

Spécification détaillées

**Détection d’anomalies pulmonaires**

**(COVID et Pneumonies Virales)**

**Plan**

1. Objectif
2. Acteurs
3. Cas d’usage
4. Présentation de l’interface utilisateur - Frontend
5. Présentation de l’API – Backend
6. Spécifications techniques – Fonctionnement Backend
7. Règles de gestion - Monitoring de l’application

# Objectif

L’objectif du client est de fournir un service de détection d’anomalie pulmonaire type COVID ou Pneumonie Virale à ses internes en médecine afin de réaliser un premier diagnostic permettant de trier les patients non malades et les patients présentant une anomalie pulmonaire.

# Acteurs

* **Les utilisateurs :**

Internes : Ce sont des praticiens en cours de formation spécialisée qui exercent sous la responsabilité d’un médecin senior. Ils effectuent un premier diagnostic sur l’état de santé d’un patient avant de faire appel à des médecins

* **Les contributeurs :**

Ce sont des utilisateurs de profil radiologue (médecin senior) qui valident le diagnostic fourni par l’application aux internes. Leur expertise permet d’améliorer les modèles de prédiction au début de la mise en place de l’outil

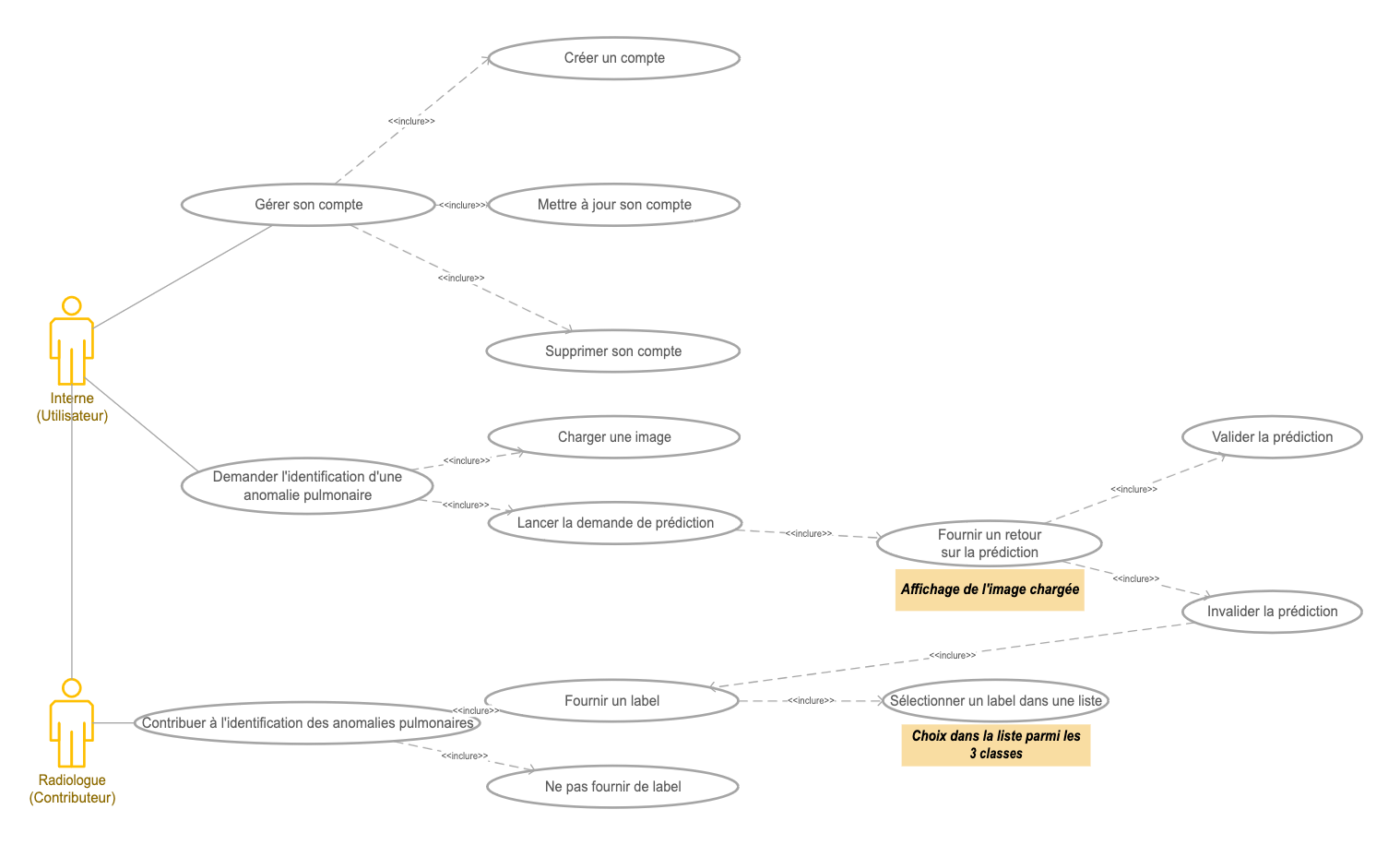
* **L’administrateur :**

Deux profils d’administrateur :

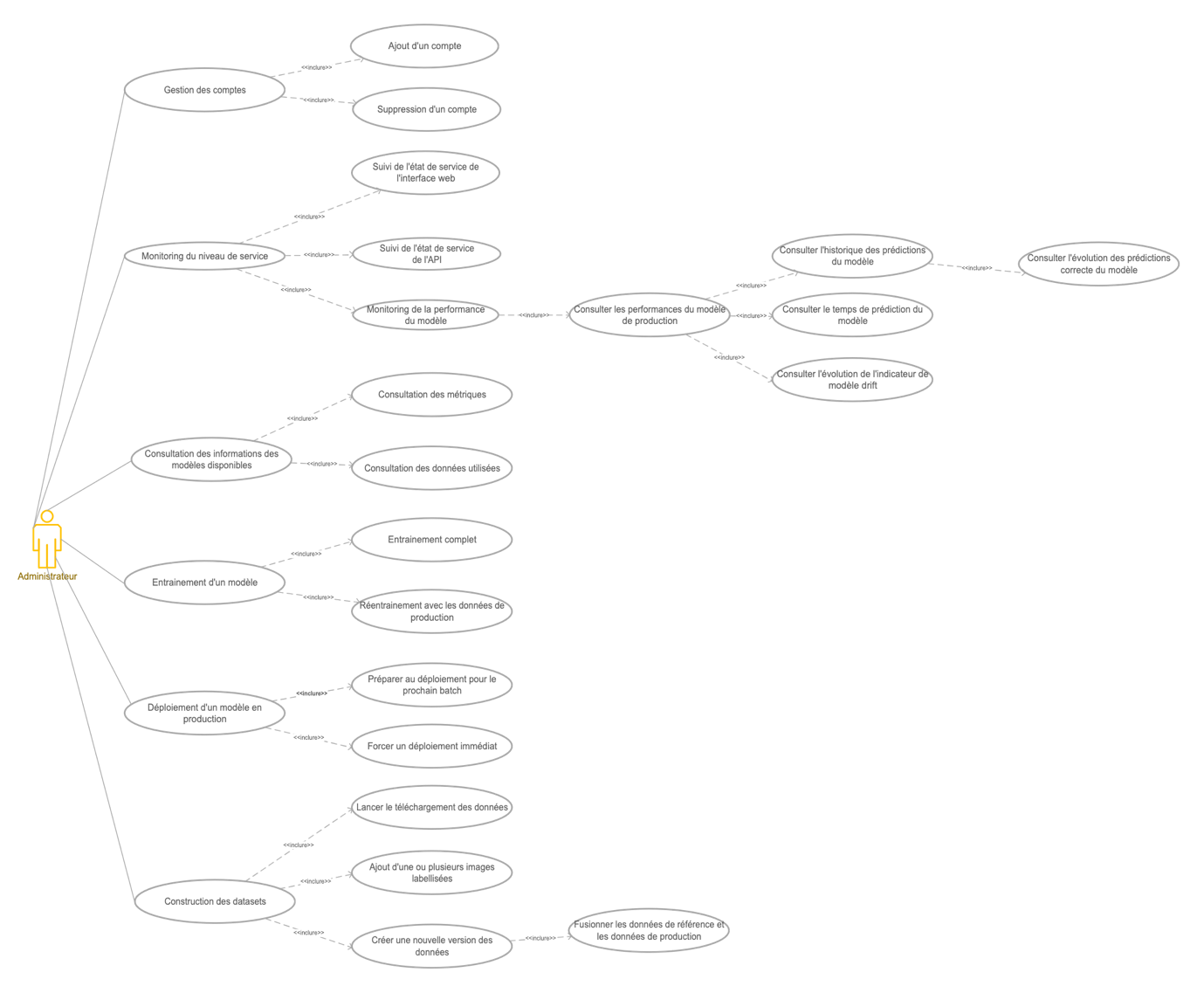
* **Datascientist :** Son rôle est de répondre aux besoins du client en matière de performance du modèle. Ainsi, il choisit les données les plus pertinentes, construit l’architecture IA adaptée et définit les métriques pour mesurer les performances après déploiement. Il effectue donc l’entrainement et l’évaluation d’un modèle au début du projet
* **MLOps Engineer :** Son rôle est de s’assurer du bon fonctionnement du service pour les utilisateurs en surveillant l’état de l’application mais aussi la performance du modèle. En cas de drift du modèle, il peut décider d’effectuer un ré-entrainement du modèle et de déployer le nouveau modèle si les performances sont conformes aux exigences du client.

