|  |
| --- |
| DUOC UC - ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES |
| Especificación de Requisitos de Software |
| Proyecto: CatCh AI |
| Logotipo  El contenido generado por IA puede ser incorrecto. |
|  |
| **Revisión*: 1.0*** |
| **fecha: 14/09/2025** |

Contenido

[Ficha del documento 3](#_Toc208795303)

[1. Introducción 4](#_Toc208795304)

[1.1. Propósito 4](#_Toc208795305)

[1.2. Ámbito del Sistema 4](#_Toc208795306)

[1.3. Definiciones, Acrónimos y Abreviaturas 5](#_Toc208795307)

[1.4. Referencias 6](#_Toc208795308)

[1.5. Visión General del Documento 6](#_Toc208795309)

[2. Descripción General 7](#_Toc208795310)

[2.1. Perspectiva del Producto 7](#_Toc208795311)

[2.2. Funciones del Producto 8](#_Toc208795312)

[2.3. Características de los Usuarios 8](#_Toc208795313)

[2.4. Restricciones 9](#_Toc208795314)

[2.5. Suposiciones y Dependencias 9](#_Toc208795315)

[2.6. Requisitos Futuros 10](#_Toc208795316)

[3. Requisitos Específicos 10](#_Toc208795317)

[3.1 Requisitos comunes de las interfaces 10](#_Toc208795318)

[3.1.1 Interfaces de usuario 11](#_Toc208795319)

[3.1.2 Interfaces de hardware 11](#_Toc208795320)

[3.1.3 Interfaces de software 12](#_Toc208795321)

[3.1.4 Interfaces de comunicación 12](#_Toc208795322)

[3.2 Requisitos funcionales 14](#_Toc208795323)

[3.2.1 Requisito funcional 1 14](#_Toc208795324)

[3.2.2 Requisito funcional 2 14](#_Toc208795325)

[3.2.3 Requisito funcional 3 14](#_Toc208795326)

[3.2.4 Requisito funcional 4 15](#_Toc208795327)

[3.2.5 Requisito funcional 5 15](#_Toc208795328)

[3.2.6 Requisito funcional 6 15](#_Toc208795329)

[3.2.7 Requisito funcional 7 16](#_Toc208795330)

[3.2.8 Requisito funcional 8 16](#_Toc208795331)

[3.3 Requisitos no funcionales 17](#_Toc208795332)

[3.3.1 Requisitos de rendimiento 17](#_Toc208795333)

[3.3.2 Seguridad 17](#_Toc208795334)

[3.3.3 Fiabilidad 18](#_Toc208795335)

[3.3.4 Usabilidad 18](#_Toc208795336)

[3.3.5 Mantenibilidad 19](#_Toc208795337)

[3.3.6 Portabilidad 20](#_Toc208795338)

[3.4 Otros Requisitos 20](#_Toc208795339)

|  |
| --- |
| Especificación de Requisitos según estándar de IEEE 830. |

# Ficha del documento

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Revisión** | **Autor** | **Modificación** |
| *14/09/2025* | *1.0* | *Eyleen Collado* | *Desarrollo de la propuesta de proyecto y especificación de requerimientos de software* |
| *15/09/2025* | *2.0* | *Eyleen Collado* | *Actualización de Requerimientos funcionales* |

# 1. Introducción

La presente Especificación de Requerimientos de Software (ERS) documenta en detalle los requisitos esenciales y funcionales para el desarrollo del sistema **CatchAI**, un producto mínimo viable (MVP) orientado a la inclusión social de personas sordas en Chile. El sistema busca traducir voz hablada a Lengua de Señas Chilena (LSCh), representando las señas mediante un avatar 3D animado.

Este documento tiene como propósito principal establecer un marco de referencia que oriente el diseño, desarrollo, implementación y validación del software. Además, servirá como contrato entre las partes interesadas, definiendo claramente qué debe cumplir el sistema, bajo qué restricciones técnicas y legales, y cómo se medirá su desempeño.

El sistema no busca reemplazar a intérpretes humanos certificados, sino constituirse como una herramienta de apoyo en situaciones de comunicación masiva, principalmente en contextos educativos, comunicacionales y de transmisión televisiva.

## 1.1. Propósito

El propósito de este documento es detallar los requisitos funcionales, no funcionales, de interfaz y de operación del sistema CatchAI. De este modo, se establecerán las bases necesarias para el ciclo de vida del desarrollo del software, asegurando la alineación con los objetivos del proyecto: entregar un prototipo funcional que traduzca al menos 50 frases básicas de voz a LSCh con una precisión mínima del 85%.

