INSTITUTO SUPERIOR TECNOLÓGICO UNIVERSITARIO RUMIÑAHUI

ESCUELA DE POSGRADOS

MAESTRÍA TECNOLÓGICA EN HERRAMIENTAS DIGITALES PARA EL ANÁLISIS DE DATOS E INTELIGENCIA ARTIFICIAL

Trabajo de titulación previo a la obtención del Título en Magister Tecnológico en Herramientas Digitales Pare el Análisis de Datos e Inteligencia Artificial.

Tema: Diseño y Desarrollo de una Aplicación Interactiva que Utiliza Inteligencia Artificial para la Práctica de Entrevistas Laborales en Inglés Dirigida a Hablantes no Nativos.

Autor: Gabriel Enrique Pacheco Maldonado

Director: PhD. Juan Carlos Minango Negrete

Fecha: 13 de septiembre de 2024-

Sangolquí – Ecuador

Autor: Pacheco Maldonado Gabriel Enrique.

Título a obtener: Magister Tecnológico en Herramientas Digitales Para el Análisis de Datos e Inteligencia Artificial.

Matriz: Sangolquí -Ecuador

Correo electrónico: gabriel.pacheco@ister.edu.ec

Dirigido por: Minango Negrete Juan Carlos.

Título: PhD.

Matriz: Sangolquí -Ecuador

Correo electrónico: juancarlos.minango@ister.edu.ec

APROBACIÓN DEL DIRECTOR DEL TRABAJO TITULACIÓN

Sangolquí, 13 de Septiembre del 2024

MSc. Elizabeth Aldás
Directora de Posgrados
Instituto Superior Tecnológico Universitario Rumiñahui
Presente

De mi consideración:

Me permito comunicar que, en calidad de director del presente Trabajo de Titulación denominado: Diseño y desarrollo de una aplicación interactiva que utiliza inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos realizado por Gabriel Enrique Pacheco Maldonado ha sido orientado y revisado durante su ejecución, así mismo ha sido verificado a través de la herramienta de similitud académica institucional, y cuenta con un porcentaje de coincidencia aceptable. En virtud de ello, y por considerar que el mismo cumple con todos los parámetros establecidos por la institución, doy mi aprobación a fin de continuar con el proceso académico correspondiente.

Particular que comunico para los fines pertinentes.

Atentamente,

Firmado electrónicamente por:
JUAN CARLOS
MINANGO NEGRETE

PhD. Juan Carlos Minango Negrete
Director del Trabajo de Titulación
C.I.: 171604244-3
Correo electrónico: juancarlos.minango@ister.edu.ec

CARTA DE CESIÓN DE DERECHOS DEL TRABAJO DE TITULACIÓN

Sangolquí,13 de Septiembre del 2024

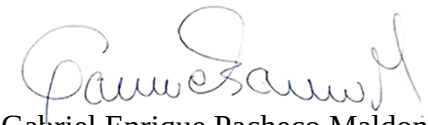
MSc. Elizabeth Aldás
Directora de Posgrados
Instituto Superior Tecnológico Universitario Rumiñahui
Presente

Por medio de la presente, yo, Gabriel Enrique Pacheco Maldonado, declaro y acepto en forma expresa lo siguiente: ser autor del trabajo de titulación denominado “Diseño y desarrollo de una aplicación interactiva que utiliza inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos", de la Maestría Tecnológica en Herramientas Digitales e Inteligencia Artificial; manifiesto mi voluntad de ceder al Instituto Superior Tecnológico Universitario Rumiñahui los derechos de reproducción, distribución y publicación de dicho trabajo de titulación, en cualquier formato y medio, con fines académicos y de investigación.

Esta cesión se otorga de manera no exclusiva y por un periodo indeterminado. Sin embargo, conservo los derechos morales sobre mi obra.

En fe de lo cual, firmo la presente.

Atentamente,



Gabriel Enrique Pacheco Maldonado

CI: 1713781878

FORMULARIO PARA ENTREGA DEL TRABAJO DE TITULACIÓN EN BIBLIOTECA DEL INSTITUTO SUPERIOR TECNOLÓGICO UNIVERSITARIO RUMIÑAHUI

MAESTRÍA TECNOLÓGICA EN HERAMIENTAS DIGITALES PARA EL ANALISIS DE DATOS E INTELIGENCIA ARTIFICIAL

AUTOR:
Gabriel Enrique Pacheco Maldonado

TUTOR:
PhD. Juan Carlos Minango Negrete

CONTACTO ESTUDIANTE:
(+593) 992001168

CORREO ELECTRÓNICO:
Gabriel.pacheco@ister.edu.ec

TEMA:
Diseño y desarrollo de una aplicación interactiva que utiliza inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos.

RESUMEN EN ESPAÑOL:
El proyecto titulado "Diseño y desarrollo de una aplicación interactiva que utiliza inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos" se centra en la creación de una herramienta digital que facilite la preparación de entrevistas en inglés para personas que no tienen el idioma como lengua materna. Esta aplicación se apoya en tecnologías avanzadas como reconocimiento de voz, procesamiento de lenguaje natural y retroalimentación en tiempo real, ofreciendo una simulación efectiva del proceso de entrevista. El marco teórico aborda los conceptos clave relacionados con la inteligencia artificial, aprendizaje automático, reconocimiento de voz y sistemas de retroalimentación automática. Por su parte, el marco metodológico describe el proceso de desarrollo de la aplicación, desde la selección de tecnologías hasta la integración de IA en las funcionalidades clave, tales como el reconocimiento de respuestas y corrección de errores.
Entre las conclusiones más destacadas, se resalta que la herramienta tiene el potencial de mejorar considerablemente la preparación de entrevistas laborales en inglés, proporcionando a los usuarios un entorno interactivo que permite mejorar sus habilidades comunicativas.

Asimismo, se proponen recomendaciones para el desarrollo futuro de la aplicación, como la ampliación de funcionalidades y la optimización del sistema de retroalimentación.
Finalmente, el proyecto no solo plantea un enfoque innovador en el aprendizaje de idiomas y preparación profesional, sino que también sugiere una base sólida para el desarrollo continuo, permitiendo escalar la aplicación hacia un entorno más robusto y completo.

PALABRAS CLAVE:
Inteligencia Artificial, Procesamiento de Lenguaje Natural, Reconocimiento de Voz, Aplicación Interactiva, Entrevistas Laborales, Práctica de Inglés, Retroalimentación, Simulación, Aprendizaje Automático, Desarrollo de Aplicaciones.

ABSTRACT:
This project presents the design and development of an interactive application using artificial intelligence to simulate job interviews in English for non-native speakers. The system leverages Natural Language Processing (NLP) and voice recognition technologies to create a realistic interview environment, enabling users to practice and improve their English skills. The app provides immediate feedback, focusing on grammatical corrections and fluency, aiming to enhance user readiness for real-life job interviews. Key technologies used include OpenAI for question generation and feedback, React for the frontend, Firebase for backend integration, and Tailwind for design optimization. The application fosters continuous improvement and user engagement through personalized interactions.

KEYWORDS:
Keywords: Artificial Intelligence, Natural Language Processing, Speech Recognition, Interactive Application, Job Interviews, English Practice, Feedback, Simulation, Machine Learning, App Development.

SOLICITUD DE PUBLICACIÓN DEL TRABAJO DE TITULACIÓN

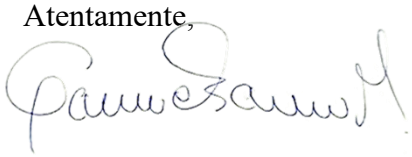
Sangolquí, 13 de Septiembre del 2024

MSc. Elizabeth Aldás
Directora de Posgrados
Instituto Superior Tecnológico Universitario Rumiñahui
Presente

A través del presente me permito aceptar la publicación del trabajo de titulación denominado: Diseño y desarrollo de una aplicación interactiva que utiliza inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos de la Unidad de Integración Curricular en el repositorio digital "DsPace" del estudiante: Gabriel Enrique Pacheco Maldinado con documento de identificación No 1713781878 estudiante de la Maestría Tecnológica en Herramientas Digitales Para el Análisis de Datos e Inteligencia Artificial.

El trabajo ha sido revisado las similitudes en el software "TURNITING" y cuenta con un porcentaje máximo de 15%; motivo por el cual, el Trabajo de titulación es publicable.

Atentamente,



Gabriel Pacheco
CI: 1713781878

AGRADECIMIENTO

Quiero agradecerme a mí mismo. Como dijo (Snoop Dog, 2018): "Quiero agradecerme por creer en mí. Quiero agradecerme por hacer todo este trabajo duro. Quiero agradecerme por no tener días libres. Quiero agradecerme por no rendirme nunca. Quiero agradecerme por ser siempre un dador y tratar de dar más de lo que recibo."

DEDICATORIA

Este trabajo es un reflejo de mi esfuerzo, me lo dedico a mí mismo por haber enfrentado el momento más difícil de mi vida sin perder la fe en lo que soy capaz de lograr. A pesar de las adversidades, seguí adelante y lo entregué todo, con más fuerza que nunca. Este trabajo es una prueba de mi capacidad para superar desafíos y mantenerme firme en mi propósito, demostrando que no hay obstáculo que pueda detener mi determinación.

RESUMEN

El proyecto titulado "Diseño y desarrollo de una aplicación interactiva que utiliza inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos" se centra en la creación de una herramienta digital que facilite la preparación de entrevistas en inglés para personas que no tienen el idioma como lengua materna. Esta aplicación se apoya en tecnologías avanzadas como reconocimiento de voz, procesamiento de lenguaje natural y retroalimentación en tiempo real, ofreciendo una simulación efectiva del proceso de entrevista. El marco teórico aborda los conceptos clave relacionados con la inteligencia artificial, aprendizaje automático, reconocimiento de voz y sistemas de retroalimentación automática. Por su parte, el marco metodológico describe el proceso de desarrollo de la aplicación, desde la selección de tecnologías hasta la integración de IA en las funcionalidades clave, tales como el reconocimiento de respuestas y corrección de errores.

Entre las conclusiones más destacadas, se resalta que la herramienta tiene el potencial de mejorar considerablemente la preparación de entrevistas laborales en inglés, proporcionando a los usuarios un entorno interactivo que permite mejorar sus habilidades comunicativas. Asimismo, se proponen recomendaciones para el desarrollo futuro de la aplicación, como la ampliación de funcionalidades y la optimización del sistema de retroalimentación. Finalmente, el proyecto no solo plantea un enfoque innovador en el aprendizaje de idiomas y preparación profesional, sino que también sugiere una base sólida para el desarrollo continuo, permitiendo escalar la aplicación hacia un entorno más robusto y completo.

Palabras Clave: Inteligencia Artificial, Procesamiento de Lenguaje Natural, Reconocimiento de Voz, Aplicación Interactiva, Entrevistas Laborales, Práctica de Inglés, Retroalimentación, Simulación, Aprendizaje Automático, Desarrollo de Aplicaciones.

ABSTRACT

This project presents the design and development of an interactive application using artificial intelligence to simulate job interviews in English for non-native speakers. The system leverages Natural Language Processing (NLP) and voice recognition technologies to create a realistic interview environment, enabling users to practice and improve their English skills. The app provides immediate feedback, focusing on grammatical corrections and fluency, aiming to enhance user readiness for real-life job interviews. Key technologies used include OpenAI for question generation and feedback, React for the frontend, Firebase for backend integration, and Tailwind for design optimization. The application fosters continuous improvement and user engagement through personalized interactions.

Keywords: Artificial Intelligence, Natural Language Processing, Speech Recognition, Interactive Application, Job Interviews, English Practice, Feedback, Simulation, Machine Learning, App Development.

Índice de Contenido

Índice de Tablas

Índice de Ilustraciones

1. INTRODUCCIÓN

1.1. TEMA

Desarrollo de una aplicación interactiva que usa inteligencia artificial (IA) para la práctica de entrevistas laborales en inglés dirigidas a hablantes no nativos. La finalidad de la aplicación es mejorar las habilidades de los usuarios en el contexto de entrevistas de trabajo, proporcionando un entorno simulado donde puedan practicar y recibir retroalimentación en tiempo real sobre su uso del inglés, con un enfoque en gramática y fluidez.

El tema responde a la creciente necesidad de profesionales que puedan desenvolverse con éxito en entrevistas en inglés, especialmente en sectores como la tecnología, donde este idioma es dominante. El uso de IA permite automatizar el rol del entrevistador, ajustando el nivel de dificultad en función de las habilidades del usuario, ofreciendo una experiencia personalizada de aprendizaje y práctica.

1.2. PLANTEMAIENTO DEL PROBLEMA.

En el contexto global actual, el dominio del inglés se ha convertido en una competencia esencial para acceder a numerosas oportunidades laborales, especialmente en el sector tecnológico, donde las entrevistas laborales en inglés son un requisito común. Sin embargo, muchos profesionales no nativos enfrentan barreras significativas debido a su nivel básico de inglés, lo que limita sus posibilidades de éxito en estas entrevistas. A menudo, la falta de acceso a un entorno de práctica adecuado, combinado con el temor a cometer errores, impide que estos profesionales mejoren sus habilidades de forma efectiva.

La problemática central radica en la ausencia de herramientas interactivas y accesibles que permitan a los hablantes no nativos de inglés practicar entrevistas laborales en este idioma de manera realista y personalizada. Aunque existen recursos tradicionales como clases de idiomas o ejercicios de gramática, estos no ofrecen un enfoque práctico y específico para situaciones laborales, ni proporcionan la retroalimentación inmediata que los usuarios necesitan para corregir sus errores y mejorar de manera progresiva.

En consecuencia, los candidatos que buscan mejorar sus competencias en entrevistas laborales en inglés carecen de un entorno donde puedan simular escenarios reales, practicar preguntas típicas y recibir correcciones instantáneas sobre aspectos clave como la gramática, la pronunciación y la fluidez. Esto provoca que muchos se sientan inseguros y mal preparados al enfrentar entrevistas reales, lo que impacta negativamente en sus posibilidades de obtener un empleo, especialmente en el competitivo mercado tecnológico.

La carencia de una solución accesible y personalizada para la práctica de entrevistas laborales en inglés genera la necesidad de desarrollar una herramienta que permita a los

usuarios mejorar sus habilidades lingüísticas en un entorno controlado, a su propio ritmo y de acuerdo con su nivel de inglés. Esta aplicación no solo debe simular el rol de un entrevistador, sino también proporcionar una retroalimentación específica y detallada que impulse la mejora continua.

Es fundamental abordar este problema para incrementar las oportunidades laborales de profesionales no nativos de inglés, facilitando su acceso a empleos que requieren competencias comunicativas en este idioma, especialmente en roles de alta demanda como el de ingeniero de software, donde el inglés es el estándar de comunicación.

1.3. OBJETIVO.

Desarrollar una aplicación interactiva basada en inteligencia artificial que permita a hablantes no nativos de inglés practicar entrevistas laborales de manera realista, con correcciones en tiempo real sobre gramática, pronunciación y fluidez, mejorando así sus competencias lingüísticas para enfrentar entrevistas en inglés.

1.4. OBJETIVOS ESPECIFICOS.

Diseñar un sistema de entrevistas simuladas que imite el formato de entrevistas laborales en inglés, adaptándose al nivel de idioma del usuario y al puesto de trabajo deseado.

Implementar la integración de un modelo de IA que actúe como entrevistador, proporcionando preguntas claras y adaptadas al nivel de inglés del usuario, así como retroalimentación en tiempo real sobre su desempeño.

