UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

TECNOLOGÍAS DE LA INFORMACIÓN

TITULO DEL TRABAJO DE TITULACIÓN

SOFTWARE TRADUCTOR DE LENGUAJE NATURAL A CÓDIGO BRAILLE Y RECONOCIMIENTO DE DISPOSITIVOS DE IMPRESIÓN

AUTOR

PILAY TOMALÁ ELVIS JOSUÉ

PROYECTO DE UNIDAD DE INTEGRACIÓN CURRICULAR

Previo a la obtención del grado académico en
INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

TUTORA

ING. LÍDICE VICTORIA HAZ LÓPEZ, MSI.

SANTA ELENA, ECUADOR

2025

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

Ing. José Sánchez Aquino. Mgt.
DIRECTOR DE LA CARRERA

Ing. Lidice Haz López. Msi.
TUTOR

Ing. Jaime Orozco Iguasnia. Mgt.
DOCENTE ESPECIALISTA

Ing. Marjorie Coronel Suárez. Mgti.
DOCENTE GUÍA UIC

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que luego de haber dirigido científica y técnicamente el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos, razón por el cual apruebo en todas sus partes el presente trabajo de titulación que fue realizado en su totalidad por **PILAY TOMALÁ ELVIS JOSUÉ**, como requerimiento para la obtención del título de Ingeniera en Tecnologías de la Información.

La Libertad, a los 05 días del mes de noviembre del año 2025

TUTOR

Firmado electrónicamente por:
LIDICE VICTORIA HAZ LOPEZ
Validar únicamente con FirmaEC

Ing. Lídice Victoria Haz López, Msi.

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Pilay Tomalá Elvis Josué**

DECLARO QUE:

El trabajo de Titulación, **SOFTWARE TRADUCTOR DE LENGUAJE NATURAL A CÓDIGO BRAILLE Y RECONOCIMIENTO DE DISPOSITIVOS DE IMPRESIÓN**, previo a la obtención del título en Ingeniero en Tecnologías de la Información, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

La Libertad, a los 16 días del mes de noviembre del año 2025

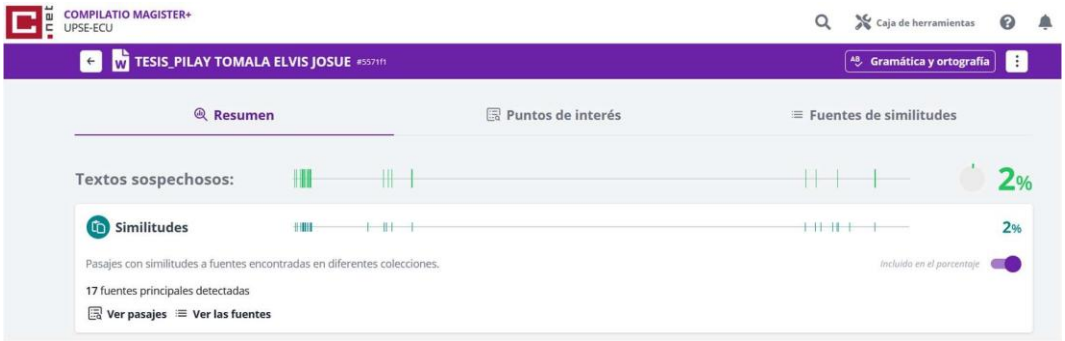
EL AUTOR

Pilay Tomalá Elvis Josué

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CERTIFICACIÓN DE ANTIPLAGIO

Certifico que después de revisar el documento final del trabajo de titulación denominado **SOFTWARE TRADUCTOR DE LENGUAJE NATURAL A CÓDIGO BRAILLE Y RECONOCIMIENTO DE DISPOSITIVOS DE IMPRESIÓN**, presentado por el estudiante, **PILAY TOMALÁ ELVIS JOSUÉ**, fue enviado al Sistema Antiplagio, presentando un porcentaje de similitud correspondiente al 2%, por lo que se aprueba el trabajo para que continúe con el proceso de titulación.



TUTORA

Firmado electrónicamente por:
LIDICE VICTORIA HAZ LOPEZ
Validar únicamente con FirmaEC

Ing. Lídice Victoria Haz López, Msi

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **Pilay Tomalá Elvis Josué**

Autorizo a la Universidad Estatal Península de Santa Elena, para que haga de este trabajo de titulación o parte de él, un documento disponible para su lectura consulta y procesos de investigación, según las normas de la Institución.

Cedo los derechos en línea patrimoniales de artículo profesional de alto nivel con fines de difusión pública, además apruebo la reproducción de este artículo académico dentro de las regulaciones de la Universidad, siempre y cuando esta reproducción no suponga una ganancia económica y se realice respetando mis derechos de autor

Santa Elena, a los 16 días del mes de noviembre del año 2025

EL AUTOR

Pilay Tomalá Elvis Josué

AGRADECIMIENTO

En primer lugar, agradezco a Dios por brindarme la vida, la fortaleza y la sabiduría necesarias para llegar hasta este momento. Su guía ha sido fundamental en cada paso de este camino académico y personal.

A mis padres Rosa y Geovanny, a mi hermana Angie y a toda mi familia, gracias por su amor infinito, por cada sacrificio silencioso y por creer en mí incluso cuando yo dudé. Ustedes han sido mi refugio, mi impulso y mi razón para seguir adelante. Todo lo que he alcanzado lleva un pedacito de ustedes.

A mi amada Nicole, por acompañarme en este proceso, gracias por tus palabras de aliento, por animarme cuando tenía miedo y por recordarme siempre de lo que soy capaz.

A mis amigos que conocí en el trayecto de la Universidad: D. Cisneros, R. Asqui, E. Merchan, C. Vera, a mis amigos de siempre: E. Mero, J. Mero, W. Mero, M. Soriano, A. Parraga, A. Velez, A. Pereira, M. Perez, quienes con su compañía, consejos y risas hicieron más ligero este proceso. Gracias por estar presentes, por escucharme y por darme su apoyo sincero en cada etapa.

Finalmente, expreso mi sincero agradecimiento a mi tutora de tesis Ing. Lídice Haz, cuya guía, conocimiento y dedicación fueron esenciales para la culminación de este trabajo. Su orientación académica y apoyo profesional hicieron posible lograr este objetivo.

Elvis Josué, Pilay Tomalá

DEDICATORIA

A Dios, por darme la fuerza, la claridad y la perseverancia necesarias para llegar hasta este momento.

A mis padres, en especial a mi madre Rosa Pilay por su amor incondicional, por creer en mí incluso cuando dudé, y por enseñarme que los logros se construyen con esfuerzo y humildad, gracias por ser mi apoyo y ayudarme en lo que necesitaba, este logro también es tuyo madrecita querida.

A mi familia, que siempre estuvo presente con palabras de aliento, paciencia y apoyo en cada etapa de este camino.

A mis profesores y mentores, quienes compartieron conmigo sus conocimientos y me guiaron con dedicación para hacer de este proyecto una realidad.

Y finalmente, a mí mismo, por no rendirme, por confiar en mi vocación y por demostrarme que con disciplina y pasión es posible alcanzar cualquier meta.

Elvis Josué, Pilay Tomalá

RESUMEN

El objetivo del proyecto “Software traductor de lenguaje natural a código Braille y reconocimiento de dispositivos de impresión”, fue desarrollar un software en Python que traduzca lenguaje natural a código Braille y facilite el acceso a la información para personas con discapacidad visual, mejorando su calidad de vida e inclusión social. La metodología fue de tipo exploratoria y diagnóstica, empleando técnicas como la encuesta a docentes de la Escuela de Educación Básica “Manuela Espejo”. Así mismo, para desarrollar la propuesta se utilizó el modelo incremental. Los resultados definen que el driver funciona correctamente, donde se concluye que, el sistema cumple totalmente con los requisitos planteados previamente.

Palabras claves: braille, discapacidad visual, software traductor.

ABSTRACT

The objective of the project "Natural Language to Braille Translation Software and Printing Device Recognition" was to develop software in Python that translates natural language to Braille and facilitates access to information for people with visual impairments, improving their quality of life and social inclusion. The methodology was exploratory and diagnostic, employing techniques such as a survey of teachers at the Manuela Espejo Elementary School. An incremental model was used to develop the proposal. The results show that the driver works correctly, concluding that the system fully meets the previously stated requirements.

Keywords: Braille, visual impairment, translation software.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

INTRODUCCIÓN

La escuela de educación básica “Manuela Espejo”, forma parte de las entidades educativas públicas, la cual está situada en la provincia de Santa Elena, cantón La Libertad, focalizándose en un ambiente inclusivo que incorpora niños con discapacidad en sus actividades académicas. Se destaca que, los individuos discapacitados han experimentado diversas deficiencias al momento de comunicarse, son un sin número de dificultades, debido a los costos elevados de los recursos ofrecidos en el sector.

Uno de los inconvenientes que poseen las personas discapacitadas en la carencia de materiales o metodologías que se adapten para los no videntes, revelando que el 10% de este grupo con deficiencia ocular emplea braille para la instrucción de la lectoescritura, de una forma que les permita participar de manera social y profesional, y por tal motivo, laborar y funcionar en el entorno; no obstante, se evidencia que en la mayoría de instituciones educativas no existen aulas especializadas para ciegos y no hay textos o libros impresos en código braille.

Debido a estos inconvenientes, se propone el desarrollo de un software que abarque la traducción automática de textos en lenguaje natural a Braille, incluyendo un módulo avanzado de reconocimiento de dispositivos de impresión que sean compatibles con esta clase de textos, con el fin de facilitar la impresión de textos traducidos, logrando una mayor eficacia y accesibilidad al momento de crear material en braille. Esto beneficiará a los estudiantes con limitaciones visuales, permitiéndoles acceder a contenido textual y académico en formato braille.

Para elaborar el proyecto, se utilizarán métodos de investigación de carácter diagnóstico y exploratorio, con el fin de recopilar datos requeridos, formulando los requerimientos del software; así mismo, se empleó la metodología de desarrollo incremental donde permite la construcción del sistema de manera progresiva.

Se utilizó el lenguaje de programación Python, en conjunto con los recursos open source de Arduino, la librería GRBL y Visual Studio Code para el desarrollo del software. Además, la arquitectura de comunicación del sistema de traducción

braille evidencia el proceso que sigue el software desde la carga de documentos hasta la conversión de los mismos e impresión final de los archivos.

La propuesta, está conformada de los siguientes puntos:

El capítulo I aborda la fundamentación, incluyendo los antecedentes, descripción del proyecto, objetivos, justificación, alcance, metodología, beneficiarios, variables, análisis de recolección de datos y metodología de desarrollo. A su vez, el capítulo II incluye la propuesta, el marco contextual, marco conceptual, marco teórico y marco legal.

Así mismo, el capítulo III incorpora los requerimientos de la propuesta, en conjunto con todas las fases definidas para el desarrollo del proyecto; además, se integran las pruebas. Finalmente, se elaboran las conclusiones y recomendaciones del trabajo, las cuales se encuentran enlazadas de los objetivos definidos.

CAPÍTULO 1. FUNDAMENTACIÓN

1.1. Antecedentes

Desde hace varias décadas, las personas discapacitadas han experimentado múltiples deficiencias al momento de comunicarse; es decir, para que estos individuos tengan oportunidad de formación de gran calidad existe un sin número de dificultades, debido a los elevados costos de los recursos ofrecidos en la industria, por su método de fabricación que es considerado como artesanal [1]. La etapa inicial de escritura se ejecuta empleando una matriz de puntos que se encuentran en relieve, siendo necesaria la vigilancia de un experto, lo que restringe al ciudadano a no ser autónomo ni aprender por sí mismo [1].

Uno de los problemas para las personas con discapacidades perceptivas es la ausencia de materiales o metodologías adaptadas para los no videntes; las estadísticas indican que apenas el 10% de este grupo con deficiencia ocular utiliza braille para la instrucción de la lectoescritura, de una manera que les permita participar social y profesionalmente y, por ende, trabajar y funcionar en el entorno; sin embargo, se observa que en la mayor parte de entidades de educación no hay aulas especializadas para ciegos y no hay textos o libros impresos en braille [2].

Según las estadísticas proporcionadas por el INEC, en el año 2023, hubo 932 casos de personas con discapacidades visuales dentro de la Provincia de Santa Elena, de los cuales 341 eran mujeres y 591 eran hombres; el 42.81% con grado de discapacidad del 30 a 49%; el 23.39% posee de 50 a 74%; el 26.39% presenta de 75 a 84% y el 7.40% tiene de 85 a 100%. Así mismo, con respecto al grupo etario, de 6 a 12 años comprende el 2.25%; de 13 a 18 años es el 4.72%; de 19 a 35 años es el 16.63%; de 36 a 50 años es 16.95%; de 51 a 64 años es el 24.03% y más de 65 años corresponde al 35.09% [3].

La escuela de educación básica “Manuela Espejo”, es parte de las instituciones educativas regulares con financiamiento público, localizadas en la Provincia de Santa Elena, Cantón de La Libertad, la cual se centra en un entorno de inclusión que integra niños con discapacidad en sus actividades escolares; además, promueve prácticas pedagógicas adaptadas que permiten la participación activa de todos los

estudiantes. Su enfoque institucional busca garantizar una educación equitativa y de calidad, fortaleciendo el desarrollo integral de la comunidad estudiantil. [4].
Por medio de la encuesta elaborada a los educadores de la escuela "Manuela Espejo" (**Ver Anexo 1**), se pudo determinar que, reconocen la importancia de la implantación de herramientas digitales para la conversión de documentos a código Braille, en especial, para brindar apoyo a los educandos no videntes. Además, se destaca que la mayoría de educadores ha contado con estudiantes que poseen esta condición; sin embargo, desconocen o no han empleado herramientas tecnológicas para la conversión a sistema Braille.

Así mismo, consideran que un sistema automático para convertir archivos PDF sería sumamente útil para el entorno educativo, ya que mejoraría la calidad del material didáctico que se dispone; no obstante, los docentes mencionan no tener los elementos requeridos como equipos tecnológicos, programas digitales o talento humano para la implementación de dichas herramientas.

Entre los principales inconvenientes para implementar el software de traducción a Braille, los educadores afirman que es la falta de información y la complejidad técnica. De la misma forma, creen que el uso de un traductor de lenguaje natural a Braille tendría una incidencia positiva en la adaptación tecnológica para individuos no videntes en la educación superior. Finalmente, destacan que, un software de conversión debe ser práctico, amigable y fácil de manejar, contando con características como soporte de imágenes y que tenga diversos idiomas, beneficiando a los educadores y alumnos.

A nivel internacional, el artículo titulado "Braille: un dispositivo tecnológico para aprender el sistema braille dirigido a niños con discapacidad visual", se propuso fomentar la evolución y el desarrollo de habilidades para aprender el método Braille en educandos de primaria que son ciegos; se utilizó metodología cualitativa como método de estudio de caso, diseñando y ejecutando el proyecto con diferentes tácticas y valoración final de dichos casos [5]. Los resultados determinaron que se lograron diversas habilidades relacionadas con la alfabetización, las cuales estaban más cerca de las destrezas de desarrollo del niño; así, se concluyó que los procesos

metodológicos que se acompañan de herramientas tecnológicas pueden estimular de manera efectiva al aprender braille [5].

En Latinoamérica, se elaboró el trabajo titulado “Validación de un dispositivo para apoyar el aprendizaje y la enseñanza del sistema Braille”, destacando que una parte poblacional globalmente presenta discapacidades visuales en Colombia, donde no poseen herramientas en el área de la educación para personas no videntes, debido a su precio y la baja distribución en este país; por tal motivo, se plantea validar un dispositivo con realimentación auditiva que sirva de apoyo en la instrucción y capacitación para los ciudadanos no videntes; se utilizó una metodología con cinco fases: revisión literaria, parámetros, diseño y construcción del dispositivo, desarrollo de la plataforma digital y verificación de todo el sistema; se pudo concluir que, se obtuvo una plataforma web funcional conectada a un dispositivo que tiene un signo generador de braille, accionado mediante realimentación auditiva, obteniendo el 99% de exactitud y 100% de precisión [6].

En Ecuador, se realizó el trabajo final “Aprendizaje del sistema de lectura y escritura Braille basado en las Tecnologías de la Información y Comunicación”, donde el desarrollo del software de aplicativo se realizó junto a un ciudadano no vidente, lo que resultó en una app móvil utilizada por individuos con la misma incapacidad, de manera que se opera con gestos táctiles y proporciona retroalimentación audible; la aplicación fue desarrollada en C# con el marco de trabajo Unity y durante las pruebas, se confirmó que la misma, cumple con los requisitos de funcionalidad para los que fue creada, obteniéndose resultados positivos [7].

En la UPSE, se desarrolló la tesis titulada “Implementación de sistema electromecánico braille, para ayuda en la enseñanza y aprendizaje de personas no videntes” con el objetivo de diseñar una impresora braille utilizando un sistema electromecánico basado en tecnología de código abierto para ayudar en el desarrollo formativo de los alumnos no videntes [8]. Se utiliza una máquina de impresión braille como dispositivo que recibe la información, con la cual se logra generar textos escritos en códigos de braille que facilitan la inclusión de personas no videntes [8].

Luego de revisar los proyectos relacionados a la temática principal, se evidencian proyectos realizados que han aportado sustancialmente a la creación de herramientas tecnológicas para la enseñanza del código Braille. Estos estudios servirán como guía para la presente propuesta, debido a sus referencias con respecto a la construcción de software traductor de lenguaje natural a código Braille.

1.2. Descripción del Proyecto

El estudio se enfoca en realizar un software utilizando Python, para la traducción de textos en lenguaje natural a código Braille, diseñado con el fin de optimizar el aprendizaje a individuos no videntes. Esta solución permite un mayor acceso a la información escrita, facilitando así la unión social y académica de los ciudadanos, brindando oportunidades para su desarrollo personal y profesional.

El software se divide en cuatro etapas, especificadas posteriormente:

Fase 1: Diseño del sistema Braille

- Definición de los formatos de entrada, ya que, el software será capaz de traducir el texto en lenguaje natural, ingresando mediante la carga de archivos en formato .docx y .pdf a código Braille.
- Se seleccionarán las bibliotecas apropiadas de Python que se emplearán para el procesamiento de archivos PDF, convirtiendo el texto y manejando la comunicación serial con Arduino.
- Establecer las reglas de convertir texto a sistema Braille en idioma español.
- Implantar el reconocimiento de forma automática en la impresora de Braille.

Fase 2: Desarrollar la traducción de texto a sistema Braille

- Desarrollar la funcionalidad para la carga de documentos PDF y .docx.
- Realizar una extracción de los archivos empleando las librerías necesarias.
- Se implementará el sistema para procesar lenguaje natural, analizando el texto y posteriormente, convertirlo al formato Braille, respetando las reglas de codificación en idioma español.
- Se brinda garantía acerca de una automatización confiable y concisa, de modo que, la codificación se completará exitosamente.

Fase 3: Pruebas de software

- Confirmar que los datos se están transmitiendo correctamente del sistema a la impresora Braille.
- Se ejecutarán pruebas unitarias e integración, garantizando el óptimo funcionamiento de cada módulo de software.
- Se tendrá en cuenta el grado de precisión de la traducción, la usabilidad y la adaptabilidad contextual en diferentes entornos sociales y educativos.

Fase 4: Guía didáctica de traductor Braille

- Desarrollo de un manual de usuario para estudiantes y profesores para que sepan cómo interactuar plenamente con el sistema.
- Incorporar las instrucciones necesarias para que se produzca una interacción adecuada.

Se plantea crear una interfaz gráfica de usuario que pueda ser accesible utilizando un lector de pantalla con navegación por teclado, lo que permite que la misma sea utilizada por niños y educadores o cuidadores con discapacidad visual, quienes son responsables de crear materiales educativos para ellos.

Además, se combinan algoritmos avanzados para el procesamiento de texto con reglas de codificación Braille para diferentes lenguajes, comenzando con el español en la primera fase. También se proporciona la posibilidad de producir archivos en Braille digital, lo que hace que los materiales educativos estén disponibles no solo en forma impresa sino también en forma electrónica, ofreciendo una mayor accesibilidad al permitir que los materiales se distribuyan en formato electrónico accesible.

La propuesta se alinea con los objetivos institucionales de investigación. Según la resolución "**RCF-FSTSO-09 No. 03-2021"** emitida por el "Consejo de la Facultad de Sistemas y Telecomunicaciones", y en concordancia con el "Reglamento de Investigación de la **Universidad Estatal Península de Santa Elena".** De la misma forma, está asociada a la línea de investigación **"Desarrollo de Software (DSS)"**, ligada a la Sublínea "**Desarrollo de Algoritmos y Procesamiento de Lenguaje**

Natural", contribuyendo con la mejora de la accesibilidad y demostrando el compromiso con la innovación tecnológica en beneficio social [9].

1.3. Objetivos del Proyecto

Objetivo general

Desarrollar un software en Python que traduzca lenguaje natural a código Braille para facilitar el acceso a la información de las personas con discapacidad visual.

Objetivos específicos

- Diseñar un software traductor de lenguaje natural a código Braille utilizando el lenguaje de programación Python.
- Elaborar módulos para la entrada de archivos PDF, procesamiento de texto, conversión de texto a código Braille y salida de archivos Braille, asegurando la calidad y precisión de la conversión.
- Crear software controlador para reconocimiento y comunicación de impresora Braille mediante puerto serial (USB).
- Evaluar el sistema mediante pruebas unitarias, de integración y con usuarios, garantizando su funcionalidad.

1.4. Justificación del Proyecto

Los aparatos tecnológicos enfocados en individuos no videntes son fundamentales por su apoyo en el proceso educativo y comunicativo, además de la inclusión social; poseen un impacto significativo, de modo que este grupo de individuos están acostumbrados a depender de alguien más, mientras que con el sistema braille se vuelven independientes en sus actividades y practican de forma autónoma [2]. Es esencial incentivar el aprendizaje a través del sistema braille, de forma que se puede complementar con herramientas de audio, como aplicaciones con comando por voz, audio libros, dispositivos para describir o detallar objetivos, entre otros [6].

La traducción de texto a Braille representa un rol esencial al permitir que los usuarios accedan a contenido escrito en un formato que puedan percibir, lo cual es esencial para su desarrollo educativo y social [10]. Si los ciudadanos no videntes

poseen recursos de sistema Braille, se considera como una forma de que este grupo de individuos puedan integrarse a la sociedad de manera productiva [10].

Debido a los inconvenientes que posee la unidad educativa en cuestión, se propone desarrollar un software que aborde la traducción automática de textos en lenguaje natural a Braille, integrando además un módulo avanzado de reconocimiento de dispositivos de impresión compatibles con este tipo de textos. Este sistema facilitará la impresión de textos traducidos, logrando mayor accesibilidad y eficacia al crear material en Braille.

La implementación del software beneficiará a los educandos con limitaciones visuales, permitiéndoles acceder a contenido textual y educativo en formato Braille, lo que favorece la igualdad de oportunidades; además, apoya los objetivos de políticas nacionales de inclusión social y accesibilidad tecnológica. Por otro lado, la creación y valoración del sistema se implanta con el fin de comprobar la viabilidad que posee, lo que asegura la integración en entidades educativas.

El trabajo se encuentra enmarcado en los fines del "Plan de Desarrollo del Nuevo Ecuador 2024 - 2025" en el enfoque descrito [11]:

Eje 1: Social

Obj. 1.- Optimizar las condiciones vitales de los ciudadanos de manera integral, lo que promueve un acceso equitativo a vivienda, salud y calidad social.

Política 1.2.- Garantizar la inclusión social de los ciudadanos y segmento prioritario de atención.

Obj 2.- Potenciar las habilidades de las personas con educación inclusiva y equitativa, lo que fomenta espacios culturales de intercambio.

Política 2.1.- Fomentar el acceso global a educación equitativa e inclusiva, idónea e intercultural para niños, jóvenes y adultos, potenciando una permanencia y finalización de sus estudios.

Política 2.2.- Garantizar una formación de calidad con enfoque innovador, inclusivo, participativo y cooperativo, fortaleciendo las capacidades de cognición, socioemocionales, comunicativas y digitales.

Política 2.4.- Crear un sistema educativo superior por medio de diversas alternativas de estudio y carreras, profundizando en la educación técnica digital como opción para profesionalizar a los alumnos.

1.5. Alcance del Proyecto

Considerando los inconvenientes antes descritos, es importante automatizar el proceso de traducción de texto a código Braille; motivo por el cual, se propone el desarrollo de un software utilizando Python, para la traducción de textos en lenguaje natural a código Braille, diseñado con el fin de brindar mejoras en la optimización dirigida a individuos discapacitados visualmente, brindando el acceso a información escrita, lo que favorece su integración social y educacional.

El software está dividido en cuatro fases, las cuales son:

La fase 1 se centrará en el diseño del sistema Braille, donde se definen los formatos de entrada, debido que, el software será capaz de traducir el texto en lenguaje natural, ingresando mediante la carga de archivos en formato .docx y .pdf a código Braille. Además, se eligen las librerías de Python para el procesamiento de los archivos PDF, la conversión del texto y posteriormente, realizar la comunicación serial con Arduino. Así mismo, se establecen las reglas de conversión de texto a sistema Braille en idioma español y se implementa la detección automática de la impresora Braille.

En la fase 2, se realiza la programación del traductor de información a código de impresión Braille, donde se desarrolla toda la funcionalidad para la carga de documentos PDF y .docx. Además, se realiza una extracción de los archivos empleando las librerías necesarias. Se implementa el motor para procesar lenguaje natural, analizar el texto y posteriormente, convertirlo al formato Braille, respetando las reglas de codificación en idioma español. Así mismo, se realiza la conversión automática, reduciendo la necesidad de la intervención manual y asegurando una transformación precisa y confiable.

En la fase 3, se ejecutan las pruebas de software, verificando que los datos sean enviados de manera correcta desde el sistema hasta la impresora Braille. De la misma forma, se realizan las pruebas unitarias y de integración, asegurando el

funcionamiento idóneo de cada parte del sistema, además de valorar elementos como la precisión de traducción, facilidad de uso y capacidad de adaptación en diferentes contextos sociales y educativos.