## 1.2. Ámbito del Sistema

El sistema, denominado CatchAI, se concibe como una herramienta tecnológica orientada a mejorar la inclusión comunicacional de las personas sordas en Chile. Su función principal consiste en recibir como entrada un archivo de video, una pista de audio o un flujo en vivo, procesar la señal de voz contenida, transcribirla a texto mediante un motor de reconocimiento automático, y finalmente consultar un diccionario digital de Lengua de Señas Chilena (LSCh) para producir la seña correspondiente. El resultado se materializa en un archivo o transmisión de video en el que un avatar 3D reproduce los gestos de la seña. En aquellos casos en que no exista una seña registrada en el diccionario, el sistema aplicará un mecanismo de deletreo manual, asegurando que ninguna palabra quede sin representación.

La propuesta busca responder a una necesidad concreta: la falta de herramientas tecnológicas que permitan a las personas sordas acceder en tiempo real a la información transmitida en espacios como la televisión, las aulas o servicios públicos. Al mismo tiempo, CatchAI sienta las bases para futuras extensiones que incrementen el vocabulario, incorporen expresiones faciales o integren modalidades bidireccionales de traducción.

Los beneficios del sistema se pueden sintetizar en cuatro ejes principales:

* Accesibilidad e inclusión: CatchAI constituye un recurso que facilita el acceso a la información hablada para personas sordas, promoviendo la equidad comunicacional en entornos educativos, mediáticos y sociales.
* Estandarización tecnológica: El desarrollo del sistema ofrece un modelo replicable que puede ser adoptado y adaptado en otros países o contextos, favoreciendo la creación de un estándar tecnológico en la traducción automática de voz a señas.
* Eficiencia comunicacional: Si bien no busca reemplazar a los intérpretes humanos certificados, CatchAI ofrece una alternativa rápida y automatizada en situaciones donde no se dispone de estos profesionales, ampliando así el acceso a la información.
* Escalabilidad: El diseño modular del sistema permite su crecimiento progresivo, tanto en el aumento del vocabulario como en la naturalidad de las animaciones y la expresividad del avatar, asegurando que la herramienta pueda evolucionar más allá del prototipo inicial.

En esta etapa la animación se visualizará en Blender/Godot y podrá ser capturada mediante herramientas externas como OBS. De este modo, se asegura la validación del flujo completo (voz → texto → seña → animación) sin necesidad de contar aún con un módulo de render automático.

A futuro, se proyecta que la API entregue como salida un archivo o flujo de video (MP4, WebM, RTMP/WebRTC), pero este objetivo queda reservado para fases posteriores.

## 1.3. Definiciones, Acrónimos y Abreviaturas

API (Application Programming Interface): Conjunto de reglas y definiciones que permiten que un software se comunique con otro. En CatchAI, se usa para recibir video/audio y devolver la salida en video con avatar.

Avatar: Personaje digital tridimensional que representa las señas en Lengua de Señas Chilena (LSCh).

CatchAI: Nombre del sistema desarrollado como proyecto de título, orientado a la traducción automática de voz a LSCh.

Deletreo manual: Estrategia de traducción usada cuando no existe una seña específica para una palabra, reproduciendo letra por letra en LSCh.

Diccionario LSCh: Base de datos que asocia palabras o frases con animaciones digitales que representan señas chilenas.

Godot Engine: Motor de videojuegos y animaciones utilizado para renderizar al avatar en este proyecto.

LSCh (Lengua de Señas Chilena): Lengua natural utilizada por la comunidad sorda en Chile, distinta del español escrito y hablado.

MVP (Minimum Viable Product): Producto mínimo viable. Primera versión funcional del sistema, centrada en un conjunto reducido de funciones para validar su factibilidad.

STT (Speech-to-Text): Tecnología de reconocimiento automático de voz que convierte audio en texto.

Whisper: Modelo de reconocimiento de voz desarrollado por OpenAI, utilizado en CatchAI para transcribir audio en español.

CITT: Centro de Innovación y Transferencia Tecnológica de Duoc UC.

## 1.4. Referencias

Ley N° 20.422: Establece normas sobre igualdad de oportunidades e inclusión social de personas con discapacidad en Chile.

IEEE 830: Estándar de especificación de requisitos de software, utilizado como base estructural de este documento.

Documentos internos del proyecto: Informe de Avance Técnico, Fase 1 de Definición del Proyecto, y minutas de reuniones con Canal 13C.

Recursos técnicos externos: Documentación de Whisper (OpenAI), Godot Engine y FreeMoCap, utilizados como referencia técnica para el desarrollo del sistema.

## 1.5. Visión General del Documento

La Especificación de Requisitos de Software (ERS) del sistema CatchAI – Traducción Automática de Voz a Lengua de Señas Chilena (LSCh) tiene como propósito establecer un marco claro, exhaustivo y verificable de todas las funcionalidades, limitaciones y condiciones bajo las cuales se desarrollará el software. El documento no solo actúa como una guía técnica para el equipo de desarrollo, sino también como un acuerdo formal entre los distintos actores involucrados: desarrolladores, docentes supervisores, instituciones interesadas y, en una fase posterior, representantes de la comunidad sorda chilena.