# Les cas d’usage

**Vue générale des cas d’usage par acteur – Utilisateur/Contributeur**



**Vue générale des cas d’usage par acteur – Administrateur**



## Utilisateur

* + 1. **Gérer son compte**

Pour utiliser notre application, un utilisateur doit être authentifié et doit donc créer un compte, il doit aussi pouvoir le mettre à jour ou le supprimer.

* ***Créer un compte***

Pour la création d’un compte, un nom d’utilisateur et mot de passe fort seront demandés. *Pour des raisons de simplicité, il ne sera pas demandé de confirmer la création du compte par email.*

***Nom d’utilisateur : 12 caractères minimum***

***Mot de passe : 12 caractères minimum, Inclure des majuscules (A, B, C, ...) et des minuscules (a, b, c, ...). Utiliser des chiffres (0-9). Intégrer des symboles et caractères spéciaux (par exemple, @, #, $, %, &, \*).***

* ***Mettre à jour son compte***

L’utilisateur aura la possibilité de mettre à jour son mot de passe. Pour cela, il doit saisir son mot de passe actuel et son nouveau mot de passe à deux reprises.

***Nouveau Mot de passe : Différent du mot de passe actuel, 12 caractères minimum, Inclure des majuscules (A, B, C, ...) et des minuscules (a, b, c, ...). Utiliser des chiffres (0-9). Intégrer des symboles et caractères spéciaux (par exemple, @, #, $, %, &, \*).***

* ***Supprimer son compte***

L’utilisateur peut supprimer son compte à tout moment.

* + 1. **Demander l'identification d'une anomalie pulmonaire**
* ***Charger une image***

Afin d’identifier une anomalie pulmonaire, un utilisateur doit charger une image.

***Les formats acceptés : png, jpg***

***Taille maximale acceptée : 500 Kb***

***Nombre maximal d’images nécessaires pour une prédiction : 1***

L’utilisateur a la possibilité de supprimer l’image et d’en recharger une nouvelle avant de lancer une demande de prédiction.

* ***Lancer une demande de prédiction***

Lorsque l’utilisateur lance une demande de prédiction, celle-ci est réalisée en Backend et sont fournis à l’utilisateur la classe et la **photo** chargée par l’utilisateur, ainsi que l’**indice de confiance**.

* + Fournir un retour sur la prédiction :
    - Valider la prédiction : A l’aide de la classe et l’affichage de l’image de radiographie fournie
    - Invalider la prédiction et fournir une classe : Choix d’une classe dans la liste
    - Invalider la prédiction sans fournir de label : Lorsque l’utilisateur ne parvient pas à identifier l’anomalie

## Contributeur (Hors MVP)

* + 1. **Contribuer à l'identification des anomalies pulmonaires**
* ***Fournir un label***

Seule l’image non labellisée est affichée

Lorsque le contributeur peut établir un diagnostic, il peut *sélectionner un label dans une liste*

* ***Ne pas fournir de label***

Lorsque l’utilisateur ne parvient pas à identifier l’anomalie, il n’effectue aucune action.

