Technical Documentation: FHIR Data Transformation Pipeline

1. Introduction

The FHIR Data Transformation Pipeline prototype is a solution designed to ingest FHIR (Fast Healthcare Interoperability Resources) data from an external system, transform it into a more workable format, deidentifying and store it in a Elasticsearch for data analysis and for visualization and dashboard creation using Kibana. AWS cloud services like S3 Bucket, Amazon Simple Queue Service (SQS), Amazon Elastic Kubernetes Service (EKS), Elastic Container Registry which is used for scalability. For Continuous Integration / Continuous Deployment (CI/CD) Pipeline, GitHub Action is used. This documentation provides an overview of the solution architecture, installation instructions, usage guidelines, and next steps for further enhancements.

2. Architecture Overview

The solution architecture consists of the following components:

- **External System / Supplier:** Sends FHIR data in JSON format to an Amazon S3 bucket.
- **Amazon S3 Bucket:** It is an object storage service. Receives FHIR data files from the external system, whenever it receives the new fie, sends the notification to Amazon SQS queue.
- **Simple Queue Services:** It makes sure the one-time delivery of the message.
- **Apache Airflow:** It’s used to scheduling the jobs and running in container to gets the works done in a distributed way.
- **Amazon Elastic Kubernetes Service (EKS):** It’s managed Kubernetes service to run Kubernetes in the AWS cloud and responsible for scheduling containers, managing application availability. Orchestrates Docker containers for Airflow, Python jobs, and other services.
- **Python:** Python libraries are used to access the data from S3 bucket and normalized, de-identified the PII information before dropping into elastic (container)
- **ElasticSearch:** Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Stores the transformed FHIR data for further analysis and it can be queried faster, and it can be used in real time analytics and extended to gives the capabilities of Machine learning, Artificial Intelligence for Enterprise plan. Nodes are runs in pod and it’s managed by EKS to provide the availability. (Container)
- **Kibana:** Provides visualization and analysis capabilities for the data stored in ElasticSearch.(Container)

- **Continuous Integration / Continuous Deployment (CI/CD) Pipeline:** Utilizes GitHub Actions for automated building and deployment of code changes to the data pipeline.

3. Installation and Setup

Prerequisites:

AWS account with appropriate plan and access permissions for the services mentioned in the architecture section.

Access to the GitHub repository containing the solution code, template for build and deploy the images.

Installation Steps:

1. Configure AWS credentials and permissions for accessing S3, SQS, EKS.
2. Using AWS console/CLI, Create the S3 buckets with Standard class and policy can be configured in way to access by the external supplies with keys and use AWS IAM for the restricted access.
   S3 Bucket Configuration:
   - **Bucket Class**: Since the data needs to be accessed frequently by various services, consider using the **Standard storage class** for the S3 bucket. This class provides high durability and availability for frequently accessed data.
   - **IAM:** Create a role Permissions **s3:GetObject and s3:ListBucket** on the specific S3 bucket where the FHIR data is stored. This allows the Airflow, Python jobs to read data from the bucket.
3. Create a private repository in Elastic Container Registry to store the codes, packages, images and control the access to the images.
   - Authenticate Docker to the ECR Registry:
     - Install and configure the AWS Command Line Interface (CLI)
     - Run the command to authenticate Docker to your ECR registry.
   - Build and tag your Docker Image
     - Navigate to the directory containing your Python job, elastic, kibana,
   - Push the Docker Image to ECR
   - Configure Kubernetes to Pull Images from ECR

   Note: Above steps can be done for testing, this step can be put it in the git action template and with necessary permission to create, build, deploy the images.

4. Deploy the Elastic DB, Kibana Docker container to the EKS cluster.
   Ideally – Go with configure 3 node cluster in elastic and one node in Kibana and it can be scaled depends on the requirement.

Running the Solution:

- After deployment, the Airflow job starts monitoring the SQS queue for incoming messages every 5 minutes.
- When a new FHIR data file is received in the S3 bucket and notification can send to the SQS message and it monitored by triggers the Airflow job.
- The Airflow job executes the dependency Python job, which reads, normalizes, and de-identifies the recent data from S3.
- The transformed data is then stored in ElasticSearch for visualization and analysis in Kibana.

Viewing and Using the Results:

- Access Kibana using the provided URL.
- Navigate to the appropriate index containing the transformed FHIR data.
- Utilize Kibana's visualization and querying capabilities to analyze the data, create dashboards, and generate reports.

5. Next Steps

Potential enhancements and future considerations for the solution include:

General Steps:

- Adding monitoring and alerting capabilities for solution components.
- Enhancing security measures such as data encryption and access control.
- Optimizing performance and scalability of the ElasticSearch cluster.
- Extending support for additional data formats or sources beyond FHIR.
- Implementing data validation and quality checks to ensure data integrity.
- Exploring cost optimization strategies such as utilizing reserved instances or spot instances for EC2 nodes in the EKS cluster.

Analytics:

- **NLP Algorithm** can used to analyse the EMR unstructured data.
- Use **Semantic search** is a search engine technology that interprets the meaning of words and phrases. The results of a semantic search will return content matching the meaning of a query, as opposed to content that literally matches words in the query.
- **Gen AI Assistant** can configured and it have the ability answer the user questions with trained EMR data.

6. Conclusion

The FHIR Data Transformation Pipeline provides a scalable and efficient solution for ingesting, transforming, and visualizing FHIR data for analytics and insights. By following the installation instructions and usage guidelines outlined in this documentation, users can effectively deploy and utilize the solution to meet their data transformation and analysis require.