|  |  |  |
| --- | --- | --- |
|  |  |  |



Réalisation d’un Data Lake interopérable pour analyse et Visualisation des données médicales via application web

par

[Votre nom]

Mémoire présenté  
MASTER de SCIENCES, TECHNOLOGIES, SANTE, mention INFORMATIQUE  
Spécialité MOBIQUITE, BASES DE DONNEES ET INTEGRATION DE SYSTÈMES (MBDS)

[mois, année]

Jury :

Dr Olivier Robinson, président  
 M. Koto Rakotondralambo, examinateur  
 M. Harena Rabemanoela, encadreur professionnel

© [votre nom] , [année]

Résumé

Abstract

Table des matières

Liste des tableaux iv

Liste des figures v

Glossaire vi

Introduction 7

1 Présentation du stage 8

1.1 Présentation de l'entreprise 8

1.2 Présentation du sujet et objectifs du projet (problème traité) 8

2 État de l’art sur le sujet traité 9

2.1 Critères de comparaison 10

2.2 Etude de chaque solution au vu des critères 10

2.3 Tableau comparatif des solutions au vu des critères 10

3 Etude de l’existant et solution envisagée 11

3.1 Étude de l’existant 11

3.1.1 Description externe du système logiciel existant (vision utilisateur) 11

3.1.2 Description interne du système logiciel existant (vision développeur/conception) 11

3.2 Critique de l’existant 11

3.3 Solutions envisagées 11

3.4 Objectifs principaux et livrables 11

4 Démarche projet 12

4.1 Principes de la démarche projet 12

4.1.1 Activités d’ingénierie logicielle 12

4.1.2 Méthode de gestion de projet utilisée 12

4.1.3 Rôles et responsabilités 12

4.1.4 Outils 13

4.1.5 Gestion de la configuration 13

4.2 Contraintes et risques sur le projet 13

4.3 Démarche projet mise en œuvre 14

4.4 Budget du projet 14

5 Exigences réalisées dans le projet (vision externe/utilisateur) 15

5.1 Exigences fonctionnelles – Cas d’utilisation 15

5.1.1 Cas d’utilisation 1 15

5.1.2 Cas d’utilisation 2 15

5.1.3 Cas d’utilisation X 16

5.2 Exigences non fonctionnelles transverses 16

5.3 Interfaces détaillées 16

5.3.1 IHM 16

5.3.2 Interfaces avec d’autres systèmes 16

6 Architecture(s) système 17

7 Conception du système logiciel réalisée dans le projet (vision interne/développeur) 18

7.1 Plate-forme technique 18

7.2 Conception du logiciel développé 18

7.2.1 Conception du code source 18

7.2.2 Le code source – vue statique 18

7.2.3 Modélisation de données 18

7.2.4 Réalisation des cas d’utilisation 19

7.2.5 Les composants et leur déploiement 19

8 Tests du système logiciel 20

9 Conclusion générale 21

10 Références et Bibliographie 22

11 Annexes (à intégrer absolument si ce n’est pas déjà fait dans le rapport principal. Elles peuvent être dans des fichiers séparés) 23

NOTES IMPORTANTES (à supprimer sur le rapport final): 23

# Liste des tableaux

[Tableau 1 : Exemple tableau 12](#_Toc142476122)

# Liste des figures

[Figure 1: Exemple figure 19](#_Toc142476236)

# Glossaire

Astéroïde. Minuscule planète dont la taille varie entre 1 000 km à moins d'un km de diamètre. Les astéroïdes gravitent généralement autour de planètes plus grandes.

Atmosphère. Masse gazeuse entourant les planètes, dont la Terre.

Densité. Nombre de particules par unité de mesure.

