

RAPPORT PROJET PROGRAMMATION ANDROID

APPLICATION DE DEPOT DE DOSSIER: CAS DE IAI



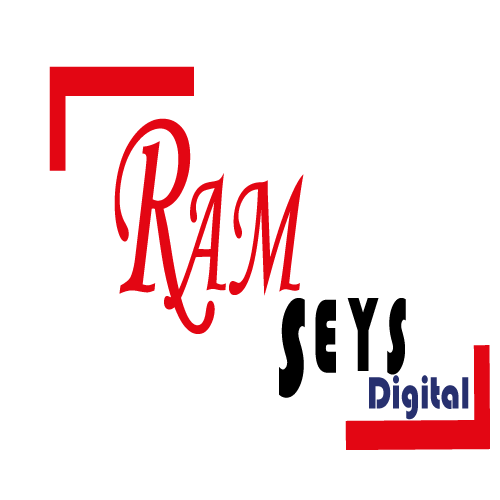
25 May 2023

IAI-CAMEROUN

Centre de Garoua

BAVITHY DIGITAL

<Tel:+227659845875>



Site : ramseys\_digital.com

BP :104 Garoua

INSTITUT AFRICAINE D ’INFORMATIQUE

AFRICAN INSTITU OF SCIENCE COMPUTER



Représentation du Cameroun

Tel : +23722211163/699990249

Site web : www.iaicameroun.cm

BP :137719 Yaoundé

RAPPORT

THEME :

CONCEPTION D’UNE APPLICATION DE DEPOT DE DOSSIER

**Stage effectué du 1erDécembre au 03 Janvier 2023 en vue de l’obtention du Diplôme de Technicien Supérieur(DTS) option Génie Logiciel.**

Rédigé par : Groupe 1 étudiants à **l’IAI-Cameroun**, Centre d’Excellence Technologique Paul Biya.

**Sous la supervision :**

Académique de :

M. Fomekong

Enseignant à IAI

Professionnel de :

M. Amadou wanie

Ingénieur Génie Logiciel

Année Académique 2022-2023

[Tableau 1:Tableau comparatif entre UML et MERISE 14](#_Toc123708334)

[Tableau 2:Différents types de relation dans un diagramme de cas d'utilisation 22](#_Toc123708335)

[Tableau 3: Listes des acteurs et leurs roles 23](#_Toc123708336)

[Tableau 4:Formalisme diagramme de séquence 31](#_Toc123708337)

[Tableau 5:Formalisme diagramme d’activité 36](#_Toc123708338)

[Tableau 6:Listes des classes 41](#_Toc123708339)

[Tableau 7:Outil logiciel utilisé 46](#_Toc123708340)

[Tableau 8:Outil materies 47](#_Toc123708341)

[Tableau 9:Formalisme diagramme de deploiement 50](#_Toc123708342)

[Figure 1: Logo d'UML (source https://fr.wikipedia.org/wiki/UML\_(informatique)) 15](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708353)

[Figure 2: Genèse d'UML( https://www.cybermedian.com/fr/a-comprehensive-guide-to-uml-class-diagram) 16](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708354)

[Figure 3Illustration du cycle de développement 2TUP (Source https://www.nassimbahri.ovh/docs/conception/Chapitre-3/) 18](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708355)

[Figure 4: Formalisme graphique d'un diagramme de cas d’utilisation 21](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708356)

[Figure 5:Formaliste d'un acteur 21](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708357)

[Figure 6:Formalisme d'un cas d'utilisation 22](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708358)

[Figure 7:Formalisme d'une association 22](#_Toc123708359)

[Figure 8: Diagramme de cas d'utilisation globale 24](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708360)

[Figure 9:Diagramme de cas d'utilisation gérer utilisateur 24](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708361)

[Figure 10: Diagramme de cas d’utilisation « Gérer matériels » 26](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708362)

[Figure 11:Diagramme de cas d’utilisation « Gérer Bureau » 28](#_Toc123708363)

[Figure 12:Diagramme de cas d’utilisation « Controle ticket» 30](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708364)

[Figure 13:Diagramme de cas d’utilisation « Gérer ticket » 30](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708365)

[Figure 14: Exemple de diagramme de séquence 32](#_Toc123708366)

[Figure 15: Diagramme de séquence "s'authentifier" 32](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708367)

[Figure 16:Diagramme de séquence "ajout" 33](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708368)

[Figure 17: Exemple diagramme d'activité 37](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708369)

[Figure 18:Diagramme d'activité "s'authentifier" 37](#_Toc123708370)

[Figure 19:Diagramme d'activité enregistrement 39](#_Toc123708371)

[Figure 20:Diagramme d'activité "modification employe" 40](#_Toc123708372)

[Figure 21:Diagramme de classe 42](#_Toc123708373)

[Figure 22: Logo java 48](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708374)

[Figure 23:Logo sql 48](file:///C:\\Users\\RAMSEYS\\Documents\\RAPPORT%20PROJET%20JAVA.docx" \l "_Toc123708375)

[Figure 24:Diagramme de déploiement du système 51](#_Toc123708376)

# RESUME

En vue de l’obtention du Diplôme de Technicien Supérieur en Informatique option Génie logiciel, l’Institut Africain d’Informatique préconise pour ses étudiants de deuxième année, un stage académique d’un mois au sein d’une entreprise. C’est ainsi que nous avons effectué notre stage à la direction informatique de l’entreprise RAMSEYS DIGITAL, du 01 Décembre au 03 Janvier 2023. Durant cette période, il nous a été attribué pour thème « **CONCEPTION D’UNE APPLICATION DE DEPOT DE DOSSIER**». Pour ce faire, nous avons identifié les besoins et les exigences des futurs utilisateurs, puis avons établis un cahier de charges qui met en exergue les besoins fonctionnels et non fonctionnels, l’estimation financière ainsi que la planification du projet.

La mise en place de la solution a nécessité l’utilisation du processus unifié 2TUP basé sur le langage de modélisation Unified Modeling Language et l’outil ASTAH Professional pour la réalisation des diagrammes. Les langages de programmation tel que le JAVA, Firebase couplés à l’architecture logiciel et l’architecture 3-tiers ont été utilisés pour le déploiement de la solution. De la réalisation de ce projet, en ressort une application accompagnée d’un cahier de charges, et de divers dossiers relatant de l’analyse, la conception et du guide utilisation.

# ABSTRACT

In order to obtain the Diploma of Higher Technician in Computer Science, software engineering option, the African Institute of Computer Science recommends for its secondyear students, a one month academic internship within a company. This is how we carried out our internship at the IT department of RAMSEYS DIGITAL, from December 01 to January 03,

2023. During this period, the theme was assigned to us "CONCEPTION OF APPLICATION FOR COMMON ENTRANCE DEPOSIT”. To do this, we identified the needs and requirements of future users, then established specifications that highlight the functional and non-functional needs, the financial estimate and the project planning. The implementation of the solution required the use of the unified process 2TUP based on the modeling language Unified Modeling Language and the ASTAH Professional tool for the realization of the diagrams. Programming languages such as JAVA, Firebase coupled with software architecture and 3-tier architecture were used for the deployment of the solution. From the realization of this project, an application comes out accompanied by specifications, and various files relating to the analysis, the conception and the user guide.

# INTRODUCTION GENERALE

La croissance rapide des technologies de l’information et de la communication ainsi que l’innovation dans les systèmes numériques sont à l’origine d’une révolution provoquée par le développement des techniques numériques, principalement l'informatique et l’Internet. Cette révolution, dite « révolution numérique » a incité les sociétés des deux secteurs publique et privé à s’informatiser et par conséquent d’établir des parcs informatiques.

Un parc informatique d’une entreprise est l’ensemble des ressources matérielles qui assurent le traitement informatique du système d’information, accumulées tout au long des années. En effet, la quantité de matériels à gérer, leur éclatement au sein de l’entreprise souvent très étendue dans l’espace, les exigences de performance et de réactivité font que la gestion de parc est devenue un processus global, complet et indispensable. La gestion du parc informatique recouvre non seulement la fonction d’inventaire de ces éléments mais aussi celles concernant son suivi et son évolution.

CAHIER DE CHARGE

Ière Partie

Contenu

INTRODUCTION

1. Contexte et justification
2. Expression des besoins

Conclusion

# Introduction

Le cahier de charge est ce document conçu par le maître d’œuvre et le maître d’ouvrage. Il devra présenter en détails le projet entre autres les objectifs visés, les besoins du client.

# Contexte et Justification

Nous nous situons ici dans le cadre du stage académique qui permet aux étudiants de mettre en pratique leurs formations académiques, de consolider leurs connaissances techniques et de s’initier au marché du travail. C’est ainsi que à notre arrivé à l’entreprise RAMSEYS, l’entreprise a exprimé le besoin d’implémenter une application de depot de dossier qui pourra dans le futur leur permettre d’avoir un inventaire en temps réel de leurs matériels informatiques.

# Expression des besoins

L’application à réaliser doit satisfaire les besoins fonctionnels qui constituent les fonctionnalités que l’application doit offrir et les besoins non fonctionnels qui perfectionnent l’application. Ce duo permettra d’obtenir une application complète, fiable avec une ergonomie conviviale.

## Les besoins fonctionnels

L’application à réaliser doit offrir un ensemble de fonctionnalités qui doivent être mises en relation avec un ensemble de besoins utilisateur. Ces derniers définissent les services auxquels les utilisateurs s’attendent à voir fournis par cette application.

La présente plateforme doit satisfaire les besoins fonctionnels suivants :

* Pour l’administrateur (Gestionnaire) :
* Authentification : S’identifier par un login et un mot de passe.
* Gerer un gestionnaire :
* Ajouter un Gestionnaire
* Modifier un Gestionnaire
* Lister gestionnaire
* Gérer Candidatures:
* Ajouter un Candidatures
* Modifier un Candidature
* Rechercher un Candidature
* Pour l’utilisateur sans privilège :
* Gerer Candidature :
* Ajouter une Candidature
* Rechercher une Candidature

## Besoins non fonctionnels

Un besoin non-fonctionnel est une contrainte liée à l’environnement, l’implémentation et les exigences en matière de performances.

