UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Gandia

XatBot: Uso de grandes modelos de lenguaje con propósitos didácticos para el juego en línea League of Legends

Trabajo Fin de Grado

Grado en Tecnologías Interactivas

AUTOR/A: Benavides Belmonte, Daniel

Tutor/a: Alberola Oltra, Juan Miguel

Cotutor/a: Sánchez Anguix, Víctor

Director/a Experimental: Canut Domínguez, Carlos María

CURSO ACADÉMICO: 2023/2024

ÍNDICE

1. Introducción

Los videojuegos competitivos se han coronado como los reyes del medio por su capacidad de retención de audiencia y la facilidad de monetizar el contenido que brindan periódicamente a sus jugadores.

Esta es una moda que lleva más de una década cogiendo tracción y actualmente no se prevé que vaya a parar por nada ni por nadie, más que algún valle en las gráficas de jugadores en activo debido a algún evento internacional o la salida de otro título muy esperado por los jugadores. Y es tanto de esta manera que este nicho de juegos son los que lideran los tops de “juegos más jugados” en los que se cuelan varios y comparten ecosistema con otros de un solo jugador; ejemplo de esto son “Epic Store” (Epic Games, 2024), “Steam” (Valve, 2024) o “Microsoft Store” (Microsoft, 2024), por nombrar algunas de las más famosas.

Mientras, la Inteligencia Artificial (IA) es un concepto bien arraigado en el mundo de los videojuegos, implementada mayoritariamente con el objetivo de dotar de cierta complejidad a las acciones de los personajes no jugables (NPC) de los juegos o los oponentes de juegos como el ajedrez, representados por la máquina, la rama de las IAs generativas es un mundo que está aún por explorar dentro de este mercado (Filipović, 2023).

Los videojuegos en línea, y en concreto los “Multiplayer Online Battle Arena” (MOBA) están constantemente produciendo datos, tanto de usuarios como de partidas (Vardal, 2022), además del contenido que se genera a su alrededor gracias a creadores de contenido en plataformas como “Youtube” o “Twitch”. A esta tendencia también contribuyen los propios usuarios en foros, compartiendo experiencias y opiniones que hacen avanzar tanto la comunidad como el videojuego.

Este hecho hace que sean socios idóneos para las herramientas de Inteligencia Artificial y análisis de datos que puede que resulten en una explosión de uso de las IAs generativas en los videojuegos como mercado, como parece pensar Andrew Wilsen -CEO de Electronic Arts- (Williams, Games Hub, 2024).

Un ejemplo de MOBA, y supertitán de los videojuegos, es el League of Legends (LoL), el cual surgió como una nueva propiedad intelectual con una clara inspiración al mod Defense of the Ancient (DotA) de otro videojuego en línea (Warcraft 3). Hablamos de “supertitán” debido a los números que arrastra en cuanto a jugadores activos, cantidad de competiciones de “e-sports” y los premios que ofrece a los mejores equipos de cada liga y torneos oficiales.

Dado el interés que suscita el LoL y la competición, una cantidad enorme de usuarios entran y salen las bases de jugadores cada año, aunque el aprendizaje de este videojuego no sea amistoso para los recién llegados. Este paso no es nada trivial para los jugadores que pueda ser el League of Legends su primer MOBA; un videojuego que lleva casi 15 años operando con una gran cantidad de variables que lo componen y en continua evolución y cambio es un reto que puede llegar a ser abismal para ciertos posibles jugadores y que se vean sobrepasados con tanta información que acaben por dejarlo.

Este Trabajo Final de Grado se centra en proporcionar una plataforma web con un chatbot en la que el usuario pueda conversar con la mencionada IA para resolver las dudas de un jugador principiante, así como brindar sugerencias -para ampliar su conocimiento- y cierto tipo de información del usuario que se presentará en pantalla.

1.1 Objetivos del proyecto

El objetivo de este proyecto que nos atañe es el desarrollo e implementación de una plataforma web que integre un agente inteligente con el que poder conversar y obtener información específica del videojuego, con un enfoque a los jugadores más casuales de los MOBA o aquellos jugadores cuya primera toma de contacto con este género sea el League of Legends.

Valorando el esfuerzo que ha de hacerse al introducirse al mundo del League of Legends, la solución implementada tiene una base de información compuesta por datos extraídos de varias fuentes contrastadas, con el objetivo de obtener una perspectiva en cada aspecto esencial del videojuego para que el chatbot sea capaz de reconocer las cuestiones que se le plantean y generar una respuesta acorde de manera fiable e informada.

Con esto en mente, podríamos listar los objetivos secundarios del proyecto como los siguientes:

- Analizar y entender la base fundamental del League of Legends
- Obtener y usar los datos necesarios para el “fine tunning” del chatbot
- Desarrollar una plataforma web con la que interactuar
- Implementar el chatbot en la plataforma web

2. Marco teórico

Durante la extensión del siguiente punto se introducirá el concepto de deporte electrónico y se hablará sobre su impacto en el ecosistema actual de los videojuegos en línea, mas concretamente en el LoL para, seguidamente, revisar varios productos con diferentes enfoques en el mercado que surge de la demanda de información y las ganas de mejorar en el MOBA que nos atañe. Finalmente, para cerrar esta sección se revisará el estado del arte de la tecnología de grandes modelos del lenguaje que conforman el grupo de los chatbots.

