



**Universitat Politècnica de Catalunya (UPC) - BarcelonaTech**

**Facultad de Informática de Barcelona (FIB)**

***Implementación de Seguridad y Privacidad de datos  
clínicos con el estándar HL7 FHIR***

***Trabajo de Final de Máster***

**Máster en Ingeniería Informática - MEI**

Autor: Benny Hammer Pérez Vásquez

Director: Jaime Delgado Mercé

Departamento de Arquitectura de Computadoras

28 de Junio, 2022

# Índice

<b>Índice de Figuras</b>	<b>4</b>
<b>Índice de Tablas</b>	<b>6</b>
<b>1 Introducción</b>	<b>8</b>
1.1 Motivación	9
1.2 Alcance y Objetivos	10
1.3 Estructura de la Tesis	11
<b>2 Estado del Arte</b>	<b>13</b>
2.1 Estándares Clínicos	13
2.1.1 Health Level Seven International (HL7)	13
2.1.2 Fast Healthcare Interoperability Resources (FHIR)	14
2.1.3 OpenEHR	15
2.1.4 Leyes y Regulaciones	17
2.2 Aplicativos actuales	19
2.2.1 Synthea Patient Population Simulator	18
2.2.2 Microsoft AZURE FHIR server	19
2.2.3 Firely	20
2.2.4 Google FHIR (FhirProto)	21
2.3 Implementaciones de Seguridad en eHealth	21
2.3.1 SMART on FHIR	22
2.3.2 FHIRCast (HTTP Web Hooks using WebSub)	23
2.3.3 Experimental Websockets support	24
<b>3 Estándares y Tecnología utilizados</b>	<b>25</b>
3.1 Análisis realizado	25
3.2 Fast Healthcare Interoperability Resources(FHIR)	26
3.1.1 Seguridad	28
3.1.2 Privacidad	28
3.1.3 Módulo de Seguridad y Privacidad en FHIR	28
3.1.3.1 Recursos	29
3.1.3.2 Tipos de Datos	30
3.1.3.3 Guías y Principios de Implementación	30
3.1.4 Características de FHIR a implementar	33
3.2 HAPI FHIR	33
3.2.1 Arquitectura HAPI FHIR JPA Server	35
3.2.2 Seguridad	36
3.2.3 Interceptor	37

3.2.4 Conclusiones características HAPI a utilizar	38
3.3 Herramientas de software	39
<b>4 Diseño de Casos de Uso</b>	<b>45</b>
<b>5 Limitaciones del proyecto</b>	<b>49</b>
5.1 Seguridad y privacidad	49
5.2 Soporte Multi Lenguaje	50
5.3 Conformidad FHIR	50
5.4 Performance	50
5.5 Manejo de errores	51
5.6 Independencia de plataforma	51
<b>6 Arquitectura e Implementación</b>	<b>52</b>
6.1 Diagrama de Arquitectura HAPI	52
6.1.1 Authorization Interceptor	55
6.2 Arquitectura del Aplicativo demostrador	56
6.3 Implementación y Desarrollo	58
6.3.1 Intentos de Implementación HAPI	59
6.3.2 Implementación estable	60
6.3.3 Lógica control de acceso HAPI FHIR	61
6.3.4 Desarrollo del aplicativo Demostrador	64
6.3.5 Generación de script de datos de prueba FHIR	71
<b>7 Experimentos y resultados</b>	<b>72</b>
7.1 Resultados por Casos de Uso	72
7.1.1 Casos de uso búsqueda de pacientes por ID	72
7.1.2 Casos de uso conteo de pacientes por criterio de búsqueda	74
<b>8 Costos y beneficios de la propuesta</b>	<b>79</b>
8.1 Costos de la implementación	79
8.2 Beneficios de la propuesta	80
<b>9 Trabajo futuro</b>	<b>82</b>
<b>10 Conclusiones</b>	<b>83</b>
<b>11 Referencias bibliográficas</b>	<b>84</b>

# Índice de Figuras

Figura 1: Modelo del ecosistema openEHR	16
Figura 2: PADARSER como modelo del marco de referencia para Synthea	18
Figura 3: Modelo de interacción de flujos FHIRCast	24
Figura 4: Niveles definidos en el estándar FHIR	27
Figura 5: Diagrama arquitecturas de seguridad	29
Figura 6: Diferentes configuraciones que pueden darse a HAPI FHIR	34
Figura 7: Arquitectura del aplicativo HAPI FHIR JPA a alto nivel	36
Figura 8: Diagrama de canalización HAPI JPA Server	38
Figura 9: Formato JSON FHIR	41
Figura 10: Formato XML FHIR	41
Figura 11: Modelo UML de FHIR RESTful	42
Figura 12: Características del servidor a utilizar	44
Figura 13: Arquitectura simplificada HAPI FHIR utilizada en el proyecto	53
Figura 14: Actuación del Authorization Service	54
Figura 15: Authorization en operaciones de Lectura	55
Figura 16: Authorization en operaciones de escritura	56
Figura 17: Arquitectura simplificada aplicativo demostrador	57
Figura 18: Resultado de la ejecución de HAPI FHIR 5.6.0	59
Figura 19: Resultado de la ejecución de hapi-fhir-jpaserver-starter 5.7.0	60
Figura 20: Resultado de la prueba del aplicativo a través de un navegador web	60
Figura 21: Inicio servidor estable HAPI FHIR 5.2	61
Figura 22: Interfaz HAPI FHIR estable HAPI FHIR 5.2	61
Figura 23: Diagrama de Flujo de la lógica implementada en HAPI FHIR	63
Figura 24: Listado de componentes modificado en HAPI	64
Figura 25: Diagrama simplificado del Front End y Back End	65
Figura 26: Modelo MVC del aplicativo demostrador	65
Figura 27: Estructura Back End	66
Figura 28: Estructura de páginas Front End	68
Figura 29: Interfaz de acceso de usuario	69
Figura 30: Interfaz de usuario Paciente	69
Figura 31: Interfaz de usuario Médico	70
Figura 32: Interfaz de usuario Organización	70
Figura 33: Interfaz de usuario Administrador	70
Figura 34: Arquitectura simplificada de script HAPI FHIR	71
Figura 36: Búsqueda de información por el mismo paciente	72
Figura 37: Búsqueda de información por parte de otro paciente	73
Figura 38: Búsqueda de información por el Médico del paciente	73
Figura 39: Búsqueda de información del paciente por parte de otro médico	73

Figura 40: Búsqueda de información por parte de la organización del paciente	74
Figura 41: Búsqueda de información por parte de otra organización	74
Figura 42: Opción de búsqueda como administrador	75
Figura 43: Cantidad de pacientes por edad	75
Figura 44: Cantidad de pacientes por presión sanguínea	76
Figura 45: Cantidad de pacientes por condición médica	76
Figura 46: Cantidad de pacientes por todos los criterios de búsqueda	77
Figura 47: Búsqueda de pacientes por rol	77
Figura 48: Búsqueda de pacientes en modo Administrador	78



# Índice de Tablas

Tabla 1: Versiones del estándar FHIR	15
Tabla 2: Descripción de las categorías de las etiquetas de seguridad FHIR	33
Tabla 3: Interacciones RESTful HAPI FHIR	43
Tabla 4: Códigos de estado HTTP en FHIR	44
Tabla 5: Reglas establecidas para la visualización de información de pacientes	46
Tabla 6: Estimación de esfuerzo del estudiante	79
Tabla 7: Costo total del proyecto en Euros	80

## Descripción

Con el avance de la tecnología muchas instituciones de salud están actualizando su infraestructura de información para lo cual están adoptando tecnologías para la gestión de datos clínicos, estas tecnologías están basadas en estándares proporcionados por Health Level Seven (HL7) u OpenEHR, las cuales son dos organizaciones que proveen de los estándares más actualizados que se están utilizando hoy en día; pero poco es lo que estas tecnologías tienen implementado en términos de seguridad y privacidad de la información dado que los estándares proporcionados están enfocados en la gestión de los datos y en interoperabilidad entre los sistemas.

De la necesidad de seguridad y privacidad de los datos clínicos de los pacientes nace la idea de este proyecto, en el cual se analizará el estándar Fast Interoperability Healthcare Resources (FHIR) de HL7 con la finalidad de extraer los métodos y características de privacidad, seguridad y control de acceso que protegen la información de los usuarios, y su aplicación en los sistemas de gestión de la información que usan las instituciones de la salud, lo que permitirá garantizar el resguardo de la información de los pacientes.

Para la validación del funcionamiento de las características de seguridad embebida en el estándar FHIR se utilizará el aplicativo HAPI FHIR de Smile CDR. Este aplicativo es un software de código abierto el cual tiene como característica principal que es el más utilizado a nivel mundial, además de que es el software más completo que se ejecuta bajo FHIR hoy en día. En términos de seguridad HAPI FHIR presenta métodos y procedimientos que nos permitirán reproducir las características de seguridad descritas en el estándar.

En resumen, en este proyecto se implementará reglas de seguridad genéricas en el aplicativo HAPI FHIR que permitirán restringir el acceso a la información de los pacientes; adicionalmente se diseñará y desarrollará un aplicativo demostrador que permitirá realizar la búsqueda de información de cada paciente a través de una interfaz de usuario, así como la de extraer cantidad de pacientes de HAPI tomando en cuenta el rol y parámetros de búsqueda que definiremos más adelante en el documento; este aplicativo también sirve como demostrador el cual nos permite validar la ejecución de los casos de uso que se diseñaron para garantizar los objetivos definidos de este trabajo.

# 1 Introducción

Con el fin de lograr los objetivos del proyecto se analizaron aplicaciones para la gestión de información médica de los cuales se eligió el aplicativo HAPI FHIR que opera bajo el estándar FHIR (Fast Healthcare Interoperability Resources) del conjunto de estándares HL7, que facilita el intercambio de información clínica entre sistemas; este aplicativo específicamente diseñado para operar en base a conexiones Restful APIs que usan el protocolo HTTP, combina las mejores características de los estándares HL7 V2, HL7 V3 y CDA, al tiempo que aprovecha las últimas tecnologías de web services.

Tomando como fuente el aplicativo HAPI FHIR se implementaron las reglas de seguridad necesarias para gestionar la información clínica de los pacientes, estas reglas fueron asignados a roles para el acceso a la información; para validar el correcto funcionamiento de la configuración/implementación se diseñaron casos de uso que permitieron modelar el funcionamiento de los servicios del aplicativo y específicamente validar el apartado de seguridad, privacidad y acceso a los datos de los pacientes.

Adicionalmente, se generó un aplicativo demostrador que interactúa con el servidor del aplicativo FHIR donde se encuentra alojada la información de los pacientes, este aplicativo cuenta con una base de datos que almacena las credenciales de acceso de cada uno de los usuarios que requieren acceso a la información de los pacientes en HAPI; este aplicativo también contiene un módulo que permite cuantificar la cantidad de pacientes de acuerdo al tipo de observación médica que posean, para eso se implementó ciertos criterios de búsqueda que se definieron en la sección de diseño de los casos de uso; este aplicativo servirá de demostrador del cumplimiento de objetivos de este trabajo.

Finalmente, se implementó un generador de datos en HAPI de los tipo de recurso “patient” (Paciente), “practitioner” (Médico), “organization” (Clínica u hospital) y “observation” (Observación clínica), con el fin de contar con información que nos permita realizar pruebas de los casos de uso.



## 1.1 Motivación

La principal motivación de este proyecto se debe a que actualmente muchas aplicaciones se centran en el almacenamiento de datos, en la funcionalidad de la herramienta de gestión de la información y en la interoperabilidad entre los sistemas; pero no se centran en la protección del acceso, seguridad y confiabilidad de la información que gestionan.

Adicionalmente, cada vez más instituciones de salud están implementando sistemas de información clínica interoperables con otros sistemas, lo que propicia que los datos estén cada vez más expuestos; para ello es necesario contar con un sistema de seguridad y privacidad de la información que proteja los datos de los pacientes a la vez que permita el flujo de la información adecuado entre los sistemas de información.

Otro de los puntos importantes es que los métodos de seguridad propuestos en otros estándares como HL7 V3, V2, CDA son muy complejos, lo cual impide que la mayoría de las instituciones de salud con bajo presupuesto las puedan implementar y con esto se deja desprotegida la información de los usuarios.

Por esta razón, surge la propuesta del uso de estándares como HL7 FHIR que se adapta a las tecnologías de la información actuales a través de un conector Restful API, que proporcionan niveles de seguridad adecuados para la información de los pacientes y que además no requiere un presupuesto elevado para su implementación.

## 1.2 Alcance y Objetivos

Este proyecto cuenta con dos objetivos específicos. El primer objetivo es analizar el estándar FHIR y el aplicativo HAPI para definir características de seguridad que se implementarán en el aplicativo demostrador como parte del segundo objetivo. El segundo objetivo es la generación de casos de uso que nos permitirán validar las características de seguridad definidos como parte del primer objetivo; este objetivo incluye la implementación de las características de seguridad en el aplicativo HAPI, además de la implementación de un aplicativo demostrador que nos permitirá realizar la validación de los casos de uso definidos.

Del análisis realizado en el primer objetivo se encontró que la característica de seguridad FHIR más viable de implementar es “Authorization/Access Control”, a partir de ello se ha elegido el modelo de control de acceso “Role-Based Access Control” (RBAC). Por otro lado, analizando las características de HAPI se encontró el módulo “Authorization Interceptor” que permite emular el modelo de control de acceso RBAC descrito en el estándar.

El segundo objetivo involucra la implementación en el aplicativo HAPI de reglas de seguridad definidas en el primer objetivo y la implementación de un aplicativo demostrador que se interconectará con HAPI para extraer la información de los pacientes y que permitirá la ejecución de los casos de uso definidos en el apartado Diseño de Casos de Uso.

El demostrador constará de dos partes importantes, la primera es implementar un buscador de pacientes a través del id del paciente; el demostrador enviará una solicitud de pacientes a HAPI para extraer la información requerida; HAPI contendrá las reglas de seguridad definidas como parte del primer objetivo. La segunda parte del aplicativo corresponde a la generación de un módulo que permitirá el conteo de pacientes a partir de criterios de búsqueda; para esta funcionalidad también se utilizarán las reglas definidas en HAPI.

Adicionalmente, el aplicativo demostrador contará con un módulo de seguridad de acceso a través de usuario y contraseña que permitirá restringir el acceso a la funcionalidad de la herramienta de ingresos no deseados.

Para habilitar la ejecución de los casos de uso de del objetivo 2, se diseñará y desarrollará un aplicativo que genere información ficticia de pacientes y que permita almacenar la información generada en el aplicativo HAPI; la importancia de la generación de estos datos

de prueba radica en poder utilizar la data generada para simular los casos de uso que se definieron como parte del objetivo 2.

## 1.3 Estructura de la Tesis

Debido a lo extenso del documento de proyecto, se procedió a dividirlo en partes que detallen las características esenciales de la investigación e implementación, que principalmente abarca la búsqueda de las características del estándar en términos de seguridad, el aplicativo a utilizar como fuente, las herramientas disponibles para la implementación, el diseño de los casos de uso que servirá para la implementación de las reglas de seguridad en HAPI y el desarrollo del aplicativo demostrador, la arquitectura del aplicativo a desarrollar, los resultados de la implementación, los costos embebidos, el análisis de la contribución del proyecto, el trabajo futuro y las conclusiones. En las siguientes líneas se brindará una breve información de cada sección del documento.

- **Introducción:** Es el apartado actual, en dónde se describen las características del trabajo que se propone realizar, en este se detallan temas importantes como la motivación, el alcance y los objetivos de este trabajo, además se brinda el detalle de cómo está estructurado el documento con una pequeña descripción del contenido de cada sección.
- **Estado del Arte:** Es la sección dónde se describe la investigación sobre lo ya existente en relación a seguridad de datos clínicos en FHIR y otros estándares. En este apartado se mencionan los estándares clínicos más usados para la gestión de información clínica; adicionalmente se describirán los productos más importantes implementados y sus principales características, entre ellos el aplicativo que se va a tomar como referencia para la implementación de este proyecto HAPI FHIR.
- **Estándares y Tecnología utilizados:** En esta parte del documento se visualizará más a fondo el estándar seleccionado para la implementación de este proyecto, sus principales características en términos de seguridad y privacidad de la información; así como los recursos que ofrece, tipo de datos, guías y principios de implementación.

Adicionalmente, se profundiza el análisis del aplicativo HAPI FHIR(JPA Server) que se estará utilizando como base de este proyecto, se analizará sus principales



características de las cuales nos centraremos en el método de cómo ofrece seguridad y privacidad de la información.

- **Diseño de los Casos de Uso:** Es una parte muy importante del documento, en dónde se diseñarán los casos de uso que han de utilizarse para realizar la validación de las características de seguridad en la implementación de este proyecto.
- **Limitaciones del proyecto:** Para delimitar el alcance del proyecto se genera este apartado, en el que se mencionan los conceptos y elementos que estarán fuera del alcance de este trabajo.
- **Arquitectura e Implementación de la propuesta:** Una parte muy importante de este proyecto es el diseño de la arquitectura que se va a utilizar para la implementación del proyecto, lo cual permitirá el cumplimiento de los casos de uso ya definidos, además se mostrará las complicaciones iniciales en la implementación del proyecto.
- **Experimentos y resultados:** En este apartado se pondrán a prueba los casos de uso, se mostrarán los resultados de la ejecución de cada uno de los casos de uso diseñado como parte del objetivo 2.
- **Costos y beneficios de la propuesta:** Una parte muy importante de este tipo de proyectos es la estimación del esfuerzo realizado y de los recursos utilizados para cuantificarlos en moneda local, esto con el fin de poder obtener el costo de ejecución del proyecto.
- **Trabajo futuro, Conclusiones y Referencias bibliográficas:** Luego de obtener los resultados se procederá al análisis del desarrollo del proyecto y del producto obtenido para plasmar ideas de lo que se podría hacer en un trabajo futuro como recomendación ante futuros proyectos; además se realizarán las conclusiones de todo el trabajo realizado; finalmente se plasmarán las referencias bibliográficas que sustentan el desarrollo e implementación del proyecto.

## 2 Estado del Arte

El enfoque propuesto en este trabajo se basa en analizar e implementar las características del aplicativo HAPI FHIR en términos de seguridad de la información de los servicios que ofrece. Además, la simulación de todo el entorno es fundamental para las pruebas y la evaluación.

El estándar FHIR dentro de los estándares de HL7 es el que ofrece muchas mejoras en comparación con otros estándares, mejoras como por ejemplo: Rápido y fácil de implementar, Múltiples librerías para la implementación, La especificación es libre para su uso sin restricciones, Facilidad de interoperabilidad (puede ser adaptado de acuerdo a las necesidades de las instituciones), entre otros[2]; lo que lo convierte en el estándar a seguir para este trabajo.

En este apartado se presenta un resumen de los trabajos existentes en las características tecnológicas de seguridad de la información antes mencionadas y relacionado con el alcance de este proyecto.

### 2.1 Estándares Clínicos

En la actualidad existen una gran cantidad de estándares que permiten la gestión de información clínica, entre ellos tenemos los proporcionados por la organización HL7 del cual mencionaremos el estándar FHIR; además también se mencionará el estándar OpenEHR; y finalmente se presentarán las principales leyes y regulaciones que están relacionadas con el desarrollo de este proyecto.

#### 2.1.1 Health Level Seven International (HL7)

Fundado en 1987, HL7 es una organización sin fines de lucro cuya finalidad es el desarrollo de estándares, acreditada por ANSI, se dedica a proporcionar un marco integral y estándares relacionados para el intercambio, integración, compartimiento y recuperación de información electrónica de salud.

Los estándares proporcionados por HL7 definen cómo se empaqueta y comunica la información de una parte a otra, estableciendo el idioma, la estructura y los tipos de datos necesarios para la integración entre sistemas.

De entre los estándares elaborados por HL7, de los más importantes tenemos los siguientes[10]:

- C-CDA (HL7 CDA® R2 Implementation Guide: Consolidated CDA Templates for Clinical Notes - US Realm).
- CDA® Release 2.
- FHIR® R4 (HL7 Fast Healthcare Interoperability Resources, Release 4).
- HL7 Context Management Specification (CCOW), Version 1.6.
- HL7 Version 2 Product Suite.
- HL7 Version 3 Product Suite.

### **2.1.2 Fast Healthcare Interoperability Resources (FHIR)**

FHIR es un estándar de Interoperabilidad destinado a facilitar el intercambio de información de atención médica entre proveedores de atención médica, pacientes, cuidadores, pagadores, investigadores y cualquier otra persona en el ecosistema de atención médica.

Consta de dos partes principales: Un modelo de contenido en forma de “recursos” y una especificación para el intercambio de estos recursos en forma de interfaces RESTful en tiempo real, así como mensajería y documentación. Entre los principales beneficios de usar este estándar, tenemos los siguientes[11]:

- Crea una especificación común mediante la cual los participantes de la atención médica pueden compartir información.
- Permite el desarrollo de aplicaciones que se benefician del acceso a información de alta calidad de una manera que los desarrolladores consideran fácil.
- Soporta mejoras en la prestación de atención médica, incluidas nuevas iniciativas como la “Atención Basada en Valor”.

En la tabla 1 se muestran las versiones de FHIR que han venido apareciendo a lo largo del tiempo.

Fecha	Versión	Descripción
<b>Current Versions</b>		
2019-10-30 (Actual)	4.0.1	FHIR Release #4: Primer contenido Normativo
<b>R5 Sequence (En curso)</b>		
2021-12-19	5.0.0-snapshot1	FHIR Release #5: Snapshot #1 (Connectathon Enero 2022)
<b>R4B Sequence (Work in Progress)</b>		
2021-12-20	4.3.0-snapshot1	FHIR Release #4B: Snapshot #1 (Connectathon Enero 2022)
<b>R4 Sequence (Current)</b>		
2019-10-30	4.0.1	FHIR Release #4 Primer Contenido Normativo con 1 errata técnica
<b>STU 3 Sequence (Historical)</b>		
2019-10-24	3.0.2	FHIR Release 3 (STU) con 2 erratas técnicas
<b>DSTU 2 Sequence (Historical)</b>		
2015-10-24	1.0.2	DSTU 2 (Versión Oficial) con 1 errata técnica
<b>DSTU 1 Sequence (Historical)</b>		
2014-09-30	0.0.82	DSTU 1 (Versión Oficial) con 2 erratas técnicas
<b>Historical Archive Sequence (Historical)</b>		
2012-05-14	0.01	Primera versión etiquetada como FHIR

Tabla 1: Versiones del estándar FHIR [12]

### 2.1.3 OpenEHR

OpenEHR es un estándar abierto para la administración electrónica de información clínica, consta de especificaciones abiertas, modelos clínicos y software que puede ser usado para crear estándares y construir información y soluciones de interoperabilidad entre sistemas de salud. La mayoría de los artefactos de openEHR son producidos por la comunidad openEHR y administrada por openEHR internacional, la cual es una institución sin fines de lucro desde el año 2003.