Cada sección de este documento está diseñada para abordar de manera integral un aspecto específico del sistema: desde la perspectiva y beneficios generales del producto, pasando por las funciones concretas que deberá cumplir, hasta las restricciones técnicas, operativas y legales que condicionarán su implementación. El nivel de detalle incluido permitirá asegurar que el sistema no solo cumpla con su propósito mínimo viable (MVP), sino que siente bases sólidas para futuras iteraciones más avanzadas.

En este sentido, el documento contempla no solo los requisitos inmediatos del proyecto académico, sino también los posibles escenarios de crecimiento del sistema. Se ha puesto especial énfasis en la trazabilidad de los requisitos, de manera que cada funcionalidad descrita pueda relacionarse con un objetivo de inclusión digital y, posteriormente, con pruebas verificables de desempeño y calidad.

# 2. Descripción General

En esta sección se ofrece una visión completa del sistema CatchAI, abarcando los factores que influyen en su diseño y desarrollo. La descripción general contextualiza al producto en su entorno, identifica las funciones clave que ofrecerá, define los perfiles de usuarios que interactuarán con él, y establece las restricciones, suposiciones y dependencias que condicionan su implementación. Asimismo, se anticipan requisitos futuros que permitirán la evolución del sistema más allá de la etapa inicial de MVP.

La finalidad de esta sección es situar al lector en un panorama amplio que no se limite a lo técnico, sino que también considere aspectos sociales, educativos y culturales, fundamentales para la pertinencia de un proyecto que aborda la accesibilidad de la comunidad sorda.

## 2.1. Perspectiva del Producto

CatchAI se concibe como un sistema de software independiente, diseñado inicialmente como un MVP, que deberá integrarse de manera transparente con entornos académicos y comunicacionales. El producto tendrá como entradas principales la voz capturada en tiempo real y como salida un avatar 3D que representará la seña correspondiente en Lengua de Señas Chilena (LSCh).

El sistema no busca reemplazar intérpretes humanos, sino constituirse como un apoyo complementario en contextos donde no exista cobertura suficiente de intérpretes certificados o donde se requiera rapidez en la traducción. Este enfoque coloca a CatchAI como una herramienta inclusiva que responde a la necesidad de accesibilidad digital, en concordancia con la Ley N° 20.422 en Chile.

El sistema se integrará mediante cuatro componentes principales: captura de voz, transcripción de voz a texto, procesamiento semántico y visualización de la seña. Cada uno de estos módulos deberá comunicarse a través de APIs internas, permitiendo modularidad y escalabilidad.

## 2.2. Funciones del Producto

El sistema CatchAI ofrecerá las siguientes funciones clave:

1. Captura de voz: El sistema deberá permitir que la voz del usuario sea capturada en tiempo real a través de un micrófono estándar.

2. Transcripción: La voz capturada se convertirá en texto mediante un modelo de reconocimiento automático de voz, como Whisper, con tiempos de respuesta adecuados.

3. Consulta de diccionario LSCh: Una vez transcrito el texto, el sistema consultará un diccionario de señas chilenas para identificar la seña correspondiente.

4. Animación con avatar 3D: Si la palabra existe en el diccionario, se reproducirá la seña mediante un avatar animado en Godot. Si no existe, se aplicará la estrategia de deletreo manual.

5. Registro de actividades: El sistema deberá almacenar logs que incluyan la entrada de voz, el resultado de la transcripción, la seña utilizada y posibles errores.

6. Captura de animaciones: Se capturará las animaciones como señal de video, integrable en medios de comunicación.

## 2.3. Características de los Usuarios

El sistema CatchAI está pensado para distintos perfiles de usuarios:

- Usuarios sordos: Personas cuyo idioma principal o preferente es la Lengua de Señas Chilena y que constituyen el público objetivo principal.

- Locutores y educadores: Profesores, periodistas u oradores que deseen que su discurso sea traducido a señas de forma automática.

- Administradores técnicos: Personal encargado de configurar, mantener y supervisar el funcionamiento del sistema, asegurando la actualización de vocabulario y la continuidad operativa.

Cada uno de estos perfiles tiene necesidades y niveles de interacción diferentes, lo que implica diseñar un sistema que sea inclusivo y adaptable.

## 2.4. Restricciones

El desarrollo de CatchAI estará sujeto a las siguientes restricciones:

* Traducción unidireccional: Solo se contempla la traducción de voz a seña, sin interacción en sentido inverso.
* Latencia máxima: El sistema deberá mantener un tiempo de respuesta no mayor a 10 segundos para asegurar fluidez.
* Vocabulario limitado: La primera versión estará restringida a 50 frases o palabras básicas.
* Dependencia de Internet: El procesamiento en la nube requiere conexión estable.
* Limitaciones de hardware: El MVP operará con equipos sin GPU dedicada, lo cual afectará la precisión y rapidez inicial.
* La expresividad facial y los matices emocionales del avatar estarán limitados en el MVP.
* La precisión de la transcripción dependerá de la calidad del audio de entrada.
* El sistema no reemplaza a intérpretes humanos certificados.

