



FACULTÉ DES SCIENCES DE MONASTIR

DÉPARTEMENT INFORMATIQUE



GREMAHTECH SOLUTIONS AGRITECH

# AgriScan Niger

*Système Intelligent de Détection des Maladies des Plantes*

CAS DE LA POMME DE TERRE

---

Type de document :	Rapport Technique
Auteur :	Amadou Habou Mahamadou
Entreprise :	GREMAHTECH – <a href="https://gremah.vercel.app">https://gremah.vercel.app</a>
Encadrant universitaire :	Mr. Anis Khalfaoui
Date :	21 décembre 2025

Année universitaire : 2025–2026

# Table des matières

1	Introduction . . . . .	2
1.1	Contexte et Motivation . . . . .	2
1.2	Objectifs du Projet . . . . .	2
2	Données et Prétraitement . . . . .	2
2.1	Acquisition des Données . . . . .	2
2.2	Prétraitement et Pipeline TensorFlow . . . . .	2
2.3	Répartition des Données . . . . .	3
3	Architecture et IA . . . . .	3
3.1	Réseau de Neurones Convolutif (CNN) . . . . .	3
3.1.1	Couches de Prétraitement . . . . .	4
3.1.2	Blocs Convolutifs . . . . .	4
3.1.3	Couches de Classification . . . . .	4
3.2	Entraînement . . . . .	4
3.3	Évaluation et Prédications . . . . .	5
4	Backend et MLOps . . . . .	6
4.1	API FastAPI . . . . .	6
4.2	Flux d'inférence . . . . .	7
5	Déploiement Cloud . . . . .	7
6	Interfaces Utilisateurs . . . . .	7
6.1	Web (React.js) . . . . .	7
6.2	Mobile (React Native) . . . . .	7
7	Conclusion . . . . .	8

# 1. Introduction

**AgriScan Niger** est une solution innovante de diagnostic agricole basée sur le **deep learning**, développée par **Gremahtech** pour soutenir les agriculteurs du Niger dans la détection automatique des maladies foliaires de la pomme de terre.

## 1.1 Contexte et Motivation

Les maladies foliaires provoquent des pertes importantes pouvant atteindre jusqu'à 30% de la production selon les années. L'identification précoce est difficile pour les agriculteurs locaux. L'intelligence artificielle, via des modèles de vision par ordinateur, offre une solution rapide et efficace pour l'analyse et la classification des feuilles malades.

## 1.2 Objectifs du Projet

- Détection automatique des types de maladies : **Mildiou Précoce**, **Mildiou Tardif**, et **Sains**.
- Développement d'interfaces web et mobile accessibles même en zones rurales.
- Mise en place d'une infrastructure cloud pour supporter l'inférence et garantir la scalabilité.
- Fournir un rapport technique structuré pour l'évolution future du projet.

# 2. Données et Prétraitement

## 2.1 Acquisition des Données

Les données proviennent du dataset **PlantVillage** et contiennent 2152 images de feuilles de pomme de terre réparties comme suit :

- Early Blight : 1,000 images
- Late Blight : 152 images
- Healthy : 1,000 images

Le déséquilibre des classes nécessite des techniques d'augmentation pour améliorer l'apprentissage.

## 2.2 Prétraitement et Pipeline TensorFlow

- Redimensionnement des images à  $256 \times 256$  pixels et normalisation  $[0, 1]$ .
- Augmentation de données : flips, rotations, modifications de luminosité.

— Pipeline optimisé avec `cache()` et `prefetch()` pour accélérer l'entraînement.

