Université Sorbonne Paris Nord
UFR Sciences Économiques et de Gestion

Projet Data Science

Réalisé par :

Khoubeib Bourbia
Tasnime Hakimi
Jinene Hamrouni

Année Universitaire 2025–2026

Table des matières

Table des figures

1 Introduction

1.1 Contexte général

La transformation numérique du secteur de la santé a profondément modifié les modalités de collecte, de stockage et d'analyse des données médicales. L'essor des dispositifs médicaux connectés et des systèmes d'information hospitaliers permet désormais la génération continue de données physiologiques telles que la pression artérielle, le rythme cardiaque ou la saturation en oxygène.

Cette évolution s'accompagne d'une augmentation significative du volume et de la vélocité des données produites. Dans un environnement clinique moderne, des centaines voire des milliers de patients peuvent transmettre simultanément leurs mesures. Le traitement manuel de ces flux devient alors inadapté face aux exigences de réactivité et de fiabilité requises pour la surveillance médicale.

Dans ce contexte, les technologies Big Data et les architectures de streaming temps réel apparaissent comme des solutions pertinentes pour assurer l'ingestion, l'analyse et la visualisation automatisée de ces données critiques.

1.2 Problématique

La surveillance continue de la pression artérielle constitue un enjeu majeur dans la prévention des pathologies cardiovasculaires. Une détection tardive des anomalies peut entraîner des complications graves, notamment des crises hypertensives ou des accidents vasculaires cérébraux.

Cependant, plusieurs défis techniques se posent dans un environnement à grande échelle :

— Comment gérer un flux continu et massif de données médicales ?
— Comment catégoriser automatiquement les mesures selon les standards médicaux (AHA) et prédire les risques futurs sans intervention humaine systématique ?
— Comment organiser le stockage des données en fonction de leur type (données normales vs données nécessitant une surveillance) ?
— Comment proposer des outils de visualisation et de monitoring en temps réel facilitant la prise de décision médicale ?
— Comment intégrer l'intelligence artificielle pour anticiper les crises hypertensives avant qu'elles ne surviennent ?

La problématique centrale du projet peut ainsi être formulée comme suit :

> Comment concevoir une architecture Big Data capable d'ingérer, traiter, analyser et visualiser en temps réel des données de pression artérielle, tout en catégorisant automatiquement les mesures selon les standards médicaux, en intégrant une dimension prédictive basée sur le Machine Learning pour anticiper les risques de crise hypertensive, et en fournissant des interfaces de monitoring adaptées aux professionnels de santé ?

1.3 Objectifs du projet

L'objectif principal de ce projet est de concevoir et d'implémenter un système intelligent de surveillance médicale reposant sur une architecture de streaming temps réel.

Plus spécifiquement, le projet vise à :

- **Simuler la génération de données médicales structurées au format FHIR :** Créer un générateur de données réalistes respectant le standard FHIR (Fast Healthcare Interoperability Resources) pour les observations de pression artérielle, garantissant ainsi l'interopérabilité avec les systèmes d'information hospitaliers.
- **Mettre en place un pipeline de streaming basé sur Apache Kafka :** Implémenter un système de messagerie distribuée permettant l'ingestion en temps réel des données médicales.
- **Implémenter une catégorisation médicale fondée sur les standards AHA :** Classifier automatiquement les mesures selon les guidelines de l'American Heart Association (AHA), en utilisant les seuils standardisés pour déterminer les catégories (NORMAL, ELEVATED, STAGE1_HYPERTENSION, STAGE2_HYPERTENSION, HYPERTENSIVE_CRISIS).
- **Mettre en œuvre une stratégie de stockage multi-niveaux :** Utiliser Elasticsearch pour l'analyse historique, un cache CSV pour le dashboard temps réel, et un archivage JSON pour les données normales.
- **Développer un tableau de bord analytique avec Kibana :** Créer des visualisations interactives pour l'analyse des tendances et des patterns dans les données historiques.
- **Développer un dashboard web Flask avec monitoring en temps réel :** Implémenter une interface web sécurisée permettant la visualisation des données en temps réel, avec authentification utilisateur.
- **Intégrer un modèle de Machine Learning pour la prédiction des risques :** Entraîner et déployer un modèle XGBoost capable de prédire la probabilité de crise hypertensive à 1h, 6h et 24h, basé sur les 30 dernières observations d'un patient.
- **Implémenter un système d'alertes intelligent :** Générer automatiquement des alertes (CRITICAL, HIGH, MEDIUM, LOW) basées sur les probabilités prédites par le modèle ML.
- **Déployer l'ensemble dans un environnement conteneurisé via Docker :** Assurer la portabilité et la facilité de déploiement grâce à Docker Compose.

Ainsi, ce projet propose une architecture Big Data complète intégrant ingestion, traitement, stockage multi-niveaux, visualisation (Kibana et Flask), et intelligence artificielle appliquée à la surveillance médicale prédictive.

2 Architecture globale du système

2.1 Vue d'ensemble

L'architecture développée dans le cadre de ce projet repose sur un modèle de traitement en flux (*stream processing*), particulièrement adapté à la gestion continue de données médicales en temps réel.

Elle couvre l'ensemble du cycle de vie de la donnée, depuis sa génération jusqu'à sa visualisation, en passant par son ingestion, son analyse et son stockage différencié.

L'objectif principal de cette architecture est de garantir :

— Une ingestion fiable et continue des données médicales ;
— Une catégorisation automatique selon les standards médicaux (AHA) ;
— Une séparation des données selon leur type (normales vs nécessitant surveillance) ;
— Une visualisation analytique et décisionnelle adaptée ;
— Une prédiction des risques futurs via Machine Learning.

La Figure 2.1 présente l'architecture globale du système.