## 2.5. Suposiciones y Dependencias

El desarrollo de CatchAI se apoya en una serie de supuestos y dependencias que condicionan su alcance y factibilidad en la etapa actual. En primer lugar, el sistema será implementado en un entorno local, sin el uso de servicios en la nube, debido a la falta de recursos económicos y técnicos para mantener infraestructura externa. Esto implica que toda la operación dependerá de la capacidad de los equipos de cómputo y cámaras disponibles en el CITT.

El MVP estará limitado a un vocabulario reducido de 50 palabras básicas en Lengua de Señas Chilena (LSCh), lo cual responde a la dificultad de construir un diccionario amplio sin contar con suficientes grabaciones ni con un número adecuado de participantes para registrar y validar señas. El equipo carece de experiencia formal en LSCh, lo que representa una dependencia crítica de actores externos —intérpretes o miembros de la comunidad sorda— para validar la fidelidad lingüística de las señas registradas. La participación de estos usuarios se contempla en pruebas piloto, pero no se puede asegurar su disponibilidad permanente debido a limitaciones logísticas.

Asimismo, el avatar digital utilizado será de mediana complejidad, con características básicas de uniformidad y vestimenta semi-formal. Esta decisión responde a la carencia de recursos gráficos avanzados y a la imposibilidad de destinar tiempo y herramientas a un diseño más elaborado. Finalmente, la creación del diccionario de señas depende exclusivamente de los recursos propios del equipo, es decir, las cámaras personales y el software de captura y animación disponibles, lo que restringe la calidad y cantidad de datos recopilados en esta etapa inicial.

## 2.6. Requisitos Futuros

Aunque el MVP de CatchAI se centra en un vocabulario limitado y enfrenta restricciones significativas de recursos, el sistema está pensado como una base escalable que permita su crecimiento en fases posteriores. Entre los principales requisitos futuros se consideran:

* Ampliación del vocabulario y expresiones idiomáticas: Una vez que se logre consolidar el diccionario inicial de 50 palabras, el siguiente paso será incrementar progresivamente el número de señas registradas, incorporando vocabulario técnico, académico y de uso cotidiano.
* Incorporación de expresiones faciales y emocionales en el avatar: Actualmente el avatar es básico, pero en el futuro se espera dotarlo de mayor naturalidad para representar adecuadamente la expresividad que caracteriza a la LSCh.
* Traducción bidireccional (voz ↔ seña): A largo plazo, el sistema no solo deberá traducir voz a señas, sino también permitir que las señas registradas por usuarios sordos se conviertan en texto o voz sintetizada, ampliando así las posibilidades de comunicación.
* Integración con plataformas educativas y gubernamentales: El objetivo final es que CatchAI pueda ser adoptado en aulas, instituciones públicas o medios de comunicación, facilitando el acceso inclusivo a la información.
* Validación con intérpretes certificados y la comunidad sorda: La falta de conocimientos profundos en LSCh por parte del equipo es una limitación en la etapa inicial. En fases futuras será indispensable trabajar junto a intérpretes y organizaciones de la comunidad sorda para validar el diccionario y garantizar precisión lingüística.
* Implementación de módulos de exportación de videos: Esto permitirá exportar las animaciones generadas como un video, lo que facilitará la implementación a diferentes plataformas con mayor facilidad.

Estos requisitos futuros muestran que, si bien la primera versión se limita a un prototipo académico con recursos mínimos, el sistema tiene el potencial de evolucionar hacia una herramienta robusta y con impacto social real, siempre que se logre la colaboración con especialistas y se cuente con recursos adicionales para su crecimiento.

# 3. Requisitos Específicos

En esta sección se detallan todos los requisitos que deberá cumplir el sistema CatchAI – Traducción Automática de Voz a Lengua de Señas Chilena (LSCh). Los requisitos se dividen en funcionales, no funcionales, de interfaz y otros requerimientos adicionales. La descripción será exhaustiva, de manera que cada requisito quede claramente definido, justificado y, en lo posible, asociado a criterios de verificación y aceptación.

## 3.1 Requisitos comunes de las interfaces

Esta sección se centra en los requisitos esenciales que guían cómo CatCh AI interactúa con usuarios, hardware y software. Aquí, se describen los detalles clave sobre cómo se diseñarán las interfaces para asegurar un funcionamiento eficiente y satisfactorio del sistema. Esto incluye la experiencia del usuario, la conectividad con dispositivos, la integración con otros programas y los protocolos de comunicación.

