OpenEye: Un vistazo a lo que no puedes mirar.

Estudiantes:

Andrés Felipe Borbón Sierra

Jefry Daniel Vargas Sierra

Joseph Alejandro Párraga Jaime

Docente: John Jairo Porras

Universidad EAN

Facultad de Ingeniería

Programa de Ingeniería de Sistemas

IFP00153: Proyecto de Grado

Bogotá, Colombia

31 de mayo de 2025

Resumen

Este proyecto propone el desarrollo de un asistente virtual con inteligencia artificial para personas con discapacidad visual, permitiendo la conversión de texto e imágenes en audio en tiempo real. La solución integrará visión artificial, reconocimiento óptico de caracteres (OCR) y síntesis de voz para ofrecer acceso eficiente a la información escrita.

El diseño garantizará accesibilidad, usabilidad y compatibilidad con distintos dispositivos, cumpliendo estándares internacionales. Con esta herramienta, se busca mejorar la autonomía, inclusión digital y calidad de vida de las personas con baja o nula visión, proporcionando una alternativa accesible y eficiente a las soluciones comerciales existentes.

Palabras Clave: discapacidad visual, asistente virtual, inteligencia artificial, visión artificial, síntesis de voz, OpenAI, OCR, accesibilidad.

Tabla de Contenido

Pág.

Índices de tablas

Pág.

Índices de figuras

Pág.

Introducción

En la actualidad, la accesibilidad tecnológica constituye un eje fundamental para promover la inclusión social en entornos digitales. Las personas con discapacidad visual, en particular, enfrentan múltiples barreras en el acceso a la información escrita y la interacción con interfaces gráficas. En este contexto, las herramientas basadas en inteligencia artificial (IA) han demostrado un gran potencial para reducir estas brechas mediante soluciones adaptativas e inclusivas.

Este proyecto de grado presenta OpenEye, un asistente virtual orientado a personas con baja o nula visión, diseñado para facilitar la interpretación de contenido visual mediante captura de pantalla, procesamiento de imágenes con reconocimiento óptico de caracteres (OCR) y generación de descripciones en lenguaje natural, convertidas a voz en tiempo real mediante síntesis de voz. La solución está desarrollada en Python, implementa modelos de lenguaje de OpenAI y se apoya en una interfaz gráfica accesible que cumple con principios de usabilidad y compatibilidad multiplataforma.

Problema

Las personas con discapacidad visual enfrentan barreras significativas en el acceso a la información escrita, lo que impacta su autonomía, educación, empleo y participación en la sociedad. Aunque existen herramientas como lectores de pantalla y asistentes de voz, muchas presentan limitaciones técnicas y económicas, ya sea por su alto costo, falta de compatibilidad con distintos dispositivos o interfaces poco intuitivas.

En un mundo donde la digitalización avanza rápidamente, es fundamental garantizar que todas las personas, independientemente de sus capacidades visuales, tengan acceso equitativo a la información. Sin embargo, la mayoría de las soluciones actuales dependen de tecnologías preconfiguradas que no siempre reconocen con precisión documentos físicos, imágenes o texto en entornos dinámicos, dificultando la interacción fluida con el entorno.

Este proyecto busca desarrollar un asistente virtual basado en inteligencia artificial que convierta contenido visual y textual en audio en tiempo real, garantizando una experiencia accesible, eficiente y adaptable. La investigación se enfocará en responder la siguiente pregunta:

¿Cómo diseñar e implementar un asistente virtual que permita a personas con discapacidad visual acceder a la información escrita en entornos físicos y digitales, utilizando tecnologías de visión artificial, OCR y síntesis de voz, optimizando precisión y usabilidad?

El estudio analizará las necesidades y desafíos de los usuarios, evaluará el rendimiento del sistema en distintos escenarios y propondrá un modelo accesible que contribuya a la inclusión digital, mejorando la calidad de vida de las personas con discapacidad visual. Cabe resaltar que, en Colombia, el marco legal respalda esta necesidad: la Ley 1680 de 2013 establece que las personas con discapacidad visual tienen derecho al acceso autónomo e independiente a la

información, garantizando así su inclusión social, educativa, cultural y laboral (Congreso de Colombia, 2013).

Objetivos

Objetivo General

Desarrollar un asistente virtual basado en inteligencia artificial que convierta imágenes y texto en audio para ayudar a personas con discapacidad visual a acceder a la información de manera efectiva y autónoma.

Objetivos Específicos

- Reconocer las principales necesidades y desafíos que enfrentan las personas con discapacidad visual en el acceso a la información escrita y visual.
- Desarrollar un sistema basado en IA que permita la conversión de texto e imágenes en audio en tiempo real.
- Evaluar la eficacia y usabilidad del asistente virtual mediante pruebas con usuarios para optimizar su desempeño.
- Garantizar la accesibilidad y facilidad de uso del asistente mediante una interfaz intuitiva y adaptable a diferentes dispositivos.

Justificación

El asistente IA para personas con discapacidad visual es una solución necesaria para eliminar las barreras de acceso a la información escrita y visual en entornos físicos y digitales. Actualmente, estas personas enfrentan dificultades para interpretar el contenido de las pantallas, lo que limita su autonomía y participación social. La falta de herramientas accesibles y eficientes

restringe su independencia en la educación, el empleo y la vida cotidiana, afectando su inclusión digital y social.

Este estudio busca desarrollar una herramienta que convierta texto e imágenes en audio en tiempo real, permitiendo una interacción inmediata con el entorno. A diferencia de soluciones existentes, esta propuesta será accesible, intuitiva y adaptable a distintos dispositivos y espacios, optimizando la accesibilidad y mejorando la seguridad en la toma de decisiones. Al proporcionar mayor autonomía, facilitar la lectura sin asistencia y promover una participación equitativa en la sociedad, el proyecto contribuirá significativamente a mejorar la calidad de vida de las personas con discapacidad visual.

Marco Teórico

El desarrollo tecnológico para personas con discapacidad visual requiere la incorporación de estrategias inclusivas que promuevan la accesibilidad digital, apoyadas por herramientas sofisticadas como la inteligencia artificial, la visión por computadora y el procesamiento del lenguaje natural. Este proyecto, orientado al diseño de un asistente visual interactivo, se fundamenta en diversos marcos teóricos que abordan la accesibilidad tecnológica. Los conceptos de inclusión digital y tecnología asistida son pilares esenciales, ya que garantizan que las personas con discapacidad puedan interactuar y desenvolverse de manera autónoma dentro de los entornos digitales. De igual manera, el uso de interfaces gráficas accesibles mediante software como CustomTkinter —con botones grandes, opciones de modo oscuro y señales auditivas— se alinea con los principios del diseño universal, buscando reducir la sobrecarga cognitiva y lograr una experiencia de uso fluida.

El marco teórico abarca seis áreas temáticas amplias. Por un lado, examina la inclusión digital y la accesibilidad en las TIC, con énfasis en la necesidad de plataformas desarrolladas con estándares universales como las WCAG. Por otro lado, se discute el papel de las tecnologías asistidas para personas con discapacidad visual, desde lectores de pantalla clásicos hasta aplicaciones móviles con visión artificial y retroalimentación multisensorial. El tercer tema aborda el uso de la inteligencia artificial para mejorar la accesibilidad, con especial enfoque en modelos generativos y multimodales como GPT-4o y Gemini, los cuales tienen la capacidad de transcribir información visual en experiencias auditivas. El cuarto tema trata sobre la visión por computadora y el análisis de imágenes, centrándose en el reconocimiento de objetos en tiempo real para asistir a personas ciegas o con baja visión. El quinto eje explora el procesamiento del lenguaje natural y la síntesis de voz, fundamentales para convertir texto en salidas de audio fluidas y expresivas. Finalmente, se analiza el diseño de interfaces accesibles, resaltando su importancia para ofrecer soluciones tecnológicas intuitivas, efectivas e inclusivas.

Inclusión digital y accesibilidad en las TIC

La inclusión digital es el proceso mediante el cual se garantiza que todos los ciudadanos, independientemente de su situación social, económica o física, tengan acceso y puedan hacer un uso efectivo de las Tecnologías de la Información y la Comunicación (TIC). Ha adquirido una importancia crucial en la sociedad contemporánea, ya que los servicios públicos, la educación, la salud y la participación ciudadana se desarrollan cada vez más sobre plataformas digitales. La transformación digital del gobierno, según la investigación de Djatmiko et al. (2025), es un elemento clave del desarrollo sostenible, ya que optimiza la eficacia administrativa y amplía la inclusión en los servicios públicos —especialmente en regiones desfavorecidas—, alineándose

con los Objetivos de Desarrollo Sostenible (ODS), en particular con el ODS 10 (reducción de las desigualdades) y el ODS 16 (sociedades pacíficas e inclusivas, e instituciones sólidas y transparentes).

Sin embargo, el acceso equitativo a las TIC sigue siendo un problema urgente, especialmente para los grupos en situación de desventaja, como las comunidades rurales, las personas mayores o las personas con discapacidad. El artículo destaca que la brecha digital está profundamente arraigada en factores socioeconómicos y culturales, en la falta de alfabetización digital, en la infraestructura deficiente y en las barreras institucionales. Las personas con discapacidad, por ejemplo, enfrentan múltiples obstáculos al intentar acceder a plataformas digitales que no han sido diseñadas bajo principios de accesibilidad universal. Esto implica que la inclusión digital no es solo un problema técnico, sino también político, cultural y estructural.

