MachinaEar
Système de Maintenance Prédictive Intelligent

Élaboré par :

Aziz BEN SALAH
Salem Aziz FRADI
Zied KALLEL
Chiheb Eddine ZIDI

Encadré par :

Dr. Mohamed Béchaa KAÂNICHE

Année Universitaire 2025 – 2026

Table des matières

1 Introduction

MachinaEar est un projet de maintenance prédictive embarquée visant à surveiller la santé des machines industrielles à faible coût.

Le système repose sur un **Raspberry Pi 4** équipé de capteurs acoustiques et vibratoires, un modèle d'intelligence artificielle embarqué (**TensorFlow Lite**), et une architecture logicielle cloud basée sur **Jakarta EE** et **MongoDB Atlas**.

L'objectif est de détecter précocement les anomalies mécaniques, de prévenir les défaillances, et d'offrir une interface web intuitive pour la supervision et la maintenance proactive.

2 Architecture Générale du Système

L'architecture de MachinaEar repose sur trois couches interdépendantes :

1. **Couche Edge (IoT) :** Raspberry Pi 4 exécutant un modèle TensorFlow Lite pour l'analyse locale des signaux.
2. **Couche Cloud :** Serveur WildFly hébergeant les services REST Jakarta EE, la base de données MongoDB Atlas et l'intégration avec une plateforme de streaming d'événements (**Apache Kafka**).
3. **Couche Application :** Interface web permettant la visualisation en temps réel, la configuration et la gestion des alertes.

3 Diagramme de Cas d'Utilisation

Le diagramme de la figure 1 présente les principaux acteurs du système MachinaEar et les cas d'utilisation associés.

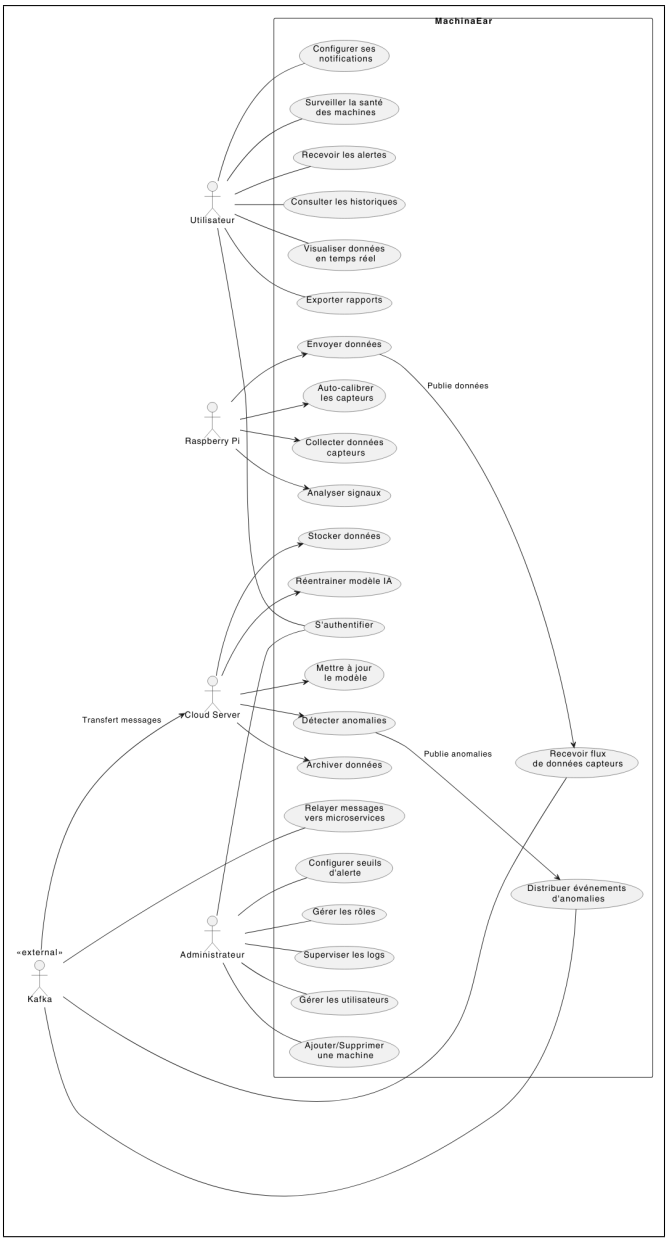

Figure 1 – Diagramme de cas d'utilisation du système MachinaEar.

