

République Tunisienne  
Ministère de l'Enseignement Supérieur  
et de la Recherche Scientifique

Université de Sfax

École nationale d'électronique  
et des télécommunications de Sfax



Ingénieur en :  
Ingénierie des données  
et Systèmes décisionnels

Option :  
Science des Données

---

# RAPPORT DE STAGE D'ETE

*présenté à*

**L'École Nationale d'Électronique et des  
Télécommunications de Sfax**

**Diplôme National d'Ingénieur en :  
Ingénierie des données et Systèmes décisionnels**

*par*

**Mohamed Amine MISSAOUI**

---

**AgriCareAi :  
Développement d'une Application Mobile  
pour la Gestion Météo, la Santé des Cultures  
et l'Assistance Agricole**

---

Année Universitaire :  
**2024-2025**

# REMERCIEMENT

Au terme de ce projet, je souhaite exprimer ma sincère reconnaissance à tous ceux qui ont contribué à sa réalisation.

Je tiens à exprimer toute ma gratitude à l'équipe de la société de fondation Djagora pour m'avoir généreusement accueilli au sein de ses rangs, pour m'avoir motivé et pour avoir témoigné une grande confiance en mes compétences tout au long de mon stage.

J'exprime ma profonde gratitude à mon encadrant professionnel, **Mr. LOUATI Mahdi**, pour sa précieuse contribution. Ses conseils avisés, son expertise, et ses suggestions constructives ont été d'une aide inestimable et ont grandement enrichi mon travail. J'ai eu la chance de bénéficier de son évaluation approfondie et de ses recommandations, pour lesquelles je lui suis particulièrement reconnaissant.

Je souhaite aussi remercier les membres du jury qui ont accepté d'examiner et d'évaluer ce modeste travail, en espérant que ce rapport saura répondre à leurs attentes et refléter la clarté qu'ils recherchent.

Enfin, mes remerciements vont à tous ceux qui ont, de près ou de loin, contribué à la réussite de ce projet. Leur soutien, tant moral que technique, a été crucial pour mener à bien cette réalisation.

# ■ TABLE DES MATIÈRES

|   |     |
|---|-----|
| <b>LISTE DES FIGURES</b>                                  | v   |
| <b>LISTE DES TABLEAUX</b>                                 | vi  |
| <b>LISTE DES ABRÉVIATIONS</b>                             | vii |
| <b>INTRODUCTION GÉNÉRALE</b>                              | 1   |
| <b>1 Cadre Général du Projet</b>                          | 2   |
| I.    Introduction . . . . .                              | 3   |
| II.    Présentation de la Société d'Accueil . . . . .     | 3   |
| II.1.    Présentation . . . . .                           | 3   |
| II.2.    Secteur d'Activité . . . . .                     | 3   |
| III.    État de l'art . . . . .                           | 4   |
| III.1.    Cadre du Projet . . . . .                       | 4   |
| III.2.    Étude de l'existant . . . . .                   | 5   |
| III.3.    La Problématique . . . . .                      | 5   |
| III.4.    Objectifs . . . . .                             | 6   |
| IV.    CONCLUSION . . . . .                               | 6   |
| <b>2 Approche Méthodologique et Outils Technologiques</b> | 7   |
| I.    Introduction . . . . .                              | 8   |
| II.    Méthodologie de Conception et Analyse . . . . .    | 8   |
| II.1.    Etude des Besoins . . . . .                      | 8   |
| II.1.1.    Besoin fonctionnels . . . . .                  | 8   |
| II.1.2.    Besoins Non Fonctionnels . . . . .             | 9   |
| II.1.3.    Besoins techniques . . . . .                   | 9   |

## TABLE DES MATIÈRES

---

|          |   |           |
|----------|---|-----------|
| II.2.    | Introduction au langage de modélisation . . . . .                               | 10        |
| II.2.1.  | Définition . . . . .  | 10        |
| II.2.2.  | Avantages . . . . .   | 10        |
| II.3.    | Typologie des Diagrammes et Outils de Modélisation . . . . .                    | 11        |
| II.3.1.  | Différents types de diagrammes d'UML . . . . .                                  | 11        |
| II.3.2.  | Outils de modélisation . . . . .  | 12        |
| III.     | Modélisation avec les Diagrammes UML . . . . .                                  | 13        |
| III.1.   | Diagramme de cas d'utilisation . . . . .  | 13        |
| III.2.   | Diagramme de classe . . . . .   | 14        |
| III.3.   | Diagramme de séquence . . . . .   | 14        |
| III.3.1. | Diagramme de séquence du cas Authentification : . . . . .                       | 14        |
| III.3.2. | Le diagramme de séquence de consultation de la météo . . . . .                  | 15        |
| III.3.3. | Le diagramme de séquence du chatbot agricole . . . . .                          | 16        |
| III.3.4. | Le diagramme de séquence de détection des maladies des plantes . . . . .        | 18        |
| III.3.5. | Le diagramme de séquence de consultation de l'encyclopédie hors ligne . . . . . | 19        |
| IV.      | Cadre Technologique et Écosystème de Développement . . . . .                    | 19        |
| IV.1.    | Écosystème de Développement . . . . .   | 19        |
| IV.1.1.  | Environnement Matériel . . . . .  | 19        |
| IV.1.2.  | Environnement Logiciel . . . . .  | 20        |
| IV.2.    | Les Outils Technologiques Utilisés . . . . .                                    | 21        |
| V.       | Conclusion . . . . .  | 22        |
| <b>3</b> | <b>Réalisation</b>  | <b>23</b> |
| I.       | Introduction . . . . .  | 24        |
| II.      | Conception du Service de Prévisions Météorologique . . . . .                    | 25        |
| II.1.    | Processus ETL . . . . .   | 25        |
| II.1.1.  | Extraction des données . . . . .  | 25        |
| II.1.2.  | Traitement des données . . . . .  | 25        |
| II.1.3.  | Chargement des données . . . . .  | 26        |

## TABLE DES MATIÈRES

---

|                            |  |           |
|----------------------------|--|-----------|
| II.2.                      | Orchestration de processus ETL . . . . .         | 26        |
| II.3.                      | Planification et exécution de workflow . . . . . | 26        |
| II.4.                      | Surveillance et gestion de workflow . . . . .    | 27        |
| II.5.                      | Création du tableau de Bord . . . . .            | 28        |
| III.                       | Détection des Maladies des Plantes . . . . .     | 29        |
| III.1.                     | Collecte et Préparation des Données . . . . .    | 29        |
| III.2.                     | Les modèles de détection . . . . .               | 32        |
| III.2.1.                   | YOLO . . . . .                                   | 32        |
| III.2.2.                   | DETR . . . . .                                   | 34        |
| III.3.                     | Entraînement et Évaluation des Modèles . . . . . | 36        |
| III.3.1.                   | Processus d’Entraînement . . . . .               | 36        |
| III.3.2.                   | Métriques d’évaluation . . . . .                 | 37        |
| IV.                        | Développement du Chatbot Agricole . . . . .      | 40        |
| IV.1.                      | Technologies et Outils Utilisés . . . . .        | 41        |
| IV.1.1.                    | Le Modèle de génération de texte . . . . .       | 41        |
| IV.1.2.                    | Méthode d’interaction : API inférence . . . . .  | 42        |
| IV.2.                      | Fonctionnement du Chatbot . . . . .              | 42        |
| IV.3.                      | Encyclopédie hors ligne . . . . .                | 44        |
| V.                         | CONCLUSION . . . . .                             | 45        |
| <b>CONCLUSION GÉNÉRALE</b> |  | <b>46</b> |
| <b>WEBLIOGRAPHIE</b>       |  | <b>46</b> |

# **LISTE DES FIGURES**

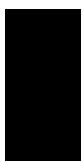
---

|      |   |    |
|------|---|----|
| 1.1  | Logo de la Fondation Djagora . . . . .  | 3  |
| 1.2  | Logo de l'application AgriCareAi . . . . .  | 4  |
| 2.1  | Types de diagrammes d'UML . . . . .   | 11 |
| 2.2  | Logo de l'outil Lucidchart . . . . .  | 13 |
| 2.3  | Diagramme de cas d'utilisation de l'application . . . . .                         | 13 |
| 2.4  | Diagramme de classe de l'application . . . . .                                    | 14 |
| 2.5  | Diagramme de Séquence du cas Authentification . . . . .                           | 15 |
| 2.6  | Diagramme de Séquence du Consultation de la météo . . . . .                       | 16 |
| 2.7  | Diagramme Séquence du Cas Utiliser Chatbot Agricole . . . . .                     | 17 |
| 2.8  | Diagramme Séquence du Cas de Détecter la maladie des plantes . . . . .            | 18 |
| 2.9  | Diagramme Séquence du Cas Consulter l'encyclopédie hors ligne . . . . .           | 19 |
| 2.10 | Logo de logiciel Android Studio . . . . .   | 20 |
| 3.1  | Interface de l'espace utilisateur . . . . .                                       | 24 |
| 3.2  | Orchestration du processus ETL . . . . .  | 26 |
| 3.3  | Tableau de bord météorologique . . . . .  | 28 |
| 3.4  | Interface de détecteur des maladies des plantes . . . . .                         | 29 |
| 3.5  | Distribution des Images par Classe pour les Ensembles de Données . . . . .        | 30 |
| 3.6  | Architecture du modèle YOLO . . . . .   | 32 |
| 3.7  | une comparaison des performances de différentes versions du modèle YOLO . . . . . | 34 |
| 3.8  | Architecture du Modèle DETR . . . . .   | 34 |
| 3.9  | Matrice de Confusion . . . . .  | 38 |
| 3.10 | Interface de ChatBot Agricole . . . . .   | 40 |
| 3.11 | Processus de fonctionnement du Chatbot Agricole . . . . .                         | 43 |
| 3.12 | Processus de fonctionnement de l'encyclopédie hors ligne . . . . .                | 44 |

# **LISTE DES TABLEAUX**

---

|     |  |    |
|-----|--|----|
| 2.1 | Présentation des outils matériels . . . . .                              | 20 |
| 2.2 | Tableau des outils technologiques utilisés . . . . .                     | 21 |
| 3.1 | Tableau descriptif des classes du dataset . . . . .                      | 31 |
| 3.2 | Tableau comparatif des caractéristiques des modèles . . . . .            | 35 |
| 3.3 | Description des paramètres d'entraînement des modèles YOLO . . . . .     | 36 |
| 3.4 | Paramètres d'entraînement du modèle DETR . . . . .                       | 37 |
| 3.5 | Comparaison des performances des modèles de détection d'objets . . . . . | 39 |



---

# LISTE DES ABRÉVIATIONS

**API** application programming interface

**CNN** convolutional neural network

**DL** Deep Learning

**DETR** DEtection TRansformer

**ETL** Extract-Transform-Load

**FPS** Frames Per Second

**GPU** Graphics Processing Unit (unité de traitement graphique)

**IA** Intelligence Artificielle

**JSON** JavaScript Object Notation

**LLaMA** Large Language Model Meta AI

**NLP** natural language processing

**OS** operating system (Système d'exploitation)

**RAM** random-access memory (mémoire vive)

**YOLO** You Only Look Once

# INTRODUCTION GÉNÉRALE

L'agriculture présente un pilier fondamental de l'économie mondiale. Elle fait face à des défis croissants dans un monde en rapide évolution. La sécurité alimentaire, la durabilité environnementale et l'efficacité de la production sont des enjeux majeurs qui exigent des solutions innovantes. Dans ce contexte, l'intégration des technologies de l'information et de l'intelligence artificielle offre des opportunités significatives pour transformer les pratiques agricoles traditionnelles et accroître la résilience du secteur face aux nouvelles contraintes.

Le projet présenté dans ce rapport se concentre sur le développement d'une application mobile intelligente, dédiée à l'accompagnement des agriculteurs dans leur travail quotidien. Cette application vise à tirer parti des technologies avancées telles que la détection des maladies des plantes via des modèles de deep learning, la prévision météorologique et l'assistance par chatbot. En fournissant des outils adaptés aux besoins des agriculteurs, AgricareAI a pour ambition de contribuer à l'amélioration des rendements agricoles, à la réduction des pertes causées par les maladies des plantes, et à l'optimisation des ressources disponibles.

L'objectif de ce projet est de démontrer comment les technologies modernes peuvent être appliquées efficacement dans le domaine de l'agriculture pour répondre aux défis actuels et futurs. En présentant les différentes composantes d'AgricareAI, ce rapport mettra en lumière le potentiel de l'intelligence artificielle à révolutionner l'agriculture et à apporter des solutions pratiques aux problèmes quotidiens des agriculteurs.