### 3.1.1 Interfaces de usuario

El sistema CatchAI no contará con una interfaz gráfica convencional para usuarios finales, dado que su naturaleza es la de un servicio de procesamiento. La interacción no será directa por parte de un usuario común, sino a través de una página web HTML básica, que permitirá cargar archivos de audio o video, o bien grabar desde micrófono, para enviarlos al sistema.

En lugar de botones o menús avanzados, esta página funcionará como punto de acceso inicial. El procesamiento interno se encargará de transcribir la voz, consultar el diccionario de señas y reproducir la animación correspondiente en Blender o Godot, la cual podrá ser visualizada en tiempo real y, si se requiere, capturada mediante OBS para obtener evidencia en video.

Desde esta perspectiva, la “interfaz de usuario” no está pensada para interacción final con el público sordo, sino como un canal de integración y prueba que asegura el flujo completo de trabajo. En el futuro, el mismo sistema se extenderá hacia una API robusta capaz de exportar y transmitir videos en formatos estándar (ej. MP4, WebM) y, en casos de transmisión en vivo, mediante protocolos como RTMP o WebRTC.

El usuario final, es decir, la persona sorda que recibe la traducción, no interactúa directamente con el sistema en esta fase inicial. Accederá a los resultados a través de los videos generados y difundidos por los operadores, ya sea en entornos educativos, televisivos o de demostración técnica.

### 3.1.2 Interfaces de hardware

El sistema CatchAI no requiere un hardware especializado para usuarios finales, ya que la entrada principal se gestionará a través de una página web HTML que permitirá cargar archivos de video o audio, o bien grabar audio desde un micrófono convencional. Los requisitos de hardware se concentran en el equipo de procesamiento local, encargado de ejecutar la transcripción y la animación del avatar.

Fuentes de entrada:

* Archivos de video en formatos estándar (MP4, MKV, AVI, WebM).
* Pistas de audio en formatos como WAV, MP3 o AAC.
* Audio capturado en tiempo real desde un micrófono, a través de la página web.

Entorno de procesamiento:

* Estación de trabajo local con capacidad de extraer audio de archivos de video (herramientas como FFmpeg).
* CPU suficiente para ejecutar el pipeline de transcripción (Whisper) y animación en Blender/Godot.
* GPU opcional para optimizar la previsualización de animaciones y, en fases futuras, acelerar renderizado o entrenamiento de modelos.

En este contexto, el micrófono no constituye un requisito obligatorio del sistema, pero sí una opción habilitada mediante la interfaz web. La salida en esta fase inicial no se genera como archivo automático, sino como animación en Blender/Godot, que puede ser registrada con software de captura como OBS Studio.

### 3.1.3 Interfaces de software

El sistema CatchAI se apoyará en un conjunto de aplicaciones y librerías que permiten implementar el flujo completo de procesamiento, desde la recepción de entradas hasta la reproducción de la animación.

Componentes principales:

* Página web HTML: interfaz simple que permite a los usuarios cargar archivos de video o audio, o grabar audio desde micrófono para su envío al sistema.
* Whisper (OpenAI, ejecución local): modelo de reconocimiento de voz utilizado para transcribir el audio a texto.
* Diccionario LSCh: base de datos digital con las señas disponibles, que permite mapear palabras o frases a animaciones predefinidas.
* Blender/Godot: entornos gráficos donde se carga el avatar 3D y se reproducen las animaciones correspondientes a las señas identificadas.
* OBS Studio (uso auxiliar): software externo que permite capturar la animación mostrada en Blender/Godot y generar un archivo de video como evidencia.

Compatibilidad y extensibilidad:

El sistema está diseñado para operar en entornos locales, utilizando software de libre acceso. En versiones futuras, se prevé complementar este flujo con una API de salida en video, que permita entregar directamente archivos o transmisiones en vivo en formatos estándar (MP4, WebM, RTMP/WebRTC).

### 3.1.4 Interfaces de comunicación

El sistema CatchAI se comunica principalmente a través de una interfaz web HTML, que actúa como punto de entrada para cargar archivos de audio o video, o para grabar audio desde un micrófono. Estos datos se envían al motor de procesamiento local, el cual realiza la transcripción, consulta el diccionario de señas y reproduce la animación correspondiente en Blender/Godot.

En esta etapa inicial, la salida del sistema no incluye respuestas en JSON ni archivos de video generados automáticamente. La única salida disponible es la visualización en tiempo real de la animación en el motor gráfico. Cuando se requiera conservar evidencia, esta animación podrá ser capturada con herramientas externas como OBS Studio.

En futuras versiones, el sistema podrá ampliar sus interfaces de comunicación para:

* Entregar respuestas en JSON con información del estado, texto transcrito y seña identificada.
* Proveer archivos de video exportados automáticamente en formatos estándar (MP4, WebM).
* Ofrecer transmisión en vivo mediante protocolos como RTMP o WebRTC, para su integración en plataformas educativas y televisivas.