Finalmente, en la fase 4 se elabora una guía didáctica o manual de usuario del funcionamiento del traductor Braille, dirigido a los estudiantes y docentes que utilizarán el software, con el fin de que conozcan cómo utilizar el sistema en su totalidad. Por ende, se incluirán todos los pasos necesarios para el uso correcto.

Se contempla el diseño de una interfaz gráfica accesible, utilizando tecnologías como lectura de pantalla y navegación por teclado, lo que garantiza la usabilidad para los escolares no videntes y para los educadores o cuidadores, encargados de generar materiales educativos adaptados a sus necesidades.

De la misma manera, se integran algoritmos avanzados de procesamiento de texto, respetando las reglas de codificación Braille en distintos idiomas, iniciando con el español como primera fase; así mismo, se ofrece la posibilidad de generar archivos en formato Braille digital, ampliando su aplicabilidad más allá del ámbito impreso, lo que permite distribuir materiales educativos en formato electrónico accesible.

1.6. Metodología del Proyecto

1.6.1. Metodología de Investigación

En este trabajo se empleará la metodología de investigación cuantitativa, con el fin de medir y analizar objetivamente la efectividad del software traductor de lenguaje natural a código Braille, así como el impacto que posee en la inclusión de ciudadanos discapacitados visualmente. Esta orientación permitirá recolectar datos medibles y numéricos por medio de encuestas, facilitando la evaluación precisa del software [12].

Además, es un proyecto investigativo descriptivo y experimental, de modo que, se recopilan datos que permitirán evaluar la funcionalidad y usabilidad del sistema, desde la conversión de texto a Braille hasta el reconocimiento de los dispositivos de impresión especializados. Por tal motivo, el enfoque descriptivo permitirá caracterizar el software; mientras que, el experimental evaluará su rendimiento en diversos entornos controlados, para identificar posibles mejoras.

1.6.2. Beneficiarios del Proyecto

El desarrollo e implantación del sistema de traducción de lenguaje natural a Braille, beneficiará a las personas no videntes de la Escuela de Educación Básica "Manuela Espejo", permitiendo el acceso a contenido textual y educativo en formato Braille, favoreciendo la igualdad de oportunidades. Así mismo, los docentes podrán navegar a través del sistema, generando materiales educativos que se adapten a los requerimientos de los escolares.

Por otro lado, también se benefician los ciudadanos en general que poseen discapacidad visual, ya que, el software puede ser utilizado en otras instituciones educativas y hogares, siendo viable para traducción de texto en formato PDF o .docx a sistema Braille.

1.6.3. Variables

El propósito de este trabajo es traducir lenguaje natural a código Braille para los individuos con ceguera. Posteriormente, se describen las variables:

- **Uso del software traductor de lenguaje natural a código Braille:** Esta variable hace referencia a la interacción que tienen las personas discapacitadas con el software diseñado para la conversión de texto de lenguaje natural a código Braille. Aquí se evaluará la implementación, usabilidad y funcionalidad del software.
- **Acceso a recurso didáctico para personas con ceguera:** Esta variable mide el resultado final del uso del software; es decir, la cantidad y calidad de herramientas educativas accesibles en Braille que serán generadas para individuos con problemas de visión. El enfoque está en como el uso del software optimiza la producción de materiales educativos accesibles.

1.6.4. Análisis de recolección de datos

En la presente propuesta, se elaboraron encuestas a los docentes de la Escuela de Educación Básica "Manuela Espejo" (**Ver Anexo 1**), cuyo propósito fue comprender los ideales de los docentes acerca de la implantación de recursos digitales que faciliten convertir documentos a código Braille, cuyos resultados

obtenidos permitieron identificar el nivel de conocimiento y disposición del personal para adoptar estas herramientas.

En la encuesta se pudo determinar que, los educadores destacan el valor de la implantación de herramientas digitales para la conversión de documentos a código Braille, en especial, para brindar apoyo a los educandos no videntes. Así mismo, la mayor parte de educadores revela que han contado con estudiantes que poseen esta condición; sin embargo, desconocen o no han empleado herramientas tecnológicas para la conversión a sistema Braille.

De la misma forma, consideran que un sistema automático para convertir archivos PDF sería muy útil para el entorno educativo, ya que mejoraría la calidad del material didáctico que se dispone; no obstante, los docentes mencionan no tener todos los instrumentos requeridos, como equipos, programas o talento humano para la implementación de dichas herramientas.

Entre los principales desafíos para la implementación del software de traducción a Braille, los educadores afirman que es la falta de información y la complejidad técnica. Así mismo, creen que el uso de un traductor de lenguaje natural a Braille tendría una incidencia positiva en la equidad tecnológica de los usuarios discapacitados visualmente en la educación superior. Finalmente, destacan que, un software de conversión debe ser práctico, amigable y fácil de manejar, contando con características como soporte de imágenes y que tenga varios idiomas, beneficiando a los educadores y alumnos.

1.7. Metodología de desarrollo

La ejecución del trabajo se llevará a cabo empleando el modelo incremental, que permite construir el sistema de manera gradual mediante incrementos, incluyendo las diversas funcionalidades en cada iteración, permitiendo realizar las modificaciones necesarias, la cual se centra en cuatro etapas: [13]:

- **Análisis:** Es la recopilación de los requisitos del sistema, por medio de diversas técnicas para comprender lo que necesitan los involucrados. En esta fase se incluye la identificación de las necesidades con respecto a los

formatos de entrada, reglas de codificación Braille y otros elementos específicos.

- **Diseño:** Creación de la estructura del software, mediante las herramientas elegidas, permitiendo la planificación de los módulos principales. Se realiza una planificación de la interfaz gráfica y se establece un flujo de trabajo para cada etapa de la conversión de texto a Braille.
- **Codificación:** Se realiza la programación en Python y con la herramienta Visual Studio Code, garantizando la funcionalidad del sistema. Se integra el desarrollo de los algoritmos para la traducción de texto a Braille y la conversión de formatos de archivos .pdf y .docx.
- **Pruebas:** Ejecución de pruebas finales, incluyendo accesibilidad, usabilidad y experiencia de los usuarios a través del software, garantizando que el mismo, cumpla con los requisitos planteados.

Para realizar los debidos incrementos, se consideran las fases del proyecto mencionadas en el alcance, siendo estas:

Incremento 1: Definición de los formatos de entrada

Incremento 2: Selección de bibliotecas y diseño de la interfaz

Incremento 3: Desarrollo de la carga de archivos y extracción de texto

Incremento 4: Implementación de la conversión básica de texto a Braille

Incremento 5: Integración con la impresora Braille

Incremento 6: Pruebas de funcionalidad

Incremento 7: Creación de manual de usuario

Figura 1: Modelo de desarrollo incremental

CAPÍTULO 2. PROPUESTA

2.1. Marco Contextual

2.1.1. Escuela de Educación Básica "Manuela Espejo"

La escuela "Manuela Espejo" ofrece instrucción primaria y está localizada en Santa Elena; esta institución de educación regular está abierta a todo el público, sin embargo, su sostenimiento es fiscal, su jurisdicción es hispánica y el tipo de enseñanza que ofrece es presencial tanto en la mañana como en la tarde, además incluye jardines de infancia y educación general básica [4].

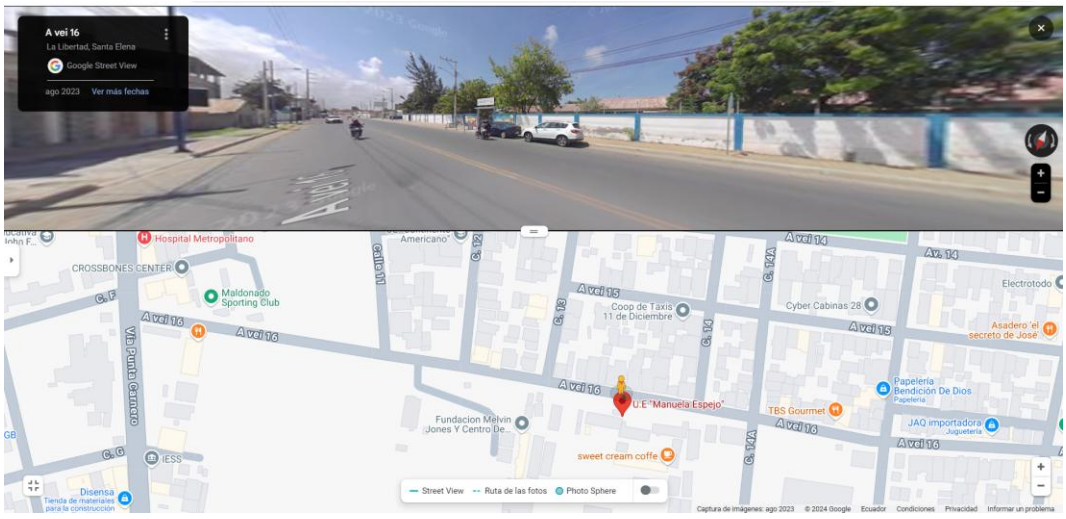


Figura 2: Ubicación geográfica de la institución

2.1.1.1. Misión

Brindar una enseñanza integral con calidez y calidad, formando a su vez, personas respetuosas de sus semejantes de la naturaleza y ambiente, inclusiva sin distinguir credos, razas y géneros; mediante derechos de moral y ética, siendo pilares fundamentales para su buen desenvolvimiento en la vida [4].

2.1.1.2. Visión

La Unidad Educativa "Manuela Espejo", facilitará el desarrollo de potencialidades de los alumnos, proporcionando una educación integral y formando personas respetuosas y del ambiente que les rodea, creando así, oportunidades para incluir a toda la comunidad educativa, respetando individualidades y necesidades de los estudiantes con el fin de formar ciudadanos responsables y dignos [4].

2.1.2. Personas no videntes en la Provincia de Santa Elena

La discapacidad es una categoría de análisis donde se hallan personas que presentan deficiencias en la salud; existen diversos tipos, como: física, mental, auditivas, entre otras [14]. Aproximadamente, 1 billón de personas presenta alguna discapacidad, lo que equivale al 15% de la población, y el 80% de este total está dentro de la edad laboral [14].

Ecuador es un país latinoamericano que tiene un nivel alto de personas discapacitadas; alrededor del 45% de ciudadanos registrados en el CONADIS tienen diversas clases de discapacidades, incluyendo la discapacidad intelectual y física [3]. Por otro lado, en la provincia de Santa Elena, viven 11,010 personas con incapacidades múltiples, de las cuales el 8.47% corresponde a discapacidad visual [3].

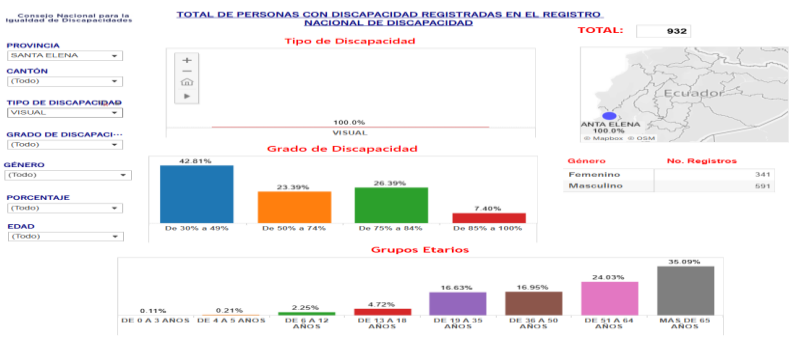


Figura 3: Total de ciudadanos no videntes en Santa Elena [3]

El INEC plantea estadísticamente que, en al año 2023, se registró que 932 personas en la Provincia de Santa Elena presentan deficiencia visual, donde 341 son mujeres y 591 son hombres. El 42.81% con grado de discapacidad del 30 a 49%; el 23.39% posee de 50 a 74%; el 26.39% presenta de 75 a 84% y el 7.40% tiene de 85 a 100%. Así mismo, con respecto al grupo etario, de 6 a 12 años comprende el 2.25%; de 13 a 18 años es el 4.72%; de 19 a 35 años es el 16.63%; de 36 a 50 años es 16.95%; de 51 a 64 años es el 24.03% y más de 65 años corresponde al 35.09% [3].

2.1.3. Desafíos de las personas no videntes

La vista puede tener problemas en alguno de sus aspectos, incluso llegar a ser completamente inactiva, y dicha incapacidad se denomina deficiencia ocular. A través de la vista, se puede adquirir un 80% de la información del mundo [15]. Se consideran dos tipos de discapacidad visual [15]:

- **Deficiencia visual:** Es aquella en la que hay una reducción importante, aunque aún se pueda distinguir la luz, y es posible orientarse hacia ella y utilizarla con motivos básicos [15].
- **Ceguera:** Es total o en algunos casos reducida y no hay la posibilidad de ver o sentir la luz de una manera que contenga sentido; esta clase de personas utilizan sus manos y oídos como su principal fuente de la obtención de información [15].

Por otro lado, los desafíos que enfrentan estas personas, son los siguientes [16]:

- Problemas para la lectura de materiales que no tienen un tipo de letra adecuado, es decir, fuentes de tamaño grande y de color sólido sobre un fondo; también incluye transcripciones a Braille o información en audio [16].
- Problemas en la detección de obstáculos [16].
- Dificultades de percepción que les permita la identificación de objetos, espacios o personas de manera visual [16].
- Inconvenientes en la orientación de espacios nuevos o desconocidos [16].
- Limitaciones en el acceso a plataformas digitales que no cuentan con opciones de accesibilidad [16].

2.1.4. Marco Legal

En la Constitución de la República del Ecuador, constan los siguientes artículos acordes al tema propuesto [17]:

- **Art. 35.-** Reconoce a los individuos como población prioritaria, donde se debe garantizar su inclusión social total.
- **Art. 47.-** El estado respaldará las estrategias preventivas de las incapacidades con la familia y comunidad, procurando equiparar oportunidades para los individuos discapacitados e inclusión social.

Por otro lado, en la Ley Orgánica de Discapacidades, se hallan los artículos [18]:

- **Artículo 4.- Principios esenciales.-** Este reglamento está sujeto y se basa en los principios:
    - **Sección 3:** Igualdad de oportunidades.- Todos los ciudadanos discapacitados son similares ante la normativa en cuanto a la integridad legal y se benefician de la ley indiscriminada.
    - **Sección 8:** Se prevé la admisión de barreras físicas, informacionales, de transporte, así como de sistemas y tecnología de comunicación para las personas con discapacidad.
    - **Sección 10:** Atención especial.- Se tomarán disposiciones en el plan o programa de vida ordinario para que los individuos discapacitados tengan atención especial y arreglos de zonas preferenciales para satisfacer sus requisitos individuales o grupales especiales.
- **Artículo 23.- Ayudas técnicas.-** El sistema de atención a la salud debe contar con las contribuciones operativas y digitales apropiadas para la atención de los ciudadanos discapacitados físicamente.
- **Artículo 28.- Educación especial.-** El titular educativo nacional podrá adoptar las normas apropiadas para fomentar la incorporación de alumnos con requerimientos específicos que necesiten ayudas tecnológicas u otras formas de apoyo.
- **Artículo 33.- Accesibilidad servicios escolares.-** La autoridad educativa nacional, dentro de su jurisdicción, debe buscar que los textos y recursos pedagógicos en Braille estén disponibles sin costo alguno.

De la misma forma, se relaciona con los **Objetivos de Desarrollo Sostenible (ODS)** en los siguientes apartados [19]:

- **ODS 4.- "Educación de calidad:** Se alinea de forma directa con la meta 4.5., la cual busca eliminar disparidades de género y asegurar un acceso equitativo a todos los niveles educativos para los individuos discapacitados. De esta forma, el desarrollo de un software que traduzca lenguaje natural a código Braille permite que los educandos con discapacidad visual tengan acceso a recursos académicos en su sistema de lectura, promoviendo una educación más inclusiva y adaptada a sus necesidades" [19].
- **ODS 9.**- "**Industria, innovación e infraestructura:** Responde a la meta 9.5, centrándose en mejorar la investigación científica y la capacidad tecnológica de sectores industriales, en especial, en países desarrollados, al incorporar la inteligencia para la traducción de lenguaje natural a Braille y compatibilidad con impresoras especializadas, el proyecto impulsa la innovación inclusiva en el área de accesibilidad, contribuyendo además con el desarrollo de infraestructura tecnológica adaptada para personas discapacitadas" [19].
- **ODS 10.**- "**Reducción de las desigualdades:** Se relaciona con la meta 10.2, promoviendo la inclusión social, política y económica de todos los ciudadanos, independientemente de su condición, al crear un recurso digital que reduce las barreras de acceso a la información escrita para personas con ceguera, donde el proyecto actúa como un medio para mejorar su autonomía, participación educativa y social" [19].

2.2. Marco Conceptual

2.2.1. Sistema Braille

El Braille hace uso de letras y números que se representan mediante seis puntos en relieve, los cuales utilizan para traducir letras, números, y hasta símbolos matemáticos, musicales, y científicos [20]. Además, posibilita a que los ciudadanos discapacitados visualmente lean mediante el tacto, lo que les garantiza una comunicación de la información, por ende, el Braille se considera como un medio comunicativo que contribuye con la independencia [20].

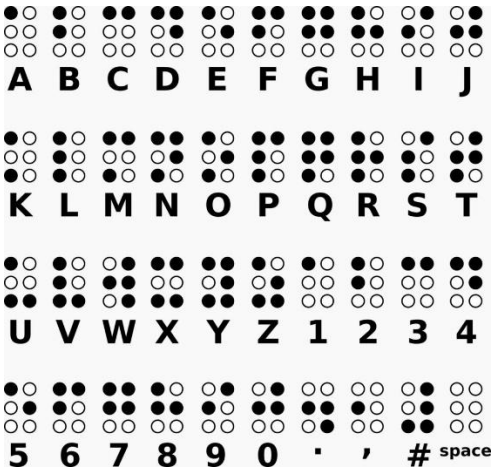


Figura 4: Sistema Braille [20]

Cada celda del braille puede tener entre uno a seis puntos en relieve, y sus combinaciones permiten a las personas la lectoescritura mediante el sentido táctil; el Braille no solo se emplea en libros y documentos impresos en braille, sino también en señales públicas, etiquetas de productos y botones de ascensores, lo que permite una accesibilidad mayor para los individuos con discapacidad visual [21].

2.2.1.1. Origen del sistema Braille

El sistema Braille se inventó por Louis Braille, que a los 15 años elaboró un sistema de lectura para individuos con deficiencias en la vista, utilizando el potencial de captar mediante las terminaciones de los nervios en las yemas de los dedos, el símbolo generador o como es conocido, celda Braille, que se conforma por 6 puntos organizados en 2 columnas o 3 ítems cada una, en función de la presencia o carencia de estos puntos, obteniendo 64 combinaciones distintas que representan diversas letras del alfabeto [22]. El código Braille es una de las herramientas más usadas para que los alumnos accedan a datos en caso de que no dispongan de un ayudante para esto, cuya simpleza hace que sea un sistema fácil de comprender y práctico a través de su memorización y repetición [22].

De la misma forma, el sistema Braille ha tenido que evolucionar notablemente desde su creación, adaptándose a diversos idiomas, avances tecnológicos y disciplinas académicas; inicialmente concebido para la representación de letras y números, con el tiempo se han ido desarrollado variantes que permiten la escritura y lectura de notación matemática, fórmulas científicas y música, ampliando de

manera considerable sus aplicaciones en el área educativa; de la misma manera, la implantación del código Braille en dispositivos electrónicos, teclados y pantallas táctiles ha permitido la incorporación más eficaz en el mundo digital, lo que ayuda a formar una comunidad con mayor inclusividad [23].

2.2.1.2. Caracteres especiales

Existen diferentes caracteres especiales dentro del sistema Braille, como signos de puntuación letras mayúsculas y números [24]. Dentro de los signos de puntuación se puede hallar el punto, coma, punto y coma, dos puntos, signos de interrogación, admiración, paréntesis y comillas. Para la representación de las letras mayúsculas, se antepone un caracter especial que se sobre entiende por cada caracter mayúsculo que se plasma en el código Braille [24]. En el caso de los números, son símbolos de doble celda o símbolos dobles, pues requieren de ello para representarse [24]. Los números se forman por las letras a, b, c, d, e, f, g, h, i y j en conjunto con el signo especial de Braille, formando números del 0 al 9 correspondientemente [24].

2.2.3. Traducción automática

La traducción automática consta del uso de la IA para la transcripción de forma automática un escrito de un lenguaje a otro sin requerir de un humano; traducir de manera autónoma no se restringe solo a la simple traducción del texto, también transmite el significado del texto de un idioma particular a un idioma objetivo y, además, examina todos los componentes del texto en relación a cómo interactúan las palabras entre sí [25].

Un recurso clave en el proceso de traducción es el uso de máquinas para hacer la transcripción autónoma, utilizándose por sí sola o combinándose, para incrementar la productividad del traductor, permitiendo [26]:

- **Traducir mayor contenido más rápido:** Se pueden traducir millones de palabras en un instante, convirtiéndola en una solución integral para proyectos grandes o con plazos rápidos de entrega.
- **Acceder a varios idiomas:** Se encuentra disponible en diversas combinaciones de idiomas distintas, por lo que obtener las traducciones en cualquier idioma es fácil y rápido.

- **Reducir costos de traducción:** Con motores neuronales, los resultados de la traducción automática fluyen más rápido y con una calidad superior significativamente, reduciendo la carga de trabajo.

2.2.4. Métodos y algoritmos para la traducción de texto a Braille

Python como único lenguaje de codificación es sencillo de aprender y muy potente, dispone de estructuras de información y codificación orientada a objetos, cuya sintaxis, por medio de su dinamismo y entorno interpretado, lo hace idóneo para la programación y la creación rápida de diversas apps [27].

Python se puede utilizar para múltiples usos, desde programar en web hasta análisis de datos, la IA y machine learning; debido a su versatilidad y sus bibliotecas extensas que facilitan la realización de tareas complejas con menor esfuerzo y mayor volumen de código; así mismo, la comunidad de Python es amplia y activa, permitiendo ofrecer muchos módulos y bibliotecas [27].

2.2.5. Discapacidad visual

La visión es, sin lugar a dudas, el sentido más importante y de mayor uso, al igual que cualquier otra actividad, se requiere de un cierto esfuerzo para trabajar, caminar, aprender, leer y participar; abusar de la vista lleva a problemas en el futuro; Las personas dejan de lado su salud, lo que hace imprescindible el acceso a atención oftalmológica [28]. La discapacidad visual producida por la afección ocular impacta al sistema visual y, además, se relaciona con las funciones asociadas a la vista; por lo tanto, la ceguera tiene un impacto negativo en una persona durante toda su vida, aunque con el acceso oportuno a una atención oftalmológica de calidad, el impacto puede ser minimizado [28].

En todo el mundo, se estiman en 1300 millones los ciudadanos que tienen alguna clase de problema ocular; en cuanto a la miopía, 188.5 millones presentan alguna deficiencia de visión leve, 217 millones tienen deficiencia en la vista de forma intermedia a grave y 3 millones son completamente ciegos, donde las causas primordiales son los problemas de cataratas, destacando que tienen una edad mayor a 50 años, lo que refleja la magnitud del impacto global de las afectaciones visuales y la necesidad de reforzar los sistemas de prevención y atención temprana [29].

2.2.5.1. Terminología de la discapacidad visual

Percepción visual

Es la facultad de los individuos para interpretar el entorno mediante sus ojos, moldeando cada aspecto de lo que perciben por medio de los mismos; la agudeza visual también destaca la oportunidad de identificar detalles y diferenciarlos entre otros que sean semejantes [30]. Los optotipos son los recursos más apropiados para medir este aspecto visual [30].

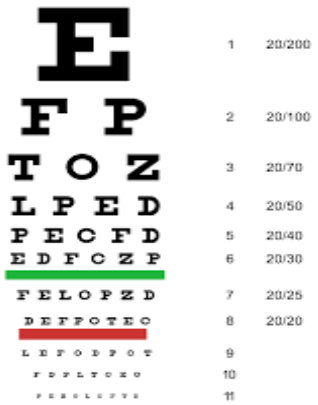


Figura 5: Optotipos [30]

Campo visual

Es la habilidad de ver y reconocer elementos que se encuentran más allá del punto enfocado, considerando que lo que se visualiza de frente se encuentra en la zona de visión de mayor claridad [31]. Para la medición de este aspecto es requerido usar campímetros, que son óptimos para saber el nivel de campo visual que percibe el individuo comúnmente [31].

Figura 6: Campímetros [31]

2.2.5.2. Clases de ceguera

Existen diversas clases de ceguera o discapacidad visual, que ayudan a clasificar mejor a los individuos según su deficiencia en la vista, las cuales son [32]:

- **Ceguera:** Se refiere a la falta completa de percepción del entorno a través de los ojos, lo que lleva a la incapacidad para realizar actividades diarias, como lo haría un individuo sin esta condición.
- **Deterioro visual profundo:** Este tipo de ceguera impide a las personas realizar tareas que requieren un gran grado de atención al detalle, como coser, fotografía y otras.
- **Deterioro visual severo:** Un tipo de deterioro visual que limita severamente la realización de tareas que requieren una atención considerable al detalle y que solo se pueden completar con la ayuda de un individuo sin discapacidad.
- **Deterioro visual leve:** Acompañando esta condición está la necesidad de una iluminación bien controlada y la realización de tareas que se consideran ordinarias.

2.2.6. Tecnologías utilizadas para reconocimiento de dispositivos de impresión

Los avances tecnológicos deben estar al servicio de los individuos con discapacidad visual, para la mejora en sus actividades cotidianas, ser independientes y llevar una vida normal, con la misma igualdad que los ciudadanos sin ninguna incapacidad [33]. Las tecnologías emergentes enriquecen el bienestar de la población con discapacidad, además de generar un impacto positivo en su inserción al trabajo.

Esto significa que, dependiendo del nivel de discapacidad, el uso de ayudas tecnológicas puede apoyar a una persona a volverse más autónoma [34]. Algunas de las apps más utilizadas, son Google Talk Back, Siri, VoiceOver, SVisual y Disabled Park [34].

2.2.7. Reconocimiento y comunicación con dispositivos de impresión

En este apartado, se abordarán los elementos técnicos fundamentales para garantizar que el software sea compatible con diferentes dispositivos de impresión Braille, analizando las clases de impresoras disponibles, protocolos de

comunicación más usuales, detección automática del hardware e interoperabilidad del software con sistemas operativos distintos.