Dans le cadre de ce travail, l'application devra répondre à ces besoins :

**✓ La sécurité** : Tous les accès des utilisateurs doivent être protégés par un login et un mot de passe ainsi qu’une gestion de privilèges et de niveaux d’accès pour les différents types de compte ;

**✓ La performance** : Pouvoir supporter des grandes quantités de données et le temps de réponse des requêtes doit être court ;

**✓ La fiabilité** : Les données fournies par l’application doivent êtres fiables et l’application doit rendre des résultats corrects.

**✓ Prise en compte de l'expérience de l'utilisateur** : Le système doit respecter le niveau d'expérience de l’utilisateur, donc il doit être simple, compréhensible et facile à utiliser.

# Conclusion

En résume, dans cette partie réservée au cahier de charges, nous avons précisé les objectifs et décrit les attentes de l’entreprise à l’égard de l’application à produire. Ainsi, cette partie sera prise en guise d’introduction au développement proprement parlé de l’application et conduira à une étape importante du projet, à savoir l’analyse.

Contenu

INTRODUCTION

1. Analyse de l’existant
2. Présentation et justification de la méthodologie de travail
3. Présentation des diagrammes

Conclusion

IIème Partie

DOSSIER D’ANALYSE

# Introduction

Le dossier d’analyse est ce document chargé de ressortir les différentes caractéristiques de l’existant tant son étude que ses critiques. Il émet et structure par un choix et une justification de méthode la/les propositions de solutions à apporter à celui-ci pour l’améliorer de part-les différents diagrammes qui le constitue.

# Analyse de l’existant

## Description de l’existant

Dans l’optique de comprendre comment est géré le depot des dossiers sur les sites de la société IAI Cammeroun, nous avons mené une enquête qui nous a conduit à constater que ces opérations sont gérées manuellement et dont le processus d’utilisation est le suivant :

* Le gestionnaire/secretaire recensent les dossiers de candidatures dans son bureau;
* Une fois les candidatures recuent, elle génére une liste de ces derniers;
* Affiche en suite la liste de ceux retenus.

## Critiques de l’existant

Suite à cette enquête, nous décelons plusieurs limites du système actuel qui sont entre autres :

* Le gestionnaire est souvent deborder avec les piles de dossiers;
* Le referencement est compliqué et souvent il y’a perte des dossiers;
* Retard suite à la complexité du travail.

## Proposition de solution

Notre solution proposée consiste à gérer les depots de dossier d’une manière informatisée avec une application qui permet à un candidat de s’incrire en ligne afin de reduire la charge du gestionnaire et qui permet aussi au gestionnaire de valider ou non la candidature, d’une manière fiable et efficace. En effet il aura une idée sur tout les candidatures et les informations qui sont joins.

D’où découle le thème de notre travail « Conception d’application de depot de dossier ».

# Présentation et justification des méthodologies de travail

## Méthode d’analyse

### Etude comparative

Les premières méthodes d’analyse ont vu le jour dans les années 1970. On compte de nos jours une diversité de méthodes parmi lesquelles la méthode Booch, Merise, OOSE, les méthodes basées sur le langage UML. Le tableau suivant présente une analyse comparative entre deux méthodes que sont Merise et UML.

Tableau 1:Tableau comparatif entre UML et MERISE

|  |  |  |
| --- | --- | --- |
| **Point de comparaison MERISE UML** | | |
| **Sigles** | Méthode d’étude et de réalisation informatique pour les systèmes d’entreprise | Unifie Modeling Language |
| **Définitions** | Méthode d’analyse et de conception de système d’information | Langage de représentation d’un système d’information |
| **Approche** | Considère le système réel selon deux points de vue : un point de vue statique (les données), un point de vue dynamique (les traitements). Il s’agit d’avoir une vision duale du système réel pour bénéficier de l’impression de relief qui en résulte, et donc consolider et valider le système final | L’approche objet associe les informations et les traitements. De cette façon, elle assure un certain niveau de cohérence |
| **Langue** | Français | International |
| **Niveaux d’abstraction** | Constitué de 03 niveaux correspondant à des degrés de stabilité et d’invariance de moins en moins élevés. Le niveau conceptuel, le niveau physique et logique | Propose différentes notions  (cas d’utilisation, paquetage, classe, composant, nœud) et différents diagrammes pour modéliser le système aux différents niveaux d’abstraction |
| **Orientation** | Plus adapté à une approche théorique | Plus orienté vers la conception |

Après cette étude comparative, nous décidons d’adopter UML comme langage de modélisation puisque nous allons utiliser le concept de l'orienté objet pour développer notre application mobile de gestion de parc informatique GPI\_GO.

### Présentation de UML



Figure 1: Logo d'UML (source https://fr.wikipedia.org/wiki/UML\_(informatique))

Pour développer une application, il ne convient pas de mettre l’accent uniquement sur l’écriture du code. Il faut d’abord organiser ses idées, les documenter, puis organiser la réalisation en définissant les modules et les étapes de la réalisation. Cette modélisation nécessite l’utilisation d’un langage permettant la description du système logiciel ainsi que sa compréhension par ses futurs utilisateurs. Pour ce faire, nous choisissons UML (Unifie Modeling Language) comme langage de modélisation de notre système car il est langage qui correspond à la programmation orienté objet.

Le langage UML a été pensé pour être un langage de modélisation visuelle commun, et riche sémantiquement et syntaxiquement. Il est destiné à l’architecture, la conception et la mise en œuvre de systèmes logiciels complexes par leur structure aussi bien que leur comportement. Il ressemble aux plans utilisés dans d’autres domaines et se compose de différents types de diagrammes. Dans l’ensemble, les diagrammes UML décrivent la limite, la structure et le comportement du système et des objets qui s’y trouvent.

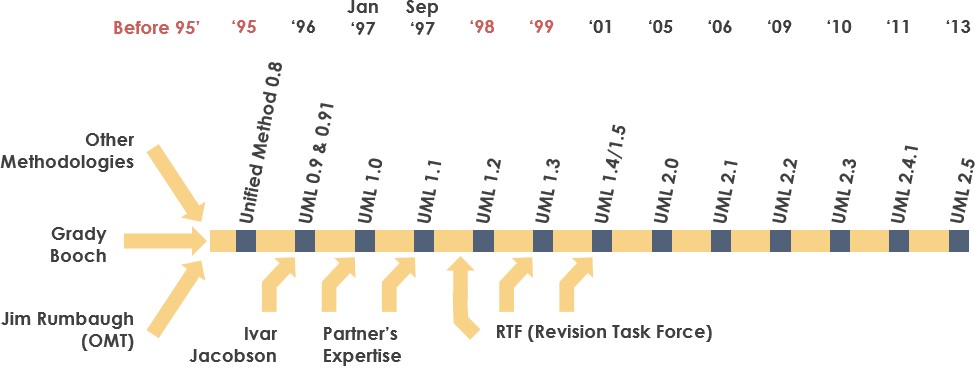
 UML est né de la fusion des langages de modélisation objet dominant à savoir OMT de James Rembauche, BOOCH de Gray Booch et OOSE d’Ivar Jacobson. Ce langage a été normalisé en janvier 1997 par l’OMG (Object Management Group) qui est une association américaine crée en 1989 dont l’objectif est de promouvoir et standardiser le modèle objet sous toutes ses formes.

Figure 2: Genèse d'UML( https://www.cybermedian.com/fr/a-comprehensive-guide-to-uml-class-diagram)

Il présente de nombreux avantages, à savoir :

* UML est un langage formel et normalisé ;
* UML permet un gain en précision ;
* UML est un support de communication performant ;
* UML facilite la compréhension des représentations abstraites complexes ;
* UML est un langage universel à cause de son caractère polyvalent et sa souplesse ;
* UML est basé sur une démarche itérative et incrémentale qui permet de s’y prendre plusieurs fois en affirmant son analyse par étapes ;
* UML est piloté par les besoins des utilisateurs.

Comme énoncé plus haut, UML se compose de diagrammes dépendants hiérarchiquement et qui se complètent de façon à permettre la modélisation d’un projet tout au long de son cycle de vie. A sa dernière version, la version 2.5, UML compte 14 diagrammes. Ainsi, UML définit 9 types de diagrammes dans deux catégories de vues, les vues statiques et les vues dynamiques.

* **Les vues statiques** 
  + Le diagramme des cas d’utilisation qui présente les possibilités d'interaction entre le système et les acteurs (intervenants extérieurs au système), c'est-à-dire de toutes les fonctionnalités que doit fournir le système ;
  + Le diagramme de classe qui décrit la structure statique, les types et les relations des ensembles d’objets ;
  + Le diagramme d’objet qui décrit les objets d’un système et leurs relations ;
  + Le diagramme de composants qui décrit les composants physiques du système d'un point de vue physique, tels qu'ils sont mis en œuvre ;
  + Le diagramme de déploiement qui présente l'utilisation de l'infrastructure physique par le système et la manière dont les composants du système sont répartis.
* **Les vues dynamiques** 
  + Le diagramme de séquence qui décrit de façon séquentielle le déroulement des traitements ;
  + Le diagramme d’activité qui présente les comportements du système ainsi que ses composants ;
  + Le diagramme de collaboration qui décrit les messages entre les objets
  + Le diagramme d’états-transitions qui présente les différents états des objets.

UML est utilisé pour spécifier, visualiser, modifier et construire les documents nécessaires au bon développement d'un logiciel orienté objet. Dans le cadre d’un projet de développement logiciel, UML doit être associé à un processus, qui permettra de définir les différentes étapes de développement. Il peut ainsi être associé à plusieurs processus (RUP, XUP,2UP…). Dans le cadre de notre projet, nous avons choisi de le lier au processus 2TUP.

