

REPUBLIQUE DU SENEGAL



UN PEUPLE – UN BUT – UNE FOI

**MINISTERE DE L'ENSEIGNEMENT SUPERIEUR DE LA RECHERCHE ET DE
L'INNOVATION**

Institut Africain de Management



**MEMOIRE DE FIN DE FORMATION POUR L'OBTENTION DU DIPLOME
D'INGENIEUR EN SYSTEME D'INFORMTION ET GENIE LOGIEL**

Option : Système d'information et Génie Logiciel

**CONCEPTION ET REALISATION D'UNE SOLUTION WEB
ET MOBILE DE GESTION D'ECOLE (MOYEN ET
SECONDAIRE) « SAMASCHOOL »**

Présenté et soutenu par :

M. Ibra NDIAYE

Directeur de mémoire

Dr Fodé CAMARA

Consultant Mr Macodou DIOUF

Année de soutenance : 2019

Dédicaces

Je dédie ce modeste travail, aux deux êtres les plus chers à mon cœur auxquels je dois mon existence :

Mon père et ma mère ; vous qui êtes toujours à mes côtés pour me soutenir et m'encourager à me battre sans jamais m'arrêter à mi-chemin ; que dieu vous protège.

A mes grands-parents.

A mes chers frères.

A mes chères sœurs.

A mes oncles et mes tantes.

A mes très chers amis.

Au responsable de programme informatique : Dr Fodé CAMARA.

A tous mes amis de la promotion Master 2 informatique 2017/2018 et tous ceux qui m'ont aidé durant cette application.

A ceux-là, et a tous ceux que j'aurais oublié citer, j'exprime mon infaillible reconnaissance et ma sincère gratitude.

Remerciements

Nous remercions le bon Dieu, tout puissant, de nous avoir donné la force pour suivre, ainsi que l'audace pour dépasser toutes les difficultés.

Au terme de ce travail, nous tenons à remercier

Nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

Nous tenons à remercier sincèrement nos directeurs de mémoire Docteur Fodé Camara et Monsieur Macodou Diouf, pour ses qualités humaines et professionnelles, pour son encadrement, ses directives, ses remarques constructives, et sa disponibilité tout au long de la réalisation de ce mémoire.

Messieurs le président et les membres du jury pour leurs efforts et leur soin apporté à notre travail.
Aux professeurs de notre institut et département informatique.

Tous les membres de DIGI221 en l'occurrence Cheikh Tidiane DIOP, Mohamed CAMARA de leurs conseils pour la réalisation de ce présent mémoire,

Enfin, nous adressons nos plus sincères remerciement à tous nos proches et amis, qui nous ont toujours soutenue et encouragée au cours de la réalisation de ce mémoire.

Merci à tous et à toutes.

SIGLES ET ABREVIATIONS

API	Application Programming Interface
CDDL	Common Development and Distribution License
CV	Curriculum Vitae
DMZ	Demilitarized Zone
EDI	Echange de données informatisées/ Electronic Data Interchange
HTTPS	Hyper Text Transfer Protocol Secure
LDAP	Lightweight Directory Access Protocol
MCD	Modèle Conceptuel de Données
MCT	Modèle conceptuel des traitements
MERISE	Méthode d'Etude et de Relation Informatique pour les Systèmes d'entreprises
MLD	Modèle Logique des Données
MOT	Modèle Logique des Données
MPD	Modèle Physique des Données
MPT	Modèle physique des traitements
OMG	Object Management Group
OMT	Object Modeling Technique
ONFP	Office Nationale de la Formation Professionnelle
OOSE	Object Oriented Software Engineering
PHP	Hypertext Preprocessor
SDK	Software Development Kit
SGBD	Système de Gestion des Bases de Données
SQL	Structured Query Language
SSI	Sécurité des Systèmes d'Informationn
UML	Unified Modeling Language
XML	Extensible Markup Language
XSS	Cross-Site Scripting

TABLE DES ILLUSTRATIONS

Tableau 1 : comparatif des méthodes de gestion de projet	7
Tableau 2 : Niveaux d'analyse	13
Tableau 3 : liste des cas d'utilisation	20
Tableau 4 : Comparatifs des différentes architectures	41
Figure 1: Cycle d'abstraction	12
Figure 2: Cas d'utilisation de l'élève	21
Figure 3: Cas d'utilisation Professeur	23
Figure 4: : Cas d'utilisation administration	25
Figure 5: Diagramme séquence authentification	28
Figure 6: Diagramme séquence consulter absence	29
Figure 7: Diagramme séquence enregistrer Planning	30
Figure 8: Diagramme séquence générer bulletin	31
Figure 12: Diagramme de classe d'analyse	32
Figure 13: Diagramme d'objet	33
Figure 15: Architecture 1 tiers	36
Figure 16: Architecture 2 tiers	37
Figure 17: Architecture 3 tiers	38
Figure 18: Architecture N tiers	39
Figure 5-19: Diagramme de déploiement	34
Ecran 1 : Page de Connexion	53
Ecran 2 : Page d'accueil	53
Ecran 3 : Formulaire d'inscription élève	54
Ecran 4 : Tableau liste des classes	54
Ecran 5 : Liste élève d'une classe	55
Ecran 6 : Relevé de note d'une élève	55
Ecran 7 : Fiche élève	56
Ecran 8 : Bulletin élève page web	56
Ecran 9 : Bulletin version PDF	57
Ecran 10 : Login et choix de classe	58
Ecran 11 : Menu principal et planning cours professeur	59
Ecran 12 : Séance de cours et appel	60
Ecran 13 : formulaire évaluation et liste des évaluations	61
Ecran 14 : noter une élève	62
Ecran 15 : écran d'accueil parent	Ecran 16 : Menu parent et élevé
Ecran 17 : notes et planning de l'élève	64
Ecran 18 : planning devoir et absence	65

SOMMAIRE

Introduction Générale.....	1
Partie 1 : PRESENTATION ET CADRE METHODOLOGIQUE	2
Chapitre 1 : Présentation Général	3
I. Présentation du projet	3
II. Problématique	3
III. Objectif du projet	3
Chapitre 2 : Cadre Méthodologique	5
I. Présentation des différentes méthodes.....	5
II. Choix de la méthode	7
III. Présentation de la méthode choisit	8
Partie 2 : Analyse et Conception	10
Chapitre 3 : Analyse	11
I. Méthode D'analyse et Choix de la méthode	11
II. Spécification des besoins du Système	16
III. règle de gestion.....	18
Chapitre 4 : Conception Case.....	19
I. Diagramme de Use Case.....	19
II. Diagramme Séquence	27
III. Diagramme de classe	32
IV. Diagramme d'objet.....	33
V. Diagramme de déploiement	34
Partie 3 : MISE EN ŒUVRE DE LA SOLUTION (REALISATION)	35
Chapitre 5 : Architecture de la solution	36
I. Architecture 1 tiers.....	36
II. Architecture 2 tiers.....	36
III. Architecture 3 tiers.....	37
IV. Architecture N tiers.....	38
V. Choix de l'architecture.....	39
Chapitre 6 : Présentation de la solution	42
I. Choix du Framework Front End.....	42
II. Choix du Framework Back End	46
III. Choix du serveur de base de données	49
IV. Présentation des outils développement	50
V. Capture D'Ecran	52
Conclusion	66

INTRODUCTION GENERALE

De nos jours l'accès à l'information est devenu de plus en plus facile grâce à la révolution numérique. On n'a plus besoin de se déplacer pour avoir accès à l'information. De nouveaux concepts comme par exemple « village planétaire » sont devenus aujourd'hui une réalité. De ce fait elle a amélioré notre façon de vivre, d'apprendre de travailler. Dans le cadre des établissements ou l'information occupe une place très importante l'utilisation des nouvelles technologies devient quelque chose d'incontournable.

C'est dans ce contexte que nous sommes fixés comme objectif de mettre en place une plateforme web et mobile qui sera dédié aux principaux acteurs du système éducatif à savoir l'administration, les élèves et les parents d'élève. Elle permettra aux parents d'élève d'avoir accès à tous les informations pour le contrôle de leurs enfants à l'école sans se déplacer. Cette plateforme permettra aussi aux élèves d'obtenir des informations comme les notes, les plannings et les devoirs à venir. Il servira à l'administration et le professeur d'automatiser les tâches traditionnelles.

Pour atteindre ces objectifs nous allons d'abord commencer par le cadre théorique afin de bien comprendre le projet à mettre place. Ensuite nous allons passer par le cadre méthodologique pour délimiter le projet à mettre en place. Enfin nous allons passer à la mise en œuvre de la solution et la présentation de la solution proposée.



Partie 1 : CADRE THEORIQUE ET CADRE METHODOLOGIQUE

Chapitre 1 : PRESENTATION GENERAL

I. Présentation du projet

Le projet « Samaschool » consiste à la création d'une plateforme web et mobile qui s'active dans le domaine scolaire (le moyen et le secondaire). Cette plateforme sera une véritable opportunité pour l'administration d'automatiser les tâches traditionnelles liés à la gestion scolaire (l'inscription, l'impression de bulletin la gestion des plannings et...).

La plateforme mobile sera dédiées aux professeurs, aux élèves et aux parents d'élèves. Elle permettra aux professeurs de programmer des devoirs, de saisir des notes, d'effectuer des appels. Pour les élèves elle leur permettra des consulter leurs notes les devoirs programmer et leur planning. Pour les parents d'élèves elles leurs permettront de consulter les notes, les plannings et les absences de leurs élèves.

II.Problématique

La plupart des écoles ne disposent pas de systèmes d'information fiables. Cela rend difficile toute action de planification rigoureuse. L'absence d'un dispositif performant d'information et la faiblesse de la circulation de l'information du niveau central vers le niveau périphérique et vice-versa contribue aussi à la mauvaise gestion des systèmes éducatifs.

Aujourd'hui les parents d'élèves sont obligés de se déplacer pour obtenir de l'information par rapport à leurs élèves. Certains parents rencontrent d'énormes difficultés pour se rendre à l'école de leurs enfants.

Les élèves ne sont pas assez surveillés par les parents concernant à leurs études. Souvent les parents reçoivent de fausses informations. Ils n'ont pas le temps toujours d'aller vérifier au près de l'école.

III.Objectifs du projet

L'objectif du projet « Samaschool » est de permettre l'administration, les parents d'élèves et les élèves d'avoir accès plus facile à l'information. Vu la taille et l'ampleur que peut avoir cette plateforme, nous nous sommes fixés les objectifs suivants :

- Permettre aux élèves d'avoir accès à leurs notes, leurs plannings et leur devoir n'importe où

- Donner aux parents un moyen supplémentaire pour surveiller le travail de leurs enfants sans se déplacer
- Faciliter la gestion des ressources pédagogique des élèves
- Permettre à l'administration d'automatiser leurs tâches quotidiennes

Chapitre 2 : CADRE METHODOLOGIQUE

I. Présentation des différentes méthodes

La planification de projet est une étape clé de la gestion d'un projet. Elle conditionne son déroulement, et à terme, son succès ou son échec. Il est donc primordial de structurer correctement le projet. Pour cela, il existe des types de planification reconnus destinés à faciliter la gestion du projet. Le choix de la méthode de planification va avant tout dépendre du type de projet. Nous allons présenter ici cinq méthodes de planification, afin de vous aider à sélectionner celle qui conviendra le mieux à votre projet.

1. Traditionnelle

La planification dite « traditionnelle » repose sur l'établissement d'une liste exhaustive des tâches composant le projet et l'estimation du temps de réalisation de chacune d'elle. Les tâches sont réparties entre les différents intervenants par le chef de projet qui se charge ensuite de superviser et de contrôler leur réalisation. C'est une méthode simple et efficace néanmoins réservée à de petites équipes, dont les membres sont très autonomes et ne dépendent pas les uns des autres pour réaliser leurs tâches. La communication au sein de l'équipe est réduite au strict minimum, le chef de projet centralisant toute la gestion des tâches.

2. PERT

Amélioration du diagramme de Gantt (du nom de son inventeur), le PERT permet d'analyser et de représenter de manière logique le réseau des tâches à réaliser au cours du projet. PERT signifie « Program Evaluation and Review Technique », ou TEEP en français, pour « Technique d'Evaluation et Examen des Programmes ». Le type de projet correspondant le mieux à la méthode PERT est sans aucun doute celui impliquant un processus de fabrication. La construction d'un réseau des tâches, mettant en lumière leurs interdépendances et la chronologie de réalisation, va permettre de construire peu à peu le canevas d'élaboration du projet. Cette méthode n'est pas toujours facile à mettre en place, car elle demande une organisation quasi-*UML*

3. Cascade

Parmi les différents types de projets, il y a ceux dont les tâches à réaliser sont particulièrement dépendantes les unes des autres. L'équipe doit dans ce cas attendre que toutes les tâches

identifiées comme prérequis soient terminées avant d'en commencer une autre. Cela nécessite de réaliser les différentes tâches dans un ordre bien précis, et donc que les membres de l'équipe communiquent efficacement entre eux afin que chacun puisse suivre l'avancement des opérations. Dans ces conditions, la méthode en cascade est idéale ! Chaque membre de l'équipe qui réalise une tâche permet à tous d'avancer, assurant ainsi le bon déroulement du projet. Cette démarche de projet permet également d'agrandir l'équipe au besoin, au fur et à mesure de la réalisation des tâches. En revanche, un contrôle absolu de la gestion des dépendances des tâches et de la chronologie de réalisation est indispensable.

4. Chemin critique

Dans le cas où le projet comprend un grand nombre de tâches interdépendantes, la méthode du chemin critique permet de faciliter la planification. Le principe est de découper le projet en tâches critiques et non critiques et en calculant le temps nécessaire à la réalisation de chacune d'elles afin de savoir lesquelles prendront le moins de temps.

Le calcul précis de la durée du projet est possible et les tâches sont mesurées et priorisées. Il est alors possible pour le chef de projet de choisir la réalisation des tâches critiques en priorité afin de continuer ensuite par les tâches dépendantes. Ce type de démarche de projet se retrouve généralement lors des recherches scientifiques ou de la mise au point de produits par les fabricants, car la durée des tâches est mise en avant.

5. Agile

Particulièrement adaptée pour les projets nécessitant une grande flexibilité (exemple projet informatique) et des développements rapides, la méthode agile consiste à découper les différentes étapes du projet en « sprints ». La réalisation des tâches se fait de façon itérative avec des cycles courts, généralement de deux à quatre semaines. Utiliser une méthode de conduite de projet informatique agile nécessite néanmoins une équipe motivée et autonome, dont les membres communiquent facilement entre eux. A l'issue de chaque sprint, la validation des livrables est effectuée, ainsi que la planification de l'itération suivante.

II. Choix de la méthode

Il est impossible de dire qu'une méthode de planification est meilleure qu'une autre. En fonction du type de projet d'entreprise, de la classification des projets en interne et d'exemples d'élaboration de projets antérieurs, il faut sélectionner la méthode la plus adaptée. Il est inutile d'utiliser la méthode traditionnelle, le PERT ou la cascade si vous souhaitez avoir beaucoup de souplesse dans votre organisation, tout comme il est inutile d'opter pour une méthode agile si vous souhaitez absolument conserver un contrôle maximum sur le déroulement du projet. Il n'est même pas forcément nécessaire de choisir une méthode et de l'appliquer à la lettre. Rien ne vous empêche d'utiliser les parties de plusieurs méthodes qui vous intéressent. L'objectif au final, est bien d'avoir une méthode la plus adaptée possible à votre projet. Il est possible d'imaginer par exemple la définition d'un chemin critique pour un projet de grande envergure, avec la mise en place de différents jalons, et de mettre en place un fonctionnement agile pour les développements situés entre ces jalons. Quel que soit le choix de la méthode de planification, et au-delà, du choix de la méthode de gestion de projet, la communication reste un point incontournable. Qu'elle soit centralisée, avec un chef de projet, ou libre entre les membres de l'équipe projet et le client, elle doit être claire et efficace. C'est à ce prix qu'il sera possible de développer un produit correspondant aux attentes du client en respectant les délais prévus.

Méthode	Agile	Flexibilité	Itératif	Taille équipe
Traditionnelle	Non	Non	Non	Petite
Agile	Oui	Oui	Oui	Petite/moyenne
Cascade	Non	Non	Non	Toute taille
PERT	Non	Non	Non	Toute taille
Chemin critique	Non	Non	Non	Toute taille

Tableau 1 : comparatif des méthodes de gestion de projet

III. Présentation de la méthode choisie

La méthode Scrum Initiée par Hirotaka Takeuchi and Ikujiro Nonaka puis formalisée par Ken Schwaber et Jeff Sutherland, cette méthode propose un cadre très structuré pour appliquer les principes de l'agilité.

Le Sprint, le cœur de Scrum

Cette approche repose sur des itérations de 2 à 4 semaines. Ce sont **les fameux "Sprints"**. Il s'agit des sous-parties d'un projet comme le définit le principe Agile. Chaque Sprint a pour objectif de livrer au client une version potentiellement utilisable du produit.

Les Sprints successifs ajoutent des fonctionnalités au produit ou améliorent celles déjà développées. On parle d'incrément de produit.

Un Sprint démarre lorsque le précédent est terminé. Il s'agit d'un processus incrémental.

Ce cadre repose sur 3 piliers que sont :

- **la transparence** : élaboration d'un standard commun pour permettre une compréhension partagée.
- **l'inspection** : des vérifications sont effectuées régulièrement.
- **l'adaptation** : en cas de dérive constatée lors de l'inspection, des ajustements sont décidés.

Les Sprints se structurent autour de plusieurs outils organisationnels (appelés événements) :

- **Sprint planning (Planification du sprint)** : réunion pour sélectionner et planifier les priorités de chaque Sprint en terme de liste des fonctionnalités produit (Sprint Backlog).
- **Scrum (Mélée quotidienne)** : réunion journalière de coordination entre les membres de l'équipe projet. Elle prend fréquemment la forme de "Stand-up meeting" (réunion de courte durée, 10-15mn, tenue debout).
- **Sprint Review (Revue de Sprint)** : réunion de synthèse à la fin de chaque Sprint afin de valider les fonctionnalités développées.