Como respuesta a los problemas actuales de inclusión digital, muchas investigaciones han instado a los gobiernos a implementar políticas regulatorias y estrategias inclusivas que integren elementos de accesibilidad digital, alfabetización tecnológica y participación multisectorial. Una de las partes más importantes de este proceso es el cumplimiento de las Pautas de Accesibilidad para el Contenido Web (WCAG), desarrolladas por el W3C, que establecen estándares para garantizar que los sitios web puedan ser utilizados por personas con diversos tipos de discapacidad. Estas pautas son fundamentales para crear equidad en el acceso a los servicios digitales públicos. Por ejemplo, el estudio de Surjit (2023) revela que la mayoría de los sitios web de gobierno electrónico en la India no cumplen ni siquiera con el nivel mínimo de conformidad A de las WCAG 2.1, lo que indica una falta de atención en la implementación de funciones básicas de accesibilidad. Los autores enfatizan que, para que la inclusión se haga realidad, los desarrolladores y diseñadores deben aplicar estas directrices desde las primeras

etapas del desarrollo web, con el fin de crear una experiencia digital accesible e igualitaria para todos, especialmente para los ciudadanos con discapacidad visual.

Tecnologías asistidas para personas con discapacidad visual

Los dispositivos y equipos de asistencia son un grupo heterogéneo de herramientas creadas para permitir que las personas con discapacidad logren una mayor independencia y funcionalidad. En el caso de las personas con discapacidad visual, estas tecnologías les permiten interactuar con la información y desplazarse por sus entornos físicos y virtuales con mayor autonomía. Tradicionalmente, se han utilizado lectores de pantalla, dispositivos Braille, ampliadores de pantalla y software de reconocimiento de voz. Sin embargo, los avances recientes en inteligencia artificial han permitido diseñar soluciones más innovadoras, como sistemas de detección de objetos en tiempo real integrados en aplicaciones móviles. El trabajo de Georgios et al. (2025), es un ejemplo perfecto de este avance al desarrollar una aplicación que combina detección inteligente de objetos con retroalimentación háptica y auditiva, proporcionando una experiencia de usuario más rica y eficaz para personas ciegas o con baja visión.

Las personas con discapacidad visual tienen necesidades especiales en cuanto a orientación espacial, movilidad segura y percepción de la información visual, tanto en el mundo físico como en el digital. En el ámbito de la tecnología musical, estas necesidades se vuelven esenciales debido al uso predominante de interfaces gráficas altamente visuales en estaciones de trabajo de audio digital (DAWs) y plugins. Los usuarios ciegos o con baja visión enfrentan importantes barreras técnicas derivadas del uso generalizado de representaciones visuales como espectros de frecuencia, curvas de ecualización y controles no etiquetados, los cuales son

difíciles o imposibles de interpretar mediante tecnologías convencionales como los lectores de pantalla. (Toro & Wu, 2025)

Ante estas limitaciones, el uso de tecnologías más recientes como la visión artificial y modelos ligeros de aprendizaje automático, como EfficientDet-lite2, ofrece nuevas posibilidades para crear herramientas más accesibles. El uso de retroalimentación háptica y auditiva, tal como se propone en el mismo estudio, puede facilitar la comprensión espacial del entorno o de parámetros complejos en aplicaciones de audio, brindando una interacción más intuitiva y eficaz. Este enfoque multisensorial no solo incrementa el nivel de independencia de los usuarios, sino que también ofrece condiciones más equitativas en entornos educativos, laborales y creativos. Por lo tanto, el desarrollo de soluciones centradas en la accesibilidad—desde software adaptable hasta dispositivos físicos como controladores con retroalimentación táctil—representa un gran paso hacia el empoderamiento tecnológico de la comunidad ciega y con baja visión.

Inteligencia Artificial Aplicada a la Accesibilidad

La inteligencia artificial (IA) también ha experimentado un crecimiento profundo en el área de la accesibilidad, revolucionando la forma en que las personas con discapacidades interactúan con los espacios digitales. Modelos multimodales y generativos como GPT-4o o Google Gemini han creado nuevas oportunidades al integrar sofisticadas capacidades de procesamiento de lenguaje natural, reconocimiento de voz y comprensión contextual, facilitando tecnologías más accesibles. Estos permiten que la experiencia visual se convierta en una experiencia auditiva o verbal, aumentando la autonomía de las personas con discapacidad visual. Algunas aplicaciones útiles son asistentes virtuales que facilitan la navegación por voz, software de reconocimiento de imágenes para identificar objetos o textos, y generadores automáticos de

descripciones que hacen accesible el contenido gráfico. Estas innovaciones no son solo mejoras técnicas, sino que responden a necesidades reales, reduciendo barreras y ayudando a lograr una interacción digital más inclusiva. (Deshakulkarni Bhaskar & Ramakrishnan, 2024)

La inteligencia artificial (IA) también ha avanzado sustancialmente en el campo de la accesibilidad, revolucionando la educación inclusiva con tecnologías que eliminan barreras para estudiantes con discapacidad. Modelos generativos y multimodales, incluyendo GPT-4o, ya han demostrado su potencial al integrar procesamiento de lenguaje natural, reconocimiento de voz y comprensión del contexto, lo que permite la generación automática de contenidos adaptados (por ejemplo, descripciones de imágenes para estudiantes ciegos o transcripciones automáticas para estudiantes sordos). (Verónica-Alexandra et al., 2025)

Estas capacidades se materializan en aplicaciones prácticas, como asistentes virtuales que simplifican la navegación por voz, sistemas de clasificación de imágenes que identifican objetos en el contexto educativo, y generadores automáticos de descripciones que otorgan acceso a la información visual. Estudios como el de Toyokawa et al. (2023) reconocen cómo estas tecnologías no solo personalizan el aprendizaje, sino que también reducen las tareas administrativas para los docentes, permitiéndoles enfocarse en pedagogías inclusivas. Sin embargo, su implementación está acompañada de retos relacionados con la infraestructura, la preparación docente y cuestiones éticas sobre la protección de datos, componentes fundamentales para garantizar una integración equitativa y sostenible. Esta tecnología, fundamentada en revisiones sistemáticas como la presente, demuestra el potencial transformador de la IA para crear espacios de aprendizaje realmente accesibles e inclusivos.

Visión por Computadora y Análisis de Imágenes

La visión por computadora es una rama de la inteligencia artificial que busca facilitar que las máquinas comprendan e interpreten el mundo visualmente, y es fundamental para el desarrollo de tecnologías accesibles para personas con discapacidad visual. El análisis de imágenes es una parte importante de este campo, ya que permite la detección, clasificación y localización de objetos en tiempo real. Bastidas-Guacho et al. (2025) proponen un sistema móvil basado en el modelo YOLOv5s, diseñado para identificar peligros en escenarios reales mediante la captura y procesamiento de imágenes con alta precisión. El sistema incluye un estudio detallado de 7,600 imágenes tomadas en un entorno educativo particular, cuidadosamente etiquetadas y categorizadas para mejorar el rendimiento del modelo en la identificación de peligros relevantes.

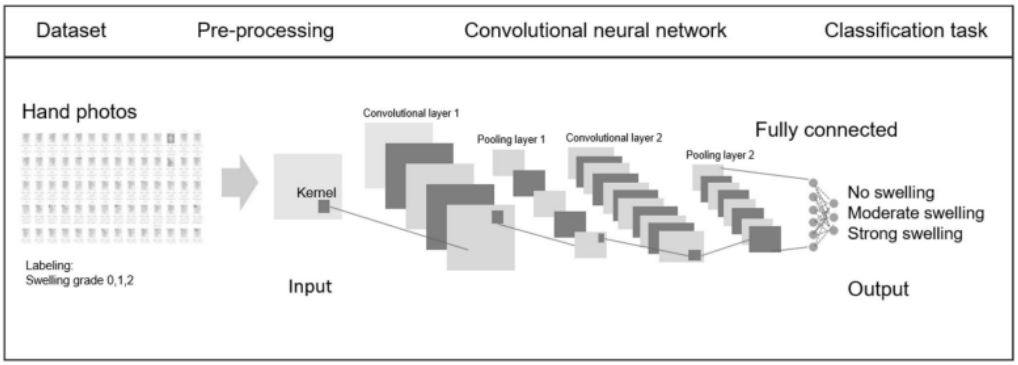
Además, el modelo fue entrenado para ofrecer respuestas rápidas en dispositivos móviles, demostrando su eficiencia computacional. Su funcionamiento se complementa con retroalimentación auditiva y háptica para que la detección visual se traduzca en información accesible para el usuario. Así, el uso de algoritmos de detección y segmentación en la visión por computadora se presenta como una herramienta esencial para convertir entornos visuales complejos en experiencias interpretables para personas con discapacidad visual, promoviendo su autonomía y seguridad en la movilidad diaria. Por otro lado, la visión por computadora y el análisis de imágenes han tenido una amplia aplicación en el ámbito clínico, especialmente en especialidades como la reumatología, donde la detección temprana y precisa de lesiones articulares es crucial para el diagnóstico y seguimiento de enfermedades inflamatorias crónicas.

En un artículo publicado en Pharmaceutical Medicine, Hügle (2024) explica cómo los modelos de redes neuronales convolucionales (CNN) han sido utilizados con éxito en la

interpretación automatizada de imágenes médicas como radiografías, resonancias magnéticas y ecografías. Estos modelos pueden detectar hallazgos clínicos importantes como erosiones, sacroilitis y osteoartritis, y ya existen algoritmos aprobados por la FDA que se utilizan en la práctica clínica. Uno de los usos más prometedores es la aplicación de CNN a imágenes tomadas directamente por el paciente, por ejemplo, fotografías de sus manos, para detectar patrones visuales asociados con la hinchazón articular. El proceso incluye la extracción de características mediante convoluciones, la reducción de datos a través de capas de pooling, y la clasificación posterior mediante aprendizaje supervisado. Este tipo de automatización no solo mejora la precisión diagnóstica, sino que también incrementa la productividad del especialista al disminuir la carga de trabajo e incluso permitir el monitoreo remoto de la enfermedad. Además, se destacan características adicionales como los mapas de calor (heatmaps), que dirigen visualmente al médico hacia las áreas de interés, así como interfaces sin código que democratizan el entrenamiento de modelos por parte del personal clínico. Estas aplicaciones consolidan a la visión por computadora como un elemento transformador en la medicina personalizada.