# Introduction

Chaque jour, les établissements de santé génèrent des volumes colossaux de données médicales, souvent cloisonnées dans des systèmes hétérogènes, rendant leur analyse complexe et coûteuse. Selon une étude de 2023, 80 % des données médicales restent inexploitées en raison de problèmes d’interopérabilité (Source : Healthcare IT News). Ce stage s’inscrit dans cette problématique cruciale en visant à concevoir un Data Lake interopérable pour centraliser, analyser et visualiser ces données via une application web intuitive.

* Plan de formation/carrière : Étudiant en informatique, spécialisé en science des données et systèmes d’information, je souhaite évoluer vers des postes d’ingénieur en data engineering ou d’architecte de solutions Big Data dans le secteur de la santé.
* Raisons du choix du stage : Ce stage m’a attiré pour son caractère innovant, combinant Big Data et santé, un secteur où l’analyse de données peut sauver des vies. L’entreprise choisie est reconnue pour ses solutions avancées en interopérabilité des systèmes médicaux.
* Objectifs/attentes : Approfondir mes compétences en gestion de données massives, développer une application web robuste, et contribuer à un projet à fort impact sociétal tout en maîtrisant des technologies comme Hadoop, Spark et des standards comme FHIR.
* Présentation de l’entreprise : L’entreprise, un acteur clé dans les technologies de la santé, développe des solutions pour l’intégration et l’analyse de données médicales, servant des hôpitaux et cliniques à l’échelle internationale.
* Missions du stage : Conception d’un Data Lake interopérable, intégration de données médicales hétérogènes, développement d’une application web pour la visualisation, et mise en place de pipelines de traitement des données.
* Problématique : Comment concevoir un Data Lake interopérable qui garantisse une intégration efficace des données médicales hétérogènes tout en offrant des performances optimales pour l’analyse et la visualisation via une interface web ?
* Plan du rapport :
  + Chapitre 1 : Présentation de l’entreprise et des objectifs du projet.
  + Chapitre 2 : État de l’art sur les Data Lakes et solutions d’interopérabilité.
  + Chapitre 3 : Analyse de l’existant et solutions envisagées.
  + Chapitre 4 : Démarche projet, incluant méthodologie, outils et gestion des risques.
  + Chapitre 5 : Exigences fonctionnelles et non fonctionnelles.
  + Chapitre 6 : Architecture système du Data Lake et de l’application web.
  + Chapitre 7 : Conception technique du système.
  + Chapitre 8 : Tests réalisés et résultats.
  + Chapitre 9 : Conclusion et perspectives.
  + Chapitre 10 : Références et bibliographie.
  + Chapitre 11 : Annexes.

# Présentation du stage

## Présentation de l'entreprise

La société Madagascar Medical Technology a été créée en 2009 afin de répondre aux besoins des professionnels de la santé à Madagascar. Notre secteur d’activité englobe la distribution et la maintenance de matériels biomédicales, la fourniture de consommables, ainsi que la gestion de stock de nos clients. Nous sommes composés d’une équipe dynamique, passionné par son métier et prêt à suivre l’évolution du monde de l’ingénierie biomédicale.

MMT a conclu une convention avec Siemens Healthineers pour avoir le statut de Business Partner à Madagascar avec un rattachement direct avec la branche Sud-Africaine de Siemens Healthineers.

## Présentation du sujet et objectifs du projet

**Caractère innovant** : Le projet propose un Data Lake interopérable capable de centraliser des données médicales issues de sources hétérogènes (dossiers médicaux électroniques, bases de données relationnelles) tout en respectant les standards d’interopérabilité comme FHIR. Contrairement aux solutions traditionnelles, ce Data Lake permet des analyses avancées et des visualisations dynamiques via une application web.

**Contexte métier** : Dans le secteur médical, les données sont souvent fragmentées entre systèmes propriétaires, rendant l’analyse globale difficile. Ce projet vise à répondre à ce défi en fournissant une solution centralisée et accessible.

**Enjeux et risques** :

* **Enjeux** : Améliorer la prise de décision clinique grâce à des visualisations en temps réel, réduire les coûts d’intégration des données, et assurer la conformité aux réglementations (RGPD, HIPAA).
* **Risques** : Complexité de l’intégration des données hétérogènes, performances du Data Lake avec des volumes massifs, et sécurité des données sensibles.