Chapitre

**1**

---

# Cadre Général du Projet

## Sommaire

---

|             |   |   |   |   |   |   |          |
|-------------|---|---|---|---|---|---|----------|
| <b>I.</b>   | <b>Introduction</b>                         | . | . | . | . | . | <b>3</b> |
| <b>II.</b>  | <b>Présentation de la Société d'Accueil</b> | . | . | . | . | . | <b>3</b> |
| II.1.       | Présentation                                | . | . | . | . | . | <b>3</b> |
| II.2.       | Secteur d'Activité                          | . | . | . | . | . | <b>3</b> |
| <b>III.</b> | <b>État de l'art</b>                        | . | . | . | . | . | <b>4</b> |
| III.1.      | Cadre du Projet                             | . | . | . | . | . | <b>4</b> |
| III.2.      | Étude de l'existant                         | . | . | . | . | . | <b>5</b> |
| III.3.      | La Problématique                            | . | . | . | . | . | <b>5</b> |
| III.4.      | Objectifs                                   | . | . | . | . | . | <b>6</b> |
| <b>IV.</b>  | <b>CONCLUSION</b>                           | . | . | . | . | . | <b>6</b> |

---

## I. Introduction

Dans ce chapitre, nous allons établir le cadre général du projet. On va commencer par introduire la fondation qui a accueilli notre stage d'été. Ensuite, on explorera l'état de l'art du projet en décrivant le cadre du projet, en effectuant une étude de l'existant, en identifiant la problématique ou les défis , et enfin en définissant les objectifs à atteindre.

## II. Présentation de la Société d'Accueil

### II.1. Présentation

Mon stage d'été est effectué au sein de **La Fondation Djagora** qui est une association professionnelle, à but non lucratif, qui regroupe les principaux acteurs des Technologies de l'Information et de l'entrepreneuriat à la Technopole de Sfax. Djagora vise à dynamiser l'industrie du numérique dans la Région.



FIGURE 1.1 – Logo de la Fondation Djagora

### II.2. Secteur d'Activité

Djagora dispose d'un FabLab en partenariat avec le groupe Orange et la société de gestion de la Technopole de Sfax. Djagora développe un programme d'accès au marché. Au cours de

ce programme, Djagora travaillera avec 6 startups, leur proposant des sessions de groupe, des sessions de pair à pair avec d'autres startups, tout en se concentrant sur des sessions individuelles avec des mentors et des experts de l'industrie pour un soutien sur mesure à chaque startup.

### III. État de l'art

#### III.1. Cadre du Projet

Le secteur agricole joue un rôle crucial dans l'économie mondiale, mais il est confronté à des défis majeurs tels que les maladies des plantes, les conditions météorologiques imprévisibles et le besoin croissant d'optimiser les ressources. Pour répondre à ces défis, l'utilisation de technologies innovantes devient essentielle.

Le projet AgricareAI s'inscrit dans ce contexte, visant à développer une application mobile destinée à aider les agriculteurs à surveiller la santé de leurs cultures, à prévoir les conditions météorologiques, et à fournir des informations pertinentes via un chatbot agricole intelligent. Ce projet exploite les avancées récentes en matière d'intelligence artificielle et d'apprentissage automatique pour améliorer la gestion agricole et contribuer à une production plus durable.



FIGURE 1.2 – Logo de l'application AgriCareAi

## III.2. Étude de l'existant

Plusieurs solutions existent actuellement pour l'assistance aux agriculteurs, allant des applications de prévisions météorologiques aux plateformes de diagnostic de maladies des plantes. Cependant, ces solutions sont souvent limitées par leur manque de personnalisation et leur complexité d'utilisation. Les applications existantes se concentrent souvent sur un seul aspect, comme la météorologie ou la détection de maladies, sans offrir une solution unifiée. De plus, peu d'entre elles intègrent un chatbot capable de répondre aux questions spécifiques des agriculteurs de manière fluide et intuitive.

AgricareAI vise à combler ces lacunes en offrant une application multifonctionnelle qui combine prévision météorologique, détection des maladies, et assistance par chatbot, le tout intégré dans une seule plateforme conviviale.

## III.3. La Problématique

Les agriculteurs font face à de nombreux défis, notamment la gestion des maladies des plantes, les variations climatiques et la nécessité d'obtenir des informations fiables et rapidement accessibles. Une réponse inadéquate à ces défis peut entraîner des pertes de rendement significatives, affectant ainsi la sécurité alimentaire et les revenus des agriculteurs. La problématique centrale à laquelle AgricareAI répond est de fournir une solution intégrée, facile d'utilisation, qui permet aux agriculteurs de surveiller la santé de leurs cultures, d'obtenir des conseils personnalisés et de planifier en fonction des conditions météorologiques, tout en étant accessible sans connexion internet.

### III.4. Objectifs

Le projet AgricareAI vise à atteindre les objectifs suivants :

- 1- **Offrir une assistance complète aux agriculteurs :** Développer une application mobile qui regroupe plusieurs services utiles aux agriculteurs, notamment la prévision météorologique, la détection des maladies des plantes, et un chatbot agricole.
- 2- **Améliorer la gestion des cultures :** Utiliser des modèles de détection de maladies basés sur l'intelligence artificielle pour permettre aux agriculteurs de diagnostiquer rapidement et précisément les problèmes affectant leurs cultures.
- 3- **Faciliter l'accès aux informations :** Intégrer une fonctionnalité d'encyclopédie hors ligne pour garantir que les utilisateurs peuvent accéder aux informations même sans connexion internet.
- 4- **Optimiser l'expérience utilisateur :** Assurer une interface conviviale et des temps de réponse rapides pour améliorer l'interaction utilisateur et promouvoir l'adoption de l'application parmi les agriculteurs.

Ces objectifs visent à améliorer la productivité agricole tout en assurant une gestion plus efficace et durable des ressources naturelles.

### IV. CONCLUSION

En conclusion, ce chapitre a permis de poser les bases du projet en détaillant son contexte général, les problématiques qu'il vise à résoudre, et les objectifs à atteindre. En explorant l'état de l'art, nous avons identifié les lacunes des solutions existantes et établi la nécessité d'une application mobile pour répondre aux besoins spécifiques des agriculteurs.

---

## **Approche Méthodologique et Outils Technologiques**

### **Sommaire**

---

|             |   |   |   |   |           |
|-------------|---|---|---|---|-----------|
| <b>I.</b>   | <b>Introduction</b>                                       | . | . | . | <b>8</b>  |
| <b>II.</b>  | <b>Méthodologie de Conception et Analyse</b>              | . | . | . | <b>8</b>  |
| II.1.       | Etude des Besoins   | . | . | . | 8         |
| II.2.       | Introduction au langage de modélisation                   | . | . | . | 10        |
| II.3.       | Typologie des Diagrammes et Outils de Modélisation        | . | . | . | 11        |
| <b>III.</b> | <b>Modélisation avec les Diagrammes UML</b>               | . | . | . | <b>13</b> |
| III.1.      | Diagramme de cas d'utilisation                            | . | . | . | 13        |
| III.2.      | Diagramme de classe                                       | . | . | . | 14        |
| III.3.      | Diagramme de séquence                                     | . | . | . | 14        |
| <b>IV.</b>  | <b>Cadre Technologique et Écosystème de Développement</b> | . | . | . | <b>19</b> |
| IV.1.       | Écosystème de Développement                               | . | . | . | 19        |
| IV.2.       | Les Outils Technologiques Utilisés                        | . | . | . | 21        |
| <b>V.</b>   | <b>Conclusion</b>   | . | . | . | <b>22</b> |

---

## I. Introduction

Ce chapitre explore l'approche méthodologique et les outils utilisés pour concevoir notre application. Il aborde les étapes de conception et d'analyse, les besoins fonctionnels et non fonctionnels, et présente les langages et outils de modélisation tels qu'UML. En mettant l'accent sur les méthodologies et les technologies modernes, ce chapitre illustre comment structurer efficacement le développement d'une application, en tenant compte à la fois des exigences utilisateurs et des contraintes techniques.

## II. Méthodologie de Conception et Analyse

### II.1. Etude des Besoins

La spécification des besoins est la première phase du cycle de vie d'une application. Le but de cette étape est de décrire l'application à développer.

#### II.1.1. Besoin fonctionnels

Dans cette partie on vise à exprimer les fonctionnalités, que le système doit fournir aux différents acteurs qui sont principalement les utilisateurs de l'application :

- **Gérer les utilisateurs** : créer un compte, se connecter, modifier les informations du compte, et se déconnecter.
- **Gérer les prévisions météo** : afficher les prévisions météo régionales et mettre à jour les données météo quotidiennement.

- **Détection des maladies des plantes** : capturer une photo, analyser l'image pour détecter la maladie, et afficher les résultats.
- **Utiliser le chatbot agricole** : poser des questions au chatbot via texte ou voix, et recevoir des réponses via l'API.
- **Consulter l'encyclopédie hors ligne** : accéder à une encyclopédie hors ligne et consulter un résumé des maladies des plantes.

### **II.1.2. Besoins Non Fonctionnels**

Ces besoins définissent les caractéristiques essentielles du système, englobant les exigences de performance, les spécifications matérielles, et les choix de conception. Pour cela on vise à fournir les caractéristiques suivantes :

- **Authentification** : l'application devra être hautement sécurisée car les informations ne devront pas être accessibles à tout le monde.
- **Ergonomie** : l'application doit offrir une interface conviviale, explicite et simple à utiliser.
- **Contrôle des champs** : l'application doit contrôler les champs de saisie pour éviter l'introduction d'informations invalides.

### **II.1.3. Besoins techniques**

- Un Smartphone Android 7 ou plus.
- Connexion Wi-Fi ou de haut débit (3G, 4G etc...).

## II.2. Introduction au langage de modélisation

### II.2.1. Définition

Pour la modélisation objet, nous avons choisi le langage commun UML (Unified Modeling Language). UML est un langage de modélisation formel et standardisé, issu de la fusion de plusieurs méthodes existantes. Il permet de représenter informatiquement un ensemble d'éléments du monde réel sous forme d'entités informatiques appelées objets. Ces objets sont décrits à travers des vues statiques et dynamiques, comprenant un ensemble de diagrammes qui collaborent pour offrir différentes perspectives d'une même représentation d'un système d'objets.

### II.2.2. Avantages

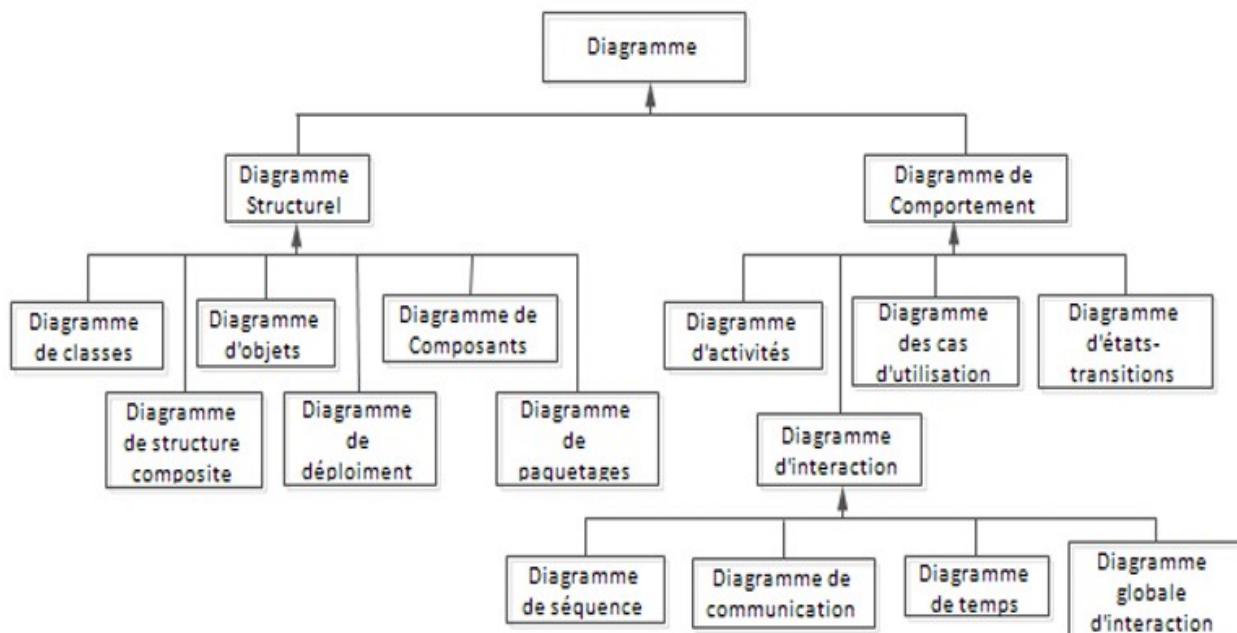
Parmi les avantages clés du langage UML qui en font un outil précieux pour la modélisation des systèmes, on cite :

- **Facilité de compréhension** : UML est un support de communication efficace qui cadre l'analyse, simplifie la compréhension du système et réduit l'ambiguïté.
- **Réduction des risques d'erreurs** : En tant que langage formel et normalisé, UML permet un gain de précision et de stabilité dans la conception.
- **Flexibilité** : UML ne prescrit aucune démarche spécifique, offrant la liberté d'utiliser les types de diagrammes souhaités dans l'ordre désiré.
- **Lisibilité** : Grâce à ses diagrammes, UML est conçu pour être compris par tous types de programmeurs, facilitant l'explication des relations dans un programme de manière simple et illustrée.

