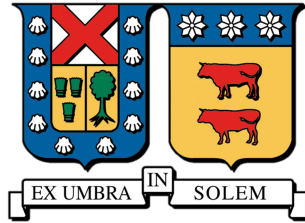


Universidad Técnica Federico Santa María
Departamento de Electrónica
Ingeniería Civil Telemática



Aplicación basada en Inteligencia Artificial que asista a radiólogos en estudios de Diagnóstico Diferencial de tomografías torácicas

Memoria para optar al título de Ingeniero Civil Telemático

Autor:
Piero Oporto Aros

Profesor Guía:
Mauricio Araya PhD

Profesor Correferente:
Patricio Olivares

Resumen

La pandemia de COVID-19 ha llevado a un aumento significativo en la demanda de servicios de salud, destacando la necesidad de herramientas y sistemas que permitan diagnósticos precisos y eficientes. Los profesionales de la salud enfrentan el desafío de proporcionar resultados tempranos y exactos, lo que requiere una exploración exhaustiva de diversos diagnósticos diferenciales.

Para abordar esta necesidad, el grupo ALPACS ha desarrollado Proximity, una plataforma de software diseñada para trabajar de manera interoperable con bases de datos de imágenes médicas (PACS) y reportes clínicos. Proximity utiliza modelos avanzados de inteligencia artificial, incluidas redes neuronales convolucionales (CNN) y procesamiento de lenguaje natural (NLP), para analizar y extraer información significativa de grandes conjuntos de datos de manera eficiente.

El objetivo principal de Proximity es la recuperación de imágenes médicas basadas en contenido (CBMIR), utilizando volúmenes tridimensionales de tomografías computarizadas (CT) de tórax como consulta inicial. El software recupera y muestra imágenes anonimizadas de pacientes previamente diagnosticados con condiciones visualmente similares, ayudando a los especialistas en salud en el proceso de diagnóstico diferencial para una amplia gama de enfermedades. Al integrar tecnologías avanzadas, Proximity mejora significativamente la atención médica, proporcionando un soporte crucial para diagnósticos precisos e investigaciones médicas avanzadas.

Abstract

The COVID-19 pandemic has led to a significant increase in the demand for healthcare services, underscoring the need for tools and systems that enable accurate and efficient diagnoses. Healthcare professionals face the challenge of providing early and precise results, necessitating thorough exploration of various differential diagnoses.

To address this need, the ALPACS group has developed Proximity, a software platform designed to work interoperably with medical image databases (PACS) and clinical reports. Proximity leverages advanced artificial intelligence models, including convolutional neural networks (CNN) and natural language processing (NLP), to efficiently analyze and extract meaningful information from large datasets.

The primary goal of Proximity is content-based medical image retrieval (CBMIR), using three-dimensional volumes of chest computed tomography (CT) scans as the initial query. The software retrieves and displays anonymized images of patients previously diagnosed with visually similar conditions, aiding healthcare specialists in the differential diagnosis process for a wide range of diseases. By integrating advanced technologies, Proximity significantly enhances medical care, providing crucial support for accurate diagnoses and advanced medical research.

Glosario

- **Backend:** Parte de un sistema o aplicación informática que se encarga del procesamiento y gestión de datos en el servidor. Incluye la lógica de negocio y la comunicación con el frontend.
- **Condensación:** Anomalía en el pulmón.
- **Comorbilidad:** Término utilizado para describir dos o más trastornos o enfermedades que ocurren en la misma persona.
- **Corte Axial:** Corte que divide al paciente en superior e inferior
- **Corte Transversal:** Volumen de imágenes 3D correspondientes a un órgano en particular y en periodo de tiempo previamente determinado.
- **Diagnóstico Diferencial:** Metodología de comparación de casos en pos de ayudar a hacer un diagnóstico final cuando se tienen síntomas sin una causa clara. Es muy utilizado en la radiología.
- **Embedding:** Representación numérica de datos como palabras, frases o imágenes, que captura características y relaciones importantes en un espacio vectorial. En el procesamiento del lenguaje natural, los embeddings representan palabras o frases, mientras que en el procesamiento de imágenes, los embeddings representan características visuales de las imágenes.
- **Estudio:** En el ámbito médico, una o más series de exámenes, pueden ser de diferentes tipos, y representan a un paciente en particular.

- **Frontend:** Es la interfaz de usuario y la experiencia que percibe el usuario al utilizar un software o sitio web. Se refiere específicamente a la tecnología y el código que se ejecuta en el navegador del usuario.
- **Hallazgo Clínico:** Conjunto de signos y síntomas encontrados cronológicamente, junto con los resultados y las interpretaciones asociadas.
- **HAPI/FHIR:** es una implementación de código abierto del estándar FHIR (Fast Healthcare Interoperability Resources), diseñada para facilitar el intercambio de datos médicos entre sistemas de salud de manera eficiente y efectiva. Proporciona herramientas y servidores que permiten a los desarrolladores crear, almacenar y gestionar información médica utilizando el formato estandarizado de FHIR. Es utilizado ampliamente en aplicaciones de salud digital para mejorar la interoperabilidad y la gestión de datos clínicos.
- **HCUCH:** Hospital Clínico de la Universidad de Chile.
- **Instancia:** En el ámbito médico, también llamado corte transversal, son imágenes que representan una sección del cuerpo, que, en su conjunto, forman una imagen tridimensional de este. Un conjunto de instancias, forman una serie.
- **Inteligencia Artificial:** Campo de estudio y desarrollo de sistemas informáticos que pueden simular y realizar tareas que normalmente requieren de la inteligencia humana. Utiliza algoritmos y técnicas para aprender, razonar, reconocer patrones y tomar decisiones, con el objetivo de realizar tareas de manera autónoma y eficiente.
- **PACS:** Acrónimo en inglés para "Picture Archiving and Communication System". Se trata de un sistema computarizado para el archivado

digital de imágenes médicas y para la transmisión de estas a estaciones de visualización dedicadas o entre estas a través de una red informática

- **Parénquima pulmonar:** Lo que compone los pulmones.
- **Red Neuronal:** Modelo computacional inspirado en el cerebro humano que procesa información mediante unidades llamadas neuronas interconectadas.
- **Stakeholder:** Cada una de las partes involucradas en el proyecto.
- **Serie:** En el ámbito médico, conjunto de instancias, que juntas forman la reconstrucción de un examen. Un conjunto de series forman un estudio.
- **Tomografía computarizada:** Procedimiento con imágenes que usa equipo especial de rayos X para crear imágenes detalladas, o exploraciones, de regiones internas del cuerpo. El resultado es un volumen de datos que recrean el área de estudio de la persona por capas o instancias.
- **Ventana Pulmonar:** La ventana pulmonar es una configuración de ventana que se utiliza en las imágenes de tomografía computarizada para visualizar el tejido pulmonar. Se define por dos parámetros: el nivel de la ventana y el ancho de la ventana.

Contenidos

Glosario	iv
Figuras	x
Tablas	xii
1 Introducción	1
1.1 Problema y oportunidad	1
1.2 Equipos de desarrollo asociados a Proximity	2
1.2.1 Programa de Memorias Multidisciplinarias	2
1.2.2 Equipos internos de Proximity	3
1.3 Objetivos	3
1.3.1 Objetivos Específicos	3
2 Estado del Arte y de la Técnica	5
2.1 Ámbito médico	5
2.1.1 Tomografías computarizadas y diagnóstico diferencial	5
2.1.2 Interoperabilidad	6
2.2 Ámbito Tecnológico	8
2.2.1 Desarrollo de software	8

<i>CONTENIDOS</i>	viii
2.2.2 MLOps	11
2.3 Competencia asociada a Proximity	12
3 Contexto de solución	14
3.1 Proximity en su versión 1.0	14
3.2 Propuesta de desarrollo	15
3.3 Definición de tecnologías médicas utilizadas	16
4 Desarrollo de la solución	18
4.1 Diseño de solución	18
4.1.1 Situación inicial y estado de oportunidad	18
4.1.2 Metodología utilizada	20
4.1.3 Etapas sistemáticas	20
4.1.4 Requisitos del Sistema	23
4.1.5 Diseño de Arquitectura de Aplicación Web	25
4.1.6 Descripción de componentes de la aplicación web	27
4.1.7 Descripción de componentes externos al desarrollo de la aplicación web	30
4.1.8 Diseño de Interfaces	31
4.1.9 Diseño MLOps	39
4.2 Implementación de la solución	44
4.2.1 Desarrollo de Software	45
4.2.2 MLOps	50
5 Validación y conclusiones	59
5.1 Validación de Aplicación Web	59
5.2 Validación de Datos	61
5.3 Conclusiones	62

CONTENIDOS

ix

Bibliografía

63

Figuras

2.1	Ciclo de vida MLOps [1]	12
4.1	Inicio Proximity v1.0	19
4.2	Resultados Proximity v1.0	19
4.3	Diagrama de contexto	25
4.4	Diagrama de arquitectura de aplicación web	26
4.5	Modelo de Navegación	31
4.6	Paleta de colores primarios	33
4.7	Paleta versión nocturna	34
4.8	Vista inicio de sesión	36
4.9	Vista principal (modo nocturno)	37
4.10	Vista resultados de consulta	37
4.11	Vista de comparación (modo nocturno)	38
4.12	Vista de comparación detallada (modo nocturno)	38
4.13	MLOps pipeline [1]	39
4.14	Pipeline para recuperación de datos	40
4.15	Entrenamiento y recuperación de resultados.	42
4.16	Stack de tecnologías	44
4.17	Vista login PMV	45

<i>FIGURAS</i>	xi
4.18 Vista dashboard PMV	46
4.19 Vista búsqueda PMV	47
4.20 Vista recuperación PMV	48
4.21 Vista comparación PMV	48
4.22 Endpoint de FastAPI para recuperación de instancias.	50
4.23 Reconstrucciones tomografía computarizada	51
4.24 Pipeline de datos dentro del servidor de Proximity	52
4.25 Árbol de carpetas ordenadas	53
4.26 Representación de distribución de instancias	57
4.27 Distribución de valores HU	58

Tablas

2.1	Ejemplos de interoperabilidad semántica y sintáctica	7
2.2	Comparación entre frameworks de frontend	9
2.3	Comparación entre frameworks de backend	10
2.4	Resumen empresas de competencia	13

CAPÍTULO 1

Introducción

1.1 Problema y oportunidad

Tanto durante, como pospandemia, se ha observado un incremento significativo en la afluencia de pacientes a los recintos de salud. Este aumento de la demanda de atención médica ha generado la necesidad de contar con herramientas y sistemas que permitan realizar diagnósticos más acertados de manera eficiente [2] [3]. Los profesionales de la salud se enfrentan al desafío de brindar un resultado temprano y preciso, lo que implica la exploración exhaustiva de diversos diagnósticos diferenciales [4].

Dado el contexto anterior, se presenta el desafío y la oportunidad que da paso al desarrollo de un software llamado Proximity en manos del grupo ALPACS: un consorcio de distintas instituciones médicas y educacionales, que trabajan en conjunto con el fin de desarrollar una plataforma destinada a alojar procesar y proveer imágenes médicas de manera interoperable.

Proximity es una plataforma de software que se diseñó para trabajar de manera interoperable con bases de datos que contienen imágenes médicas (PACS) y reportes clínicos. Este software emplea modelos avanzados de inteligencia artificial que utilizan técnicas sofisticadas como redes neuronales convolucionales (CNN), y procesamiento de lenguaje natural (NLP). Estas técnicas permiten analizar y extraer información significativa de grandes volúmenes de datos en períodos de tiempo considerablemente reducidos, una tarea que resultaría prácticamente imposible de llevar a cabo de manera eficiente por humanos.

El objetivo de esta aplicación es la recuperación de imágenes médicas basadas en contenido (CBMIR) [5], empleando volúmenes tridimensionales de tomografías computarizadas (CT) de tórax como consulta inicial. El software recupera y muestra por pantalla imágenes anonimizadas de pacientes previamente diagnosticados con enfermedades visualmente similares, facilitando el proceso de diagnóstico diferencial realizado por especialistas de la salud. Así, Proximity integra tecnologías avanzadas para mejorar significativamente la atención médica, proporcionando soporte crucial tanto para diagnósticos precisos como para investigaciones médicas avanzadas.

1.2 Equipos de desarrollo asociados a Proximity

1.2.1 Programa de Memorias Multidisciplinarias

Para desarrollar la solución presentada en esta memoria se trabajó en conjunto con un grupo de alumnos de la Universidad Técnica Federico Santa María, mediante el Programa de Memorias Multidisciplinarias (PMM), en el cual cada integrante asumió una responsabilidad en el desafío con el propósito de desarrollar tanto el frontend como el backend asociado a la

plataforma. Sin embargo, en la presente memoria se trabajó de manera más específica en tareas asociadas al backend y al flujo de datos.

1.2.2 Equipos internos de Proximity

Dentro de la orgánica interna del proyecto Proximity se encuentran tres equipos principales:

- **Data Science:** encargado de realizar investigaciones necesarias con el fin de obtener modelos de IA que se adapten al proyecto, procurando que la recuperación de imágenes médicas más precisas.
- **Infraestructura:** encargado de mantener los servicios de redes informáticas activas. Además son encargados de brindar soporte a los servidores que son relevantes para proyecto, como el data center donde se realizan entrenamientos de Data Science, servidor de almacenamiento de archivos, página web informativa y aplicación web Proximity.
- **Equipo Médico:** grupo de médicos del HCUCH encargados de guiar y brindar conocimiento del área médica, tanto para el desarrollo de la aplicación web, como para la validación de resultados de los modelos de inteligencia artificial.