2.1. League of Legends: introducción al deporte electrónico

Los esports son competiciones de videojuegos organizadas en un formato profesional, en las que jugadores individuales o equipos compiten entre sí en diferentes títulos de videojuegos en línea. Estas competiciones arrastran a una gran cantidad de personas aficionadas al videojuego en el que se compite y reparten grandiosas cuantías a los equipos mejor clasificados del torneo (Lehnert, 2022).

Habiendo establecido DotA lo que serían las bases para los juegos MOBA, se había creado una fórmula para juegos altamente competitivos, de los cuales el LoL no es una excepción.

Y es tanto así que el 18-06-2011, poco menos de dos años después de salir al mercado, se celebraría su primer torneo mundial como parte de la DreamHack de dicho año, en la que ya se hospedaban otros deportes electrónicos, con un total de 99.500$ de premios a repartir entre los 8 equipos participantes (Esports earnings, 2011). Este fue el primer torneo mundial de muchos que estarían por venir, con premios que rondan el millón de dólares para el equipo vencedor en el último torneo celebrado.

Actualmente, la competición ha permeado todos los estratos de la base de jugadores, habiendo competiciones en los diferentes niveles territoriales, desde una comunidad relativamente pequeña, como lo puede ser una universidad y sus torneos deportivos en los que ahora figuran de la misma manera deportes tradicionales, como el futbol, con deportes electrónicos, como el LoL.

En un marco más amplio de competiciones, existen aquellas clasificadas en una escala de D a S -escala determinada por Riot Games-, que engloba tanto torneos en línea como presenciales, de carácter nacional, como lo es la Superliga española (Liquipedia, 2024)

de la que han salido jugadores españoles de nivel internacional como los de la figura 1, organizada por la LVP (Liga Profesional de Videojuegos) que cuenta con varias divisiones (Liquipedia, 2024), e incluso el ya mencionado torneo internacional de los Worlds (Liquipedia, 2024).

Ilustración 1: "Elyoya", jugador español, levantando la copa de Superliga por segunda vez [Fuente: Movistar eSports - Diario AS]

Y si bien es cierto que el concepto de deportes electrónicos no es novedoso, teniendo los primeros ejemplos a principios de milenio con torneos de otro videojuego en línea, Counter-Strike, la bola de nieve que se ha formado con la explosión exponencial de las bases de jugadores ha hecho que este deporte se convierta en la manera de vivir de mucha gente y las aspiraciones de muchísima más.

Hoy en día, en cuanto al League of Legends se refiere, la escena competitiva, que no representa ni 1% de los jugadores totales, rige cómo se juega al juego, qué personajes se escogen, qué objetos se compran e incluso influyen en las decisiones de la compañía a la hora de modificar aspectos del juego. Tal es la influencia de la escena competitiva que los juegos actuales que aparecen en el mercado y tienen un componente multijugador en ellos están casi obligados a implementar un modo clasificatorio debido a una práctica que ya se ha convertido en un estándar de la industria; apelar al mercado de los deportes electrónicos (Nalli, 2019).

Con lo anterior mencionado, surgieron una gran cantidad de productos y servicios que capitalizarían el nicho de mercado poco explotado que había, enfocados al análisis de la información de partidas, tanto las de los usuarios casuales, como las competitivas de mayor nivel, hojas de ruta para las habilidades y objetos para los campeones e incluso sesiones de entrenamiento que ofrecen ciertos jugadores en la cúspide de la pirámide, con el objetivo de aumentar la accesibilidad de la información.

2.2. League of Legends: mercado del deporte

Debido a la necesidad de mejorar inherente en el espíritu competitivo en la base de jugadores de actividades competitivas, se forma un mercado a explotar en el que ofrecer servicios que brindan información analítica y/o recomendaciones para el juego que puedan conseguir dar al jugador una ventaja sobre sus oponentes. En esta sección analizaremos varios ejemplos exitosos.

La primera aplicación web que se mencionará es op.gg. Esta página, aunque hay varias con el mismo propósito, ayuda a contextualizar los resultados del jugador y el trayecto por las divisiones del juego. Esto no significa que no sea útil para los jugadores más casuales que quieran comprobar los perfiles de sus amigos, por ejemplo, ya que el cliente del LoL ofrece poca información.

Ilustración 2: Interfaz de usuario de la página principal de la página web de op.gg [Recurso propio]

Dicha página brinda unos datos estadísticos detallados sobre los diferentes aspectos del videojuego, como puede apreciarse en la figura 2 (OP.GG, 2024). La funcionalidad principal es el resumen de partidas, en los que te brinda tu resultado y los diferentes objetos que has comprado a lo largo de la partida, información que ya está en el cliente del juego.

