Architecture du Projet PING

Pipeline EDS $\leftrightarrow$ FHIR + API

1 Objectif global du système

Le projet **PING** vise à concevoir une architecture complète permettant de gérer un flux bidirectionnel de données médicales entre deux mondes :

- un **Entrepôt de Données de Santé (EDS)** structuré en Parquet, jouant le rôle de référence locale,
- le standard international **HL7 FHIR**, utilisé pour l'échange et l'interopérabilité,
- une **API FastAPI**, qui orchestre l'ensemble des opérations.

L'objectif principal est d'assurer :

1. la production de données médicales synthétiques,
2. leur ingestion et transformation en un EDS propre et standardisé,
3. leur exportation vers le standard FHIR,
4. la réintégration de données FHIR externes dans l'EDS,
5. la fusion intelligente des données internes et externes,
6. la génération de rapports JSON et PDF,
7. et l'exposition de toutes ces opérations via une API unifiée.

2 Architecture globale du système

Cette section décrit l'architecture dans sa globalité, en présentant chaque composant, ses responsabilités et la logique de circulation des données, sans schéma visuel.

2.1 Production des données

Le point d'entrée du système est **Synthea**, un générateur open-source de données médicales synthétiques. Il simule :

- des patients,
- leurs diagnostics,
- leurs observations,
- leurs consultations,
- et leurs traitements.

Synthea produit deux types de sorties :

- des fichiers **CSV**, qui représentent les données sous forme tabulaire,
- des fichiers **FHIR JSON**, qui utilisent la norme HL7.

Ces données alimentent l'ensemble du pipeline PING.

2.2 Conversion et préparation des données

Les fichiers CSV sont convertis en **Parquet**, un format colonne efficace permettant :

- une lecture très rapide,
- une réduction de la taille,
- un typage explicite des colonnes,
- une compatibilité naturelle avec les outils analytiques modernes.

Les fichiers FHIR JSON, lorsqu'ils sont fournis par Synthea, peuvent également être prétraités pour faciliter leur ingestion.

2.3 Nettoyage et normalisation (Polars)

Le système utilise **Polars**, une bibliothèque de traitement de données hautement performante, pour nettoyer et normaliser les données converties :

- typage automatique,
- correction de valeurs incohérentes,

— harmonisation des noms de colonnes,
— standardisation des formats de dates,
— normalisation des unités cliniques.

Cette étape garantit que l'EDS reposera sur un socle fiable et propre.

2.4 Construction de l'EDS interne

L'**Entrepôt de Données de Santé (EDS)** constitue la structure interne centrale du projet. Il est organisé sous forme de tables Parquet :

— patient.parquet,
— observation.parquet,
— encounter.parquet,
— condition.parquet, etc.

L'EDS agit comme :

— un référentiel stable et local,
— une vue normalisée de l'ensemble des données,
— une base pour les transformations ultérieures.

3 Exportation de l'EDS vers FHIR

L'export représente la transformation des données internes en ressources FHIR standardisées.

3.1 Mapping EDS → FHIR

Un module de mapping lit les tables Parquet de l'EDS et les convertit en ressources FHIR telles que :

- **Patient**,
- **Observation**,
- **Encounter**,
- **Condition**.

Chaque ressource respecte la structure formelle imposée par HL7.

3.2 Construction des bundles FHIR

Les ressources générées sont regroupées en un **Bundle FHIR JSON**. Ce bundle représente une unité cohérente d'échange, pouvant contenir :

- un patient,
- toutes ses observations,
- toutes ses rencontres,
- ses diagnostics.

3.3 API Export

L'API expose l'endpoint :

`/export/fhir`

Il permet à un utilisateur ou un système tiers de récupérer un bundle FHIR généré à partir de l'EDS.

4 Importation de données FHIR dans l'EDS

Ce flux inverse permet de réintégrer des données externes dans l'EDS interne.

4.1 Réception via l'API

Les données FHIR externes sont envoyées au système via :

`/import/fhir`

Le corps de la requête contient un **bundle FHIR JSON**.

4.2 Parsing du bundle FHIR

Le système analyse chaque ressource :

- extraction des identifiants FHIR,
- reconstruction des relations (Patient → Observation → Encounter),
- validation de la structure des données,
- lecture des valeurs contenues dans les attributs FHIR.

4.3 Mapping FHIR → EDS

Chaque ressource FHIR est traduite en une structure tabulaire correspondant aux tables de l'EDS.

Par exemple :

- une ressource **Patient** devient une ligne dans `patient.parquet`,
- une **Observation** devient une ligne dans `observation.parquet`.

Cette étape garantit la compatibilité entre les données externes et internes.

4.4 Fusion intelligente des données

Lors de la réintégration, les situations suivantes doivent être gérées :

- patients déjà présents dans l'EDS,
- observations identiques ou contradictoires,
- dates incohérentes,
- valeurs mises à jour par le système externe.

La fusion applique des règles telles que :

— conserver la valeur la plus récente,
— créer une nouvelle entrée si nécessaire,
— mettre à jour une entrée existante,
— signaler les conflits irrésolus.

4.5 Mise à jour finale de l'EDS

Une fois fusionnées, les données sont réécrites proprement dans les tables Parquet. L'EDS devient alors une version enrichie et cohérente des données internes et externes.

4.6 Rapport JSON / PDF

Après chaque importation, le système génère un rapport accessible via :

`/report`

Ce rapport contient :

— un résumé de l'opération,
— les conflits détectés,
— les décisions de fusion,
— les éventuelles erreurs,
— des statistiques de mise à jour.

5 Rôle central de l'API FastAPI

L'API agit comme une façade unique. Elle expose :

— `/export/fhir` : exportation FHIR,
— `/import/fhir` : ingestion FHIR,
— `/report` : génération de rapport,
— `/stats` : métriques du système.

Elle permet :

— la séparation claire entre logique métier et interface,
— l'automatisation des transformations,
— la réutilisation des outils par des tiers.