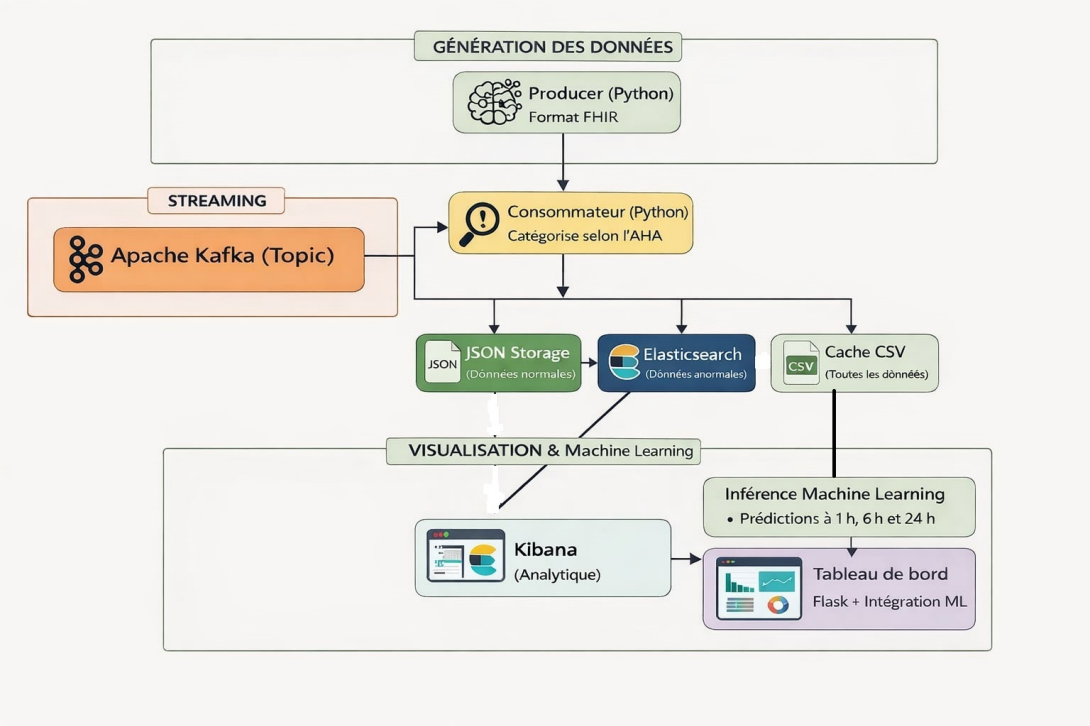


FIGURE 2.1 – Architecture globale du système de traitement en temps réel

Cette architecture s'articule autour des composants suivants :

— Module de génération des données médicales (format FHIR) ;
— Plateforme de streaming Apache Kafka ;
— Module de traitement et de catégorisation médicale ;
— Système de stockage multi-niveaux (JSON, CSV, Elasticsearch) ;
— Couche de Machine Learning pour prédictions ;

— Couche de visualisation et de monitoring (Kibana et Flask) ;
— Infrastructure de déploiement conteneurisée.

Chaque composant joue un rôle spécifique dans le pipeline global, garantissant une organisation modulaire, évolutive et robuste.

2.2 Description des composants

2.2.1 Génération des données (format FHIR)

Le système débute par un module de simulation générant des observations médicales de pression artérielle.

Les données produites respectent le standard FHIR (*Fast Healthcare Interoperability Resources*), assurant une structuration compatible avec les systèmes d'information hospitaliers.

Chaque observation comprend :
— Un identifiant patient ;
— Une pression systolique ;
— Une pression diastolique ;
— Un rythme cardiaque ;
— Un horodatage ;
— Un identifiant d'appareil de mesure.

Ces données sont sérialisées au format JSON avant leur transmission vers la plateforme de streaming.

Le générateur simule 50 patients avec des profils cohérents (baseline physiologique, tendances évolutives) et une distribution de risque réaliste.

2.2.2 Ingestion et streaming avec Apache Kafka

Apache Kafka constitue le cœur du système de communication inter-composants. Il assure la gestion distribuée et asynchrone des flux de données médicaux.

L'architecture d'ingestion repose sur deux entités principales :
— **Producer Kafka :** responsable de l'envoi continu des observations vers un topic dédié (*blood-pressure-data*) ;
— **Consumer Kafka :** chargé de la lecture des messages et de leur traitement en temps réel.

Cette organisation permet de reproduire un environnement hospitalier dans lequel les mesures physiologiques sont transmises de manière continue (toutes les 15 minutes par patient).

2.2.3 Module de traitement et catégorisation médicale

Le module de traitement est intégré au Consumer Kafka. Il applique une stratégie d'analyse en deux étapes :
— **Catégorisation selon les standards AHA :** classification automatique des mesures selon les guidelines de l'American Heart Association (NORMAL, ELEVATED, STAGE1_HYPERTENSIO
STAGE2_HYPERTENSION, HYPERTENSIVE_CRISIS) ;

— **Routage selon la catégorie :**
  — Données NORMAL → Archivage JSON ;
  — Données non-NORMAL → Indexation Elasticsearch ;
  — Toutes les données → Cache CSV (dashboard Flask).

Cette combinaison permet une organisation efficace du stockage tout en garantissant un accès rapide aux données critiques.

2.2.4 Stockage multi-niveaux des données

La stratégie de stockage repose sur trois niveaux complémentaires :

— **JSON (archivage) :** Les données normales sont archivées sous forme de fichiers JSON locaux (*blood_pressure_normal.json*) ;
— **CSV (cache) :** Toutes les données sont sauvegardées dans un fichier CSV (*blood_pressure_data.csv*) servant de cache pour le dashboard Flask ;
— **Elasticsearch (indexation) :** Les données nécessitant une surveillance sont indexées dans Elasticsearch pour l'analyse historique et la visualisation via Kibana.

Ce choix permet :

— Une optimisation des ressources de stockage ;
— Une indexation rapide des cas critiques ;
— Une exploitation analytique efficace via Kibana ;
— Un accès rapide pour le dashboard Flask.

2.2.5 Couche de Machine Learning

Le système intègre un module de prédiction ML indépendant du pipeline de streaming.

Son fonctionnement est le suivant :

— Lecture des données depuis le cache CSV ;
— Extraction des 30 dernières observations de chaque patient ;
— Génération des features temporelles ;
— Prédiction via un modèle XGBoost entraîné (probabilité de crise hypertensive à 1h, 6h et 24h) ;
— Conversion des probabilités en niveaux d'alerte (CRITICAL, HIGH, MEDIUM, LOW).

Le module ML est appelé par l'application Flask lors des requêtes de prédiction, permettant une visualisation en temps réel des risques futurs.

2.2.6 Couche de visualisation

La couche de visualisation comprend deux niveaux complémentaires :

— **Kibana :** analyse exploratoire et supervision en temps réel des données indexées dans Elasticsearch, via des dashboards interactifs ;
— **Application web Flask :** interface de monitoring en temps réel avec authentification utilisateur, fournissant :
  — Visualisation des données récentes ;
  — Prédictions ML intégrées (risques 1h, 6h, 24h) ;

— Système d'alertes basé sur les probabilités ML ;
— Filtres par patient et par date.

Cette séparation distingue la visualisation analytique technique (Kibana) de la visualisation décisionnelle orientée utilisateur (Flask).

2.3 Déploiement et conteneurisation

L'ensemble de l'infrastructure est déployé à l'aide de conteneurs Docker afin de garantir la portabilité et la reproductibilité du système.

Chaque service (Zookeeper, Kafka, Producer, Consumer, Elasticsearch, Kibana, Flask Dashboard) est isolé dans un conteneur distinct et orchestré via Docker Compose.

