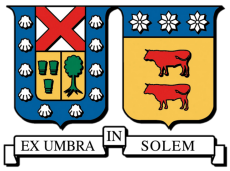
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE ELECTRÓNICA

Santiago – REGIÓN METROPOLITANA



“Modelo acústico de lenguaje natural (NLP) para conversaciones oncológicas de salud”

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL TELEMÁTICO

HSIEN-I LEE

Profesor Guía: Matías Zañartu
Profesor Correferente : Nicolás Jara

Santiago, Chile
04 de diciembre de 2024

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título; ☐ Tesis de Postgrado;

Título del trabajo: Modelo acústico de lenguaje natural en el área oncológica

Nombre del candidato(a): Hsien i lee

Carrera / Grado: Ingeniería Civil Telemática

Campus: Santiago San Joaquin **;** **Departamento**: Electrónica

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Hsien I Lee , en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

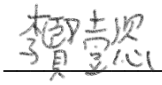
Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 19/08/25 ; **Firma:** 

Estudiante o Candidato(a):

Fecha: 19/08/2025 ; **Firma:** 

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM

Resumen

Este proyecto desarrolla una plataforma web para generar informes médicos personalizados en consultas oncológicas. La plataforma captura y procesa conversaciones oncológicas mediante transcripción y procesamiento de lenguaje natural. Genera dos informes: uno para el paciente, simplificando términos médicos; y otro para el médico, con detalles técnicos y códigos CIE-10. Por ejemplo, en caso de *linfoma no Hodgkin*, se proporciona una definición simplificada para el paciente y detalles técnicos para el médico. Esto busca mejorar la retención de información médica por parte de los pacientes y facilitar el registro clínico para los médicos.

Se utilizan tecnologías avanzadas tanto en el *frontend* como en el *backend*: *React* para la interfaz de usuario, *Node.js* para la lógica del sistema, *AssemblyAI* para la transcripción de audio a texto, y *OpenAI GPT-4* para el procesamiento de lenguaje natural. Estas herramientas trabajan juntas para asegurar una solución eficiente y precisa en la creación de informes médicos.

Como resultado, la plataforma ayuda a los pacientes a comprender mejor su información médica y proporciona a los médicos informes detallados.

Palabras clave: React, Node.js, frontend, backend, AssemblyAI, OpenAI, modelo GPT-4, procesamiento de lenguaje natural, transcripción de audio, generación de informes médicos.

Abstract

This project develops a web platform to generate personalized medical reports for oncology consultations. Its goal is to record, transcribe, and analyze conversations between patients and doctors. The platform produces two types of reports: one for patients, presented in simple and easy-to-understand language, and another for doctors, using appropriate technical terminology. This aims to improve patients' retention of medical information and facilitate clinical documentation for doctors.

Advanced technologies are used in both the frontend and backend: React for the user interface, Node.js for system logic, AssemblyAI for audio-to-text transcription, and OpenAI GPT-4 for natural language processing. These tools work together to ensure an efficient and accurate solution for creating medical reports.

As a result, the platform helps patients better understand their medical information and provides doctors with detailed reports. This project represents a significant step toward the digitization and personalization of healthcare, enhancing communication and documentation in oncology.

Keywords: React, Node.js, frontend, backend, AssemblyAI, OpenAI, GPT-4 model, natural language processing, audio transcription, medical report generation.

Glosario

- **Frontend:** Parte visual del sistema con la que interactúa el usuario.
- **Backend:** Lógica del sistema y gestión de datos.
- **React:** Biblioteca de JavaScript para crear interfaces de usuario.
- **Node.js:** Entorno para ejecutar JavaScript en el servidor.
- **AssemblyAI:** Servicio para convertir audio en texto.
- **OpenAI GPT-4:** Modelo avanzado de lenguaje de OpenAI.
- **NLP:** Procesamiento automatizado del lenguaje humano.
- **Transcripción de Audio:** Conversión de audio en texto escrito.
- **Informe Médico Personalizado:** Informe adaptado para el paciente o médico.
- **CIE-10:** Sistema de códigos para clasificar enfermedades.
- **API:** Conjunto de reglas para la comunicación entre sistemas.
- **EMR:** Sistema digital para gestionar registros médicos.

Índice general

Capítulo 1

Marco general

1.1. Introducción

La atención oncológica presenta desafíos complejos debido a la necesidad de diagnósticos precisos y tratamientos adaptados a cada caso. Durante las consultas, es esencial que el paciente comprenda información crítica como el tipo de cáncer, las opciones terapéuticas y los efectos secundarios esperados. Sin embargo, estudios como el de [1] indican que una gran proporción de pacientes tienen dificultades para comprender completamente las explicaciones recibidas y participar en la toma de decisiones. Esta plataforma busca resolver esta barrera de comunicación mediante informes personalizados que refuercen lo discutido en consulta.

No se trata solo de proporcionar información. Es fundamental que el paciente entienda claramente lo que le comunica el médico para poder tomar decisiones informadas y seguir el tratamiento de manera adecuada. Una comunicación efectiva no solo mejora la seguridad y confianza del paciente, sino que también puede optimizar su experiencia y contribuir a una mejor recuperación.

1.2. Problema a Resolver

Aunque una buena comunicación en oncología es muy importante, muchos estudios muestran que los pacientes olvidan gran parte de lo que se les dice durante una consulta médica. Se estima que los pacientes recuerdan solo un 10 % de la información dada por el médico. Esto afecta su capacidad para seguir las indicaciones y cumplir con el tratamiento de manera correcta.

Este problema puede tener varias causas. Por un lado, los médicos usan términos muy técnicos que son difíciles de entender para los pacientes. Por otro lado, el estrés y la ansiedad que sienten los pacientes durante la consulta pueden hacer que se distraigan y no retengan bien la información. Además, las consultas suelen durar entre 10 y 30 minutos, lo que no siempre es suficiente para explicar todo de manera clara.

y esto puede afectar directamente a los paciente, cuando los pacientes no entienden o no recuerdan bien lo que se habló en la consulta, pueden cometer errores al seguir su tratamiento. Esto puede afectar su recuperación y la eficacia del tratamiento. Por eso, es necesario encontrar soluciones que ayuden a los pacientes a recordar y entender mejor la información de sus consultas, para que puedan seguir el tratamiento correctamente y mejorar sus resultados médicos.

1.3. Solución Propuesta

La solución propuesta consiste en desarrollar una plataforma web para generar y entregar informes médicos después de cada consulta. Al finalizar la conversación, se crearán dos informes: uno dirigido al paciente y otro al médico.

El informe dirigido al paciente contendrá información relevante sobre su diagnóstico y tratamiento, eliminando términos médicos complejos o explicándolos de manera sencilla. Por ejemplo, si el diagnóstico es carcinoma ductal infiltrante, el informe describirá esto como un tipo de cáncer de mama que se origina en los conductos mamarios y puede invadir otros tejidos. Esta simplificación busca asegurar que el paciente comprenda su condición y pueda tomar decisiones informadas. El objetivo es asegurar que el contenido sea lo más

claro y accesible posible para el paciente.

Por otro lado, el informe destinado al médico contendrá terminología técnica y detalles específicos adecuados para describir con precisión la situación médica del paciente. Este informe estará redactado en un lenguaje formal y adaptado a la especialidad de oncología, garantizando así que proporcione la información necesaria para el diagnóstico y tratamiento.

El proyecto utilizó diversas tecnologías para su desarrollo. **React** fue implementado en el *frontend* para ofrecer una interfaz de usuario intuitiva dirigida a los médicos. **AssemblyAI** se encargó de segmentar el audio de cada usuario y realizar la transcripción de audio a texto. **Node.js** funcionó como *backend* para gestionar la lógica del negocio y facilitar la integración de servicios externos, como **OpenAI** y **AssemblyAI**. Además, se implementó un modelo de procesamiento de lenguaje natural (NLP) utilizando los servicios de **OpenAI**, lo que permitió analizar y sintetizar textos médicos para personalizar los informes tanto para los médicos como para los pacientes.

El proyecto se desarrolló en cuatro etapas principales:

1. **Transcripción de Audio a Texto y Segmentación de la Conversación**: Se logró segmentar las conversaciones, eliminando los ruidos externos, y se garantizó que la transcripción a texto fuera precisa. Esto proporcionó una base sólida para el análisis y permitió trabajar eficazmente con el modelo NLP.

2. **Adaptación a un Lenguaje Médico Formal**: Se implementó la clasificación **CIE-10** para generar transcripciones con un lenguaje médico formal tanto para el paciente como para el médico.