2.2.7.1. Tipos de impresoras Braille

Se hallan diferentes clases de impresoras Braille, tal como se muestran a continuación:

Tipo de impresora	Detalle
Embosadoras	Impresoras mecánicas especializadas que crean puntos en relieve por medio de cabezales de impacto sobre papel grueso, usualmente cartulina, las cuales funcionan semejante a una impresora de matriz de puntos, donde cada carácter Braille está compuesto por combinaciones de 6 a 8 puntos de relieve. Además, algunas versiones permiten la impresión a doble cara, optimizando el uso de papel [35].
Impresoras térmicas	Funcionan aplicando calor sobre el papel termosensible formando puntos Braille, sin requerir impacto mecánico, reduciendo el ruido y desgaste, siendo más rápidas y silenciosas, lo cual las hace idóneas para bibliotecas o aulas; no obstante, necesitan papel especial con una capa química reaccionando al calor, lo que puede ser costoso y poco accesible en diversos contextos [36].
Electro – mecánicas	Combinan componentes eléctricos con mecanismos mecánicos de precisión alta, empleando señales electrónicas para activar punzones que generan puntos Braille en el papel, siendo más robustas, capaces de imprimir de forma rápida y con calidad excelente de relieve, lo que las hace apropiadas para entidades especializadas o centros de producción de material accesible; así mismo, su durabilidad y eficacia permiten entender volúmenes grandes de impresión [37].

Tabla 1: Tipos de impresoras Braille

2.2.7.2. Protocolos de comunicación

Con el fin de que el software traductor funcione de forma correcta con distintos modelos de impresoras Braille, es fundamental entender los protocolos de comunicación que permiten transferir datos entre el hardware y el sistema. A continuación, se definen los principales usados en la actualidad:

- **USB:** Es el protocolo más extendido en impresores modernas Braille, ofreciendo una velocidad alta de transferencia, configuración fácil y compatibilidad con la mayor parte de sistemas operativas, cuya capacidad de detección automática facilita la conexión inmediata del dispositivo sin requerir de configuraciones complejas manuales [38].
- **Serial:** Empleado en modelos con mayor antigüedad, este protocolo necesita una configuración manual de parámetros como baud rate, aunque es más lento que el USB, es confiable y estable en entornos con limitada infraestructura tecnológica, siendo útil para equipos que operan en un hardware legado [39].
- **Bluetooth:** Brinda conectividad inalámbrica de alcance corto, siendo idóneo para dispositivos móviles o aplicaciones portátiles que necesitan imprimir sin conexión de cables; aunque la velocidad de transferencia es menor, ofrece gran comodidad en entornos domésticos y pedagógicos, requiriendo emparejamiento previo [40].
- **Wifi:** Permite conectar la impresora a una red local inalámbrica, habilitando el acceso remoto desde diversos dispositivos, siendo especialmente útil en entidades educativas y centros de impresión, permitiendo imprimir en red, aunque depende también de una configuración red y puede afectarse por la calidad del Internet [41].

2.2.7.3. Detección y reconocimiento automático de hardware

La detección automática del hardware de impresión es un elemento fundamental que contribuye con una experiencia accesible y sin complicaciones técnicas para los individuos; dicha funcionalidad permite que el sistema detecte de manera inmediata el modelo de la impresora Braille conectada. Esto optimiza el tiempo de configuración y reduce los errores de compatibilidad entre dispositivos.

- **Drivers universales:** Permiten que el S.O. detecte y conecte impresoras Braille sin requerir la instalación de controladores específicos para cada uno de los dispositivos, facilitando la instalación y minimizando los problemas de compatibilidad con distintos modelos de impresoras [42].
- **Interfaces estándar:** Se utilizan interfaces, tales como HID permitiendo que el sistema operativo pueda reconocer a la impresora automáticamente sin ajustes adicionales, lo que mejora la usabilidad y compatibilidad con diversos tipos de equipos [43].
- **Vinculación inmediata:** Posterior a que el sistema se enlaza por medio de los puertos inalámbricos, se detectan instantáneamente las impresoras Braille sin requerir que intervenga el usuario, por lo cual se potencia la usabilidad [44].

2.2.7.4. Integración entre software traductor y S.O

Es esencial considerar la conectividad entre el software traductor Braille y los diversos S.O. para que las personas tengan acceso a la información sin tomar en cuenta el ambiente tecnológico que empleen; el mismo se debe adaptar a diferentes portales, permitiendo su utilización en diferentes configuraciones; a su vez, se incluyen herramientas y marcos que faciliten el uso de archivos traducidos en código braille [45].

En este sentido, la interacción entre el software y S.O. implica que el sistema se debe mantener actualizado con versiones vigentes, previniendo fallas en el empleo del mismo; por otro lado, debe ser fácil de instalar y ajustar, siendo dinámico para los usuarios que lo utilicen [46].

2.2.8. Ventajas del uso de software libre para la inclusión tecnológica

El software libre representa un recurso esencial para promover la inclusión tecnológica basada en principios de colaboración, libertad y acceso abierto, cuyo uso contribuye a sectores vulnerables o con menos recursos, como personas discapacitadas visualmente o entidades sociales, permitiendo soluciones personalizadas y sostenibles [47].

Las principales ventajas del software libre, son [47]:

- Cualquier persona o institución puede acceder, distribuir y modificar el software sin restricciones económicas ni legales.
- No necesita pagos por licencias ni mantenimiento obligatorio, haciendo que sea ideal para entidades educativas.
- Al tener una comunidad activa que revisa el código fuente, se identifican y corrigen vulnerabilidades con mayor agilidad.
- El desarrollo cooperativo garantiza una mejora constante en el funcionamiento del software.
- Las libertades que ofrece impulsan el desarrollo de soluciones e ideas, especialmente adaptadas a requerimientos específicos.
- Favorece entornos donde el conocimiento es compartido para la resolución de problemas de manera colectiva.

2.2.9. Herramientas para la creación del software

2.2.9.1. Python

Es un lenguaje de codificación usado en diseñar portales web, codificar software, ciencia de datos, así como también en machine learning; su empleo es popular por su sencillez y eficiencia, así como por su capacidad de ser ejecutado en todas las plataformas [48]. Posee diversos beneficios, como [48]:

- Por su sintaxis simple similar al inglés, los desarrolladores pueden comprender y leer programas en Python con sencillez.
- Resulta más productivo para los programadores porque permite que se puedan realizar programas en Python en un menor número de pasos.
- Cuenta con una gran librería que proporciona innumerables códigos que se pueden utilizar para cualquier tipo de trabajo.
- Python se puede usar por programadores junto con otros lenguajes como C, Java o C++.
- Este lenguaje de programación tiene a su disposición gran cantidad de recursos para su uso en forma de tutoriales o videos en la web.
- Es un lenguaje de código abierto.
- Su amplia compatibilidad con frameworks y herramientas lo convierte en una opción versátil para el desarrollo de aplicaciones modernas.

2.2.9.2. Visual Studio Code

Permite editar la codificación, diseñado para ser rápido, ligero y personalizable, en comparación a otros editores de código; además, se centra en ofrecer una experiencia ágil para escribir y editar el código fuente, sin requerir características avanzadas, entre las cuales destacan [49]:

- Es intellisense y posee autocompletado inteligente.
- Cuenta con depuración integrada, es decir, se puede ejecutar el código directamente en el editor, configurando los breakpoints.
- Tiene control de versiones con Git, gestionando los repositorios y realizando commits cuando sean necesarios.
- Posee un enfoque en la personalización, destacando ante otros editores de código fuente.

2.2.9.3. Arduino

Figura 7: Arduino [50]

Es un entorno de diseño electrónico open source, centrado en software y hardware libre, fácil de emplear y flexible para los desarrolladores, permitiendo la creación de distintas clases de microordenadores de placa única a los que los programadores puede dar distintas clases de uso [50]. Arduino brinda bases para que un individuo u organización pueda elaborar sus propias placas, siendo funcionales a partir de la misma base [50].

2.2.9.4. GRBL

Es un firmware open source que permite controlar el movimiento en máquinas CNC, donde se puede instalar de forma sencilla el firmware GRBL en un Arduino, permitiendo obtener rápidamente un controlador CNC de alto rendimiento y económico [51]. El GRBL emplea código G como entrada, emitiendo el manejo del movimiento por medio de Arduino [51].

A continuación, se muestra este proceso en el siguiente diagrama [51]:

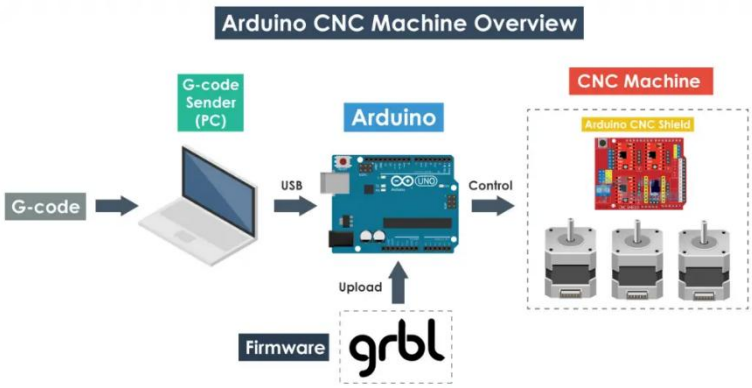


Figura 8: Diagrama del rol del GRBL [51]

El diagrama evidencia el rol del GRBL en el funcionamiento general de la máquina CNC, que se trata de un firmware que se debe transferir al Arduino con el fin de poder manejar la función de los motores de la máquina CNC, cuyo propósito es convertir código G en el desplazamiento de motores [51].

Para la configuración del GRBL se requiere un Arduino o placa compatible, basada en Atmega 328; sin embargo, Arduino uno es la alternativa más adecuada, debido a su disponibilidad amplia y facilidad de uso [52]. Para reducir el cableado y conexiones entre la placa y componentes de la máquina CNC, se recomienda una placa CNC, que incluye controladores de motores paso a paso y conexiones para control de husillo, interruptores de límites, entre otros periféricos [52].

2.3. Marco Teórico

2.3.1. El braille como herramienta clave para la inclusión y educación

El Braille es un método táctil de lectoescritura en uso desde el siglo XIX, representando un código lingüístico y un camino hacia la educación, igualdad e independencia, siendo imprescindible para que los individuos que poseen discapacidad visual tengan acceso al conocimiento, logrando una vida plena; el 4 de enero es el día del Braille, recordando el papel esencial que tiene en la protección de los DDHH [53].

Aunque este método facilita la inclusión de múltiples personas, su acceso no es equitativo en todos los sitios del mundo, por ende, las unidades educativas y

sociales han comenzado a cerrar las brechas que existen, promoviendo proyectos de innovación que incluyen al sistema Braille con tecnologías emergentes [53]. En Ecuador, la UTPL ha empezado a fomentar la inclusión de individuos discapacitados visualmente, desarrollando una impresora modular de sistema Braille a bajo costo, que se diseñó por alumnos de la carrera de Ingeniería en Electrónica y Telecomunicaciones en el año 2021, cuyo propósito se centró básicamente en la democratización del acceso a recursos impresos [53].

Por otro lado, la lectura y escritura son considerados elementos orales que derivan de la lengua, por ende, es fundamental obtener una correcta implantación del mismo; así mismo, cuando comenzó el auge de nuevas tecnologías, se creía que el sistema Braille iba a desaparecer, de forma que las personas jóvenes que poseen discapacidad visual perdían el interés de aprenderlo, ya que tenían lectores de pantalla para celulares y computadoras; sin embargo, el Braille siempre se ha ido desarrollando para ser bien utilizado en el área de las tecnologías accesibles [54].

El aprendizaje en Braille es una parte fundamental del proceso que lo enmarca, siendo un medio para llegar a un fin, que es la comunicación; motivo por el cual, los estudiantes, docentes y familias en general, pueden vivirlo de una manera más natural, como una ganancia, para poder comunicarse con ciudadanos no videntes [54].

2.3.2 Estructura del código Braille

El sistema Braille se centra en una estructura lógica que se conforma por una celda rectangular de 6 puntos en relieve, en conjunto con 2 columnas verticales de 3 puntos cada una, permitiendo la generación de hasta 64 combinaciones distintas, representando letras, números, signos de puntuación y caracteres especiales; la numeración de los puntos sigue un orden correspondiente: de arriba hacia abajo en columna izquierda (1, 2, 3) y en la derecha (4, 5, 6), que, a partir de dichas combinaciones, el braille es convertido en un lenguaje táctil accesible para individuos con ceguera [55].

El alfabeto en Braille se organiza progresivamente, donde las letras de la A hasta la J se representan con combinaciones básicas de los primeros puntos, luego para formar las letras de la K a la T, se agrega el punto 3 a las configuraciones previas;

finalmente, una excepción que destaca es la letra W, la cual no sigue el mismo patrón porque inicialmente no formaba parte del alfabeto francés en el momento que se creó el sistema Braille [55].

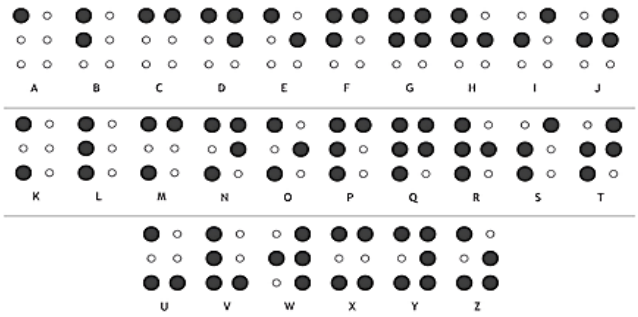


Figura 9: Alfabeto en Braille [55]

Con respecto a los números, el Braille reutiliza las configuraciones de las letras de la A hasta la J, pero antecedidas por el signo numérico, representando los puntos 3,4, 5 y 6, cuya señal indica que los caracteres siguientes deben ser interpretados como números; por ejemplo, la combinación del signo numérico seguido del símbolo de la letra A representa el 1; dicho esto, este sistema permite una codificación práctica y eficaz sin requerir símbolos completamente nuevos [2].

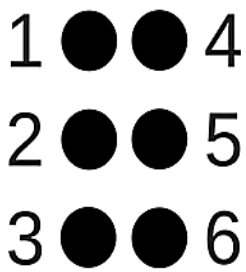


Figura 10: Números en Braille [55]

El código braille también posee varios signos especiales y de puntuación, fundamentales para la comprensión correcta de textos, entre los que se hallan el punto 2, 5 y 6, la coma, signo de interrogación, de exclamación, entre otros; así mismo, existen versiones del código Braille que se adaptan a diversos idiomas y requerimientos, como Braille de grado 1, grado 2 o en inglés americano, lo que demuestra una flexibilidad por parte del sistema en diferentes contextos de comunicación [2].

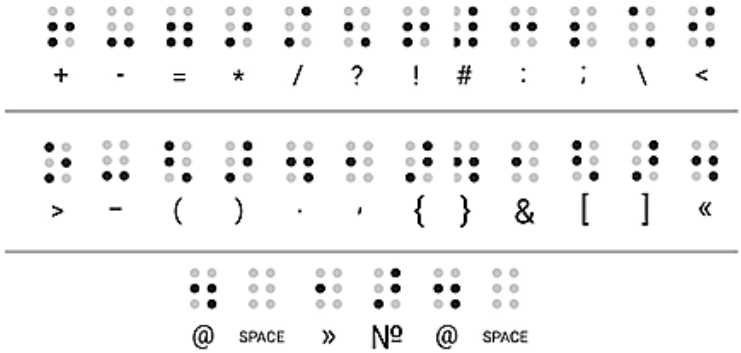


Figura 11: Signos especiales en Braille [55]

2.3.3. Importancia de un sistema de traducción de textos impresos a Braille

El código Braille es señalado como un recurso fundamental para individuos invidentes, al considerarlo como un medio de comunicación escrita; además, se señala que la instrucción comprende más que el dominio del código, es más un proceso social, intelectual y emocional; así mismo, parte de su intervención es poder construir espacios donde la escritura y lectura vaya más allá de simplemente codificar palabras en un símbolo o grafema, donde este sistema se elabora con el objetivo de brindar información básica a las personas invidentes utilizando estratégicamente ubicados puntos en relieve [56].

Tal y como se describe previamente, este sistema se estructura y adapta de forma fisiológica y psicológica al sentido del tacto, es decir a las células nerviosas en la punta de los dedos; los puntos en el relieve, cuya unidad mínima o signo generador es la celdilla o cajetín, están compuestos por 6 puntos, que se distribuyen en 2 matrices de 3 puntos [56].

En la actualidad, existen muchos jóvenes invidentes que no pueden integrarse al entorno escolar; sin embargo, debido a las políticas de inclusión que han establecido diversos países durante los últimos años, los gobiernos toman la decisión de que los individuos con esta clase de discapacidad deben incluirse al entorno educativo como los demás alumnos; no obstante, el tiempo para aprender a leer sistema Braille es aproximadamente de 4 meses para la versión no contraída [57]. La impresión de código Braille en papel con relieve por medio de impresoras especiales dificulta la

producción, ya que, los libros en este sistema poseen desventajas con respecto al costo y tamaño, de modo que el relieve de las letras ocupa mayor espacio que la escritura a tinta y papel comúnmente utilizada; por tanto, limita la accesibilidad e incrementa los costos para el aprendizaje, dejando excluida a una gran parte de la población [57]. Es sumamente importante desarrollar un prototipo que permita realizar la conversión rápida de cualquier texto en Braille sin requerir de un ordenador personal, mejorando el acceso al conocimiento para los individuos con discapacidad visual, reduciendo los costos de autonomía y aprendizaje [57].

2.3.4. Discapacidad visual, sistema Braille e inclusión educativa

Diversos docentes tienen desconocimiento acerca del método Braille, complicando la valoración y corrección de los proyectos escritos, además que los estudiantes tampoco conocen mucho con respecto a este tema y de la discapacidad, lo que resulta en el escaso apoyo o ayuda requerida como la repetición verbal de instrucciones, adaptaciones u otras; así mismo, los educadores especializados que existen en las instituciones educativas como soporte de ayuda a los alumnos no videntes poseen poco manejo del sistema Braille, causando poca retroalimentación, corrección o evaluación de cómo participa el educando dentro del proceso escolar [58].

Los educandos invidentes se encuentran en actividades pedagógicas en el contexto educativo donde no se ejecutan cambios necesarios para incluirlos satisfactoriamente; razón por la cual, los escolares con dicha condición presentan escasa habilidad en lectura, dificultando los procesos de abstracción de ideas principales y secundarias, reconocer clases de textos, entre otras actividades relevantes dentro del aula de clases [58].

Por otro lado, se destaca que en la actualidad el sistema braille puede ser leído por medio de pantallas electrónicas que convierten de manera instantánea la información en código Braille, con puntos más altos y pronunciados que el papel; además, es cómodo y simple, permitiendo realizar distintas combinaciones de dichos puntos, formando números, letras y signos de puntuación, considerando que este sistema es compatible con más de 100 idiomas en todo el mundo [59].

El primer paso para el proceso de lectura en braille, comienza con la percepción táctil del grafema que se percibe con los dedos y resulta analítico, distinguiéndose grafía por grafía; por lo cual, la velocidad de lectura se dificulta debido a la aproximación al significado del texto que se realiza por las rutas fonológicas, decodificando las palabras; es por esto que se requiere de la coordinación eficiente de los dedos y manos, siguiendo con la línea del texto, haciendo transiciones entre las mismas [59].

2.4. Requerimientos

2.4.1. Requerimientos Funcionales

Código	Descripción
RF – 01	El software permitirá cargar archivos en formato .docx y PDF para su posterior procesamiento.
RF – 02	El software deberá extraer el texto a partir de archivos cargados, empleando diversas librerías de Python.
RF – 03	El software deberá convertir el texto extraído a código Braille, siguiendo las reglas de codificación en el idioma español.
RF – 04	El software deberá realizar la detección automática de la impresora Braille conectada por medio del puerto serial (USB).
RF – 05	El software deberá enviar información convertida a sistema Braille a la impresora correctamente.
RF – 06	El software deberá contar con interfaces gráficas amigables y funcionales, compatible con tecnologías de lectura y navegación por teclado.
RF – 07	El software deberá generar archivos en formato Braille para poder distribuirse de manera electrónica.
RF – 08	El software deberá incluir pruebas unitarias y de integración, verificando su funcionamiento correcto.
RF – 09	El software deberá integrar un manual de usuario, explicando a detalle el funcionamiento del sistema.
RF – 10	El software permitirá la extensión a otros idiomas, comenzando con el español.

Tabla 2: Requerimientos Funcionales

2.4.2. Requerimientos no Funcionales

Código	Descripción
RNF – 01	El software debe ser fácil de utilizar, en especial, para las personas que poseen discapacidad visual y docentes.
RNF – 02	La conversión de texto a sistema Braille debe ser precisa, con un margen de error del 2%.
RNF – 03	El software debe realizar el procesamiento y conversión de archivos en un tiempo de 30 segundos aproximadamente, dependiendo del tamaño del documento.
RNF – 04	El software debe ser compatible con el sistema operativo Windows.
RNF – 05	El software debe diseñarse para incluir actualizaciones posteriores, como la integración de idiomas nuevos o funciones.
RNF – 06	El entorno visual debe ser accesible para las personas con discapacidad visual.
RNF – 07	El código debe documentarse y estructurarse, facilitando su actualización y mantenimiento.
RNF – 08	El software debe garantizar la seguridad e integridad de los datos, evitando la pérdida de información.
RNF – 09	El software debe ser fácil de instalar y ejecutar en diversos entornos, sin necesitar configuraciones complicadas.
RNF – 10	El software debe funcionar de forma establece, sin presentar fallas durante su utilización.
RNF – 11	El software debe contar con consumo de recursos optimizado, evitando un uso excesivo de la memoria o procesamiento.

Tabla 3: Requerimientos no Funcionales

2.5. Componentes de la Propuesta

2.5.1. Diagrama de proceso del sistema

El proceso de uso del software traductor Braille comienza cuando el usuario selecciona la opción de cargar archivo, abriendo el explorador de documentos para escoger un archivo en formato Pdf o Docx, donde una vez cargado se verifica que el documento haya sido subido de manera correcta, evidenciando el nombre y extensión en la pantalla. El usuario optará por traducir todo el archivo o seleccionar un rango de páginas específico, para luego dar clic en el botón convertir y confirmar la acción cuando se le solicite; si el usuario cancela la conversión, regresará a la pantalla de selección del documento.

Si la conversión es correcta, el software desplegará dos ventanas, una que mostrará el texto original y otra que evidenciará el código binario o Braille traducido; a continuación, el usuario decide si imprimir todo el archivo traducido presionando el botón de imprimir; de lo contrario, finaliza el proceso sin la impresión.

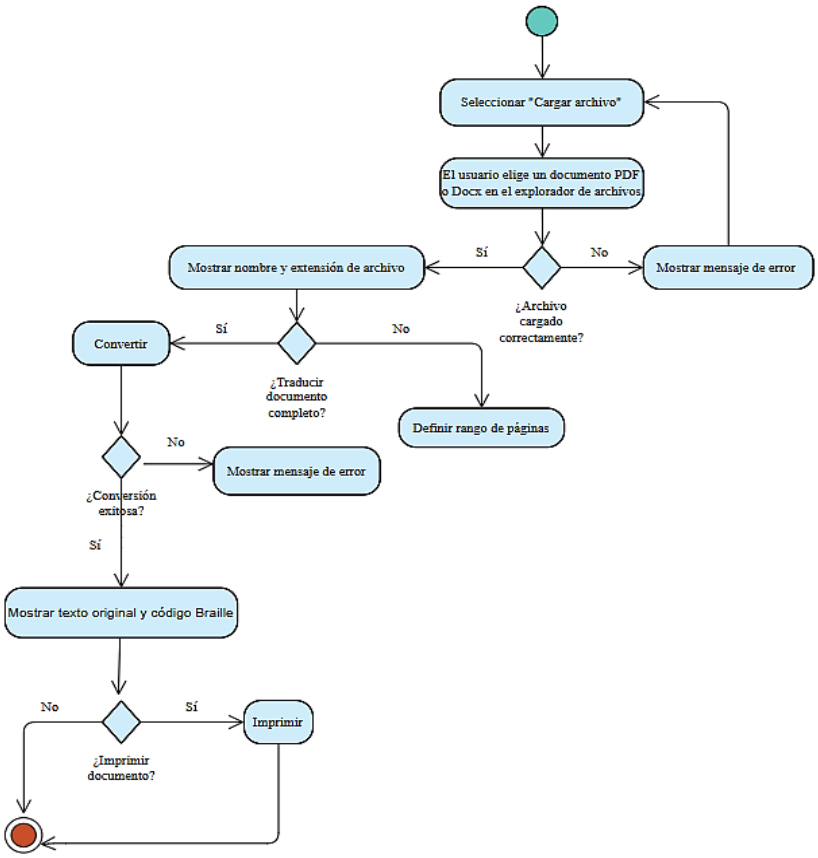


Figura 12: Diagrama de proceso del software traductor Braille

2.5.2. Fase 1: Diseño del sistema Braille

En esta fase inicial de la propuesta se diseñó de forma estructural el sistema de conversión de texto en lenguaje natural a código Braille, con el fin de garantizar que el software sea accesible, funcional y fácil de utilizar por personas con discapacidad visual; siendo una etapa esencial para determinar los principios operativos y técnicos que guiarán el desarrollo posterior. A continuación, se presentan las actividades más relevantes llevadas a cabo en la fase:

Definición de los formatos de entrada

Uno de los pasos iniciales para el diseño del sistema fue establecer que dos tipos de archivos ampliamente utilizados en el ambiente educativo y profesional se pudiesen procesar: Docx y Pdf. En este caso, se tuvo en cuenta la disponibilidad de tales documentos y que era necesario para el usuario poder cargar archivos sin necesidad de tener restricciones.

La manipulación de los archivos PDF fue realizada por medio de la biblioteca fitz o PyMuPDF. Esta biblioteca proporciona una interfaz adecuada y eficiente para extraer y leer documentos PDF, puesto que permite la manipulación de documentos que poseen una cierta complejidad estructural, pudiendo no solo obtener el texto, sino también ordenarlo.