1.3 Objetivos

1.3.1 Objetivos Específicos

OE1: Generar prototipos de aplicación implementando las historias de usuario

Iterar prototipos adecuando la aplicación a las necesidades y recomendaciones de los especialistas.

OE2: Crear Interfaz gráfica

Generar una interfaz gráfica para la aplicación, con las funcionalidades asociadas a esta.

OE3: Definir arquitectura de servicios externos necesarios para la aplicación

Crear arquitectura y definir datos de entrada para habilitar funcionalidades de repositorio PACS y HAPI/FHIR.

OE4: Definir un flujo de datos

Crear un pipeline destinado a una eficiente entrada y salida de datos para ser consumidos según se estime conveniente.

OE5: Integrar modelo de IA junto al backend

Crear un backend robusto, capaz de utilizar un modelo de IA para procesar las solicitudes de los usuarios.

OE6: Testear aplicación

Realizar pruebas funcionales a la aplicación, velando por el correcto funcionamiento y seguridad de datos.

CAPÍTULO 2

Estado del Arte y de la Técnica

2.1 Ámbito médico

2.1.1 Tomografías computarizadas y diagnóstico diferencial

A medida que pasan los años, se crean importantes avances tecnológicos en el área médica. Específicamente, en el diagnóstico diferencial estos avances han permitido desarrollar nuevas herramientas que dan paso a diagnósticos más precisos y sensibles.

El diagnóstico diferencial consiste en elaborar una lista de posibles enfermedades para un paciente, basada en un análisis exhaustivo de un estudio [6]. Este es un proceso que requiere de un alto grado de experiencia y pericia, incluso los profesionales más experimentados pueden verse sesgados por sus propias experiencias o prejuicios. Esto se debe a que es un tema muy complejo, con muchas posibles causas para un mismo conjunto de síntomas. Por lo tanto, esperar que un experto tenga un diagnóstico acertado el cien

por ciento de las veces, es simplemente imposible. Sin embargo, con la ayuda de un software como Proximity el objetivo es reducir en gran medida la tasa de error en los diagnósticos.

La tomografía computarizada (CT por sus iniciales en inglés) es una técnica conocida en el área de la medicina, la cual es una prueba de diagnóstico por imágenes que utiliza rayos X para crear imágenes volumétricas del interior del cuerpo. Las CT se pueden utilizar para diagnosticar una amplia gama de condiciones, incluyendo enfermedades del corazón, del sistema respiratorio, digestivo, músculo-esquelético y nervioso. [7]

Las tomografías computarizadas son una herramienta valiosa en el diagnóstico diferencial, ya que pueden ayudar a los médicos a descartar o confirmar una amplia gama de posibles causas de los síntomas de un paciente, pero son particularmente útiles para diagnosticar enfermedades que pueden causar cambios en la estructura interna del cuerpo. Por ejemplo, una CT puede ayudar a diagnosticar enfermedades ligadas al tórax como la neumonía, ya que se presenta como áreas opacas en los pulmones.

2.1.2 Interoperabilidad

La interoperabilidad, en simples palabras, se entiende como poder ocupar un mismo recurso en varios servicios a la vez de manera íntegra. La IEEE define interoperabilidad como "la capacidad de dos o más sistemas o componentes para intercambiar información y utilizar dicha información que ha sido intercambiada". Esta rama de la información puede ser catalogada en dos sub ramas, las cuales dependen del contexto del que se necesite [8]. En primer lugar, se tiene la interoperabilidad sintáctica, cuyo foco es lograr que los sistemas de información puedan leer y escribir los archivos médicos de manera estándar en un mismo formato, sin importar el terminal en el que se utilice

dicha data. El tipo archivo DICOM [9] (foo.dcm), con el cual se trabaja en la presente memoria, es un ejemplo de la implementación de lo anteriormente mencionado. La interoperabilidad sintáctica es muy importante para poder lograr implementar la segunda sub rama: La Interoperabilidad Semántica. La interoperabilidad semántica se refiere a la capacidad de los sistemas informáticos para comprender el significado de los datos que intercambian. Esto significa que los sistemas deben tener un entendimiento común de los términos y conceptos utilizados en los datos. En la Tabla 2.1 se presentan tres ejemplos de cada uno de los tipos de interoperabilidad.

	Sintáctica	Semántica
Ejemplo	Archivos .dcm Formato XML Formato JSON	Vocabulario controlado SNOMED CT.[10] Lenguaje de modelado unificado (UML) Lenguaje de consulta estructurado (SQL)

Tabla 2.1: Ejemplos de interoperabilidad semántica y sintáctica

Lograr una correcta implementación de ambos conceptos en conjunto, permite que los sistemas informáticos intercambien datos de manera eficiente y efectiva. Esto puede conducir a una mejora en la calidad de los datos, una reducción de los costos y una mejora en la eficiencia. Es por eso, que en el área médica, la organización Health level Seven international (HL7) [11] creó un conjunto de reglas para transmitir datos de salud entre sistemas de atención médica, y un sistema de representación de datos de salud llamado Fast Healthcare Interoperability Resources (FHIR) [12] para garantizar la interoperabilidad de los datos. Por otra parte, OpenHIE, una iniciativa comunitaria, se presenta para facilitar intercambio de datos de salud como tal, utilizando para ello los estándares de FHIR y HL7.

2.2 Ámbito Tecnológico

2.2.1 Desarrollo de software

En un proyecto como Proximity, la escalabilidad es una arista muy importante al momento de desarrollar una solución. Es por esto, que se investigaron las ventajas y desventajas de los principales frameworks utilizados en la industria por grandes empresas. En la Tabla 2.2 se muestran algunos de los frameworks de frontend más utilizados. Por otra parte, para el backend, como decisión de proyecto, se optó por utilizar el lenguaje de programación Python para facilitar la integración del modelo de inteligencia artificial, el cual se encuentra escrito en el mismo lenguaje. En la Tabla 2.3 se presentan los principales frameworks de backend para el lenguaje Python.

	Angular	Vue.js	React
Pros	Completo y Robusto. Gran cantidad de funcionalidades. Comunidad activa.	Buen rendimiento general. Simple y ligero.	Flexibilidad para diferentes tipos de aplicaciones. Gran comunidad de usuarios y desarrolladores.
Contras	Puede resultar complejo de aprender. Para aplicaciones pequeñas puede no resultar útil por su peso.	Documentación desorganizada en comparación a otros frameworks. Menos flexible que su competencia.	Alta complejidad en proyectos muy grandes. Necesita utilizar paquetes de terceros.
Empresas que lo utilizan	Google[13] Netflix PayPal	Gitlab[14] Adobe Xiaomi	Deezer[15] Wallmart Facebook
Foco	Proyectos complejos de gran envergadura.	Proyectos pequeños y medianos.	Proyectos con alto contenido dinámico.

Tabla 2.2: Comparación entre frameworks de frontend

	Django	Flask	Fast API
Pros	Robusto. Gran cantidad de documentación.	Muy ligero. Buen rendimiento en proyectos pequeños.	Renderización virtual: solo renderiza los cambios en la IU. Ligero y flexible.
Contras	Poca flexibilidad debido a la gran cantidad de funcionalidades. Mas pesado que su competencia.	Falta de seguridad. Falta de documentación.	Al ser relativamente nuevo, no existe una gran comunidad. La documentación no es tan detallada lo es la de su competencia.
Empresas que lo utilizan	Mozilla Instagram Pinterest	Reddit Netflix GitHub	Spotify Twitch NASA
Foco	Proyectos con alta seguridad y funcionalidades integradas	Proyectos pequeños y ligeros que no necesitan mayores funcionalidades.	Proyectos que necesitan rendimiento con alto contenido dinámico.

Tabla 2.3: Comparación entre frameworks de backend

Es importante hacer notar que el proyecto Proximity, no guarda en un almacenamiento o base de datos local la data de usuarios o imágenes, debido a que se consume desde repositorios interoperables, además debido al flujo que tienen los usuarios en la interfaz, esta se encuentra en constante cambio. En este contexto, es importante notar que React es un framework flexible y puede ser utilizado en proyectos con alto contenido dinámico como lo es Proximity. Por otra parte, Fast API es un framework liviano, pero flexible que permite la renderización virtual, es decir, solo renderiza los cambios en la interfaz de usuario, lo cual lo hace ideal para aplicaciones que tienen

mucha interacción en el Front-end. Ambas tecnologías son utilizadas por grandes empresas para proyectos complejos, lo que justifica su elección para el desarrollo de la versión 2.0 de Proximity.

2.2.2 MLOps

Machine Learning Operations es un área relativamente nueva que surge a partir de la masificación de modelos de Machine Learning y Deep Learning y su necesidad por pasar a producción de manera limpia y segura, además de mantenerse en el tiempo como un elemento funcional [1]. Por lo mismo, es un área que no se encuentra tan desarrollada como DevOps, pero como es lógico pensar, MLOps implementa varios principios de este último campo mencionado.

Por ejemplo, al igual que una nueva funcionalidad de una aplicación web que pasa a producción, utiliza las prácticas de DevOps para agilizar el proceso de entrega, MLOps propicia la creación, implementación y mantención de un modelo de inteligencia artificial para que su despliegue pueda ser lo más sencillo, seguro y rápido posible. Para llevar a cabo este desafío, MLOps se preocupa, a grandes rasgos, desde la curación de la data, facilitación del proceso de entrenamiento y validación, hasta el monitoreo del código en producción, velando porque sus métricas no decaigan en el tiempo (revisar Figura 2.1). Esta última etapa de entrenamiento continuo es una fase vital del pipeline de MLOps, debido a que se adapta el modelo, reentrenándolo con los nuevos datos que los usuarios proporcionan.

A pesar de que existen marcos de trabajo y herramientas para poder implementar un sistema de MLOps, son muchos los data scientists que siguen realizando este proceso de forma manual [16]. Si bien el hecho de automatizar el entrenamiento y ajuste de los modelos, quita flexibilidad y

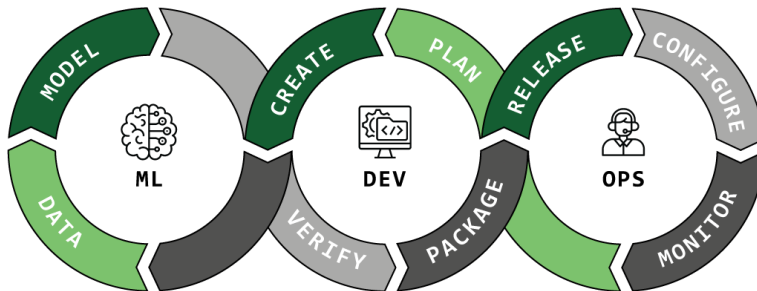


FIGURA 2.1: Ciclo de vida MLOps [1]

control sobre lo que se entrena, sigue siendo una muy buena oportunidad para mantener el constante desarrollo, despliegue y entrenamiento en los proyectos que utilizan machine learning y/o deep learning. Se espera que los proyectos, cuyos datos de entrada y salida no cambian drásticamente a lo largo del tiempo, se agilicen e implementen eficientemente, gracias a estas áreas de desarrollo que se crean en torno a la inteligencia artificial, en aras de facilitar su despliegue y disponibilidad.

2.3 Competencia asociada a Proximity

Al explorar sobre la competencia de Proximity, es posible encontrar diversas empresas en distintos lugares del mundo como NeuralMed en América, Infervision en Asia, y ContextVision en Europa, pero para realizar una comparativa con Proximity se seleccionan a criterio cuatro empresas que desarrollan un software médico con una mirada parecida a la que se tiene en esta memoria.

En primer lugar, se tiene a Gleamer Solutions [17], empresa francesa que trabaja con radiografías de rayos X, utilizando la inteligencia artificial para

	País	Tipo de Estudio
Gleamer Solutions	Francia	Rayos X
Avicenna.ai	Francia	CT
Oxipit.ai	España	Rayos X, CT, Mamografía
Alumbra Toraxia	Chile	Rayos X, CT, Resonancia Magnética

Tabla 2.4: Resumen empresas de competencia

obtener traumatismos en huesos a lo largo del cuerpo. Además, se encuentran trabajando en un modelo capaz de utilizar tomografías computarizadas. En el mismo país, se encuentra la empresa Avicenna.ai [18] obtiene, a partir de tomografías computarizadas, diagnósticos de pecho y cerebro. También en Europa, la empresa española Oxipit.ai [19], actualmente trabaja con rayos X de pecho y músculo esquelético, mamografías, y tomografías computarizadas. Utiliza imágenes médicas, para obtener como resultado una lista de posibles hallazgos que son validadas por el médico asignado. Por último, en Chile, se encuentra Alumbra Toraxia [20], que trabaja con radiografías de tórax (Rayos X), CT de tórax y resonancias magnéticas. Su foco es analizar patrones en imágenes torácicas que presenten anomalías. Además, realizar una asignación de urgencia de pacientes en función de los hallazgos presentes en los estudios. En la Tabla 2.4, se muestra un resumen de los tipos de estudios que utiliza cada empresa y el país de procedencia.

Contexto de solución

3.1 Proximity en su versión 1.0

El desarrollo de Proximity comienza el año 2020, con investigadores, científicos de datos, y desarrolladores encargados de frontend y backend, pertenecientes al grupo ALPACS, siendo ellos los artífices de la versión 1.0.