## Processus de développement 2TUP

Le processus unifié (UP : Unified Process) est un processus générique de développement logiciel mené par UML. Générique signifie car il est nécessaire d’adapter UP au contexte du projet, de l’équipe, du domaine, et/ou de l’organisation. Parmi les adaptations on distingue par exemple :

* RUP (Rational Unified Process) ;
* EUP (Enterprise Unified Process) ;
* AUP (Agile Unified Process) ;
* 2TUP (Two Tracks Unified Process).

2TUP est un processus de développement logiciel qui implémente le processus unifié (c.à.d. itératif, incrémental, basé sur UML). Il propose un cycle de développement qui sépare les aspects techniques des aspects fonctionnels en partant du constat que toute évolution peut se traiter parallèlement, suivant un axe fonctionnel et un axe technique. Ensuite, et en fusionnant les résultats de ces deux axes (branches), on arrive à réaliser le système désiré. Ce qui nous donne un cycle de développement sous forme de Y. Il commence par une étude préliminaire qui consiste essentiellement à identifier les acteurs qui vont interagir avec le système à construire, identifier les messages qu'échangent les acteurs et le système, et à produire le cahier des charges

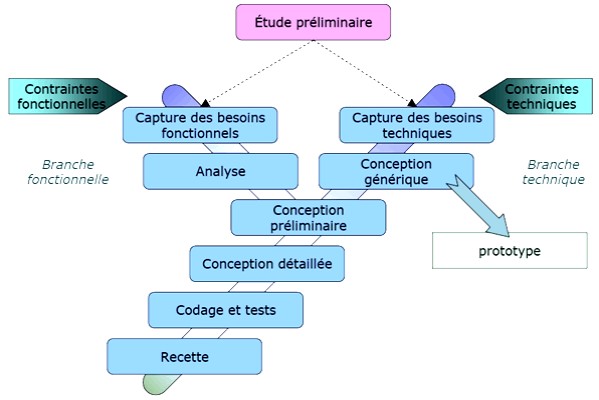
La figure ci-dessous présente l’illustration graphique du processus 2TUP :

Figure 3Illustration du cycle de développement 2TUP (Source https://www.nassimbahri.ovh/docs/conception/Chapitre-3/)

Le processus 2TUP est ainsi composé de 3 branches à savoir :

* **La branche gauche (fonctionnelle) comporte :**

* + **La capture des besoins fonctionnels**, qui produit un modèle des besoins focalisé sur le métier des utilisateurs. Elle qualifie au plus tôt le risque de produire un système inadapté aux utilisateurs. De son côté, la maîtrise d’œuvre consolide les spécifications et en vérifie la cohérence et l’exhaustivité l’analyse, qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en termes de métier. Les résultats de l’analyse ne dépendent d’aucune technologie particulière ;
  + **L’analyse**, qui consiste à étudier précisément les spécifications fonctionnelles à obtenir une idée de ce qui va réaliser le système en termes de métier.
* **La branche droite (technique) comporte :**

* + **La capture des besoins techniques**, qui recense toutes les contraintes et les choix qui circonscrivent la conception du système. Les outils et les matériels sélectionnés ainsi que la prise en compte de contraintes d’intégration avec l’existant conditionnent généralement des prérequis d’architecture technique ;
  + **La conception générique**, qui définit les composants nécessaires à la construction de l’architecture technique. Cette conception est la moins dépendante possible des aspects fonctionnels. Elle a pour objectif d’uniformiser et de réutiliser les mêmes mécanismes pour tout un système. L’architecture technique construit le squelette du système informatique et écarte la plupart des risques de niveau technique.

L’importance de sa réussite est telle qu’il est conseillé de réaliser un prototype pour assurer sa validité.

* **La branche du milieu (réalisation) comporte :** 
  + **La conception préliminaire**, qui représente une étape délicate, car elle intègre le modèle d’analyse fonctionnelle dans l’architecture technique de manière à tracer la cartographie des composants du système à développer. Cette étape permet de produire le modèle de conception système. Ce dernier organise le système en composants, délivrant les services techniques et fonctionnels ;
  + **La conception détaillée,** qui permet d’étudier comment réaliser chaque composant. Cette étape produit le modèle de conception des composants. Il fournit l’image prête à fabriquer du système complet. La conception détaillée permet de minimiser l’incapacité de l’architecture technique à répondre aux contraintes opérationnelles ;
  + **Le codage** qui quant à lui permet la production des composants et des tests des unités de code au fur et à mesure de leurs réalisations
  + **L’étape de recette**, qui consiste enfin à valider les fonctions du système développé

# Présentation des diagrammes

L’étude fonctionnelle nous a permis de définir les contraintes (ce que notre système doit réaliser en termes de métier). On y relève ainsi la capture des besoins fonctionnels, d’analyse et de spécification.

## Diagramme de cas d’utilisation

### Présentation

Les diagrammes de cas d’utilisation identifient les fonctionnalités fournies par le système (cas d’utilisation), les utilisateurs qui interagissent avec le système (acteurs), et les interactions entre ces derniers. Les cas d’utilisation sont utilisés dans la phase d’analyse pour définir les besoins de haut niveau du système. Les objectifs principaux des diagrammes des cas d’utilisation sont :

* Fournir une vue de haut-niveau de ce que fait le système ;
* Identifier les utilisateurs (acteurs) du système ;
* Déterminer des secteurs nécessitant des interfaces hommes-machine (IHM).

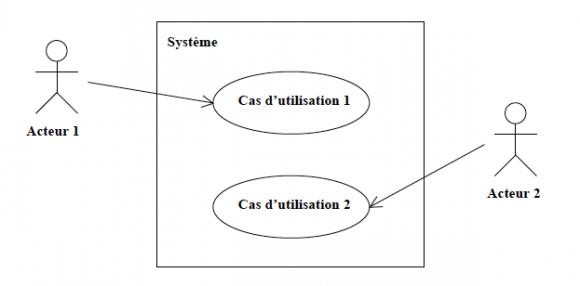
Les cas d’utilisation se prolongent au-delà des diagrammes imagés. En fait, des descriptions textuelles des cas d’utilisation sont souvent employées pour compléter ces derniers et représentent leurs fonctionnalités plus en détail.

### Formalisme

Les éléments constitutifs d’un diagramme de cas d’utilisation sont : les acteurs, les cas d’utilisation, et les associations.

La figure suivante présente de façon graphique le formalisme d’un diagramme de cas d’utilisation :

Figure 4: Formalisme graphique d'un diagramme de cas d’utilisation



#### Acteur

Un acteur est un type stéréotypé représentant une abstraction qui réside juste en dehors du système à modéliser.

En UML, l’acteur n’est pas nécessairement une personne physique, mais toute entité (une personne, un service, une société, un système informatique, bases de données, des équipements…) qui interagit avec le système.

Une même personne physique peut donc être représentée par plusieurs acteurs en fonction des rôles qu’elle joue. Formellement, un acteur est représenté en UML comme suit :

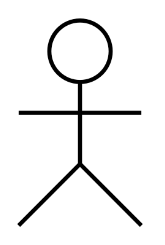


Figure 5:Formaliste d'un acteur

Nom acteur

#### Cas d’utilisation

Un cas d'utilisation représente une fonctionnalité fournie par le système, typiquement décrite sous la forme Verbe + objet.

Il correspond à un objectif du système, motivé par un besoin d’un ou plusieurs acteurs. L'ensemble des cas d’utilisation (use cases) décrit les objectifs (le but) du système.

Les cas d'utilisation sont représentés par une ellipse contenant leur nom comme illustré ci-après :

Nom du cas

Figure 6:Formalisme d'un cas d'utilisation

#### Association

Les associations sont utilisées pour lier des acteurs avec des cas d'utilisation. Elles indiquent qu'un acteur participe au cas d'utilisation sous une forme quelconque. Les associations sont représentées par une ligne reliant l'acteur et le cas d'utilisation.

Figure 7:Formalisme d'une association

#### Les Relations

En UML, les acteurs sont liés à leurs différents cas d’utilisation par des diverses relations qui leur lient. Celles-ci présentent un lien d’association entre ledit acteur et l’action qu’il mène dans le système. C’est ainsi que l’on distingue les relations suivantes :

Tableau 2:Différents types de relation dans un diagramme de cas d'utilisation

|  |  |  |
| --- | --- | --- |
| **Type de relation** | **Descriptions** | **Représentation graphique** |
| **Inclusion** | Un cas d’utilisation A inclus un cas d’utilisation B signifie que pour effectuer le cas d’utilisation A, il faut obligatoirement effectuer le cas d’utilisation B |  |
| **Extension** | On dit qu’un cas d’utilisation B étend un cas d’utilisation A si pendant que l’on effectue le cas   1. on a la possibilité d’effectuer 2. indépendamment de A |  |
| **Généralisation** | Un cas A est une généralisation d’un cas B si le cas B est une autre manière d’effectuer le cas A (spécialisation). |  |

### Identification des acteurs et cas d’utilisation

Tableau 3: Listes des acteurs et leurs roles

|  |  |
| --- | --- |
| Acteur | Rôle |
| **Utilisateur** | un employé ou bien un enseignant qui a la possibilité d’authentifier, |
| **Administrateur** | Le technicien informatique qui a la possibilité de gérer l’application, d’ajouter, supprimer ou modifier un matériel, consulter l’état du parc en temps réel, gérer les tickets (les interventions) |