3. **Generación de un Informe Resumido para el Paciente y uno con Lenguaje Técnico para el Médico**: Se capturó solo la información relevante sobre la enfermedad discutida, omitiendo diálogos no relacionados con el diagnóstico. Esto aseguró que el paciente recibiera un resumen claro y útil de su consulta. Por otro lado, el informe para el médico incluyó un lenguaje técnico formal, con el respaldo de la clasificación CIE-10.

4. **Inyección de Análisis en el Registro Clínico Electrónico**: Se almacenó en el **EMR** toda la información generada durante las consultas, beneficiando a los pacientes con un registro estructurado y detallado de su atención médica.

1.4. Estado del Arte y de la Técnica

El artículo Advancing Medical Imaging with Language Models: A Journey from N-grams to ChatGPT analiza cómo los modelos de lenguaje, como ChatGPT, pueden mejorar los informes médicos. Estos modelos tienen la capacidad de analizar imágenes médicas y generar informes precisos, aumentando la eficiencia y reduciendo los errores de diagnóstico. Por ejemplo, ChatGPT puede incrementar la precisión del diagnóstico entre un 10 % y un 15 %, y mejorar la precisión de los informes de radiología hasta en un 20 %. En histopatología, la precisión también mejora en un 20 %. Sin embargo, aproximadamente entre el 5 % y el 10 % de los informes contienen frases duplicadas que deben ser eliminadas [2].

Aunque ChatGPT aún no logra generar informes con una precisión del 100 %, sigue siendo una opción destacada. OpenAI ofrece una plataforma robusta y versátil que incluye todas las herramientas necesarias para desarrollar proyectos de generación de informes médicos personalizados.

Por otro lado, el artículo Measuring the Accuracy of Automatic Speech Recognition Solutionsevalúa la precisión de varias soluciones de ASR, incluyendo AssemblyAI. El estudio destaca la importancia de las transcripciones precisas para la accesibilidad de personas sordas y con problemas de audición, y mide el rendimiento de 11 servicios ASR con grabaciones de conferencias de educación superior. Los resultados muestran una variabilidad significativa en la precisión entre los proveedores y señalan que los servicios de transmisión en vivo tienden a tener una calidad menor en comparación con las transcripciones por lotes. A pesar de los avances, los servicios comunes de ASR aún carecen de confiabilidad en términos de precisión [3].

AssemblyAI es uno de los servicios evaluados y ha demostrado ser una opción viable para la transcripción automática de audio, lo cual es crucial para la plataforma de generación de informes médicos oncológicos. La integración de AssemblyAI puede mejorar la precisión

y eficiencia de la transcripción de consultas médicas, asegurando que los pacientes reciban información clara y relevante, y que los médicos obtengan los detalles técnicos precisos necesarios para el tratamiento.

Bibliografía

[1] A. M. García del Río and J. A. González. La información al paciente y su participación en la toma de decisiones. *Atención Primaria*, 35(3):142–146, 2005.

[2] Mingzhe Hu, Shaoyan Pan, and Yuheng Li y Xiaofeng Yang. Advancing medical imaging with language models: A journey from n-grams to chatgpt. 2023.

[3] Korbinian Kuhn, Verena Kersken, Benedikt Reuter, Niklas Egger, and Gottfried Zimmermann. Measuring the accuracy of automatic speech recognition solutions. *ACM Transactions on Accessible Computing*, 16(4):25, 2023.

Capítulo 2

Desarrollo del tema

En el ámbito de la oncología, la comunicación efectiva entre médico y paciente es esencial para garantizar el entendimiento del diagnóstico y tratamiento. Sin embargo, las barreras en la retención de información médica, la complejidad del lenguaje técnico y las limitaciones de tiempo en las consultas plantean un desafío significativo. A continuacion se aborda los conceptos, tecnologías y normativas fundamentales que sustentan el desarrollo de una plataforma web orientada a la generación de informes médicos personalizados.

2.1. Antecedentes

En el contexto de la atención médica, especialmente en la rama de la oncología, la interacción entre médico y paciente es crucial para el entendimiento de diagnósticos y tratamientos. Sin embargo, diversos estudios han demostrado que los pacientes olvidan hasta un 80 % de la información discutida durante una consulta médica, lo que afecta directamente para seguir las indicaciones y los tratamientos prescritos. Este problema se grave en la consultas oncológicas, donde cada detalle puede ser crítico para la evolución de la enfermedad.

2.2. Bases Teóricas

Para el desarrollo de este proyecto, se llevaron a cabo dos pasos fundamentales. El primero fue la **transcripción de audio a texto**, que permitió convertir las conversaciones médico-paciente en texto claro y estructurado. El segundo fue el **procesamiento de lenguaje natural (NLP)**, el cual analizó y sintetizó este texto para crear informes personalizados. Estos procesos facilitaron una comunicación efectiva, asegurando que los pacientes comprendieran la información y que los médicos recibieran detalles técnicos precisos para el registro clínico.

2.2.1. Transcripción de Audio a Texto

AssemblyAI es una API de reconocimiento automático de voz (ASR) que permite transcribir grabaciones de consultas médicas con alta precisión. Durante el desarrollo de esta plataforma, se realizaron pruebas con grabaciones de consultas simuladas, donde AssemblyAI logró identificar a distintos hablantes (médico, paciente y acompañante) y filtrar el ruido ambiental.

AssemblyAI fue seleccionado para este proyecto debido a las siguientes:

- **Segmentación por hablantes:** Es capaz de distinguir entre diferentes voces en una conversación. Esto resulta esencial en un contexto médico, donde cada rol tiene un peso importante en el diálogo.
- **Precisión en términos difíciles:** Aunque no está diseñado específicamente para el área médica, su transcripción se puede mejorar con herramientas adicionales para ajustar el texto a un contexto más especializado.
- **Facilidad de integración:** AssemblyAI se ofrece como un servicio API REST, lo que simplifica su integración en aplicaciones web como la desarrollada en este proyecto.

La implementación de AssemblyAI comienza con la captura del audio de la consulta médica, que luego se envía al servicio exterior AssemblyAI para su procesamiento. Este servicio analiza el audio, genera una transcripción precisa y segmenta los diálogos según

los diferentes hablantes. El resultado es un texto transcrito que sirve como base para las siguientes etapas del proyecto, como el procesamiento de lenguaje natural (NLP) y la generación de informes personalizados.

2.2.2. Procesamiento de Lenguaje Natural

Actualmente, **OpenAI** es reconocido como uno de los líderes en el desarrollo de tecnologías de procesamiento de lenguaje natural. Entre sus herramientas, el modelo **GPT-4** destaca por ser una de las tecnologías más avanzadas en este campo, con una gran capacidad para comprender y generar texto de manera precisa y coherente. Esta tecnología facilita tareas complejas, como la generación de informes médicos personalizados tanto para el paciente como para el médico.

OpenAI y su modelo *GPT-4* fue seleccionado para este proyecto debido a las siguientes:

La elección de *OpenAI* y su modelo *GPT-4* para este proyecto responde a las siguientes características únicas:

- **Avanzada capacidad de comprensión:** GPT-4 puede interpretar con precisión conversaciones médicas, incluso en escenarios donde se utiliza lenguaje técnico. Esto permite extraer información clave y diferenciar cada hablante, tales como el médico, el paciente y cualquier acompañante (donde puede haber más de un acompañante).

- **Flexibilidad en la generación de informes:** El modelo se adapta a las necesidades específicas del proyecto, generando:

  - **Informe para el paciente:** Un resumen claro y accesible, escrito en un lenguaje sencillo, que facilita la comprensión de la consulta médica.

  - **Informe para el médico:** Un documento técnico y conciso del estado del paciente y su futuros tratamiento incluyendo el diagnostico.

- **Integración:** La API de OpenAI es fácil de integrar con otros servicios, como *AssemblyAI*, permitiendo un flujo de trabajo eficiente desde la transcripción de

audio hasta la generación de informes.

La implementación del modelo **GPT-4** comienza con la **identificación de roles**, que clasifica las intervenciones en las transcripciones para asignar roles específicas a cada hablante (médico, paciente o acompañante). Este paso es esencial, ya que en muchos casos los pacientes o acompañantes suelen realizar **autodiagnósticos** y muchas veces estas informaciones son incorrectas. Incluir estos autodiagnósticos en los informes podría afectar negativamente su precisión y generar errores en los resultados. Por eso, una vez que se identifiquen correctamente los roles, se procede con la **generación de informes**.