L'**Utilisateur** se connecte à l'application pour surveiller la santé des machines, consulter les historiques, visualiser les données en temps réel, recevoir les alertes, configurer ses notifications et exporter des rapports.

L'**Administrateur** dispose de fonctionnalités supplémentaires : gestion des utilisateurs et des rôles, ajout ou suppression de machines, configuration des seuils d'alerte et supervision des logs du système.

Le **Raspberry Pi** collecte les données provenant des capteurs, analyse les signaux localement, peut auto-calibrer les capteurs puis envoie les données vers le cloud.

Le **Cloud Server** stocke les données, détecte les anomalies, réentraîne et met à jour le modèle d'IA, archive les données et relaie certains messages vers d'autres microservices.

Le système externe **Kafka** reçoit les flux de données capteurs ainsi que les événements d'anomalies et les distribue à d'autres systèmes consommateurs (analytique, supervision globale, etc.).

4 Diagramme de Classes

La figure 2 décrit les principales classes du système MachinaEar et leurs relations.

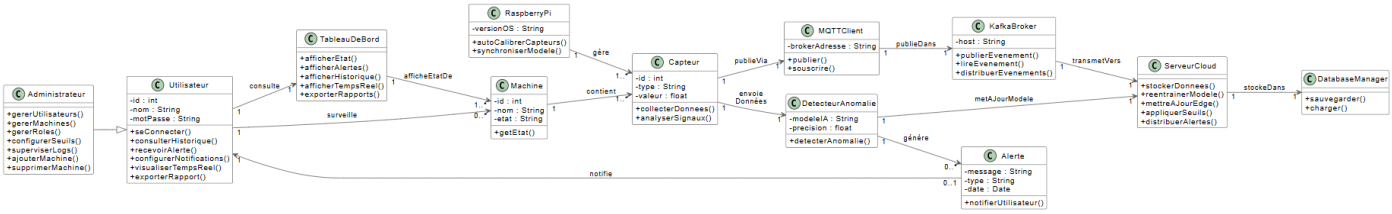

Figure 2 – Structure des classes du système MachinaEar.

Analyse des relations principales

La classe **Utilisateur** regroupe les informations d'authentification et les opérations de base (connexion, consultation des historiques, réception d'alertes, configuration des notifications, visualisation temps réel, exportation de rapports). La classe **Administrateur** hérite de **Utilisateur** et ajoute des opérations de gestion des utilisateurs, des machines et des rôles.

Chaque **Utilisateur** consulte un **TableauDeBord** qui affiche l'état des machines, les alertes et les indicateurs synthétiques. Une **Machine** est caractérisée par son identifiant, son nom et son état. Elle contient un ou plusieurs **Capteurs**, qui collectent et analysent les signaux physiques.

Les données issues des capteurs sont envoyées au **DétecteurAnomalie** via un **MQTTClient**. Le détecteur applique le modèle d'IA, calcule des scores d'anomalie et génère des **Alertes** qui notifient l'utilisateur. Les événements sont publiés dans un **KafkaBroker**, qui se charge de distribuer les événements aux autres composants.

La classe **ServeurCloud** coordonne le stockage des données, le réentraînement du modèle, l'application des seuils et la distribution des alertes. La persistance est assurée par le **DatabaseManager**, qui interagit avec **MongoDB Atlas**.

5 Diagramme de Déploiement

Le diagramme de la figure 3 illustre la distribution physique du système entre la couche Edge, le serveur d'application et les services externes.

