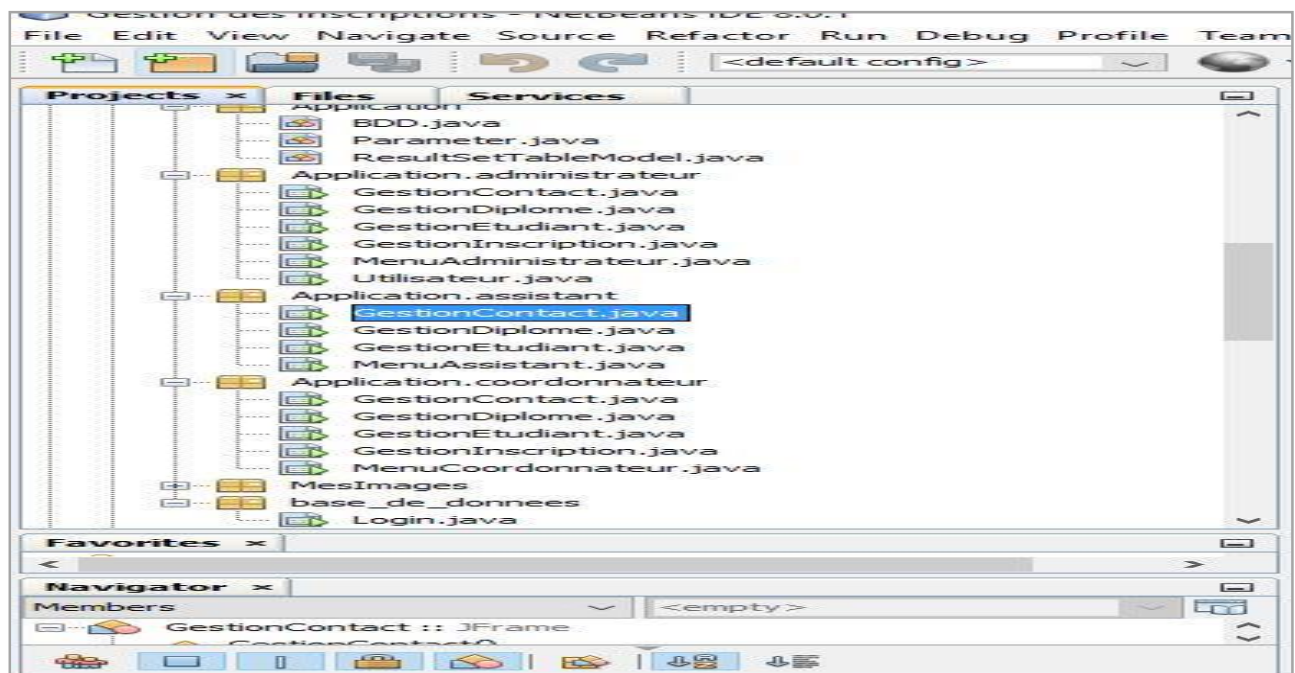





Figure 6 : Architecture de l'application

**L'architecture de notre application est organisée comme suit :**

**Elle est composée de six (6) packages :**

**Application** : Ce package gère l'ensemble des traitements SQL appliqués à la base de données de l'application.

-  La classe **BDD.java** gère les différentes opérations SQL (création de table – sélection des données d'une table – ajout de données – suppression de données – modification de données).
-  La classe **Parameter.java** gère la connexion entre notre base de données et notre application.
-  La classe **ResultSetTableModel.java** gère les données issues de la base de données dans des tableaux Java au niveau de l'application.

**Application.administrateur** : Gère le **système CRUD** de l'application (Gestion des Contacts – Gestion des Diplômes – Gestion des Etudiants – Gestion des Utilisateurs)

**Application.coordonnateur** : Gère aussi le **système CRUD** de l'application excepté la Gestion des Utilisateurs qui est réservée à l'administrateur qui a tous les droits d'accès.

**Application.assistant** : Gère la consultation des données – seule l'opération **recherche** est valable dans cet espace.

**Mes images** : toutes les images utilisées dans les différentes interfaces de l'application sont issues à ce niveau.

**Base de données** : on y trouve **login.java** qui gère le **système d'authentification** l'aspect **sécurité** de l'application.

## **Conclusion**

Cette application gestion des inscriptions a pour objectif d'automatiser le processus des inscriptions et de mieux réorganiser ou gérer les dossiers des étudiants.

Pour réaliser ces tâches, le problème a été abordé en s'appuyant sur la méthode UML et la documentation de Java pour l'implémentation de l'application.

Ce travail a été un thème de recherche dans le monde professionnel. Cette expérience nous a permis de joindre l'utile agréable en évaluant aussi les profondeurs théoriques et pratique dans ce domaine.

## **BIBLIOGRAPHIE**

**Pascal Roques**, UML2 modéliser une application web, 4EME EDITION EYROLLES, 2008

**Cysboy**, Apprenez à programmer en Java, SITE DU ZERO, 29/07/2011

**Chantal Gribaumont (Taguan)**, Administrez vos bases de données avec MySQL, SITE DU ZERO, 29/02/2012