medx Healthcare system design

AWY

SUPINFO

5DESI

# **Introduction**

**MedX** est une plateforme de santé innovante conçue pour faciliter l’accès aux soins médicaux grâce aux technologies numériques. Elle propose un ensemble de services intégrés visant à simplifier le parcours de soin des patients tout en soutenant les professionnels de santé dans leur pratique quotidienne.

Parmi ses principales fonctionnalités, MedX offre la possibilité de réaliser des téléconsultations médicales, permettant aux patients de consulter un médecin à distance, rapidement et en toute sécurité, sans avoir à se déplacer.

La plateforme intègre également un système de gestion sécurisée des dossiers médicaux, garantissant la confidentialité des données de santé tout en assurant leur accessibilité aux professionnels autorisés. Cela favorise une meilleure coordination entre les différents acteurs du système de santé.

Enfin, MedX permet le suivi des prescriptions médicales par les pharmacies, en assurant une transmission claire et instantanée des ordonnances, ce qui renforce la sécurité des traitements et facilite leur gestion pour les patients.

Grâce à cette approche connectée, MedX ambitionne de moderniser l’accès aux soins et d’optimiser l’expérience santé pour tous.

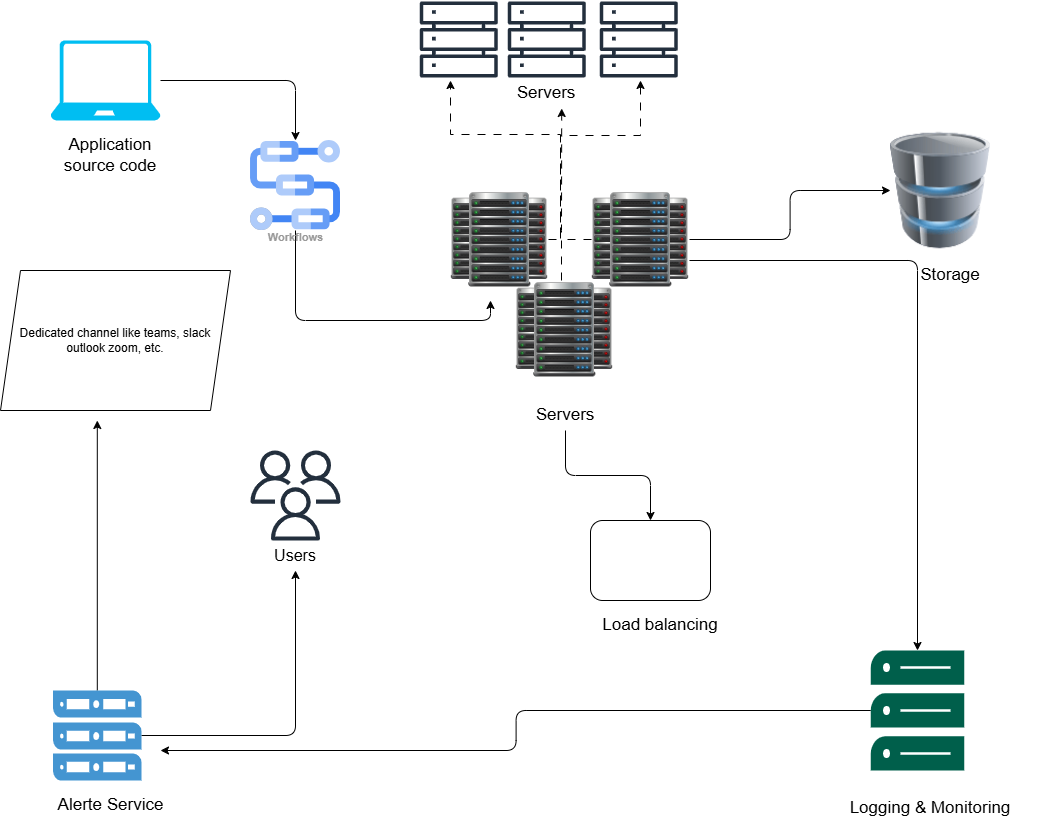
Ce document présente la solution technique complète pour MedX, incluant :

* L'architecture du système
* La conception détaillée des composants
* Les stratégies de données et stockage
* Les considérations de scalabilité et performance
* Les mesures de sécurité et conformité réglementaire.

## **Architecture Globale**

### **Vu d’ensemble du système**

Le diagramme ci-dessous présente l’application dans son ensemble, présentant ainsi comment elle fonction et les différents composant essentielles à son fonctionnement



**Figure 1**: Architecture globale de MEDX

En premier lieu, nous avons le code source de l'application, développé par l'équipe et versionné dans un repository Git, grâce à un pipeline CI/CD automatisé (via GitHub Actions et Jenkins), le code est testé, validé et déployé automatiquement sur nos serveurs de production. Les serveurs d'application ont été répliqués sur plusieurs instances pour garantir la haute disponibilité et pallier d'éventuelles interruptions de service. Un load balancer répartit intelligemment la charge entre ces serveurs. Les données fournies et produit par les utilisateurs au cours de l’utilisation du logiciel sont stockées dans une base de données sécurisée avec chiffrement. Pour la bonne gestion des erreurs, les logs sont stockés sur un serveur de monitoring dédié, séparé de la base de données principale. Un serveur de messagerie assure les notifications tant pour les développeurs (alertes techniques) que pour les utilisateurs (rappels, confirmations).

L'ensemble du système communique via des connexions chiffrées TLS pour garantir la sécurité des données médicales sensibles.

