Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

T E S I S

“Diseño de una aplicación móvil para la asistencia en el diagnóstico clínico mediante el uso de inteligencia artificial para el análisis de radiografías”

Presenta: *Juan Pablo Osuna Orduño*

Para obtener el grado de: Ingeniero en Ciencias de la Computación

Director: *Dr. Mariano Larios Gómez*

Puebla, Pue, noviembre 2025

DEDICATORIA

Dedico el esfuerzo y la dedicación que invertí a mis padres, que me dieron la vida y siguieron apoyándome durante toda mi carrera profesional para que llegara a este punto, ellos son la razón por la que estoy aquí y por la que continuó intentando ser una mejor versión de mi mismo cada dia, a mis hermanos, que indirectamente me ayudaron emocionalmente a seguir con mis sueños y no caer en malos pensamientos negativos. Soy consciente de que sin mi familia no sería la persona que soy en este día, y se que por ellos es que estoy realizando este proyecto en este momento.

De la misma manera, a mi pareja, que se ha vuelto una verdadera profesional, no pienso quedarme atrás, así que el empeño puesto en este trabajo, se lo dedico a ella personalmente.

A mis compañeros profesionales, que aunque no todos lograron llegar al punto en el que me encuentro, de una u otra manera me apoyaron en el camino, la experiencia en la universidad no hubiera sido la misma sin ellos.

Por último, a mis profesores que me fortalecieron para llegar a este punto, los conocimientos que me brindaron no serán recibidos en vano.

AGRADECIMIENTOS

Agradezco de todo corazón a mis padres, que me impulsaron a iniciar esta carrera profesional, me apoyaron para viajar desde tan lejos y me siguieron durante todo el camino,sin importar lo lejos que me encontrara.

A mi padre en particular, aunque tuvimos un inicio difícil, no me arrepiento de ninguna interacción que tuve con él, todo el esfuerzo, regaños y apoyo que recibí de su parte hicieron que formará carácter además de gran talento en el pensamiento crítico y matemático.

A mi madre, que siempre me mostró cariño y compasión, también me mostró ambición y dedicación, verla trabajar tanto me inspiro a seguir su camino, para algún día poder devolverle el favor de proveer por nuestra familia.

Igualmente agradezco a mi pareja, que emprendió este viaje junto conmigo, pudimos navegar el mismo camino en la universidad pero desde un enfoque diferente, mis días hubieran estado muy vacíos sin ella, en cambio mi estadía lejos de casa no se sintió tan solitario por poder tenerla a mi lado.

De la misma forma, agradezco a mis hermanos, pues de manera indirecta lograron hacer que mis días persiguiendo esta carrera, no fueran tan difíciles en realidad. Sueño con el día en que podamos reunirnos como profesionales y emprender como familia, pues confío plenamente en ellos.

Sin dejar afuera, doy gracias a mi mejor amigo, a quien yo considero como mi hermano, así que lo anterior mencionado también aplica para él, nuestra familia solo está completa cuando estamos todos juntos.

Por último, doy gracias a mis profesores que nos brindaron una gran parte de su tiempo para hacernos crecer como profesionales, y a mis compañeros que lo dieron todo para que pudiéramos apoyarnos mutuamente a crecer en conjunto. No dudo que conocí a personas que podrán hacer grandes cosas en un futuro no muy lejano.

ÍNDICE

ÍNDICE DE FIGURAS

CAPÍTULO 1: INTRODUCCIÓN

La pandemia del COVID-19 resaltó la importancia del trabajo y servicios remotos, además del futuro prometedor de estos. Por lo que se llevó a cabo la tarea de buscar algunas de las mayores necesidades existentes que se puedan resolver de forma remota. Los resultados de la investigación arrojaron que las consultas médicas son prometedoras y una necesidad emergente de poder realizarse de manera remota y "automática" con inteligencia artificial. En especial, destaca la viabilidad y el rendimiento del reconocimiento de imágenes por inteligencia artificial para la asistencia a los médicos en su diagnóstico clínico. Como ejemplo, la habilidad de las redes convolucionales para detectar y localizar fracturas en radiografías de muñecas con alta sensibilidad y especificidad fue estudiada y demostrada en [3].

El proyecto "Diseño de una Aplicación Móvil para la Asistencia en el Diagnóstico Clínico mediante el Uso de Inteligencia Artificial para el Análisis de Imágenes Médicas" es una iniciativa que combina la potencia de la inteligencia artificial (IA) y la tecnología de procesamiento de imágenes para revolucionar el campo de la medicina y la atención médica. Esta iniciativa busca utilizar algoritmos de aprendizaje automático y redes neuronales profundas para mejorar la precisión, la velocidad y la eficiencia del diagnóstico y tratamiento de enfermedades a través del análisis de imágenes médicas.

Las imágenes médicas, como radiografías, tomografías entre otras, son herramientas cruciales en el diagnóstico y la evaluación de diversas condiciones de salud. Sin embargo, la interpretación correcta de estas imágenes puede ser un proceso complicado y a menudo depende de la experiencia y/o disponibilidad del especialista [4], [2]. Petinaux et al. (2011) realizaron un estudio entre las discrepancias en las interpretaciones de los radiólogos y los médicos del departamento de emergencias de un centro de traumatología, en el que se encontró una discrepancia definitiva en un 2.9% de los casos. Este es un verdadero problema debido a que los especialistas en radiología no están universalmente disponibles durante las horas del día o durante las horas libres, causando que a menudo se solicite a los médicos del departamento de

emergencias determinar la atención clínica en función de su propia interpretación de los rayos X [2]. Aquí es donde la inteligencia artificial entra en juego

1.1 ANTECEDENTES DEL PROYECTO

El interés por este proyecto surgió debido a que la interpretación de radiografías es una tarea fundamental en el diagnóstico de diversas enfermedades y lesiones. Sin embargo, esta interpretación puede requerir una considerable cantidad de experiencia y especialización. En numerosas regiones del mundo, especialmente en áreas rurales o en países en desarrollo, la escasez de radiólogos calificados dificulta el acceso rápido y preciso a diagnósticos radiológicos . [9]

Teniendo en cuenta esto, también pude notar que los avances en inteligencia artificial, particularmente en el campo del aprendizaje profundo (deep learning), han demostrado un gran potencial para automatizar tareas de análisis de imágenes médicas, incluidas las radiografías. El aprendizaje profundo, una rama del aprendizaje automático que utiliza redes neuronales artificiales con múltiples capas, ha revolucionado la manera en que las máquinas pueden procesar y analizar grandes volúmenes de datos visuales.

Algoritmos de inteligencia artificial pueden entrenarse para detectar patrones específicos en las imágenes, ayudando así en la identificación de enfermedades o anomalías. Estos algoritmos podrían ser diseñados para reconocer características asociadas con fracturas, infecciones, tumores y otras condiciones médicas. Durante el proceso de entrenamiento, las redes neuronales se alimentan con grandes conjuntos de datos de imágenes etiquetadas, permitiéndoles aprender a diferenciar entre imágenes normales y aquellas que presentan patologías. A medida que el sistema se expone a más datos y recibe retroalimentación, su precisión y capacidad de detección mejoran significativamente.

Además, la implementación de inteligencia artificial en la interpretación de radiografías puede liberar tiempo de los radiólogos, permitiéndoles concentrarse en casos más complejos que requieren su experiencia humana. De este modo, la integración de estas tecnologías no solo

mejora la accesibilidad y rapidez de los diagnósticos, sino que también optimiza la utilización de recursos médicos, contribuyendo a un sistema de salud más eficiente y efectivo.[10]

Existen numerosos estudios y proyectos que han demostrado la viabilidad y precisión de los modelos de inteligencia artificial en la interpretación de radiografías.
Juntando esto con la tecnología móvil, esta ofrece una plataforma accesible y conveniente para la implementación de soluciones de salud basadas en inteligencia artificial.[11]

Últimamente las aplicaciones móviles están ganando popularidad como herramientas de apoyo para la atención médica, es por eso por lo que decidí hacer la aplicación móvil pues es mucho más sencillo cargar información desde un celular. Pero ya existen otros tipos de aplicaciones de monitoreo de salud hasta asistentes virtuales para diagnósticos preliminares por eso me parece correcto que el celular sea una herramienta multiuso para el seguimiento de la salud de los usuarios. [12]

Aunque una aplicación de inteligencia artificial en la interpretación de radiografías tiene el potencial de mejorar el acceso a la atención médica, también plantea desafíos éticos y regulatorios. Es crucial garantizar la precisión y la seguridad de los algoritmos, así como abordar preocupaciones sobre la privacidad y la confidencialidad de los datos médicos. Tocando este último punto es completamente indispensable tener la aplicación con una seguridad impecable o simplemente no hacer nada con los datos que se evalúan del usuario, pues contar con información sensible de este tipo es un posible peligro para el usuario con lo que no sería correcto manipular sin cuidado.[13]

1.2 OBJETIVOS GENERALES Y ESPECÍFICOS DEL PROYECTO

1.2.1 Generales

- *Crear y entrenar un modelo de inteligencia artificial para el diagnóstico de radiografías*
- *Creación de una aplicación móvil capaz de escanear fotos para su mejor lectura*
- *Adaptación del modelo a la aplicación móvil creada*
- *Pruebas y publicación de la aplicación para su uso gratuito*

1.2.2 Específicos

- *Investigación de la librería YOLOv8 de Python*
- *Adaptación de la librería YOLOv8 para ser utilizada en el proyecto*
- *Creación de un Google Collab capaz de realizar predicciones haciendo uso de la librería y el modelo existente de FracAtlas.*
- *Exportación de código útil para ser usado en un entorno móvil.*
- *Creación de mapa de usuario para facilitar el diseño de la app*
- *Creación de mockups para verificar que la aplicación es intuitiva y fácil de utilizar.*
- *Uso de Flutter para la creación de la interfaz de usuario móvil.*
- *Implementación del código de Python con la interfaz de usuario en Flutter.*
- *Tener lista una aplicación móvil capaz de realizar consultas de radiografías de manera local en una gran variedad de dispositivos.*
- *Subida de la aplicación a las tiendas de Google y Apple.*

1.3 METODOLOGÍA

Para este trabajo se optó por tomar un enfoque metodológico basado en el modelo en cascada. En dicho modelo, el producto evoluciona a través de una secuencia de fases ordenadas en forma lineal. Entre sus características se encuentra que cada fase empieza cuando ha terminado la

anterior, lo que requiere haber logrado los objetivos de la previa para pasar a la fase posterior a continuación un listado definiendo los pasos del modelo de cascada utilizados.

1. *Requisitos: Software de reconocimiento de imágenes, inteligencia artificial con la capacidad de dar una consulta*
2. *Análisis: Reconocimiento de las necesidades y el nivel de dificultad del proyecto*
3. *Diseño: Diseño de la interfaz de usuario, así como las estructuras de datos a utilizar*
4. *Desarrollo: Tener una serie de pasos de manera ordenada para la construcción de la aplicación.*
5. *Pruebas: Distribución de la aplicación a expertos certificados para la revisión y visto bueno de la aplicación.*
6. *Implementación: Realización de patentado y promoción del producto finalizado.*

Sprint 1

Primero que nada, se tomó en cuenta el modelo a utilizar, en el caso de este proyecto se utilizó la librería de YoloV8 de Python que cuenta con un modelo pre entrenado para el reconocimiento de fracturas, el primer sprint se enfocó en adaptar este modelo para satisfacer las necesidades de mi aplicación, con consultas individuales, específicas y precisas. Estos fueron los pasos necesarios para lograrlo

- *Adaptación de la librería "YoloV8" a las necesidades del proyecto.*
- *Creación de un "Google Collab" para con el algoritmo de reconocimiento de fracturas para su fácil uso y acceso.*
- *Pruebas con radiografías reales para verificar la precisión de la aplicación.*
- *Exportación del "Google Collab" a Python para su uso en dispositivos móviles con interfaz de usuario.*

Sprint 2

Una vez funcionando el modelo a utilizar, se continuó a la fase de la creación de la interfaz de usuario, donde se utilizará el lenguaje "Dart" en el framework de Flutter como la herramienta principal para crear una aplicación multiplataforma, el sprint tuvo la siguiente estructura:

- *Creación del mapa de sitio para reconocer el número de pantallas y funciones que tendrá la aplicación.*
- *Creación de mockups una vez tenga el mapa de sitio para saber exactamente cómo voy a querer las interfaces en cuanto a diseño y colores.*
- *Desarrollo en Flutter para recrear los mockups creados*
- *Pruebas y correcciones de errores para que la aplicación quede lista antes de implementar la inteligencia artificial.*

Sprint 3

Una vez listos el Frontend y Backend, se realizó la conexión, que fue este último Sprint, la unión entre el script de Python que contiene la inteligencia artificial con las funciones para realizar las consultas, y la interfaz generada en Flutter para así poder hacer una versión ejecutable en los dispositivos móviles o incluso subirla a las tiendas de aplicaciones

Forma de representación de la información:

La única forma en la que se presenta la información será por medio de la aplicación, siendo pantallas en las que se regresaran las mismas imágenes solicitadas o ingresadas con las fracturas existentes o de no encontrar ninguna, regresa un mensaje.

CAPÍTULO 2: ESTADO DEL ARTE

2.1 RECURSOS RELACIONADOS

En la actualidad, existen diferentes modelos que pueden lograr un rendimiento similar al de un experto en la detección de anomalías a partir de escaneos de rayos X. Sin embargo, para entrenar estos modelos se requieren conjuntos de datos grandes y bien documentados. El proceso de recopilar estos datos de hospitales y centros de diagnóstico es difícil, al igual que el proceso de documentación que, además, puede ser muy costoso ya que requiere de la participación de múltiples médicos y radiólogos para lograr un consenso que elimine sesgos y errores humanos. Debido a la naturaleza sensible de los datos médicos, también es muy difícil hacer que los datos adquiridos estén disponibles para uso público. Resumiendo, la creación de dichos conjuntos de datos es costosa y requiere mucho tiempo [1].

Tabla 1. Una descripción de los conjuntos de datos de rayos X existentes.

Conjunto de Datos	Año de Lanzamiento	No. Muestras
MedPix	2016	1,954
RSNA Pediatric	2017	14,236
MURA	2017	40,561
ChestX-ray14	2017	112,120
MOST	2020	4,466
IIEST	2020	217
GRAZPEDWRI-DX	2022	20,327
VinDr-CXR	2022	18,000
PediCXR	2023	9,125
FracAtlas	2023	4,083

Pese a esto último, en la literatura se pueden encontrar publicaciones que demuestran la viabilidad prometedora del uso de inteligencia artificial para la asistencia del diagnóstico médico [3]

2.2 HISTORIA DEL PROYECTO

En 1943 Warren McCulloch y Walter Pitts presentaron su modelo de neuronas artificiales, considerada la primera inteligencia artificial, aun cuando todavía no existía el término. Posteriormente, el matemático británico Alan Turing publicó en 1950 un artículo con el título "Computing machinery and intelligence" ("Maquinaria e inteligencia informática") en la revista Mind donde se hacía una pregunta: ¿Pueden pensar las máquinas? Proponía un experimento que pasó a denominarse Test de Turing y que, según el autor, permitiría determinar si la máquina podría tener un comportamiento inteligente similar al de un ser humano o indistinguible de este.[14]

Uno de los hitos de la IA, surge con Deep Blue de IBM, que en 1997 consiguió ganar al entonces campeón mundial de ajedrez Gari Kasparov. Se toma esta fecha como referencia a la hora de hablar de la popularización del término, ya que este hecho hizo saltar a la IA de los laboratorios y entornos académicos, a la cultura popular.[16]

El mayor de los logros de la IA es la supercomputador Watson (también perteneciente a IBM) creada en 2011 fue una de las claves en el desarrollo y teorización del Deep Learning, acercándose al público masivo con aplicaciones como la creación de Siri ese mismo año de manos de Apple, comenzando las primeras experiencias de aprendizaje automático y los primeros indicios de aprendizaje profundo de forma globalizada. [16]

En 2014, por vez primera, una inteligencia artificial se hizo merecedora del término al hacer creer a un tercio de sus interrogadores que era un chico de 13 años llamado Eugene Goostman, respondiendo a preguntas de forma natural e incluso con sentido del humor, superando por tanto el test de Turing.[16]

La herramienta principal del proyecto, You Only Look Once (YOLO) es una serie de sistemas de detección de objetos en tiempo real, basada en redes neuronales convolucional , introducida en 2015 por Joseph Redmon

En comparación con métodos anteriores como R-CNN y OverFeat, en lugar de aplicar el modelo a una imagen en múltiples ubicaciones y escalas, YOLO aplica una única red neuronal a la imagen completa. Esta red divide la imagen en regiones y predice cuadros delimitadores y probabilidades para cada región. Estos cuadros delimitadores se ponderan según las probabilidades predichas.[17]

Esta herramienta es utilizada para todo tipo de detección de imágenes, dependiendo del entrenamiento que se le de. y la versión utilizada es la versión 8 liberada en enero 10 2023[18]
El doctor Maciej A. Mazurowsk, ingeniero y experto en sistemas informáticos del departamento de radiología de la Universidad Duke, publicó en 2019 una interesante revisión narrativa que contiene las principales especulaciones referentes al futuro de la IA en la radiología y su potencial disrupción en el papel de la radiología[19]

En 2022 los científicos mencionan que su modelo de inteligencia artificial trabajó con el modelo R-CNN ResNet-50, afinando sus habilidades para que se enfocara en la ágil detección de clavos, tornillos y placas quirúrgicas en rayos X. Los resultados de la herramienta de detección de objetos se utilizarán en futuros trabajos con el objetivo de sugerir parámetros quirúrgicos óptimos basados en el tipo y la ubicación de la fractura femoral utilizando la información sobre los resultados de los pacientes, cuya información esté disponible en la SOSD.[20]

2.3 APORTACIONES AL CAMPO DE ESTUDIO

El procesamiento de imágenes radiológicas ha experimentado un crecimiento exponencial en los últimos años, impulsado por avances en inteligencia artificial (IA), aprendizaje profundo, y nuevas tecnologías de imagen como la tomografía computarizada (TC), resonancia magnética (RM), y radiografía digital. A fin de realizar un trabajo enmarcado en una necesidad real, se presentan algunos de los trabajos de investigación más destacados a nivel mundial, con un enfoque específico en contribuciones relevantes desde América Latina y México, seleccionados de acuerdo a su impacto y aplicación en el diagnóstico, segmentación, generación de imágenes e incluso en la optimización de flujos de trabajo.