## II.3. Typologie des Diagrammes et Outils de Modélisation

### II.3.1. Différents types de diagrammes d'UML

UML comprend désormais 13 diagrammes différents, dont quatre ont été ajoutés avec UML 2.0, chacun représentant un aspect spécifique d'un système logiciel. UML modélise le système de deux façons : la structure statique du système "au repos" et sa dynamique en fonctionnement. Ces deux représentations sont complémentaires et essentielles pour illustrer la composition du système et l'interaction de ses composants. La figure suivante présente les différents types de diagrammes UML.



**FIGURE 2.1 – Types de diagrammes d'UML**

Dans notre projet on a recours seulement aux diagrammes suivants :

- **Diagramme de classes** : Un schéma UML qui décrit la structure statique d'un système en représentant ses classes, leurs attributs, méthodes, et les relations entre elles (héritage,

association, etc.). Il sert à modéliser l'architecture logicielle et les interactions entre les composants du système.

- **Diagramme de séquence** : Un schéma UML qui montre l'ordre des interactions entre les objets d'un système au fil du temps. Il représente le flux de messages échangés pour accomplir une tâche spécifique, illustrant comment les objets collaborent dynamiquement.
- **Diagramme de cas d'utilisation** : Un outil de modélisation UML qui décrit les interactions entre les utilisateurs (ou acteurs) et un système. Il représente les différents scénarios d'utilisation (ou cas d'utilisation) que les utilisateurs peuvent effectuer avec le système, en montrant les fonctionnalités offertes par ce dernier et la manière dont les utilisateurs les utilisent.

### II.3.2. Outils de modélisation

Les outils de modélisation UML sont essentiels pour concevoir et documenter des systèmes logiciels. Ils permettent de visualiser et de communiquer la structure et le comportement d'un système à l'aide de diagrammes standardisés, tout en facilitant la gestion des projets et la collaboration.

Parmi ces outils, **Lucidchart** se distingue particulièrement par sa simplicité d'utilisation et sa flexibilité. Lucidchart est une plateforme en ligne, basée sur le cloud, développé par **Lucid Software Inc.** et il a été publié pour la 1ère fois le 1er décembre 2008. Elle permet de créer facilement des diagrammes UML, la visualisation de données, et autres schémas conceptuels grâce à une interface conviviale et à des modèles prédéfinis.



FIGURE 2.2 – Logo de l'outil Lucidchart

### III. Modélisation avec les Diagrammes UML

Il est essentiel de comprendre comment UML permet de représenter un système complexe à travers divers diagrammes. Chaque type de diagramme UML a un rôle spécifique dans la modélisation des aspects dynamiques et statiques du système.

#### III.1. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation ci-dessus illustre les interactions possibles entre un utilisateur de l'application AgricareAI et les fonctionnalités principales du système.

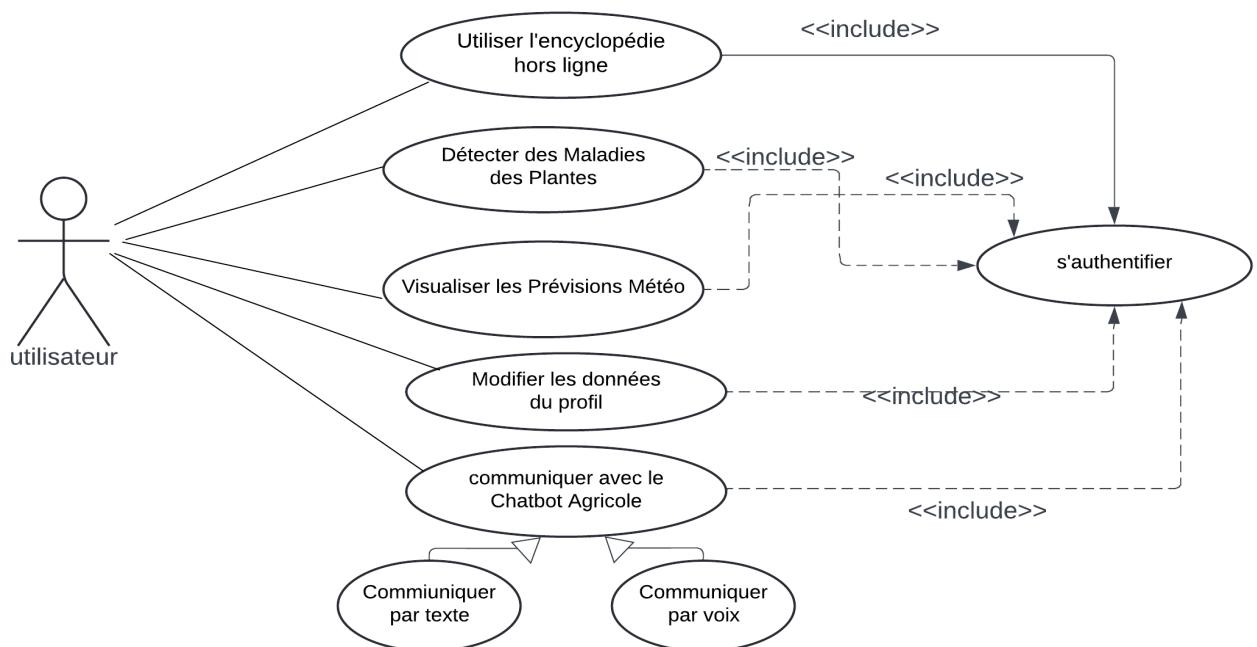
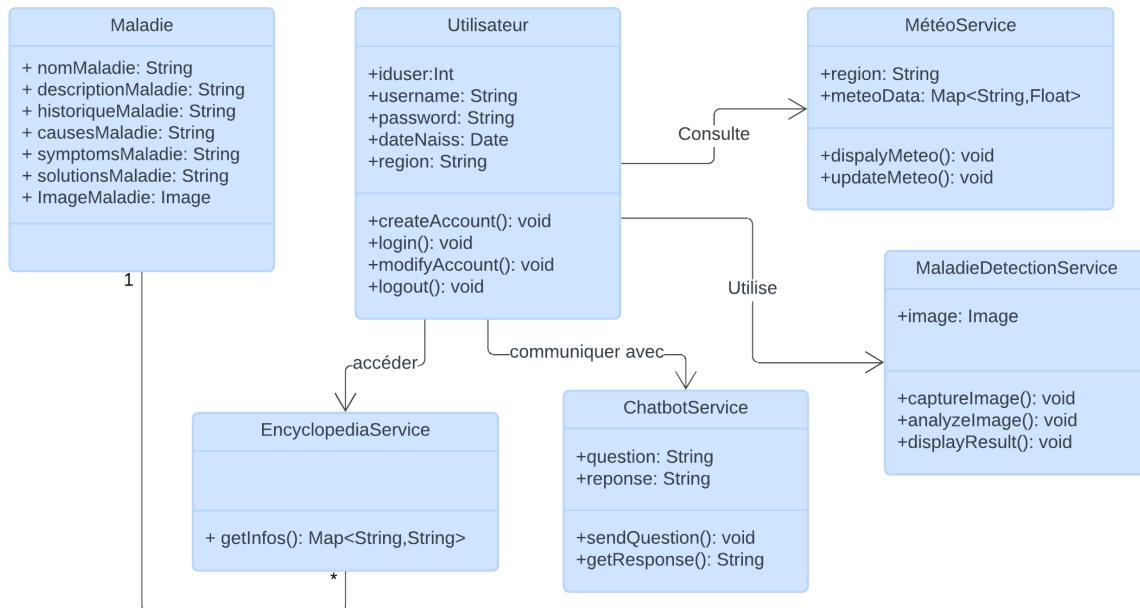


FIGURE 2.3 – Diagramme de cas d'utilisation de l'application

## III.2. Diagramme de classe

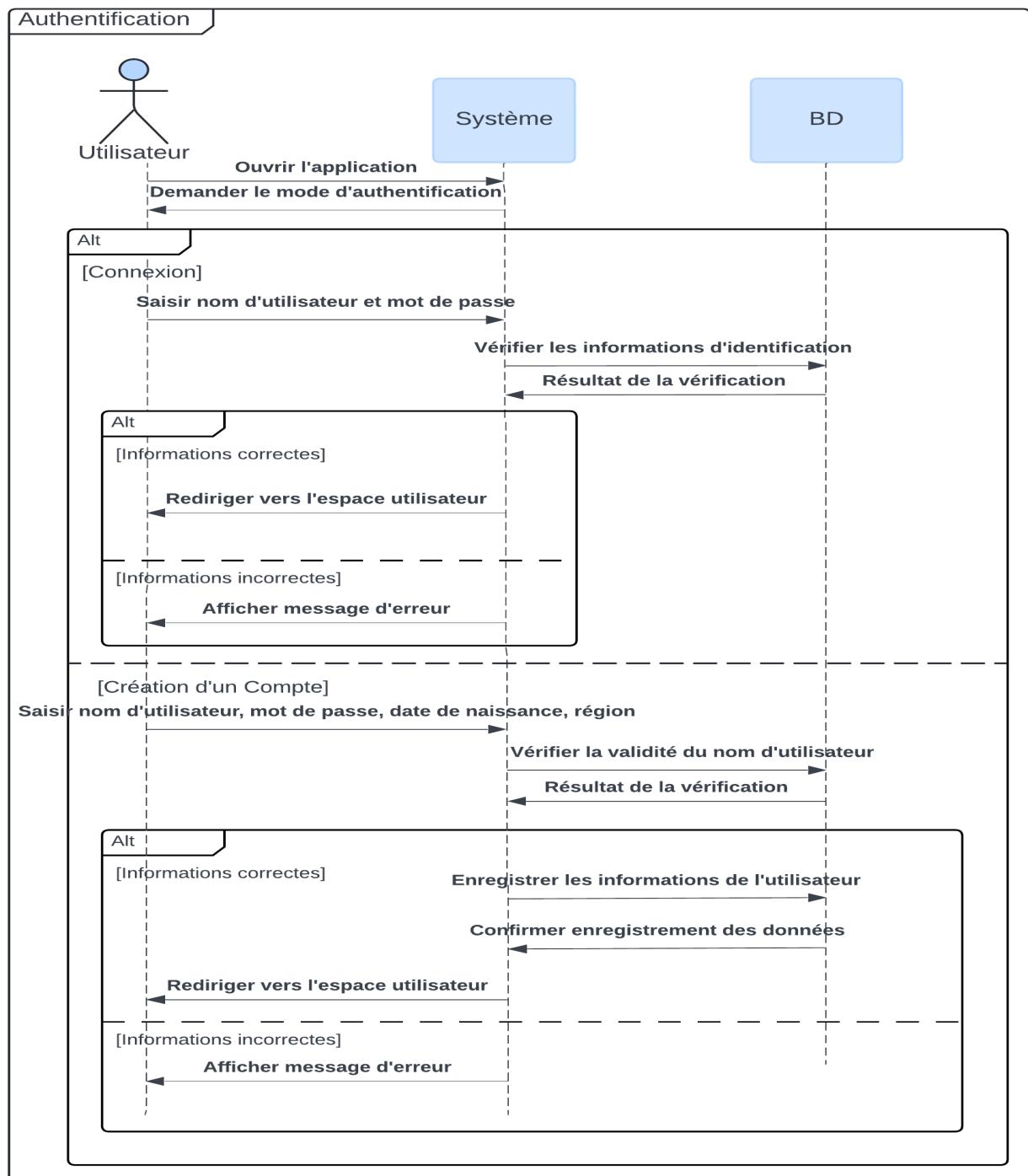


**FIGURE 2.4 – Diagramme de classe de l’application**

## III.3. Diagramme de séquence

### III.3.1. Diagramme de séquence du cas Authentification :

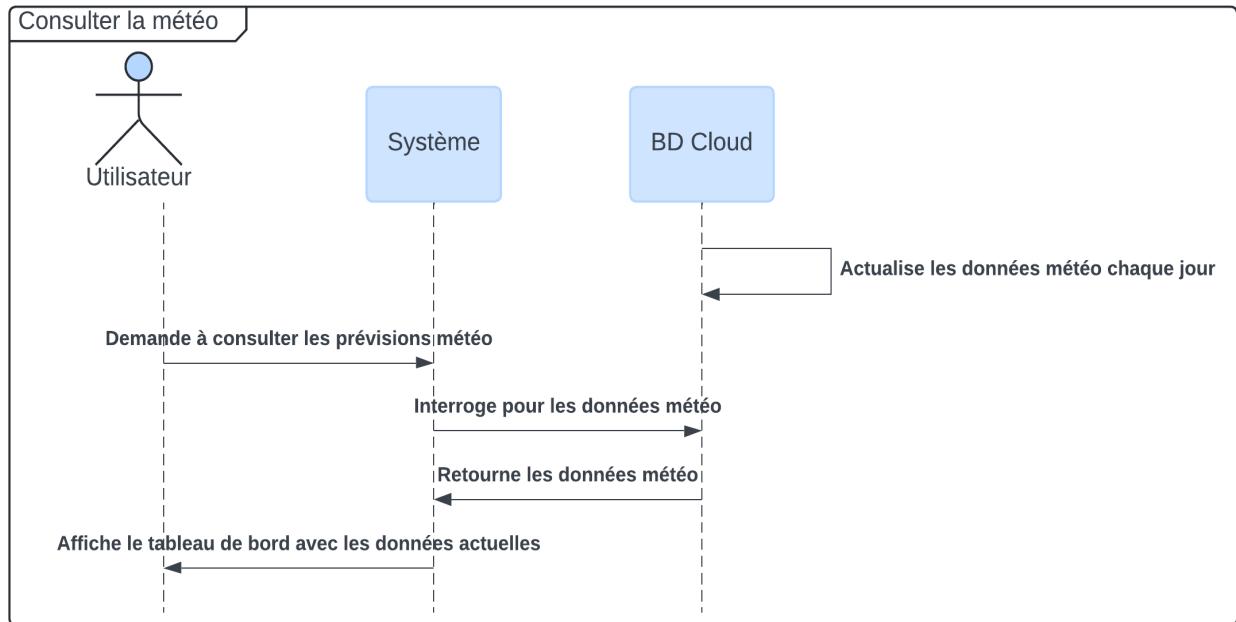
Le diagramme montre le processus par lequel un utilisateur interagit avec le système pour accéder à son espace personnalisé. Il montre les étapes clés, telles que la saisie des informations d'identification, la vérification des données par le système, et la redirection vers l'espace utilisateur.



