

# Compte-rendu de projet

# SCHOLASTICA

Emanuelle Laurent

Nabile Ouchane

Romain-Pascal Saillot

**Université Claude Bernard**



**Lyon 1**

## Table des matières

1. Introduction .....	3
1.1. Contexte .....	3
1.2. Objectif .....	3
1.3. Collecte d'informations .....	3
2. Planification.....	4
2.1. Cahier des charges .....	4
2.2. Diagramme de GANTT .....	5
3. Choix techniques .....	7
3.1. IDE .....	7
3.2. Java .....	7
3.3. Java IHM et la palette de NetBeans .....	7
3.4. Frameworks Java .....	7
3.5. UML .....	7
3.6. Conception Merise .....	9
3.7. MySQL .....	11
3.8. Git et GitHub.....	12
4. Bilan.....	13
4.1. Points réussis.....	13
4.1.1. Le logiciel est fonctionnel .....	13
4.1.2. Le travail d'équipe.....	13
4.1.3. Respect des procédures.....	13
4.1.4. Phase de test.....	13
4.1.5. Maîtrise de nouveaux outils.....	13
4.2. Axes d'amélioration .....	13
4.2.1. Ce qui était dans le cahier des charges et qui n'a pas été fait.....	13
4.2.2. La planification .....	14
4.2.3. Les objets Java.....	14
4.2.4. Le modèle MVC .....	14
5. Conclusion .....	15

## 1. Introduction

### 1.1. Contexte

Il existe de nombreux logiciels pour la gestion des établissements scolaires mais ceux-ci sont plutôt destinés aux collèges et aux lycées, sont payants ou ne répondent pas aux besoins réels d'une école primaire.

### 1.2. Objectif

On propose de créer un logiciel gratuit qui permette la gestion d'une école primaire, des élèves, de leurs parents, de l'équipe enseignante ainsi que des classes.

### 1.3. Collecte d'informations

Les informations sont collectées sous forme de questionnaire papier distribué aux élèves en début d'année.

## 2. Planification

### 2.1. Cahier des charges

Après nous être mis d'accord sur le projet à réaliser, nous avons rédigé le cahier des charges.

Dans celui-ci, nous avons défini les besoins d'une école primaire et donc les fonctionnalités que devra intégrer notre logiciel.

Exemple de fonctionnalités :

#### Besoins fonctionnels

Fonction	Connexion
<b>Objectif</b>	Se connecter au logiciel
<b>Description</b>	Le premier écran à l'ouverture du logiciel comprend un champ d'identifiant et un de mot de passe.
<b>Contraintes/règles de gestion</b>	Il existe deux niveaux d'accès : un niveau en modification, et un niveau en consultation seulement.
<b>Niveau de priorité</b>	Haute



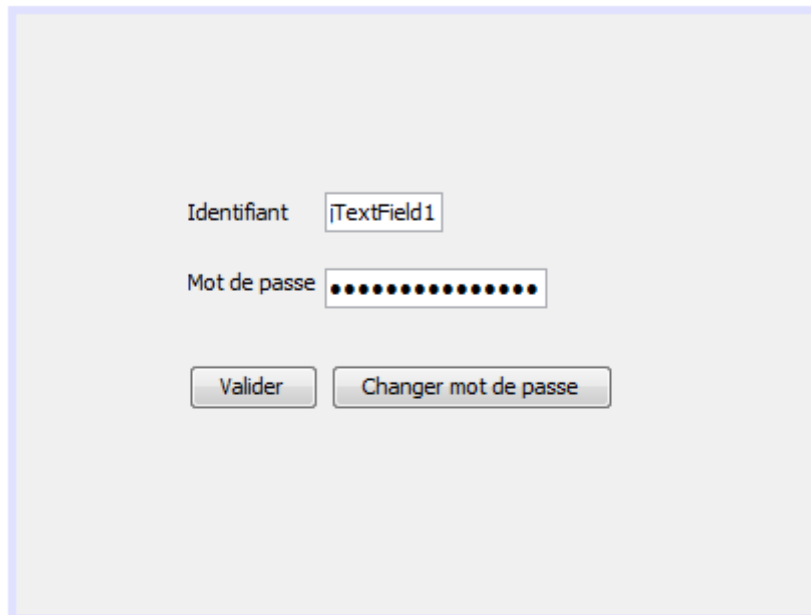
Fonction	Enregistrer/modifier un élève
<b>Objectif</b>	Enregistrer les données associées à un élève
<b>Description</b>	Un bouton du menu principal ouvre une fenêtre en modification. La fenêtre permet d'enregistrer directement les informations suivantes : <ul style="list-style-type: none"> <li>▫ Nom</li> <li>▫ Prénom</li> <li>▫ Photographie</li> <li>▫ Date de naissance</li> <li>▫ Lieu de naissance</li> <li>▫ Nationalité</li> <li>▫ Date d'inscription</li> <li>▫ Date de radiation</li> <li>▫ Adresse</li> <li>▫ N° de téléphone</li> <li>▫ Port de lunettes</li> </ul>