***Règle de validation définitive d’un label : 1 validation par le contributeur***

## Administrateur

* + 1. **Gestion des comptes**
    2. **Monitoring du niveau de service**
       1. Suivi de l'état de service de l'interface web
       2. Suivi de l'état de service de l'API
       3. Monitoring de la performance du modèle
* Consulter les performances du modèle de production
  + Consulter l'historique des prédictions du modèle
    - Consulter l'évolution des prédictions correcte du modèle
  + Consulter le temps de prédiction du modèle
  + Consulter l'évolution de l'indicateur de modèle drift
    1. **Consultation des informations des modèles disponibles**
       1. Consultation des métriques
       2. Consultation des données utilisées
    2. **Entrainement d'un modèle**
       1. Entrainement complet
       2. Réentrainement avec les données de production
    3. **Déploiement d'un modèle en production**
       1. Préparer au déploiement pour le prochain batch
       2. Forcer un déploiement immédiat
    4. **Construction des datasets** 
       1. Lancer le téléchargement des données
       2. Ajout d'une ou plusieurs images labellisées
       3. Créer une nouvelle version des données
    5. **Traitement des alertes reçues**

L’administrateur reçoit des alertes par mail sur le fonctionnement de l’application. Le type de message reçu :

Dysfonctionnement de l’API (Retour erreur 503) – Date et Heure

* ***Monitoring du niveau de service***
  + Suivre l’état de service de l’interface web

Dysfonctionnement de l’application web (Retour erreur 503) – Date et Heure

* + Suivre l’état de service de l’API

Dysfonctionnement de l’API (Retour erreur 503) – Date et Heure

* ***Monitoring de la performance du modèle***

Une alerte est reçue par l’administrateur lorsqu’un « drift » est détecté, ce qui est un indicateur de l’altération de la performance d’un modèle. Celui-ci est basé sur la précision des prédictions demandées par les utilisateurs et par la validation des prédictions par les utilisateurs et les contributeurs.

Règles de calcul du drift et seuil

* + 1. **Gestion des performances du modèle**
  + Réentrainement du modèle

L’action de réentraînement d’un modèle est manuelle. La décision de relancer un réentrainement dépendra de l’analyse par l’administrateur :

* De la performance et la non régression du nouveau modèle.
* De l’intégration d’une nouvelle espèce.
* Du volume de nouvelles images labellisées (pour les espèces existantes) afin d’améliorer les performances du modèle

Toutes ces décisions de réentrainement du modèle dépendent du volume de données.

* + Déployer un nouveau modèle

Le déploiement d’un nouveau modèle dépendra de la performance cible souhaitée.

# Présentation de l’interface utilisateur – Frontend

***« Streamlit »*** sera utilisé pour développer une interface utilisateur simple et intuitive.

Seront proposés les cas d’usage des utilisateurs et contributeurs.

Une API sert de passerelle entre les actions demandées par le métier et le Back End.

La page d’accueil invite l’utilisateur à s’authentifier ou à s’inscrire.

Lorsque la connexion est établie, cela activera trois sections du menu principal :

* Reconnaitre un oiseau
* Contribuer
* Mon compte

# Présentation de l’API – Backend

***« FastAPI »*** est le framework utilisé pour le développement de l’API du projet de reconnaissance des oiseaux.

L’appel à l’API sera sécurisé par l’utilisation de clé API

* ***(CU Administrateur - Suivi de l’état de service de l’API)***

GET

@/

Permet de faire une simple requête vers l’API afin de vérifier que l’on obtient bien un status code “200”. Elle retourne l’erreur 503 si le service n’est pas démarré

* ***(CU Utilisateur - S’authentifier)***

POST

***@/***token

Permet à l’utilisateur de s’authentifier avec son login et mot de passe

* ***(CU Utilisateur - Lancer une demande de prédiction)***

GET

@/get\_class\_image

Permet de charger une image sur laquelle appliquer la prédiction.

*Format de l’entrée : .jpg, .jpeg, .png*

*Format de réponse : JSON, Format FileResponse qui correspond au chemin du fichier chargé sur le serveur*

POST

@/predict

Permet de faire une prédiction de classe à partir d’une image envoyée à l’API.

*Format de l’entrée : .jpg, .jpeg, .png*

*Format de réponse : JSON, contenant le score et la classe + image d’exemple de la classe prédite*

* ***(CU Utilisateur - Valider la prédiction) + Résultat des contributions***

*POST*

@/add\_image

Permet d’ajouter une image dans la base de données et la classe déjà existante associée

*Format de l’entrée: .jpg, .jpeg, .png + label*

* ***(CU Utilisateur - Invalider la prédiction – Fournir un label) + (CU Contributeur – Fournir un label)***

*GET*

@/get\_species

Permet d’obtenir la liste des espèces d’oiseaux

*Format de réponse : JSON, contenant la liste des espèces*

# Spécifications techniques – Fonctionnement Backend

A venir (API et Architecture)

# Règles de gestion - Monitoring de l’application