Esta versión fue una prueba de concepto acerca de lo que se quiere lograr más adelante. Se creó un modelo de inteligencia artificial, el cual es capaz de diferenciar aquellos casos de personas con COVID-19 de las personas que están sanas, analizando tomografías computarizadas de tórax. Esta solución trae consigo una aplicación web básica, que despliega los resultados de los casos más parecidos a una consulta no clasificada por pantalla. Si se pretende comparar la solución mencionada, con la propuesta plasmada en la versión 2.0, son las iteraciones con los médicos que permitieron que el producto sea a medida, una interfaz gráfica más ajustada a las necesidades presentadas

por radiólogos en su día a día. En cuanto al modelo de inteligencia artificial diseñado, el distintivo que permite a ALPACS resaltar sobre la competencia es el uso de volúmenes de imágenes de CT, en vez de realizar consultas con una sola instancia.

Proximity 1.0 concluyó su fase de validación de concepto y ahora en su versión 2.0, se centra en desarrollar un Producto Mínimo Viable (PMV) enfocado en investigación y educación, antes de pasar a la siguiente fase y ser lanzado al mercado. La oportunidad fue convertir los requerimientos de los médicos en funciones y características de la aplicación tanto en el frontend como en el backend, abordando los desafíos clave y considerando las mejores prácticas de desarrollo.

3.2 Propuesta de desarrollo

En el desarrollo de Proximity 2.0 se pretendió realizar mejoras a una prueba de concepto realizado anteriormente en su versión 1.0. Para lograrlo, se decidió crear una plataforma web que permita la interconexión con diferentes repositorios de datos médicos pertinentes y cumpla con los requerimientos de funcionalidad y diseño establecidos por el equipo médico. Para ello se iteraron, en conjunto a los expertos del área de salud, diversos bosquejos.

Luego de obtener una maqueta de la interfaz gráfica validada por los médicos, se pasó a la elección del stack de tecnologías del frontend necesarias para comenzar el desarrollo de la aplicación. Para esto se investigó, en primer lugar, que tecnologías utilizaba Proximity en su versión 1.0, para así reutilizar o mejorar lo ya existente. Sin embargo, para obtener un resultado más fiel a lo conversado con los médicos, se optó por empezar un desarrollo desde cero, con tecnologías seleccionadas con base en lo que se utiliza en el mercado actual y que además manejan los integrantes del grupo PMM.

En cuanto al backend, el foco fue encontrar una arquitectura que sea compatible con el modelo de inteligencia artificial, debido a que dicho modelo se incrustó en el backend. Además, se buscó garantizar la escalabilidad, seguridad y eficiencia del sistema, permitiendo un acceso rápido a los datos médicos relevantes y facilitando el trabajo de los profesionales de la salud. Es por esto que se utilizó el mismo lenguaje con el que se realizó el modelo de inteligencia artificial, y un framework ligero, potente y escalable para qué brinda microservicios tipo API REST. Además, se añadió una capa de MLOps que se encarga de mantener el correcto flujo de datos y la posibilidad de integración de nuevos modelos en un futuro.

3.3 Definición de tecnologías médicas utilizadas

A lo largo del proyecto, se emplearon diversas tecnologías fundamentales en el ámbito médico, cruciales para comprender la propuesta de solución. Estas tecnologías incluyen:

- **Archivos DICOM:** Estándar internacional para el manejo, almacenamiento, impresión y transmisión de información en imágenes médicas. Los archivos DICOM no solo contienen la imagen médica en sí, sino también los metadatos esenciales para su interpretación, permitiendo la interoperabilidad entre dispositivos y sistemas en radiología y otras especialidades médicas.
- **PACS (Sistema de Comunicación y Archivo de Imágenes):** Utilizado mayoritariamente en servicios de radiología, facilita el almacenamiento y la comunicación de imágenes médicas mediante el estándar DICOM. Ejemplos de implementaciones incluyen Orthanc

[21], un servidor de código abierto diseñado para gestionar y compartir estudios radiológicos de manera eficiente y segura, integrándose fácilmente con otros sistemas de salud a través de APIs.

- **HAPI FHIR (Fast Healthcare Interoperability Resources):** Servidor de código abierto que almacena datos médicos conforme al estándar HL7/FHIR.
- **HL7/FHIR:** HL7 (Health Level Seven) es una organización internacional que desarrolla estándares para la interoperabilidad en el ámbito de la salud. FHIR (Fast Healthcare Interoperability Resources) es uno de estos estándares, diseñado para facilitar el intercambio de datos de salud de manera rápida y eficiente mediante APIs modernas y basadas en recursos.

Estas tecnologías juegan un papel fundamental en la mejora de la interoperabilidad y eficiencia dentro de los entornos clínicos, garantizando un manejo efectivo de la información médica.

Desarrollo de la solución

4.1 Diseño de solución

4.1.1 Situación inicial y estado de oportunidad

El diseño de la interfaz inicial de Proximity v1.0 (ver Figura 4.1) es simple. El usuario (médico) ingresa su credencial y la cantidad de nodos para luego visualizar, (ver Figura 4.2) los distintos posibles diagnósticos junto con información del examen y del paciente.

El estado de oportunidad surgió cuando se unió el desarrollo inicial para la validación de concepto y la necesidad de obtener la validación por parte de los médicos. Durante la versión 2.0 de la aplicación, los médicos tuvieron la oportunidad de expresar sus necesidades y expectativas, las cuales se convirtieron en requerimientos esenciales para el desarrollo de una solución que se ajuste a sus demandas. Para resolver esto, se estableció un diálogo constante y una estrecha colaboración entre los especialistas de la salud y

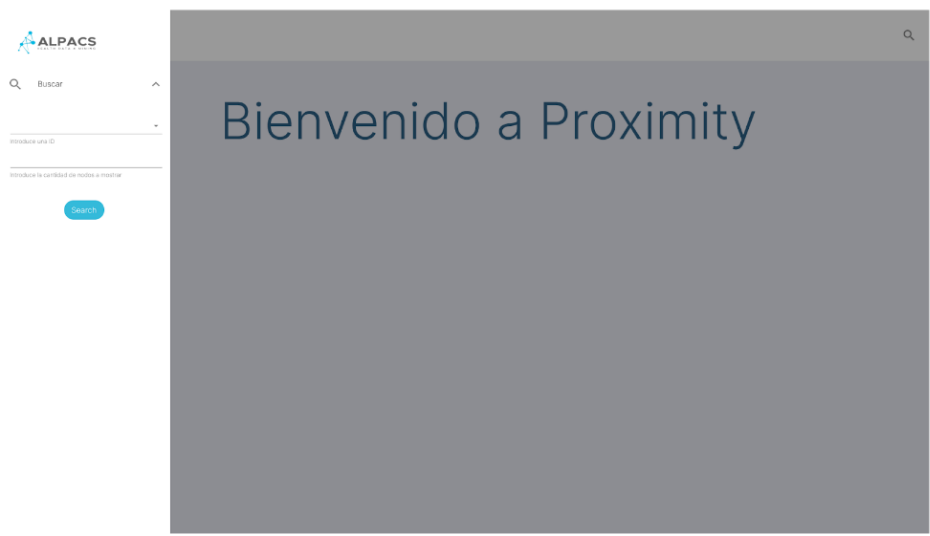


FIGURA 4.1: Inicio Proximity v1.0

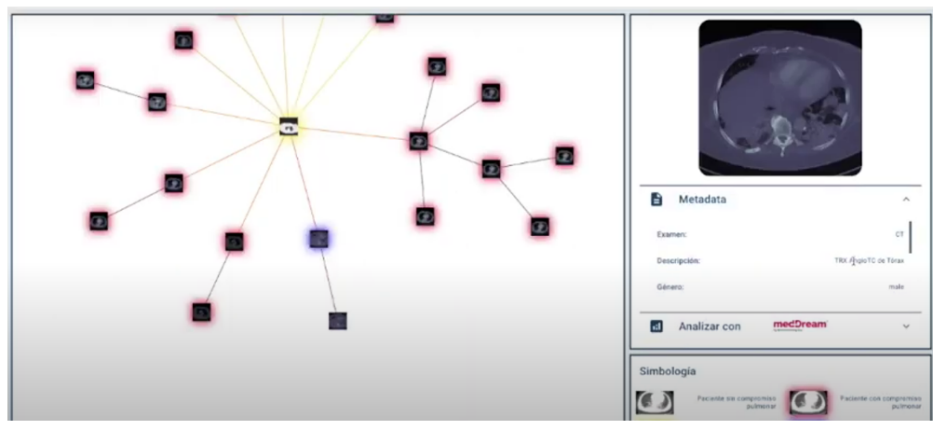


FIGURA 4.2: Resultados Proximity v1.0

los desarrolladores. El objetivo fue lograr un producto final que satisfaga de manera efectiva las necesidades y expectativas de los médicos, a través de la iteración y mejora continua del desarrollo de la aplicación.

4.1.2 Metodología utilizada

El objetivo principal fue desarrollar un Producto Mínimo Viable (PMV) que permita ingresar a la fase de validación con los usuarios. Para lograr esto, se establecieron objetivos específicos basados en los requerimientos mencionados por los stakeholders. El proceso de desarrollo del PMV se llevó a cabo en estrecha colaboración con el equipo médico, quienes desempeñaron un papel clave en la identificación de las características y funcionalidades más importantes para la solución. Mediante la metodología Diseño Centrado en el Usuario (DCU) [22], se busca satisfacer sus necesidades y expectativas de manera efectiva, garantizando que el producto final cumpla con los estándares de calidad y funcionalidad requeridos. Además, se utiliza el Método Proyectual [23], destacando la importancia del enfoque iterativo y multidisciplinario, promoviendo la experimentación, la colaboración y la retroalimentación constante. A través de esta metodología, se buscó generar soluciones innovadoras y efectivas, teniendo en cuenta tanto las necesidades de los usuarios como los aspectos técnicos y estéticos del diseño.

4.1.3 Etapas sistemáticas

4.1.3.1 Etapas sistemáticas para abordar el desafío.

Las etapas sistemáticas y estructuradas para abordar el desafío de desarrollar la aplicación fueron:

1. **Investigación y comprensión del problema:** se realizó la investigación comenzando con la documentación proporcionada por Alpacs, comprendiendo el contexto y el problema. Luego, en reuniones con el cliente y usuarios, se obtuvieron requerimientos. Finalmente, se realizó una investigación de soluciones similares para comenzar a identificar patrones de estandarización.
2. **Definición de objetivos y requerimientos:** El objetivo principal fue desarrollar la interfaz del usuario, la lógica del software y el flujo de datos, cumpliendo con los requerimientos principales de los médicos y llegando así al producto mínimo viable.
3. **Ideación y generación de conceptos:** En esta etapa, se generaron ideas y conceptos para la aplicación, considerando las necesidades y deseos de los radiólogos. Se utilizaron técnicas de pensamiento creativo, como el brainstorming y los bocetos rápidos, para explorar diferentes enfoques y posibilidades de diseño.
4. **Diseño y desarrollo iterativo:** A partir de los conceptos generados, se comenzó a diseñar la interfaz de usuario, la arquitectura de la información y la unión con los algoritmos de inteligencia artificial. Se realizaron iteraciones del diseño, prototipado y pruebas con los médicos para evaluar la usabilidad, la eficacia y la experiencia general de la aplicación.
5. **Evaluación y validación:** se efectuaron evaluaciones y pruebas para validar la aplicación. Esto implicó realizar pruebas de usabilidad con radiólogos, recopilar su feedback y realizar ajustes con base en sus comentarios.

6. **Implementación y despliegue:** Una vez que la aplicación fue validada, se procedió a su implementación y despliegue en entornos reales. Se realizó la integración con los sistemas y equipos existentes en los centros médicos y se brindó soporte técnico durante el proceso de implementación.
7. **Monitoreo y mejora continua:** Luego de que la aplicación está en funcionamiento, se realizó un monitoreo continuo para evaluar su rendimiento y recopilar feedback de los médicos. Se realizaron mejoras iterativas con base en la retroalimentación recibida y a los avances realizados por el sistema de MLOps.

Estas etapas permitieron abordar el problema de manera sistemática y centrada en el usuario, asegurando que la aplicación cumpla con las necesidades y expectativas de los radiólogos. La metodología de Diseño Proyectual y DCU garantizó que la solución final sea efectiva y funcional.

4.1.3.2 Identificación de recursos.

- **Recursos:**

- **Datos clínicos:** Acceso a repositorios de datos clínicos anonimizados.
- **Equipo multidisciplinario:** Apoyo informático, clínico, investigadores, expertos en el área de Data Science, expertos en el área de Infraestructura, interoperabilidad médica y radiología.
- **Financiamiento:** Adquisición de recursos necesarios para el desarrollo del proyecto.

- **Recursos humanos:**

- **Ingenieros:** Equipo de Data Science, Equipo de infraestructura, equipo de desarrollo e ingeniería de software, gerencia.
- **Médicos:** Usuarios.

4.1.4 Requisitos del Sistema

4.1.4.1 Requisitos Funcionales

- **RF1:** El entorno web, una vez confirmada las credenciales de acceso, permitirá al usuario consultar por algún caso de estudio según un ID específico.
- **RF2:** La plataforma debe estar interconectada con los servicios asociados a repositorios PACS y HAPI/FHIR para la recuperación de volúmenes 3D de tomografías torácicas solicitadas.

