Proyecto Final

FigurOut: Extensión de Accesibilidad Digital al Lenguaje Figurado mediante Inteligencia Artificial

Autor: Ulises Fritz Serain

Carrera: Tecnicatura Universitaria en Programación

Tutor: Dr. Miguel Méndez-Garabetti

Agosto 2025

Universidad CAECE

Tecnicatura Universitaria en Programación

1. Resumen

El presente proyecto desarrolla FigurOut, una extensión de navegador orientada a mejorar la accesibilidad web para personas que enfrentan dificultades al interpretar el lenguaje figurado, como metáforas, ironías o hipérboles. Estas expresiones, comunes en la comunicación digital, pueden convertirse en barreras cognitivas para niños, estudiantes de idiomas o personas con trastornos del desarrollo o dificultades semánticas.

La solución consiste en una extensión que permite al usuario seleccionar un texto con lenguaje figurado y recibir una versión literal o explicativa, adaptada a su perfil. Para lograr esto, el sistema utiliza modelos de inteligencia artificial provistos por OpenAI, que procesan los textos en tiempo real a partir de configuraciones personalizables. FigurOut incluye un panel de control intuitivo, opciones de interpretación ajustables y filtros de moderación que garantizan la seguridad del contenido.

El desarrollo se realizó con tecnologías como Node.js, JavaScript y EJS, integrando interfaces accesibles y un backend optimizado. Además, se evaluaron distintas estrategias de generación de texto mediante una prueba profesional, que permitió afinar los prompts de respuesta según su claridad, adecuación y fidelidad semántica.

Se concluye que la aplicación constituye una herramienta innovadora, funcional y con impacto social inmediato, al abordar una dimensión poco explorada de la accesibilidad digital: la comprensión del lenguaje figurado. El sistema actual sienta bases sólidas para su expansión futura, tanto en términos técnicos como en su aplicación a nuevos perfiles de usuario.

Palabras claves: API, Inteligencia Artificial, Lenguaje Figurado, Accesibilidad Web, extensión de navegador.

Índice

2. Introducción

El presente trabajo final de tecnicatura presenta el desarrollo de *FigurOut*, una extensión de navegador diseñada para mejorar la accesibilidad web en contextos donde el lenguaje figurado representa un obstáculo para la comprensión. Este tipo de lenguaje, que incluye expresiones como metáforas, ironías, hipérboles y frases idiomáticas, está presente en noticias, redes sociales, literatura digital, entre otros entornos. Si bien enriquece la comunicación, puede generar confusión, malentendidos o exclusión para ciertos grupos de usuarios.

La aplicación surge como respuesta a esta problemática, ofreciendo una herramienta que detecta y reformula expresiones figuradas en tiempo real, brindando al usuario una explicación literal o contextualizada, según su perfil. Está orientada especialmente a personas a las que estos recursos se le presentan cómo una barrera, estudiantes de lenguas extranjeras y niños en etapa de desarrollo lingüístico. La solución se apoya en inteligencia artificial, integrando la API de OpenAI, y proporciona una interfaz adaptable y segura, con funciones de moderación y personalización.

El documento se encuentra estructurado en varios apartados, siendo el número 3 el núcleo del desarrollo, donde se detallan las etapas del proyecto en los siguientes bloques:

- Capítulo 1 – Marco Teórico: Se contextualiza la problemática desde el enfoque de la accesibilidad web, la comprensión del lenguaje figurado, la tecnología adaptativa y el procesamiento del lenguaje natural. También se presenta el concepto de accesibilidad cognitiva como parte del derecho a la inclusión digital.

- Capítulo 2 – Objetivos: Se detallan el objetivo general del proyecto y los objetivos específicos que guiaron el desarrollo de *FigurOut*, incluyendo aspectos técnicos, metodológicos y sociales.

- Capítulo 3 – Diseño, Implementación y Evaluación del Sistema: Se describe cómo fue concebido y construido el sistema. Incluye requisitos, casos de uso, metodología de trabajo, evaluación profesional de respuestas generadas por IA y la visualización de interfaces del sistema.

- Capítulo 4 – Arquitectura del Sistema: Se explica el enfoque cliente-servidor adoptado, se presentan diagramas de contexto y flujo, los principios de diseño utilizados, los componentes principales, las tecnologías implementadas y los trabajos futuros proyectados.

Finalmente, el documento incluye un apartado de conclusiones, seguido de las referencias bibliográficas. A su vez también incluye anexos, donde se encuentra la construcción del formulario de evaluación, resultados de encuestas profesionales, un tester de la API moderadora de respuestas y contenido técnico relevante como los prompts utilizados y especificaciones del sistema.

La composición del documento se realizó siguiendo el Reglamento de Trabajos Finales propuesto por la Universidad CAECE [7].

3. Desarrollo

3.1 Capítulo 1: Marco Teórico

A continuación, se presentan los fundamentos teóricos que sustentan el desarrollo del sistema.

3.1.1 Accesibilidad Web y Comprensión del Lenguaje

La accesibilidad web, definida por la W3C como la práctica de garantizar que todas las personas puedan percibir, comprender, navegar e interactuar con la web, ha evolucionado para considerar no solo discapacidades físicas, sino también cognitivas y lingüísticas [8]. Dentro de este marco, el lenguaje utilizado en los contenidos digitales cobra una importancia central.

La accesibilidad cognitiva es una de las dimensiones más ignoradas en el diseño digital. La comprensión del lenguaje figurado, como metáforas, ironías o exageraciones, requiere habilidades interpretativas que no todos los usuarios poseen. Esto representa una barrera de acceso a la información, especialmente para personas con Trastornos del Espectro Autista (TEA), dislexia, o para quienes están aprendiendo un nuevo idioma [9][11].

Evidencia empírica reciente muestra este desbalance: un estudio sobre portales de COVID-19 halló que la gran mayoría de barreras detectadas se concentraban en el criterio “perceptible” (77,8%), mientras que “entendible” (donde suelen ubicarse aspectos cognitivos) apenas representó el 0,9% de los hallazgos. Los autores advierten que esto refleja limitaciones de las herramientas automáticas actuales y la necesidad de evaluar criterios cognitivos con métodos complementarios [27].

3.1.2 El Lenguaje Figurado como Obstáculo de Comprensión

El lenguaje figurado se aleja del significado literal y depende del contexto cultural y social. Su procesamiento implica inferencias complejas, lo cual puede resultar especialmente problemático en entornos automatizados o hipertextuales como la web.

Estudios han evidenciado que usuarios con dificultades cognitivas enfrentan desafíos significativos al interpretar expresiones figuradas en textos escolares, redes sociales o noticias, afectando su capacidad de comprensión y participación plena en entornos digitales [9][10][25].

3.1.3 Tecnología Adaptativa y Procesamiento del Lenguaje Natural (NLP)

Frente a la problemática que representa el lenguaje figurado para ciertos grupos de usuarios, la inteligencia artificial y el procesamiento de lenguaje natural (NLP) emergen como herramientas clave para detectar, interpretar y transformar expresiones complejas en versiones más accesibles. Entre estas tecnologías, se destaca el uso de modelos generativos como los provistos por OpenAI, cuya API permite acceder a modelos de lenguaje avanzados (como GPT-3.5 y GPT-4o-mini) entrenados sobre grandes volúmenes de texto y capaces de generar explicaciones claras, concisas y detalladas según el contexto y el perfil del usuario [5][12].

La capacidad de estos modelos de generar texto a partir de indicaciones específicas (prompts), sumada a sus sistemas de moderación y control de contenido, la hacen especialmente útil para aplicaciones educativas, accesibles y seguras.

3.1.4 Accesibilidad Cognitiva como Derecho Digital

Desde la perspectiva de la ética digital, el acceso comprensible a la información se vincula con el derecho a la inclusión. En consecuencia, la incorporación de herramientas automatizadas que simplifiquen o expliquen el lenguaje figurado se inscribe dentro de una tendencia mayor por construir entornos digitales más justos, comprensibles y accesibles [11][8].

El propio W3C, a través de la iniciativa *Cognitive Accessibility at W3C*, enfatiza que las personas con limitaciones cognitivas suelen quedar fuera de la planificación de accesibilidad y plantea lineamientos específicos para atender esta deuda histórica [26]. En este marco, soluciones como FigurOut cobran relevancia, ya que aportan una vía práctica para hacer más entendible el contenido digital y garantizar la igualdad de acceso a la información [26].

3.2 Capítulo 2. Objetivos

En este capítulo se establecen los propósitos que guiaron el desarrollo de FigurOut, tanto desde una perspectiva general como en sus objetivos técnicos específicos.

3.2.1 Objetivo General

Desarrollar y validar un prototipo funcional de extensión de navegador orientado a mejorar la accesibilidad web para usuarios que enfrentan dificultades en la interpretación del lenguaje figurado, facilitando la comprensión de expresiones complejas (como metáforas, ironías o exageraciones) mediante el uso de inteligencia artificial y estrategias de interpretación adaptadas a distintos perfiles de usuario.

3.2.2 Objetivos Específicos

- Diseñar e implementar una interfaz accesible, intuitiva y adaptable que permita a los usuarios seleccionar distintos niveles de explicación del lenguaje figurado según sus necesidades.
- Integrar la API de OpenAI como motor principal para el procesamiento de texto y la generación de interpretaciones figurativas en tiempo real, configurando los prompts según los perfiles definidos de usuario.
- Aplicar técnicas de preprocesamiento textual con el fin de optimizar el consumo de tokens y garantizar la sostenibilidad del sistema bajo un modelo freemium.
- Investigar y analizar modelos de inteligencia artificial y alternativas open source para la detección, interpretación y moderación de lenguaje figurado, evaluando su desempeño y adaptabilidad al contexto hispanohablante.
- Relevar y analizar la problemática asociada a la comprensión del lenguaje figurado, explorando artículos y casos que fundamenten la importancia y pertinencia de la solución propuesta.
- Desarrollar y documentar la arquitectura de la extensión de navegador.
- Conformar un sistema de prompts adecuados al perfil y el tipo de respuesta configurado por el usuario, analizarlos bajo el punto de vista profesional y comparar las respuestas.

3.3 Capítulo 3. Diseño, Implementación y Evaluación del Sistema

Este capítulo aborda el proceso de construcción del sistema FigurOut, desde sus decisiones iniciales de diseño hasta la implementación funcional y su posterior evaluación. Se describen las características clave de la solución propuesta, los requisitos que guiaron su

desarrollo, la metodología adoptada y la validación realizada con participación de profesionales del área.

3.3.1 Visión general del sistema.

FigurOut es una herramienta tecnológica orientada a mejorar la accesibilidad web mediante la interpretación del lenguaje figurado, se usa cómo extensión de navegador. Su función principal es detectar expresiones como metáforas, ironías o hipérboles en textos seleccionados por el usuario, y ofrecer una versión alternativa que sea más literal, explicativa o comprensible, según las necesidades del perfil del usuario. Esta solución está pensada para personas que la comprensión lingüística les resulta una barrera, niños o estudiantes de idiomas, y se integra de forma directa en la experiencia de navegación.

3.3.2 Requisitos del sistema.

A continuación, se detallan los requisitos funcionales y no funcionales que el sistema debía cumplir para alcanzar sus objetivos. Estos requisitos fueron definidos en función de las necesidades del usuario, las capacidades tecnológicas disponibles y los principios de accesibilidad cognitiva que sustentan el proyecto. Además se detalla el conjunto de casos de uso de la aplicación.

Requerimientos Funcionales.

- **RF01.** El sistema debe permitir al usuario seleccionar un plan de uso (gratuito, básico o premium).
- **RF02.** El sistema debe ofrecer al usuario dos modos de interpretación del lenguaje figurado (a- interpretación literal y concisa y b- explicativa y contextual).
- **RF03**. El sistema debe permitir al usuario seleccionar texto en la web y presentar un botón para que el respectivo sea analizado.

- **RF04.** El sistema debe detectar las configuraciones del usuario y generar la respuesta en función.
- **RF05**. El sistema debe enviar el contenido textual a la API de OpenAI para su análisis e interpretación.
- **RF06.** El sistema debe pasar por el filtro de Moderation API el resultado generado por la IA, bloqueando respuestas ofensivas.
- **RF07.** El sistema debe mostrar la respuesta en un tooltip.
- **RF08**. El sistema debe permitir al usuario que cierre el tooltip.
- **RF09.** El sistema debe optimizar el uso de tokens según el plan contratado y el modo seleccionado.
- **RF10.** El sistema debe almacenar y recuperar la configuración del usuario mediante chrome.storage.
- **RF11.** El sistema debe permitir cambiar de perfil o modo de uso desde el panel de control.