OpenEHR está diseñado para satisfacer esta necesidad, proporcionando una arquitectura integral para especificar, diseñar y construir soluciones de salud electrónica, el cual tiene las siguientes características:

- Un marco de modelado multinivel que separa la representación de datos del contenido del dominio.
- Una arquitectura de plataforma abierta que se puede utilizar para representar datos de salud centrados en el paciente, a los que acceden instituciones y productos, pero que no la controlan.
- Un formalismo de modelado de dominio que admite composición, especialización, localización y enlace flexible a la terminología.
- Un entorno de fábrica de modelado que crea una biblioteca de puntos de datos, conocidos como arquetipos, desarrollados por expertos en el dominio, en cualquier idioma.
- La capacidad de recombinar puntos de datos en conjuntos de datos, conocidos como plantillas, para casos de uso locales.
- Herramientas que convierten modelos de dominio en formas técnicas que se pueden usar para construir:
  - Aplicaciones (por ejemplo, definiciones de pantalla).
  - Componentes de interoperabilidad (definiciones de mensajes) y
  - Ser consumido por las implementaciones de la plataforma en tiempo de ejecución (definiciones de conjuntos de datos).

Podemos visualizar el ecosistema de tecnología openEHR en la figura 1.

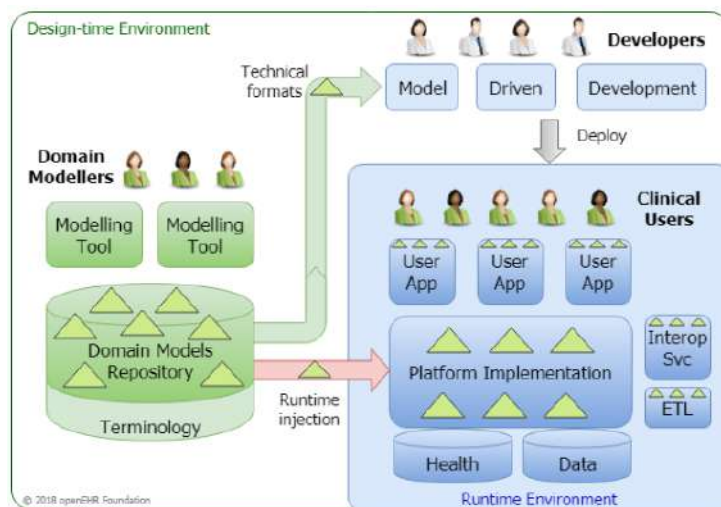


Figura 1: Modelo del ecosistema openEHR[31]



Con respecto a la seguridad y privacidad en openEHR, estos se encuentran en los sistemas más que en el modelo openEHR, particularmente la implementación de autenticación, control de acceso y encriptación de la información. Las especificaciones de openEHR y la implementación de los componentes principales no definen explícitamente muchos mecanismos concretos ya que existe una gran variedad en los requisitos de diferentes sitios. Lo que se hace en openEHR es admitir algunos de los requisitos clave de una manera lo suficientemente flexible como para que las implementaciones con requisitos y configuraciones sustancialmente diferentes puedan implementar los principios básicos de manera estándar [32].

### 2.1.4 Leyes y Regulaciones

Debido a que en este proyecto se realiza el tratamiento de datos clínicos de usuarios, en España entran a tallar dos instituciones que regulan el tratamiento de datos médicos, estas son la Ley Orgánica de Protección de Datos(LOPD) y el Reglamento General de Protección de Datos(RGPD o GPDR en inglés).

- **Ley Orgánica de Protección de Datos(LOPD):** Es una ley Española establecida en el artículo 18 de la constitución en la que se menciona el derecho a la protección de datos de los ciudadanos como un derecho fundamental que tiene cada individuo y que se traduce en la potestad de control de sus datos personales[33].
- **Reglamento General de Protección de Datos(RGPD):** Establece las normas relativas a la protección de datos de las personas naturales en lo que respecta al tratamiento de sus datos personales y normas relativas a la libre circulación de datos personales. Este Reglamento protege los derechos y libertades fundamentales de las personas naturales y, en particular, su derecho a la protección de datos personales.[34].

## 2.2 Aplicativos actuales

En el mercado existe una variedad muy grande de aplicativos basados en FHIR, por lo que en este apartado se mostrará los principales aplicativos desarrollados en base a este estándar. Algunos de estos aplicativos cuentan con características para la seguridad y privacidad de la información clínica que gestionan. Entre estas implementaciones, tenemos las siguientes:

## 2.2.1 Synthea Patient Population Simulator

Synthea™ es un generador de pacientes sintéticos que modela el historial médico de pacientes sintéticos. Su misión es producir datos de pacientes sintéticos, realistas pero no reales, de alta calidad y registros de salud asociados que cubren todos los aspectos de la atención médica. Los datos resultantes están libres de restricciones de costos, privacidad y seguridad. Se puede usar sin restricciones para una variedad de usos secundarios en la academia, la investigación, la industria y el gobierno.

Este aplicativo usa PADARSER como modelo de referencia el cual cuenta con un millón de pacientes sintéticos que son libremente accedidos a través de un buscador web, esta información está codificada en formatos HL7 Fast Healthcare Interoperability Resources (FHIR) que pueden ser accedidos a través de una aplicación HL7 FHIR [3], en la figura 2 se muestra la estructura de PADARSER.

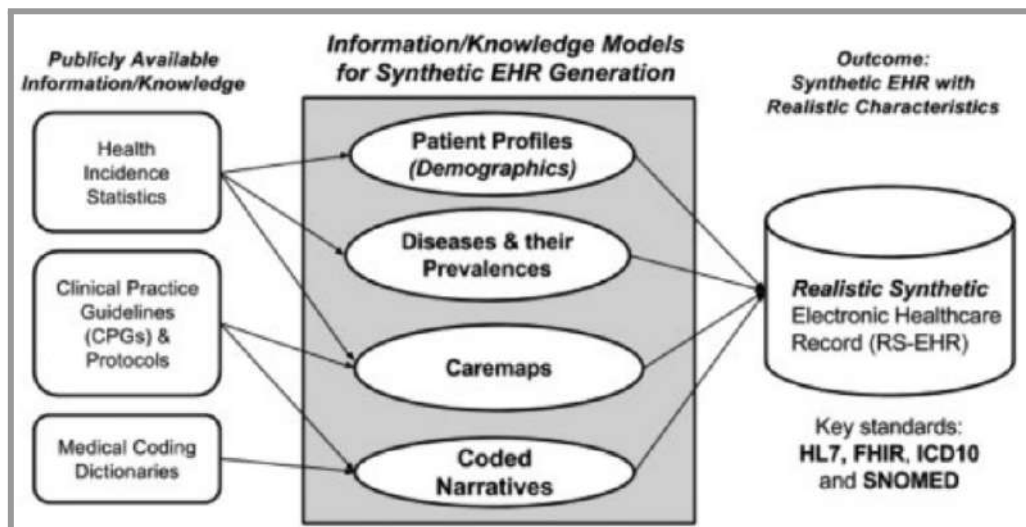


Figura 2: PADARSER como modelo del marco de referencia para Synthea[3]

De lo mencionado anteriormente, se observa que Synthea no considera en su estructura la característica que permitiría garantizar la seguridad de la información clínica, dado que el aplicativo no está diseñado para trabajar en ese segmento de la tecnología.

## 2.2.2 Microsoft AZURE FHIR server

El servidor FHIR de Microsoft Azure es una implementación de código abierto que utiliza el estándar HL7 Healthcare Interoperability Resources diseñado para la nube de Microsoft. El estándar FHIR define cómo los datos clínicos pueden ser interoperables entre los sistemas y el servidor FHIR para AZURE ayuda a facilitar la interoperabilidad en la nube. El principal objetivo de este proyecto de Microsoft es el de habilitar a los desarrolladores a implementar rápidamente servicios FHIR.

En esta aplicación de Microsoft Azure la privacidad y la seguridad son las principales prioridades con la finalidad de proteger la información clínica que usan los servicios de Azure.

Existen dos servicios ofrecidos por Azure; el primero es uno que generalmente está disponible y es llamado "Azure API for FHIR", el segundo es el "Azure Healthcare APIs". Este último incluye la funcionalidad para implementar un servidor FHIR y un DICOM(Digital Imaging and Communication On Medicine) en un simple espacio de trabajo. Estas plataformas FHIR as a Service(PaaS) están respaldadas por un proyecto de código abierto y ofrecen una solución llave en mano para aprovisionar un servicio FHIR seguro y confiable.[4]

El servicio de FHIR en Azure Healthcare APIs habilita rápidamente el intercambio de datos a través de las APIs FHIR, lo cual hace sencillo para los desarrolladores el consumir datos clínicos, gestionarlos y garantizar la protección de los datos personales en la nube. De entre los beneficios que ofrece Azure Healthcare API, tenemos los siguientes[5]:

- Servicio FHIR administrado, aprovisionado en la nube.
- Endpoint basado en FHIR de nivel empresarial en Azure para acceso a datos y almacenamiento en formato FHIR.
- Alto rendimiento, baja latencia.
- Gestión segura de información médica protegida (PHI) en un entorno de nube compatible.
- SMART en FHIR para implementaciones móviles y web.
- Control de acceso basado en roles (RBAC).
- Seguimiento de registro de auditoría para acceso, creación, modificación y lecturas dentro de cada almacén de datos.

Entre las características más importantes que buscamos para este trabajo, observamos que los servicios de Azure Healthcare ofrecen características destacables de Privacidad, Control de Acceso y Seguridad de la información clínica, lo que lo hace muy demandable por los desarrolladores, adicionalmente al estar alojado en la nube garantiza la confiabilidad del servicio.

### 2.2.3 Firely

El “Firely Server” es un servidor con un aplicativo basado en FHIR que está listo para usar, está construido en Microsoft .NET y se ejecuta en cualquiera de las plataformas para la que está disponible .NET Core Runtime: Linux, Windows, MacOS, Docker, etc. Este aplicativo servidor cuenta con las siguientes características para ser configuradas[6]:

- Permite elegir la Base de Datos que se quiere instalar, por defecto viene con SQLite.
- Permite configurar el nivel de validación de las solicitudes que se le hagan, teniendo la facilidad de configurar niveles de acceso flexibles o estrictos de acuerdo a la validez de las consultas que se le hagan.
- Se puede configurar la versión de FHIR que el usuario necesita en el aplicativo; cuenta con las versiones FHIR STU3 y FHIR R4.
- Se puede añadir una Licencia para su uso.
- Se pueden agregar y retirar complementos al aplicativo (Plugins).

Además de la configuración de los ajustes, el servidor Firely cuenta con una API de administración que permite la configuración de los “Recursos de Conformidad” que permiten el análisis, serialización, validación y terminología. La API de administración está cargada con recursos de conformidad, como las definiciones de estructura, los parámetros de búsqueda, los sistemas de código y los conjuntos de valores que vienen con la especificación FHIR. Adicionalmente, se puede usar la API de administración para que el servidor Firely pueda distinguir lo siguiente:

- Perfiles personalizados: Se pueden configurar restricciones nacionales o institucionales sobre los recursos.
- Recursos personalizados: Se pueden definir recursos más allá de los FHIR y se pueden gestionar como si fueran recursos FHIR estándar.

- Pueden albergar recursos CodeSystem y ValueSet para terminología.
- Parámetros de búsqueda personalizados: Se pueden configurar los parámetros de búsqueda para restringir las propiedades que no se pueden buscar.

## 2.2.4 Google FHIR (FhirProto)

FhirProto es una implementación de Google del estándar FHIR para datos de atención médica; esta implementación se ha hecho mediante Buffers de Protocolo (Lenguaje de Google). En esta aplicación se aprovecha el formato de almacenamiento de datos central de Google, lo que permite a FhirProto proporcionar un formato FHIR fuertemente validado y con seguridad de tipo compatible con varios idiomas en una fracción de tamaño de disco, lo que lo convierte en un excelente modelo de datos para desarrollar una aplicación de atención médica.

Los códigos estructurados y las extensiones de FhirProto garantizan que la información esté en el formato correcto. El soporte para generar y validar guías de implementación personalizadas permite personalizar FhirProto según su conjunto de datos y requisitos; además las bibliotecas de análisis e impresión facilitan alternar entre el formato FhirProto y JSON.

FhirProto usa un repositorio que contiene un script para usar Synthea que permite crear datos sintéticos en formato FHIR JSON [7].

## 2.3 Implementaciones de Seguridad en eHealth

Actualmente los avances en las tecnologías de la información y la comunicación mejoran distintos ámbitos de la prestación de servicios de salud modernos, además la implementación de sistemas de historias clínicas electrónicas es una parte fundamental de un sistema de eHealth. El término eHealth es generalmente utilizado para referirse a una aplicación que realiza la comunicación, intercambio, retención y eliminación de información de salud. Una aplicación importante de la tecnología en información clínica es la generación de Registros Médicos Electrónicos (EHR por sus siglas en inglés), el cual ofrece sustanciales beneficios con respecto de sus antecesores debido a que permite una recuperación rápida, simplicidad en la búsqueda, almacenamiento remoto, tiene la potencialidad de generar nuevos modelos de prestación de atención, intercambio de información pública, entre otros[46].

A pesar de los beneficios que ofrecen los EHRs hay ciertas consideraciones que se deben de tomar en cuenta durante la implementación de un sistema eHealth; entre ellos tenemos Seguridad, Privacidad y Control de Acceso los cuales permitirían cubrir principales retos de seguridad de la información según su prioridad dada por ENISA[47]:

- Disponibilidad del sistema.
- Falta de interoperabilidad.
- Control de acceso y autenticación.
- Integridad de la información.
- Seguridad en la red.
- Falta de expertise en la seguridad.
- Pérdida de de datos.
- Falta de cumplimiento y veracidad.
- Falta de estandarización.
- Incidentes de borde.

A continuación mostramos algunas herramientas de seguridad que se usan hoy en día y que permiten emular las condiciones de seguridad y privacidad de datos clínicos.

### 2.3.1 SMART on FHIR

El proyecto SMART por sus siglas en inglés “Substitutable Medical Applications and Reusable Technologies” ha trabajado una serie de soluciones que pueden construir y reutilizar las características de un sistema de información médica. Esta plataforma proporciona aplicaciones que pueden integrarse a los sistemas de salud de forma segura[36].

SMART también se integra con FHIR en su versión R4; el “framework” describe un conjunto de patrones fundamentales basados en OAuth 2.0 para que las aplicaciones cliente autoricen, autenticuen e integren con sistemas de datos basados en FHIR. Los patrones definidos en esta especificación se presentan en las secciones siguientes[37]:

- **Descubrimiento de capacidades y configuración del servidor:** SMART define un documento de descubrimiento disponible en `.well-known/smart-configuration` en relación con una URL base del servidor FHIR, lo que permite a los clientes conocer las URL del punto final de autorización y las características que admite un servidor. Esta información ayuda al cliente a dirigir las solicitudes de autorización al extremo



correcto y ayuda a los clientes a construir una solicitud de autorización que el servidor pueda admitir.

SMART define dos patrones para la autorización del cliente:

- **Autorización a través del lanzamiento de la aplicación SMART:** Autoriza una aplicación cliente orientada al usuario ("Aplicación") para conectarse a un servidor FHIR.
- **Autorización a través de SMART Backend Services:** Autoriza una aplicación de cliente sin cabeza o automatizada ("Servicio de back-end") para conectarse a un servidor FHIR.

SMART define dos patrones para la autenticación de clientes:

- **Autenticación asimétrica ("clave privada JWT"):** Autentica a un cliente mediante un par de claves asimétricas. Este es el método de autenticación preferido de SMART porque evita enviar un secreto compartido por cable.
- **Autenticación simétrica ("secreto del cliente"):** Autentica a un cliente utilizando un secreto que se ha compartido previamente entre el cliente y el servidor.

### 2.3.2 FHIRCast (HTTP Web Hooks using WebSub)

Este aplicativo sincroniza las aplicaciones de atención médica en tiempo real para mostrar el mismo contenido clínico a un usuario común; se basa en el modelo abstracto CCOW para especificar un modelo de sincronización de contexto simple y basada en http que no requiere un administrador de contexto separado. FHIRcast está destinado a ser menos sofisticado y más fácil de implementar que CCOW, por lo tanto, no es un reemplazo uno a uno de CCOW, aunque resuelve muchos de los mismos problemas.

Al adoptar la terminología WebSub, FHIRcast describe una aplicación como un suscriptor que se sincroniza con un EHR en el rol de Hub, pero cualquier aplicación orientada al usuario puede sincronizarse con FHIRcast. Si bien es menos común, también es posible la comunicación bidireccional entre múltiples aplicaciones.

Este aplicativo se puede implementar en conjunto con Smart FHIR; pero no es absolutamente necesario; en la figura 3 se muestra un ejemplo de interacción de esta aplicación [38].

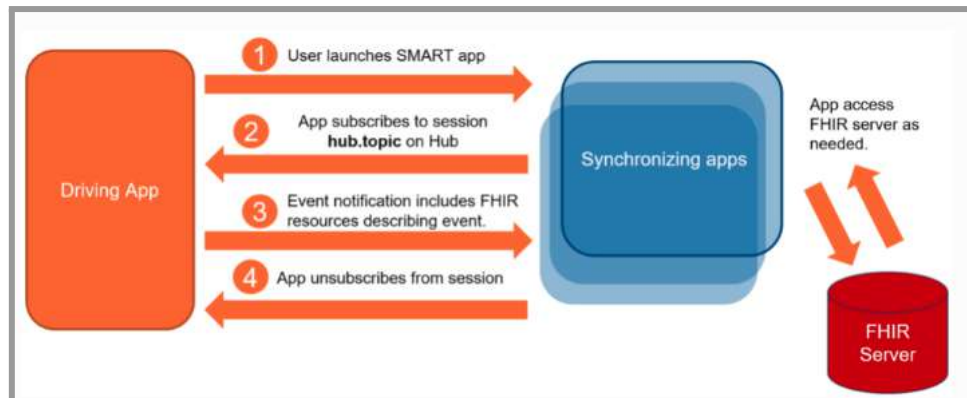


Figura 3: Modelo de interacción de flujos FHIRCast[38]

- Las notificaciones de eventos están basadas en JSON.
- La aplicación puede solicitar cambios de contexto enviando una notificación de evento al identificador de sesión `hub.topic` de Hub. El estado de la respuesta HTTP indica éxito o fracaso.
- El catálogo de eventos documenta los eventos de flujo de trabajo que se pueden comunicar en FHIRcast. Cada evento siempre llevará el mismo tipo de recursos FHIR.

### 2.3.3 Experimental Websockets support

Si bien la interfaz principal para los servidores FHIR es la API REST de FHIR, no es necesario que las notificaciones se realicen a través de REST. De hecho, es posible que algunos suscriptores no puedan exponer un servidor HTTP externo para recibir notificaciones activadas. Por ejemplo, una aplicación web o una aplicación móvil pura del lado del cliente puede querer suscribirse a una fuente de datos. Esto se puede lograr usando un canal de notificación websocket.

Si bien el uso de los Websocket está especificado en la versión R4 del estándar FHIR, actualmente no es muy utilizado para proveer una capa de seguridad a los aplicativos FHIR y su uso está en fase de pruebas [39].



## 3 Estándares y Tecnología utilizados

Para el desarrollo de este proyecto se ha tenido como referencia el estándar HL7 FHIR y el aplicativo HAPI basado en FHIR de los cuales se describirán sus características más esenciales que nos permitirán alcanzar los objetivos deseados; en este apartado inicialmente se muestra el análisis realizado de los estándares y las herramientas utilizadas para posteriormente entrar en temas más técnicos como el contenido del estándar utilizado en términos de seguridad y las características de seguridad del aplicativo HAPI; además se menciona la lista de herramientas que permitieron la implementación del aplicativo con el cual se dará cumplimiento a cada uno de los requerimientos del proyecto.

### 3.1 Análisis realizado

La elección de los estándares y tecnologías utilizados en este proyecto ha pasado por una serie de análisis que ha permitido poder discernir entre las diferentes opciones que existen actualmente.

En el análisis del estándar a utilizar, se optó por la revisión en HL7 que es el marco de estándares clínicos más utilizados hoy en día en el sector de la gestión de la información clínica en comparación de otros estándares como los de OpenEHR; de esta revisión se extrajeron las características de seguridad más viables de implementar, en la que el estándar FHIR de HL7 se antepuso a sus pares debido a su facilidad de implementación en comparación con HL7 v2, HL7 v3 y CDA.

Una vez elegido el estándar, se continuó con la búsqueda de un aplicativo que nos permita emular las características de seguridad viables de implementar en FHIR; se analizaron aplicativos como Synthea Patient Population Simulator, Microsoft AZURE FHIR Server, Firely, Google FHIR y HAPI FHIR; de los cuales el más adaptable a la implementación de este trabajo fue HAPI debido a que es un aplicativo basado en el estándar FHIR además que es un aplicativo de código abierto, que nos permite navegar entre su infraestructura interna y modificar sus características para adaptarlo a los requerimientos de este proyecto.

Una vez elegido el estándar FHIR y el aplicativo HAPI se procedió a un análisis más riguroso de los componentes de este último para validar los módulos que nos permitirán implementar el objetivo de seguridad de este proyecto. Inicialmente el enfoque era poder implementar seguridad de la información clínica basado en etiquetas, característica que según FHIR es viable de implementar, pero que llevado a la práctica con el aplicativo no se

encontró información de este módulo. De lo anterior, se encontró en la documentación de HAPI que contaba con módulos que nos permitían implementar seguridad en datos clínicos basado en Roles y Control de Acceso (RBAC) y que además está descrito como una de las características de seguridad de FHIR, por lo que se procedió a implementar este módulo en HAPI.