De igual manera, para la manipulación de los archivos Docx se utilizó la librería Python-docx que permitía extraer contenido básico de un Word, sin alterar el texto y formato básico, cuya capacidad para leer documentos de las dos partes aumenta su aplicabilidad en ámbitos educativos; además, ambas bibliotecas aseguran un tratamiento consistente del contenido, evitando pérdidas o alteraciones durante el proceso de extracción.

Diseño modular del sistema

La arquitectura del software fue diseñada con un enfoque modular con el fin de proporcionar flexibilidad y crecimiento escalable, así como facilidad de mantenimiento. Con esta estructura, se pueden incorporar nuevas funciones o modificar las ya existentes sin comprometer la estabilidad general del sistema. Los módulos principales que componen este software son los siguientes:

- **FileConverter:** Se encarga del procesamiento de archivos de entrada, como PDF y DOCX, extrayendo el texto y preparándolo para convertirlo en formato Braille. La actividad principal de este módulo es interpretar de manera correcta los distintos tipos de documentos y extraer de ellos el contenido textual relevante, dejando a un lado las imágenes o aspectos no procesables.
- **ArduinoManager:** Administra la comunicación con la placa Arduino por medio de un puerto serial; una vez que el texto se convirtió en código Braille, el ArduinoManager es responsable de enviar las secuencias binarias al dispositivo, transformándolo en representaciones físicas en relieve, empleando una impresora Braille o actuador.
- **ImageLoader y MessageHandler:** Se encargan de la carga de imágenes en la interfaz gráfica y del manejo de mensajes o errores dentro del software, puesto que, ImageLoader es responsable de cargar íconos y otros aspectos gráficos; mientras que, MessageHandler maneja las notificaciones y mensajes de error para los usuarios, lo que asegura que se le indique de manera adecuada en caso de cualquier falla durante el proceso.

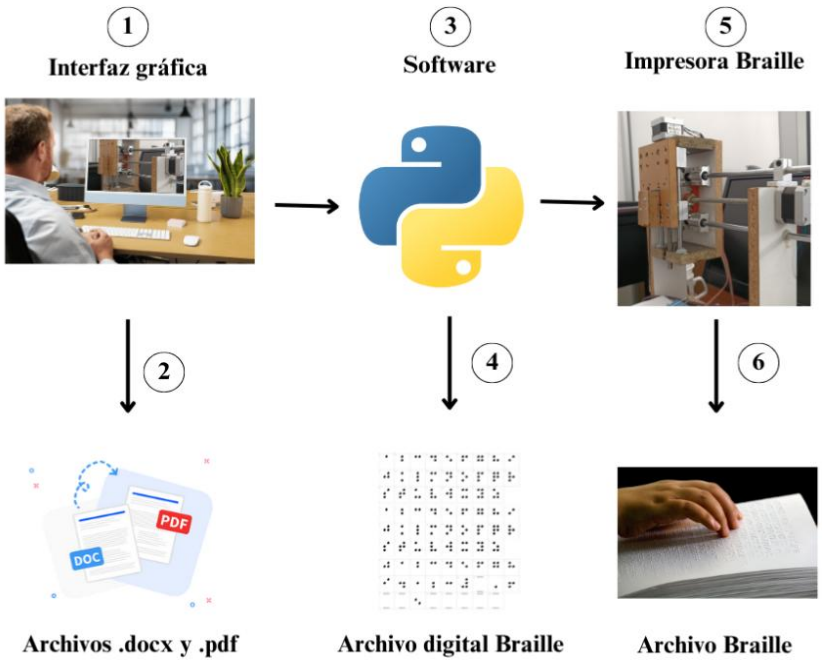


Figura 13: Esquema general del funcionamiento del sistema

El esquema evidencia el flujo general del funcionamiento del sistema, comenzando con la carga de archivos de entrada en formato Docx o Pdf; por medio del módulo FileConverter, se extrae el contenido textual que posteriormente se procesa para su conversión a código Braille a través de la tabla de equivalencias establecida. El texto convertido se transmite al módulo de ArduinoManager, estableciendo la comunicación serial con el dispositivo Arduino. Se concluye con el Arduino que se encarga de enviar instrucciones a la impresora Braille o actuador, permitiendo materializar de forma física el texto en código Braille. Así mismo, los módulos MessageHandler e ImageLoader asisten en la administración de recursos gráficos y mensajes del sistema, lo que asegura una interacción amigable y fluida para los usuarios.

Flujo de proceso del sistema de impresión de código Braille

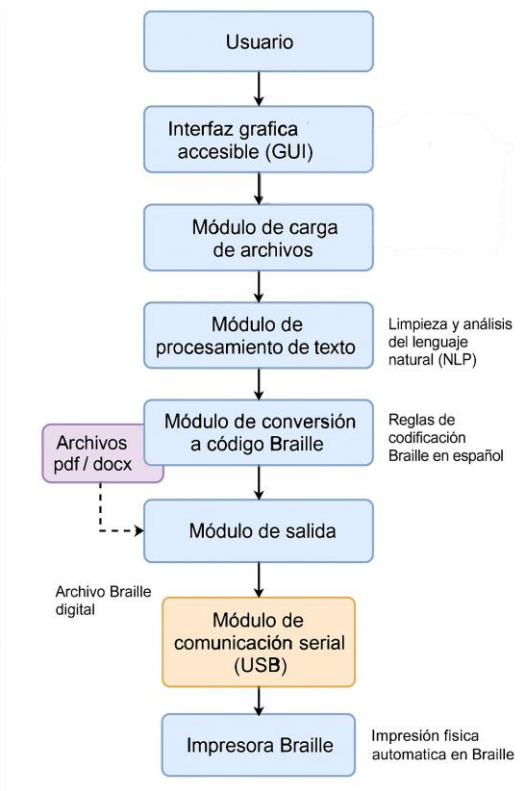


Figura 14: Flujo de proceso del sistema de impresión de código Braille

En la Figura 13, se evidencia los diferentes elementos que conforman la solución desarrollada, comenzando con el usuario quien interactúa con el sistema por medio de la interfaz gráfica de usuario accesible a través del lector de pantalla y

navegación por teclado; desde la GUI, el usuario puede emplear el módulo de carga de archivos que permite la importación de documentos en formato .docx o .pdf.

Luego de la carga de documentos, el texto se envía al módulo de procesamiento de texto, encargándose de extraer y limpiar el contenido usando bibliotecas especializadas de Python, que, posteriormente, dirige el archivo al módulo de conversión Braille, aplicando reglas de codificación en idioma español para la traducción del texto plano.

Una vez que se convirtió el texto, se lo transfiere al módulo de salida que administra la comunicación del puerto serial hacia el dispositivo de impresión; concluyendo con la impresora Braille que recibe los datos y genera el documento en Braille, siendo este, en formato físico.

Esquema de la arquitectura de comunicación del sistema Braille

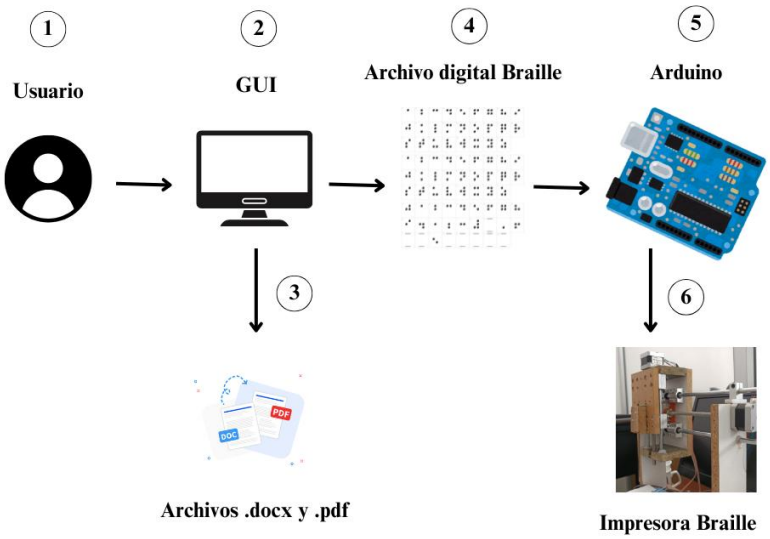


Figura 15: Esquema de la arquitectura de comunicación del sistema Braille

La Figura 14, evidencia la arquitectura de comunicación del sistema de traducción Braille, cuyo proceso comienza con el usuario que, por medio de la interfaz gráfica, carga los documentos en formato PDF o DOCX; posteriormente, la GUI procesa el contenido de los archivos y los convierte en documento digital con codificación binaria. Luego, el archivo se envía al microcontrolador Arduino, el cual interpreta

el código Braille y lo transmite a una impresora física de Braille, que culmina realizando la impresión del texto adaptado para individuos con discapacidad visual.

Establecimiento del sistema de codificación Braille

El núcleo del sistema es la conversión de texto en lenguaje natural a su representación respectiva en código Braille, siendo necesario para esto, establecer una tabla de equivalencias que asociará cada caracter de idioma español con su equivalente en Braille, incorporando las letras del alfabeto, signos de puntuación, números y caracteres especiales, lo que asegura que la conversión sea precisa y completa.

El sistema Braille se centra en un patrón de puntos dispuestos en celdas de 2x3; por esto, el sistema desarrollado traducirá cada caracter del texto extraído en su secuencia respectiva de puntos Braille, adaptándose para cumplir con las normas del idioma español, considerando caracteres como la ñ, acentos y otros signos de puntuación propios de la lengua.

De la misma manera, se establecieron medidas específicas para el control de espacios y saltos de línea para que el proceso de traducción se realizara de forma estructural y brindando la posibilidad a la persona discapacitada a que pueda leer de manera fluida y coherente el contenido.

Elección de bibliotecas y herramientas de desarrollo

Para lograr la eficiencia y la robustez del sistema propuesto, se optó por determinadas herramientas y bibliotecas en Python que minimizaron el esfuerzo durante la implementación:

- **Fitz:** Esta biblioteca se seleccionó por la alta exactitud que brinda en la extracción de textos de archivos en PDF, ya que Fitz es capaz de manejar textos que tienen escritos en cursivas, negritas y hasta subrayados, favoreciendo mucho en la conversión del texto original.
- **Python-docx:** Este tipo de archivos puede leerse y producirse usando este tipo de librería, porque su empleo es habitual tanto en el ámbito docente como en el profesional, por lo que permitir su utilización en el software, brindando más libertad en el manejo de archivos.

- **Pyserial:** Para las comunicaciones con Arduino, se utilizó la biblioteca Pyserial, que permite la conexión del sistema de puerto serie a los microcontroladores y a la computadora, lo cual admite el envío y recepción de las secuencias en Braille generadas para su posterior visualización en el dispositivo físico.
- **Tkinter:** Con relación a la UIX, se empleó este software que se utiliza de manera amplia en aplicaciones desarrolladas con lenguaje Python, puesto a su facilidad de creación de módulos dinámicos y autónomos para los usuarios.

Comunicaciones con Arduino

Se pone en marcha la conversión en tiempo real con el microcontrolador Arduino junto con una conexión pyserial y un canal habilitado en la computadora con el dispositivo, siendo posible enviar información a la impresora conectada, decodificando el texto en números binarios. Así mismo, se desarrollaron protocolos de sincronización en Arduino y el sistema, garantizando una transmisión de datos confiable, lo que resuelve la transferencia de datos sin fallos.

Integración con la impresora Braille

Se plantea la integración del software con una impresora Braille, diseñando cómo el sistema imprime los archivos transformados, abordando el diseño de una secuencia de impresión que cumpla con todos los requerimientos para lograr el cumplimiento total; Además, se consideró la configuración automática de parámetros como intensidad de relieve y formato de página.

2.5.3. Fase 2: Desarrollo del traductor de texto a sistema de impresión Braille

En esta fase se pone en marcha cada paso de la propuesta, asegurándose de que la función principal del software, que es la traducción automática de textos en lenguaje natural al Braille, se ejecute correctamente, teniendo en cuenta tanto la conversión como las reglas de la lengua. Para ello se diseñaron varios módulos en Python que se encargaron de la parte de carga de archivo, extracción de texto, manipulación de escritos y transformación a Braille, los cuales se detallan en el Anexo del Manual de codificación (Ver Anexo 3).

Nombre función	Parámetros	Descripción	Código fuente
upload_files()	self.	Se dedica a la incorporación de los documentos al sistema de manera lógica, mediante la obtención del archivo a través del método FileManager.get_file.path(). Si se seleccionó un archivo, se almacena su ruta en self.path, mostrando una barra de progreso y actualizando el botón de conversión para evidenciar que el documento está siendo procesado. Posteriormente, se extrae el nombre y la extensión del archivo empleando PathBuilder.extract_file_name y PathBuilder.extract_file_extension. Luego, se oculta la barra de progreso, mostrando el ícono que corresponde a la extensión del documento y se llama al método process_file_extension para administrar las acciones correspondientes según la clase del archivo. Se crea un hilo utilizando threading.Thread para la ejecución de la función load_file, encargándose del procesamiento del archivo sin afectar la interactividad de la interfaz, pasando como argumentos la ruta del documento y su extensión, el cual se ejecuta de forma asincrónica, permitiendo que el usuario siga interactuando con la app mientras el archivo es procesado en segundo plano.	Ver Anexo 3. Figura 1

load_file()	self, file_path, file_extension.	La función load_file se encarga de procesar el contenido del documento según su extensión; si el archivo es PDF, emplea la librería fitz para abrir y leer el contenido, extrayendo el texto de cada página y actualizando la barra de progreso conforme avance la lectura. Si el archivo es docx, se usa la librería docx para el procesamiento de cada párrafo, a la vez que se actualiza la barra de progreso de forma semejante. En el caso de otra clase de archivos, se lee el documento como un bloque de bytes, procesando fragmentos de 4096 bytes, para luego completar el procesamiento y actualizando el ícono que corresponde a la extensión del documento a través del método process_file_extension.	Ver Anexo 3. Figura 2
extract_text_from_pdf	file_path.	Está diseñada para extraer texto a partir de archivos con formato PDF por medio de la librería PyPDF2, donde se abre el archivo, recorriendo todas sus páginas y posteriormente, extrayendo el texto de cada una empleando el método extract_text(). Luego, devuelve todo el escrito que se extrajo del documento y si se halla algún inconveniente, como un archivo inaccesible o corrupto, se genera el mensaje respectivo.	Ver Anexo 3. Figura 3
extract_text_from_word	file_path.	Es la función encargada de realizar la extracción del texto de documentos docx empleando la librería Python-docx, donde se abre el archivo desde la ruta brindada, separando cada párrafo con un salto de línea; en caso de	Ver Anexo 3. Figura 4

		haber algún error, como inconvenientes al acceder al documento, se muestra un mensaje.	
extract_text_from_pdf_range	file_path, start, end.	Es una variante de la función anterior, donde se permite especificar un rango de páginas, desde las cuales se puede extraer el texto; cuyos parámetros Start y end definen las páginas del documento PDF a procesar y la función transforma el texto de esas páginas en conjunto; si hay una falla durante la extracción, también se manejan los mensajes de errores.	Ver Anexo 3. Figura 5
create_convert_button	self, parent.	Se encarga de crear y configurar el botón "Convertir" en la interfaz gráfica, empleando el método Button para diseño del botón, estableciendo el texto y asignando la función convert_files como el comando a ejecutar cuando sea presionado.	Ver Anexo 3. Figura 6
convert_files	self.	Comienza el proceso de conversión de archivos a un formato binario para Braille, verificando que exista un archivo cargado, para luego solicitar una confirmación al usuario antes de continuar. Si se confirma, detecta la clase de documento y extrae el texto completo o un rango específico de páginas. Una vez que se obtiene el escrito, llama a la función handle_text_conversion con el fin de realizar la conversión.	Ver Anexo 3. Figura 7

handle_text_conversion	self, text.	Toma el texto extraído a partir del documento y lo convierte en una representación binaria usando BytesConverter.convert_to_bytes, adaptándose al contenido para su posterior traducción al código Braille. Así mismo, se muestra el texto original como su versión transformada en una ventana de resultados, informando al usuario que la conversión se realizó satisfactoriamente. Además, marca la conversión como completa y en caso de trabajarse con un rango de páginas, se almacenan los datos de inicio y fin.	Ver Anexo 3. Figura 8
handle_conversion_error	self, error.	La función handle_conversion_error posee como propósito manejar cualquier error que ocurra durante el proceso de conversión de documentos, recibiendo el objeto de error capturado y empleando MessageHandler, que despliega una ventana de error evidenciando sobre el problema encontrado, lo que asegura una comunicación correcta durante la ejecución del software.	Ver Anexo 3. Figura 9
convert_to_bytes	textToConvert.	Esta función se encarga de realizar la conversión del texto extraído de un archivo en una representación binaria compatible con el sistema Braille, recorriendo cada carácter del texto recibido y aplica reglas específicas, si se encuentra caracteres de escape los conserva tal cual, detectando letras	Ver Anexo 3. Figura 10

		mayúsculas añade el prefijo de mayúscula definido en el diccionario braille_translation y las convierte en minúsculas.	
Diccionario de caracteres	-	El diccionario define la correspondencia entre los caracteres numéricos, alfabéticos y de puntuación del lenguaje natural, así como su representación en código binario Braille. Cada número, letra o símbolo especial se relaciona a una secuencia de bits específica, permitiendo que el texto procesado se traduzca de manera precisa al sistema Braille.	Ver Anexo 3. Figura 11
create_print_button	self, parent.	Se encarga de gestionar la funcionalidad de impresión del proyecto por medio del método de impresión, creando un botón en la interfaz gráfica que al presionarse ejecuta la función print_files.	Ver Anexo 3. Figura 12
print_files	self.	Verifica que previamente se haya realizado la conversión de texto a bits; una vez se confirma se abre el documento que contiene el texto en formato binario, se lee su contenido y se convierte a instrucciones en G – code, siendo un lenguaje que Arduino comprende para controlar dispositivos como impresoras o motores. Concluyendo con este código generado que se envía al microcontrolador para que ejecute la impresión del patrón Braille respectivo.	Ver Anexo 3. Figura 13

Tabla 4: Fase 2

2.5.4. Fase 3: Resultados del software controlador

Prueba	Escenario	Procedimiento	Resultado	Descripción
Verificación de conectividad física	Corroborar las conexiones de alimentación y comunicación entre el Arduino, CNC Shield y PC.	1. Conectar la impresora Braille a la PC. 2. Verificar en el administrador de dispositivos.	El sistema se reconoce por el administrador de dispositivo en el puerto serial correspondiente, en este caso, COM5.	Ver Anexo 4
Verificación lógica (Software)	Conectar la impresora por medio del puerto serial al software de traducción Braille.	1. Se declaró el puerto serial en el archivo Arduino.manager.py, configurando el puerto identificado en el administrador de dispositivos para establecer la comunicación con el Arduino. 2. Se ejecutó el archivo principal main.py, que incluye los módulos del sistema y permite cargar archivos .docx o .pdf, convirtiendo su contenido a	El sistema permite realizar las diversas acciones, como subir el archivo, convertirlo e imprimir el documento.	Ver Anexo 5

		braille por medio del diccionario binario y generando las instrucciones de impresión para el prototipo.		
Prueba de impresión básica	Imprimir un texto corto, verificando la funcionalidad óptima.	1. Subir un archivo con un texto corto. 2. Convertir el archivo. 3. Realizar la impresión del "hola mundo" en Braille.	Se imprimió el archivo de texto corto, verificando la correcta marcación y relieve de los puntos.	Ver Anexo 6
Prueba de consistencia	Imprimir el mismo archivo varias veces.	1. Subir un archivo con un texto. 2. Convertir el archivo 3. Realizar la impresión varias veces.	Se imprimió el archivo varias ocasiones, comprobando la consistencia. Se pudo verificar que las copias resultantes fueron iguales y sin irregularidades.	Ver Anexo 7
Prueba de límites y errores físicos	Ejecutar una prueba de papel mal colocado.	1. Colocar mal el papel en la impresora. 2. Identificar las limitaciones del tamaño máximo de página.	El sistema reaccionó de manera apropiada ante dichos casos.	Ver Anexo 8

Tiempo de impresión de palabras	Verificar el tiempo de impresión de palabras x minutos	1. Imprimir el archivo en código Braille. 2. Contar el tiempo que se toma en imprimir el documento, dependiendo del número de palabras.	El sistema está apto para imprimir 25 palabras por minuto.	Ver Anexo 8

Tabla 5: Pruebas del software controlador

Las pruebas realizadas confirmaron que el software braille funciona de forma óptima al convertir texto en código braille, identificando los archivos que se cargan por medio del software y realizando la conversión sin problemas. En este sentido, los archivos impresos fueron legibles al tacto, con una alineación idónea y con los puntos correctamente establecidos. Además, se considera que el sistema cumple con el rol principal de imprimir en código braille en función de archivos digitales, demostrando funcionalidad y confiabilidad.

2.5.5. Fase 4: Pruebas de software

En esta fase se realizan las pruebas unitarias, de integración, de precisión y usabilidad. A continuación, se describe cada una de ellas:

Pruebas unitarias

Prueba	Detalle	Método	Resultado	Evidencia
Conversión a Braille	Comprobar que el sistema transforma el texto a código Braille satisfactoriamente.	1. Utilizar un texto corto. 2. Ejecutar el proceso de conversión. 3. Comprobar que la salida corresponde en el Braille correcto.	El texto se convierte en código Braille sin errores.	Ver Anexo 9. Figura 1
Creación de un archivo	Verificar que el archivo Braille se almacena correctamente en el sistema.	Se corrobora que el archivo generado posee la extensión y los formatos apropiados.	El archivo se almacena correctamente en la ubicación especificada.	Ver Anexo 9. Figura 2
Compatibilidad de caracteres	Comprobar que caracteres especiales son traducidos correctamente a Braille.	1. Introducir caracteres con tildes. 2. Escribir caracteres especiales, como la “ñ”.	Los caracteres especiales deben ser convertirse correctamente a código Braille.	Ver Anexo 9. Figura 3

		3. Corroborar la salida en Braille.		

Tabla 6: Pruebas unitarias

Pruebas de integración

Prueba	Detalle	Método	Resultado	Evidencia
Conversión y archivo	Comprobar que la conversión y creación de archivo funcionen de forma correcta juntos.	1. Escribir un texto. 2. Convertirlo en Braille. 3. Almacenar el archivo generado. 4. Revisar si el documento está correcto.	La conversión y creación del archivo funcionan sin inconvenientes.	Ver Anexo 10. Figura 1
Generación de código binario en consola	Verificar que el documento se convierta en código binario por consola.	1. Introducir un archivo. 2. Convertirlo a código binario. 3. Revisar la consola.	El código binario generado se evidencia en consola.	Ver Anexo 10. Figura 2

Tabla 7: Pruebas de integración

Precisión de la traducción

Prueba	Detalle	Método	Resultado	Evidencia
Exactitud de caracteres	Corroborar que cada carácter en el texto se convierte de manera correcta a Braille.	1. Introducir un texto con caracteres comunes. 2. Verifica si el Braille generado es correcto.	El Braille generado debe coincidir con los estándares internacionales.	Ver Anexo 11. Figura 1
Consistencia de traducción	Comprobar que la misma entrada siempre se traduzca de la misma manera.	1. Introducir el mismo texto múltiples veces. 2. Corroborar que el Braille sea consistente.	La traducción debe ser siempre consistente sin fallas.	Ver Anexo 11. Figura 2

Tabla 8: Precisión de la traducción

Usabilidad del sistema

Prueba	Detalle	Método	Resultado	Evidencia
Interfaz de usuario	Verificar que la interfaz del sistema sea fácil de utilizar y clara.	1. Interactuar con la interfaz del sistema.	La interfaz debe ser navegable e intuitiva.	Ver Anexo 12. Figura 1

		2. Evaluar si los botones y comandos con claros.		
Manejo de errores	Corroborar que el sistema muestre mensajes de error claros, cuando algo esté fallando.	1. Ingresar un archivo incorrecto. 2. Verificar si el sistema muestra una alerta.	Los mensajes de error deben ser útiles y claros.	Ver Anexo 12. Figura 2

Tabla 9: Usabilidad del sistema

Adaptabilidad en distintos entornos sociales y educativos

Prueba	Detalle	Método	Resultado	Evidencia
Adaptación a textos educativos	Verificar que el sistema maneje textos técnicos y científicos	1. Introducir un texto académico o técnico. 2. Validar si la traducción es correcta.	El sistema debe manejar textos con términos técnicos de forma precisa.	Ver Anexo 13. Figura 1
Adaptación a textos sociales	Corroborar que el software maneje textos informales o sociales fin fallas.	1. Introducir un texto informal. 2. Comprobar que la traducción sea correcta.	El software debe manejar texto social con precisión.	Ver Anexo 13. Figura 2

Tabla 10: Adaptabilidad a diversos entornos

CONCLUSIONES

- Se diseñó el software traductor de lenguaje natural a código Braille empleando Python, integrando una interfaz gráfica amigable que contiene botones y opciones de control que facilitan la interacción con los usuarios. Dicho sistema permite la carga de archivos de texto en lenguaje natural, para luego procesarlos y convertirlos en código binario y posteriormente a Braille, generando una salida que sea compatible con la impresora; la misma, imprimió correctamente los puntos en relieve, asegurando exactitud y legibilidad del contenido que se tradujo.

- Se implementaron bibliotecas Python – docx y Fitz para la carga y lectura de archivos en formato DOCX y PDF, asegurando una extracción que sea fiel al contenido; luego de presionar el botón de convertir, el diccionario de datos transforma el texto natural a código Braille en menos de 5 segundos, donde el sistema permite visualizar los caracteres Braille en una pestaña, así como el texto original y el código binario, lo que garantiza un proceso de conversión eficaz.

- El software pudo establecer comunicación con la impresora Braille por medio del puerto serial (USB), empleando el módulo ArduinoManager, para gestionar la transmisión de datos y asegurar que cada carácter en código Braille sea enviado de manera ordenada y sin pérdida de datos. La incorporación de protocolos de verificación y sincronización garantiza que la impresora reciba las instrucciones solamente cuando el equipo se encuentre listo, previniendo fallos de impresión y confirmando la materialización correcta del texto en Braille.