Combinaciones posibles entre perfiles y tipos de respuesta:

- El sistema contempla distintos perfiles de usuario y ofrece dos modos de interpretación del lenguaje figurado. A continuación, en la Tabla 1 se presentan las combinaciones previstas, que definen cómo debe comportarse el sistema ante cada configuración:

OP	Perfil de Usuario	Tipo de Respuesta
1.a	1. Usuario con barreras de comprensión	a. Literal y concisa
1.b	1. Usuario con barreras de comprensión	b. Explicación y contextualización
2.a	2. Usuario infantil	a. Literal y concisa
2.b	2. Usuario infantil	b. Explicación y contextualización
3.a	3. Usuario estudiante de idioma extranjero	a. Literal y concisa
3.b	3. Usuario estudiante de idioma extranjero	b. Explicación y contextualización

Tabla 1 - Cuadro de combinaciones de perfiles del sistema.

Requerimientos No Funcionales

- **RNF01.** El sistema debe funcionar como una extensión de navegador compatible con Google Chrome.
- **RNF02.** La interfaz debe ser accesible y comprensible para usuarios con diferentes niveles de competencia digital.
- **RNF03**. El sistema debe asegurar la persistencia de configuraciones entre sesiones mediante almacenamiento local.
- **RNF04.** El sistema debe ser seguro frente a respuestas inadecuadas de la IA mediante filtros de validación previos a su despliegue.

Casos de Uso

- Caso de Uso 001: Elegir plan de contratación.
- Caso de Uso 002: Seleccionar una opción en el panel de control.
- Caso de Uso 003: Aplicar traducción literal o explicativa según configuración del usuario
- Caso de Uso 004: Seleccionar texto en la web.
- Caso de Uso 005: Almacenar preferencias del usuario en chrome.storage
- Caso de Uso 006: Enviar texto al back-end y recibir respuesta procesada
- Caso de Uso 007: Mostrar resultados mediante tooltip.
- Caso de Uso 008: Cerrar tooltip.
- Caso de Uso 009: Seleccionar “prueba nuestro intérprete”.
- Caso de Uso 010: Enviar texto en el intérprete.

3.3.4 Metodología.

El desarrollo de *FigurOut* se llevó a cabo mediante una metodología ágil e iterativa, centrada en el incremento funcional del sistema y en la mejora continua a partir del feedback recibido durante su construcción. Esta elección metodológica permitió ajustar rápidamente aspectos técnicos y de experiencia de usuario a lo largo del proyecto.

Durante todo el proceso se trabajó en un repositorio remoto de GitHub, que funcionó como entorno de control de versiones y bitácora de avance.

Vínculo del repositorio: https://github.com/UlisesFritzSerain/node-corrector

3.3.5 Evaluación Profesional de Respuestas Generadas por IA

Se usó la Metodología exploratoria-experimental basada en pruebas con diferentes combinaciones de prompts y análisis cualitativo de los resultados generados con participación de profesionales del área de la salud mental. Previamente se analizaron 90 frases pero se seleccionaron las que más fallas presentaban para que puedan ser juzgadas por los profesionales.

Métricas a evaluar:

- **Claridad:** ¿La explicación es comprensible según el perfil?
- **Adecuación léxica:** ¿Usa un vocabulario adecuado al usuario?
- **Fidelidad semántica:** ¿Respeta la idea original?

Perfiles evaluados: Usuario con barreras de comprensión. (perfil 1).

- Tipos de respuesta evaluadas:
    a. Literal y concisa.
    b. Explicativa y contextual.

Resultados de evaluación profesional sobre las respuestas generadas por IA.

- Resultado sobre respuestas de Usuario al que el lenguaje figurado le resulta una barrera (perfil 1) - respuesta literal y concisa (a):

❖ Prompt seleccionado como el más eficaz: **Index1**.

❖ Este prompt ofreció las respuestas más completas y equilibradas. La claridad fue consistente en casi todos los casos, y las explicaciones fueron fieles al sentido original, sin agregar ambigüedad.

A continuación, en la Figura 1 se detallan los resultados globales sobre el análisis de las respuestas de los diferentes prompts de este perfil:

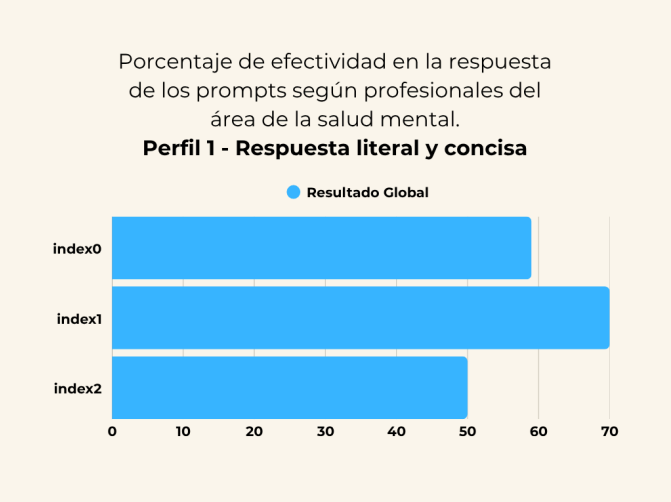


Figura 1 - Resultados sobre el análisis de las respuestas de los prompts del perfil 1-a.

En la Figura 2 se presenta el contenido del Prompt perfil 1-a index1:

```
{
  systemPrompt: "Tu tarea es traducir frases complicadas o con lenguaje figurado a una versión
  simple, clara y literal para facilitar su comprensión.",
  buildUserPrompt: (text) => `Decime qué significa esta frase, sin adornos ni metáforas: ${text}`
},
```

Figura 2 - Prompt seleccionado como el más eficiente para perfil 1-a.

Resultado sobre respuestas de Usuario al que el lenguaje figurado le resulta una barrera (perfil 1) - respuesta explicativa y contextual (b):

- Prompt seleccionado como el más eficaz: **Index0**.
- Este prompt sostuvo una paridad considerable con el index1, pero su diferencial según los profesionales se pudo ver en la claridad de la explicación que mostraban sus respuestas.

A continuación los resultados globales sobre el análisis de las respuestas de los diferentes prompts de este perfil detallados en la Figura 3:

Figura 3 - Resultados sobre el análisis de las respuestas de los prompts del perfil 1-b.

Se presenta el contenido del Prompt perfil 1-b index0 en la Figura 4:

```
{
  systemPrompt: "Tu función es facilitar la comprensión de frases difíciles, explicando el sentido
  figurado y dando un ejemplo simple para que cualquier adulto lo entienda.",
  buildUserPrompt: (text) => `Explicá qué significa esta frase figurada de forma breve y clara: $
  {text}`
},
```

Figura 4 - Prompt seleccionado como el más eficiente para perfil 1-b.

3.3.6 Diseño de interfaces del sistema

La Figura 5 refleja la interfaz del panel de control de *FigurOut (/)* dónde se tiene acceso a la información del sitio y a la personalización de opciones:

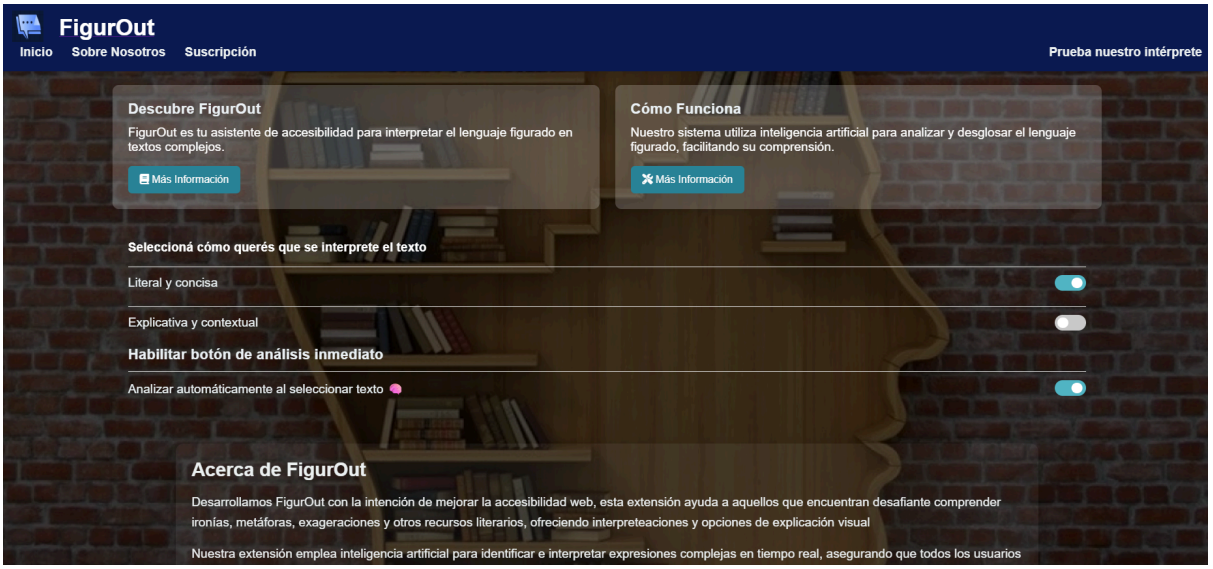


Figura 5 - Captura de pantalla del Panel de control de FigurOut.

En la Figura 6 se observa la interfaz del modo intérprete manual de *FigurOut* (/traductor):

Figura 6 - Captura de pantalla del intérprete manual de la aplicación (/traductor).

La Figura 7 muestra el Popup que da acceso al panel de control de *FigurOut* desde la barra de navegación:

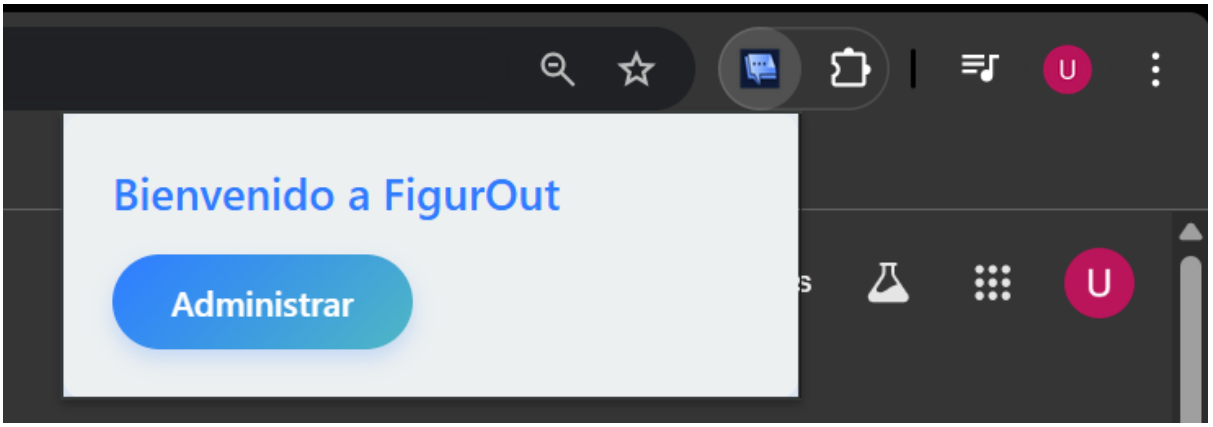


Figura 7 - Captura de Pantalla del Popup de FigurOut.

En la Figura 8 se observa el botón Mouse-up con emoji para solicitar la interpretación del texto:

Figura 8 - Captura de pantalla del botón mouse-up con emoji de cerebro.

En la Figura 9 podemos ver la presentación de la opción del menú contextual para interpretar el texto seleccionado.