Figura 1

Red neuronal convolucional



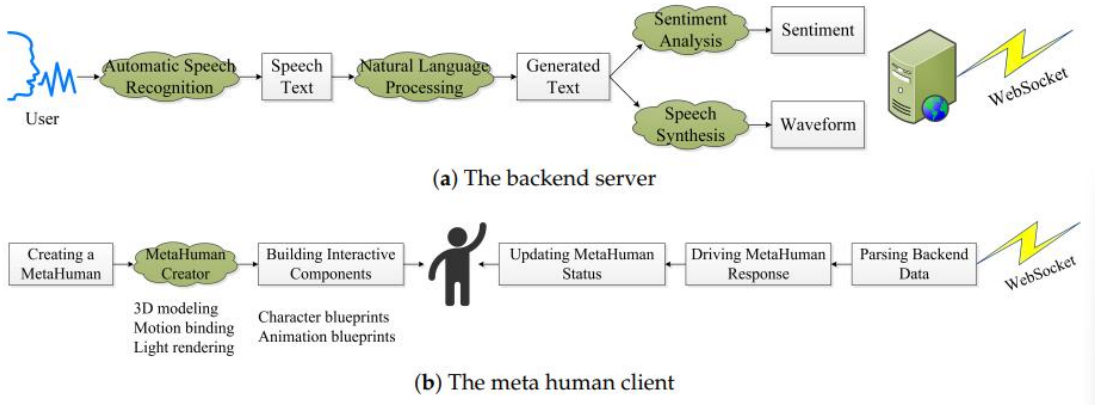
Nota. Tomado de (Hügle, 2024). Tarea de clasificación para predecir la inflamación articular a partir de fotos de manos de pacientes con artritis reumatoide.

Conversión de Texto a Voz y Diseño de Interfaces Accesibles

El Procesamiento de Lenguaje Natural (PLN) es un área importante de la inteligencia artificial que permite a las computadoras comprender, leer y producir el lenguaje humano. En el sistema digital humano hiperrealista desarrollado por Song & Xiong (2025), el PLN es el componente clave al apoyar la comprensión semántica de los textos y la creación de respuestas personalizadas mediante modelos de lenguaje a gran escala (LLMs) como ChatGLM y ChatGPT.

Figura 2

Sistema digital humano impulsado por LLM



Nota. Tomado de (Song & Xiong, 2025). Arquitectura de un sistema interactivo inteligente impulsado por un modelo de lenguaje de gran escala (LLM) con visualización 3D hiperrealista

Estos modelos, entrenados con enormes conjuntos de datos, pueden realizar tareas como análisis de sentimientos, extracción de información, generación de textos y diálogo conversacional. Este procesamiento del lenguaje se combina luego con tecnologías de texto a voz (TTS), es decir, motores de redes neuronales como Microsoft Edge TTS, que transforman el texto generado por el modelo en una voz humana expresiva y emocional. Esta combinación de PLN y TTS no solo aumenta el nivel de naturalidad en la comunicación humano-máquina, sino que también permite una comunicación inclusiva y accesible, especialmente en entornos virtuales con interfaces animadas o avatares 3D, como se sugiere en el proyecto. Mediante el uso de marcadores como SSML (Speech Synthesis Markup Language), visemas sincronizados y una prosodia detallada, el sistema ofrece una experiencia auditiva inmersiva que refleja, en gran medida, la emoción y los matices del habla humana natural. (Song & Xiong, 2025)

Por último, la creación de interfaces gráficas usables es fundamental en el desarrollo de aplicaciones para personas ciegas, ya que reduce la carga cognitiva y facilita una interacción fluida. En el proyecto desarrollado, se utilizó CustomTkinter para crear una interfaz centrada en la accesibilidad, con botones grandes, navegación mediante clics, modo oscuro y retroalimentación por voz — características que ofrecen legibilidad visual y navegación sencilla. Estas están en consonancia con propuestas actuales que integran múltiples modalidades de entrada y salida (por ejemplo, retroalimentación háptica y auditiva) que permiten a los usuarios ubicarse con mayor precisión y recibir información rica mediante una experiencia personalizada, simplificada y eficiente. Este enfoque está alineado con la necesidad de evitar la sobrecarga cognitiva, promoviendo la creación de sistemas inclusivos que respondan a las necesidades de personas con discapacidad visual. (Alzalabny et al., 2024)

Análisis de requerimientos

El desarrollo exitoso del proyecto “OpenEye” depende del cumplimiento riguroso de los requerimientos planteados desde la etapa inicial. Este análisis busca establecer las especificaciones de diseño, el alcance de la solución de ingeniería y evitar modificaciones críticas en etapas avanzadas del desarrollo.

Requerimientos de negocio

El principal objetivo de OpenEye es mejorar la autonomía, inclusión y calidad de vida de las personas con discapacidad visual, facilitando el acceso a contenidos digitales mediante un asistente virtual que combina visión artificial, procesamiento de lenguaje natural y síntesis de voz.

La iniciativa responde a la necesidad de contar con herramientas tecnológicas accesibles que permitan a los usuarios con baja o nula visión interpretar en tiempo real la información visual disponible en una pantalla. Este proyecto busca cerrar la brecha digital que afecta a este grupo poblacional, promoviendo la equidad y la inclusión social.

Indicadores de éxito asociados:

- Reducción del tiempo de interpretación de texto visual a menos de 10 segundos.
- Acceso autónomo a imágenes y documentos digitales sin ayuda de terceros.
- Aumento del uso del sistema por parte de estudiantes con discapacidad visual durante pruebas piloto.

Requerimientos de los interesados

Diversos actores tienen interés en el desarrollo y funcionalidad del sistema OpenEye. A continuación, se resumen sus principales necesidades:

- Usuarios con discapacidad visual: requieren un sistema intuitivo que describa en voz alta la información visual de la pantalla, sin necesidad de asistencia externa.
- Docentes y evaluadores: esperan una solución funcional, bien documentada y con impacto social real, que refleje conocimientos avanzados en ingeniería y accesibilidad.
- Familiares o cuidadores: desean que el usuario pueda interactuar con dispositivos digitales de forma más independiente.
- Organizaciones promotoras de inclusión: buscan herramientas tecnológicas alineadas con los principios de accesibilidad universal.

Requerimientos de la solución

Requerimientos funcionales

- El sistema debe permitir la captura completa o parcial de la pantalla, según elección del usuario.
- Debe aplicar OCR (Reconocimiento Óptico de Caracteres) para extraer el texto de la imagen capturada.
- El texto extraído debe ser procesado por un modelo de lenguaje (GPT de OpenAI) para generar una descripción coherente y enriquecida del contenido.
- La descripción generada debe ser convertida en voz mediante el motor de síntesis de voz gTTS (Google Text-to-Speech).
- El asistente debe contar con una interfaz gráfica accesible, desarrollada en Python con la librería customtkinter.
- El sistema debe iniciarse desde una ventana con dos opciones: captura de pantalla completa o recorte manual.
- Debe utilizar hilos (threading) para no bloquear la interfaz durante el procesamiento.
- El sistema debe incluir teclas de acceso rápido o botones visuales para facilitar su uso por parte de personas con baja visión.

Requerimientos no funcionales

- Compatibilidad multiplataforma: el software debe ejecutarse correctamente en sistemas Windows y Linux.
- Tiempo de respuesta: el tiempo máximo desde la captura hasta la reproducción del audio no debe superar los 20 segundos.

- Usabilidad: la interfaz debe tener botones grandes, colores contrastantes y pocas opciones para facilitar su navegación.
- Accesibilidad: debe permitir el uso sin necesidad de interacción con el ratón, mediante combinaciones de teclas o comandos de voz en el futuro.
- Estabilidad: el sistema debe funcionar sin fallos durante sesiones prolongadas de uso.

Requerimientos de transición

- Se debe desarrollar una guía de uso en audio y texto para facilitar la capacitación de nuevos usuarios con discapacidad visual.
- Se planea una fase de pruebas con usuarios reales para validar la efectividad del sistema y recoger retroalimentación.
- No se requiere migración de datos ni integración con sistemas anteriores, pero se propone realizar pruebas comparativas con lectores de pantalla tradicionales.
- Se deberá realizar una validación formal con usuarios al finalizar el desarrollo para documentar su grado de utilidad y facilidad de uso.

Análisis de restricciones

El diseño y desarrollo de sistemas en ingeniería, como el proyecto OpenEye, implica enfrentar diversas restricciones que limitan la implementación de ciertas soluciones tecnológicas. Estas restricciones pueden ser de carácter técnico, económico, normativo, ambiental o social, y deben ser consideradas desde las primeras fases del proyecto para garantizar su viabilidad y sostenibilidad. A continuación, se presentan las principales restricciones identificadas en el desarrollo de este asistente visual con inteligencia artificial.

Restricciones técnicas

- Compatibilidad del sistema operativo: actualmente, el script de OpenEye ha sido probado y desarrollado exclusivamente en entornos Windows. Esto restringe su uso inmediato en sistemas operativos como macOS o distribuciones Linux, limitando la portabilidad del software a otras plataformas.
- Procesamiento local limitado: el sistema depende del uso de servicios externos (como la API de OpenAI y gTTS) para el procesamiento del lenguaje y la síntesis de voz. Esto requiere una conexión constante a internet y limita su funcionalidad en entornos desconectados o con conectividad inestable.
- Requisitos de hardware: aunque el sistema está optimizado para equipos de gama media, el uso de múltiples hilos (threading), operaciones gráficas y conexión a APIs externas puede generar una carga significativa en computadores con bajos recursos.

Restricciones económicas

- Costo de acceso a la API de OpenAI: el sistema se basa en el modelo GPT-4o de OpenAI, cuyo uso implica la suscripción a una membresía de pago (al menos en

su modalidad básica). Este costo representa una barrera económica tanto para los desarrolladores como para instituciones o usuarios con recursos limitados.

- Dependencia de servicios externos gratuitos con limitaciones: la biblioteca gTTS, aunque es gratuita, presenta limitaciones en cuanto a velocidad, calidad y conectividad. En el futuro, podría ser necesario migrar a soluciones comerciales más robustas, lo cual incrementaría los costos operacionales.