### Diagramme de cas d’utilisation globale

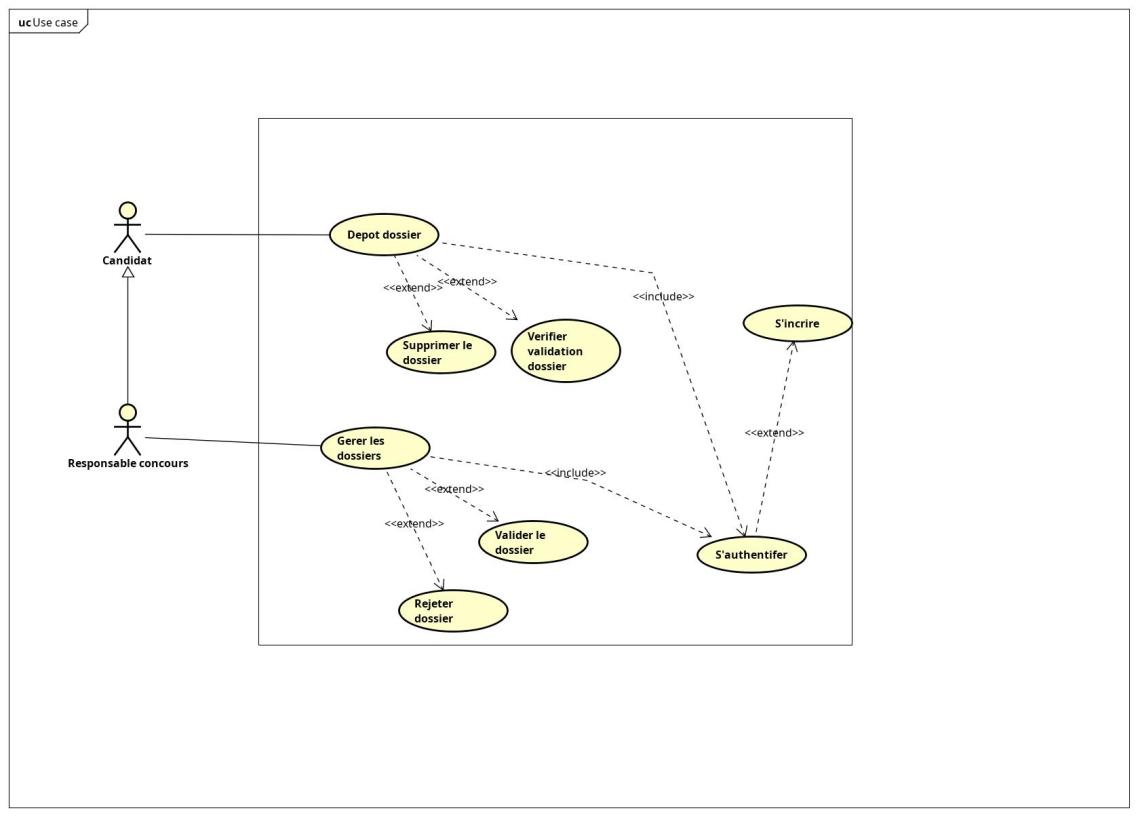


Figure 8: Diagramme de cas d'utilisation globale

|  |  |
| --- | --- |
| Présupposé | Le système fonctionne correctement |
| Acteur | Gestionnaire |
| Précondition | Le gestionnaire s’est authentifié |
| Déclencheur | L’administrateur veut gérerles candidatures (cas d’ajout) |
| Scénario | Scénario nominal :   1. Le gestionnaire ouvre la fenêtre d’administration 2. Il clique sur le bouton d’ajout 3. Il choisit l’action à exécuter 4. Le système renvoi le formulaire d’ajout 5. Le gestionnaire entre les informations du nouveau candidat 6. Le gestionnaire clique sur le bouton d’ajout   Scénario alternatif : si les valeurs des champs du formulaire d’ajout sont incorrectes (Action 5)   1. Le système envoie un message d’avertissement 2. Le scenario recommence à l’action 4. |
| Postcondition | Candidat ajouté |

|  |  |
| --- | --- |
| Présupposé | Le système fonctionne correctement |
| Acteur | Administrateur |
| Précondition | L’administrateur s’est authentifié |
| Déclencheur | L’administrateur veut gérer les matériels (cas d’ajout) |
| Scénario | Scénario nominal :   1. Le technicien ouvre la fenêtre d’administration 2. Il clique sur le bouton d’ajout 3. Il choisit l’action à exécuter 4. Le système renvoi le formulaire d’ajout 5. Le technicien entre les informations du nouvel matériel 6. Le technicien clique sur le bouton d’ajout   Scénario alternatif : si les valeurs des champs du formulaire d’ajout sont incorrectes (Action 5)   1. Le système envoie un message d’avertissement 2. Le scenario recommence à l’action 4. |
| Post condition | Matériel ajouté |

|  |  |
| --- | --- |
| Présupposé | Le système fonctionne correctement |
| Acteur | Administrateur |
| Précondition | L’administrateur s’est authentifié |
| Déclencheur | L’administrateur veut gérer les Gestionnaires(cas d’ajout) |
| Scénario | Scénario nominal :   1. Le gestionnaire ouvre la fenêtre d’administration 2. Il clique sur le bouton d’ajout 3. Il choisit l’action à exécuter 4. Le système renvoi le formulaire d’ajout 5. Le gestionnaire entre les informations du nouveau bureau 6. Le gestionnaire clique sur le bouton d’ajout   Scénario alternatif : si les valeurs des champs du formulaire d’ajout sont incorrectes (Action 5)   1. Le système envoie un message d’avertissement 2. Le scenario recommence à l’action 4. |
| Post condition | Gestionnaire ajouté |

## Diagramme de séquence

### Présentation

Le diagramme de séquence nous permettre de représenter les interactions entre les utilisateurs et la plateforme de gestion ainsi que la base de données tout en précisant la chronologie des échanges de message. Son but principal est de donner une description chronologique sur le déroulement des cas d’utilisation entre les acteurs et les objets ou objets entre eux.

### Formalisme

Les éléments constitutifs d’un diagramme de séquence sont : l’objet, l’acteur, la ligne de vie, l’activation, le message, le cadre du diagramme.

Tableau 4:Formalisme diagramme de séquence

|  |  |  |
| --- | --- | --- |
| **Eléments Descriptions** | | **Représentation graphique** |
| **Objet** | Les objets sont les instances de classe et sont rangés horizontalement | |  | | --- | | Objet | |
| **Acteur** | Un acteur est une personne qui interagit ou communique avec le système et les objets |  |
| **Ligne de vie** | La ligne de vie identifie l’existence de l’objet par rapport au temps |  |
| **Activation** | L’activation représente l’ensemble des opérations exécutées par un objet. |  |
| **Message synchrone** | Est une communication où l’émetteur reste en attente de la réponse à son message avant de poursuivre ses actions |  |
| **Message asynchrone** | Est une communication où l’émetteur n’attend pas la réponse à son message, il poursuit l’exécution de ses opérations. |  |

*:*

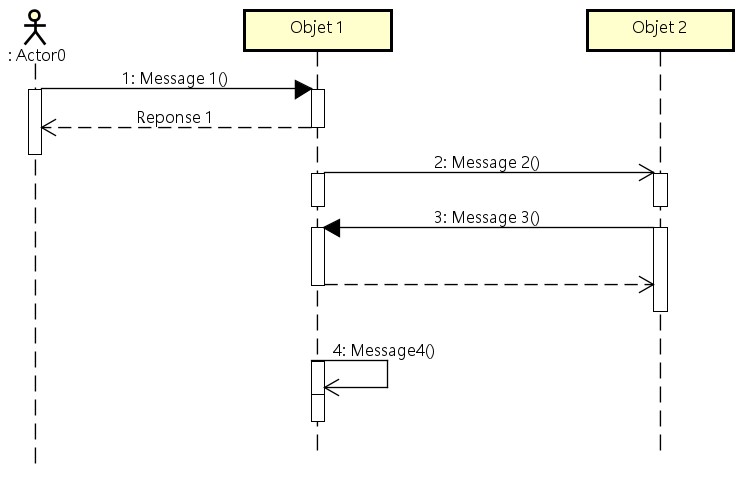


Figure 14: Exemple de diagramme de séquence

### Diagramme de séquence « s’authentifier »

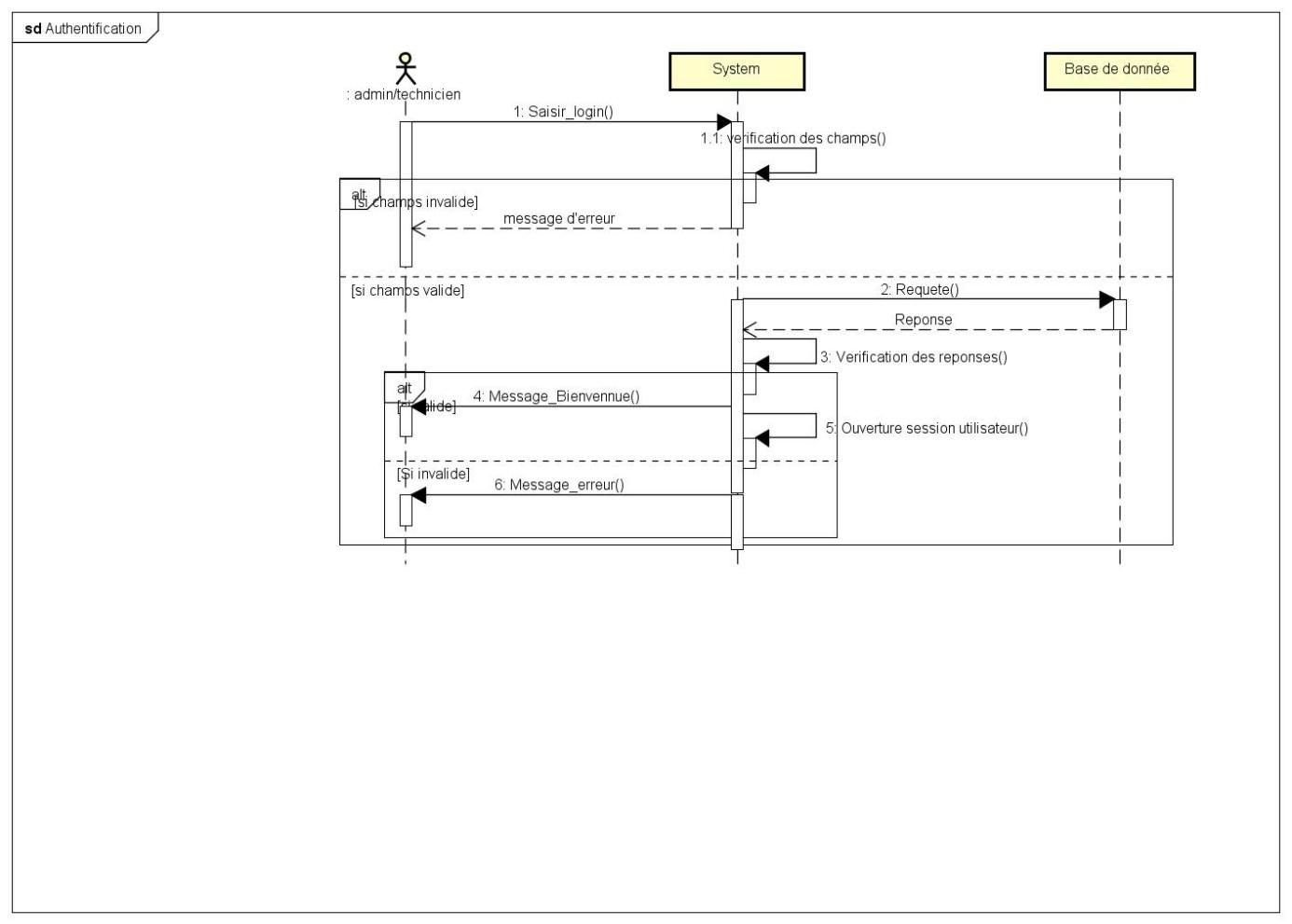


Figure 15: Diagramme de séquence "s'authentifier"