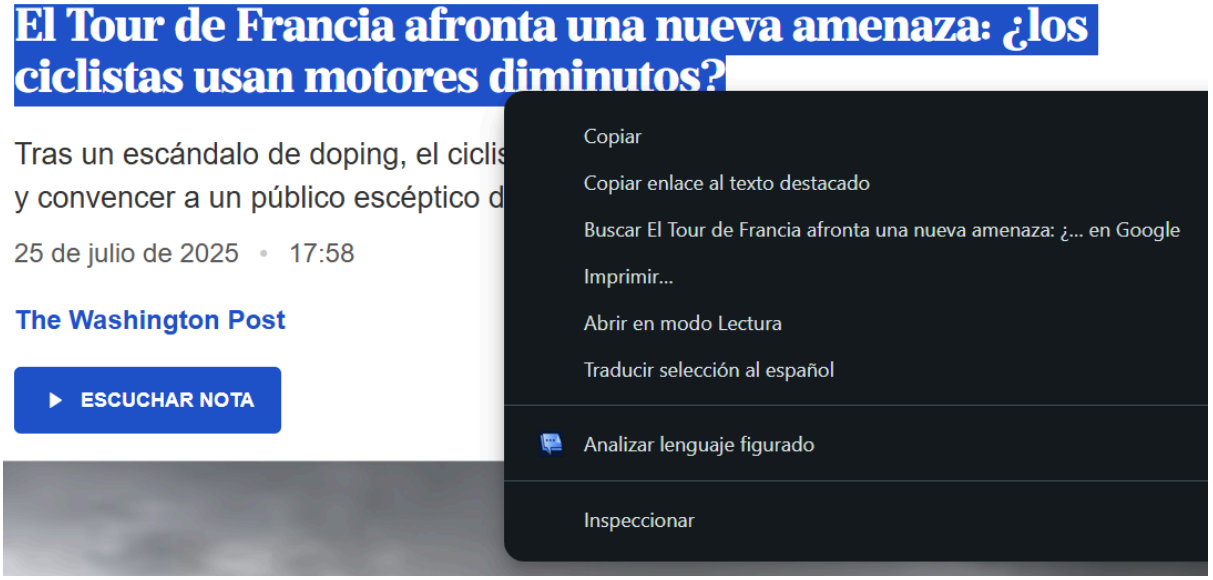


Figura 9 - Captura de pantalla de la opción de menú contextual.

Visualización del cartel de procesamiento del texto para lograr feedback con el usuario en la Figura 10:

Figura 10 - Captura de pantalla del tooltip “Analizando”.

La Figura 11 enseña el tooltip con la respuesta procesada por el sistema interpretando posible lenguaje figurado:

Figura 11 - Tooltip con la respuesta del sistema.

3.4 Capítulo 4. Arquitectura del sistema

En este capítulo se describe la arquitectura técnica de FigurOut, detallando la organización de sus componentes principales, los principios de diseño adoptados, las tecnologías utilizadas y el flujo de funcionamiento del sistema. También se incluyen esquemas gráficos que ilustran cómo interactúan la extensión de navegador, el backend y los servicios de inteligencia artificial integrados.

3.4.1 Enfoque

El enfoque adoptado en la implementación corresponde a una arquitectura cliente-servidor, donde el cliente está representado por la extensión instalada en el navegador, y el servidor consiste en un backend desarrollado con Node.js que procesa los textos seleccionados. Este backend interactúa con la API de OpenAI para generar respuestas interpretativas y, en caso necesario, aplica filtros de moderación antes de mostrar los resultados al usuario.

El sistema fue desarrollado inicialmente como una extensión para el navegador Google Chrome, considerando que este navegador es el más utilizado a nivel mundial y que su entorno de desarrollo ofrece un soporte sólido, documentación extensa y una comunidad activa [17].

Chrome sirvió como plataforma base debido a la madurez de su API para extensiones, lo que permitió construir prototipos funcionales rápidamente e integrar características como contextMenus, runtime messaging (comunicación entre componentes) y a futuro, chrome.storage [18].

Si bien el desarrollo actual está optimizado para el ecosistema de Chrome, se proyecta la adaptación futura a otros navegadores compatibles con WebExtensions, como Microsoft Edge, Brave o Mozilla Firefox, garantizando una mayor portabilidad y cobertura de usuarios [19].

3.4.2 Diagrama de contexto.

A continuación, la Figura 12 presenta el diagrama de contexto del sistema *FigurOut*. Este esquema permite visualizar, de manera simplificada, cómo interactúan los componentes de la aplicación entre sí.

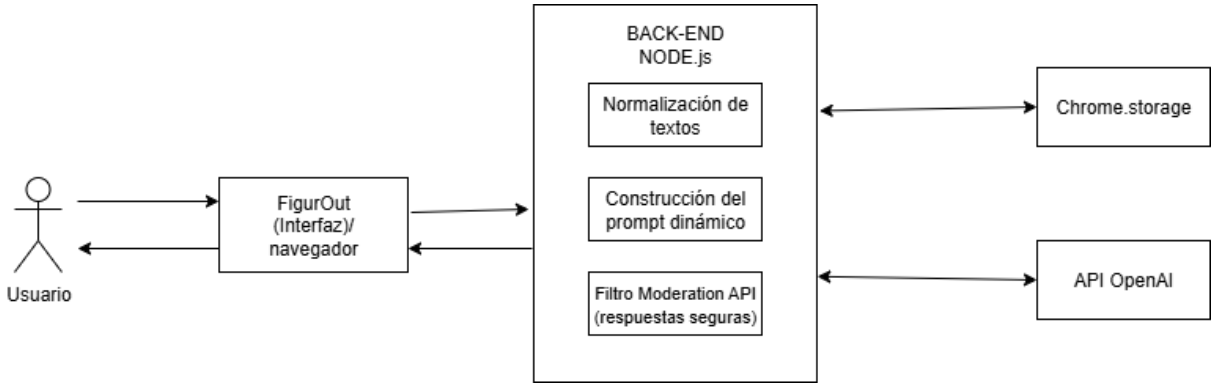


Figura 12 - Diagrama de contexto del sistema FigurOut.

3.4.3 Principios de diseño.

El diseño del sistema responde a principios de modularidad, permitiendo que cada componente (preprocesamiento, generación de prompts, procesamiento de IA, almacenamiento de logs, moderación, etc.) sea reutilizable o escalable; escalabilidad, al permitir futuras expansiones como nuevos perfiles, bases de datos o autenticación de usuarios; y accesibilidad, asegurando que las respuestas generadas sean comprensibles y adaptadas al nivel lingüístico o cognitivo del usuario final.

3.4.4 Componentes principales.

A continuación se describen los componentes más relevantes del sistema:

- **Extensión de navegador:** Es el punto de interacción directa con el usuario. Permite seleccionar fragmentos de texto en cualquier sitio web y enviarlos al backend para su análisis. Cuenta con un botón flotante, un menú contextual y un sistema de tooltips para presentar los resultados de forma clara e inmediata. Esta extensión captura el texto seleccionado y lo transmite mediante una solicitud al servidor.
- **Panel de control web front-end (EJS-CSS-JavaScript):** Se trata de una interfaz web accesible desde el navegador o el popup de la extensión, donde el usuario puede acceder a la información de la aplicación, configurar su perfil y tipo de respuesta y además ingresar texto manualmente para obtener una interpretación figurada en una vista de estilo “traductor”. Esta vista muestra tanto el texto original como el resultado procesado.
    - Para un fácil acceso al panel, la extensión cuenta con un **popup** que desde la barra de navegación redirige a la vista mencionada.

- **Servidor backend (Node.js + Express):** Es el núcleo del procesamiento. Recibe los textos desde la extensión, normaliza el contenido para reducir ambigüedad y consumo de tokens, genera el prompt adecuado según el perfil de usuario y tipo de respuesta, y se comunica con la API de OpenAI para obtener la interpretación. Luego, devuelve el resultado al navegador para ser mostrado al usuario.
- **API de OpenAI (GPT-4o-mini):** Es el componente central de procesamiento inteligente. A través de ella se generan las reescrituras interpretativas del lenguaje figurado utilizando modelos avanzados de lenguaje natural. La comunicación se realiza mediante solicitudes POST que incluyen un prompt específico.
- **Moderation API (filtro de contenido sensible):** Antes de mostrar el resultado al usuario, el sistema puede aplicar un filtro de moderación que evalúa si la respuesta generada contiene contenido ofensivo, violento o inadecuado. En caso de detección, se evita su visualización y se ofrece una alternativa segura. Además almacena las respuestas inadecuadas para un futuro estudio.

3.4.5 Tecnologías utilizadas.

La elección de estas herramientas para el desarrollo de *FigurOut* responde a criterios de eficiencia, escalabilidad, compatibilidad multiplataforma y facilidad de mantenimiento. A continuación, se justifican las principales decisiones:

- **Node.js + Express:**

  Constituyen la base del servidor backend. Node.js permite ejecutar JavaScript del lado del servidor de forma eficiente y no bloqueante, mientras que Express es un framework minimalista que facilita la creación de rutas, manejo de solicitudes HTTP y gestión de vistas. Esta combinación permitió desarrollar un backend ligero, rápido y fácilmente extensible [1][22].

- **JavaScript (front-end)**:

  Se utilizó para implementar el comportamiento interactivo de la extensión, capturar texto seleccionado por el usuario, gestionar eventos como clicks o selección de texto, y mostrar resultados mediante tooltips flotantes. También se usó para comunicar la extensión con el backend mediante fetch [23].

- **HTML + CSS**:

  Estas tecnologías se usaron para definir la estructura y estilo visual tanto de la extensión como del panel web [3][4].

- **EJS (Embedded JavaScript)**:

  Motor de plantillas utilizado en el panel web del sistema, que permite renderizar dinámicamente contenido HTML a partir de datos recibidos desde el backend. Esto fue clave para construir el modo “intérprete manual", donde se muestran los textos originales y corregidos en tiempo real [24].

- **Bootstrap**:

  Se utilizó como framework de estilos para la construcción del intérprete manual accesible desde el panel de control. Esta herramienta permitió acelerar el desarrollo de la vista de dicha sección [2].

- **Moderation API:**

  Como mecanismo de validación final del contenido generado por el sistema, *FigurOut* incorpora la Moderation API de OpenAI en su backend. Esta herramienta

realiza un análisis automático del texto producido por el modelo GPT, antes de ser mostrado al usuario, con el objetivo de detectar y bloquear respuestas que contengan lenguaje ofensivo, violento, sexual o inadecuado para ciertos perfiles [6].

El filtro se ejecuta inmediatamente después de recibir la respuesta del modelo de lenguaje y antes de enviarla al frontend, garantizando que ningún contenido potencialmente inapropiado llegue al usuario final, incluso si fue generado por una interpretación errónea o descontextualizada del modelo.

La integración se realiza mediante el uso de node-fetch, enviando el texto generado a la Moderation API. Si el campo flagged === true, se considera que la respuesta no es apta y se bloquea automáticamente. Además, todas las respuestas rechazadas quedan registradas localmente en un archivo .json, junto con el timestamp, el texto original y el contenido generado, permitiendo auditorías o revisiones posteriores.

A continuación, en la Figura 13 se muestra un ejemplo de respuesta de la Moderation API. En este caso, flagged es true debido a que al menos una categoría (como harassment o violence) fue identificada como riesgosa. La confianza de detección se refleja en el campo category_scores, con valores entre 0 y 1.

```
{
"id": "modr-abc123",
"model": "text-moderation-007",
"results": [
  {
    "flagged": true,
    "categories": {
      "sexual": false,
      "hate": false,
      "harassment": true,
      "self-harm": false,
      "violence": true,
      "violence/graphic": false
    },
    "category_scores": {
      "sexual": 0.0001,
      "hate": 0.002,
      "harassment": 0.9231,
      "self-harm": 0.0000,
      "violence": 0.8724,
      "violence/graphic": 0.0002
    }
  }
]
}
```


Figura 13 - Objeto respuesta de Moderation API.

3.4.6 Diagrama de flujo

A continuación, se presenta en la Figura 14 el diagrama de flujo que describe el recorrido que realiza un texto seleccionado por el usuario a través del sistema *FigurOut*. Este esquema detalla las etapas que componen el proceso de interpretación: desde la captura del fragmento en el navegador, pasando por la normalización y análisis del backend, hasta la generación de la respuesta adaptada mediante inteligencia artificial. También se incluyen los filtros de moderación y la entrega del resultado al usuario. Este flujo refleja la secuencia lógica y operativa del sistema.