Finalmente, para poder desarrollar el trabajo que permitió el cumplimiento de los objetivos se procedió a realizar un análisis de las herramientas a utilizar, entre ellas se encontró una gran variedad de las cuales se hizo la elección en base a la experiencia del implementador.

## 3.2 Fast Healthcare Interoperability Resources(FHIR)

FHIR es un marco de estándares de próxima generación creado por HL7. FHIR combina las mejores características de las líneas de productos HL7 v2, HL7 v3 y CDA mientras aprovecha los últimos estándares web y aplica un enfoque estricto en la implementación.

Las soluciones FHIR se construyen a partir de un conjunto de componentes modulares llamados “Recursos”. Estos recursos se pueden ensamblar fácilmente en sistemas de trabajo que resuelven problemas clínicos y administrativos del mundo real a una fracción del precio de las alternativas existentes. FHIR es adecuado para su uso en una amplia variedad de contextos [12]:

- Aplicaciones de teléfono móvil.
- Comunicación en la nube.
- Intercambio de Datos basados en EHR.
- Comunicación de servidor en grandes proveedores de atención médica institucional.
- Entre otros.

Los recursos de FHIR están organizados en 5 módulos(niveles), cada uno de los módulos representa una diferente área funcional de la especificación. En el primer nivel tenemos los “framework” básicos en la cual está construida la especificación; el nivel 2 es el nivel que soporta implementaciones y enlaces de especificaciones externas, y es el nivel que nos permite implementar las características de seguridad que necesitamos para este proyecto; el nivel 3 genera vínculos con los objetos reales de un sistema de salud como por ejemplo: Paciente, Profesional, Dispositivo, Organización, Ubicación y Servicio de Salud; en el nivel 4 el estándar permite el mantenimiento de los registros y el intercambio para el proceso de atención médica; finalmente, el nivel 5 permite contar con el conocimiento clínico en dónde

se guardan los reportes, definiciones, documentación del proceso, entre otros. En la figura 4 se definen los niveles de la estructura del estándar, en dónde se resalta el nivel en el que se encuentra el apartado de seguridad y privacidad de la información.

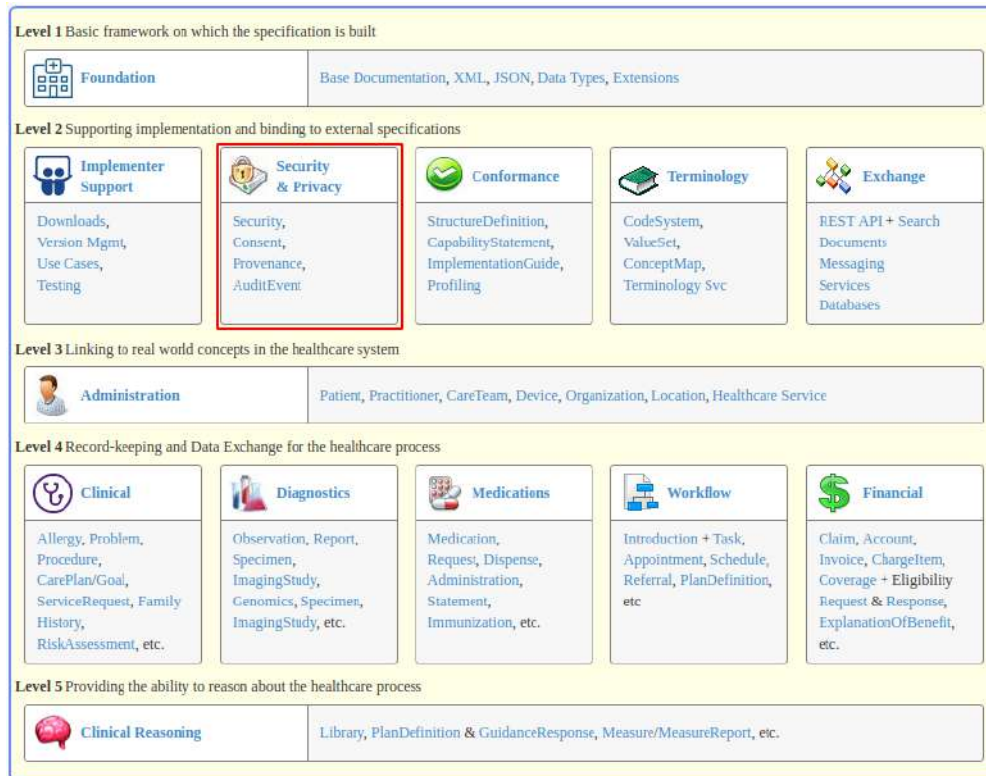


Figura 4: Niveles definidos en el estándar FHIR [27]

Los casos de uso que es posible implementar usando HL7 FHIR son los siguientes:

- Seguridad y privacidad.
- Autorización y control de acceso.
- Consideraciones de autorización con parámetros de consulta.
- Identidad de usuario y contexto de acceso.
- Registro de auditoría de seguridad y privacidad.
- Contabilidad de divulgaciones e informes de acceso.
- Consentimiento de privacidad.
- Procedencia.
- Firmas digitales y electrónicas.
- Desidentificación, anonimización y seudonimización.
- Consideraciones de seguridad sobre los datos de prueba.

### 3.1.1 Seguridad

FHIR se focaliza en los métodos de acceso a datos y la codificación aprovechando las soluciones de seguridad existentes. La seguridad en FHIR debe centrarse en el conjunto de consideraciones necesarias para garantizar que los datos puedan descubrir, acceder o modificar solo de acuerdo con las expectativas y las políticas. La implementación debe aprovechar los estándares de seguridad y las implementaciones existentes para garantizar lo siguiente:

- Todas las comunicaciones se pueden cifrar para evitar el acceso no autorizado.
- No hay fugas de información cuando se producen errores.
- No se puede inyectar contenido de “script activo” en los recursos narrativos.
- Se pueden construir y utilizar pistas de auditoría completas para detectar patrones de acceso anómalos.

### 3.1.2 Privacidad

La privacidad en FHIR incluye el conjunto de consideraciones requeridas para garantizar que los datos individuales se procesen de acuerdo con los Principios de privacidad individual y de privacidad por diseño.

FHIR incluye la siguiente guía de implementación para garantizar que:

- Las preferencias individuales se pueden comunicar a través de protocolos basados en estándares (por ejemplo, OAuth, acceso administrado por el usuario) o mediante una representación FHIR explícita (Consentimiento).
- Los recursos se pueden etiquetar para indicar la sensibilidad o confidencialidad de los datos que representan (Etiquetas de seguridad).
- Los registros de acceso a datos y los registros de auditoría se pueden compartir con personas, por ejemplo para contabilidad de revelaciones (Evento de Auditoría).

### 3.1.3 Módulo de Seguridad y Privacidad en FHIR

El módulo de seguridad y privacidad describe cómo proteger la información de un servidor FHIR(A través del control de acceso y autorización), cómo documentar los permisos que se ha otorgado a un usuario(Consentimiento) y cómo mantener registros sobre los eventos que se han realizados(Registro de auditoría y procedencia). FHIR no exige un enfoque técnico

único para la seguridad y la privacidad; de hecho la especificación proporciona un conjunto de componentes básicos que se pueden aplicar para crear sistemas privados y seguros [13].

Para la implementación de un sistema FHIR se necesitará un subsistema de seguridad que administre, autentifique y autorice el acceso a los datos de los usuarios; el lugar dónde se ubicará este subsistema será dependiendo del diseño y la arquitectura del sistema; en la figura 5 se muestra el diagrama de la arquitectura de seguridad en FHIR.

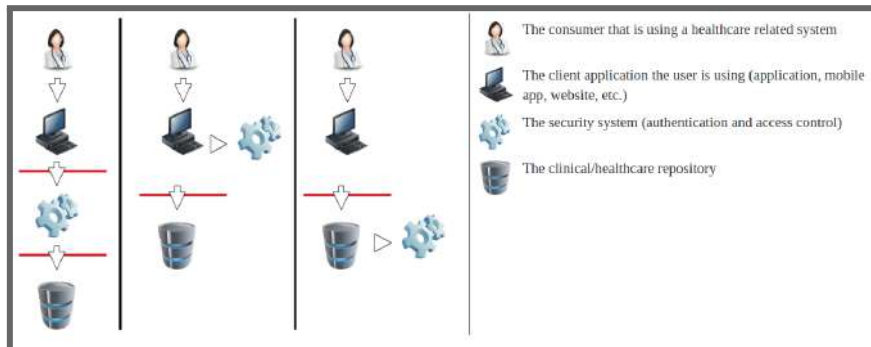


Figura 5: Diagrama arquitecturas de seguridad [26]

En el diagrama anterior se muestran unas líneas de color rojo que representan las interfaces FHIR, el cliente (consumidor de los servicios FHIR) puede interactuar con un sistema de seguridad que se manifiesta como un servidor FHIR y que depende de una interfaz FHIR posterior para proporcionar el almacenamiento real, o bien el cliente o el servidor interactúa con el sistema de seguridad de forma independiente. En cada uno de estos 3 escenarios, los diferentes componentes pueden ensamblarse en aplicaciones o componentes de red de manera diferente, pero se aplica el mismo diseño lógico. La especificación FHIR asume que existe un sistema de seguridad y que puede implementarse delante o detrás de la API FHIR[26].

Los módulos de seguridad y privacidad contienen los siguientes materiales:

### 3.1.3.1 Recursos

Los recursos utilizados para poder atender la funcionalidad de seguridad y privacidad de FHIR son las siguientes:

- Consent:** Contiene un registro de las elecciones de seguridad de un usuario de atención médica. Este módulo permite o deniega el acceso a la información de un usuario; para ello utiliza registro de usuarios y roles para realizar una o más



acciones dentro del contexto de una política determinada, para propósitos o periodos de tiempo específicos. Por ejemplo:

- **Directiva de procedimiento de privacidad:** Acuerdo para recopilar, acceder, usar o divulgar la información.
  - **Directiva de consentimiento de tratamiento médico:** Consentimiento para someterse a un tratamiento en específico.
  - **Directiva de consentimiento de investigación:** Se requiere consentimiento para participar en el protocolo de investigación e intercambio de información.
  - **Instrucciones anticipadas de Atención:** Consentimiento para recibir instrucciones sobre tratamientos médicos potencialmente necesarios.
- **Procedencia:** La procedencia de un recurso es un registro que describe las entidades y los procesos involucrados en la producción y entrega o que de otro modo influyen en ese recurso. La procedencia proporciona una base fundamental para evaluar la autenticidad, permitir la confianza y permitir la reproducibilidad. Las afirmaciones de procedencia son una forma de metadatos contextuales y pueden convertirse en registros importantes con su propia procedencia.

La declaración de procedencia indica la importancia clínica en términos de confianza en la autenticidad, confiabilidad, integridad y la etapa del ciclo de vida.

- **Eventos de Auditoría:** Es un registro de un evento realizado con el fin de mantener un registro de seguridad. Los usos típicos incluyen la detección de intentos de intrusión y el control del uso inapropiado.

### 3.1.3.2 Tipos de Datos

- **Firma:** Contiene una representación electrónica de una firma y su contexto de soporte en una forma accesible FHIR. La firma puede ser de tipo criptográfico (XML DigSig o JWS), que puede proporcionar una “prueba de no repudio”, o puede ser una imagen gráfica que representa una firma o un proceso de firma.

### 3.1.3.3 Guías y Principios de Implementación

- **Principios de Seguridad:** Fast Healthcare Interoperability Resources (FHIR) no es un protocolo de seguridad ni define ninguna funcionalidad relacionada con la



seguridad. Sin embargo, FHIR define protocolos de intercambio y modelos de contenido que deben usarse con varios protocolos de seguridad definidos en otros lugares.

A continuación un resumen de los métodos de seguridad utilizados por FHIR:

- Tiempo de vida.
  - Autenticación.
  - Autorización/Control de Acceso.
  - Auditoría.
  - Firmas digitales.
  - Adjuntos.
  - Etiquetas.
  - Narrativa.
  - Validación de entrada.
- 
- **Firmas:** Esta especificación recomienda el uso de firmas digitales W3C o firmas digitales JSON para firmas digitales. Los recursos se pueden firmar utilizando el recurso Procedencia para llevar una firma digital independiente. El tipo de datos Signature está disponible para admitir varios tipos de firma, incluidos los fines de no repudio.
  - **Etiquetas de Seguridad:** Una etiqueta de seguridad es un concepto adjunto a un recurso o paquete que proporciona metadatos de seguridad específicos sobre la información a la que está fijado. El motor de decisiones de control de acceso utiliza la etiqueta de seguridad junto con los recursos de procedencia asociados con el recurso y otros metadatos (p. ej. el tipo de recurso, el contenido del recurso, etc.).

Entre los tipos de restricciones basados en etiquetas tenemos:

- Aprobar lectura, cambio y/o otras operaciones.
- Determinar qué tipo de información puede ser retornada.
- Determinar qué advertencias de gestión se deben de transmitir con los datos.

A continuación se muestran las etiquetas disponibles en FHIR para el sistema de códigos y conjunto de valores HCS(Healthcare System), incluye todos los códigos

de los cuales las implementaciones de SLS y PPS deberán seleccionar el campo de etiqueta de seguridad y los valores de etiqueta utilizados en documentos, mensajes y recursos FHIR[1]; en la tabla 2 se muestra la descripción de las etiquetas definidas en FHIR.

<b>Etiqueta de Seguridad</b>	<b>Descripción</b>
<b>Clasificación de la confidencialidad</b>	<p>Metadatos de etiquetas de seguridad que clasifican un recurso de TI (Hecho clínico, datos, objeto de información, servicio o capacidad del sistema) según su nivel de sensibilidad, que se basa en un análisis de las políticas de privacidad aplicables y el riesgo de daños financieros, de reputación o de otro tipo a una persona o entidad que podría resultar si se pone a disposición o se revela a personas, entidades o procesos no autorizados.</p> <p>Ejemplos de usos: sin restricciones, normal, muy restringido.</p>
<b>Categoría Sensibilidad</b>	<p>Metadatos de etiquetas de seguridad que "segmentan" un recurso de TI al categorizar el valor, la importancia y la vulnerabilidad de un recurso de TI que se percibe como no deseable para compartir.</p> <p>Ejemplos de usos: enfermedades de transmisión sexual, atención psiquiátrica, estatus de celebridad.</p>
<b>Categoría Compartimiento</b>	<p>Metadatos de etiquetas de seguridad que "segmentan" un recurso de TI al indicar que el acceso y el uso está restringido a miembros de una comunidad o proyecto definido.</p> <p>Nota: este es un uso diferente de "Compartimiento" al uso del Compartimiento del paciente.</p> <p>Ejemplos de uso: investigación, registros de recursos humanos.</p>
<b>Categoría Integridad</b>	<p>Metadatos de etiquetas de seguridad que "segmentan" un recurso de TI transmitiendo la integridad, veracidad, confiabilidad, confiabilidad y procedencia de un recurso de TI.</p> <p>Ejemplos de usos: anónimo, firmado, informado por el paciente.</p>
<b>US Privacy Law</b>	<p>Metadatos de etiquetas de seguridad que "segmentan" un Recurso TI indicando las disposiciones legales a los que se ajusta la asignación de una Clasificación de Confidencialidad en los EE.UU.</p>



	Las jurisdicciones pueden desarrollar leyes de privacidad específicas del ámbito o códigos de categorías de políticas para su uso en etiquetas de seguridad en sus dominios.
<b>Advertencia de manejo</b>	Metadatos de etiquetas de seguridad que transmiten controles de difusión y manejo de información, instrucciones tales como obligaciones y políticas de abstención a las que un custodio de recursos de TI o el receptor debe cumplir. Este tipo de advertencia de manipulación DEBERÁ asignarse a un hecho clínico si así lo requiere política jurisdiccional u organizacional, que puede activarse mediante el consentimiento de un paciente directiva Ejemplos de usos: no divulgar, diversas restricciones de uso y marcas de política.

Tabla 2: Descripción de las categorías de las etiquetas de seguridad FHIR [1]

### 3.1.4 Características de FHIR a implementar

En los acápites anteriores se mencionan las características de FHIR en términos de seguridad y privacidad de la información, con esta información se procedió a validar la viabilidad de implementación.

El análisis se inició con la consigna de lograr un sistema de seguridad basado en etiquetas; sin embargo, del análisis a la documentación del aplicativo HAPI nos encontramos que no estaba especificado para su versión más actual.

De lo anterior, se continuó con el análisis de las características FHIR obteniendo como resultado que el modelo de seguridad más viable para su implementación en HAPI es el de Autorización/Control de Acceso usando “Role-Based Access Control” (RBAC).

## 3.2 HAPI FHIR

HAPI FHIR, producto de la empresa Smile CDR, es una implementación de código abierto de la especificación HL7 FHIR para Java. Es una aplicación que permite intercambiar datos clínicos de una forma moderna y fácil de ser implementada por los desarrolladores; esta aplicación está diseñada con una intención principal: Proveer un modo flexible de añadir características FHIR al funcionamiento.

HAPI FHIR proporciona una serie de mecanismos para implementar un servidor FHIR, debido a que cuenta con un numeroso grupo de tipos de servidores que pueden ser usados por los implementadores para su instalación[8]. Entre estos tenemos:

- **Plain Server:** Es una implementación de un servidor FHIR con un backend arbitrario que el usuario proporciona. En este modo, el usuario escribe el código que maneja el almacenamiento de recursos y la lógica de recuperación, y HAPI FHIR se encarga de lo siguiente:
  - Procesamiento HTTP.
  - Análisis / Serialización.
  - Semántica FHIR REST.
- **JPA Server:** El servidor HAPI FHIR JPA es una implementación completa de FHIR contra una base de datos relacional; a diferencia del servidor simple, el servidor JPA proporciona su propio esquema de base de datos y gestiona toda la lógica de almacenamiento y recuperación sin necesidad de de codificación.
- **JAX-RS Server:** La implementación de este servidor es como un JEE estándar, lo que significa que se puede implementar en cualquier contenedor de web compatible con JEE. El módulo JAX-RS es un módulo respaldado por la comunidad que no fue desarrollado por el equipo central de HAPI FHIR.

En la figura 6 se muestran algunos ejemplos de arquitecturas que se pueden configurar en HAPI FHIR.

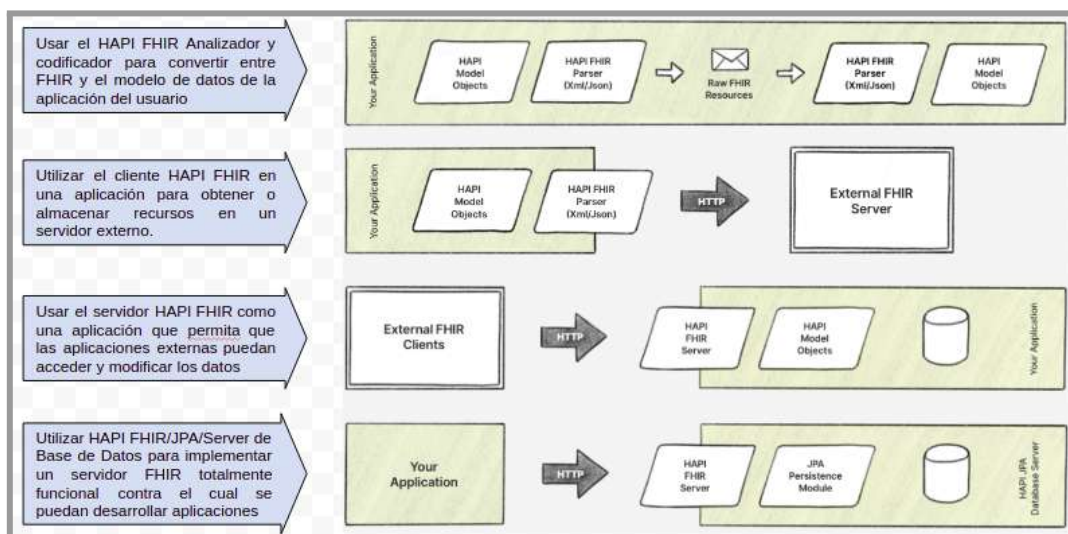


Figura 6: Diferentes configuraciones que pueden darse a HAPI FHIR [8]

### 3.2.1 Arquitectura HAPI FHIR JPA Server

La versión que utilizaremos para el proyecto es HAPI FHIR JPA Server debido a que es la versión más completa del proyecto HAPI, soporta todas las operaciones estándares(read/create/ delete, etc.). Este aplicativo contiene una instancia de base de datos por defecto por lo que este servidor puede ejecutarse sin la necesidad de una base de datos externa; pero también puede utilizar una base de datos externa si es que el desarrollador así lo requiere; esta arquitectura tiene los siguientes componentes [14].

- **Proveedores de Recursos:** Un servidor proveedor de recursos RESTful es proporcionado para cada tipo de recurso en la versión determinada de FHIR que se esté trabajando. Cada proveedor de recursos implementa un método “@Search” que implementa el conjunto completo de parámetros de búsqueda definidos en la especificación FHIR para el tipo de recurso dado. Los proveedores de recursos también amplían una superclase que implementa todos los demás métodos de FHIR como: Leer, Crear, Eliminar, etc.
- **HAPI DAOs:** Los DAO implementan toda la lógica comercial de la base de datos relacionada con el almacenamiento, la indexación y la recuperación de recursos FHIR, utilizando la API JPA.
- **Hibernate:** El servidor HAPI JPA usa la librería JPA implementada por Hibernate. No se utilizan funciones específicas de Hibernate, por lo que la biblioteca también debería funcionar con otros proveedores.
- **Database:** El servidor RESTful server usa una base de datos Derby integrada, pero se puede configurar para poder usar una base de datos externa.

En la figura 7 se muestra la arquitectura a alto nivel de HAPI JPA server.