## 3.2 Requisitos funcionales

A continuación, se detallan los principales requisitos funcionales del sistema.

## 3.2.1 Requisito funcional 1

**Nombre: RF. 1 Ingesta de video**

El sistema deberá proporcionar un endpoint /process/video que permita enviar un archivo de video en formatos estándar (MP4, MKV, AVI, WebM).

* Descripción: El sistema deberá extraer de manera automática la pista de audio del video, normalizarla y enviarla al módulo de transcripción.
* Entrada esperada: Video de hasta 1 hora de duración, con audio en español estándar.
* Salida: JSON con estado del procesamiento y URL del video resultante con el avatar.
* Criterio de aceptación: El sistema procesa al menos el 90% de los videos cargados sin errores de compatibilidad.

## 3.2.2 Requisito funcional 2

**Nombre: RF.2 Ingesta de pista de audio**

El sistema deberá proporcionar un endpoint /process/audio que reciba archivos de audio en formatos como WAV, MP3 o AAC.

* Descripción: El sistema procesará directamente la pista de audio sin necesidad de extraerla de un video.
* Entrada esperada: Archivo de audio con voz clara y en español.
* Salida: Video generado con avatar en LSCh correspondiente a la transcripción.
* Criterio de aceptación: El sistema devuelve un archivo de video con avatar en menos de 10 segundos para audios menores a 60 segundos.

## 3.2.3 Requisito funcional 3

**Nombre: RF.3 Procesamiento de transmisión en vivo**

El sistema deberá proporcionar un endpoint /stream que permita recibir un flujo de audio en vivo mediante protocolos RTMP o WebRTC.

* Descripción: El sistema deberá procesar el audio recibido en tiempo real, transcribirlo y devolver una transmisión de video en vivo del avatar ejecutando las señas.
* Entrada esperada: Flujo continuo de audio desde micrófono de estudio o transmisión online.
* Salida: Flujo de video en tiempo real con el avatar en LSCh.
* Criterio de aceptación: La latencia entre audio recibido y seña generada no debe superar los 10 segundos.

## 3.2.4 Requisito funcional 4

**Nombre: RF.4 Transcripción automática**

El sistema deberá transcribir el audio recibido (desde video, pista o flujo en vivo) utilizando el motor Whisper u otro STT.

* Descripción: El resultado de la transcripción se devolverá como objeto JSON con el texto reconocido, porcentaje de confianza y timestamps de inicio y fin.
* Entrada esperada: Audio en español claro, hasta con ruido moderado.
* Salida: Objeto JSON con transcripción y confianza ≥ 0.85.
* Criterio de aceptación: El sistema logra al menos 85% de precisión en frases básicas del diccionario definido.

## 3.2.5 Requisito funcional 5

**Nombre: RF.5 Interpretación semántica con diccionario LSCh**

El sistema deberá consultar un diccionario digital de señas chilenas para encontrar la animación correspondiente a la palabra transcrita.

* Descripción: Si la palabra está en el diccionario, devuelve la animación asociada; si no, invoca el módulo de deletreo manual.
* Entrada esperada: Texto transcrito.
* Salida: Identificador de animación en formato GLB/FBX.
* Criterio de aceptación: El sistema debe encontrar coincidencia en al menos el 95% de las palabras del vocabulario inicial (50 frases).

## 3.2.6 Requisito funcional 6

**Nombre: RF.6 Generación de animación de avatar**

El sistema deberá reproducir la animación asociada a la seña mediante el motor gráfico Blender o Godot.

* Descripción: En esta versión inicial, la animación se mostrará únicamente en tiempo real dentro del motor gráfico. No se generará automáticamente un archivo de video; en caso de requerirse evidencia, podrá capturarse con software externo como OBS Studio.
* Entrada esperada: Identificador de animación proveniente del diccionario de señas.
* Salida: Animación en tiempo real visualizada en Blender/Godot.
* Criterio de aceptación: El avatar debe ejecutar señas comprensibles en más del 90% de los casos, validadas en pruebas internas.

## 3.2.7 Requisito funcional 7

**Nombre: RF.7 Registro y logging**

* Descripción: El log incluirá fecha, tipo de entrada (video, audio, stream), texto transcrito, seña ejecutada, confianza del modelo y estado del procesamiento.
* Entrada esperada: Cualquier interacción con la API.
* Salida: Registro estructurado en base de datos o archivo seguro.
* Criterio de aceptación: El 100% de las solicitudes deben quedar registradas con información trazable.

## 3.2.8 Requisito funcional 8

**Nombre: RF.8 Respuesta del sistema**

El sistema deberá entregar el resultado del procesamiento en la forma definida para el MVP.