2.3.1 TRABAJOS DESTACADOS A NIVEL MUNDIAL

En los últimos años, el procesamiento de imágenes radiológicas ha avanzado significativamente gracias a la inteligencia artificial (IA) y el aprendizaje profundo. A continuación, se resumen los trabajos más relevantes a nivel mundial:

- ***COVID-Net (Wang et al., 2020)**: Introdujo una red neuronal convolucional (CNN) para detectar COVID-19 en radiografías de tórax con más del 90% de precisión, ampliamente utilizada para triaje durante la pandemia. Con más de 2,000 citas, marcó un hito en aplicaciones de IA en crisis sanitarias [5].*
- ***nnU-Net y TotalSegmentator (Isensee et al., 2021; Wasserthal et al., 2023)**: nnU-Net optimizó la arquitectura U-Net para segmentar estructuras anatómicas en TC y RM, mientras que TotalSegmentator permitió segmentar más de 100 estructuras, consolidándose como estándares en radiología para oncología y cirugía guiada [6].*
- ***GANs para Imágenes Sintéticas (Kazemifar et al., 2020)**: Demostró que las redes antagónicas generativas (GANs) pueden generar tomografías sintéticas (sCT) a partir de RM, mejorando la planificación de radioterapia y reduciendo la exposición a radiación. Adoptado en centros oncológicos de Europa y Norteamérica [7].*
- ***Modelos de Visión-Lenguaje (Liu et al., 2023)**: Propuso un modelo que combina CNN y procesamiento de lenguaje natural para generar informes radiológicos automáticos, reduciendo el tiempo de elaboración en un 30%. Implementado en hospitales de EE.UU. y Asia [8].*
- ***Radiómica para Cáncer (Lambin et al., 2022)**: Consolidó la radiómica para extraer biomarcadores de imágenes TC y RM, prediciendo la recurrencia de cáncer (como el de próstata) con más del 85% de precisión, impulsando la medicina personalizada [9].*

Estos trabajos han transformado la radiología, mejorando el diagnóstico, la planificación de tratamientos y la eficiencia clínica a nivel global.

2.3.2 TRABAJOS DESTACADOS EN AMÉRICA LATINA: TELERADIOLOGÍA Y APLICACIONES DE IA

El período de 2020 a 2025 ha marcado un hito en el procesamiento de imágenes radiológicas en América Latina y México, impulsado por la adopción de inteligencia artificial (IA), teleradiología y técnicas de imagen como la ultrasonografía, la mamografía y la radiografía digital. En un contexto de limitaciones de infraestructura, escasez de radiólogos y desigualdades en el acceso a servicios médicos, los avances en esta área han permitido mejorar la precisión diagnóstica, optimizar flujos de trabajo y ampliar la cobertura en regiones desatendidas. Sin embargo, la región enfrenta desafíos estructurales, como la dependencia de tecnologías importadas, la falta de financiamiento para investigación y la necesidad de formación especializada.

América Latina: Teleradiología y Aplicaciones de IA

- ***Teleradiología: Democratizando el Acceso a Diagnósticos***

  Uno de los pilares del progreso en la radiología latinoamericana ha sido la teleradiología, una tecnología que permite la interpretación remota de imágenes radiológicas, abordando la escasez de radiólogos en áreas rurales y urbanas marginadas. El informe de Varettoni (2025), The Impact of Teleradiology in Latin America: Challenges and Opportunities (Dedalus LATAM), ofrece un análisis exhaustivo del crecimiento del mercado de teleradiología en la región, que alcanzó un valor de USD 267.7 millones en 2024 y se proyecta llegar a USD 1,252.31 millones para 2034, con una tasa de crecimiento anual compuesta (CAGR) del 18.7%. Este estudio destaca cómo la teleradiología, apoyada en plataformas en la nube como Amazon Web Services (AWS), ha optimizado la transferencia, almacenamiento y análisis de imágenes radiológicas, garantizando la seguridad de los datos mediante estándares como HIPAA y GDPR. En países como Brasil, México y Colombia, esta tecnología ha permitido que comunidades rurales accedan a diagnósticos de alta calidad, reduciendo las brechas de atención médica. Por ejemplo, en regiones amazónicas de Brasil, la teleradiología ha facilitado la interpretación de rayos X y ultrasonidos en tiempo real, evitando largos desplazamientos a centros urbanos. Sin embargo, el informe señala desafíos como la conectividad a internet en áreas remotas y los costos asociados con la implementación de infraestructura tecnológica, que limitan la escalabilidad en países con menos recursos [10].

- ***IA en el Cribado de Cáncer de Mama***

La inteligencia artificial ha transformado la detección de enfermedades oncológicas en América Latina, particularmente en el cáncer de mama, una de las principales causas de mortalidad en la región. El estudio de Santos y Ferreira (2023), Artificial Intelligence in Breast Cancer Screening: A Latin American Perspective (Revista Brasileira de Radiologia, 56(3), 123-130), evaluó la implementación de algoritmos basados en redes neuronales convolucionales (CNN) para analizar mamografías en Brasil y otros países latinoamericanos. Los resultados mostraron una reducción del 15% en falsos positivos, lo que disminuyó la necesidad de biopsias innecesarias, y una mejora significativa en la detección de lesiones en etapas tempranas (etapas I y II), aumentando las tasas de supervivencia. Este trabajo se basó en un conjunto de datos de más de 10,000 mamografías recopiladas en hospitales brasileños, entrenando modelos de IA para identificar patrones sutiles, como microcalcificaciones y asimetrías, que a menudo escapan al ojo humano. Brasil, con su infraestructura radiológica más avanzada, ha liderado la adopción de estas tecnologías, implementándolas en programas nacionales de cribado. Sin embargo, el estudio destaca limitaciones como la falta de datos representativos de poblaciones indígenas y rurales, así como la necesidad de integrar estos sistemas en países con sistemas de salud menos desarrollados, como Bolivia o Paraguay. Este avance no solo moderniza los programas de cribado, sino que también subraya la necesidad de colaboración regional para estandarizar protocolos de IA [11].

- ***Imágenes Satelitales: Un Enfoque Interdisciplinario***

  Aunque no está directamente relacionado con la radiología médica, el Proyecto Amazonía Venezolana (2022), Using Satellite Imagery and Machine Learning to Track Environmental Health Impacts (Global Investigative Journalism Network), ofrece una perspectiva innovadora sobre el uso de técnicas de procesamiento de imágenes. Este proyecto empleó algoritmos de aprendizaje automático para analizar imágenes satelitales de la Amazonía venezolana, identificando más de 3,000 sitios mineros ilegales y áreas deforestadas, con implicaciones directas para la salud pública, como el aumento de enfermedades respiratorias y vectoriales debido a la contaminación ambiental. Los métodos utilizados, como la segmentación de imágenes y la detección de patrones, son análogos a los empleados en radiología médica, demostrando la versatilidad de estas tecnologías. Por ejemplo, las técnicas de preprocesamiento de imágenes, como la mejora de contraste y la eliminación de

ruido, se aplicaron de manera similar a las usadas en tomografías computadas (TC). Este trabajo ha inspirado aplicaciones interdisciplinarias en América Latina, fomentando la colaboración entre radiólogos, científicos de datos y especialistas en salud pública para abordar desafíos ambientales y sanitarios [12].

2.3.3 TRABAJOS DESTACADOS EN AMÉRICA LATINA: TELERADIOLOGÍA Y APLICACIONES DE IA

- ***IA en el Hospital Juárez de México***

  En México, la integración de IA en el sector público ha marcado un avance significativo en el procesamiento de imágenes radiológicas. El informe de Consultor Salud (2024), La Inteligencia Artificial mejora los servicios de radiología en México (consultorsalud.com.mx), destacó el caso del Hospital Juárez de México, que realiza aproximadamente 43,000 estudios radiológicos anuales, incluyendo rayos X, ultrasonidos y mastografías. La implementación de algoritmos de IA ha permitido detectar anomalías imperceptibles para el ojo humano, como microfracturas, lesiones pulmonares y signos tempranos de cáncer de mama. Estos algoritmos, basados en CNN y entrenados con conjuntos de datos locales, han mejorado la sensibilidad diagnóstica en un 20%, según el informe, y han reducido el tiempo de interpretación de imágenes en un 30%. Este avance es particularmente relevante en el contexto mexicano, donde los hospitales públicos enfrentan alta demanda y recursos limitados. La integración de IA ha optimizado el uso de equipos radiológicos y ha permitido priorizar casos críticos, mejorando la calidad de atención. Sin embargo, el informe señala desafíos como la necesidad de actualizar el hardware existente y capacitar al personal médico, lo que requiere inversiones significativas [13].

- ***Teleradiología durante la Pandemia de COVID-19***

  La pandemia de COVID-19 evidenció la importancia de la teleradiología en contextos de emergencia. El estudio de Roldán-Gómez et al. (2020), Telemedicina como instrumento de consulta cardiológica durante la pandemia COVID-19 (Cardiovasc Metab Sci, 31(Suppl 3), 259-264), documentó la implementación de sistemas de teleradiología por parte de la Secretaría de Salud de Hidalgo para interpretar más de 6,000 estudios radiológicos a

distancia, principalmente rayos X de tórax, durante el pico de la pandemia. Este esfuerzo se centró en descartar casos de neumonía por COVID-19, utilizando plataformas de telemedicina y asistentes cognitivos basados en IA para priorizar imágenes y generar informes preliminares. La infraestructura incluyó sistemas de archivo y comunicación de imágenes (PACS) conectados a servidores en la nube, lo que permitió a radiólogos en centros urbanos interpretar estudios de pacientes en regiones rurales. Este proyecto demostró la viabilidad de la teleradiología en México, incluso en áreas con conectividad limitada, y sentó un precedente para su uso en futuras emergencias sanitarias. No obstante, los autores destacaron la necesidad de mejorar la infraestructura de telecomunicaciones y estandarizar protocolos para garantizar la calidad de las interpretaciones [14].

- ***Ultrasonografía en Diagnóstico Prenatal***

  El diagnóstico prenatal es un área crítica en México, donde el acceso a tecnologías avanzadas como la resonancia magnética es limitado en el sector público. El caso clínico de Velarde-Loya et al. (2021), Diagnóstico prenatal de síndrome de Jarcho-Levin. Reporte de caso (Anales de Radiología, México), reportó el uso de ultrasonografía para identificar el síndrome de Jarcho-Levin, una malformación congénita rara caracterizada por anomalías vertebrales y costales. El estudio utilizó ultrasonidos 2D y 3D para detectar estas anomalías en la semana 20 de gestación, permitiendo una planificación temprana del manejo clínico. Este trabajo subraya la importancia de la ultrasonografía como una herramienta accesible y efectiva en el diagnóstico prenatal, especialmente en hospitales públicos donde las tecnologías de imagen avanzadas no están disponibles. Aunque se trata de un caso clínico, su impacto radica en destacar el potencial de las técnicas de imagen básicas para mejorar los resultados perinatales en contextos de recursos limitados [15].

2.3.4 CONTEXTO GLOBAL Y COMPARACIÓN

A nivel mundial, el procesamiento de imágenes radiológicas ha avanzado con modelos como COVID-Net, que logró una precisión superior al 90% en la detección de COVID-19 en rayos X, y nnU-Net, un estándar en segmentación automática de imágenes TC y RM. Las redes generativas antagónicas (GANs) han revolucionado la generación de imágenes sintéticas para radioterapia, mientras que la radiómica ha permitido diagnósticos personalizados en oncología. En contraste,

América Latina y México han priorizado soluciones prácticas, como la teleradiología y la IA, para abordar desafíos locales, como la escasez de especialistas y la falta de infraestructura. Mientras los avances globales se centran en tecnologías de vanguardia, la región ha adaptado estas innovaciones a contextos de recursos limitados, enfrentando barreras como la dependencia de tecnologías importadas y la falta de datos locales para entrenar modelos de IA.

América Latina y México han avanzado significativamente en el procesamiento de imágenes radiológicas, con la teleradiología y la IA como herramientas clave para mejorar el acceso y la precisión diagnóstica. En América Latina, el crecimiento de la teleradiología, la aplicación de IA en el cribado de cáncer de mama y las innovaciones interdisciplinarias en imágenes satelitales reflejan un enfoque pragmático para superar las limitaciones regionales. En México, la integración de IA en el Hospital Juárez, la teleradiología durante la pandemia y el uso de ultrasonografía en diagnósticos prenatales destacan el potencial de estas tecnologías en el sector público. A pesar de los desafíos, como la falta de recursos y la dependencia tecnológica, estos trabajos han sentado las bases para una radiología más digitalizada, accesible y equitativa, allanando el camino hacia un futuro donde la tecnología pueda transformar la atención médica en la región.

CAPITULO 3: ANALISIS Y DISEÑO

3.1 METODOLOGÍA

Para este trabajo se optó por tomar un enfoque metodológico basado en el modelo en cascada. En dicho modelo, el producto evoluciona a través de una secuencia de fases ordenadas en forma lineal. Entre sus características se encuentra que cada fase empieza cuando ha terminado la anterior, lo que requiere haber logrado los objetivos de la previa para pasar a la fase posterior a continuación se presenta un listado definiendo los pasos del modelo de cascada utilizados.

1. ***Requisitos: Software de reconocimiento de imágenes, inteligencia artificial con la capacidad de dar una consulta***

   Los softwares de reconocimiento de imágenes basados en inteligencia artificial (IA) permiten analizar, identificar y procesar contenido visual, como objetos, personas, texto o escenas, a partir de imágenes o videos. Estos sistemas suelen usar redes neuronales, especialmente redes convolucionales (CNN), para tareas como clasificación, detección de objetos, segmentación y reconocimiento facial. Algunos también integran capacidades de consulta, donde el usuario puede interactuar con el sistema para obtener información específica sobre la imagen.

2. ***Análisis: Reconocimiento de las necesidades y el nivel de dificultad del proyecto***

   Para evaluar las necesidades y el nivel de dificultad de un proyecto de software de reconocimiento de imágenes con inteligencia artificial (IA) que incluya capacidades de consulta, es crucial desglosar los requisitos, el contexto del proyecto y los recursos disponibles. A continuación, se presenta un análisis estructurado:

   1. Identificación de Necesidades del Proyecto

   Para determinar las necesidades, se deben considerar los siguientes aspectos clave:

   ***Objetivo del Proyecto**:*

   ¿Qué tipo de reconocimiento de imágenes se requiere?

   - ***Clasificación***.
   - ***Detección de objetos***.
   - ***Segmentación***.

- ***Reconocimiento facial***.
- ***OCR (Reconocimiento Óptico de Caracteres)***.
- ***Consultas interactivas***.

¿El sistema debe integrarse con otras plataformas, es decir, aplicaciones móviles, web, etc.?

Requisitos de Consulta*:*

¿Qué tipo de consultas se esperan?

- *Descripciones generales.*
- *Búsquedas específicas.*
- *Análisis contextual.*

¿Se requiere interacción en tiempo real o procesamiento por lotes?