# État de l’art sur le sujet traité

## Critères de comparaison

Pour évaluer les solutions de Data Lakes dans le domaine médical, les critères suivants ont été définis :

* **Interopérabilité** : Support des standards comme FHIR, HL7.
* **Performance** : Temps de traitement des données et scalabilité.
* **Sécurité** : Conformité RGPD/HIPAA, chiffrement des données.
* **Facilité d’utilisation** : Qualité des interfaces de visualisation.
* **Coût** : Infrastructure et maintenance.

## Etude de chaque solution au vu des critères

AWS HealthLake : Solution cloud d’Amazon, intégrant FHIR, avec des outils d’analyse avancés. Points forts : scalabilité, sécurité robuste. Limites : coûts élevés, dépendance au cloud AWS.

Google Cloud Healthcare API : Supporte FHIR et HL7, avec intégration BigQuery pour l’analyse. Points forts : performance, intégration IA. Limites : complexité de configuration.

Microsoft Azure Health Data Services : Compatible FHIR, avec Azure Synapse pour l’analyse. Points forts : intégration avec l’écosystème Azure. Limites : personnalisation limitée.

Solution maison (projet) : Data Lake basé sur Hadoop/Spark avec une application web React. Points forts : flexibilité, coût maîtrisé. Limites : maturité moindre par rapport aux géants du cloud.

## Tableau comparatif des solutions au vu des critères

Tableau 1 : Exemple tableau

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Position Jan ‘14 | Position Jan ‘13 | Language | Share Jan ‘14 | Trend |
| 1 | 1 | Java | 26.2% | -0.6% |
| 2 | 2 | PhP | 13.2% | -1.6% |
| 3 | 6 | Python | 10.2% | +1.3% |
| 4 | 3 | C# | 9.6% | -0.4% |
| 5 | 4 | C++ | 8.9% | 0% |
| 6 | 5 | C | 8.1% | -0.2% |
| 7 | 7 | JavaScript | 7.6% | +0.3% |

# Etude de l’existant et solution envisagée

## Étude de l’existant

### Description externe du système logiciel existant (vision utilisateur)

L’existant comprend des bases de données relationnelles (PostgreSQL) pour les dossiers médicaux et des fichiers CSV issus de capteurs IoT. Les utilisateurs accèdent aux données via des rapports statiques générés manuellement, ce qui limite l’analyse en temps réel.

### Description interne du système logiciel existant (vision développeur/conception)

**Plate-forme technique** : Serveurs locaux avec PostgreSQL et scripts Python pour l’extraction de données.

**Structure** : Bases de données relationnelles avec tables pour patients, diagnostics, et traitements. Aucun système centralisé pour les données non structurées (images, textes)

## Critique de l’existant

**Positif** : Stabilité des bases PostgreSQL, fiabilité des données structurées.

**Négatif** : Absence d’intégration des données non structurées, manque d’outils de visualisation dynamique, et absence de conformité FHIR.

## Solutions envisagées

**Data Warehouse** : Centralisation dans un entrepôt de données relationnel. Rejeté car inadapté aux données non structurées.

**Data Lake cloud (AWS/Google)** : Solution performante mais coûteuse. Rejetée pour des raisons budgétaires.

**Data Lake maison** : Basé sur Hadoop/Spark avec une application web React. Choisi pour sa flexibilité et son coût modéré.

## Objectifs principaux et livrables

**Objectifs** : Intégrer des données médicales hétérogènes, permettre des analyses statistiques, et offrir une visualisation interactive.

**Livrables** : Data Lake opérationnel, application web, pipelines de données, documentation technique.

# Démarche projet

## Principes de la démarche projet

### Activités d’ingénierie logicielle

Exigences : Recueillir les besoins des utilisateurs (médecins, analystes).