Nous avons aussi commencé à réfléchir aux différentes fenêtres de l'application, leurs rôles et leurs liens. Nous avons donc établi la maquette pour avoir un aperçu simplifié de la vue.

Exemple d'écrans tirés de la maquette :

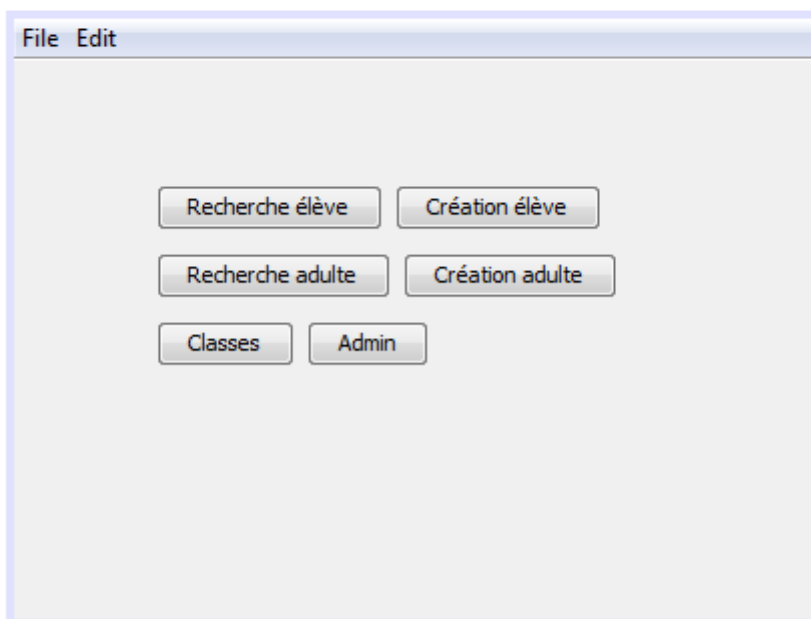
## Maquette

### Connexion



Maquette de la page de connexion. Elle contient deux champs de saisie : 'Identifiant' (contenant 'jTextField1') et 'Mot de passe' (contenant des points). En dessous, il y a deux boutons : 'Valider' et 'Changer mot de passe'.