Por otro lado, encima de esta esta funcionalidad básica añade varias estadísticas adicionales, como lo son:

- El “winrate”, que muestra en un gráfico de donut el porcentaje de victorias y derrotas en las últimas veinte partidas
- El KDA, la cual calcula una ratio que combina tus asesinatos y asistencias y los compara a las tus muertes totales de la partida
- El porcentaje de contribución en asesinatos “C/Kill”
- La cantidad de wards de control
- Los súbditos asesinados, tanto totales como por minutos

- La media de nivel de los rangos de la partida en la que has jugado

Para finalizar, están las etiquetas que vemos debajo de las estadísticas comentadas, que aun forman parte de una beta, que analizan las estadísticas de la partida y representan un resumen de esta.

Es importante resaltar que el historial de partidas existe en tres diferentes pestañas, las cuales las representan dependiendo del modo de juego que haya escogido jugar el usuario, teniendo un resumen del groso de estas encima de las colas clasificatorias de solo/duo -en las que se puede buscar partida un máximo de dos personas juntas-, clasificatorias flexibles -en las que pueden hacer grupo para realizar la búsqueda hasta un máximo de cinco personas- ARAM (All Random All Mid) y el resto de modos rotatorios que implementan en los diferentes eventos a lo largo del año.

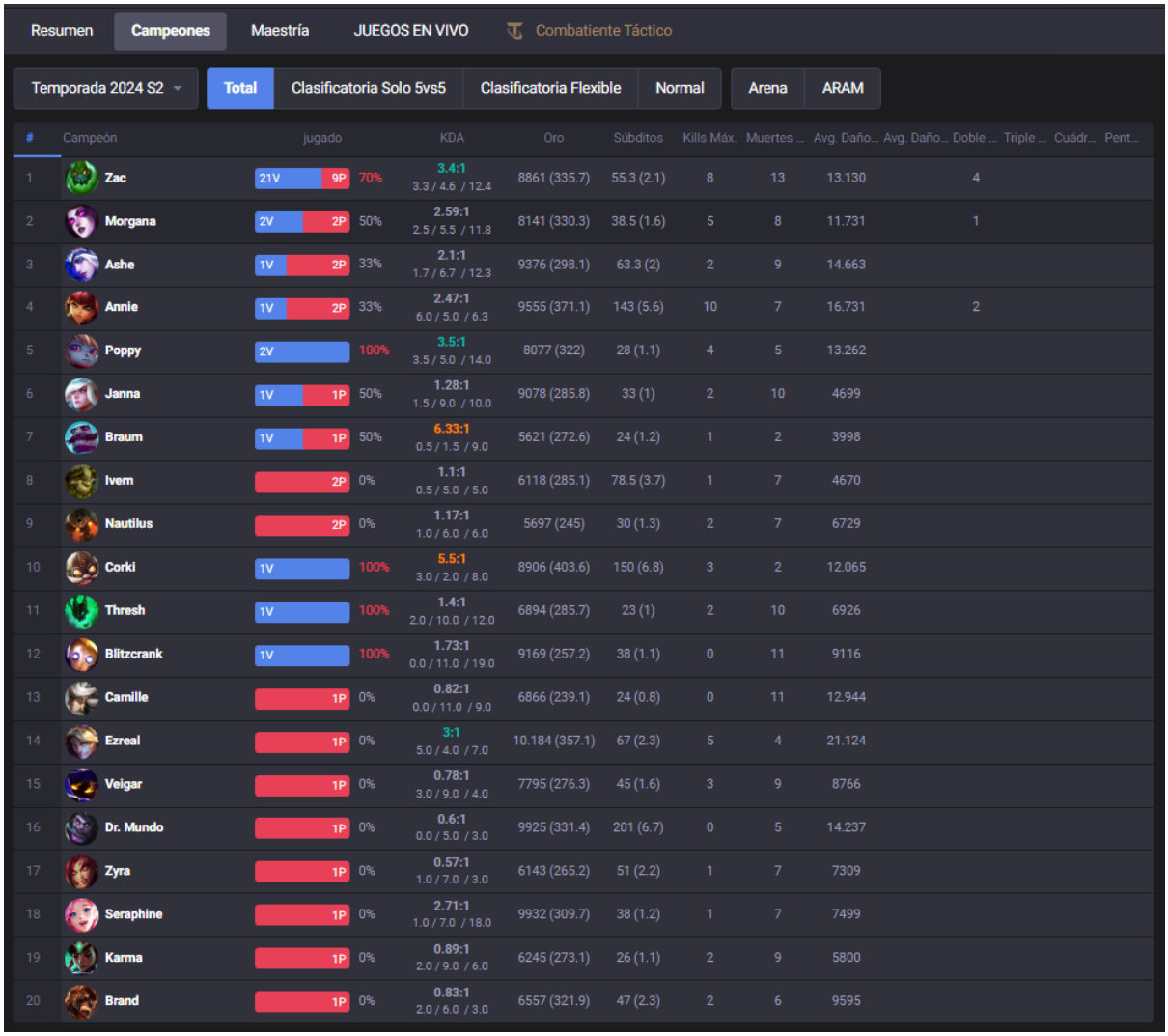


Ilustración 3: Interfaz de usuario de la sección dedicada a los campeones [Recurso propio]

Otro de los puntos fuertes de esta página es la representación estadística de tu desempeño con cada uno de los campeones que has jugado en el total de partidas, en este caso dentro de un periodo de tiempo denominado “split” -que son las divisiones temporales que conforman una temporada regular-. De la misma forma que brinda datos concretos de las diferentes partidas, lo hace de los campeones jugados como puede verse plasmada en la figura 3.