Cette approche présente plusieurs avantages :
— Isolation des dépendances ;
— Déploiement simplifié ;
— Scalabilité ;
— Reproductibilité de l'environnement sur différentes machines.

Ainsi, l'architecture proposée respecte les standards actuels des systèmes distribués Big Data et constitue une base robuste pour une application de surveillance médicale en temps réel intégrant des prédictions basées sur le Machine Learning.

3 Pipeline de traitement des données

3.1 Simulation des données médicales

Le pipeline débute par la génération de données médicales simulées représentant des mesures de pression artérielle pour un ensemble de patients suivis.

Caractéristiques de la simulation

- **Nombre de patients :** 50 patients actifs simultanément ;
- **Intervalle de génération :** une nouvelle observation toutes les 15 minutes par patient (rotation *round-robin*) ;
- **Profils patients cohérents :** chaque patient possède un profil unique comprenant :
  - Une baseline de pression artérielle (systolique et diastolique) ;
  - Une tendance évolutive (amélioration, détérioration ou stabilité) ;
  - Une baseline de rythme cardiaque ;
  - Un identifiant d'appareil de mesure.

Distribution de risque des patients

- 10 % de patients à risque critique ;
- 20 % de patients à risque élevé ;
- 30 % de patients à risque moyen ;
- 40 % de patients à risque faible.

Format des données

Les observations sont structurées selon le standard FHIR (*Fast Healthcare Interoperability Resources*) et contiennent :

- Identifiant unique de l'observation (UUID) ;
- Identifiant du patient (format : *PATIENT_XXXX*) ;
- Pression systolique (mmHg) ;
- Pression diastolique (mmHg) ;
- Rythme cardiaque (bpm) ;
- Identifiant de l'appareil (format : *DEVICE_X*) ;
- Horodatage (ISO 8601).

Les données sont sérialisées au format JSON avant leur transmission au système de streaming via le Producer Kafka.

Cette simulation permet de tester le comportement du système dans un contexte de flux continu et réaliste, avec des profils patients cohérents évoluant dans le temps.

3.2 Streaming temps réel avec Kafka

Une fois générées, les données sont transmises via Apache Kafka, qui joue le rôle de broker central pour le streaming distribué.

Configuration Kafka

- **Topic :** *blood-pressure-data* ;
- **Producer :** publication continue des observations FHIR ;
- **Consumer :** groupe *blood-pressure-consumer-group* ;
- **Format des messages :** JSON ;
- **Clé de partition :** identifiant patient (garantit la cohérence par patient).

Mécanisme de génération

Le Producer applique une stratégie de rotation *round-robin* : chaque patient génère une observation à tour de rôle, garantissant une distribution équilibrée des données dans le temps.

Avantages de l'architecture Kafka

- Ingestion asynchrone et distribuée ;
- Tolérance aux pannes (persistance des messages) ;
- Scalabilité via partitions multiples ;
- Découplage entre production et consommation.

Kafka garantit ainsi une architecture robuste adaptée aux environnements Big Data à fort débit et faible latence.

3.3 Catégorisation médicale selon les standards AHA

Dès la réception d'un message par le Consumer, une catégorisation est effectuée selon les guidelines de l'American Heart Association (AHA).

Le Consumer extrait les valeurs systoliques et diastoliques et applique les règles suivantes :

- **HYPERTENSIVE_CRISIS :**
  - Systolique > 180 mmHg **ou** Diastolique > 120 mmHg ;
  - Consultation médicale immédiate requise.
- **STAGE2_HYPERTENSION :**
  - Systolique ≥ 140 mmHg **ou** Diastolique ≥ 90 mmHg.
- **STAGE1_HYPERTENSION :**
  - Systolique 130–139 mmHg **ou** Diastolique 80–89 mmHg.
- **ELEVATED :**
  - Systolique 120–129 mmHg **et** Diastolique < 80 mmHg.
- **NORMAL :**
  - Systolique < 120 mmHg **et** Diastolique < 80 mmHg.

Cette approche déterministe garantit une classification immédiate, interprétable et conforme aux standards médicaux reconnus.

3.4 Gestion du stockage multi-niveaux

Après catégorisation, le système applique une stratégie de stockage différenciée.

Routage des données

1. Données NORMAL → Archivage JSON

— Fichier : *blood_pressure_normal.json* ;

— Format : tableau JSON ;

— Objectif : archivage léger des données non-critiques.

2. Données non-NORMAL → Elasticsearch

— Index : *blood-pressure-observations* ;

— Format : documents JSON avec mapping optimisé ;

— Objectif : analyse historique et visualisation via Kibana.

3. Toutes les données → Cache CSV

— Fichier : *blood_pressure_data.csv* ;

— Colonnes : (observation_id, timestamp, patient_id, systolic, diastolic, heart_rate, status, device_id) ;

— Objectif : accès rapide pour le dashboard Flask et le module ML.

Avantages de la stratégie

— Optimisation des ressources de stockage ;