Restricciones de calidad

- Limitaciones en la precisión del OCR: el reconocimiento óptico de caracteres puede verse afectado por la calidad de la imagen, el tipo de letra, el contraste o el idioma. Esto puede reducir la calidad de las descripciones generadas por el asistente.
- Fiabilidad del audio generado: el motor gTTS puede generar voces poco naturales en algunos casos, afectando la experiencia del usuario final. Además, no cuenta con soporte nativo para retroalimentación o personalización avanzada de voz.

Restricciones de tiempo

- Dependencia de la latencia en red: como el sistema depende de servicios en la nube, el tiempo de respuesta total (desde la captura de pantalla hasta la reproducción de audio) puede verse afectado por la velocidad de internet, aumentando los tiempos de espera más allá del objetivo ideal de 20 segundos.
- Duración del proceso de pruebas con usuarios reales: validar la efectividad del sistema con población con discapacidad visual requiere tiempo adicional para

convocatoria, adaptación, sensibilización y documentación de resultados, lo cual impacta el cronograma general del proyecto.

Restricciones de recursos

- Limitado equipo humano: el desarrollo del sistema ha sido realizado por un equipo reducido, lo cual restringe la posibilidad de implementar funcionalidades avanzadas o realizar pruebas extensivas en múltiples plataformas o idiomas.
- Infraestructura limitada: se carece de servidores propios o de una infraestructura de despliegue profesional, por lo que todo el procesamiento y ejecución se realiza a nivel local, lo cual reduce la escalabilidad de la solución.

Restricciones de alcance

- Enfoque limitado a imágenes en pantalla: OpenEye actualmente sólo interpreta información visual de la pantalla del dispositivo, sin capacidad para interpretar entornos físicos o texto en tiempo real captado por cámaras externas.
- Alcance funcional inicial: el sistema no incluye interacción por voz, retroalimentación adaptativa ni configuraciones personalizadas de accesibilidad, lo cual limita su personalización por parte del usuario final.

Restricciones normativas y sociales

- Accesibilidad parcial: si bien se han implementado criterios básicos de accesibilidad, el sistema aún no ha sido evaluado formalmente con estándares como WCAG (Web Content Accessibility Guidelines), lo cual podría restringir su adopción en entornos oficiales o educativos.

Método de selección

La selección tecnológica para el desarrollo de OpenEye respondió a un proceso riguroso que evaluó diversas alternativas según su capacidad para satisfacer los requerimientos funcionales del proyecto, especialmente en términos de accesibilidad, bajo costo, facilidad de implementación y rendimiento en equipos de gama media. Se descartaron opciones populares que, aunque potentes, resultaban inviables por su complejidad de configuración, requerimientos técnicos elevados o modelos de licencia restrictivos.

En cuanto a la interfaz gráfica, se analizaron bibliotecas como JavaFX, Electron, Kivy y PyQt. Aunque PyQt ofrece una mayor personalización y soporte robusto para accesibilidad, su curva de aprendizaje y complejidad de despliegue en entornos no técnicos limitaron su viabilidad para un equipo reducido y con recursos limitados. Electron, por su parte, fue descartado debido a su alto consumo de recursos y dependencias basadas en tecnologías web. Finalmente, se eligió CustomTkinter, una extensión moderna de Tkinter que permite crear interfaces más estilizadas sin sacrificar simplicidad ni ligereza. A pesar de sus limitaciones visuales frente a otros frameworks, su integración nativa con Python y su bajo consumo de memoria la convirtieron en una opción práctica para el desarrollo rápido de un prototipo funcional.

Para el procesamiento de imágenes y descripción visual, se consideraron soluciones locales como Tesseract OCR y modelos preentrenados como YOLOv5 o MobileNet, que pueden ejecutarse sin conexión. Sin embargo, estas alternativas presentaron limitaciones en la comprensión semántica del contenido visual, especialmente en escenarios donde se requiere contextualizar elementos (por ejemplo, "un botón azul con el texto 'Enviar' en la esquina inferior derecha"). Por esta razón, se seleccionó GPT-4o de OpenAI, que combina visión computacional y comprensión de lenguaje natural en un solo modelo multimodal. Su capacidad para generar

descripciones comprensibles para usuarios ciegos fue determinante, aunque se reconocen riesgos como la dependencia de conexión estable a internet y la exposición a costos variables por el uso de la API. Se mitigó parcialmente este riesgo utilizando la versión GPT-4o mini, que conserva una alta precisión a un menor costo por consulta.

En cuanto a la síntesis de voz, se compararon motores comerciales como Amazon Polly y Microsoft Azure TTS, que ofrecen excelente calidad de audio, pero requieren configuración compleja de autenticación y credenciales de uso. Además, sus planes gratuitos son limitados y poco sostenibles para un proyecto en fase de prueba. En contraste, se eligió gTTS (Google Text-to-Speech), una biblioteca de código abierto que, aunque no ofrece control detallado sobre la entonación o velocidad del habla, genera resultados suficientemente claros, especialmente en español latinoamericano, y permite una integración directa con Python sin procesos adicionales de autenticación o despliegue.

En términos de arquitectura, se optó por una estructura modular con procesamiento asincrónico mediante threading para evitar que operaciones de red o procesamiento bloqueen la interfaz del usuario. Esta decisión responde a la necesidad de mantener la aplicación funcional en equipos con recursos limitados, considerando que muchos usuarios con discapacidad visual en Colombia acceden a computadores de segunda mano o de características técnicas modestas.

En conjunto, la selección de estas tecnologías refleja una postura crítica y realista frente al entorno en el que se implementará OpenEye: un contexto académico con limitaciones técnicas, presupuestales y de infraestructura, pero con un alto compromiso social. Se priorizaron soluciones que, sin ser las más sofisticadas del mercado, permiten alcanzar un equilibrio adecuado entre funcionalidad, costo y accesibilidad.

Método de desarrollo

Para el desarrollo del asistente virtual OpenEye se empleó la metodología Kanban, una técnica ágil orientada a la mejora continua del flujo de trabajo y visualización de tareas. Esta metodología permite gestionar eficazmente el proceso de desarrollo mediante tableros que muestran el estado de cada actividad: por hacer, en progreso y terminado.

Kanban fue seleccionado por su adaptabilidad y enfoque visual, lo cual resultó ideal para un proyecto ejecutado por un equipo reducido. Además, favorece la identificación de cuellos de botella, la priorización de tareas y una respuesta ágil frente a cambios o mejoras requeridas en la solución. A través del uso de tarjetas con historias de usuario (HU), se estructuraron las tareas según su prioridad y dependencia.

Análisis de costos

El presente análisis de costos tiene como objetivo estimar los recursos económicos requeridos para el desarrollo y uso continuo del asistente visual accesible, diseñado para personas con discapacidad visual. Se contemplan tanto los costos de desarrollo como los operativos, con énfasis en servicios de inteligencia artificial en la nube, cuyo uso diario puede representar un gasto significativo si no se gestiona adecuadamente.

Clasificación de Costos

Hardware.

Se utiliza un computador personal con cámara y micrófono integrados. Costo adicional: 0 USD.

Licencias de Software.

Python, CustomTkinter y gTTS son herramientas de código abierto, sin costo de licencia.

Servicios en la Nube (OpenAI GPT-4o mini).

En cuanto a los servicios en la nube, el proyecto utiliza la API de OpenAI GPT-4o mini para procesar imágenes y generar descripciones en lenguaje natural. Esta API tiene un costo accesible, estimado en función del número de tokens utilizados por consulta. Para un uso frecuente con hasta 100 interacciones diarias, se calcula un consumo mensual de aproximadamente 3 millones de tokens de entrada y 1.5 millones de salida, lo que representa un costo total cercano a los $1.35 USD mensuales. Esta solución ofrece un balance óptimo entre capacidad tecnológica y viabilidad económica, siendo ideal para prototipos funcionales de bajo costo.

El valor estimado de $1.35 USD mensuales para el uso de la API de OpenAI GPT-4o mini se calculó tomando en cuenta un escenario de uso frecuente diario, propio de una aplicación operativa que asiste a personas con discapacidad visual en tiempo real. Se estimó un total de 100 consultas por día, cada una utilizando en promedio 1,000 tokens de entrada y 500 tokens de salida. En términos mensuales, esto representa un consumo aproximado de 3 millones de tokens de entrada (1000 × 100 × 30) y 1.5 millones de tokens de salida (500 × 100 × 30). De acuerdo con las tarifas oficiales de OpenAI para GPT-4o mini, los costos son de $0.15 USD por millón de tokens de entrada y $0.60 USD por millón de tokens de salida (Reuters, 2024). Por lo tanto, el costo mensual es:

- Entrada: 3 × 0.15 = $0.45 USD
- Salida: 1.5 × 0.60 = $0.90 USD

Sumando ambos valores, se obtiene un costo total mensual aproximado de $1.35 USD por el uso de la API, lo que demuestra que, incluso con un uso constante, esta tecnología sigue siendo económicamente viable para proyectos accesibles.

Costos indirectos.

En cuanto a los costos indirectos, se considera la conexión a internet como un recurso esencial para el funcionamiento continuo del asistente visual, ya que permite acceder a servicios en la nube como la API de OpenAI. Suponiendo un uso constante diario, se estima la necesidad de un plan de internet estable con un costo aproximado de $15 USD mensuales, suficiente para soportar las consultas de texto e imagen que realiza la aplicación.

Respecto a los costos de desarrollo, se calculó el valor del tiempo invertido en la programación, pruebas y puesta en marcha del proyecto. Se estimó un total de 80 horas de trabajo, con una valorización promedio de $5 USD por hora, lo que representa un costo total de $400 USD. Este valor refleja el esfuerzo técnico necesario para construir una solución funcional y accesible, considerando tanto el diseño de la interfaz como la integración de tecnologías como visión por computadora, síntesis de voz y procesamiento de lenguaje natural.