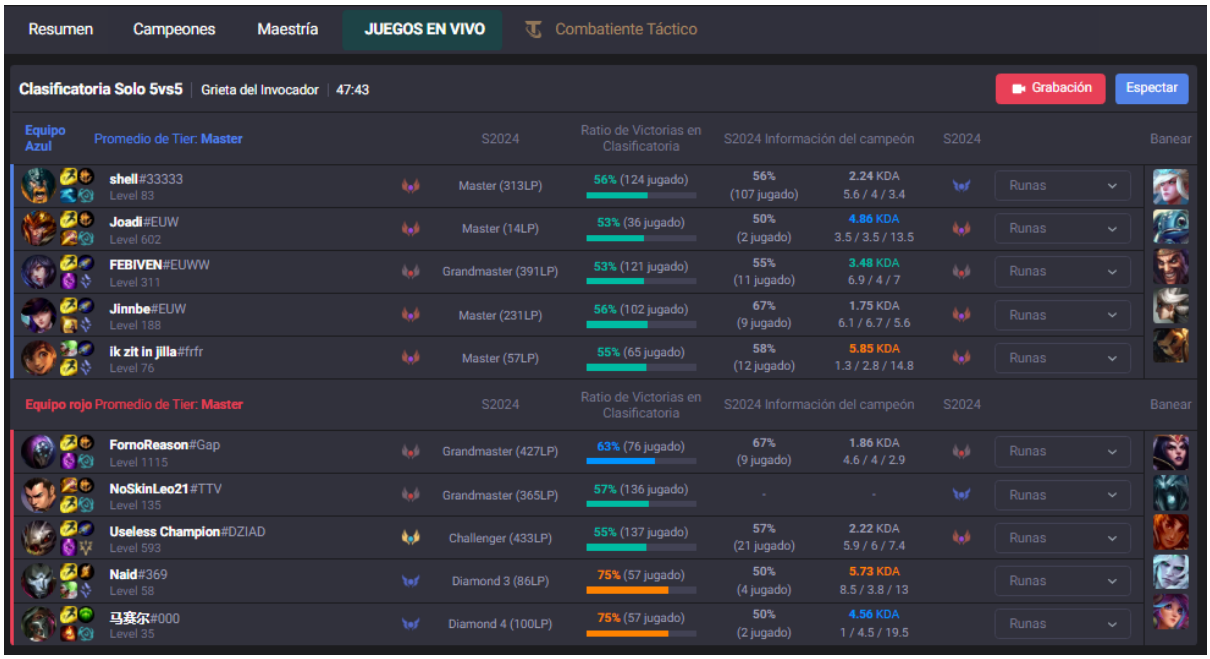


Ilustración 4: Interfaz de usuario de la sección de partidas en vivo [Recurso propio]

La última de las pestañas que podemos ver en la figura 4, la de “JUEGOS EN VIVO”, está destinada a otorgar al usuario información sobre los jugadores de la partida en la que está jugando, pero específicamente de la cola en la que ha buscado partida.

Esta se centra en la ratio de victorias del modo y del campeón en el modo, añadiendo el KDA medio que el jugador tiene con el personaje en concreto que haya decidido escoger, así como su rango en el “split” anterior y un desplegable de las runas que ha seleccionado seguido de los personajes que han prohibido para esa partida. El desplegable te permite copiarlas para poder utilizarlas tú mismo.

Por último, la funcionalidad más útil de esta sección ha de ser la de grabar y espectar. Estas le permiten al jugador ser capaz de descargar y revisitar la partida que han jugado para, por ejemplo, analizarla y destacar errores que hayan cometido o recortar ciertas partes para su uso personal. Por otro lado, el espectar puede ser usado de la misma manera que grabar

para partidas de alto nivel, viendo desde el punto de vista de un jugador en el top del juego como afronta una partida de tal nivel.

El siguiente producto que vamos a discutir es Backseat AI, el cual es desarrollado por una “start-up” fundada por el famoso “streamer” y “youtuber” Tyler1 –el cual alcanzó el máximo rango en clasificatorias con todos los roles-. Este está aún en fase temprana del desarrollo, contando con un servicio de suscripción de pago por 4.99$ mensuales.

Esta herramienta se centra en suplir un rol de entrenador profesional en tiempo real, generando una perspectiva de conocimiento y comentario en base a tus acciones en la partida, proveyendo al usuario de directrices y hojas de ruta para proseguir con su partida, con el añadido de reflejar la personalidad distendida y cercana del patrocinador, Tyler1 al que podemos ver en la figura 5 (Backseat GG, 2024).