Incorporar tecnologías de reconocimiento y síntesis de voz para que el usuario pueda interactuar con el sistema mediante el uso de comandos de voz, facilitando la práctica de conversación fluida.

Evaluar la efectividad de la aplicación mediante métricas de mejora en las habilidades de los usuarios, obteniendo retroalimentación y adaptando el sistema para maximizar el aprendizaje y preparación para entrevistas reales.

1.5. JUSTIFICACIÓN.

En el contexto actual, el inglés se ha consolidado como una lengua global esencial, particularmente en el ámbito laboral. Para hablantes no nativos, la falta de dominio del idioma puede representar una barrera significativa al momento de postularse a puestos de trabajo que exigen entrevistas en inglés. Esta situación es aún más desafiante cuando los candidatos carecen de oportunidades de practicar en un entorno realista que les permita recibir retroalimentación específica sobre sus errores gramaticales, de pronunciación y fluidez.

La implementación de una aplicación interactiva que utiliza inteligencia artificial para simular entrevistas laborales responde a esta necesidad. Ofrecer un entorno donde los usuarios puedan practicar de forma personalizada no solo mejora sus habilidades lingüísticas, sino que también incrementa su confianza y preparación para enfrentar entrevistas reales en inglés. Además, el uso de tecnologías como Speech-to-Text y Text-to-Speech hace que la experiencia sea más inmersiva y cercana a situaciones reales de interacción verbal.

El proyecto es altamente relevante debido a la creciente demanda de perfiles profesionales con competencias en inglés, especialmente en áreas técnicas como la ingeniería de software, donde la competencia es internacional. A través de esta aplicación, se contribuye a la inserción laboral de un segmento de la población que de otro modo podría quedar excluido por su nivel de inglés, ayudando a superar esta barrera con un enfoque práctico y accesible.

1.6. IDEA A DEFENDER.

El desarrollo de una aplicación interactiva basada en inteligencia artificial para la práctica de entrevistas laborales en inglés dirigida a hablantes no nativos es una solución innovadora y eficaz para mejorar las habilidades comunicativas y de empleabilidad en un

mercado globalizado. Este enfoque permite a los usuarios prepararse de manera personalizada, superando barreras lingüísticas y técnicas a través de la simulación de entrevistas reales, lo que incrementa sus oportunidades de éxito en procesos de selección laborales en inglés.

2. CAPITULO I

MARCO TEÓRICO

2.1. Inteligencia Artificial

La Inteligencia Artificial (IA) se ha consolidado como una de las tecnologías más disruptivas en la actualidad, con aplicaciones en diversos campos. En términos simples, la IA se refiere a la capacidad de las máquinas para realizar tareas que normalmente requieren inteligencia humana, como el reconocimiento de patrones, la toma de decisiones, y el aprendizaje autónomo. (Brown & al., 2020)

En el contexto de este proyecto, se utiliza la IA como una herramienta para simular entrevistas laborales en inglés, proporcionando correcciones gramaticales y feedback personalizado a hablantes no nativos. Esto se alinea con los avances recientes en procesamiento del lenguaje natural (NLP), que permiten a las máquinas comprender, procesar y generar lenguaje humano de manera efectiva. (Young, 2018)

2.2. Procesamiento del Lenguaje Natural (NLP)

El procesamiento del lenguaje natural (NLP) es una rama de la IA que se enfoca en la interacción entre las computadoras y el lenguaje humano. El NLP combina la lingüística computacional y el aprendizaje automático para permitir a las máquinas interpretar y responder a entradas en lenguaje natural. Entre sus principales aplicaciones se encuentran los chatbots, asistentes virtuales y sistemas de traducción automática. (Jurafsky & Martin, 2021)

Para el desarrollo de este proyecto, el NLP es clave, ya que el sistema debe entender y analizar las respuestas del usuario, corregir errores gramaticales y proporcionar retroalimentación clara. Herramientas como el modelo GPT-3.5, desarrollado por OpenAI, son

ejemplos de sistemas avanzados de NLP que generan respuestas coherentes y precisas en lenguaje humano.

2.3.Modelos de Lenguaje Grandes (LLM)

Los modelos de lenguaje grandes (*Large Language Models, LLM*) son modelos de aprendizaje profundo entrenados con enormes cantidades de datos de texto. Estos modelos, como GPT-3 y GPT-4, utilizan redes neuronales avanzadas para generar, entender y manipular texto de manera coherente y significativa. Los LLM se destacan por su capacidad para realizar una amplia gama de tareas relacionadas con el lenguaje, como traducción, resumen de texto, generación de contenido y diálogos conversacionales. (Radford A. , 2019)

La clave de los LLM es su tamaño y la cantidad de parámetros que pueden procesar, lo que les permite capturar matices complejos del lenguaje humano. Utilizan técnicas de atención para enfocarse en diferentes partes del texto de entrada y generar respuestas que tienen en cuenta el contexto a lo largo de una conversación o documento.

2.4. Modelos de Lenguaje: OpenAI y GPT-3.5

Los modelos de lenguaje como GPT-3.5 representan un avance significativo en la creación de sistemas de IA conversacionales. Estos modelos utilizan grandes volúmenes de datos para generar lenguaje humano con un alto grado de coherencia. El GPT-3.5, en particular, se entrena en una amplia variedad de textos, lo que le permite comprender contextos complejos y generar respuestas apropiadas para diversas situaciones. (Brown & al., 2020)

2.5.Redes Neuronales Profundas (DNN)

Las **redes neuronales profundas (*Deep Neural Networks, DNN*)** son un tipo de red neuronal artificial con múltiples capas entre la entrada y la salida. Estas capas permiten que la

red aprenda representaciones más complejas de los datos, haciendo que las DNN sean extremadamente útiles en tareas de procesamiento de lenguaje, visión artificial y reconocimiento de patrones. Las DNN han impulsado avances significativos en aplicaciones como el reconocimiento de voz y la generación de texto, que son esenciales para sistemas de entrevistas simuladas basados en IA.

Las DNN consisten en neuronas artificiales organizadas en capas: una capa de entrada, una o más capas ocultas y una capa de salida. Cada neurona procesa información mediante una función de activación y pasa los resultados a la siguiente capa. Los pesos entre las neuronas se ajustan durante el entrenamiento para minimizar los errores entre la predicción del modelo y la realidad, lo que permite al modelo mejorar con el tiempo.

2.6.Transferencia de Aprendizaje (*Transfer Learning*)

El transfer learning o aprendizaje por transferencia es una técnica que permite reutilizar un modelo previamente entrenado para una tarea específica y adaptarlo a una nueva tarea relacionada. En lugar de entrenar un modelo desde cero, el transfer learning aprovecha los conocimientos adquiridos en el modelo original, lo que reduce significativamente el tiempo de entrenamiento y mejora el rendimiento en tareas con menos datos disponibles.

2.7. Fine-Tuning

El *fine-tuning* es un proceso de ajuste fino de un modelo preentrenado para una tarea específica. Consiste en tomar un modelo entrenado con un gran corpus de datos generales y adaptarlo a un conjunto de datos más pequeño y específico, ajustando sus parámetros para optimizar su rendimiento en esa nueva tarea. Este proceso es particularmente efectivo en aplicaciones de procesamiento de lenguaje natural, donde se requiere que el modelo responda

a tareas especializadas, como simulaciones de entrevistas o corrección gramatical en diálogos. (Howard, 2018)

2.8.Reconocimiento Automático de Voz (ASR)

El reconocimiento automático de voz (*Automatic Speech Recognition*, ASR) es una tecnología que convierte el habla en texto mediante algoritmos de procesamiento de señales. ASR se basa en modelos de aprendizaje automático y redes neuronales profundas que son entrenados con grandes cantidades de datos de audio para identificar patrones en el lenguaje hablado. Este proceso incluye la extracción de características acústicas y su mapeo a secuencias de texto. (Graves & Schmidhuber, 2014)

El reconocimiento de voz es un componente crucial en aplicaciones que requieren interacción por medio de comandos hablados, ya que permite que los sistemas comprendan el lenguaje humano en tiempo real. ASR ha evolucionado con el uso de modelos avanzados como los de redes neuronales recurrentes (RNN) y modelos de atención, mejorando su precisión, especialmente en entornos ruidosos o con variaciones en acentos y pronunciaciones. (Bahdanau, Cho, & Bengio, 2014)

2.9. Procesamiento en Tiempo Real

El procesamiento en tiempo real se refiere a la capacidad de un sistema para analizar y generar respuestas instantáneamente, en el momento en que se recibe una entrada. En aplicaciones de lenguaje natural, como asistentes virtuales y chatbots, es crucial que el procesamiento en tiempo real sea eficiente para proporcionar respuestas inmediatas y precisas al usuario. (Krizhevsky, Sutskever, & Hinton, 2012)

El tiempo real también juega un papel importante en sistemas que dependen de la voz, donde la entrada de audio debe ser convertida rápidamente en texto y procesada por el modelo para generar una respuesta casi instantánea. Esto permite una experiencia fluida e interactiva.

2.10. Bases de Datos NoSQL

Las bases de datos *NoSQL* son sistemas de gestión de bases de datos que no siguen el modelo relacional tradicional. En lugar de utilizar tablas y columnas como las bases de datos *SQL*, las *NoSQL* permiten almacenar y gestionar datos en formatos más flexibles, como documentos, clave-valor, gráficos y columnas anchas. Estas bases de datos son especialmente útiles para aplicaciones que manejan grandes volúmenes de datos no estructurados o semi-estructurados, como redes sociales, sistemas de recomendación, y análisis de grandes cantidades de texto. (Abelló & García-Solache)

Un ejemplo común es *Firebase*, una base de datos *NoSQL* basada en documentos, que almacena la información en formato *JSON*. La naturaleza flexible de *Firebase* permite el almacenamiento de datos sin la rigidez de los esquemas tradicionales, lo que facilita su uso en aplicaciones modernas de tiempo real, como chats y *dashboards* interactivos. (Firebase Documentation, 2024)

2.11. Firebase y Autenticación

Firebase es una plataforma de desarrollo de aplicaciones web y móviles que proporciona servicios backend como bases de datos en tiempo real, almacenamiento en la nube, autenticación y hosting. Uno de sus servicios más importantes es Firebase Authentication, que facilita la gestión del acceso de usuarios mediante diferentes métodos de autenticación, como correo electrónico y contraseña, Google, Facebook, o incluso proveedores personalizados. (Firebase Documentation, 2024)

Este servicio es fundamental para proteger los datos del usuario y garantizar que solo personas autorizadas accedan a las funcionalidades de la aplicación. Además, Firebase facilita la integración con otras plataformas, como React, para gestionar la autenticación y los permisos de manera sencilla y escalable.

2.12. React y la Construcción de Interfaces Dinámicas

React es una biblioteca de JavaScript desarrollada por Facebook para la creación de interfaces de usuario. Se basa en el uso de componentes reutilizables que permiten desarrollar aplicaciones web de manera modular y eficiente. React es especialmente popular en el desarrollo de aplicaciones de una sola página (SPA), donde el contenido se actualiza dinámicamente sin necesidad de recargar toda la página. (Erikson, 2017)

Uno de los puntos clave de React es su capacidad para gestionar el estado de la aplicación utilizando herramientas como hooks y context, lo que permite que los desarrolladores gestionen datos de manera eficiente y reactiva. Su combinación con Firebase permite crear aplicaciones interactivas y en tiempo real, como chats o paneles de control.

2.13. API: Definición e Importancia

Una API (*Application Programming Interface*) es un conjunto de reglas y protocolos que permiten a diferentes aplicaciones comunicarse entre sí. Las APIs son fundamentales en el desarrollo de software moderno, ya que facilitan la interacción entre diferentes servicios y plataformas. En el contexto del desarrollo web, las APIs permiten que aplicaciones frontend (como las desarrolladas en React) se comuniquen con servicios backend (como bases de datos o servicios en la nube). (Axelrod & Hamilton, 2015)

Las APIs REST (*Representational State Transfer*) son uno de los enfoques más comunes, ya que permiten realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) a través de solicitudes HTTP. En el desarrollo de aplicaciones con Firebase y React, las APIs juegan un papel clave para enviar y recibir datos de manera eficiente. (Axelrod & Hamilton, 2015)

2.14. Tailwind CSS

Tailwind CSS es un framework de utilidades de CSS que permite diseñar interfaces de usuario de manera rápida y eficiente. A diferencia de los frameworks CSS tradicionales como Bootstrap, que proporcionan componentes predefinidos, Tailwind ofrece una gran variedad de clases de utilidad de bajo nivel, lo que da mayor control sobre el diseño sin tener que escribir CSS personalizado. (Tailwind CSS, 2024)

Tailwind es ideal para aplicaciones que requieren personalización del diseño sin depender de componentes rígidos, como ocurre en proyectos donde el diseño es parte esencial de la experiencia de usuario.

2.15. Vite

Vite es una herramienta de construcción moderna para aplicaciones web que mejora considerablemente el proceso de desarrollo, ofreciendo tiempos de compilación y recarga en caliente mucho más rápidos que las herramientas tradicionales como Webpack. Vite utiliza ES Modules (ESM) y se enfoca en optimizar tanto la experiencia de desarrollo como el rendimiento en producción. (Smith, 2023)

2.15.1. Características clave de Vite:

Desarrollo rápido: Vite proporciona un servidor de desarrollo que usa ESM nativo en el navegador, lo que permite que solo los módulos necesarios sean cargados y ejecutados, resultando en tiempos de arranque muy cortos.

Compilación optimizada: Vite usa Rollup bajo el capó para empaquetar la aplicación para producción, generando un bundle altamente optimizado.

Integración con frameworks modernos: Vite ofrece soporte nativo para frameworks populares como React, Vue, y Svelte, proporcionando configuraciones preestablecidas que agilizan la configuración y el desarrollo.

2.16. Aplicaciones Web

Una aplicación web es un programa que se ejecuta en un navegador a través de Internet. A diferencia de las aplicaciones de escritorio, las aplicaciones web no requieren instalación en el dispositivo del usuario y están disponibles de manera instantánea desde cualquier lugar con conexión a Internet. (Duckett, 2011) Estas aplicaciones son altamente interactivas y dinámicas, gracias al uso de tecnologías como HTML, CSS, JavaScript, y marcos de trabajo como React o Angular.

2.17. Entrevistas de Trabajo

Las entrevistas de trabajo son procesos formales donde el entrevistador evalúa las habilidades, experiencia y actitud de un candidato para determinar su idoneidad para un puesto. Las entrevistas son un paso crucial en el proceso de contratación y pueden adoptar diferentes formatos. (Barrick, Shaffer, & DeGrassi, 2009)

2.18. Entrevistas de Trabajo en Inglés

Las entrevistas de trabajo en inglés son comunes en empresas internacionales y en sectores que requieren el dominio del idioma, como la tecnología, finanzas y marketing. En estas entrevistas, se evalúan tanto las habilidades profesionales como la capacidad del candidato para comunicarse en inglés. (Barrick, Shaffer, & DeGrassi, 2009)

2.19. Gestión de Estados en Aplicaciones Web

La gestión de estados es un aspecto crucial en aplicaciones web interactivas. Se refiere a cómo una aplicación maneja y sincroniza los datos que cambian a lo largo del uso de la aplicación. En aplicaciones complejas, el estado puede incluir información sobre el usuario, las preferencias, datos de formularios, y otros elementos que cambian dinámicamente. Herramientas como React utilizan soluciones como el Context API o Redux para gestionar estos estados de manera eficiente, asegurando que los componentes de la interfaz se actualicen correctamente cuando los datos cambian. (Medina, 2022)

2.20. Text-to-Speech (TTS)

Text-to-Speech (TTS) es una tecnología que convierte texto escrito en audio hablado. Utiliza un motor de síntesis de voz para generar una representación auditiva del contenido escrito, facilitando la interacción con aplicaciones que requieren accesibilidad o respuestas de voz automatizadas. TTS es crucial en muchas aplicaciones, incluyendo asistentes virtuales, sistemas de navegación, y herramientas de aprendizaje de idiomas. (Jurafsky & Martin, 2021)

2.20.1. Componentes clave del TTS

Entrada de texto: El texto que será procesado y convertido en habla.