Tabla 1

Tabla de análisis de costos

Concepto	Descripción	Costo Estimado (USD)
Computador personal	Para desarrollo y pruebas del software	0 (propio)
Cámara y micrófono integrados	Captura de imágenes y entrada de voz	0 (propio)
API de OpenAI (GPT-4o mini)	Uso diario (≈100 consultas/día)	1.35 / mes

Concepto	Descripción	Costo Estimado (USD)
gTTS (Google Text-to-Speech)	Librería gratuita para síntesis de voz	0
CustomTkinter y Python	Librerías de código abierto	0
Conexión a Internet	Uso constante para acceso a API	15 / mes
Tiempo de desarrollo	80 horas x 5 USD/hora	400
Total, estimado inicial	(Primer mes con desarrollo y servicios)	$416.35 USD
Total, mensual operativo	(Sin desarrollo)	$16.35 USD/mes

Nota. Elaboración propia.

Plan de implementación

la aplicación práctica de la metodología Kanban en el desarrollo del sistema OpenEye, se desarrolla a través de la definición del Product Backlog, el análisis arquitectónico y el modelado de casos de uso.

Product Backlog

El Product Backlog representa el conjunto priorizado de funcionalidades requeridas por el sistema, organizadas en historias de usuario estructuradas bajo el formato: “Yo como… quiero… para…”. Cada historia incluye sus respectivos criterios de aceptación.

Tabla 2

Product Backlog del sistema OpenEye

Código	Nombre de la Historia	Descripción	Criterios de Aceptación
HU001	Captura de pantalla	Yo como usuario con discapacidad visual quiero capturar la pantalla completa o una región específica, para interpretar la información visual de forma autónoma.	* El sistema debe permitir seleccionar entre captura completa o recorte manual desde la interfaz.
HU002	Generación de descripción con IA	Yo como usuario con discapacidad visual quiero que el sistema genere una descripción del contenido visual, para comprender lo que aparece en la imagen capturada.	* La imagen debe enviarse correctamente a la API de ChatGPT GPT-4, y este debe devolver una descripción clara y coherente del contenido. * La respuesta no debe superar los 30 segundos.
HU003	Conversión a voz (TTS)	Yo como usuario con discapacidad visual quiero escuchar en voz alta la descripción generada, para acceder al contenido sin leerlo.	* La síntesis de voz debe ser clara, fluida y sin errores de pronunciación. * El audio debe reproducirse automáticamente tras generarse la descripción.
HU004	Activación por clic derecho	Yo como usuario con discapacidad visual quiero que el sistema tome una captura y genere una descripción al hacer clic derecho dentro de la interfaz, para facilitar el uso sin	* Al hacer clic derecho en cualquier parte de la interfaz, se debe tomar una captura de pantalla completa. * El flujo completo (captura → descripción → voz) debe

Código	Nombre de la Historia	Descripción	Criterios de Aceptación
		necesidad de apuntar a un botón específico.	ejecutarse sin necesidad de presionar un botón adicional.
HU005	Fluidez del sistema (multihilos)	Yo como desarrollador quiero que la aplicación use hilos para ejecutar procesos en segundo plano, para evitar bloqueos en la interfaz gráfica.	* La interfaz debe mantenerse operativa durante la ejecución de OCR, IA y TTS.
HU006	Interfaz accesible	Yo como usuario con discapacidad visual quiero una interfaz con botones grandes y colores contrastantes, para facilitar su uso y navegación.	* Los elementos visuales deben tener tamaños grandes y contraste alto. * La interfaz debe ejecutarse correctamente en Windows

Nota. Elaboración propia.

Análisis Arquitectura y Diseño

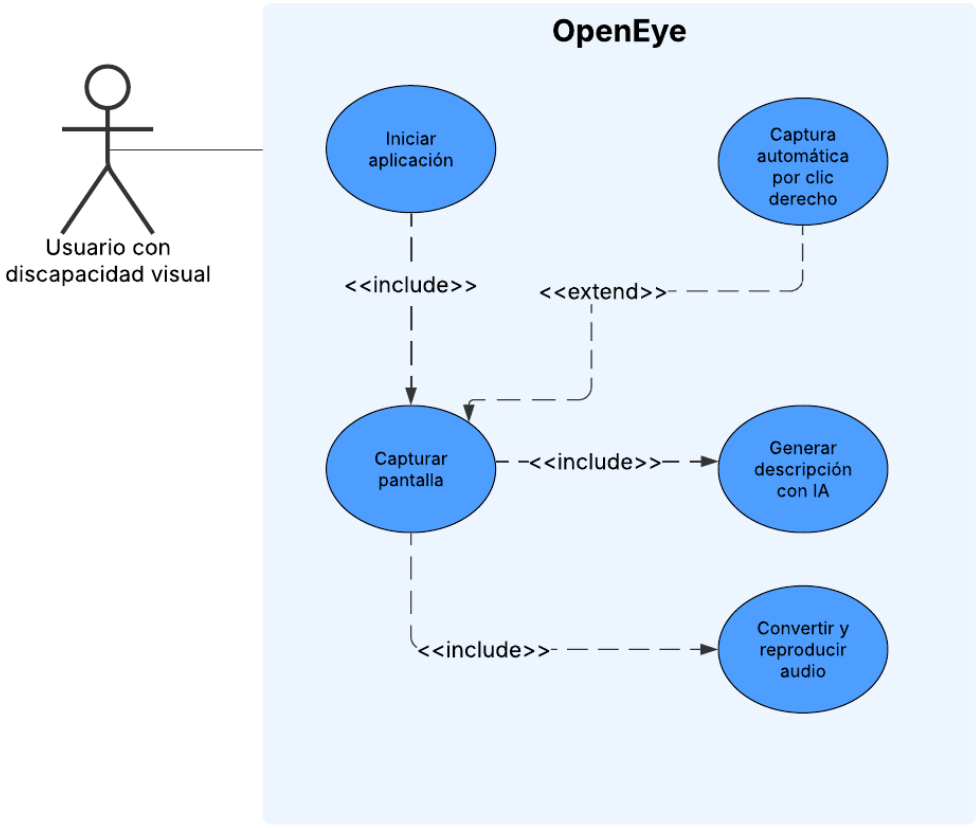
Este apartado describe la funcionalidad general del sistema mediante el modelado de casos de uso, los cuales representan las interacciones esperadas entre el usuario y el sistema.

Diagrama de Casos de uso

El siguiente diagrama representa los casos de uso del sistema "OpenEye", las relaciones entre ellos y la participación del actor externo "Usuario con discapacidad visual".

Figura 3

Diagrama de Casos de uso del sistema OpenEye



Nota. Elaboración propia. Elaborado con lucidchart.com (s.f.). https://www.lucidchart.com/.

Descripción de casos uso

La siguiente tabla detalla los principales casos de uso definidos para OpenEye, incluyendo su funcionalidad, historias de usuario relacionadas, flujo principal y posibles escenarios alternos.

Tabla 3

Descripción de casos de uso del sistema OpenEye

Nombre del caso de uso	Funcionalidad	Historias de usuario vinculadas	Flujo principal	Flujo alterno
Iniciar aplicación	Permite al usuario ejecutar OpenEye e iniciar su interfaz	HU001, HU004	1. El usuario abre la aplicación. 2. Se presenta la interfaz con las opciones de captura.	Error al cargar dependencias.
Capturar pantalla	Permite tomar una imagen completa o recortada de la pantalla del dispositivo	HU001	1. Usuario elige tipo de captura. 2. El sistema guarda la imagen de forma temporal.	Cancelación de la acción por parte del usuario.
Generar descripción con IA	Procesa la imagen que extrae el contenido y genera una descripción en lenguaje natural	HU002	1. La imagen es enviada a GPT-4o. 2. Se aplica OCR y procesamiento semántico. 3. Se recibe una descripción detallada del contenido visual.	Fallo en la conexión o imagen ilegible.
Convertir y reproducir audio	Convierte la descripción en voz y la reproduce automáticamente	HU003	1. El texto es enviado a gTTS. 2. Se genera un archivo de audio. 3. El sistema reproduce el audio generado.	Fallo en la conversión o conectividad.

Nombre del caso de uso	Funcionalidad	Historias de usuario vinculadas	Flujo principal	Flujo alterno
Captura automática por clic derecho	Permite al usuario tomar una captura desde cualquier punto de la interfaz haciendo clic derecho, y activa automáticamente la descripción y su lectura en voz	HU004, HU006	1. El usuario hace clic derecho en cualquier parte de la interfaz. 2. El sistema captura la pantalla completa. 3. La imagen se envía a GPT-4o. 4. Se genera una descripción. 5. El texto se convierte en voz y se reproduce automáticamente.	Error en captura o falta de conexión a internet.

Nota. Elaboración propia. Los flujos principales representan la secuencia ideal de ejecución del sistema, mientras que los flujos alternos contemplan fallos comunes como errores de red o cancelación por parte del usuario.

Arquitectura de la solución

La arquitectura del sistema OpenEye se presenta utilizando el modelo C4 (Contexto, Contenedores, Componentes y Código), una metodología de documentación arquitectónica que ofrece una representación jerárquica y estructurada del sistema. Este enfoque permite visualizar la solución desde distintos niveles de abstracción, facilitando su comprensión tanto para públicos técnicos como no técnicos.

Esta representación arquitectónica permite identificar dependencias críticas, optimizar el diseño de los componentes y establecer una base sólida para la implementación del asistente visual OpenEye, garantizando escalabilidad, mantenibilidad y eficiencia.