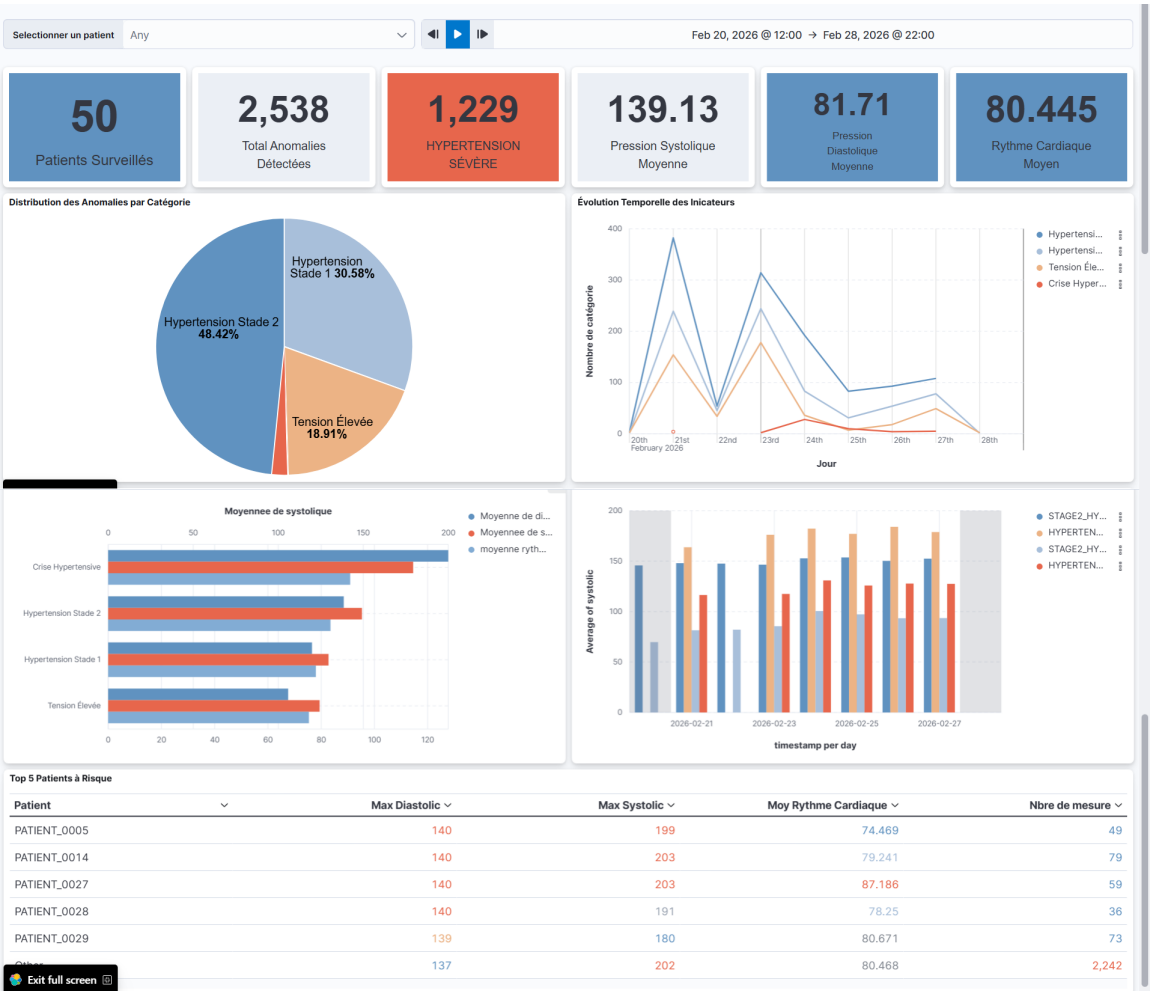
— Indexation ciblée des cas critiques ;

— Performance du dashboard via cache local ;

— Support direct pour le module de prédiction ML.

Ainsi, le pipeline assure un traitement fluide des flux entrants tout en optimisant le stockage et en garantissant un accès rapide aux données critiques ainsi qu'aux données nécessaires aux prédictions basées sur le Machine Learning.

4 Visualisation avec Kibana



Patient	Max Diastolic	Max Systolic	Moy Rythme Cardiaque	Nbre de mesure
PATIENT_0005	140	199	74.469	49
PATIENT_0014	140	203	79.241	79
PATIENT_0027	140	203	87.186	59
PATIENT_0028	140	191	78.25	36
PATIENT_0029	139	180	80.671	73
	137	202	80.468	2,242

FIGURE 4.1 – Dashboard Kibana de surveillance et d'analyse des anomalies d'hypertension

Le dashboard Kibana constitue l'interface principale de supervision du système de surveillance de la pression artérielle. Il permet aux professionnels de santé d'accéder en temps réel à une vision synthétique et analytique de l'état cardiovasculaire des patients suivis.

L'organisation visuelle repose sur une structure hiérarchique en trois niveaux :

— Une ligne supérieure d'indicateurs clés (KPIs) pour une lecture immédiate de la situation globale ;

— Des graphiques analytiques permettant l'étude des tendances temporelles et de la répartition des anomalies ;

— Un tableau détaillé identifiant les patients les plus à risque.

Des filtres permettent de sélectionner une période temporelle ou un patient particulier.

Le tableau de bord est actualisé automatiquement toutes les dix secondes, garantissant une surveillance continue et une réactivité rapide face à l'apparition d'anomalies critiques.

4.1 Indicateurs clés de performance (KPIs)

FIGURE 4.2 – Indicateurs clés de performance (KPI)

La première ligne du dashboard présente six métriques essentielles :

— **Patients surveillés (50)** : agrégation *unique count* sur l'identifiant patient.

— **Total des anomalies détectées (2 290)** : nombre total d'observations anormales indexées dans Elasticsearch.

— **Hypertension sévère (1 118)** : nombre d'observations classées en STAGE2_HYPERTENSION.

— **Pression systolique moyenne (139 mmHg)** : moyenne des valeurs systoliques des anomalies.

— **Pression diastolique moyenne (81 mmHg)** : moyenne des valeurs diastoliques.

— **Rythme cardiaque moyen (80 bpm)** : moyenne du rythme cardiaque.

4.2 Visualisations analytiques

4.2.1 Distribution des anomalies par catégorie

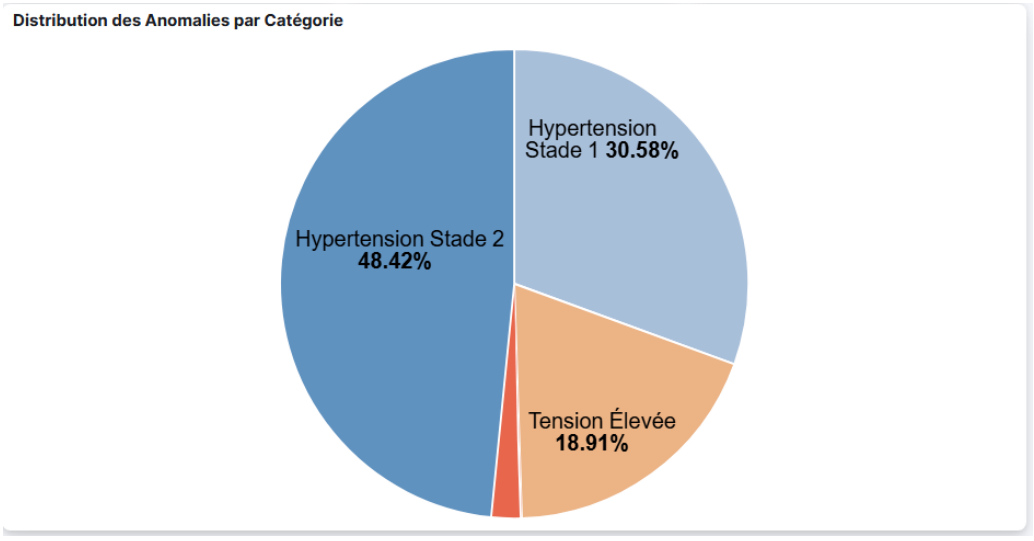


FIGURE 4.3 – Distribution des anomalies par catégories

Cette visualisation montre une prédominance de l'hypertension de stade 2, suivie du stade 1, avec une part plus réduite de tension élevée et un faible pourcentage de crises hypertensives.

La représentation en pourcentage facilite la lecture rapide et la communication clinique.