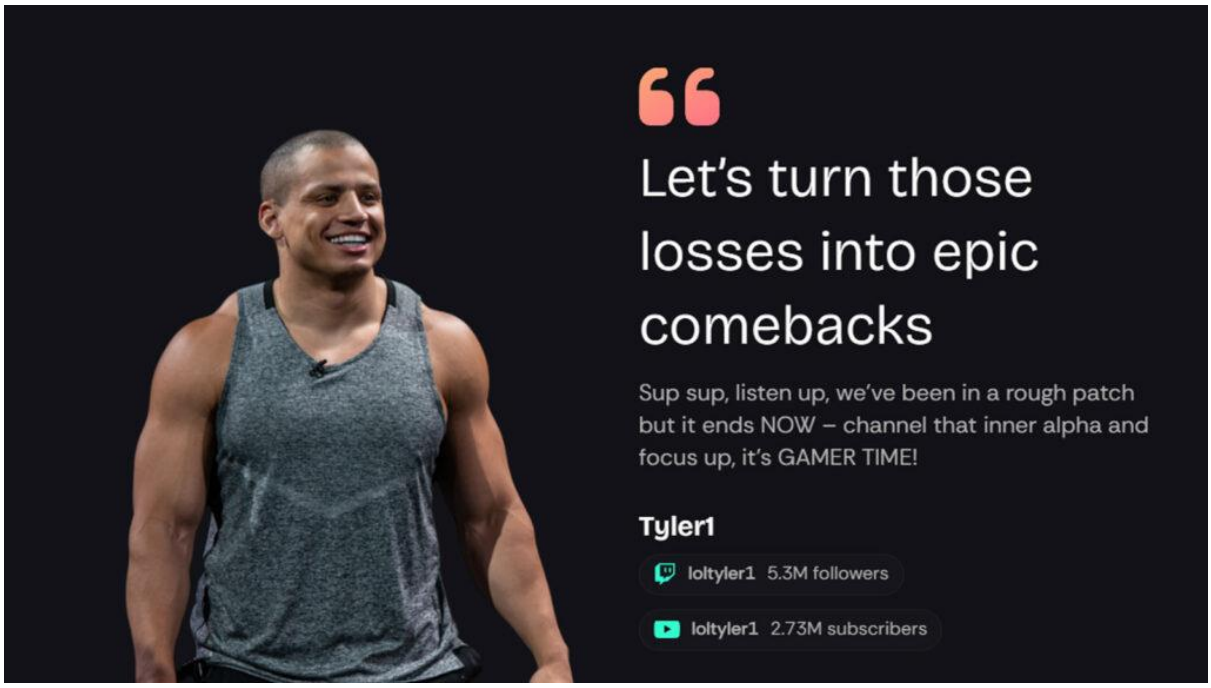


Ilustración 5: Imagen promocional de Backseat AI [Recurso extraído de Backseat AI]

Las funcionalidades no acaban ahí, sino que también es capaz de ofrecer análisis antes de empezar la partida, en la selección de campeones -donde dará consejos sobre que personaje escoger o incluso como jugar tu enfrentamiento-, así como un resumen una vez acabada la partida, dando información personalizada en base a las estadísticas personales y del equipo como de tu enfrentamiento en concreto.

Por otro lado, también contará con una opción de conversación en tiempo real en la que podrás preguntar a la Inteligencia Artificial desarrollada por el equipo de Backseat AI, tanto por voz como por texto, y generará una respuesta basada en los diferentes aspectos a tener en cuenta de la partida en la que estés en ese momento.

Como punto final, el servicio no se centrará únicamente en el público anglosajón. En la propia página web se anuncia la adición de por lo menos dos lenguas más: el coreano y el chino -dos de los mercados más grandes en el League of Legends, tanto por número de usuarios como por talento probado en competiciones internacionales-.

2.3. “Chatbots” y tecnologías

Con el auge de la Inteligencia Artificial en el último lustro se ha cimentado como una de las tecnologías más disruptivas de nuestra era, con un gran impacto en muchos ámbitos de nuestra vida. En lo que al usuario final respecta, la faceta de esta tecnología de la que más uso hemos dado o que más ha impactado en nuestra vida diaria son las IAs generativas (Adhithya, 2023).

Las Inteligencias Artificiales generativas se dedican a generar contenido propio u original, mediante sus propios algoritmos y redes neuronales complejas, a partir de un “prompt” o una entrada de un usuario. Estos contenidos pueden ser textos, como lo podrían ser un ejemplo de ejercicio de sintáxis o un “dataset” para entrenar a otra IA, o imágenes, dentro del rango de cuadros independientes o una sucesión de imágenes formando un video.

Los “chatbots” son un uso específico de la rama de IA generativa con el objetivo de poder mantener una conversación con un usuario, respondiendo de manera coherente a las entradas de dicho usuario.

Esto no es para nada un concepto novedoso, ya que tan pronto como en el año 1966 se desarrolló en el Instituto de la Tecnología de Massachusetts (MIT) el primer prototipo de chatbot, apodado “Elizza”, que consistia en una interfaz de usuario rudimentaria con la que comunicarte con la máquina, que gracias a algoritmos de concordancia y sistitución de patrones podia generar una respuesta Weizenbaum, Joseph (1966-01). ELIZA—a computer program for the study of natural language communication between man and machine.