- **Sprint Retrospective (Rétrospective de Sprint) :** venant immédiatement après la revue de Sprint, il s'agit d'un bilan dont l'objectif est l'amélioration continue des pratiques. L'équipe échange sur les réussites, les difficultés, relève ce qui a fonctionné ou non. Avec toujours des leçons à tirer pour les prochains Sprints.

Comprenant des entrants et sortants du processus, appelés "artéfacts"

- **Product Backlog :** liste des fonctionnalités du produit.
- **Sprint Backlog :** planification des éléments du Product Backlog à mettre en œuvre lors du Sprint pour livrer l'incrément de produit doté des fonctionnalités requises pour cette étape. Le Sprint Backlog n'est pas figé, mais est amené à évoluer durant le Sprint.
- **L'incrément de produit :** déjà évoqué plus haut.

Avec des rôles définis pour chacun :

- **Product Owner - PO (propriétaire du produit) :** l'expert métier, **le maître d'ouvrage**, représente le client et intervient sur le côté fonctionnel.
- **Scrum Master (maître de mêlée) :** le coordinateur du projet et le garant du respect de la méthode Scrum.
- **Team (équipe) :** les autres intervenants sur le projet (notamment les développeurs).



Partie 2 : ANALYSE ET CONCEPTION

Chapitre 3 : ANALYSE

I. Méthode d'analyse et Choix de la méthode

Un projet informatique, quelle que soit sa taille et la portée de ses objectifs, nécessite la mise en place d'un planning organisationnel tout au long de son cycle de vie. Il est ainsi nécessaire de définir la notion de méthode.

Une méthode, dans le contexte informatique, peut être définie comme une démarche fournissant une méthodologie et des notations standards qui aident à concevoir des logiciels de qualité. Une méthode d'analyse assure donc la pertinence, la représentativité et la hiérarchisation des besoins.

L'analyse se focalise sur quatre grands types d'information:

- A qui est destinée l'application ? Quelles sont les caractéristiques des utilisateurs finaux?
- Quels objectifs professionnels cette application va apporter aux utilisateurs?
- Dans quel contexte l'utilisation de cette application va-t-elle s'inscrire?
- De quelle façon, ces utilisateurs parviennent-ils à réaliser ses objectifs?

Dans cette partie, nous allons présenter deux outils de modélisation à savoir MERISE et UML :

1. MERISE

Merise propose une méthode de conception et de développement de Systèmes d'informations complète, détaillée, en grande partie formalisée, qui garantit (en principe) une informatisation réussie. Elle fut créée en 1978 sous l'impulsion du Ministère français de la Recherche et de l'industrie. Dès 1980, c'est devenu un standard dans le domaine des systèmes intégrés de gestion. Ce système existe seulement dans les pays francophones. Le but de cette méthode est d'arriver à concevoir un système d'information. La méthode MERISE est basée sur la séparation des données et des traitements à effectuer en plusieurs modèles conceptuels et physiques. C'est d'ailleurs son point fort.

La conception du système d'information avec la méthode MERISE se fait par étapes, afin d'aboutir à un système d'information fonctionnel reflétant une réalité physique. Il s'agit donc de valider une à une chacune des étapes en prenant en compte les résultats de la phase précédente. D'autre part, les données étant séparées des traitements, il faut vérifier la concordance entre données et traitements afin de vérifier que toutes les données nécessaires aux traitements sont présentes et qu'il n'y ait pas de données superflues. Cette succession d'étapes est appelée cycle d'abstraction pour la conception des systèmes d'information :

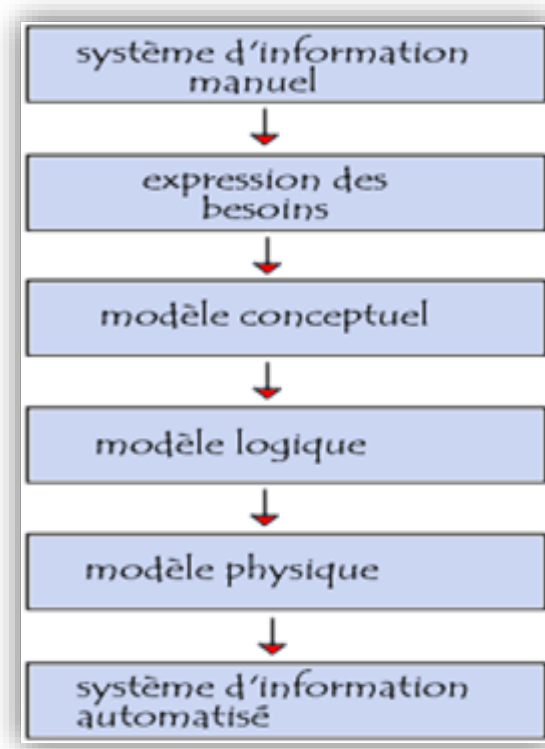


Figure 1: Cycle d'abstraction

Ainsi, la méthode MERISE s'appuie sur un certain nombre d'outils qui modélisent le système selon les niveaux d'analyse (abstraction) ou degré d'invariance :

NIVEAU CONCEPTUEL : s'attache aux invariants de l'entreprise ou de l'organisme du point de vue métier. Le compte rendu de ce niveau est de

Matérialiser, sous forme de dessins normalisés, de modèles complètes par un dossier explicatif.

LE NIVEAU LOGIQUE : Ce niveau décrit la nature des ressources qui sont utilisées pour supporter la description statique et dynamique du système d'information. Ces ressources peuvent être humaines et/ou matérielles et logicielles. Il s'attache à préciser comment on organise les données de l'entreprise ou d'une organisation.

LE NIVEAU PHYSIQUE : permet d'établir la manière concrète dont le système sera mis en place. Il représente le résultat informatique, il dépend des logiciels de développement nécessaires à la programmation et la manipulation des données.

Niveaux d'analyse	Données	Traitements
Niveau conceptuel	Modèle Conceptuel de	Modèle conceptuel des

	Données (MCD)	traitements (MCT)
Niveau logique	Modèle Logique des Données (MLD)	Modèle Organisationnel des Traitements (MOT)
Niveau physique	Modèle Physique des Données (MPD)	Modèle physique des traitements (MPT)

Tableau 2 : Niveaux d'analyse

Modèle Conceptuel de Données (MCD) : décrit la signification des données sur lesquelles reposent le système d'information et les structures.

Modèle conceptuel des traitements (MCT) : formalise les activités du domaine étudié.

Modèle Logique des Données (MLD) : fournit une description des données tenant compte des moyens informatiques mis en œuvre.

Modèle Organisationnel des Traitements (MOT) : décrit le fonctionnement du domaine étudié en présentant les ressources mises en œuvre et leur organisation.

Modèle Physique des Données (MPD) : est une description de la base de données ou de l'ensemble des fichiers correspondants aux données gérées par le système d'information.

Modèle physique des traitements (MPT) : décrit les spécificités des différents modules de traitement.

❑ Avantages

Pour de petites bases de données, MERISE est généralement l'une des meilleures solutions du point de vue de l'architecture. Un des avantages de MERISE est quasiment de pouvoir être utilisé par un non informaticien. Ce qui permet de faire évoluer en temps réel la structure informatique d'une entreprise à mesure que celle-ci évolue.

MERISE est une méthodologie qui dispose de beaucoup d'outils de développement informatique tel que AMC designer, Power Designer, et est adaptée aux nouvelles technologies : architecture client/serveur, interfaces graphiques, intranet/internet, démarche de développement rapide.

❑ Inconvénients

MERISE n'est pas orientée objet (même si des évolutions en ce sens sont apparues en même temps que d'autres méthodes plus populaires aujourd'hui). Ce n'est pas une méthode cognitive mais une méthode technique, d'autant plus qu'elle manque d'abstraction.

Elle est un peu adaptée pour les environnements distribués où plusieurs applications sont externes à un domaine d'interagir avec le modèle d'application. En outre, elle n'est pas en mesure de modéliser les données à caractère sémantique.

2. UML

UML (en anglais Unified Modeling Language ou « langage de modélisation unifié ») est un langage de modélisation graphique à base de pictogrammes. Il est apparu dans le monde du génie logiciel, dans le cadre de la « conception orientée objet ». Couramment utilisé dans les projets logiciels, il peut être appliqué à toutes sortes de systèmes ne se limitant pas au domaine informatique. UML est l'accomplissement de la fusion de précédents langages de modélisation objet : Booch, OMT, OOSE. Principalement issu des travaux de Grady Booch, James Rumbaugh et Ivar Jacobson, UML est à présent un standard défini par l'Object Management Group (OMG). La dernière version diffusée par l'OMG est UML 2.4.1.

UML définit 6 modèles pour la représentation des systèmes :

Modèle des classes : capture la structure statique,

Modèle des états : exprime le comportement dynamique des objets,

Modèle des cas d'utilisation : décrit les besoins de l'utilisateur,

Modèle d'interaction : représente les scénarios et les flots de messages,

Modèle de réalisation : montre les unités de travail,

Modèle de déploiement : précise la répartition des processus.

UML définit 9 diagrammes pour élaborer les 6 modèles :

- ★ Diagramme de classe : représentation de la structure statique en termes de classes et relations,
- ★ Diagramme de séquence : représentation temporelle des objets et leurs interactions,
- ★ Diagramme de collaboration : représentation spatiale des objets, des liens et des interactions,
- ★ Diagramme d'objet : représentation des objets et leurs relations (diagramme de collaboration simplifiée sans envois des messages),
- ★ Diagramme d'états-transitions : représentation du comportement d'une classe d'objet en termes d'états,
- ★ Diagramme d'activités : représentation d'une opération en termes d'actions,

- ★ Diagramme de cas d'utilisation : représentation des fonctions du système du point de vue de l'utilisateur,
- ★ Diagramme de composant : représentation des composants physiques d'une application,
- ★ Diagramme de déploiement : représentation du déploiement des composants sur les dispositifs matériels.

❑ **Avantages**

Étant donné qu'il est le fruit de la fusion de plusieurs méthodes objets, il utilise l'approche objet en présentant un langage de description universel. Il permet grâce à un ensemble de diagrammes très explicites, de représenter l'architecture et le fonctionnement des systèmes informatiques complexes en tenant compte des relations entre les concepts utilisés et l'implémentation qui en découle.

UML est un langage formel et normalisé, il permet le gain de précision, encourage l'utilisation d'outils et constitue à cet effet un gage de stabilité UML est un support de communication performant car il cadre l'analyse et facilite la compréhension de représentations abstraites complexes Son caractère polyvalent et sa souplesse en font un langage universel

❑ **Inconvénients**

La mise en pratique d'UML nécessite un apprentissage et passe par une période d'adaptation Un inconvénient que certains développeurs pourraient trouver en utilisant UML est le temps qu'il faut pour gérer et maintenir des diagrammes UML.

UML a une notation majoritairement graphique pouvant se révéler insuffisante ou trop chargée d'un point de vue expressivité et les liens sont parfois difficiles entre les vues et les diagrammes d'une même application.

3. Choix de la méthode

La méthode Merise et le langage UML sont deux approches de modélisation d'un système d'information. Cela dit, la méthode Merise, s'arrête au niveau organisationnel pour les traitements et les données et ne s'occupe pas de l'interface utilisateur. Elle est très adaptée à un contexte de création d'application mais pas forcément à un problème de maintenance ou de seconde informatisation.

UML, contrairement à son prédécesseur MERISE qui pourtant est utilisée de nos jours, donne un sens intéressant à l'approche objet et couvre de plus tout le cycle de réalisation du logiciel.

UML est avant tout un support de communication performant, qui facilite la représentation et la compréhension de solutions objet.

Sa notation graphique permet d'exprimer visuellement une solution objet, ce qui facilite la comparaison et l'évaluation de solutions. L'aspect formel de sa notation, limite les ambiguïtés et les incompréhensions. Son indépendance par rapport aux langages de programmation et aux domaines d'application ainsi aux processus, en fait un langage universel. UML est donc bien plus qu'un simple outil qui permet de "dessiner" des représentations graphiques... Il permet également de parler un langage commun, normalisé mais accessible, car visuel. C'est pourquoi UML a été choisi comme outil de modélisation de notre système.

II. Spécification des besoins du Système

1. Analyse des besoins

L'analyse des besoins est la phase initiale de toute mise en œuvre de projet. Elle permet de définir de manière concise et détaillée les fonctions à implémenter, l'accessibilité et la confidentialité des informations qui y seront stockées.

Dans la démarche d'analyse, la phase de modélisation est l'une des étapes à franchir. Cette méthode permet d'atteindre plusieurs de nos objectifs préalablement fixés qui sont :

- Faciliter l'accès à l'information du tuteur par rapport à son élève
- Avoir un meilleur suivi de l'élève de la sixième au terminal
- Avoir

2. Analyse des rôles et des acteurs

Un acteur est un rôle joué par une entité externe qui agit sur le système en échangeant de l'information. Dans notre cas nous avons le client et l'ouvrier comme acteurs externes. La manière d'accéder aux services de l'application est la même pour tous les prestataires (entreprises et clients), la différence réside sur les droits d'accès et les limites applicatives et fonctionnelles de chacun. Ce qui nous permet donc de distinguer plusieurs rôles :

- L'élève
- Le tuteur
- Le professeur
- L'administration de l'école

3. Spécifications des besoins fonctionnelles du système

Pour s'assurer qu'un produit puisse satisfaire un client on se préoccupe en priorité des services qu'il rend. Dans notre cas on se préoccupe des fonctionnalités du système. Ces fonctionnalités sont les contraintes environnementales et d'implémentation aidant à la performance, à la fiabilité du système à mettre en place.

Ainsi, nous pourrions noter que le système doit :

- ❖ Être capable de sécuriser l'accès par l'authentification de chaque utilisateur (professeur élève, administration).
- ❖ Doit offrir aux utilisateurs une interface ergonomique, facile d'adaptation.
- ❖ Doit être fiable et interactif.
- ❖ Doit permettre aux élève de consulter ses notes, son emploi du temps, ses absences, ses dates de devoir et examen
- ❖ Doit permettre aux professeurs de gérer (consulter, enregistrer et supprimer) les notes, les absences, les devoirs et les examens
- ❖ Doit permettre à l'administration de gérer (consulter, enregistrer et supprimer) les notes, les absences, les professeurs, les élèves, les emplois du temps, les devoirs, les examens Et les bulletins.
- ❖ Garantir l'intégrité et la cohérence des données.
- ❖ Pouvoir envoyer des notifications.

4. Spécification des besoins non fonctionnelles du Système

A part les besoins fondamentaux, notre système doit répondre aux critères suivants :

La rapidité de traitement : En effet, vu le nombre important des transactions quotidiennes, il est impérativement nécessaire que la durée d'exécution des traitements s'approche le plus possible du temps réel.

La performance : Un logiciel doit être avant tout performant c'est-à-dire à travers ses fonctionnalités, répond à toutes les exigences des usagers d'une manière optimale.

La convivialité : Le futur logiciel doit être facile à utiliser. En effet, les interfaces utilisateurs doivent être conviviales c'est-à-dire simples, ergonomiques et adaptées à l'utilisateur.

III. Règles de gestion

Règle 1 : l'élève est inscrit une et une seule fois dans une année scolaire.

Règle 2 : chaque inscription concerne une et une seule classe.

Règle 3 : une classe peut faire l'objet de plusieurs inscriptions.

Règle 4 : une inscription concerne une année scolaire et une classe et un programme

Règle 5 : une classe peut avoir plusieurs programmes

Règle 6 : chaque élève est identifié par un code

Règle 7 : une matière appartient à un seul programme

Règle 8 : un programme est composé de plusieurs matières

Règle 9 : un élève a un seul tuteur

Règle 10 : un tuteur peut concerner une ou plusieurs élèves

Règle 11 : un professeur enseigne une ou plusieurs matières

Règle 12 : un professeur enseigne une ou plusieurs classes

Règle 13 : un tuteur a une ou plusieurs élèves

Chapitre 4: Conception

I. Diagrammes de use case :

Les diagrammes de cas d'utilisation sont des diagrammes UML utilisés pour donner une vision globale du comportement fonctionnel d'un système logiciel. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d'un projet, mais pour le développement, les cas d'utilisation sont plus appropriés. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système. Il est une unité significative de travail. Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs (actors), ils interagissent avec les cas d'utilisation (use cases).