Linguistic analysis: El análisis lingüístico interpreta el texto para identificar la estructura gramatical y semántica.

Motor de síntesis de voz: Genera la voz a partir del texto utilizando diferentes técnicas, como la concatenación de fragmentos pregrabados o la síntesis basada en modelos de redes neuronales.

2.21. Speech-to-Text (STT)

Speech-to-Text (STT) es la tecnología inversa de TTS, y convierte el audio hablado en texto. Utiliza reconocimiento de voz para transcribir la entrada de voz de los usuarios, permitiendo interacciones más fluidas y naturales en aplicaciones como chatbots, asistentes de voz, o dictado en tiempo real. (Dutoit, 1997)

2.21.1. Componentes clave del STT:

Captura de audio: El sistema recoge el audio del usuario a través de un micrófono.

Reconocimiento de voz: Mediante algoritmos de procesamiento de lenguaje natural (NLP) y modelos de reconocimiento de patrones, el audio se convierte en texto.

Post-procesamiento: Correcciones de gramática, puntuación y adecuación del texto resultante para adaptarlo al contexto o requerimientos específicos de la aplicación.

3. CAPITULO II

MARCO METODOLÓGICO

En este capítulo se define las estrategias utilizadas para el desarrollo de la aplicación interactiva que usa inteligencia artificial. En este proyecto, se adoptó una metodología de desarrollo ágil para gestionar de manera eficiente las etapas de diseño, implementación y pruebas. La elección de esta metodología se basó en la capacidad para adaptarse a cambios continuos y permitir una integración rápida de retroalimentación, lo que es crucial en el desarrollo de aplicaciones centradas en la experiencia del usuario.

3.1. Metodología de Desarrollo.

Para el desarrollo del proyecto, se utilizó la metodología ágil Scrum, un marco ágil que facilita la planificación y ejecución en ciclos cortos de trabajo llamados *Sprints*. Esta metodología permitió dividir el desarrollo en fases más pequeñas y manejables, con entregas incrementales. Cada Sprint duró entre una y dos semanas, y al final de cada uno, se revisaron los avances y se adaptaron las funcionalidades en base a la retroalimentación. Podemos ver la planificación del desarrollo en la *Tabla 1*.

Tabla 1 Metodología de Desarrollo.

Fase	Actividad	Descripción	Responsable	Duración (días)
Análisis	Recolección de requisitos	Reuniones con stakeholders para definir requisitos	Gabriel Pacheco	7

Diseño	Diseño del sistema y arquitectura	Creación de diagramas y diseño de la arquitectura	Gabriel Pacheco	10
Desarrollo	Implementación de funcionalidades	Codificación de módulos y funcionalidades	Gabriel Pacheco	20
Pruebas	Pruebas de integración y validación	Pruebas de funcionalidad y corrección de errores	Gabriel Pacheco	10
Implementación	Despliegue del sistema	Implementación del sistema en el entorno de producción	Gabriel Pacheco	5
Mantenimiento	Soporte y mantenimiento post-lanzamiento	Monitoreo y resolución de problemas	Gabriel Pacheco	Continuo

3.2.Product Backlog.

El *Product Backlog* es una lista priorizada de requisitos y características que el producto debe cumplir. La *Tabla 2* presenta los elementos clave del backlog, describiendo cada requisito o funcionalidad, su prioridad, estado actual y la estimación de tiempo necesaria para su implementación. La correcta gestión del *Product Backlog* permite una planificación efectiva y asegura que los recursos se dirijan a las tareas más importantes y valiosas para el proyecto.

Tabla 2 Product Backlog

ID	Elemento	Descripción	Prioridad	Días
1	Requerimiento de autenticación	Implementar un sistema de autenticación de usuarios	Alta	5
2	Funcionalidad de entrevista	Crear módulos para realizar entrevistas simuladas	Alta	8
3	Integración de IA para evaluación	Implementar IA para evaluar respuestas de entrevistas	Alta	10
4	Interfaz de usuario	Diseñar y desarrollar la interfaz de usuario	Alta	7
5	Reportes de progreso	Implementar sistema de reportes de progreso	Baja	4

3.3.Planificación de Funcionalidades.

En la fase de planificación, se definieron las funcionalidades clave que la aplicación debía tener para cumplir con los objetivos propuestos. Estas funcionalidades incluyeron:

Registro de Usuario: Este ítem específico enfoca en la implementación de la funcionalidad de registro de usuarios, que incluye la creación de la interfaz para el registro y la integración con Firebase para la autenticación.

Sistema de entrevistas simuladas: Un módulo que genera preguntas basadas en el nivel de inglés del usuario y el contexto de la entrevista.

Reconocimiento de voz y conversión de texto a voz: Un componente crucial para que los usuarios puedan interactuar con la aplicación mediante comandos de voz.

Evaluación de respuestas: Un sistema de evaluación que proporciona retroalimentación en tiempo real sobre el desempeño del usuario.

Cada una de estas funcionalidades fue integrada y probada en ciclos de desarrollo iterativos. La *Tabla 3* muestra la planificación de otras funcionalidades.

Tabla 3 Planificación de funcionalidades

Funcionalidad	Descripción	Prioridad	Responsable
Creación de Usuario	Permite a los usuarios crear nuevas cuentas en el sistema.	Alta	Gabriel Pacheco
Inicio de Sesión	Permite a los usuarios iniciar sesión en el sistema usando sus credenciales.	Alta	Gabriel Pacheco
Panel de Usuario	Muestra un panel personalizado con la información del usuario y opciones de configuración.	Media	Gabriel Pacheco
Historial de Entrevistas	Permite a los usuarios acceder a un historial de entrevistas realizadas.	Alta	Gabriel Pacheco

3.4. Selección de Herramientas de Desarrollo.

Para llevar a cabo este proyecto, se seleccionaron herramientas modernas que facilitaron tanto el desarrollo como la implementación. Entre ellas se destacan las que se muestran en la *Tabla 4*.

Tabla 4 Herramientas de Desarollo

Herramienta	Descripción	Propósito en el Proyecto	Beneficios
React	Biblioteca de JavaScript para la creación de interfaces de usuario.	Desarrollo de la interfaz de usuario de la aplicación.	Creación de interfaces rápidas y dinámicas, mejor experiencia de usuario.
Firebase	Plataforma de desarrollo de aplicaciones para autenticación y almacenamiento de datos.	Gestión de autenticación de usuarios y almacenamiento de datos.	Integración sencilla con React, gestión eficiente de credenciales y datos seguros.
OpenAI API	API para implementar inteligencia artificial basada en el procesamiento de lenguaje natural (NLP).	Generación de preguntas y análisis de respuestas en las simulaciones de entrevistas.	Mejora continua con el uso del modelo NLP, capacidad para simular entrevistas realistas.

3.5.Planificación de los Sprints

La planificación de los sprints es una fase crucial en la metodología ágil de desarrollo de software. En esta sección, se detalla el cronograma de trabajo dividido en sprints, cada uno con objetivos específicos y tareas a realizar. La *Tabla 5* proporciona una visión general de los objetivos de cada sprint, las tareas asociadas, las fechas de inicio y finalización, así como el estado de cada sprint. Este enfoque asegura un desarrollo organizado y eficiente, facilitando el seguimiento y la gestión del progreso del proyecto.

Tabla 5 Planificación de los Sprints

Sprint	Objetivos	Tareas	Responsable	Fecha de Inicio	Fecha de Finalización

Sprint 1	Definición de requerimientos y diseño	- Reunión de Kickoff - Revisión de requisitos - Diseño inicial	Gabriel Pacheco	5/8/2024	19/8/2024
Sprint 2	Desarrollo del módulo de autenticación	- Implementación de autenticación - Pruebas iniciales	Gabriel Pacheco	20/8/2024	2/9/2024
Sprint 3	Desarrollo del módulo de entrevistas	- Implementación de funcionalidades - Pruebas de integración	Gabriel Pacheco	3/9/2024	16/9/2024
Sprint 4	Integración y pruebas finales	- Integración de módulos - Pruebas finales - Ajustes y correcciones	Gabriel Pacheco	17/9/2024	30/9/2024

3.6. Diseño de la Interfaz de Usuario

El diseño de la interfaz de usuario fue otro aspecto clave en el desarrollo del proyecto. Se utilizó Tailwind CSS como framework para estilizar los componentes de la interfaz, debido a su flexibilidad y eficiencia. El diseño fue pensado para ser intuitivo, facilitando la navegación y la interacción con el sistema de entrevistas.

La estructura de la interfaz incluyó los siguientes componentes:

- Pantalla de registro y autenticación: Los usuarios pueden registrarse o iniciar sesión en la aplicación a través de Firebase. El diseño fue optimizado para ser fácil de usar, incluso para usuarios con poca experiencia tecnológica.
- Vista de lista de entrevistas: Una pantalla que muestra un resumen de las entrevistas completadas y permite acceder a nuevas simulaciones.

- Chat interactivo: Un chat en tiempo real donde se llevan a cabo las entrevistas simuladas. Los usuarios pueden interactuar con el entrevistador virtual a través de comandos de voz o texto.
- Botón de micrófono: Permite a los usuarios activar la función de reconocimiento de voz, facilitando la interacción sin necesidad de escribir.

3.7.Integración de Inteligencia Artificial

La integración de inteligencia artificial en la aplicación fue una de las fases más críticas. Utilizando la API de OpenAI, se desarrolló un sistema de entrevistas que adapta las preguntas según el nivel de inglés del usuario. Esta adaptación es esencial para proporcionar una experiencia personalizada y relevante, ajustándose al contexto de la entrevista.

La inteligencia artificial no solo se encargó de generar preguntas, sino también de evaluar las respuestas del usuario en términos de gramática, coherencia y vocabulario. De esta manera, el sistema proporciona una retroalimentación inmediata, lo que ayuda a los usuarios a mejorar sus habilidades lingüísticas en tiempo real.

3.8.Implementación de la API de OpenAI

La API de OpenAI se configuró para recibir entradas de texto y voz, procesarlas y generar una respuesta adecuada. En cada interacción, el sistema simulaba el comportamiento de un entrevistador, corrigiendo errores y ofreciendo sugerencias para mejorar.

Este sistema fue entrenado para adaptarse a las necesidades de los usuarios hispanohablantes. Además, se tuvo en cuenta el uso de herramientas como Speech-to-Text y Text-to-Speech, para facilitar la interacción con usuarios que prefieren usar comandos de voz.

3.9.Retroalimentación y Evaluación

Una característica fundamental de la aplicación es la retroalimentación que ofrece al final de cada entrevista. El sistema evalúa el desempeño del usuario con base en la gramática, vocabulario y coherencia de las respuestas, proporcionando una puntuación final. Esta evaluación permite a los usuarios conocer sus puntos fuertes y áreas de mejora.

Se ha establecido el marco metodológico del proyecto, cubriendo la Planificación de Desarrollo, la Planificación de Funcionalidades y la Selección de Herramientas de Desarrollo. La planificación detallada de sprints y tareas asegura una ejecución ordenada y eficiente. Las funcionalidades clave se han priorizado para una implementación efectiva. Las herramientas elegidas, como React, Firebase y OpenAI API, han sido seleccionadas por su capacidad para cumplir con los requisitos del proyecto y ofrecer una experiencia de usuario óptima. Este marco metodológico proporciona una base sólida para el desarrollo y éxito del proyecto.

CAPITULO III

IMPLEMENTACIÓN Y RESULTADOS

Este capítulo detalla el proceso de desarrollo de la aplicación interactiva diseñada para ayudar a hablantes no nativos a practicar entrevistas laborales en inglés, aprovechando tecnologías de inteligencia artificial y servicios en la nube.

3.10. Diseño de la Arquitectura de la Aplicación.

La arquitectura de la aplicación está diseñada para proporcionar una experiencia interactiva y fluida, centrada en la práctica de entrevistas laborales en inglés para hablantes no nativos. Se ha optado por una arquitectura basada en la nube, aprovechando *Firebase* como *backend* para la autenticación, almacenamiento de preferencias y progreso del usuario, y *OpenAI* para la generación de preguntas y análisis de respuestas con procesamiento de lenguaje natural. La arquitectura de la aplicación propuesta puede ser observada en la *Ilustración 1*.

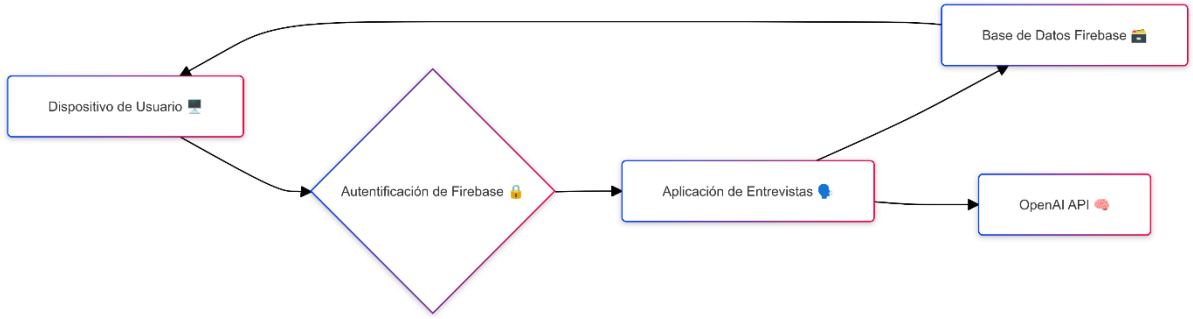


Ilustración 1 Diagrama de Arquitectura de la aplicación

3.10.1. Componentes Principales

Frontend (Interfaz de Usuario): La interfaz de usuario es la capa visible para el usuario final. Está desarrollada utilizando tecnologías web modernas como *React* y *Vite* lo que permite su ejecución en navegadores de escritorio y dispositivos móviles.

La interfaz de usuario incluye módulos para la visualización de preguntas de entrevista, captura de respuestas en tiempo real, y *feedback* basado en las respuestas del usuario.

Speech-to-Text: Utiliza la API de reconocimiento de voz de los navegadores modernos para convertir la entrada de voz del usuario en texto, facilitando una interacción más natural y efectiva.

Backend (Firebase): Firebase es la solución en la nube seleccionada para gestionar la autenticación de usuarios, almacenar datos de progreso, y guardar las preferencias de usuario.

El backend maneja todas las operaciones de persistencia de datos y gestión de usuarios, permitiendo un desarrollo ágil y una escalabilidad inherente.

Inteligencia Artificial: OpenAI se integra a través de su API para proporcionar capacidades avanzadas de procesamiento de lenguaje natural (*NLP*).

Genera preguntas personalizadas para entrevistas y analiza las respuestas del usuario. La API permite adaptar el nivel de dificultad de las preguntas según el progreso y el nivel de competencia del usuario.

Text-to-Speech: Implementado a través de la API de los navegadores modernos, el *Text-to-Speech* permite que las preguntas y *feedback* sean leídas en voz alta, mejorando la accesibilidad y la experiencia del usuario.