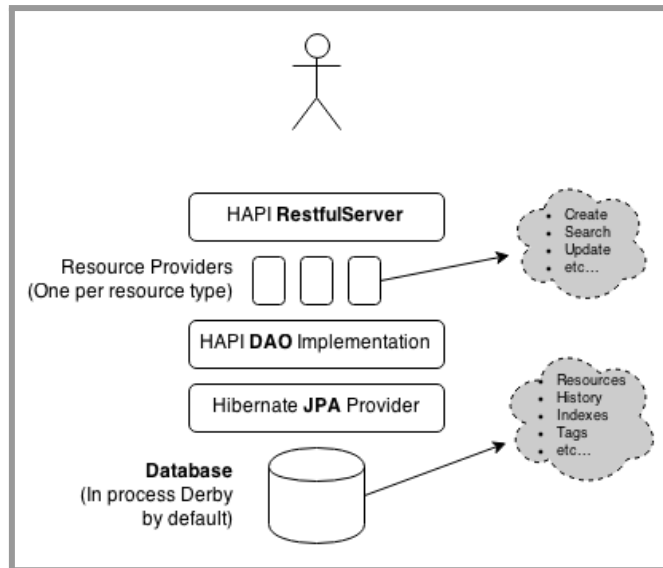


Figura 7: Arquitectura del aplicativo HAPI FHIR JPA a alto nivel[14]

### 3.2.2 Seguridad

La seguridad es un tema complejo que va mucho más allá del alcance de FHIR o HAPI ya que cada sistema y arquitectura opera en un conjunto diferente de reglas y tiene diferentes requisitos de seguridad. HAPI FHIR no proporciona una única capa de seguridad completa del sistema; en su lugar proporciona una serie de herramientas útiles y bloques de construcción que se pueden construir como parte de su arquitectura de seguridad general.

Dado que el servidor REST de HAPI FHIR se basa en la API Servlet, puede utilizar cualquier mecanismo de seguridad que funcione en ese entorno; algunos contenedores de servlet pueden proporcionar capas de seguridad a las que puede conectarse [16].

La seguridad en HAPI FHIR se divide en 3 tópicos:

- **Authentication (AuthN):** Se usa para identificar si el usuario es quien dice ser, se logra probando un nombre de usuario y contraseña en la solicitud.
- **Authorization (AuthZ):** Sirve para verificar si un usuario tiene permiso para realizar alguna acción. Por ejemplo en una aplicación FHIR se puede usar esta funcionalidad para probar que el usuario que intenta realizar una solicitud al servidor FHIR tiene el permiso de acceder a este, con lo cual si es que el usuario tiene permiso se le permite el acceso; en caso contrario, puede bloquearse la operación que el usuario desea realizar sobre el servidor FHIR. Este ejemplo es un AuthN y AuthZ.



- **Consent and Audit:** Se usa para verificar que un usuario tiene derechos para leer/modificar los recursos específicos que solicita, aplicando directivas para enmascarar los datos son devueltos al cliente(ya se de forma parcial o completa) y crea un registro de que ocurrió el evento.

Para la aplicación de seguridad en HAPI FHIR se utiliza el concepto de Interceptors; entre ellos tenemos:

- Authentication Interceptors.
- Authorization Interceptors.
- Consent Interceptors.
- Search Narrow Interceptors.
- CORS Interceptor.

### 3.2.3 Interceptor

La versión 3.8.0 de HAPI FHIR introduce el interceptor “framework” que es usado por toda la librería; los interceptors permiten “engancharse” a varios puntos de la cadena de procesamiento tanto en aplicativos cliente y servidor; y la función principal de estos objetos que son capaces de interponerse a la ejecución de los métodos y priorizar la realización de acciones antes o después de la ejecución del método [15].

El esquema de un interceptor en HAPI FHIR utiliza los siguientes términos clave para su funcionamiento.

- **Interceptor:** Una clase que tiene uno o más métodos Hook. La anotación de los interceptores puede anotarse o no con el término @Interceptor.
- **Hook:** Es un método de interceptor individual que se invoca en respuesta a una acción específica que tiene lugar en el marco de HAPI FHIR. Los métodos Hook deben anotarse con el término @Hook.
- **Pointcut:** Es un punto específico en la canalización de procesamiento de HAPI FHIR que está siendo interceptado. Cada método Hook debe declarar que punto de corte está interceptando, todos los puntos de corte disponibles están definidos por HAPI FHIR en la enumeración Pointcut.

- Hook Params:** Cada Pointcut define una lista de parámetros que se pueden pasar a un método Hook para un Pointcut determinado.

En la figura 8 se muestra un diagrama de la canalización de procesamiento para procesar una solicitud al servidor.

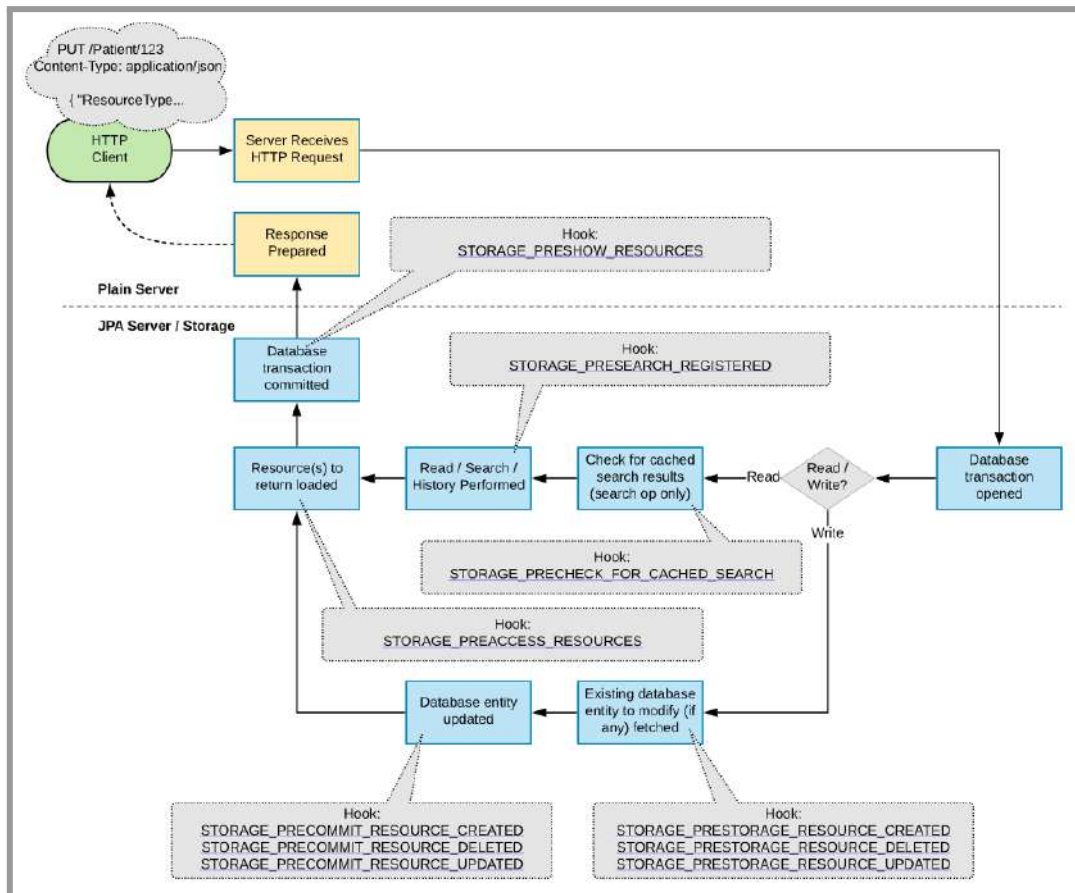


Figura 8: Diagrama de canalización HAPI JPA Server [15]

### 3.2.4 Características HAPI a utilizar

Del análisis realizado en el apartado 3.2 se observa que el aplicativo HAPI cuenta con un set de módulos de seguridad que pueden ser utilizados para la implementación de seguridad y privacidad de la información requerida en este proyecto.

Realizando la comparación de la información en HAPI con el estándar FHIR se observó que el modelo de seguridad más viable de implementar es el modelo de Autorización/Control de Acceso; y la implementación se adapta perfectamente a través del módulo "Authorization Interceptor" definido en HAPI.

### 3.3 Herramientas de software

Para lograr el cumplimiento de los objetivos del proyecto se modificará el aplicativo HAPI FHIR agregando código, lo que permitirá su configuración deseada para el proyecto; adicionalmente, se generará un aplicativo “Front End” y ”Back End” para lo cual se usará herramientas que a continuación se describen.

- **IntelliJ IDEA:** Es un entorno de desarrollo integrado (IDE) que permite el desarrollo de programas informáticos, este IDE es desarrollado por JetBrains. Este IDE permite trabajar en Java y otros lenguajes JVM como Kotlin, Scala y Groovy en todo tipo de aplicaciones. Además, IntelliJ IDEA Ultimate puede ayudarlo a desarrollar aplicaciones web completas, gracias a sus potentes herramientas integradas, soporte para JavaScript y tecnologías relacionadas, y soporte avanzado para marcos populares como Spring, Spring Boot, Jakarta EE, Micronaut, Quarkus, Helidon. Además, puede ampliar IntelliJ IDEA con complementos gratuitos desarrollados por JetBrains, lo que le permite trabajar con otros lenguajes de programación, incluidos Go, Python, SQL, Ruby y PHP [19].
- **Postman:** Postman es una plataforma API para construir y usar API. Postman simplifica cada paso del ciclo de vida de la API y agiliza la colaboración para que pueda crear mejores API, y más rápido. Postman puede almacenar y administrar especificaciones de API, documentación, recetas de flujo de trabajo, casos de prueba y resultados, métricas y todo lo demás relacionado con las API. La plataforma de Postman incluye un conjunto completo de herramientas que ayudan a acelerar el ciclo de vida de la API, desde el diseño, las pruebas, la documentación, y burlarse del uso compartido y la capacidad de descubrimiento de sus API.

Postman se integra con las herramientas más importantes en su proceso de desarrollo de software para habilitar las prácticas API-first. La plataforma Postman también es extensible a través de la API de Postman y mediante tecnologías de código abierto [20].

- **Netbeans 12.0:** Es un ambiente de desarrollo de código abierto hecho inicialmente para el lenguaje de programación Java, y que actualmente se extiende hacia otros lenguajes de programación. La plataforma de Netbeans permite desarrollar aplicaciones a partir de un conjunto de componentes llamados módulos; un módulo





es un archivo de Java que contiene clases de Java y que permiten interactuar con las APIs de Netbeans [21].

- **GitHub:** Es una plataforma que permite a los desarrolladores generar repositorios de código y guardarlos en la nube, actualmente propiedad de Microsoft. El objetivo principal de GitHub es brindar a los desarrolladores la posibilidad de poder guardar el código de sus aplicaciones usando un sistema de control de versiones llamado GIT. En este proyecto se utilizará para realizar las actualizaciones del código del aplicativo HAPI FHIR [22].
- **Apache Maven Project:** Apache Maven es una herramienta de comprensión y gestión de proyectos de software que nos permite simplificar los procesos de build (Compilar y generar ejecutables a partir del código fuente). Basado en el concepto de un modelo de objetos de proyecto (POM), Maven puede administrar la construcción, los informes y la documentación de un proyecto desde una pieza central de información. En este proyecto se usará para los procesos build del aplicativo HAPI FHIR [23].
- **Jetty:** Es un motor de servlet ligero y altamente escalable basado en Java que admite protocolos web como HTTP, HTTP/2 y WebSocket en una forma de baja latencia de alto volumen que brinda el máximo rendimiento. Jetty es un servidor web moderno totalmente asíncrono que tiene una larga historia como una tecnología orientada a componentes que se integra fácilmente en las aplicaciones y, al mismo tiempo ofrece una distribución tradicional sólida para la implementación de aplicaciones web[35]. En el proyecto el aplicativo HAPI FHIR será instalado usando Jetty.
- **Lenguaje Unificado de Modelado (UML):** Lenguaje de modelado visual común y semántica y sintácticamente rico para la arquitectura, el diseño y la implementación de sistemas de software complejos, tanto en estructura como en comportamiento; los diagramas UML describen los límites, la estructura y el comportamiento del sistema y los objetos que contiene.

UML no es un lenguaje de programación, pero existen herramientas que se pueden usar para generar código en diversos lenguajes usando los diagramas UML; UML guarda una relación directa con el análisis y el diseño orientados a objetos [40].



- **JSON(JavaScript Object Notation) y XML(Extensible Markup Language):** JSON es un formato para guardar e intercambiar información legible por el ser humano. La representación JSON para un recurso se basa en el formato JSON descrito en el estándar STD 90 (RFC 8259).

JSON se describe usando el formato de la figura 9.

```
{
  "resourceType" : "[Resource Type]",
  // from source: property0
  "property1" : "<[primitive]>", // short description
  "property2" : { [Data Type] }, // short description
  "property3" : { // Short Description
    "propertyA" : { CodeableConcept }, // Short Description (Example)
  },
  "property4" : [{ // Short Description
    "propertyB" : { Reference(ResourceType) } // R! Short Description
  }]
}
```

Figura 9: Formato JSON FHIR [24]

XML es el acrónimo de Extensible Markup Language, es decir, es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos. XML se describe usando el formato de la figura 10.

```
<name xmlns="http://hl7.org/fhir" (attrA="value")>
  <!-- from Resource: id, meta, implicitRules, and language -->
  <nameA><!-- 1..1 type description of content --></nameA>
  <nameB[x]><!-- 0..1 type1|type1 description --></nameB[x]>
  <nameC> <!-- 1..* -->
    <nameD><!-- 1..1 type>Relevant elements --></nameD>
  </nameC>
</name>
```

Figura 10: Formato XML FHIR [30]

- **HAPI FHIR RESTful:** FHIR es descrito como una especificación “RESTful” de acuerdo a cómo la industria define el término REST[17]. En la práctica FHIR solamente soporta el nivel 2 del modelo de madurez de REST[18] como parte de su especificación central, aunque es posible usar el nivel 3 con el uso de extensiones. En la figura 11 se muestra un diagrama UML de la interfaz RESTful de HAPI.

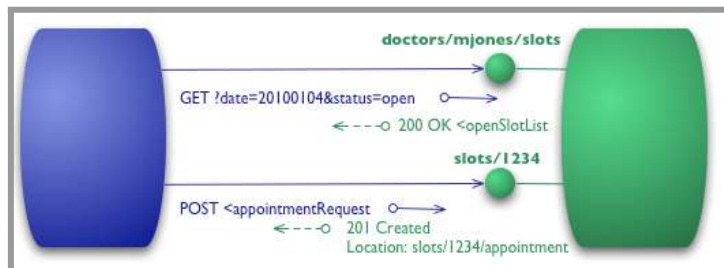


Figura 11: Modelo UML de FHIR RESTful [18]

La API describe a los recursos de FHIR como un set de operaciones en recursos llamadas “Interacciones”, donde una instancia de un recurso individual es gestionado en colecciones por su tipo. Los servidores pueden elegir cual de estas interacciones están disponibles y qué tipo de recursos soportan. En la tabla 3 se encuentran la definición de las interacciones en FHIR.

<b>Instance Level Interactions</b>	
<b>read</b>	Lee el estado actual del recurso
<b>vread</b>	Lee el estado de una versión específica del recurso
<b>update</b>	Actualiza un recurso existente, tomando como referencia su id (O lo crea si este es nuevo)
<b>patch</b>	Actualiza un recurso existente publicando un conjunto de cambios en el recurso
<b>delete</b>	Elimina un Recurso
<b>history</b>	Recupera el historial de cambios de un recurso en particular
<b>Type Level Interactions</b>	
<b>create</b>	Crea un nuevo recurso con una identificación asignada por el servidor
<b>search</b>	Busca el tipo de recurso en función de algunos criterios de filtro

<b>history</b>	Recupera el historial de cambios para un tipo de recurso en particular
<b>Whole System Interactions</b>	
<b>capabilities</b>	Obtiene una declaración de capacidad para el sistema
<b>batch/transaction</b>	Actualiza, crea o elimina un conjunto de recursos en una sola interacción
<b>history</b>	Recupera el historial de cambios de todos los recursos
<b>search</b>	Busca en todos los tipos de recursos en función de algunos criterios de filtro

Tabla 3: Interacciones RESTful HAPI FHIR [17]

En términos de seguridad en la especificación FHIR el uso de HTTP es opcional, pero todo el intercambio de datos clínicos deben de hacerse usando SSL y seguridad adicional. La mayoría de las aplicaciones requerirán la autenticación de los usuarios, y todas las operaciones que lo hagan están sujetas a RBAC [28] y/o ABAC[29], y algunas operaciones pueden depender de que se otorgue el consentimiento adecuado.

Los servidores pueden rechazar las solicitudes de información debido a problemas de integridad o de reglas comerciales; a continuación mostramos los códigos de estado HTTP comunes devueltos en errores relacionados con FHIR; además de los errores HTTP normales relacionados con problemas de negociación de seguridad, encabezado y tipo de contenido; en la tabla 4 se define los códigos de estado HTTP en FHIR.

Code	Name	Description
400	Bad Request	El recurso no se pudo analizar o falló las reglas básicas de validación de FHIR (o se encontraron múltiples coincidencias para los criterios condicionales)
401	Unauthorized	El intento de acceso por parte del usuario falló
403	Forbidden	El acceso no fue autorizado, el usuario no tiene acceso al recurso
404	Not Found	Tipo de recurso no compatible, o no es un punto final FHIR
405	Method Not Allowed	El recurso no existía antes de la actualización y el servidor no permite identificadores definidos por el cliente
409/412	-	Gestión de conflictos de versión
422	Unprocessable Entity	El recurso propuesto violó los perfiles FHIR aplicables o las reglas comerciales del servidor

Tabla 4: Códigos de estado HTTP en FHIR[26]

- **Sistema Operativo y Recursos de Hardware:** El sistema operativo que se usará como base para la implementación es el Ubuntu en su versión 20.04; adicionalmente se muestra el detalle de este y los recursos de hardware disponibles serán los que se muestran en la figura 12.

Memory	15,3 GiB
Processor	Intel® Core™ i7-10510U CPU @ 1.80GHz × 8
Graphics	Mesa Intel® UHD Graphics (CML GT2)
Disk Capacity	1,0 TB
OS Name	Ubuntu 20.04.3 LTS
OS Type	64-bit

Figura 12: Características del servidor a utilizar

## 4 Diseño de Casos de Uso

Este proyecto tiene dos objetivos específicos importantes, el primero es el análisis del estándar FHIR y del aplicativo HAPI basado en FHIR para la definición de características de seguridad. El segundo de ellos es generar casos de uso para emplearlos como ejemplo en un demostrador; lo que nos permitirá validar la usabilidad de la herramienta HAPI FHIR en términos de seguridad y privacidad.

De acuerdo a la definición del segundo objetivo, que también incluye contar con un demostrador que realiza dos funcionalidades principales, se generan dos apartados para los casos de uso; el primero será para la funcionalidad de búsqueda de pacientes utilizando el ID de paciente y el segundo grupo de casos de uso para la funcionalidad de conteo de pacientes según criterios de búsqueda .

Para definir el primer apartado de casos de uso se tomó como referencia el análisis realizado de FHIR y de la herramienta HAPI. Se determinó implementar un módulo de “Autorización/Control de Acceso” debido a que era una de las características de seguridad de FHIR que más se ajustaba para poder implementarlo en HAPI debido a la información contenida en la documentación del aplicativo. Del estándar se tomó el modelo de control de acceso “Role-Based Access Control” (RBAC) y del HAPI la herramienta “Authorization Interceptor” con el cual se implementó una lógica en el mismo aplicativo a partir de los siguientes casos de uso:

- **Caso de uso 1: Autoconsulta de la información del paciente**  
El paciente podrá acceder a toda su información.
- **Caso de uso 2: Consulta de la información de un paciente por parte de otro paciente**  
El acceso a la información de un paciente estará totalmente restringida para los otros pacientes.
- **Caso de uso 3: Médico consulta la información de su paciente**  
El médico del paciente podrá acceder a toda la información de cada uno de sus pacientes.

- **Caso de uso 4: Médico consulta información de pacientes de otros médicos**  
Solamente el médico del paciente podrá acceder a la información del paciente; no se permite el acceso a dicha información por parte de otros médicos.
- **Caso de uso 5: La clínica u hospital consulta la información de sus pacientes**  
La organización dónde el paciente es atendido solamente podrá acceder a los datos esenciales del paciente, no podrá acceder a la información de su condición médica.
- **Caso de uso 6: La clínica u hospital consulta la información de pacientes que no tiene registrados**  
La organización no podrá acceder a alguna información del paciente si esta no es la institución dónde el paciente es atendido.

En la tabla 5 se muestra el resumen de los casos de uso que se utilizarán para validar las características de seguridad que se implementaran en el aplicativo y que servirán de referencia para validar el resultado de este trabajo.

Rol	Casos de Uso	ID	Nombres	Edad	Nacimiento	Médico	Observación
Paciente	Auto-consulta	Si	Si	Si	Si	Si	Si
	Consulta a otro paciente	No	No	No	No	No	No
Médico	Consulta a su paciente	Si	Si	Si	Si	Si	Si
	Consulta a otro paciente	No	No	No	No	No	No
Organización	Consulta a su paciente	Si	Si	Si	Si	Si	No
	Consulta a otro paciente	No	No	No	No	No	No

Tabla 5: Reglas establecidas para la visualización de información de pacientes

Los casos de uso anteriores fueron elegidos a criterio del que desarrolló este trabajo, siendo importante mencionar que el diseño de las restricciones al acceso de la información de los pacientes puede ser modificado a criterio de las regulaciones de cada país y/o normas gubernamentales, necesidades de la institución, entre otros.

Para el segundo apartado se definirán casos de uso relacionados con el conteo de pacientes por criterios de búsqueda y solamente será accesible para los roles Médico, Organización y Administrador. La data para este set de casos de uso se obtendrá de HAPI a la que el demostrador se conectará a través de una interfaz API, una vez obtenida la información de HAPI el aplicativo demostrador es el que realiza el filtro de información.