A partir de las transcripciones procesadas(con los roles identificados), se generan dos tipos de informes.

Una vez generados los informes, se procede a la **clasificación de los códigos CIE-10**. El modelo identifica los términos médicos presentes en las transcripciones y les asigna códigos de la **Clasificación Internacional de Enfermedades (CIE-10)**. Finalmente, se actualiza el informe médico para el médico con estos códigos.

La clasificación **CIE-10** es fundamental para elaborar informes médicos precisos dirigido a los profesionales de la salud. Su uso permite estandarizar la terminología médica utilizada, lo que facilita una comprensión clara y uniforme del diagnóstico y tratamiento. Gracias a esta codificación, cuando un paciente es atendido posteriormente por otro médico, el profesional puede revisar el informe generado y entender correctamente todo lo discutido en la consulta anterior, asegurando así una continuidad efectiva en la atención médica.

Capítulo 3

Desarrollo de la Solución

3.1. Análisis de Requerimientos

Antes de comenzar con el desarrollo del proyecto, es esencial definir de manera clara todos los requisitos necesarios. Esto incluye identificar los **requisitos funcionales** y los **requisitos no funcionales** que el sistema debe cumplir.

Tener claros estos requisitos permite avanzar de manera más eficiente y evitar problemas durante el desarrollo. De esta forma, se asegura que el proyecto cumpla con sus objetivos y se puedan prevenir posibles dificultades en el proceso.

3.1.1. Requisitos Funcionales

A continuacion se nombrara los requisitos funcionales donde especifican las características y funciones que el sistema debe cumplir para alcanzar los objetivos propuestos:

- **RF1:** El sistema debe iniciar la grabación de audio cuando el médico haga clic en el botón correspondiente en el *frontend*.
- **RF2:** Implementar un sistema de codificación *CIE-10* que asigne códigos a los diagnósticos médicos extraídos de las transcripciones.
- **RF3:** Generar dos informes diferenciados al finalizar la consulta:

  - Un informe técnico para el médico con terminología médica formal y concisa.
  - Un informe simplificado para el paciente, con términos médicos explicados en un lenguaje accesible.

- **RF4:** Permitir la transcripción automática de audio a texto utilizando la API de *AssemblyAI*, segmentando las intervenciones según los roles de los hablantes (médico, paciente, acompañantes).
- **RF5:** Integrar el sistema con el registro médico electrónico (*EMR*), almacenando los informes generados de forma segura para su consulta futura.
- **RF6:** Proporcionar al usuario la capacidad de pausar, detener o cancelar la grabación en cualquier momento durante la consulta.
- **RF7:** Facilitar la interacción con los servicios externos (*OpenAI*, *AssemblyAI*) a través de un *backend* eficiente desarrollado en *Node.js*.

3.1.2. Requisitos No Funcionales

A continuacion se nombraran algunos de los requisitos no funcionales donde describen las cualidades y restricciones del sistema que aseguran su desempeño y usabilidad:

- **RNF1:** El sistema debe ser capaz de procesar y generar informes médicos en un tiempo máximo de 5 minutos tras finalizar la consulta.
- **RNF2:** La interfaz de usuario, desarrollada en *React*, debe ser intuitiva, fácil de usar y accesible desde cualquier dispositivo con un navegador web moderno.
- **RNF3:** Los datos sensibles, como las grabaciones de audio y los informes médicos, deben almacenarse y transmitirse utilizando protocolos de seguridad robustos para garantizar la confidencialidad y la integridad de la información.
- **RNF4:** El sistema debe ser escalable, capaz de manejar múltiples consultas y grabaciones simultáneamente sin afectar su rendimiento.

- **RNF5:** Los informes generados deben estar libres de errores ortográficos y gramaticales, presentando la información de manera clara y profesional.
- **RNF6:** El sistema debe integrarse de manera eficiente con servicios externos (*AssemblyAI* y *OpenAI*), garantizando una comunicación fluida y rápida.
- **RNF7:** Las grabaciones de audio deben almacenarse temporalmente en el sistema, eliminándose automáticamente una vez que se haya completado la generación de los informes.

3.1.3. Requisitos de Interfaces

A continuación, se describen los eventos externos necesarios para la interacción del sistema:

- **Evento:** Acceso del médico al EMR.
- **Descripción:** El médico necesita acceder al sistema de registros médicos electrónicos (EMR) para revisar los datos clínicos del paciente.
- **Iniciador:** Inicio de sesión del médico en el sistema.
- **Parámetros:** Credenciales de inicio de sesión (usuario y contraseña).
- **Respuesta:** Acceso otorgado al EMR con los registros clínicos del paciente disponibles para su revisión y actualización.

Para la respuesta del sistema, se consideran las siguientes especificaciones:

- **Respuesta:** Acceso a los registros clínicos del paciente.
- **Descripción:** El sistema proporciona acceso seguro a los registros clínicos del paciente, permitiendo al médico revisar y actualizar la información necesaria.
- **Parámetros:** Credenciales válidas proporcionadas durante el inicio de sesión en el EMR.

3.1.4. Requisitos de Ambiente

Hardware de Desarrollo

Se requerirá como hardware de desarrollo un computador capaz de soportar las distintas actividades que se irán realizando a medida que avanza en el modelo

Software de Desarrollo

El desarrollo del sistema se basa en las siguientes herramientas de software:

- **Entorno de Desarrollo Integrado (IDE):** Visual Studio Code, que permite un desarrollo eficiente en *JavaScript* y *Node.js*.
- **Frameworks y Bibliotecas:**
  - *React:* Para el desarrollo del *frontend*.
  - *Node.js:* Para la implementación del *backend*.
- **Servicios externos:**
  - *AssemblyAI:* Para la transcripción de audio a texto.
  - *OpenAI:* Para el procesamiento de lenguaje natural y generación de informes médicos personalizados.
- **Sistema de Gestión de Versiones:** Git, con almacenamiento en GitHub para el control de versiones y colaboración.

3.2. Elección de Tecnologías

3.2.1. React

La elección de *React* para el desarrollo del *frontend* de la plataforma se debe a varias razones:

React permite dividir la interfaz en componentes reutilizables, lo que simplifica el mantenimiento y facilita la escalabilidad del proyecto. Cada componente puede manejar su propio estado y lógica, permitiendo desarrollos más organizados y eficientes.

Además, su amplia comunidad proporcionan bibliotecas y herramientas adicionales, como *React Router* para el manejo de rutas, lo que facilita el desarrollo de aplicaciones complejas.

Otra ventaja de *React* es su compatibilidad con *JavaScript*, uno de los lenguajes de programación más utilizados y conocidos.

En este proyecto, el *frontend* desarrollado con *React* incluyó:

- Componentes interactivos para la captura de audio, ingreso de datos del paciente y gestión de informes.
- Diseño adaptativo que garantiza una experiencia de usuario óptima en distintos dispositivos.
- una comunicación fluida con el *backend* implementado en *Node.js*.

3.2.2. Node.js

Para la implementación del *backend* de este proyecto, se seleccionó *Node.js* como la tecnología principal. *Node.js* es un entorno de ejecución para *JavaScript*.

La elección de *Node.js* es debido a distintos factores. En primer lugar, Node.js utiliza la arquitectura «Single Threaded Event Loop» para manejar múltiples clientes al mismo tiempo. Para entender en qué se diferencia de otros tiempos de ejecución y orientada a eventos permite manejar múltiples solicitudes concurrentes con un alto nivel de eficiencia, lo que es esencial para una aplicación que interactúa con servicios externos como *AssemblyAI* y *OpenAI*. Además, *Node.js* ofrece una integración nativa con *JavaScript*, lo que facilita la comunicación fluida entre el *frontend* desarrollado en *React* la cual tambien utiliza el lenguaje javascrip y el *backend*.

En el contexto de este proyecto, *Node.js* fue utilizado para:

- **Gestión de la lógica del negocio:** Manejar la lógica necesaria para procesar las transcripciones, identificar roles, generar informes y clasificar diagnósticos con CIE-10.
- **Integración con servicios externos:** Establecer comunicación con las APIs de *AssemblyAI* y *OpenAI* para la transcripción de audio y el procesamiento de lenguaje natural.
- **Manejo de datos:** Asegurar la recepción, procesamiento y envío de datos entre el *frontend* y el *backend*, garantizando una experiencia de usuario fluida.