4.2.2 Évolution temporelle des anomalies

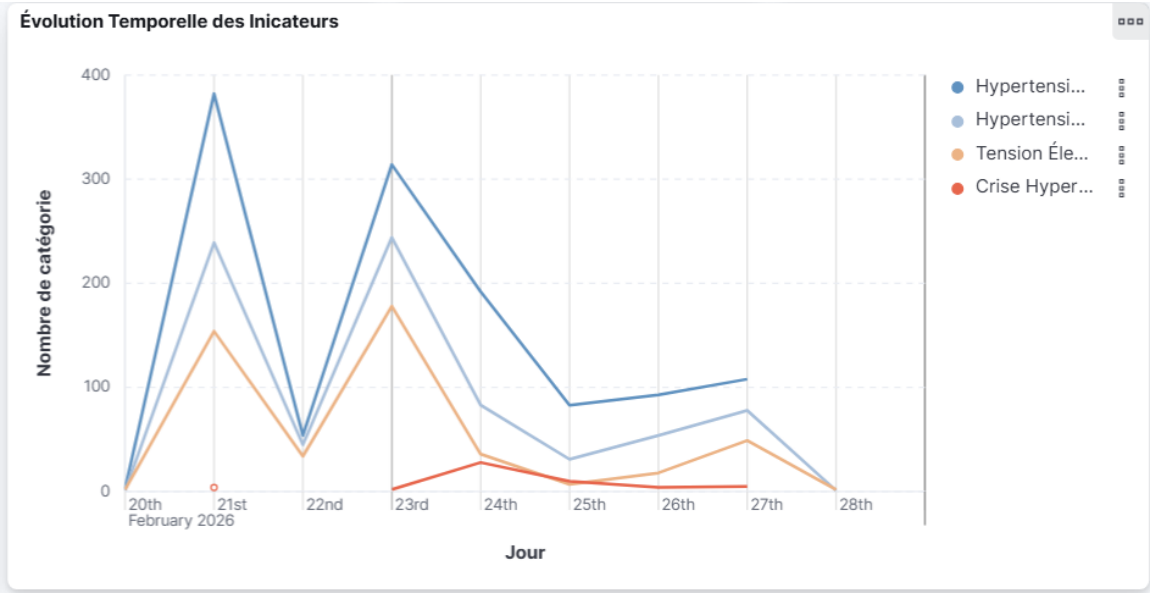


FIGURE 4.4 – Évolution temporelle des indicateurs

Le graphique met en évidence :

— Des pics simultanés pour toutes les catégories à certaines dates ;

— Une décroissance progressive en fin de période ;

— Une évolution parallèle des différentes catégories.

Cette synchronisation suggère des facteurs communs influençant les mesures.

4.2.3 Analyse circadienne – pics de pression par heure

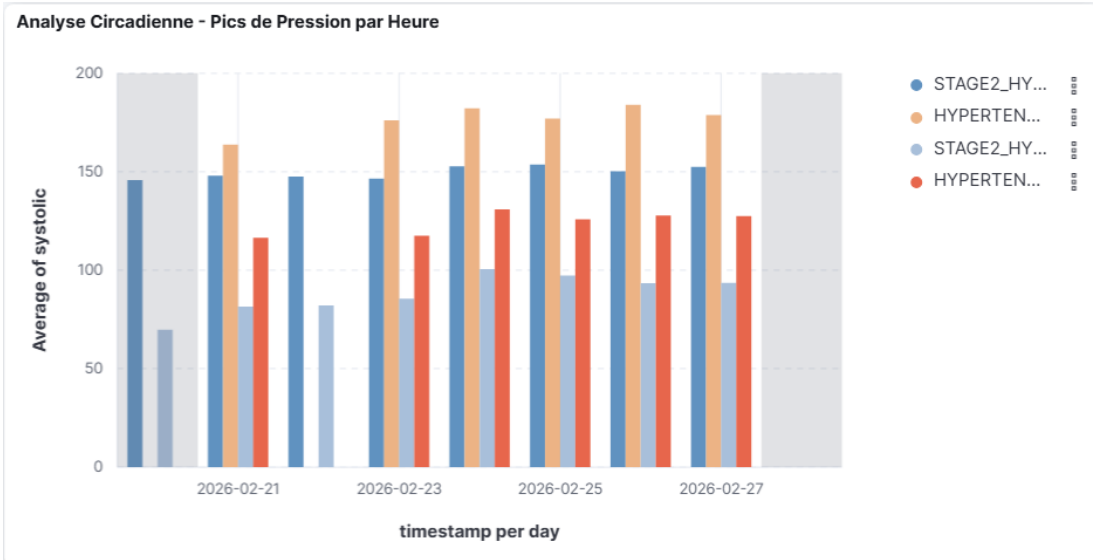


FIGURE 4.5 – Analyse circadienne – pics de pression par heure

Les observations montrent :

— Une réduction des mesures pendant la nuit ;
— Des pics matinaux ;
— Une stabilité relative de la moyenne systolique.

Ces résultats correspondent au phénomène physiologique du *morning surge*.

4.2.4 Moyennes des indicateurs par catégorie

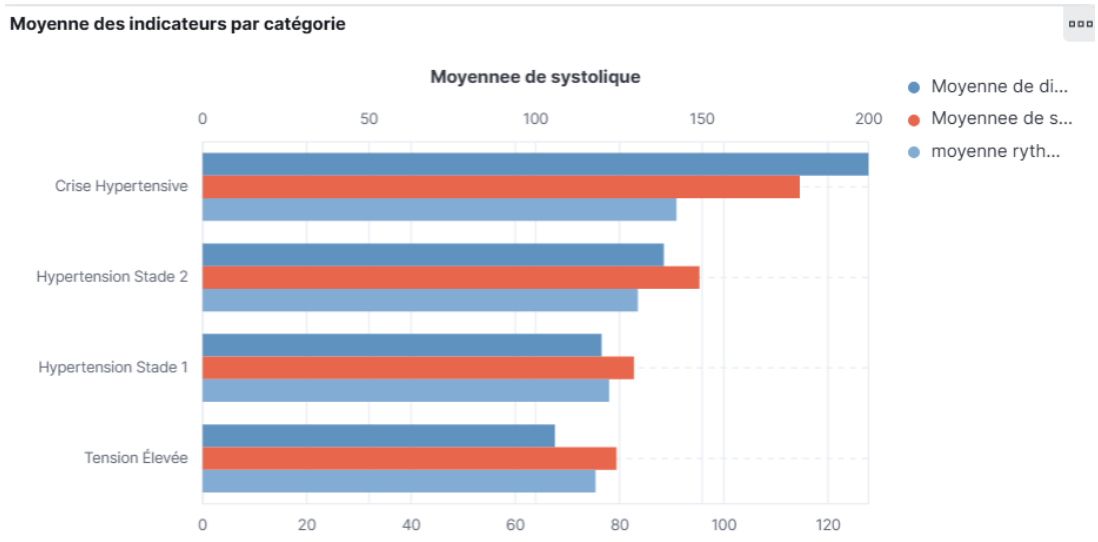


FIGURE 4.6 – Moyennes des indicateurs par catégorie

On observe une progression cohérente des valeurs moyennes :
— Valeurs maximales pour les crises hypertensives ;
— Valeurs intermédiaires pour les stades 1 et 2 ;
— Valeurs plus faibles pour la catégorie ELEVATED.