❖ Liste des cas d'utilisations

ACTEURS	CAS D'UTILISATION
Elève	<ul style="list-style-type: none">• Consulter notes• Consulter absence• Consulter planning cours• Consulter planning devoir• Consulter convocation• S'authentifier
Parent d'élève	<ul style="list-style-type: none">• Consulter notes• Consulter absence• Consulter planning cours• Consulter planning devoir• Consulter convocation
Professeur	<ul style="list-style-type: none">• Gérer évaluation• Gérer notes• Ajouter séances de cours• Gérer absence

	<ul style="list-style-type: none"> • Consulter planning • Gérer informations • S'authentifier
Administration	<ul style="list-style-type: none"> • Gérer planning • Gérer professeur • Gérer matière • Gérer programme • Gérer absence • Générer bulletin • Gérer inscription • S'authentifier • Gérer compte

Tableau 3 : liste des cas d'utilisation

❖ Cas d'utilisation élève

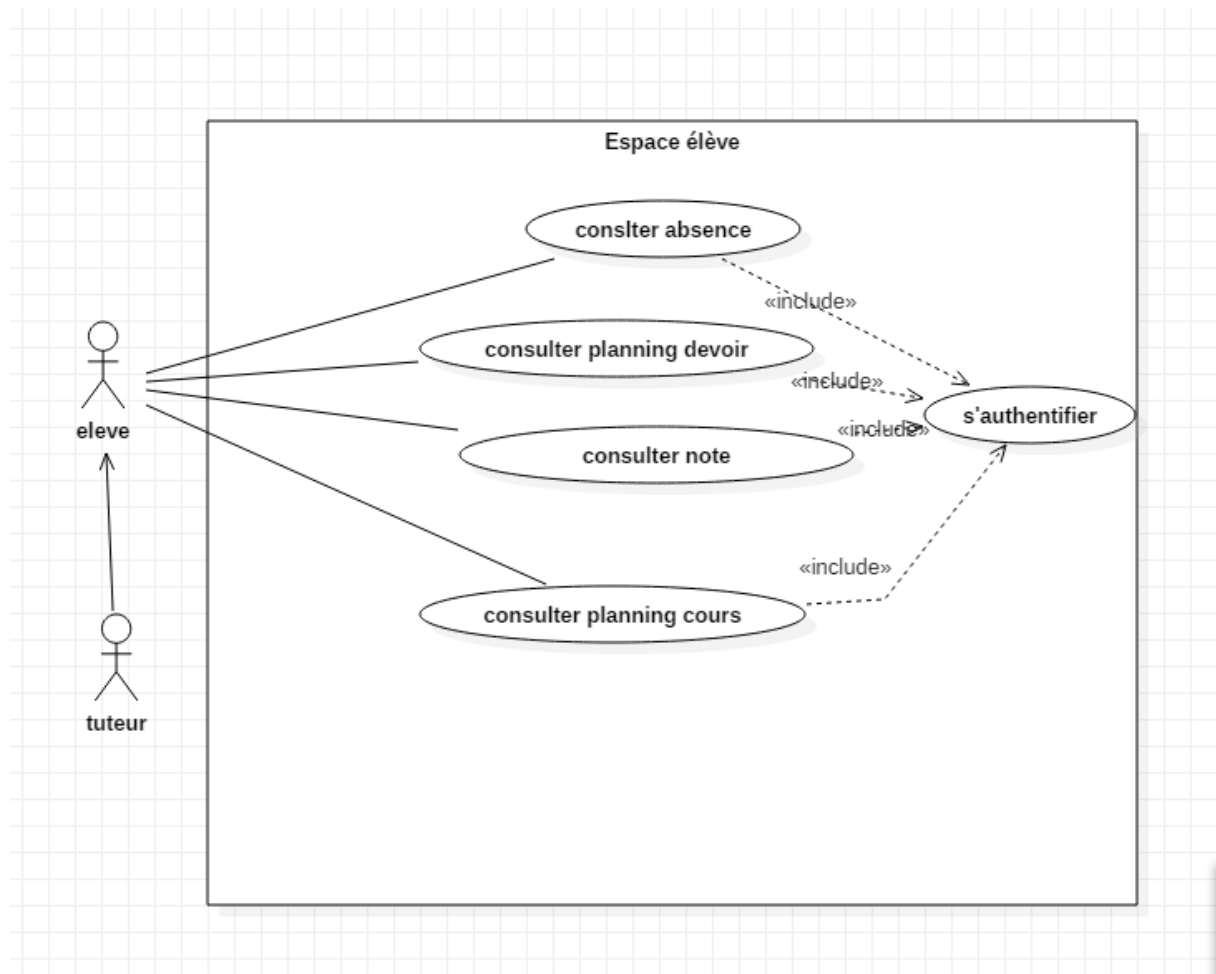


Figure 2: Cas d'utilisation de l'élève

❖ Description textuelle:

Cas n°1

Nom : Consulter note

Acteur(s) : élevé

Description : tous les élevés peuvent consulter les notes

Précondition(s) : L'utilisateur doit être authentifié en tant qu'élevé (Cas d'utilisation <<se connecter>>).

Démarrage : L'utilisateur a choisi la page <<voir notes>>.

Le scénario nominal :

1. Le système envoie une notification à l'utilisateur

2. L'utilisateur choisit l'action ouvrir l'application par le biais de la notification.
3. Le système affiche les informations concernant l'élève

Cas n°2

Nom : Consulter planning cours

Acteur(s) : élève

Description : tous les élèves peuvent consulter leur planning

Précondition(s) : L'utilisateur doit être authentifié en tant qu'élève (Cas d'utilisation <<se connecter>>).

Démarrage : L'utilisateur a choisi la page << accueil>>.

Le scénario nominal :

1. Le système lui affiche la page d'accueil de l'application
2. L'utilisateur choisit l'action voir le planning.
3. Le système affiche les informations du planning concernant l'élève

❖ Cas du Professeur

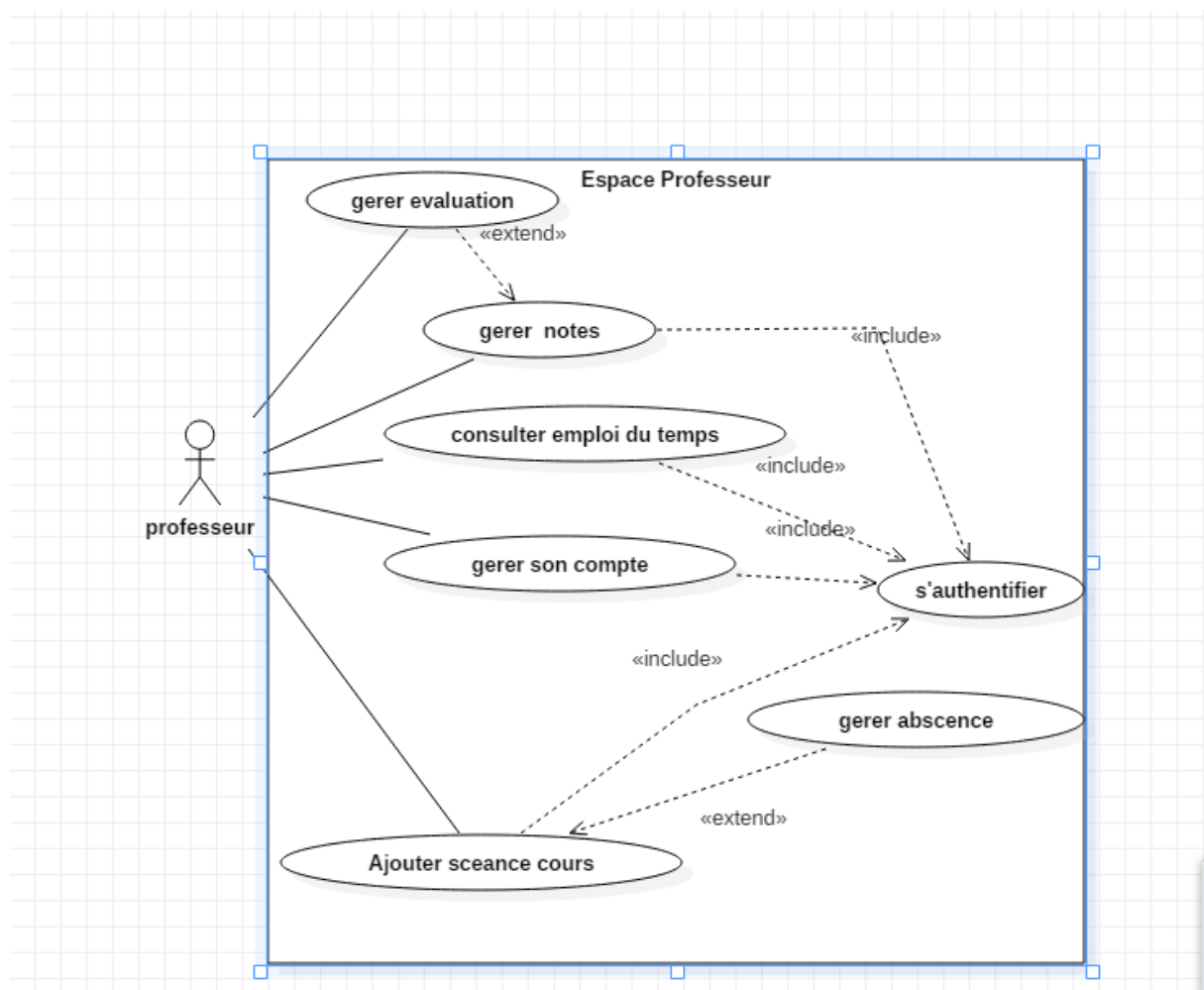


Figure 3: Cas d'utilisation Professeur

Cas n°1

Nom : Gérer Notes

Acteurs(s) principal : Professeur

Acteur(s) secondaire: élève

Description : La gestion des notes rassemble toutes les actions qu'un professeur peut effectuer sur les notes des élèves.

Précondition(s) : L'utilisateur doit être authentifié en tant que professeur (Cas d'utilisation <<Authentification>>).

Démarrage : L'utilisateur a choisi la page << gérer notes >>.

Le scénario nominal :

1. Le professeur choisit la classe
2. Le professeur choisit l'évaluation
3. Le professeur lui affecte une note

Cas n°2

Nom : Ajouter séance de cours

Acteurs(s) principal : Professeur

Description : l'ajout de la séance de cours rassemble l'action d'ajout qu'un professeur peut effectuer pour les séances de cours.

Précondition(s) : L'utilisateur doit être authentifié en tant que professeur (Cas d'utilisation <<Authentification>>).

Démarrage : L'utilisateur a choisi la page << Ajouter séance de cours >>.

Le scénario nominal :

1. Le système lui affiche la liste des plannings des cours
2. Le professeur choisit le planning du cours
3. Le système lui affiche un formulaire de saisie
4. Le professeur saisit les informations par rapport au séance du cours

❖ Use Case Administrateur

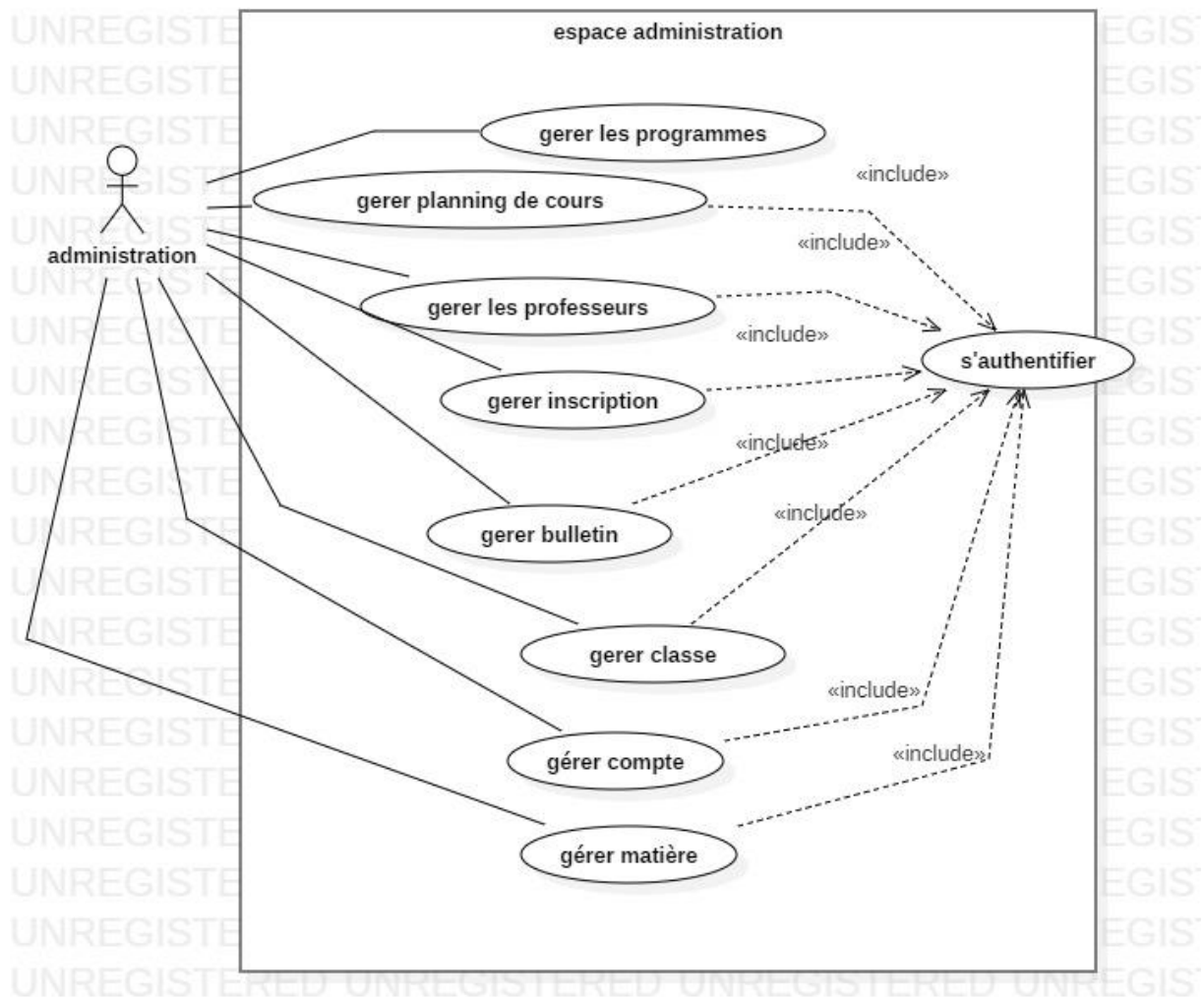


Figure 4: : Cas d'utilisation administration

Cas n°1

Nom : gérer emploi temps

Acteur(s) : administration

Description : La gestion (ajout, modification suppression) + de l'emploi du temps doit être possible pour l'administration

Précondition(s) : L'utilisateur doit être authentifié en tant que membre de l'administration (Cas d'utilisation <<se connecter>>).

Démarrage : L'utilisateur a choisi la page << gérer des emplois du temps>>.

Le scénario nominal :

1. L'utilisateur choisit la classe
2. L'utilisateur choisit le programme
3. L'utilisateur choisit la matière
4. L'utilisateur choisit l'heure début
5. L'utilisateur choisit l'heure de fin
6. L'utilisateur ajoute dans le planning

Cas n°2

Nom : gérer bulletin

Acteur(s) : administration

Description : La gestion (ajout, modification suppression) des bulletins -doit être possible pour l'administration

Précondition(s) : L'utilisateur doit être authentifié en tant que membre de l'administration (Cas d'utilisation <<se connecter>>).

Démarrage : L'utilisateur a choisi la page << gestion des élèves>>.

Le scénario nominal :

1. L'utilisateur choisit la classe
2. L'utilisateur choisit l'élève
3. L'utilisateur le semestre
4. L'utilisateur génère les bulletins

Cas n°3

Nom : gérer élève

Acteur(s) : administration

Description : La gestion (ajout, modification suppression) des élèves du doit être possible pour l'administration

Précondition(s) : L'utilisateur doit être authentifié en tant que membre de l'administration (Cas d'utilisation <<se connecter>>).

Démarrage : L'utilisateur a choisi la page << gestion des élèves>>.

Le scénario nominal :

1. L'utilisateur saisit les informations de l'élève
2. L'utilisateur choisit la classe de l'élève

L'utilisateur valide l'inscription de l'élève

II. Diagrammes de Séquence

Les diagrammes de cas d'utilisation modélisent à QUOI sert le système, en organisant les interactions possibles avec les acteurs.

Les diagrammes de classes permettent de spécifier la structure et les liens entre les objets dont le système est composé : il spécifie QUI sera à l'œuvre dans le système pour réaliser les fonctionnalités décrites par les diagrammes de cas d'utilisation.

Les diagrammes de séquences permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs :

- a. Les objets au cœur d'un système interagissent en s'échangeant des messages.
- b. Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine).