3.2.3. AssemblyAI

En este proyecto, se utilizó *AssemblyAI* como la tecnología principal para la transcripción de audio a texto.

En este proyecto, *AssemblyAI* desempeñó un papel fundamental al:

- Recibir grabaciones de audio generadas durante las consultas médicas.
- Transcribir el contenido del audio a texto de manera precisa.
- Identificar los diferentes roles de los hablantes, clasificando sus intervenciones como médico, paciente o acompañante.
- Proveer datos textuales procesables para su posterior análisis mediante el modelo *GPT-4* de *OpenAI*.

El flujo de trabajo inicia con el envío de los audios grabados al servicio de *AssemblyAI*, que procesa y devuelve una transcripción detallada en formato JSON. Esta transcripción incluye información segmentada, como los intervalos de tiempo de cada intervención, y está estructurada para facilitar su uso en las siguientes etapas del proyecto.

3.2.4. OpenAI

En este proyecto, *OpenAI* y su modelo avanzado *GPT-4* desempeñaron un papel clave en el análisis y procesamiento del lenguaje natural. *OpenAI* es una de las plataformas líderes en inteligencia artificial, conocida por su capacidad de comprender, procesar y generar texto con una alta precisión y adaptabilidad, lo que la convierte en una herramienta esencial para aplicaciones complejas como la generación de informes médicos personalizados.

El modelo *GPT-4* fue utilizado en diferentes etapas del proyecto, incluyendo:

- **Identificación de roles:** Clasificó las intervenciones en las transcripciones, etiquetando a cada hablante como médico, paciente o acompañante.
- **Generación de informes:** Procesó las transcripciones segmentadas para crear informes adaptados a las necesidades de los usuarios finales (pacientes y médicos).
- **Clasificación de diagnósticos:** Identificó términos médicos y los vinculó con códigos CIE-10, estandarizando la información para su integración en sistemas de registro clínico.

3.3. Arquitectura de la plataforma web

3.3.1. Diagrama de Contexto

El diagrama muestra la interacción entre los usuarios y el sistema en la generación de informes médicos:

- **Médico:** Es el usuario que interactúa directamente con el sistema. Inicia la grabación y recibe el informe técnico.
- **Paciente:** No interactúa directamente con el sistema, pero su diálogo es grabado para generar un informe adaptado a su comprensión.
- **Sistema:** Graba la conversación, procesa los datos y entrega dos informes: uno técnico para el médico y uno simplificado para el paciente.

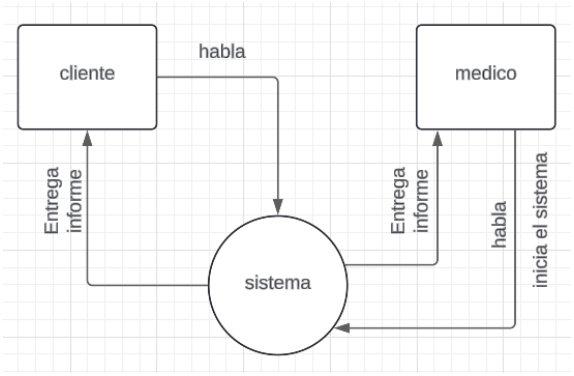


Figura 3.1: diagrama de contexto

3.3.2. Diagrama de Arquitectura

La Figura 3.3 muestra el diagrama de arquitectura del sistema, destacando las interacciones entre sus componentes principales. Se observa que el **Backend (Node.js)** es el núcleo que conecta todos los elementos del sistema. A continuación, se describe el flujo de información y la interacción entre los componentes:

- **UI (React):** Es la interfaz de usuario donde el médico interactúa con el sistema. Envía los datos capturados al Backend (en este caso sería audios de las consultas medicas).

- **Backend (Node.js):** Recibe la información desde la UI, la procesa y realiza solicitudes a diferentes servicios externos. Es responsable de coordinar el flujo de datos en el sistema.

- **AssemblyAI:** El Backend envía el audio grabado a AssemblyAI, que se encarga de procesarlo, segmentarlo y convertirlo de audio a texto una ves hecho eso esta devuelve de vuelta el dialogo segmentado.

- **OpenAI (GPT-4):** Una vez obtenida la transcripción de AssemblyAI, el Backend envía una solicitud a OpenAI (GPT-4). Este servicio se encarga de identificar los roles de los hablantes y generar los informes personalizados, lo devuelve nuevamente al backend

- **Base de Datos (BD):** El Backend está conectado a la base de datos, donde se almacenan los datos necesarios para el procesamiento y gestión de los informes.
- **EMR (Electronic Medical Record):** Una vez generados los informes, los datos son enviados al *EMR* para ser almacenados y accesibles para futuras consultas. En este caso, no se tiene un acceso directo al *EMR* del Instituto *FALP*, pero se proporcionó un *endpoint* que contenía toda la información necesaria del instituto en la parte de la inyección de datos.
- **OncoData y FALPComm:** Estos módulos también interactúan con el Backend mediante solicitudes RESTful para facilitar la comunicación y el intercambio de información específica de oncología.

Este diagrama de arquitectura refleja cómo cada componente interactúa para garantizar el flujo eficiente de información, desde la grabación inicial hasta la generación y almacenamiento de los informes médicos.

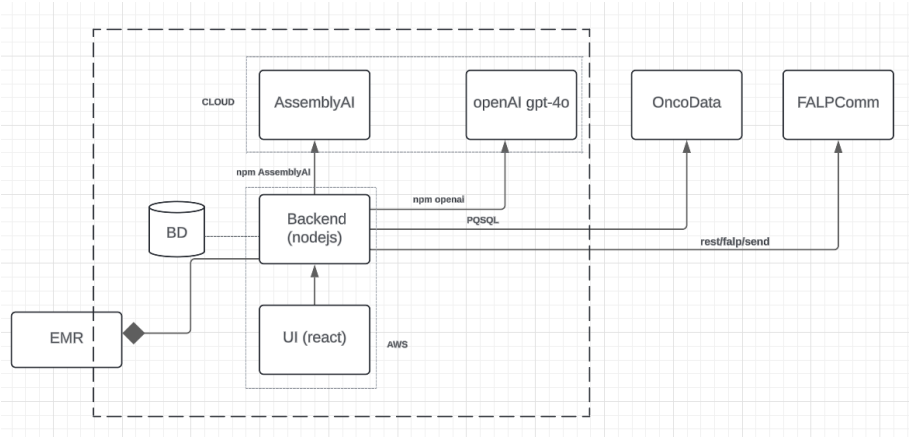


Figura 3.2: diagrama de arquitectura

3.4. Módulos del Sistema

A continuación, se presenta una descripción de los módulos que conforman el sistema de generación de informes médicos personalizados. Cada módulo tiene una función específica que facilita el proceso de transcripción de audio, procesamiento de lenguaje natural y entrega de informes. Estos módulos trabajan de manera coordinada para asegurar que el

sistema funcione correctamente y cumpla con su objetivo.

En la siguiente tabla se detallan los módulos del sistema, su propósito.

Módulo	**Propósito**	**Sección**
Módulo de Gestión de Lógica de Negocios (Backend)	Gestiona la lógica del negocio y la comunicación entre los módulos del sistema	5.1
Módulo de Interfaz de Usuario (Frontend)	Proporciona una interfaz interactiva para médicos, comunicándose con el backend para mostrar datos, visualización de informes y entrada de datos.	5.2
Módulo de Almacenamiento de Datos (ERM)	Almacena todos los datos utilizados y generado de los informes personalizados.	5.3
Módulo de Transcripción de Audio (AssemblyAI)	Transcribe el audio de las consultas médicas en texto utilizando la API de AssemblyAI, proporcionándonos texto transcrito de audio a texto desde múltiples speakers .	5.4
Módulo de Procesamiento de Lenguaje Natural (Servicio OpenAI gpt-4o)	Proporciona capacidades avanzadas de NLP, como resumen de información y formalización del lenguaje, utilizando la API de OpenAI.	5.5

Cuadro 3.1: Módulos de la arquitectura del sistema

3.4.1. Módulo de Gestión de Lógica de Negocios

- **Propósito:**
    - Centralizar y manejar la lógica de negocio dentro del sistema, asegurando que todas las operaciones de negocio se realicen de manera eficiente. Este módulo se comunica con distintas API, incluyendo AssemblyAI, OpenAI, la base de datos, EMR y otros servicios, garantizando coherencia y seguridad.