- Se ejecutó la evaluación del sistema por medio de pruebas unitarias y de integración, confirmando que los módulos del software funcionan de manera apropiada y se comunican de forma estable con la impresora Braille a través del puerto serial. Es así que, la conversión de archivos a código Braille se ejecutó de manera precisa, y las pruebas de impresión

evidenciaron que los puntos en relieve se reproducen de forma consistente, inclusive al imprimir varias veces el mismo documento; además, el sistema maneja apropiadamente condiciones adversas, como archivos de tamaño máximo o papel mal colocado, garantizando un proceso accesible y confiable para los usuarios.

RECOMENDACIONES

- Se recomienda implementar opciones avanzadas en el sistema Braille, con el fin de fortalecer la accesibilidad del mismo, como integrar el modo oscuro en el software para reducir la fatiga visual, compatibilidad con diversos dispositivos Braille e incorporar la capacidad de controlar el sistema a través de comandos de voz y gestos; esto permitirá que el software se adapte a los requerimientos de los usuarios, logrando una experiencia más eficiente en su uso.

- Se sugiere añadir la opción de detallar y describir imágenes dentro de los archivos, de forma que los usuarios puedan incorporar texto descriptivo para el contenido gráfico, ampliando la capacidad del software, lo que permitirá que tanto el texto natural como las descripciones de imágenes se conviertan a Braille, ofreciendo una experiencia accesible y completa para los individuos con discapacidad visual.

- Es recomendable incluir un sistema de evaluación y supervisión que actúe en tiempo real con respecto a la conexión de la impresora Braille, incorporando la verificación de los puertos, verificación de errores y mensajes automáticos para los usuarios administradores en caso de algún fallo o desconexión con el software, facilitando la identificación y resolución ágil de problemas operativos, lo que garantiza que el proceso de impresión no se interrumpa.

- Sería relevante establecer un programa constante en cuanto a pruebas o evaluaciones que integre la ejecución de casos fundamentales para todas las funciones de los módulos, con el propósito de recopilar, analizar y responder

a la retroalimentación de los usuarios finales, permitiendo detectar de forma temprana cualquier falla o área de mejora, preservando la calidad del software de manera continua.

BIBLIOGRAFÍA

[1] D. Orna, «Implementación de un sistema electrónico convertidor de texto a lenguaje,» Riobamba, 2024.

[2] P. Cabrera y J. Mejía, «DESARROLLO DE UN PROTOTIPO PARA APRENDIZAJE DE LENGUAJE BRAILLE ASISTIDO POR AUDIO,» Quito, 2021.

[3] INEC, «Estadísticas de Discapacidad,» 2023. [En línea]. Available: https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/.

[4] Infoescuelas, «Infoescuelas,» [En línea]. Available: https://www.infoescuelas.com/ecuador/santa-elena/escuela-de-educacion-basica-manuela-espejo-en-la-libertad/. [Último acceso: 19 Agosto 2024].

[5] M. Álvarez y R. Sumba, «Braillet: dispositivo tecnológico para el aprendizaje del sistema Braille dirigido a niños con discapacidad visual,» *Educación en contexto,* vol. 6, nº 1, p. 31, 2020.

[6] M. Hernández, «Validación de un dispositivo para apoyar el aprendizaje y la enseñanza del sistema Braille,» Bogotá, 2020.

[7] J. Sarango, «Aprendizaje del sistema de lectura y escritura Braille basado en las Tecnologías de la información y comunicación,» Loja, 2021.

[8] F. López, 4 7 2024. [En línea]. Available: https://repositorio.upse.edu.ec/bitstream/46000/11586/4/UPSE-TET-2024-0008.pdf.

[9] Facsistel, «UPSE,» 2024. [En línea]. Available: https://facsistel.upse.edu.ec/index.php?option=com_content&view=article&id=58&Item.

[10] S. Alejandro González, «La importancia del braille para las personas con discapacidad visual,» INCI, 15 Noviembre 2019. [En línea]. Available: https://www.inci.gov.co/blog/la-importancia-del-braille-para-las-personas-con-discapacidad-visual.

[11] Secretaría Nacional de Planificación, «Plan de desarrollo para el nuevo Ecuador,» 2024.

[12] Centro Virtual Cervantes, «Metodología cuantitativa,» 2022. [En línea]. Available: https://cvc.cervantes.es/ensenanza/biblioteca_ele/diccio_ele/diccionario/metodologiacuantitativa.htm.

[13] A. León, J. Acosta y R. Díaz, «Aplicación de la metodología incremental en el desarrollo de sistemas de información,» *Revista Universidad y Sociedad,* vol. 13, nº 5, 2021.

[14] Comunicando UPSE, «Santa Elena cuenta con más de 10 mil personas con discapacidad,» 20 1 2022. [En línea]. Available: https://comunicandoupse.wixsite.com/comunicandoupseec/post/personas-con-discpacidad-en-la-provincia-de-santa-elena.

[15] Puntodis, «Discapacidad visual,» 2021. [En línea]. Available: https://puntodis.com/featured_item/discapacidad-visual/#:~:text=Dificultad%20para%20leer%20textos%20que,de%20orientaci%C3%B3n%20en%20nuevos%20espacios..

[16] M. Aguilera, «Discapacidad visual en el aula: desafíos y adaptaciones,» 4 12 2023. [En línea]. Available: https://www.rededuca.net/blog/educacion-y-docencia/discapacidad-visual.

[17] CONSTITUCIÓN DE LA REPÚBLICA DEL ECUADOR, «CONSTITUCIÓN DE LA REPÚBLICA DEL ECUADOR,» 25 1 2021. [En línea]. Available: https://www.defensa.gob.ec/wp-content/uploads/downloads/2021/02/Constitucion-de-la-Republica-del-Ecuador_act_ene-2021.pdf.

[18] Asamblea Nacional del Ecuador, «Ley orgánica de discapacidades,» 2021.

[19] Naciones Unidas, «Objetivos de Desarrollo Sostenible,» 24 5 2022. [En línea]. Available: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/.

[20] Canal Institucional TV, «¿Qué es el sistema Braille y cómo funciona?,» 4 1 2021. [En línea]. Available: https://www.canalinstitucional.tv/te-interesa/que-es-el-sistema-braille-y-como-funciona.

[21] D. Labrador, «Conociendo el Braille: Definición, Historia y Alfabeto,» 29 9 2024. [En línea]. Available: https://www.discapnet.es/accesibilidad/accesibilidad-en-la-comunicacion/braille.

[22] National Geographic, «Louis Braille, el inventor de la lectura táctil,» 2025. [En línea]. Available: https://historia.nationalgeographic.com.es/a/louis-braille-inventor-lectura-tactil_17545.

[23] Once, «El Braille cumple 200 años de evolución: del punzón a la tecnología,» 2 1 2025. [En línea]. Available: https://www.once.es/noticias/el-braille-cumple-200-anos-de-evolucion-del-punzon-a-la-tecnologia.

[24] Hable, «Alfabeto y números en Braille,» 15 5 2024. [En línea]. Available: https://www.iamhable.com/es-am/blogs/articulo/braille-alphabet-numbers?srsltid=AfmBOorZMRycv6d5JTRWbqvLI8kzxAljgTyiubxRkiOFEeiQoATV9ktp.

[25] AWS, «¿En qué consiste la traducción automática?,» 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/machine-translation/.

[26] Trados, «¿En qué consiste la traducción automática?,» 2022. [En línea]. Available: https://www.trados.com/es/learning/topic/machine-translation/.

[27] GoDaddy, «¿Python? Características, usos y ventajas,» 27 Febrero 2024. [En línea]. Available: https://www.godaddy.com/resources/latam/desarrollo/python-que-es.

[28] OMS, «Ceguera y discapacidad visual,» 10 8 2023. [En línea]. Available: https://www.who.int/es/news-room/fact-sheets/detail/blindness-and-visual-impairment.

[29] OPS, «Salud visual,» 2023. [En línea]. Available: https://www.paho.org/es/temas/salud-visual.

[30] Barraquer, «¿Qué es la agudeza visual?,» 2 2 2021. [En línea]. Available: https://www.barraquer.com/noticias/agudeza-visual#:~:text=La%20agudeza%20visual%20es%20la,cercanos%20est%C3%A1n%2C%20efectivamente%2C%20separados..

[31] MedlinePlus, «Campo visual,» 12 11 2020. [En línea]. Available: https://medlineplus.gov/spanish/ency/article/003879.htm.

[32] E. Arranz, «Cuáles son los tipos de discapacidad visual,» 22 5 2023. [En línea]. Available: https://fundacionadecco.org/blog/cuales-son-los-tipos-de-discapacidad-visual/.

[33] Orcam, «Orcam,» 2024. [En línea]. Available: https://www.orcam.com/es-es/blog/la-tecnologia-al-servicio-de-las-personas-con-discapacidad-visual. [Último acceso: 24 Agosto 2024].

[34] Fundación Adecco, «Fundación Adecco,» 2023. [En línea]. Available: https://fundacionadecco.org/blog-diversidad-inclusion/las-nuevas-

tecnologias-al-servicio-de-la-discapacidad/. [Último acceso: 25 Agosto 2024].

[35] Roland, «Impresión braille,» 2025. [En línea]. Available: https://www.rolanddg.eu/es-es/aplicaciones/braille-rotulos-tactiles.

[36] S. Kravitz, «Impresión Braille: Cómo funciona,» 2 11 2022. [En línea]. Available: https://ironmarkusa.com/braille-printing-how-it-works/.

[37] F. López, «Implementación de un sistema electromecánico braille, para ayuda en la enseñanza y aprendizaje de personas no videntes.,» 2024.

[38] Concepto, «USB,» 2025. [En línea]. Available: https://concepto.de/usb/.

[39] J. Castillo, «Puerto serie – Qué es, para qué sirve y tipos,» 7 3 2020. [En línea]. Available: https://www.profesionalreview.com/2020/03/07/puerto-serie-que-es-para-que-sirve-y-tipos/.

[40] Intel, «¿Qué es la tecnología Bluetooth?,» 2025. [En línea]. Available: https://www.intel.la/content/www/xl/es/products/docs/wireless/what-is-bluetooth.html.

[41] Telefónica, «Qué es el wifi,» 8 4 2025. [En línea]. Available: https://www.telefonica.com/es/sala-comunicacion/blog/wifi-que-es/.

[42] CholloTinta, «Qué son los drivers de la impresora,» 18 3 2024. [En línea]. Available: https://www.chollotinta.com/blog/que-son-los-drivers-de-la-impresora.

[43] IBM, «Operaciones básicas del sistema,» 2022.

[44] RedesZone, «¿Por qué deberías usar un adaptador WiFi USB y para qué sirve?,» 5 5 2025. [En línea]. Available: https://www.redeszone.net/tutoriales/redes-wifi/adaptador-wifi-usb-que-es-caracteristicas/.

[45] UTPL, «El braille es una herramienta clave para la inclusión y la educación,» 9 1 2025. [En línea]. Available: https://noticias.utpl.edu.ec/el-braille-es-una-herramienta-clave-para-la-inclusion-y-la-educacion.

[46] BrowserStack, «¿Qué es el modo de compatibilidad?,» 9 12 2024. [En línea]. Available: https://www.browserstack.com/guide/compatibility-mode.

[47] Ixiam, «Software libre: características y ventajas de su uso,» 15 11 2022. [En línea]. Available: https://www.ixiam.com/es/blog/software-libre-caracteristicas-y-ventajas-de-su-uso/.

[48] Python, «Python,» 2024. [En línea]. Available: https://www.python.org/.

[49] Visual Studio Code, «Visual Studio Code,» 2024. [En línea]. Available: https://code.visualstudio.com/.

[50] Y. Fernández, «Qué es Arduino, cómo funciona y qué puedes hacer con uno,» 14 11 2024. [En línea]. Available: https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno.

[51] HowToMechatronics, «Cómo configurar GRBL y controlar una máquina CNC con Arduino,» 2022. [En línea]. Available: https://howtomechatronics.com/tutorials/how-to-setup-grbl-control-cnc-machine-with-arduino/#google_vignette.

[52] P. Pal, «Software GRBL: Todo lo que necesitas saber,» 16 8 2024. [En línea]. Available: https://all3dp.com/2/grbl-software-guide/.

[53] UTPL, «El braille es una herramienta clave para la inclusión y la educación,» 9 1 2025. [En línea]. Available: https://noticias.utpl.edu.ec/el-braille-es-una-herramienta-clave-para-la-inclusion-y-la-educacion#:~:text=El%20braille%2C%20sistema%20de%20lectura,y%20logren%20una%20vida%20plena..

[54] INCI, «La importancia del braille para las personas con discapacidad visual,» 15 11 2020. [En línea]. Available: https://www.inci.gov.co/blog/la-importancia-del-braille-para-las-personas-con-discapacidad-visual.

[55] Hable, «Alfabeto y números en Braille,» 15 5 2024. [En línea]. Available: https://www.iamhable.com/es-am/blogs/articulo/braille-alphabet-numbers?srsltid=AfmBOopWrZ-ULoyR2XEFyZywt_mLEJiBsEkL_bg0X3-4k5DxO0uqBt5P.

[56] M. Serpa, I. Rojas, Y. González, B. Hernández y S. Rodríguez, «Consideraciones históricas sobre el sistema braille,» *Revista Cubana de Oftalmología,* vol. 35, nº 3, 2022.

[57] U. Hernández, F. Mirelez, J. Díaz, J. Letechipia y R. Cruz, «Desarrollo de un sistema de traducción de textos impresos al braille mediante la metodología de diseño mecatrónico electricidad,» *Científica,* vol. 29, nº 1, p. 19, 2025.

[58] S. Sánchez y E. Díaz, «Discapacidad visual, sistema Braille e inclusión educativa,» *Gestión I+D,* vol. 5, nº 2, p. 27, 2020.

[59] F. Quispe, «La comprensión lectora en Braille y los procesos psicológicos en personas con discapacidad visual,» Lima, 2022.

[60] «Plan de Creación de Oportunidades 2021-2025,» [En línea]. Available: https://www.planificacion.gob.ec/plan-de-creacion-de-oportunidades-2021-2025/.

[61] J. Cole, «¿Qué es el PLN (procesamiento del lenguaje natural)?,» 11 8 2024. [En línea]. Available: https://www.ibm.com/es-es/topics/natural-language-processing.

[62] AWS, «¿Qué es el Procesamiento de lenguaje natural (NLP)?,» 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/nlp/.

[63] «Scielo,» [En línea]. Available: http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-55462018000401487. [Último acceso: 26 Agosto 2024].

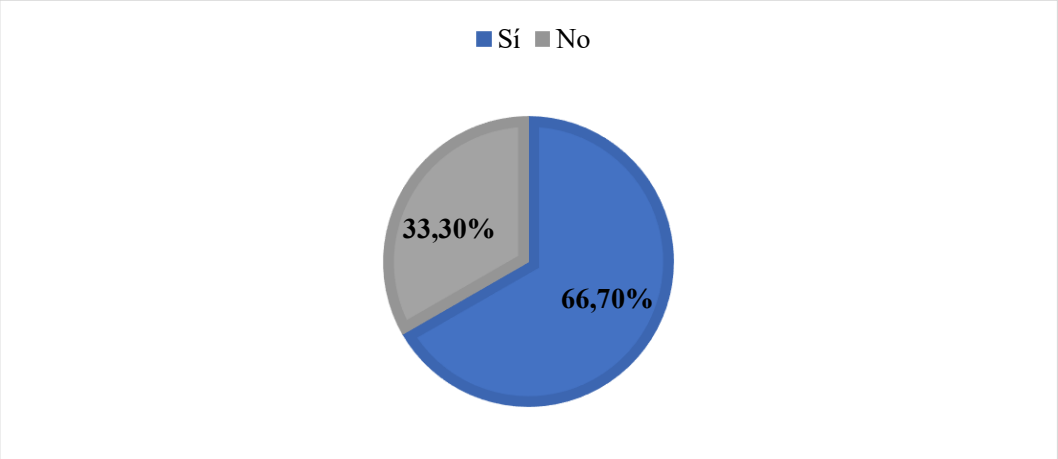
[64] «Fundación Adecco,» [En línea]. Available: https://fundacionadecco.org/notas-de-prensa/las-personas-discapacidad-apps-especificas-facilitar-sus-tareas/. [Último acceso: 25 Agosto 2024].

ANEXOS

Anexo 1. Encuesta a los docentes de la Escuela de Educación Básica "Manuela Espejo"

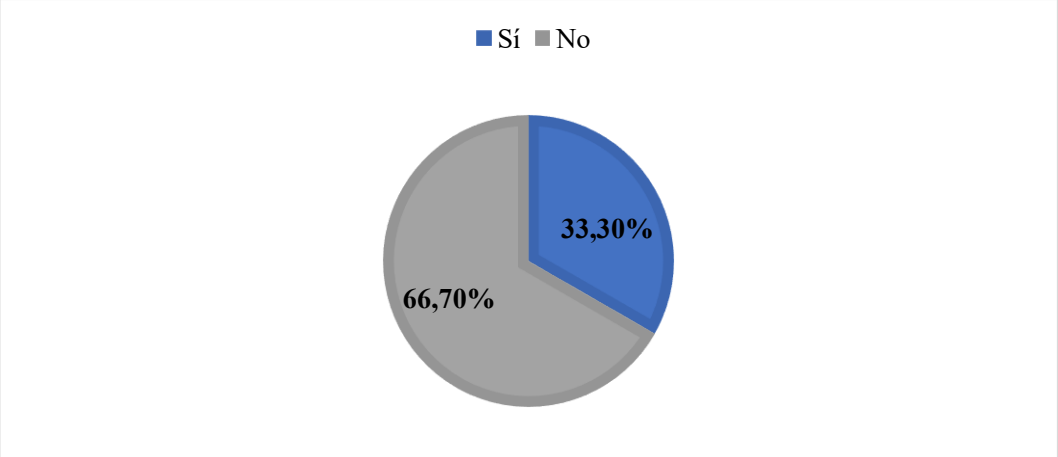
Objetivo: Conocer la percepción de los docentes acerca de la implementación de herramientas tecnológicas que faciliten convertir documentos a código Braille.

1. **¿Ha tenido estudiantes con discapacidad visual en sus clases?**



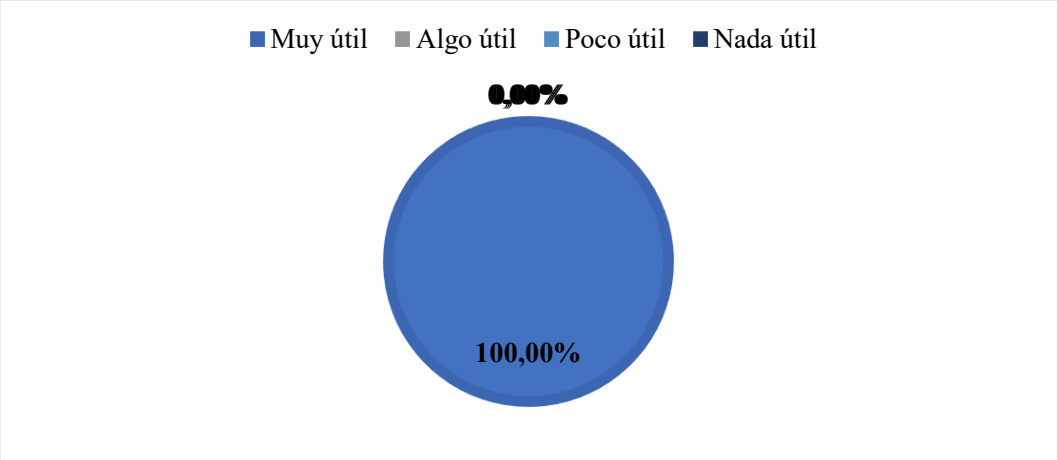
Análisis: El 66.7% de los docentes mencionan que sí han tenido estudiantes con discapacidad visual en sus clases; mientras que, el 33.3% indican que no.

2. **¿Ha utilizado o conoce alguna herramienta tecnológica que facilite la conversión de documentos a código Braille?**



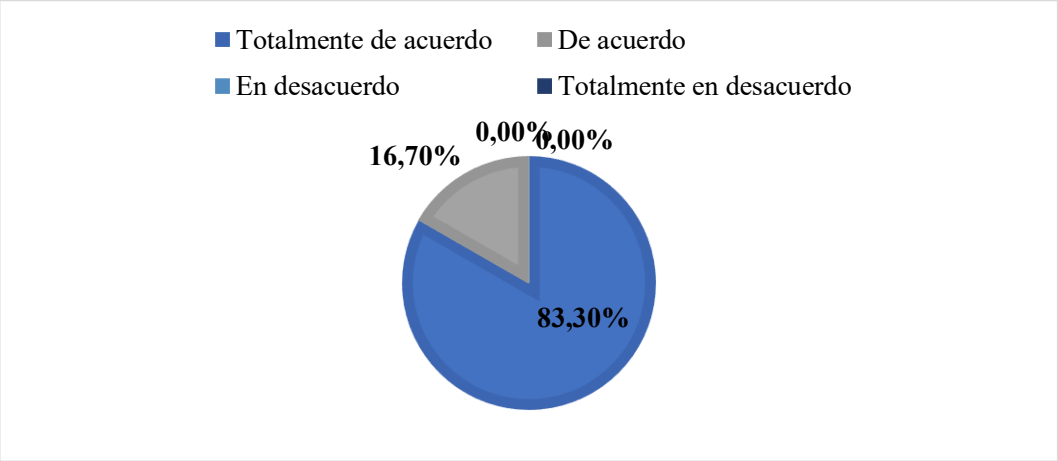
Análisis: El 66.7% de los encuestados manifiestan que no han utilizado o conocen alguna herramienta tecnológica que facilite la conversión de documentos a código Braille; mientras que, el 33.3% destacan que sí han empleado estos recursos.

3. **¿Qué tan útil considera que sería un sistema automático para convertir archivos PDF a código Braille en su entorno educativo?**



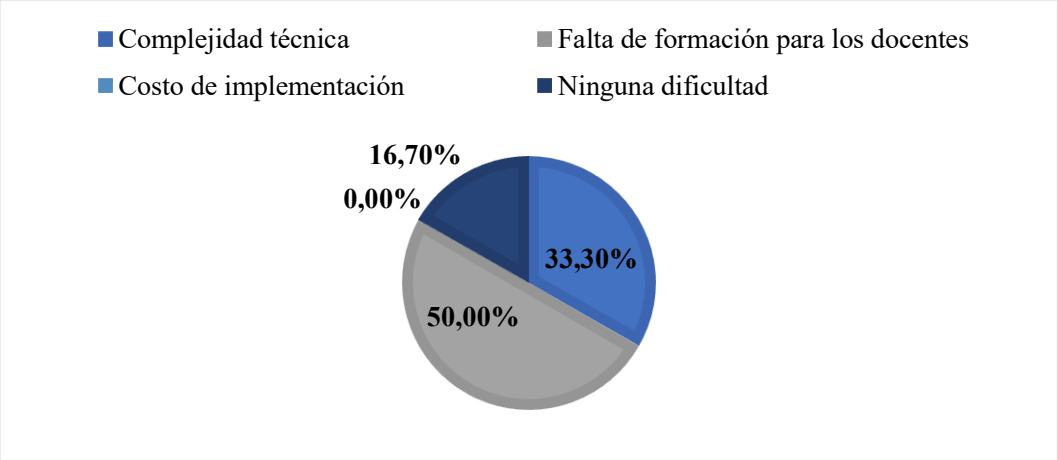
Análisis: El 100% de la población encuestada indica que, sería muy útil contar con un sistema automático para convertir archivos PDF a código Braille en su entorno educativo.

4. **¿Cree que la implementación de un sistema de conversión automática de PDF a Braille mejoraría la calidad del material didáctico disponible para estudiantes con discapacidad visual?**



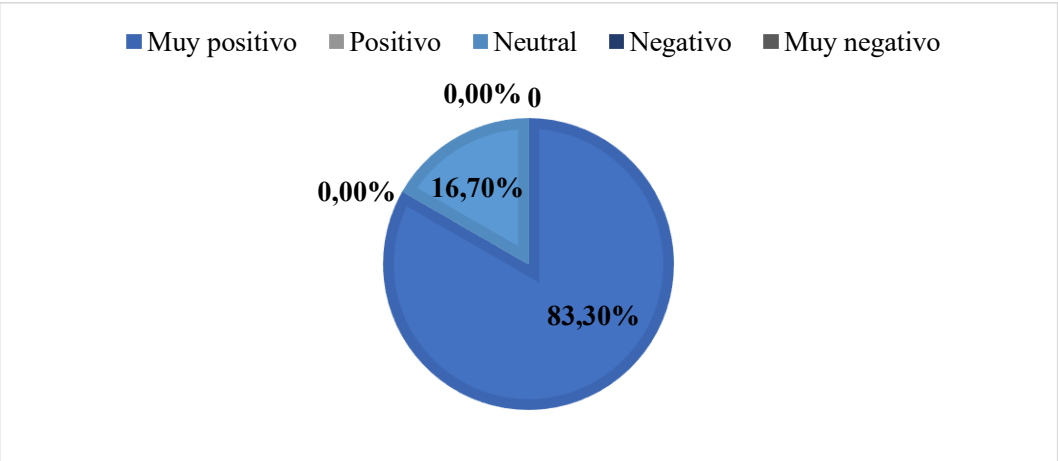
Análisis: El 83.3% de los docentes está totalmente de acuerdo con que la implementación de un sistema de conversión automática de PDF a Braille mejoraría la calidad del material didáctico disponible para estudiantes con discapacidad visual; mientras que, el 16.7% manifiesta estar de acuerdo con esta afirmación.

5. **¿Qué dificultades cree que podrían presentarse al implementar un software de traducción de lenguaje natural a código Braille?**



Análisis: El 50% de los encuestados destaca que, la falta de información para los docentes podría presentarse como dificultad al implementar un software de traducción de lenguaje natural a código Braille; mientras que, el 33.3% menciona que podría ser la complejidad técnica y el 16.7% considera que no habría ninguna dificultad.