- **RF3:** Una vez obtenido el caso de estudio solicitado, la aplicación pasará el CT por la Red Neuronal para que este sea procesado e indexado encontrado así sus anomalías.
- **RF4:** La aplicación buscará y retornará los casos de estudio con diagnósticos disímiles pero con anomalías visuales similares.
- **RF5:** El software desplegará por pantalla el CT solicitado, además de los demás casos de estudio similares para permitir el diagnóstico diferencial por parte del profesional médico.
- **RF6:** Al analizar uno de los casos retornados, la plataforma deberá mostrar por pantalla toda la información médica técnica disponible sobre la tomografía escogida.
- **RF7:** Luego de montar el proyecto en un servidor y ponerlo a disposición del usuario final, se medirán y visualizarán métricas tanto para la aplicación web, como para el modelo.

4.1.4.2 Requisitos No Funcionales

- **RNF1:** Tras la consulta a la plataforma, esta no deberá tardar más de 1 minuto en presentar por pantalla la información pertinente.
- **RNF2:** La visualización de la información debe ser clara y de carácter técnico para el área médica.
- **RNF3:** La plataforma debe estar desplegada en la nube.
- **RNF4:** El software debe ser flexible y escalable a futuras funcionalidades aún no consideradas para esta versión.

4.1.5 Diseño de Arquitectura de Aplicación Web

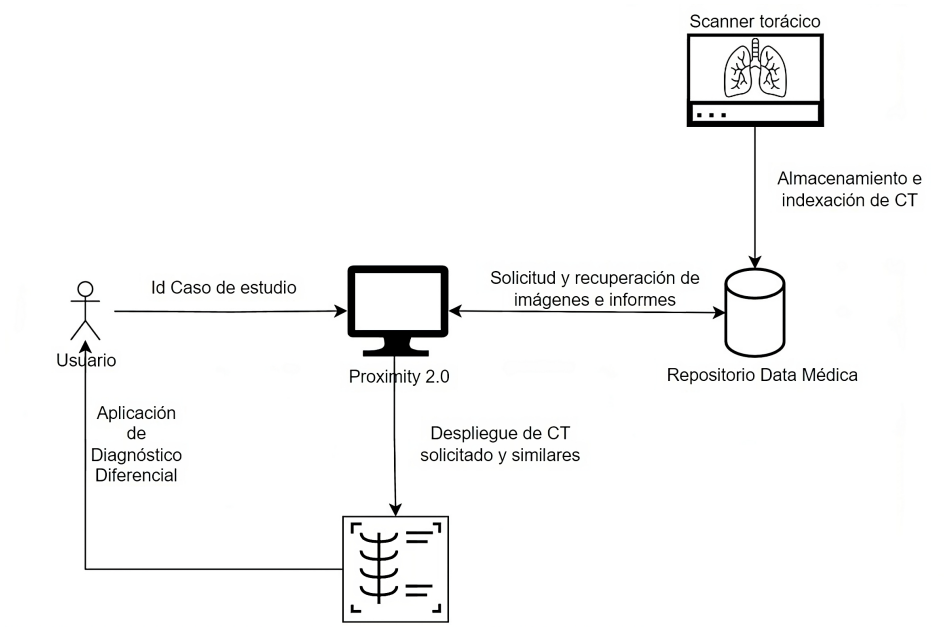


FIGURA 4.3: Diagrama de contexto

En las Figuras 4.3 y 4.4 se puede observar la estructura en la que se basó Proximity 1.0 y que además se tomó como base para realizar un acercamiento de Proximity 2.0. El pipeline es el siguiente:

1. Llega una query en forma de URI.
2. El backend consulta al repositorio PACS para buscar la imagen correspondiente a la URI.

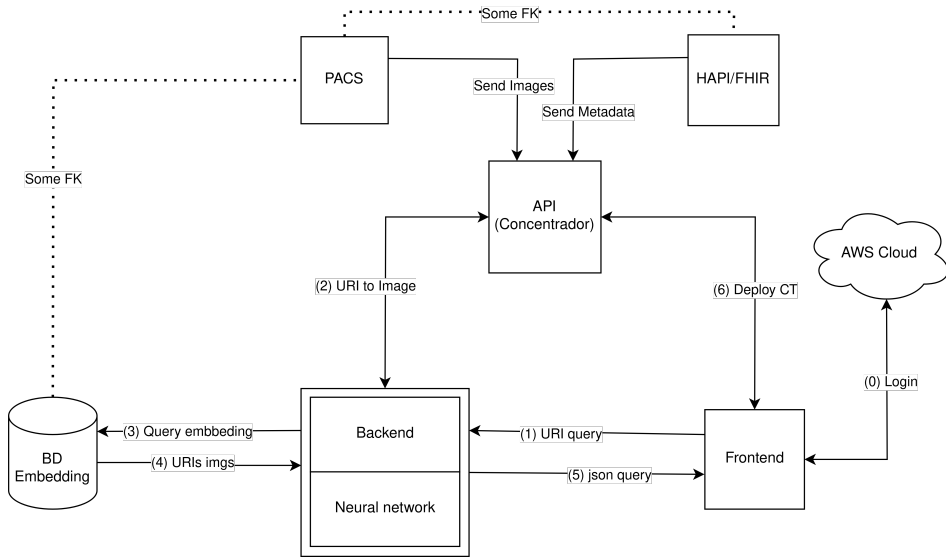


FIGURA 4.4: Diagrama de arquitectura de aplicación web

3. La query pasa por la red neuronal y se compara con los índices almacenados de los embeddings de las imágenes en el servidor PACS.
4. Como respuesta se envía una estructura con datos de los embeddings más cercanos a la query.
5. El backend envía la información al frontend.
6. El frontend con la información dada en la estructura, es capaz de recuperar las imágenes y metadata del paciente, en los servidores PACS y HAPI/FHIR respectivamente

4.1.6 Descripción de componentes de la aplicación web

4.1.6.1 Frontend

- **Propósito:** Permitir a los usuarios identificarse, buscar cortes transversales, mostrar su información de entendible, mostrar diagnósticos similares a este y por último mediante el uso de herramientas de visualización revisar estos datos.
- **Alcance:** El componente está pensado para ser una interfaz completa, incluyendo proceso de autenticación, búsqueda y edición de imágenes médicas.
- **Dependencias:** Depende de los siguientes módulos, **Backend** Para la consulta de diagnósticos similares, **Concentrador** para la recuperación de imágenes de Cortes transversales, **AWS** Para la autenticación de las cuentas institucionales.
- **Supuestos:** Se cuenta con acceso a las cuentas institucionales y está montado en la misma red que el módulo **Concentrador**
- **Restricciones:** las imágenes médicas de los cortes transversales se encuentran en series de imágenes con formato .dcm .png o .jpg. Los cuales son aptos para la visualización. la metadata recibida del corte transversal se encuentra en formato **JSON**, esto debido a que se trata de un formato ampliamente utilizado y apto para la aplicación
- **Estructura General:** El módulo primeramente autentifica al usuario, luego presenta a este distintos cortes transversales, ya sea mediante la búsqueda por ID, mediante el uso de filtros o a través imágenes pre visualizadores en la interfaz principal, luego de seleccionar el

corte transversal la aplicación muestra al usuario distintos cortes transversales similares, destacando información relevante presente en la metadata de estos y por último la aplicación permite al usuario manipular estos con tal de compararlos exactitud.

4.1.6.2 Backend/Neural Network

- **Propósito:** Retornar información de diagnósticos similares.
- **Alcance:** El módulo se propone como un intermedio para la consulta de diagnósticos diferenciales según proximidad.
- **Dependencias:** Depende de los módulos **Frontend** Para obtener el identificador del corte transversal a procesar y **Concentrador** para obtener las imágenes de estas, para luego enviar esta información al **BD-Embedding**.
- **Supuestos:** La base **BD-Embedding** se encuentra siempre activa, los módulos de **Concentrador** y **FrontEnd** están activos al momento de realizar la consulta.
- **Restricciones:** La URI ingresada debe ser el número de acceso único de cada estudio, también llamado Accession Number en la metadata de los archivos DICOM
- **Estructura General:** El módulo de Backend, recibe una petición del **FrontEnd**, con tal de obtener los diagnósticos similares de una imagen. Dicha imagen se busca en el **Concentrador**, pasa por la red neuronal y se convierte en un vector de características. El vector es comparado con los demás vectores indexados en la **BD-Embedding** y finalmente, esta retorna los identificadores de las imágenes similares. Por último, el

módulo consulta por las imágenes respectivas de los identificadores al **Concentrador** y estas imágenes son enviadas devuelta al **FrontEnd**

4.1.6.3 AWS

- **Propósito:** Módulo externo, asociado netamente a la autenticación.
- **Alcance:** Pensado netamente en la autenticación web, se trata de una API externa que recibe las credenciales enviadas a la página.
- **Dependencias:** Este módulo depende únicamente del módulo **FrontEnd**.
- **Supuestos:** Se encuentra operativo, monitoreado por Equipo de Infraestructura.
- **Restricciones:** Las credenciales son enviadas en un formato JSON, encriptado según el estándar Data Encryption Standard (DES).
- **Estructura General:** Las credenciales son ingresadas a la aplicación de la nube de Amazon, luego dependiendo del resultado de esta, se retorna un código 200 de respuesta y dependiendo si la autenticación fue exitosa se envía el resultado en conjunto a la respuesta a la página.

4.1.6.4 Concentrador

- **Propósito:** interconectar los repositorios de datos e imágenes con la plataforma.
- **Alcance:** El módulo abarca las consultas dentro de la aplicación: imágenes médicas, metadatos y reportes.

- **Dependencias:** depende únicamente de los módulos externos **HAPI/FHIR** y **PACS**.
- **Supuestos:** Los repositorios PACS y HAPI/FHIR se encuentran operativos en todo momento.
- **Restricciones:** La consulta debe ser con la URI (Accession Number) para la extracción de los datos. Además, la URL retornada difiere si se debe utilizar solo las imágenes o las imágenes con metadatos incluidos.
- **Estructura General:** El módulo tiene un servidor operativo dentro de la red local para realizar consulta de imágenes, a la vez posee los permisos necesarios para interconectarse con los repositorios PACS y HAPI/FHIR.

4.1.7 Descripción de componentes externos al desarrollo de la aplicación web

- **BD-Embedding:** Base de datos creada a partir de las representaciones numéricas de las imágenes almacenadas en Orthanc, cuya función es poder realizar comparaciones y entregar los nombres de los CTs más cercanos a la query ingresada. En este caso, la Base de datos opera con el espacio latente de la imagen (llamada Pixel Array en la metadata), de cada archivo DICOM
- **HAPI/FHIR:** Servidor Open Source que permite el almacenamiento de datos médicos. Permite el almacenamiento de los reportes clínicos anonimizados provenientes del HCUCH
- **PACS (Picture Archiving and Communication System):** Sistemas estándares para el almacenamiento y comunicación de imágenes

médicas. Se utilizó para guardar todos los estudios anonimizados provenientes del HCUCH

4.1.8 Diseño de Interfaces

4.1.8.1 Modelo de navegación

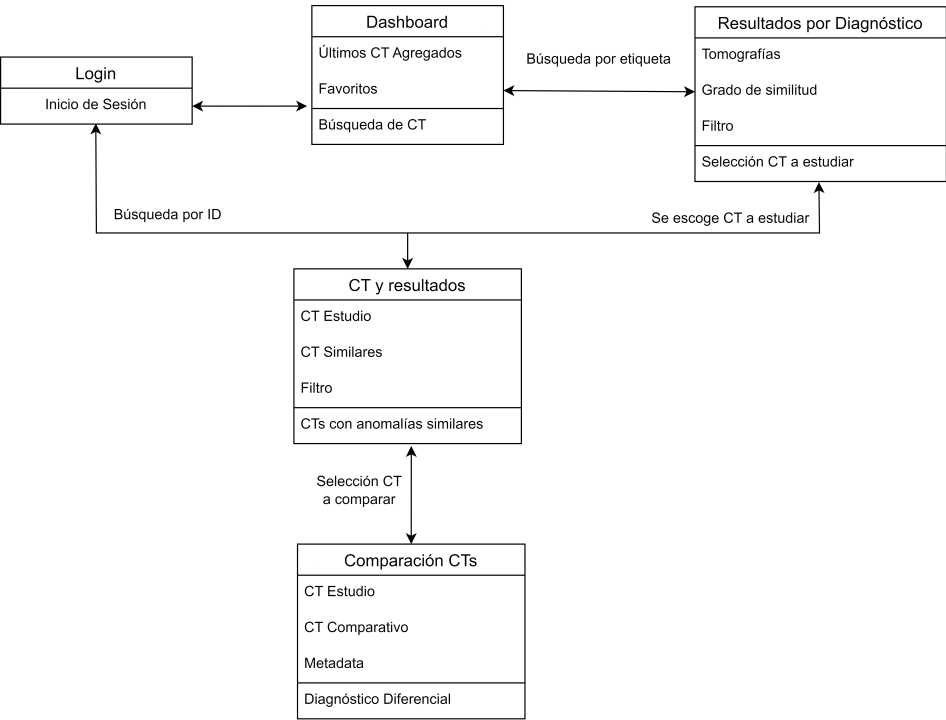


FIGURA 4.5: Modelo de Navegación

El modelo de navegación, presente en la Figura 4.5, comienza con un módulo de Inicio de sesión, tras el cual se pasa a un Dashboard donde el usuario puede

hacer el proceso de Búsqueda de CT, ya sea por ID o por etiquetas. Luego, se le presenta la tomografía seleccionada junto con los CT con similitudes resultantes. Al elegir uno de estos se da paso a la última vista, donde se puede hacer la comparación entre los casos de estudio.