## **Composant détaillée**

### **Couche Application**

La couche application est représenté dans le diagramme ci-dessus par le « **1** » elle contient le code source de la solution, qui a suivi une découpe en micro-service. Nous avons exactement 8 microservices, dans les prochaines lignes nous tâcherons d’expliquer nous choix pour chaque microservices.

#### **User Management Service**

Cette couche est Responsable de :

* Gestion des inscriptions et profils (patients, médecins, pharmaciens)
* Authentification multi-facteurs (MFA avec TOTP)
* Autorisation basée sur les rôles (RBAC) ex :**Patient**, **Médecin,** **Pharmacien**, **Administrateur, Urgentiste**
* Gestion des sessions et token JWT

Les principales technologies utilisées sont :

* Backend : Node.js avec Express.js
* Base de données : PostgreSQL (données structurées, ACID)
* Authentification : OAuth 2.0 + JWT
* MFA : TOTP (Google Authenticator) + SMS backup

#### **Teleconsultation Scheduling Service**

Cette couche est responsable de la :

* Réservation, modification, annulation de consultations
* Gestion des disponibilités médecins
* Notifications automatiques (rappels)
* Gestion liste d'attente et priorités urgentes
* Intégration calendriers externes (Google Calendar, Outlook)

Les principales technologies utilisées sont :

* Backend : Python avec FastApi
* Base de données : PostgreSQL (intégrité transactionnelle)
* Authentification : OAuth 2.0 + JWT
* Cache : Redis pour disponibilités temps réel
* Calendrier : Intégration CalDAV / Google Calendar API

#### **Consultation Service**

Elle prend en compte la :

* Création de salles de consultation virtuelles
* Gestion streaming audio/vidéo sécurisé(WebRTC)
* Chat en temps réel pendant consultation
* Partage de documents médicaux
* Enregistrement des sessions (avec consentement)

Pour les technologies :

* Serveur média : Jitsi Meet (self-hosted) ou Twilio
* Backend : Node.js avec Socket.io pour chat
* Signaling : WebSocket sécurisé (WSS)
* Stockage enregistrements : S3 avec chiffrement (uniquement en opt-in)

En guise de mésure de sécurité les salles sont avec tokens JWT à usage unique, expiration automatique après consultation, chiffrement DTLS pour transport média, watermarking des flux vidéo (identification des captures)

#### **Patient Records Management Service**

Elle est chargée de :

* Stockage sécurisé des dossiers médicaux
* Gestion de l'historique des consultations
* Contrôle d'accès granulaire aux données patients
* Versioning des modifications
* Accès d'urgence pour professionnels autorisés
* Conformité RGPD/HIPAA

Il est attendue des patients :

* Informations démographiques (nom, date naissance, contact)
* Antécédents médicaux
* Allergies et contre-indications
* Historique des consultations
* Résultats d'examens
* Documents médicaux (images, PDF)

Pour les technologies :

* Backend : Java Spring Boot
* Base de données : PostgreSQL (données structurées)
* Stockage documents : AWS S3 avec chiffrement côté serveur
* Chiffrement : AES-256-GCM pour données au repos
* Versioning : Table d'audit avec triggers PostgreSQL

Elle a aussi des particularités comme un/une :

* **Schéma flexible** : MongoDB permet structure variable selon type de dossier
* **Auto-sauvegarde** : Intégration automatique avec Consultation Service
* **Filtrage par rôle** : Patients voient version simplifiée (non-technique)
* **Audit automatique** : Chaque accès enregistré avec timestamp et justification

#### **Prescription Management Service**

Elle est chargée de :

* Création de prescriptions par médecins
* Modification et révocation de prescriptions
* Notification automatique aux pharmacies
* Suivi du statut de délivrance
* Historique des prescriptions patient

Pour les technologies**:**

* Backend : Java Spring Boot
* Base de données : PostgreSQL
* Signature numérique : PKI avec certificats X.509
* QR codes : Bibliothèque ZXing

En guise de sécurité : les prescriptions sont non modifiables après création (sauf révocation), la signature numérique est obligatoire, audit trail complet, détection de fraudes (patterns anormaux).

#### **Pharmacy Interface Service**

Elle est chargée de :

* Confirmation réception prescriptions
* Mise à jour statuts préparation
* Vérification disponibilité stock,
* Communication avec patients

Comme techno elle utilise :

#### **Notification Service**

**Ce service fait le :**

* Envoi notifications multi-canal (SMS, Email, Push, In-app)
* Gestion préférences utilisateurs
* Tracking délivrance et lecture
* Template management

**Avec plusieurs types de notifications comme :**

* Rappels de RDV (24h, 1h avant)
* Nouvelle prescription disponible
* Prescription prête en pharmacie
* Accès d'urgence à dossier médical
* Résultats d'examens disponibles

**Et les technologies utilisées sont**

* Message Queue : RabbitMQ pour asynchrone
* Email : SendGrid / AWS SES
* SMS : Twilio
* Push : Firebase Cloud Messaging
* Backend : Node.js worker processes

#### **Billing and Payment Service**

**Cette couche est responsable du :**

* Traitement paiements consultations
* Génération factures
* Remboursements
* Intégration passerelles paiement

**Les technologies utilisées sont**

* Payment Gateway : Stripe / PayPal
* Compliance : PCI-DSS (via gateway externe)
* Base de données : PostgreSQL

### **Modèle de Données et Stockage**

Cette contient toutes les ressources produites et fournit par l’utilisateur au cours de l’utilisation de MedX.