La funcionalidad de conteo de pacientes por criterios de búsqueda se implementará tomando estrictas medidas de anonimato de la información de los pacientes. La elección de estos criterios de búsqueda se realizó tomando como referencia el documento “Indicadores de Salud” publicado por la Organización Mundial de la Salud y la Organización Panamericana de la Salud de donde se extrajo lo siguiente [50]:

- Rango de Edad.
- Rango de Presión Sanguínea.

Adicionalmente, se diseñarán 3 criterios de búsqueda basados en la condición médica del paciente que permitirá al usuario implementar sus propios criterios de búsqueda.

Para garantizar la funcionalidad de este módulo se generaron los siguientes casos de uso:

- **Caso 1: Búsqueda de pacientes por rango de edad**

Se permitirá el conteo de pacientes según su rango de edad, este es uno de los criterios más usados debido a que las enfermedades se presentan con distinta frecuencia en los distintos grupos etarios.

- **Caso 2: Búsqueda por rango de presión sanguínea**

Se permitirá la búsqueda por rango de presión sanguínea debido a que está relacionada con problemas del corazón el cual hoy en día es la principal causa de muerte en el mundo [51].

- **Case 3: Búsqueda por condición médica del paciente**

Este criterio de búsqueda permite al usuario realizar un conteo de pacientes con alguna condición médica; en el aplicativo se implementarán 3 de estas opciones de



búsquedas para garantizar que el usuario pueda acceder al dato exacto de su búsqueda.

- **Caso 4: Criterios de búsqueda combinables**

Esta característica es muy importante para que el usuario pueda realizar búsquedas más exactas, el objetivo es combinar los criterios de búsqueda de los casos de uso descritos en los 3 primeros casos de uso.

- **Caso 5: Estadísticas de pacientes por rol**

Esta característica de la búsqueda y conteo de pacientes se realiza por rol, en dónde las organizaciones y los médicos solamente podrán aplicar esta funcionalidad a los pacientes que atienden. Adicionalmente se cuenta con el rol paciente, pero éste no tiene permisos de búsqueda como los mencionados anteriormente.

- **Caso 6: Búsqueda de pacientes con rol de administrador**

Adicionalmente a los roles de organización y médico se cuenta con el rol tipo administrador, el cual tiene permiso de acceso a la información de todos los pacientes que se encuentran en la base de datos de HAPI. Además esta funcionalidad de búsqueda como administrador fue extendida a los roles organización y médico a través de una casilla de selección.



## 5 Limitaciones del proyecto

Los objetivos del proyecto son analizar el estándar FHIR y el aplicativo HAPI para extraer las alternativas de seguridad que son viables de implementar en este último, y generar casos de uso para emplearlos como ejemplo en un demostrador; lo que nos permitirá validar la usabilidad de la herramienta HAPI FHIR en términos de seguridad y privacidad usando recursos enmarcados en el estándar HL7 FHIR; para ello se implementó un aplicativo (FrontEnd y BackEnd) que se conecta a HAPI a través de una interfaz API para lograr extraer la información requerida por los usuarios. Sin embargo, existen algunas limitaciones que es importante enmarcar en el proyecto para un mejor entendimiento del lector.

### 5.1 Seguridad y privacidad

La implementación completa de un sistema de autorización y autenticación no es parte del objetivo del proyecto; sin embargo, dentro del desarrollo de este proyecto hay ciertos elementos que soportan estos requerimientos.

FHIR no es un protocolo de seguridad, ni define ninguna funcionalidad de seguridad; sin embargo FHIR define protocolos de intercambio y modelos de contenido que deben de usarse con protocolos de seguridad definidos en otros estándares [26].

La información que se extrae de la base de datos HAPI FHIR a través del aplicativo en términos del acceso de un paciente en particular, solamente es permitida si el usuario de búsqueda tiene los permisos para poder acceder a ella, caso contrario es denegada la información.

En el caso de la obtención de cantidad de pacientes por criterios de búsqueda particulares, la información es registrada en memoria RAM y solamente es utilizada para realizar la cuenta de pacientes, por lo que se descarta el almacenamiento de información en mecanismos persistentes como base de datos; adicionalmente, debido a que el aplicativo ha sido desarrollado en java, este lenguaje de programación cuenta con un proceso que se ejecuta cada cierto periodo de tiempo y que libera la información que ya no está siendo accedida de la memoria RAM [52], de esta manera se garantiza el cumplimiento de la seguridad y privacidad de los datos clínicos.

## 5.2 Soporte Multi Lenguaje

La especificación FHIR soporta múltiples lenguajes en una variedad de conceptos, siendo el Inglés el lenguaje utilizado por defecto; todos los recursos tienen un elemento de lenguaje opcional.

En este proyecto, el lenguaje del prototipo es el Inglés, siendo posible implementar y soportar otros lenguajes; esta opción no está definida como parte del alcance de este proyecto.

## 5.3 Conformidad FHIR

La especificación central de FHIR describe un conjunto de recursos, marcos y API que se utilizan en muchos contextos diferentes en el cuidado de la salud. Sin embargo, existe una amplia variabilidad entre jurisdicciones y en todo el ecosistema de atención médica en torno a prácticas, requisitos, regulaciones, educación y qué acciones son factibles y/o beneficiosas. Por esta razón, la especificación FHIR es una "especificación de plataforma": crea una plataforma o base común sobre la cual se implementan una variedad de soluciones diferentes. Como consecuencia, esta especificación generalmente requiere una mayor adaptación a contextos de uso particulares [49].

En este proyecto solamente se han utilizado los tipo de recurso: Patient, Observation, Practitioner, Organization; no es del alcance de este proyecto la utilización de recursos adicionales proporcionados por el estándar FHIR.

## 5.4 Performance

En la sección 6 se muestra la arquitectura del aplicativo HAPI FHIR, así como del aplicativo demostrador generado para la demostración. El acceso a la información de HAPI es realizada a través de la interfaz API que expone; este procedimiento es costoso por lo que se ha tratado de minimizar su ocurrencia; además, la información obtenida para el conteo de pacientes por parámetros de búsqueda, se almacena en memoria RAM evitando la operación en base de datos.

Si bien es cierto, no es parte del alcance de este proyecto la optimización de las operaciones realizadas en los aplicativos, el código implementado en el aplicativo HAPI

FHIR y en el aplicativo demostrador se ha optimizado para reducir operaciones innecesarias.

## 5.5 Manejo de errores

El código implementado se ha diseñado para soportar los casos de uso definidos en la sección 4, en dónde los escenarios de error son conocidos y se ha definido su solución. El prototipo desarrollado no ha sido probado para casos de uso extendidos aparte de los definidos en este capítulo.

## 5.6 Independencia de plataforma

La base del código implementado en el aplicativo FHIR y en el aplicativo demostrador es Java; el segundo se implementó utilizando la arquitectura Modelo-Vista-Controlador (MVC), adicionalmente, se realizó utilizando el ambiente de desarrollo Apache Netbeans, con el cual se lograron los objetivos buscados a nivel de desarrollo; por otro lado, las vistas generadas en el Front End del aplicativo demostrador fueron probadas en navegadores como Chrome, Firefox y Microsoft Edge.

No se garantiza el funcionamiento del aplicativo Front End en navegadores adicionales a los mencionados.

## 6 Arquitectura e Implementación

Para la implementación de este proyecto se tomó como referencia el aplicativo HAPI FHIR JPA Server, su elección se debe a que este aplicativo servidor es el más completo de la familia HAPI que cuenta con la mayoría de las características de FHIR versión R4; este aplicativo cuenta con todas las operaciones estándar (leer/crear/eliminar entre otros); además cuenta con una instancia de base de datos integrada H2 Java lo que permite que el servidor pueda ejecutarse sin necesidad de conectarse a una base de datos externa; pero también se puede conectar a una base de datos externa si el implementador así lo necesitara; en este aplicativo se implementaron las reglas de control de acceso requeridas para este proyecto.

Adicionalmente, se implementó un aplicativo demostrador que permite la interacción con HAPI a través de una interfaz amigable para el usuario que permite a los usuarios realizar consultas de la información de los pacientes; además cuenta con un módulo que permite realizar un conteo de pacientes según los criterios de búsqueda definidos por el usuario y que toma como base las reglas embebidas en HAPI. Este aplicativo también cuenta con un Front End en donde el usuario tiene la posibilidad de realizar la búsqueda de la información de los pacientes a través del id del paciente, además permite ingresar los criterios de búsqueda para el conteo de pacientes; el resultado de ambas búsquedas será mostrado en el mismo Front End del aplicativo.

En adelante se mostrarán las arquitecturas definidas para cada uno de los aplicativos, en la que se profundizará en los detalles para un mayor entendimiento del lector; además se mostrará la implementación del aplicativo HAPI en el servidor y se mencionará cada una de las implementaciones realizadas en código.

### 6.1 Diagrama de Arquitectura HAPI

Uno de los elementos importantes para la elaboración de este proyecto es el diseño de la arquitectura en el aplicativo HAPI, para ello utilizaremos como fuente la información vertida en el apartado 3 que nos permite basar la arquitectura en función de los casos de uso propuestos.

En la figura 13 se presentan los componentes que integrarán la arquitectura, así como un esbozo simplificado de la arquitectura a utilizar; para la elaboración de ésta se tomaron en cuenta las características definidas en la sección 3 de este documento; en dónde y en modo

de resumen se definió las características que ofrece FHIR en términos de seguridad y su viabilidad de implementación; del análisis realizado al aplicativo HAPI se observó que una de las características más viables de implementar y que es compatible con FHIR es la de un módulo de control de acceso basado en roles (RBAC), para ello el elemento en HAPI que nos permite implementarlo es el “Authorization Interceptor” que en adelante se volverá el elemento principal con el cual se elaboró el código que permitió la generación del sistema de control de acceso requerido para este proyecto.

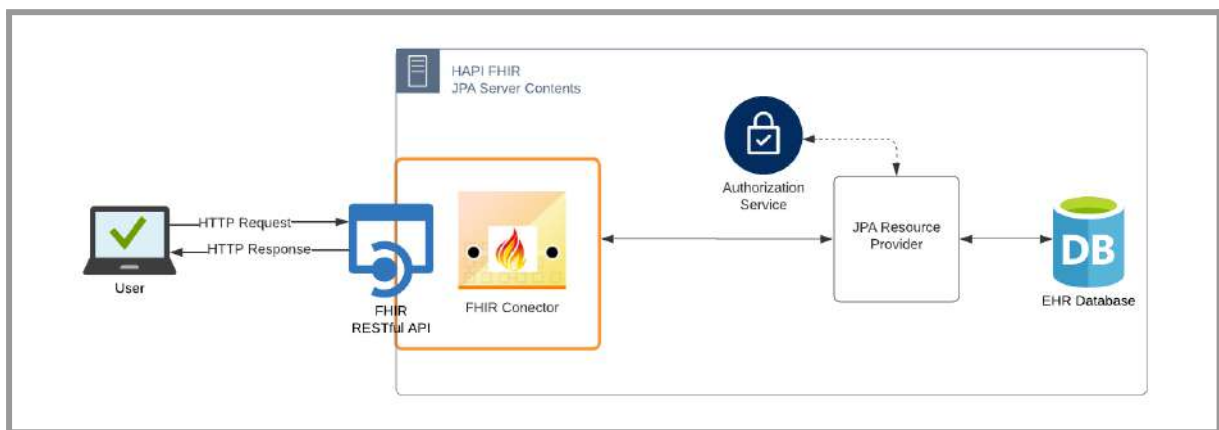


Figura 13: Arquitectura simplificada HAPI FHIR utilizada en el proyecto

En la arquitectura anterior se muestra cómo es el flujo de información en HAPI para lograr las reglas de control de acceso; esta inicia con la solicitud del usuario a través de alguna herramienta o un aplicativo (Caso de este proyecto).

El acceso a la información dentro de HAPI pasa por diferentes etapas, entre ellas tenemos el acceso a través del RESTful API que es la interfaz que expone el aplicativo HAPI para brindar acceso a la información que posee.

El conector FHIR envía la solicitud al proveedor de recursos de HAPI, este a su vez se vale del “Authorization Service”, que es el módulo dónde se ha codificado las reglas de control de acceso, este módulo es la parte esencial de la implementación de este proyecto dado que de acuerdo al rol de usuario que solicita el acceso a la información permite o deniega la solicitud; la respuesta del “Authorization Service” es interpretada por el proveedor de recursos de HAPI y según las reglas definidas se permitirá o denegará totalmente o parcialmente el acceso a la información requerida y almacenada en la base de datos.

El proveedor de recursos de HAPI extrae la información permitida de la base de datos y es enviada al conector FHIR para que a través de la API envíe la información al usuario, caso contrario se le enviará como respuesta el código 403 de acción no autorizada.

A continuació se mostra una breu descripció de lo que involucra cada mòdul de la arquitectura antes mencionada.

- **User:** Representa al usuari que requiera hacer la consulta, que a través de una interfaz solicita información al servidor FHIR; este usuario puede ser un profesional de la salud, paciente u otro usuario.
- **FHIR Conector:** El conector de FHIR expone una API RESTful la cual se puede invocar a través de HTTP/HTTPS; el conector FHIR convierte cada solicitud de entrada en estándar HL7 y lo envía a los recursos disponibles del servidor FHIR.
- **Authorization Service:** El servicio de autorización analiza cada solicitud incluido los detalles de la solicitud y los datos que se devuelven, el resultado de este análisis determina si se permite la operación o se bloquea. Para saber qué decisión tomar, el servicio de autorización utiliza Roles y Permisos configurados para cada usuario de la aplicación; el servicio de autorización realiza una determinación binaria simple; si la operación es permitida el usuario podrá ejecutar la operación requerida, caso contrario el cliente verá una respuesta HTTP 403 no autorizada; en la figura 14 se muestra el funcionamiento del “Authorization service”.

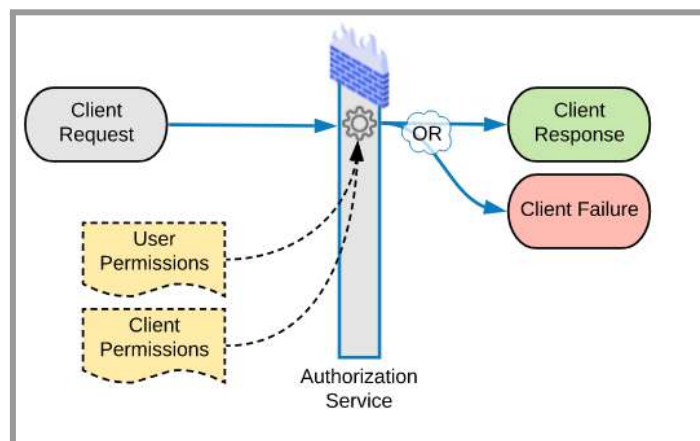


Figura 14: Actuación del Authorization Service [43]

- **JPA Resource Provider:** El proveedor de recursos implementa un método @search que utiliza el conjunto completo de parámetros de búsqueda definidos en la especificación FHIR para el tipo de recurso dado; los proveedores de recursos también amplían una super clase que implementa todos los demás métodos FHIR, como Leer, Crear, Eliminar, entre otros.

- EHR Database:** Esta base de datos se carga regularmente con un conjunto de datos de prueba y también este servidor puede almacenar cualquier dato relacionado con conceptos administrativos como pacientes, proveedores, organizaciones y dispositivos, así como una variedad de conceptos clínicos que incluyen problemas, medicamentos, diagnósticos, planes de cuidados, temas económicos, entre otros.

### 6.1.1 Authorization Interceptor

El authorization interceptor es una de las particularidades de seguridad que ofrece el aplicativo HAPI FHIR, mediante el cual el aplicativo por sí mismo es capaz de permitir o restringir el acceso a la información, basado en las necesidades de seguridad requeridas para la implementación, este es el módulo que se utilizará para codificar las reglas de control de acceso requeridas para el proyecto.

Para entender a profundidad su funcionamiento, en las siguientes líneas se procederá a explicar las características de dicho funcionamiento.

- “Authorization” operaciones de lectura:** Al autorizar una operación de lectura, “Authorization Interceptor” siempre permite que el código del cliente se ejecute y genere una respuesta. Luego examina la respuesta antes de devolvérsela al cliente, y si las reglas no permiten que se muestre los datos al cliente, el interceptor aborta la solicitud. En la figura 15 se muestra el diagrama UML de una operación de lectura usando el Authorization Interceptor.

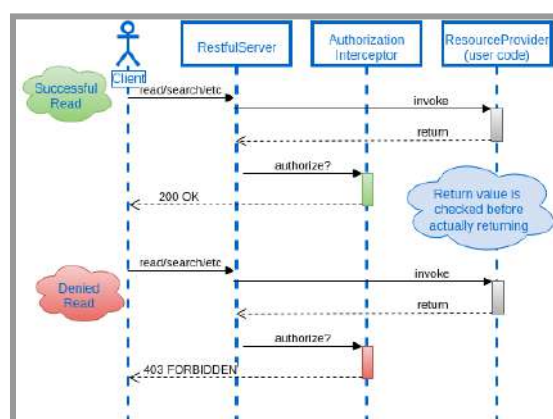


Figura 15: Authorization en operaciones de Lectura [41]

- **“Authorization” operaciones de escritura:** Las operaciones de escritura (crear, actualizar, etc.) generalmente las autoriza el interceptor al examinar la URL analizada y tomar una decisión sobre si autorizar la operación antes de permitir que continúe el código del proveedor de recursos. Esto significa que el código del cliente no tendrá la oportunidad de ejecutar y crear recursos para los que el cliente no tiene permisos de creación. En la figura 16 se muestra el diagrama UML de una operación de escritura usando el Authorization Interceptor.

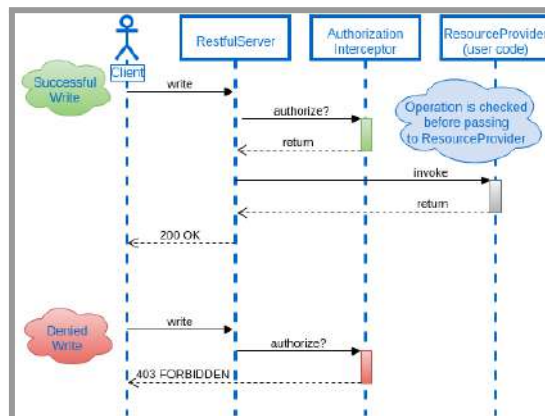


Figura 16: Authorization en operaciones de escritura [41]

## 6.2 Arquitectura del Aplicativo demostrador

El diseño del aplicativo demostrador es una de las implementaciones más importantes de este proyecto, debido a que es el aplicativo de interfaz que permite la interacción con el usuario y el acceso a las funcionalidades definidas para cada rol.

El ingreso al aplicativo se realizará a través de la validación de las credenciales usuario y contraseña que se ingrese con las credenciales almacenadas en base de datos; si la validación es correcta el acceso al aplicativo será exitoso.

Para la identificación del rol, este aplicativo se comunicará con el aplicativo HAPI a través de la interfaz RESTful API para identificar al usuario y extraer el tipo de usuario que intenta acceder; el campo llave que permitirá identificar al usuario en ambos aplicativos es el id FHIR.

Una vez el usuario tenga acceso al Front End del aplicativo, podrá navegar entre las opciones de búsqueda que se presentan entre ellas: la búsqueda de información por id de paciente y el conteo de pacientes basado en criterios de búsqueda.

En la figura 17, se muestra una arquitectura simplificada de este aplicativo y las interacciones entre los diferentes elementos que la componen.



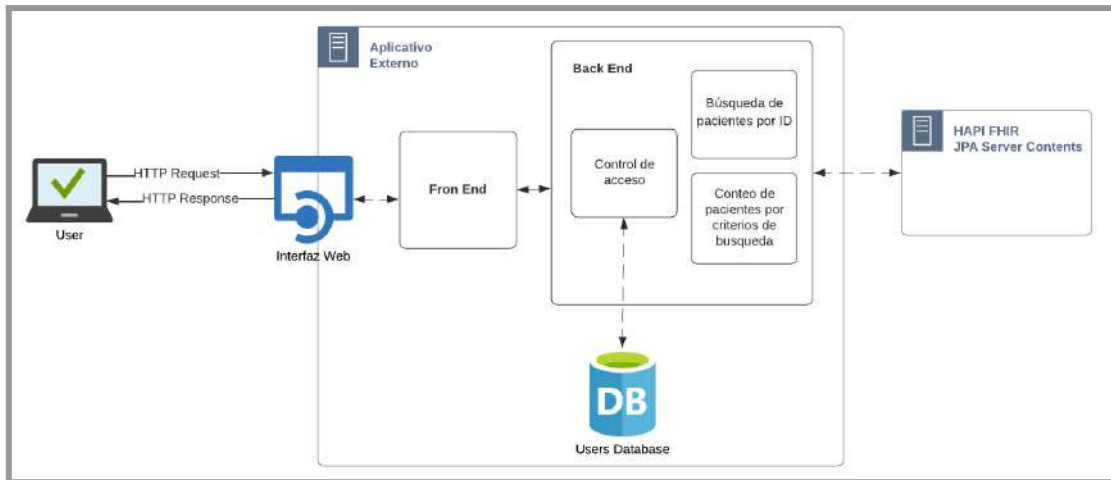


Figura 17: Arquitectura simplificada aplicativo demostrador

A continuación se brindará una breve descripción de la funcionalidad de cada módulo.

- Front End:** Es el módulo que contiene la interfaz de usuario del aplicativo demostrador. Para el acceso a la interfaz de este aplicativo inicialmente aparecerá un formato para introducir el usuario y la contraseña de la persona que requiere el acceso; la validación del acceso se hace realizando la comparación del usuario y contraseña registrados en la base de datos; y para la identificación del tipo de rol del usuario se toma el id FHIR de la base de datos del aplicativo demostrador y se realiza la búsqueda en HAPI con el fin de extraer el tipo de recurso(rol) que corresponde.

Una vez logrado el acceso a través de las credenciales del usuario, el sistema dará paso a otra pantalla en dónde según el rol del usuario podrá acceder a las siguientes funcionalidades:

- Paciente:** Solamente se le permitirá el acceso a su información de paciente, no tendrá acceso a la funcionalidad de conteo de pacientes usando criterios de búsqueda.
- Médico, Organización y Administrador:** Se le permite el acceso a la información de los pacientes utilizando el id de cada uno de ellos; además tendrá el acceso a la funcionalidad para hacer conteo de pacientes según criterios de búsqueda. En el caso del rol de médico u organización este debe ser el médico u organización del paciente para que pueda acceder a la información de este.