❖ *Authentification*

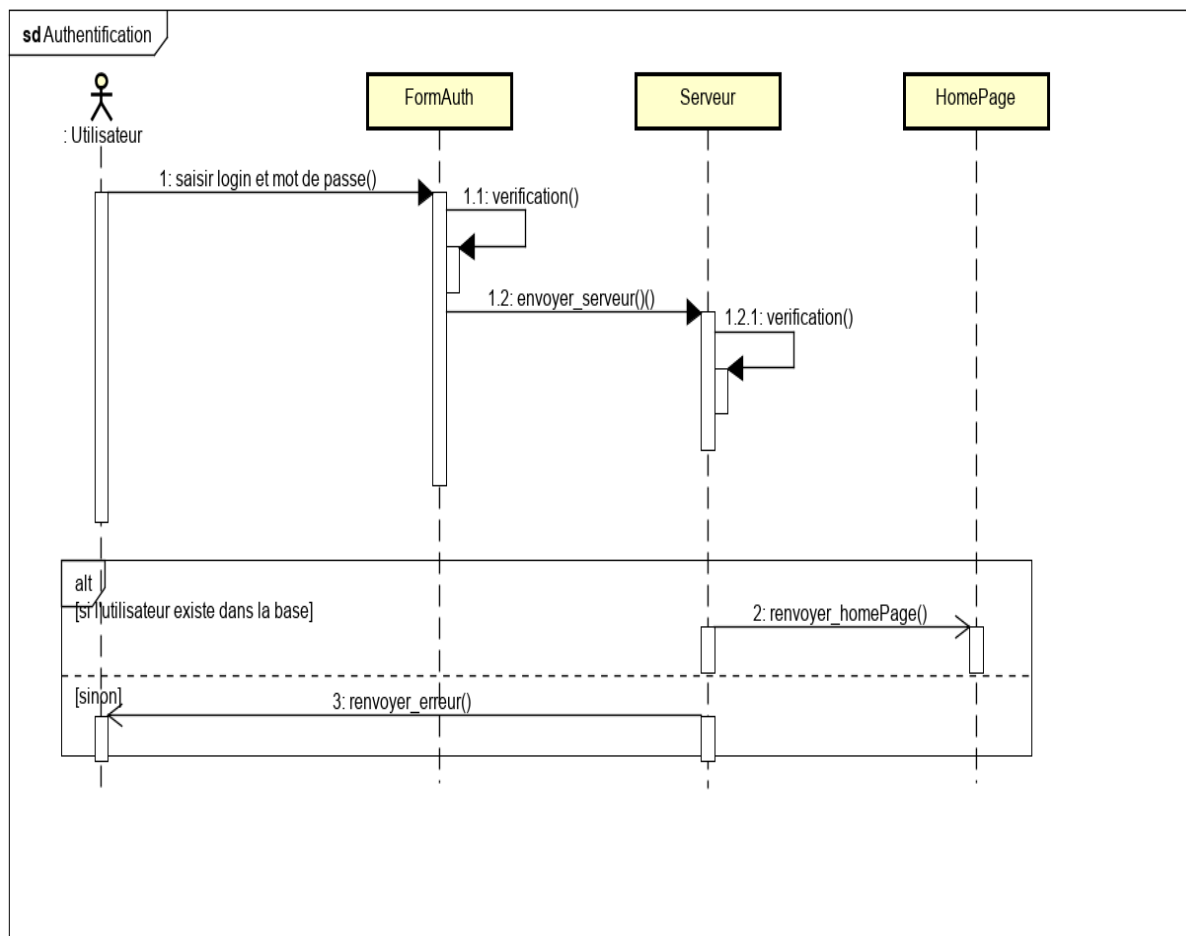


Figure 5: Diagramme séquence authentification

❖ Consulter Absence

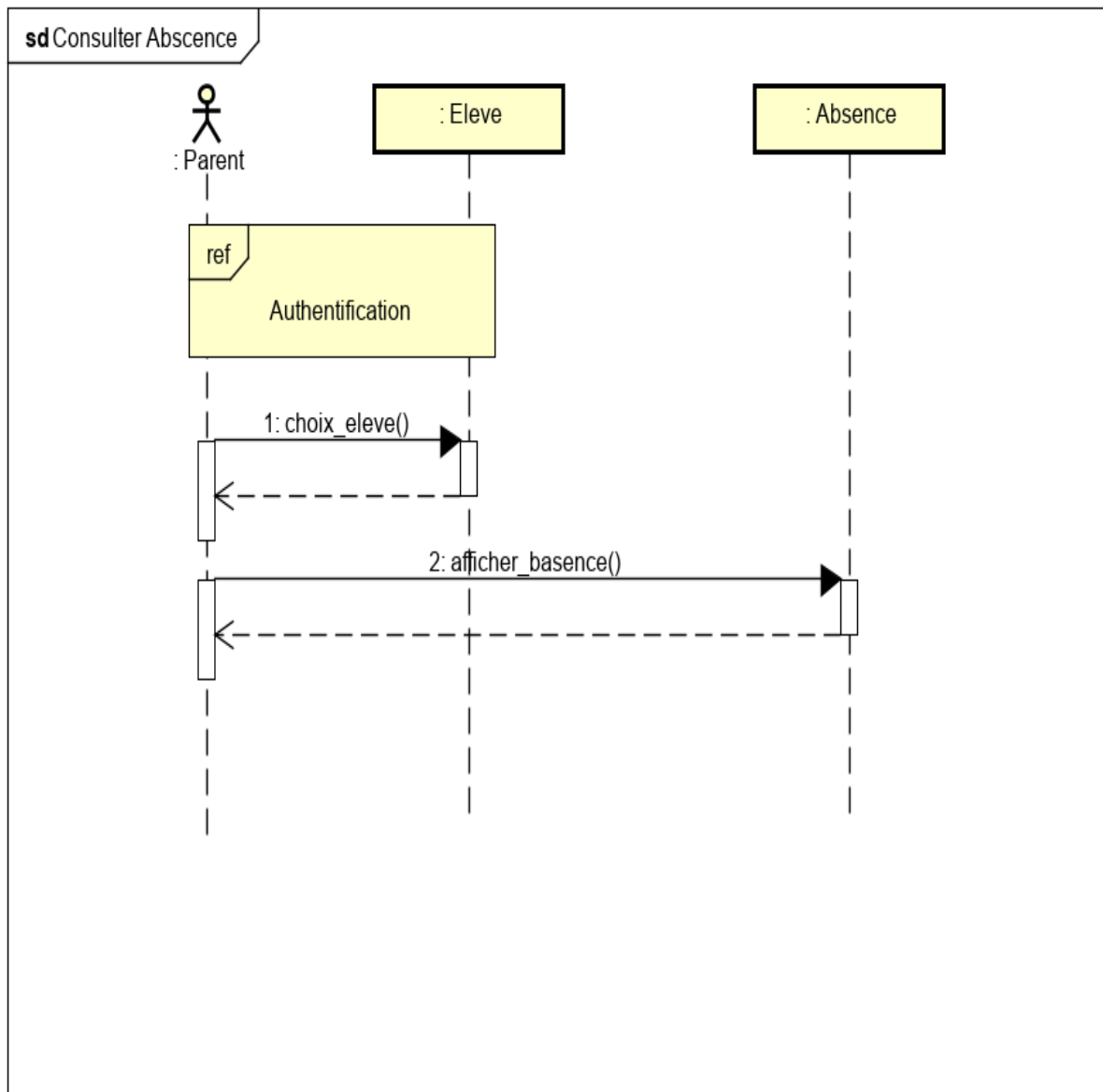


Figure 6: Diagramme séquence consulter absence

❖ *Générer Planning*

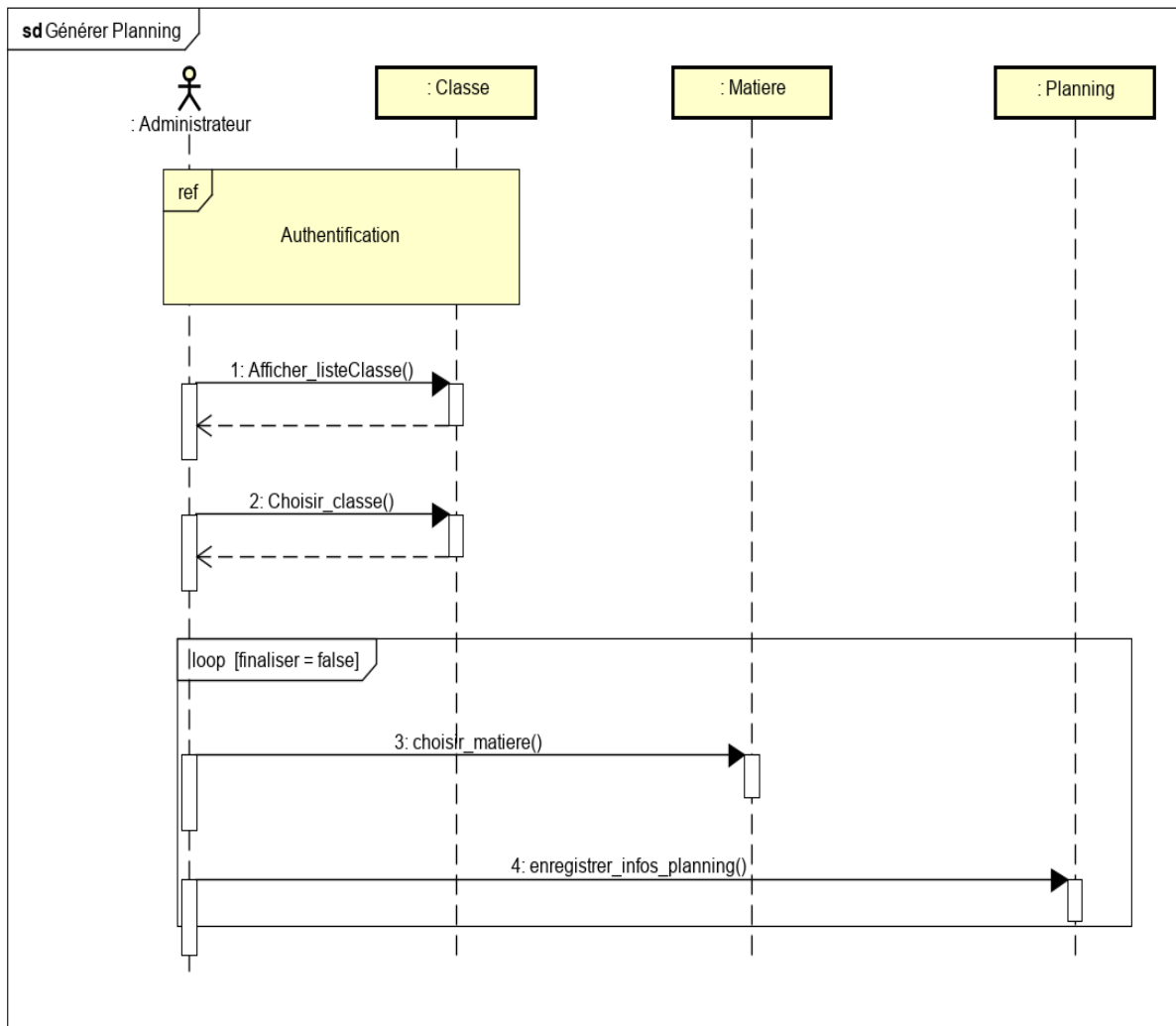


Figure 7: Diagramme séquence enregistrer Planning

❖ *Générer Bulletin*

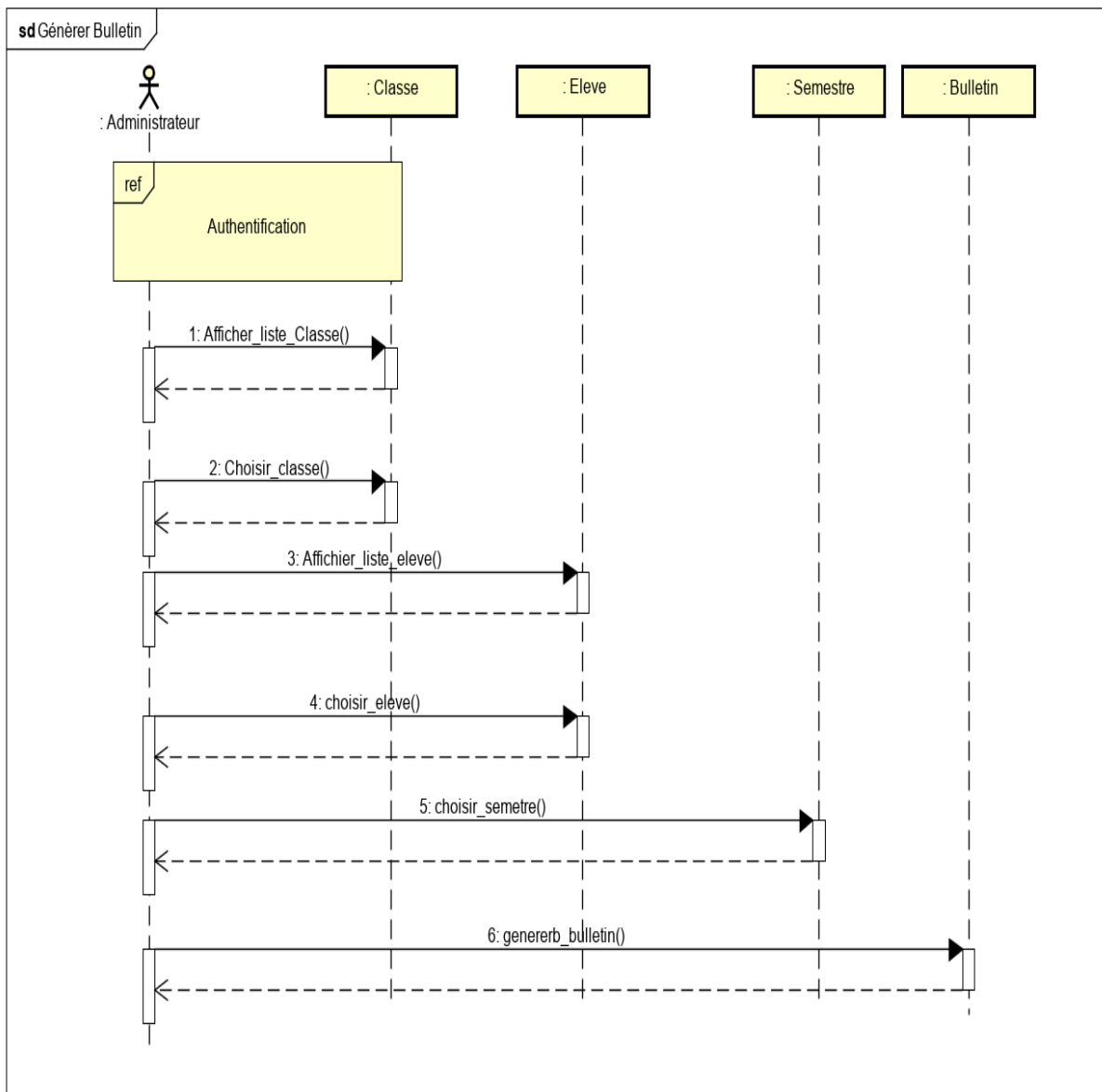


Figure 8: Diagramme séquence générer bulletin

IV. Diagramme d'objet

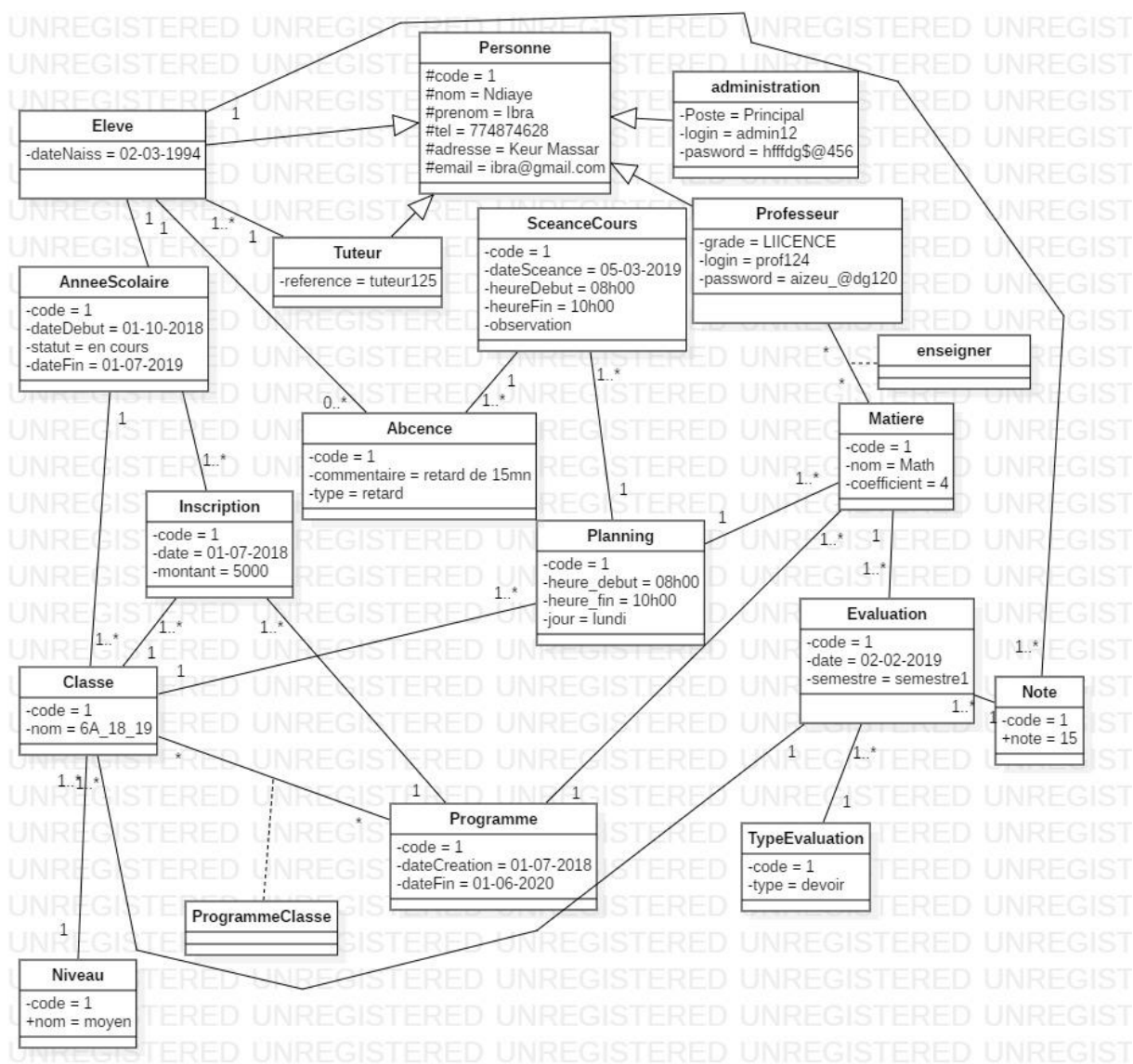


Figure 10: Diagramme d'objet

V. Diagramme de déploiement

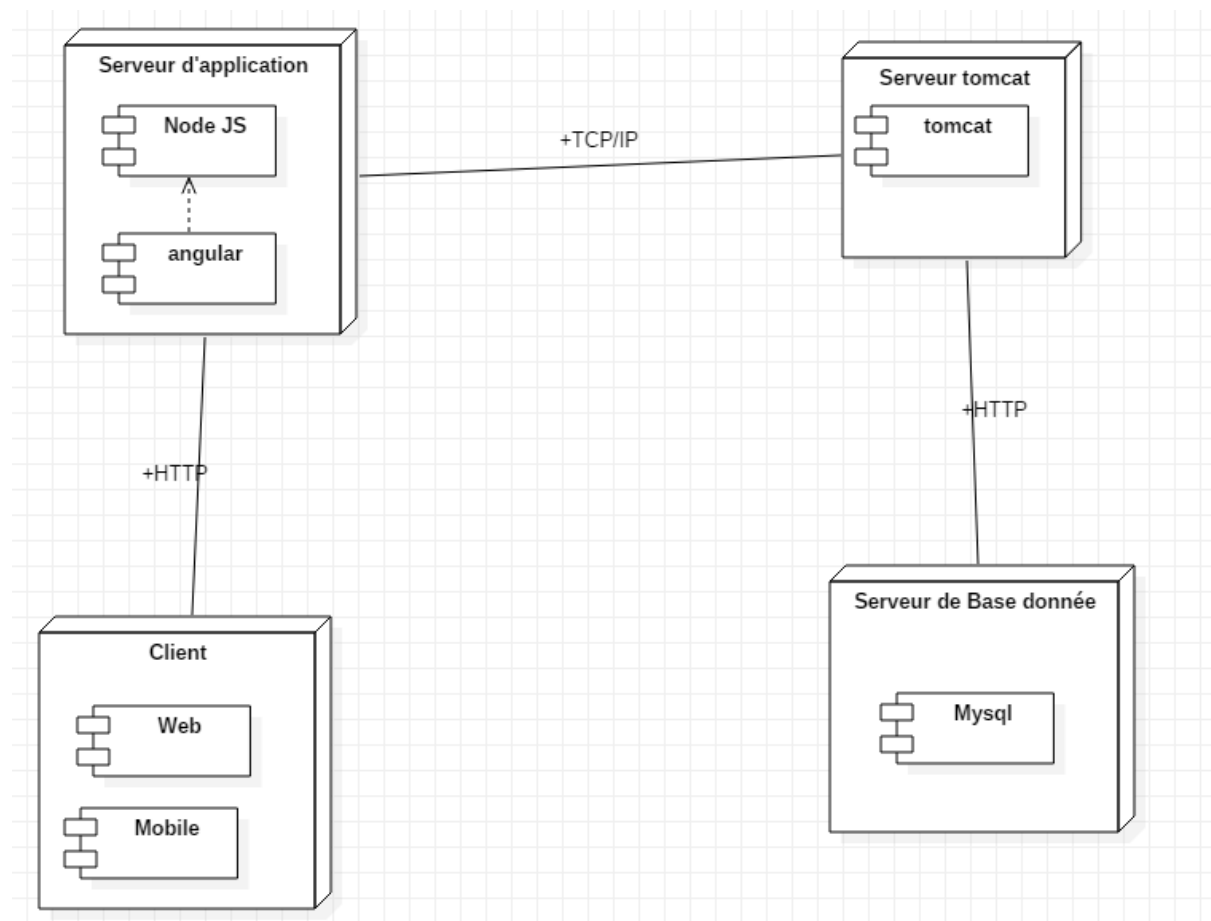


Figure 5-11: Diagramme de déploiement



Partie 3 : MISE EN ŒUVRE DE LA SOLUTION (REALISATION) ET POLITIQUE DE SECURITE

Chapitre 5 : Architecture de la solution

L'architecture de SGBD facilite la conception, le développement, la mise en œuvre et la maintenance d'une base de données. Une base de données stocke des informations critiques pour une entreprise. La sélection de l'architecture de base de données appropriée facilite un accès rapide et sécurisé à ces données.