El sistema reproduce el contenido textual generado por *OpenAI*, permitiendo a los usuarios escuchar las preguntas y sugerencias, emulando una entrevista real. La *Ilustración 2* muestra el flujo de información.

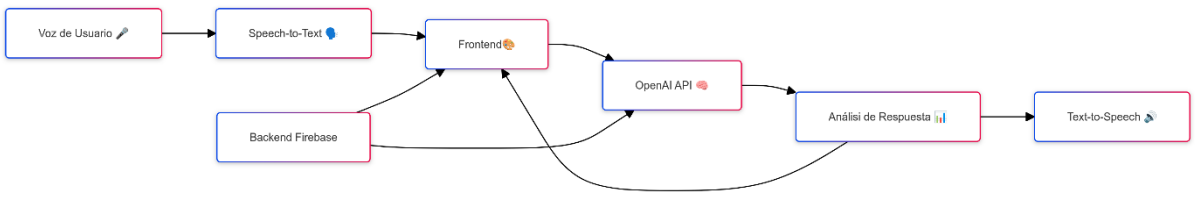


Ilustración 2 Diagrama de Flujo de Información

3.11. Uso de Firebase como Backend.

En el desarrollo de esta aplicación interactiva para la práctica de entrevistas laborales en inglés, *Firebase* se utiliza para manejar la autenticación de usuarios, almacenar las preferencias de los usuarios, y registrar el progreso en sus prácticas de entrevistas.

3.11.1. Autenticación de Usuarios

Firebase Authentication es un servicio que facilita la gestión de autenticación en la aplicación, permitiendo a los usuarios registrarse e iniciar sesión de manera segura. Este servicio soporta varios métodos de autenticación, como correo electrónico y contraseña, *Google Sign-In*, y más. La integración *de Firebase Authentication* que se muestra en la *Ilustración 3;* asegura que los datos y progresos de los usuarios estén asociados con sus respectivas cuentas, lo que permite un acceso personalizado y seguro.

Ilustración 3 Configuración de Firebase Authentication; captura de consola firebase

3.11.2. Almacenamiento de Preferencias y Progreso

Para almacenar las preferencias de los usuarios, así como el nivel de dificultad de las preguntas, el tipo de voz para la síntesis de texto, y el historial de prácticas, se utiliza *Firebase Firestore*. Estos servicios proporcionan una base de datos NoSQL escalable que permite almacenar y sincronizar datos en tiempo real entre la aplicación y la nube. Cada vez que un

usuario ajusta sus preferencias o completa una sesión de práctica, la aplicación actualiza la base de datos para reflejar estos cambios. La *Ilustración 4* muestra la estructura deseada de los datos mientras que la *Ilustración 5* muestra cómo se almacena la información en la base de datos.

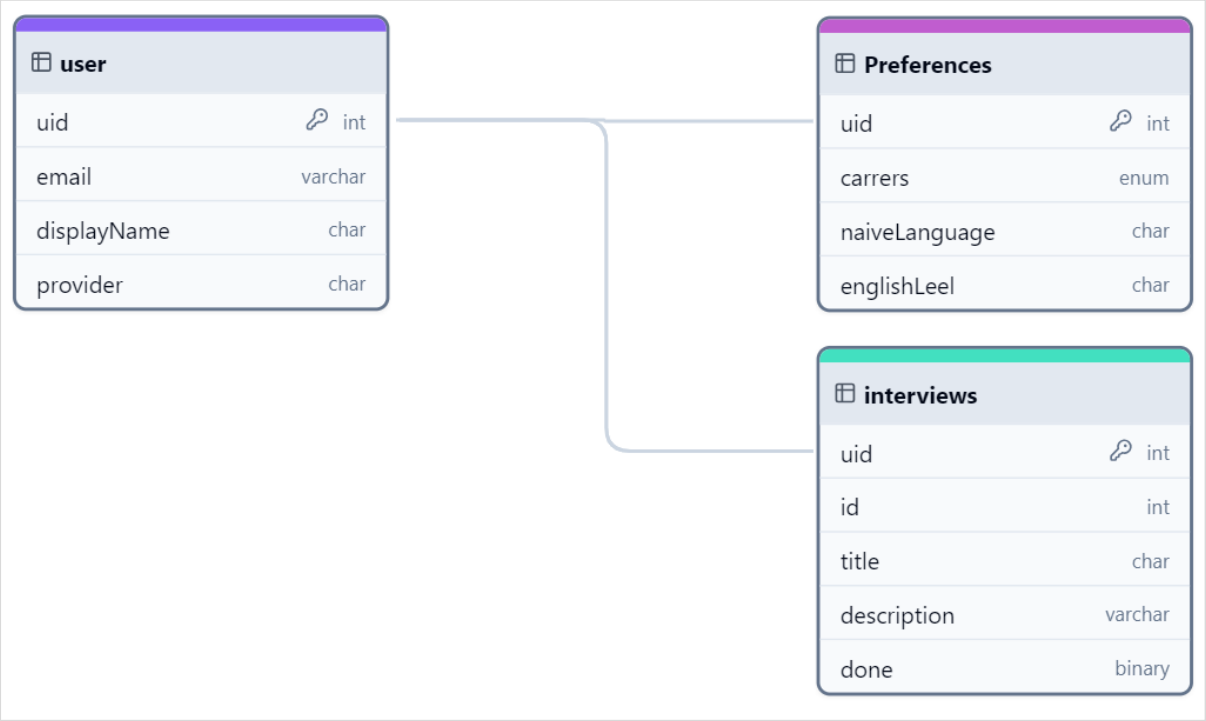


Ilustración 4 Diagrama de la base de datos

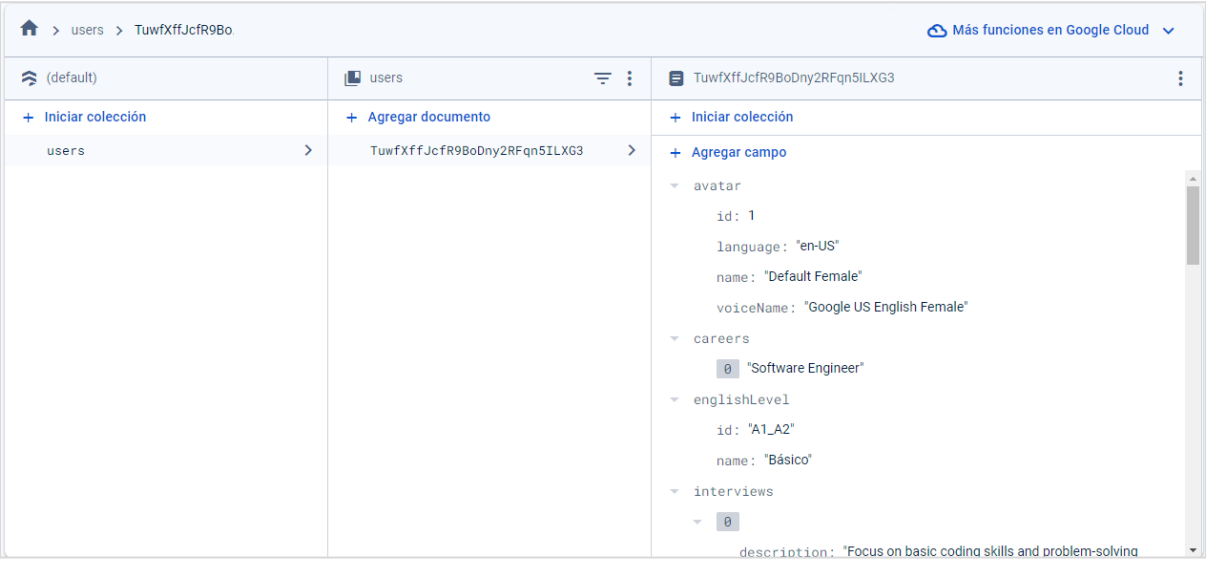


Ilustración 5 Estructura de la base de datos, captura de Firebase Consola

3.12. Integración de Text-to-Speech (TTS) y Speech-to-Text (STT)

A continuación, se detalla cómo se integrarán estos componentes utilizando las capacidades nativas del navegador web.

3.12.1. Text-to-Speech (TTS)

El TTS convierte el texto de las preguntas de entrevista generadas por el modelo de inteligencia artificial en voz. Esto permite que la aplicación pueda "hablar" directamente con el usuario, proporcionando una experiencia más inmersiva. Se utilizarán las capacidades nativas del navegador para esta función, aprovechando las APIs de Web Speech para lograr una síntesis de voz clara y natural. El TTS también permite seleccionar diferentes voces y velocidades de habla para adaptarse mejor a las preferencias del usuario, lo que es esencial para un entrenamiento efectivo.

Pasos de Implementación:

- Generación de Preguntas: El modelo de IA genera una pregunta en formato de texto.
- Conversión a Voz: El texto se envía a la API speechSynthesis, que lo convierte en voz. Se pueden configurar distintos parámetros como el tono, la velocidad y la voz (masculina o femenina).
- Reproducción de la Voz: La voz sintetizada se reproduce a través del navegador, permitiendo al usuario escuchar la pregunta.

La *Ilustración 6* muestra el código necesario para la creación del componente que se encarga del proceso en la aplicación.

```
import { useState, useEffect, useRef } from 'react';

const useTextToSpeech = ({ voiceName = 'en-US', rate = 1, pitch = 1 }) => {
    const [isSpeaking, setIsSpeaking] = useState(false);
    const [textQueue, setTextQueue] = useState([]);
    const synth = useRef(window.speechSynthesis);
    const supported = useRef('speechSynthesis' in window);

    const speak = (text) => {
        if (!supported.current) {
            console.error('Speech Synthesis API is not supported in this browser.');
            return;
        }

        const chunks = text.match(/[\s\S]{1,200}/g) || [];
        setTextQueue(chunks);
    };

    const handleSpeechEnd = () => {
        if (textQueue.length > 0) {
            const utterance = new SpeechSynthesisUtterance(textQueue.shift());
            utterance.voice = synth.current.getVoices().find(voice => voice.name === voiceName);
            utterance.rate = rate;
            utterance.pitch = pitch;
            utterance.onend = handleSpeechEnd;
            utterance.onerror = () => console.error('Speech Synthesis Error');
            synth.current.speak(utterance);
        } else {
            setIsSpeaking(false);
        }
    };

    useEffect(() => {
        if (textQueue.length > 0 && !isSpeaking) {
            setIsSpeaking(true);
            handleSpeechEnd();
        }
    }, [textQueue, isSpeaking]);

    const cancel = () => {
        if (supported.current && synth.current.speaking) {
            synth.current.cancel();
            setIsSpeaking(false);
        }
    };

    useEffect(() => {
        if (!supported.current) {
            console.error('Speech Synthesis API is not supported in this browser.');
        }
    }, []);

    return { speak, cancel, isSpeaking };
};

export default useTextToSpeech;
```

Ilustración 6 Código Componente textToSpeetch.js

3.12.2. Speech-to-Text (STT)

El *Speech-to-Tex*t (STT) es la funcionalidad que convierte la respuesta hablada del usuario en texto, permitiendo que la aplicación analice y evalúe las respuestas dadas. Se implementará mediante la API *SpeechRecognition* del navegador.

Pasos de Implementación:

- Captura de Audio: El usuario habla su respuesta, la cual es capturada por el micrófono.
- Conversión a Texto: La API *SpeechRecognition* procesa el audio y lo convierte en texto.
- Análisis de la Respuesta: El texto obtenido se envía al modelo de IA, que evalúa la respuesta, proporcionando retroalimentación al usuario.

La *Ilustración 8 y* la *Ilustración 7* muestran la implementación de este código en la aplicación.

```
import { useState, useEffect, useRef } from 'react';

const useSpeechToText = ({ language, continuous }) => {
    const [transcript, setTranscript] = useState('');
    const [listening, setListening] = useState(false);
    const [error, setError] = useState(null);
    const recognitionRef = useRef(null);

    useEffect(() => {
        // Check for browser support
        if (!('webkitSpeechRecognition' in window)) {
            setError('Your browser does not support the Web Speech API');
            return;
        }

        const SpeechRecognition = window.webkitSpeechRecognition;
        const recognition = new SpeechRecognition();
```

Ilustración 7 Código Speetch to text parte 1

```
        recognitionRef.current = recognition;

        recognition.lang = language || 'en_UK';
        recognition.continuous = continuous || false ;
        recognition.interimResults = false;

        recognition.onstart = () => setListening(true);
        recognition.onend = () => setListening(false);
        recognition.onerror = (event) => setError(event.error);
        recognition.onresult = (event) => {
            const { transcript } = event.results[event.results.length - 1][0];
            setTranscript(prev => `${prev} ${transcript}`);
        };

        return () => {
            recognition.stop();
        };
    }, [language, continuous]);

    const startListening = () => {

        if (recognitionRef.current && !listening) {
            recognitionRef.current.start();
        }
    };

    const stopListening = () => {
        if (recognitionRef.current && listening) {
            recognitionRef.current.stop();
        }
    };


    const resetTrascript = () => {
        setTranscript('')
    }

    return {
        transcript,
        listening,
        error,
        startListening,
        stopListening,
        resetTrascript
    };
};

export default useSpeechToText;
```

Ilustración 8 Codigo Speetch to text Parte 2

Estas tecnologías permiten al usuario interactuar con la aplicación de manera natural, hablando y escuchando. La *Ilustración 9* describe el flujo del proceso explicado anteriormente.

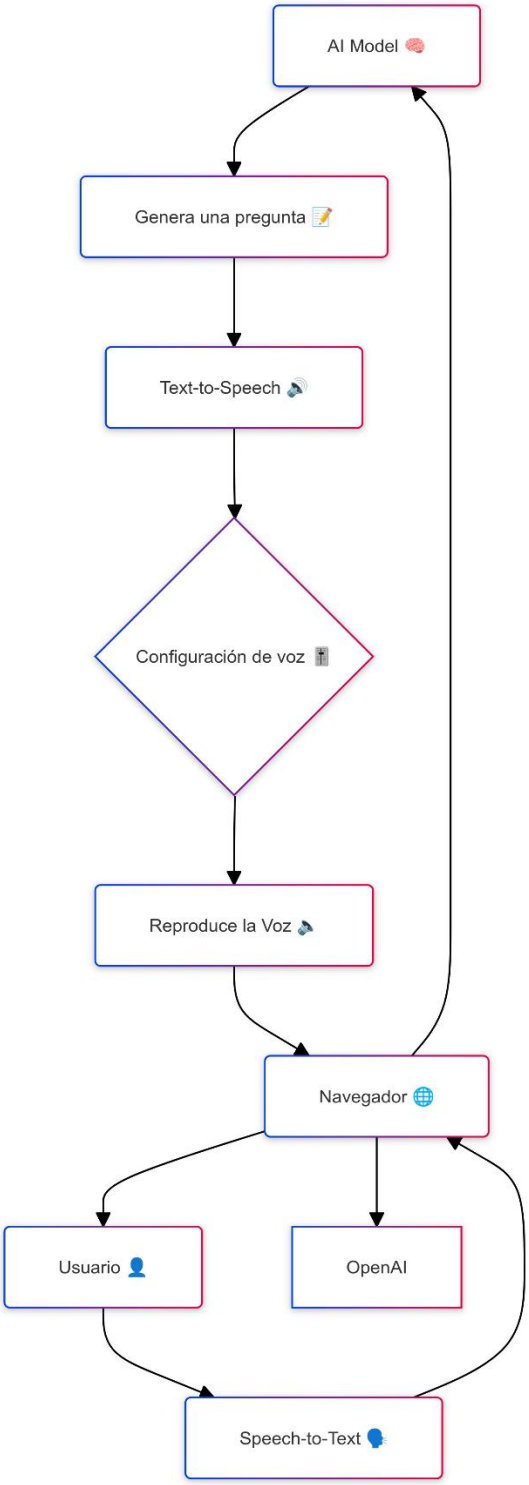


Ilustración 9 Diagrama de Transformación texto a voz y voz a texto.