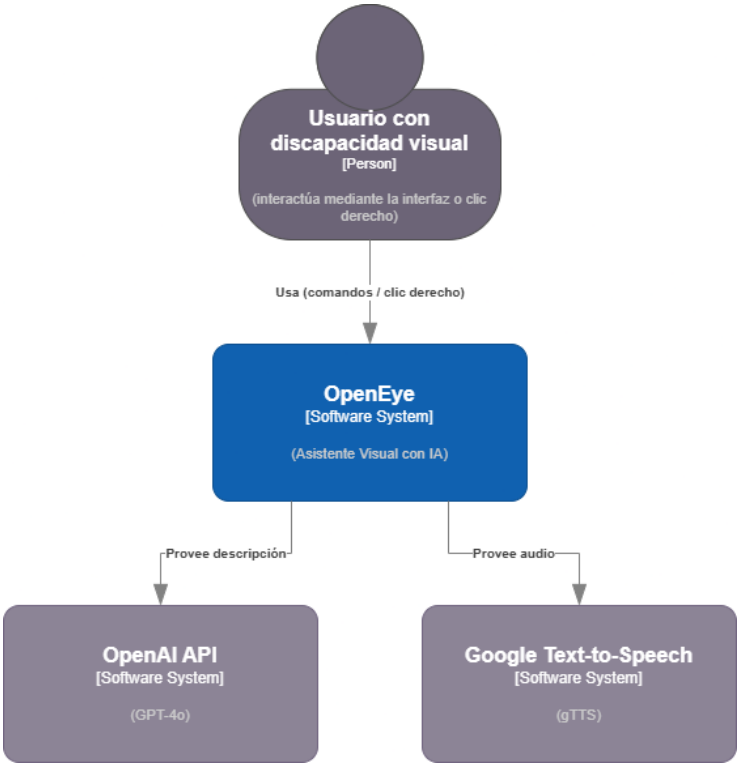
Diagrama de Contexto C4

Este diagrama muestra cómo interactúan los usuarios con discapacidad visual con el sistema, y cómo OpenEye se relaciona con los servicios externos de inteligencia artificial y síntesis de voz. Se describe la integración del sistema dentro de su entorno.

- **Persona externa:** Usuario con discapacidad visual que interactúa mediante una interfaz gráfica o comandos contextuales.
- **Sistema central:** OpenEye, un asistente visual basado en inteligencia artificial.
- **Sistemas externos:**
    - **OpenAI API (GPT-4o):** Analiza imágenes y genera descripciones textuales.
    - **Google Text-to-Speech (gTTS):** Convierte texto en audio comprensible para el usuario.

Figura 4

C4 Nivel 1: Contexto del Sistema OpenEye



Nota. Elaboración propia. Elaborado con diagrams.net (s.f.). https://app.diagrams.net/.

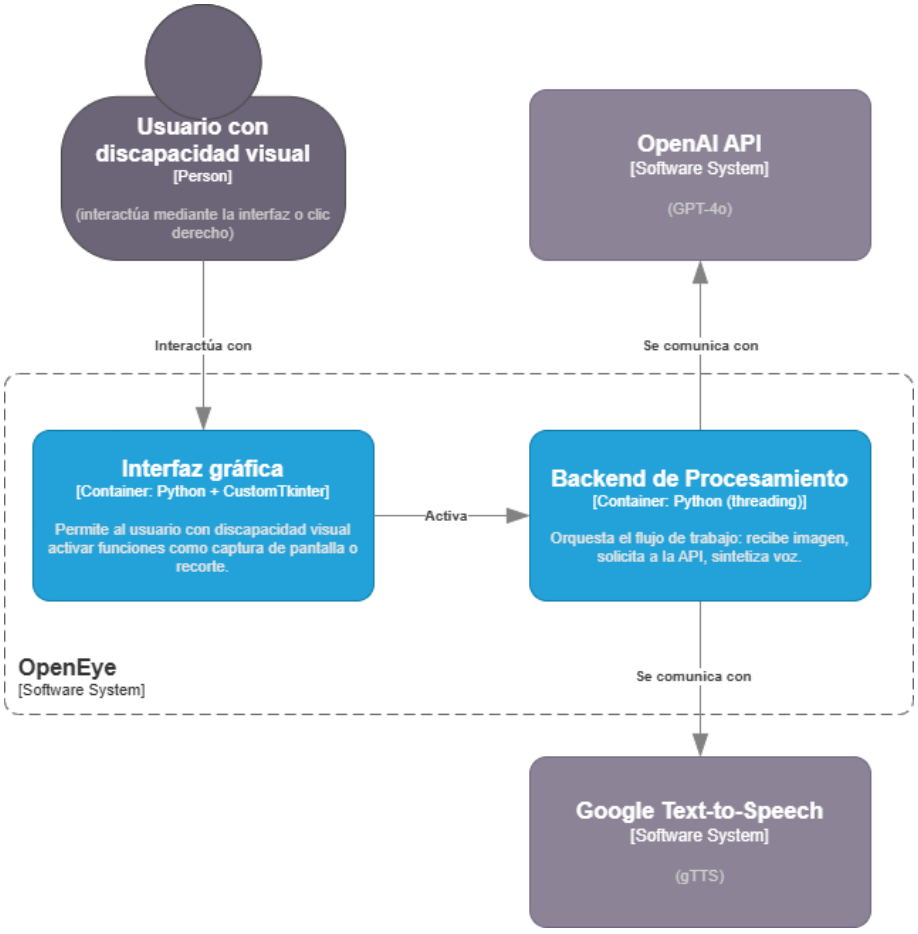
Diagrama de contenedores C4

Este nivel ilustra la organización de los principales contenedores del sistema y sus interacciones. Se destacan los elementos clave que conforman la arquitectura lógica de OpenEye.

- **Contenedores principales:** Interfaz gráfica (Python + CustomTkinter) para interacción del usuario, Backend de Procesamiento (Python con threading) para orquestar el flujo de trabajo, y APIs externas (OpenAI y Google TTS) para procesamiento inteligente.

Figura 5

C4 Nivel 2: Contenedores del Sistema OpenEye



Nota. Elaboración propia. Elaborado con diagrams.net (s.f.). https://app.diagrams.net/.

Diagrama de componentes C4

Este nivel descompone el contenedor de backend de procesamiento en sus módulos internos, describiendo cómo colaboran para convertir imágenes en audio. Resulta útil para identificar dependencias internas y mejorar el diseño modular.

Componentes internos del Backend de Procesamiento:

- **Captura de imagen:** Módulo para capturar pantalla o recortes usando PIL_pyanutgui.

- **Procesamiento con GPT-4o:** Servicio para enviar imágenes en base64 a OpenAI y recibir descripciones textuales.
- **Conversión de texto a voz:** Componente gTTS para sintetizar audio a partir de las descripciones.
- **Gestor de sonido:** Módulo pygame.mixer para reproducir audio y gestionar interrupciones.

Figura 6

C4 Nivel 3: Componentes del Backend de Procesamiento

Nota. Elaboración propia. Elaborado con diagrams.net (s.f.). https://app.diagrams.net/.

Modelo y Flujo de Datos

El prototipo actual de *OpenEye* no incorpora una base de datos relacional ni mecanismos de almacenamiento persistente para la gestión de información. En su lugar, adopta un modelo de datos basado en la interacción dinámica y en tiempo real con la API de OpenAI, específicamente

mediante el modelo GPT-4o. Este enfoque permite procesar imágenes capturadas por el usuario y generar descripciones textuales accesibles, orientadas a personas con discapacidad visual. Al prescindir del almacenamiento permanente, se priorizan la simplicidad operativa y la eficiencia, características clave en entornos donde la adaptabilidad y la respuesta inmediata son fundamentales.

La información —como las capturas de pantalla, las descripciones generadas y los archivos de audio— se maneja de forma temporal, creándose y eliminándose en cada sesión, lo cual reduce significativamente la complejidad arquitectónica. Sin embargo, esta arquitectura introduce una dependencia crítica de la conectividad a internet, dado que el procesamiento de datos depende de servicios externos.

El flujo de datos se inicia con la captura de una imagen, ya sea de toda la pantalla o de una región específica, utilizando la biblioteca PyAutoGUI. Esta imagen se guarda temporalmente en formato PNG. Posteriormente, se codifica en base64 para ser enviada a la API, acompañada de una instrucción clara que solicita una descripción accesible y breve, con una duración estimada de lectura inferior a 30 segundos. El modelo GPT-4o interpreta la imagen y devuelve un texto descriptivo, que se transforma en audio mediante gTTS (Google Text-to-Speech). La reproducción se realiza inmediatamente con Pygame y el archivo de audio se elimina tras su uso, evitando acumulaciones innecesarias en el sistema.

Este flujo puede describirse como una secuencia lineal: captura, codificación, análisis, síntesis y reproducción. La eficacia del sistema depende en gran medida de la estabilidad de la conexión a internet, que puede influir en la latencia, especialmente en entornos con conectividad limitada. Para mitigar este riesgo, se emplea GPT-4o mini, una variante optimizada del modelo que permite mantener el tiempo total de procesamiento dentro del objetivo de 20 segundos.

Cabe destacar que la ausencia de almacenamiento persistente impide la reutilización de descripciones o imágenes previamente procesadas. Aunque esta limitación responde a un enfoque minimalista en el diseño del prototipo, se contempla su abordaje en futuras versiones, posiblemente mediante la integración de modelos locales de inteligencia artificial.

En caso de errores —como fallos en la conectividad o imágenes no válidas— el sistema emite un mensaje predeterminado que informa al usuario sobre la imposibilidad de generar una descripción. Este mecanismo contribuye a una experiencia de uso consistente. Asimismo, se han implementado rutinas específicas para la gestión y eliminación de archivos temporales durante la ejecución y al cerrar el programa, lo que favorece la eficiencia y limpieza del entorno de ejecución.

Prototipo No Funcional

El diseño de la interfaz y la experiencia de usuario de *OpenEye* se orienta a garantizar la accesibilidad para personas con discapacidad visual, incluyendo tanto a usuarios ciegos como a aquellos con baja visión. La propuesta cumple con los principios establecidos por las Pautas de Accesibilidad para el Contenido Web (WCAG 2.1, nivel AA), priorizando la claridad, simplicidad e interacción intuitiva.

Se emplea la biblioteca CustomTkinter para la creación de una interfaz gráfica moderna, con un tema oscuro, contrastes adecuados y elementos visuales diseñados para una lectura óptima. La ventana principal presenta un diseño vertical con jerarquía clara: un título prominente, un subtítulo explicativo, dos botones funcionales y un indicador de estado. Esta organización facilita la navegación y minimiza la carga cognitiva del usuario.