I. Architecture 1 tiers



Figure 12: Architecture 1 tiers

L'architecture de base de données la plus simple est **un niveau sur** lequel le client, le serveur et la base de données résident tous sur le même ordinateur. Chaque fois que vous installez une base de données sur votre système et y accédez pour vous entraîner aux requêtes SQL, il s'agit d'une architecture à 1 niveau. Mais une telle architecture est rarement utilisée en production.

II. Architecture 2 tiers

Une architecture à deux niveaux est une architecture de base de données où

1. La couche de présentation s'exécute sur un client (PC, mobile, tablette, etc.)
2. Les données sont stockées sur un serveur.

Une interface d'application appelée ODBC (Open Data base Connectivite) est une API qui permet au programme côté client d'appeler le SGBD. Aujourd'hui, la plupart des SGBD offrent des pilotes ODBC pour leur SGBD. L'architecture à 2 niveaux fournit une sécurité supplémentaire au SGBD, car il n'est pas exposé directement à l'utilisateur final.

Un exemple d'architecture à deux niveaux est un système de gestion de contacts créé à l'aide de MS-Access.

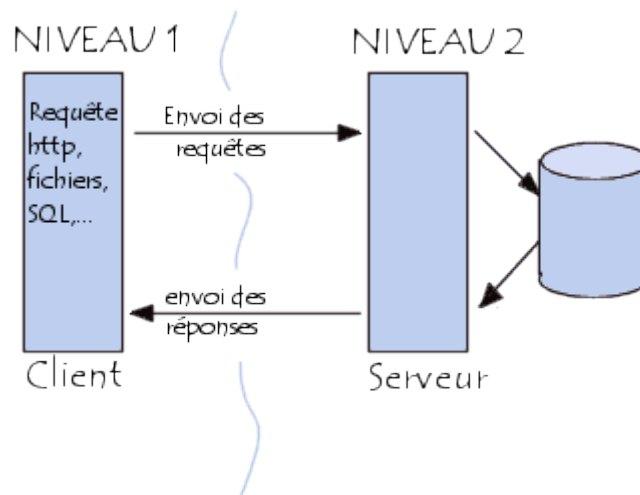


Figure 13: Architecture 2 tiers

Dans l'architecture à deux niveaux ci-dessus, nous pouvons voir qu'un serveur est connecté aux clients 1, 2m et 3. Cette architecture permet une communication directe et plus rapide.

III. Architecture à 3 niveaux

Le schéma à 3 niveaux est une extension de l'architecture à 2 niveaux. L'architecture à 3 niveaux comporte les couches suivantes

1. Couche de présentation (votre PC, tablette, mobile, etc.)
2. Couche application (serveur)
3. Serveur de base de données

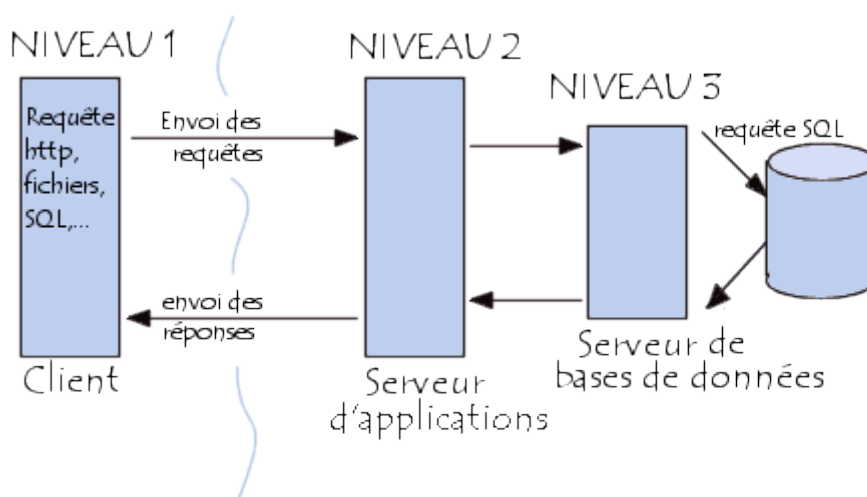


Figure 14: Architecture 3 tiers

Cette architecture de SGBD contient une couche Application entre l'utilisateur et le SGBD, qui est chargée de communiquer la demande de l'utilisateur au système SGBD et d'envoyer la réponse du SGBD à l'utilisateur.

La couche application (couche de logique métier) traite également la logique fonctionnelle, les contraintes et les règles avant de transmettre des données à l'utilisateur ou vers le SGBD.

L'architecture à trois niveaux est l'architecture de SGBD la plus répandue.

Le but de l'architecture à trois niveaux est:

- Pour séparer les applications utilisateur et la base de données physique
- Proposé pour supporter les caractéristiques du SGBD
- Indépendance du programme de données
- Prise en charge de plusieurs vues des données

IV. Architecture N tiers

Les architectures n-tiers doivent permettre de pallier les limites des architectures trois tiers et de concevoir des applications puissantes et simples à maintenir. Ce type d'architecture permet de **distribuer plus librement la logique applicative**, ce qui facilite la répartition de la charge entre tous les niveaux. Cette évolution des architectures trois tiers met en œuvre une **approche objet** pour offrir une plus grande souplesse d'implémentation et faciliter la réutilisation des

développements. Théoriquement, ce type d'architecture supprime tous les inconvénients des architectures précédentes. Elle permet l'utilisation d'interfaces utilisateurs riches et sépare nettement tous les niveaux de l'application. Elle offre de **grandes capacités d'extension** et facilite la gestion des sessions.

L'appellation "n-tiers" peut laisser penser que cette architecture met en œuvre un nombre indéterminé de niveaux de services. Mais ces niveaux restent à un nombre maximum de trois (les trois niveaux d'une application informatique). En fait, l'architecture n-tiers qualifie la distribution d'applications entre de multiples services et non la multiplication des niveaux de service. Ainsi, l'architecture prend toujours en compte les **trois niveaux d'abstraction** d'une application.

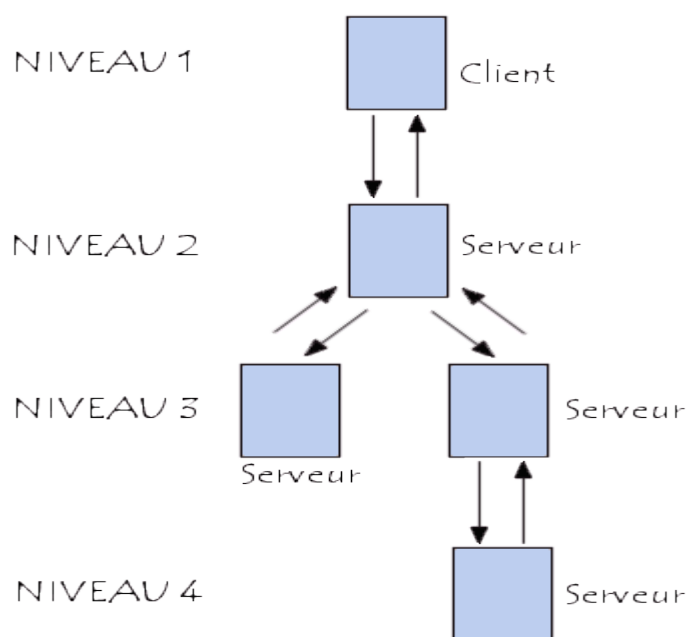


Figure 15: Architecture N tiers

V. Choix de l'architecture

- Le tableau ci-dessous présente les différences qu'apportent les architectures 3-tier aux anciennes architectures 2-tier.

	2 tiers	3 et n tiers
--	---------	--------------

Administration du système	Complexe (la couche application est physiquement répartie sur plusieurs postes clients)	Moins complexe (les applications peuvent être gérées centralement sur le serveur)
Sécurité	Faible (sécurité au niveau des données)	Elevée (raffinée au niveau des services ou des méthodes)
Encapsulation des données	Faible (les tables de données sont directement accessibles)	Elevée (le client fait appel à des services ou méthodes)
Performance	Faible (plusieurs requêtes SQL sont transmises sur le réseaux, les données sélectionnées doivent être acheminées vers le client pour analyse)	Bonne (seulement les appels de services et les réponses sont mis sur le réseau)
Extensibilité	Faible (gestion limitée des liens réseaux avec le client)	Excellente (possibilité de répartir dynamiquement la charge sur plusieurs serveurs)
Réutilisation	Faible (application monolithique sur le client)	Excellente (réutilisation des services et des objets)
Facilité de développement	Elevée	En progression (des outils intégrés pour développer la partie du client et du serveur)
Lien Serveur-serveur	Non	Oui (via le middleware Serveur/Serveur)

Intégration des systèmes déjà en place	Non	Oui (via des passerelles encapsulées par les services ou objets)
Soutien Internet	Faible (les limitations de la bande passante pénalisent le téléchargement d'applications de type "fat-client")	Excellente (les applications de type "thin-client" sont facilement téléchargeable et les appels aux services repartissent la charge sur un ou plusieurs serveurs)
Sources de données hétérogènes	Non	Oui (les applications 3-tier peuvent utiliser plusieurs bases de données dans la même transaction)
Choix de communications de type "riche"	Non (synchrone et RPC)	Oui (gestion asynchrone de messages , files de livraison, publication et abonnement, "broadcast")
Flexibilité d'architecture matérielle	Limitée	Excellente (possibilité de faire résider les couches 2 et 3 sur une ou plusieurs machines)
Relève en cas de pannes	Faible	Excellente (possibilité d'avoir la couche du centre "middle-tier" sur plusieurs serveurs)

Tableau 4 : Comparatifs des différentes architectures

Pour moi utiliser une architecture à 3 niveaux est beaucoup plus propre. Cela permet de diviser les tâches et par conséquent d'avoir des développeurs spécialisés sur un des trois niveaux. De plus, la flexibilité qu'offre ce genre d'infrastructure est à prendre en considération surtout si vous travaillez sur un projet qui peut être amenée à évoluer.

Chapitre 6 : Présentation de la solution (REALISATION)

I. Choix du Framework de Front end

1. Partie WEB

a) REACT

ReactJS est une bibliothèque JavaScript, ouverte par Facebook en **2013**, qui est utilisée pour la construction d'énormes applications web où les données sont changeables sur une base régulière.

AVANTAGES DE REACTJS :

- Facile à apprendre, en raison de sa simplicité en ce qui concerne la syntaxe
- Haut niveau de flexibilité et de réactivité
- DOM virtuel (modèle d'objet de document)
- Combiné avec ES6 / 7, ReactJS peut travailler avec la charge élevée d'une manière facile.
- Librairie JavaScript 100% open source (reçoit beaucoup de mises à jour et d'améliorations quotidiennes)
- La migration entre les versions est généralement très facile, avec « codemods », fournit par Google, pour automatiser une grande partie du processus.

INCONVÉNIENTS DE REACTJS :

- Manque de documentation officielle – le développement ultrarapide de ReactJS ne laisse aucune place à la documentation appropriée
- React est unopinionated, ce qui signifie que les développeurs ont parfois trop de choix.
- Long à maîtriser, ReactJS nécessite une connaissance approfondie de la façon d'intégrer l'interface utilisateur dans le Framework MVC.

b) Angular

Angular est un Framework JavaScript orienté Composant, dont la version initiale (2.0) fut publiée en septembre **2016**, qui est très utilisé pour la construction d'**applications web hautement interactives**. Une nouvelle version est sortie en début mai 2018.

AVANTAGES D'ANGULAR

- Nouvelles fonctionnalités améliorées telles que RXJS, compilation plus rapide (en moins de 3 secondes)
- Documentation détaillée
- Liaison de données bidirectionnelle qui minimise les risques d'erreurs possibles
- MVVM (Model-View-ViewModel) qui permet aux développeurs de travailler séparément sur la même section de l'application en utilisant le même ensemble de données
- Injection de dépendance des fonctionnalités liées aux composants avec modules et modularité en général

INCONVÉNIENTS D'ANGULAR

- La syntaxe complexe qui vient de la première version d'Angular. Néanmoins, Angular 6 utilise TypeScript 2.7 qui est moins difficile à apprendre
- Les problèmes de migration qui peuvent apparaître lors du passage de l'ancienne version aux plus récentes.

c) VUEJS

Vue.js est un Framework JavaScript, lancé en **2013**, qui convient parfaitement à la création d'**interfaces utilisateur hautement adaptables** et d'**applications Single-page complexes**.

AVANTAGES DE VUE.JS :

- HTML optimisé. Vue.js a de nombreuses caractéristiques similaires avec Angular et cela peut aider à optimiser la gestion des blocs HTML avec une utilisation de différents composants.
- Documentation détaillée
- Adaptabilité. Il fournit une période de transition rapide d'un autre Framework à Vue.js, en raison de la similitude avec Angular et React en termes de conception et d'architecture.
- Intégration impressionnante
- Grande échelle. Vue.js peut aider à développer de très grands gabarits réutilisables qui peuvent être réalisés sans temps supplémentaire
- Petite taille. Vue.js peut peser environ 20Ko en gardant sa vitesse et sa flexibilité qui permet d'atteindre de bien meilleures performances par rapport aux autres Framework.

INCONVÉNIENTS DE VUE.JS :

- Manque de ressources
- Il peut y avoir des problèmes d'intégration dans d'énormes projets

- Antécédents chinois. Dans la mesure où Vue.js a un peu d'historiques chinois, beaucoup d'éléments et de descriptions sont encore en chinois, néanmoins, de plus en plus de matériaux sont traduits en anglais.

d) Choix du Framework du front end web

Parmi ces Framework front end, tous présentent des avantages et des inconvénients. Cependant, ces avantages et inconvénients n'ont pas été assez discriminants pour permettre de choisir un Framework parmi tous les autres. Ainsi, pour choisir celui qui serait le plus intéressant à utiliser, nous avons encore une fois basé notre choix sur la popularité de l'outil. Ainsi, notre choix s'est porté sur Angular.

2. Partie Mobile

a) Applications natives, Web, Hybride

➤ Applications natives

Une application native est conçue spécialement pour les appareils mobiles et dans un langage de programmation spécifique à un système d'exploitation (Swift pour iOS, par exemple). Cela signifie, en reprenant l'exemple précédent, que si l'on souhaite faire fonctionner son application iOS sur Android, il faudra entièrement la reprogrammer en Java ou Kotlin. Les applications natives sont donc très coûteuses, mais sont les plus performantes. De plus, elles ont directement accès aux fonctionnalités relatives aux différents composants de l'appareil tels que le GPS ou l'appareil photo, sont utilisables hors-ligne et sont directement téléchargeables depuis les plateformes de téléchargement d'applications type Google Play ou Appstore.

➤ Applications Web

Une application web est une application mobile exécutable via le navigateur internet de votre appareil mobile, comme un site internet classique. Elle a plusieurs avantages :

- Entièrement en HTML, CSS et JavaScript, donc un apprentissage du code bien moins long et coûteux que pour les applications natives.
- Un seul code à écrire, pas besoin de se soucier des compatibilités avec Android ou iOS car elles seront compatibles avec tous les navigateurs modernes.
- D'autres avantages propres au web comme la facilité d'insertion de nombreux types de médias, la gestion de l'affichage sur différents types d'écrans, etc.

L'application web n'offrira pas toutes les fonctionnalités dont sont dotées les applications natives, comme par exemple l'interaction avec d'autres applications ou le contrôle direct d'un composant de l'appareil, et ne sera pas aussi performante. Cependant, JavaScript reste un langage très puissant, et couplé à l'un de ses nombreux Framework comme **Angular.js**, il est tout à fait possible de créer d'excellentes applications qui seront puissantes, multiplateformes, et surtout faciles et rapides à développer, donc peu coûteuses.