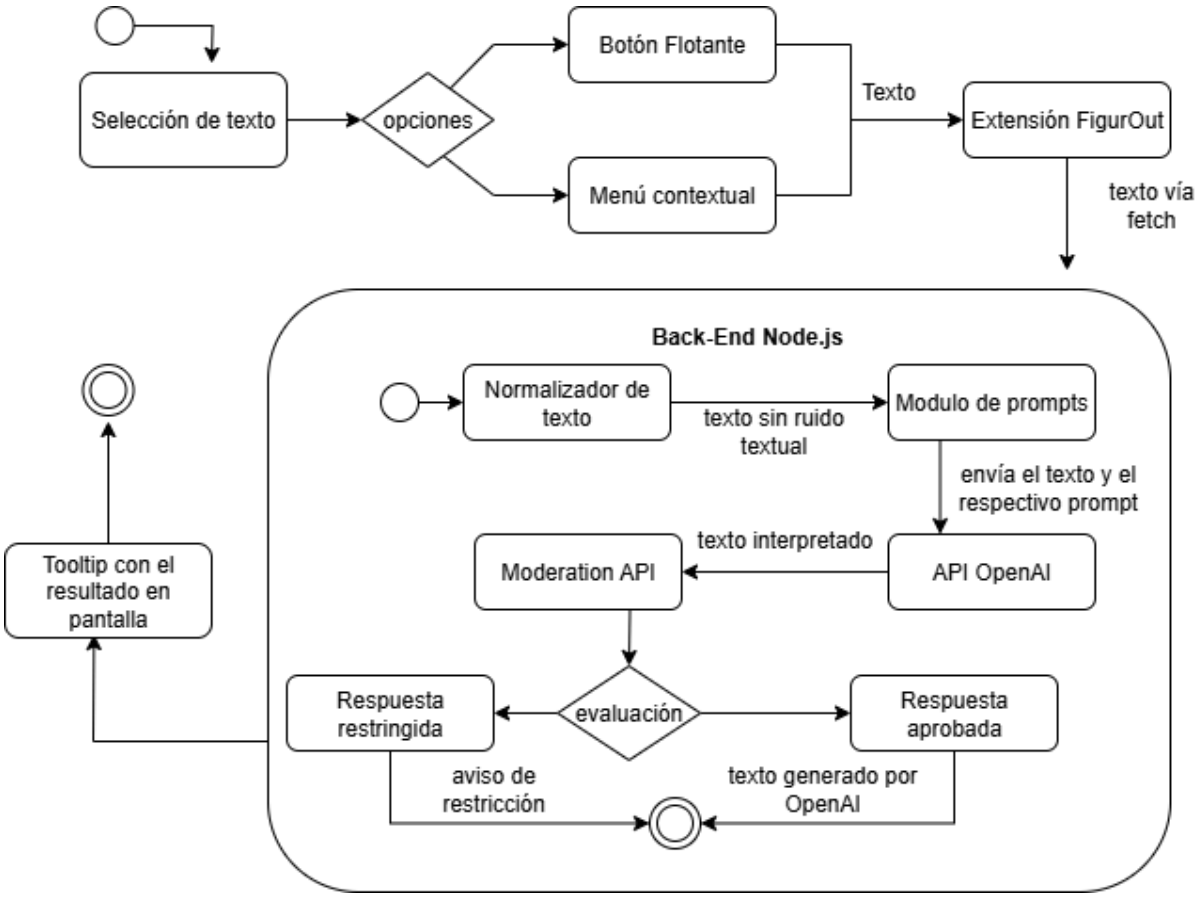


Figura 14 - Diagrama de flujo de la interpretación de un texto en FigurOut.

3.5 Capítulo 5. Resultados

El desarrollo del sistema FigurOut culminó con un prototipo funcional que cumple con los objetivos planteados en cuanto a accesibilidad cognitiva, procesamiento del lenguaje figurado y personalización de la respuesta según la necesidad del usuario. A lo largo del proyecto, se alcanzaron avances significativos en tres niveles: técnico, funcional y evaluativo.

Desde el punto de vista técnico, se logró implementar una extensión de navegador operativa para Google Chrome, que permite al usuario seleccionar fragmentos de texto y obtener, mediante inteligencia artificial, una reinterpretación adaptada. Esta funcionalidad puede activarse desde el menú contextual o a través de un botón flotante. También se desarrolló un panel de control complementario donde se puede seleccionar configuraciones como el tipo de respuesta (literal o explicativa) y activar o desactivar el botón flotante. Además, en él se accede a un intérprete desde donde se pueden ingresar textos manualmente y recibir la respuesta en el momento.

En cuanto a la arquitectura, el sistema adoptó un enfoque cliente-servidor: la extensión actúa como cliente, mientras que el backend, desarrollado en Node.js, se encarga de la construcción de prompts dinámicos, conexión con la API de OpenAI, moderación de contenido y envío de respuestas procesadas. El sistema también hace uso de chrome.storage para almacenar preferencias del usuario, y contempla posibles integraciones futuras con modelos open source como BETO.

Se comprobó el recorrido completo: captura de texto en el navegador, normalización y armado de prompt dinámico en el backend, consulta al modelo, moderación previa al despliegue y presentación en tooltip. El resultado es tiempo de respuesta adecuado para escenarios de prueba y una experiencia coherente entre panel, intérprete y contenido en sitio externo.

A nivel funcional, FigurOut logró el diseñado para adaptarse al perfil por defecto, que es el de personas a las que el lenguaje figurado le representa una barrera por una cuestión de comprensión.

Como parte del proceso de ajuste, se llevó a cabo una evaluación exploratoria sobre la calidad de las interpretaciones generadas, con participación de profesionales del ámbito educativo y clínico de la salud mental. Esta instancia permitió identificar las configuraciones de prompt más adecuadas para cada modo de respuesta, y verificar que el sistema ofrece explicaciones claras, pertinentes y respetuosas del sentido original de las frases. No obstante, si bien la evaluación profesional validó la eficacia para el perfil de usuarios con barreras de comprensión, se requiere una validación similar para los perfiles infantil y de estudiantes de idiomas, lo cual se plantea como parte de los trabajos futuros.

Desde el punto visual, las interfaces se implementaron con EJS, CSS y JavaScript, favoreciendo una composición modular de vistas y estilos reutilizables. En la Figura 5 se observa el panel de control (ruta /) con opciones de configuración y acceso al intérprete; allí puede verse el uso de componentes EJS y CSS para una interfaz intuitiva, limpia y consistente con el resto del sistema.

En resumen, el proyecto alcanzó la construcción de un MVP funcional, validado en escenarios controlados y abierto a futuras mejoras. FigurOut demostró ser una herramienta viable para abordar la accesibilidad semántica, brindando asistencia a usuarios reales mediante un enfoque escalable, adaptable y éticamente responsable.

Asimismo, los avances del proyecto trascendieron el ámbito académico local: el artículo derivado de este trabajo, titulado “Accesibilidad Digital al Lenguaje Figurado mediante Inteligencia Artificial”, fue aceptado para su exposición y publicación en el Congreso Argentino de Ciencias de la Computación (CACIC 2025). La evaluación de pares resaltó la claridad, pertinencia y aplicabilidad de la propuesta, destacando la calidad

experimental y su potencial de aporte al campo de la accesibilidad digital. Este reconocimiento refuerza la relevancia del sistema FigurOut como línea de investigación aplicada y valida su contribución dentro del panorama científico y tecnológico actual.

3.5.1 Trabajos futuros

Si bien la aplicación alcanzó un nivel funcional que permite validar su utilidad y viabilidad, el sistema presenta múltiples posibilidades de expansión y mejora. A continuación, se describen los trabajos futuros propuestos, que apuntan tanto a la optimización técnica del sistema como a su escalabilidad funcional y portabilidad multiplataforma. Estas proyecciones permitirán consolidar *FigurOut* como una herramienta más robusta, autónoma y accesible, adaptable a distintos entornos y perfiles de usuario.

Extensión de perfiles: Infantil y Estudiantes de Lengua Extranjera

A futuro se propone ampliar *FigurOut* hacía dos públicos adicionales (usuarios infantiles y estudiantes de español como lengua extranjera) sin alterar la topología actual del sistema. La arquitectura cliente–servidor ya implementada permite escalar de forma orgánica: el panel de control incorporará un selector de tipo de cuenta/perfil (barreras de comprensión, infantil, ELE) y, a partir de esa preferencia, el backend resolverá el conjunto de prompts y el tono de salida más adecuados, conservando el mismo flujo técnico.

La integración requiere cambios puntuales y compatibles con el diseño vigente. En la interfaz se añadirá el selector de perfil y su persistencia entre sesiones; en el servidor se reutilizará el mecanismo de construcción de prompts, ahora indexado por perfil y modo de respuesta (literal o explicativo). La capa de moderación ajustará *criterios por audiencia*: umbrales más estrictos y vocabulario protegido para el perfil infantil, español neutro y breves aclaraciones culturales para el perfil de estudiantes de lengua extranjera. De este modo, la

personalización se incorpora como una variación de configuración y no como un cambio estructural del sistema.

En paralelo, se plantea un estudio específico de ingeniería de prompts para cada nuevo público. El procedimiento replicará el protocolo ya utilizado: elaboración de bancos de prompts iniciales, pilotaje con corpus representativo y evaluación cualitativa junto a profesionales (docentes de primaria/psicopedagogía para infantil y profesores para estudiantes). Las métricas de decisión se mantendrán: claridad, adecuación léxica y fidelidad semántica, y se definirá como criterio de aceptación un desempeño promedio alto y consistente, además de la reducción de rechazos por moderación en el perfil infantil.

Escalabilidad del backend

Si bien la arquitectura actual basada en Node.js y Express resulta adecuada para un prototipo funcional y pruebas controladas, el crecimiento del sistema en escenarios de uso masivo requerirá estrategias adicionales de escalabilidad. Entre las alternativas posibles se proyecta el uso de contenedores Docker, que permitirían empaquetar e implementar la aplicación de manera portable y replicable en diferentes entornos de servidor. Asimismo, se contempla la adopción de arquitecturas serverless (como AWS Lambda, Google Cloud Functions o Azure Functions), que posibilitarían un escalado automático de recursos en función de la demanda real del sistema.

La incorporación de estas tecnologías garantizaría que FigurOut pueda sostener un aumento progresivo de usuarios y solicitudes, manteniendo tiempos de respuesta adecuados y asegurando la disponibilidad del servicio en contextos de mayor exigencia.

Almacenamiento con chrome.storage

En la versión actual se implementó una funcionalidad básica con chrome.storage.local para persistir preferencias clave de la extensión (por ejemplo, el modo de interpretación:

literal o explicativo; y, de forma experimental, banderas de interfaz como la activación del botón mouseup). Esta capa evita depender de una base de datos externa y permite que la experiencia del usuario sea consistente entre interacciones dentro del navegador.

Como trabajo futuro, se proyecta la expansión completa de este mecanismo para persistir todos los perfiles y preferencias del usuario, incluyendo: tipo de perfil (barreras, infantil, estudiante), tipo de explicación preferida (literal/ contextual), último acceso y métricas de uso no sensibles.

En la Figura 15 se ilustra el esquema JSON previsto para estas preferencias. La extensión inicializará estos valores mediante chrome.storage.local.set() al instalarse o reconfigurarse, y los recuperará con get() en cada sesión para mantener la personalización activa. Esta ampliación impacta positivamente en la arquitectura, al separar con mayor claridad la lógica de presentación del estado del usuario en una capa de persistencia en el cliente, incorporando validaciones del esquema, manejo de cambios con storage.onChanged y políticas de retención alineadas con privacidad.


```
{
"email": "usuario@ejemplo.com",
"profile": 2,
"option": 1,
"subscription": 1,
"lastLogin": "2025-06-24T12:00:00Z"
}
```


Figura 15 - Ejemplo de estructura JSON almacenada.

Este cambio generará un impacto en la arquitectura, ya que implica una mayor separación entre la lógica de presentación y la lógica de usuario, incorporando una capa de persistencia en el cliente.

Alternativa a OpenAI: funcionamiento local con modelos open source.

Si bien la versión actual de *FigurOut* se basa en la API de OpenAI para interpretar el lenguaje figurado, se proyecta el desarrollo de una arquitectura autónoma y autoalojada, utilizando modelos de código abierto entrenados para el español. Esto representa un avance hacia la soberanía tecnológica, la reducción de costos y la mejora en privacidad de los datos procesados.

Modelo Propuesto: BETO

BETO es un modelo de lenguaje basado en BERT, pre entrenado sobre corpus en español (colección estructurada de textos). Esto último le aporta valor al proyecto ya que el modelo aprende estructuras gramaticales reales, capturando expresiones figuradas, coloquiales o modismos [14][15][16].