**FIGURE 2.5 – Diagramme de Séquence du cas Authentification**

### III.3.2. Le diagramme de séquence de consultation de la météo :

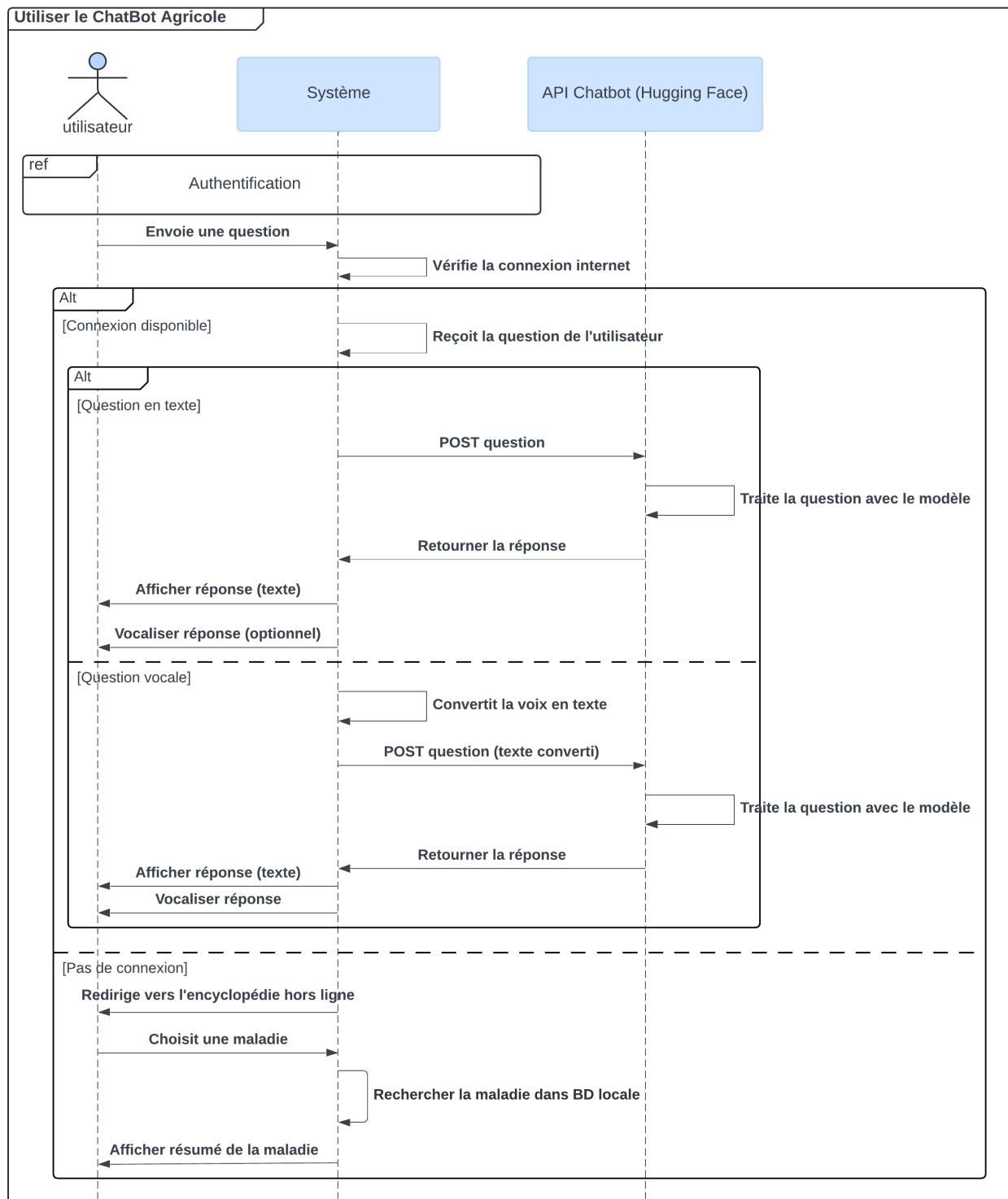
Il montre comment l'utilisateur interagit avec le système pour obtenir les prévisions météorologiques de sa région.



**FIGURE 2.6 – Diagramme de Séquence du Consultation de la météo**

### III.3.3. Le diagramme de séquence du chatbot agricole

Le diagramme montre comment l'utilisateur pose une question, et comment le système, via l'API du modèle, traite la requête et retourne la réponse soit textuellement ou vocalement. Cela permet une interaction rapide et efficace entre l'utilisateur et le chatbot.



**FIGURE 2.7 – Diagramme Séquence du Cas Utiliser Chatbot Agricole**

### III.3.4. Le diagramme de séquence de détection des maladies des plantes

Le diagramme montre comment l'utilisateur capture une image de la feuille, comment le système l'analyse à l'aide du modèle, et retourne ensuite le résultat de la détection.

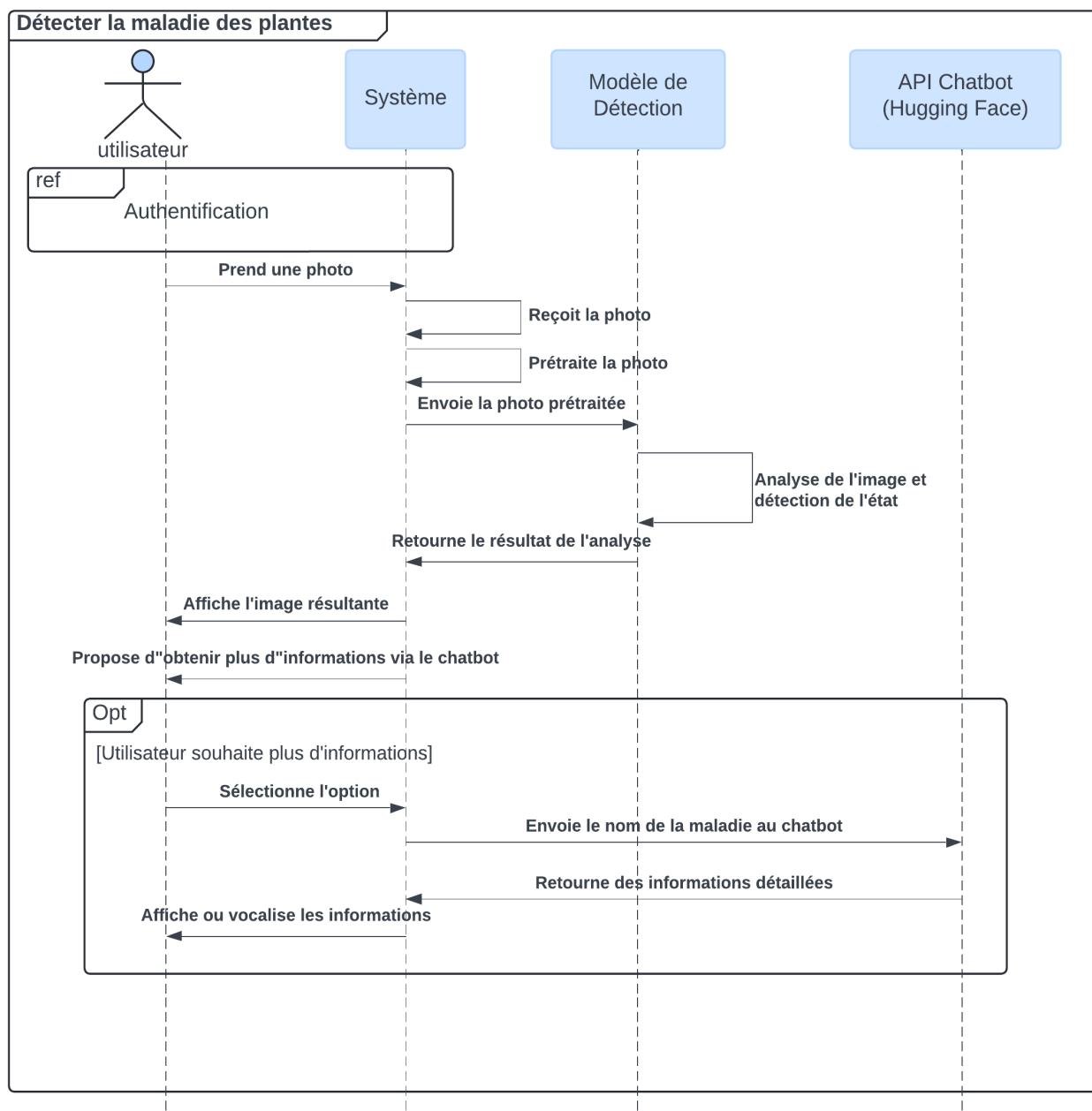


FIGURE 2.8 – Diagramme Séquence du Cas de DéTECTER la maladie des plantes

### III.3.5. Le diagramme de séquence de consultation de l'encyclopédie hors ligne

Le diagramme montre comment l'utilisateur peut chercher des informations brève à propos une maladie donnée (nom, image, description, historique, Symptômes, causes, solutions) s'il n'a pas accès au internet.

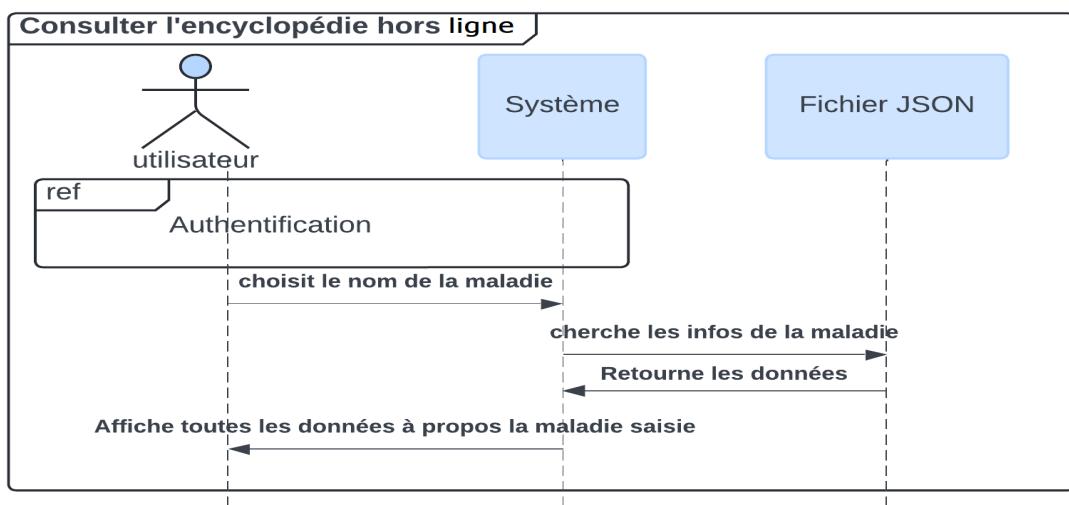


FIGURE 2.9 – Diagramme Séquence du Cas Consulter l'encyclopédie hors ligne

## IV. Cadre Technologique et Écosystème de Développement

### IV.1. Écosystème de Développement

#### IV.1.1. Environnement Matériel

L'application a comme un environnement matériel les composantes suivantes :

| Caractéristiques  | PC             | Smartphone |
|-------------------|----------------|------------|
| <b>Marque</b>     | MSI            | Realme C51 |
| <b>RAM</b>        | 16 Go          | 8 Go       |
| <b>Processeur</b> | Intel® Core i5 | Octa-Core  |
| <b>Disque dur</b> | 512 Go         | 32 Go      |
| <b>O.S</b>        | Windows 10     | Android 13 |

**TABLE 2.1 – Présentation des outils matériels**

#### IV.1.2. Environnement Logiciel

Android est un système d’exploitation mobile open source fondé sur le noyau Linux et développé par un consortium d’entreprises, le Open Handset Alliance, sponsorisé par Google. Le choix d’Android comme plateforme de développement s’explique par sa popularité mondiale, offrant une large base d’utilisateurs et une flexibilité accrue grâce à son code source ouvert.