➤ **Applications Hybride**

Une application hybride est un mélange des deux premiers types d'applications que nous venons de voir. Ce sont des applications web qui disposent de fonctionnalités propres à des applications natives, comme le GPS par exemple. Elles peuvent constituer une alternative intéressante aux applications natives car moins coûteuses et téléchargeables depuis leur plateforme de téléchargement d'applications respective, mais elles restent moins performantes que les natives et d'une ergonomie pas toujours optimisée.

b) Les langages de développement mobile

➤ **TypeScript**

Il s'agit d'un langage open-source développé par Microsoft qui vise à renforcer la sécurité et l'efficacité de JavaScript. TypeScript est « Trans compilé » en JavaScript, ce qui veut dire qu'il peut être interprété par n'importe quel navigateur web ou moteur JavaScript. Couplez ce langage à des Framework comme Angular.js ou NativeScript pour créer de puissantes application web ou hybrides.

➤ **Swift**

C'est le langage à choisir pour la programmation d'applications natives iOS. Swift a le vent en poupe et est en train de prendre petit à petit le dessus sur Objective-C, le langage de programmation principal pour iOS, bien qu'ils soient encore souvent utilisés ensemble. Swift a de nombreux avantages : il clarifie la syntaxe d'Objective-C, facilite la détection d'erreurs de programmation, etc. Si vous aimez la pomme et ne jurez que par elle, ces deux langages sont pour vous.

➤ **Java**

C'est le langage de programmation le plus utilisé de la planète, et le langage par défaut pour le développement d'applications natives Android. La force de Java est que, contrairement à des

langages plus « fermés » comme Swift, conçu pour iOS seulement, Java n'est pas uniquement destiné à Android, et peut être utilisé comme code de base pour le développement de programmes qui tourneront sur presque tous les appareils possibles et imaginables. De plus, la communauté de programmeurs Java est tellement immense qu'il est presque impossible de rester bloqué sur une ligne de code : votre problème a sûrement déjà été rencontré et résolu. Java est une valeur sûre, vous ne pouvez pas vous tromper en le choisissant.

c) Choix du front end Mobile

Nous avons choisi d'utiliser la technologie hybride grâce à ses avantages :

- ❖ C'est une technologie multiplateforme, ce qui est un gain de temps considérable si le projet n'est pas trop complexe et ne nécessite pas le développement de plugins spécifiques ;
- ❖ Les compétences web sont plus répandues que les compétences mobiles, les coûts de développement sont donc généralement moins importants ;
- ❖ Conçue pour être téléchargeable, l'application mobile hybride peut être monétisée et bénéficie des classements dans les stores.

Pour le choix du langage de programmation nous allons utiliser le TypeScript avec le Framework IONIC qui est un Framework pour créer des applications mobiles semi-hybride. Ces dernières peuvent fonctionner sous IOS, Android et Windows Phone. Ionic peut aussi créer des applications Chrome ou Electron JS. Son principal atout est d'écrire une seule fois le code pour les multiples plateformes.

Ce Framework s'utilise en HTML, CSS, JavaScript mais il offre aussi la possibilité d'utiliser le langage TypeScript. Le langage TypeScript est converti en JavaScript lors de la compilation de l'application. L'avantage du TypeScript est d'offrir un langage plus propre et plus structuré que le JavaScript, notamment pour la création des classes en orienté objet.

II. Choix du Framework Back End

Un Framework n'est pas indispensable pour la création de notre application web. Cependant, pour que l'application soit robuste, facile à faire évoluer et réalisable en un temps minimum, un Framework représente un outil idéal.

Il existe une grande quantité de Framework. Chacun présentant des avantages et des inconvénients, il a fallu réduire le choix de ces Framework pour permettre ainsi de ne pas perdre trop de temps sur le comparatif et passer plus rapidement à la phase de modélisation.

1. Django Rest Framework

Django Rest Framework est une extension à Django pour développer rapidement des API REST robustes au goût du jour. Reprenant la philosophie Django, la prise en main est rapide et efficace.

La structure Django REST est une boîte à outils puissante et flexible pour la construction d'API Web.

Voici quelques raisons pour lesquelles vous voudrez peut-être utiliser la structure REST:

- L' API navigable sur le Web est un gain énorme en termes de convivialité pour vos développeurs.
- Stratégies d'authentification, y compris les packages pour OAuth1a et OAuth2 .
- Sérialisation prenant en charge les sources de données ORM et non ORM .
- Personnalisable tout le chemin vers le bas - il suffit d'utiliser des vues sur la base de la fonction régulières si vous n'avez pas besoin des plus puissantes fonctionnalités .
- Une documentation complète et un excellent soutien de la communauté .
- Utilisé et approuvé par des entreprises de renommée internationale, notamment Mozilla , Red Hat , Heroku et Eventbrite .

2. Symfony

Symfony est un Framework PHP qui a été lancé en 2005. Il est aujourd'hui stable et reconnu. Il est également orienté objet, respecte le modèle MVC et est développé sous licence MIT.

C'est un Framework très utilisé et reconnu internationalement. Il a été développé par la société SensioLabs qui l'utilise et le maintient régulièrement. Il est considéré comme un ensemble d'outils rassemblant des composants préfabriqués, rapides et faciles à utiliser.

Un des avantages de Symfony est de proposer une évolutivité et une maintenance efficace en permettant à d'autres développeurs de prendre en main rapidement le projet sans avoir participé à son élaboration. Il existe également un nombre important de ressources sur le web pour rendre

la maintenance encore plus facile. Enfin, il est très flexible car il permet de n'utiliser que certains de ces modules sans forcément avoir à utiliser tout le Framework. Laravel possède beaucoup de composants issus du Symfony.

3. Spring Boot

Spring Boot fournit aux développeurs Java une bonne plate-forme pour développer une application Spring autonome et de niveau production que vous pouvez **simplement exécuter**. Vous pouvez commencer à utiliser des configurations minimales sans avoir besoin d'une configuration complète de la configuration Spring.

Avantages :

Spring Boot offre à ses développeurs les avantages suivants:

- Facile à comprendre et à développer des applications de printemps
- Augmente la productivité
- Réduit le temps de développement

Buts :

Spring Boot est conçu avec les objectifs suivants -

- Pour éviter une configuration XML complexe au printemps
- Développer plus facilement des applications Spring prêtes pour la production
- Pour réduire le temps de développement et exécuter l'application indépendamment
- Offrir un moyen plus simple de démarrer avec l'application

4. Choix du Framework de Back-end

Parmi ces Framework, tous présentent des avantages et des inconvénients. Cependant, notre choix c'est porté sur le Framework spring boot grâce à ses nombreux avantages :

- Il fournit un moyen flexible de configurer Java Beans, des configurations XML et des transactions de base de données.
- Il fournit un traitement par lots puissant et gère les points de terminaison REST.
- Dans Spring Boot, tout est configuré automatiquement. Aucune configuration manuelle n'est nécessaire.

- Il offre une application de ressort basée sur des annotations
- Facilite la gestion des dépendances
- Il comprend un conteneur de servlets intégré

III. Choix du système de gestion de base de données

Une fois le langage et le Framework choisis, la question de la base de données à utiliser a elle aussi été très importante. Toujours dans l'optique d'une optimisation de l'outil, il faut choisir le système de gestion de bases de données le plus efficace possible. Son adéquation avec les besoins du programme impacte directement le temps de développement et la stabilité du système.

1. Oracle

Oracle Database est un système de gestion de base de données relationnelle objet. Il est développé par Oracle Corporation a été distribué pour la première fois en 1980. Il est implémenté en C et C++ et est disponible sur la plupart des systèmes d'exploitation (Linux, Solaris, OSX, Windows). Il est également compatible avec une grande variété de langage de programmation, dont PHP. Il respecte le principe des transactions ACID.

2. MySQL

MySQL est un Système de Gestion de Base de Données (SGBD) parmi les plus populaires au monde. Il est distribué sous double licence, une licence publique générale GNU et une propriétaire selon l'utilisation qui en est faite. La première version de MySQL est apparue en 1995 et l'outil est régulièrement entretenu.

Ce système est particulièrement connu des développeurs pour faire partie des célèbres quatuors: WAMP (Windows, Apache, MySQL et PHP), LAMP (Linux) et MAMP (Mac). Ces packages sont si populaires et simples à mettre en œuvre que MySQL est largement connu et exploité comme système de gestion de base de données pour des applications utilisant PHP. C'est d'ailleurs pour cette raison que la plupart des hébergeurs web proposent PHP et MySQL.

3. PostgreSQL

PostgreSQL est un Système de Gestion de Base de Données (SGBD) libre disponible sous licence BSD. Ce système multiplateforme est largement connu et réputé à travers le monde,

notamment pour son comportement stable et pour être très respectueux des normes ANSI SQL. Ce projet libre n'est pas géré par une entreprise mais par une communauté de développeurs.

4. Choix du SGBD :

De par sa très grande popularité et ses nombreuses, le SGBD qui a retenu notre attention : c'est MySQL. Il est un serveur de base de données relationnelles SQL qui fonctionne sur de nombreux systèmes d'exploitation (dont Linux, Mac OS X, Windows, Solaris, FreeBSD...) et qui est accessible en écriture par de nombreux langages de programmation, incluant notamment PHP, Java, Ruby, C, C++, .NET, Python ...

L'une des spécificités de MySQL c'est qu'il inclut plusieurs moteurs de bases de données et qu'il est par ailleurs possibles au sein d'une même base de définir un moteur différent pour les tables qui composent la base. Cette technique est astucieuse et permet de mieux optimiser les performances d'une application. Les 2 moteurs les plus connus étant MyISAM (moteur par défaut) et InnoDB.

La réplication est possible avec MySQL et permet ainsi de répartir la charge sur plusieurs machines, d'optimiser les performances ou d'effectuer facilement des sauvegardes des données

IV. Présentation des outils de développement

1. STARUML

Robuste, complet et gratuit StarUML est un outil spécialisé dans la modélisation UML pratique dans le domaine du développement d'applications. Un logiciel complet pour les utilisateurs aguerris dans le développement d'applications.

Utilisé dans le développement logiciel et dans la conception orientée objet, la modélisation UML est un langage reposant sur une représentation en diagrammes et pictogrammes. Elle permet une visualisation et une représentation d'une architecture d'un projet en montrant les acteurs, processus et composants.

StarUML est un logiciel de modélisation UML, cédé comme open source par son éditeur, à la fin de son exploitation commerciale, sous une licence modifiée de GNU GPL.

StarUML gère la plupart des diagrammes spécifiés dans la norme UML 2.0.

StarUML est écrit en Delphi, et dépend de composants Delphi propriétaires (non open-source).

StarUML 2.5.1 est la version que nous avons utilisée.

2. Visual studio code

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et MacOS.

Visual Studio Code est présenté lors de la conférence des développeurs Build d'avril 2015 comme un éditeur de code cross-Platform, open source et gratuit, supportant une dizaine de langages.

Le code source est fourni sous la licence libre MIT (plus précisément la licence Expat) sur le site du projet sur Github. En revanche, l'exécutable est proposé sur le site officiel de Microsoft sous une licence privative.

VSC propose différents éléments qui peuvent être intéressants pour des développeurs à tous niveaux, si bien que, comparé à d'autres éditeurs de texte, le niveau visé est plutôt intermédiaire/avancé. Néanmoins, VSC peut représenter un bon choix de départ même pour un débutant dans la perspective d'atteindre ensuite une certaine expertise. De plus, les fonctionnalités d'édition avancées de VSC peuvent être également exploitées dans d'autres domaines tels que le formatage/nettoyage de fichiers textuels ou données brutes.

Parmi les caractéristiques principales du logiciel figurent :

- ❖ IntelliSense : une technologie avancée qui propose, outre à la mise en évidence de la syntaxe et la complétion automatique du code, un système d'inférence articulé et basé directement sur la logique du code source ;
- ❖ Intégration native avec Git : le logiciel implémente le système de gestion de versions Git directement dans l'interface de l'éditeur, ce qui représente un avantage pour pouvoir effectuer les opérations de versioning plus aisément ;
- ❖ Ligne de commande intégrée : toujours dans l'interface de l'éditeur, il est possible de lancer la ligne de commande et exécuter tous les commandes disponibles sur le système d'exploitation;
- ❖ Ecosystème d'extensions : les extensions sont au cœur du projet et il existe même un système assez simple pour développer/publier ses propres extensions ;

- ❖ **Debugging intégré** : pour les développeurs plus avancés, il existe également des fonctionnalités de débogging directement à l'intérieur de l'éditeur.

3. Eclipse

Eclipse est une plate-forme de développement Java gratuite, connue pour ses plug-ins permettant aux développeurs de développer et de tester du code écrit dans d'autres langages de programmation. Eclipse est commercialisé sous les termes de la licence publique Eclipse.

Eclipse a vu le jour en 2001, lorsque IBM a fait don de trois millions de lignes de code de ses outils Java pour développer un environnement de développement intégré (IDE) open source . L'IDE a été initialement supervisé par un consortium de fournisseurs de logiciels cherchant à créer et à promouvoir une nouvelle communauté qui viendrait compléter Apache. la communauté open source de. Selon des rumeurs, le nom de la plate-forme découlerait d'un objectif secondaire, qui était d'éclipser le populaire IDE de Microsoft, Visual Studio .

4. Xamp

XAMPP est un ensemble de logiciels servant à mettre en place aisément un serveur Web, un serveur FTP et un serveur de messagerie électronique. C'est une distribution de logiciels libres (X Apache MySQL Perl PHP) offrant une bonne souplesse d'utilisation, reconnue pour son installation simple et rapide. Ainsi, il est à la portée de la plupart de personnes dans la mesure où il ne requiert pas de connaissances spécifiques et fonctionne, qui plus est, sur les dispositifs d'exploitation les plus communs.

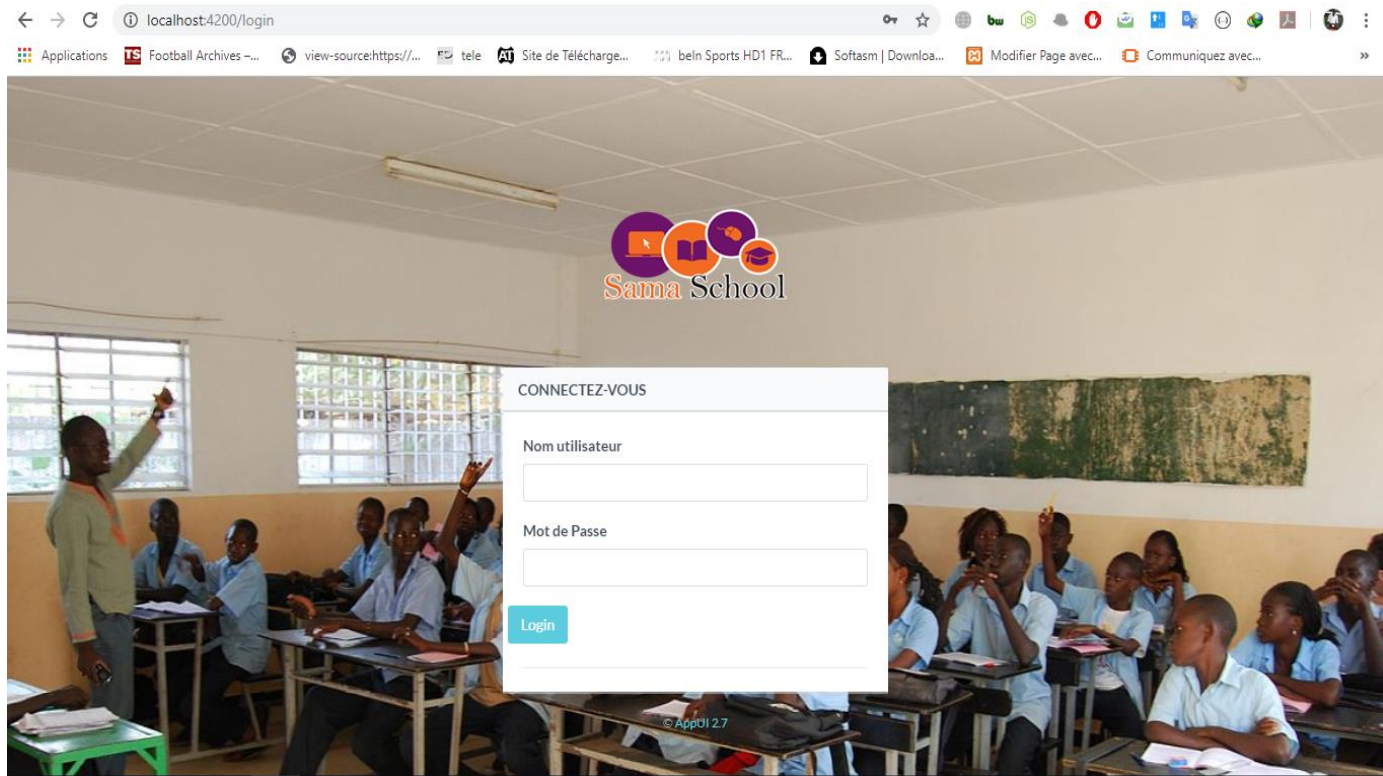
Il est distribué avec différentes bibliothèques logicielles qui élargissent la palette des services de façon notable : OpenSSL, Expat (parseur XML), PNG, SQLite, zlib, ... mais aussi différents modules Perl et Tomcat. Bon nombre de personnes critiquent la quantité d'extensions ajoutées qui sont pour la majorité inutiles aux débutants. Une version lite a donc été mise en place

V. Capture d'écran

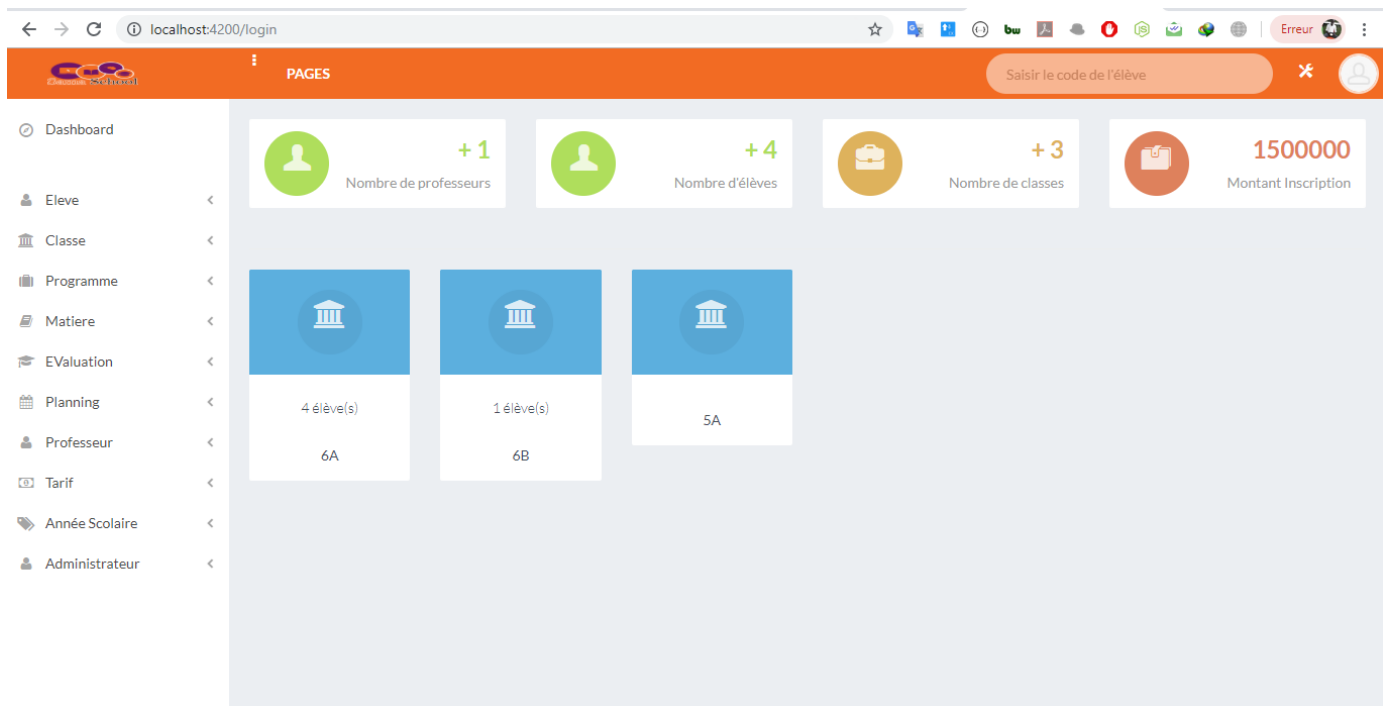
1. Partie Web

La plateforme web est destinée à l'administration. Pour y accéder il faut au préalable disposer d'un login et d'un mot de passe.