4.1.8.2 Paleta de colores

En el proceso de selección de colores destinada al sector médico, el enfoque se centró en la creación de una paleta que se adapte tanto al uso diurno como nocturno. Esta elección resulta fundamental, especialmente en el ámbito profesional de la aplicación médica, ya que nuestro objetivo principal es dirigir la atención del médico hacia la comparación de imágenes para diagnóstico diferencial. Para lograrlo, se ha diseñado una que prioriza la neutralidad, evitando cualquier tipo de ruido visual que pueda interferir con el objetivo de mejorar el contraste y facilitar la comparación de imágenes de tomografías. En este proceso de selección, también se ha tenido en cuenta la identidad de la marca, asegurándonos de que los colores elegidos estén en línea con el logo del grupo Alpacs y Proximity.

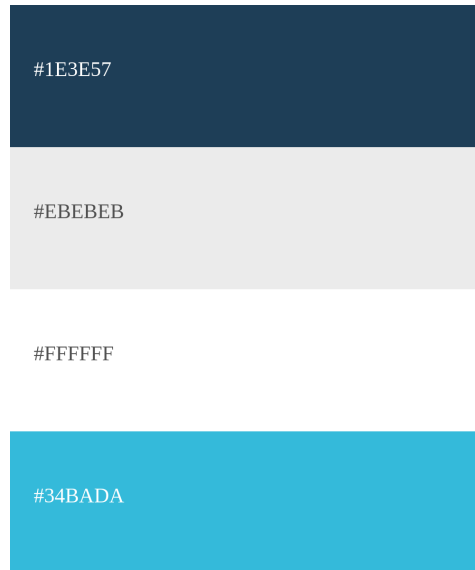


FIGURA 4.6: Paleta de colores primarios

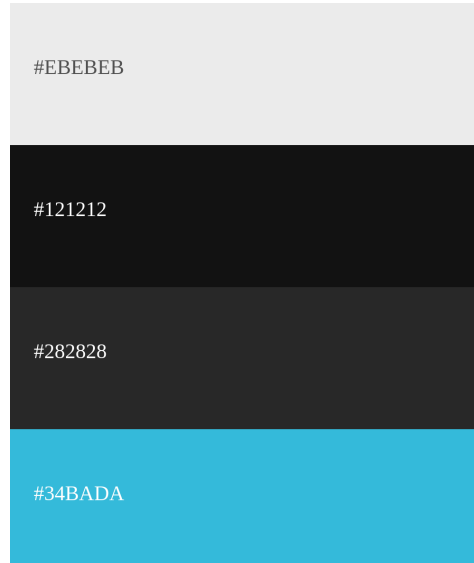


FIGURA 4.7: Paleta versión nocturna

En el proceso de elección de colores, se siguieron las pautas de la identidad de Proximity, enfocando particularmente en los tonos azules como se ve en las Figuras 4.7 4.6. Esta elección fue respaldada por consideraciones clave para alcanzar los objetivos.

4.1.8.3 Diseño de Interfaces Usuarías

Para el diseño del primer prototipo de Proximity 2.0 se inició un proceso de estudio tecnológico, investigando aplicaciones médicas usadas en la actualidad como AgendaPro, Clinic Cloud, y Nubimed, además de las reuniones de brainstorming en las que participaron ingenieros y médicos. Se decidió mantener la paleta de colores presentada en la versión 1.0 acentuando así la marca Proximity y para su construcción hacemos uso de la herramienta

de generación de prototipos Figma, la cual permite crear prototipos de interfaz sin la necesidad de programar. La finalidad de esta maqueta fue la validación de requerimientos captados en entrevistas con médicos mediante un proceso de Testing de Papel [24]. Dicha metodología consiste en entrevistas individuales con el fin de evitar influenciar a los demás participantes con las respuestas proporcionadas previamente. En estas entrevistas, al médico se le otorgó un contexto para luego asignarle tareas a resolver en el prototipo. El médico debió realizar un Think Aloud mientras interactuaba con la aplicación, es decir, transmitió en voz alta sus apreciaciones, confusiones, disgustos o cualquier otro aspecto relevante en la realización de las tareas asignadas, con el fin de que el equipo entienda en tiempo real lo que debe ser mejorado o se encuentra dentro de las directrices esperadas. Como resultado se logró obtener las interfaces mostradas en las Figuras 4.8, 4.9, 4.10, 4.11 4.12



The image shows a login interface for the PRX Proximity system. At the top center is the logo, which consists of the letters 'PRX' in a bold, black, sans-serif font, followed by a stylized graphic of three connected nodes (two blue, one black) and the word 'PROXIMITY' in a smaller, blue, sans-serif font. Below the logo, there are two input fields. The first is labeled 'Usuario' and contains the text 'Dra.Doctorina'. The second is labeled 'Contraseña' and contains a series of asterisks '*****'. Below these fields, there are two links: 'Recuérdame' (with a small square icon) and 'Recuperar contraseña'. A large blue button with the text 'Iniciar sesión' is positioned below the links. In the bottom left corner, the version 'V 2.2' is displayed. In the bottom right corner, there is a logo for 'ALPACS' with the tagline 'HEALTH DATA & MINING'.

PRX
PROXIMITY

Usuario
Dra.Doctorina

Contraseña

☐ Recuérdame [Recuperar contraseña](#)

Iniciar sesión

V 2.2

ALPACS
HEALTH DATA & MINING

FIGURA 4.8: Vista inicio de sesión

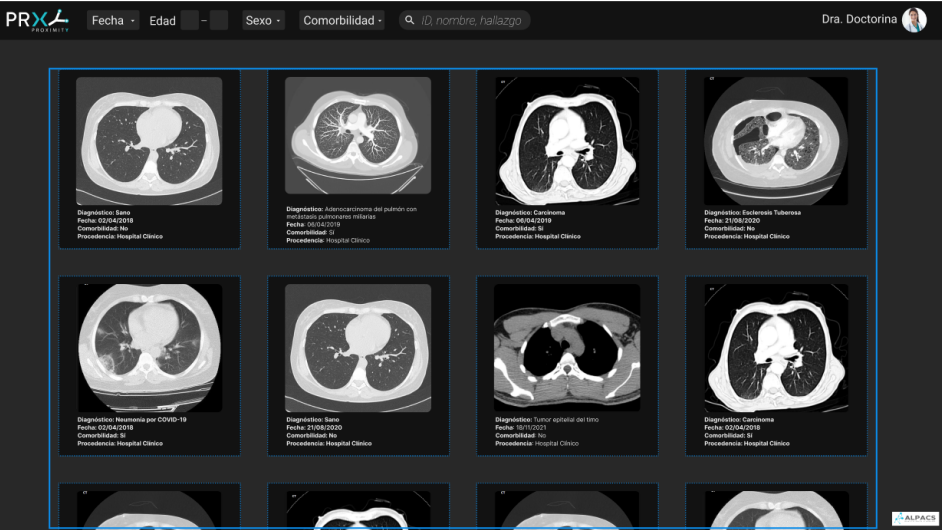


FIGURA 4.9: Vista principal (modo nocturno)

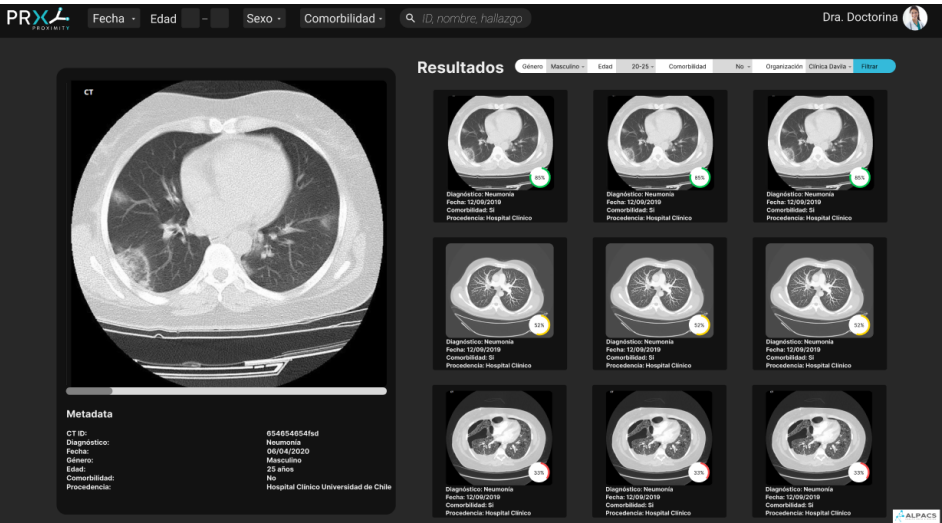


FIGURA 4.10: Vista resultados de consulta

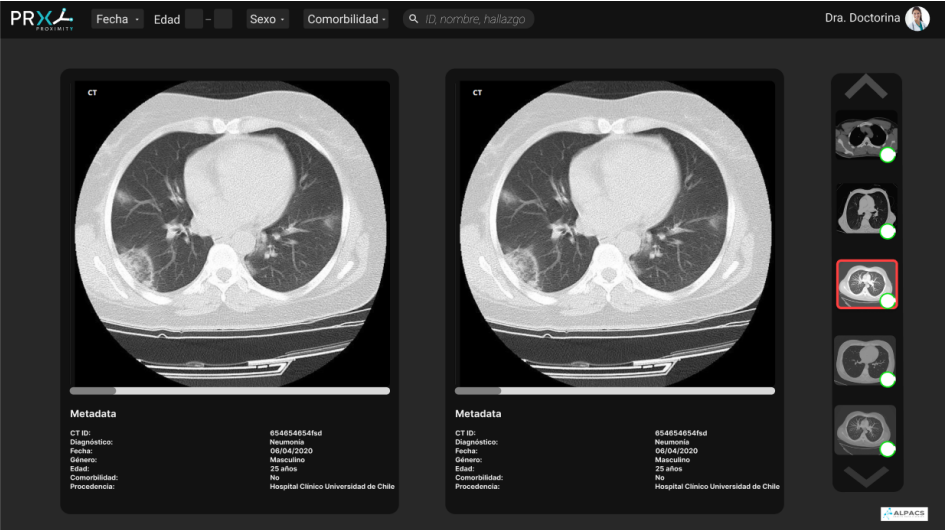


FIGURA 4.11: Vista de comparación (modo nocturno)

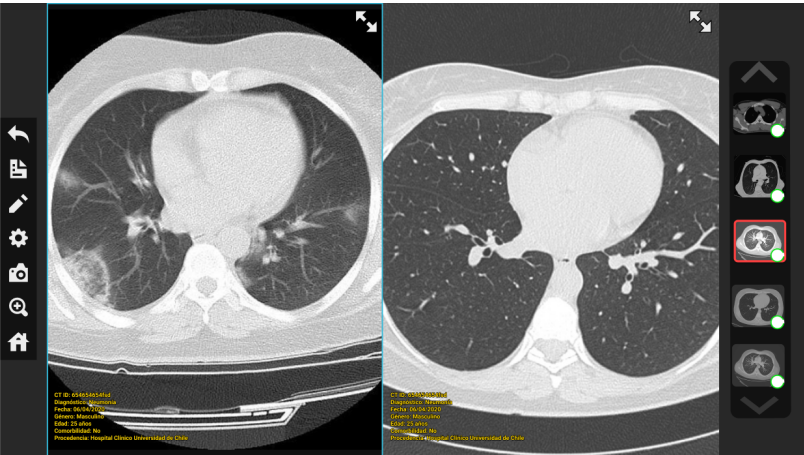


FIGURA 4.12: Vista de comparación detallada (modo nocturno)

4.1.9 Diseño MLOps

El área de MLOps se centra principalmente en tres puntos principales de la vida de un modelo de machine learning que sale a producción: procesamiento y curación de datos, entrenamiento y optimización del modelo y monitoreo del estado del modelo en producción, como se muestra en la Figura 4.13.

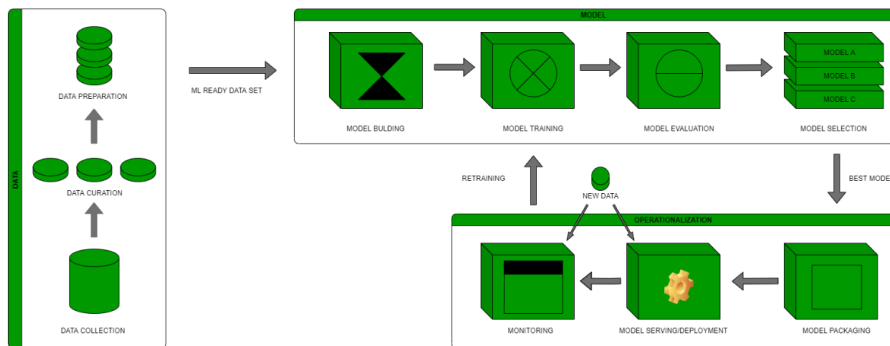


FIGURA 4.13: MLOps pipeline [1]

En el proyecto desarrollado con el grupo de memorias multidisciplinarias, la tarea principal de esta memoria, fue el procesamiento y curación de datos, ya que los datos provenientes del PACS, no están listos para su uso, es necesario realizar una serie de pasos previos y así dejarlos preparados para el consumo de la aplicación web. Con el fin de lograr una línea de trabajo eficiente, se propuso un primer pipeline a medida de los servidores y tecnologías con las que se trabaja. Los datos son consumidos desde el servidor Orthanc, servidor que funciona como PACS.