Es responsable de procesar las solicitudes de los usuarios, aplicar las reglas del negocio y coordinar la comunicación entre los diferentes componentes del sistema.

- **Alcance:**
  - Recibir y procesar las solicitudes desde el frontend.
  - Interactuar con el EMR para almacenar y recuperar datos clínicos.
  - Coordinar la comunicación con servicios externos, como la transcripción de audio a texto (AssemblyAI) y el procesamiento de lenguaje natural (OpenAI).
- **Dependencias:**
  - Este módulo depende del frontend para recibir solicitudes y enviar respuestas. Además, depende de servicios externos para la conexión con APIs, permitiendo la generación adecuada de los informes médicos.
- **Supuestos:**
  - Se espera que los servicios externos (AssemblyAI, OpenAI) y el EMR respondan en tiempos adecuados para mantener la eficiencia del sistema.
  - Se asume que el servidor que ejecuta Node.js cuenta con suficiente capacidad de procesamiento y memoria para manejar múltiples solicitudes concurrentes.
  - Se requiere una conexión de red estable y rápida para la comunicación entre los módulos y los servicios externos.
- **Restricciones:**
  - Debe mantener tiempos de respuesta bajos para garantizar una experiencia de usuario fluida.
  - Debe implementar medidas de seguridad para proteger los datos sensibles y evitar accesos no autorizados.

- **Estructura General:**
  - **Entradas:**
    - Solicitudes del frontend para operaciones CRUD en el sistema de gestión de datos.
    - Datos de audio de consultas médicas para transcripción.
    - Solicitudes para generar informes personalizados.
  - **Ejecución:**
    - Recibir solicitud del frontend.
    - Enviar datos al servicio de transcripción de voz a texto (AssemblyAI).
    - Enviar el texto transcrito al servicio de procesamiento de lenguaje natural (OpenAI).
    - Interactuar con el EMR para realizar operaciones CRUD.
  - **Salidas:**
    - Generación y entrega de dos informes personalizados, tanto para el médico como para el paciente, en formato JSON.

3.4.2. Módulo de Interfaz de Usuario (react)

- **Propósito:**
  - Proveer una interfaz interactiva para los médicos, permitiéndoles iniciar la grabación de consultas y generar informes personalizados.
- **Alcance:**
  - El módulo de UI se encarga de la interacción directa con los usuarios. Aquí es donde el médico inicia la grabación y, una vez finalizada, se generan los

informes personalizados.

- **Dependencias:**
  - Depende del Módulo de Gestión de Lógica de Negocios para obtener y enviar datos.
  - Depende del Módulo de Transcripción de Audio (Servicio AssemblyAI) para transcribir el audio a texto y del Módulo de Procesamiento de Lenguaje Natural (Servicio OpenAI gpt-4o) para generar informes personalizados tanto para el médico como para el paciente.
- **Supuestos:**
  - Se asume que los usuarios tendrán acceso a dispositivos con navegadores web modernos.
  - Se asume que el backend y otros servicios estarán disponibles y funcionando correctamente.
- **Restricciones:**
  - Limitaciones en el tiempo de generación de los informes personalizados, debido a la dependencia de varios servicios externos como AssemblyAI y OpenAI.
- **Estructura General:**
  - **Entradas:** El médico hace clic en el botón de grabar, iniciando la grabación. La interfaz incluye cinco botones: grabar, pausar, terminar, anular y generar informe.
  - **Salidas:** Visualización de los informes generados.

3.4.3. Módulo de Almacenamiento de Datos (EMR)

- **Propósito:**

- El propósito de este módulo es gestionar de manera eficiente y segura el almacenamiento, recuperación y manipulación de todos los datos del sistema. Se almacenarán tanto los informes personalizados del paciente sus datos personales , incluyendo : nombre , apellido y rut, garantizando su accesibilidad para futuras consultas y análisis.

- **Alcance:**
  - Se encarga de almacenar los informes personalizada tanto del paciente como el del medico y ademas de las informaciones personales del paciente

- **Dependencias:**
  - Este módulo interactúa con el EMR, encargado de integrar los datos y asegurar que todo quede registrado.

- **Supuestos:**
  - Se espera que las consultas y operaciones en EMR se realicen dentro de tiempos aceptables para no afectar la eficiencia del sistema.
  - Se asume una conexión de red estable y rápida para la comunicación entre el servidor de la EMRs y otros módulos del sistema.

- **Restricciones:**
  - Debe mantener tiempos de respuesta bajos para garantizar una experiencia de usuario fluida.
  - Debe implementar medidas de seguridad para proteger los datos sensibles y evitar accesos no autorizados.

- **Estructura General:**
  - **Entradas:**
    - Solicitudes del backend para actuar con el enpind del EMR.

- **Ejecución:**
  - El backend recibe las solicitudes del frontend y, una vez procesadas, los informes generados se almacenan en el EMR.
- **Salidas:**
  - Entrega de datos para futuros tratamientos o análisis.

3.4.4. Módulo de Transcripción de Audio (Servicio AssemblyAI)

- **Propósito:**
  - Este módulo se encarga de convertir el audio de las consultas médicas en texto utilizando el servicio de AssemblyAI.
- **Alcance:**
  - El módulo de transcripción de audio toma las grabaciones de audio generadas durante las consultas médicas y las transcribe a texto. Este texto se utiliza posteriormente para generar informes médicos personalizados y mejorar la retención de información por parte del paciente.
- **Dependencias:**
  - Este módulo depende principalmente del backend, ya que es el backend el encargado de conectarse a los servicios externos. En este caso, utilizará la API de AssemblyAI para realizar la transcripción de audio.
- **Supuestos:**
  - Se asume que los archivos de audio proporcionados están en un formato compatible con AssemblyAI y que la conexión a internet es estable.
- **Restricciones:**
  - Las transcripciones pueden estar limitadas por la calidad del audio grabado,

el ruido de fondo y la claridad del discurso. También existen limitaciones en términos de la capacidad de procesamiento del servicio AssemblyAI y el tiempo de respuesta.

- **Estructura General:**
  - **Entrada:** Archivo de audio de la consulta médica.
  - **Ejecución:** Enviar el archivo de audio a AssemblyAI. Recibir la transcripción del audio en texto. Procesar y almacenar la transcripción.
  - **Salida:** Texto transcrito y segmentado del audio de la consulta médica.

3.4.5. Módulo de Procesamiento de Lenguaje Natural (Servicio OpenAI GPT-4)

- **Propósito:**
  - Este módulo se encarga de procesar las transcripciones de audio para generar resúmenes utilizando el servicio de procesamiento de lenguaje natural GPT-4 de OpenAI.
- **Alcance:**
  - El módulo de procesamiento de lenguaje natural toma las transcripciones de audio y las analiza para generar informes médicos personalizados, resúmenes de consultas y respuestas a preguntas específicas basadas en el contenido de las consultas.
- **Dependencias:**
  - Este módulo depende del modulo del backend, ya que es el encargado de incluir la API de OpenAI GPT-4 para el procesamiento de lenguaje natural. También requiere el módulo de transcripción de audio, que proporciona el texto transcrito a procesar e incluir en la plataforma.

- **Supuestos:**
  - Se asume que las transcripciones de audio están en un formato adecuado para ser procesadas por GPT-4 y que la conexión a internet es estable.
- **Restricciones:**
  - Las capacidades de procesamiento están limitadas por la complejidad de las consultas médicas y la precisión de las transcripciones. También existen limitaciones en términos de la capacidad de procesamiento del servicio GPT-4 y el tiempo de respuesta.
- **Estructura General:**
  - **Entrada:** Texto transcrito de la consulta médica.
  - **Ejecución:** Enviar el texto transcrito a la API de GPT-4. Recibir el análisis y las respuestas generadas por GPT-4.
  - **Salida:** Informes médicos personalizados.

3.5. Diseño del Prototipo

3.5.1. Contexto General del Prototipo

El diseño del prototipo de la plataforma se enfoca en ofrecer una interfaz intuitiva y fácil de usar para los médicos. Se incorporaron los colores institucionales del Instituto FALP para reforzar la identidad visual y crear un entorno familiar.

La interfaz cuenta con botones para *iniciar*, *pausar*, *detener* y *cancelar* la grabación de las consultas. Además, el flujo de trabajo es simple y estructurado, guiando al médico desde el ingreso de datos del paciente hasta la generación de informes personalizados.

3.5.2. Descripción de Componentes

En la plataforma encontramos seis componentes, que son los siguientes:

1. **Iniciar la grabación**: Este botón redirige al usuario a la sección donde debe ingresar las informaciones del paciente donde sería el nombre, apellido y rut.