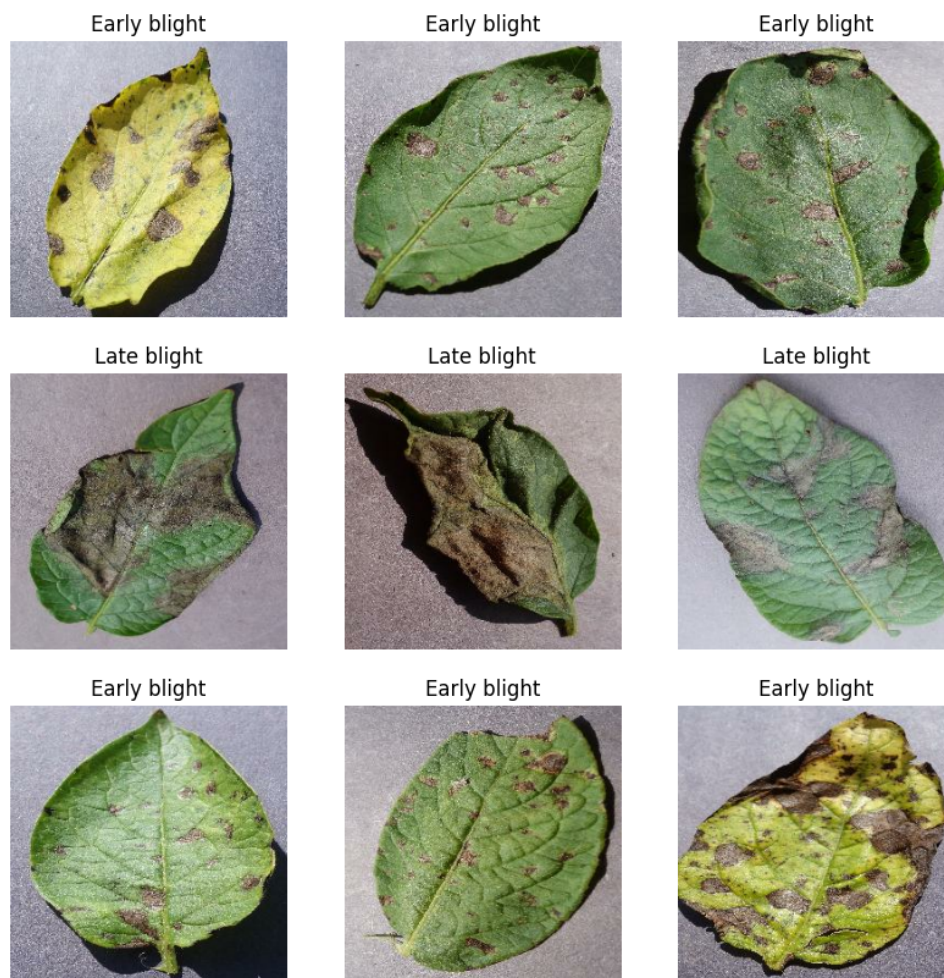


FIGURE 1 – Exemple de feuilles de pomme de terre utilisées pour l'entraînement.

## 2.3 Répartition des Données

- Entraînement : 80%
- Validation : 10%
- Test : 10%

## 3. Architecture et IA

### 3.1 Réseau de Neurones Convolutif (CNN)

Le modèle est un CNN séquentiel adapté à la classification multi-classes de feuilles.

### 3.1.1 Couches de Prétraitement

- Redimensionnement et rescaling
- Augmentation de données en temps réel

### 3.1.2 Blocs Convolutifs

Le modèle comporte 6 blocs Conv2D + MaxPooling pour extraire les caractéristiques :

- Bloc 1 : Conv2D(32, 3x3) + ReLU + MaxPooling
- Bloc 2 : Conv2D(64, 3x3) + ReLU + MaxPooling
- Bloc 3 : Conv2D(64, 3x3) + ReLU + MaxPooling
- Bloc 4 : Conv2D(64, 3x3) + ReLU + MaxPooling
- Bloc 5 : Conv2D(64, 3x3) + ReLU + MaxPooling
- Bloc 6 : Conv2D(64, 3x3) + ReLU + MaxPooling

### 3.1.3 Couches de Classification

- Flatten
- Dense(64) + ReLU
- Dense(3) + Softmax

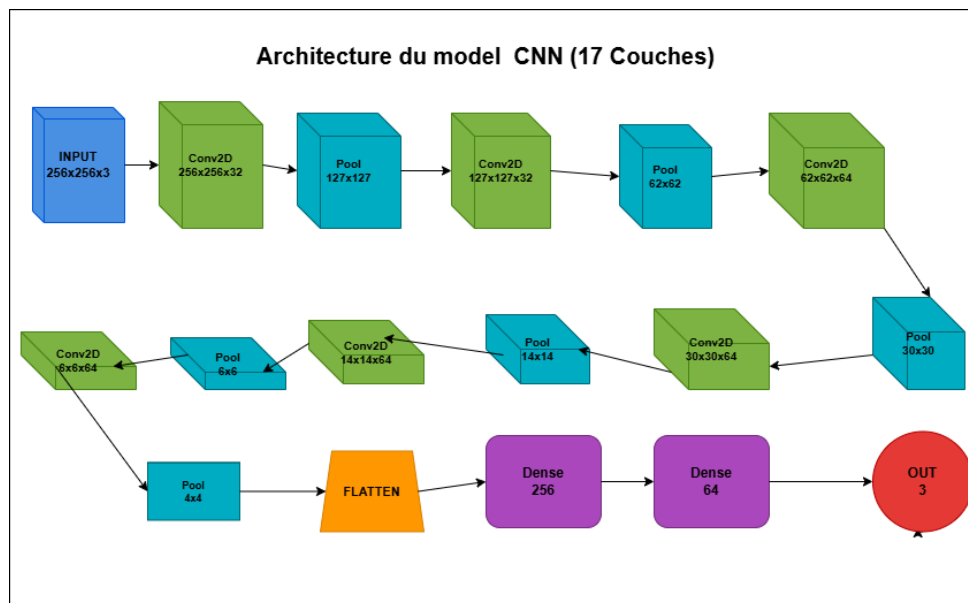


FIGURE 2 – Architecture du modèle CNN.