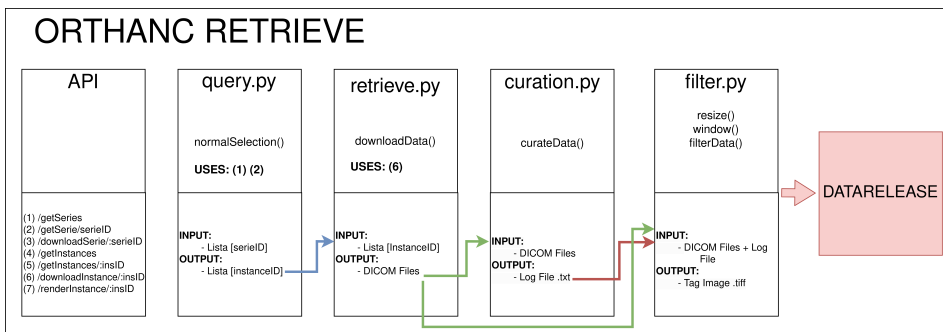


FIGURA 4.14: Pipeline para recuperación de datos

En la Figura 4.14, se creó un pipeline con el objetivo de obtener y preparar datos para entrenar o probar modelos de inteligencia artificial mediante la disponibilización de una API que permita la descarga de dichos datos. El proceso comenzaba consumiendo datos desde una API de Orthanc, la cual ofrecía varios endpoints útiles, aunque no todos fueron utilizados en este flujo específico.

Para gestionar esta interacción, se desarrolló el archivo *query.py*, el cual generaba diferentes salidas según se requirieran datos para pruebas (retornando todas las URLs de las instancias disponibles) o para entrenamiento. En este último caso, se aplicaba una selección basada en una distribución normal (Figura 4.26) con el parámetro "k", determinando cuántas instancias se seleccionaban por serie, resultando en una lista de URLs de descarga correspondientes.

Posteriormente, el archivo *retrieve.py* se encargaba de descargar los datos seleccionados. Aunque este paso preparaba los datos para el consumo del frontend, se recomendaba realizar una curación previa para evitar posibles problemas con la biblioteca encargada de leer los archivos DICOM.

Para este propósito, el archivo *curation.py* verificaba la integridad de la información, asegurando la corrección de las imágenes adjuntas en los archivos DICOM y verificando la metadata relevante, como el tipo de estudio analizado. Como resultado, se generaban dos archivos de registro ("log files") que indicaban las rutas a los archivos DICOM locales, clasificados según su integridad.

Finalmente, en *filter.py* se implementaban funciones para permitir al usuario ajustar la data, como el tamaño de las imágenes y los cambios de contraste, produciendo así un conjunto de datos de entrenamiento denominado "DataRelease".

Este pipeline cubrió la primera fase del flujo de MLOps. Para la etapa de entrenamiento, se utilizó la versión 1.0 del modelo neuronal de Proximity. Es crucial mencionar las limitaciones de este modelo: primero, solo aceptaba una instancia en lugar de utilizar el estudio completo; segundo, sus embeddings eran notablemente grandes, por lo que se aplicó PCA para reducir el número de componentes y la memoria utilizada; tercero, este modelo ya no se utiliza en el proyecto Proximity.

El enfoque principal de este pipeline fue diseñar un proceso que pudiera replicarse con mínimos ajustes en un entorno controlado, posiblemente utilizando herramientas como Docker, para ser aplicado a otros modelos de IA.

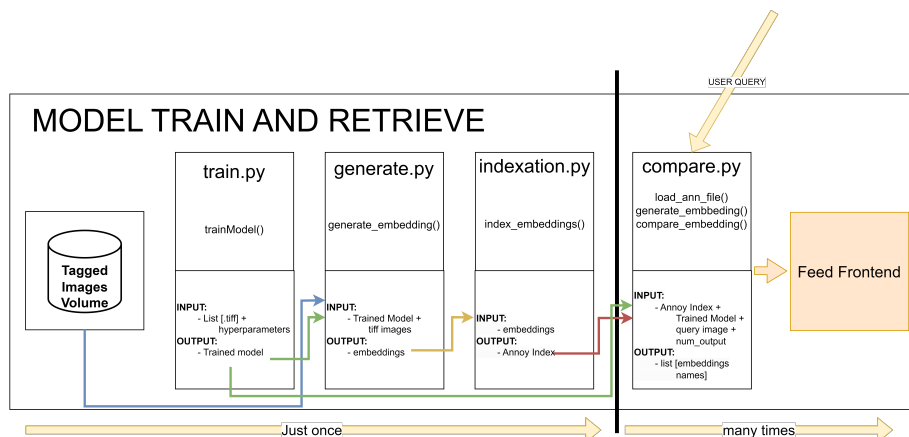


FIGURA 4.15: Entrenamiento y recuperación de resultados.

En la Figura 4.15 se muestra el flujo de datos desde el entrenamiento del modelo hasta su utilización en el proceso de diagnóstico diferencial. El objetivo principal de este pipeline fue facilitar la integración de nuevos modelos de inteligencia artificial en el proyecto. Solo es necesario considerar las entradas y salidas correspondientes.

Para comenzar, se creó el archivo *train.py*, donde se incorporaba el nuevo modelo que sería entrenado. Este proceso utilizaba el conjunto de datos previamente liberado (DataRelease) junto con los hiperparámetros adecuados, generando como salida el modelo entrenado. Una vez completado el entrenamiento, el archivo *generate.py* se encargaba de producir los embeddings del modelo y construir lo que se denominaba en la arquitectura como la base de datos de embeddings (BD Embeddings).

Posteriormente, en *indexation.py*, se utilizaron los embeddings con la biblioteca ANNOY para indexar las distancias en el espacio euclidiano y almacenarlas en un archivo. Esto permitía que, al generar un embedding

de una consulta en *compare.py*, se pudiera cargar el índice de ANNOY y determinar qué instancias estaban más cercanas a la consulta realizada. Como resultado final, estas distancias, junto con los nombres de los archivos recuperados, se enviaban al backend para su procesamiento y uso conforme fuera necesario.

Algunas ideas de la lógica de la Figura 4.15 estaban basadas en el proceso utilizado en Proximity 1.0. Por ejemplo, la biblioteca ANNOY [25] podía ser sustituida por otros métodos de indexación como MRPT [26] o búsqueda exacta, que fueron utilizados anteriormente, pero no lograron resultados tan eficientes como ANNOY. Del mismo modo, el comportamiento de *compare.py* se debe adaptar ligeramente según el funcionamiento de los nuevos modelos desarrollados, dependiendo del orden en el que se indexan.

4.2 Implementación de la solución

Luego de recolectar la información necesaria y realizar las iteraciones pertinentes, se dio paso a la implementación de la aplicación. Lo primero fue definir el stack de tecnología a utilizar en el proyecto. Para esto se separaron en tres grandes aristas del proyecto: MLOps, Desarrollo de software y Red Neuronal* (ver Figura 4.16)

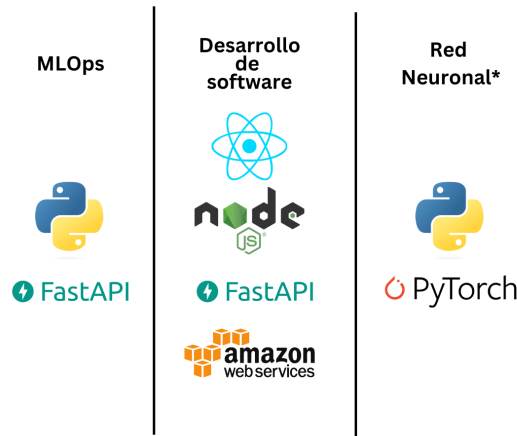


FIGURA 4.16: Stack de tecnologías

*Basado en la versión 1.0 del modelo Proximity[27]

4.2.1 Desarrollo de Software

4.2.1.1 Vista Login PMV

Para la implementación de la aplicación, cada uno de los integrantes del grupo de memoria multidisciplinarias trabajó de manera local. Se utilizó Bitbucket para mantener el control de versiones y poner a disposición el código.

Se comenzó por desarrollar la vista de inicio de sesión (ver Figura 4.17)



FIGURA 4.17: Vista login PMV

En esta vista, se podía ingresar el usuario y la contraseña para acceder a la cuenta. Además, contaba con una opción para mantener la sesión abierta en caso de cerrar el navegador y un botón para recuperar la contraseña en caso de olvidarla. La validación y el uso de tokens se realizaban a través de un servidor externo, AWS, proporcionado por el equipo de infraestructura.

4.2.1.2 Vista dashboard PMV

Después de iniciar sesión, el usuario se encontraba con la Figura 4.18. En esta vista se mostraban las últimas series agregadas y una barra de navegación. Había dos flujos principales en esta vista: el primero permitía filtrar por diferentes parámetros, como el número de resultados, edad, comorbilidad o algún hallazgo en particular, funcionando solo como filtro de imágenes en la base de datos médicas. El segundo flujo permitía al usuario hacer clic en uno de los últimos CTs agregados para realizar una recuperación de los hallazgos más cercanos a la serie consultada.



N° resultados: 9

Edad:

Comorbilidad: No

Hallazgos...

Buscar CT

Dr. Doctora Doctorina 

Últimos CTs Agregados					
FECHA	DIAGNÓSTICO	PROCEDENCIA	GÉNERO	COMORBILIDAD	HALLAZGO
23/08/2023	Cáncer	Hospital Clínico	Femenino	Sí	Silueta
18/08/2023	Neumonía	Hospital Clínico	Masculino	No	Condensación
20/08/2023	Tuberculosis	Hospital Clínico	Masculino	Sí	Condensación
19/08/2023	Covid-19	Hospital Clínico	Femenino	Sí	Silueta
21/08/2023	Cáncer	Hospital Clínico	Masculino	Sí	Condensación
19/08/2023	Covid-19	Hospital Clínico	Femenino	No	Broncograma aéreo
20/08/2023	Tuberculosis	Hospital Clínico	Masculino	Sí	Condensación
19/08/2023	Cáncer	Hospital Clínico	Femenino	Sí	Silueta
21/08/2023	Covid-19	Hospital Clínico	Masculino	Sí	Condensación
19/08/2023	Cáncer	Hospital Clínico	Femenino	No	Broncograma aéreo
20/08/2023	Tuberculosis	Hospital Clínico	Masculino	Sí	Condensación
19/08/2023	Cáncer	Hospital Clínico	Femenino	Sí	Silueta
21/08/2023	Neumonía	Hospital Clínico	Masculino	Sí	Condensación
19/08/2023	Cáncer	Hospital Clínico	Femenino	No	Broncograma aéreo

FIGURA 4.18: Vista dashboard PMV

4.2.1.3 Vista búsqueda PMV

Esta vista era simplemente un visualizador de la base de datos de imágenes médicas. Permitía revisar los resultados buscados a través de la barra de navegación y hacer clic en uno de ellos para utilizar la inteligencia artificial,

con el fin de encontrar los resultados cuyos hallazgos fueran similares al CT consultado. (ver Figura 4.19).

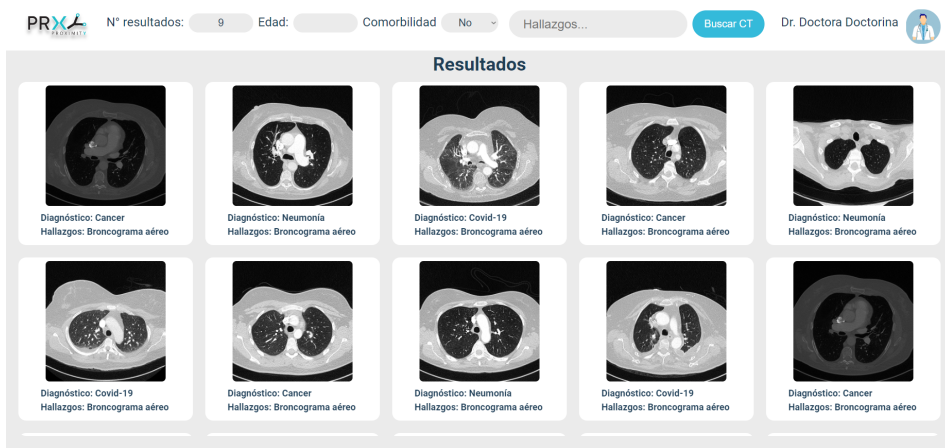


FIGURA 4.19: Vista búsqueda PMV

4.2.1.4 Vista recuperación y comparación PMV

Al momento de realizar una consulta, el modelo de inteligencia artificial era capaz de retornar 9 CTs con hallazgos similares al CT consultado (ver Figura 4.20). El usuario podía hacer clic en uno de los resultados y pasar a una vista comparativa (ver Figura 4.21). En esta última, en la esquina superior de cada visualizador, se podía obtener el informe asociado al estudio. En la esquina inferior izquierda de cada imagen observada, se encontraba la metadata de cada serie. En el centro de la vista, se podían utilizar herramientas para mover las imágenes, hacer zoom, medir o limpiar todas las anotaciones. También se podían agregar filtros útiles para los usuarios médicos.