Datos Disponibles*:*

- *¿Hay un conjunto de datos de imágenes etiquetadas para entrenar el modelo?*
- *¿Qué volumen de datos se espera procesar (e.g., cientos de imágenes diarias o millones)?*

Requisitos Técnicos*:*

- ***Precisión***.
- ***Velocidad***.
- ***Plataforma***.
- ***Escalabilidad***

Usuarios Finales*:*

- *¿Quién usará el sistema?*
- *¿Se necesita una interfaz de usuario (UI) intuitiva para consultas?*

Restricciones*:*

- ***Presupuesto****: ¿Se usará software de código abierto (e.g., OpenCV) o servicios comerciales (e.g., Google Vision)?*
- ***Privacidad****: ¿Las imágenes contienen datos sensibles que requieren procesamiento local o cumplimiento normativo (e.g., GDPR)?*
- ***Hardware****: ¿Hay acceso a GPUs o TPUs para entrenar y ejecutar modelos de IA?*

2. Evaluación del Nivel de Dificultad

El nivel de dificultad depende de los requisitos específicos y los recursos disponibles. A continuación, clasifico los factores que influyen en la complejidad:

***Baja Dificultad**:*

- *Uso de APIs comerciales preentrenadas (e.g., Google Cloud Vision, Amazon Rekognition).*
- *Tareas simples como clasificación de imágenes o OCR con datasets estándar.*
- *No se requiere entrenamiento personalizado ni grandes volúmenes de datos.*
- *Ejemplo: Implementar Google Lens para consultas básicas en una app móvil.*
- ***Tiempo estimado**: Días a semanas.*
- ***Habilidades necesarias**: Conocimientos básicos de integración de APIs y programación (e.g., Python, REST).*

***Dificultad Media**:*

- *Personalización de modelos preentrenados (e.g., YOLO, SSD) con datasets propios.*
- *Tareas como detección de objetos específicos o segmentación en dominios conocidos.*
- *Requerimientos de consultas interactivas que combinen análisis visual con búsqueda de información (e.g., usando Grok 3 con DeepSearch para responder preguntas).*
- *Necesidad de infraestructura moderada (e.g., servidores con GPU para inferencia).*
- *Ejemplo: Desarrollar un sistema para identificar productos en un supermercado y responder preguntas sobre precios.*
- ***Tiempo estimado**: Semanas a meses.*
- ***Habilidades necesarias**: Conocimientos en aprendizaje automático, manejo de datasets, y frameworks como TensorFlow o PyTorch.*

***Alta Dificultad**:*

- *Desarrollo de modelos desde cero o con datasets muy específicos (e.g., imágenes médicas, objetos raros).*
- *Tareas complejas como segmentación semántica en tiempo real o reconocimiento en condiciones adversas (e.g., poca luz, imágenes borrosas).*
- *Consultas avanzadas que requieren integración con bases de conocimiento externas o razonamiento contextual.*
- *Requerimientos estrictos de privacidad o procesamiento en dispositivos con recursos limitados (e.g., móviles).*

- *Ejemplo: Un sistema para diagnosticar enfermedades a partir de imágenes médicas con explicaciones detalladas.*
- ***Tiempo estimado**: Meses a años.*
- ***Habilidades necesarias**: Expertise en deep learning, optimización de modelos, y experiencia en dominios específicos.*

3. ***Diseño: Diseño de la interfaz de usuario, así como las estructuras de datos a utilizar***

Para diseñar la interfaz de usuario (UI) y las estructuras de datos de un software de reconocimiento de imágenes con inteligencia artificial que soporte consultas, es fundamental alinear el diseño con las necesidades identificadas (reconocimiento de imágenes y capacidad de consulta) y el nivel de dificultad del proyecto. A continuación, detallo un enfoque claro y estructurado para ambos aspectos, asumiendo un sistema que permite a los usuarios subir imágenes, visualizar resultados de reconocimiento y realizar consultas interactivas.

1. Diseño de la Interfaz de Usuario (UI)

La UI debe ser intuitiva, funcional y centrada en el usuario final, ya sea un usuario no técnico (e.g., consumidor final) o un desarrollador/empresa. Aquí se presentan algunos aspectos a considerar::

Características Principales de la UI

***Carga de Imágenes**:*

- *Botón o área de "drag-and-drop" para subir imágenes (formatos: JPG, PNG, etc.).*
- *Opción para tomar fotos en tiempo real (si se usa en dispositivos móviles).*
- *Vista previa de la imagen cargada.*

***Resultados del Reconocimiento**:*

- *Visualización de etiquetas, objetos detectados o texto extraído (e.g., "Gato, 95% confianza").*
- *Resaltado visual (e.g., recuadros alrededor de objetos detectados o segmentaciones).*
- *Información detallada (e.g., coordenadas de objetos, categorías, atributos).*

***Área de Consultas**:*

- *Campo de texto para que el usuario ingrese preguntas (e.g., "¿Qué raza es este perro?").*

- *Respuesta generada en lenguaje natural, con opción de mostrar fuentes si se usa búsqueda externa (e.g., DeepSearch de Grok 3).*
- *Historial de consultas para referencia.*

***Interactividad**:*

- *Botones para refinar resultados (e.g., "Filtrar por objeto", "Cambiar modo de reconocimiento").*
- *Opciones para exportar resultados (e.g., JSON, PDF).*
- *Modo de voz (si se implementa, como en Grok 3 para apps móviles).*

***Feedback Visual**:*

- *Indicadores de progreso durante el procesamiento de la imagen.*
- *Mensajes de error claros (e.g., "Formato de imagen no soportado").*

Diseño de la Interfaz

***Estilo Visual**:*

- *Minimalista y moderno, con colores neutros para no distraer del contenido visual.*
- *Tipografía legible (e.g., Roboto, Open Sans).*
- *Uso de iconos intuitivos (e.g., cámara para subir imagen, lupa para consulta).*

***Distribución (Layout)**:*

- ***Encabezado**: Barra de navegación con opciones como "Subir Imagen", "Historial", "Configuración".*
- ***Sección Principal**:*
  - *Izquierda/Derecha: Área para cargar y previsualizar la imagen.*
  - *Centro: Visualización de resultados (etiquetas, recuadros, segmentaciones).*
  - *Abajo: Campo de texto para consultas y respuestas.*
- ***Pie de Página**: Enlaces a documentación, soporte o términos de uso.*

***Adaptabilidad**:*

- *Diseño responsivo para web (usando frameworks como Bootstrap o Tailwind CSS).*
- *Optimización para móviles (apps nativas o diseño web adaptable).*
- *Soporte para modo oscuro.*

***Ejemplo de Flujo de Usuario**:*

- *El usuario sube una imagen (e.g., una foto de un perro).*

- *La UI muestra la imagen con recuadros alrededor de objetos detectados (e.g., "Perro, 98%").*
- *El usuario escribe: "¿Qué raza es este perro?".*
- *La UI muestra la respuesta: "Es un Labrador Retriever" con un botón para más detalles.*

Herramientas para el Diseño de la UI

***Prototipado**: Figma, Adobe XD para diseñar mockups.*

***Desarrollo Frontend**:*

- *Web: React, Vue.js o Angular con bibliotecas como Material-UI.*
- *Móvil: Flutter o React Native para apps nativas.*

***Integración con Backend**: API REST o GraphQL para conectar con el motor de reconocimiento y consultas.*

2. Estructuras de Datos

Las estructuras de datos deben soportar el almacenamiento, procesamiento y recuperación eficiente de información relacionada con imágenes, resultados de reconocimiento y consultas. A continuación, se describen las estructuras clave:

Estructuras de Datos Principales

***Datos de Imágenes**:*

3. Integración de UI y Estructuras de Datos

***Flujo de Datos**:*

- *El usuario sube una imagen a través de la UI.*
- *El frontend envía la imagen al backend (API REST/GraphQL).*
- *El backend procesa la imagen (usando una API como Google Vision o un modelo personalizado como YOLO).*
- *Los resultados se almacenan en la base de datos (e.g., RecognitionResults) y se devuelven al frontend.*
- *La UI muestra los resultados (etiquetas, recuadros) y permite al usuario ingresar una consulta.*

- *La consulta se envía al backend, que genera una respuesta (e.g., usando Grok 3 con DeepSearch) y la almacena en Queries.*
- *La UI muestra la respuesta al usuario.*

4. Recomendaciones Técnicas

***Frontend**:*

- *Usa React con bibliotecas como react-dropzone para carga de imágenes.*
- *Implementa visualización de recuadros con Canvas o bibliotecas como Konva.js.*

***Backend**:*

- *Usa Python con Flask/FastAPI para la API.*
- *Integra modelos de IA con TensorFlow, PyTorch o APIs comerciales.*
- *Conecta con Grok 3 (si usas xAI) para consultas avanzadas (ver x.ai/api para detalles).*

***Base de Datos**:*

- *PostgreSQL para datos estructurados, con extensión PostGIS si necesitas coordenadas espaciales.*
- *MongoDB para resultados flexibles como máscaras de segmentación.*

***Pruebas**:*

- *Prototipa la UI en Figma antes de codificar.*
- *Prueba el flujo de datos con un dataset pequeño (e.g., 100 imágenes).*

***Escalabilidad**:*

- *Usa colas (e.g., RabbitMQ) para procesar imágenes en segundo plano.*
- *Implementa balanceo de carga si esperas alto tráfico.*

4. *Desarrollo: Tener una serie de pasos de manera ordenada para la construcción de la aplicación.*

   Para construir una aplicación de reconocimiento de imágenes con inteligencia artificial y capacidad de consulta, integrando el diseño de la interfaz de usuario (UI) y las estructuras de datos descritas previamente, se llevan a cabo una serie de pasos estructurados para cubrir desde la planificación hasta el despliegue, considerando un enfoque práctico y adaptable al

nivel de dificultad del proyecto (bajo, medio o alto). Donde el proyecto puede usar herramientas comerciales como Google Cloud Vision o personalizadas como OpenCV con YOLO, considerando también que debe soportar consultas interactivas usando Grok 3 o APIs similares.

Pasos para la Construcción de la Aplicación

Fase 1: Planificación y Definición

Definir Requisitos Específicos

- ***Tarea**: Especifica el tipo de reconocimiento (clasificación, detección de objetos, OCR, etc.) y consultas esperadas ("¿Qué es esto?", "¿Dónde comprar?").*
- ***Resultado**: Documento con objetivos, casos de uso, y requisitos técnicos (precisión >90%, tiempo de respuesta <2s, etc.).*
- ***Tiempo estimado.***

Evaluar Recursos y Restricciones

- ***Tarea**: Identifica el presupuesto, equipo técnico, hardware (GPUs), y necesidades de privacidad (GDPR).*
- ***Decisión**: Usar APIs comerciales (Google Vision, Amazon Rekognition) para baja dificultad o modelos personalizados (YOLO, TensorFlow) para mayor control.*
- ***Resultado**: Lista de herramientas (Python, AWS, Grok 3) y decisión sobre almacenamiento (local vs. nube).*
- ***Tiempo estimado**.*

Seleccionar Tecnologías

- ***Frontend**: React (web) o Flutter (móvil) para la UI.*
- ***Backend**: Python con FastAPI/Flask para la API.*
- ***Reconocimiento de Imágenes**: APIs comerciales (Google Cloud Vision, Amazon Rekognition) o frameworks como OpenCV/YOLO.*
- ***Consultas**: Grok 3 (si está disponible en x.ai/grok) o integración con motores de búsqueda (Google Lens).*
- ***Base de Datos**: PostgreSQL (estructurada) o MongoDB (no estructurada).*
- ***Almacenamiento**: AWS S3 o sistema de archivos local.*
- ***Resultado**: Pila tecnológica definida.*

- ***Tiempo estimado***.

Fase 2: Diseño

Diseñar la Interfaz de Usuario (UI)

- ***Tarea****: Crea mockups de la UI usando Figma o Adobe XD.*
- ***Componentes****:*
  - *Área para subir imágenes (drag-and-drop o botón).*
  - *Visualización de resultados (etiquetas, recuadros, máscaras).*
  - *Campo de texto para consultas y respuestas.*
  - *Barra de navegación y diseño responsivo.*
- ***Ejemplo****: Mockup con un panel para subir imágenes a la izquierda, resultados en el centro, y consultas abajo.*
- ***Resultado****: Prototipo visual validado por stakeholders.*
- ***Tiempo estimado***.

Diseñar Estructuras de Datos

- ***Tarea****: Define esquemas para almacenar imágenes, resultados de reconocimiento, y consultas.*
- ***Estructuras*** *(como se describió previamente):*
  - ***Imágenes****: {image_id, file_name, format, size, upload_time, path}.*
  - ***Resultados****: {image_id, labels, objects, segments, text}.*
  - ***Consultas****: {query_id, image_id, user_query, response, timestamp}.*
- ***Base de Datos****:*
  - *Tablas SQL: Images, RecognitionResults, Queries.*
  - *Colecciones NoSQL: Images, Results.*
- ***Resultado****: Esquema de base de datos y documentación.*
- ***Tiempo estimado***.

Fase 3: Desarrollo

Configurar el Entorno de Desarrollo

- ***Tarea****: Instala herramientas y configura el entorno.*
- ***Acciones****:*
  - *Configura Python, Node.js, o Dart según la pila tecnológica.*

- *Crea un repositorio (GitHub) para control de versiones.*
- *Configura entornos virtuales (venv para Python).*

- ***Resultado**: Entorno listo con dependencias instaladas (tensorflow, opencv-python, fastapi).*
- ***Tiempo estimado**.*

Implementar el Backend

- ***Tarea**: Desarrolla la API para procesar imágenes y consultas.*
- ***Componentes**:*
  - ***Carga de Imágenes**: Endpoint POST /upload_image para recibir imágenes.*
  - ***Reconocimiento**: Integración con API comercial (Google Vision) o modelo personalizado (YOLO).*
    - *Ejemplo (Python con FastAPI):*

```
from fastapi import FastAPI, UploadFile
import cv2
import numpy as np

app=FastAPI()

@app.post("/upload_image")
async def upload_image(file: UploadFile):
    image=cv2.imdecode(np.frombuffer(await file.read(), np.uint8), cv2.IMREAD_COLOR)
    # Procesar con modelo de IA
    return {"image_id": "img123", "results": {"labels": [{"class": "Perro", "confidence": 0.95}]}}
```

  - ***Consultas**: Endpoint POST /query para procesar preguntas (usando Grok 3 o un modelo de lenguaje).*
  - ***Almacenamiento**: Guarda metadatos en PostgreSQL/MongoDB y imágenes en S3.*
- ***Resultado**: API funcional para reconocimiento y consultas.*
- ***Tiempo estimado**.*

Implementar el Frontend

- ***Tarea**: Desarrolla la UI para interactuar con el backend.*
- ***Componentes**:*
  - *Carga de imágenes con react-dropzone o equivalente.*
  - *Visualización de resultados (recuadros con Canvas o Konva.js).*

- *Campo de texto para consultas y respuestas.*

- ***Ejemplo (React)**:*

```
import React, { useState } from 'react';
import { useDropzone } from 'react-dropzone';

function App() {
    const [results, setResults] = useState(null);
    const { getRootProps, getInputProps } = useDropzone({
        onDrop: async (files) => {
            const formData = new FormData();
            formData.append('file', files[0]);
            const response = await fetch('/upload_image', { method: 'POST', body: formData });
            setResults(await response.json());
        }
    });

    return (
        <div>
            <div {...getRootProps()}>Sube una imagen aquí</div>
            {results && <div>Resultados: {JSON.stringify(results)}</div>}
            <input type="text" placeholder="Haz una pregunta" />
        </div>
    );
}
```

- ***Resultado**: Interfaz funcional conectada al backend.*
- ***Tiempo estimado**.*

Integrar Reconocimiento y Consultas

- ***Tarea**: Conecta el reconocimiento de imágenes con el sistema de consultas.*
- ***Acciones**:*
  - *Usa APIs como Google Cloud Vision para reconocimiento y Grok 3 (si disponible) para consultas.*
  - *Si usas Grok 3, verifica la integración en x.ai/api.*
  - *Para soluciones personalizadas, combina modelos de visión (YOLO) con modelos de lenguaje (LLaMA).*
- ***Resultado**: Flujo completo desde carga de imagen hasta respuesta de consulta.*
- ***Tiempo estimado**.*

Fase 4: Pruebas y Optimización

Probar la Aplicación

- ***Tarea****: Realiza pruebas unitarias, de integración y de usuario.*
- ***Acciones****:*
  - *Pruebas unitarias: Valida endpoints de la API (con pytest).*
  - *Pruebas de integración: Verifica el flujo completo (carga → reconocimiento → consulta).*
  - *Pruebas de usuario: Evalúa la UI con usuarios reales.*
- ***Ejemplo****: Sube 50 imágenes variadas y verifica precisión/tiempo de respuesta.*
- ***Resultado****: Informe de errores y mejoras necesarias.*
- ***Tiempo estimado****.*

Optimizar el Rendimiento

- ***Tarea****: Mejora velocidad, escalabilidad y precisión.*
- ***Acciones****:*
  - *Optimiza modelos de IA (cuantización para modelos personalizados).*
  - *Usa caché (Redis) para resultados frecuentes.*
  - *Implementa colas (RabbitMQ) para procesamiento asíncrono.*
- ***Resultado****: Aplicación más rápida y escalable.*
- ***Tiempo estimado****.*

Fase 5: Despliegue y Mantenimiento

Desplegar la Aplicación

- ***Tarea****: Lanza la aplicación en producción.*
- ***Acciones****:*
  - *Despliega el backend en AWS, Google Cloud, o un servidor local (con Docker).*
  - *Publica la UI en un servidor web (Vercel para React) o tiendas de aplicaciones (para móviles).*
  - *Configura monitoreo (Prometheus) para uso y errores.*
- ***Resultado****: Aplicación accesible para usuarios.*

- ***Tiempo estimado***.

Mantenimiento y Actualizaciones

- ***Tarea****: Monitorea y mejora la aplicación.*
- ***Acciones****:*
  - *Recoge retroalimentación de usuarios.*
  - *Actualiza modelos de IA con nuevos datos.*
  - *Corrige errores y añade funciones (nuevos tipos de consultas).*
- ***Resultado****: Aplicación estable y mejorada con el tiempo.*
- ***Tiempo estimado***.