- **Control de acceso:** Este módulo permite validar las credenciales de acceso del usuario, para ello se realizará una comparación de las credenciales ingresadas con las credenciales definidas en la base de datos. Adicionalmente, para la definición del rol que le corresponde al usuario se realiza un cruce de información con la información del usuario en la base de datos de HAPI a través del id FHIR.
- **Búsqueda de pacientes por ID:** Este es el módulo que permite realizar la búsqueda de la información del paciente a través del id generado en FHIR. Lo que hace este módulo es realizar la búsqueda del id ingresado por el usuario para realizar la búsqueda en el aplicativo HAPI; el resultado será el acceso total, parcial o la denegación de la consulta y se mostrará en el Front End.
- **Conteo de pacientes por criterios de búsqueda:** Mediante este módulo se le permitirá a los usuarios autorizados realizar el cálculo de la cantidad de pacientes con criterios de búsqueda predefinidos; lo que hace este módulo es realizar una búsqueda de pacientes tomando en consideración los criterios de búsqueda predefinidos por el usuario y considerando el universo al cual pertenecen (Mismo médico, organización) o una búsqueda total si el usuario tiene perfil de administrador. El resultado será mostrado en el Front End del aplicativo.

## 6.3 Implementación y Desarrollo

Como ya se describió en la sección 3.2, se usará el aplicativo HAPI FHIR como base para la implementación de los casos de uso que nos permitirán alcanzar el objetivo del proyecto; el módulo y la versión del aplicativo será el “HAPI FHIR JPA Server Starter” en su versión 5.4.0.

A continuación se mostrará el proceso que se siguió hasta llegar a la versión más estable del aplicativo que nos permitirá la simulación de los casos de uso descritos en apartados anteriores.

Además se describe el trabajo realizado para la implementación de la lógica de control de acceso en el aplicativo HAPI, así como el desarrollo y la implementación del aplicativo demostrador; ambas implementaciones se realizaron considerando el criterio de arquitectura definido anteriormente.

### 6.3.1 Intentos de Implementación HAPI

Existen variadas herramientas que nos permiten gestionar información médica, las cuales fueron presentadas en el apartado 2.3; en este sentido la elección para este proyecto fue por el aplicativo HAPI FHIR, el cual es una herramienta que se ajusta a las necesidades requeridas para el proyecto además de ser uno de los aplicativos más confiables y estables de la industria.

En este apartado se mencionan los intentos de implementación realizados hasta llegar con la versión más estable de HAPI.

- **HAPI FHIR 5.6.0 (Raccoon):** Es la última versión estable de HAPI FHIR, la cual cuenta con todos los recursos ofrecidos por HAPI FHIR. Este release cuenta con mejoras realizadas a las versiones anteriores en los siguientes aspectos [44]:
  - Seguridad.
  - Cambios generales en el módulo Servidor, Cliente y “Parser”.
  - Herramienta CLI.
  - Cambios en el servidor JPA.
  - Cambios en la partición del servidor JPA.
  - Validación de cambios y terminología del servidor.
  - Mejoras en el módulo MDM del servidor JPA.

Al momento de instalarlo se presentó el error de la figura 18.



```
[INFO] HAPI FHIR - Distribution Archive ..... SKIPPED
[INFO] HAPI FHIR JAX-RS Server Kotlin Test ..... SKIPPED
[INFO] HAPI FHIR - Minimal Dependency Test - Client ..... SKIPPED
[INFO] HAPI FHIR - Minimal Dependency Test - Server ..... SKIPPED
[INFO] hapi-fhir-spring-boot-autoconfigure ..... SKIPPED
[INFO] hapi-fhir-spring-boot-starter ..... SKIPPED
[INFO] hapi-fhir-spring-boot ..... SKIPPED
[INFO] hapi-fhir-spring-boot-samples ..... SKIPPED
[INFO] hapi-fhir-spring-boot-sample-client-apache ..... SKIPPED
[INFO] hapi-fhir-spring-boot-sample-client-akhttp ..... SKIPPED
[INFO] hapi-fhir-spring-boot-sample-server-jersey ..... SKIPPED
[INFO] ..... SKIPPED
[INFO] BUILD FAILURE
[INFO] -----
[INFO] Total time: 01:33 min
[INFO] Finished at: 2022-02-16T05:07:29+01:00
[INFO] -----
[ERROR] Plugin co.uhn.hapi.fhir:hapi-tinder-plugin:6.0.0-PR2-SNAPSHOT or one of its dependencies could not be resolved: Could not find artifact co.uhn.hapi.fhir:hapi-tinder-plugin:jar:6.0.0-PR2-SNAPSHOT -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] https://wiki.apache.org/confluence/display/MAVEN/PluginResolutionException
root@benny-thinkpad-x1:/home/Benny/Downloads/hapi-fhir-master
```

Figura 18: Resultado de la ejecución de HAPI FHIR 5.6.0

Se investigó en los foros del desarrollador, sin llegar a la solución; por lo que se procedió a desestimar esta versión del release.

- **hapi-fhir-jpaserver-starter 5.7.0:** Es la última versión estable del módulo JPA Server Starter, según especificaciones de la página web HAPI FHIR, este es el módulo más completo en el que se podría implementar reglas de acceso y seguridad para el acceso a datos clínicos[45].

Al momento de ejecutar el aplicativo, la instalación fue exitosa, como se muestra en la figura 19.

```

root@benny-thinkpad-x1:/home/Benny/Programas/Programas_FHIR/hapi-fhir-jpaserver-starter-master# mvn jetty:run
[INFO] Scanning for projects...
[WARNING] The project ca.uhn.fhir:hapi-fhir-jpaserver-starter:war:5.7.0-PRE9-SNAPSHOT uses prerequisites which is only intended for maven-plugin projects but not for non maven-plugin projects. For such purposes you should use the maven-enforcer-plugin. See https://maven.apache.org/enforcer/enforcer-rules/requireMavenVersion.html
[INFO]
[INFO] ----- ca.uhn.fhir:hapi-fhir-jpaserver-starter > -----
[INFO] Building HAPI FHIR JPA Server - Starter Project 5.7.0-PRE9-SNAPSHOT
[INFO] -----[ war ]-----
[INFO]
[INFO] >>> jetty-maven-plugin:9.4.44.v20210927:run (default-cli) > test-compile @ hapi-fhir-jpaserver-starter >>>
[INFO]
[INFO] --- maven-enforcer-plugin:3.0.0:enforce (enforce-maven) @ hapi-fhir-jpaserver-starter ---
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.7:prepare-agent (default-prepare-agent) @ hapi-fhir-jpaserver-starter ---
[INFO] argLine set to "-javaagent:/root/.m2/repository/org/jacoco/org.jacoco.agent/0.8.7/org.jacoco.agent-0.8.7-runtime.jar=destfile=/home/Benny/Programas/Programas_FHIR/hapi-fhir-jpaserver-starter-master/target/jacoco.exec,excludes=ca/uhn/fhir/model/dstu2/**/*.*.class:ca/uhn/fhir/jpa/rp/r5/*.*.class:ca/uhn/fhir/jpa/rp/r4/*.*.class:ca/uhn/fhir/jpa/rp/dstu3/*.*.class:ca/uhn/fhir/jpa/rp/dstu2/*.*.class,dumponexit=true"
[INFO]
[INFO] --- maven-resources-plugin:3.0:resources (default-resources) @ hapi-fhir-jpaserver-starter ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 3 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ hapi-fhir-jpaserver-starter ---
[INFO] Nothing to compile - all classes are up to date

```

Figura 19: Resultado de la ejecución de hapi-fhir-jpaserver-starter 5.7.0

Sin embargo, al momento de acceder a la interfaz que genera el aplicativo a través de un navegador de internet, nos muestra el error de la figura 20.



Figura 20: Resultado de la prueba del aplicativo a través de un navegador web

Se buscó una solución en foros del desarrollador y en el internet en general, sin éxito; por lo que se procede a desestimar esta versión.

### 6.3.2 Implementación estable

Debido a los intentos infructuosos anteriores y con la recomendación del fabricante de utilizar JPA server en su versión “Starter”; se procedió a probar versiones anteriores del aplicativo, en la que llegamos a la versión 5.2.

En la figura 21 y 22 se muestra el resultado de la instalación satisfactoria de esta versión del release.

```
[INFO] Started o.e.j.m.p.JettyWebAppContext@421ead7e[, [file:///home/Benny/Programas/Programas%20FHIR/hapi-fhir-hir-jpaserver-starter-5.2.0/target/jetty_overlays/hapi-fhir-testpage-overlay-5_2_0_war/, jar:file:///root/.m2/repository/org/webjars/bower/awesome-bootstrap-checks/4.17.43.jar!/META-INF/resources, jar:file:///root/.m2/repository/org/webjars/bower/awesome-bootstrap-checks/4.17.43.jar!/META-INF/resources, jar:file:///root/.m2/repository/org/webjars/font-awesome/5.8.2/font-awesome-5.8.2.jar!/META-INF/resources, jar:file:///root/.m2/repository/org/webjars/jstimezonedetect/1.0.6/jstimezonedetect-1.0.6.jar!/META-INF/resources, jar:file:///root/.m2/repository/org/webjars/jquery/1.11.1/jquery-1.11.1.jar!/META-INF/resources, jar:file:///root/.m2/repository/org/webjars/bootstrap/3.3.7/bootstrap-3.3.7.jar!/META-INF/resources, jar:file:///root/.m2/repository/org/webjars/momentjs/2.10.3/momentjs-2.10.3.jar!/META-INF/resources, jar:file:///home/Benny/Programas/Programas%20FHIR/hapi-fhir-jpaserver-starter-5.2.0/src/main/webapp/]
[INFO] Started ServerConnector@485262be{HTTP/1.1, (http/1.1)}{0.0.0.0:8080}
[INFO] Started @27907ms
[INFO] Started Jetty Server
```

Figura 21: Inicio servidor estable HAPI FHIR 5.2

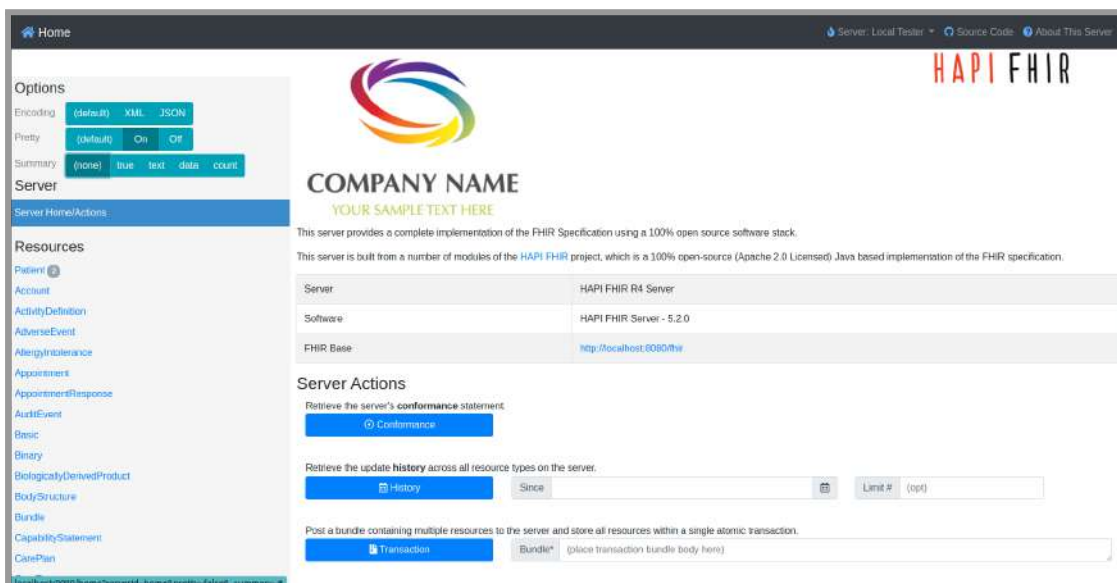


Figura 22: Interfaz HAPI FHIR estable HAPI FHIR 5.2

### 6.3.3 Lógica control de acceso HAPI FHIR

Con el fin lograr el cumplimiento de los casos de uso mostrados en la sección 4 de este documento, se procedió a realizar modificaciones en el aplicativo HAPI FHIR; esto debido a que al ser el aplicativo fuente brinda mayor versatilidad al momento de permitir los accesos de lectura a la base de datos de los pacientes, el cual cuenta con atributos que permiten interrelacionar a los pacientes con sus médicos y/o centros de salud (Organización).

En esta implementación se utilizó el módulo “Authorization Interceptor” de HAPI, para poder utilizar esta característica del aplicativo se realizó un desarrollo en el mismo HAPI que permite generar la reglas de control de acceso que requiere el proyecto.

En la figura 23 se muestra el diagrama de flujo utilizado para dicha implementación en donde el inicio es la solicitud de lectura de la información de un usuario, el interceptor



analiza la solicitud validando el usuario que realiza la consulta e identificandolo con su rol de usuario.

- **Paciente:** Permite el acceso a la información de sí mismo.
- **Médico:** Permite el acceso a la información de sus pacientes.
- **Organización:** Permite el acceso parcial a la información de cada paciente, no se permite acceso hacia la condición médica del paciente.
- **Administrador:** Permite acceso a la información de cualquier paciente, sin restricciones.



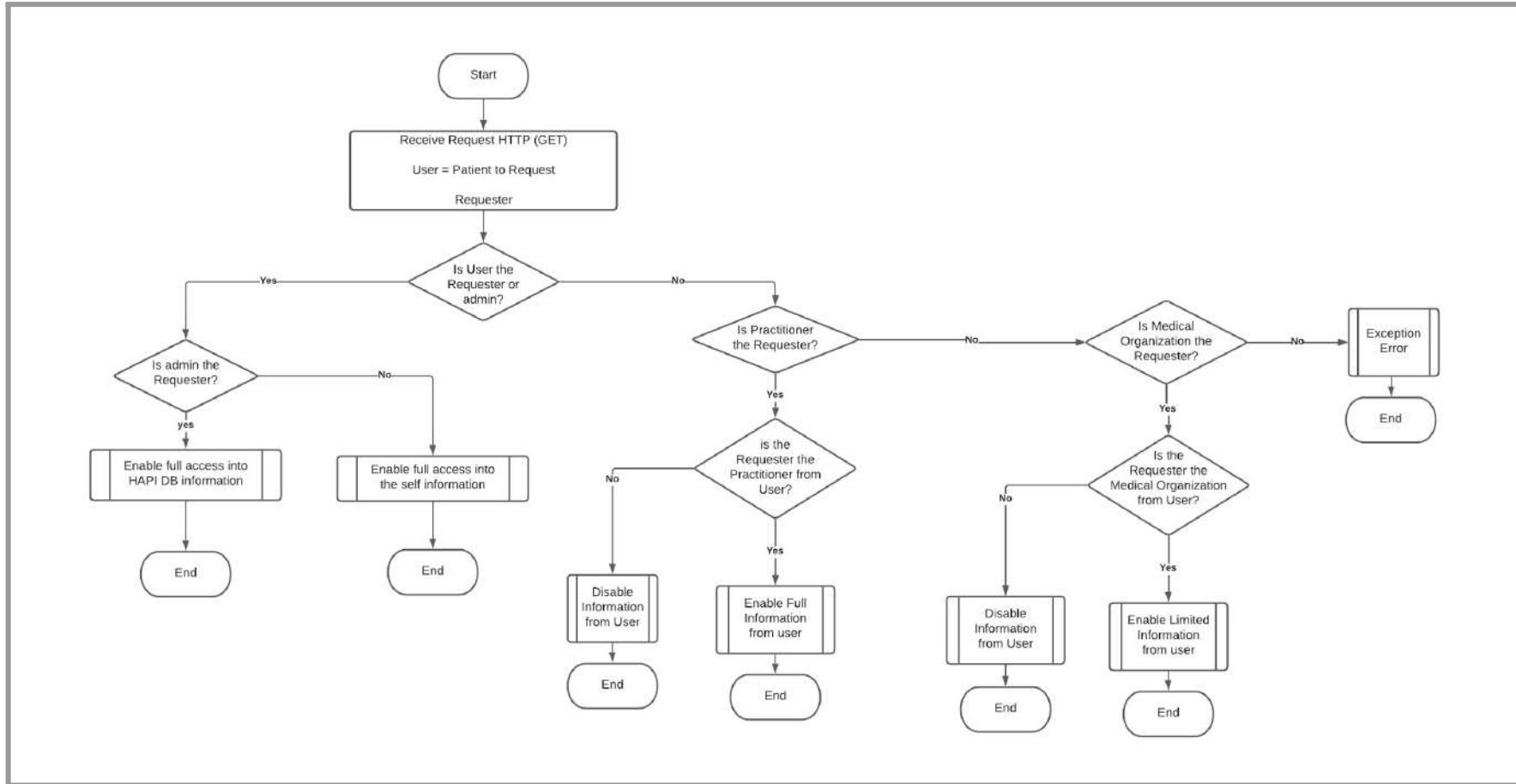


Figura 23: Diagrama de Flujo de la lógica implementada en HAPI FHIR

Adicionalmente en la figura 24 se muestra el listado de componentes que se han modificado en HAPI (Recuadro verde) para poder implementar el flujo mencionado en la figura 23.

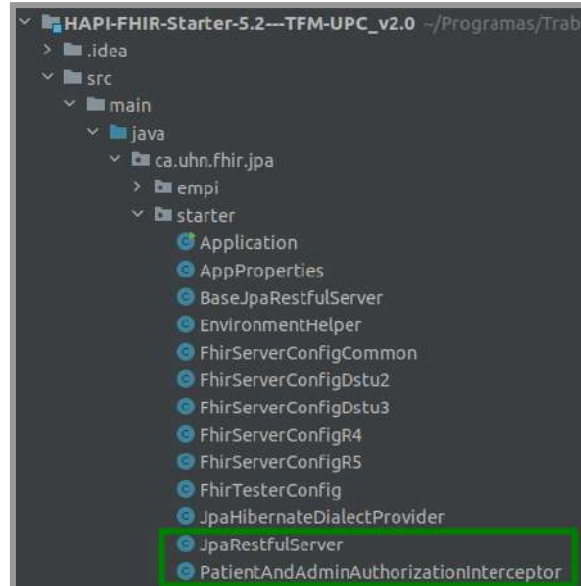


Figura 24: Listado de componentes modificado en HAPI

### 6.3.4 Desarrollo del aplicativo Demostrador

El desarrollo del aplicativo Demostrador es una de las partes esenciales del proyecto dado que permitirá generar una interfaz amigable con el usuario para su acceso a la herramienta, además contará con las funcionalidades de Back End “Control de Acceso”, “Información de Pacientes” y “Conteo de Pacientes”.

El desarrollo de este aplicativo se realizó siguiendo el esquema “Model-View-Controller” (MVC) y el desarrollo del programa se realizó en base a páginas “.jsp” y servlets, y el lenguaje de programación utilizado es Java.

En la Figura 25 se muestra el diagrama de la estructura implementada en Java, el cual responde al cumplimiento de las casuísticas mencionadas en el apartado 4, y en la figura 26 se muestra un pantallazo de la implementación MVC.



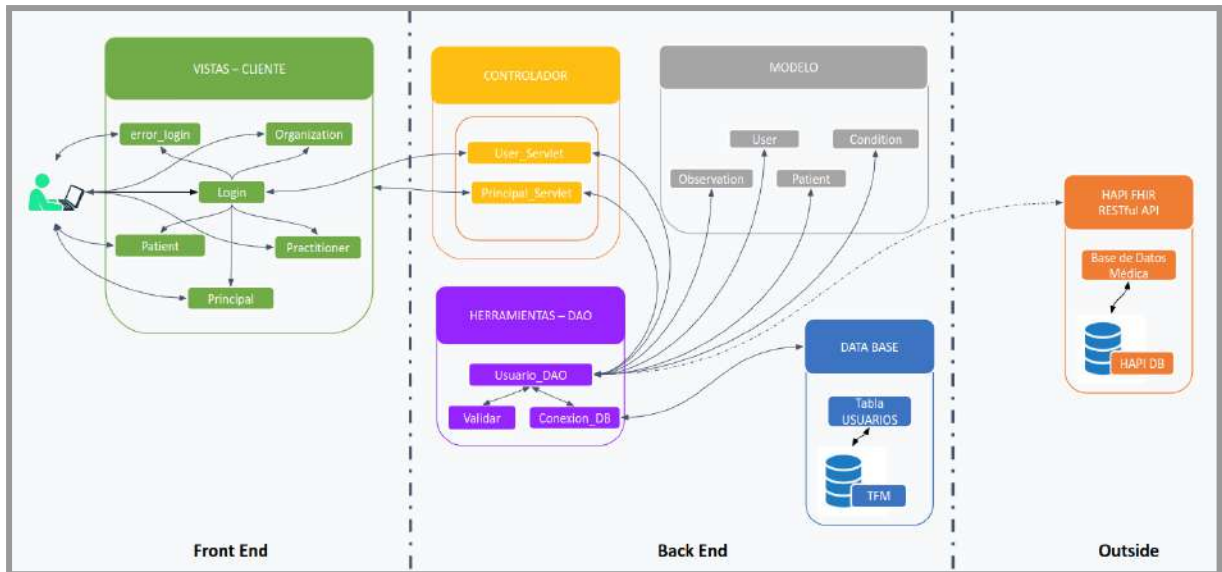


Figura 25: Diagrama simplificado del Front End y Back End

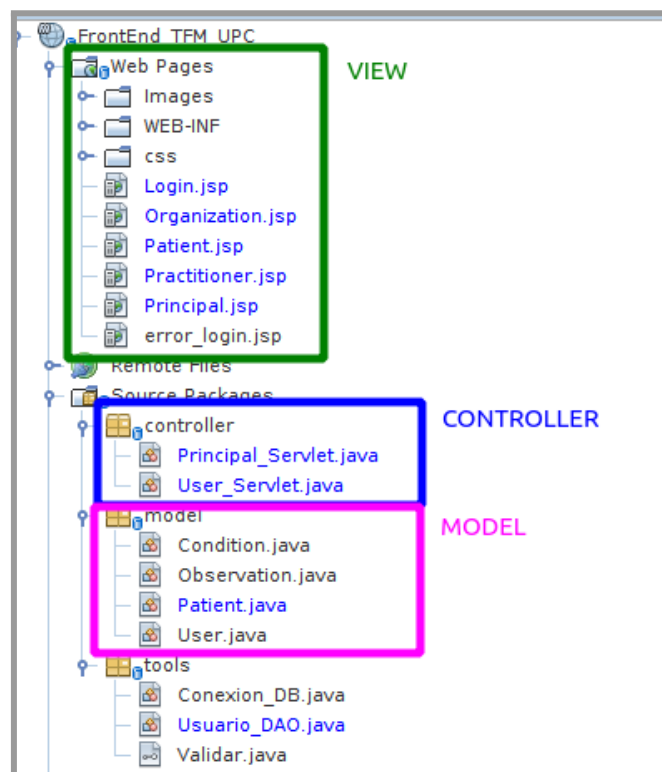


Figura 26: Modelo MVC del aplicativo demostrador

## ● Desarrollo del Back End

Una parte importante de este proyecto es la implementación del “Back End”. En este apartado se implementan llamadas a la API que ofrece el aplicativo HAPI FHIR; el Back End del aplicativo está basado en el lenguaje de programación java y presenta

las funciones que permitirán el acceso a la información para los casos de uso del apartado 4.