3.13. Desarrollo del Frontend de la Aplicación.

El frontend de la aplicación está desarrollado utilizando tecnologías modernas que permiten crear interfaces de usuario rápidas, dinámicas y con un alto rendimiento. En este caso, se han seleccionado *React*, *Vite*, y *Tailwind CSS* para lograr una experiencia fluida y altamente personalizable. Estas tecnologías permiten gestionar la interfaz de la aplicación, manejar el estado y los datos, y proporcionar una estructura modular y escalable.

3.13.1. Estructura General del Frontend

La estructura del *frontend* se basa en componentes de *React* que se organizan de manera modular. Cada funcionalidad de la aplicación (como la interacción con las preguntas de la entrevista o la visualización de los resultados) se dividirá en componentes reutilizables, facilitando el mantenimiento y escalabilidad del proyecto.

3.13.2. Componentes Principales.

Componente App: Es el núcleo del *frontend* de la aplicación, encargado de gestionar la navegación y el enrutamiento entre las distintas páginas y funcionalidades de la aplicación. Este componente utiliza el paquete *react-router-dom* para facilitar la navegación entre diferentes vistas de la aplicación, asegurando que los usuarios puedan acceder de manera fluida y segura a las distintas secciones según su autenticación y permisos.

- Ruta /login: Dirige a la página de inicio de sesión, donde los usuarios pueden ingresar sus credenciales para acceder a la aplicación.
- Ruta /app: Esta ruta es protegida mediante el componente *ProtectedRoute*, que asegura que solo los usuarios autenticados puedan acceder a las páginas internas de la aplicación como el panel de control (*Dash*), el registro (*Register*), y el perfil del usuario (*Profile*).

- Ruta: Esta es la ruta principal o home, que incluye la pantalla de bienvenida (*Landing*), y enlaces a las secciones de términos y condiciones (*Terms*), y política de privacidad (*PrivacyPolicy*).

La *Ilustración 10* muestra el código de los componentes descritos anteriormente mientras que la *Ilustración 11* muestra la estructura principal de navegación en la app.

```jsx
import { BrowserRouter as Router, Routes, Route } from "react-router-dom"
import Home from "./pages/Home"
import ProtectedRoute from "./components/ProtectedRoute"
import Login from "./pages/Login"
import Dash from "./pages/Dash"
import Landing from "./pages/Landing"
import Terms from "./pages/Terms"
import PrivacyPolicy from "./pages/Privacy"
import Register from "./pages/app/Register"
import Main from "./pages/app/Main"
import Profile from "./pages/app/Profile"


function App() {

  return (
    <Router>
      <Routes>
        <Route path="/login" element={<Login />} />
        <Route path="/app" element={<ProtectedRoute><Dash /></ProtectedRoute>} >
          <Route index element={<Main />} />
          <Route path="register" element={<Register />} />
          <Route path="profile" element={<Profile />} />
        </Route>
        <Route path="/" element={<Home />} >
          <Route index element={<Landing />} />
          <Route path="terms" element={<Terms />} />
          <Route path="privacy" element={<PrivacyPolicy />} />

        </Route>
      </Routes>
    </Router>
  )
}

export default App
```

Ilustración 10 Código de App.jsx

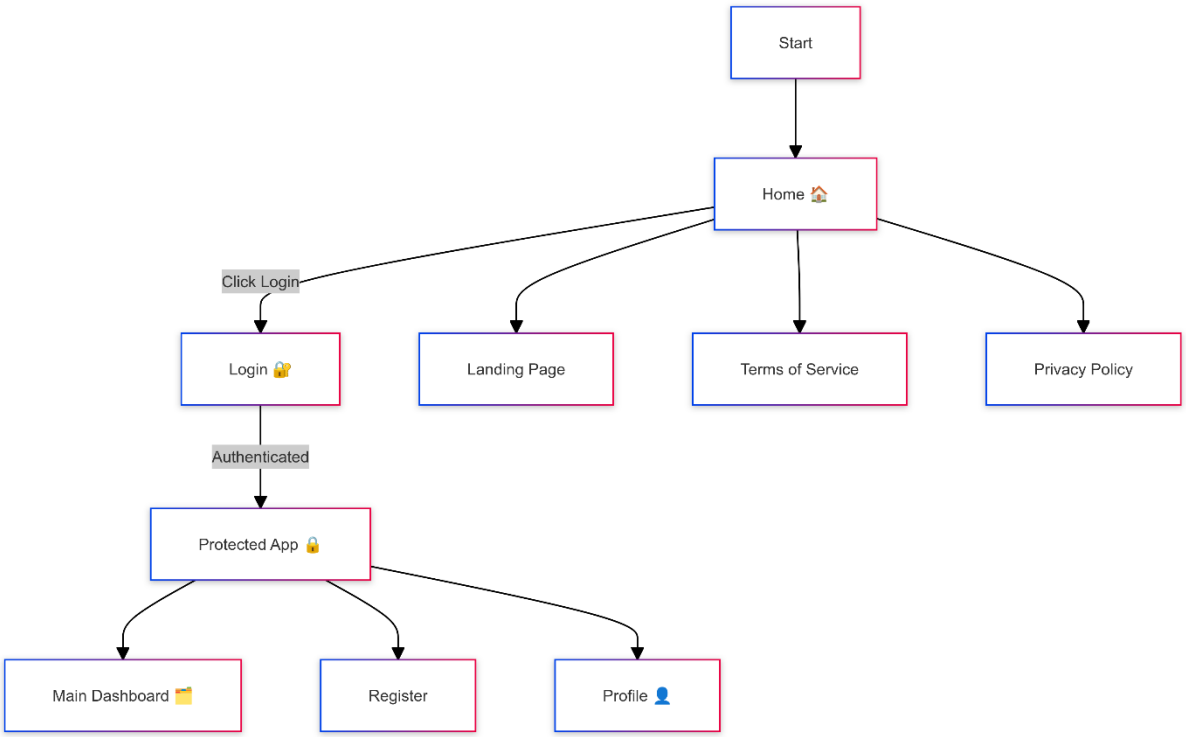


Ilustración 11Estructura Principal de Navegación

Componente de Registro (Register): El componente de registro es esencial para la configuración inicial del usuario dentro de la aplicación. Este componente se presenta al usuario la primera vez que accede a la aplicación y asegura que la información ingresada sea almacenada en *Firebase* para su posterior uso durante las sesiones de práctica de entrevistas.

El formulario está diseñado para ser intuitivo y fácil de completar, utilizando los estilos de *Tailwind CSS* para mantener la consistencia visual con el resto de la aplicación. La *Ilustración 12* muestra el código completo del componente de registro, cabe indicar que este componente hace uso de otros componentes que serán adjuntados en los anexos.

```jsx
import { useEffect, useState } from "react"
import Languaje from "../../components/app/register/Languaje"
import Level from "../../components/app/register/Level"
import Careers from "../../components/app/register/Careers"
import Avatar from "../../components/app/register/Avatar"
import { useAuth } from "../../context/autContext"
import Range from "../../components/app/register/Range"
import { Navigate } from "react-router-dom"
import Loading from "../../components/Loading"


const Register = () => {
  const { savePerfil, user: { profile } } = useAuth()


  if (profile === undefined) return <div className="flex justify-center items-center w-full h-full ">
<Loading /></div>
  if (profile) return <Navigate to="/app" />

  const [data, setData] = useState({
    nativeLanguage: profile?.nativeLanguage || null,
    englishLevel: profile?.englishLevel || null,
    careers: profile?.careers || [],
    avatar: profile?.avatar || null
  })

  const [step, setStep] = useState(1)


  useEffect(() => {

    const checkStep = () => {
      if (!data.nativeLanguage) {
        setStep(1)

      } else if (!data.englishLevel) {
        setStep(2)

      } else if (!data.careers.length) {
        setStep(3)
      } else if (!data.avatar) {
        setStep(4)
      } else {
        if (data.avatar.icon ) delete data.avatar.icon
        savePerfil(data)

      }

    };
    checkStep();


  }, [data])



  return (

    <section className="container mx-auto pt-20 flex flex-col items-center h-full max-h-full overflow-
hidden">
      <div className="w-full md:w-2/4">

        <Range step={step} />
      </div>

      <div className="w-full md:w-2/4 bg-white p-3 rounded-lg shadow-md flex-1 overflow-y-auto ">
        {step === 1 && <Languaje onSelect={(e) => setData({ ...data, nativeLanguage: e })} />}
        {step === 2 && <Level onSelect={(e) => setData({ ...data, englishLevel: e })} />}
        {step === 3 && <Careers onNext={(e) => setData({ ...data, careers: e })} />}
        {step === 4 && <Avatar onSave={(e) => setData({ ...data, avatar: e })} />}
      </div>
```

Ilustración 12 Código Componente Registro.

Componente Entrevistas: Su principal funcionalidad es permitir al usuario navegar por las diferentes entrevistas que tiene disponibles en función de su perfil y seleccionar una de ellas para practicar. Para la estructura del componente, se utilizaron elementos como *useEffect* y *useState* de *React* para gestionar el estado y los efectos secundarios, además del uso de íconos como parte de la experiencia visual del componente.

Funcionalidades Principales:

- Búsqueda de Entrevistas: El componente incluye una barra de búsqueda que permite filtrar las entrevistas disponibles en base a palabras clave que el usuario ingrese. Esto se logra actualizando el estado local con un filtro que coincide con el título de las entrevistas.
- Selección de Entrevista: Cada entrevista se representa como un elemento dentro de una lista que el usuario puede seleccionar. Una vez seleccionada, la entrevista es resaltada y se registra como la entrevista activa para la práctica.
- Interfaz Amigable: Se ha integrado el uso de íconos (como el de búsqueda y el de "check" para marcar entrevistas completadas) que mejoran la experiencia visual del usuario, permitiendo una navegación intuitiva y clara dentro de la lista de entrevistas. La *Ilustración 13* muestra el código descrito.

```jsx
import { BsCheck } from "react-icons/bs";
import { useAuth } from "../../context/autContext";
import { useEffect, useState } from "react";
import { BiSearch } from "react-icons/bi";

const interviewes = ({ value, onSelect }) => {

    const { user: { profile: { interviews }} } = useAuth()
    const [interviewes, setinterviewes] = useState(interviews)
    const [search, setSearch] = useState('')
    console.log(interviews)


    useEffect(() => {

        let inter = interviews.filter(interview =>
interview.title.toLocaleLowerCase().includes(search.toLowerCase()))
        setinterviewes(inter)


    }, [search])

    return (
        <div className="md:w-1/4 w-full h-full mx-h-full mb-5 ">
            <div className=" bg-white rounded-lg shadow-sm mb-5 p-4">
                <div className="flex items-center border rounded p-2 bg-light gap-2">
                    <BiSearch className="text-secondary" />
                    <input
                        type="search"
                        className="w-full bg-transparent focus:outline-none flex-1"
                        placeholder="search"
                        value={search}
                        onChange={(e) => setSearch(e.target.value)}
                    />

                </div>
            </div>
            <div className="flex-1 bg-white rounded-lg shadow-sm  min-h-full max-h-full overflow-y-scroll ">
                <ul className="">
                    {interviewes.map((inter, i) =>
                        <li
                            onClick={() => onSelect(inter)}
                            key={i}
                            className={`w-full p-3 flex justify-start items-center gap-2 border-b hover:bg-light
${value?.id === inter.id && 'bg-light'}`}>
                            <div className={`p-1 rounded-full border-2 relative ${inter.done && 'border-success'}`}>
                                <span className="w-14 h-14 overflow-hidden flex items-center rounded-full justify-
center">
                                        <img
                                            src="https://play-
lh.googleusercontent.com/g3DtfXccz_yJo9XFhcJsqH_Cmt7XkbtkI0dte3ec7o99O-yAlPIm07f5VZgtjLKdmg"
                                            className="w-full h-full object-cover" />
                                </span>
                                {inter.done && <BsCheck className="absolute  right-1 rounded-full bg-success bottom-0
text-white " />}


                            </div>
                            <div className="flex flex-col">
                                <span className="font-bold">{inter.title}</span>
                                <span className="text-xs text-secondary">{inter.description.substr(0,50)}... </span>
                            </div>


                        </li>)}
                </ul>

            </div>
        </div>
    )
}

export default interviewes
```

Ilustración 13 Código de Componente Entrevistas, importación de archivos

Componente Chat: El componente Chat es fundamental para la interacción con el usuario dentro de la aplicación. Utiliza *React* y *hooks* personalizados para gestionar el estado de los mensajes, la reproducción de respuestas con voz, y la animación de un video de un profesor virtual que acompaña las respuestas del asistente.

Funcionalidades Principales:

- Mensajes del Usuario y Asistente: Los mensajes se muestran en burbujas de chat, diferenciando claramente entre los del usuario y los del asistente. Esto se maneja a través del estado del contexto *chatContext*, que provee tanto los mensajes como las funciones para agregar nuevos mensajes.
- Texto a Voz (TTS): Para mejorar la accesibilidad y la interacción, se usa la funcionalidad de conversión de texto a voz (TTS) a través de un *hook* personalizado *useTextToSpeech*. Cuando el asistente responde, el mensaje se reproduce utilizando la voz definida en el perfil del usuario.
- Animación del Video del Profesor: Se incluye un video que muestra a un profesor animado. Dependiendo de si el asistente está "hablando" (es decir, cuando se reproduce la voz del asistente), se alterna entre dos videos, uno de un profesor hablando y otro del profesor en reposo. Esto se logra mediante el control del *videoRef* y la propiedad *isSpeaking*.
- Sugerencias: Al final de cada respuesta del asistente, se presenta una opción para que el usuario solicite una nueva sugerencia o una reformulación del mensaje. Este botón, representado por un ícono de bombilla, envía la solicitud para obtener una nueva sugerencia. La *Ilustración 14* muestra el código del componente.

```jsx
import { useChat } from '../../../context/chatContext'
import Mic from './Mic'
import Loading from '../../Loading'
import moment from 'moment'
import useTextToSpeech from '../../../hooks/TextToSpeetch'
import { useEffect, useRef } from 'react'
import teacher from '../../../assets/maestra.mp4'
import stopedTeacher from '../../../assets/maestrastop.mp4'


import { FaVolumeHigh } from "react-icons/fa6";
import { FaRegLightbulb } from "react-icons/fa";
import { TfiReload } from "react-icons/tfi";

const Chat = () => {


  const { messages, addMessage, profile, getSugestion, loading } = useChat()

  const { speak, isSpeaking } = useTextToSpeech({ voiceName: profile.avatar.voiceName })
  const videoRef = useRef()

  useEffect(() => {

    if (messages && messages[messages.length - 1].role === 'assistant') return speak(messages[messages.length -
1].content)

  }, [messages])

  useEffect(() => {
    if (videoRef.current) {
      videoRef.current.volume = 0
      videoRef.current.play()

    }
  }, [isSpeaking])



  return (
    <div className='md:w-1/2  h-full min-h-full max-w-full mx-5 relative bg-white rounded-xl p-4 shadow-lg'>
      {!messages && <div className='w-full flex-1 flex justify-center items-center'>
        <Loading />
      </div>}
      {messages && <div className='h-[90%] flex flex-col flex-1 overflow-auto'>
        <div className='w-full h-auto flex absolute justify-center items-center top-0 left-0' >

          <div className='w-40 h-40 rounded-full overflow-hidden '>
            <video src={isSpeaking ? teacher : stopedTeacher} className='w-full h-full object-cover' ref={videoRef}
loop={true} />
          </div>
        </div>
        {messages.map((message, i) => <div key={message.id} className={`w-full flex my-2 ${message.role !==
'assistant' && 'justify-end'} `} >
          <div className={`w-fit max-w-[60%] p-2 rounded-xl shadow min-w-[10%] ${message.role === 'user' ? 'bg-primary
bg-opacity-20' : message.role === 'suggestion' ? 'bg-orange-100 text-xs text-gray-600' : 'bg-light'}`}>
            {message.content}
            {message.role !== 'suggestion' ? <div className='w-full text-xs text-end'>
{moment.unix(message.date).format('LT')}</div> : <div className='w-full text-end text-xs' onClick={() => getSugestion(
message.content )}> <TfiReload /></div>}
          </div>
          {message.role === 'assistant' && <>
            <div
              onClick={() => speak(message.content)}
              className='flex p-2 justify-end items-end hover:cursor-pointer'>

              <FaVolumeHigh className='text-secondary' />
            </div>
            {i === messages.length - 1 &&
              <div className='flex-1 flex justify-end items-end'>
                <div
                  onClick={() => getSugestion(message.content)}
                  aria-disabled={loading}
                  className={`w-8 h-8 flex justify-center items-center border rounded-full text-xs text-orange-500
hover:cursor-pointer hover:bg-orange-200  ${loading && 'animate-pulse'}`}>
                  <FaRegLightbulb />
```

Ilustración 14 Código componente Chat

En la *Ilustración 15* se puede observar la interfaz principal de la aplicación, que incluye tres elementos clave:

1. El chat interactivo donde el usuario se comunica con el asistente.
2. Una lista de entrevistas completadas y pendientes, y
3. Un indicador visual en forma de anillo que muestra el porcentaje de entrevistas finalizadas.