Cronograma Estimado

- ***Fase 1 (Planificación)****: 3-5 días.*
- ***Fase 2 (Diseño)****: 5-8 días.*
- ***Fase 3 (Desarrollo)****: 3-7 semanas.*
- ***Fase 4 (Pruebas y Optimización)****: 2-4 semanas.*
- ***Fase 5 (Despliegue y Mantenimiento)****: 3-5 días iniciales + continuo.* ***Total****: 6-12 semanas (depende de la complejidad y recursos).*

Consideraciones Adicionales

- ***Dificultad Baja****: Usa APIs comerciales (Google Vision) para reducir el tiempo de desarrollo (4-6 semanas).*
- ***Dificultad Media/Alta****: Modelos personalizados (YOLO) requieren más tiempo para entrenamiento y optimización (8-12 semanas).*
- ***Grok 3****: Si usas Grok 3 para consultas, verifica disponibilidad en x.ai/grok o x.ai/api. La versión gratuita tiene límites, pero SuperGrok ofrece mayor capacidad.*
- ***Presupuesto****: APIs comerciales tienen costos por uso (consultar en sus sitios). Soluciones personalizadas requieren inversión en hardware (GPUs).*
- ***Escalabilidad****: Diseña para manejar picos de uso con colas y balanceo de carga desde el inicio.*

5. ***Pruebas: Distribución de la aplicación a expertos certificados para la revisión y visto bueno de la aplicación.***

   Esta es una etapa crítica para garantizar la calidad, funcionalidad, seguridad y cumplimiento de la aplicación de reconocimiento de imágenes con inteligencia artificial y capacidad de consulta. A continuación, se detallan el proceso ordenado para distribuir la aplicación a expertos certificados, obtener su retroalimentación y lograr el visto bueno, integrando los pasos previos de construcción de la aplicación. Este proceso supone que la aplicación ya está desarrollada, probada y lista para revisión, y que los expertos son profesionales con experiencia en IA, desarrollo de software, UI/UX, seguridad o dominios específicos relevantes (visión por computadora).

   Pasos para la Distribución y Revisión por Expertos Certificados

   Fase 1: Preparación para la Revisión

   Definir el Alcance de la Revisión

   - ***Tarea****: Especifica qué aspectos de la aplicación deben ser revisados por los expertos.*
   - ***Áreas clave****:*
     - ***Funcionalidad****: Precisión del reconocimiento de imágenes (clasificación, detección de objetos, OCR) y calidad de las respuestas a consultas.*
     - ***Interfaz de Usuario (UI/UX)****: Usabilidad, accesibilidad y diseño responsivo.*
     - ***Rendimiento****: Tiempo de respuesta, escalabilidad y uso de recursos.*
     - ***Seguridad****: Protección de datos de imágenes, cumplimiento normativo (GDPR, CCPA).*
     - ***Código****: Calidad, documentación y mantenibilidad.*
     - ***Dominio específico****: Si aplica (precisión en imágenes médicas para expertos en salud).*
   - ***Resultado****: Documento con objetivos de revisión y criterios de aceptación (precisión >90%, sin vulnerabilidades críticas).*
   - ***Tiempo estimado****.*

   Identificar Expertos Certificados

   - ***Tarea****: Selecciona expertos con credenciales relevantes según el enfoque del proyecto.*

- ***Perfiles sugeridos**:*
  - ***Especialistas en IA/Visión por Computadora**: Para evaluar modelos de reconocimiento (certificados en TensorFlow, experiencia con YOLO).*
  - ***Diseñadores UI/UX**: Certificados en diseño de interacción (Nielsen Norman Group).*
  - ***Especialistas en Seguridad**: Certificaciones como CISSP, CEH o experiencia en auditorías de aplicaciones.*
  - ***Ingenieros de Software**: Con experiencia en desarrollo full-stack (certificados AWS, Google Cloud).*
  - ***Expertos de Dominio**: Si la aplicación es específica (radiólogos para imágenes médicas).*
- ***Fuentes para encontrar expertos**:*
  - *Plataformas profesionales: LinkedIn, Upwork, Toptal.*
  - *Comunidades de IA: Kaggle, GitHub, conferencias como CVPR.*
  - *Organizaciones de certificación: IEEE, CompTIA, o partners de xAI (consultar x.ai para referencias).*
- ***Resultado**: Lista de 3-5 expertos con roles definidos (1 para IA, 1 para UI/UX, 1 para seguridad).*
- ***Tiempo estimado**.*

Preparar Material para la Revisión

- ***Tarea**: Compila documentación y recursos para facilitar la evaluación.*
- ***Materiales**:*
  - ***Código fuente**: Acceso controlado (repositorio privado en GitHub con permisos de solo lectura).*
  - ***Documentación**: Arquitectura de la aplicación, esquemas de base de datos, endpoints de API, y flujo de trabajo (carga de imágenes → reconocimiento → consultas).*
  - ***Guía de Uso**: Instrucciones para probar la aplicación (cómo subir imágenes, ejemplos de consultas).*
  - ***Dataset de Prueba**: Conjunto de imágenes representativas (50-100 imágenes etiquetadas) y consultas de ejemplo.*

- ***Métricas de Rendimiento**: Reportes de pruebas previas (precisión, tiempo de respuesta).*
- ***Acceso a la Aplicación**: Entorno de prueba (servidor en AWS con credenciales temporales) o APK para móviles.*

- ***Consideraciones**:*
  - *Asegura que los datos sensibles estén anonimizados o protegidos.*
  - *Firma acuerdos de confidencialidad (NDA) con los expertos.*
- ***Resultado**: Paquete de revisión completo y seguro.*
- ***Tiempo estimado**.*

Fase 2: Distribución a Expertos

Configurar un Entorno de Revisión

- ***Tarea**: Proporciona acceso controlado a la aplicación y los materiales.*
- ***Métodos**:*
  - ***Entorno en la Nube**: Despliega la aplicación en un servidor de prueba (AWS EC2, Google Cloud, Heroku) con autenticación (OAuth).*
  - ***Acceso Local**: Si la aplicación es local, proporciona máquinas virtuales o contenedores Docker.*
  - ***Frontend**: Comparte la UI a través de un enlace web (Vercel) o apps móviles (TestFlight para iOS, APK para Android).*
  - ***API**: Documenta y comparte endpoints (Swagger/OpenAPI) para pruebas.*
- ***Seguridad**:*
  - *Usa credenciales temporales y revoca acceso tras la revisión.*
  - *Limita el acceso a datos sensibles (imágenes subidas por usuarios).*
- ***Resultado**: Entorno accesible y seguro para los expertos.*
- ***Tiempo estimado**.*

Distribuir la Aplicación a Expertos

- ***Tarea**: Envía el paquete de revisión y coordina con los expertos.*
- ***Acciones**:*
  - *Envía un correo formal con:*

    - *Objetivos de la revisión.*
    - *Instrucciones para acceder al entorno (credenciales, URLs).*
    - *Materiales (documentación, dataset).*
    - *Plazo para la revisión (1-2 semanas).*
  - *Organiza una reunión inicial (virtual o presencial) para aclarar dudas.*
  - *Usa herramientas de colaboración (Slack, Microsoft Teams) para comunicación continua.*
- ***Resultado**: Expertos con acceso y claridad sobre la tarea.*
- ***Tiempo estimado**.*

Fase 3: Revisión por Expertos

Facilitar la Revisión

- ***Tarea**: Asegura que los expertos puedan evaluar la aplicación de manera efectiva.*
- ***Acciones**:*
  - ***Funcionalidad (IA)**: Los expertos prueban el reconocimiento de imágenes con el dataset proporcionado y consultas específicas ("¿Qué raza es este perro?"). Evalúan precisión, falsos positivos/negativos y robustez.*
  - ***UI/UX**: Los expertos navegan la interfaz, evalúan usabilidad (con heurísticas de Nielsen) y accesibilidad (WCAG 2.1).*
  - ***Rendimiento**: Miden tiempos de respuesta, uso de CPU/memoria y escalabilidad bajo carga (con herramientas como JMeter).*
  - ***Seguridad**: Realizan auditorías (pruebas de penetración, análisis OWASP) para identificar vulnerabilidades.*
  - ***Código**: Revisan calidad, modularidad y documentación (con herramientas como SonarQube).*
- ***Soporte**:*
  - *Responde preguntas de los expertos rápidamente.*
  - *Proporciona datasets adicionales o casos de prueba si se solicitan.*
- ***Resultado**: Retroalimentación inicial de los expertos.*
- ***Tiempo estimado**.*

Recopilar Retroalimentación

- ***Tarea****: Consolida los informes de los expertos.*
- ***Formato sugerido****:*
  - ***Resumen Ejecutivo****: Puntos clave (fortalezas, debilidades).*
  - ***Detalles por Área****: Resultados específicos ("Precisión del modelo: 92%", "UI no intuitiva en móviles").*
  - ***Recomendaciones****: Mejoras sugeridas ("Optimizar tiempo de respuesta", "Añadir autenticación de dos factores").*
  - ***Prioridad****: Clasificación de problemas (críticos, altos, bajos).*
- ***Herramientas****:*
  - *Usa plantillas (Google Docs, Notion) para estandarizar informes.*
  - *Reúne feedback en reuniones o formularios (Google Forms).*
- ***Resultado****: Informe consolidado de retroalimentación.*
- ***Tiempo estimado****.*

Fase 4: Iteración y Visto Bueno

Implementar Mejoras

- ***Tarea****: Corrige problemas identificados por los expertos.*
- ***Acciones****:*
  - ***Funcionalidad****: Ajusta modelos de IA (reentrena con más datos si la precisión es baja).*
  - ***UI/UX****: Rediseña elementos problemáticos (mejora botones en móviles).*
  - ***Rendimiento****: Optimiza código o infraestructura (añade caché con Redis).*
  - ***Seguridad****: Corrige vulnerabilidades (implementa HTTPS, sanitización de entradas).*
- ***Prioridad****: Aborda problemas críticos primero (fallos de seguridad) y luego mejoras secundarias.*
- ***Resultado****: Versión actualizada de la aplicación.*
- ***Tiempo estimado****.*

Segunda Revisión (si es necesario)

- ***Tarea****: Envía la versión actualizada a los expertos para verificar correcciones.*

- ***Acciones**:*
  - *Proporciona un informe de cambios realizados.*
  - *Permite acceso al entorno actualizado.*
  - *Solicita revisión enfocada en los problemas corregidos.*
- ***Resultado**: Confirmación de que los problemas fueron resueltos.*
- ***Tiempo estimado**.*

Obtener el Visto Bueno

- ***Tarea**: Consigue la aprobación formal de los expertos.*
- ***Acciones**:*
  - *Reúne firmas o certificados de aprobación (documento firmado o correo de confirmación).*
  - *Verifica que todos los criterios de aceptación se cumplan (precisión, usabilidad, seguridad).*
  - *Si el proyecto requiere cumplimiento normativo, obtén certificaciones específicas (ISO 27001 para seguridad).*
- ***Resultado**: Documento de aprobación firmado por los expertos.*
- ***Tiempo estimado**.*

Fase 5: Finalización

Documentar la Revisión

- ***Tarea**: Archiva los informes y aprobaciones para referencia futura.*
- ***Acciones**:*
  - *Guarda informes de revisión, cambios realizados y aprobaciones en un repositorio seguro (Google Drive, AWS S3).*
  - *Actualiza la documentación de la aplicación con los hallazgos clave.*
- ***Resultado**: Registro completo del proceso de revisión.*
- ***Tiempo estimado**.*

Preparar para Lanzamiento

- ***Tarea**: Finaliza el despliegue tras la aprobación.*
- ***Acciones**:*

- *Despliega la versión final en producción (AWS, Google Cloud, tiendas de aplicaciones).*
- *Comunica el lanzamiento a los stakeholders.*
- *Configura monitoreo continuo (New Relic) para detectar problemas post-lanzamiento.*

- ***Resultado**: Aplicación lista para usuarios finales.*
- ***Tiempo estimado**.*

Cronograma Estimado

- ***Fase 1 (Preparación)**: 6-11 días.*
- ***Fase 2 (Distribución)**: 3-5 días.*
- ***Fase 3 (Revisión)**: 1-2 semanas.*
- ***Fase 4 (Iteración y Visto Bueno)**: 1-4 semanas.*
- ***Fase 5 (Finalización)**: 3-4 días.*
- ***Total**: 4-8 semanas (depende de la complejidad y el número de iteraciones).*

Consideraciones Clave

- ***Selección de Expertos**: Asegurarse de que los expertos tengan certificaciones relevantes y experiencia en el dominio (visión por computadora para IA, OWASP para seguridad).*
- ***Seguridad de Datos**: Usar entornos de prueba aislados y NDAs para proteger datos sensibles (especialmente si las imágenes contienen información personal).*
- ***Presupuesto**: La revisión por expertos puede costar entre $50-$200/hora por experto (según Upwork, 2025). APIs comerciales (Google Vision) también incurren en costos (verificar en sus sitios).*
- ***Grok 3**: Si se utiliza Grok 3 para consultas, debe verificarse la integración en x.ai/api. La versión gratuita tiene límites, pero SuperGrok ofrece mayor capacidad (consultar x.ai/grok para detalles).*
- ***Pruebas Rigurosas**: Proporcionar datasets variados para que los expertos evalúen casos extremos (imágenes borrosas, iluminación baja).*

6. Implementación: Realización de patentado y promoción del producto finalizado.

En este proceso se incluye el patentamiento en el contexto de una aplicación de software con IA, así como las estrategias de promoción para maximizar el impacto del producto en el mercado. Los pasos consideran las normativas de patentes en Estados Unidos, con extensiones internacionales y estrategias de marketing adaptadas para aplicaciones de IA.

Fase 1: Patentado de la Aplicación

El patentado de una aplicación de software con IA, como una solución de reconocimiento de imágenes, es un proceso complejo debido a las restricciones legales sobre la patentabilidad de software y lo abstracto de los algoritmos. Las Oficinas de Patentes permiten patentar software siempre y cuando cumpla con criterios específicos de novedad, no obviedad, utilidad y elegibilidad, en seguida se detallan los pasos para patentar una aplicación.

1. Evaluar la Patentabilidad

- ***Tarea**: Determina si la aplicación o sus componentes son elegibles para una patente.*
- ***Acciones**:*
  - ***Realizar una búsqueda del arte previo**: Buscar patentes existentes y publicaciones técnicas relacionadas con reconocimiento de imágenes y consultas basadas en IA (usando USPTO Patent Public Search, Google Patents o servicios de búsqueda profesional). Esto identifica si tu invención es novedosa.*
  - ***Identificar elementos patentables**: La aplicación debe ofrecer una mejora técnica específica (un nuevo algoritmo de detección de objetos, una integración única de hardware y software, o un método novedoso para procesar consultas). Según la USPTO, las invenciones de IA deben demostrar una aplicación práctica y no ser meras ideas abstractas (evitar reclamos que solo describan algoritmos matemáticos).*
  - ***Consultar con un abogado de patentes**: Contratar un abogado especializado en IA/software para evaluar la elegibilidad bajo la decisión Alice (que requiere una mejora técnica tangible).*
  - ***Consideraciones**:*
    - *Las patentes de software suelen cubrir algoritmos específicos, interfaces de usuario únicas o integraciones con hardware. Por ejemplo, un método novedoso*

para optimizar el reconocimiento de imágenes en tiempo real podría ser patentable.