2. **Información del paciente**: En esta sección, el usuario debe ingresar los datos del paciente para su identificación y posterior búsqueda en el sistema de registros médicos electrónicos (EMR). Los campos presentes en este componente incluyen:

    - **Nombre**: Campo de texto donde se ingresa el nombre del paciente.
    - **Apellido**: Campo de texto donde se ingresa el apellido del paciente.
    - **RUT**: Campo de texto para el número de identificación del paciente (RUT).
    - **Botón Confirmar**: Al presionar este botón, se validan los datos ingresados y se permite continuar a la siguiente etapa de grabación.

3. **Confirmar**:Una vez que se presiona el botón, el sistema determinará si se navega a la siguiente sección o si se deben completar los datos correctamente. En caso de que falten datos o el RUT no tenga todos los números requeridos, no se podrá procesar para avanzar a la siguiente sección.

4. **Grabar**: Este componente se encarga de iniciar la grabación del audio de la consulta. Al presionar el botón “Grabar”, el sistema comienza a captar y almacenar la conversación entre el médico y el paciente.

5. **Pausar**: Este botón permite pausar la grabación en cualquier momento. Es útil en caso de que se necesite un descanso en la conversación o una interrupción temporal.

6. **Terminar**: Este botón finaliza la grabación y empieza el proceso de generación de informes. Una vez que el audio ha sido capturado, se envía a AssemblyAI, que se encarga de segmentarlo y transcribirlo a texto. A continuación, el texto transcrito es procesado por OpenAI, que identifica los roles de los participantes y genera dos informes personalizados: uno para el médico y otro para el paciente.

7. **Anular**: Este botón cancela la grabación actual y permite reiniciar el proceso desde el principio. Al utilizarlo, cualquier audio capturado en la sesión actual es descartado.

En la tabla que se muestra a continuación, se detallan las diferentes rutas API utilizadas en el proyecto. Cada una de estas rutas facilita una funcionalidad específica para la generación y entrega de informes médicos. Para estas interacciones se han utilizado solicitudes de tipo **POST**, debido a que estas permiten enviar datos al servidor y recibir una respuesta con la información procesada.

Las rutas definidas permiten realizar tareas como la identificación de roles de hablantes, la generación de códigos **CIE-10**, y la creación de informes personalizados tanto para el médico como para el paciente. A continuación se describen las rutas y sus respectivas funciones:

Método HTTP	**Ruta**	**Recibe**	**Entrega**	**Función**
POST	/api/openai/roles	Texto generado por AssemblyAI	JSON con roles de hablantes	Identificar roles (Doctor, Paciente, Acompañante)
POST	/api/cie/generar-cie10	Diálogo con los roles identificados	JSON con códigos CIE-10	Generar códigos CIE-10
POST	/api/paciente/generarinformepaciente	Diálogo con los roles identificados	Informe para el paciente	Generar informe personalizado para el paciente
POST	/api/doctor/informedoctor	Diálogo con los roles identificados	Informe para el doctor	Generar informe técnico para el doctor
POST	/api/transcripcion	Archivo de audio e información del paciente	JSON con los datos del paciente, informe del paciente, informe del médico y códigos CIE-10	Transcribir y etiquetar diálogos

Cuadro 3.2: Rutas API utilizadas en el proyecto

Estas rutas aseguran el correcto flujo de información entre los diferentes componentes del sistema, garantizando una comunicación eficiente entre el *frontend*, el *backend* y los servicios externos utilizados en el proyecto.

3.5.3. Esquema general de los componente del prototipo

El siguiente esquema se observa la interacción entre los principales componentes del prototipo desarrollado para la grabación y generación de informes médicos personalizados.

En este flujo, se destacan tres áreas principales: el frontend, el backend, y los servicios externos (AssemblyAI y OpenAI) que facilitan la transcripción y análisis de los datos.

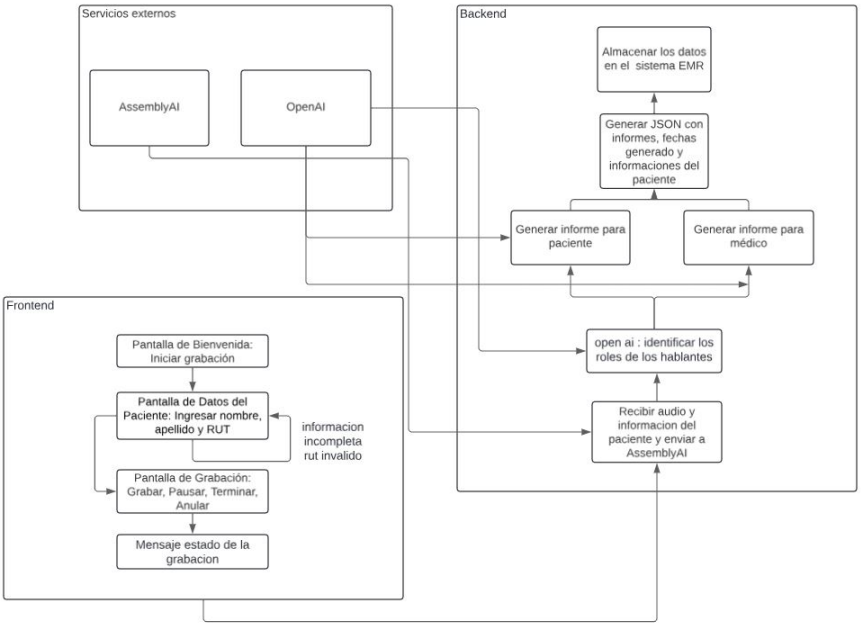


Figura 3.3: Esquema general de los componente del prototipo

3.6. Módulo de Gestión de Lógica de Negocios (Backend)

El **Backend** de la plataforma es el núcleo central encargado de gestionar la lógica del sistema y coordinar el funcionamiento entre los distintos módulos y servicios externos. Se desarrolló utilizando *Node.js* debido a su eficiencia en el manejo de múltiples solicitudes simultáneas y su amplia compatibilidad con APIs y bibliotecas modernas.

3.6.1. Responsabilidades del Backend

El backend se encarga de:

- **Recepción y procesamiento de solicitudes:** Interpreta las solicitudes provenientes del frontend y coordina las operaciones necesarias para satisfacer dichas solicitudes.
- **Comunicación con APIs externas:**

- **AssemblyAI:** Se utiliza para la transcripción de audio a texto. El backend envía los archivos de audio grabados durante las consultas y recibe las transcripciones generadas por AssemblyAI.
- **OpenAI GPT-4:** Se utiliza para el procesamiento de lenguaje natural (NLP). El backend envía las transcripciones procesadas para que el modelo GPT-4 genere resúmenes personalizados tanto para el paciente como para el médico.
- **EMR:** El backend interactúa con el sistema de registros médicos electrónicos para almacenar y recuperar información clínica del paciente.

- **Procesamiento de datos y generación de informes:** Una vez recibidas las transcripciones y los análisis de NLP, el backend organiza y formatea esta información para generar dos tipos de informes:
  - Un informe simplificado para el paciente.
  - Un informe técnico para el médico, con terminología especializada y códigos CIE-10.

3.6.2. Flujo de Trabajo del Backend

1. **Inicio de solicitud:** El médico utiliza la interfaz de usuario para iniciar la grabación de una consulta.
2. **Envío a AssemblyAI:** El audio se envía al servicio de AssemblyAI para su transcripción.
3. **Procesamiento con OpenAI:** La transcripción se envía a OpenAI GPT-4 para analizar el texto y generar resúmenes personalizados.
4. **Generación de informes:** El backend procesa los resultados de NLP y genera los dos informes.
5. **Almacenamiento en EMR:** Los informes se almacenan en el sistema de registros médicos electrónicos para su consulta futura.