Está disponible gratuitamente en la plataforma Hugging Face y puede integrarse localmente en el backend del sistema.

Es especialmente apto para tareas como:

- Detección de lenguaje figurado.
- Clasificación semántica de frases.
- Análisis de entidades y modismos.

Se puede extender mediante fine-tuning para ajustarse a tareas específicas del proyecto. Su integración con el backend se lograría a través de frameworks como FastAPI o Flask, recibiendo texto desde el frontend y evaluando su contenido en tiempo real.

Arquitectura proyectada con BETO:

- Paso 1: BETO clasifica si una frase contiene lenguaje figurado.
- Paso 2: Un segundo modelo, genera una explicación literal o contextual del texto detectado.
- Paso 3: Se aplica lógica de moderación y se entrega el resultado al usuario.

Entre las principales ventajas de esta alternativa se destacan la independencia total de servicios de terceros, lo cual reduce la dependencia tecnológica externa; una mejora sustancial en la privacidad y el control de los datos procesados, al mantener todo el análisis en entornos locales; la posibilidad de implementar el sistema en contextos offline o institucionales, como escuelas, y además, una optimización de costos a mediano plazo, al eliminar tarifas por uso de servicios comerciales como la API de OpenAI.

Filtro de Moderación Local (Alternativa Open Source)

Como parte de los desarrollos futuros de *FigurOut*, se proyecta la implementación de un sistema de moderación de contenido independiente, basado en tecnologías de código abierto, que reemplace la dependencia actual de la Moderation API.

La alternativa propuesta consiste en incorporar un microservicio de moderación local, utilizando modelos de clasificación entrenados para detectar lenguaje ofensivo, tóxico, violento o sexual en español. Entre los modelos investigados se destacan Detoxify y HateBERT que abarcan todo tipo de contextos, lo cuál es muy adecuado para la navegación de los usuarios de *FigurOut*, estos modelos pueden ejecutarse localmente sin depender de proveedores externos [16][20][21].

El funcionamiento espera que, tras la generación de la respuesta por el modelo de lenguaje principal, el texto sea evaluado por este microservicio de moderación local. Si el

modelo detecta una probabilidad significativa de toxicidad u otra categoría sensible (umbral configurable), la respuesta es bloqueada y registrada en el sistema de auditoría, así cómo funciona actualmente. Además, este sistema puede complementarse con listas negras y blancas personalizables, optimizando la detección de contextos locales, jerga y expresiones propias del idioma español.

Integración con Otros Navegadores: Facilidad y Escalabilidad

La extensión FigurOut fue desarrollada siguiendo el estándar WebExtensions, una arquitectura adoptada por la mayoría de los navegadores modernos, incluyendo Google Chrome, Microsoft Edge, Brave y Mozilla Firefox. Esta elección tecnológica tiene un impacto directo en la escalabilidad y portabilidad del sistema [18].

Dado que tanto Microsoft Edge como Brave están basados en Chromium y comparten la misma API de extensiones que Chrome, la integración de FigurOut en estos navegadores es prácticamente inmediata. El proceso consiste en empaquetar la extensión ya existente y cargarla como desarrollador en Edge o Brave. En la mayoría de los casos, no se requieren modificaciones en el código fuente ni en la estructura de archivos.

Para la inclusión de Firefox sabemos que también adopta el estándar WebExtensions, pero su implementación presenta algunas diferencias menores respecto a Chromium. La mayoría de las funciones desarrolladas para Chrome, Edge o Brave funcionarán directamente en Firefox; sin embargo, puede requerirse el ajuste de ciertos permisos, el uso de propiedades específicas en el archivo manifest.json o la adaptación de algunas APIs [19].

4. Conclusiones

El desarrollo del proyecto representó un verdadero desafío, no solo desde el punto de vista técnico de la carrera, sino por la complejidad de problemática abordada. La accesibilidad al lenguaje figurado requiere una mirada empática y responsable. Entender aquello me llevó a profundizar en bibliografía específica sobre trastornos del espectro autista, dificultades de comprensión semántica y barreras cognitivas en entornos digitales. Aprendiendo así a cómo utilizar el lenguaje y adaptando al sistema en base a lo investigado.

En el transcurso del proyecto consolidé mis habilidades de análisis, diseño y sobre todo de desarrollo y aprendiendo nuevas tecnologías en el proceso. La metodología iterativa y el feedback con mi tutor fueron fundamentales para ir superando los obstáculos y mejorar el producto.

Desde lo formativo, el proyecto evidenció los conocimientos adquiridos a lo largo de la carrera, conceptos como planificación por etapas, especificación de requisitos (ERS), trazabilidad entre requisitos–casos de uso–interfaces, y diseño modular de componentes (extensión, backend, moderación, panel) incorporados en materias de Ingeniería de Software. También criterios de eficiencia y claridad en preprocesamiento de textos, manejo de estructuras simples para logs y configuración, y decisiones de complejidad para mantener tiempos de respuesta fueron aprendidos en materias de lógica de programación como Algoritmos y estructuras de datos 1 y 2, Programación Avanzada y Lenguajes de Programación.

Considero que el sistema, en su estado actual, es robusto y versátil, y podría ser lanzado al mercado con impacto social inmediato, ya que resuelve una necesidad poco visibilizada en la accesibilidad web. El MVP desarrollado es funcional, adaptable a múltiples perfiles y deja sentadas las bases para evoluciones futuras, tanto desde el punto de vista técnico como en el abordaje de nuevas dimensiones de la accesibilidad digital.

5. Referencias

1. Node.js. (n.d.). *Node.js documentation*. Node.js Foundation. Recuperado el 15 de julio de 2025, de https://nodejs.org/docs/latest/api/
2. Bootstrap. (n.d.). *Introduction · Bootstrap v5.3*. The Bootstrap Authors. Recuperado el 15 de julio de 2025, de https://getbootstrap.com
3. W3Schools. (n.d.). *HTML Tutorial*. Recuperado el 15 de julio de 2025, de https://www.w3schools.com/html/
4. W3Schools. (n.d.). *CSS Reference*. Recuperado el 15 de julio de 2025, de https://www.w3schools.com/CSSref/index.php
5. OpenAI. (2023). *GPT-4 Technical Report*. Recuperado el 15 de julio de 2025, de https://cdn.openai.com/papers/gpt-4.pdf
6. OpenAI. (n.d.). *Moderation*. OpenAI Documentation. Recuperado el 15 de julio de 2025, de https://platform.openai.com/docs/guides/moderation
7. Universidad CAECE - Reglamento de Trabajos Finales: Documento oficial consultado para guiar el desarrollo del presente proyecto.
8. World Wide Web Consortium (W3C). (n.d.). *Web Standards Recuperado el 15 de julio de 2025, de* https://www.w3.org/
9. Norbury, C. F. (2005). The relationship between Theory of Mind and metaphor: Evidence from children with language impairment and autistic spectrum disorder. British Journal of Developmental Psychology, 23(3), 383–399. https://doi.org/10.1348/026151005X26732
10. Booth, R., & Happé, F. (2010). “Hunting with a knife and … fork”: Examining central coherence in autism, attention deficit/hyperactivity disorder, and typical development

with a linguistic task. Journal of Experimental Child Psychology. https://www.sciencedirect.com/science/article/pii/S0022096510001244

11. Chemnad, M., & Othman, M. (2024). Digital accessibility in the era of artificial intelligence—Bibliometric analysis and systematic review. Journal of Medical Internet Research, 26, e51355. Recuperado el 15 de julio de 2025, de https://www.jmir.org/2024/1/e51355/
12. Orynbay, Y., Yermukanov, A., Aidos, D., & Ali, F. (2025). The role of cognitive computing in NLP: A new path for accessible digital interaction. Frontiers in Computer Science, 6, 1486581. Recuperado el 15 de julio de 2025, de https://www.frontiersin.org/articles/10.3389/fcomp.2024.1486581/full
13. Google Developers. (n.d.). *chrome.storage - Chrome Extensions API*. Recuperado el 15 de julio de 2025, de https://developer.chrome.com/docs/extensions/reference/api/storage
14. Hugging Face. (n.d.). *dccuchile/bert-base-spanish-wwm-cased [Modelo BETO]*. Hugging Face. Recuperado el 15 de julio de 2025, de https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased
15. González, L., & Chaperon, G. (2023). *Spanish Pre-trained BERT Model and Evaluation Data*. *ResearchGate*. https://www.researchgate.net/publication/372962444_Spanish_Pre-trained_BERT_Model_and_Evaluation_Data
16. Cañete, J., Chaperon, G., Fuentes, R., & Pérez, J. (2020). *Spanish Pre-trained BERT Model and Evaluation Data*. arXiv. https://arxiv.org/abs/1910.03771
17. StatCounter. (2024). *Browser Market Share Worldwide*. StatCounter Global Stats. Recuperado el 15 de julio de 2025, de https://gs.statcounter.com/browser-market-share

18. Google. (n.d.). *Chrome Developers - Extensions Documentation*. Google Developers. Recuperado el 15 de julio de 2025, de https://developer.chrome.com/docs/extensions?hl=es-419
19. Mozilla. (n.d.). *WebExtensions - Browser Extension APIs*. Mozilla Developer Network. Recuperado el 15 de julio de 2025, de https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions
20. Unitary. (n.d.). *Detoxify: Transformer-based offensive content classifier*. GitHub. Recuperado el 15 de julio de 2025, de https://github.com/unitaryai/detoxify
21. Caselli, T., Basile, V., Mitrović, J., & Vossen, P. (2020). *HateBERT: Retraining BERT for abusive language detection in English.* arXiv preprint arXiv:2010.12472. Recuperado el 15 de julio de 2025, de https://arxiv.org/abs/2010.12472
22. Express.js. (n.d.). *Express – Node.js web application framework*. Recuperado el 15 de julio de 2025, de https://expressjs.com/
23. Mozilla Developer Network (MDN). (n.d.). *JavaScript – MDN Web Docs*. Recuperado el 15 de julio de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript
24. GeeksforGeeks. (n.d.). *How to use EJS as template engine in Node.js?* Recuperado el 15 de julio de 2025, de https://www.geeksforgeeks.org/node-js/use-ejs-as-template-engine-in-node-js/
25. Martín-Gascón, B. (2024). *The effects of cognitive-based instruction on figurative constructions comprehension and production: a case study*. *Porta Linguarum: Revista Internacional de Didáctica de las Lenguas Extranjeras*, 41, 97-117. https://produccioncientifica.ucm.es/documentos/65c3da167eb17e0a47b7d750?.com
26. W3C WAI. (s. f.). *Cognitive Accessibility at W3C*. https://www.w3.org/WAI/cognitive/?.com

27. Aguirre, A., & Iglesias, A. (2023). *Accessibility of COVID-19 information portals: An evaluation of barriers and challenges*. Applied Sciences, 13(15), 8715. MDPI. https://www.mdpi.com/2076-3417/13/15/8715

Anexos

Construcción del formulario "Evaluación de interpretaciones de Inteligencia Artificial sobre textos con recursos figurados"

a. Instrucción:

Leer con atención:

Esta investigación tiene el objetivo de evaluar la efectividad en la interpretación de frases con recursos del lenguaje figurado --metáforas, ironías, hipérboles, etc--. En el contexto de desarrollar una aplicación que elimine esta barrera para los usuarios a los que se les presente como tal.

La consigna es evaluar si la respuesta generada por la IA logra reinterpretar el contenido figurado a una versión literal y concisa.
En cada caso se presentarán:

- La frase original (con lenguaje figurado)

- La respuesta generada por la IA (versión corregida).

A partir de eso, deberá calificar los siguientes tres criterios, usando una escala de 0 a 3 (donde 0 es el puntaje más bajo y 3 el más alto):

1. Claridad: ¿La explicación es fácil de entender?

2. Adecuación al usuario: ¿Está redactada de forma apropiada para el perfil de usuario indicado?

3. Fidelidad semántica: ¿Mantiene el significado original de la frase figurada?

Tenga en cuenta que los textos pueden repetirse, ya que las correcciones son evaluadas con diferentes métodos de IA, entonces la respuesta variará. El contenido fue sacado de páginas web abarcando distintas temáticas.

b. Contenido del formulario:

Información personal:

- Nombre y Apellido
- Profesión/carrera
- Año de estudio (si corresponde).

