**GAMMIS PROJECT DOCUMENTATION**

**Digital HIE Inc**

**270**: The Edi X12 270 transaction set is part of the X12 series of EDI standards, which cover various types of business transactions across different industries. The 270 transactions specifically allow providers to check a patient’s insurance coverage rapidly and efficiently, including details like eligibility and benefits, before or at the time of service.

**271** **(Health Care Eligibility/Benefit Inquiry and Response):** This standard allows healthcare providers to inquire about a patient’s health insurance benefits and eligibility electronically. In response, the insurance company sends back a X12 271 message, detailing the patient's coverage information, including benefits, co-pays, deductibles, and other relevant insurance policy details. This process streamlines the exchange of insurance information, making it faster and more efficient compared to traditional methods.

**837I (Institutional Claim)**: The 837I is used for transmitting institutional healthcare claims, such as those from hospitals, inpatient facilities, and other institutional providers. It includes information about the patient, healthcare provider, diagnosis, procedures performed, and other details relevant to the institutional healthcare claim. The 837I is utilized for electronic submission of healthcare claims for inpatient and outpatient services provided by institutional healthcare providers.

**837P (Professional Claim)**: The 837P is used for transmitting professional healthcare claims, typically from individual healthcare practitioners such as physicians, therapists, and other professional service providers. It includes information like the 837I but tailored to professional services, including details about the patient, healthcare provider, diagnosis, procedures performed, and other relevant billing information. The 837P is employed for electronic submission of healthcare claims for professional services rendered by individual healthcare practitioners.

**TECH STACK:**

* Java
* SpringBoot (STS)
* JSON
* Excel (Crosswalk/ Data Mapping)
* Jackson library
* AWS Lambda function
* Postman

**DOCUMENTS:**

* GAMMIS Companion Guides (<https://www.mmis.georgia.gov/portal/PubAccess.EDI/Companion%20Guides/tabId/47/Default.aspx> )
* Availity (<https://apps.availity.com/availity/documents/EDI_ConnectionServices_QuickStartGuide_332.pdf>)
* Palmetto (<https://palmettogba.com/Palmetto/Providers.Nsf/files/EDI_GPNet_Comm.pdf/$File/EDI_GPNet_Comm.pdf>)

**PROJECT IMPLEMENTATION FLOW:**

A screenshot of a computer

Description automatically generated

1. Go Over the Documentation of GAMMIS
2. Created crosswalks of the data mapping between the Claim Form and the GAMMIS documents.

* Understand Structure
* Identify Key Data Elements
* Create Crosswalk Spreadsheet

1. Create JSON files using the created crosswalk and GAMMIS companion Guides.

* The structure needs to be an envelope (ISA-IEA) segments, because when converted it needs to match and transform to a valid EDI x12 format with delimiters (~ \*)
* 3 JSON files (837P/ 837I/ 270)

1. Create a Spring boot application with base code.

* Use Spring Initializer to generate basic project structure.
* Maven Project
* Language: Java
* Spring Boot: 3.2.1
* Packaging: Jar
* Java: 17
* Dependencies: Spring Web, Spring Boot Dev Tools (pom.xml)

1. Divide into modules based on structure, updates, and regular fixes.
2. Deploy on AWS Lambda function.

* Developed Lambda Function
* Deployed it using API Gateway URL

**Steps to test in Postman:**

1. Open Postman

2. Create a Request

* HTTP request

3. Select the Request Type

* Request Type: “POST”

1. Enter Request URL: Enter the URL of your API endpoint. (AWS Lambda Function)

* <https://svad5abgii.execute-api.us-east-2.amazonaws.com/dev>

5. Select the Body Tab:

* Go to the "Body" tab in your Postman request.
* Select the "raw" option to input raw data.
* Choose the content type as "JSON (application/Json)" from the dropdown next to the raw data input.

6. Input Encoded JSON Payload

7. Send Request

* Once you have entered the JSON payload, click the "Send" button to make the request.

8. Check Response:

* Check the response in the lower part of the Postman window. If there are any errors or issues, they will be displayed there.
* Make sure that the JSON payload you enter in Postman matches the structure expected by your server.
* To validate (decode) the encoded Input/ Output, use [Base64 Decode and Encode - Online](https://www.base64decode.org/)