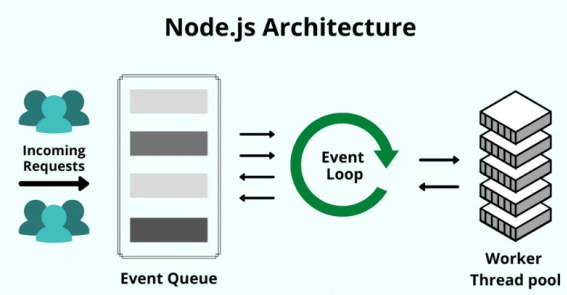


Figura 3.4: Diagrama de Contexto

Capítulo 4

Resultado

4.0.1. Interfaces Implementadas en el Prototipo

Se ha implementado una interfaz de usuario intuitiva y diseñada para ser lo más accesible posible. Al ingresar a la plataforma, el usuario es recibido con un mensaje de bienvenida y un botón para iniciar la grabación. Al presionar este botón, el sistema redirige al usuario a una página donde se le solicita ingresar la información del paciente, incluyendo nombre, apellido y RUT. Estos datos son fundamentales para identificar al paciente y permitir su búsqueda en el sistema de registros médicos electrónicos (EMR) en el futuro.

Una vez que la información del paciente ha sido ingresada y verificada, el sistema permite avanzar a la sección de grabación y proceder con la generación del informe.

A continuación, se describen las interfaces de usuario implementadas:

Pantalla de Bienvenida: Al iniciar la plataforma, el usuario es recibido con una pantalla de bienvenida que incluye el logotipo de la institución y un botón para comenzar la grabación de la consulta.

Figura 4.1: Pantalla de bienvenida

Pantalla de Información del Paciente: Tras seleccionar el botón "Iniciar grabación" en la pantalla de bienvenida, el usuario es redirigido a una página donde debe ingresar los datos del paciente, como nombre, apellido y RUT. Esta información es esencial para facilitar la identificación del paciente en el sistema de registros médicos electrónicos (EMR) y permitir el acceso a su historial en futuras consultas. Para continuar con el proceso de grabación, el usuario debe ingresar todos los datos requeridos del paciente de manera correcta.

A continuación, se muestran las diferentes variaciones de esta pantalla:

- **Pantalla de Información del Paciente (Normal)**: Esta es la pantalla estándar, donde el usuario introduce los datos del paciente sin errores o mensajes de advertencia.

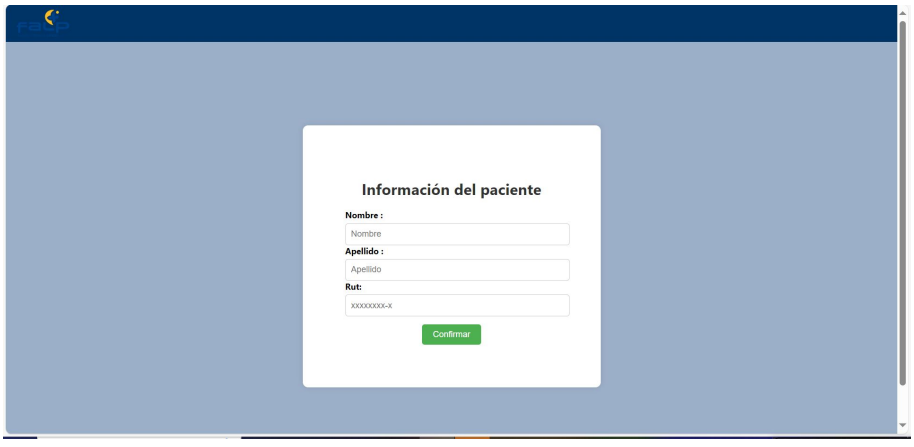


Figura 4.2: Pantalla de información del paciente: vista normal

- **Pantalla de Información del Paciente (Campos incompletos)**: Si el usuario intenta continuar sin completar todos los campos, se muestra un mensaje de advertencia en rojo solicitando que se completen todos los campos obligatorios.

Figura 4.3: Pantalla de información del paciente: campos incompletos

- **Pantalla de Información del Paciente (RUT incorrecto)**: En caso de que el usuario ingrese un RUT incorrecto, el sistema muestra un mensaje de error en rojo indicando que el RUT ingresado es inválido, solicitando al usuario que lo corrija antes de continuar.

Figura 4.4: Pantalla de información del paciente: RUT incorrecto

Pantalla para Iniciar la Grabación: Luego de ingresar la información del paciente y confirmar los datos, el usuario es llevado a la pantalla de grabación, en la cual se presentan cuatro botones: “Grabar”, “Pausar”, “Terminar” y “Anular”. El usuario puede iniciar la

grabación de la consulta médica presionando el botón “Grabar”. También cuenta con las opciones de pausar o cancelar la grabación según sea necesario.

Al presionar el botón “Terminar”, el sistema finaliza la grabación y procede con la generación del informe. En este momento, el audio se envía al sistema para ser transcrito, identificado por roles y procesado, generando así los informes personalizados tanto para el paciente como para el médico.

Figura 4.5: Pantalla para iniciar la grabación

Pantalla de Confirmación: Una vez completada la grabación y generados los informes, el sistema muestra un mensaje de confirmación indicando que el informe ha sido generado exitosamente. Este mensaje informa que el informe será enviado tanto al correo del usuario como al correo del paciente, concluyendo así el proceso de grabación.

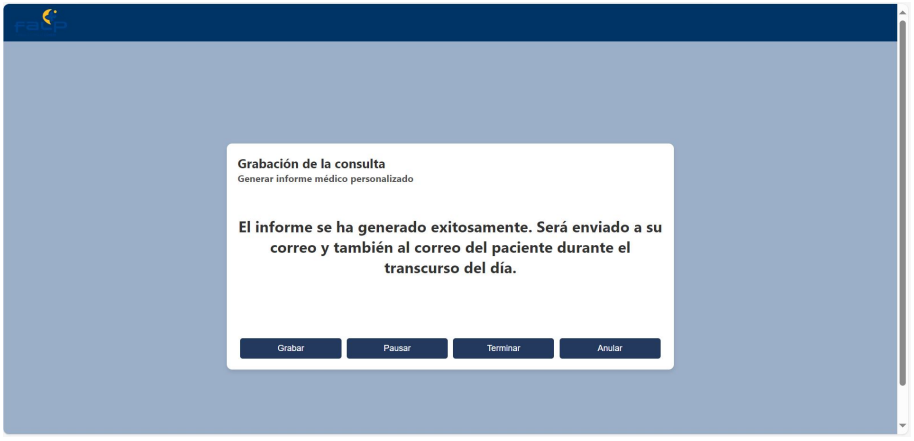


Figura 4.6: Confirmación de informe generado exitosamente

4.1. Gestión de Riesgos

4.1.1. Supuestos

1. **Feedback del cliente:** Se recibirá retroalimentación del cliente durante cada ciclo de desarrollo, lo que asegurará que el proyecto esté alineado con las expectativas del usuario.

2. **Calidad de audio:** Se asume que los micrófonos utilizados por los usuarios (médicos) serán de alta calidad, lo que garantizará que el audio capturado sea lo suficientemente claro para permitir transcripciones precisas a través del servicio AssemblyAI.

3. **Acceso continuo a Internet:** Se contará con acceso a internet estable y continuo para permitir la comunicación eficiente entre los módulos del sistema.

4. **Compatibilidad de dispositivos:** Los dispositivos utilizados por los médicos serán compatibles con el sistema y podrán ejecutar aplicaciones web sin problemas.

4.1.2. Dependencias

1. **Conexión a Internet Estable:** Es imprescindible contar con una conexión a internet continua y confiable en los lugares donde se utilizará el sistema, ya que la generación y envío de informes dependen de este acceso.

2. **Adquisición de Licencias y Derechos de Software:** Es necesario adquirir las licencias correspondientes para el uso de tecnologías y herramientas requeridas, incluidas las APIs externas de AssemblyAI y OpenAI.

4.1.3. Restricciones

1. **Dependencia de servicios externos:** El sistema depende de la disponibilidad y estabilidad de servicios externos como AssemblyAI y OpenAI. Cualquier interrupción o baja disponibilidad de estos servicios puede afectar el funcionamiento y la capacidad del sistema para generar informes de manera oportuna.

2. **Limitaciones en la transcripción en tiempo real:** La transcripción de audios y la generación de informes en tiempo real pueden verse limitadas por el tiempo de respuesta de servicios externos como AssemblyAI y OpenAI, así como por la capacidad de procesamiento del propio sistema, lo que podría afectar la rapidez de generación de los informes en situaciones de alta demanda.

3. **Calidad del entorno de grabación:** La calidad del audio puede verse afectada por el entorno en el que se realiza la grabación, como en consultas ruidosas o con interferencias. Esto puede dificultar la precisión de la transcripción y la correcta identificación de los hablantes.