CONCEPTION ET REALISATION D'UNE SOLUTION WEB ET MOBILE DE GESTION D'ECOLE (MOYEN ET SECONDAIRE) « SAMASCHOOL »



Ecran 1 : Page de Connexion



Ecran 2 : Page d'accueil

CONCEPTION ET REALISATION D'UNE SOLUTION WEB ET MOBILE DE GESTION D'ECOLE (MOYEN ET SECONDAIRE) « SAMASCHOOL »

Enregistrement Eleve

Dashboard

Eleve

Enregistrer eleve

Classe

Programme

Matiere

Evaluation

Planning

Professeur

Tarif

Année Scolaire

Administrateur

FORMULAIRE D'ENREGISTREMENT D'UNE ELEVE

Nom eleve

Prenom eleve

Adresse eleve

Telephone eleve

Date de Naissance

Lieu de Naissance

Choisir Une Image

Nom Tuteur

Prenom Tuteur

Adresse Tuteur

Telephone tuteur

Enregistrer

Ecran 3 : Formulaire d'inscription élève

Liste Classe

Dashboard

Eleve

Classe

Enregistrer Classe

Liste Classe

Programme

Matiere

Evaluation

Planning

Professeur

Tarif

Année Scolaire

Administrateur

LISTE DES CLASSE

ID	nom Classe	Niveau	Année Scolaire		Action
209	5A	Moyen	2018-2019	Affecter Programme voir eleve voir planning	voir modifier
119	6B	Moyen	2018-2019	Affecter Programme voir eleve voir planning	voir modifier
13	6A	Moyen	2018-2019	Affecter Programme voir eleve voir planning	voir modifier

Ecran 4 : Tableau liste des classes

CONCEPTION ET REALISATION D'UNE SOLUTION WEB ET MOBILE DE GESTION D'ECOLE (MOYEN ET SECONDAIRE) « SAMASCHOOL »

localhost:4200/liste-eleve/13

PAGES

Saisir le code de l'élève

Dashboard

Liste eleve

Classe / Blank

LISTE DES ELEVES

Imprimer

classe : 6A

code	Prenom	Nom	Telephone	
100002	Cheikh Tidjane	Diop	774874628	Affecter Note note Ajouter absence Generer bulletin Infos eleves
100003	Soda	Lo	763025237	Affecter Note note Ajouter absence Generer bulletin Infos eleves
100004	Ndack	Ndiaye	774874525	Affecter Note note Ajouter absence Generer bulletin Infos eleves
100001	fatou	ndiaye	774874628	Affecter Note note Ajouter absence Generer bulletin Infos eleves

Ecran 5 : Liste élève d'une classe

localhost:4200/liste-note/13/232

PAGES

Saisir le code de l'élève

Dashboard

Eleve

Classe

Enregistrer Classe

Liste Classe

Programme

Matiere

Evaluation

Planning

Professeur

Tarif

Année Scolaire

Administrateur

Ndiaye Ndack
née 02/03/1999 à Keur Massar

Imprimer

Matiere	Prenom professeur	Nom professeur	Note Obtenue	semestre
SVT	Ibra	Ndiaye	18	semestre1
Science Physique	Ibra	Ndiaye	10	semestre2
Philosophie	Ibra	Ndiaye	13	semestre2
Education Physique	Ibra	Ndiaye	15	semestre1
Education Physique	Ibra	Ndiaye	18	semestre1
Mathématique	Ibra	Ndiaye	15	semestre1
Mathématique	Ibra	Ndiaye	13.75	semestre1
HG	Ibra	Ndiaye	18	semestre1

Ecran 6 : Relevé de note d'une élève

CONCEPTION ET REALISATION D'UNE SOLUTION WEB ET MOBILE DE GESTION D'ECOLE (MOYEN ET SECONDAIRE) « SAMASCHOOL »

The screenshot shows the 'Eleve' profile page in the SAMASCHOOL web application. The page has a sidebar with navigation links: Dashboard, Eleve, Classe, Programme, Matière, Evaluation, Planning, Professeur, Tarif, Année Scolaire, and Administrateur. The main content area displays the student's profile with a photo, name, code, and birth date. It also includes tabs for 'Information Eleve', 'Information Tuteur', and 'Information Scolaire'.

Eleve

Code: 100005
née 02/03/2003 à Keur Massar

Modifier Inscrire

Information Eleve Information Tuteur Information Scolaire

Diop
Amina
Keur Massar
774874525

Ecran 7 : Fiche élève

The screenshot shows the 'Bulletin élève' page in the SAMASCHOOL web application. The page has a sidebar with navigation links: Dashboard, Eleve, Classe, Programme, Matière, Evaluation, Planning, Professeur, Tarif, Année Scolaire, and Administrateur. The main content area displays the student's profile with a photo, name, and birth date. It also includes a table of grades for various subjects and a summary table at the bottom.

Bulletin élève

Générer bulletin


ndiaye fatou
née 2/2/1994 à Dakar
classe: 6A
semestre1

Cours privée Excellence Ainoumady
Tel : 33 878 49 21
Adresse: Keur Massar
Email: cpecc@cppeccan
BP : 15030

Matiere	Notes Devoir	Note Composition	Coefficient	nombre de points
Education Physique	10	15	1	12.5
ESPAGNOL	13	10	3	34.5
HG	10.67	15	2	25.67
Mathématique	11	12	2	23
SVT	14.33	16	2	30.33

Total Général	Total Coef	Moyenne / 20	Rang
126	10	12.6	3 ieme

Ecran 8 : Bulletin élève page web



ndiaye fatou
née 2/2/1994 à Dakar
classe: 6A
semestre 1

Cours privée Excellence Ainourady
Tel : 33 878 49 21
Adresse: Keur Messar
Email: cpao@cpao.sn
BP : 15030


Matiere	Notes Devoir	Note Composition	Coefficient	nombre de points	
Education Physique	10	15	1	12.5	
ESPAGNOL	13	10	3	34.5	
HG	10.67	15	2	25.67	
Mathématique	11	12	2	23	
SVT	14.33	16	2	30.33	

Total Général	Total Coef	Moyenne / 20	Rang
126	10	12.6	3 ieme

Ecran 9 : Bulletin version PDF

2. Partie Mobile

a. Professeur



Nom utilisateur


Mot de Passe


Vous êtes ▼


Login

Design by Digi221

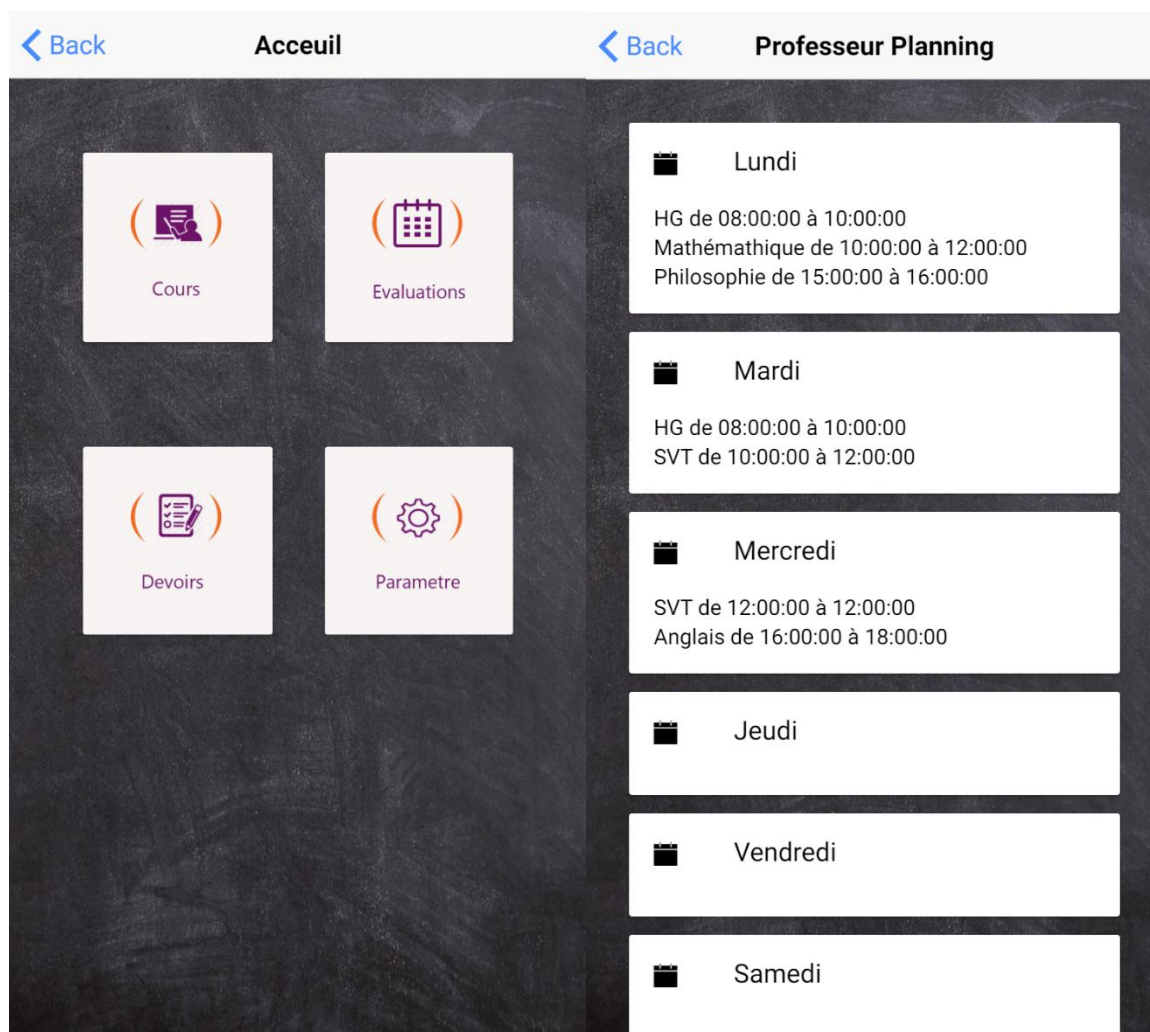
professeur-menu

 6A
Moyen

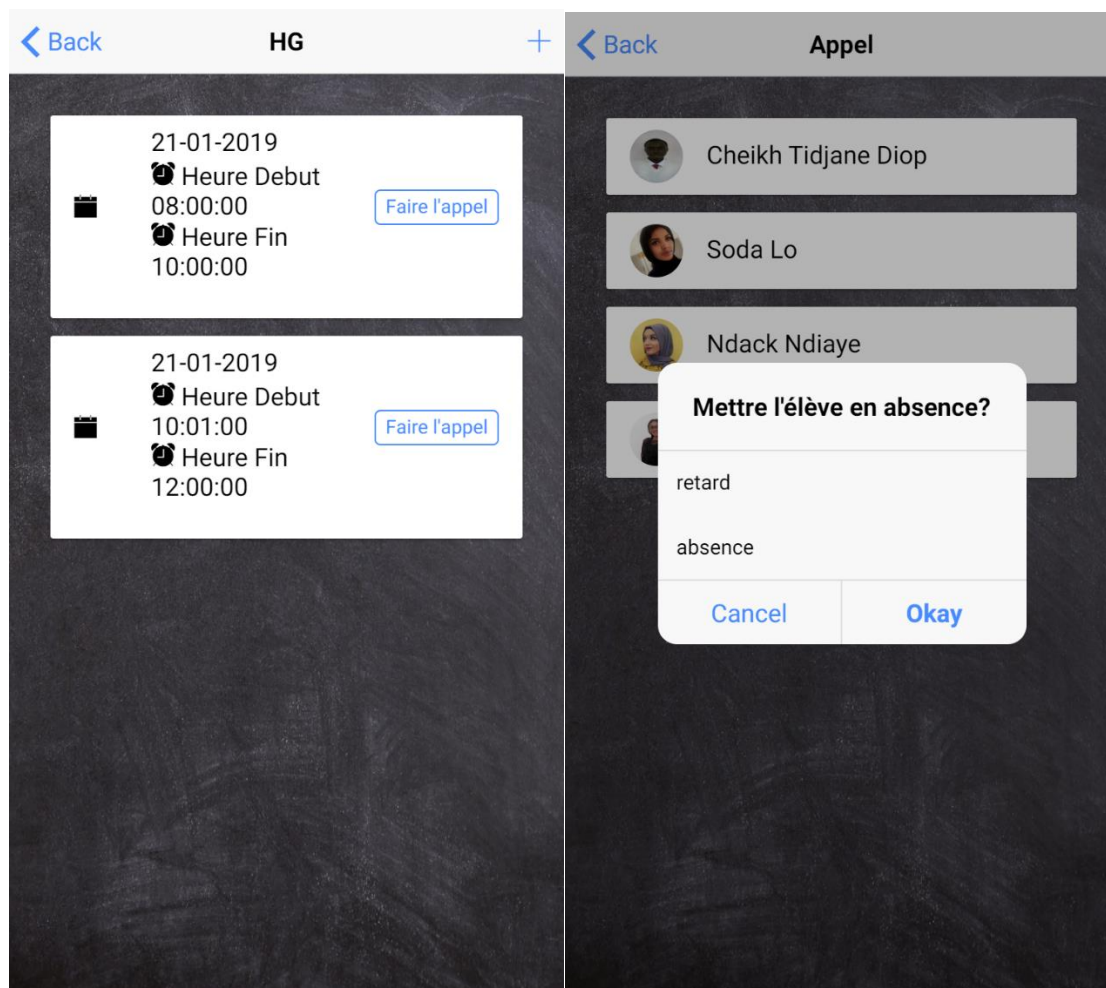
 6B
Moyen

 5A
Moyen

Ecran 10 : Login et choix de classe



Ecran 11 : Menu principal et planning cours professeur



Ecran 12 : Séance de cours et appel

Back

Evaluation

Ajouter une évaluation

Date Debut

Semestre

Libelle

Type Evaluation

Matiere

Ajouter

Back

professeur-list-evaluati...