- *Si la aplicación usa Grok 3 (de xAI), hay que verificar si la innovación está en el uso de la API o en un componente propio, ya que no se puede patentar tecnologías de terceros.*

- ***Resultado**: Informe de patentabilidad y lista de elementos patentables (un algoritmo de segmentación de imágenes o un sistema de consulta contextual).*
- ***Tiempo estimado**.*

2. Preparar la Solicitud de Patente

- ***Tarea**: Redacta una solicitud de patente completa, ya sea provisional o no provisional.*
- ***Acciones**:*
  - ***Elegir el tipo de patente**:*
    - ***Patente de utilidad**: Para el proceso, algoritmo o sistema de la aplicación (e.g., método de reconocimiento de imágenes).*
    - ***Patente de diseño**: Para la interfaz de usuario visual, si es única y ornamental (e.g., diseño de la UI para mostrar resultados de reconocimiento).*
    - ***Solicitud provisional**: Menos costosa ($130-$260 en tarifas USPTO), permite reservar una fecha de prioridad mientras desarrollas la aplicación. Ideal para iteraciones rápidas.*
  - ***Redactar la solicitud**:*
    - ***Título**: Breve y específico ("Sistema de Reconocimiento de Imágenes con Consultas Contextuales").*
    - ***Resumen**: Describir la invención en ~150 palabras.*
    - ***Descripción detallada**: Explicar el proceso de reconocimiento (uso de redes neuronales convolucionales), la integración de consultas (procesamiento de lenguaje natural) y cómo se implementa (en la nube con AWS). Incluye diagramas de bloques del sistema.*
    - ***Reclamaciones**: Definir los elementos novedosos ("Un método para detectar objetos en imágenes usando un modelo de IA optimizado para baja latencia"). Evitar reclamaciones amplias que puedan ser rechazadas por ser abstractas.*

- ***Diagramas****: Incluye diagramas de flujo del proceso de reconocimiento y consulta, y esquemas de la UI.*
- ***Usar herramientas de IA (opcional)****: Herramientas como Solve Intelligence o DeepIP pueden asistir en la redacción de la solicitud, reduciendo el tiempo de redacción en un 25-35%. Asegurarse de que un abogado revise el resultado para evitar errores o "alucinaciones" de IA.*
- ***Cumplir con la USPTO****: Asegurarse de que la solicitud describe cómo la invención mejora la tecnología (mayor precisión en detección de objetos) y no solo implementa un algoritmo genérico.*

- ***Resultado****: Solicitud de patente lista para presentación.*
- ***Tiempo estimado****.*
- ***Costo estimado****: $8,000-$20,000 para una solicitud no provisional en una firma boutique; más en firmas grandes.*

3. Presentar la Solicitud

- ***Tarea****: Envíar la solicitud a la USPTO y, si aplica, a oficinas internacionales.*
- ***Acciones****:*
  - ***Presentación en la USPTO****: Usar Patent Center para cargar la solicitud electrónicamente en formato DOCX (evita la tarifa adicional de $400 por no-DOCX).*
  - ***Solicitud internacional (opcional)****: Si se busca protección global, usar el Tratado de Cooperación en Materia de Patentes (PCT) para presentar una solicitud internacional, que te da hasta 30 meses para entrar en países específicos.*
  - ***Track One (opcional)****: Pagar una tarifa adicional (~$4,000) para una revisión prioritaria en la USPTO, reduciendo el tiempo de examen a ~12 meses.*
  - ***Poder de abogado****: Designar un abogado de patentes como representante ante la USPTO.*
- ***Consideraciones****:*
  - *Solo personas naturales pueden ser inventores, no sistemas de IA (según Thaler v. Vidal). Si se utilizó Grok 3 u otra IA, hay que asegurarse de que un humano haya hecho una contribución significativa.*

- *Evitar usar una cuenta USPTO.gov compartida con herramientas de IA, ya que viola las políticas de la USPTO.*

- ***Resultado**: Solicitud presentada con número de registro.*
- ***Tiempo estimado**.*
- ***Costo estimado**: Tarifas USPTO ($100-$1,600 según el tipo de entidad) + costos legales.*

4. Responder a Acciones de la USPTO

- ***Tarea**: Gestionar las objeciones o rechazos de la USPTO durante el examen.*
- ***Acciones**:*
  - ***Revisar acciones de la oficina**: La USPTO puede emitir rechazos basados en arte previo, falta de novedad o elegibilidad (considerar la invención un "proceso mental").*
  - ***Responder con argumentos**: Trabaja con tu abogado para aclarar cómo la invención es técnica (mejora la eficiencia computacional) o modificar las reclamaciones. Herramientas como DeepIP pueden sugerir respuestas, pero deben ser validadas.*
  - ***Iterar**: Puede requerir varias rondas de respuestas (1-3 típicamente).*
- ***Resultado**: Aprobación de la patente o rechazo final.*
- ***Tiempo estimado**: 6-18 meses (o ~12 meses con Track One).*
- ***Costo estimado**: $2,000-$5,000 adicionales por respuestas legales.*

5. Mantener la Patente

- ***Tarea**: Pagar tarifas de mantenimiento para mantener la patente activa.*
- ***Acciones**:*
  - *Pagar tarifas en los años 3.5, 7.5 y 11.5 tras la concesión ($1,600-$7,400 según el tipo de entidad).*
  - *Monitorear infracciones y considera acciones legales si otros usan tu tecnología sin permiso.*
- ***Resultado**: Patente activa durante 20 años (utilidad) o 15 años (diseño).*
- ***Tiempo estimado**: Continuo.*

Fase 2: Promoción del Producto Finalizado

Una vez que la aplicación está desarrollada, revisada por expertos, desplegada y (opcionalmente) patentada, la promoción es clave para alcanzar a los usuarios objetivo y

generar impacto en el mercado. Las estrategias deben destacar las capacidades únicas de la aplicación (reconocimiento de imágenes y consultas) y aprovechar el atractivo de la IA. A continuación, se detallan los pasos.

6. Definir el Mercado Objetivo

- ***Tarea**: Identifica a los usuarios y sectores que se beneficiarán de la aplicación.*
- ***Acciones**:*
  - ***Segmentos**:*
    - ***Consumidores**: Usuarios individuales que usan la app para identificar objetos o hacer consultas (e.g., similar a Google Lens).*
    - ***Empresas**: Industrias como comercio electrónico (identificación de productos), salud (análisis de imágenes médicas) o seguridad (reconocimiento facial).*
    - ***Desarrolladores**: Si ofreces una API (como Grok 3), apunta a programadores que integren tu tecnología.*
  - ***Análisis de necesidades**: Investiga qué problemas resuelve la aplicación (e.g., identificación rápida de productos, consultas contextuales en tiempo real).*
  - ***Competencia**: Analiza competidores como Google Vision, Amazon Rekognition o Clarifai para diferenciar tu propuesta de valor (e.g., mejor UI, consultas más precisas).*
- ***Resultado**: Perfil del mercado objetivo y propuesta de valor clara.*
- ***Tiempo estimado**: 1-2 semanas.*

7. Desarrollar una Estrategia de Marketing

- ***Tarea**: Diseña una estrategia para promocionar la aplicación.*
- ***Acciones**:*
  - ***Branding**:*
    - *Crea un nombre y logotipo atractivos (e.g., "ImageQuery AI").*
    - *Destaca la IA como un diferenciador: "Identifica cualquier cosa, pregunta cualquier cosa".*
  - ***Canales de promoción**:*
    - ***Redes sociales**: Usa X, LinkedIn y TikTok para mostrar demos de la aplicación (e.g., videos cortos de reconocimiento de objetos).*

- ***Marketing de contenido****: Publica blogs o videos explicando casos de uso (e.g., "Cómo identificar plantas con nuestra app").*
- ***Publicidad digital****: Anuncios en Google Ads o Meta dirigidos a sectores específicos (e.g., comercio electrónico).*
- ***Eventos y conferencias****: Presenta la aplicación en eventos de IA como CVPR o CES.*

- ***Asociaciones****:*
  - *Colabora con empresas que puedan integrar tu tecnología (e.g., apps de comercio electrónico).*
  - *Si usas Grok 3, menciona la integración con xAI como un punto fuerte (consultar x.ai/grok para detalles).*
- ***Pruebas gratuitas****: Ofrece acceso gratuito limitado (similar al modelo de Grok 3) para atraer usuarios.*

- ***Resultado****: Plan de marketing con canales y mensajes definidos.*
- ***Tiempo estimado****: 2-3 semanas.*

8. Lanzar una Campaña de Promoción

- ***Tarea****: Ejecuta la estrategia de marketing para generar interés.*
- ***Acciones****:*
  - ***Lanzamiento suave****: Invitar a beta testers (a través de X o foros de IA) para probar la aplicación y recopilar retroalimentación.*
  - ***Lanzamiento oficial****: Anunciar la aplicación en redes sociales, comunicados de prensa y newsletters.*
  - ***Demostraciones****: Publicar videos mostrando el reconocimiento de imágenes y consultas en acción (identificar un producto y responder "¿Dónde comprarlo?").*
  - ***Testimonios****: Usar reseñas de los expertos que revisaron la aplicación para generar credibilidad.*
  - ***SEO y ASO****: Optimizar la página web y la descripción en tiendas de aplicaciones (Google Play, App Store) con palabras clave como "reconocimiento de imágenes" y "IA".*
- ***Resultado****: Mayor visibilidad y primeros usuarios.*
- ***Tiempo estimado****: 2-4 semanas.*

9. Monetización y Escalamiento

- ***Tarea****: Generar ingresos y expandir el alcance de la aplicación.*
- ***Acciones****:*
  - ***Modelos de monetización****:*
    - ***Suscripción****: Similar a SuperGrok, ofrece planes premium con mayor capacidad (más imágenes procesadas).*
    - ***Licencias****: Licenciar la tecnología a empresas (minoristas para identificación de productos).*
    - ***Publicidad****: Integrar anuncios en la versión gratuita.*
    - ***API****: Ofrecer acceso a la API para desarrolladores, como hace xAI con Grok 3 (consultar x.ai/api).*
  - ***Expansión internacional****: Adaptar la aplicación a otros idiomas y mercados, usando el PCT para protección de patentes en otros países.*
  - ***Mejoras continuas****: Usar retroalimentación de usuarios para añadir funciones (soporte para nuevos formatos de imagen).*
- ***Resultado****: Flujo de ingresos y crecimiento de usuarios.*
- ***Tiempo estimado****: Continuo.*

10. Monitorear y Ajustar la Promoción

- ***Tarea****: Evaluar el éxito de la promoción y optimiza la estrategia.*
- ***Acciones****:*
  - ***Métricas clave****: Medir descargas, usuarios activos, tasas de conversión (de gratuito a premium) y engagement en redes sociales.*
  - ***Análisis de datos****: Usar herramientas como Google Analytics o Mixpanel para rastrear el comportamiento de los usuarios.*
  - ***Ajustes****: Refinar anuncios, contenido o funciones según retroalimentación (e.g., mejorar la UI si los usuarios la encuentran confusa).*
- ***Resultado****: Estrategia de promoción optimizada y sostenible.*
- ***Tiempo estimado****: Continuo.*

Cronograma Estimado

***Fase 1 (Patentado)**: 3-24 meses (depende de la USPTO y Track One).*

- *Evaluación: 1-2 semanas.*
- *Preparación: 2-4 semanas.*
- *Presentación: 1-2 días.*
- *Examen: 6-18 meses.*
- *Mantenimiento: Continuo.*

***Fase 2 (Promoción)**: 2-3 meses iniciales + continuo.*

- *Definición de mercado: 1-2 semanas.*
- *Estrategia de marketing: 2-3 semanas.*
- *Lanzamiento: 2-4 semanas.*
- *Monetización y escalamiento: Continuo.*
- *Monitoreo: Continuo.*

***Total inicial**: 2-3 meses para promoción + tiempo de patentado (paralelo).*

Consideraciones Clave

- ***Patentado**:*
  - *Las patentes de IA son complejas debido a la decisión Alice y la necesidad de demostrar una mejora técnica. Trabajar con un abogado experimentado en IA (costos: $8,000-$25,000).*
  - *Considerar alternativas como secretos comerciales si la tecnología no es pública (algoritmos internos).*
  - *Si se usa Grok 3, no se puede patentar su tecnología, solo las innovaciones específicas.*
- ***Promoción**:*
  - *Destacar la facilidad de uso y la precisión de la aplicación en la promoción.*
  - *Aprovechar el auge de la IA (mercado proyectado en $1.8 billones para 2030) para atraer inversores y usuarios.*
  - *Considerar asociaciones con empresas de comercio electrónico o salud para casos de uso específicos.*
- ***Seguridad y cumplimiento**: Asegurarse de que la aplicación cumpla con normativas de privacidad (GDPR si opera en Europa) y proteger los datos de los usuarios.*

ETAPAS DE IMPLEMENTACIÓN

Sprint 1

Primero que nada, se tomó en cuenta el modelo a utilizar, en el caso de este proyecto se utilizó la librería de YoloV8 de Python que cuenta con un modelo pre entrenado para el reconocimiento de fracturas, el primer sprint se enfocó en adaptar este modelo para satisfacer las necesidades de mi aplicación, con consultas individuales, específicas y precisas. Estos fueron los pasos necesarios para lograrlo

- *Adaptación de la librería "YoloV8" a las necesidades del proyecto.*
- *Creación de un "Google Collab" para con el algoritmo de reconocimiento de fracturas para su fácil uso y acceso.*
- *Pruebas con radiografías reales para verificar la precisión de la aplicación.*
- *Exportación del "Google Collab" a Python para su uso en dispositivos móviles con interfaz de usuario.*

Sprint 2

Una vez funcionando el modelo a utilizar, se continuó a la fase de la creación de la interfaz de usuario, donde se utilizará el lenguaje "Dart" en el framework de Flutter como la herramienta principal para crear una aplicación multiplataforma, el sprint tuvo la siguiente estructura:

- *Creación del mapa de sitio para reconocer el número de pantallas y funciones que tendrá la aplicación.*
- *Creación de mockups una vez tenga el mapa de sitio para saber exactamente cómo voy a querer las interfaces en cuanto a diseño y colores.*
- *Desarrollo en Flutter para recrear los mockups creados*
- *Pruebas y correcciones de errores para que la aplicación quede lista antes de implementar la inteligencia artificial.*

Sprint 3

Una vez listos el Frontend y Backend, se realizó la conexión, que fue este último Sprint, la unión entre el script de Python que contiene la inteligencia artificial con las funciones para realizar las

consultas, y la interfaz generada en Flutter para así poder hacer una versión ejecutable en los dispositivos móviles o incluso subirla a las tiendas de aplicaciones

Forma de representación de la información:
La única forma en la que se presenta la información será por medio de la aplicación, siendo pantallas en las que se regresaran las mismas imágenes solicitadas o ingresadas con las fracturas existentes o de no encontrar ninguna, regresa un mensaje.

ETAPA DE INVESTIGACIÓN

De acuerdo con la metodología propuesta, el primer paso fue hacer una investigación de la tecnología YOLOV8, para ponerse en marcha en google collab, python nativo y por último, su implementación en flutter. Para esto recurrí directamente con la documentación de You only look once y de Ultralytics

3.2 ANALISIS DE IA Y YOLOV8

Como modelo de visión computacional, YOLO fue la clara opción gracias a su accesibilidad, siendo este un proyecto "Open source", y su versión 8 es una gran mejora sobre las anteriores con su menor tamaño y grandes velocidades, además de su facilidad de uso.
YOLOv8 fue lanzado por Ultralytics el 10 de enero de 2023, ofreciendo un rendimiento de vanguardia en términos de precisión y velocidad. Basándose en los avances de las versiones anteriores de YOLO, YOLOv8 introdujo nuevas funciones y optimizaciones que lo convierten en la opción ideal para diversas tareas de detección de objetos en una amplia gama de aplicaciones.

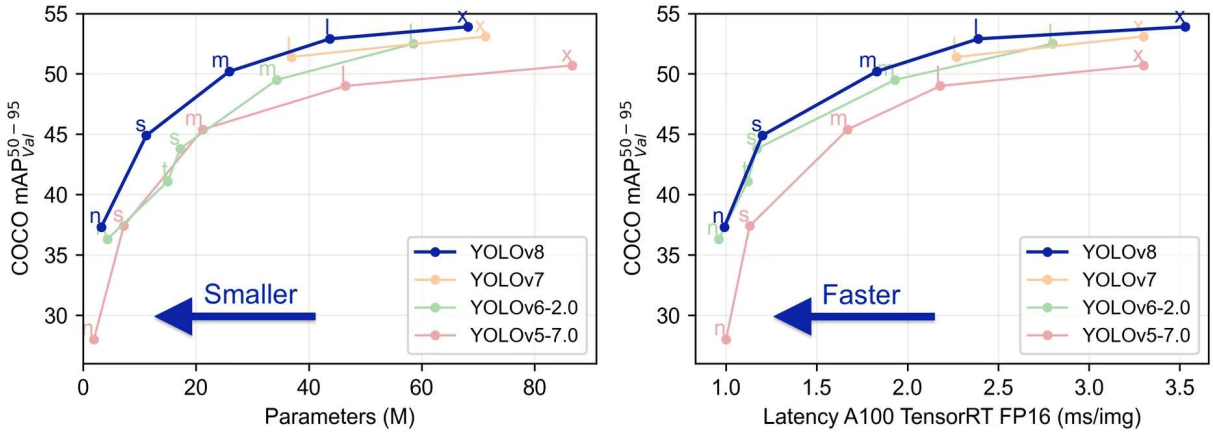

Fig1. YOLOv8 a comparación de las versiones anteriores

- *YOLOv8 emplea arquitecturas backbone y neck de última generación*
- *YOLOv8 adopta un cabezal dividido sin anclajes Ultralytics , que contribuye a una mayor precisión y a un proceso de detección más eficaz en comparación con los enfoques basados en anclajes.*
- *Centrado en mantener un equilibrio óptimo entre precisión y velocidad, YOLOv8 es adecuado para tareas de detección de objetos en tiempo real en diversas áreas de aplicación.*
- *YOLOv8 ofrece una gama de modelos pre entrenados para atender a diversas tareas y requisitos de rendimiento, lo que facilita encontrar el modelo adecuado para su caso de uso específico.[18]*

3.3 SELECCIÓN DE TECNOLOGÍAS EN DESARROLLO

Para el uso de la IA fue un proceso sencillo pues YOLOv8 fue seleccionado y por lo tanto, este es principalmente compatible con PYTHON, pero para la selección de la presentación o el lenguaje para trabajo con dispositivos móviles, un análisis más a detalle tuvo que ser realizado. esto se puede ver en la tabla siguiente:

Tabla 2. Lenguajes móviles, con ventajas y desventajas

Lenguajes	Ventajas	Desventajas
Kotlin	Seguridad Facilidad de uso Compatibilidad con android	Curva de aprendizaje inicial Exclusividad con android
Java	Soporte multiplataforma Frameworks y herramientas disponibles	Rendimiento inferior
Swift	Lenguaje principal de apple	Compatibilidad con otros dispositivos
React native	Gran compatibilidad multiplataforma Rendimiento	Curva de aprendizaje alta Dependencia en bibliotecas terceras
Flutter	Rapidez de desarrollo Compatibilidad con una gran cantidad de dispositivos Amplia colección de widgets personalizables	Estabilidad

Siendo sinceros, cualquiera de estos lenguajes pudo ser el seleccionado para la realización del proyecto, pues cada uno tiene sus ventajas y desventajas, siendo que ninguno es perfecto, pero el enfoque de este trabajo va hacia la disponibilidad de la mayor cantidad de personas y dispositivos posibles. Es por esto que Flutter fue seleccionado por la experiencia que tuve anteriormente con el framework y sus capacidades de compatibilidad con las plataformas móviles existentes.[21]