4.1.4. Riesgos

Riesgo	**Medida de Mitigación**
Recursos limitados (servicios externos)	Considerar el uso de servicios alternativos en caso de que se excedan los límites de solicitudes a OpenAI o AssemblyAI.
Dependencia de servicios externos (AssemblyAI/OpenAI)	Mantener un plan de contingencia que incluya proveedores alternativos. Establecer un sistema que notifique al usuario en caso de fallas en estos servicios y permita reintentar el procesamiento.

Cuadro 4.1: Riesgos y Medidas de Mitigación

4.1.5. Pruebas y Audios de Simulación

El objetivo principal de este proyecto era realizar pruebas utilizando audios reales de consultas médicas, lo cual permitiría evaluar de manera precisa el funcionamiento del sistema en un entorno real. Sin embargo, debido a la dificultad de obtener el consentimiento de los pacientes para compartir audios de sus consultas, fue necesario crear audios simulados. Estos audios fueron generados por nuestro equipo, recreando situaciones de atención médica para simular escenarios de consultas reales. En total, se generaron aproximadamente

diez audios de prueba con las siguientes donde unas de las mas destacada son las siguientes :

- **Audio 1**: Conversación entre dos personas, un paciente y un médico. En esta prueba, el sistema logró segmentar correctamente el audio, identificar a los hablantes y generar los informes médicos de manera precisa.

- **Audio 2**: Conversación entre tres personas: un paciente, un médico y un acompañante. El sistema fue capaz de identificar correctamente a cada hablante y generar los informes médicos sin problemas.

- **Audio 3**: Conversación entre cuatro personas: un paciente, un médico y dos acompañantes. En esta prueba, el sistema generó un informe médico personalizado correctamente. Sin embargo, surgió un problema menor con la identificación del segundo acompañante, quien en ocasiones no fue reconocido de forma precisa. A pesar de esto, el problema no afectó significativamente la calidad del informe generado.

- **Audio 4**: Conversación entre tres personas con ruido de fondo. Aunque el sistema logró generar el informe, al revisar en detalle, observamos que algunas palabras no fueron detectadas correctamente debido al ruido. No obstante, este problema fue menor y no comprometió la integridad del informe.

- **Audio 5**: Conversación entre dos personas con ruidos fuertes e inesperados en el fondo. En esta prueba, se presentaron problemas en la segmentación del audio, lo que resultó en errores de transcripción. Debido a esto, el informe generado no coincidió completamente con las palabras originales de la conversación.

Durante el desarrollo del proyecto, fue necesario ajustar las instrucciones proporcionadas a OpenAI, ya que, inicialmente, el sistema no generaba correctamente los informes médicos a partir de las transcripciones de los diálogos. Se realizaron varias iteraciones en las instrucciones para lograr que el sistema identificara de manera precisa los roles de los participantes y generara informes de alta calidad, adaptados tanto para el médico como para el paciente.

A continuación, se presentará un ejemplo basado en el audio de una conversación con tres hablantes. Este audio tiene una duración aproximada de 10 minutos y fue utilizado para generar tanto el informe para el médico como el informe para el paciente. El tiempo de procesamiento para la generación de ambos informes fue de aproximadamente 30 segundos. A continuación, se muestra el JSON generado con las características respectivas de cada informe.


```
{
    "message": "Datos enviados al sistema de inyección",
    "response": {
        "nombre": "Fabian",
        "apellido": "Soto",
        "rut": "1111111-7",
        "informePaciente": "Síntomas: \nEl paciente ha experimentado un aumento en el cansancio y el dolor abdominal, especialmente después de
            comer. Además, ha tenido dificultades para dormir, con un promedio de 3 horas de sueño por noche en la última semana. También se ha
            sentido estresado.\n\nDiagnóstico: \nEl paciente tiene un tumor, cuyo tamaño no ha aumentado desde la última revisión, lo cual es
            positivo. Sin embargo, el aumento en el cansancio y el dolor abdominal son preocupantes. Se requiere una resonancia magnética (una
            prueba que usa imanes y ondas de radio para ver dentro del cuerpo) para investigar más a fondo estos síntomas.\n\nTratamiento: \nEl
            paciente debe someterse a una resonancia magnética lo más pronto posible, preferiblemente en la misma semana. Además, se le ha
            proporcionado un remedio natural para ayudar a aliviar sus síntomas. \n\nInformación adicional: \nEl médico ha enfatizado la
            importancia de realizar la resonancia magnética con urgencia para determinar la causa de los síntomas del paciente. El acompañante del
            paciente se ha comprometido a ayudar a programar los exámenes. El médico ha expresado su preocupación por la situación y ha asegurado
            al paciente y al acompañante que están trabajando en equipo para mejorar la salud del paciente.",
        "informeMedico": "El paciente presenta síntomas de fatiga intensa, insomnio y dolor abdominal, especialmente postprandial. Los resultados
            de los exámenes previos indican que el tumor existente no ha crecido, sin embargo, la intensificación de los síntomas es preocupante.
            Se ha recomendado una resonancia magnética para una evaluación más detallada de la condición del paciente. Se ha enfatizado la
            urgencia de realizar estos exámenes en la misma semana. Además, se ha proporcionado un tratamiento natural para ayudar al paciente a
            manejar sus síntomas actuales.",
        "codigoCIE10": "{\n\"C80\": \"Tumor maligno de sitio no especificado\",\n\"F43.0\": \"Trastorno de estrés agudo\",\n\"G47.0\":
            \"Insomnio\",\n\"R10.13\": \"Dolor epigástrico\",\n\"R53\": \"Malestar y fatiga\"\n}",
        "fecha": "2024-11-03T16:45:05.255Z"
    }
}
```


Figura 4.7: JSON generado a partir del audio de 10 minutos

La figura anterior muestra el JSON generado por la plataforma, el cual contiene los datos del paciente y los informes médicos creados a partir de la conversación. Este formato permite visualizar tanto la información básica del paciente como los detalles específicos de los informes generados para la consulta.

En el JSON generado se puede notar una diferencia clara entre el informe para el paciente y el informe para el médico. El informe del paciente es mucho más largo y detallado, ya que incluye secciones como síntomas, diagnóstico, tratamiento e información adicional importante. Esto ayuda a que el paciente entienda mejor su condición y los pasos a seguir. Además, en el informe del paciente, cuando se utilizan palabras técnicas, se incluye una breve explicación entre paréntesis para aclarar su significado, facilitando así la comprensión del paciente.

Por otro lado, el informe para el médico es mucho más breve y directo. Solo contiene un resumen rápido de los síntomas, el diagnóstico y el tratamiento. Este informe está diseñado

para que el médico tenga a la mano solo la información esencial sin detalles adicionales, lo que le permite una revisión rápida y eficiente de la consulta.

4.2. Resultados y Limitaciones de las Pruebas

Las pruebas realizadas en la plataforma han sido exitosas, logrando cumplir con el objetivo principal de generar informes médicos personalizados tanto para el médico como para el paciente. Sin embargo, es importante destacar que los audios utilizados fueron simulaciones, creadas específicamente para probar el sistema en escenarios controlados. Por esta razón, aún no es posible considerar el proyecto como finalizado, ya que el uso de audios reales en un entorno clínico podría presentar nuevos desafíos y modificar los resultados obtenidos.

Una de las principales desventajas identificadas en la plataforma es su alta dependencia de las instrucciones proporcionadas para procesar la información de manera adecuada. Debido a que cada escenario clínico puede variar considerablemente, se requiere ajustar las instrucciones en función del contexto específico. Esto puede derivar en una gran cantidad de instrucciones personalizadas, lo que aumenta la complejidad del sistema y su mantenimiento.

Además,Durante el uso del modelo **GPT-4** de **OpenAI**, se identificó que solo puede procesar alrededor de **8,000 palabras por solicitud**. Esto puede ser un problema en consultas médicas largas. Además, el costo del servicio aumenta con la cantidad de datos procesados.

Por otro lado, aunque AssemblyAI ha demostrado ser eficaz en la transcripción de audio a texto, se han observado errores cuando los ruidos ambientales son demasiado intensos. En situaciones donde el ruido de fondo es considerable, la precisión de la transcripción puede verse afectada, lo que podría influir en la calidad de los informes generados.

En conclusión, aunque el prototipo ha mostrado resultados prometedores en entornos simulados, será necesario realizar pruebas con audios reales y en situaciones clínicas reales para validar completamente su funcionamiento. Asimismo, se deben considerar

estrategias para optimizar las instrucciones y reducir los costos operativos, además de mejorar la capacidad de la plataforma para manejar condiciones adversas durante la captura del audio.