Las principales funcionalidades que se realizaron en este aplicativo son las descritas en el apartado 6.2, las cuales solamente se van a enumerar ya que su funcionamiento fue descrito anteriormente.

- Control de Acceso.
- Búsqueda de pacientes por ID.
- Conteo de pacientes por criterios de búsqueda.

En la figura 27 se muestra la lista de elementos implementados en la sección de Back End.

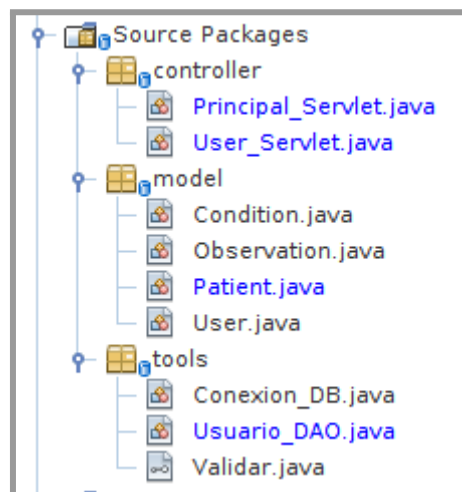


Figura 27: Estructura Back End

A continuación se mostrará una breve descripción de cada uno de los módulos que conforman el Back End.

- **User\_Servlet:** Este servlet tiene por funcionalidad servir de interfaz entre la página de acceso y la herramienta Usuario\_DAO la cual contiene el método de validación de las credenciales de usuario; además permitirá brindar el acceso a las páginas que contienen las funcionalidades que se requieren por cada rol.
- **Principal\_Servlet:** Es un servlet que nos permite acceder a las funcionalidades de “Búsqueda de Pacientes por ID” y “Conteo de Pacientes por Criterio de Búsqueda”, interactúa con la clase usuarioDAO que contiene los métodos que permiten acceder a la información de FHIR además de



contar con los métodos que permiten el conteo de pacientes usando los criterios de búsqueda definidos en el Front End.

- **Condition:** Es la clase que permite la funcionalidad de poder utilizar los criterios de búsqueda de pacientes definidos en el Front End; en este apartado se definen la cantidad de criterios que el usuario ha definido para el conteo de pacientes y permite al usuario utilizar todos los criterios o parte de estos.
- **Observation:** Es la clase dónde se encuentra el modelo del tipo de recurso “Observation” y está alineado con la estructura de parámetros del Json del tipo de recurso “Observation” en HAPI, permite recolectar la información recibida a través de Json de HAPI y almacena la información en memoria siguiendo la misma estructura recibida.
- **Patient:** Es la clase dónde se encuentra el modelo del tipo de recurso “Patient”, permite recolectar la información recibida por el Json del tipo de recurso “Patient” que viene desde HAPI.
- **User:** Es la clase que almacena la información del usuario recopilada desde la base de datos, es utilizada para realizar la validación de las credenciales que permitirán el acceso de los usuario a la herramienta.
- **Usuario\_DAO:** Es la clase en java que contiene los métodos de acceso a la base de datos para el control de acceso y los de conexión con la API de HAPI que permitirán la recolección de los pacientes a través de mensajes Json. Además, contiene los métodos que permitirá definir si es que la información del paciente es accesible por el usuario, así como el conteo de pacientes basado en criterios de búsqueda.
- **Conexion\_DB:** Es la clase donde se define el método de conexión a la base de datos del aplicativo demostrador; es reutilizada por la clase Usuario\_DAO para recolectar la información de las credenciales del usuario en base de datos.
- **Validar:** Es un recurso tipo interfaz que es utilizada en conjunto con Conexion\_DB para lograr el acceso a base de datos.

## ● Desarrollo del Front End

Parte esencial de este proyecto es la implementación de un módulo que permita visualizar el resultado del procesamiento realizado por el Back End; la implementación de este módulo se realizó en base a páginas “.jsp” y “servlets”, esto debido a que es la tecnología de la cual se tiene mayor grado de conocimiento; de hecho, la lógica implementada puede ser desarrollada en cualquier lenguaje de programación orientada a Front End; en la Figura 28 se muestra la estructura de páginas utilizada.

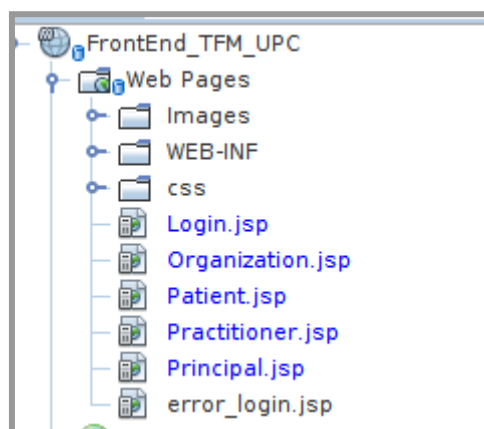


Figura 28: Estructura de páginas Front End

A continuación se describirán cada una de las páginas implementadas.

- **Login:** En esta página se encuentra la interfaz de usuario que permite ingresar las credenciales de acceso.
- **Organization:** Es la página a la que es redirigido el usuario y que cuenta con el acceso a la funcionalidad brindado al rol “Organización”.
- **Patient:** Es la página a la que es redirigido el usuario y que cuenta con el acceso a la funcionalidad brindado al rol “Paciente”.
- **Practitioner:** Es la página a la que es redirigido el usuario y que cuenta con el acceso a la funcionalidad brindado al rol “Practitioner”.
- **Principal:** Es la página a la que es redirigido el usuario y que cuenta con el acceso a la funcionalidad brindado al rol “administrador”.

- **Error\_login:** Es la página a la cual se redirecciona cuando se da un intento fallido de identificación en la página de Login.

La implementación realizada cuenta con los siguientes funcionalidades:

- Control de acceso mediante usuario y password.
- Acceso a una página distinta por tipo de rol (Paciente, Médico, Organización, Administrador).
- Formato de búsqueda de pacientes utilizando el ID.
- Formato de respuesta de búsqueda de pacientes por ID.
- Formato de ingreso de criterios para el conteo de pacientes.
- Formato de respuesta de conteo de pacientes por criterio de búsqueda.

En la figura 29 se muestra la pantalla de acceso y la interfaz de usuario.

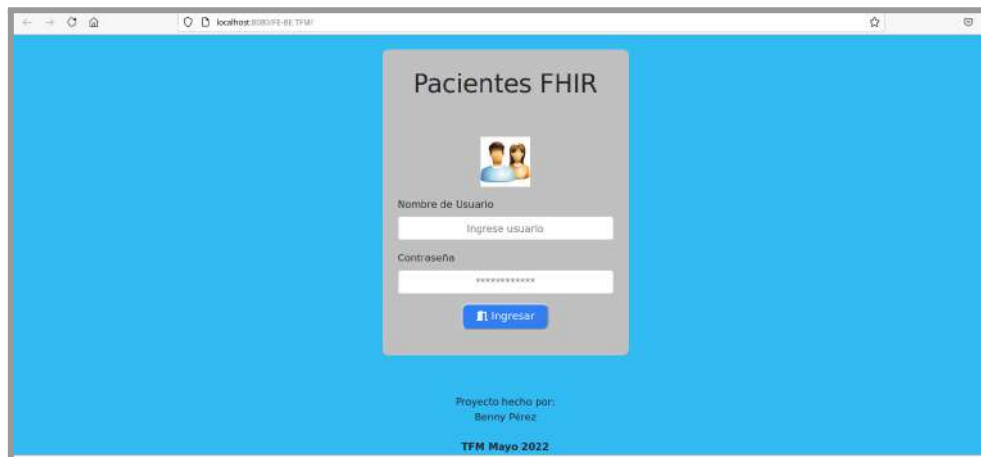


Figura 29: Interfaz de acceso de usuario

Adicionalmente, para cada rol que acceda al aplicativo se mostrará las características con las que podrá interactuar; en las figuras 30, 31, 32 y 33 se muestran el tipo de interfaz que se ofrecerá a cada rol.

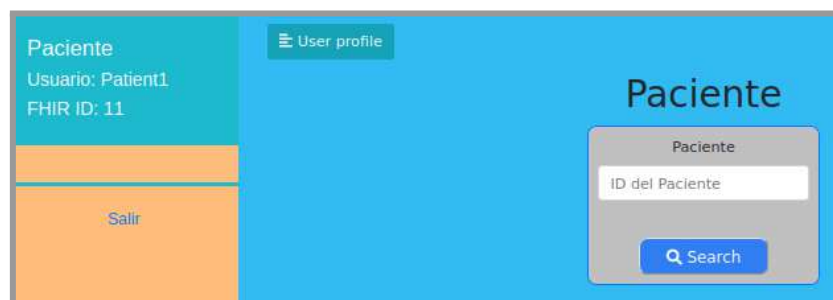


Figura 30: Interfaz de usuario Paciente

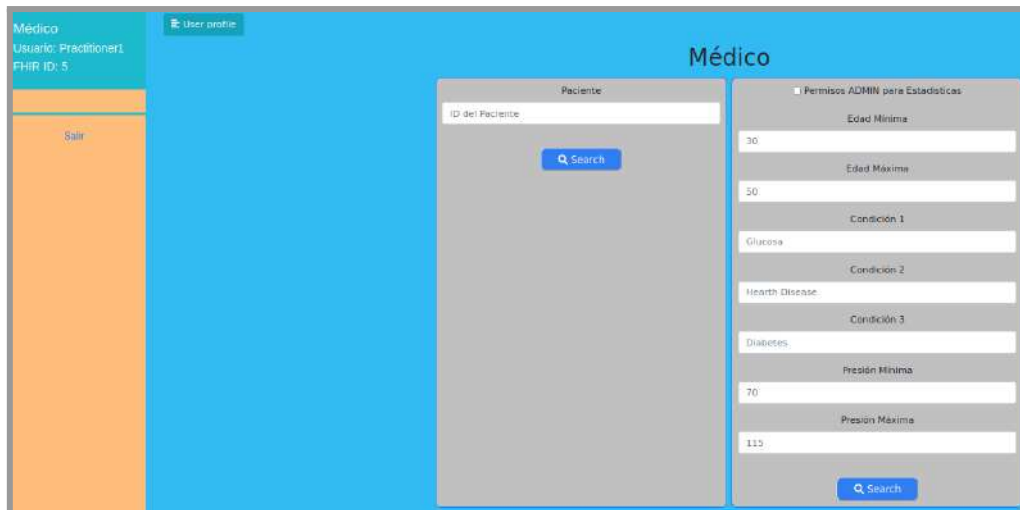


Figura 31: Interfaz de usuario Médico

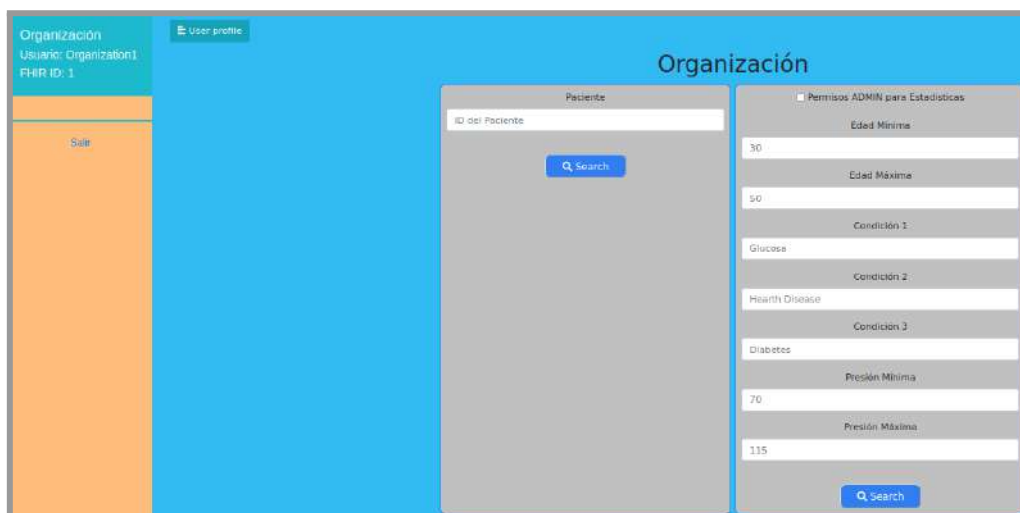


Figura 32: Interfaz de usuario Organización

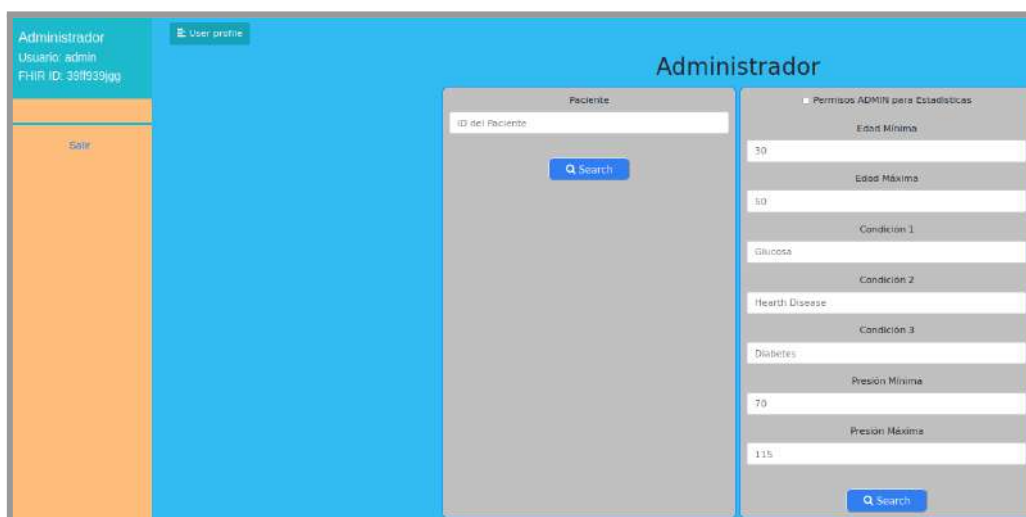


Figura 33: Interfaz de usuario Administrador

### 6.3.5 Generación de script de datos de prueba FHIR

Debido a que el aplicativo base de HAPI FHIR viene sin datos de pacientes, se procede a generar el script que permitirá añadir información de pacientes sintéticos a la base de datos de HAPI, el cual posteriormente podremos utilizar en los casos de uso mencionados en el apartado 4. Los tipos de recurso generados son los siguientes: “patient”(Paciente), “practitioner”(Médico), “organization”(Clínica u hospital) y “observation”(Observación clínica). En la figura 34 se muestra la estructura del script implementado.

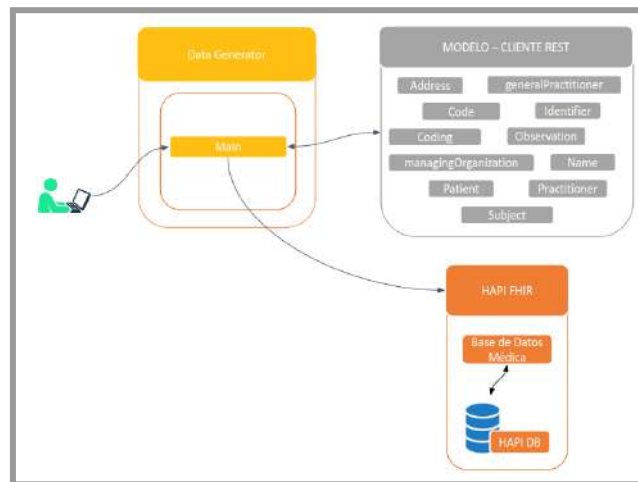


Figura 34: Arquitectura simplificada de script HAPI FHIR

En la figura 35 se muestra el resultado de la ejecución del script en donde se muestra que se han generado 500 pacientes HAPI.

```

{
  "resourceType": "Bundle",
  "id": "d22eb757-c886-46c5-a9df-027551c860c3",
  "meta": {
    "lastUpdated": "2022-06-10T08:13:29.958-05:00"
  },
  "type": "searchset",
  "total": 500,
  "link": [
    {
      "relation": "self",
      "url": "http://localhost:8080/fhir/Patient/"
    }
  ],
  "entry": [
    {
      "fullUrl": "http://localhost:8080/fhir/Patient/11",
      "resource": {
        "resourceType": "Patient",
        "id": "11",
        "meta": {
          "versionId": "1",
          "lastUpdated": "2022-06-10T08:06:35.255-05:00",
          "source": "#1UFBXbAojMgBa3pN"
        },
        "text": {
          "status": "generated",
          "div": "<div xmlns='http://www.w3.org/1999/xhtml'><div class=
class='hapiPropertyTable'><tbody><tr><td>Address</td><td>
1981</td></tr></tbody></table></div>"
        },
        "name": [

```

Figura 35: Se muestra el resultado de la ejecución del Script

## 7 Experimentos y resultados

En este apartado se hará una breve reseña de los casos de uso descritos en la sección 4, estos casos de uso nos permitirán comprobar la funcionalidad del aplicativo, además de validar a detalle pruebas y resultados obtenidos como producto de la ejecución de los mismos.

### 7.1 Resultados por Casos de Uso

Los casos de uso se encuentran separados en dos instancias; la primera corresponde a los casos de uso de la funcionalidad de búsqueda de información de pacientes por ID de paciente, y la segunda corresponde los casos de uso de la funcionalidad de conteo de pacientes en base a criterios de búsqueda, ambos definidos como parte del segundo objetivo del proyecto.

#### 7.1.1 Casos de uso búsqueda de pacientes por ID

Se mostrará la información de los pacientes, en donde usuarios como la Organización, el Médico o el mismo paciente pueden hacer la búsqueda de información de los pacientes; para estas casuísticas se han aplicado las restricciones propuestas en el apartado 4.1.

- **Lectura de información de un paciente por el mismo paciente**

Se muestra que cuando el paciente hace búsqueda de su información se le permite ver todos sus datos sin restricciones, en la figura 36 se muestra el resultado de la búsqueda.



Nombre	Identificador	Género	Fecha de Nacimiento	Médico	Clínica	Problema de Salud
Laura Berrocal	11	female	2007-04-25	7	2	Glucosa [Moles/Volumen] in Blood + Hearth Disease + High Pressure(Blood Pressure 125 )

Figura 36: Búsqueda de información por el mismo paciente



- **Lectura de la información de un paciente por otro paciente**

Se muestra que el usuario que realiza la búsqueda es un tipo paciente, y lo que está buscando es la información de otro paciente; en este sentido debido a las definiciones se procede a negar el acceso a la información del paciente buscado; como se muestra en la figura 37.



The screenshot shows a user profile for 'Paciente' (User: Patient1, FHIR ID: 11). The main heading is 'Paciente'. Below it is a button 'Cambio Búsqueda de Paciente'. A table displays search results with columns: Nombre, Identificador, Género, Fecha de Nacimiento, Médico, Clínica, and Problema de Salud. The first row shows 'No Permitido' for all fields.

Nombre	Identificador	Género	Fecha de Nacimiento	Médico	Clínica	Problema de Salud
No Permitido	12	No Permitido	No Permitido	No Permitido	No Permitido	No Permitido

Figura 37: Búsqueda de información por parte de otro paciente

- **Lectura de la información del paciente por parte de su médico**

Para el caso de las búsquedas de los datos de los pacientes por su médico, en la figura 38 se muestra que el médico puede acceder a la información de su paciente.



The screenshot shows a user profile for 'Médico' (User: Practitioner1, FHIR ID: 5). The main heading is 'Médico'. Below it is a button 'Cambio Búsqueda de Paciente'. A table displays search results with columns: Nombre, Identificador, Género, Fecha de Nacimiento, Médico, Clínica, and Problema de Salud. The first row shows patient information: Laura Pérez, ID 12, female, born 2020-04-23, associated with doctor 5 and clinic 2. The health problem is 'Glucosa [Moles/Volumen] in Blood + Hearth Disease + High Pressure(Blood Pressure 125 )'.

Nombre	Identificador	Género	Fecha de Nacimiento	Médico	Clínica	Problema de Salud
Laura Pérez	12	female	2020-04-23	5	2	Glucosa [Moles/Volumen] in Blood + Hearth Disease + High Pressure(Blood Pressure 125 )

Figura 38: Búsqueda de información por el Médico del paciente

- **Lectura de la información del paciente por parte de otro médico**

En caso de que el solicitante de información no sea el médico del paciente, entonces no se le mostrará la información, como se muestra en la figura 39.



The screenshot shows a user profile for 'Médico' (User: Practitioner1, FHIR ID: 5). The main heading is 'Médico'. Below it is a button 'Cambio Búsqueda de Paciente'. A table displays search results with columns: Nombre, Identificador, Género, Fecha de Nacimiento, Médico, Clínica, and Problema de Salud. The first row shows 'No Permitido' for all fields.