Cette hiérarchie confirme la cohérence de la classification automatique.

4.2.5 Top 5 patients à risque

Top 5 Patients à Risque

Patient	Max Diastolic	Max Systolic	Moy Rythme Cardiaque	Nbre de mesure
PATIENT_0005	140	199	74.469	49
PATIENT_0014	140	203	79.241	79
PATIENT_0027	140	203	87.186	59
PATIENT_0028	140	191	78.25	36
PATIENT_0029	139	180	80.671	73
	137	202	80.468	2,242

Exit full screen

FIGURE 4.7 – Top 5 patients à risque

Le tableau présente pour chaque patient :
— La pression systolique maximale ;
— La pression diastolique maximale ;
— Le rythme cardiaque moyen ;

— Le nombre total de mesures enregistrées.

Cette vue permet de prioriser les patients nécessitant une prise en charge immédiate et d'orienter les décisions médicales vers les profils les plus critiques.

Ce tableau de bord illustre l'intégration réussie des technologies Big Data (Kafka, Elasticsearch, Kibana) avec des standards médicaux (FHIR et recommandations cliniques) dans une logique de médecine numérique et de surveillance proactive.

5 Application Web et Intelligence Artificielle

5.1 Objectif de l'intégration du Machine Learning

Le système de surveillance développé repose initialement sur une catégorisation médicale basée sur les standards AHA, permettant une classification immédiate des mesures selon des règles déterministes.

Afin d'enrichir cette approche et d'apporter une dimension prédictive, un module de Machine Learning supervisé a été intégré.

L'objectif principal est d'anticiper la probabilité de survenue d'une crise hypertensive à différents horizons temporels (1h, 6h, 24h), permettant une surveillance proactive avant que les seuils critiques ne soient atteints.

5.2 Génération des données d'entraînement

Le dataset d'entraînement a été généré synthétiquement afin de simuler des trajectoires physiologiques réalistes.

Caractéristiques

— 5 000 séquences temporelles ;
— 30 observations successives par séquence ;
— Distribution : 40% LOW, 40% MEDIUM, 20% HIGH.

Chaque observation contient :

— Pression systolique ;
— Pression diastolique ;
— Rythme cardiaque ;
— Heure normalisée.

Les variables cibles continues sont :

— *critical_risk_1h*
— *critical_risk_6h*
— *critical_risk_24h*

Le problème est formulé comme une régression multi-sortie (120 features, 3 targets).

5.3 Modélisation et comparaison des algorithmes

Deux modèles ont été évalués :

— Random Forest ;
— XGBoost.

Split : 80% entraînement, 20% test.

Métriques : MAE, RMSE, R^2.

Comparaison des performances

— Risque 1h : XGBoost (0.802) > Random Forest (0.792)
— Risque 6h : XGBoost (0.860) > Random Forest (0.848)
— Risque 24h : XGBoost (0.872) > Random Forest (0.868)

XGBoost présente :

— MAE plus faible ;
— RMSE plus faible ;
— Meilleure stabilité inter-horizons ;
— Meilleure performance court et moyen terme.

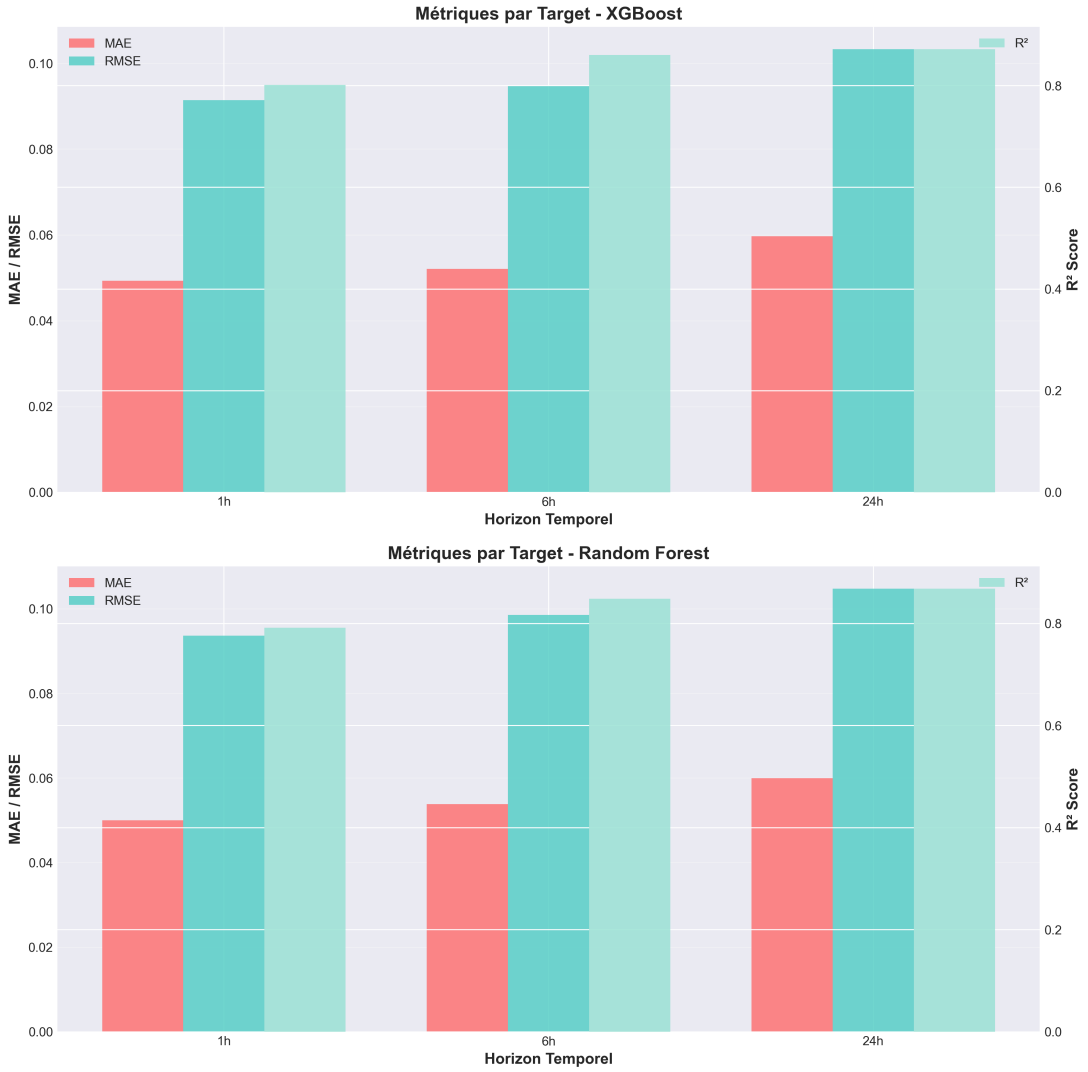


FIGURE 5.1 – Comparaison des métriques – XGBoost vs Random Forest

XGBoost a donc été retenu comme modèle final de production.