* Descripción: En esta primera versión, no se generarán respuestas en formato JSON ni archivos de video automáticos. La salida se limita a la animación mostrada en Blender/Godot. En fases futuras, se incorporará una respuesta JSON con información de estado, transcripción, seña identificada y enlace al video generado.
* Entrada esperada: Solicitud de procesamiento desde la interfaz web HTML (audio, video o micrófono).
* Salida: Animación visualizada en tiempo real dentro del motor gráfico.
* Criterio de aceptación: La animación debe reproducirse sin errores en más del 95% de los casos de prueba internos.

## 3.3 Requisitos no funcionales

Los requisitos no funcionales de CatchAI establecen las características de calidad y condiciones de operación que el sistema deberá cumplir más allá de sus funciones principales. Estos requisitos garantizan que el sistema no solo haga “qué” debe hacer, sino que lo haga de forma confiable, segura, accesible y mantenible.

### 3.3.1 Requisitos de rendimiento

**RNF.01 Tiempo de respuesta en entorno local**

* Descripción: El sistema deberá procesar archivos de video o audio de hasta 1 minuto y entregar el video con avatar en un tiempo máximo de 10 segundos.
* Respuesta: CatchAI utilizará un pipeline optimizado con extracción directa de audio y transcripción eficiente mediante Whisper en CPU local, reduciendo pasos intermedios.
* Medida de Respuesta: En pruebas de laboratorio, al menos el 95% de los archivos de ≤1 minuto deberán generar su salida en ≤10 segundos.

**RNF.02 Latencia en transmisión en vivo**

* Descripción: En flujos en vivo, la latencia entre la voz recibida y la seña animada no deberá superar 10 segundos.
* Respuesta: CatchAI procesará fragmentos de audio en lotes pequeños (chunks) de 5 a 10 segundos para reducir el tiempo de espera acumulado.
* Medida de Respuesta: Durante simulaciones de transmisión, la diferencia entre el audio recibido y la salida en video deberá medirse y mantenerse ≤10 segundos en al menos el 90% de los casos.

### 3.3.2 Seguridad

**RNF.05 Protección de archivos procesados**

* Descripción: Los archivos de entrada y salida deberán almacenarse de forma temporal en directorios protegidos.
* Respuesta: CatchAI utilizará permisos de sistema operativo para limitar accesos a carpetas de trabajo, eliminando archivos temporales tras 24 horas.
* Medida de Respuesta: En inspecciones de servidor, los directorios de trabajo deberán mostrar permisos restringidos y ausencia de archivos temporales antiguos.

**RNF.06 Gestión de logs de auditoría**

* Descripción: Cada solicitud deberá generar un registro con datos de entrada, salida y estado.
* Respuesta: CatchAI generará logs en formato estructurado (JSON/CSV) y los almacenará cifrados localmente.
* Medida de Respuesta: Los registros deberán revisarse en auditorías internas, verificando consistencia en el 100% de las solicitudes.

### 3.3.3 Fiabilidad

**RNF.03 Procesamiento continuo de solicitudes**

* Descripción: El sistema deberá ser capaz de procesar al menos 10 solicitudes consecutivas sin caídas ni reinicios.
* Respuesta: El diseño modular permitirá encolar solicitudes y procesarlas secuencialmente, evitando sobrecargas en la memoria.
* Medida de Respuesta: En pruebas internas, se ejecutarán 10 solicitudes consecutivas y se verificará que todas produzcan una salida válida.

**RNF.04 Manejo de errores y recuperación automática**

* Descripción: Ante entradas inválidas (ej. formatos no soportados), el sistema deberá notificar el error y continuar funcionando.
* Respuesta: CatchAI validará los formatos de entrada en la etapa de ingestión y devolverá un mensaje de error estandarizado en JSON sin detener el servicio.
* Medida de Respuesta: En pruebas de estrés con archivos no soportados, el sistema deberá devolver respuesta con "status": "ERROR" y permitir que nuevas solicitudes sean atendidas sin reinicio.

### 3.3.4 Usabilidad

**RNF.07 Documentación de integración de la API**

* Descripción: La API deberá contar con documentación clara y accesible para desarrolladores.
* Respuesta: CatchAI dispondrá de un manual en PDF/Markdown con ejemplos de uso en Python, además de un diagrama de flujo de los procesos.
* Medida de Respuesta: Validación con pruebas de integración: un desarrollador externo deberá poder usar la API en ≤2 días con la documentación entregada.

**RNF.08 Estándar de respuestas unificado**

* Descripción: Todas las respuestas deberán cumplir con el formato JSON establecido en RF.08.
* Respuesta: Los endpoints de CatchAI estarán programados para devolver siempre un objeto JSON con campos obligatorios (status, message, metadata).
* Medida de Respuesta: En pruebas de validación, el 100% de las respuestas (éxito o error) deberán cumplir el formato esperado.