### /home/ramseys/Documents/IAI DEPOSIT/sequence depot.jpgsequence depotDigramme de séquence « d’ajout »

Figure 16:Diagramme de séquence "ajout"

# Conclusion

Au terme de cette partie, il était question de faire une analyse détaillée et profonde de notre projet. Il en ressort que, l’analyse de notre système, en passant par la détermination des cas d’utilisations nous a permis d’avoir une vision poussée, mais aussi une spécification plus précise des besoins du client du fait des actions qu’il aura à mener sur la plateforme. Ainsi elle nous a servi de base à la réflexion sur les mécanismes du système. Le langage de modélisation et le processus choisis, nous serons utiles par la suite dans l’énoncé du dossier de conception.

Contenu

INTRODUCTION

1. Objectif de conception
2. Présentation des diagrammes

intervenants

Conclusion

IIIème Partie

DOSSIER DE CONCEPTION

# Introduction

Ce document permet de modéliser dans son ensemble la solution proposée et de recueillir les informations nécessaires à la mise sur pied d’une base de données complexe et efficace. De manière globale, il nous offre une vue panoramique sur l’ensemble des éléments et interactions prise en compte dans le dossier d’analyse.

# Objectif de conception

La conception a pour objectif de permettre de formaliser les étapes préliminaires du développement d’un système afin de rendre son développement plus fidèle aux besoins du client. Pour ce faire tout à bord on est parti d’un énoncé informel (tels que les futurs utilisateurs du logiciel l’on spécifié) ainsi que l’étude et l’analyse de l’existant, la conception sera donc définie comme la description de l’objet à développer selon une vue interne, cette partie permet de décrire très simplement le fonctionnement du futur système pour en faciliter sa réalisation.

# Présentation des diagrammes

## Diagramme d’activité

### Présentation

Un diagramme d'activités (activités et transitions) est une variante du diagramme d'états-transitions (états et transitions). Il nous permet de mettre l’accent sur les différents traitements qui s’exécutent dans notre projet. Il permet de représenter la dynamique de notre système d’informations et montre les règles d’enchainement des activités entre l’utilisateur, le système (l’application) et la base de données.

### Formalisme

Les éléments constitutifs d’un diagramme d’activités sont : les objets, les acteurs, les lignes de vie, les activations, les messages entre les acteurs et objet ou objet et objet, le cadre du diagramme.

Tableau 5:Formalisme diagramme d’activité

|  |  |  |
| --- | --- | --- |
| **Eléments Descriptions Représentation graphique** | | |
| **Etat initial** | L’état initial marque le point d’entrée de la première activité. Il est représenté par un cercle plein. Il ne peut qu’y avoir un seul état initial sur un diagramme. |  |
| **Activité** | Marque une action faite par un objet. Il est matérialisé par un rectangle | Activité 1 |
| **Transition** | Quand une activité est accomplie, le traitement passe à un autre état d’acticité. Les transitions sont modélisées par des flèches. |  |
| **Décision** | Représente la fusion des différents flux, ou une décision possède au moins deux embranchements avec le texte de la condition permettant, aux utilisateurs de voir les options |  |
| **Barre de synchronisation** | Associe deux activités simultanées et les réintroduit dans un flux ou n’a lieu qu’une seule activité à la fois |  |
| **Etat final** | Marque la fin d’une activité et représente l’achèvement de tous les flux d’un procédé. |  |

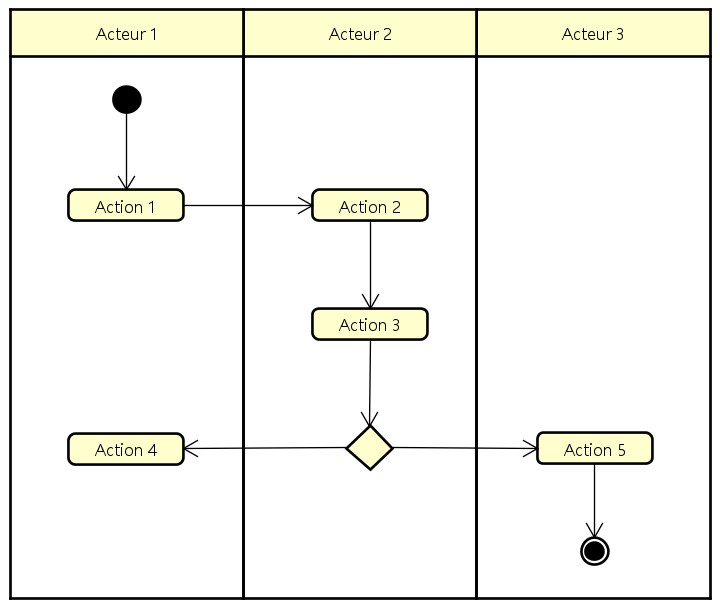


Figure 17: Exemple diagramme d'activité

### Diagramme d’activité « s’authentifier »

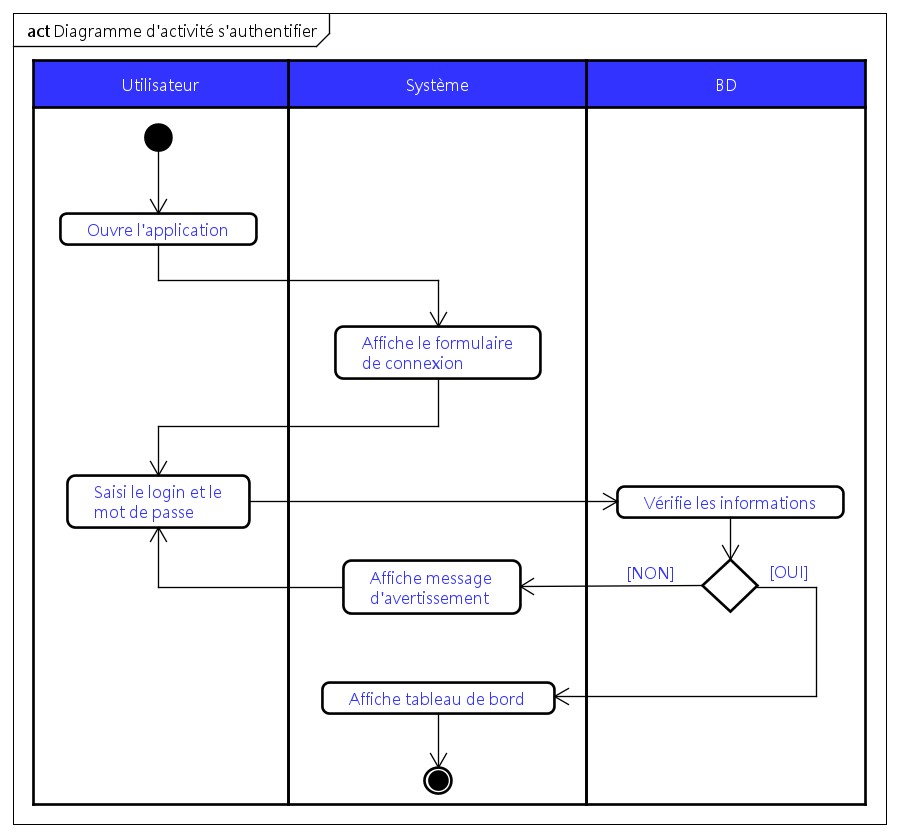


Figure 18:Diagramme d'activité "s'authentifier"