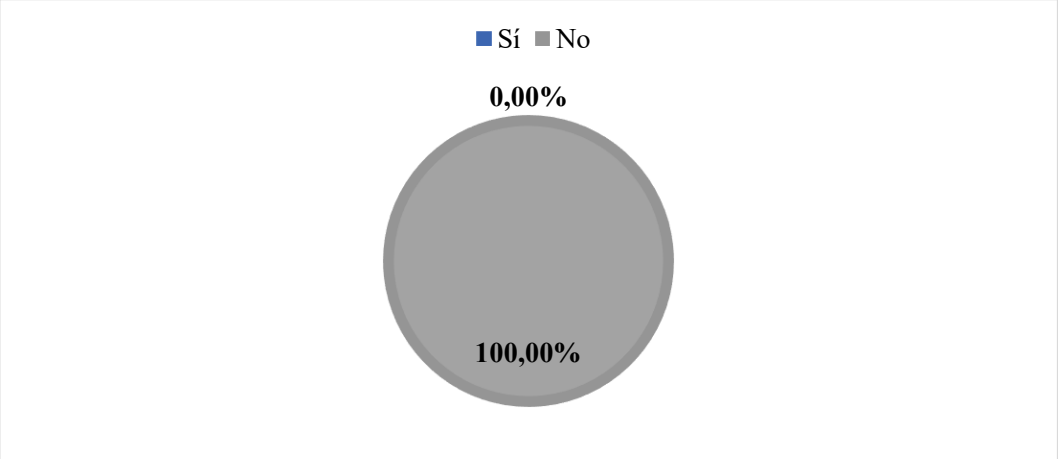
6. **¿Qué impacto cree que tendría el uso de un traductor de lenguaje natural a código Braille en la inclusión digital de personas con discapacidad visual en la educación superior?**



Análisis: El 83.3% de la población encuestada cree que tendría un impacto muy positivo el uso de un traductor de lenguaje natural a código Braille en la inclusión

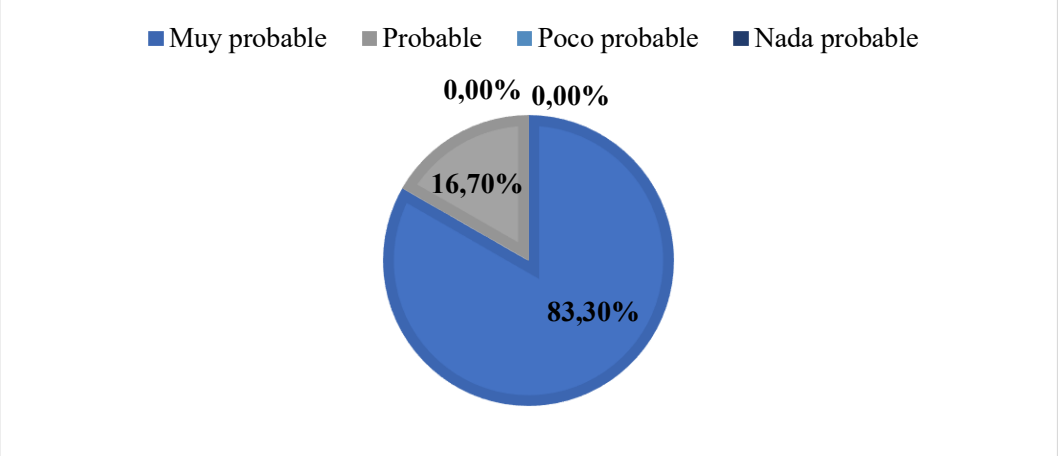
digital de personas con discapacidad visual en la educación superior; mientras que, el 16.7% se encuentra en estado neutral ante esta afirmación.

7. **¿Dispone de los recursos necesarios (hardware, software, personal) para implementar estas herramientas?**



Análisis: El 100% de los docentes manifiestan que, no disponen de los recursos necesarios, como hardware, software y personal, para implementar estas herramientas.

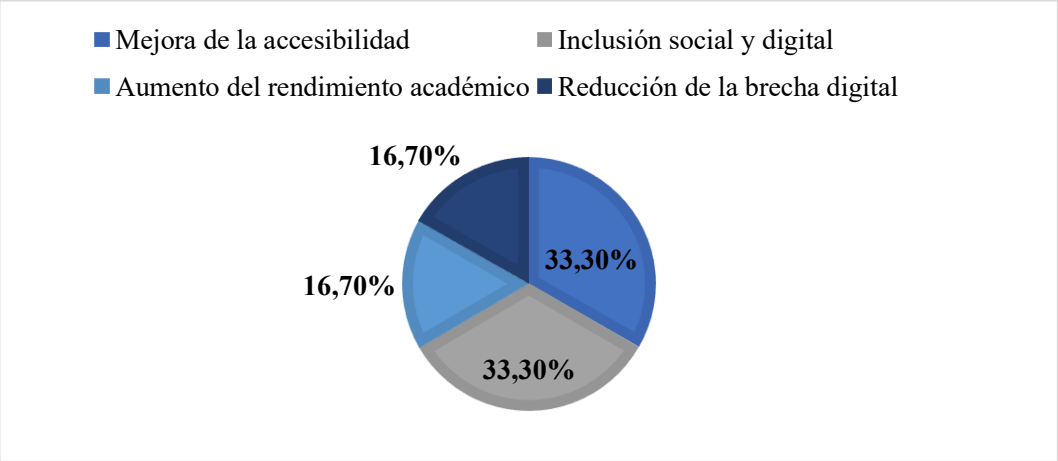
8. **¿Qué tan probable es que recomiende a otros docentes el uso de un sistema de conversión de PDF a código Braille si se demostrara su efectividad?**



Análisis: El 83.3% de los encuestados indican que, es muy probable que recomiende a otros docentes el uso de un sistema de conversión de PDF a código

Braille si se demostrara su efectividad; mientras que, el 16.7% menciona que es probable.

9. **¿Cuáles cree que son los principales beneficios de implementar un sistema de conversión automática a código Braille en la educación superior?**

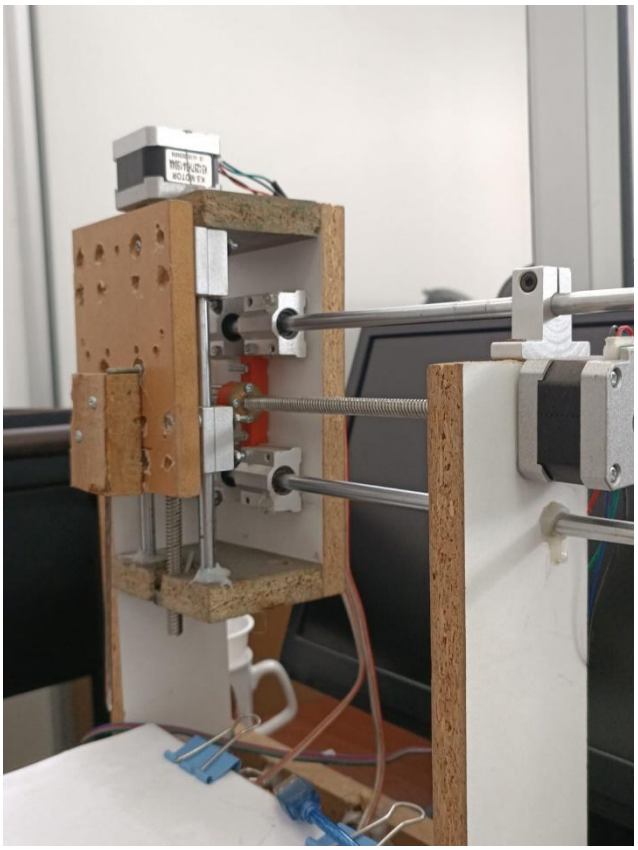


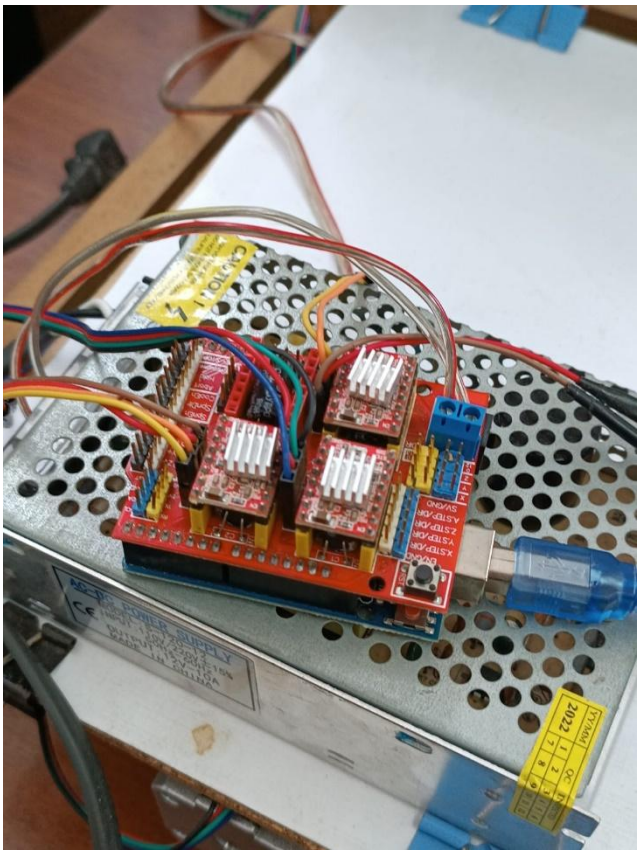
Análisis: El 33.3% de los docentes revela que, la mejora de la accesibilidad y la inclusión social y digital son los principales beneficios de implementar un sistema de conversión automática a código Braille en la educación superior; mientras que, el 16.7% manifiesta que los beneficios serían el aumento del rendimiento académico y la reducción de la brecha digital.

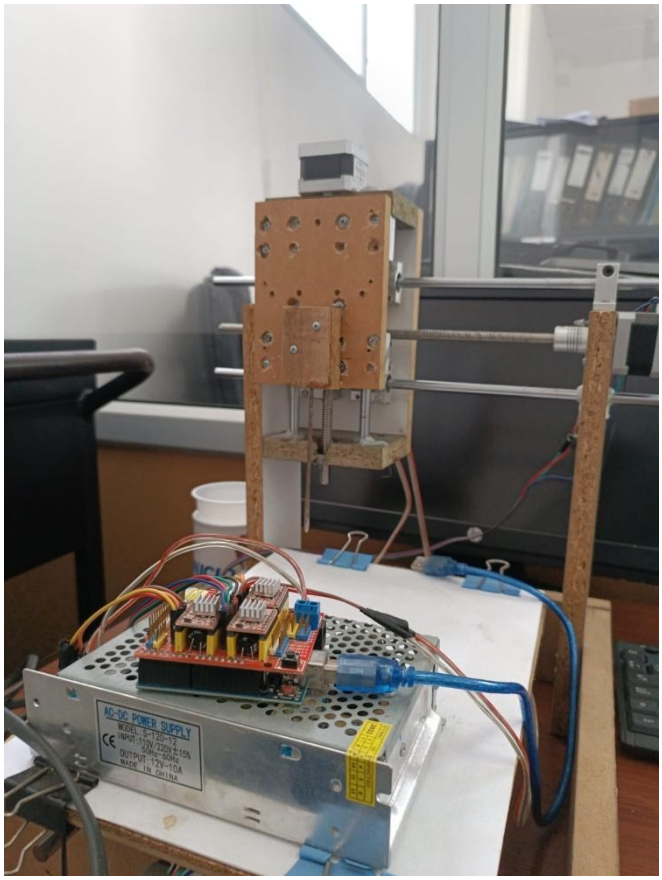
10. **¿Qué características adicionales considera importantes para un software de conversión de texto a Braille en el contexto educativo?**

Análisis: Entre las características adicionales que se consideran importantes para un software de conversión de texto a Braille en el contexto educativo, se encuentran: amigable y accesible, el idioma, las imágenes, que sea práctico, accesible y sencillo. Así mismo, que debe ser de fácil manejo y que todos puedan acceder, tanto estudiantes y docentes.

Anexo 2. Impresora Braille







Anexo 3. Manual de codificación del software Braille

Figura 1. Integración de archivos

```
# def upload_files(self): # Maneja la subida de archivos
    file_path = FileManager.get_file_path()

    if file_path:
        self.path = file_path
        self.mostrar_barra_de_progreso()

        self.update_convert_button_state() # Actualiza el estado del botón de conversión

        file_name = PathBuilder.extract_file_name(file_path)

        self.display_file_name(file_name)

        file_extension = PathBuilder.extract_file_extension(file_path)

        self.ocultar_barra_de_progreso()
        self.display_icon(file_extension)

        self.process_file_extension(file_extension, file_path)
```

Figura 2. Subida de archivos

```python
def upload_files(self):  # Maneja la subida de archivos
    file_path = FileManager.get_file_path()

    if file_path:
        self.path = file_path

        self.mostrar_barra_de_progreso()
        self.update_convert_button_state()  # Actualiza el estado del botón de conversión
        file_name = PathBuilder.extract_file_name(file_path)
        self.display_file_name(file_name)

        file_extension = PathBuilder.extract_file_extension(file_path)

        # Crear un hilo para cargar el archivo sin bloquear la UI, pasando ambos parámetros
        threading.Thread(target=self.load_file, args=(file_path, file_extension), daemon=True).start()
```

Figura 3. Procesamiento de documentos

```python
def load_file(self, file_path, file_extension):
    try:
        if file_extension == '.pdf':
            doc = fitz.open(file_path)
            total_pages = len(doc)
            self.progress_bar['maximum'] = total_pages

            for i, page in enumerate(doc):
                text = page.get_text()
                # Aquí puedes guardar o procesar el texto
                self.progress_bar['value'] = i + 1
                self.update_idletasks()

        elif file_extension == '.docx':
            doc = docx.Document(file_path)
            total_paragraphs = len(doc.paragraphs)
            self.progress_bar['maximum'] = total_paragraphs

            for i, para in enumerate(doc.paragraphs):
                text = para.text
                # Aquí puedes guardar o procesar el texto
                self.progress_bar['value'] = i + 1
                self.update_idletasks()

        else:
            # Otros tipos de archivos: leer por tamaño de bytes
            file_size = os.path.getsize(file_path)
            self.progress_bar['maximum'] = file_size

            with open(file_path, 'rb') as f:
                bytes_read = 0
                chunk_size = 4096
                while True:
                    chunk = f.read(chunk_size)
                    if not chunk:
                        break
                    bytes_read += len(chunk)
                    self.progress_bar['value'] = bytes_read
                    self.update_idletasks()

        self.display_icon(file_extension)
        self.process_file_extension(file_extension, file_path)
        self.ocultar_barra_de_progreso()
```

Figura 4. Extracción de documentos PDF

```python
def extract_text_from_pdf(file_path):
    try:

        with open(file_path, "rb") as file:
            pdf_reader = PyPDF2.PdfReader(file)
            text = ""
            for page_num in range(len(pdf_reader.pages)):
                page = pdf_reader.pages[page_num]
                text += page.extract_text()

            return text
    except Exception as e:
        messagebox.showerror(
            "Error", f"Error durante la extracción del PDF: {e}")
```

Figura 5. Extracción de documentos Word

```
def extract_text_from_word(file_path):
    try:
        doc = docx.Document(file_path)
        text = ""
        for paragraph in doc.paragraphs:
            text += paragraph.text + "\n"
        return text
    except Exception as e:
        messagebox.showerror(
            "Error", f"Error durante la extracción del documento DOCX: {e}")
```

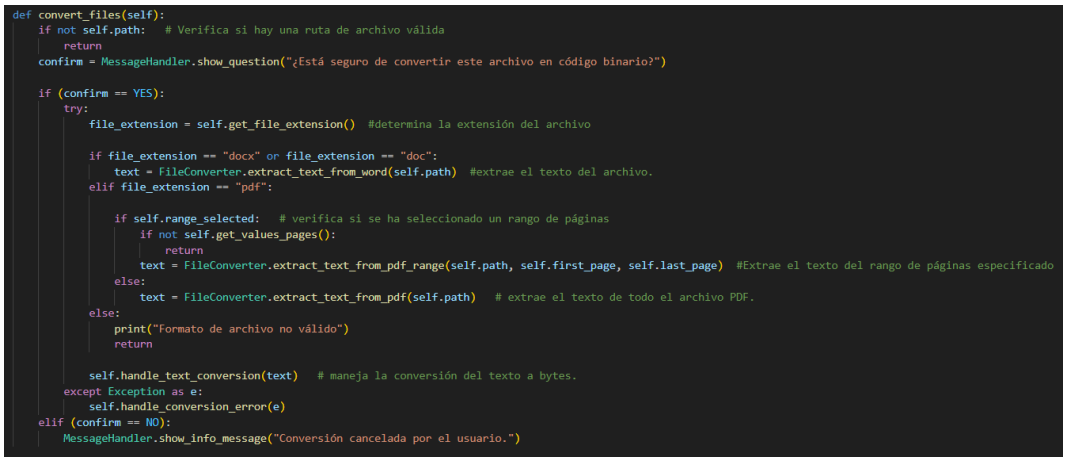
Figura 6. Filtrado de escritos por límites predefinidos

```
@staticmethod
def extract_text_from_pdf_range(file_path, start, end):
    try:
        print(file_path)
        with open(file_path, "rb") as file:
            pdf_reader = PyPDF2.PdfReader(file)
            text = ""
            for page_num in range(start - 1, end):
                page = pdf_reader.pages[page_num]
                text += page.extract_text()

            return text
    except Exception as e:
        messagebox.showerror(
            "Error", f"Error durante la extracción del rango de páginas del PDF: {e}")
```

Figura 7. Botón de convertir

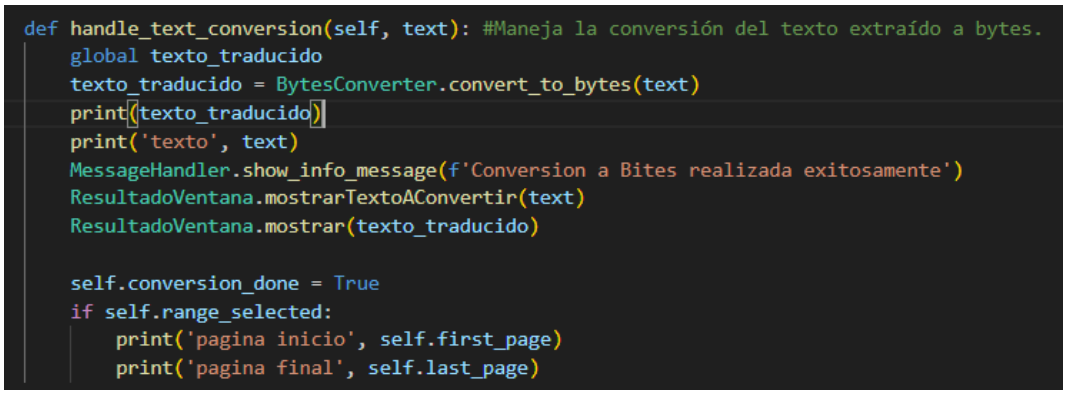
```
def create_convert_button(self, parent):
    # Botón "Convertir"
    self.convert_button = Button(parent, text="Convertir", command=self.convert_files)
    StyleManager.configure_style_global(self.convert_button, BUTTON_STYLE)
    StyleManager.configure_style_global(self.convert_button, CONVERT_BUTTON_STYLE)
    self.convert_button.pack(side=LEFT, padx=(0, 10))
```

Figura 8. Conversión de archivos



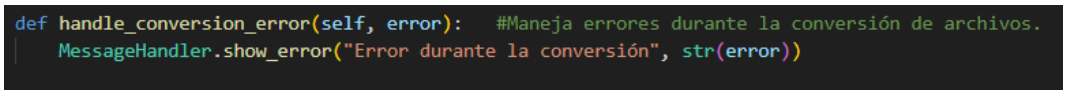
```
def convert_files(self):
    if not self.path:   # Verifica si hay una ruta de archivo válida
        return
    confirm = MessageHandler.show_question("¿Está seguro de convertir este archivo en código binario?")

    if (confirm == YES):
        try:
            file_extension = self.get_file_extension()  #determina la extensión del archivo

            if file_extension == "docx" or file_extension == "doc":
                text = FileConverter.extract_text_from_word(self.path)  #extrae el texto del archivo.
            elif file_extension == "pdf":

                if self.range_selected:   # verifica si se ha seleccionado un rango de páginas
                    if not self.get_values_pages():
                        return
                    text = FileConverter.extract_text_from_pdf_range(self.path, self.first_page, self.last_page)  #Extrae el texto del rango de páginas especificado
                else:
                    text = FileConverter.extract_text_from_pdf(self.path)   # extrae el texto de todo el archivo PDF.
            else:
                print("Formato de archivo no válido")
                return

            self.handle_text_conversion(text)   # maneja la conversión del texto a bytes.
        except Exception as e:
            self.handle_conversion_error(e)
    elif (confirm == NO):
        MessageHandler.show_info_message("Conversión cancelada por el usuario.")
```

Figura 9. Conversión de texto extraído a bytes



```
def handle_text_conversion(self, text): #Maneja la conversión del texto extraído a bytes.
    global texto_traducido
    texto_traducido = BytesConverter.convert_to_bytes(text)
    print(texto_traducido)
    print('texto', text)
    MessageHandler.show_info_message(f'Conversion a Bites realizada exitosamente')
    ResultadoVentana.mostrarTextoAConvertir(text)
    ResultadoVentana.mostrar(texto_traducido)

    self.conversion_done = True
    if self.range_selected:
        print('pagina inicio', self.first_page)
        print('pagina final', self.last_page)
```

Figura 10. Maneja errores en la conversión de documentos



```
def handle_conversion_error(self, error):   #Maneja errores durante la conversión de archivos.
    MessageHandler.show_error("Error durante la conversión", str(error))
```

Figura 11. Conversión de archivos a binarios

```python
from mock.diccionario import braille_translation
from decouple import config #variables de entorno
from mock.constants import numberPunctuations, escapeCharacters
from modules.messageHandler.message import MessageHandler


file_translated_global = config('FILE_TRASLATED_BITS')

class BytesConverter:
    @staticmethod
    def convert_to_bytes(textToConvert): #PERFECTO
            try:
                convertedBits = ''
                output_file_path = file_translated_global
                isNumber = False
                for character in textToConvert:
                    if character in escapeCharacters:
                        convertedBits += character
                        continue
                    if character.isupper():
                        convertedBits += braille_translation['caps']
                        character = character.lower()
                    if character.isdigit():
                        if not isNumber:
                            isNumber = True
                            convertedBits += braille_translation['num']
                    else:
                        if isNumber and character not in numberPunctuations:
                            isNumber = False
                    convertedBits += braille_translation.get(character, '')

                with open(output_file_path, 'w') as file:
                    file.write(convertedBits)
                return convertedBits
            except Exception as e:
                MessageHandler.showerror("Error", f"Error durante la conversión a bits: {e}")
```

Figura 12. Alfabeto alfanumérico

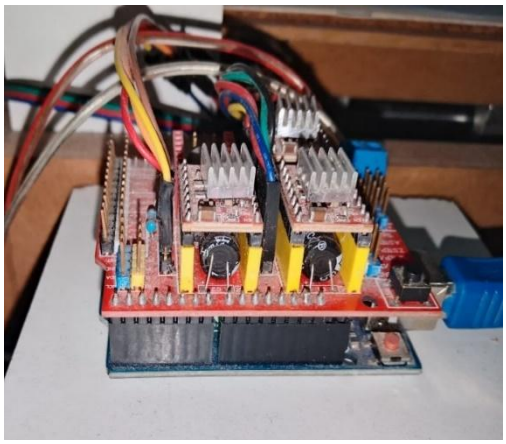
```python
braille_translation = {
    'a': '100000', 'b': '101000', 'c': '110000', 'd': '110100', 'e': '100100', 'f': '111000',
    'g': '111100', 'h': '101100', 'i': '011000', 'j': '011100', 'k': '100010', 'l': '101010',
    'm': '110010', 'n': '110110', 'ñ': '111101', 'o': '100110', 'p': '111010', 'q': '111110',
    'r': '101110', 's': '011010', 't': '011110', 'u': '100011', 'v': '101011', 'w': '011101',
    'x': '110011', 'y': '110111', 'z': '100111', ' ': '000000', '1': '100000', '2': '101000',
    '3': '110000', '4': '110100', '5': '100100', '6': '111000', '7': '111100', '8': '101100',
    '9': '011000', '0': '011100', 'num': '010111', 'caps': '010001', '.': '000010', ',': '001000',
    ';': '001010', ':': '001100', '-': '000011', '¿': '001001', '?': '001001', '¡': '001110',
    '!': '001110', '"': '001011', '(': '101001', ')': '010110', 'á': '101111', 'é': '011011',
    'í': '010010', 'ó': '010011', 'ú': '011111', 'ü': '101101',
    '#': '00100011', '$': '00100100', '%': '00100101', '&': '00100110', '/': '00101111',
    '*': '00101010', '+': '00101011', '<': '00111100', '=': '00111101', '>': '00111110',
    '[': '01011011', '\\': '01011100', ']': '01011101', '^': '01011110', '`': '01100000',
    '{': '01111011', '|': '01111100', '}': '01111101', '~': '01111110'
}


# Define the Braille alphabet in binary format

braille_alphabet = {
    'a': '100000', 'b': '110000', 'c': '100100', 'd': '100110',
    'e': '100010', 'f': '110100', 'g': '110110', 'h': '110010',
    'i': '010100', 'j': '010110', 'k': '101000', 'l': '111000',
    'm': '101100', 'n': '101110', 'o': '101010', 'p': '111100',
    'q': '111110', 'r': '111010', 's': '011100', 't': '011110',
    'u': '011010', 'v': '011000', 'w': '010111', 'x': '101111',
    'y': '101011', 'z': '101101', ' ' : '000000'
}
```