El tema oscuro implementado asegura una relación de contraste mínima de 4.5:1 entre el texto y el fondo, mejorando así la visibilidad para personas con visión reducida. Los botones tienen un tamaño de 280 por 45 píxeles, lo que permite su selección incluso en condiciones de motricidad limitada. Se complementan con emojis representativos —como una cámara para la captura completa o unas tijeras para la captura selectiva— que refuerzan visualmente su propósito. Además, un marco informativo destaca una funcionalidad clave: la activación de una captura rápida mediante clic derecho en cualquier parte de la pantalla, lo que representa una alternativa útil para usuarios que prefieren evitar el uso de botones.

La retroalimentación auditiva constituye un componente esencial de la experiencia de usuario. Cada interacción —desde la captura hasta la generación de la descripción— culmina en la reproducción automática de un archivo de audio generado por gTTS. Esto permite a los usuarios recibir información de manera fluida, sin necesidad de confirmaciones adicionales.

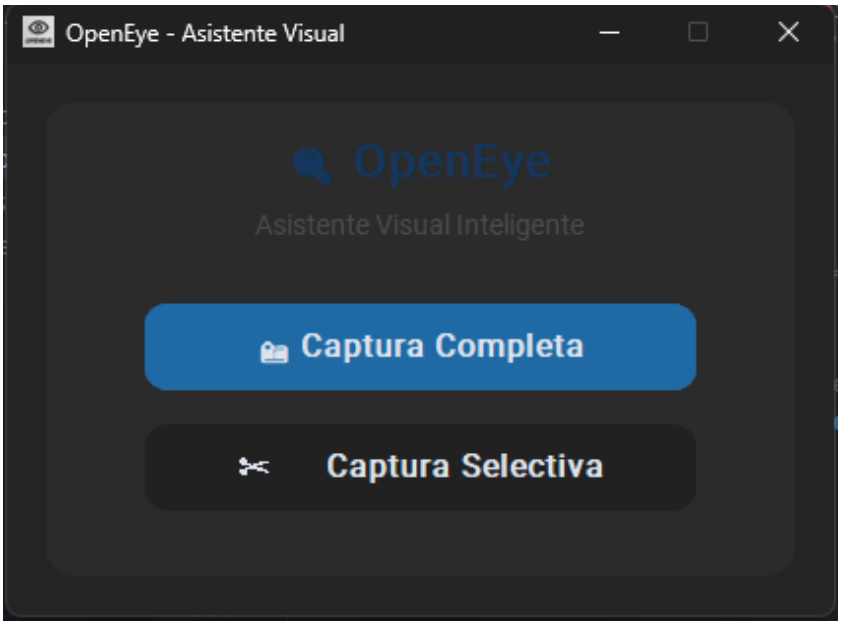
El sistema es compatible con los sistemas operativos Windows y Linux, gracias a la portabilidad que ofrece CustomTkinter. En el modo de captura selectiva, la interfaz oscurece la pantalla hasta un 30 % de opacidad, permitiendo al usuario definir una región de interés mediante un rectángulo rojo con borde de 2 píxeles, guiado por un cursor en forma de cruz. Aunque esta función requiere cierto grado de visión para su uso efectivo, representa una experiencia visual clara y precisa. En versiones futuras, se considera la inclusión de mecanismos alternativos, como comandos de voz o interfaces táctiles, para mejorar la accesibilidad. La arquitectura de la interfaz ha sido diseñada para ser escalable, permitiendo la integración de nuevas funcionalidades sin comprometer la estructura base.

Wireframe: Interfaz Principal

El wireframe correspondiente a la ventana principal representa una interfaz compacta de 420 por 280 píxeles. En su parte superior se ubica el título "🔍 OpenEye", centrado y en fuente de 24 puntos en negrita, con color blanco sobre un fondo oscuro para maximizar la legibilidad. Justo debajo se encuentra el subtítulo "Asistente Visual Inteligente", en fuente de 14 puntos y tono gris claro, que contextualiza la funcionalidad de la aplicación. Dos botones ocupan el área central: "📸 Captura Completa" y "✂️ Captura Selectiva", ambos con fondo azul y efecto de resaltado al pasar el cursor, diseñados para facilitar la interacción.

Figura 7

Interfaz Principal - OpenEye



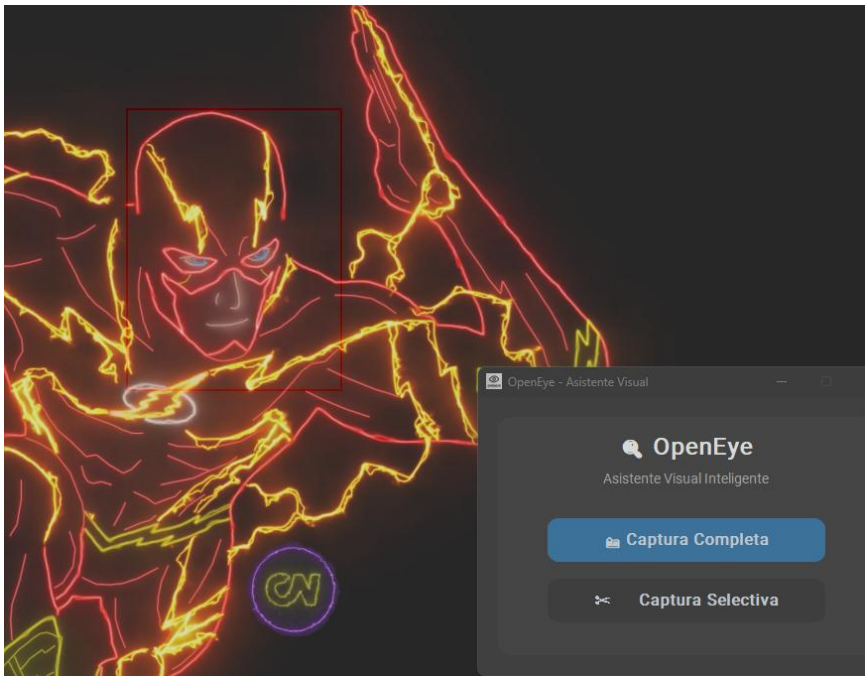
Nota: Elaboración Propia

Wireframe: Captura Selectiva

Este wireframe ilustra la experiencia visual durante el proceso de selección manual. La pantalla se atenúa con un gris translúcido al 30 %, permitiendo al usuario distinguir el contenido subyacente. A medida que se arrastra el cursor —con forma de cruz para mayor precisión— se dibuja un rectángulo rojo con borde visible. Al finalizar la selección, el área delimitada es procesada inmediatamente. Esta visualización es especialmente útil para usuarios con visión parcial que requieren asistencia para enfocar regiones específicas de la pantalla.

Figura 8

OpenEye en escritorio



Nota: Elaboración Propia. Pantallazo de un escritorio junto a la interfaz

Implementación

La implementación de *OpenEye* ha sido desarrollada utilizando Python, aprovechando su versatilidad y compatibilidad con bibliotecas especializadas en inteligencia artificial,

manipulación gráfica y procesamiento de audio. La arquitectura del sistema sigue un enfoque modular, segmentando las funcionalidades en componentes independientes encargados de la captura de pantalla, análisis de imagen, síntesis de voz e interfaz gráfica. Esta organización favorece la mantenibilidad y escalabilidad del software.

La biblioteca CustomTkinter se encarga de la interfaz visual, mientras que PyAutoGUI y PIL permiten capturar y manipular imágenes en tiempo real. La API de OpenAI, a través del modelo GPT-4o, interpreta estas imágenes y genera descripciones textuales adaptadas para personas con discapacidad visual. Posteriormente, gTTS transforma el texto en un archivo de audio, que se reproduce con Pygame, asegurando una transición fluida entre los distintos componentes del sistema.

El flujo de ejecución comienza con la captura de una imagen, la cual se codifica en base64 y se envía a la API de OpenAI para su análisis. Una vez obtenida la descripción, se genera un archivo de audio MP3 que se reproduce automáticamente. La gestión de archivos temporales es rigurosa: tanto imágenes como audios se eliminan al cerrar la aplicación o en puntos clave durante la ejecución, lo que evita la acumulación de residuos digitales.

Para garantizar una interfaz responsiva, se emplea procesamiento en segundo plano mediante hilos. Asimismo, un sistema de bloqueo asegura que los audios no se superpongan, interrumpiendo cualquier reproducción activa antes de iniciar una nueva.

Entre los principales desafíos identificados se encuentra la dependencia de una conexión a internet estable para el uso de la API de OpenAI y gTTS. Esta limitación puede introducir latencia en contextos de conectividad deficiente. No obstante, la utilización del modelo GPT-4o mini contribuye a mitigar este problema, permitiendo alcanzar el tiempo de respuesta deseado. En cuanto a la compatibilidad, el sistema ha sido probado en entornos Windows y Linux, con

planes de extender su validación a macOS. Finalmente, se han implementado mecanismos de manejo de errores que aseguran una respuesta adecuada ante fallos, manteniendo la experiencia de usuario bajo control en todo momento.

Pruebas y QA de Software

El aseguramiento de la calidad del prototipo *OpenEye* se fundamenta en la aplicación del modelo en V, un enfoque metodológico que vincula de manera estructurada cada fase del desarrollo con su respectiva etapa de pruebas. Este modelo resulta especialmente adecuado para proyectos académicos con requisitos previamente definidos y equipos de trabajo reducidos, permitiendo validar rigurosamente cada funcionalidad implementada. En el caso de *OpenEye*, el énfasis está puesto en la accesibilidad y la usabilidad para personas con discapacidad visual, desde la captura de pantalla hasta la reproducción de audio mediante síntesis de voz.