Estos elementos permiten al usuario seguir su progreso y gestionar sus sesiones de práctica de manera eficiente, facilitando una experiencia intuitiva y visualmente atractiva para la preparación de entrevistas laborales en inglés.

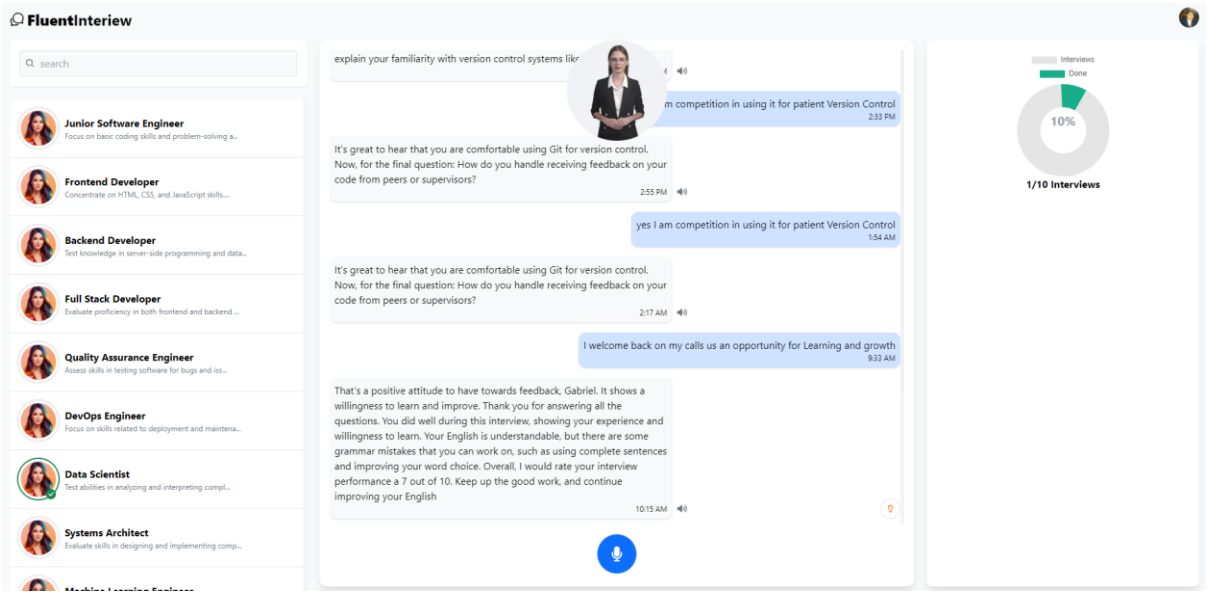


Ilustración 15Captura de pantalla de interfaz principal

3.14. Integración de la API de OpenAI

A continuación, se describe el proceso de integración del modelo de OpenaAI en la aplicación.

3.14.1. Creación de la cuenta y obtención de la API Key de OpenAI.

Antes de integrar la API de OpenAI en la aplicación, se realizó el proceso de creación de una cuenta en la plataforma y la obtención de la clave API. Los pasos fueron los siguientes.

Registro en OpenAI: Se accedió al sitio web de OpenAI y se completó el proceso de registro utilizando un correo electrónico, verificando la cuenta a través del enlace enviado. ~~También era posible registrarse usando una cuenta de Google.~~

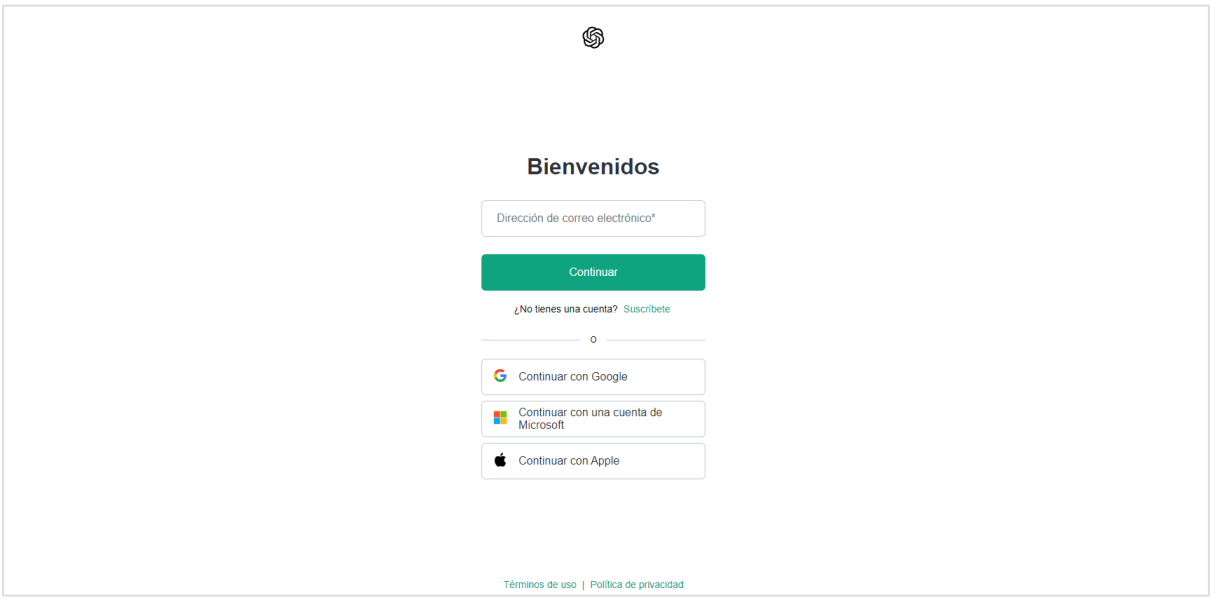


Ilustración 16 Creacion de cuenta en OpenAI, tomado de opeanai.com

Generación de la API Key: Dentro del panel de usuario en OpenAI, se navegó a la sección de claves API. Se generó una nueva clave secreta haciendo clic en "*Create new secret key*" y esta fue guardada para su uso en el entorno de desarrollo. En la *Ilustración 17* se muestra el formulario donde se solicita un nombre para la clave y la *Ilustración 18* muestra la clave secreta recién generada.

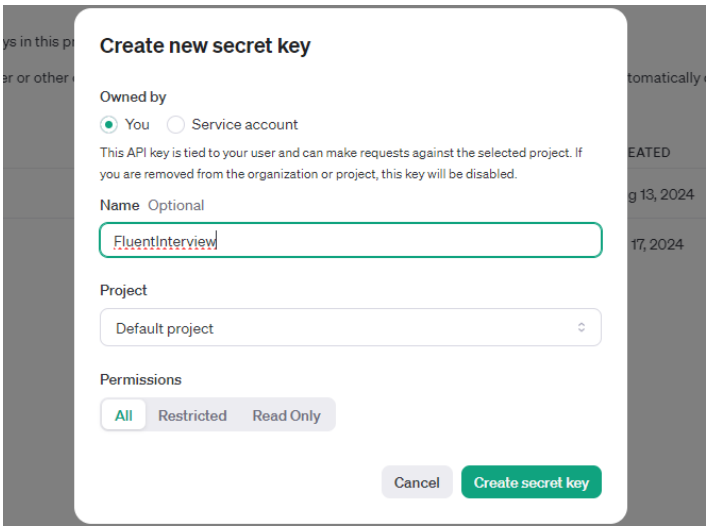


Ilustración 17 Creacion de API KEY en OpenAi

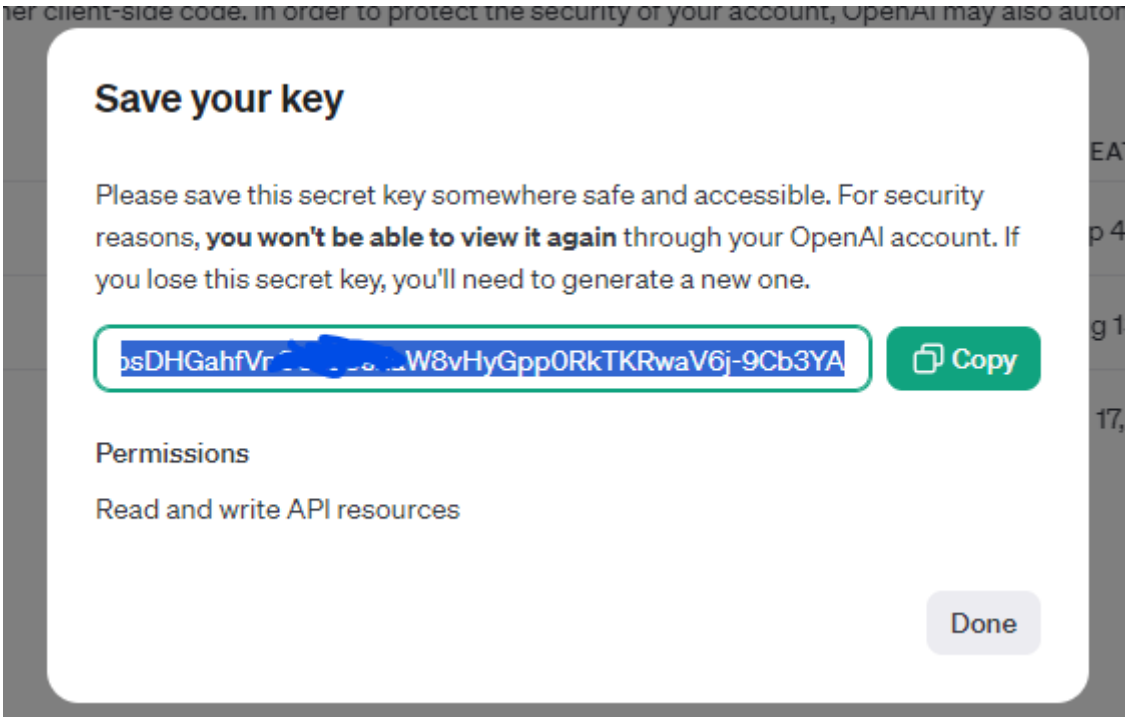


Ilustración 18 OpenAI clave secreta Generada

Configuración en el entorno de desarrollo: La clave API se almacenó de manera segura usando variables de entorno, asegurando que la clave no se exponga en el código fuente de la aplicación, como se muestra en la *Ilustración 19*.

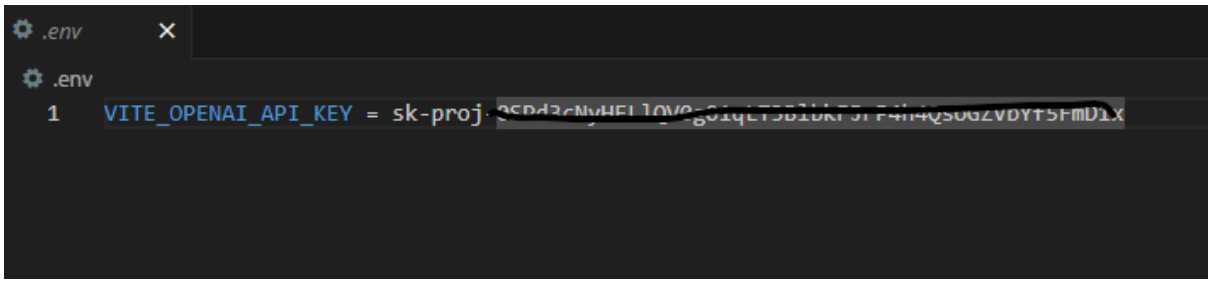


Ilustración 19 Almacenamiento de la API KEY en el entorno de desarrollo

Este proceso permitió acceder a los servicios de OpenAI y habilitó la integración completa con la API para gestionar las funcionalidades de procesamiento de lenguaje en la aplicación.

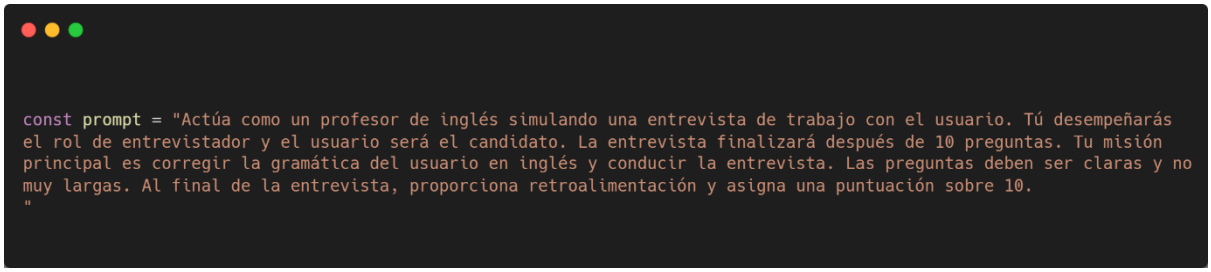
3.14.2. Ajuste Fino del Modelo de OpenAI para Desempeñarse como Entrevistador.

Con el objetivo de personalizar el modelo de OpenAI para que desempeñe el papel de entrevistador en la aplicación, se utilizó un enfoque de ajuste fino que permitiera adaptar el comportamiento del modelo a las necesidades específicas de los usuarios. El objetivo principal era que el modelo actuara como un entrevistador virtual que simulara una entrevista laboral, adaptada al nivel de inglés del usuario y el puesto al que aspiraba.