Conception : Modélisation UML, architecture Data Lake.

Codage : Développement des pipelines (Python, Spark) et de l’application web (React).

Tests : Tests unitaires (pytest), tests d’intégration, et tests utilisateurs.

Stratégie de test : Tests automatisés pour les pipelines, tests manuels pour l’IHM.

Activités réalisées : Conception, codage, tests unitaires.

### Méthode de gestion de projet utilisée

Méthode **Scrum** avec des sprints de 2 semaines. Mon rôle : développeur et participation aux daily scrums. Communication via réunions hebdomadaires et rapports sur Jira

### Rôles et responsabilités

**Client** : Équipe médicale définissant les besoins.

**Équipe projet** : 1 chef de projet, 1 data engineers, 1 développeur web (moi).

**Mon rôle** : Développement des pipelines et de l’application web, rédaction de la documentation.

### Outils

IDE : VS Code pour Python et React.

Gestion de projet : Jira pour le suivi des tâches.

Modélisation : Draw.io pour les diagrammes UML.

Mon rôle : Proposition de l’utilisation de Draw.io et configuration de VS Code

### Gestion de la configuration

**Organisation des livrables** : Répertoire Git avec dossiers pour code source (src/), documentation (docs/), et tests (tests/).

**Règles de nommage** : Préfixe dl\_ pour les scripts Data Lake, web\_ pour l’application web.

**Sauvegardes** : GitLab avec branches main, dev, et versionnage sémantique.

## Contraintes et risques sur le projet

Présenter les contraintes et les risques du projet.

Les risques peuvent être présentés dans un tableau. Par exemple:

Un plan de risque doit être exécutable. Les risques, les facteurs contributions et les solutions doivent être précis.

Être Malade n’est pas risque. C’est facteur contribuant.

Un risque c’est de ne pas pouvoir atteindre un de vos objectifs (une ou plusieurs de

vos exigences fonctionnelles ou non fonctionnelles, …).

Une solution proposée peut consister à faire quelque chose dans une des premières itérations pour lever le risque.

## Démarche projet mise en œuvre

Le projet a été découpé en quatre grandes étapes, alignées avec les catégories du timesheet : **Préparation & Installation**, **Backend – Construction du Data Lake**, **Frontend – Next.js + D3.js**, et **Test + Validation + Déploiement**. Chaque étape a été réalisée dans le cadre d’une méthodologie **Scrum** avec des sprints de deux semaines. Les tâches ont été priorisées dans un backlog, et les sprints ont été planifiés pour couvrir les activités listées dans le timesheet.

#### Backlog (extrait)

* **User Story 1** : En tant que médecin, je veux visualiser les tendances des données patient via des graphes dynamiques pour faciliter le diagnostic.
* **User Story 2** : En tant qu’administrateur, je veux intégrer des données médicales hétérogènes dans un Data Lake interopérable.
* **User Story 3** : En tant qu’analyste, je veux exécuter des requêtes statistiques sur les données du Data Lake.
* **Tâches techniques** : Installation Hadoop, développement Spark, création API, composants D3.js, etc.

#### Découpage en sprints

* **Sprint 1 (01/03/2025 - 15/03/2025)** : Prise en main, apprentissage Next.js/D3.js, étude architecture Data Lake, installation VM (Vagrant), configuration Hadoop/Hive.
* **Sprint 2 (16/03/2025 - 30/03/2025)** : Définition schéma logique, développement Spark (ingestion, nettoyage, transformation), intégration HBase/MongoDB.
* **Sprint 3 (01/04/2025 - 15/04/2025)** : Initialisation projet Next.js, authentification (NextAuth + JWT), connexion API backend, composants D3.js.
* **Sprint 4 (16/04/2025 - 30/04/2025)** : Finalisation graphes dynamiques, gestion erreurs UI, création tables Hive, API Hive/MongoDB.
* **Sprint 5 (01/05/2025 - 15/05/2025)** : Tests cohérence données, tests UI/API, corrections backend/frontend.
* **Sprint 6 (16/05/2025 - 31/05/2025)** : Export VM Vagrant, déploiement web app, documentation, formation utilisateur.