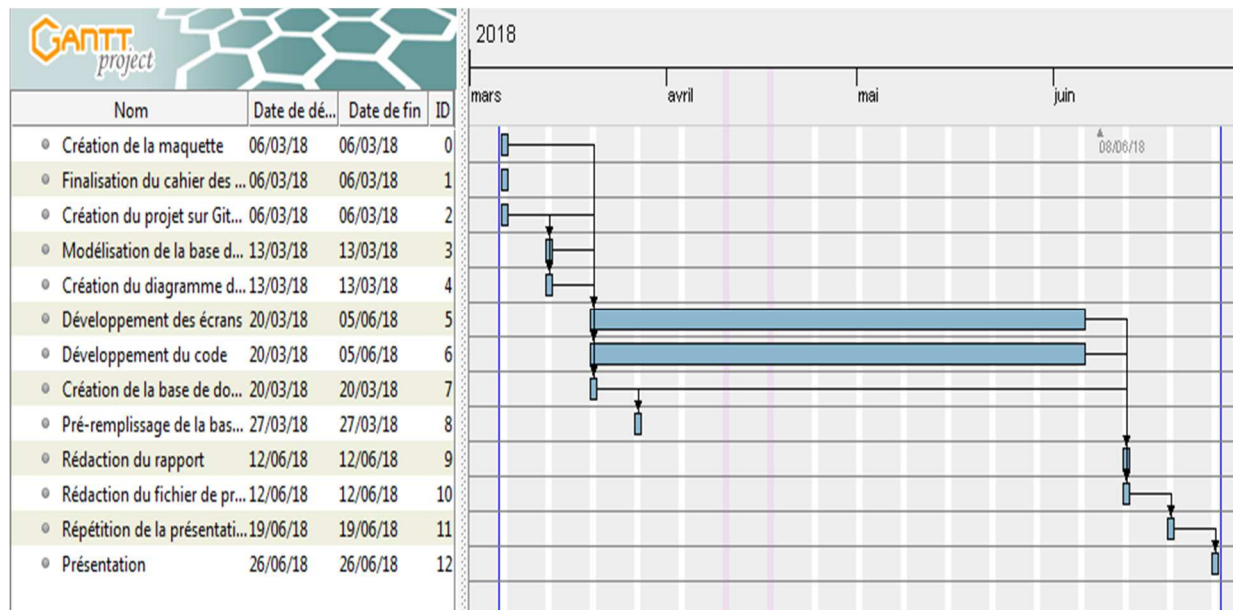
### Écran d'accueil



Maquette de l'écran d'accueil. Elle présente une barre de menu avec 'File' et 'Edit'. Le contenu principal est composé de six boutons disposés en trois lignes : 'Recherche élève' et 'Création élève' (première ligne), 'Recherche adulte' et 'Création adulte' (deuxième ligne), et 'Classes' et 'Admin' (troisième ligne).

## 2.2. Diagramme de GANTT

Nous avons établi un diagramme de GANTT afin de pouvoir organiser les différentes tâches à effectuer pour mener le projet à bien en respectant le temps qui nous a été imparti.



## 3. Choix techniques

### 3.1. IDE

Nous avons précédemment utilisé NetBeans en cours, donc nous avons naturellement décidé de continuer à l'utiliser pour le projet, d'autant qu'il est disponible gratuitement.

### 3.2. Java

Le choix du langage de développement nous a été imposé par la consigne.

### 3.3. Java IHM et la palette de NetBeans

La consigne nous a aussi imposé d'utiliser les IHM natives Java pour l'interface utilisateur. Pour gagner du temps et faciliter la mise en page, et parce que nous utilisons tous le même IDE, nous avons décidé d'utiliser la palette de NetBeans.

Ce choix de la palette a parfois posé problème, parce que nous l'avions très peu utilisée précédemment et nous n'avions pas encore acquis d'automatismes. De plus, les morceaux de code trouvés sur internet ont généralement dû être adaptés pour pouvoir être utilisés avec du code généré automatiquement par la palette.

### 3.4. Frameworks Java

Nous avons réfléchi à la possibilité d'utiliser le framework Hibernate, et nous avons finalement décidé de ne pas le faire. Le temps que nous aurions pu gagner en utilisant un framework aurait été perdu en cherchant à apprendre comment utiliser ce framework.

Nos connaissances actuelles nous ont suffi pour écrire le projet.

### 3.5. UML

Nous avons utilisé un diagramme de classe en UML sur PowerAMC pour générer le squelette du code Java. Les limitations de PowerAMC nous ont obligé à modifier manuellement les noms de méthodes dans NetBeans.