En la actualidad los “chatbots” existen como producto de los Grandes Modelos del Lenguaje (LLM), que utilizan la lingüística computacional y modelos estadísticos junto con la

gran capacidad de computación de datos del modelo para generar las respuestas, ahora basadas en entradas de texto y voz .

Estos están implementados en infinidad de sitios, mayoritariamente en soporte de usuario en comercios o como un interlocutor multiusos, como puede ser el ejemplo de ChatGPT de OpenAI.[1]

Si bien la intención original es la de interactuar consensualmente con un usuario, últimamente se está viviendo una tendencia en la que se utilizan estos “chatbots” como una versión mejorada de los típicos “bots” para aumentar artificialmente las interacciones de tus cuentas de redes sociales, o las de otras personas, a demanda. Las consecuencias se pueden ver en plataformas como Facebook, en las que hay comunidades completas de IAs generativas creando imágenes y teniendo conversaciones entre ellas mismas (Shirazi, 2021).

Como en cualquier nicho de mercado, hay diferentes productos con los que el usuario pueda interactuar y en el mercado actual los LLM más populares y potentes esán respaldados por grandes compañias:

- **Llama** (Meta, 2023): Gran Modelo del Lenguaje lanzado por la división de Inteligencia artificial de Meta (Meta AI), con la característica de ser “open source” y contando con un código escrito en el lenguaje de programación Python
- **ChatGPT** (OpenAI, 2022): LLM lanzado por la empresa OpenAI, que poco después sería adquirida por Microsoft, contando con una licencia privativa y con su código escrito en el lenguaje de programación Python
- **Bing LLM** (Microsoft, 2023): chatbot integrado en el buscador Bing, de la empresa Microsoft y desarrollado por OpenAI
- **Claude** (Anthropic, 2023): LLM desarrollado por la empresa Anthropic, fundada por exmiembros de OpenAI.

Cabe destacar que varios de estos modelos están montados en la nube de la plataforma Voiceflow, la cual ofrece servicios, mediante llamadas a su API, para poder tener las conversaciones con el usuario final de manera sencilla y relativamente rápida, ya sea de temas generales o de los datos de entrenamiento proporcionados por el desarrollador. Y esto último es otra de las ventajas que ofrece este servicio, ya que tiene una sección dedicada a

[1] ChatGPT

los datos de entrenamiento para el modelo, “Knowledge Base” como la denominan ellos, que hace de este proceso uno casi sin interrupciones.

Por un lado, una de las opciones que se barajó como posible solución para resolver el problema que este trabajo planteaba fue el modelo Llama2, que si bien era interesante la propuesta de ser código abierto y poder hacer tuyo, de alguna manera el modelo, se acabó descartando por las limitaciones técnicas de las máquinas a nuestro alcance, ya que el proceso de entrenamiento y generación de texto, esto último sobretodo, parecían completamente inviables para un producto final en términos de tiempo de ejecución,

Finalmente, y por lo que respecta a este proyecto, después de probar varios productos y versiones de los mismos, se ha optado por ChatGPT, en concreto su versión 3.5-turbo la cual, aun sufriendo de alucinaciones de IA generando texto careciente de sentido, como cabe esperar, se centra en la generación únicamente de texto, que es lo que en este trabajo se busca. Y es por estas especificaciones y las ventajas que ofrece la plataforma de Voiceflow que se ha acabado usando este modelo en concreto como pilar sobre el que desarrollar la solución tecnológica que atañe a este trabajo final de grado.

Habiendo revisado varias aplicaciones existentes podemos lograr ver los resquicios donde aparece la viabilidad y funcionalidad de nuestra aplicación, ya que en las anteriores se presupone un conocimiento moderado, cuanto menos, del funcionamiento del videojuego, de sus parámetros óptimos y de los términos y conceptos generados por el paso del tiempo.

Podemos afirmar que la aplicación que nos atañe no solo no se solapa con las previamente revisadas, sino que puede ser un valioso recurso complementario para el público objetivo de este proyecto. Los jugadores menos experimentados o aquellos que quieran embarcarse en el proyecto de aprender a jugar tendrán una plataforma en la que esté la mayoría de la información del videojuego de manera centralizada y que cuenta con una experiencia de usuario intuitiva y liviana gracias al uso de un chatbot.

3. Propuesta

En este trabajo se pretende conseguir una aplicación web en la que se integre un chatbot con el objetivo didáctico de ofrecer información a jugadores poco experimentados en la grieta del invocador.

A continuación, se detallarán los requisitos que deberán formar parte de la plataforma como funcionalidades y una breve descripción de los diseños con los que se han desarrollado este proyecto de final de grado, tanto la arquitectura del proyecto como representaciones del flujo de datos.

Para finalizar se comentará el proceso de implementación de todas las partes que trabajan al unísono para hacer funcionar esta plataforma web.