##### a) Android Studio

Android Studio est un environnement de développement intégré (IDE) pour développer des applications mobiles Android. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il peut être téléchargé sous les systèmes d’exploitation Windows, macOS, Chrome OS et Linux.



**FIGURE 2.10 – Logo de logiciel Android Studio**

## IV.2. Les Outils Technologiques Utilisés

Les outils technologiques, tels que les plateformes en ligne, les logiciels, et les services cloud, sont présentés dans ce tableau :

| Site web   | Description  |
|--|--|
|  Apache Airflow | Apache Airflow est une plateforme open source pour la gestion des workflows et des pipelines de données. Développée à l'origine par Airbnb, Airflow permet de programmer, planifier et surveiller des tâches complexes dans des pipelines de données.[1]                           |
|  Power BI       | Power BI est une solution d'analyse de données de Microsoft. Il permet de créer des visualisations de données personnalisées et interactives avec une interface suffisamment simple pour que les utilisateurs finaux créent leurs propres rapports et tableaux de bord. [2]        |
|  roboflow      | Roboflow est une plateforme de vision par ordinateur qui permet aux utilisateurs de créer des modèles de vision par ordinateur plus rapidement et avec plus de précision grâce à de meilleures techniques de collecte de données, de prétraitement et de formation de modèles. [3] |
|  Hugging Face | Hugging Face est une plateforme et une communauté d'apprentissage automatique (ML) et de science des données qui aide les utilisateurs à créer, déployer et former des modèles d'apprentissage automatique.  |
|  Azure SQL    | Azure SQL est une famille de produits gérés, sécurisés et intelligents qui utilisent le moteur de base de données SQL Server dans le cloud Azure.[4]   |
|  W&B          | Weights & Biases (wandb) est une plateforme de métro-apprentissage machine qui permet de construire des modèles pour des applications concrètes. La plateforme permet de suivre, comparer, décrire et reproduire les expériences d'apprentissage machine.[5]                       |

TABLE 2.2 – Tableau des outils technologiques utilisés

## **V. Conclusion**

En conclusion, ce chapitre a mis en lumière l'approche méthodologique et les outils technologiques adoptés pour le développement de notre application. En détaillant les étapes de conception et les différents diagrammes UML, nous avons illustré comment une méthodologie bien structurée et des outils adaptés permettent de répondre efficacement aux exigences fonctionnelles et non fonctionnelles. L'utilisation de technologies modernes et de plateformes de développement robustes garantit une solution performante, sécurisée et facile à maintenir.

---

## Réalisation

### Sommaire

---

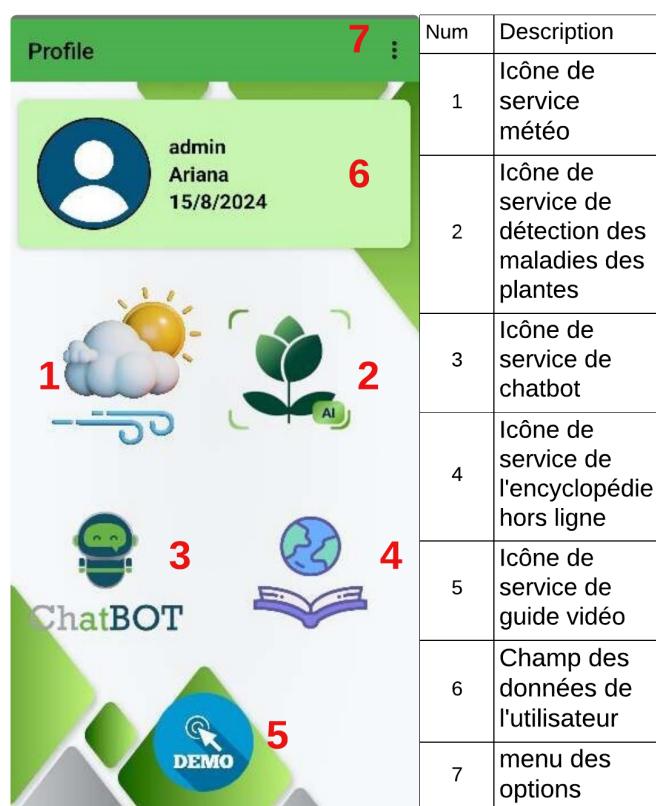
|             |   |           |
|-------------|---|-----------|
| <b>I.</b>   | <b>Introduction</b>                                       | <b>24</b> |
| <b>II.</b>  | <b>Conception du Service de Prévisions Météorologique</b> | <b>25</b> |
| II.1.       | Processus ETL   | 25        |
| II.2.       | Orchestration de processus ETL                            | 26        |
| II.3.       | Planification et exécution de workflow                    | 26        |
| II.4.       | Surveillance et gestion de workflow                       | 27        |
| II.5.       | Création du tableau de Bord                               | 28        |
| <b>III.</b> | <b>Détection des Maladies des Plantes</b>                 | <b>29</b> |
| III.1.      | Collecte et Préparation des Données                       | 29        |
| III.2.      | Les modèles de détection                                  | 32        |
| III.3.      | Entraînement et Évaluation des Modèles                    | 36        |
| <b>IV.</b>  | <b>Développement du Chatbot Agricole</b>                  | <b>40</b> |
| IV.1.       | Technologies et Outils Utilisés                           | 41        |
| IV.2.       | Fonctionnement du Chatbot                                 | 42        |
| IV.3.       | Encyclopédie hors ligne                                   | 44        |
| <b>V.</b>   | <b>CONCLUSION</b>   | <b>45</b> |

---

## I. Introduction

Dans ce chapitre, on détaillera le processus de réalisation du projet en expliquant l'architecture et le fonctionnement de chaque service de notre application mobile. On commencera par le service de prévision météorologique locale en temps réel. Ensuite, on présentera le service de détection des maladies des plantes à partir de leurs feuilles. On continuera avec le service chatbot agricole, ainsi que d'autres services supplémentaires.

Tous ces services sont intégrés dans l'interface utilisateur suivante :



**FIGURE 3.1 – Interface de l'espace utilisateur**

## II. Conception du Service de Prévisions Météorologique

### II.1. Processus ETL

#### II.1.1. Extraction des données

Nous avons opté pour l'extraction des données depuis le site web **AccuWeather** en raison de la richesse et du niveau de détail qu'il offre sur des paramètres tels que la température, la vitesse du vent, la qualité de l'air, les précipitations, etc. Cela permet aux agriculteurs d'obtenir une vue claire et précise des conditions météorologiques quotidiennes.

Nous avons adopté des techniques de web scraping pour extraire les données en utilisant la bibliothèque Python **Selenium**. En effet, grâce aux fonctionnalités offertes par cette bibliothèque, nous avons développé un script automatique pour naviguer entre les pages web et extraire les données que nous avions définies.

#### II.1.2. Traitement des données

Après l'extraction, les données sont transformées pour les préparer à être intégrées dans la base de données. Les transformations peuvent inclure le nettoyage des données, la conversion des formats, la combinaison de différentes sources, la normalisation et l'enrichissement des données, ainsi que la création de nouvelles valeurs dérivées ci dessous quelques traitements .

Dans notre cas, nous avons effectué quelques opérations de nettoyage pour stocker uniquement les données dont nous avions besoin dans la base de données.

### II.1.3. Chargement des données

Le chargement de données est l'étape cruciale du processus ETL où les données extraites et transformées trouvent leur destination finale pour être exploitées. Dans cette phase, les données sont acheminées vers la base de données Nous avons choisi une base de données hébergée dans le cloud pour garantir sa disponibilité depuis n'importe quel endroit, à tout moment.

## II.2. Orchestration de processus ETL

L'orchestration de notre processus ETL est réalisée à l'aide de l'outil Apache Airflow. Airflow offre une plateforme puissante pour planifier, exécuter et surveiller nos workflows d'extraction de données, de traitement et de stockage.

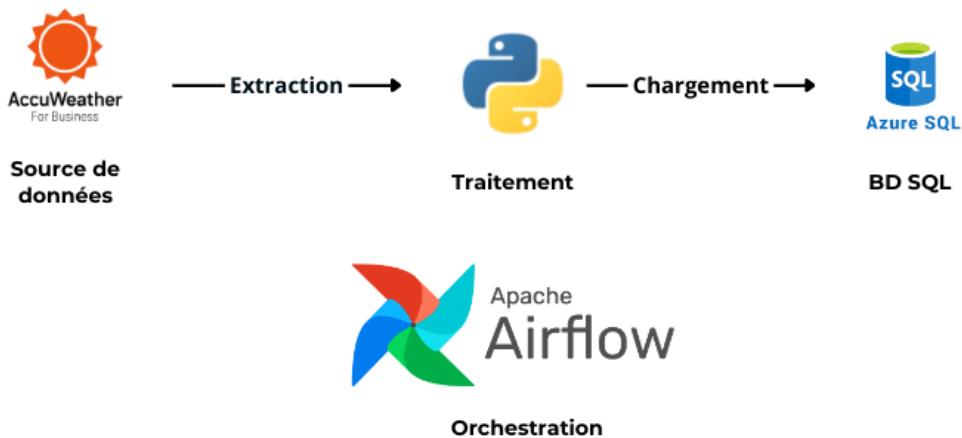


FIGURE 3.2 – Orchestration du processus ETL

## II.3. Planification et exécution de workflow

Airflow facilite la planification et l'exécution des workflows ETL en utilisant des DAGs. Dans notre cas, nous avons créé un DAG personnalisé pour notre processus ETL, définissant des tâches suivantes :

- **Extraction des données (Tâche 1) :** Cette tâche utilise des techniques de web scraping pour récupérer des données météorologiques.
- **Traitements des données scrapées (Tâches 2 à 5) :** Cette étape est responsable du traitement des données scrapées, elle est composée de trois tâches.
- **Suppression des données passées (Tâches 6 à 8) :** Cette tâche consiste à supprimer les données météorologiques correspondant aux heures ou dates passées.
- **Chargement des données actuelles (Tâche 9) :** Cette tâche alimente la base de données par des données météorologiques actuelles.
- **Envoi d'un courriel (Tâches 10) :** Cette tâche implique l'envoi d'un courriel pour confirmer que toutes les tâches sont exécutées.

## II.4. Surveillance et gestion de workflow

Une fois le DAG créé et planifié, Apache Airflow offre des fonctionnalités avancées de surveillance et de gestion de workflow, telles que la surveillance en temps réel de l'exécution des tâches, la visualisation des dépendances entre les tâches, la gestion des erreurs et des reprises, ainsi que la possibilité d'ajuster dynamiquement le calendrier d'exécution selon les besoins opérationnels. Dans notre cas nous avons mis en place des paramètres suivants :

- **Gestion des Erreurs et Reprises :**

En cas d'erreur survenue lors de l'exécution d'une tâche, notre configuration dans Apache Airflow prévoit **un délai de reprise de 3 minutes** avant de relancer automatiquement la tâche. Cette approche permet de gérer efficacement les erreurs mineures et de garantir une exécution fluide du flux de travail global.

## RÉALISATION

En cas de nouvel échec de la tâche après sa relance, **un courriel est envoyé** pour signaler l'erreur rencontrée.

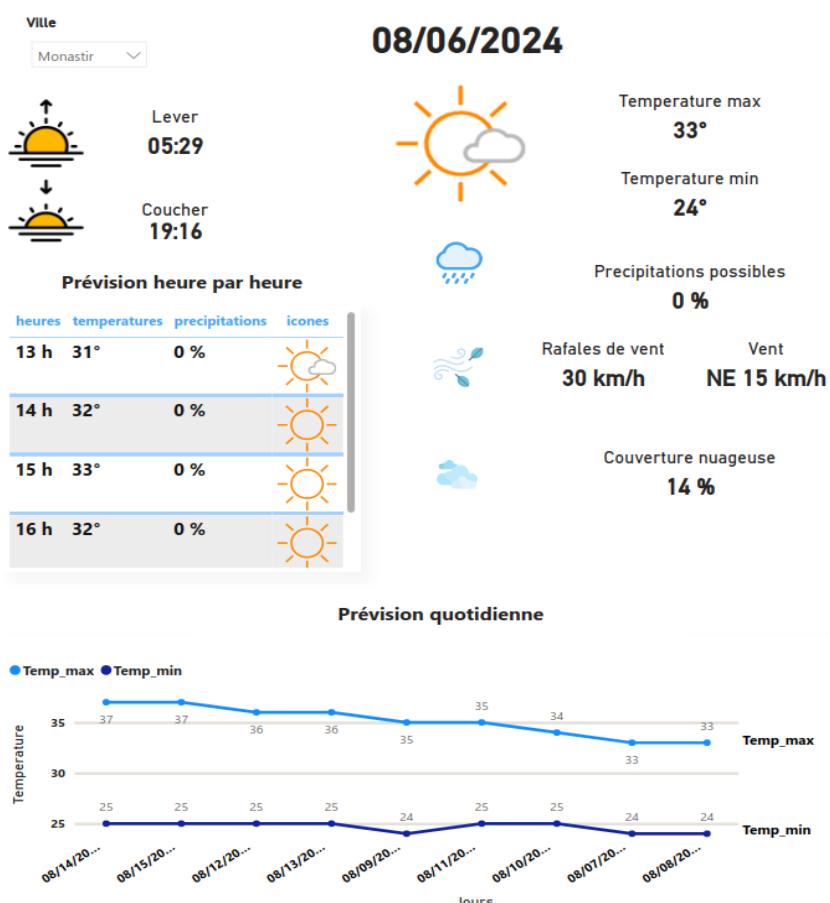
### - Calendrier d'Exécution Planifié

Le DAG est planifié pour être exécuté quotidiennement à partir de la date du **06/20/2024**.