5.4 Inférence et détermination du niveau d'alerte

Processus :

— Lecture des 30 dernières observations ;
— Vectorisation (120 features) ;
— Prédiction des 3 probabilités ;
— Clamp entre 0 et 1.

Niveaux d'alerte (basé sur risque 6h)

— CRITICAL : $\geq 60\%$
— HIGH : $\geq 40\%$
— MEDIUM : $\geq 20\%$
— LOW : $< 20\%$

5.5 Dashboard Flask

5.5.1 Dashboard principal – Monitoring temps réel

FIGURE 5.2 – Interface principale du dashboard Flask – Monitoring temps réel

Indicateurs affichés :
— Total mesures ;
— Moyenne systolique ;
— Moyenne diastolique ;
— Moyenne rythme cardiaque ;
— Nombre mesures normales ;
— Nombre cas hypertensifs.

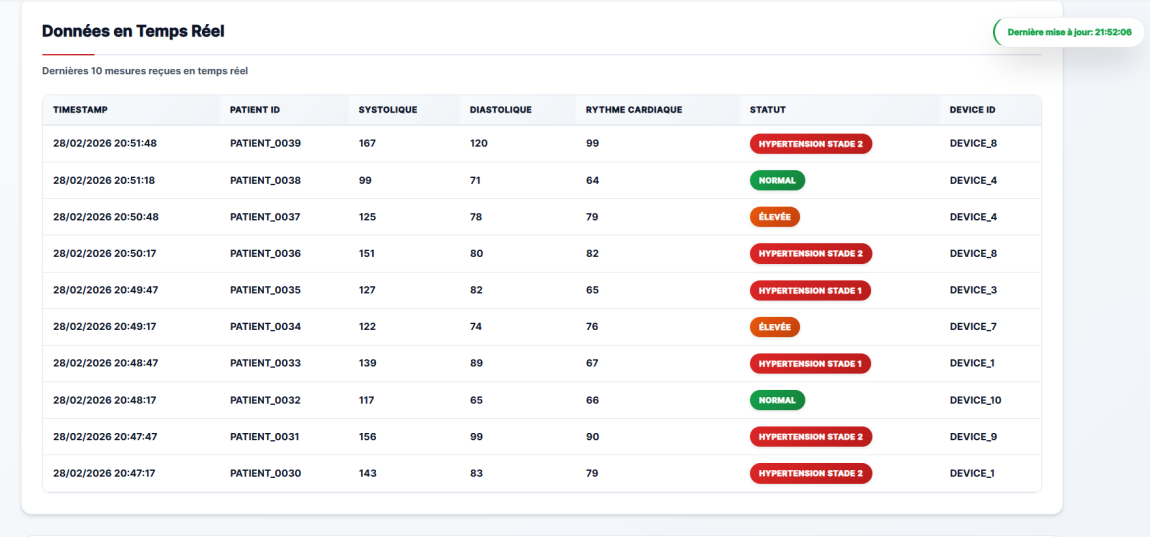


FIGURE 5.3 – Tableau des données en temps réel – Dernières mesures reçues

Code couleur statut :

— Vert : NORMAL

— Jaune : ELEVATED

— Orange : STAGE1

— Rouge : STAGE2

— Rouge foncé : HYPERTENSIVE_CRISIS

5.5.2 Interface des prédictions ML

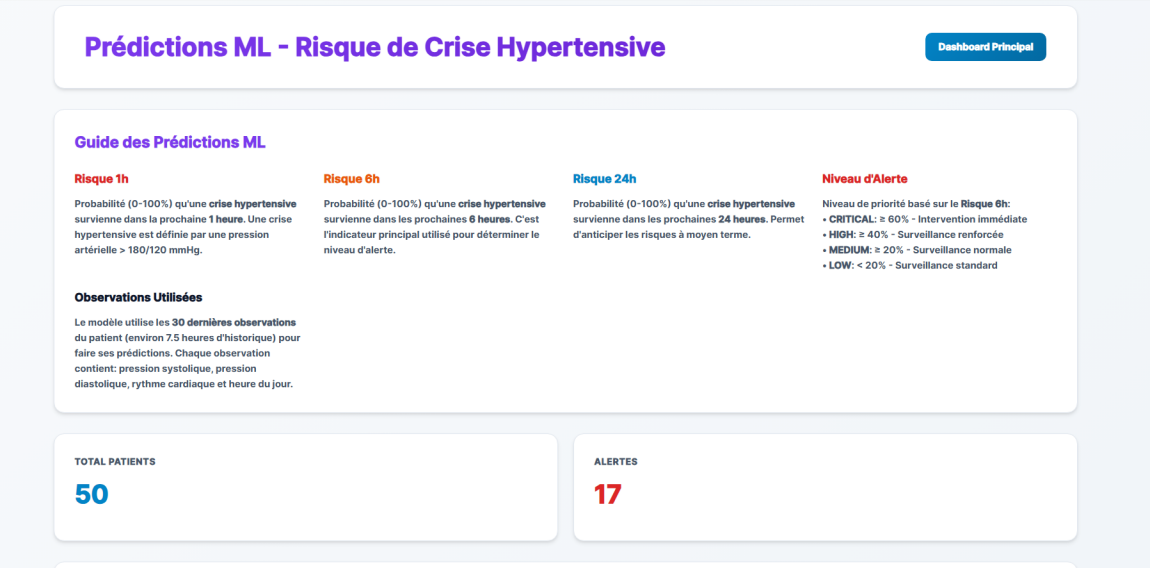


FIGURE 5.4 – Interface globale des prédictions Machine Learning

Vue globale :

— Nombre total patients ;

— Nombre alertes CRITICAL/HIGH.

