Metz Numeric School

CD2IA 2024/2025

GreenCheck by AgroNovaTech

Diagnostic intelligent de plantes

Présentation de l'entreprise

AgroNovaTech est une PME fictive développant des outils numériques pour les jardiniers amateurs et professionnels. Elle souhaite proposer une webapp permettant aux utilisateurs d'obtenir un **diagnostic automatique de l'état de santé de leurs plantes** à partir d'images envoyées et de quelques informations contextuelles.

Objectifs du projet

Développer une application web sécurisée et accessible qui :

- Permet de soumettre une photo d'une plante avec un formulaire de contexte (type de plante, localisation, exposition, etc.).
- Utilise un modèle de **machine learning (classification d'image)** pour détecter d'éventuelles maladies ou carences.
- Fournir une fiche de diagnostic automatisée avec des conseils.
- Enregistre l'historique utilisateur dans un espace personnel.

Fonctionnalités attendues

a. Espace utilisateur

- Authentification, gestion du compte.
- Formulaire de soumission : image + données complémentaires.
- Historique des diagnostics.
- Fiche de diagnostic PDF générée automatiquement.

b. Moteur d'IA (Machine Learning)

- Classification d'images à partir d'un modèle ML (préentraîné ou entraîné sur mesure, ex : `MobileNet`, `ResNet`).

- Modèle hébergé dans un microservice Python (Flask API ou FastAPI).
- Entrées du modèle : image + données du formulaire (optionnel).
- Sortie : label de maladie ou "sain" + niveau de confiance + conseils associés (règle métier).

c. Back-end

- Base de données utilisateur / diagnostics.
- API sécurisée pour front et moteur IA.
- Gestion de quotas ou limites d'usage (freemium possible).

d. Front-end

- UI responsive (mobile/tablette/desktop).
- Interface de téléversement, prévisualisation d'image.
- Rendu clair et structuré des résultats d'analyse.

Contraintes fonctionnelles & techniques

Domaine	**Contraintes**
Sécurité	Authentification sécurisée, validation des fichiers, contrôle des entrées
Accessibilité	RGAA 4.1 (contrastes, navigation clavier, aria-labels)
Performance	Limitation taille image, mise en cache des diagnostics
Confidentialité	Données pseudonymisées, image supprimée après traitement si demandé (RGPD)
IA	Modèle ML léger et rapide en inférence, avec API REST documentée (Swagger/OpenAPI)

Technologies recommandées

Couches	**Technologies suggérées**
Front-end	React ou Vue + TailwindCSS

Couches	Technologies suggérées
Back-end	Python / Flask ou Django
IA	Python (TensorFlow / Keras ou PyTorch), FastAPI ou Flask
BDD	MySQL ou MangoDB
DevOps	Docker, docker-compose, GitHub Actions
Tests	Jest, Pytest, OWASP ZAP, Postman

Livrables attendus

a. Techniques

- Code source structuré, documenté
- Dossier projet (architecture, BDD, diagrammes) - **TITRE***
- Scripts d'installation / déploiement (Docker, CI/CD)
- Fichier `.env.example` avec variables nécessaires
- Journal de veille (IA, sécurité, accessibilité) - **TITRE***

b. Fonctionnels

- Documentation utilisateur (PDF ou web)
- Présentation orale avec diaporama - **TITRE***
- Jeu de tests automatisés + plan de test manuel - **TITRE***
- Rapport d'audit d'accessibilité et de sécurité

* **Obligatoire pour le passage du titre !**

Plan de développement indicatif

Phase	Durée	Objectif
Analyse & conception	1 semaine	Dossier de conception complet
Front & back-end de base	2 semaines	Connexion, téléversement, DB

Phase	Durée	Objectif
Intégration modèle IA (API Flask)	1 semaine	Classification image + réponse API
Tests, sécurité, validation RGPD	1 semaine	Tests complets + respect des normes
CI/CD + Dockerisation + soutenance	1 semaine	Déploiement final & présentation