### 

### Diagramme d’activité « Enregistrement »

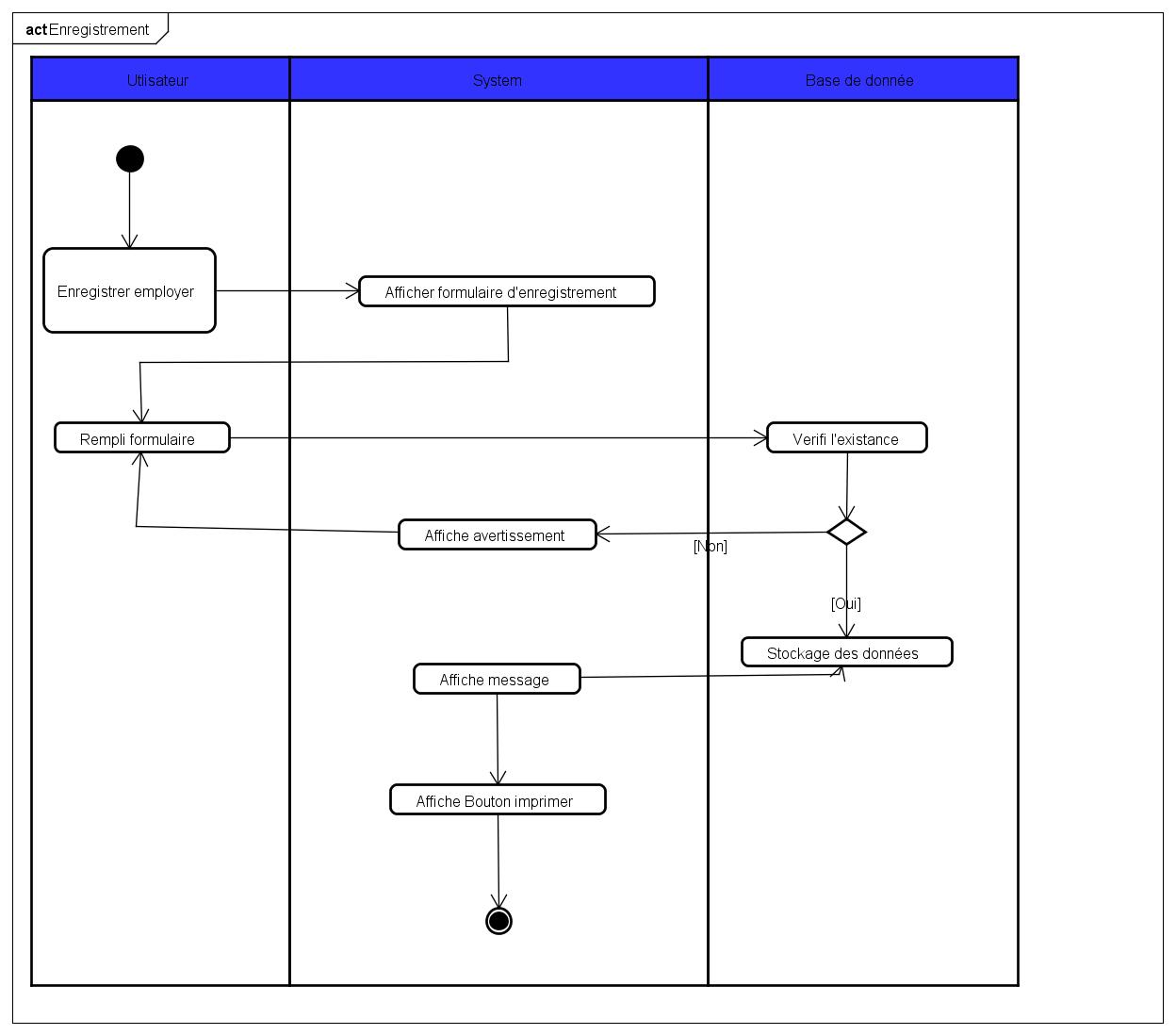


Figure 19:Diagramme d'activité enregistrement

### Diagramme d’activité « Modifier information employé »

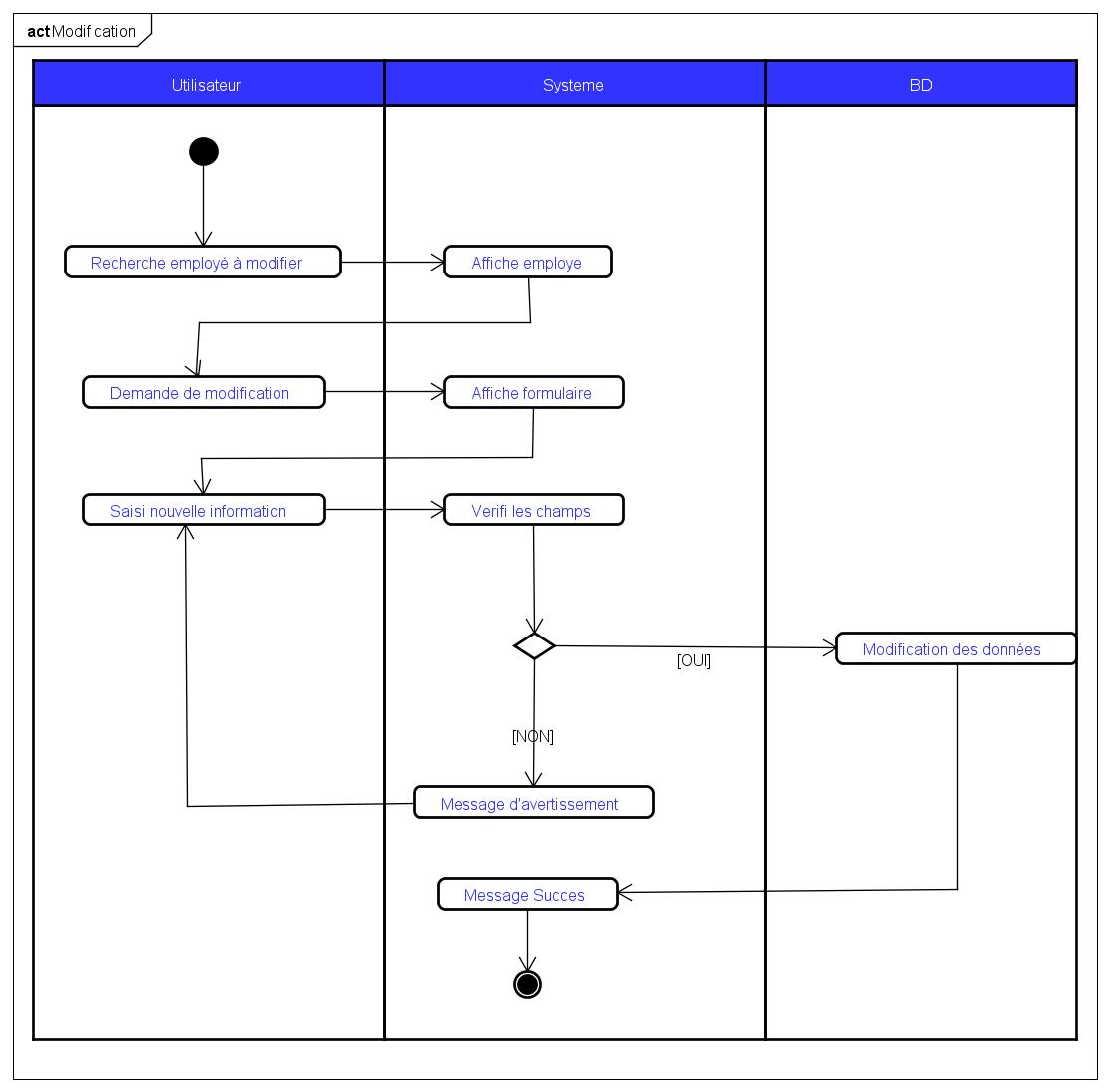


Figure 20:Diagramme d'activité "modification employe"

## Diagramme de classe

Un diagramme de classes est une collection d'éléments de modélisation statiques (classes, paquetages...), qui montre la structure d'un modèle. Il permet de définir quelles seront les composantes du système final : il ne permet en revanche pas de définir le nombre et l’état des instances individuelles. Néanmoins, on constate souvent qu’un diagramme de classes proprement réalisé permet de structurer le travail de développement de manière très efficace.

Cette phase de conception nous permet de déterminer les champs nécessaires pour la création de la base de donnée.

### Identification des classes et des attributs

Tableau 6:Listes des classes

|  |  |
| --- | --- |
| **Classe** | **Attributs** |
| Utilisateurs | Identifiant (N° C.I.N.), nom, prénom, login, mot de passe |
| **Gestionnaire** | post, lieu, date debut |
| **Candidat** | nom tuteur, numero tuteur |
| **Candidature** | documents pdf |
| **Dossier** |  |