3.1. Requisitos del chatbot

Para el desarrollo del trabajo que pretende brindar una solución con una integración de un chatbot, antes siquiera de elegir la tecnología o la herramienta que más se acercase a las necesidades que esta requiera, se debían dejar claros los objetivos que debería cumplir el mencionado chatbot.

Y es de esta manera que surgen las preguntas “¿Qué debe saber? ¿En qué profundidad debería bucear para brindar respuestas? ¿Debería centrarse en un aspecto del producto en concreto?”, y es que en un juego en línea que lleva cerca de 15 años disponible en el mercado, son cuestiones más que válidas.

Después de sopesar las necesidades y el público objetivo para la magnitud de este trabajo, surgieron los requisitos por los que se regiría el desarrollo de dicho proyecto:

- Ofrecer información sobre los campeones: estos son los personajes jugables del juego en línea League of Legends
  (LoL), que en el momento de escribir este texto hay un total de 167 opciones entre las que elegir. El chatbot deberá ser capaz de dar una descripción general sobre el campeón y tener el conocimiento de las habilidades y sus efectos.
- Ofrecer información sobre los objetos: estos se compran dentro de una partida con el oro que consigues jugando la partida en sí, deberá ser capaz de dar una respuesta con la descripción de las propiedades que brinde el objeto final. Se dividen en las siguientes categorías:
  - Consumibles
  - Baratijas (trinkets, solo se puede poseer uno en un momento en concreto)
  - Distribuidos (objetos específicos de los diferentes modos de juego, los campeones, de las propiedades pasivas llamadas “runas” o del objetivo neutral)
  - Botas

    - Objetos básicos
    - Objetos épicos
    - Objetos legendarios
    - Objetos míticos
- Ofrecer información sobre los términos y conceptos: la información del chatbot deberá ser suficiente para responder las preguntas de los usuarios sobre los diferentes términos que han surgido en la vida del League of Legends, de igual manera tendrá que ser capaz de responder las dudas sobre los conceptos clave del juego que ayudarán a contextualizar de mejor manera las acciones, tanto del usuario como del resto de jugadores de la partida.

3.2. Tecnologías

Para esta tarea se ha elegido el “framework” de desarrollo web Next.js, que a su vez es una ampliación de la librería con la que se puede trabajar en js o typescript llamado React. Uno de los puntos fuertes de dicho “framework” es el concepto de los componentes -trozos de código que forman partes de la plataforma web resultante y en los que se ejecuta código de manera autocontenida-.

El objetivo de Facebook al desarrollar la mencionada herramienta de creación de aplicaciones web era crear una forma de manejar la representación de la información del usuario en las páginas webs de manera dinámica, de manera que el desarrollo fuese escalable y sencillo (Gackenheimer, 2015), y actualmente se ha convertido en el estándar en el desarrollo de web.

En cuanto a otorgar un estilo a la aplicación, se ha optado por utilizar “Tailwind” -un “framework” para CSS de alto nivel basado en la definición de clases de estilo para los “tags” HTML-. Este “framework” tiene claras ventajas al desarrollo desde cero de un estilo para la plataforma, combinando la potencia de las clases preestablecidas -las cuales cuentan con un diseño sólido y coherente con el conjunto de clases disponibles- con la posibilidad de poder crear clases propias personalizadas o reescribir las clases base del “framework”.

Finalmente desarrollaremos la implementación del “back-end” y las características que incidieron en la decisión final de escoger Python como la tecnología a usar por encima de la funcionalidad de enrutación dinámica que ofrece el “framework” de Nextjs para montar las APIs para las páginas web.

Por un lado, el framework de Nextjs ofrece la funcionalidad de “API Routes”, la cual es una solución para construir APIs públicas con dicho software. Funciona de una manera sencilla, rigiéndose por un directorio “/api” dedicado para los archivos pertenecientes a la API, que está ubicado dentro del directorio “pages” donde están conjuntamente los archivos que conforman la parte de “front-end” de la aplicación.

Todos los archivos dentro de la carpeta “/api/*” serán tratados como una ruta única por el “framework”, lo que conlleva que para cada “endpoint” se corresponda a un único archivo con la extensión JS, aún que puede evitar este obstáculo declarando varias respuestas dependiendo del tipo de petición -GET, POST, UPDATE, DELETE-.

Además de lo anteriormente mencionado, ofrece rutas dinámicas para ofrecer una funcionalidad más personalizada, en el caso de que no sepas la convención de la ruta de la API, funcionando de la siguiente manera: en el caso de querer obtener o rediriguir a un usuario a su página personal, se llamaría a la ruta “/pages/api/xat/[usuario].js” a la cual respondería si llamas a la ruta “…/xat/Antonio” como a la ruta “…/xat/Carla” de la misma manera, pero con un contenido diferente.

Por otro lado, se barajó el “framework” para Python “Flask”, el cual es homólogo a otros frameworks para “Javascript” como lo podría ser Express. Este, en su versión más básica, funciona declarando reglas o rutas estáticas, con un símil al direccionamiento de directorios, a los cuales realizar peticiones HTTP. También puede ampliarse el funcionamiento de este framework con la implementación de clases como la de “View” para aportar información de manera dinámica para diferentes usuarios, exprimiendo al máximo el jugo que ofrece el lenguaje de programación orientado a objetos.