Pregunta 1: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]
Texto original: el presidente pateó la pelota para adelante
Corrección de la IA: El presidente hizo que la situación avanzara.

Pregunta 2: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[1]

Texto original: el presidente pateó la pelota para adelante

Corrección de la IA: El presidente tomó una decisión que pospone el problema para más tarde.

Pregunta 3: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[2]

Texto original: el presidente pateó la pelota para adelante

Corrección de la IA: El presidente dio una patada a la pelota hacia adelante.

Pregunta 4: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: El nuevo guiño entre Tini Stoessel y Rodrigo de Paul

Corrección de la IA: El nuevo saludo amistoso entre Tini Stoessel y Rodrigo De Paul.

Pregunta 5: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]

Texto original: El nuevo guiño entre Tini Stoessel y Rodrigo de Paul

Corrección de la IA: La frase se refiere a una señal amistosa o un gesto de conexión entre Tini Stoessel y Rodrigo De Paul, que puede interpretarse como un indicio de una relación entre ellos.

Pregunta 6: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [2]

Texto original: El nuevo guiño entre Tini Stoessel y Rodrigo de Paul

Corrección de la IA: El nuevo gesto de complicidad entre Tini Stoessel y Rodrigo De Paul.

Pregunta 7: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: los romanos construyeron un imperio tan grande que podían ver el amanecer y el atardecer al mismo tiempo.

Corrección de la IA: Los romanos construyeron un imperio tan extenso que en algunas partes podían ver el sol salir y ponerse al mismo tiempo.

Pregunta 8: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]

Texto original: los romanos construyeron un imperio tan grande que podían ver el amanecer y el atardecer al mismo tiempo.

Corrección de la IA: Los romanos tenían un imperio tan extenso que en algunas partes podían ver el sol salir en una dirección y ponerse en la otra al mismo tiempo.

Pregunta 9: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[2]

Texto original: los romanos construyeron un imperio tan grande que podían ver el amanecer y el atardecer al mismo tiempo.

Corrección de la IA: Los romanos construyeron un imperio muy extenso que se extendía por muchas regiones, tanto que en un lugar del imperio podían ver salir el sol y en otro lugar, al mismo tiempo, podían ver el sol ponerse.

Pregunta 10: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[0]

Texto original: se ahogó en un vaso de agua.

Corrección de la IA: Se preocupó demasiado por un problema pequeño.

Pregunta 11: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[1]

Texto original: se ahogó en un vaso de agua.

Corrección de la IA: La frase significa que alguien se descontroló o se preocupó mucho por un problema pequeño o insignificante.

Pregunta 12: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[2]

Texto original: se ahogó en un vaso de agua.

Corrección de la IA: Se murió porque se metió en un vaso con agua y no pudo respirar.

Pregunta 13: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[0]

Texto original: Bochorno en Diputados: se cayó la sesión en medio de gritos e insultos entre el kirchnerismo y La Libertad Avanza

Corrección de la IA: Problemas en la reunión de diputados: la sesión se interrumpió debido a gritos e insultos entre los miembros del kirchnerismo y los de La Libertad Avanza.

Pregunta 14: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[1]

Texto original: Bochorno en Diputados: se cayó la sesión en medio de gritos e insultos entre el kirchnerismo y La Libertad Avanza

Corrección de la IA: La frase significa que hubo un problema en una reunión de los diputados, donde las cosas se pusieron muy tensas y se pelearon verbalmente entre dos grupos políticos: el kirchnerismo y la libertad avanza. Esto hizo que la sesión se detuviera.

Pregunta 15: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[2]

Texto original: Bochorno en Diputados: se cayó la sesión en medio de gritos e insultos entre el kirchnerismo y La Libertad Avanza

Corrección de la IA: El clima en el Congreso fue muy incómodo: la reunión se canceló porque hubo muchas peleas y ofensas entre los miembros del kirchnerismo y el partido La Libertad Avanza.

Pregunta 16: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[0]

Texto original: despierta tus sentidos con cada sorbo

Corrección de la IA: Disfruta de cada trago y presta atención a lo que sientes.

Pregunta 17: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[1]

Texto original: despierta tus sentidos con cada sorbo

Corrección de la IA: Significa que disfrutes y sientas cada trago que tomas.

Pregunta 18: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[2]

Texto original: despierta tus sentidos con cada sorbo

Corrección de la IA: Haz que tus sentidos se pongan alerta con cada vez que tomas un trago.

Construcción del formulario "Evaluación de interpretaciones de Inteligencia Artificial sobre textos con recursos figurados 2 (respuesta explicativa)"

c. Instrucción:

Leer con atención:

Esta investigación tiene el objetivo de evaluar la efectividad en la interpretación de frases con recursos del lenguaje figurado --metáforas, ironías, hipérboles, etc--. En el contexto de desarrollar una aplicación que elimine esta barrera para los usuarios a los que se les presente como tal.

La consigna es evaluar si la respuesta generada por la IA logra explicar y contextualizar el contenido figurado, se esperan respuestas más largas.

En cada caso se presentarán:

- La frase original (con lenguaje figurado)

- La respuesta generada por la IA (versión corregida).

A partir de eso, deberá calificar los siguientes tres criterios, usando una escala de 0 a 3 (donde 0 es el puntaje más bajo y 3 el más alto):

1. Claridad: ¿La explicación es fácil de entender?

2. Adecuación al usuario: ¿Está redactada de forma apropiada para el perfil de usuario indicado?

3. Fidelidad semántica: ¿Mantiene el significado original de la frase figurada?

Tenga en cuenta que los textos pueden repetirse, ya que las correcciones son evaluadas con diferentes métodos de IA, entonces la respuesta variará. El contenido fue sacado de páginas web abarcando distintas temáticas.

d. Contenido del formulario:

Información personal:

- Nombre y Apellido
- Profesión/carrera
- Año de estudio (si corresponde).

Pregunta 1: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: el pueblo habló en las urnas.

Corrección de la IA: La frase "el pueblo habló en las urnas" significa que la población expresó su opinión o tomó una decisión a través de su voto en una elección. En este contexto, "caminos" se refiere a las votaciones realizadas en las urnas, que son los lugares donde se depositan los votos.

Pregunta 2: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[1]

Texto original: el pueblo habló en las urnas.

Corrección de la IA: La frase "el pueblo habló en las urnas" significa que los ciudadanos expresaron su opinión o tomaron una decisión a través de su voto en una elección. Es una forma de decir que la voluntad del pueblo se dejó ver mediante el resultado electoral.

Pregunta 3: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original?

[2]

Texto original: el pueblo habló en las urnas.

Corrección de la IA: La frase "el pueblo habló en las urnas" significa que las personas de una comunidad (el pueblo) expresaron su opinión o tomaron una decisión a través de su voto en una elección (las urnas son donde se depositan los votos). Así que, en un sentido más literal, implica que la gente se pronunció sobre un tema importante usando el sufragio.

Pregunta 4: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: el plan económico quedó en pañales

Corrección de la IA: La frase "el plan económico quedó en pañales" significa que el plan no logró desarrollarse o avanzar, quedando en una etapa muy inicial o inmadura, como un bebé que aún usa pañales y no ha crecido lo suficiente.

Ejemplo simple: Imagina que alguien intenta empezar un negocio pero solo ha hecho algunos bocetos y no ha tomado medidas concretas, diríamos que su plan está "en pañales", porque aún no ha comenzado de verdad.

Pregunta 5: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]

Texto original: El plan económico quedó en pañales.

Corrección de la IA: La frase "el plan económico quedó en pañales" significa que el plan está muy poco desarrollado o en sus etapas iniciales. Es como cuando un bebé recién nace y todavía necesita mucho cuidado, lo que implica que el plan no está listo para ser implementado o tiene muchas cosas por mejorar.

Pregunta 6: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [2]

Texto original: El plan económico quedó en pañales.

Corrección de la IA: La frase "el plan económico quedó en pañales" sugiere que el plan es muy temprano en su desarrollo y todavía no está maduro o completamente formado. Utiliza la metáfora de un bebé en pañales, que representa algo que necesita tiempo y atención para crecer y evolucionar.

Pregunta 7: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: "categoría a". brad pitt reveló cuáles fueron las únicas dos actrices que lo deslumbraron a lo largo de toda su carrera

Corrección de la IA: La frase "categoría A" se refiere a personas o cosas que son de la más alta calidad o destacadas en su campo. En este contexto, al mencionar que Brad Pitt reveló las "únicas dos actrices que lo deslumbraron", él está diciendo que esas actrices son excepcionales, sobresalientes o muy talentosas en comparación con otras.

Pregunta 8: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]

Texto original: "categoría a". brad pitt reveló cuáles fueron las únicas dos actrices que lo deslumbraron a lo largo de toda su carrera

Corrección de la IA: La frase "categoría A" se refiere a las actrices que Brad Pitt considera de un nivel de talento y fama muy alto. Al decir que solo dos actrices lo deslumbraron, implica que esas mujeres son excepcionales en su oficio y han dejado una impresión duradera en él a lo largo de su carrera.

Pregunta 9: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [2]

Texto original: "categoría a". brad pitt reveló cuáles fueron las únicas dos actrices que lo deslumbraron a lo largo de toda su carrera

Corrección de la IA: La frase "categoría A" se refiere a las actrices que son consideradas de la más alta calidad o prestigio en la industria del cine. Cuando Brad Pitt dice que solo dos actrices lo deslumbraron, implica que esas actrices son excepcionales en su talento o carisma,

destacándose por encima de otras en su carrera. En un sentido más literal, sugiere que, en su opinión, hay un nivel de calidad muy alto que solo unas pocas pueden alcanzar.

Pregunta 10: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]
Texto original: le echaron más leña al fuego en el debate.
Corrección de la IA: La frase "le echaron más leña al fuego en el debate" significa que alguien intensificó una discusión o conflicto en lugar de calmarla. La "leña" representa argumentos o provocaciones que aumentan la tensión, mientras que el "fuego" simboliza la situación ya conflictiva.

Pregunta 11: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]
Texto original: le echaron más leña al fuego en el debate.
Corrección de la IA: La frase "le echaron más leña al fuego en el debate" significa que alguien intensificó una discusión o conflicto, haciendo que la situación se volviera más tensa o apasionada. Es como si se estuviera alimentando una pelea, haciéndola aún más intensa.

Pregunta 12: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [2]
Texto original: le echaron más leña al fuego en el debate.
Corrección de la IA: La frase "le echaron más leña al fuego en el debate" significa que las personas involucradas en el debate hicieron comentarios o acciones que intensificaron el conflicto o la tensión en la discusión, en lugar de calmarla. Literalmente, "echar leña al

fuego" indica agregar combustible a un fuego, lo que lo hace arder más fuerte; en este caso, se refiere a avivar una situación ya conflictiva.

Pregunta 13: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: la oposición salió con los tapones de punta

Corrección de la IA: La frase "la oposición salió con los tapones de punta" significa que la oposición (un grupo político o personas que critican o se oponen a algo) está muy enojada o agresiva, lista para enfrentar o atacar a sus oponentes.

Pregunta 14: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]

Texto original: la oposición salió con los tapones de punta

Corrección de la IA: La frase "la oposición salió con los tapones de punta" significa que la oposición está lista para enfrentar un desafío de manera muy decidida y agresiva. La expresión "con los tapones de punta" sugiere que están preparados para atacar o defenderse con fuerza, sin miedo a confrontar a sus oponentes.

Pregunta 15: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [2]

Texto original: la oposición salió con los tapones de punta

Corrección de la IA: La frase "la oposición salió con los tapones de punta" sugiere que la oposición está muy agresiva o lista para atacar. Literalmente, "tapones de punta" hace referencia a algo afilado que puede causar daño, lo que implica que la oposición está preparada para confrontar o criticar con mucha fuerza.

Pregunta 16: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [0]

Texto original: milei, sobre las auditorías en las universidades: “los chorros no van a tener paz en la argentina."

Corrección de la IA: La frase “los chorros no van a tener paz en la Argentina” significa que las personas que han robado o han cometido actos de corrupción (los "chorros") no podrán vivir tranquilas, porque habrá un control más estricto y consecuencias para sus acciones, como las auditorías en universidades.

Pregunta 17: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [1]

Texto original: milei, sobre las auditorías en las universidades: “los chorros no van a tener paz en la argentina."