## 3.2 Entraînement

- Optimiseur : Adam

- Fonction de perte : SparseCategoricalCrossentropy
- Epochs : 50
- Batch size : 32

### 3.3 Évaluation et Prédictions

- Accuracy finale sur test : 98.44%
- Courbes d'accuracy et loss visualisées pour convergence et stabilité
- Exemple de prédiction sur batch de test avec confiance affichée

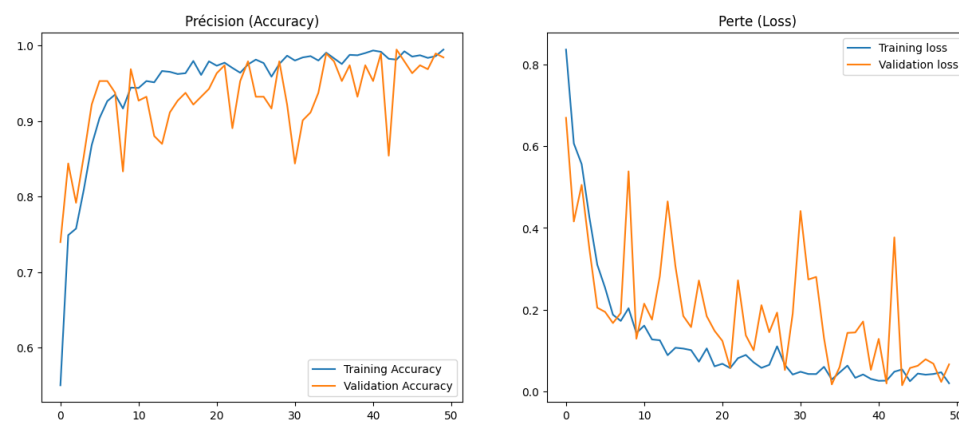


FIGURE 3 – Performance du modèle.