**Rôle dans l’élaboration** : J’ai défini les tâches du backlog et participé à la planification des sprints avec l’équipe. Aucun écart significatif n’a été observé, car les délais ont été respectés grâce à une priorisation efficace.

## Budget du projet

Le coût de projet au moins par rapport à vous. Locaux, machines, salaires, …

# Exigences réalisées dans le projet (vision externe/utilisateur)

Dans le cas d’un existant, décrire ici uniquement les nouvelles exigences réalisées au sein du projet.

S'inspirer de l’étude de cas du cours démarche de conception.

Notes : Les cas d’utilisations peuvent être remplacés par les User Stories si vous utilisez SCRUM par exemple.

## Exigences fonctionnelles – Cas d’utilisation

Donner ici le tableau de toutes vos cas d’utilisations.

Choisir ensuite 3 ou 4 cas d’utilisation représentatifs que vous allez décrire dans ce cette section. Mettre en annexe la description des autres cas d’utilisation. Expliquer la valeur de ces cas d'utilisation pour l'utilisateur.

### Cas d’utilisation 1

Description textuelle détaillée (voir un cours UML). Un cas doit contenir :

* Un nom
* Des préconditions
* Des postconditions
* Une description textuelle détaillée
* Des codes et messages d’erreurs.
* …

Un cas peut être accompagné de certains diagrammes UML :

* Des diagrammes de séquence (niveau système logiciel boîte noire, interactions entre le système et les acteurs extérieurs)
* Des diagrammes d’états
* Des diagrammes d’activités
* Etc.

Un cas peut être accompagné d’écrans (voir la section 6.3)

### Cas d’utilisation 2

…

### Cas d’utilisation X

## Exigences non fonctionnelles transverses

Utilisabilité (charte graphique, ergonomie générale) Performances (temps de réponses, volumétrie) Robustesse (tolérances aux pannes, disponibilité) Sécurité

Supportabilité (maintenabilité, évolutivité)

Les exigences fonctionnelles doivent être décrites de façons concrètes et chiffrés. Par exemple

si le temps de réponse d’une page web.

## Interfaces détaillées

### IHM

CU d’utilisation X

écran 1

écran 2

Faire suivre chaque écran d’une description de ce qui s’y passe.

### Interfaces avec d’autres systèmes

Description d’API par exemple

# Architecture(s) système

Dans ce chapitre vous devez décrire les architectures cibles de vos projets :

* L´architecture logicielle est une vue tournée sur l'organisation interne et le découpage en couches et modules du ou des logiciels du système informatique. Les responsabilités de chaque module et la nature et la structure des relations entre modules.
* L´architecture technique est une vue tournée vers les différents éléments matériels et l'infrastructure dans laquelle le système informatique s'inscrit, les liaisons physiques et logiques entre ces éléments et les informations qui y circulent.

Il est important de décrire ces architectures et de positionner votre contribution.



Figure 1: Exemple figure

# Conception du système logiciel réalisée dans le projet (vision interne/développeur)

Dans le cas d’un existant, faire référence à l’existant. Bien faire ressortir ce qui change ou ne change pas par rapport à l’existant. Justifier les choix. Préciser votre rôle dans ces choix. Dans le cas d'un existant, faire ressortir ce qui change par rapport à l'existant (si aucun changement sur un aspect, faire référence au chapitre décrivant l'existant).

(voir l’étude de cas du cours démarche de conception).

## Plate-forme technique

Architecture matérielle et framework logiciel

## Conception du logiciel développé

### Conception du code source

Expliquer les choix structurants sur concernant la structuration en unités de code (par exemple en Java, la structuration en classes et packages) et en tables.

Expliquer dans quelle mesure cette structuration applique les bonnes pratiques de conception. Expliquer les règles de nommage.

