**FHIR Data Pipeline**

**Overview The FHIR Data Pipeline is a Python-based ETL (Extract, Transform, Load) solution built on top of Apache Airflow. It automates the process of transforming healthcare claims data stored in CSV format into FHIR-compliant JSON resources, facilitating interoperability and standardized data exchange.**

**Project Objectives • Convert raw healthcare data into standardized FHIR (Fast Healthcare Interoperability Resources) format. • Automate the data pipeline using Apache Airflow. • Store the output in a structured JSON format for downstream analytics or API usage.**

**Project Structure**

**Project Structure**

**fhir-pipeline/**

**│**

**├── dags/**

**│ └── csv\_to\_fhir\_dag.py # DAG definition**

**│**

**├── data/**

**│ ├── input/ # Contains raw CSV files**

**│ └── output/ # Stores generated FHIR JSON**

**│**

**├── scripts/**

**│ ├── transform\_csv\_to\_fhir.py # Main transformation script**

**│ └── post\_to\_hapi.py # Script for posting it to HAPI**

**│**

**├── Dockerfile # Docker build file for Airflow image**

**├── docker-compose.yml # Containerized Airflow + HAPI setup**

**├── requirements.txt # Python dependencies**

**└── README.md # Project documentation**

**Tech Stack • Python 3.12 • Apache Airflow (Dockerized setup) • FHIR.resources (Python client for FHIR models) • Pandas (CSV handling and preprocessing) • Docker (for portability and ease of deployment)**

**Input Format Input is a CSV file with the following representative fields: CLAIM\_ID,MEMBER\_KEY,MEMBER\_ID,MEM\_GENDER,MEM\_DOB,MEM\_ZIP, AdmitDate,DischargeDate,Inpatient/ outpatient,AMT\_BILLED, PayerType,DRG\_CODE,DRG\_DESC,ICD\_DIAG1,ICD\_DIAG1\_DESC, proc1\_code,proc1\_desc,...,proc4\_code,proc4\_desc**

**Output Format A JSON file containing FHIR-compliant resources: • Patient • Encounter • Procedure • Claim Each resource uses appropriate FHIR data types, such as: • Reference • CodeableConcept • Period • Money Output is saved to: /opt/airflow/data/output/fhir\_output.json**

**Key Components**

1. **transform\_csv\_to\_fhir.py This is the transformation logic executed by the DAG: • Reads the input CSV file using Pandas. • Loops through each row to generate FHIR resources. • Uses fhir.resources library to create resource models. • Converts models to dicts and saves as JSON.**
2. **post\_to\_hapi.py This script posts the generated FHIR resources to a running HAPI FHIR server endpoint: • Reads the output JSON file. • Sends POST/PUT requests to the configured FHIR endpoint. • Displays HTTP response code and success/failure status.**
3. **Airflow DAG (csv\_to\_fhir\_dag.py) Defines the pipeline: • PythonOperator task for transform\_csv\_to\_fhir • PythonOperator task for post\_to\_hapi • Can be scheduled or manually triggered from Airflow UI**
4. **Docker Setup Docker Compose sets up: • Airflow Webserver • Scheduler • Worker • Postgres (metadata DB) • Redis (Celery broker) • HAPI FHIR Server (optional - can be included in docker-compose.yml)**

**How to Run**

**Step 1: Set Up Docker and Airflow**

**docker-compose up -d**

**Step 2: Place CSV file**

**./data/input/healthcare\_data.csv**

**Step 3: Trigger DAG • Open Airflow UI at http://localhost:8080 • Enable and trigger the DAG: csv\_to\_fhir\_pipeline**

**Step 4: Output Check for the generated file:**

**./data/output/fhir\_output.json**

**Step 5 (Optional): View on HAPI Server • Ensure HAPI FHIR server is running at http://localhost:3000 • Use Swagger or FHIR client to view uploaded data**

**Troubleshooting • Make sure input dates are in MM/DD/YYYY format. • Airflow logs are available in /opt/airflow/logs/ for debugging. • If TypeError: Object of type date is not JSON serializable, ensure date.isoformat() is used.**

**License MIT License**

**Author Nagapraneeth For questions, raise an issue on the GitHub repo or email the author.**