ESPAGNOL

25/05/2019

semestre2

Science Physique

16/02/2019

semestre2

SVT

31/01/2019

semestre1

Philosophie

18/03/2019

semestre2

Education Physique

21/12/2018

semestre1

Ecran 13 : formulaire évaluation et liste des évaluations

[Back](#) professeur-noter

Noter

Veuillez saisir la note de Diop Cheikh Tidjane

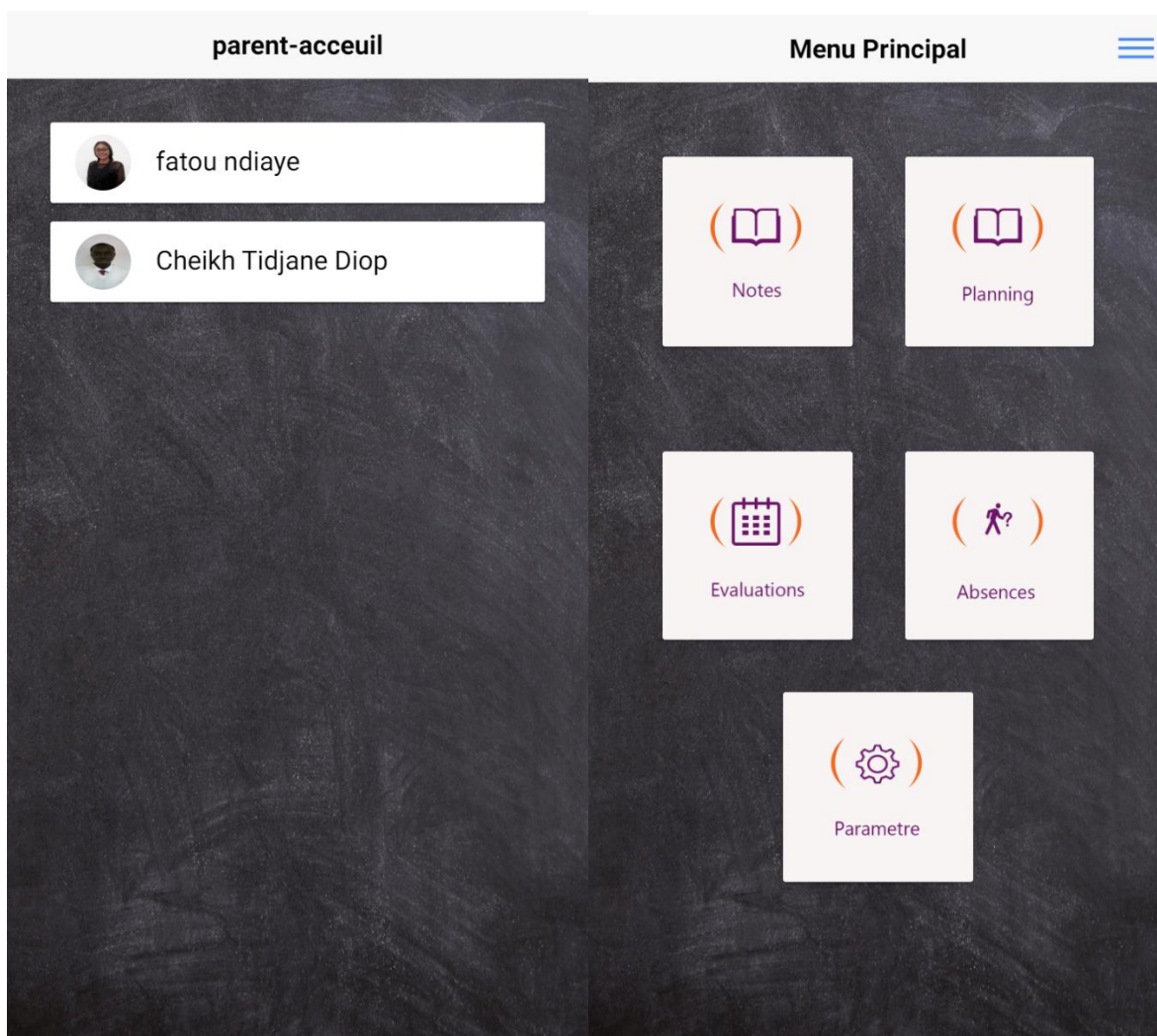
[Annuler](#)[Enregister](#)

/20

fatou ndiaye
15/20

Ecran 14 : noter une élève

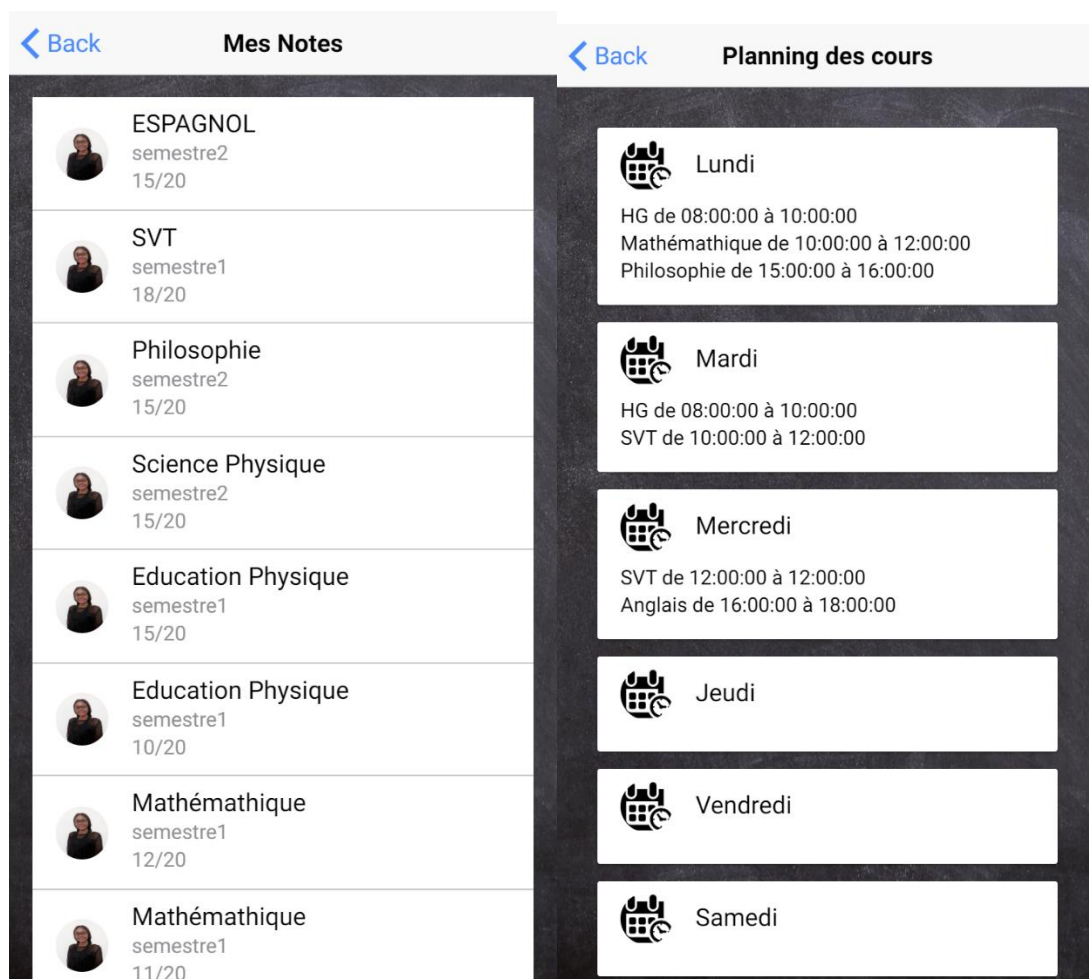
b. Parent et élève



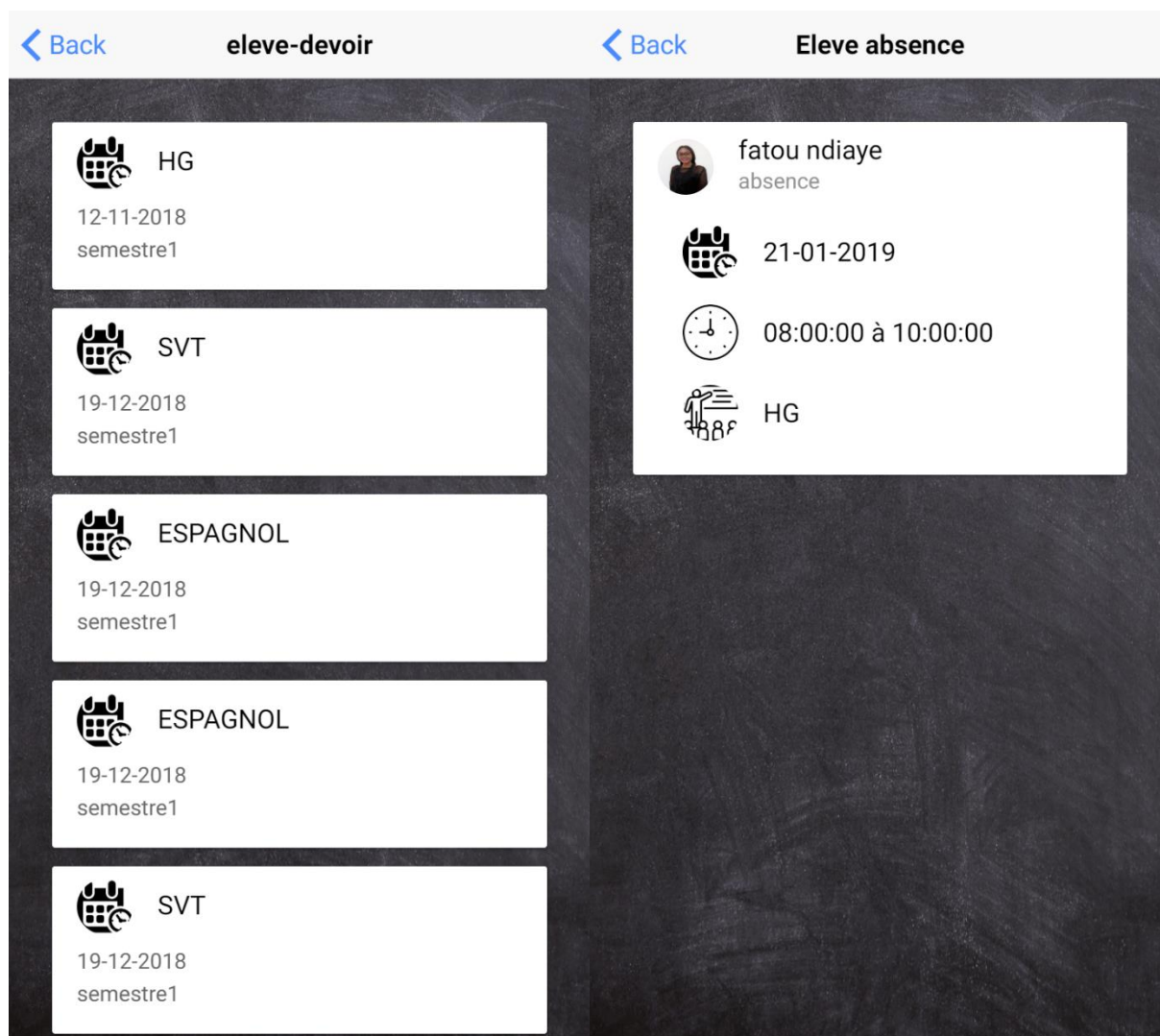
Ecran 15 : écran d'accueil parent

Ecran 16 : Menu parent et élevé

Si l'utilisateur est un parent l'écran 15 s'affiche en premier pour qu'ils puissent afficher un de ses élèves dont il est tuteur. Sinon s'il est une élève la deuxième l'écran 16 s'affiche comme écran d'accueil.



Ecran 17 : notes et planning de l'élève



Ecran 18 : planning devoir et absence

CONCLUSION

L'objectif de ce présent mémoire a été de créer une application mobile et Web innovante intitulé 'SAMASCHOOL' aidant les parents d'élève et l'école d'avoir un meilleur suivi de leurs élèves. Nous pouvons affirmer avec certitude que l'objectif a été atteint.

La mise en place de cette application est un point important qui contribue l'amélioration du suivi des élèves dans les écoles. Cette application facilite l'accès parents d'élèves à des informations de leurs élèves pour un meilleur suivi. Il permettra aussi à l'administration d'avoir une vision plus globale de leurs élèves.

Le travail que nous venons d'effectuer dans le cadre de l'obtention d'ingénieur en système d'information et génie logiciel, nous a permis d'approfondir nos connaissances acquises en programmation et en conception. Mais aussi d'autres aptitudes comme l'autonomie, la gestion de projet et du temps.

Résumé

Dans le cadre de mon Projet de Fin d'Etudes, j'ai choisi de mettre en place une application de gestion d'école, qui permettra de gérer l'ensemble de ses élèves, professeurs et parents. À travers une interface web et mobile simple et pratique.

Pour ce projet on a opté comme démarche, les étapes suivantes :

Recenser les besoins fonctionnels et non fonctionnels du projet.

L'étude technique et la conception détaillée de l'application.

Réalisation.

Pour bien mener le développement de ce projet, la méthodologie agile qui part du principe que spécifier et planifier dans les détails l'intégralité d'un produit avant de le développer, semble plus adéquate à notre contexte, et plus précisément la méthode **SCRUM**, avec Le langage de modélisation **UML**.

Pour ce qui concerne le volet technique, il y avait eu recours aux technologies comme :

- Spring Boot et Spring Security pour le Back-end
- Angular pour le front-end Web et ionic pour le front-end mobile

Ma mission consiste à développer un ensemble de modules de gestion d'école, tels que la gestion des : élèves, professeurs, parents... En utilisant les différentes techniques et outils de développement mentionné auparavant.

Abstract

As part of my final project, I chose to implement a school management application, which will manage all of its students, teachers and parents. Through a simple and practical web and mobile interface.

For this project we chose the following steps:

- Identify the functional and non-functional needs of the project.
- Technical study and detailed design of the application.
- Production.

To successfully lead the development of this project, the agile methodology that assumes that specifying and planning in detail the entirety of a product before developing it, seems more appropriate to our context, and more specifically the SCRUM method, with The UML modeling language.

On the technical side, technologies such as:

- **Spring Boot** and **Spring Security** for the Back-end
- **Angular** for the Web front-end and **ionic** for the mobile front-end

My mission is to develop a set of school management modules, such as the management of: students, teachers, parents. Using the different techniques and development tools mentioned before.

REFERENCES

Bibliographie

Mémoire de fin de fin d'étude de Mouhamed El Moustapha KONTE pour l'obtention de maitrise en système d'information et génie logiciel en 2018 « **Etude et mise en place d'une plateforme Web de souscription d'assurance Automobile : cas SAHAM Assurance** »

Mémoire de fin de fin d'étude de Ibra Ndiaye, Mamadou Lamine Sall, et Cheikh tidiane Diop pour l'obtention de la licence en informatique en 2016 « **conception et réalisation d'une plateforme web et mobile de géolocalisation des ouvriers du Sénégal 'moammo'** »

Webographie

- ✓ <https://www.supinfo.com/articles/single/574-architecture-2-tiers-vs->
Consulté le 06-03-2019
- ✓ <http://www-igm.univ-mlv.fr/~dr/XPOSE2001/perrot/Intro-Comparatif.htm>
Consulté le 06-03-2019
- ✓ <https://www.ambient-it.net/reactjs-vs-angular-vs-vuejs-que-choisir-en-2018/>
Consulté le 06-03-2019
- ✓ <https://mastercaweb.u-strasbg.fr/developpement-mobile-bons-outils-pour-debuter/>
Consulté le 06-03-2019
- ✓ <https://www.mobizel.com/developpement-dune-application-mobile-hybride-33/>
Consulté le /06-03-2018
- ✓ <https://www.supinfo.com/articles/single/5778-presentation-ionic>
Consulté le
- ✓ <https://makina-corpus.com/blog/metier/2015/django-rest-framework-les-serializer-et-les-exceptions-partie-1>
Consulte le 07-03-2019
- ✓ <https://sql.sh/sqbd/mysql>
Consulte le 07-03-2019
- ✓ <https://sql.sh/sqbd/postgresql>
Consulté le 07-03-2019

- ✓ <https://www.supinfo.com/articles/single/5174-installer-serveur-xampp-son-pc>

Consulté le 07-03-2019

- ✓ <https://searchmicroservices.techtarget.com/definition/Eclipse>

consulté le 07-03-2019

OUTILS UTILISES

- ✓ StarUml: <http://staruml.io/download/release/v2.5.1/StarUML-v2.5.1.msi>
- ✓ Xamp Server : <https://www.apachefriends.org/fr/download.html>
- ✓ Visual Studio Code : https://code.visualstudio.com/?wt.mc_id=vscom_downloads:
- ✓ Eclipse: <https://spring.io/tools3/sts/all>

TABLE DES MATIERES

Table des matières

Dédicaces	I
Remerciements	II
Sigles et abréviations	III
Table des illustrations	IV
Sommaire	V
Introduction Générale.....	1
Partie 1 : PRESENTATION ET CADRE METHODOLOGIQUE.....	2
Chapitre 1 : Présentation Général	3
I. Présentation du projet	3
II. Problématique	3
III. Objectif du projet	3
Chapitre 2 : Cadre Méthodologique	5
I. Présentation des différentes méthodes	5
1. Traditionnelle	5
2. Pert	
3. Cascade	5
4. Chemin critique.....	6
5. Agile.....	6
II. Choix de la méthode	7
III. Présentation de la méthode choisit	8
Partie 2 : Analyse et Conception.....	10
Chapitre 3 : Analyse.....	11
I. Méthode D'analyse et Choix de la méthode.....	11
1. Merise	11
2. UML	14
3. Choix de la méthode.....	15
II. Spécification des besoins du Système	16
1. Analyse des besoins	16
2. Analyse des rôles et des acteurs	16
3. Spécification des besoins fonctionnelles	17
4. Spécification des besoins non fonctionnelles.....	17
III. Règle de gestion	18

Chapitre 4 : Conception	19
I. Diagramme de Use Case	19
II. Diagramme Séquence	27
III. Diagramme de classe	32
IV. Diagramme d'objet	33
V. Diagramme de déploiement	34
Partie 3 : MISE EN ŒUVRE DE LA SOLUTION (REALISATION)	35
Chapitre 5 : Architecture de la solution	36
I. Architecture 1 tiers	36
II. Architecture 2 tiers	36
III. Architecture 3 tiers	37
IV. Architecture N tiers	38
V. Choix de l'architecture	39
Chapitre 6 : Présentation de la solution	43
I. Choix du Framework Front End	42
1. Partie Web	42
a. REACT	42
b. Angular	42
c. Vue Js	43
d. Choix du Framework front end web.....	44
2. Partie Mobile	44
a. Application native, web, hybride.....	44
b. Les langages de développement mobile	45
c. Choix du front end mobile	46
II. Choix du Framework Back End	46
1. Django Rest Framework	47
2. Symfony.....	47
3. Spring Boot	48
4. Choix du Framework Back-End.....	48
III. Choix du serveur de base de données	49
1. Oracle	49
2. MySQL	49
3. PostgreSQL	49
4. Choix du SGBD	50
IV. Présentation des outils développement.....	50
1. STARUML	50
2. Visual Studio Code	51
3. Eclipse.....	52
4. Xamp	52
V. Capture D'Ecran	52
1. Partie Web	52
2. Partie Mobile	57

a. Professeur.....	57
b. Parent et élève	63
Conclusion.....	66
Références	69
Outils utilisés.....	71
Table de Matière.....	72