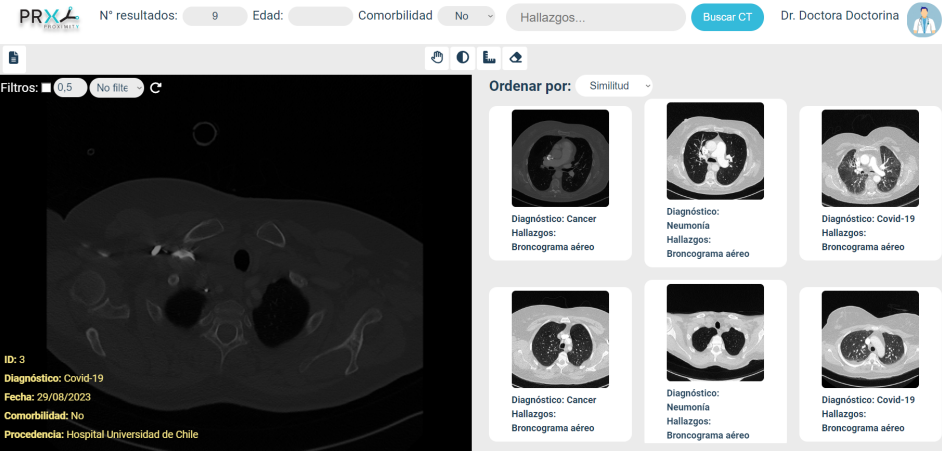


FIGURA 4.20: Vista recuperación PMV

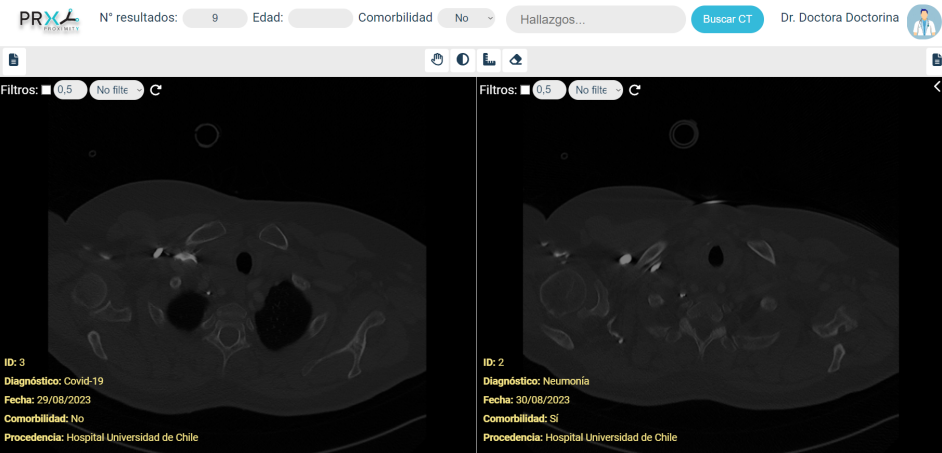


FIGURA 4.21: Vista comparación PMV

4.2.1.5 Backend

Para la realización del backend se utilizó la biblioteca FastAPI de Python, ya que es un framework simple de utilizar y escalable. Su finalidad es crear una API capaz de aplicar una renderización virtual, es decir, que solo renderiza los cambios cuando se necesite en la interfaz de usuario. La segunda razón por la cual es utilizada esta biblioteca, es la posibilidad de integrar de un modelo de inteligencia artificial escrito en Python, que es capaz de realizar recuperación de metadata de las imágenes más cercanas a la imagen de consulta.

Si bien, en esta versión se probó el modelo 1.0 de inteligencia artificial de Proximity, el cual trabajó solo con una instancia, puede ser escalado a una nueva versión que utilice el volumen de instancias completo, solo cambiando la forma en que recupera la información.

En la Figura 4.22, se muestra el endpoint creado para obtener una lista con la información de un número a determinar por el usuario de instancias. En este caso solo se devolvió el nombre de las instancias cercanas en el espacio euclidiano, pero puede retornar información necesaria para identificar las instancias en otros servidores. Dicha información, es utilizada para obtener imágenes en ORTHANC y metadata en HAPI/FHIR, que más adelante fueron desplegadas por el frontend.

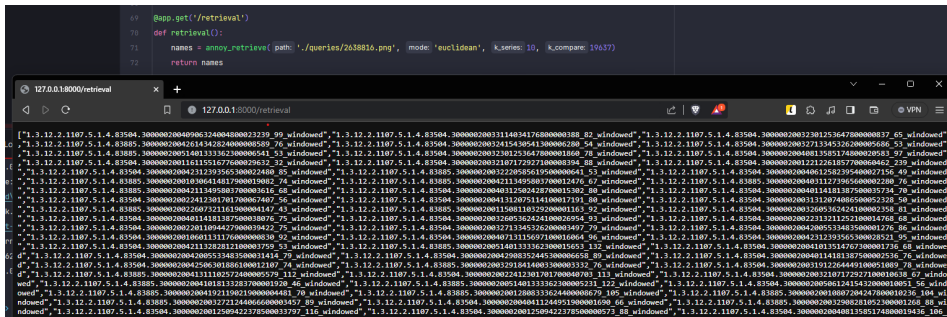


FIGURA 4.22: Endpoint de FastAPI para recuperación de instancias.

4.2.2 MLOps

En el estado del arte del MLOps, se presentan tres partes principales del flujo: preprocesamiento de datos, entrenamiento y monitoreo. Para la solución implementada en la aplicación web, solamente se aplicó el preprocesamiento de datos, ya que las instancias mostradas en pantalla debían tener un orden específico y ser archivos íntegros para evitar errores en producción. Por otro lado, Entrenamiento y Monitoreo no pudieron ser probados ni validados.

El preprocesamiento de los datos se realizó en tres pasos: curación, selección y aplicación de filtros.

4.2.2.1 Datos

En el proyecto Proximity se trabajó con imágenes médicas de tipo tomografía computarizada en formato DICOM (*foo.dcm*). En este tipo de exámenes, se crea un mapa 3D con imágenes que simulan cortes transversales de una zona del paciente para estudiarlas. En la Figura 4.23, se ve una reconstrucción de tórax, tanto el modelo de inteligencia artificial, como el frontend consume datos de cortes axiales (imagen inferior derecha de la Figura 4.23). Además,

este tipo de imágenes médicas, están en escala de grises y sus píxeles, en vez de ir de 0 a 255 (2^8), van de 0 a 65536 (2^{16}) por lo que pesan el doble de una imagen común.

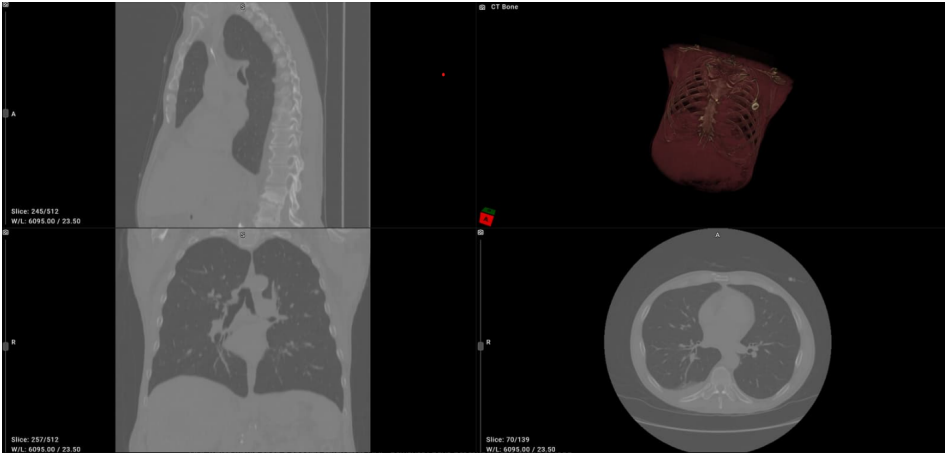


FIGURA 4.23: Reconstrucciones tomografía computarizada

4.2.2.2 Orden de datos dentro de servidor

Para lograr una búsqueda eficiente de datos en el servidor, se llevó a cabo un proceso interno de ordenamiento de datos (Figura 4.24), que a diferencia de la API mostrada en la Figura 4.14, no se disponibilizó mediante una API, sino que se ejecutaba manualmente dentro del servidor cada vez que llegaban nuevos datos. Inicialmente, los datos llegaban en lotes no ordenados y se almacenaban en carpetas con nombres predeterminados como inboundX, donde X es un número entero que incrementa con cada nuevo lote de datos, desde 1 hasta la enésima vez que se transmitían los datos. Dentro de

estas carpetas, se encontraban numerosos archivos DICOM cuyos nombres consistían en cadenas de caracteres sin información valiosa.

Utilizando la biblioteca Pydicom [28] en el módulo *sort_files.py*, fue posible analizar la metadata de cada archivo DICOM y organizarlos en carpetas según el estudio correspondiente. Además, se asignó a cada instancia un nombre que detallaba completamente su procedencia (Figura 4.25).

En el mismo módulo, el submódulo *copy_to_warehouse.py* guardaba todas las instancias de manera ordenada en un DataWarehouse. Una vez ordenados los datos, se procesaban en el módulo principal *curation.py*, donde se registraban las direcciones de los datos que cumplían (*Logs_clean.txt*), no cumplían (*Logs_error.txt*) y las razones por las que no cumplían (*Log.txt*) con ciertos requisitos que se detallan más adelante.

Finalmente, el módulo *selected_data.py* leía las rutas de los archivos que pasaban los filtros y los copiaba a una carpeta de espera para más datos o los pasaba directamente a la creación de un nuevo DataRelease, según las necesidades del equipo.

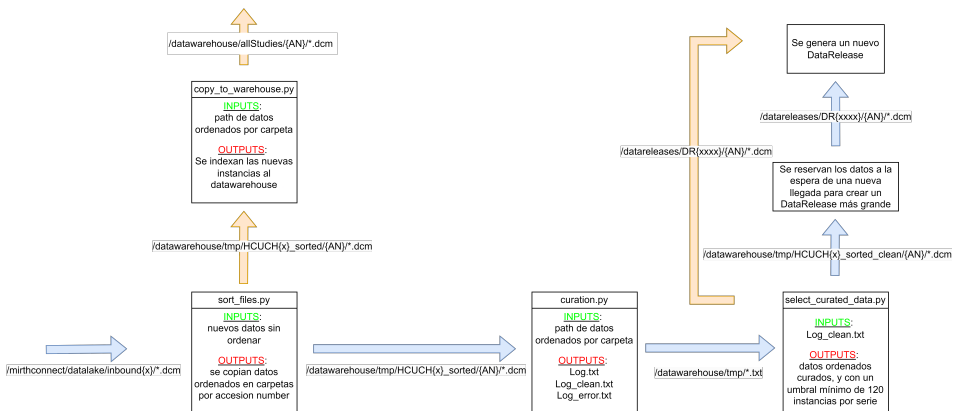


FIGURA 4.24: Pipeline de datos dentro del servidor de Proximity

4.2.2.3 Descripción del nombre de los archivos

```
sorted_data_folder/
├── <Accesion-Number1>/
│   ├── <serie-in-folder-id1>_<Serie-UID1>_<Instance-Number>_<Accesion-Number1>AN.dcm
│   ├── ...
│   ├── <serie-in-folder-id2>_<Serie-UID2>_<Instance-Number>_<Accesion-Number1>AN.dcm
│   └── ...
├── <Accesion-Number2>/
│   ├── <serie-in-folder-id1>_<Serie-UID1>_<Instance-Number>_<Accesion-Number2>AN.dcm
│   ├── ...
│   ├── <serie-in-folder-id2>_<Serie-UID2>_<Instance-Number>_<Accesion-Number2>AN.dcm
│   ├── ...
│   ├── <serie-in-folder-id3>_<Serie-UID3>_<Instance-Number>_<Accesion-Number2>AN.dcm
│   └── ...
└── ...
```

FIGURA 4.25: Árbol de carpetas ordenadas

Como se observa en la Figura 4.25, el nombre de cada archivo separa cuatro identificadores mediante un guion bajo, esto con la justificación de que en caso de pasar todos los archivos a una sola carpeta, o al tener que buscar archivos específicos, estos sean autorreferentes, y, por lo tanto, solo se aplique un procesamiento de nombre de archivo, en vez de tener que gastar memoria en el reprocesado de todos los archivos. A continuación se muestra el detalle del nombre de los archivos:

- **serie_in_folder-id**: corresponde a un identificador corto que permite separar las series de manera más fácil, en otras palabras, simplemente es para ver al ojo humano el número de series diferentes y estimar el número que tiene cada una dentro de cada estudio.
- **Serie_UID**: identificador único de cada serie

- **Instance _Number:** Posición de una instancia dentro de una serie. Si se extrapola a un video, equivale a la posición que debería ocupar cada cuadro dentro de la secuencia de imágenes
- **Accesion _Number:** identificador único de cada estudio. En la gran mayoría de los casos coincide con el Study ID.

4.2.2.4 Criterios aplicados en curación de datos

Curación es el proceso de identificar y eliminar los datos que son incorrectos o irrelevantes. Esto puede incluir datos faltantes, duplicados, valores atípicos o datos que no son relevantes para el problema de aprendizaje automático que se está abordando.