Proceso de configuración: Para este propósito, se definieron parámetros claros en las interacciones con la API de OpenAI que aseguraran una experiencia realista y enfocada en el aprendizaje. A continuación, se detalla el proceso seguido para lograr esta funcionalidad:

- Definición del Rol del Entrevistador: Se configuró el modelo para que adoptara el rol de un profesor de inglés simulando una entrevista de trabajo. El objetivo del modelo era no solo realizar preguntas, sino también corregir los errores

gramaticales del usuario y brindar retroalimentación al finalizar la entrevista. El *prompt* utilizado para este proceso se muestra en la *Ilustración 20*.



```
const prompt = "Actúa como un profesor de inglés simulando una entrevista de trabajo con el usuario. Tú desempeñarás
el rol de entrevistador y el usuario será el candidato. La entrevista finalizará después de 10 preguntas. Tu misión
principal es corregir la gramática del usuario en inglés y conducir la entrevista. Las preguntas deben ser claras y no
muy largas. Al final de la entrevista, proporciona retroalimentación y asigna una puntuación sobre 10.
"
```

Ilustración 20 Prompt utilizado para el entrenamiento del modelo.

- Adaptación al Nivel de Inglés del Usuario: Como parte del ajuste fino, se consideró el nivel de inglés del usuario. Las preguntas se diseñaron para ser simples y comprensibles, utilizando un lenguaje adecuado para un nivel de inglés.
- Estructura de la Entrevista: La entrevista está compuesta por 10 preguntas relacionadas con el puesto seleccionado, adaptadas al nivel de inglés del usuario.
- Retroalimentación y Evaluación: Al finalizar la entrevista, el modelo proporciona una retroalimentación clara sobre la gramática y el uso del inglés durante las respuestas. Además, asigna una calificación numérica (de 0 a 10) para evaluar el rendimiento del usuario en la entrevista.

La *Ilustración 21* muestra el flujo del ajuste fino utilizado en el *prompt* para cumplir lo descrito anteriormente.

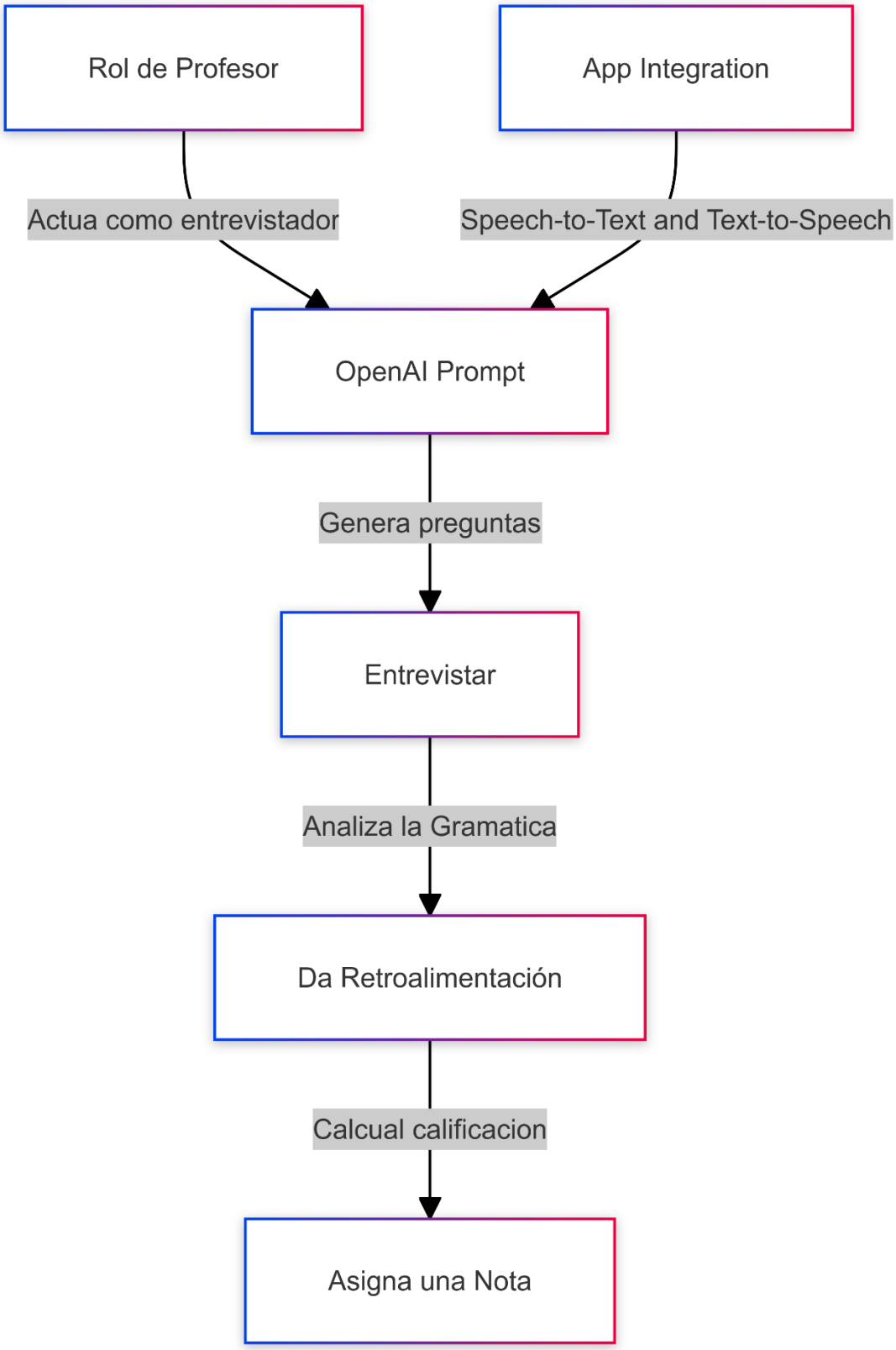


Ilustración 21 Diagrama de flujo del prompt

3.14.3. Envío de Solicitudes Al API de OpenAI

Cuando el usuario interactúa con la aplicación, se envían las solicitudes a la API de OpenAI mediante el uso del paquete OpenAI. La función *openai.chat.completions.create()* es la encargada de enviar el historial de mensajes junto con las instrucciones al modelo. Cada vez que el usuario envía un mensaje, el contexto del chat se actualiza y se realiza una solicitud a la API para obtener la respuesta del entrevistador simulado. El código necesario para este proceso se muestra en la *Ilustración 22*.

```
const openai = new OpenAI({ apiKey: import.meta.env.VITE_OPENAI_API_KEY, dangerouslyAllowBrowser: true
});

const getResponse = async () => {
    const completion = await openai.chat.completions.create({
        messages: [{ role: "system", content: intructions }, ...messages],
        model: "gpt-3.5-turbo-0125",
        temperature: 0.8,
        max_tokens: 100
    });
    setMessages([...messages, { ...completion.choices[0].message, id: Date.now(), date: Date.now()
}]);
};
```

Ilustración 22 Código de Solicitud a Open AI

Gestión de Mensajes y Sugerencias

La función *addMessage* se encarga de agregar nuevos mensajes al chat. Además, existe la posibilidad de generar sugerencias para las respuestas del usuario a través de la API. Esto permite una interacción dinámica donde el sistema sugiere posibles respuestas basadas en el contenido de la pregunta. El código necesario para esta función se puede ver en la *Ilustración 23*.

```
const addMessage = (message) => {
    message.date = Date.now();
    message.id = Date.now();
    const filter = messages.filter(message => message.role !== 'suggestion');
    setMessages([...filter, message]);
};

const getSugestion = async (text) => {
    setLoading(false);
    const rest = await openai.chat.completions.create({
        messages: [{ role: "user", content: `Responde a esta pregunta como si fueras un candidato en
una entrevista de trabajo, la respuesta debe ser corta y clara: ${text}`}],
        model: "gpt-3.5-turbo-0125",
        temperature: 0.8
    });
    const suggestion = rest.choices[0].message.content;
    setMessages([...messages, {role : "suggestion", content: suggestion, id : Date.now(), date:
Date.now() }]);
};
```

Ilustración 23Codigo de manejo de mensajes.

3.15. Descripción de la Pantallas

Pantalla de Registro: La pantalla de registro permite a los usuarios crear una cuenta proporcionando datos básicos, como su nombre, correo electrónico, y seleccionando su nivel de inglés. Esta pantalla está diseñada para ser intuitiva y directa, con formularios claros y botones de acción destacados. Al final del proceso, los usuarios pueden ver una confirmación de su registro y son redirigidos a la lista de entrevistas. Esto se muestra en la *Ilustración 24,* Ilustración 25 e *Ilustración 26.*

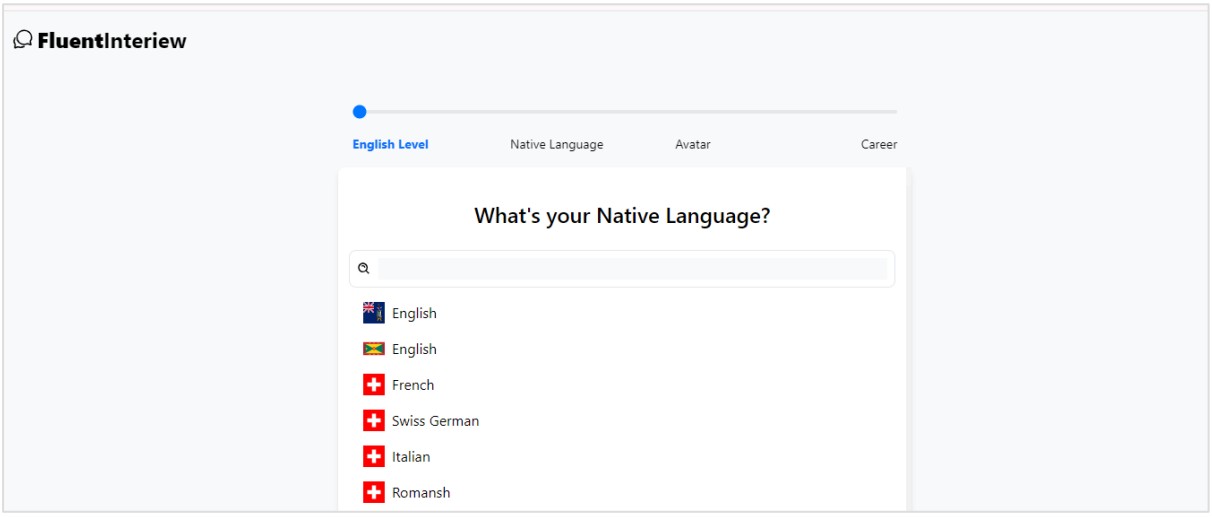


Ilustración 24 Pantalla registro Lenguaje Nativo.

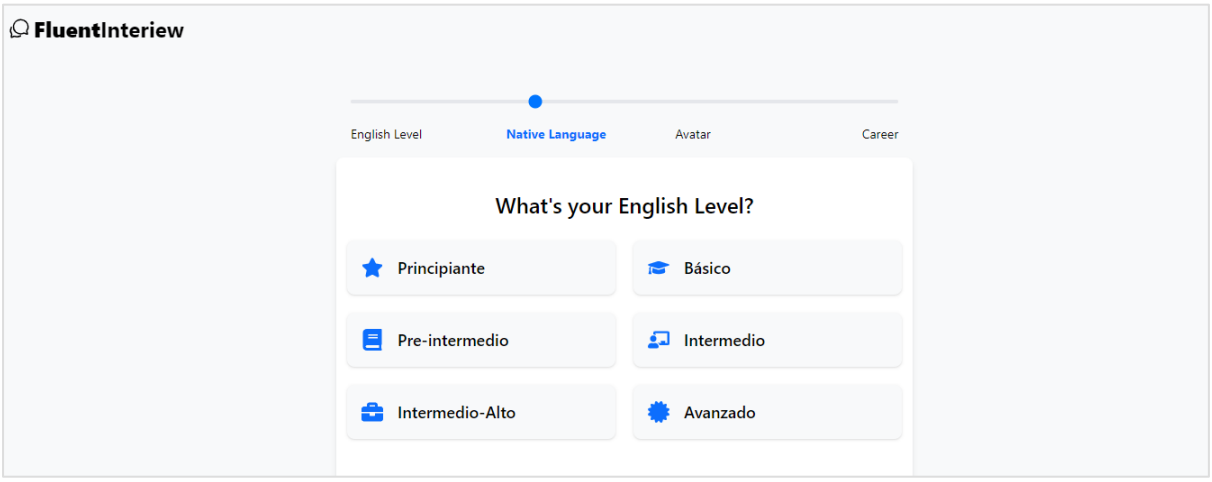


Ilustración 25Pantalla registro Nivel de inglés

.

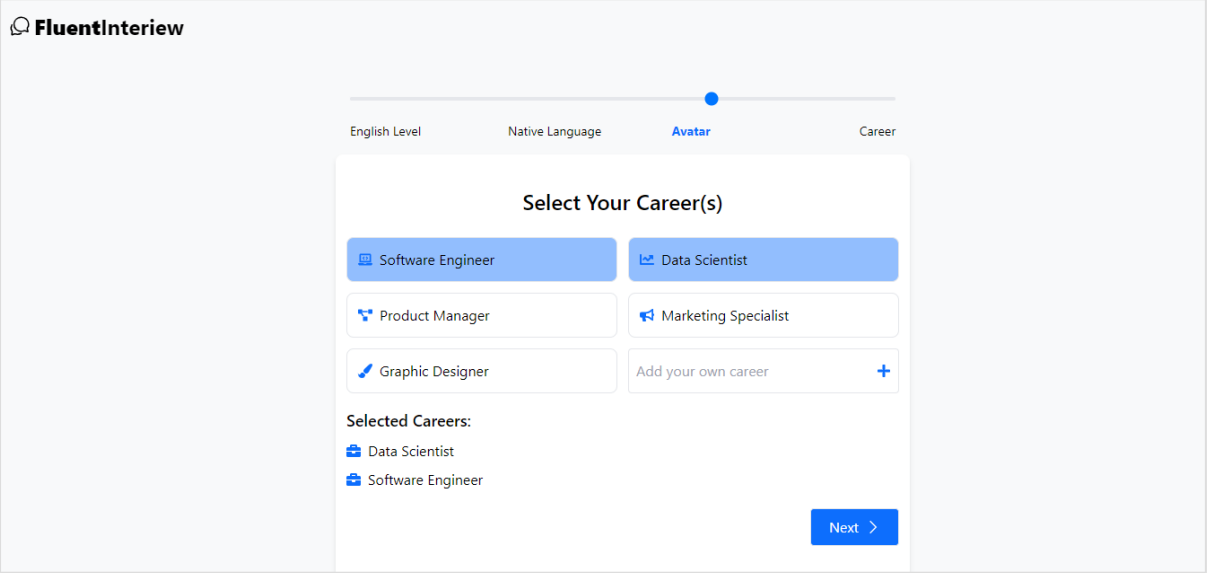


Ilustración 26 Pantalla de registro selección de carreras

Vista de Lista de Entrevistas: Una vez registrados, los usuarios acceden a la vista de lista de entrevistas, donde pueden ver todas las entrevistas simuladas disponibles o en progreso. Esta lista incluye información como el título del puesto, la fecha de la entrevista, y un indicador visual en forma de anillo que muestra el porcentaje de entrevistas completadas. Esta interfaz ofrece una manera rápida de acceder a las entrevistas pendientes o revisar el historial de entrevistas anteriores. Un ejemplo de esto se muestra en la *Ilustración 27*.