### 3.3.5 Mantenibilidad

**RNF.09 Modularidad de componentes internos**

* Descripción: Cada módulo del sistema deberá ser independiente.
* Respuesta: CatchAI dividirá el pipeline en módulos (ingestión, STT, diccionario, animación) conectados mediante APIs internas.
* Medida de Respuesta: En pruebas de reemplazo, deberá poder sustituirse un módulo sin alterar el resto del sistema.

**RNF.10 Registro de versiones del avatar y diccionario**

* Descripción: El sistema deberá registrar la versión del avatar y del diccionario utilizados en cada procesamiento.
* Respuesta: Los metadatos generados incluirán campos avatar\_version y dictionary\_version.
* Medida de Respuesta: Cada video procesado deberá contener dicha información en el JSON de salida.

### 3.3.6 Portabilidad

**RNF.11 Compatibilidad multiplataforma**

* Descripción: El sistema deberá ejecutarse en Windows 10/11 y Linux Ubuntu 22.04 LTS.
* Respuesta: CatchAI se distribuirá mediante contenedores Docker opcionales o scripts de instalación documentados.
* Medida de Respuesta: El sistema deberá instalarse y funcionar correctamente en ambos entornos en pruebas de laboratorio.

**RNF-12 – Reproducibilidad de resultados en distintos equipos**

* Descripción: El sistema deberá generar salidas idénticas en diferentes equipos al procesar el mismo archivo de entrada.
* Respuesta: CatchAI procesará con parámetros estandarizados y dependencias fijas para garantizar resultados consistentes.
* Medida de Respuesta: En pruebas de validación cruzada, dos estaciones deberán producir archivos de salida binariamente iguales a partir de la misma entrada.

## 3.4 Otros Requisitos

**RNF-13 – Validación lingüística de señas**

* Descripción: Las señas reproducidas por el avatar deberán representar de forma fiel el vocabulario definido en el diccionario LSCh del proyecto.
* Respuesta: El equipo validará el diccionario y las animaciones mediante consulta con usuarios sordos y comparación con material oficial disponible en asociaciones lingüísticas.
* Medida de Respuesta: En pruebas piloto, al menos el 85% de las señas deberá ser reconocida correctamente por participantes sordos.

**RNF.14 Consistencia de animaciones**

* Descripción: Las animaciones de señas deberán mantener uniformidad en estilo, velocidad y proporciones para evitar confusiones.
* Respuesta: Todas las animaciones se crearán con las mismas herramientas (FreeMoCap y Blender), asegurando un estándar visual consistente.
* Medida de Respuesta: Revisión de 100% de las animaciones por parte del equipo de QA antes de ser incorporadas al sistema.

**RNF.15 Manual técnico y de integración**

* Descripción: El sistema deberá incluir un manual técnico dirigido a desarrolladores y un manual de integración para usuarios institucionales (ej. canales de TV).
* Respuesta: Se elaborarán documentos en formato PDF y Markdown que incluyan: explicación del pipeline, endpoints de API, ejemplos de uso y pasos de instalación.
* Medida de Respuesta: La documentación deberá ser evaluada por un usuario externo, quien deberá poder instalar y usar el sistema en ≤2 días siguiendo las guías entregadas.

**RNF.16 Registro de pruebas y resultados**

* Descripción: Se deberá mantener un repositorio local con resultados de pruebas de desempeño, validaciones y auditorías internas.
* Respuesta: CatchAI almacenará reportes en carpetas estructuradas con evidencias (logs, capturas de pantalla, métricas de rendimiento).
* Medida de Respuesta: En revisiones académicas o de control de calidad, deberá estar disponible al menos un informe completo de cada iteración de pruebas.

**RNF.17 Precisión mínima de transcripción**

* Descripción: El motor de transcripción (Whisper u otro) deberá alcanzar un nivel de confianza promedio ≥0.85 en las pruebas de vocabulario definido.
* Respuesta: CatchAI configurará el motor de STT con modelos optimizados para español y reducirá interferencias mediante normalización de audio.
* Medida de Respuesta: En las pruebas de validación, el 85% de las frases básicas deberán transcribirse sin errores graves.

**RNF.18** **Calidad visual del avatar**

* Descripción: El video generado deberá mantener una resolución mínima de 720p y una tasa de cuadros estable de al menos 24 fps.
* Respuesta: El motor gráfico Godot se configurará para renderizar animaciones en ese estándar mínimo.
* Medida de Respuesta: En la salida final, cada archivo de video será evaluado automáticamente para verificar resolución y fps.

**RNF.19 Identificación única de cada solicitud**

* Descripción: Cada procesamiento deberá generar un identificador único que permita rastrear los archivos de entrada, salida y los logs asociados.
* Respuesta: CatchAI asignará un UUID a cada solicitud recibida por la API.
* Medida de Respuesta: En auditorías internas, deberá ser posible rastrear un archivo procesado consultando su identificador único en los logs.