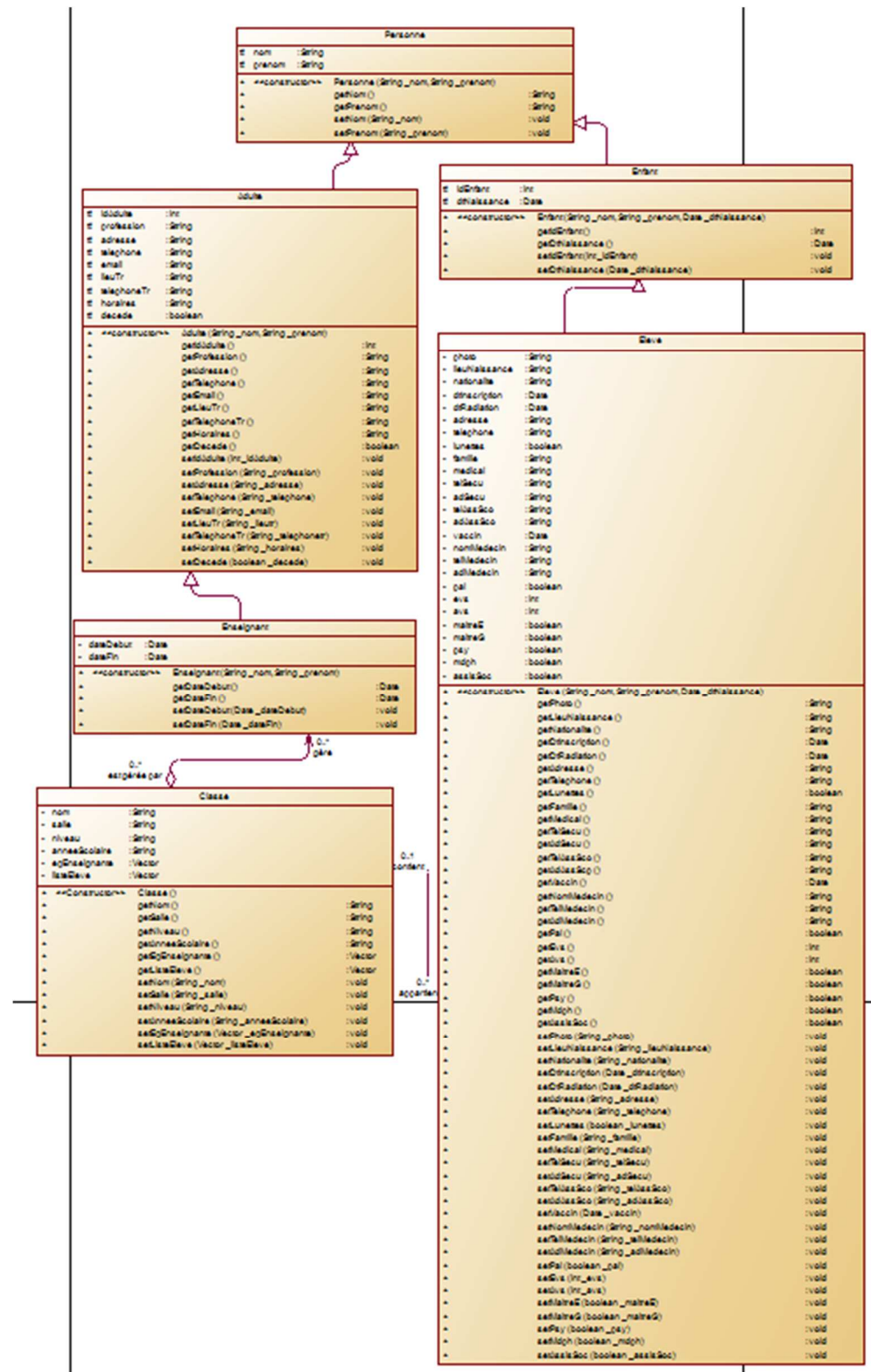


Figure 1 Vision globale du diagramme de classes



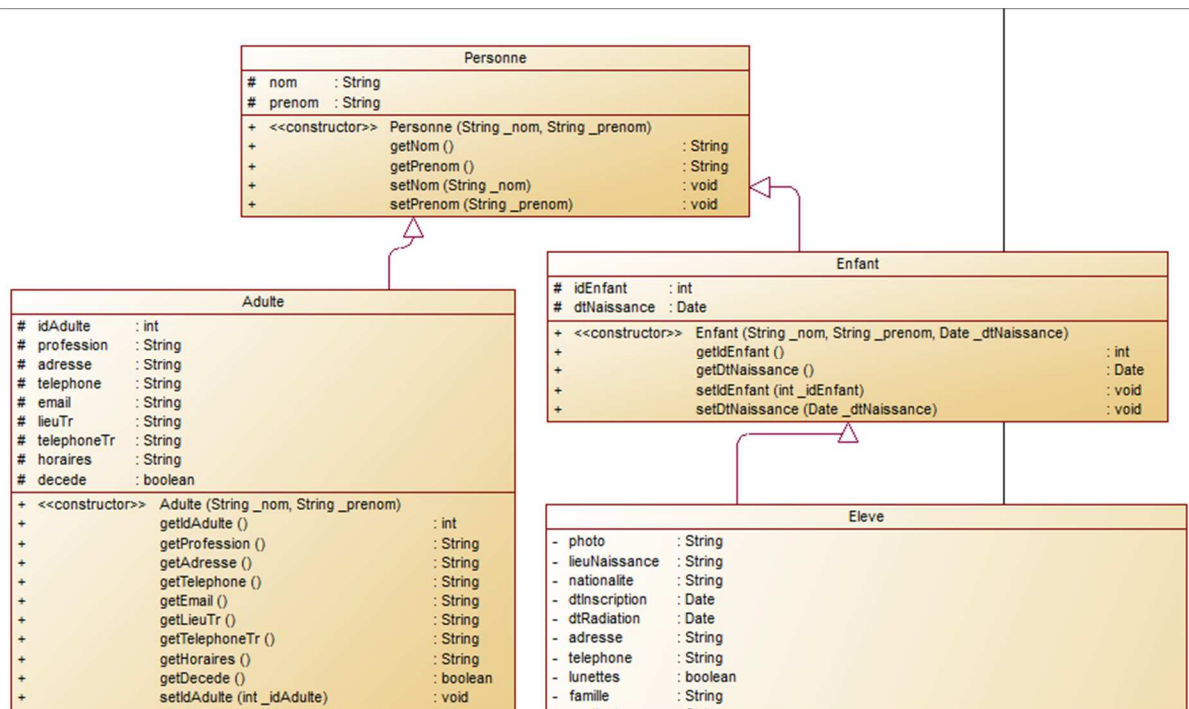


Figure 2 Détail du diagramme de classe

### 3.6. Conception Merise

Nous avons modélisé la base de données en utilisant Merise sur PowerAMC.

Le code SQL généré par PowerAMC pour créer la base ne fonctionnant pas, nous avons finalement créé la base de données manuellement.