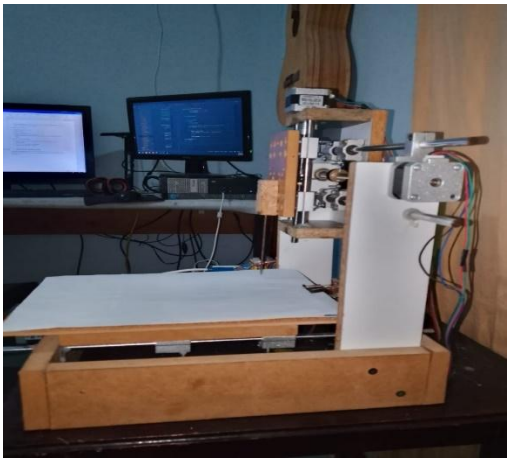
Figura 13. Función para ajuste del botón de impresión

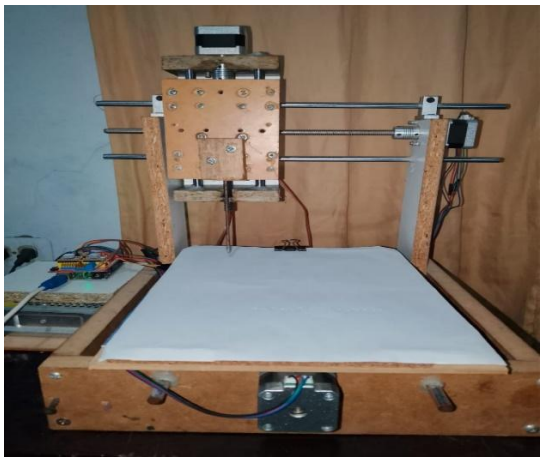
```python
def create_print_button(self, parent):  #PERFECTO
    # Botón "Imprimir"
    print_button = Button(parent, text="Imprimir", command=self.print_files)
    StyleManager.configure_style_global(print_button, BUTTON_STYLE)
    StyleManager.configure_style_global(print_button, PRINT_BUTTON_STYLE)
    print_button.pack(side=LEFT, padx=(0, 10))
```

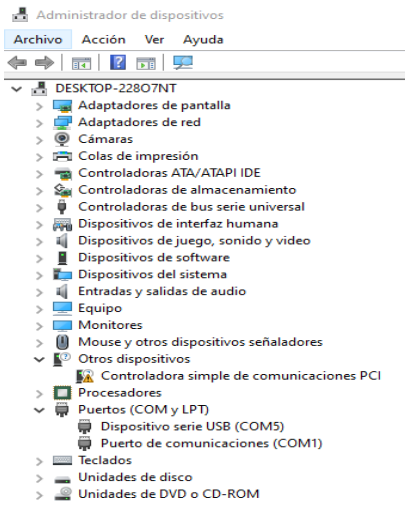
```python
# ARDUINO
def print_files(self):
    if not self.conversion_done:
        MessageHandler.show_error("Error", "Primero debe convertir el archivo")
        return

    # Abre el archivo de texto para lectura
    with open(file_traslated_bits, 'r') as file:
        binary_text = file.read().strip()   # Lee y limpia el contenido del archivo
        print("binary_text", binary_text)

        # Convierte el texto binario a líneas de G-code
        gcode_lines = BinaryToGcode.binary_to_gcode(binary_text)
        print("gcode_lines", gcode_lines)

        #letra g
        # gcode_lines = [
        #     "G90",
        #     "G1 F5000",
        #     "G1 X0.00 Y0.00 Z10.00",
        #     "G1 X150.00 Y105.00",
        #     "G1 Z-2.00",
        #     "G1 Z10.00",
        #     "G1 X147.46 Y105.00",
        #     "G1 Z-2.00",
        #     "G1 Z10.00",
        #     "G1 X150.00 Y102.46",
        #     "G1 Z-2.00",
        #     "G1 Z10.00",
        #     "G1 X147.46 Y102.46",
        #     "G1 Z-2.00",
        #     "G1 Z10.00",
        #     "G1 X0.00 Y0.00 Z10.00",
        # ]



        # Llama a la función para enviar el G-code a Arduino
        ArduinoManager.communicate_with_arduino(gcode_lines)
```

Anexo 4. Verificación de la conectividad física



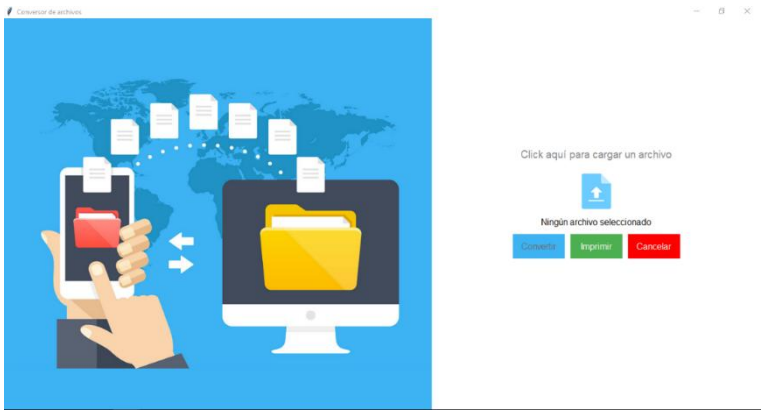


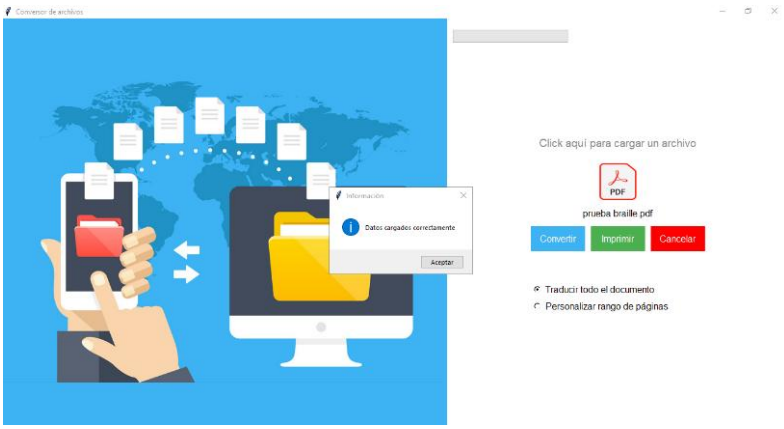


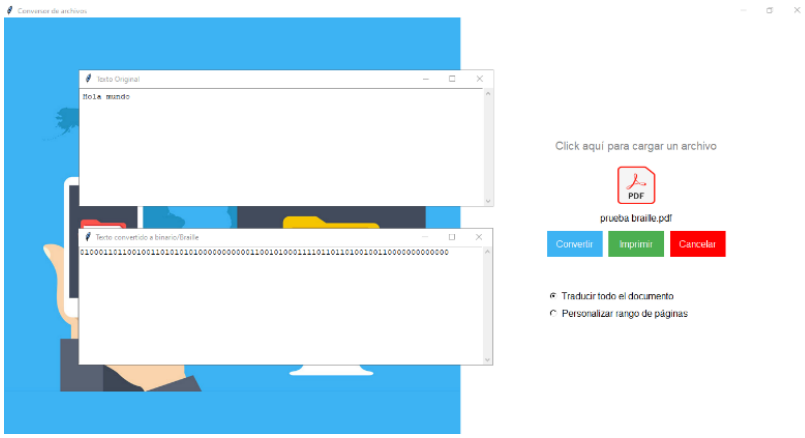


Anexo 5. Verificación lógica (software)

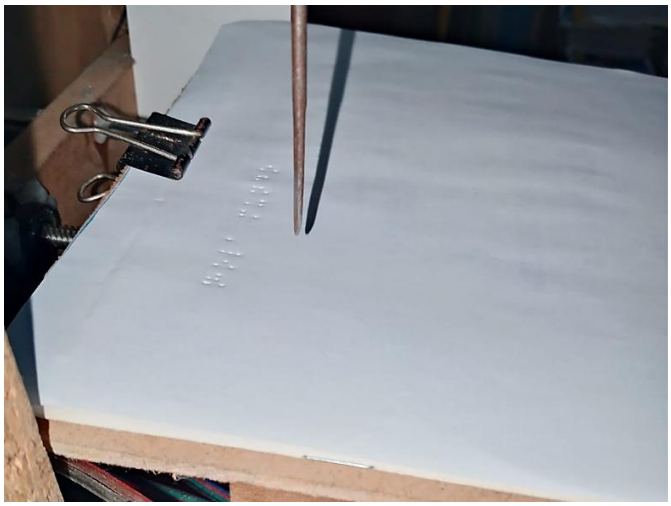
```python
import serial    La importación "serial" no se ha podido resolver desde el origen
import time
from decouple import config #variables de entorno
from modules.messageHandler.message import MessageHandler

serial_port='COM5'
# serial_port =config('SERIAL_PORT_W')
# 115200,  9600
```

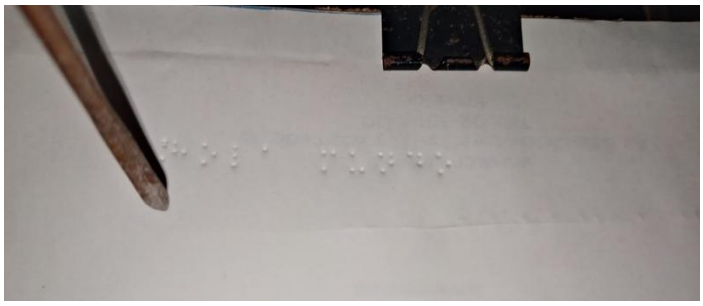


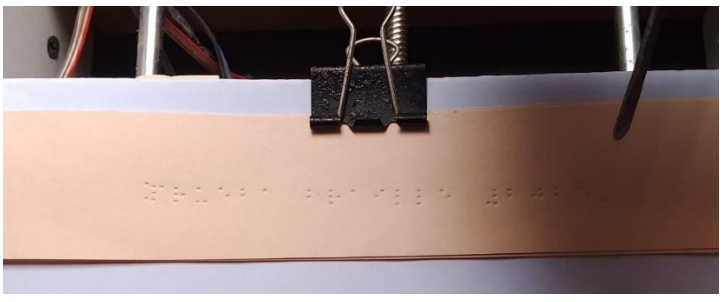


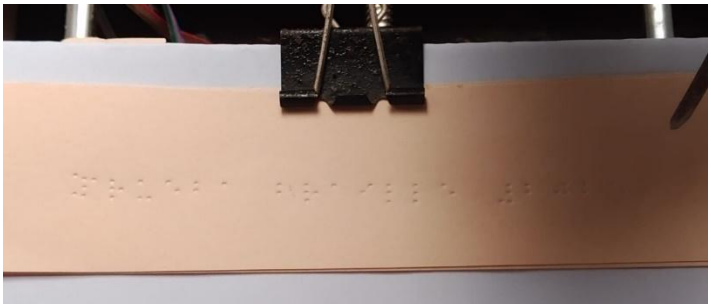


Anexo 6. Prueba de impresión básica

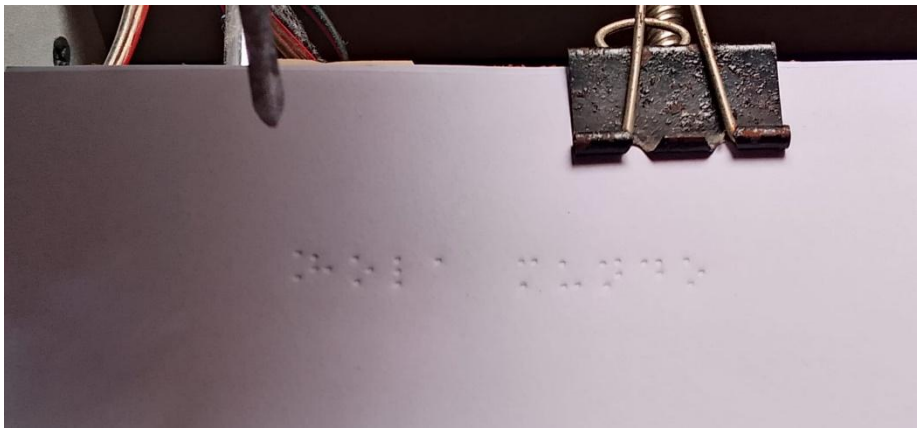


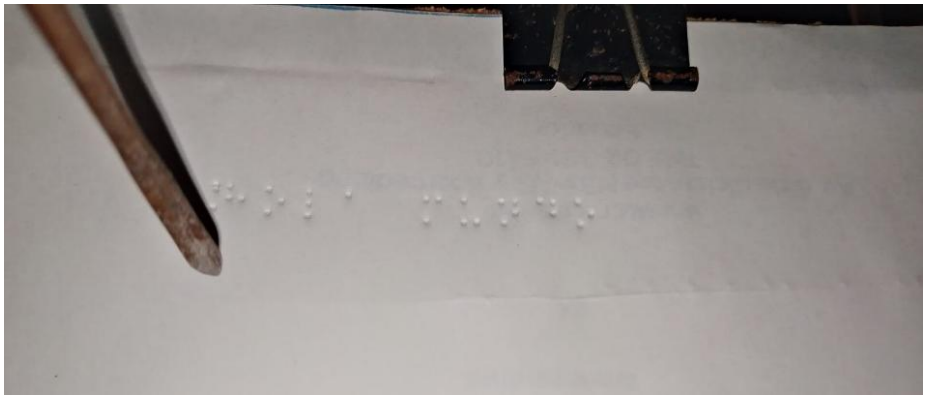




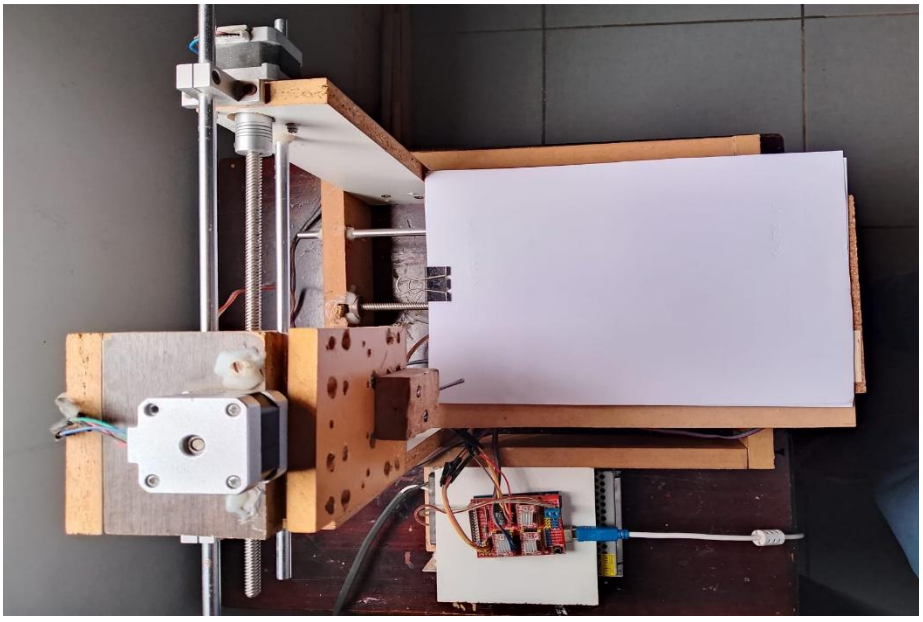


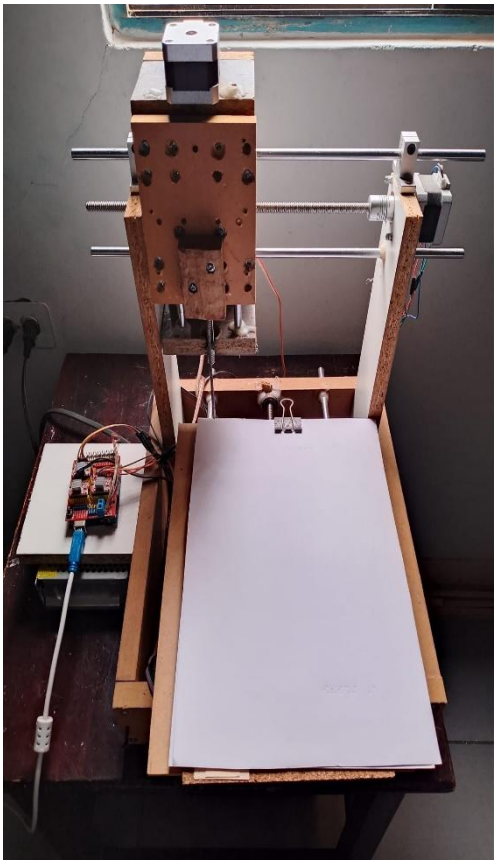
Anexo 7. Test de consistencia

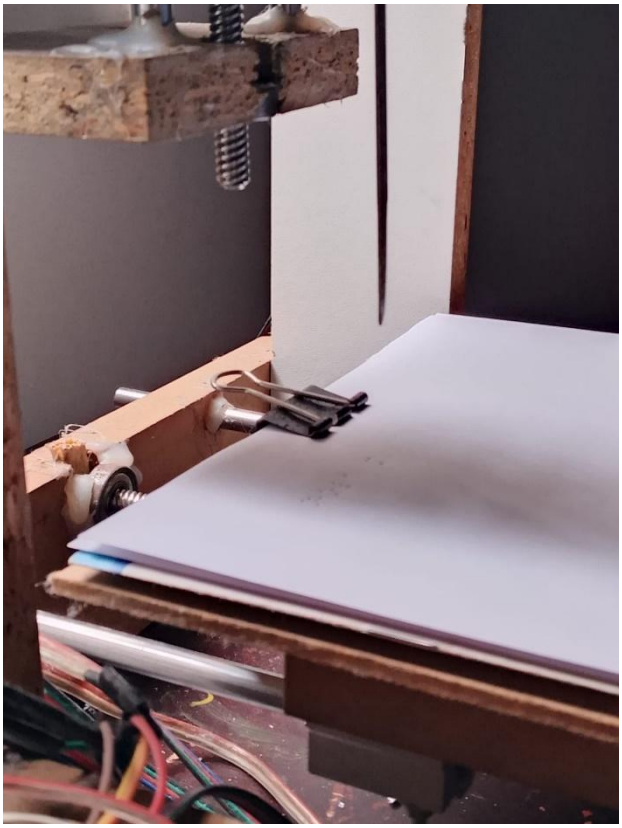




Anexo 8. Evaluación de tolerancia y fallos físicos







Anexo 9. Verificación de módulos

Figura 1. Escrito convertido a braille

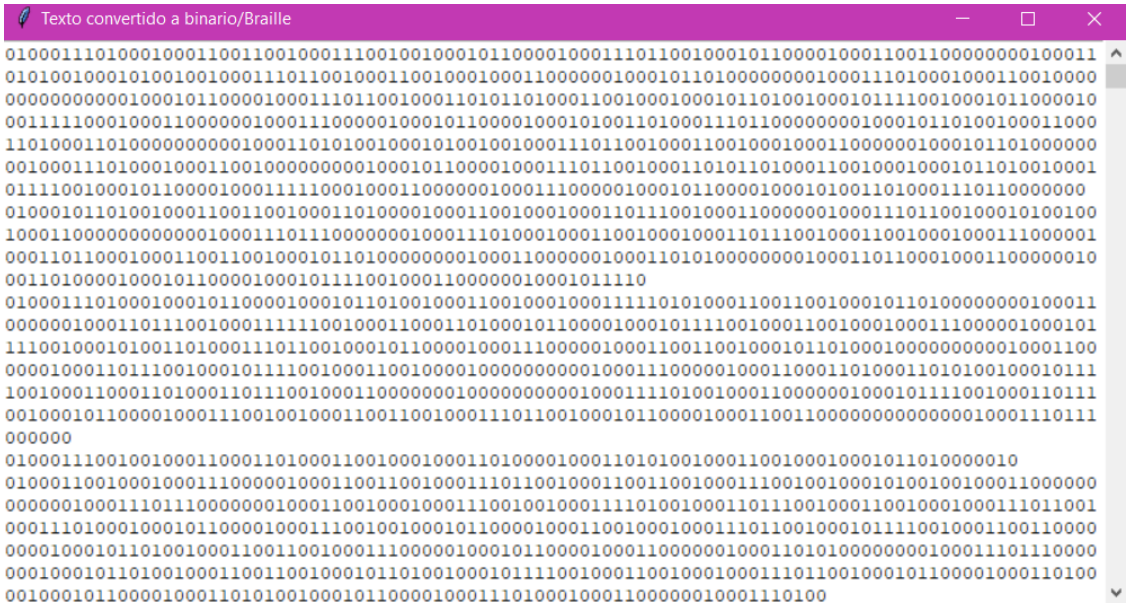


Figura 2. Escrito procesado

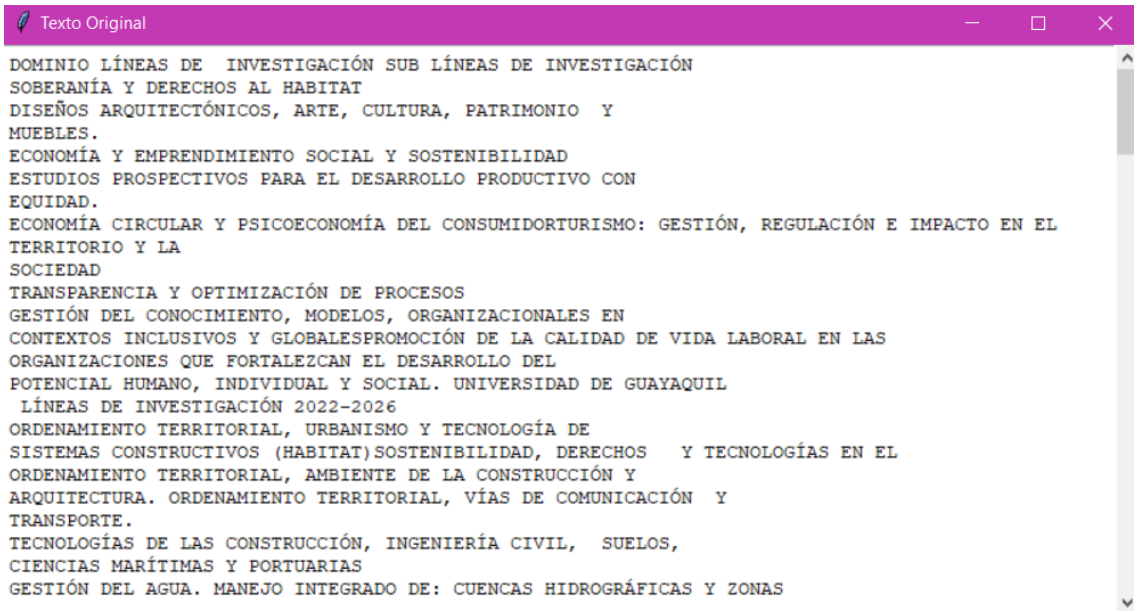
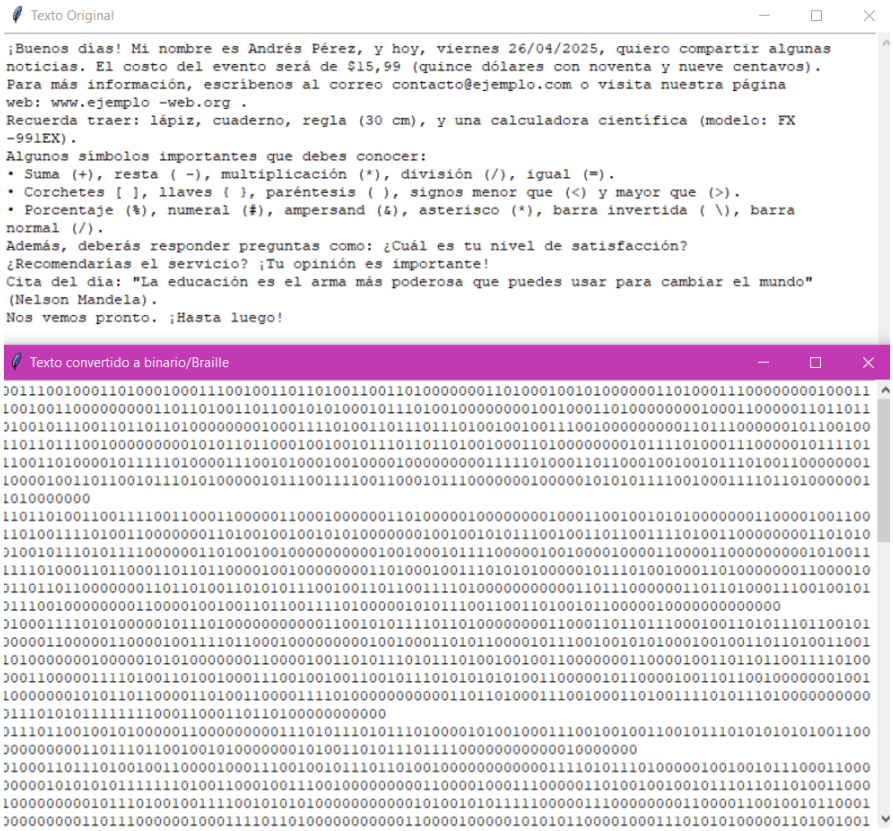