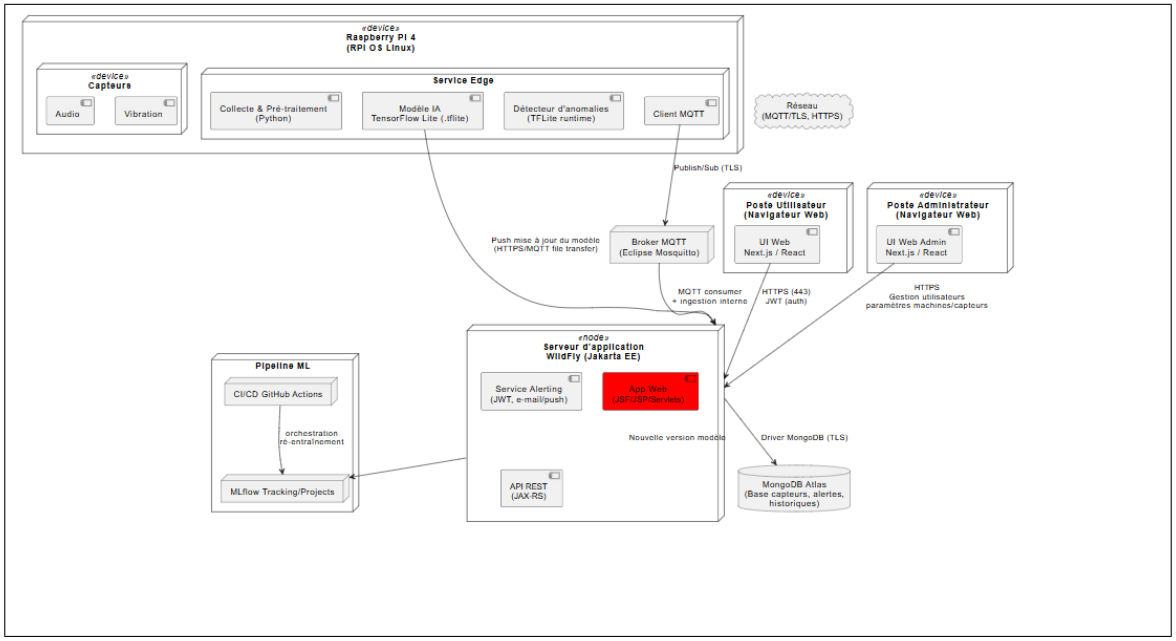

Figure 3 – Diagramme de déploiement du système MachinaEar (Edge–Cloud–Pipeline IA).

Description du déploiement

Sur la couche **Edge**, un **Raspberry Pi 4** exécute un service de collecte et de prétraitement (Python), le modèle IA TensorFlow Lite, un détecteur d'anomalies et un client MQTT. Les capteurs audio et vibration sont connectés à ce dispositif.

Sur la couche **Cloud**, un serveur d'application **WildFly (Jakarta EE)** héberge l'application web, les API REST, le service d'alerting (JWT, e-mail/push) et les consommateurs MQTT qui ingèrent les données. La base **MongoDB Atlas** stocke les mesures, les alertes et les historiques.

Un **pipeline ML** basé sur **GitHub Actions** et **MLflow** gère le réentraînement et le déploiement des nouvelles versions du modèle. Les interfaces utilisateur (poste opérateur et poste administrateur) accèdent au système via le navigateur, en HTTPS, avec authentification par **JWT**.

6 Sécurité et Communication

— Transmission sécurisée via **HTTPS** et **MQTTs**.

— Authentification par jeton **JWT**.

— Contrôle d'accès basé sur les rôles (utilisateur / administrateur).
— Sauvegarde automatique des données et versioning des modèles IA.

7 Améliorations Futures

— Intégration d'une application mobile Android connectée à l'API REST.
— Ajout de nouveaux capteurs : température, courant, pression.
— Support du protocole industriel OPC-UA.
— Visualisation enrichie via PrimeFaces et JSF.

8 Conclusion

Le projet **MachinaEar** combine **IoT**, **IA embarquée** et **architecture cloud** pour offrir une solution complète de maintenance prédictive.

Grâce à l'utilisation de **Jakarta EE** et de **Raspberry Pi 4**, le système garantit une détection en temps réel, une grande fiabilité et une intégration simple dans les environnements industriels modernes.