La fréquence d'exécution du DAG a été ajustée pour qu'il s'exécute **tous les 3 heures**.

## II.5. Création du tableau de Bord

Nous avons opté pour l'outil Microsoft Power BI pour la création de tableaux de bord en raison de ses fonctionnalités avancées.



**FIGURE 3.3 – Tableau de bord météorologique**

### III. Détection des Maladies des Plantes

Le service de détection des maladies des plantes vise à fournir aux agriculteurs un outil puissant et accessible pour identifier rapidement les maladies affectant leurs cultures en s'appuyant sur des modèles de deep learning avancés de détection permettant une analyse précise des images de feuilles de plantes, ce qui aboutit à la création de l'interface suivante :



FIGURE 3.4 – Interface de détecteur des maladies des plantes

#### III.1. Collecte et Préparation des Données

Le dataset intitulé "*Tomato and Strawberry*", disponible sur Roboflow, comprend un ensemble d'images annotées représentant les feuilles de plantes telles que les fraises et les tomates. Ces images ont été étiquetées dans le but de détecter et de classifier diverses anomalies affectant les feuilles de ces espèces. Le dataset est fourni au format YOLOv8, avec chaque image accompagnée

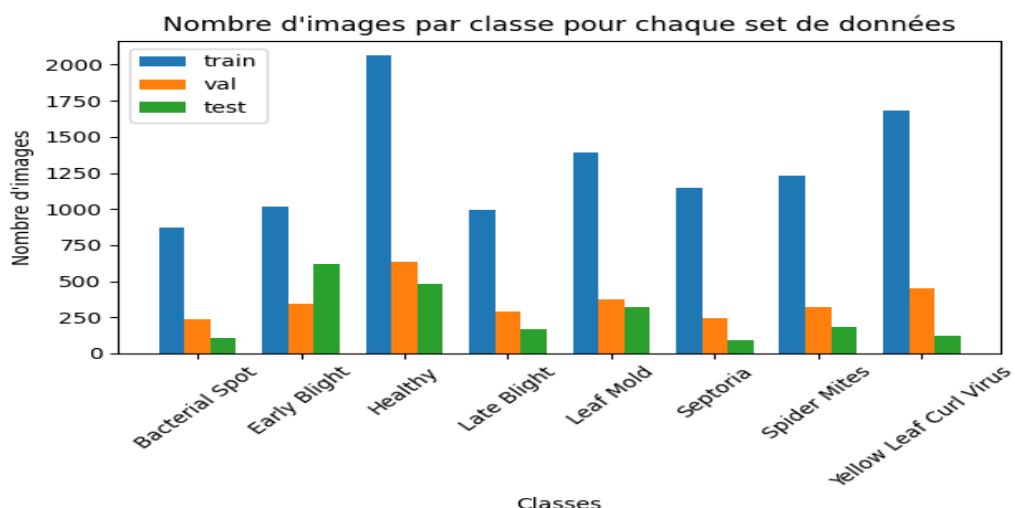
## RÉALISATION

---

d'un fichier texte qui contient les annotations sous forme de boîtes englobantes, délimitant les zones touchées sur les feuilles. Les annotations spécifient :

- Le label de la classe correspondant à l'anomalie détectée.
- Les coordonnées normalisées de la boîte englobante (centre x, centre y, largeur, hauteur).

Ce dataset comprend un total de 7 626 images au format JPEG, chacune ayant une résolution de 640x640 pixels. Les images sont réparties en trois sous-ensembles : 5 340 images (70%) pour l'entraînement, 1 524 images (20%) pour la validation, et 762 images (10%) pour le test.



**FIGURE 3.5 – Distribution des Images par Classe pour les Ensembles de Données**

Le dataset comprend 8 classes correspondant à différentes anomalies affectant les feuilles :

Ce dataset est conçu pour développer et tester des modèles de vision par ordinateur dans le domaine de l'agriculture, en particulier pour la détection et la classification des maladies des plantes.

## RÉALISATION

---

| Maladie                | Image | Description   |
|------------------------|-------|---|
| Bacterial Spot         |       | Une infection bactérienne qui provoque des taches sombres et angulaires sur les feuilles des plantes, affectant principalement les tomates et les poivrons.   |
| Early Blight           |       | Une maladie fongique causée par <i>Alternaria solani</i> , se manifestant par des taches brunes sur les feuilles, souvent avec des cercles concentriques, et affecte principalement les tomates.      |
| Healthy                |       | Représente les feuilles de plantes saines. [2]  |
| Late Blight            |       | Une maladie destructrice causée par le champignon <i>Phytophthora infestans</i> , caractérisée par des lésions aqueuses qui s'étendent rapidement sur les feuilles, les tiges et les fruits.          |
| Leaf Mold              |       | Une maladie fongique qui apparaît sous forme de taches jaunes sur le dessus des feuilles et de moisissure grise ou brune sur le dessous, affectant principalement les tomates cultivées sous serre.   |
| Septoria               |       | Causée par le champignon <i>Septoria lycopersici</i> , cette maladie se manifeste par des petites taches grises ou brunes entourées d'un halo jaunâtre sur les feuilles, conduisant à la défoliation. |
| Spider Mites           |       | De minuscules acariens qui se nourrissent des cellules des feuilles, provoquant des taches jaunâtres ou blanchâtres, et peuvent rapidement dévaster les cultures.                                     |
| Yellow Leaf Curl Virus |       | Un virus transmis par des aleurodes, provoquant un enroulement des feuilles vers le haut, un jaunissement et une croissance réduite, particulièrement dévastateur pour les tomates.                   |

TABLE 3.1 – Tableau descriptif des classes du dataset

## III.2. Les modèles de détection

### III.2.1. YOLO

YOLO (You Only Look Once), est un algorithme de détection d'objets dans des images ou des vidéos. Il a été introduit pour la première fois en 2016 par Joseph Redmon et Santosh Divvala dans leur article intitulé "You Only Look Once : Unified, Real-Time Object Detection". Depuis sa première version, YOLO a connu plusieurs versions améliorées, chacune apportant des améliorations en termes de précision, de vitesse et de capacité à détecter une plus grande variété d'objets.

YOLO a une architecture CNN, Comme illustré ci-dessous, comportant au total 24 couches convolutionnelles, quatre couches de pooling maximal et deux couches entièrement connectées.[6]

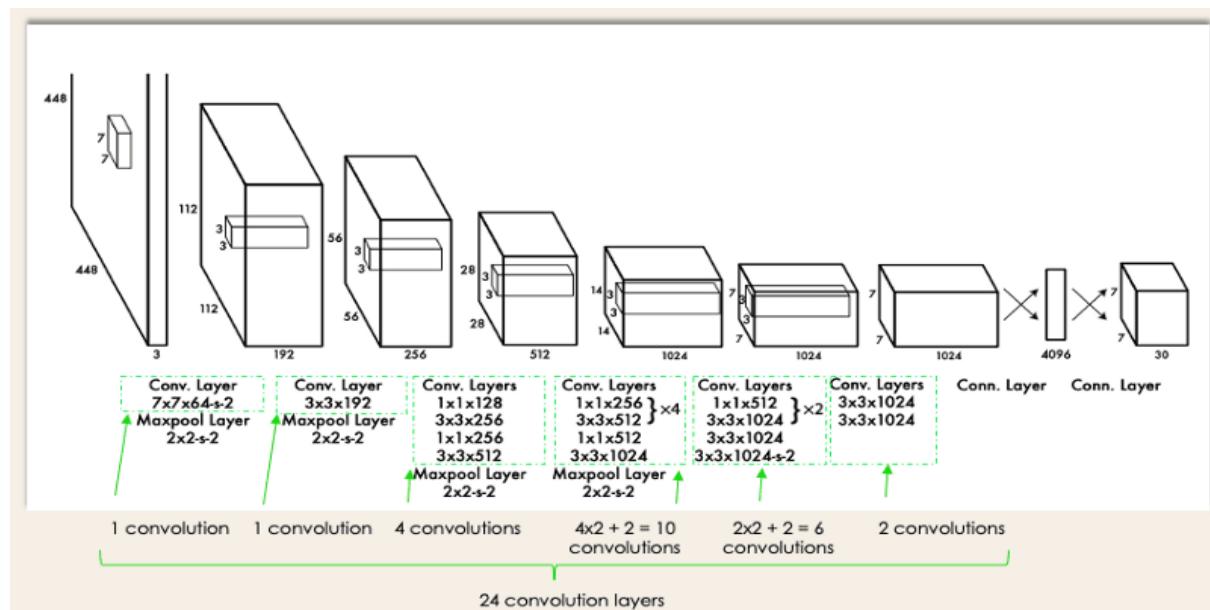


FIGURE 3.6 – Architecture du modèle YOLO

YOLO fonctionne de la manière suivante :

- **Division en grille :** Il divise l'image d'entrée en une grille de taille  $S \times S$ . Chaque cellule de la grille est responsable de la détection des objets dont le centre se trouve à l'intérieur de cette cellule.
- **Prédiction des boîtes englobantes :** Pour chaque cellule de la grille, YOLO prédit un nombre fixe de boîtes englobantes ainsi que les scores de confiance, qui indiquent la probabilité qu'une boîte contienne un objet et la précision de la boîte prédite.
- **Prédiction de classes :** Chaque boîte englobante prédit également les probabilités des classes, indiquant la probabilité que l'objet détecté appartienne à une certaine classe (par exemple, personne, voiture).
- **Détection finale :** YOLO multiplie le score de confiance par la probabilité de classe pour obtenir le score final. Seules les prédictions avec des scores élevés sont conservées, et les boîtes se chevauchant sont filtrées à l'aide de la suppression non maximale (NMS).

Pour mieux comprendre les évolutions de YOLO, On a choisi de se pencher sur les dernières versions les plus stables de cet algorithme qui sont :

- **YOLOv7** qui se caractérise par des Améliorations significatives en termes de rapidité et de précision par rapport aux versions précédentes avec l'Utilisation d'ancres optimisées pour la détection des petits objets dans les images.
- **YOLOv8n** qui se caractérise par :Réduction du nombre de paramètres pour une meilleure efficacité sans sacrifier la précision, Amélioration de la vitesse de traitement et la Capacité à gérer plusieurs tâches comme la segmentation, la classification et la détection de poses<sup>1</sup>.

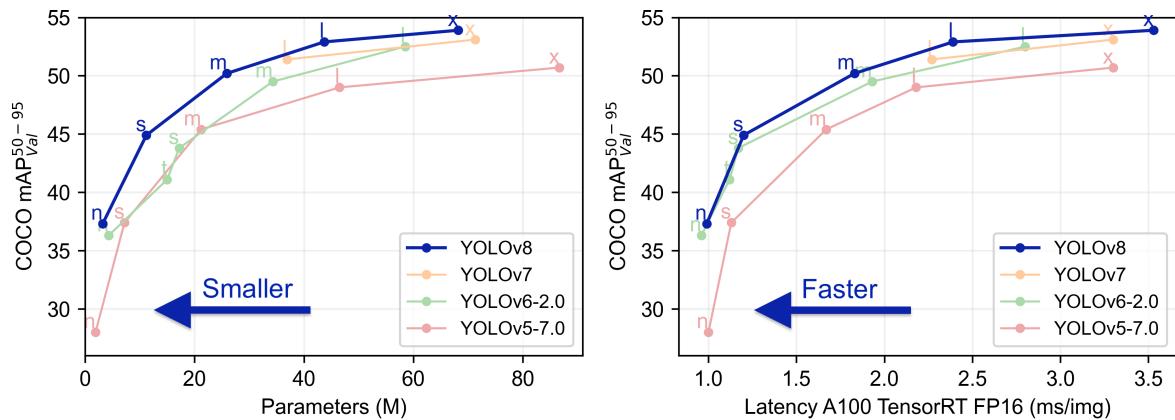


FIGURE 3.7 – une comparaison des performances de différentes versions du modèle YOLO

### III.2.2. DETR

Le DETR (DEtection TRansformer) est un algorithme de détection d'objets relativement récent, introduit en 2020 par des chercheurs de Facebook AI Research (FAIR). Il est basé sur l'architecture des transformateurs, un modèle puissant utilisé pour diverses tâches de traitement du langage naturel. DETR est un modèle de prédiction directe qui utilise une architecture encodeur-décodeur pour prédire tous les objets simultanément.