Seuils utilisés :

— CRITICAL $\geq 60\%$

— HIGH $\geq 40\%$

— MEDIUM $\geq 20\%$

— LOW $< 20\%$

Prédictions par Patient

PATIENT	RISQUE 1H	RISQUE 6H	RISQUE 24H	NIVEAU ALERTE	OBSERVATIONS
PATIENT_0001	49.8%	56.5%	53.2%	HIGH	98
PATIENT_0002	38.3%	36.8%	32.9%	MEDIUM	98
PATIENT_0003	48.5%	47.3%	48.5%	HIGH	96
PATIENT_0004	45.0%	51.3%	51.5%	HIGH	94
PATIENT_0005	43.5%	45.7%	44.8%	HIGH	93
PATIENT_0006	41.4%	35.5%	32.3%	MEDIUM	91
PATIENT_0007	36.8%	37.0%	30.8%	MEDIUM	90
PATIENT_0008	37.9%	30.7%	36.5%	MEDIUM	90
PATIENT_0009	40.9%	38.0%	36.3%	MEDIUM	90
PATIENT_0010	37.0%	32.0%	28.9%	MEDIUM	90
PATIENT_0011	40.6%	42.1%	48.3%	HIGH	89

FIGURE 5.5 – Tableau détaillé des prédictions individuelles

Chaque ligne affiche :

— Patient ID ;

— Risque 1h ;

— Risque 6h ;

— Risque 24h ;

— Niveau d'alerte ;

— Nombre d'observations utilisées.

5.5.3 Génération automatique d'alertes

Le système extrait automatiquement les patients CRITICAL ou HIGH.

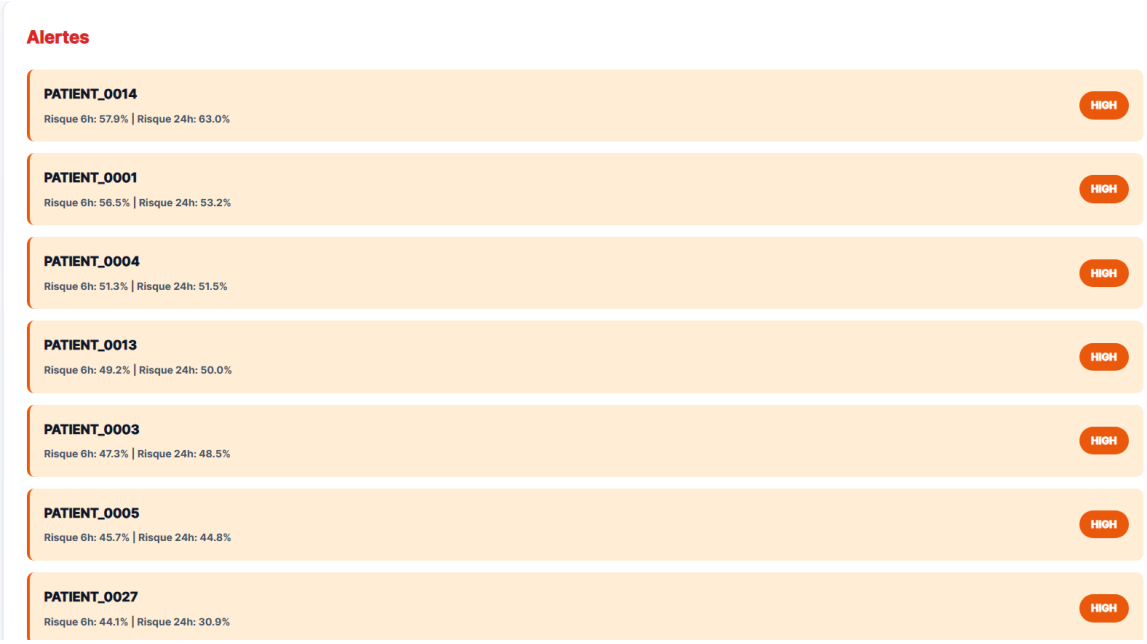


FIGURE 5.6 – Liste des alertes critiques et élevées générées par le modèle

Les alertes sont :

— Triées par risque 6h décroissant ;

— Affichées avec badge coloré ;

— Les CRITICAL déclenchent une mise en évidence visuelle.

Ce mécanisme transforme la prédiction probabiliste en outil opérationnel d'aide à la décision clinique, permettant priorisation et réactivité.

6 Conclusion

Ce projet avait pour objectif de concevoir et d'implémenter une architecture Big Data appliquée à la surveillance médicale en temps réel, plus précisément au suivi de la pression artérielle. Face à l'augmentation du volume et de la vélocité des données issues des dispositifs médicaux connectés, il était nécessaire de proposer une solution capable d'ingérer, traiter, analyser et visualiser ces flux de manière fiable, tout en intégrant une dimension prédictive basée sur le Machine Learning.

L'architecture développée couvre l'ensemble du cycle de vie de la donnée. La génération de mesures physiologiques au format FHIR a permis de simuler un environnement clinique réaliste impliquant 50 patients. L'ingestion via Apache Kafka assure un traitement distribué et asynchrone des flux en temps réel. Le Consumer Kafka applique une catégorisation immédiate selon les standards AHA, fournissant une première interprétation médicale standardisée et immédiatement exploitable.

La stratégie de stockage multi-niveaux optimise l'utilisation des ressources : archivage JSON pour les données normales, indexation Elasticsearch pour les cas nécessitant une surveillance, et cache CSV pour l'accès rapide par le dashboard Flask et le module de Machine Learning. Cette organisation garantit à la fois performance, scalabilité et efficacité analytique.

L'intégration de Kibana et du dashboard Flask permet une double approche complémentaire : une analyse exploratoire technique via Elasticsearch et Kibana, et une interface décisionnelle orientée utilisateur via Flask, intégrant monitoring en temps réel et système d'alertes visuelles.

L'ajout du modèle XGBoost, sélectionné après comparaison expérimentale avec Random Forest, apporte une dimension prédictive avancée. Les performances obtenues (R^2 compris entre 0.802 et 0.872 selon l'horizon temporel) démontrent la capacité du modèle à anticiper les risques de crise hypertensive jusqu'à 24 heures à l'avance. La transformation des probabilités continues en niveaux d'alerte discrets (CRITICAL, HIGH, MEDIUM, LOW) rend ces prédictions directement exploitables dans un contexte opérationnel.

Le déploiement via Docker Compose assure la portabilité, la reproductibilité et la modularité du système, conformément aux standards actuels des architectures distribuées modernes.

En définitive, ce projet illustre la convergence entre technologies Big Data, visualisation interactive et intelligence artificielle au service de la santé numérique. L'architecture proposée dépasse le cadre d'une simple simulation académique : elle constitue un prototype fonctionnel de surveillance médicale prédictive, combinant catégorisation standardisée, prédiction avancée et interfaces décisionnelles adaptées aux environnements cliniques modernes.