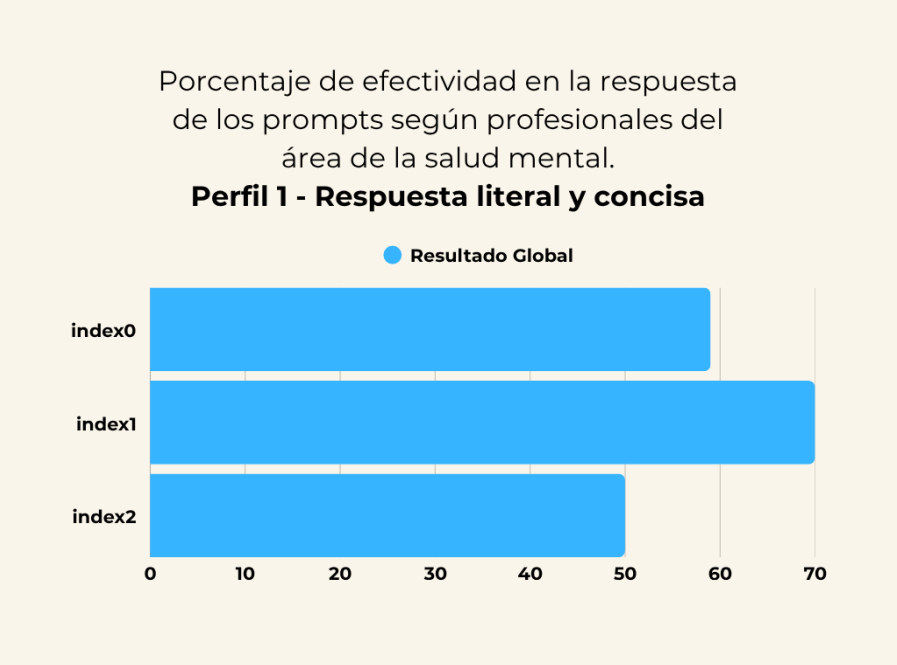
Corrección de la IA: La frase de Milei sugiere que las personas corruptas (“chorros”) no podrán estar tranquilas o seguir actuando como lo han hecho en el pasado debido a las auditorías que se están implementando en las universidades. Es un mensaje sobre la intención de erradicar la corrupción y asegurar que aquellos que han cometido actos ilícitos enfrenten consecuencias.

Pregunta 18: (1) Claridad de la explicación, (2) Adecuación al usuario, (3) Fidelidad semántica --¿mantiene la idea original? [2]

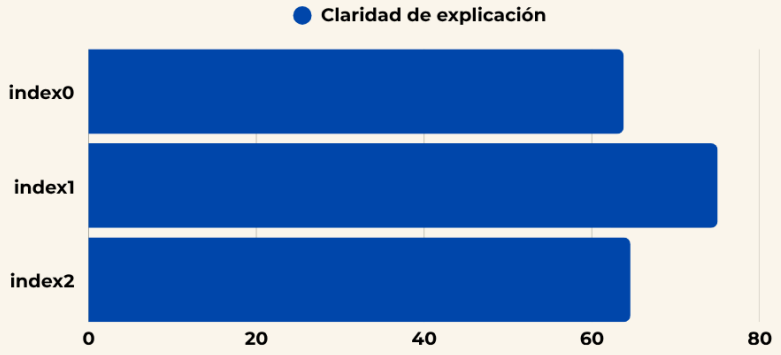
Texto original: milei, sobre las auditorías en las universidades: “los chorros no van a tener paz en la argentina."

Corrección de la IA: La frase de Milei sugiere que las personas corruptas ("chorros") no podrán actuar con impunidad en Argentina. Al mencionar las auditorías en las universidades, implica que estas revisiones buscarán detectar y sancionar la corrupción, lo que les generará problemas y ansiedad a quienes están involucrados en actos deshonestos. En resumen, es una advertencia de que se están tomando medidas para combatir la corrupción.

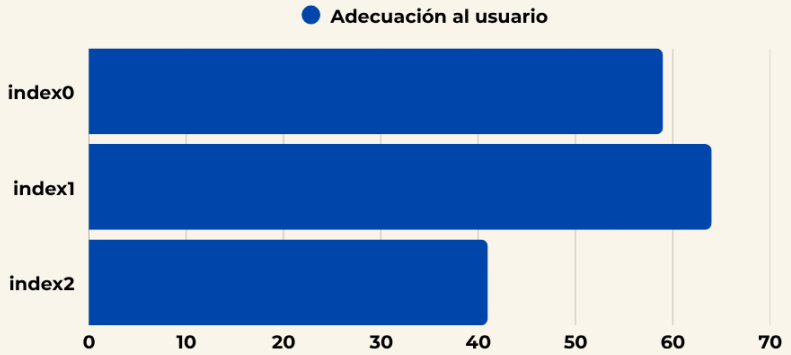
e. Resultados de la encuesta: Evaluación de interpretaciones de Inteligencia Artificial sobre textos con recursos figurados.

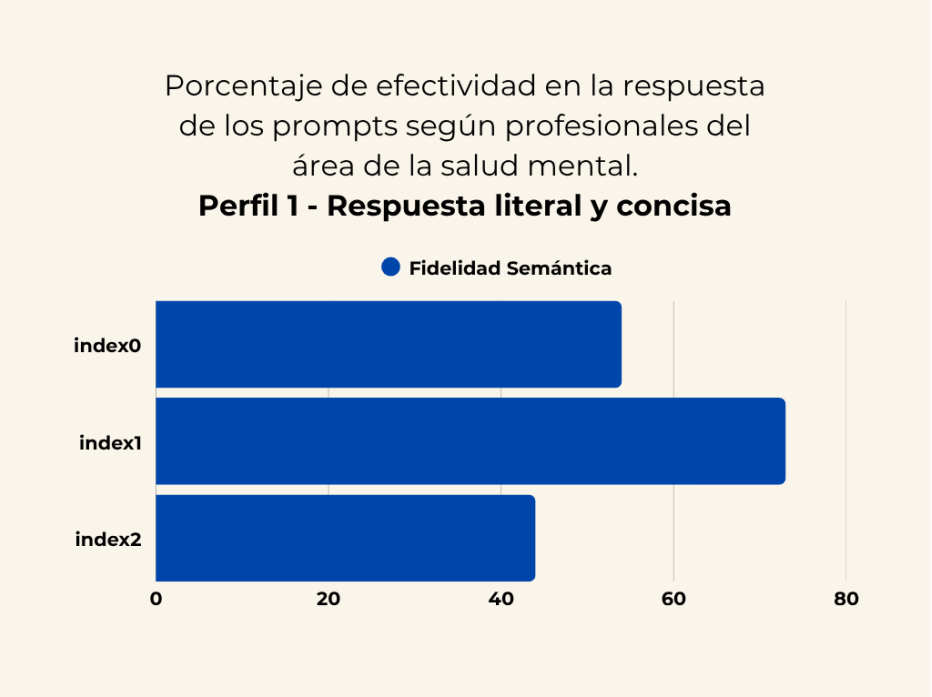


Porcentaje de efectividad en la respuesta de los prompts según profesionales del área de la salud mental.
Perfil 1 - Respuesta literal y concisa

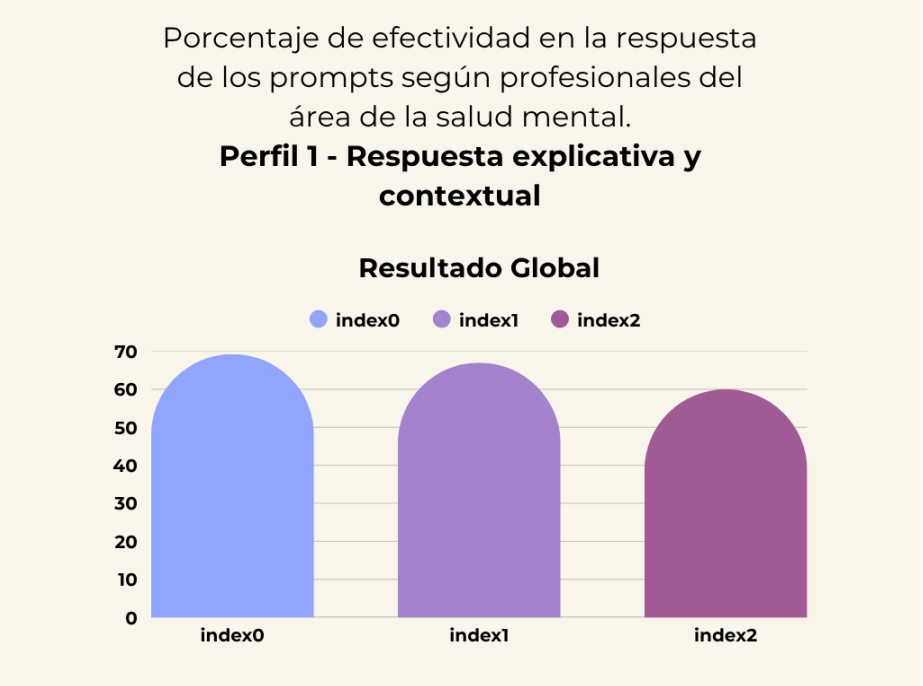


Porcentaje de efectividad en la respuesta de los prompts según profesionales del área de la salud mental.
Perfil 1 - Respuesta literal y concisa

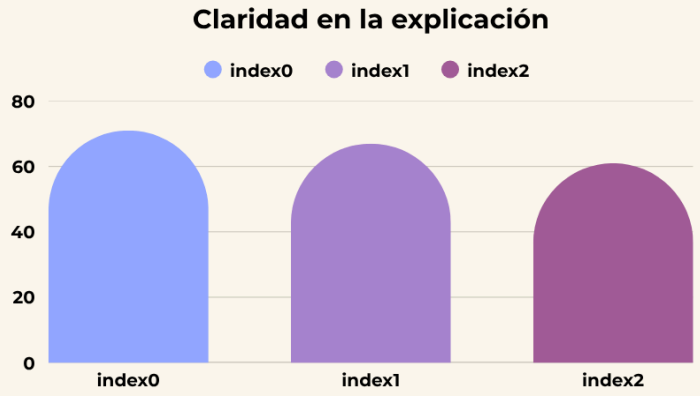




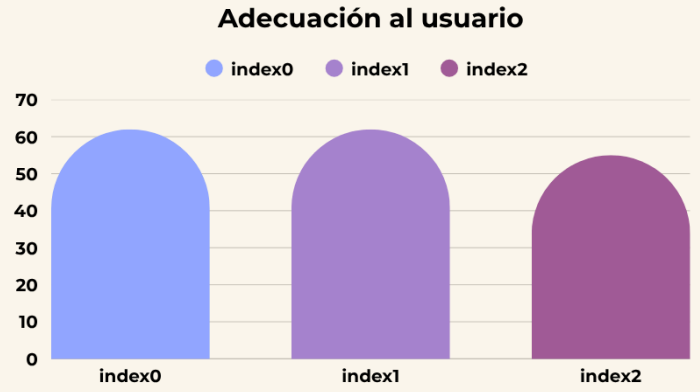
f. Resultados de la encuesta: Evaluación de interpretaciones de Inteligencia Artificial sobre textos con recursos figurados.