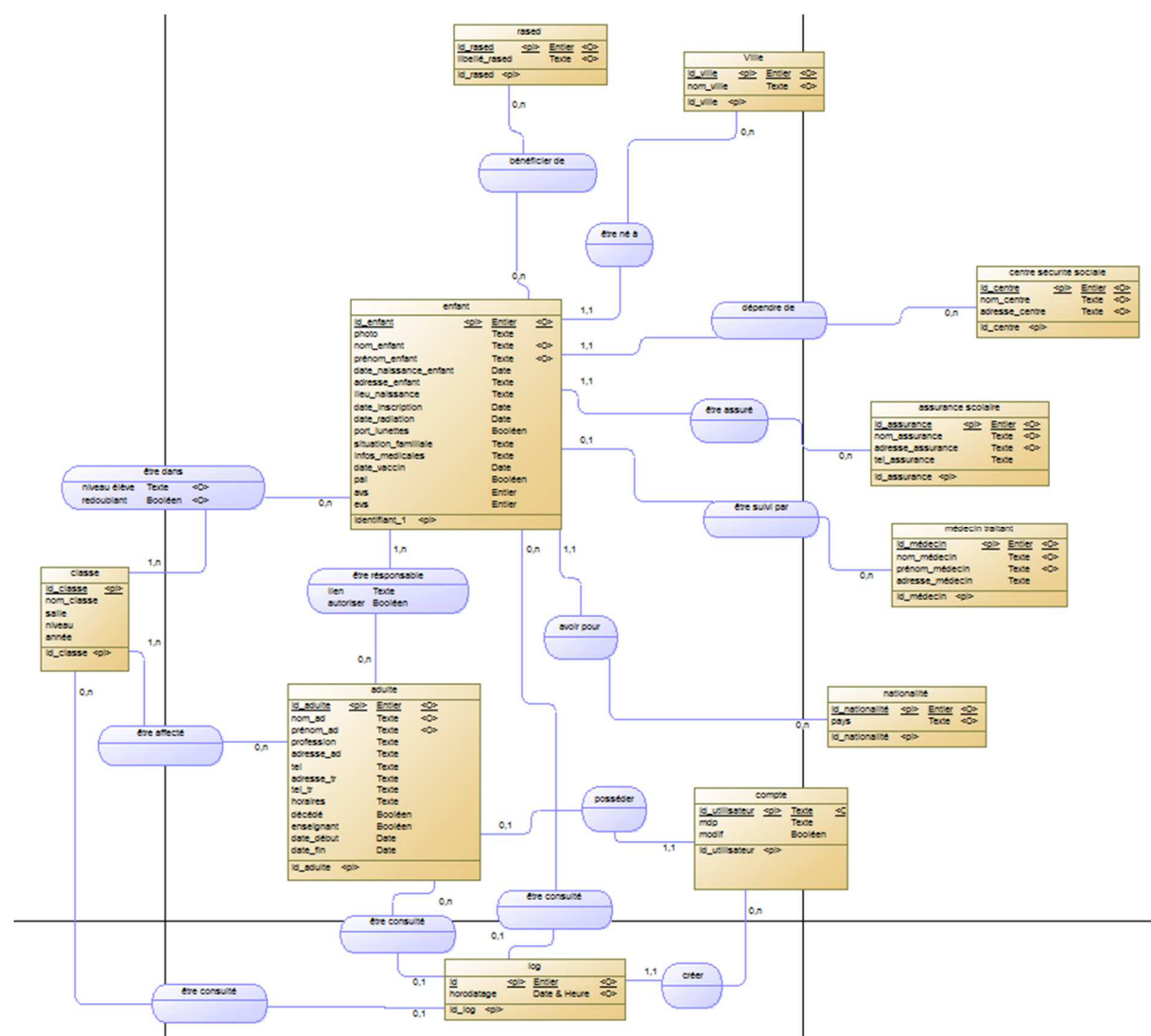


Figure 3 Vision globale du MCD

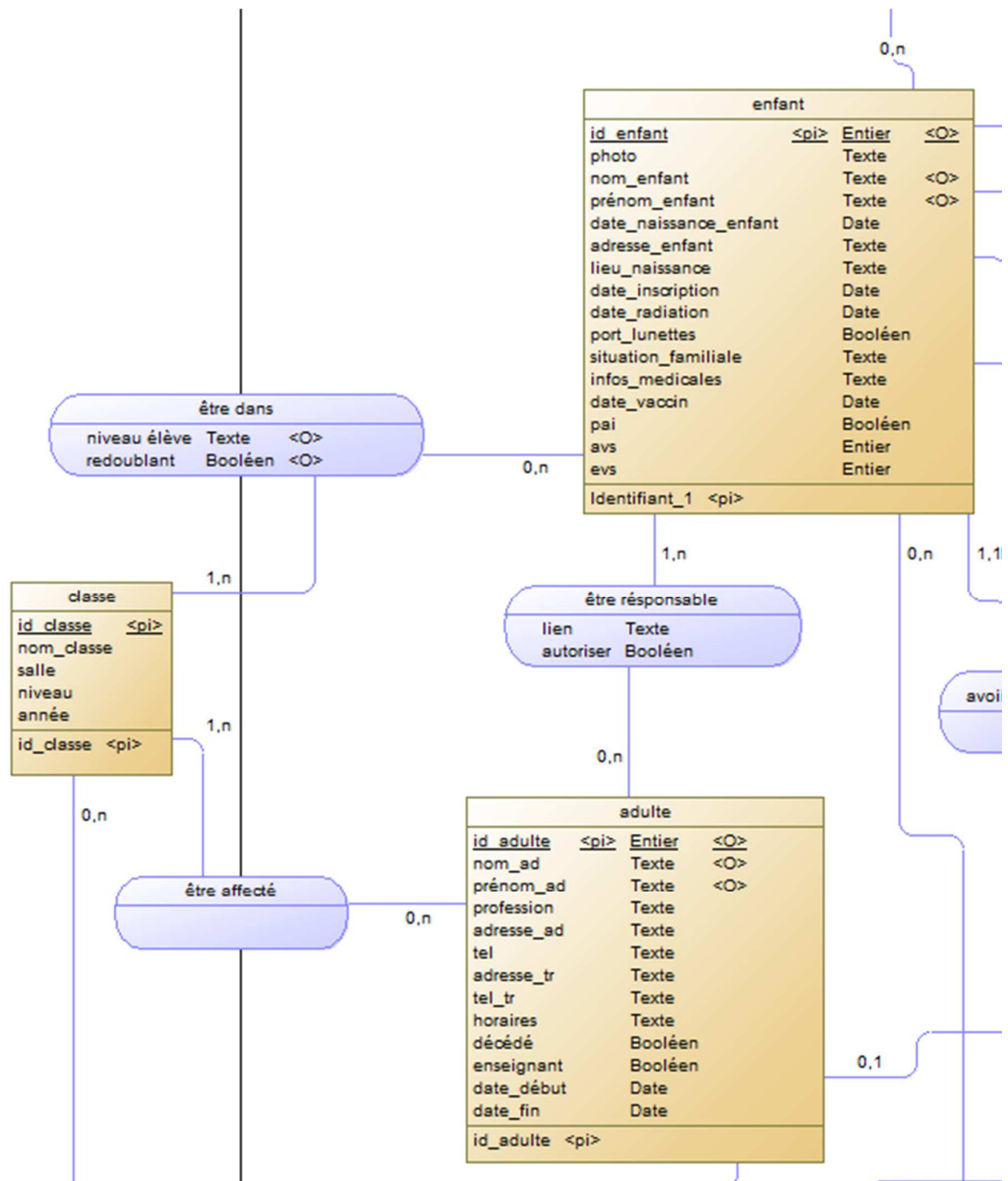


Figure 4 Détail du MCD

### 3.7. MySQL

Nous avons hésité entre plusieurs RDB : Oracle Server ou MySQL que nous avons utilisé en cours ou PostgreSQL que certains d'entre nous utilisent professionnellement.

Notre choix s'est finalement porté sur MySQL parce que, en tant qu'étudiants de Lyon 1, nous avons accès à la base implémentée pour chaque étudiant, ce qui résolvait le problème de l'installation. Le fait que la base soit accessible en

ligne permettait aussi de facilement travailler à plusieurs depuis différents endroits.

### 3.8. Git et GitHub

Nous avons utilisé Git en tant qu'outil de gestion de version. Nous avons créé un dépôt sur GitHub sur lequel nous nous sommes connectés en utilisant Git Bash. Certains d'entre nous utilisent professionnellement ce logiciel et ce site, et cette expérience nous a permis d'approprier plus facilement l'utilisation de Git, et de résoudre les quelques problèmes de merge qui se sont posés au cours du projet.

Nous avons chacun créé des branches sur lesquelles nous avons travaillé indépendamment sur certaines fonctionnalités, et nous avons ensuite mis notre travail en commun sans problèmes majeurs. Git nous a aussi permis de travailler aussi bien de l'IUT que de chez nous.