### Représentation du diagramme des classes

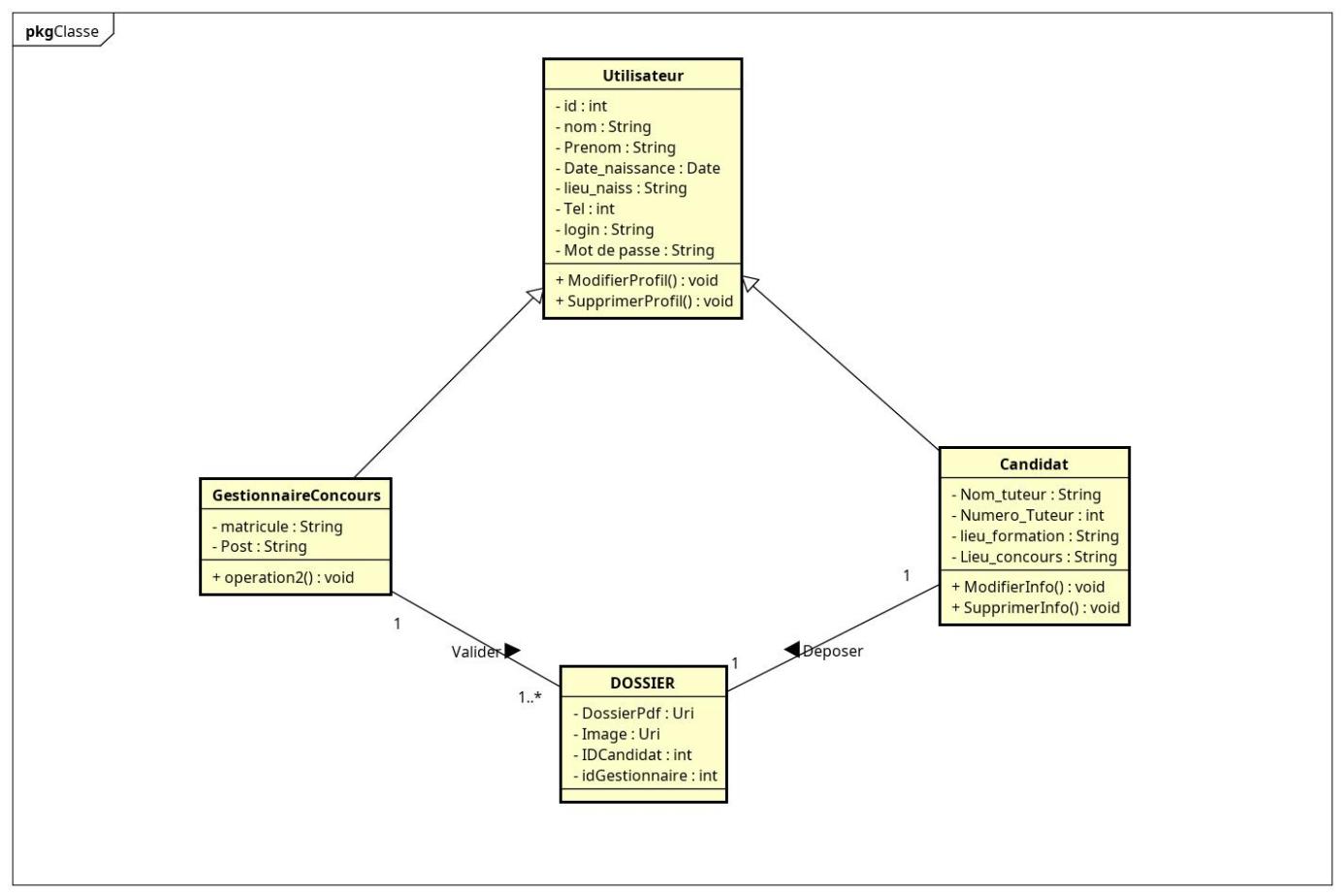


Figure 21:Diagramme de classe

# Conclusion

Le dossier de conception nous a permis de faire une analyse approfondie, de ressortir les données validées et les traitements à automatiser et nous offre tous les outils nécessaires pour une bonne réalisation. La branche technique se situe dans le processus de modélisation, un maillon important qui permet de définir et de spécifier les différents éléments architecturaux constituant l’application ou le système d’information étudié. Dans le dossier qui suivra, il sera question de dénombrer les solutions nous permettant de mettre en exergue l’architecture logicielle, matérielle et physique dudit système.

DOSSIER DE REALISATION

IVème Partie

Contenu

INTRODUCTION

1. Technologie utilisée
2. Présentation des diagrammes

intervenants

Conclusion

# Introduction

Le dossier de conception influence considérablement le dossier de réalisation ceci dans la mesure où il offre tous les outils nécessaires pour une bonne réalisation. Le maillon important dans cette chaine est la branche de réalisation. Elle permet de produire de façon concrète les objets issus de la branche technique. Il sera question pour nous tout au long de cette phase de mettre en exergue l’architecture logicielle, physique et matérielle dudit système, à travers les diagrammes tels que celui de composant, de déploiement, le modèle physique et l’architecture du code.

# Technologie utilisée

## Technologie Logiciel

Dans le cadre de la réalisation de notre projet, nous avons décidé d’utiliser les logiciels présentés dans le tableau ci-dessous :

Tableau 7:Outil logiciel utilisé

|  |  |  |
| --- | --- | --- |
| **Logo** | **Nom du logiciel** | **Rôle** |
| /home/ramseys/Documents/IAI DEPOSIT/téléchargement (2).pngtéléchargement (2) | Android Studio | Environnement de développement |
| /home/ramseys/Documents/IAI DEPOSIT/téléchargement.pngtéléchargement | Firebase | Serveur donnée |
|  | Google Chrome | Navigateur web |
|  | Adobe Photoshop | Création et traitement des images |

## Technologie Matériels

Pour la réalisation de notre projet, nous avons utilisés les matériels suivants :

Tableau 8:Outil materies

|  |  |
| --- | --- |
| **Matériel** | **Caractéristiques** |
| **Ordinateur portable** | Marque : FUJITSU  RAM : 4 Go  CPU : INTEL Core i5 2.5 GHz  DD : 500 Go |
| **Modem Wi-Fi** | Marque : MTN MIFI  Type de réseau : 4G  Opérateur : MTN |
| **ANDROID** | Marque: Samsung J6  Memoire: 4go RAM  ROM: 32GO |

## Les langages de programmations

Pour la mise sur pied de notre application et selon les besoins des utilisateurs finaux, le langage de programmation retenu est le **JAVA**. À ce langage, nous allons associer le **no Sql** de firebas**e**. Chacun de ces langages sera présenté de façon détaillée dans les paragraphes qui suivront.

### Le JAVA

Le **JAVA** est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982) présenté officiellement le 23 mai 1995. La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java.

Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être très largement utilisé pour le développement d'applications de types très disparates. Java est notamment largement utilisé pour le développement d'applications d'entreprises et mobiles.



Figure 22: Logo java

### Le SQL

Firebase est une plateforme de développement mobile et web qui offre des outils pour créer des applications de qualité tout en étant facile à utiliser. Elle a été acquise par Google en 2014 et est actuellement l’un des services cloud les plus populaires pour la création d'applications modernes.

Firebase a été créée en 2011 par deux ingénieurs de Google, James Tamplin et Andrew Lee, comme une plate-forme de développement pour les applications mobiles. À l'origine, il s'agissait d'un service d'hébergement de bases de données en temps réel, mais la plate-forme a rapidement évolué pour devenir une suite complète d'outils de développement d'applications mobiles et web.

Firebase offre un large éventail d'outils pour le développement d'applications, y compris un backend as a service (BaaS), des fonctionnalités d'authentification et de stockage de données cloud, des fonctionnalités de notification push, de la messagerie en temps réel et des analyses en temps réel. Tout cela peut être géré à partir d'une console unique et intégrée.

Firebase fonctionne avec une architecture en nuage, qui permet aux développeurs de se concentrer sur le développement en utilisant les outils fournis à la fois pour le frontend et le backend de l'application. Le backend est géré par les serveurs Firebase, ce qui simplifie la gestion de l'infrastructure pour les développeurs.



Figure 23:Logo firebase

# Architecture de déploiement

Pour avoir une architecture évolutive, robuste et modulable, il nous faut utiliser une architecture basée sur le système dit de « couche ». Nous allons donc séparer au maximum les différents types de traitements de l’application (présentation, métier, accès aux données). Ceci correspond à une architecture 3-tiers.

Dans cette approche, les couches communiquent entre elles et chacune propose un ensemble de service. Les services d’une couche sont mis à la disposition de la couche supérieure. On empêche donc qu’une couche invoque les services d’une couche plus basse que la couche immédiatement inférieure ou plus haute que la couche immédiatement supérieure. Le rôle de chaque couche étant bien défini, les fonctionnalités de l’une peuvent évoluer sans impliquer des changements dans les autres. Il s’agit donc d’un modèle logique d’architecture applicative qui vise à modéliser une application comme un empilement d trois couches logicielles dont le rôle est clairement défini :

## Couche de présentation (Premier niveau)

Correspond à l’affichage, la restitution sur les appareils clients, le dialogue avec les utilisateurs. On parle d’interface Homme-machine. Cette interface peut prendre multiples facettes, sans pour autant changer la finalité de l’application. La couche présentation relaie les requêtes de l’utilisateur à destination de la courbe métier et en retour lui présente les informations renvoyées par les traitements de cette couche.

## Couche de traitement (Deuxième niveau)

Correspond à la mise en œuvre de l’ensemble des règles de gestion et de la logique applicative. C’est la partie fonctionnelle de l’application, celle qui implémente la logique et qui décrit les opérations que l’application opère sur les données en fonction des requêtes des utilisateurs, effectués au travers de la couche présentation. Les différentes règles de gestion et de contrôle du système sont mises en œuvre dans cette couche. La courbe métier offre des services applicatifs et métiers à la couche présentation. Pour fournir ces services, elle s’appuie sur les données du système accessibles au travers des services de la couche inférieure. En retour, elle renvoie à la couche présentation les résultats qu’elle a calculés.

## Couche d’accès aux données (Troisième niveau)

Correspond aux données qui sont destinées à être conservées sur une durée finir voire même infinie. C’est la partie qui gère l’accès aux données du système. Ces données peuvent être propres au systèmes ou gérées par un autre système. La courbe métier n’a pas à s’adapter à ces deux cas, ils sont transparents pour elle car l’accès aux données se fait de manière uniforme.

# Présentation des diagrammes

## Diagramme de déploiement

Le diagramme de déploiement est une vue statique qui sert à représenter l’utilisation de l’infrastructure physique par le système et la manière dont les composants du système sont repartis ainsi que les relations entre eux. Le diagramme de déploiement se rapproche de la réalité physique puisqu’il identifie les éléments matériels, leur disposition physique et la disposition des composants sur ces éléments matériels. Le diagramme de déploiement permet donc de représenter l’architecture physique d’un système.

### Formalisme

Les éléments qui constituent le diagramme de déploiement sont :

Tableau 9:Formalisme diagramme de deploiement

|  |  |  |
| --- | --- | --- |
| **Nom** | **Description** | **Symbole** |
| **Composant** | Unité logicielle d’un système qui fournit et utilise les comportements via des interfaces et est représentée par un bloc rectangulaire | C  omposant |
| **Noeud** | Equipement matériel du système représenté par un bloc pavé |  |
| **Dépendance** | C’est un lien entre deux composants qui traduit une dépendance |  |
| **Association** | C’est un lien qui indique une ligne de communication entre les éléments matériels. |  |