En Proximity, los archivos DICOM son curados con parámetros particulares, extraídos de la metadata. Para que el corte o instancia, sea aceptable y pueda conformar una serie, debe tener:

- OBLIGATORIAS:
 - Slice Thickness: 1.5
 - Modality: CT
 - Pixel Data: 524288 (puede variar, pero es lo normal para que sea de 512x512 y 2^{16} valores de bits por píxel)
 - 'Image Orientation Patient'[4]: 1 (representa que el paciente está en supino o "boca arriba")
- SITUACIONALES

Si el estudio fue tomado con una máquina marca Siemens modelo Sensation 64:

- Series Description: B70f (debe contener el string)

- OPCIONALES

- Rescale Slope: No importa el valor pero debe existir un valor
- Rescale Intercept: No importa el valor pero debe existir un valor

Como salida, debido a que la data con errores sirve para pedir que se vuelva a enviar, o para investigar sobre cuál es el origen del problema, se crearon dos archivos de texto: uno con las direcciones de los archivos correctos, y otro con las direcciones de los archivos que contienen errores.

4.2.2.5 Selección de estudios

Para explicar este paso, se debe mostrar, en primer lugar, las entradas y las salidas.

- ENTRADAS

- Número de series a seleccionar (INT)
- Número de instancias a seleccionar (INT)¹
- Series a seleccionar ("random" o LIST)²
- Instancias random (BOOLEAN)³

- SALIDAS

¹Si instancias random = False, este parámetro no se toma en cuenta

²"random" para seleccionar número de series fijo aleatorias, o una lista específica de Accession Number y debe coincidir con el número de series a seleccionar

³True para obtener instancias aleatorias, False para devolver todas las instancias de la serie

■ Lista de direcciones de instancias seleccionadas

En el caso de seleccionar "random" como parámetro en la selección de series, se utilizó una distribución normal $\mathcal{N}(\mu = \frac{x}{2}, \sigma^2 = \frac{x^2}{16})$, Esta decisión se basó en conclusiones obtenidas durante conversaciones con médicos del HCUCH. Se determinó que los datos con mayor información se encuentran principalmente en el centro de la serie, mientras que las primeras y últimas instancias suelen contener información inexistente o simplemente irrelevante, como se muestra en la Figura 4.26.

Para efectos de la aplicación web, la configuración de parámetros es siempre:

- Número de series a seleccionar: cambia según la vista
- Series a seleccionar: lista de series a mostrar por pantalla
- Instancias random: False

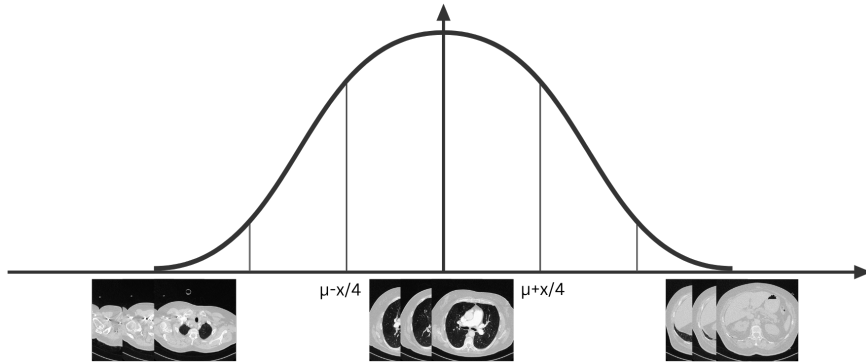


FIGURA 4.26: Representación de distribución de instancias

4.2.2.6 Aplicación de filtros

Esta etapa final de la preparación de los datos consistió principalmente en aplicar dos filtros a las imágenes. En primer lugar, el médico podía especificar el tamaño deseado para las imágenes, originalmente de $1 \times 512 \times 512$. Para ello, se utilizó la función `resize` de la biblioteca `skimage`, con los parámetros `preserve_range=True` para mantener el rango original de los píxeles de las imágenes y `anti_aliasing=True` para evitar efectos de distorsión al realizar el reescalado.

Además, el usuario tenía la opción de mantener la imagen original o ajustar el contraste. Este ajuste se realizaba convirtiendo los valores de los píxeles a Unidades de Hounsfield (HU), utilizando los parámetros `width=1000` a `1500` HU y `center=-700` HU [29]. En la Figura 4.27, se

observa la distribución de los valores de HU, que iban desde -1000 hasta 1000, y se describía a qué correspondía cada rango de valores. Con los parámetros mencionados, los valores de los píxeles podían comprender desde -200 hasta -1200 HU, lo cual representaba una vista de ventana pulmonar.

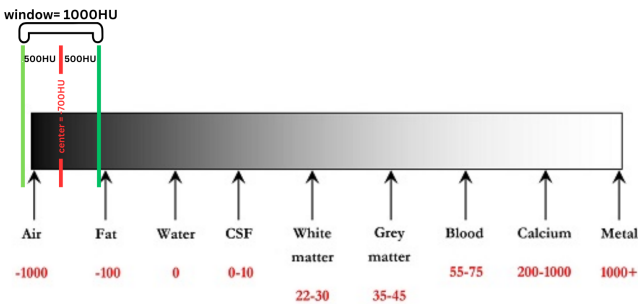


FIGURA 4.27: Distribución de valores HU

Validación y conclusiones

5.1 Validación de Aplicación Web

Para realizar la validación de la aplicación Proximity, se creó una serie de pasos que realizaron los médicos radiólogos del HCUCH, con el fin de dar una retroalimentación acerca de su experiencia de usuario. Para esto, Proximity se montó en un servidor de pruebas de la Universidad Técnica Federico Santa María, con el fin de hacer llegar la aplicación web más fácilmente a los médicos.

Plan de testing para las distintas funcionalidades de Proximity.

1. Inicio de sesión

- a) Iniciar sesión con credenciales de Proximity
- b) Cambiar contraseña
- c) Iniciar sesión con cuenta personal

2. **Revisión de Dashboard** Hacer clic uno de los últimos CTs agregados, dando paso a la vista de visualización y recuperación de casos similares.

3. **Búsqueda por etiquetas**

- a) Ingresar diferentes tipos de datos para realizar búsqueda personalizada de CTs.
- b) Hacer clic en uno de los CTs filtrados en el paso anterior para realizar recuperación de casos de estudio similares.

4. **Vista de Resultados de CTs**

- a) Comprobar que los CTs devueltos son consistentes con la imagen de la consulta.
- b) Utilizar herramientas de visualización
- c) Seleccionar un CT para obtener comparación 1 a 1.

5. **Comparación 1 a 1**

- a) Utilizar herramientas disponibles para el análisis correspondiente.
- b) Revisar informes y metadata asociada.
- c) Regresar a la vista de resultados de CTs mediante el botón dispuesto en el visualizador derecho.
- d) Comparar con otro CT.

Al finalizar, se pudieron obtener comentarios o sugerencias para realizar mejoras tanto a la interfaz de usuario, como a requisitos funcionales y no funcionales.

5.2 Validación de Datos

El proceso realizado en el procesamiento de datos de MLOps, fue validado por el equipo de Data Science, quienes gracias a los datarelease obtenidos logran entrenar diferentes modelos y comparar sus resultados con las métricas que existen en la literatura.

5.3 Conclusiones

Como se observa a lo largo de la memoria desarrollada en conjunto con el grupo de Memorias Multidisciplinarias, en la parte de desarrollo de software se logró obtener un producto mínimo viable. Este fue creado a partir de sugerencias y necesidades reales de médicos radiólogos del Hospital Clínico de la Universidad de Chile. Además, gracias al apoyo de equipos como el de Infraestructura, se pudo establecer una comunicación y acceso más rápido a los datos, tanto para su consumo como para su procesamiento. En términos de preprocesamiento de datos, el pipeline desarrollado fue concebido principalmente para el equipo de Data Science, aunque también puede ser utilizado por el frontend o backend de la aplicación, extrayendo datos con parámetros preestablecidos para diferentes vistas.

Es importante destacar que, aunque esta solución se basa en el estado del arte y en el conocimiento adquirido en múltiples cursos en la UTFSM, aún puede optimizarse para mejorar requisitos no funcionales como los tiempos de recuperación y procesamiento.

A pesar de las limitaciones mencionadas, el producto mínimo viable desarrollado representa una versión funcional del objetivo inicial al abordar este desafío. Así, proporciona una implementación efectiva de Proximity 2.0.

Bibliografía

- [1] G. Symeonidis, E. Nerantzis, A. Kazakis, and G. A. Papakostas. Mlops – definitions, tools and challenges, 2022. URL <https://arxiv.org/abs/2201.00162>.
- [2] Simon A. Lewin, Lijun Jiang, and Roberto and... Lopez-Blanco. Global genomic diversity of sars-cov-2 and the role of a novel lineage. *Pediatric Research*, 90(2):1–8, 2021. doi: 10.1007/s00431-023-05101-1. URL <https://link.springer.com/article/10.1007/s00431-023-05101-1>.
- [3] Matthew McGough, Krutika Amin, and Cynthia Cox. How has healthcare utilization changed since the pandemic? *Peterson-KFF Health System Tracker*. URL <https://www.healthsystemtracker.org/chart-collection/how-has-healthcare-utilization-changed-since-the-pandemic/>.
- [4] Bimal Jain. The key role of differential diagnosis in diagnosis. *Diagnosis*, 4, 01 2017. doi: 10.1515/dx-2017-0005.
- [5] Lisa A. Schwartz and Steven Woloshin. Lost in transmission – Dna-based diagnostics. *Nature Biotechnology*, 30(4):318–321, 2012.

- doi: 10.1038/nbt.2199. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3317765/>.
- [6] Bimal Jain. The key role of differential diagnosis in diagnosis. *Diagnosis (Berlin)*, 4(4):239–240, 2017. doi: 10.1515/dx-2017-0005.
- [7] A. Calzado and J. Geleijns. Tomografía computarizada. evolución, principios técnicos y aplicaciones. *Revista de Física Médica*, 11(3): 163–180, 2010.
- [8] Ioannis Arvanitis. Semantic interoperability in healthcare. In *Integrating Information Technology and Management for Quality of Care*, pages 1–10, 2014. doi: 10.3233/978-1-61499-423-7-5.
- [9] Neil E. Smith, John C. Newell, and J. Michael Fitzpatrick. Di-com: Digital imaging and communications in medicine. *IEEE Transactions on Medical Imaging*, 4(4):277–283, 1985. doi: 10.1109/TMICS.1985.4785318.
- [10] J. J. Cimino, J. H. Campbell, and J. W. Rogers. Snomed ct: The standard reference terminology for health care. *Journal of the American Medical Informatics Association*, 9(3):288–291, 2002. doi: 10.1109/AMIA.2002.1034225.
- [11] Health Level Seven International. Health level seven international, 2023. URL <https://www.hl7.org/>.
- [12] FHIR. Fast healthcare interoperability resources, 2024. URL <https://www.hl7.org/fhir/>.
- [13] Angularjs web application development case studies: A curated list. <https://colorwhistle.com/angular-js-case-studies/>, 2023.

- [14] Google, apple and other users of vue.js. <https://medium.com/notonlycss/google-apple-and-other-users-of-vue-js-e4505359e5d5>, 2022.
- [15] What is react used for: When and why to use react. <https://jaydevs.com/what-is-react-used-for/>, 2023.
- [16] Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. Machine learning operations (mlops): Overview, definition, and architecture, 2022.
- [17] Gleamer solutions, 2024. URL <https://www.gleamer.ai/solutions>.
- [18] Avicenna.ai, 2024. URL <https://avicenna.ai/>.
- [19] Oxipit.ai, 2024. URL <https://oxipit.ai/>.
- [20] Toraxia, 2024. URL <https://www.alumbra.ai/toraxia/>.
- [21] Orthanc - a lightweight, restful dicom server for healthcare and medical research. <https://www.orthanc-server.com/>. Consultado en julio 2024.
- [22] Roy Pea. User centred system design-new perspectives on human/computer interaction. *J Educ Comput Res*, 01 1987.
- [23] Bruno Munari. *Título del Libro*. Editorial, Año. URL https://editorialgg.com/media/catalog/product/9/7/9788425228650_inside.pdf.
- [24] UXPin. Paper prototyping: The 10-minute practical guide, 2024. URL <https://www.uxpin.com/studios/blog/paper-prototyping-guide/>.

- [25] Erik Bernhardsson. Annoy, 2024. URL <https://github.com/spotify/annoy>.
- [26] Ville Hyvönen, Teemu Pitkänen, Sotiris Tasoulis, Elias Jääsaari, Risto Tuomainen, Liang Wang, Jukka Corander, and Teemu Roos. Fast nearest neighbor search through sparse random projections and voting. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1147–1156, 2016. doi: 10.1145/2939672.2939770. URL <https://www.cs.helsinki.fi/u/ttonteri/pub/bigdata2016.pdf>.
- [27] Camilo G. Sotomayor, Marcelo Mendoza, Víctor Castañeda, Humberto Farías, Gabriel Molina, Gonzalo Pereira, Steffen Härtel, Mauricio Solar, and Mauricio Araya. Content-based medical image retrieval and intelligent interactive visual browser for medical education, research and care. *Diagnostics*, 11(8), 2021. ISSN 2075-4418. doi: 10.3390/diagnostics11081470. URL <https://www.mdpi.com/2075-4418/11/8/1470>.
- [28] Guillaume Lemaitre et al. pydicom: Read, modify and write dicom files with python code, 2023. URL <https://pydicom.github.io/pydicom/stable/>.
- [29] Young Rae Choi, Bo Mi Gil, Myung Hoon Chung, Woong Jin Yoo, Na Young Jung, Young Hwan Kim, Sun Woo Kwon, and Jun Young Kim. Comparing attenuations of malignant and benign solitary pulmonary nodule using semi-automated region of interest selection on contrast-enhanced ct. *Journal of Thoracic Disease*, 11(5):2392–2401, 2019. doi: 10.21037/jtd.2019.05.56.