Figura 3. Representación de texto traducido



Anexo 10. Verificación de traducción

Figura 1. Conversión y creación de archivo

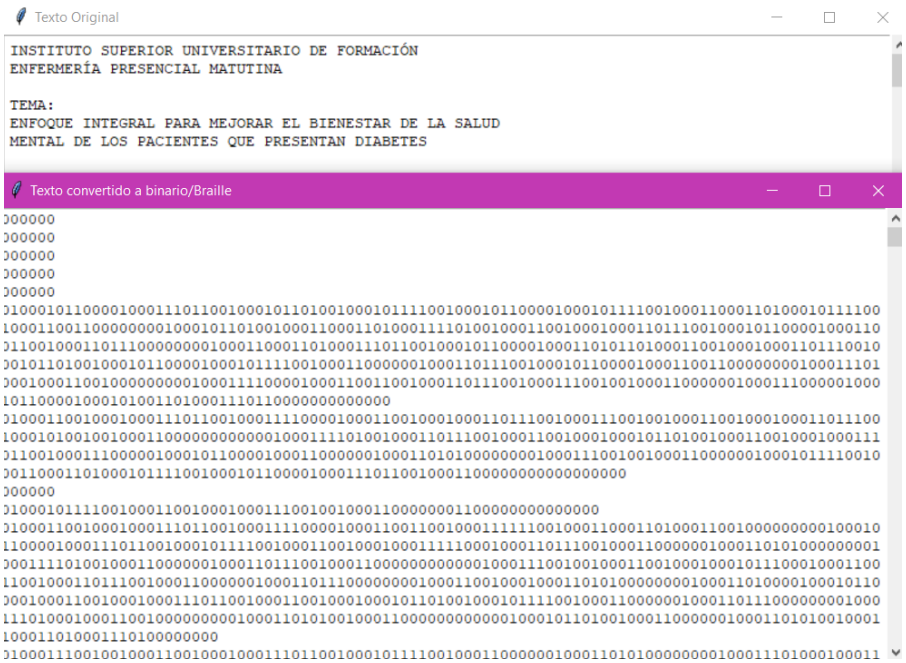
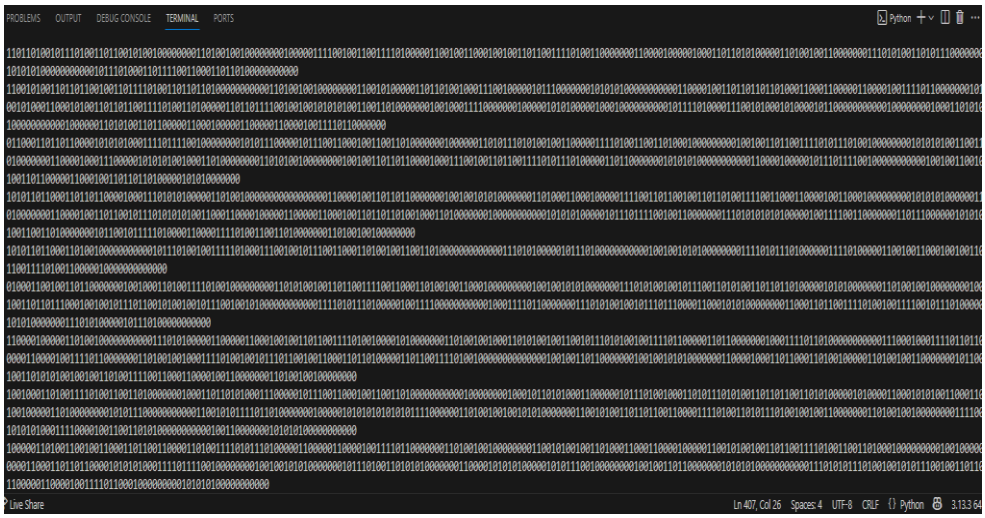


Figura 2. Traducción en la terminal



Anexo 11. Evaluación de conversión

Figura 1. Precisión de caracteres en el código

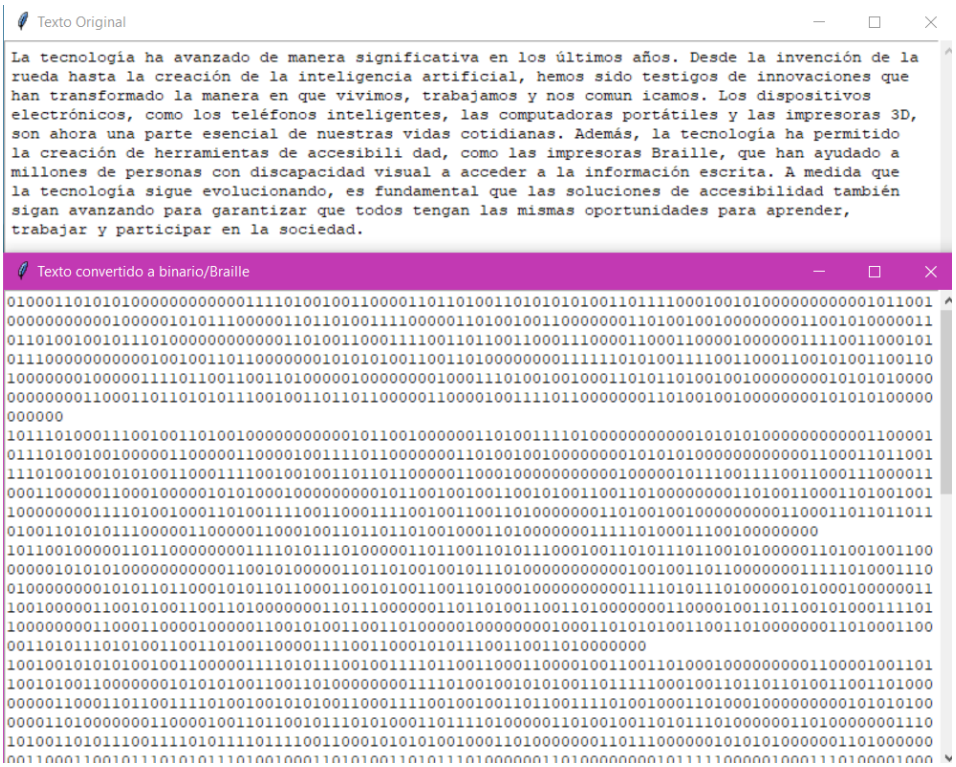
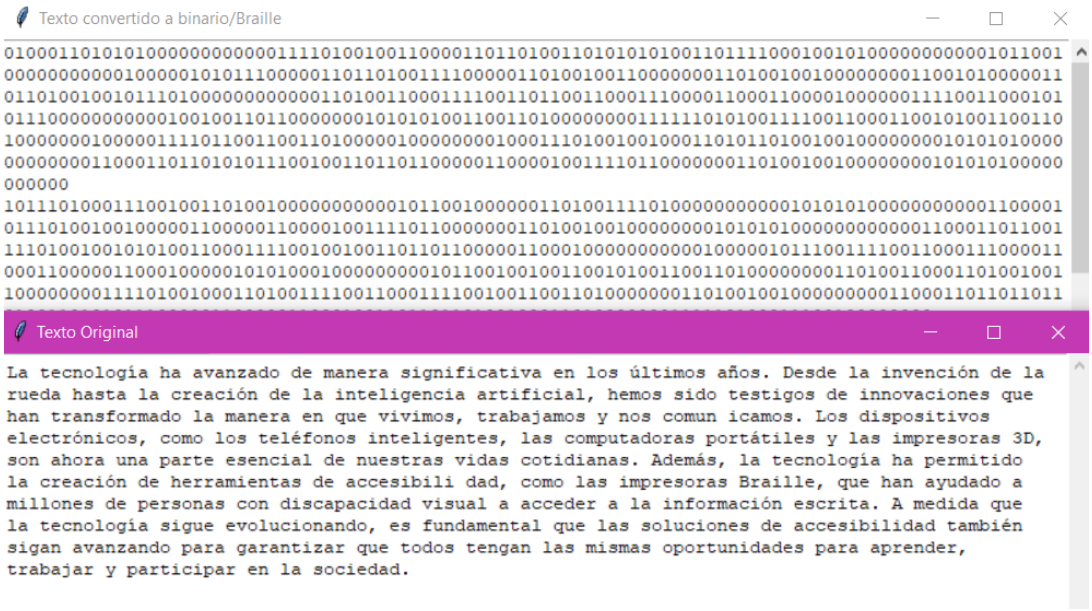


Figura 2. Uniformidad de traducción en documento



Anexo 12. Test de usabilidad

Figura 1. Página principal

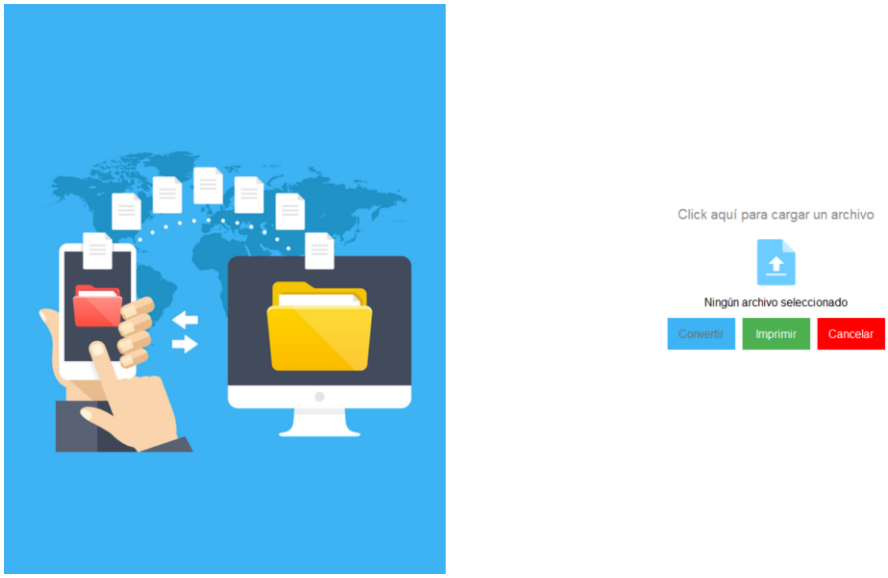
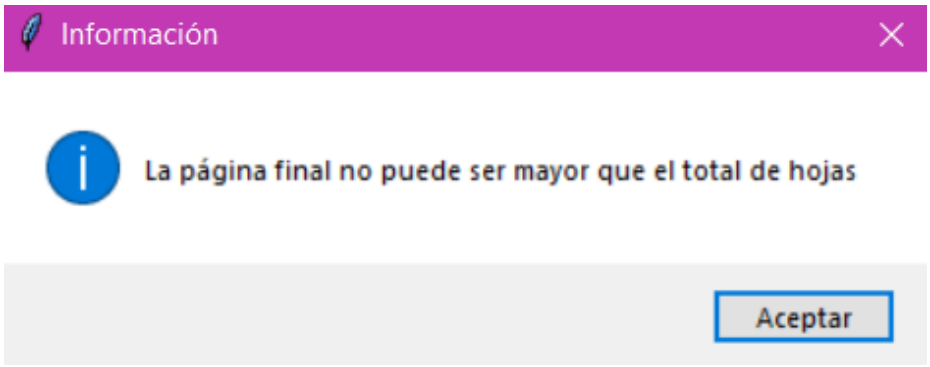


Figura 2. Advertencias en el sistema



Anexo 13. Ajuste eficaz a distintos contextos sociales y de aprendizaje

Figura 1. Adecuación a escritos académicos

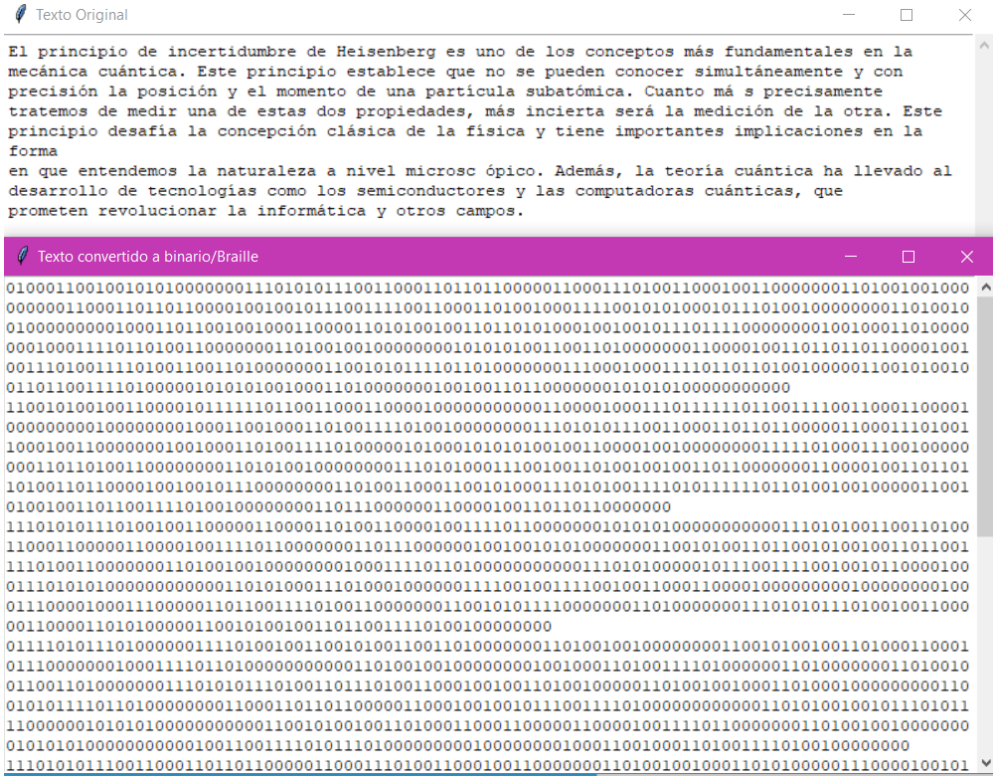
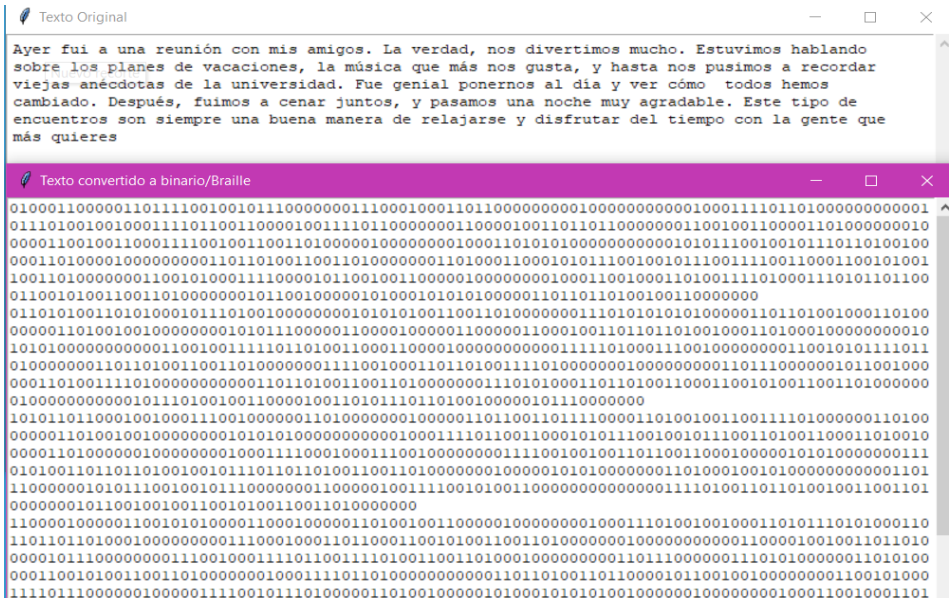


Figura 2. Integración a material social



Anexo 14. Manual de usuario del software traductor

Descripción

El presente manual de usuario proporciona instrucciones concisas con respecto al software traductor Braille desarrollado, el cual es una herramienta que facilita el proceso pedagógico e incluye en sus actividades a los individuos discapacitados visualmente, permitiendo traducir escritos en código Braille automáticamente, en poco tiempo; de tal forma, el sistema acepta archivos en formato docx o pdf. Esta guía se realiza con el fin de explicar los módulos principales y funcionalidades en cada interfaz.

Objetivo de la guía

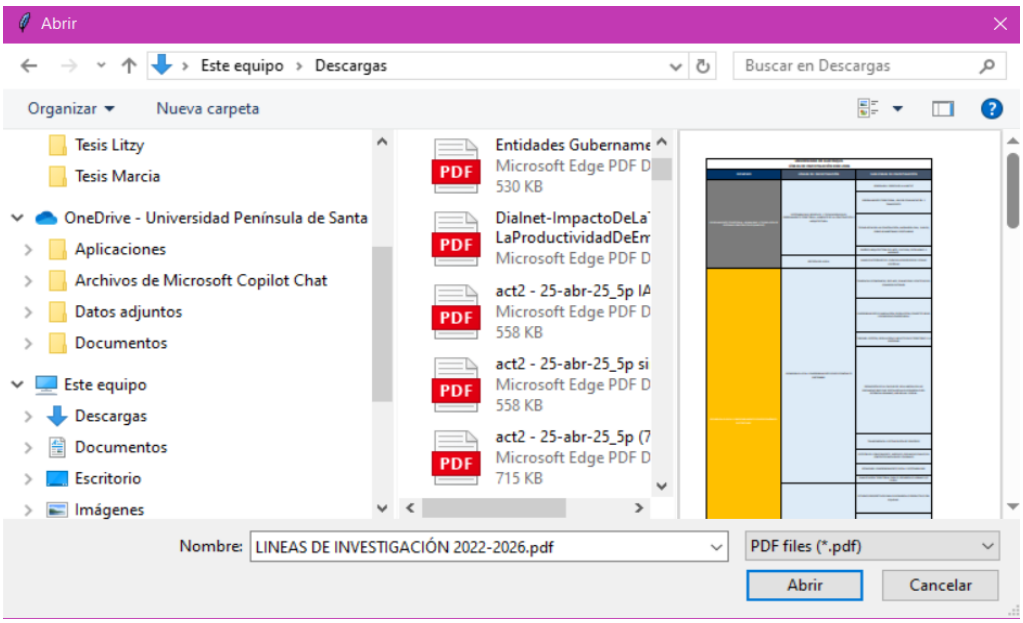
Proporcionar las orientaciones necesarias para el manejo eficaz del software traductor Braille.

Contenido principal

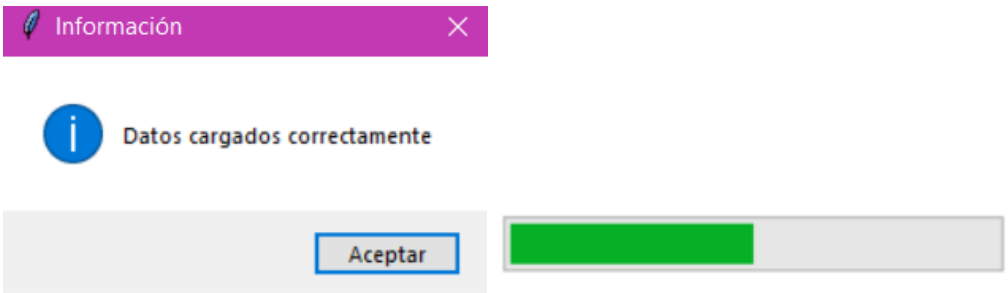
En esta interfaz, el usuario tiene la posibilidad de cargar archivos desde la computadora para convertirlos en formato Braille. Los elementos principales en el módulo, son:

- **Área de carga de archivos:** En el centro de la pantalla se evidencia un ícono de carga, que al presionarla se puede subir un archivo desde el dispositivo.
- **Estado del archivo:** Abajo del ícono se muestra el texto que indica si un documento ha sido seleccionado o no.

- **Botones:** Se encuentran las opciones de convertir, imprimir y cancelar.
    - **Convertir:** Comienza el proceso de traducción del archivo en formato Braille.
    - **Imprimir:** Permite enviar de manera directa el documento traducido a una impresora compatible.
    - **Cancelar:** Cancela la opción y regresa al estado inicial de la interfaz.



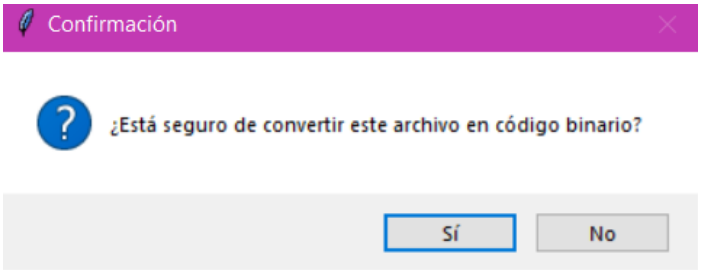
Para cargar un nuevo archivo, debe presionar en el espacio “Click aquí para cargar un archivo”, posteriormente, se abrirá el explorador de archivos donde se podrá seleccionar documentos en PDF o extensión DOCX.



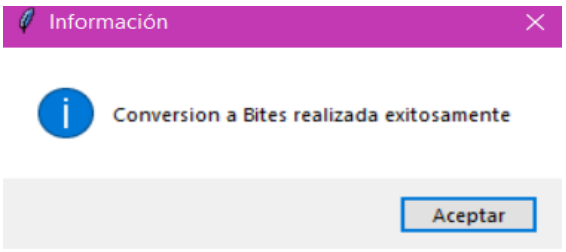
Una vez que se seleccione el archivo correspondiente, se mostrará en la pantalla la alerta “Datos cargados correctamente” y una barra de progreso en la parte superior que indica que el proceso fue exitoso.

Luego de cargarse el archivo, se puede visualizar la información del mismo en la pantalla, con el nombre del documento y la extensión correspondiente. El usuario puede escoger entre dos opciones: traducir todo el documento o personalizar el rango de páginas.

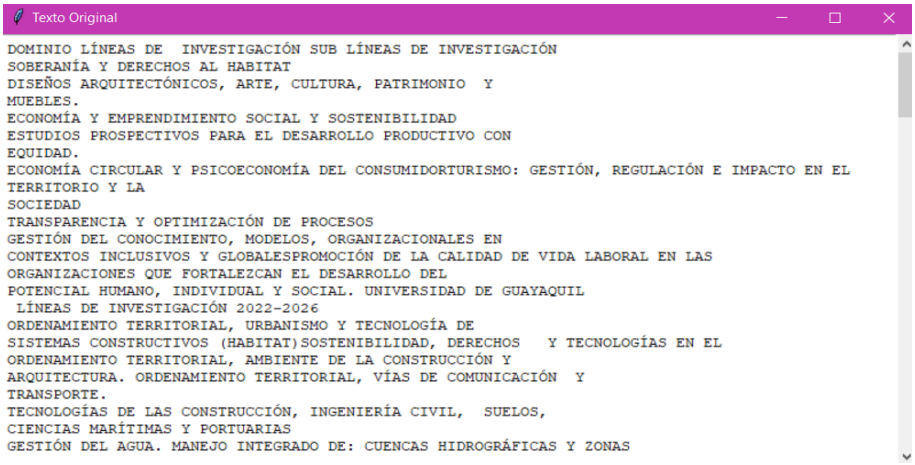
Si el usuario escoge la segunda opción, deberá señalar la página de inicio y fin que requiere traducir a Braille, para posteriormente dar clic en el botón “Convertir”.



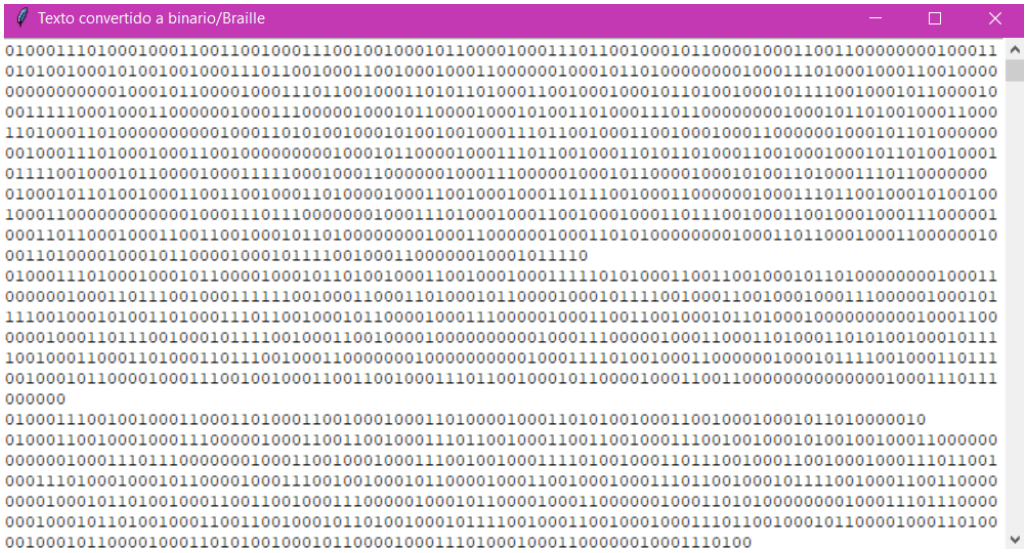
Aparecerá la siguiente confirmación, donde el usuario puede seguir con el proceso dando clic en “Sí”, o si desea cancelar la acción, deberá presionar “No”.



Si la confirmación es exitosa, se mostrará la siguiente alerta que revela que la conversión a Bites fue realizada exitosamente. Además, se cargan dos ventanas adicionales, una con el texto original del documento seleccionado y otra con el código binario.



Se cargará la ventana evidenciando el texto original del archivo seleccionado, reflejando que el documento se transformó correctamente.



De la misma forma, se puede visualizar la transformación del archivo a código binario, lo que indica un proceso satisfactorio del software traductor a Braille.

Consejos o buenas prácticas

- **Corrobora el formato del documento antes de cargarlo:** Es importante asegurarse que el documento que selecciones se encuentre en formato Docx o Pdf, ya que, otros archivos son incompatibles con el software y podrían generar algún error en el proceso de conversión.
- **Supervisa la calidad del texto:** Antes de cargar el documento, comprueba que el texto no cuente con errores tipográficos o figuras que puedan interferir con el proceso de conversión, ya que, el sistema funciona de mejor forma con un texto claro y limpio.
- **Emplea archivos simples para mejores resultados:** Si el archivo posee gráficos, tablas complejas o columnas, intenta utilizar un formato más simple para asegurar que el texto se convierta de manera correcta, puesto que, los elementos complejos puede que no se traduzcan bien a Braille.
- **Personaliza el rango de páginas cuando lo requieras:** Si solo se necesita una sección específica del archivo, se debe utilizar la alternativa de personalizar el rango de páginas, permitiendo ahorrar tiempo y recursos, centrándose solamente en la parte que se requiere.
- **Almacena el archivo original antes de la conversión:** Se debe almacenar una copia siempre antes de comenzar con el proceso de conversión, ya que,

en caso de necesitar realizar ajustes o corregir posibles errores. Se tendrá el archivo original disponible.

- **Inspecciona el documento traducido:** Posterior al proceso de conversión, supervisa que el documento que se tradujo para asegurar que todo se encuentre de manera correcta en código Braille.
- **Impresiones:** Para la impresión del archivo, se debe ajustar el equipo y colocar los medios necesarios para que todo sea satisfactorio.
- **Pedir ayuda:** En caso de existir problemas o fallos durante cualquier proceso que conlleva el sistema, se debe contactar con los encargados del software para las orientaciones requeridas.