Nombre	Identificador	Género	Fecha de Nacimiento	Médico	Clínica	Problema de Salud
No Permitido	11	No Permitido	No Permitido	No Permitido	No Permitido	No Permitido

Figura 39: Búsqueda de información del paciente por parte de otro médico

- **Lectura de la información de un paciente por parte de la organización**

En este caso de uso solamente la organización dónde está afiliado el paciente podrá acceder a la información personal del paciente; pero no podrá acceder a la información de su condición médica, como se muestra en la figura 40.



Nombre	Identificador	Género	Fecha de Nacimiento	Médico	Clínica	Problema de Salud
Julia Morais	17	female	1953-01-04	6	1	No Permitido

Figura 40: Búsqueda de información por parte de la organización del paciente

- **Lectura de la información de un paciente por parte de otra organización**

Si la organización que quiere acceder a la información del paciente no es dónde el paciente está registrado, entonces ésta no podrá acceder a la información del paciente, tal como se muestra en la figura 41.



Nombre	Identificador	Género	Fecha de Nacimiento	Médico	Clínica	Problema de Salud
No Permitido	11	No Permitido	No Permitido	No Permitido	No Permitido	No Permitido

Figura 41: Búsqueda de información por parte de otra organización

### 7.1.2 Casos de uso conteo de pacientes por criterio de búsqueda

En este apartado se mostrará las casuísticas de búsqueda por cantidad de los pacientes que presenten ciertas características definidas en los criterios de búsqueda: Rango de Edad, Condición (Implementada 3 veces) y Rango de Presión Sanguínea; estos criterios de búsqueda se realizan sobre la base de pacientes que tiene cada médico u organización; pero también estos dos roles mencionados tienen la opción de hacer búsquedas como administrador, como se muestra en la figura 42.

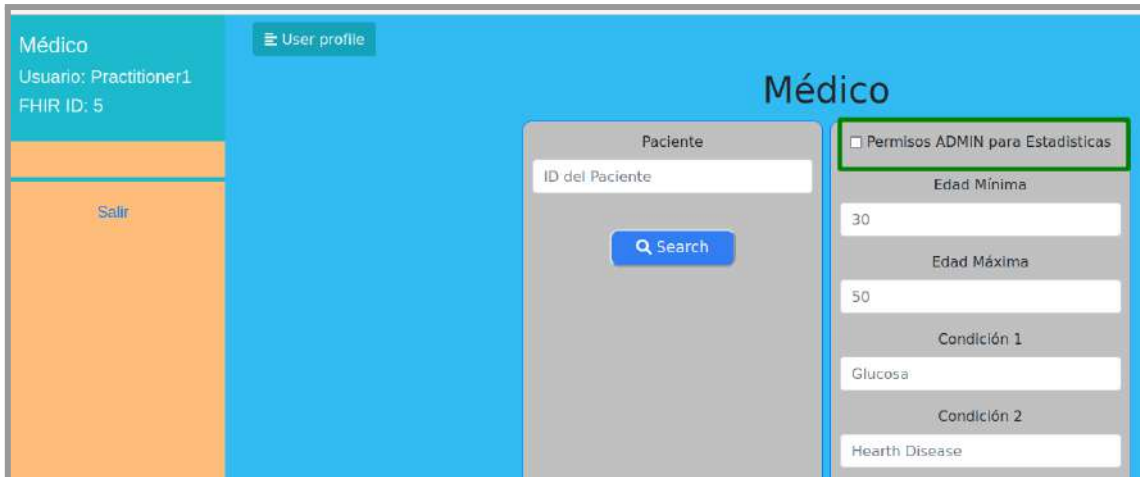


Figura 42: Opción de búsqueda como administrador

- **Búsqueda de pacientes por rango de edad**

En este apartado se mostrará el cálculo de la cantidad de pacientes que están dentro de un rango de edad; en la figura 43 se muestra el resultado de una búsqueda.



Figura 43: Cantidad de pacientes por edad

- **Búsqueda de pacientes por Rango de presión sanguínea**

Una de los criterios de búsqueda es el rango de presión sanguínea del paciente, del cual el aplicativo calcula la cantidad de pacientes con este criterio; en la figura 44 se muestra el resultado de una búsqueda.



Figura 44: Cantidad de pacientes por presión sanguínea

- **Búsqueda por condición médica del paciente**

Para esta búsqueda, el usuario elegirá alguna condición particular del cual se quiere encontrar la cantidad de pacientes; en el demostrador se pueden elegir hasta 3 parámetros. En la figura 45 se muestra una de las búsquedas realizadas.



Figura 45: Cantidad de pacientes por condición médica

- **Criterios de búsqueda combinables**

Una característica importante del cálculo del número de pacientes por criterio de búsqueda es que el usuario podrá buscar por los criterios que requiera, no será necesario que complete todas las casillas de la búsqueda. En la figura 46 se muestra el detalle de esta búsqueda.



Figura 46: Cantidad de pacientes por todos los criterios de búsqueda

- **Estadística de pacientes por rol**

Una de las características de la implementación realizada es poder realizar búsqueda de pacientes por rol de usuario, en dónde las cantidades de pacientes con los mismos criterios para el rol de médico es diferente para el rol de organización. En la figura 47 se muestra la misma consulta que la realizada en la figura 46, en este caso el resultado de la búsqueda es 16.



Figura 47: Búsqueda de pacientes por rol

- **Búsqueda de pacientes con rol de Administrador**

Otra de las características importantes en el conteo de pacientes por criterio de búsqueda es la facultad que tienen usuarios como los Médicos y las Organizaciones de poder hacer búsquedas en modo administrador. En la figura 46 se muestra la búsqueda con los mismos criterios de la figura 48, en dónde se muestra que el



conteo de pacientes es muy superior al realizado a la lista de pacientes que el médico atiende.



Figura 48: Búsqueda de pacientes en modo Administrador

## 8 Costos y beneficios de la propuesta

En este apartado se describirán en detalle los costos de la implementación realizada, en donde se analiza y se cuantifica en moneda local el esfuerzo invertido en cada instancia del proyecto; adicionalmente, se muestra los beneficios del trabajo realizado y su aporte en contribución académica.

### 8.1 Costos de la implementación

En este apartado se muestra los costos en la implementación de este proyecto, este incluye el detalle de las horas insumidas en cada una de las etapas del proyecto; en la tabla 6 se muestra la lista de tareas realizadas con su respectiva estimación de esfuerzo.

Código de Trabajo	Nombre	Tiempo estimado (h)
WP1	Investigación y Documentación	100
WP2	Experimentación en el uso de HAPI FHIR	20
WP3	Generación de Script HAPI FHIR	16
WP4	Código lógica Authorization Interceptor HAPI FHIR	100
WP5	Back End Aplicativo Demostrador	100
WP6	Front End Aplicativo Demostrador	10
<b>Total WP</b>		<b>346</b>
-	Reuniones semanales	20
-	Propuesta de proyecto y plan de trabajo	10
-	Revisiones a la documentación	30
-	Documento de TFM	60
-	Preparación de la defensa y defensa	20
<b>Total</b>		<b>486</b>

Tabla 6: Estimación de esfuerzo del estudiante

Considerando el trabajo de un desarrollador, el cual según páginas como “Glasdoor”, el sueldo promedio por hora es de 15€; este costo será multiplicado por la cantidad total de horas invertidas con el fin de conseguir el costo del esfuerzo del estudiante; en la tabla 7 se muestra el balance del costo insumido para la implementación de este proyecto.

<b>Actividad</b>	<b>Costo (€)</b>
Tareas de Investigación y Desarrollo	7290
Electricidad (4 meses)	200
Internet (4 meses)	200
<b>Costo Total</b>	<b>7690</b>
Contingencia	1000
<b>Presupuesto Total</b>	<b>8690</b>

Tabla 7: Costo total del proyecto en Euros

## 8.2 Beneficios de la propuesta

La importancia de la elaboración de este proyecto radica en la contribución que realiza tanto académicamente así como su aporte a la mejora de la gestión de la información médica de las instituciones, en términos de seguridad y privacidad de la información. A continuación se describe a detalle lo que el autor considera la contribución de este proyecto.

- Generar información base para el desarrollo de nuevos proyectos relacionados con seguridad basados en el estándar HL7 FHIR.
- Brindar una estrategia distinta de ofrecer seguridad y privacidad de la información usando como base el estándar HL7 FHIR.
- Demostrar que se puede implementar un sistema de seguridad y privacidad de la información en instituciones de salud con bajo presupuesto.
- El desarrollo de un aplicativo demostrador que permite el acceso a la información de los pacientes solamente por parte de las personas autorizadas.
- La implementación de un Authorization Interceptor en el aplicativo HAPI FHIR que permite centralizar el control de acceso desde la fuente de información.





- La implementación del módulo de estadísticas, lo que permite realizar la cuenta de pacientes de acuerdo a los criterios de búsqueda.
- La implementación de una interfaz de usuario que restringe el acceso al aplicativo a través de un usuario y contraseña.
- Lograr cierto grado de eficiencia en el código evitando realizar operaciones costosas en tiempo de procesamiento.

## 9 Trabajo futuro

Una de las ideas iniciales del proyecto era ofrecer un sistema de seguridad de pacientes usando el sistema de etiquetas mencionado en el estándar FHIR, pero debido a la poca o casi nula información disponible por parte del aplicativo HAPI FHIR no fue posible implementarlo, por lo que el desarrollar un sistema de seguridad basado en etiquetas queda como un trabajo a futuro.

Este proyecto involucra la Implementación de Seguridad y Privacidad de datos clínicos con el estándar HL7 FHIR, para lo cual el diseño utilizó una de las características principales del aplicativo HAPI el “Authorization Interceptor” para construir un “Role Based Access Control”; sin embargo, no es parte de este proyecto implementar un protocolo específico de seguridad para la web como es el caso de OAuth, lo cual se dejará para realizarlo en un trabajo futuro.

En este proyecto se ha implementado un control de acceso simple basado en usuario y contraseña, para lo cual se ha implementado una base de datos que contenga esta información para cada usuario, la información de esta base de datos se enlaza con la de la base de datos de HAPI a través del campo llave ID FHIR, pero para garantizar la seguridad se podría utilizar una llave aleatoria basado en un “Bearer Token” lo que quedará para un trabajo futuro el implementarlo.

La implementación se basa en la construcción de un aplicativo “Front End” y “Back End” que se alimentan de la información de los pacientes a través de la API del aplicativo HAPI FHIR, el consumo de esta información está basada en archivos “json” simples sin encriptar, queda para un trabajo futuro el poder enviar y recibir información a través de archivos “json” o “xml” encriptados con el fin de proteger de punta a punta la información de los pacientes.

## 10 Conclusiones

En las pruebas realizadas a la implementación de reglas en el aplicativo HAPI FHIR se logró la restricción de acceso a la información de pacientes por parte de usuarios que no estaban autorizados, con lo cual se recomienda el uso de los métodos de HAPI que permiten aplicar reglas a la información de los usuarios como parte del sistema de seguridad y privacidad de los usuarios.

La implementación de los módulos “Front End” y “Back End” permitió una mayor interoperabilidad con el aplicativo HAPI FHIR en la medida de poder acceder a la información de pacientes para poder realizar el conteo de acuerdo a los criterios de búsqueda requerido por el usuario; adicionalmente, es muy viable poder implementar todo un módulo de estadísticas para aprovechar a gran escala la infraestructura implementada.

Para el cálculo del número pacientes según los criterios de búsqueda requeridos por el solicitante, se aplicó los criterios a cada paciente sin leer su información sensible como domicilio, identificador clínico, sexo, entre otros; y además la información leída de los pacientes es almacenada en memoria RAM por cortos periodos de tiempo y sin posibilidad de almacenamiento en una memoria de largo plazo como una base de datos; con esto se cumple con lo requerimiento de privacidad de información de los usuarios.

Si bien es cierto, el módulo de control de acceso implementado desde HAPI y con el uso del aplicativo demostrador provee protección de datos de pacientes, este puede ir acompañado de métodos de seguridad de la información como “Anonymization”, “Pseudonymization”, seguridad web OAuth2, entre otros. Con esto se lograría el robustecimiento del sistema de seguridad ya implementado.

## 11 Referencias bibliográficas

- [1] Patrick Pyette, John Mike Davis, Kathleen Connor. HL7 Version 3 Standard: Privacy, Access and Security Services; Security Labeling Service, Release 1. (2019). Health Level Seven International.  
[https://www.hl7.org/implement/standards/product\\_brief.cfm?product\\_id=360](https://www.hl7.org/implement/standards/product_brief.cfm?product_id=360). [Accedido 13 01 2022]
- [2] ©2007-2022 Health Level Seven International. FHIR Infrastructure.  
<http://www.hl7.org/Special/committees/fiwwg/overview.cfm>. [Accedido 21 01 2022]
- [3] Jason Walonoski. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. (2017). Journal of the American Medical Informatics Association.  
<https://academic.oup.com/jamia/article/25/3/230/4098271>. [Accedido 21 02 2022]
- [4] Microsoft (2020) fhir-server [Código Fuente]. <https://github.com/microsoft/fhir-server>
- [5] Copyright Microsoft (2022) [Documentación FHIR Service].  
<https://docs.microsoft.com/en-us/azure/healthcare-apis/fhir/overview>. [Accedido 08 01 2022]
- [6] Copyright Firely (2021) [Documentación FHIR Service].  
<https://docs.fire.ly/projects/Firely-Server/en/latest/overview.html>. [Accedido 10 01 2022]
- [7] Copyright Google (2021) [Documentación Google FHIR]. <https://github.com/google/fhir>. [Accedido 08 01 2022]
- [8] Copyright HAPI FHIR (2022) [Documentación HAPI FHIR].  
[https://hapifhir.io/hapi-fhir/docs/getting\\_started/introduction.html](https://hapifhir.io/hapi-fhir/docs/getting_started/introduction.html). [Accedido 06 01 2022]
- [9] Copyright HAPI FHIR (2022) [Documentación HAPI FHIR]. <https://hapifhir.io/hapi-fhir/>. [Accedido 06 01 2022]
- [10] HL7 Standards - Section 1 (2022) [Principales Estándares].  
[https://www.hl7.org/implement/standards/product\\_section.cfm?section=1](https://www.hl7.org/implement/standards/product_section.cfm?section=1). [Accedido 05 01 2022]

- [11] FHIR® R4 (HL7 Fast Healthcare Interoperability Resources, Release 4)(2022)  
[Descripción]. [https://www.hl7.org/implement/standards/product\\_brief.cfm?product\\_id=491](https://www.hl7.org/implement/standards/product_brief.cfm?product_id=491).  
[Accedido 01 03 2022]
- [12] FHIR 4.0.1 (2022) [Summary]. <http://hl7.org/fhir/summary.html>. [Accedido 01 03 2022]
- [13] FHIR 4.0.1 Seguridad y Privacidad (2022) [Módulo de Seguridad y Privacidad].  
<http://hl7.org/fhir/secpriv-module.html>. [Accedido 05 01 2022]
- [14] JPA SERVER (2022) [Introduction].  
[https://hapifhir.io/hapi-fhir/docs/server\\_jpa/introduction.html](https://hapifhir.io/hapi-fhir/docs/server_jpa/introduction.html). [Accedido 01 03 2022]
- [15] HAPI FHIR [Interceptors]. <https://hapifhir.io/hapi-fhir/docs/interceptors/interceptors.html>.  
[Accedido 02 01 2022]
- [16] HAPI FHIR [REST Server Security].  
<https://hapifhir.io/hapi-fhir/docs/security/introduction.html>. [Accedido 21 12 2021]
- [17] HL7 Organization , "RESTful API," HL7 Organization , 19 4 2017. [Online]. Available:  
<https://www.hl7.org/fhir/http.html#2.21.0>. [Accedido 21 12 2021]
- [18] M. Fowler, "Richardson Maturity Model," martinowler, 18 3 2010. [Online]. Available:  
<https://martinfowler.com/articles/richardsonMaturityModel.html>. [Accedido 14 01 2022]
- [19] IntelliJ IDEA (2022) [Descripción]. <https://www.jetbrains.com/idea/features/>. [Accedido 14 01 2022]
- [20] Postman (2022) [Introducción]. <https://www.postman.com/product/what-is-postman/>.  
[Accedido 14 01 2022]
- [21] NetBeans (2022) [Introducción]. <https://ca.wikipedia.org/wiki/NetBeans>. [Accedido 14 01 2022]
- [22] Github (2022) [Introducción]. <https://github.com/>. [Accedido 14 01 2022]
- [23] Apache Maven Project (2022). [Introducción]. <https://maven.apache.org/>. [Accedido 14 01 2022]
- [24] HL7 FHIR JSON (2022) [Representación de los recursos]. <http://hl7.org/fhir/r4/json.html>.  
[Accedido 14 01 2022]

[25] HL7 FHIR (2022) [Versionamiento]. <http://hl7.org/fhir/directory.html>. [Accedido 14 01 2022]

[26] HL7 FHIR Security (2022) [Definición]. <https://www.hl7.org/fhir/security.html>. [Accedido 14 01 2022]

[27] HL7 FHIR [Niveles de Implementación]. <https://www.hl7.org/fhir/modules.html>. [Accedido 14 01 2022]

[28] Role-based Access Control (2022) [Definición].  
[https://en.wikipedia.org/wiki/Role-based\\_access\\_control](https://en.wikipedia.org/wiki/Role-based_access_control). [Accedido 14 01 2022]

[29] Attribute -based Access Control (2022) [Definición].  
[https://en.wikipedia.org/wiki/Attribute-based\\_access\\_control](https://en.wikipedia.org/wiki/Attribute-based_access_control). [Accedido 14 01 2022]

[30] HL7 FHIR XML (2022) [Representación de los recursos].  
<https://www.hl7.org/fhir/xml.html>. [Accedido 14 01 2022]

[31] openEHR (2022) [Definición]. [https://www.openehr.org/about/what\\_is\\_openehr](https://www.openehr.org/about/what_is_openehr). [Accedido 14 01 2022]

[32] openEHR (2022) [Seguridad y Confidencialidad]  
<https://specifications.openehr.org/releases/1.0.1/html/architecture/overview/Output/security.html>. [Accedido 14 01 2022]

[33] Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. Jefatura del Estado. BOE.  
<https://www.boe.es/buscar/pdf/2018/BOE-A-2018-16673-consolidado.pdf>.

[34] Reglamento General de Protección de Datos (RGPD) (2022) [Información].  
<https://gdpr-info.eu/>. [Accedido 14 01 2022]

[35] Eclipse Foundation (2022) [Proyecto Jetty] <https://www.eclipse.org/jetty/>. [Accedido 14 01 2022]

[36] SMART (2022) [Descripción] <https://smarthealthit.org/>. [Accedido 14 01 2022]

[37] HL7 FHIR, SMART (2022) [Smart App Launch].  
<http://www.hl7.org/fhir/smart-app-launch/>. [Accedido 14 01 2022]

[38] HL7 FHIRcast (2017) [Información General]. <https://fhircast.org/>. [Accedido 14 01 2022]

[39] HL7 FHIR WebSocket (2022) [R5 Backport].

<https://build.fhir.org/ig/HL7/fhir-subscription-backport-ig/channels.html>. [Accedido 14 01 2022]

[40] Lenguaje Unificado de Modelado (2022) [Definición UML].

<https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml>. [Accedido 14 01 2022]

[41] Smile CDR Consent Service (2022) [Descripción].

[https://smilecdr.com/docs/security/consent\\_service.html](https://smilecdr.com/docs/security/consent_service.html). [Accedido 14 01 2022]

[42] HAPI FHIR Authorization Interceptor (2022) [Descripción].

[https://hapifhir.io/hapi-fhir/docs/security/authorization\\_interceptor.html](https://hapifhir.io/hapi-fhir/docs/security/authorization_interceptor.html). [Accedido 14 01 2022]

[43] Smile CDR Authorization and Consent Service (2022) [Descripción].

[https://smilecdr.com/docs/security/authorization\\_and\\_consent.html](https://smilecdr.com/docs/security/authorization_and_consent.html). [Accedido 14 01 2022]

[44] HAPI FHIR 5.6.0 (Raccoon) (2022) [Último Release Estable].

<https://github.com/hapifhir/hapi-fhir/releases/tag/v5.6.0>

[45] hapi-fhir-jpaserver-starter 5.7.0 (2022) [Último Release].

<https://github.com/hapifhir/hapi-fhir-jpaserver-starter>

[46] Tony Sahama, Leonie Simpson (2013). Information Security Discipline Science and Engineering Faculty Queensland University of Technology, Brisbane, Australia: Security and Privacy in eHealth: is it possible?.

[https://www.researchgate.net/publication/269329631\\_Security\\_and\\_Privacy\\_in\\_eHealth\\_Is\\_it\\_possible](https://www.researchgate.net/publication/269329631_Security_and_Privacy_in_eHealth_Is_it_possible)

[47] Dimitra Liveri, Anna Sarri, Christina Skouloudi, ENISA (2015). Security and Resilience in eHealth: Security Challenges and Risks.

<https://www.enisa.europa.eu/publications/security-and-resilience-in-ehealth-infrastructures-and-services>

[48] HL7 FHIR (2022) [Lenguajes]. <https://www.hl7.org/fhir/languages.html>. [Accedido 11 06 2022]



[49] HL7 FHIR (2022) [Conformance-module].

<https://www.hl7.org/fhir/conformance-module.html>. [Accedido 11 06 2022]

[50] Indicadores de Salud, Aspectos conceptuales y operativos (Washington DC - 2018).

[https://iris.paho.org/bitstream/handle/10665.2/49058/9789275320051\\_spa.pdf?sequence=5](https://iris.paho.org/bitstream/handle/10665.2/49058/9789275320051_spa.pdf?sequence=5).

[Accedido 16 06 2022]

[51] Enfermedades Cardiovasculares.

[https://www.who.int/es/health-topics/cardiovascular-diseases#tab=tab\\_1](https://www.who.int/es/health-topics/cardiovascular-diseases#tab=tab_1). [Accedido 16 06

2022]

[52] Java Garbage Collector.

<https://www.arquitecturajava.com/java-garbage-collector/>. [Accedido 16 06 2022]