3.4 SELECCIÓN DEL CONJUNTO DE DATOS

El conjunto seleccionado fue el de FracAtlas para entrenar el modelo de inteligencia artificial, esto debido a la modernización de sus datos y su libertad de uso, con un total de 4083 imágenes estas conteniendo un aproximado de 62% de hombres y 38% de mujeres y el rango de edad es de 8 meses a 78 años, fue la cantidad de imágenes suficientes para poder entrenar la inteligencia artificial, además de que la calidad de estas es impecable[22]

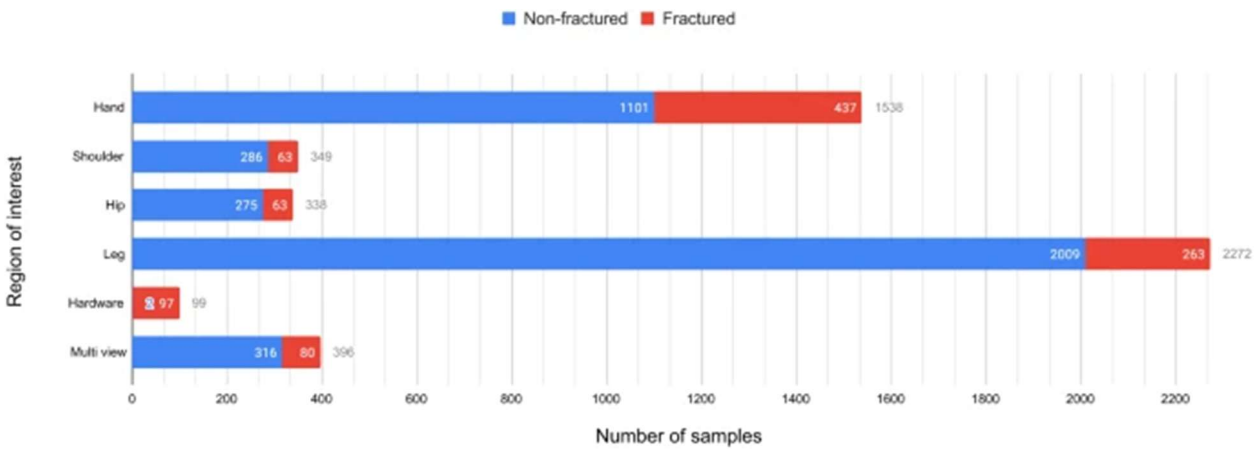

Fig. 2. Distribución de los tipos de imagenes que contiene el conjunto de datos FracAtlas

A comparación de otros conjuntos de datos, FracAtlas es muy libre con cómo se usan los datos, y tiene una gran distribución de edades, conteniendo ambos huesos sanos y con fracturas, lo que hizo que la selección de este fuera ideal para el proyecto.

3.5 DISEÑO DE ESTRUCTURA DEL PROYECTO

Para este trabajo se optó por tomar un enfoque metodológico basado en el modelo en cascada. En dicho modelo, el producto evoluciona a través de una secuencia de fases ordenadas en forma lineal. Entre sus características se encuentra que cada fase empieza cuando ha terminado la anterior, lo que requiere haber logrado los objetivos de la previa para pasar a la fase posterior. La Fig. 3. muestra un esquema que representa la metodología usada.

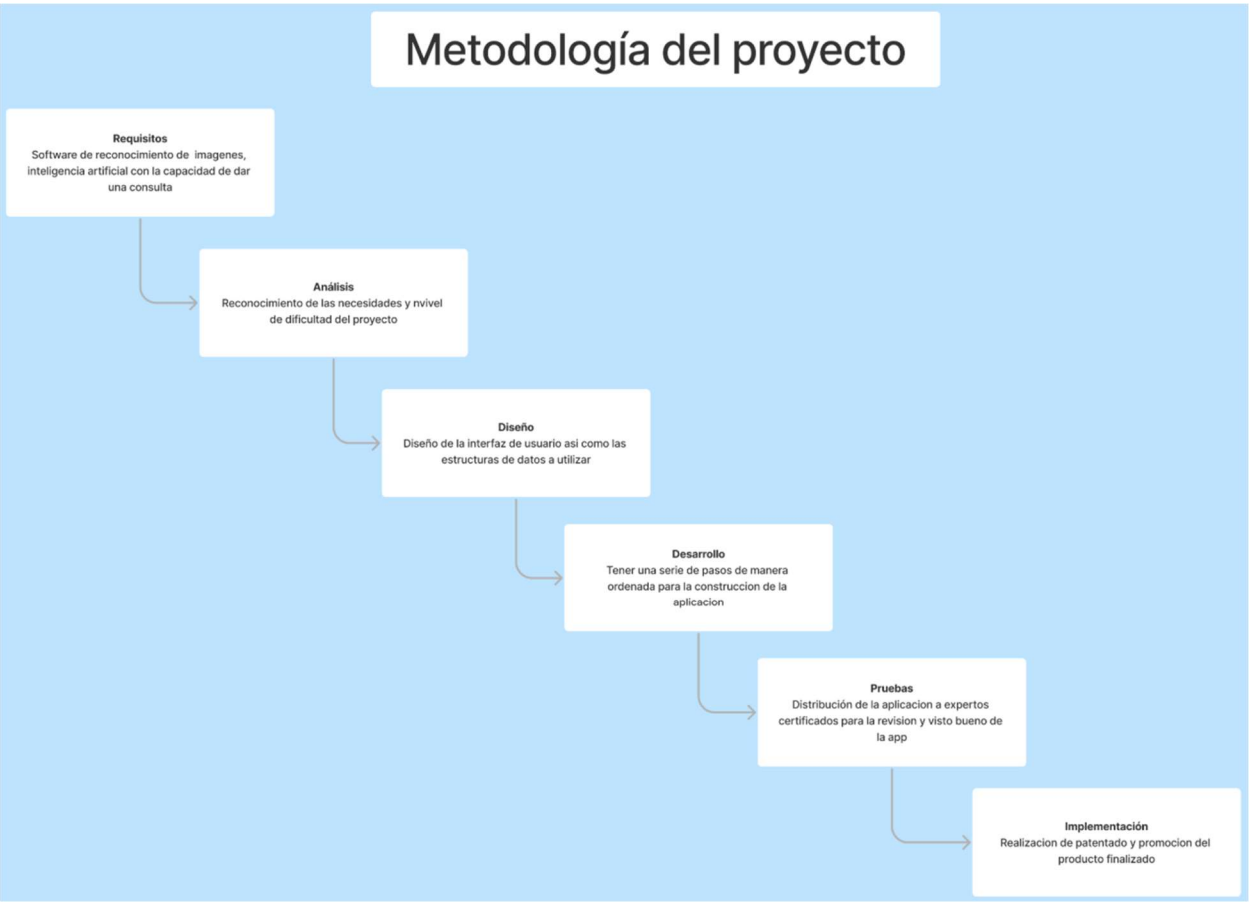

Fig. 3. Un esquema que representa la metodología en cascada aplicada en cómo se llevó a cabo este trabajo.

En la siguiente figura podemos ver los casos de uso del sistema donde solo participan el usuario y el modelo de inteligencia artificial.

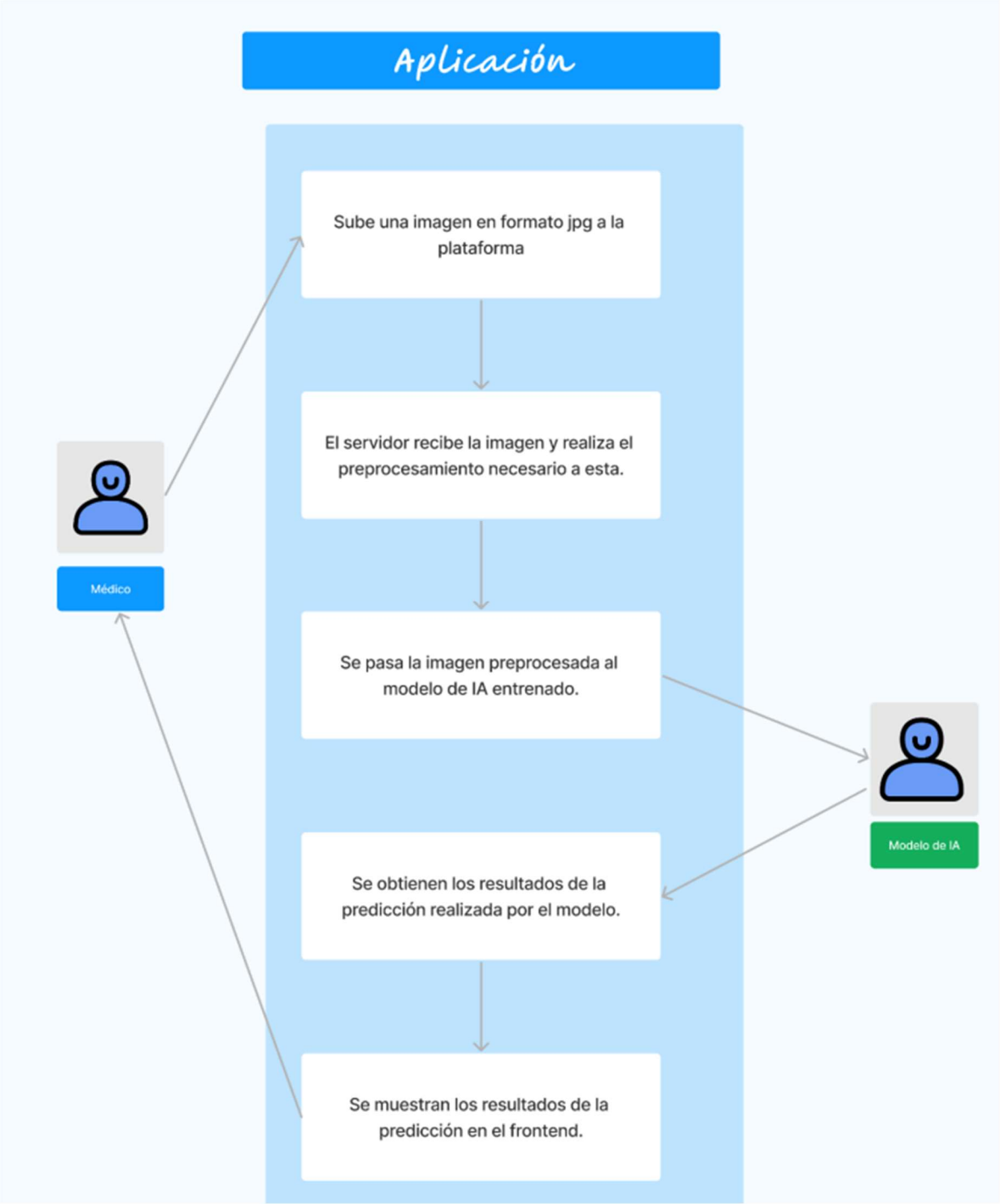

Fig. 4. Un esquema que representa los casos de uso esperados en la aplicación en una forma general

3.6 DISEÑO E INTERFACES GRAFICAS DE USUARIO

A continuación, se presenta un resumen conciso del diseño e implementación de las interfaces gráficas de usuario (GUI) para la aplicación de reconocimiento de imágenes con inteligencia artificial y capacidad de consulta, basado en los requisitos y pasos descritos previamente. Este resumen abarca los aspectos clave del diseño, desarrollo, pruebas y despliegue de la GUI, integrándola con las funcionalidades de la aplicación (carga de imágenes, visualización de resultados y consultas interactivas).

1. Objetivos y Requisitos

- ***Propósito**: Crear una GUI intuitiva, responsiva y accesible para que los usuarios (consumidores, empresas o desarrolladores) puedan cargar imágenes, visualizar resultados de reconocimiento (etiquetas, recuadros, texto extraído) y realizar consultas interactivas (e.g., "¿Qué es este objeto?").*
- ***Requisitos clave**:*
  - ***Funcionalidad**: Soporte para carga de imágenes (drag-and-drop, selección de archivos, captura en tiempo real), visualización de resultados dinámicos (e.g., recuadros de detección) y consultas en lenguaje natural.*
  - ***Usabilidad**: Interfaz simple, con retroalimentación visual clara (e.g., indicadores de carga).*
  - ***Accesibilidad**: Cumplir con WCAG 2.1 (alto contraste, soporte para lectores de pantalla).*
  - ***Compatibilidad**: Funcionar en web (Chrome, Firefox, Safari) y móviles (iOS, Android).*
  - ***Integración**: Conectar con la API del backend (endpoints como /upload_image, /query) y, si aplica, con Grok 3 para consultas (ver x.ai/api).*

2. Diseño de la GUI

- ***Estructura**:*
  - ***Encabezado**: Barra de navegación con opciones ("Subir Imagen", "Historial", "Configuración") y logotipo.*
  - ***Sección principal**:*
    - *Área de carga: Drag-and-drop o botón "Seleccionar Imagen" con vista previa.*

- *Panel de resultados: Muestra etiquetas (e.g., "Perro, 95% confianza"), recuadros de detección (usando Canvas/Konva.js) o texto extraído (OCR).*
- *Área de consultas: Campo de texto para preguntas y respuestas en formato de chat.*

- ***Pie de página**: Enlaces a soporte, términos de uso o documentación.*

- ***Estilo visual**:*
  - ***Colores**: Paleta minimalista (blanco, gris, acentos en azul).*
  - ***Tipografía**: Fuentes legibles (Roboto, Inter).*
  - ***Iconos**: Intuitivos (e.g., cámara para carga, lupa para consultas) de Material Icons.*
- ***Prototipo**:*
  - *Diseñado en **Figma** o **Adobe XD**, con flujos interactivos (carga → resultados → consulta).*
  - *Ejemplo: Imagen con recuadros rojos alrededor de objetos detectados, campo de texto abajo con "Pregunta: ¿Qué raza es este perro?" y respuesta: "Labrador Retriever".*
- ***Accesibilidad**:*
  - *Contraste 4.5:1, etiquetas ARIA, navegación por teclado.*

3. Implementación

- ***Tecnologías**:*
  - ***Web**: React + Tailwind CSS + Konva.js para visualización dinámica.*
  - ***Móvil**: Flutter o React Native con bibliotecas como image_picker.*
  - ***Backend**: Conexión con API REST (FastAPI) para procesar imágenes y consultas.*
- ***Componentes clave**:*
  - ***Carga de imágenes**:*

```
import { useDropzone } from 'react-dropzone';
function ImageUploader({ onUpload }) {
  const { getRootProps } = useDropzone({
    accept: 'image/*',
    onDrop: (files) => {
      const formData = new FormData();
      formData.append('file', files[0]);
      onUpload(formData);
```

```
    },
  });
  return <div {...getRootProps()}>Arrastra una imagen</div>;
}
```

- ***Visualización de resultados***:
  - *Usa Canvas/Konva.js para dibujar recuadros sobre la imagen.*
- ***Consultas***:
  - *Campo de texto conectado al endpoint /query para respuestas en lenguaje natural.*
- ***Integración con Grok 3*** *(si aplica): Usa la API de xAI para procesar consultas (consultar x.ai/api).*

4. Pruebas y Optimización

- ***Pruebas***:
  - ***Usabilidad****: Evalúa con 5-10 usuarios (heurísticas de Nielsen).*
  - ***Funcionalidad****: Verifica carga, visualización y consultas.*
  - ***Rendimiento****: Mide tiempos de carga (Lighthouse) y optimiza (e.g., WebP, lazy loading).*
  - ***Accesibilidad****: Usa WAVE/axe para WCAG 2.1.*
- ***Optimización***:
  - *Corrige problemas de usabilidad (e.g., botones más claros).*
  - *Reduce recursos (e.g., minimiza CSS/JS).*
  - *Añade funciones (e.g., modo oscuro, zoom).*

5. Despliegue

- ***Web****: Vercel, Netlify o AWS Amplify.*
- ***Móvil****: Google Play, App Store (TestFlight para pruebas).*
- ***Monitoreo****: Sentry para errores en producción.*

6. Cronograma y Costos

- ***Tiempo****: 3-4 semanas (diseño: 3-5 días, implementación: 1-2 semanas, pruebas: 1 semana, despliegue: 2-3 días).*
- ***Costos***:
  - *Diseño: Figma (gratis o ~$12/mes por usuario).*
  - *Desarrollo: Gratuito con herramientas open-source (React, Tailwind).*
  - *Despliegue: Vercel (gratis o $20/mes), AWS ($10-$50/mes).*

CAPÍTULO 4: IMPLEMENTACIÓN DEL SOFTWARE YOLOV8

4.1 ENTRENAMIENTO DE LA IA

*El entrenamiento del modelo **YOLOv8** utilizando el dataset **FracAtlas** en **Google Colab** es muy útil para desarrollar una aplicación de reconocimiento de imágenes, especialmente si el objetivo es detectar fracturas óseas en imágenes médicas. En seguida se muestra el proceso para entrenar YOLOv8 con FracAtlas en Google Colab, integrando los elementos mencionados (entrenamiento de IA, dataset FracAtlas, y el contexto de la aplicación con GUI y consultas interactivas).*

4.1.1 DESCRIPCIÓN DE HERRAMIENTAS CLAVE

- ***FracAtlas**: Es un dataset de imágenes médicas (radiografías) diseñado para la detección de fracturas óseas. Contiene imágenes etiquetadas con recuadros (bounding boxes) y clases ("fractura", "no fractura") para tareas de detección de objetos. El dataset está disponible en plataformas como Kaggle o GitHub, con imágenes en formatos como JPG/PNG y etiquetas en formato YOLO (txt por imagen).*
- ***YOLOv8**: Es una versión avanzada de YOLO desarrollada por Ultralytics, conocida por su precisión y eficiencia en detección de objetos, ideal para aplicaciones médicas que requieren alta precisión en entornos críticos.*
- ***Google Colab**: Proporciona acceso gratuito a GPUs (NVIDIA T4) para acelerar el entrenamiento, con soporte para Python y bibliotecas como Ultralytics.*
- ***Objetivo**: Entrenar un modelo YOLOv8 para detectar fracturas en imágenes de FracAtlas, integrarlo con la GUI descrita (carga de imágenes, visualización de recuadros, consultas) y, opcionalmente, usar Grok 3 para consultas interactivas ("¿Qué tipo de fractura es esta?").*

- 4.1.2 PASOS PARA EL ENTRENAMIENTO
- 4.1.2.1 Configurar Google Colab

Preparar el entorno en Google Colab para el entrenamiento.