L'architecture de DETR est simple et se compose des composants principaux comme dans le schéma suivant :

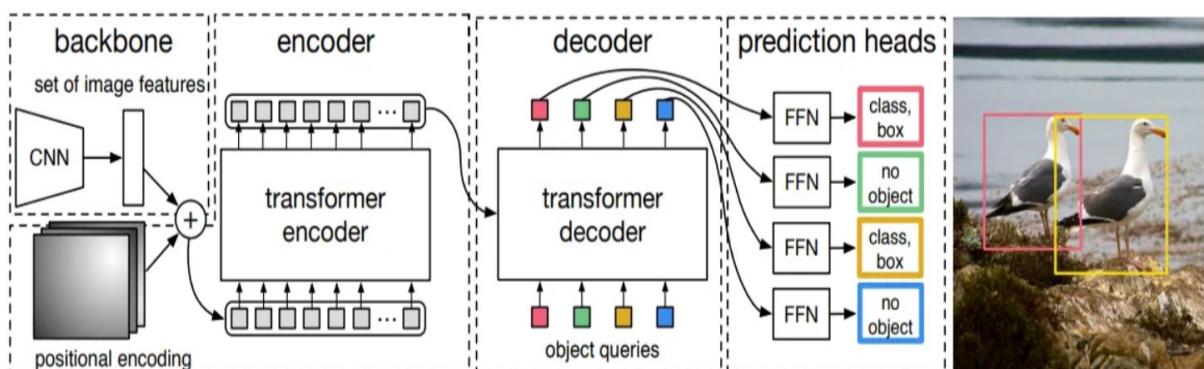


FIGURE 3.8 – Architecture du Modèle DETR

Le modèle DEtection TRansformer (DETR) fonctionne de la manière suivante :

- **Encodage de l'image :** L'image d'entrée est d'abord traitée par un réseau de convolution (CNN) pour extraire des caractéristiques visuelles en une séquence de vecteurs de caractéristiques.
- **Transformateur :** Un transformateur est utilisé pour modéliser les relations spatiales et contextuelles des objets dans l'image. Il utilise des mécanismes d'attention pour capturer les dépendances entre différentes parties de l'image, permettant ainsi de comprendre le contexte global.
- **Détection d'objets :** Au lieu de prédire des boîtes englobantes comme dans les modèles traditionnels, DETR prédit directement les classes des objets et les coordonnées des boîtes englobantes à l'aide de prédicteurs d'objets fixés. Chaque prédicteur est responsable de détecter un seul objet dans l'image.
- **Suppression des doublons :** DETR utilise un mécanisme de suppression non maximale (NMS) implicite via le transformateur pour éviter les prédictions en double.

Pour résumer les caractéristiques des trois modèles choisis, nous avons élaboré le tableau suivant :

| Caractéristiques                      | YOLOv7                | YOLOv8n               | DETR                              |
|---------------------------------------|-----------------------|-----------------------|-----------------------------------|
| Architecture                          | CNN                   | CNN                   | Transformer                       |
| Précision                             | Élevée                | Très élevée           | Moyenne                           |
| Vitesse (FPS)                         | Très rapide           | Rapide                | Moins rapide                      |
| Complexité de l'Entraînement          | Modérée               | Modérée               | Élevée                            |
| Capacité de Généralisation            | Bonne                 | Excellent             | Bonne                             |
| Gestion des Petits Objets             | Bonne                 | Très bonne            | Moyenne                           |
| Robustesse aux Variations d'Éclairage | Bonne                 | Très bonne            | Excellent                         |
| Capacité de Détection Multi-échelle   | Très bonne            | Excellent             | Bonne                             |
| Facilité d'Implémentation             | Facile                | Facile                | Modérée                           |
| Type de Modèle                        | Détecteur à une étape | Détecteur à une étape | Détecteur à deux étapes           |
| Profondeur du Réseau                  | 50-100 couches        | 60-120 couches        | Variable (backbone + transformer) |
| Nombre de Paramètres                  | 36.9 M                | 3.2 M                 | 41 M                              |

TABLE 3.2 – Tableau comparatif des caractéristiques des modèles

### III.3. Entraînement et Évaluation des Modèles

L’entraînement des modèles de détection d’objets commence par la collecte et l’annotation d’un grand ensemble de données d’images. Ensuite, un modèle est entraîné pour reconnaître ces objets en ajustant ses paramètres internes afin de minimiser l’erreur entre les prédictions du modèle et les annotations réelles.

#### III.3.1. Processus d’Entraînement

Pour effectuer l’entraînement des modèles YOLO, on doit configurer les hyperparamètres suivants :

| Paramètre | Description  |
|-----------|--|
| translate | Assure l’augmentation de données qui contrôle le décalage aléatoire des images.  |
| scale     | Ajuste la mise à l’échelle des images comme forme d’augmentation de données.   |
| erasing   | Fait référence à l’effacement aléatoire (random erasing), une technique d’augmentation où une partie de l’image est masquée aléatoirement. |
| epochs    | Le nombre d’époques pour lesquelles le modèle sera entraîné.   |
| imgsz     | Taille des images d’entrée utilisée pendant l’entraînement. Les images seront redimensionnées à une taille de 640x640 pixels par défaut.   |
| batch     | La taille du lot (batch size) détermine le nombre d’images traitées ensemble en une seule fois pendant l’entraînement.                     |

**TABLE 3.3 – Description des paramètres d’entraînement des modèles YOLO**

Pour effectuer l’entraînement des modèles DETR, on doit configurer les paramètres suivants :

| Paramètre                   | Définition  |
|-----------------------------|---|
| num_train_epochs            | Nombre total d'époques d'entraînement                                     |
| learning_rate               | Taux d'apprentissage pour ajuster les poids                               |
| per_device_train_batch_size | Taille du lot d'entraînement par dispositif                               |
| warmup_steps                | Nombre d'étapes pour augmenter progressivement le taux d'apprentissage    |
| max_grad_norm               | Norme maximale des gradients pour éviter des gradients trop grands        |
| metric_for_best_model       | Métrique utilisée pour évaluer le meilleur modèle                         |
| eval_strategy               | Fréquence d'évaluation du modèle durant l'entraînement                    |
| save_strategy               | Fréquence de sauvegarde du modèle durant l'entraînement                   |
| load_best_model_at_end      | Charge le meilleur modèle basé sur la métrique à la fin de l'entraînement |

TABLE 3.4 – Paramètres d'entraînement du modèle DETR

### III.3.2. Métriques d'évaluation

Pour évaluer la performance des modèles entraînés, nous nous référons aux métriques d'évaluation présentées dans le tableau ci-dessous :

- **Mean Average Precision (mAP)** : Le mAP est une mesure utilisée pour évaluer les performances des modèles de détection d'objets. Il est calculé comme la moyenne des précisions moyennes (AP) pour toutes les classes d'objets.
- **Précision (Precision)** : La précision mesure la proportion de prédictions positives correctes parmi toutes les prédictions positives. Elle est calculée comme :

$$\text{Précision} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Positifs}}$$

- **Rappel (Recall)** : Le rappel mesure la proportion de vrais positifs parmi tous les échantillons réellement positifs. Il est calculé comme :

$$\text{Rappel} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Négatifs}}$$

- **F1 Score** : Le F1 Score est la moyenne harmonique de la précision et du rappel. Il est calculé comme :

$$\text{F1 Score} = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

- **Vrais Positifs (TP)** : Prédictions correctes pour la classe positive.
- **Faux Positifs (FP)** : Prédictions incorrectes pour la classe positive.
- **Faux Négatifs (FN)** : Prédictions incorrectes pour la classe négative.
- **Vrais Négatifs (TN)** : Prédictions correctes pour la classe négative.

|           |                 | Vraie condition    |                    |
|-----------|-----------------|--------------------|--------------------|
|           |                 | Condition positive | Condition negative |
| Détection | Détecté positif | TP                 | FP                 |
|           | Détecté négatif | FN                 | TN                 |

FIGURE 3.9 – Matrice de Confusion

- **cls\_loss (classification loss)** : Il s'agit de la perte de classification, qui mesure l'exactitude de la prédiction des catégories des objets dans les boîtes englobantes.
- **box\_loss (bounding box loss)** : Il s'agit de la perte de localisation des boîtes englobantes, qui mesure la précision de la prédiction des positions et des dimensions des boîtes par rapport à la vérité terrain.

On a illustré les métriques calculés dans ce tableau :

| Modèle | Precision | Recall  | F1-score | mAP50   | mAP50-95 | box_loss | cls_loss |
|--------|-----------|---------|----------|---------|----------|----------|----------|
| YOLOv7 | 0.78692   | 0.66109 | 0.7      | 0.75445 | 0.6265   | 0.32     | 0.15     |
| YOLOv8 | 0.81935   | 0.71427 | 0.748    | 0.812   | 0.6993   | 0.4346   | 0.3486   |
| DETR   | 0.21112   | 0.418   | 0.2813   | 0.4770  | 0.5186   | 1.034    | 0.7021   |

**TABLE 3.5 – Comparaison des performances des modèles de détection d’objets**

- **YOLOv7** montre une bonne précision et un rappel raisonnable, ce qui se traduit par un F1-score de 0.7. Les métriques mAP50 et mAP50-95 sont également solides, indiquant une bonne performance globale en détection d’objets. Les pertes de boîte et de classification sont relativement faibles, ce qui est positif.
- **YOLOv8** améliore les performances par rapport à YOLOv7 avec une précision, un rappel et un F1-score plus élevés. Les métriques mAP50 et mAP50-95 sont également meilleures, indiquant une meilleure performance en détection d’objets. Cependant, les pertes de boîte et de classification sont plus élevées, ce qui pourrait indiquer une complexité accrue du modèle.
- **DETR** montre des performances inférieures en termes de précision, de rappel et de F1-score par rapport aux modèles YOLO. Les métriques mAP50 et mAP50-95 sont également plus faibles. Les pertes de boîte et de classification sont significativement plus élevées, ce qui indique que le modèle a plus de difficulté à localiser et à classifier les objets correctement.

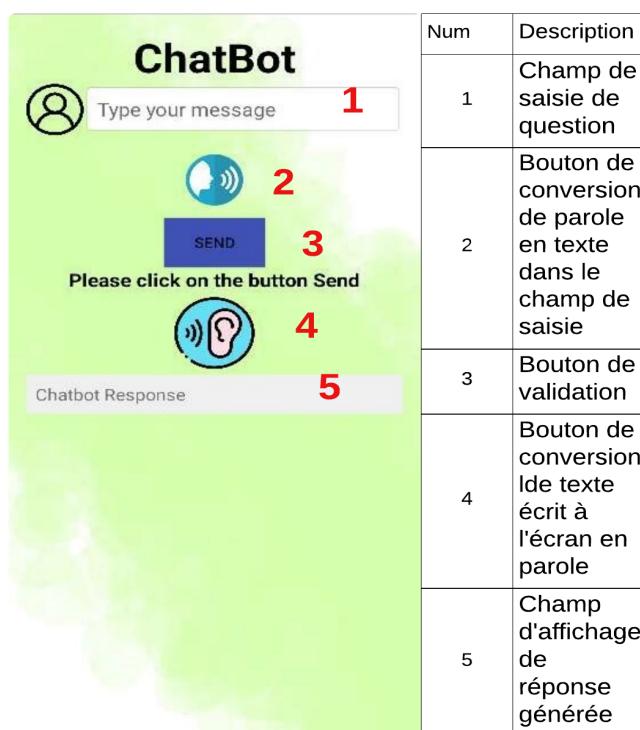
Sur la base des métriques, YOLOv8 est le meilleur modèle parmi les trois. Il offre le meilleur équilibre entre précision, rappel, et score mAP, ce qui le rend plus efficace pour des tâches de détection d’objets dans divers scénarios. Bien que DETR ait un certain potentiel pour la généralisation, ses performances globales sont inférieures à celles des modèles YOLO.

## IV. Développement du Chatbot Agricole

Un chatbot moderne est un programme informatique basé sur l'IA qui simule une conversation humaine avec un utilisateur final. [10]

À cet égard, nous nous efforçons de développer un chatbot agricole offrant divers avantages aux agriculteurs :

- Accès rapide et simple à l'information en zones rurales.
- Support en temps réel pour une prise de décision rapide.
- Conseils personnalisés en fonction des besoins spécifiques.
- Accessibilité multilingue et multimodalité (texte et voix).



**FIGURE 3.10 – Interface de ChatBot Agricole**

## IV.1. Technologies et Outils Utilisés

Pour développer un chatbot, il est crucial d'utiliser un modèle d'IA de génération de texte performant capable de comprendre les instructions et de fournir des réponses précises et rapides. Une réponse imprécise peut entraîner une interaction erronée et potentiellement nuire aux récoltes, tandis qu'une latence de réponse élevée peut provoquer l'insatisfaction de l'utilisateur.

### IV.1.1. Le Modèle de génération de texte

Notre modèle utilisé Meta-LLaMA-3-8B-Instruct est un modèle hébergé sur la plateforme Hugging Face. Il est spécifique de la famille Meta-LLaMA (Large Language Model Meta AI) qui est une famille de modèles de langage développée par Meta (anciennement Facebook). Elle se distingue par sa capacité à générer du texte et à comprendre des instructions.