El proceso de validación se estructura en múltiples niveles de prueba, diseñados para abordar tanto componentes individuales como la experiencia integral del usuario. En la fase de análisis de requerimientos, se verifica que las funcionalidades especificadas en las historias de usuario —como la captura completa de pantalla o la generación de audio— respondan a las necesidades de usuarios con discapacidad visual, docentes evaluadores y potenciales organizaciones de inclusión interesadas en la adopción del sistema. Las pruebas de aceptación en esta etapa se llevan a cabo con usuarios reales, quienes interactúan directamente con el sistema para confirmar su facilidad de uso e impacto. Por ejemplo, se evalúa si la función de captura rápida mediante clic derecho permite ejecutar acciones de manera autónoma y sin asistencia.

Durante la etapa de diseño, se valida que la arquitectura del sistema —que integra captura de imágenes, procesamiento mediante la API de OpenAI, conversión a audio y una interfaz

gráfica basada en CustomTkinter— cumpla con los requerimientos establecidos. Las pruebas de sistema comprueban la funcionalidad del flujo completo: desde la selección de una región de pantalla hasta la reproducción auditiva de su descripción. Se establece un umbral de desempeño máximo de 20 segundos por ciclo funcional en condiciones de conexión estable. Adicionalmente, se verifica la compatibilidad del prototipo con los sistemas operativos Windows y Linux, asegurando el funcionamiento correcto de bibliotecas clave como PyAutoGUI, gTTS y Pygame.

A nivel de diseño modular, se desarrollan pruebas unitarias para validar el comportamiento de componentes específicos. Se comprueba, por ejemplo, que la función de codificación genere cadenas base64 válidas, que las descripciones generadas por la API de OpenAI sean coherentes con la imagen analizada, y que la síntesis de voz produzca archivos de audio claros y funcionales. Estas pruebas se automatizan mediante herramientas como unittest o pytest, lo que permite una detección oportuna de errores antes de la integración de componentes.

En la fase de implementación, se ejecutan pruebas de integración para verificar la interacción adecuada entre módulos. Se analiza, por ejemplo, si la captura selectiva de pantalla activa correctamente los procesos de análisis y reproducción de audio, y si el uso de subprocesos mantiene la interfaz gráfica responsiva durante tareas de alta demanda computacional. Dada la dependencia del sistema con servicios externos —en particular la API de OpenAI—, estas pruebas son esenciales para garantizar un funcionamiento robusto y cohesivo.

Las pruebas de aceptación final se realizan con un grupo mínimo de diez usuarios con discapacidad visual, idealmente en colaboración con organizaciones especializadas en inclusión. Estas sesiones permiten recopilar retroalimentación sobre aspectos clave como la facilidad de uso, la comprensión del audio y la satisfacción general. Los participantes evalúan elementos como la claridad de los botones o la efectividad del clic derecho como método de interacción.

Para facilitar la participación, las encuestas de retroalimentación se ofrecen en formatos accesibles como texto ampliado o audio.

La accesibilidad es un eje transversal en todo el proceso de validación. Se garantiza el cumplimiento del nivel AA de las directrices WCAG 2.1, mediante la evaluación de contraste, navegación por teclado y compatibilidad con lectores de pantalla como NVDA (Windows) y Orca (Linux). Se emplean herramientas como WAVE o axe DevTools para el análisis automatizado, complementado con pruebas manuales que aseguren una experiencia inclusiva. Particular atención se presta a la detección de elementos por parte de los lectores y a la disponibilidad de accesos alternativos mediante teclas rápidas.

Los criterios de éxito del plan de pruebas se definen con métricas específicas. Se establece que al menos el 95 % de los casos de prueba deben completarse sin errores críticos, tales como fallos en la API o congelamiento de la interfaz. El tiempo de respuesta —desde la captura hasta la reproducción del audio— debe mantenerse por debajo de 20 segundos en el 90 % de los escenarios con conexión estable. En las pruebas de aceptación, se espera que al menos el 80 % de los participantes califiquen la experiencia como satisfactoria, destacando la facilidad de uso y la claridad del audio. Además, se exige estabilidad del sistema durante al menos 100 sesiones consecutivas, sin fallos en capturas repetitivas.

Finalmente, se promueve la mejora continua mediante la incorporación sistemática de retroalimentación en ciclos iterativos de desarrollo. Se propone ajustar variables como la velocidad de lectura o la distribución de los elementos visuales en función de la experiencia del usuario. Asimismo, se analizarán los registros de errores para identificar patrones recurrentes y optimizar el código. Aunque el prototipo actual depende de servicios en línea, se contempla a futuro la integración de modelos locales de inteligencia artificial que permitan operar en entornos

sin conexión, ampliando así el alcance y la accesibilidad de *OpenEye*. Este enfoque integral garantiza no solo el cumplimiento de los objetivos técnicos del proyecto, sino también su relevancia y utilidad para la comunidad a la que está dirigido.

Conclusiones

El desarrollo del sistema OpenEye demostró que es posible construir una solución funcional, accesible y de bajo costo que permita a personas con discapacidad visual acceder a la información visual contenida en pantallas digitales. A través de la integración de tecnologías como visión artificial, modelos de lenguaje avanzados y síntesis de voz, se logró crear un asistente virtual capaz de interpretar y transformar contenido visual en audio de forma automatizada y en tiempo real.

La metodología Kanban permitió organizar el trabajo de forma ágil y estructurada, favoreciendo la entrega continua de funcionalidades y la adaptación del proyecto a las limitaciones técnicas, de tiempo y de recursos. El enfoque modular del sistema, junto con el uso de herramientas de código abierto, facilitó su desarrollo y garantiza su escalabilidad para futuras mejoras.

Entre los principales logros del proyecto se destacan la implementación efectiva de hilos para mantener la fluidez de la interfaz, la generación de descripciones claras mediante GPT-4o, y una interfaz adaptada con elementos accesibles. Asimismo, la activación por clic derecho ofrece una interacción sencilla y efectiva para usuarios con baja visión.

Finalmente, OpenEye no solo constituye una herramienta funcional, sino también una propuesta con impacto social real. Su desarrollo evidencia cómo la inteligencia artificial puede ser aplicada con responsabilidad para cerrar brechas de accesibilidad, promoviendo la inclusión digital y mejorando la calidad de vida de personas con discapacidad visual.

Referencias

Alzalabny, S., Moured, O., Müller, K., Schwarz, T., Rapp, B., & Stiefelhagen, R. (2024). Designing a Tactile Document UI for 2D Refreshable Tactile Displays: Towards Accessible Document Layouts for Blind People. *Multimodal Technologies and Interaction*, *8*(11), 102. https://doi.org/10.3390/mti8110102

Bastidas-Guacho, G. K., Alejandro, P. A. M., Moreno-Vallejo, P. X., Moreno-Costales, P. R., Samanta, O. Y. N., & Carlos, T. C. J. (2025). Computer Vision-Based Obstacle Detection Mobile System for Visually Impaired Individuals. *Multimodal Technologies and Interaction*, *9*(5), 48. https://doi.org/10.3390/mti9050048

Deshakulkarni Bhaskar, N., & Ramakrishnan, I. V. (2024). Enhancing Accessibility of Desktop Applications for Blind Users Through Multimodal AI Integration. In *ProQuest Dissertations and Theses*. https://login.bdbiblioteca.universidadean.edu.co/login?url=https://www.proquest.com/dissertations-theses/enhancing-accessibility-desktop-applications/docview/3109572181/se-2?accountid=34925

Djatmiko, G. H., Sinaga, O., & Pawirosumarto, S. (2025). Digital Transformation and Social Inclusion in Public Services: A Qualitative Analysis of E-Government Adoption for Marginalized Communities in Sustainable Governance. *Sustainability*, *17*(7), 2908. https://doi.org/10.3390/su17072908

Georgios, V., Ioannis, D., Nikolaos, T., Kokkonis, G., & Sotirios, K. (2025). Development and Evaluation of a Tool for Blind Users Utilizing AI Object Detection and Haptic Feedback. *Machines*, *13*(5), 398. https://doi.org/https://doi.org/10.3390/machines13050398

Hügle, T. (2024). Advancing Rheumatology Care Through Machine Learning. *Pharmaceutical Medicine*, *38*(2), 87–96. https://doi.org/10.1007/s40290-024-00515-0

Reuters. (2024, July 18). *OpenAI unveils cheaper small AI model GPT-4o mini*. https://www.reuters.com/technology/artificial-intelligence/openai-unveils-cheaper-small-ai-model-gpt-4o-mini-2024-07-18/

Song, Y., & Xiong, W. (2025). Large Language Model-Driven 3D Hyper-Realistic Interactive Intelligent Digital Human System. *Sensors*, *25*(6), 1855. https://doi.org/10.3390/s25061855

Surjit, P. (2023). Accessibility analysis using WCAG 2.1: evidence from Indian e-government websites. *Universal Access in the Information Society*, *22*(2), 663–669. https://doi.org/10.1007/s10209-021-00861-9

Toro, D., & Wu, C. (2025). Enhancing Accessibility in Digital Audio Workstations: Barriers, Breakthroughs, and the Future of Inclusive Plugin Design for Visually Impaired Users. In *ProQuest Dissertations and Theses*. https://login.bdbiblioteca.universidadean.edu.co/login?url=https://www.proquest.com/dissertations-theses/enhancing-accessibility-digital-audio/docview/3204181606/se-2?accountid=34925

Toyokawa, Y., Horikoshi, I., Majumdar, R., & Ogata, H. (2023). Challenges and opportunities of AI in inclusive education: a case study of data-enhanced active reading in Japan. *Smart Learning Environments*, *10*(1). https://doi.org/10.1186/s40561-023-00286-2

Verónica-Alexandra, M.L., Andrea, B.A., Carla-Belén, G.M., & Hernández-Martínez, E. (2025). The Impact of Artificial Intelligence on Inclusive Education: A Systematic Review. *Education Sciences*, *15*(5), 539. https://doi.org/10.3390/educsci15050539

Congreso de Colombia. (2013). *Ley 1680 de 2013: Por la cual se garantiza el acceso a la información, a las comunicaciones y al conocimiento a las personas con discapacidad visual y se dictan otras disposiciones*. https://www.wipo.int/wipolex/es/legislation/details/14792