### Diagramme de déploiement du système

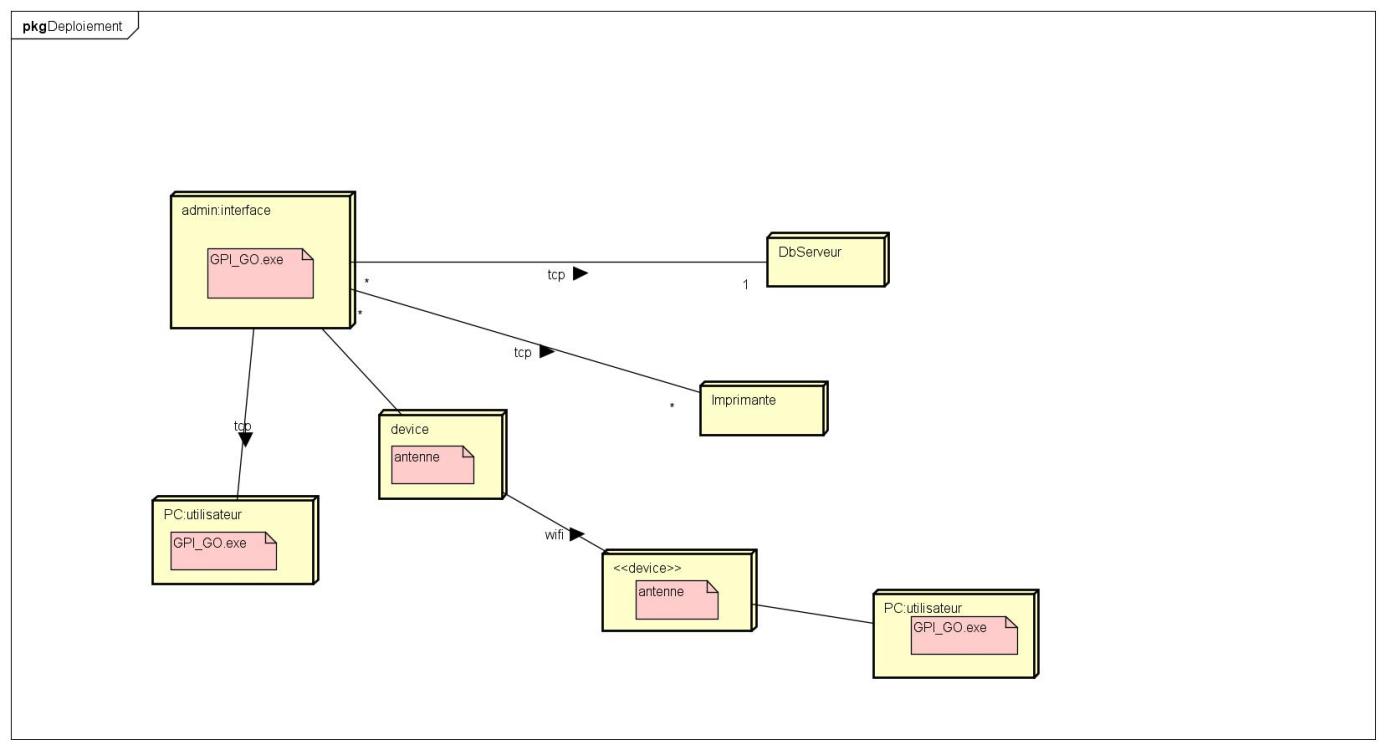


Figure 24:Diagramme de déploiement du système

# Conclusion

Dans cette partie, nous avons abordé plusieurs notions tour à tour l’architecture logicielle, matérielle, les ressources matérielles et logicielles entrant dans le projet. A la question de savoir comment sera déployée notre application, des éléments de réponses ont été apportés au travers des diagrammes tels que le diagramme de déploiement. Parvenus donc au terme de cette phase de notre cycle de développement, il est important de se poser la question de savoir si les utilisateurs du futur système seront à même de l’utiliser sans éprouver de difficultés ? La question à cette interrogation fera l’objet de la prochaine phase, dite guide d’utilisation.

# CONCLUSION GENERALE

En conclusion, le présent rapport a pour objectif de proposer une solution de création d'application de dépôt de dossier informatisé pour l'Institut Africain d'Informatique (IAI) du Cameroun. La solution proposée consiste en la création d'une application qui permettra de gérer les dépôts de dossier de manière efficace, tout en offrant des fonctionnalités telles que la saisie et le traitement des données, la gestion des autorisations et la sécurité des informations.

Cette solution permettra également aux utilisateurs de l'IAI du Cameroun de circonscrire un incident ou une demande de service via l'utilisation de tickets, tandis que les techniciens de l'IAI pourront suivre et gérer leur parc de manière fiable et efficace. L'application de gestion de parc informatique permettra de recenser les différents types de machines de l'Institut, leur localisation, leur nombre et leur état en cas de panne.

La mise en place de cette application permettra à l'IAI du Cameroun de bénéficier d'une meilleure gestion des processus de dossier, d'une réduction des temps de traitement, d'une gestion plus efficace des activités ainsi qu'une amélioration de la productivité générale. En somme, notre équipe est convaincue que la solution proposée répondra aux besoins de l'Institut Africain d'Informatique du Cameroun et constituera une référence en matière de dépôt de dossier informatisé.

# BIBLIOGRAPHIE

* Mme Mekongo Marie-Thérèse, Support de cours Rédaction scientifique, Institut

Africain d’informatique – Centre d’excellence technologique Paul Biya, 2021-2022 ;

* Mme Ada Rihanatou, Support de cours Conception des bases des données, Institut

Africain d’informatique – Centre d’excellence technologique Paul Biya, 2021-2022 ;

* M. Fomekong NZE Stephane, Support de cours UML 1.4, Institut Africain d’informatique – Centre d’excellence technologique Paul Biya, 2021-2022 ;

* M. Fomekong NZE Stephane, Support de cours Programmation Android, Institut

Africain d’informatique – Centre d’excellence technologique Paul Biya, 2021-2022 ;

* M. Mohamed Ali, Rapport de stage DTS sur « PLATEFORME DE STREAMING EN VOD (VIDEO ON DEMAND) », Institut Africain d’informatique – Centre d’excellence technologique Paul Biya, 2018-2019 ;

# WEBOGRAPHIE

* [https://www.technoscience.net/definition/670.html#:~:text=2TUP%20(2%20track%20 unified%20process,aspects%20techniques%20des%20aspects%20fonctionnels](https://www.technoscience.net/definition/670.html" \l ":~:text=2TUP%20(2%20track%20unified%20process,aspects%20techniques%20des%20aspects%20fonctionnels)

* <https://fr.wikipedia.org/wiki/Processus_unifi%C3%A9>
* [https://www.simplifiedcoding.net/android-mysql-tutorial-to-perform-basic-crudoperation](https://www.simplifiedcoding.net/android-mysql-tutorial-to-perform-basic-crud-operation)

* <https://www.androidhire.com/insert-data-from-app-to-mysql-android/>

* <https://waytolearnx.com/2019/10/mysql-foreign-key.html>

Table des matières

[RESUME 3](#_Toc262757645)

[ABSTRACT 4](#_Toc806180808)

[INTRODUCTION GENERALE 5](#_Toc1088659546)

[Introduction 7](#_Toc1880196590)

[I. Contexte et Justification 8](#_Toc2138444437)

[II. Expression des besoins 8](#_Toc1267009847)

[1. Les besoins fonctionnels 8](#_Toc1673039276)

[2. Besoins non fonctionnels 9](#_Toc1956383578)

[Conclusion 9](#_Toc2115330738)

[Introduction 11](#_Toc740469976)

[I. Analyse de l’existant 12](#_Toc2128414684)

[1. Description de l’existant 12](#_Toc897032923)

[2. Critiques de l’existant 12](#_Toc1182810365)

[3. Proposition de solution 12](#_Toc136022042)

[II. Présentation et justification des méthodologies de travail 13](#_Toc1544909765)

[1. Méthode d’analyse 13](#_Toc71728175)

[a) Etude comparative 13](#_Toc399480884)

[b) Présentation de UML 14](#_Toc1132775319)

[2. Processus de développement 2TUP 17](#_Toc2115181855)

[III. Présentation des diagrammes 19](#_Toc1636412757)

[1. Diagramme de cas d’utilisation 19](#_Toc2113713457)

[a) Présentation 19](#_Toc1249110403)

[b) Formalisme 20](#_Toc967495831)

[c) Identification des acteurs et cas d’utilisation 23](#_Toc1444259082)

[d) Diagramme de cas d’utilisation globale 23](#_Toc1142557155)

[2. Diagramme de séquence 27](#_Toc925325710)

[a) Présentation 27](#_Toc22444851)

[b) Formalisme 27](#_Toc672765357)

[c) Diagramme de séquence « s’authentifier » 28](#_Toc906171878)

[d) Digramme de séquence « d’ajout » 29](#_Toc1801032329)

[Conclusion 29](#_Toc1600324937)

[Introduction 31](#_Toc1168929523)

[I. Objectif de conception 31](#_Toc459729490)

[II. Présentation des diagrammes 31](#_Toc541500836)

[1. Diagramme d’activité 31](#_Toc901642465)

[a) Présentation 31](#_Toc450690279)

[b) Formalisme 31](#_Toc1808510683)

[c) Diagramme d’activité « s’authentifier » 33](#_Toc427198094)

[d) Diagramme d’activité « Enregistrement » 35](#_Toc259590210)

[e) Diagramme d’activité « Modifier information employé » 36](#_Toc1776357773)

[2. Diagramme de classe 37](#_Toc1167668070)

[a) Identification des classes et des attributs 37](#_Toc240521246)

[b) Représentation du diagramme des classes 38](#_Toc525907049)

[Conclusion 38](#_Toc202994787)

[Introduction 40](#_Toc376543288)

[I. Technologie utilisée 41](#_Toc2070816814)

[1. Technologie Logiciel 41](#_Toc274722962)

[2. Technologie Matériels 42](#_Toc776024172)

[3. Les langages de programmations 42](#_Toc1056108485)

[a) Le JAVA 43](#_Toc242421170)

[b) Le SQL 43](#_Toc264953281)

[II. Architecture de déploiement 44](#_Toc1022338295)

[1. Couche de présentation (Premier niveau) 45](#_Toc1491531573)

[2. Couche de traitement (Deuxième niveau) 45](#_Toc1232449113)

[3. Couche d’accès aux données (Troisième niveau) 45](#_Toc319113729)

[III. Présentation des diagrammes 46](#_Toc486605081)

[1. Diagramme de déploiement 46](#_Toc10291175)

[a) Formalisme 46](#_Toc341558581)

[b) Diagramme de déploiement du système 47](#_Toc1159370438)

[Conclusion 47](#_Toc916463053)

[CONCLUSION GENERALE 48](#_Toc2142590910)

[BIBLIOGRAPHIE 49](#_Toc612211727)

[WEBOGRAPHIE 50](#_Toc2085392576)