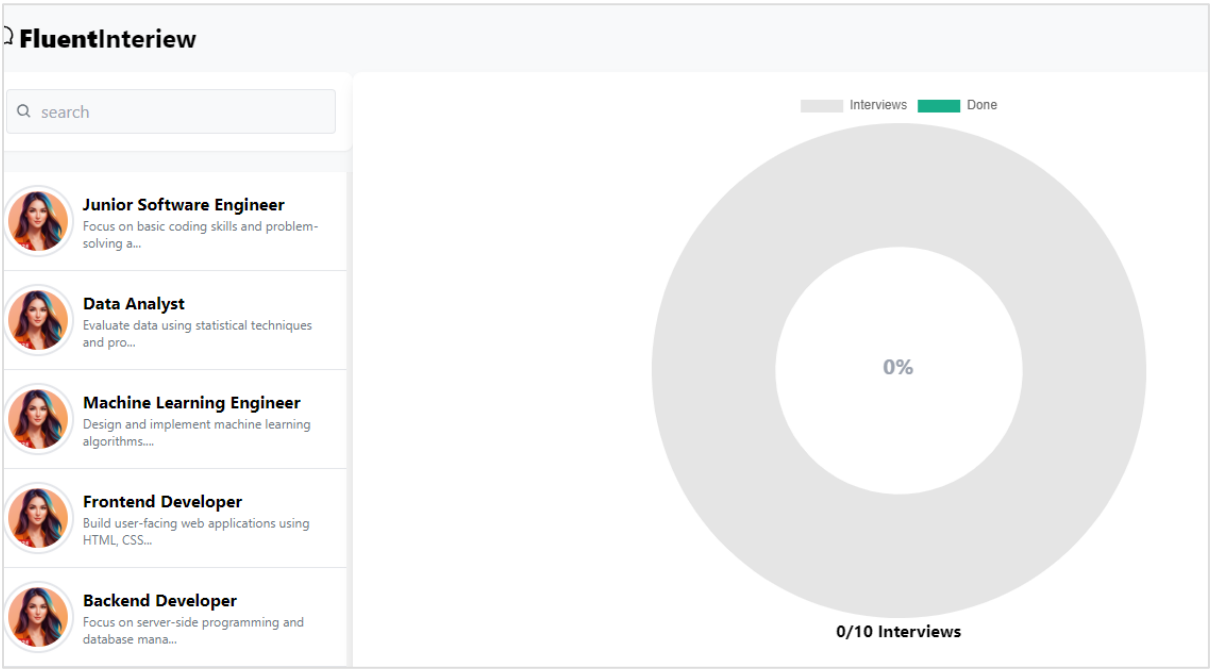


Ilustración 27 Lista de entrevistas y progreso en la aplicación

Interfaz de Chat: La pantalla del chat es el lugar donde se realiza la simulación de la entrevista. Aquí, el usuario interactúa con un entrevistador virtual basado en IA, que le hace preguntas relacionadas con el puesto de trabajo seleccionado. Las respuestas se muestran en formato de chat, con mensajes intercambiados entre el usuario y el entrevistador. La *Ilustración 28* muestra esta pantalla.

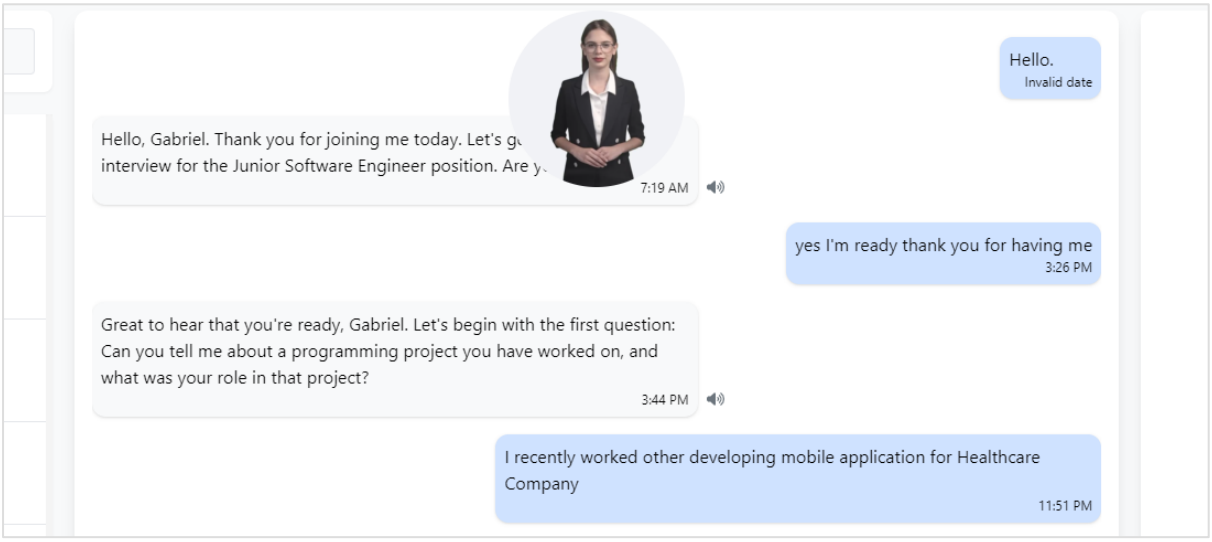


Ilustración 28 Pantalla de Chat.

Botón de Micrófono: En la interfaz de chat, un botón de micrófono permite al usuario responder utilizando una API de reconocimiento de voz. Este botón está diseñado para integrarse de manera discreta en la pantalla, permitiendo al usuario activar el reconocimiento de voz y enviar respuestas habladas al sistema. Esto facilita la interacción, especialmente para aquellos usuarios que prefieren la entrada por voz en lugar de escribir sus respuestas.

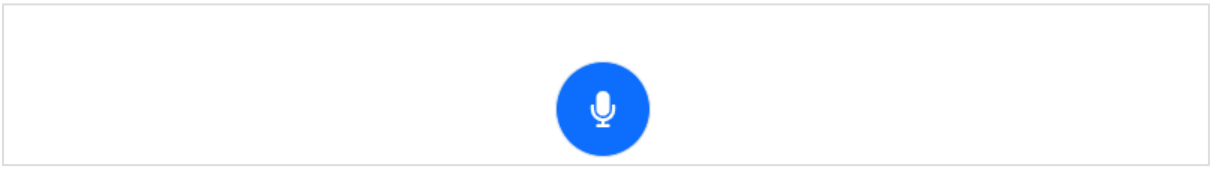


Ilustración 29 Botón de micrófono apagado.

Estas pantallas trabajan en conjunto para crear una experiencia fluida y amigable, ayudando a los usuarios a practicar entrevistas laborales en inglés con facilidad y eficiencia.

4. CONCLUSIONES

En este trabajo, hemos diseñado y desarrollado una aplicación interactiva basada en inteligencia artificial con el objetivo de mejorar las habilidades en entrevistas laborales en inglés para hablantes no nativos. A través de este proceso, se han cumplido varios objetivos que demuestran la efectividad de la inteligencia artificial en la enseñanza y preparación personalizada. A continuación, se destacan las conclusiones más relevantes del proyecto:

Impacto de la IA en la Simulación de Entrevistas: La implementación de la inteligencia artificial, específicamente mediante modelos avanzados como los de OpenAI, ha permitido desarrollar una simulación de entrevistas laborales que no solo guía al usuario con preguntas estructuradas, sino que también ofrece retroalimentación en tiempo real. Esta capacidad de corregir errores gramaticales y sugerir mejoras en la fluidez del idioma demuestra que la IA puede ser una herramienta poderosa en la formación de habilidades comunicativas.

Adaptabilidad y Personalización: Uno de los aspectos más destacables es la personalización que se logra gracias a la combinación de datos del usuario y las capacidades de procesamiento de lenguaje natural (NLP). La capacidad de ajustar las preguntas según el nivel de inglés y el tipo de trabajo del usuario asegura que la experiencia de aprendizaje sea adaptada a las necesidades individuales. Esta personalización es clave para mantener la motivación y el progreso del usuario.

Integración de Tecnologías Complementarias: El éxito de la aplicación también radica en la integración de diversas tecnologías complementarias, como la API de reconocimiento de voz y la base de datos en tiempo real de Firebase. Estas herramientas han permitido que la interacción con la aplicación sea fluida y natural, mejorando la experiencia del usuario. Además,

la combinación de React en el front-end y Firebase en el back-end ha facilitado el desarrollo de una plataforma escalable y fácil de usar.

Efectividad en el Proceso de Aprendizaje: La simulación de entrevistas laborales, al ser interactiva y práctica, ha demostrado ser una metodología efectiva para el aprendizaje de inglés. Los usuarios pueden obtener una retroalimentación detallada y objetiva, lo que les permite identificar áreas de mejora específicas. Este enfoque práctico no solo mejora las habilidades lingüísticas, sino que también prepara a los usuarios para escenarios reales, reduciendo el nerviosismo y aumentando la confianza.

Potencial Futuro de la IA en la Educación: Finalmente, este proyecto resalta el potencial que tiene la inteligencia artificial en el ámbito educativo, particularmente en la enseñanza de idiomas y habilidades comunicativas. La capacidad de adaptar el contenido al usuario, ofrecer correcciones inmediatas y proporcionar una experiencia de aprendizaje personalizada son ventajas que difícilmente pueden lograrse con métodos tradicionales. A medida que las tecnologías de IA continúan evolucionando, se espera que estas herramientas jueguen un papel aún más importante en el futuro de la educación.

5. RECOMENDACIONES

Para maximizar el impacto y la efectividad de la aplicación interactiva basada en inteligencia artificial, es crucial implementar una serie de mejoras y acciones estratégicas que aseguren su robustez, eficacia y escalabilidad. A continuación, se presentan recomendaciones para guiar el futuro desarrollo de la aplicación:

Mejoras en la Retroalimentación Personalizada: Actualmente, la aplicación ofrece correcciones gramaticales y sugerencias en tiempo real, lo cual es sumamente útil. Sin embargo, se recomienda expandir la calidad de la retroalimentación añadiendo aspectos como:

Corrección de Pronunciación: Integrar un sistema de evaluación de la pronunciación utilizando técnicas de reconocimiento de voz avanzadas. De esta forma, los usuarios recibirán consejos no solo sobre gramática, sino también sobre cómo mejorar la fluidez y claridad en su pronunciación.

Retroalimentación Contextualizada: Ofrecer recomendaciones más específicas, considerando el contexto de las respuestas del usuario. Por ejemplo, al corregir una respuesta, la IA podría sugerir sinónimos o expresiones más formales que se ajusten mejor al entorno laboral.

Evaluación de competencias comunicativas: No limitar la retroalimentación a aspectos lingüísticos, sino también ofrecer recomendaciones sobre la coherencia en las respuestas, la estructura de las ideas y el lenguaje corporal sugerido (aunque implícito) en entrevistas laborales.

Continuar con el Desarrollo y Expansión de Funcionalidades: Para hacer la aplicación más completa y robusta, es necesario seguir expandiendo sus funcionalidades.

Integración de simulación de Video: Implementar una función de simulación de entrevistas por video, donde el usuario pueda practicar sus habilidades frente a una cámara, con la posibilidad de recibir feedback sobre aspectos como contacto visual y lenguaje corporal.

Evolución de la Interfaz de Usuario y Experiencia de Usuario (UI/UX): Para atraer y retener a más usuarios, se debe mejorar la experiencia de usuario mediante una interfaz más intuitiva, limpia y fácil de navegar. Las mejoras podrían incluir:

Rediseño de la Interfaz: Optar por un diseño que permita a los usuarios tener un acceso más fácil a las simulaciones y resultados. Incluir paneles de control claros que muestren su progreso y áreas a mejorar de manera visual e interactiva.

Guías Personalizadas: Proporcionar guías interactivas que ayuden a los usuarios a aprovechar todas las funcionalidades de la aplicación, adaptando el contenido a su nivel de conocimiento y experiencia.

Escalabilidad y Seguridad: A medida que la aplicación crece en número de usuarios, es esencial garantizar que la infraestructura técnica pueda soportar una mayor carga sin comprometer la velocidad o la funcionalidad. Para ello, se recomienda:

Feedback del Usuario: Recopilar continuamente información de los usuarios a través de encuestas y análisis de uso para ajustar la funcionalidad según sus necesidades.

Pruebas de usabilidad: Realizar pruebas de usabilidad en diferentes etapas de desarrollo para identificar áreas de mejora en la interfaz y el flujo de la aplicación.

Con estas mejoras, la aplicación puede convertirse en una herramienta esencial no solo para la preparación de entrevistas laborales en inglés, sino también en un recurso integral para el desarrollo profesional. Apostar por su escalabilidad y seguir integrando nuevas tecnologías permitirá que la aplicación sea una referencia en el ámbito de la enseñanza de inglés y la preparación profesional.

6. BIBLIOGRAFÍA

Abelló, A., & García-Solache, J. (s.f.). *Bases de datos NoSQL.* Marcombo. doi:9788426725369

Axelrod, M., & Hamilton, N. (2015). *GraphQL: A data query language.* Apress. doi:9781484217374

Bahdanau, D., Cho, K., & Bengio, Y. (2014). *Neural machine translation by jointly learning to align and translate.* ICLR. doi:arXiv:1409.0473

Barrick, M. R., Shaffer, J. A., & DeGrassi, S. W. (2009). *What You See May Not Be What You Get: Relationships Among Self-Presentation Tactics and Ratings of Interview and Job Performance.* American Psychological Association. doi:978-1557982416

Brown, T. B., & al., e. (2020). *Language Models are Few-Shot Learners.* arXiv. doi:arXiv:2005.14165

Devlin, J. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv. doi:arXiv:1810.04805

Duckett, J. (2011). *HTML and CSS: Design and Build Websites.* Indianapolis, IN: Wiley. doi:978-1118008188

Dutoit, T. (1997). *An Introduction to Text-to-Speech Synthesis.* Dordrecht: Springer. doi:978-0792345643

Erikson, M. (2017). *React in Action.* Manning. doi:9781617293972

Firebase Documentation. (2024). *Firebase Documentation.* Google. Obtenido de https://firebase.google.com/docs/database

Graves, A., & Schmidhuber, J. (2014). *Framewise phoneme classification with bidirectional LSTM and other neural network architectures.* Neural Networks. doi:10.1016/j.neunet.2013.10.005

Howard, J. (2018). *Universal Language Model Fine-tuning for Text Classification.* arXiv. doi:arXiv:1801.06146

Jurafsky, D., & Martin, J. H. (2021). *Speech and Language Processing.* Pearson.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *ImageNet Classification with Deep Convolutional Neural Networks.* Neural Information Processing Systems. Obtenido de https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Medina, B. (2022). *Gestión de estados en aplicaciones React.* HackMD. Obtenido de https://hackmd.io/@blasmedina/H1_r5JieC

Radford, A. (2019). *Language Models are Unsupervised Multitask Learners.* OpenAI. Obtenido de https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

Radford, A. (2019). *Language Models are Unsupervised Multitask Learners.* OpenAI.

Smith, J. (2023). *A Comparative Analysis of Vite and Webpac.* Journal of Web Development .

Snoop Dog. (19 de 11 de 2018). I wanna thank me. El poder de la gratitud. Obtenido de https://youtu.be/Ad3iXEG8o7M?si=fMjpOMumMGWEhWiE

Tailwind CSS. (2024). *Tailwind CSS Documentation* . Tailwind Labs Inc.

Vaswani, A. (2017). *Attention Is All You Need.* arXiv. doi:arXiv:1706.03762

Young, T. (2018). *Recent Trends in Natural Language Processing.* ACL Anthology. Obtenido de https://aclanthology.org/P18-1038/