## 4. Bilan

### 4.1. Points réussis

#### 4.1.1. Le logiciel est fonctionnel

Nous avons respecté 95% du cahier des charges, le logiciel livré est fonctionnel, même si le code pourrait être optimisé.

#### 4.1.2. Le travail d'équipe

Nous avons réussi à répartir les tâches de manière égale et à communiquer. Nous avons aussi réussi à mettre en place une relecture de code.

#### 4.1.3. Respect des procédures

Nous avons respecté le processus de création d'un logiciel : établissement du cahier des charges, planification avec un diagramme de GANTT, conception des classes et de la base de données avec UML et Merise, développement, tests, livraison.

#### 4.1.4. Phase de test

Nous avons procédé à des tests de fonctionnalité avant chaque livraison d'écran, ainsi qu'à la livraison de l'ensemble du logiciel.

#### 4.1.5. Maîtrise de nouveaux outils

Nous avons très peu utilisé la palette de NetBeans en cours, mais nous avons pu au cours du projet l'utiliser et découvrir ses avantages mais aussi ses faiblesses.

Nous avons déjà utilisé MySQL en cours, mais de manière très basique. Le projet nous a permis de découvrir des fonctionnalités plus avancées et notamment le langage procédural SQL/PSM via la création de procédures stockées et de triggers.

De même, nous avons abordé le fonctionnement de Git en classe, mais l'utilisation concrète du logiciel via une interface en ligne de commande nous a permis d'appréhender le fonctionnement un peu complexe de l'outil.

### 4.2. Axes d'amélioration

#### 4.2.1. Ce qui était dans le cahier des charges et qui n'a pas été fait

Nous avons prévu de pouvoir imprimer une fiche de renseignements vierge pré-remplie avec les infos de l'école et aussi d'imprimer les fiches élève renseignées. Bien que nous ayons trouvé la bibliothèque Java nécessaire pour

créer des PDF, l'implémentation aurait été trop longue par rapport au temps qu'il nous restait.

Nous avons aussi prévu de nous mettre en conformité avec le RGPD, en permettant le téléchargement des enregistrements relatifs aux personnes, ainsi qu'un log enregistrant l'identifiant de l'utilisateur ayant modifié ou consulté les écrans adultes ou enfants. Encore une fois, des problèmes de gestion du temps nous ont empêchés d'implémenter ces fonctionnalités.

#### 4.2.2. La planification

Au vu du précédent point, il est clair que notre planification a péché. Nous aurions probablement dû passer une séance supplémentaire au début du projet pour établir une documentation fonctionnelle beaucoup plus précise qui nous aurait permis de réfléchir aux problèmes en amont au lieu de les résoudre au fur et à mesure que nous les soulevions.

#### 4.2.3. Les objets Java

Nous avons conçu un diagramme de classe UML dans le but d'utiliser des objets JAVA pour interagir à la fois avec les données collectées via l'IHM et avec la base de données, ce qui aurait permis de limiter les connexions à la base de données et d'éviter de créer des enregistrements temporaires dans la base. Des difficultés initiales et un manque de temps nous ont incités à court-circuiter cette option pour interagir directement avec la base de données, ce qui a posé d'autres problèmes par la suite.

#### 4.2.4. Le modèle MVC

Comme pour les objets Java, nous avons l'intention d'organiser notre code selon le modèle MVC, mais les lacunes de notre planification nous ont empêché de mettre ce système en place, et il a été ensuite trop difficile et chronophage de revenir en arrière pour adapter le code déjà écrit.

## 5. Conclusion

Le projet a été l'occasion pour nous de faire la synthèse des différents domaines étudiés au cours des trois ans précédents.

Ce fut aussi une première approche de ce que peut être le développement dans un cadre professionnel et non universitaire, même si l'exercice est un peu artificiel. Elle nous a permis de découvrir les difficultés que peut rencontrer un projet informatique, aussi bien organisationnelles que purement techniques.