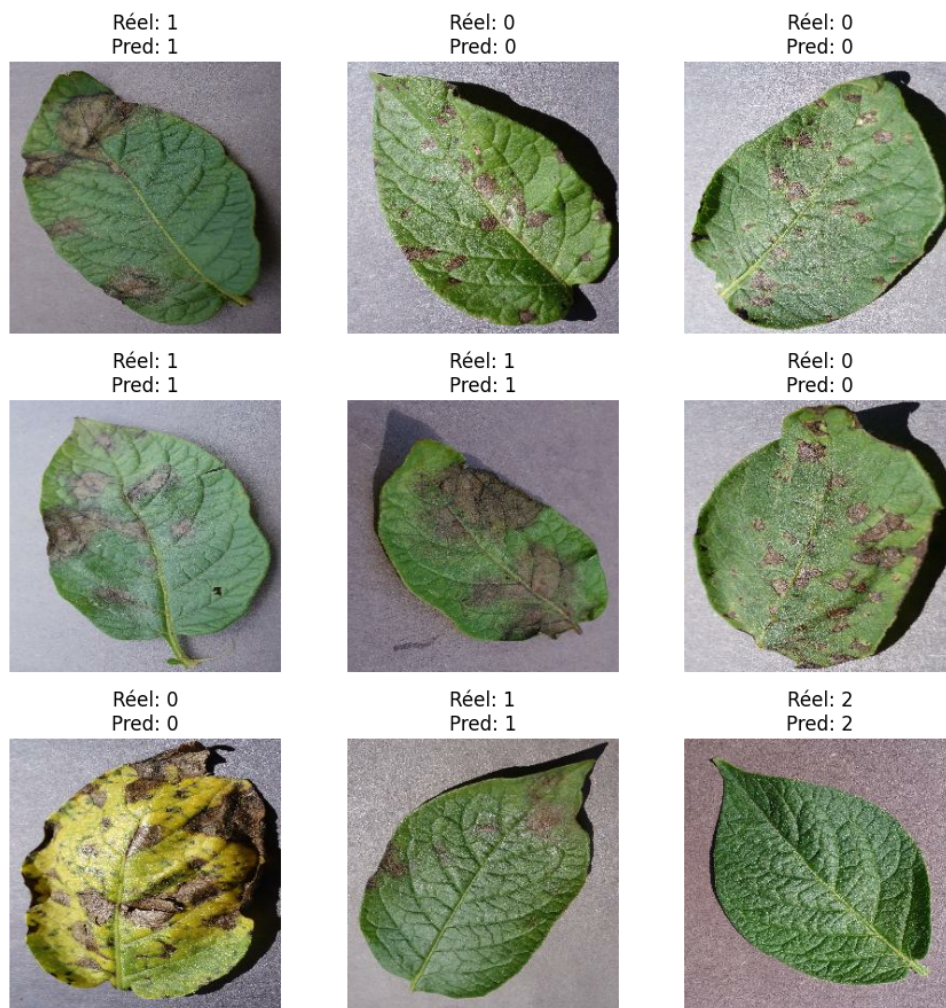


FIGURE 4 – Exemple de pr  dictions et labels r  els.

## 4. Backend et MLOps

### 4.1 API FastAPI

- Re  oit images des utilisateurs
- Convertit en tableaux NumPy
- Effectue l'inf  rence via le mod  le Keras
- Retourne les r  sultats en temps r  el



## 4.2 Flux d'inférence

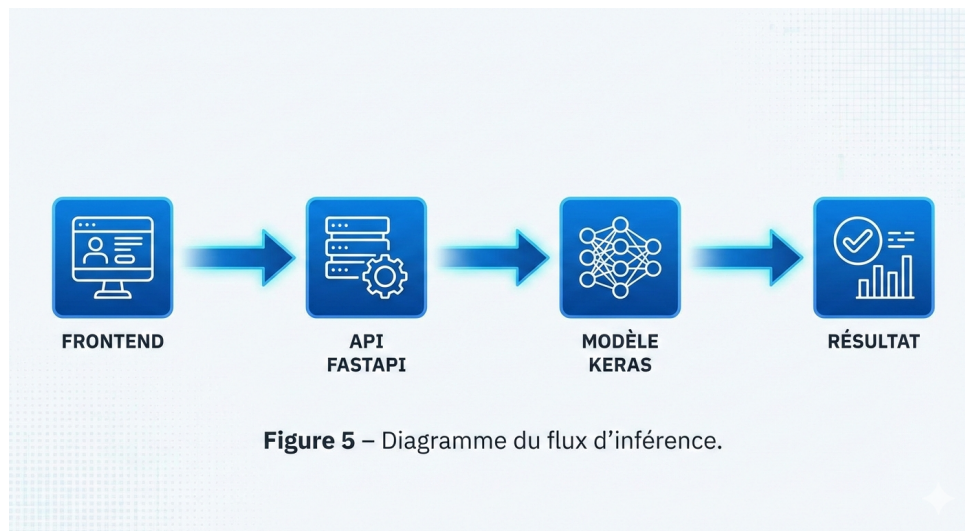


FIGURE 5 – Diagramme du flux d'inférence : Frontend → API FastAPI → Modèle Keras → Résultat.

## 5. Déploiement Cloud

- Hébergement de l'API et du modèle sur **Microsoft Azure** (Azure App Service pour le backend et Azure Storage pour les images/modèles).
- Endpoint REST accessible depuis le web et mobile via l'URL Azure.
- Scalabilité et fiabilité assurées grâce aux services managés d'Azure pour supporter un grand nombre d'utilisateurs simultanés.
- Surveillance et logs centralisés via Azure Monitor et Application Insights.

## 6. Interfaces Utilisateurs

### 6.1 Web (React.js)

Interface web intuitive pour upload d'images et affichage instantané des résultats.

### 6.2 Mobile (React Native)

Application mobile adaptée aux zones rurales avec accès direct à l'API cloud et interface optimisée pour faible bande passante.



## 7. Conclusion

**AgriScan Niger** propose une solution complète d'IA pour l'agriculture, alliant précision, robustesse et accessibilité. Le projet montre la capacité de **Gremahtech** à développer des solutions technologiques innovantes pour améliorer la productivité et la sécurité alimentaire au Niger.

## Références

1. PlantVillage Dataset. <https://www.kaggle.com/datasets/emmarex/plantdisease>
2. TensorFlow Documentation. <https://www.tensorflow.org/>
3. FastAPI Documentation. <https://fastapi.tiangolo.com/>
4. Microsoft Azure. <https://azure.microsoft.com/fr-fr/>
5. Chollet, F. *Deep Learning with Python*, Manning Publications, 2017.