Cet Modèle a été ajusté (finetuned) à partir de versions précédentes pour répondre efficacement aux instructions. [9]

Ce modèle possède certaines caractéristiques, comme suit :

- **3** fait référence à la troisième génération de la série LLaMA.
- **8B** indique la taille du modèle, qui se compose de 8 milliards de paramètres.
- **Instruct** ce modèle est spécialement optimisé pour la compréhension et l'exécution des instructions utilisateur, particulièrement adapté à des applications interactives et des chatbots sophistiqués.

Le processus de génération de texte par un modèle MetaLLaMA commence par l'entrée d'une séquence de texte, qui sert de contexte initial. Le modèle utilise son architecture basée sur les Transformers pour analyser ce contexte en appliquant des mécanismes d'attention, identifiant

les relations et les dépendances entre les mots. Ensuite, il prédit le mot ou la phrase suivante en se basant sur les connaissances acquises durant le pré-entraînement et le fine-tuning. Chaque prédiction est faite de manière séquentielle, construisant progressivement une réponse complète, cohérente et contextuellement adaptée aux informations fournies par l'utilisateur.

**NB :** Les modèles de génération de texte sont généralement très volumineux, atteignant parfois jusqu'à 9 Go. Cette taille importante pose des problèmes d'intégration directe dans une application mobile, temps d'attente assez long pour traiter les requêtes saisies. Par conséquent, l'utilisation d'une API d'un modèle hébergé sur le cloud se présente comme une solution efficace.

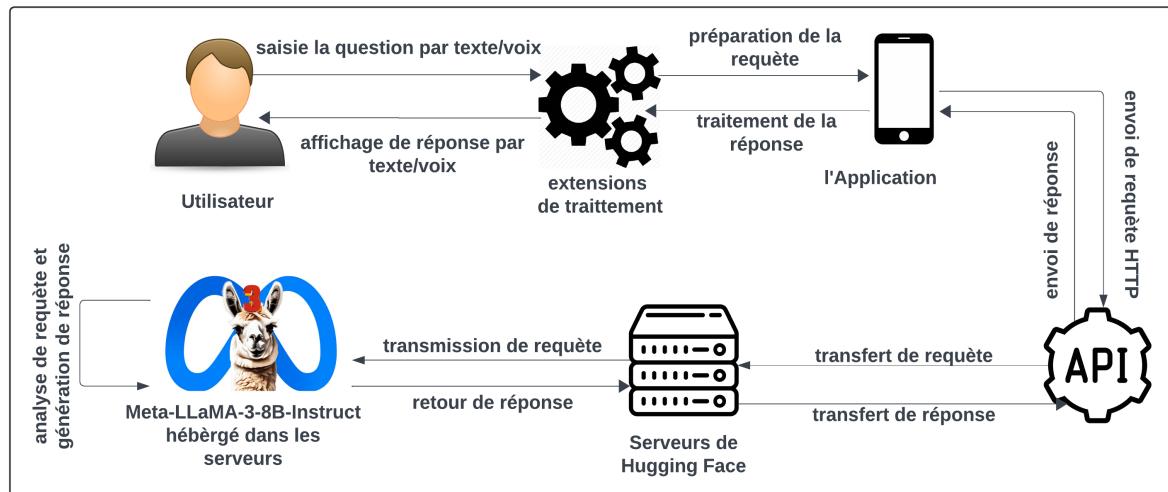
### IV.1.2. Méthode d'interaction : API inférence

Une API, ou interface de programmation d'application, est un ensemble de règles ou de protocoles qui permet aux applications logicielles de communiquer entre elles pour échanger des données, des fonctionnalités et des fonctionnalités.[7]

API d'inférence est un service qui vous permet l'utilisation d'un modèle entraîné pour faire des prédictions sur de nouvelles données. Comme ce processus peut être gourmand en calcul, l'exécution sur un serveur dédié peut être une option intéressante. La bibliothèque huggingface-hub fournit un moyen simple d'appeler un service qui exécute l'inférence pour les modèles hébergés. [8]

## IV.2. Fonctionnement du Chatbot

Le fonctionnement de notre chatbot agricole peut être décrit en plusieurs étapes, depuis l'interaction de l'utilisateur jusqu'à la fourniture de la réponse. Voici un aperçu du processus :



**FIGURE 3.11 – Processus de fonctionnement du Chatbot Agricole**

### 1. Interaction de l'Utilisateur

L'utilisateur peut poser sa question en la saisissant dans le champ de texte ou en utilisant sa voix, qui sera convertie en texte par un module de reconnaissance vocale intégré à Android Studio.

### 2. Envoi de la Requête

Après que l'utilisateur ait saisi sa requête, l'application mobile traite cette entrée et l'envoie à l'API d'inférence hébergée sur Hugging Face via Internet. Cela implique de convertir la requête en un format compatible avec l'API, généralement une requête HTTP structurée.

### 3. Traitement par le Modèle NLP

Une fois la requête reçue par l'API, elle est acheminée vers le modèle Metalama 3B Instruct hébergé sur les serveurs de Hugging Face. Le modèle analyse la requête en utilisant des techniques avancées de traitement du langage naturel (NLP) pour comprendre le sens et le contexte de la question. Il génère ensuite une réponse textuelle pertinente et adaptée en se basant sur ses vastes capacités d'apprentissage et ses compétences acquises.

### 4. Retour de la Réponse

Après avoir générée une réponse précise et pertinente, elle sera ensuite renvoyée aux serveurs. L'API reformate cette réponse, si nécessaire, et la transmet à l'application mobile sous forme d'un fichier JSON (format de données textuel dérivé de la notation des objets du langage JavaScript). L'application reçoit alors le fichier et le prépare pour extraire la réponse voulu et l'afficher ou la vocaliser selon les préférences de l'utilisateur.

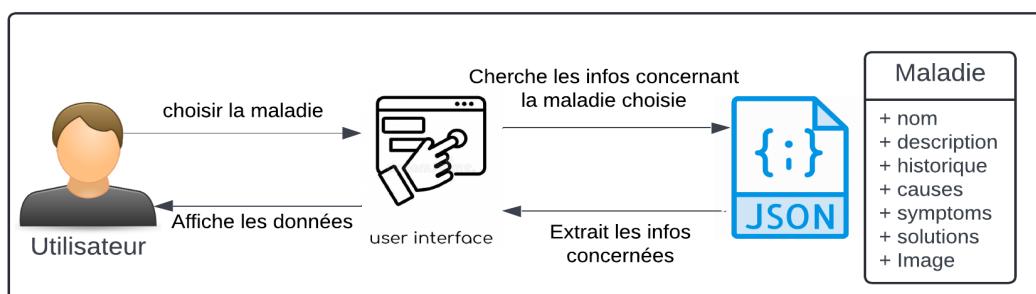
### 5. Affichage de la Réponse

La réponse est ensuite affichée sur l'interface utilisateur. Si la question initiale était vocale, la réponse peut être lue à haute voix par l'application. L'utilisateur peut poser d'autres questions, enchaînant ainsi les interactions pour obtenir des informations détaillées ou clarifications supplémentaires.

Ce processus garantit une interaction fluide et réactive, permettant à l'utilisateur d'accéder rapidement à des informations essentielles pour ses besoins agricoles.

## IV.3. Encyclopédie hors ligne

Si il n'a pas de connexion internet, l'utilisateur sera redirigé vers une encyclopédie hors ligne intégrée à l'application, où il peut rechercher des informations brèves sur les maladies des plantes.



**FIGURE 3.12 – Processus de fonctionnement de l'encyclopédie hors ligne**

## V. CONCLUSION

En conclusion, ce chapitre a détaillé le processus de réalisation du projet en exposant les différentes composantes et services de notre application mobile. On a abordé la mise en œuvre du service de prévisions météorologiques, la détection des maladies des plantes grâce à des modèles de deep learning, ainsi que l'intégration d'autres fonctionnalités innovantes d'assistance intelligente via un chatbot interactif puissant. L'accent a été mis sur l'architecture, le flux de travail et les technologies utilisées, démontrant ainsi l'efficacité et la robustesse de notre solution pour les besoins agricoles.

# CONCLUSION GÉNÉRALE

Le projet AgriCareAI vise à intégrer des technologies modernes dans le secteur agricole, en offrant aux agriculteurs des outils innovants pour améliorer la gestion de leurs cultures. Grâce à des fonctionnalités telles que la détection des maladies des plantes à l'aide de modèles de DL, la prévision météorologique locale et l'assistance via un chatbot intelligent, l'application répond à des besoins réels et pressants des agriculteurs en fournissant des solutions efficaces.

Bien que les résultats obtenus jusqu'à présent soient prometteurs, il existe encore des opportunités pour améliorer notre application. D'abord, on cherche à améliorer les modèles de détection de maladies, en intégrant des bases de données plus vastes et plus large de pathologies végétales. De plus, l'intégration d'une extension d'administrateur est envisagée pour faciliter la gestion des utilisateurs, offrant un meilleur contrôle sur l'accès aux services et garantissant la sécurité des données et la conformité aux réglementations.

Pour financer ces améliorations et assurer la durabilité du projet, certains services avancés pourraient devenir payants comme les analyses détaillées des maladies et des rapports de santé des cultures personnalisés à long terme pourrait être proposé sous forme d'abonnement.

En conclusion, AgricareAI a démontré son potentiel à transformer la pratique agricole en un processus plus efficace, durable et résilient, tout en ouvrant la voie à de futures innovations et améliorations.



---

## WEBLIOGRAPHIE

- [1] **Apache Airflow : qu'est-ce que c'est et comment l'utiliser ?** consulté le 18/08/2024 sur <https://datascientest.com/apache-airflow>
- [2] **Microsoft Power BI** : Disponible sur : [https://fr.wikipedia.org/wiki/Microsoft\\_Power\\_BI](https://fr.wikipedia.org/wiki/Microsoft_Power_BI) consulté le 17/08/2024
- [3] **Roboflow** consulté le 18/08/2024 sur : <https://medium.com/analytics-vidhya/converting-annotations-for-object-detection-using-roboflow-5d0760bd5871>
- [4] **Qu'est-ce qu'Azure SQL ?** : consulté le 18/08/2024 sur <https://learn.microsoft.com/en-us/azure/azure-sql/azure-sql-iaas-vs-paas-what-is-overview?view=azuresql>
- [5] **Weights & Biases (wandb) - Alliance Doc** : consulté le 18/08/2024 sur [https://docs.alliancecan.ca/wiki/Weights\\_%26\\_Biases\\_\(wandb\)/f](https://docs.alliancecan.ca/wiki/Weights_%26_Biases_(wandb)/f)
- [6] **YOLO Object Detection Explained - DataCamp** consulté le 18/08/2024 sur : <https://www.datacamp.com/blog/yolo-object-detection-explained>
- [7] **What Is an API (Application Programming Interface) ?** consulté le 18/08/2024 sur <https://www.ibm.com/topics/api>
- [8] **Inference - Hugging Face** consulté le 18/08/2024 sur : [https://huggingface.co/docs/huggingface\\_hub/package\\_reference/inference\\_client](https://huggingface.co/docs/huggingface_hub/package_reference/inference_client)
- [9] **llama-3-8b-instruct - Workers AI** consulté le 18/08/2024 sur <https://developers.cloudflare.com/workers-ai/models/llama-3-8b-instruct/>
- [10] **What Is a Chatbot ?** consulté le 18/08/2024 sur : <https://www.ibm.com/topics/chatbots>

**AgriCareAi :**  
**Développement d'une Application Mobile pour la Gestion Météo, la Santé des Cultures**  
**et l'Assistance Agricole**

---

---

**Mohamed Amine MISSAOUI**

---

---

**Résumé :**

Le projet AgricareAI vise à révolutionner le secteur agricole en fournissant une application mobile innovante dédiée aux agriculteurs. Cette application propose plusieurs services clés : la détection des maladies des plantes via des modèles de deep learning, des prévisions météorologiques personnalisées, un chatbot intelligent pour l'assistance agricole, et une encyclopédie hors ligne pour fournir des informations essentielles en l'absence de connexion internet. AgricareAI combine des technologies avancées pour aider les agriculteurs à optimiser leurs pratiques, réduire les pertes et améliorer leurs rendements, tout en rendant la gestion agricole plus accessible et efficace.

**Mots clés :** AgriCareAi, agriculture, application mobile, intelligence artificielle, ChatBot, détection des maladies des plantes, apprentissage profond, Visualisation de météo.

**Abstract :**

The AgricareAI project aims to revolutionize the agricultural sector by providing an innovative mobile application dedicated to farmers. This application offers several key services : plant disease detection via deep learning models, personalized weather forecasts, an intelligent chatbot for agricultural assistance, and an offline encyclopedia to provide essential information when internet connection is unavailable. AgricareAI combines advanced technologies to help farmers optimize their practices, reduce losses, and improve yields, while making agricultural management more accessible and efficient.

**Keywords :** AgriCareAI, agriculture, mobile application, artificial intelligence, chatbot, plant disease detection, deep learning, weather visualization.