- *Crear un nuevo notebook en Google Colab.*

- *Habilitar la GPU: Ir a Entorno de ejecución → Cambiar tipo de entorno de ejecución → Selecciona GPU (T4).*
- *Instalar dependencias necesarias:*

```
!pip install -U ultralytics
```

- *Verificar la instalación:*

```
import ultralytics
ultralytics.checks()
```

4.1.2.2 Obtener y Preparar FracAtlas

Descargar y organizar el dataset FracAtlas en formato YOLO.

- ***Descargar FracAtlas**:*
  - *Si está en Kaggle, usa la API de Kaggle:*

```
!pip install kaggle
!mkdir ~/.kaggle
!echo '{"username":"TU_USUARIO","key":"TU_API_KEY"}' > ~/.kaggle/kaggle.json
!chmod 600 ~/.kaggle/kaggle.json
!kaggle datasets download -d ali-hussain/fractional-atlas-fracture-detection
!unzip fractional-atlas-fracture-detection.zip -d /content/FracAtlas
```

  - *O descargarlo manualmente desde el repositorio (e.g., GitHub: https://github.com/axium/FracAtlas) y subirlo a Google Drive.*
- ***Estructura del dataset**:*
  - *FracAtlas incluye imágenes (e.g., images/) y etiquetas (e.g., labels/) con archivos txt en formato YOLO: clase x_centro y_centro ancho alto (normalizados).*
  - *Divide el dataset en train (70%), val (20%) y test (10%):*

```
FracAtlas/
├── images/
│   ├── train/
│   ├── val/
```

```
│   ├── test/
├── labels/
│   ├── train/
│   ├── val/
│   ├── test/
```

- ***Crear archivo*** *data.yaml:*

```
train: /content/FracAtlas/images/train
val: /content/FracAtlas/images/val
test: /content/FracAtlas/images/test
nc: 2  # Número de clases (e.g., "fractura", "no fractura")
names: ['fractura', 'no fractura']  # Nombres de clases
```

- ***Aumento de datos****: FracAtlas puede beneficiarse de aumentos como rotaciones, cambios de contraste o recortes para mejorar la generalización en radiografías (configuradas automáticamente en YOLOv8 con Mosaic).*

4.1.2.3 Entrenar el Modelo YOLOv8

Entrenar YOLOv8 usando FracAtlas y un modelo preentrenado.

- ***Cargar modelo preentrenado****:*
  - *Usar yolov8n.pt para nano o ligero, y yolov8m.pt para mediano ó más preciso, para transfer learning:*

```
from ultralytics import YOLO
model = YOLO("yolov8n.pt")  # Descarga automática desde Ultralytics
```

- ***Configurar entrenamiento****:*
  - *Ajustar hiperparámetros como épocas, tamaño de imagen, lote, etc.:*

```
model.train(
    data="/content/FracAtlas/data.yaml",
    epochs=50,  # Ajustar según dataset (50-100 para FracAtlas)
    imgsz=640,  # Resolución típica para radiografías
```

```
    batch=16,    # Ajustar según GPU (T4 soporta ~16-32)
    device=0     # GPU
)
```

- ***Monitoreo**:*
  - *Colab muestra métricas en tiempo real (mAP@0.5, pérdida).*
  - *Los pesos se guardan en /runs/detect/train/weights/best.pt.*
- ***Consideraciones para FracAtlas**:*
  - *Las radiografías tienen bajo contraste; usa aumentaciones de brillo/contraste.*
  - *FracAtlas puede tener clases desbalanceadas (más "no fractura" que "fractura"); ajustar pesos de pérdida si es necesario (cls en hiperparámetros).*

4.1.2.4 Validar y Probar el Modelo

Evaluar el rendimiento del modelo en el conjunto de validación y prueba.

- ***Validación**:*

```
model.val(data="/content/FracAtlas/data.yaml")
```

  - *Mide mAP@0.5 (IoU=0.5) y mAP@0.5:0.95. Para FracAtlas, espera mAP@0.5 > 0.7 en clases como "fractura".*
- ***Pruebas**:*
  - *Prueba con imágenes de prueba:*

```
results = model.predict(source="/content/FracAtlas/images/test", save=True)
```

  - *Verifica recuadros y clases en /runs/detect/predict.*
  - *Evalúa casos extremos (e.g., fracturas pequeñas, imágenes borrosas).*
- ***Optimización**:*
  - *Si el mAP es bajo, aumenta épocas (hasta 100) o añade más imágenes etiquetadas.*
  - *Usa SAHI (Slicing Aided Hyper Inference) para mejorar la detección de fracturas pequeñas:*

```
pip install sahi
yolo predict model=best.pt source=test.jpg slice=True
```

4.1.2.4 Integrar con la GUI

Conectar el modelo YOLOv8 entrenado con la interfaz gráfica descrita es decir, cargar las imágenes, visualización de resultados y consultas.

- ***API del backend**:*
  - *Configura un endpoint en FastAPI para procesar imágenes:*

```
from fastapi import FastAPI, UploadFile
from ultralytics import YOLO
import cv2
import numpy as np

app = FastAPI()
model = YOLO("best.pt")

@app.post("/upload_image")
async def upload_image(file: UploadFile):
    image = cv2.imdecode(np.frombuffer(await file.read(), np.uint8), cv2.IMREAD_COLOR)
    results = model.predict(image)
    return {
        "image_id": "img123",
        "objects": [
            {
                "class": model.names[int(cls)],
                "confidence": float(conf),
                "bounding_box": {
                    "x_min": float(box[0]),
                    "y_min": float(box[1]),
                    "x_max": float(box[2]),
                    "y_max": float(box[3])
                }
            } for box, conf, cls in results[0].boxes.data
        ]
    }
```

- ***Visualización en la GUI**:*
  - *Usa Konva.js para dibujar recuadros en la imagen (como en el resumen de GUI):*

```
import React from 'react';
import Konva from 'konva';

function ImageCanvas({ image, results }) {
  React.useEffect(() => {
    const stage = new Konva.Stage({ container: 'canvas', width: image.width, height: image.height });
    const layer = new Konva.Layer();
    stage.add(layer);

    results.objects.forEach(obj => {
      const rect = new Konva.Rect({
        x: obj.bounding_box.x_min,
        y: obj.bounding_box.y_min,
        width: obj.bounding_box.x_max - obj.bounding_box.x_min,
        height: obj.bounding_box.y_max - obj.bounding_box.y_min,
        stroke: 'red',
        strokeWidth: 2,
      });
      layer.add(rect);
      layer.add(new Konva.Text({
        x: obj.bounding_box.x_min,
        y: obj.bounding_box.y_min - 20,
        text: `${obj.class} (${(obj.confidence * 100).toFixed(1)}%)`,
        fontSize: 16,
        fill: 'red'
      }));
    });
  }, [image, results]);

  return <div id="canvas" />;
}
```

- ***Consultas interactivas***:
  - *Integrar con Grok 3 para responder preguntas sobre fracturas:*

```
import requests
def query_image(image_id, question):
    response = requests.post("https://x.ai/api/query", json={"image_id": image_id, "query": question})
```

```
return response.json()["answer"]
```

- *Ejemplo: El usuario sube una radiografía, YOLOv8 detecta una fractura, y Grok 3 responde a "¿Qué tipo de fractura es esta?" con "Posible fractura de Colles, consulta a un médico".*

4.1.2.5 Desplegar el Modelo

Implementar el modelo en producción.

- ***Exportar modelo**:*

```
yolo export model=best.pt format=onnx
```

- ***Despliegue**:*
  - ***Web**: Usa AWS/Vercel con FastAPI.*
  - ***Móvil**: Convierte a CoreML (iOS) o TensorFlow Lite (Android).*
- ***Monitoreo**: Configurar Sentry para errores en producción.*

CAPÍTULO 5: PRUEBAS Y RESULTADOS DEL YOLOV8

En esta sección se presentan los resultados obtenidos durante el diseño de la app mobil presentado en este trabajo. Incluímos los resultados de las pruebas de rendimiento del modelo de segmentación recuperado de [1]. Para asegurarse que el conjunto de datos presentado en [1] es viable para el entrenamiento de algoritmos de aprendizaje máquina, los autores entrenaron los modelos de visión por computadora YOLOv8s y YOLOv8s-seg para probar el conjunto de datos para localización de fracturas y segmentación, respectivamente. Las imágenes del conjunto de datos fueron separadas aleatoriamente en 80% (574) para entrenamiento, 12% (82) para validación y 8% (61) para pruebas. Esta división fue utilizada para probar los modelos de localización y segmentación, ambos modelos fueron pre entrenados con COCO40 y se ejecutaron por 30 epochs. El tamaño de entrada para ambos casos fue de 600 pixeles. El rendimiento para el modelo de segmentación, que fue el que se utilizó para el prototipo del sistema presentado en este trabajo, tuvo una precisión de 71.8% y un recuerdo de 60.7%. Mediante una serie de pruebas realizadas a un prototipo del diseño propuesto del servidor se obtuvieron los siguientes resultados:

1. *El servidor es capaz de recibir múltiples imágenes en una misma solicitud.*
2. *La comunicación entre el servidor y el modelo entrenado permite realizar una predicción de una o más imágenes médicas en formato JPEG recibidas en una petición al servidor.*
3. *Las imágenes y archivos de formatos diferentes a los permitidos son ignorados en las predicciones..*
4. *Una solicitud al servidor correcta resulta en una o más predicciones, dependiendo del número de imágenes con un formato aceptado en la solicitud.*

En cuestiones de privacidad y seguridad, las decisiones de limitar los formatos de imágenes médicas y no solicitar ni almacenar datos de los pacientes han resultado en un sistema que no representa un riesgo de difusión o mal uso de datos sensibles.

5.1 Prototipo

Las siguientes imágenes forman son

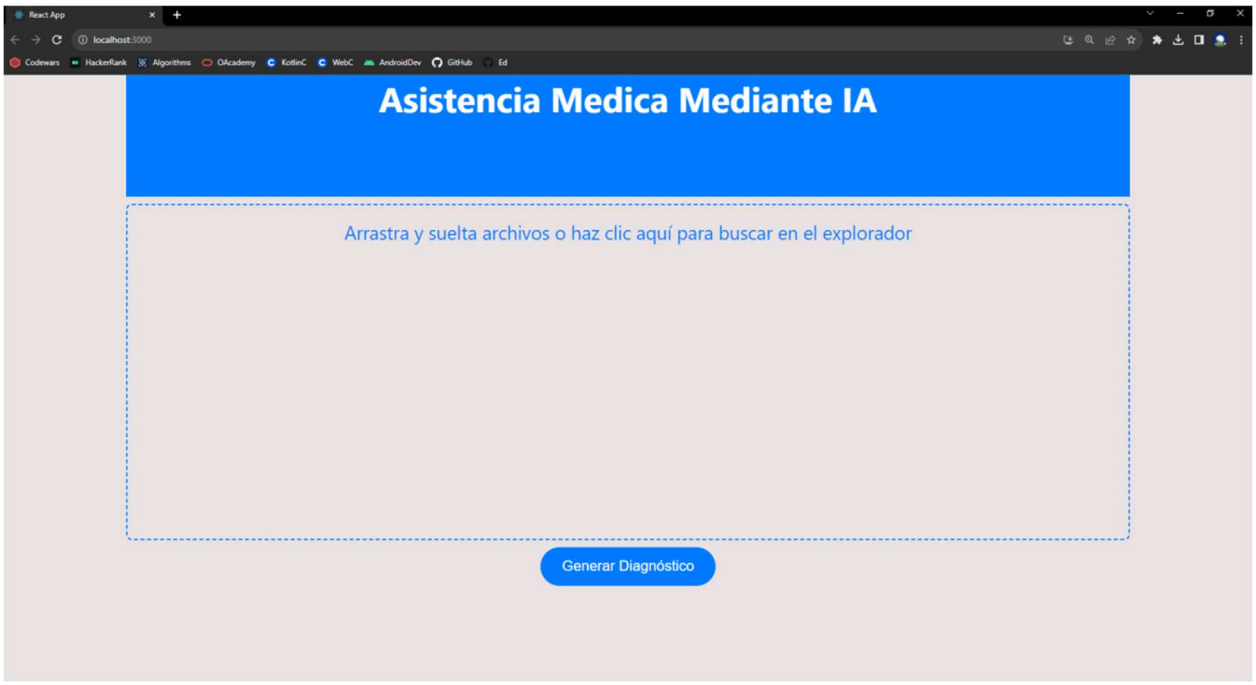

Fig. 5. Pantalla principal en espera del ingreso de una imagen

La pantalla principal nos muestra el espacio donde se cargan las imágenes a procesar, por el momento solo puede cargar una imagen a la vez.

Fig. 6. Carga de la imagen de una pierna con dislocación para su análisis

Una vez cargada la imagen solo se debe pulsar el botón de la generación del diagnóstico.

Fig. 7. Resultado del análisis de la figura 5 donde se marca el origen de la fractura.

Despues de procesamiento interno, la aplicación lanzará un resultado del diagnóstico con su probabilidad de acertar

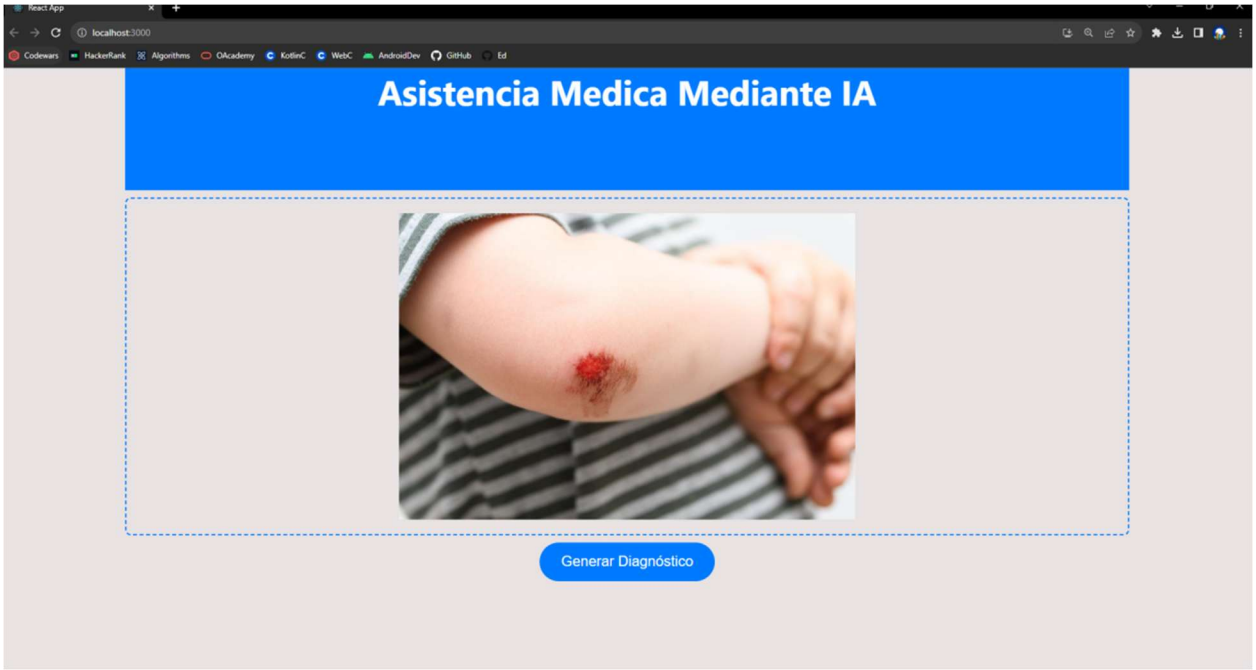

Fig. 8. Carga de la imagen de un raspon común para su análisis

En un caso extremo de no presentar una radiografía o imagen válida, se procesa pero como no es nada parecido a una radiografía, simplemente lo desecha y mostrará el mensaje