### Le code source – vue statique

Architecture du code (structuration en packages)

Pour chaque package: structuration en classes/fichiers de code source – des exemples d'implémentation de code source peuvent être présentés (ou bien mis en annexe et référencés ici).

### Modélisation de données

Décrire ici le modèle conceptuel MERISE (dictionnaire de données et MCD) ou UML (modèles de classes).

Décrire ensuite le modèle logique de données.

### Réalisation des cas d’utilisation

Choisir un cas d'utilisation significatif (1 de ceux présentés dans le chapitre précédent) et décrire le scénario de cas d’utilisation dans une vision interne/développeur (diagramme de séquence système logiciel boîte blanche)

### Les composants et leur déploiement

Par exemple, en JEE, la structuration en fichiers .war, ejb-jar …

Préciser les règles de nommage.

# Tests du système logiciel

En cohérence avec la stratégie de test décrite au chapitre 3, décrire les tests réalisés et leurs résultats. Préciser votre rôle dans la définition et l'exécution des tests.

# Conclusion générale

La conclusion est l’occasion pour vous de faire un bilan sur votre projet. Ce bilan doit

comporter les parties suivantes :

* Le bilan des résultats obtenus pour l’entreprise
* Le bilan des problèmes rencontrés et des solutions apportés
* Les perspectives du projet
* Le bilan personnel

**Le bilan des résultats obtenus pour l’entreprise**

Précisions sur le travail réalisé (nombre de programmes réalisés, nombre de classes implantées, ..., nombre de lignes de code, ...)

Précisions sur la situation finale de l’application réalisée (Le statu des livrables : en production, ..., en test, finis à X %, etc.)

Les livrables peuvent être : des releases d’une application, des uses cases traités, des users stories traitées, des sprints traités, des modules traités, etc.

**Le bilan des problèmes rencontrés et des solutions apportés**

Il s’agit ici de rapporter les principaux problèmes rencontrés et la façon dont ils ont été résolus.

**Les perspectives du projet**

Quelle est la suite possible à votre projet :

* Amélioration
* Nouveaux services
* Etc.

**Le bilan personnel**

Indiquer ici ce que ce stage et/ou projet a pu vous apporter.

# Références et Bibliographie

1. Auteur1, auteur 2, …, Titre\_du\_livre, Edition, date\_parution, lien s’il y a

…

1. Auteur1, auteur 2, …, Titre\_article, Revue de parution ou site Web, date\_parution, lien s’il y a …

Les liens web seuls ne peuvent être une bibliographie. Vous devez mettre dans la bibliographie : des articles de revues, des livres, des documents en respectant les exemples ci-dessus. Si vous avez un doute, merci d’aller voir les références et bibliographies de n’importe quel article WIKIPEDIA.

# Annexes (à intégrer absolument si ce n’est pas déjà fait dans le rapport principal. Elles peuvent être dans des fichiers séparés)

Annexe 1 : Exigences détaillées (correspond à peu près au cahier des charges)

Annexe 2 : Modèle conceptuel de données (diagramme de classes)

Annexe 3 : Dossier technique (Code réalisé y compris sa documentation)

Annexe 4 : Présentation des outils de développement utilisés

…

NOTES IMPORTANTES (à supprimer sur le rapport final):

* Les 40 pages maximums du rapport n’intègrent pas les annexes
* Concernant le stage, il y’aura trois soutenances (une première intermédiaire en Mars Orienté Organisation en présence du professeur de gestion de projet, un deuxième intermédiaire fin mai/début juin et la finale Début Septembre). Le professeur de gestion de projet donnera une note à lui au vu de la 1ère soutenance. Vous devez donc donner un maximum d’informations dans le rapport et ses annexes.
* Les éléments d’organisation du projet (risques, planning, coûts, etc.) sont des objets vivants. Ils doivent évoluer avec le projet. Ils doivent être mis à jour régulièrement. Ils doivent être utilisables.