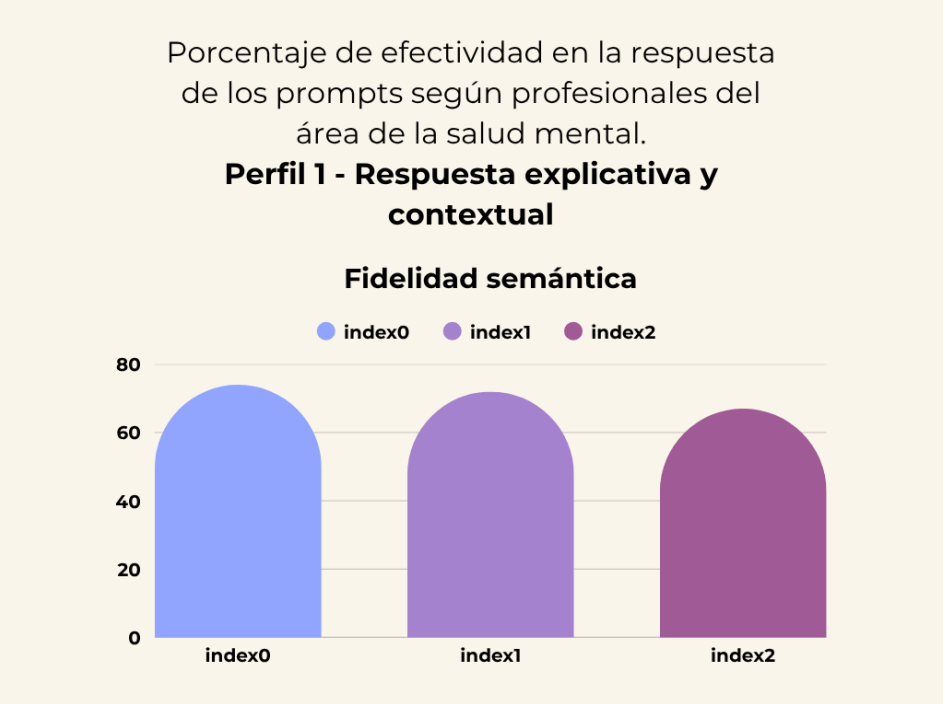


Porcentaje de efectividad en la respuesta de los prompts según profesionales del área de la salud mental.
Perfil 1 - Respuesta explicativa y contextual



Porcentaje de efectividad en la respuesta de los prompts según profesionales del área de la salud mental.
Perfil 1 - Respuesta explicativa y contextual





g. Participación de profesionales:

A continuación se presenta la respuesta del campo “Profesión” de la encuesta tomada:

Lic.en Psicología
Lic en psicologia
Estudiante de licenciatura en psicología
Lic. en Psicología
Acompañante terapeutico
Psicología
Lic. en Psicopedagogía
Estudiante avanzado de Lic. En Psicología

h. Extracción de [ModulePrompts.js]

Prompts analizados en la encuesta número 1.

```
literal: [
  {
    systemPrompt: "Tu función es hacer accesible cualquier texto que presente ambigüedad, lenguaje complejo o lenguaje
    figurado, traduciendo la frase al lenguaje más simple y directo posible.",
    buildUserPrompt: (text) => `Convierte esta frase figurada en una versión literal y clara: ${text}`
  },
  {
    systemPrompt: "Tu tarea es traducir frases complicadas o con lenguaje figurado a una versión simple, clara y literal para
    facilitar su comprensión.",
    buildUserPrompt: (text) => `Decime qué significa esta frase, sin adornos ni metáforas: ${text}`
  },
  {
    systemPrompt: "Explicá el sentido literal de expresiones difíciles usando el lenguaje más directo posible, para que
    cualquier persona pueda entender.",
    buildUserPrompt: (text) => `Reescribí esta frase para que sea completamente literal y fácil de entender: ${text}`
  }
],
```

Prompts analizados en la encuesta número 2:

```
explicacion: [
  {
    systemPrompt: "Tu función es facilitar la comprensión de frases difíciles, explicando el sentido figurado y dando un
    ejemplo simple para que cualquier adulto lo entienda.",
    buildUserPrompt: (text) => `Explicá qué significa esta frase figurada de forma breve y clara: ${text}`
  },
  {
    systemPrompt: "Explicá frases complejas o figuradas de forma breve, brindando una interpretación clara que facilite su
    comprensión y el acceso a esa información.",
    buildUserPrompt: (text) => `¿Qué quiere decir esta frase? Explicalo de forma simple: ${text}`
  },
  {
    systemPrompt: "Sos un asistente que ayuda a entender contextos con lenguaje figurado. Dás una explicación corta que
    facilite la interpretación para quienes no entienden ese contexto.",
    buildUserPrompt: (text) => `Ayudame a entender brevemente esta frase figurada de manera más literal: ${text}`
  }
]
```

Acceso al tester de Moderation API.

https://github.com/UlisesFritzSerain/testModerationAPI

Contenido:

Moderation API Tester – FigurOut

Este repositorio contiene un entorno de prueba simple para verificar el uso de la **Moderation API de OpenAI** integrada al proyecto **FigurOut**. El objetivo es comprobar si un texto determinado sería marcado como inapropiado según los filtros de contenido de la plataforma.

Contenido

- `main.js` : versión básica por consola. Pide un texto, evalúa y responde si contiene lenguaje sensible.
- `test-openai-output.js` : evalúa si las respuestas generadas por la IA cumplen con criterios seguros.
- `utils/moderationCheck.js` : validación de contenido con `fetch` a la API de moderación.

Requisitos

- Node.js instalado
- Clave de API válida de OpenAI en un .env

Uso

1. Instalá las dependencias:

```
npm install dotenv node-fetch
```

Especificación de requerimientos del sistema:

Se presentará la ERS construida en los inicios del proyecto, a modo de entender la evolución que tuvo el mismo:

Especificación de Requisitos de Software (ERS)

1. **Introducción**

1.1. Propósito

El propósito de este documento es definir los requisitos del sistema “FigurOut“ en el marco del desarrollo de un proyecto para la carrera Tecnicatura Universitaria en programación. Este documento servirá como una guía para el desarrollo y la validación

del sistema, asegurando que cumpla con las necesidades de los usuarios y otros interesados.

1.2. Alcance

Esta extensión tiene el objetivo de mejorar la accesibilidad de las personas con problemas para interpretar el lenguaje figurado. El sistema permitirá detectar recursos de lenguaje figurado en el texto de los sitios web y lo reescribirá a versiones más literales o explicativas.
Esta extensión está pensada para ponerse a prueba sin planes de pago y en un marco teórico a modo de muestra.

1.3 Definiciones, siglas, y abreviaciones

- API (Application Programming Interface): Una API es una pieza de código que permite a dos aplicaciones comunicarse entre sí para compartir información y funcionalidades. En este proyecto, se utilizará la API de OpenAI.
- IA (Inteligencia Artificial): La inteligencia artificial (IA) es una tecnología que permite a ordenadores y máquinas simular la inteligencia humana y su capacidad para resolver problemas.
- Accesibilidad Web: La accesibilidad web combina la programación, el diseño y la tecnología para construir un Internet sin barreras que permita a todos los usuarios el entendimiento, el aprendizaje, la navegación y la plena interacción con la web.
- Lenguaje figurado: Que usa figuras retóricas. Dicho de un sentido: Que no corresponde al literal de una palabra o expresión, pero está relacionado con él por una asociación de ideas.

1.4. Referencias

- IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

1.5. Overview

Este documento consta de 3 secciones, en la sección de introducción se proporciona una visión general de la ERS.
En la sección 2 se da una descripción general del sistema con el fin de conocer las principales funciones que debe realizar, los datos asociados, factores y restricciones que afectan el desarrollo de la aplicación.
En la sección 3 se definen detalladamente los requisitos que debe satisfacer el sistema.

2. Descripción general

2.1. Perspectiva del producto

El sistema "FigurOut" es un producto enfocado en la accesibilidad web. El mismo se podrá acceder como una extensión de navegador y los usuarios podrán personalizar su experiencia para acceder a la información de manera literal. La cuál podrá realizar ese proceso de "interpretación" mediante el consumo de la API de OpenAI que es una inteligencia artificial con distintos modelos creada para generación e interpretación de texto.

2.2 Funciones del producto

El sistema proporcionará diferentes funcionalidades, diseñadas para facilitar la comprensión del lenguaje figurado y mejorar la accesibilidad en la web. Estas funcionalidades estarán categorizadas según el nivel de detalle en las respuestas y el consumo de recursos de la API de OpenAI, permitiendo una escalabilidad del servicio según el nivel de uso requerido por los usuarios. Las principales funciones son las siguientes:

***A**. Funciones con explicaciones detalladas*

- **A1**. Explicación mediante carteles emergentes.
  Las explicaciones detalladas de los elementos figurativos se presentarán como carteles emergentes (pop-ups) sobre el texto. Esta opción permite que las aclaraciones se muestren por fuera y el contenido original permanezca intacto.

- **A2**. Explicación embebida en el texto:
  La extensión proporcionará una explicación detallada de los recursos de lenguaje figurado, como metáforas, ironías o exageraciones, integrando directamente las aclaraciones dentro del texto original. Esta opción ofrece mayor detalle en las aclaraciones del lenguaje, pero con un mayor consumo de tokens y, por ende, un costo mayor.

B. *Función de respuestas concretas con menor consumo de recursos.*

- **B1**. Carteles explicativos simples.
  En esta opción los carteles presentan una descripción concisa del significado del lenguaje figurado, reduciendo así el consumo de recursos.

- **B2**. Cambio de contenido en el texto.
  El sistema hará una "traducción" simple del lenguaje figurado a un lenguaje literal y accesible, reemplazando el contenido original sin proporcionar explicaciones adicionales. Esta función minimiza el consumo de tokens y es adecuada para usuarios que prefieren una interpretación rápida y clara del texto.

2.3 Características del usuario

El usuario al que está dirigido este sistema es aquel al que el lenguaje figurado se le presenta como un barrera para navegar en diferentes sitios web. Estos usuarios necesitan una traducción literal del lenguaje figurado o explicaciones más claras y detalladas para mejorar su comprensión del contenido en línea.
A futuro, se proyecta extender su utilidad a otros públicos, como niños en etapa escolar que están aprendiendo a interpretar expresiones complejas y para también proteger su consumo en internet, otra alternativa a la que la extensión apunta a futuro es en estudiantes de idiomas extranjeros que puedan beneficiarse de versiones más accesibles del contenido.

2.4 Restricciones

- Dependencia de la API de OpenAI.

- Restricciones de Plataforma.
- Capacidad Limitada para Procesar Sitios Complejos.

2.5 Suposiciones y dependencias

- Se asume que la API de OpenAI estará disponible durante todo el proceso de desarrollo y uso de la extensión, sin interrupciones significativas que afecten el funcionamiento del sistema. También se asume que la API mantendrá una política de precios y tokens accesible para el proyecto.
- Se asume que los navegadores web en los que se instalará la extensión soportarán las tecnologías utilizadas, como Node.js y EJS, y que no habrá restricciones significativas en cuanto a la modificación del contenido HTML de las páginas web.
- Se asume que los usuarios tendrán acceso a Internet.
- El éxito del proyecto depende de la capacidad de la extensión para funcionar en múltiples navegadores populares.
- Se asume que la extensión no violará políticas de privacidad o términos de uso de los sitios web que modifique.

2.6 Requisitos futuros

- Se considerará para una versión futura la opcionalidad de mejorar el contraste visual.
- Se considerará para una versión futura agregar la autorización de pagos en el panel de control de la aplicación.
- Se considerará una extensión de tokens para los usuarios premium que lleguen al límite establecido en una futura versión.
- Se considerará para una versión futura una función que cambie la petición a la IA dependiendo la necesidad del usuario (niños, estudiantes, extranjeros, personas a las que el lenguaje figurado les representa una barrera).

3. Requisitos específicos

3.1 Requerimientos funcionales.

- La extensión debe ser capaz de analizar el contenido textual de una página web y detectar recursos como ironía, metáforas, exageraciones, y otros elementos del lenguaje figurado.
- La extensión debe ofrecer una interpretación en lenguaje literal o sencillo, accesible para el usuario.
- Además de la traducción literal, la extensión puede ofrecer explicaciones sobre el uso y significado del recurso figurado en un formato desplegable. (Funciones A2 y B2).
- La extensión debe permitir al usuario la capacidad de personalizar cómo se presenta la explicación o traducción, seleccionando entre:
    - ***A****. Funciones con explicaciones detalladas*
        - ***A1****. Explicación embebida en el texto*
        - ***A2.*** *Explicación mediante carteles emergentes.*
    - **B.** *Función de respuestas concretas con menor consumo de recursos.*
        - ***B1****. Cambio de contenido en el texto.*
        - ***B2****. Carteles explicativos simples.*
- La extensión debe integrarse con la API de OpenAI para realizar la interpretación del lenguaje figurado en tiempo real. Esto incluye el envío de solicitudes de texto y la recepción de respuestas procesadas.
- La extensión deberá optimizar el uso de tokens para minimizar el costo de las llamadas a la API según la opción seleccionada por el usuario.
- La extensión debe ofrecer diferentes planes de pago basados en el consumo de tokens.
- La extensión debe incluir un panel de control que actuará cómo interfaz de usuario en el cuál podrá personalizar su experiencia.
- La extensión debe guardar las configuraciones de cada usuario usando chrome storage.