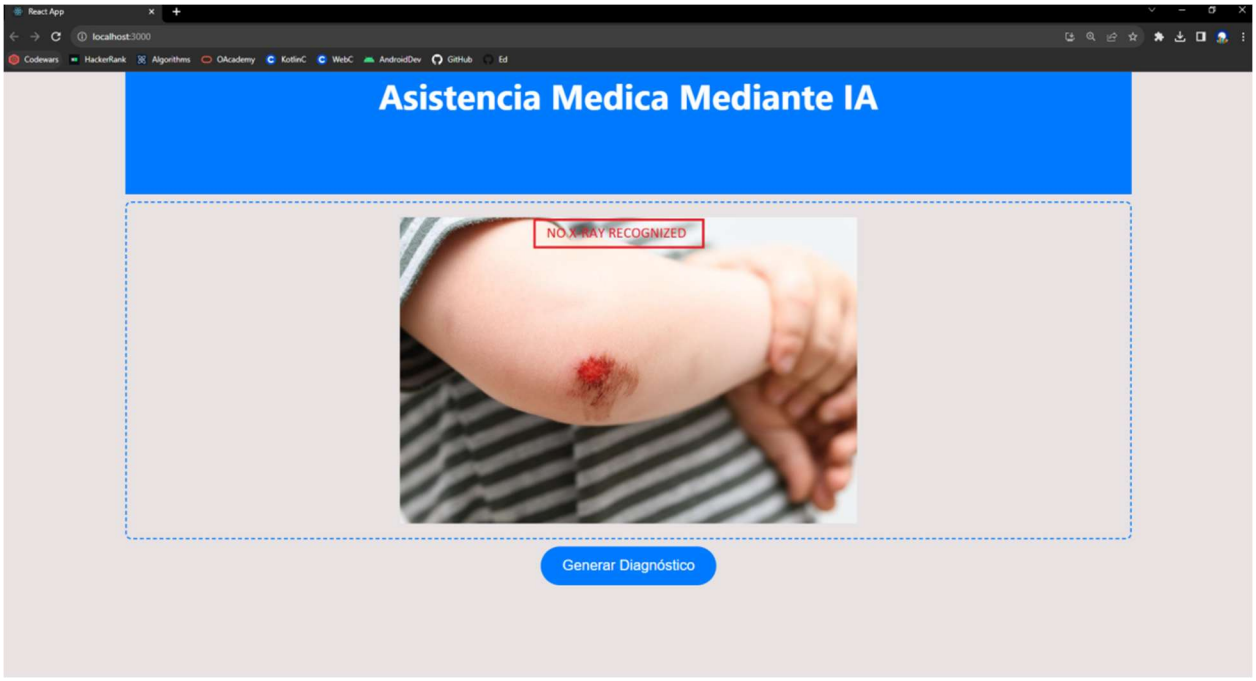

Fig. 9. Resultado del análisis de la figura 7 en donde se muestra como resultado que la imagen no es una radiografía.

Como la imagen presentada no es una radiografía, muestra el mensaje de que no se cuenta con una radiografía a analizar.

Fig. 10. Cargado de una radiografía de una mano sana para su análisis.

En caso de cargar una imagen de una estructura sana la aplicación lo detectará y mandará un mensaje de respuesta como lo veremos en la imagen siguiente.

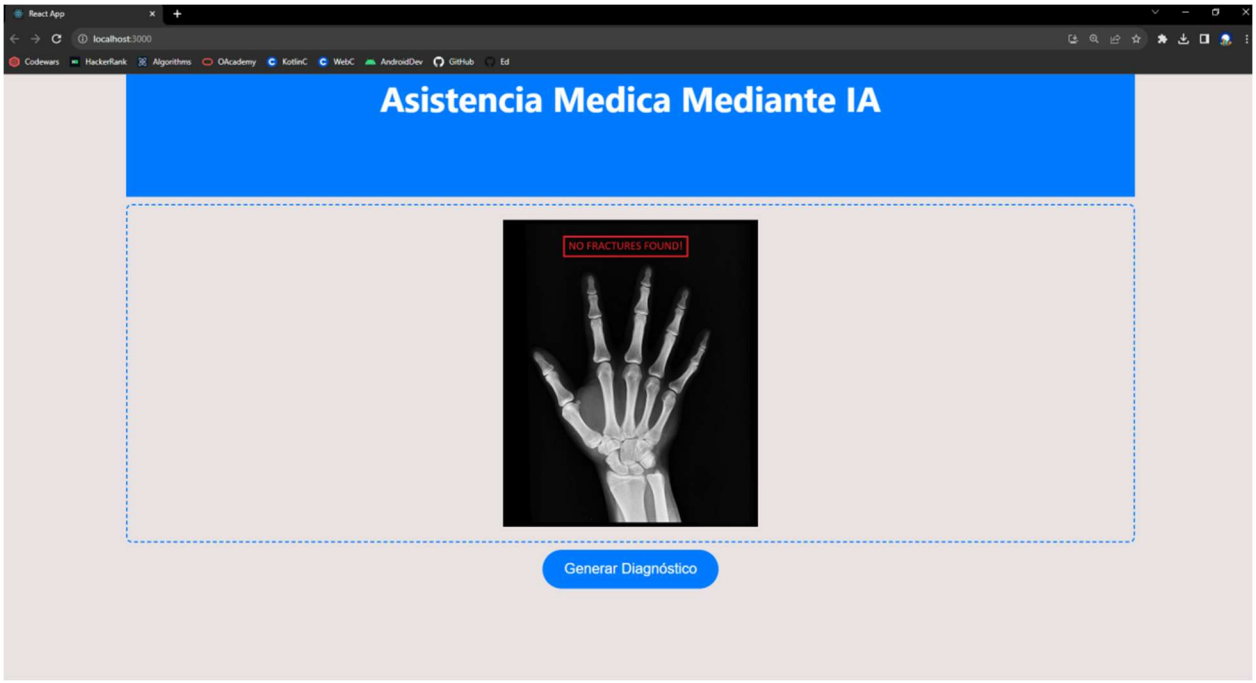

Fig. 11. Debido a que la radiografía muestra una mano sin fracturas, se muestra un mensaje mostrando esto.

Como se ve en la figura 10, la mano sana obtiene como respuesta que no se encontraron fracturas

5.2 Aceptacion de los usuarios

Copias de la aplicación fueron brindadas a 30 usuarios para que le dieran uso a la aplicación y así pudieran contestar una encuesta para poder brindar el grado de aceptación de la aplicación.

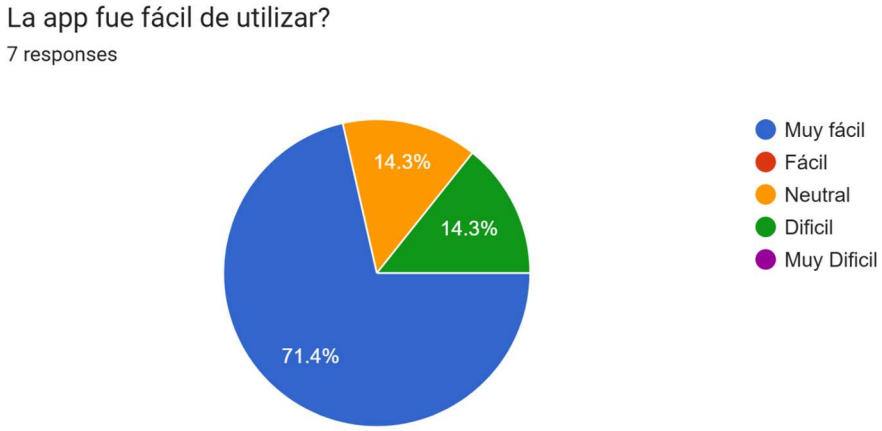

Fig. 12. Respuestas de la encuesta sobre la facilidad de uso

Este dato nos puede ayudar a entender si hubo un grado de error humano a la hora de utilizar la aplicación.

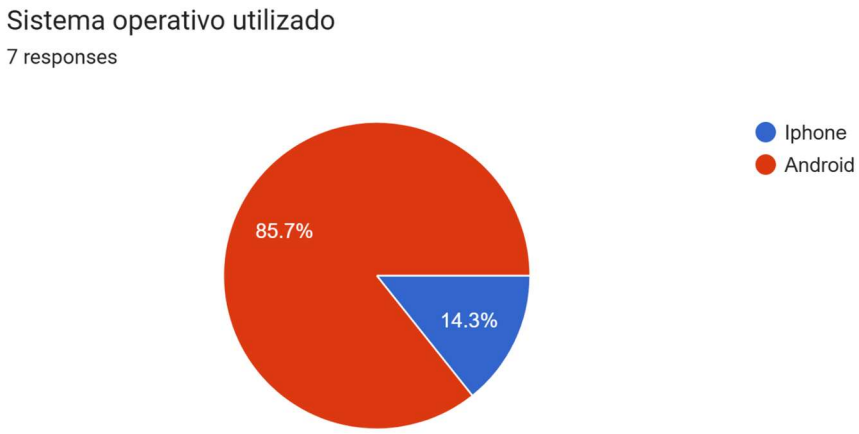

Fig. 13. Respuestas de la encuesta sobre el sistema operativo utilizado

Al igual que el anterior, puede que el sistema operativo esté relacionado al grado de aceptación así que esta pregunta fue implementada.

Fig. 14. Respuestas de la encuesta sobre el número de imágenes que se utilizó

Podemos concluir que al menos 50 imágenes fueron cargadas en la aplicación, número que será útil en el siguiente resultado

Fig. 15. Respuestas de la encuesta sobre el promedio de aciertos que se obtuvo

Tenemos un promedio de aciertos del noventa por ciento, siendo que el número de puntos negativos pudo ser gracias a la incomprensión de la app o de algún error con sistemas operativos diferentes

CAPÍTULO 6: CONCLUSIÓN

No cabe duda que el futuro de la tecnología y la computación son las inteligencias artificiales, el punto es saber utilizarlas y desarrollando para ayudar en ciertos sectores, y aunque por el momento es imposible que una computadora haga el mismo trabajo que un profesional del área de medicina, por ese camino vamos pues, la inteligencia artificial con ayuda de otras tecnologías podría llegar a hacer grandes cosas, por ahora, lo correcto es ir dando un paso a la vez para intentar llegar a este futuro deseado.

Este trabajo me ayudó a comprender más la inteligencia artificial y sus casos de uso, siendo estos casi ilimitados, uno puede agobiarse por las nuevas tecnologías, lo importante es poder adaptarse a estas para poder aprovecharlas. La emergencia de chatbots mucho más inteligentes, reconocimiento de imágenes para búsquedas, y generación de multimedia hace pensar que ya no se puede crear nada nuevo, pero la realidad es que la inteligencia artificial apenas está llegando a su pico de interés, y es el momento apropiado para estudiarla y seguirla desarrollando.

La carrera nos preparó para estos casos, donde no se nos enseñó un lenguaje o a programar en realidad, si no a resolver problemas, con un enfoque de ingeniería para llevar un proceso metódico y científico a nuestras soluciones. La investigación y la implementación de soluciones son lo que nos reconoce como ingenieros.

REFERENCIAS BIBLIOGRÁFICAS

1. *Petinaux, B., Bhat, R., Boniface, K., & Aristizabal, J. (2011). Accuracy of radiographic readings in the emergency department. The American journal of emergency medicine, 29(1), 18-25.*

2. *Thian, Y. L., Li, Y., Jagmohan, P., Sia, D., Chan, V. E. Y., & Tan, R. T. (2019). Convolutional Neural Networks for Automated Fracture Detection and Localization on Wrist Radiographs. Radiology. Artificial intelligence, 1(1), e180001. https://doi.org/10.1148/ryai.2019180001.*

3. *Pinto, A., Reginelli, A., Pinto, F., Lo Re, G., Midiri, F., Muzj, C., Romano, L., & Brunese, L. (2016). Errors in imaging patients in the emergency setting. The British journal of radiology, 89(1061), 20150914. https://doi.org/10.1259/bjr.20150914.*

4. *Cataldi, Z., Lage, F., Pessacq, R., & García Martínez, R. (1999, August). Ingeniería de software educativo. In Proceedings del V Congreso Internacional de Ingeniería Informática (pp. 185-199).*

5. *Wang, L., Lin, Z. Q., & Wong, A. (2020). "COVID-Net: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases from Chest X-Ray Images." Scientific Reports, 10(1), 19549.*

6. *Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). "nnU-Net: A Self-Configuring Method for Deep Learning-Based Biomedical Image Segmentation." Nature Methods, 18(2), 203-211.*

7. *Kazemifar, S., Nguyen, D., & Xing, L. (2020). "Synthetic CT Generation from MRI Using Improved Generative Adversarial Networks." Physics in Medicine & Biology, 65(23), 235015.*

8. *Liu, F., Chen, X., & Zhou, Z. (2023). "Vision-Language Models for Automated Radiology Report Generation." Medical Image Analysis, 85, 102745.*

9. *Lambin, P., Leijenaar, R. T. H., & Deist, T. M. (2022). "Radiomics: The Bridge Between Medical Imaging and Personalized Medicine." Nature Reviews Clinical Oncology, 19(4), 263-273.*

10. *Varettoni, A. (2025). "The Impact of Teleradiology in Latin America: Challenges and Opportunities." Dedalus LATAM.*

11. *Santos, M. K., & Ferreira, J. R. (2023). "Artificial Intelligence in Breast Cancer Screening: A Latin American Perspective." Revista Brasileira de Radiologia, 56(3), 123-130.*

12. *Proyecto Amazonía Venezolana (2022). "Using Satellite Imagery and Machine Learning to Track Environmental Health Impacts." Global Investigative Journalism Network.*

13. *Consultor Salud (2024). "La Inteligencia Artificial mejora los servicios de radiología en México." consultorsalud.com.mx.*

14. *Roldán-Gómez, F. J., et al. (2020). "Telemedicina como instrumento de consulta cardiológica durante la pandemia COVID-19." Cardiovasc Metab Sci, 31(Suppl 3), 259-264.*

15. *Velarde-Loya, M., Sáenz-Nieto, J. C., Ortega-Flores, J. M., & Gamboa-Torres, J. A. (2021). "Diagnóstico prenatal de síndrome de Jarcho-Levin. Reporte de caso." Anales de Radiología, México.*

16. *Jocher, G., Chaurasia, A., & Qiu, J. (2023). YOLO by Ultralytics (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics.*

17. *Grinberg, M. (2018). Flask web development: developing web applications with python. O'Reilly Media, Inc.*

18. *FigMa: The Collaborative Interface Design Tool. (n.d.). Figma. Recuperado el 14 de noviembre del 2023 de https://www.figma.com/*

19. *React. (n.d.). Facebook. Recuperado el 14 de noviembre del 2023 de https://es.react.dev/*

20. *Medicarama. (n.d.). La importancia de la interpretación de las radiografías. Medicarama. Recuperado Mayo 15, 2024, de https://www.medicarama.com/la-importancia-de-la-interpretacion-de-las-radiografias/#:~:text=La%20importancia%20de%20la%20interpretaci%C3%B3n%20radiol%C3%B3gica&text=M%C3%A1s%20all%C3%A1%20de%20ser%20una,r%C3%A1pida%20y%20complicaciones%20potencialmente%20graves.*

21. *Universidad Autónoma de Guadalajara. (2023, November). Inteligencia artificial: Una herramienta en el diagnóstico de enfermedades. UAG. Recuperado Mayo 15, 2024, de https://www.uag.mx/es/mediahub/inteligencia-artificial-una-herramienta-en-el-diagnostico-de-enfermedades/2023-11.*

22. *Radiological Society of North America. (2023, March). AI identifies normal, abnormal X-rays. RSNA. Retrieved June 7, 2024, from https://www.rsna.org/news/2023/march/ai-identifies-normal-abnormal-xrays.*

23. *Sharma, P., Allison, M. A., Heiss, G., Malhotra, R., Solomon, A. J., Goel, M., & Criqui, M. H. (2014). Leg muscle mass assessed by computed tomography and incident peripheral arterial disease: The multi-ethnic study of atherosclerosis (MESA). Vascular Medicine, 19(3), 192-199. https://doi.org/10.1177/1358863X14533178*

24. *Red Seguridad. (2012, February 29). La importancia y la necesidad de proteger la información sensible. Red Seguridad. Recuperado Mayo 25, 2024, de https://www.redseguridad.com/especialidades-tic/proteccion-de-datos/la-importancia-y-la-necesidad-de-proteger-la-informacion-sensible_20120229.html#:~:text=El%20uso%20de%20estas%20tecnolog%C3%ADas,incumplimiento%20de%20la%20pol%C3%ADtica%20de*

25. *https://www.iberdrola.com/innovacion/historia-inteligencia-artificial#:~:text=En%201943%20Warren%20McCulloch%20y,today%C3%ADa%20no%20exist%C3%ADa%20el%20t%C3%A9rmino.*

26. *https://es.wikipedia.org/wiki/Historia_de_la_inteligencia_artificial*

27. *https://www.elternativa.com/historia-inteligencia-artificial/*

28. *https://en.wikipedia.org/wiki/You_Only_Look_Once*

29. *https://docs.ultralytics.com/es/models/yolov8/*

30. *https://revistamedicina.net/index.php/Medicina/article/view/1648/2138*

31. *https://www.bioplusmedicalcare.com/publicaciones/actualidad/crean-metodo-que-permite-la-lectura-de-rayos-x-con-ia*

32. *https://teclab.edu.ar/tecnologia-y-desarrollo/lenguajes-de-programacion-para-aplicaciones-moviles/*

33. *https://www.nature.com/articles/s41597-023-02432-4*