Merece la pena destacar el hecho de que ambos tienen un servicio para el desarrollo, o un “development server”, integrado para el desarrollo e implementación de las funcionalidades del servidor final, aunque solo el “framework” de Nextjs tiene un servicio nativo para desplegar las aplicaciones finales en un servidor de producción, utilizando los servidores de Vercel -empresa desarrolladora del “software”- que hospeda y ofrece una dirección URL para el acceso al servidor.

Para terminar, se acabó decantando la balanza a favor del “framework” de Python por varias razones:

- Conocimiento del procesado de datos: para esta tarea se posiciona “Python” delante de “Javascript”, tanto en el estándar de la industria como en la experiencia en la materia que se tenía a la hora de desarrollar la aplicación
- Requisitos del “back-end”: el proyecto tiene unos requisitos pequeños para la parte del servidor, contando con pocas reglas
- “Framework” para las consultas: en el lenguaje de “Python” ya había desarrolladas herramientas para hacer más simples las peticiones a la API de Riot Games

3.3. Diseño

En el apartado de diseño de mostrarán en figuras las ideas sobre las que se cimienta la aplicación web que se ha desarrollado, así como comentar los componentes que las conforman.

3.3.1Diagrama general del proyecto

Para la solución de este TFG, habiéndose desarrollado una aplicación web con la que el usuario pueda interactuar, se ha construido con el diseño general de la figura 6, el cual tiene en cuenta tanto la interfaz con la que interactuará el usuario como los engranajes invisibles que dotan de funcionalidad a esta aplicación.

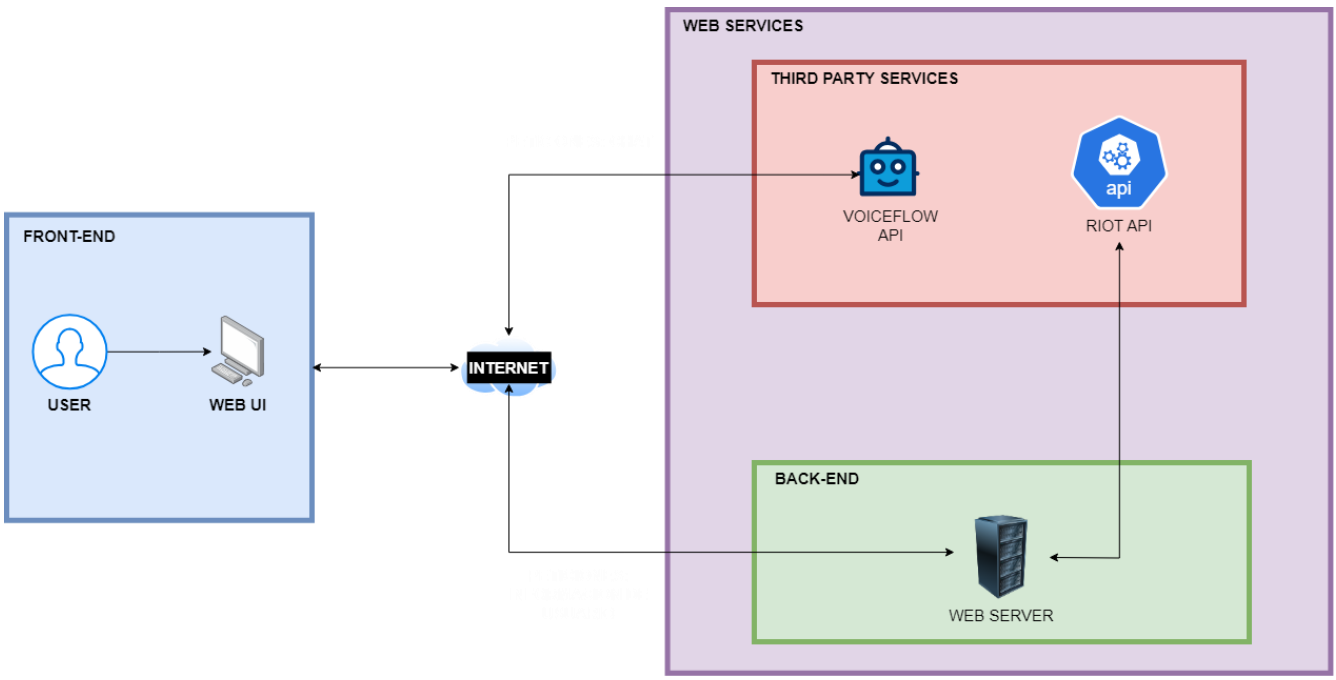


Ilustración 6: Diseño de la arquitectura general del proyecto [Recurso propio]

Esta tiene en cuenta tanto la interfaz con la que el usuario se relacionará, así como los servicios de la web que engloban las peticiones a la API de Voiceflow para la conversación con el "bot" y finalmente las peticiones, y procesado de la información del server, a la API de Riot Games, de la que se obtiene la información sobre el juego.

3.3.2 Diagrama de los servicios del servidor

En la figura 7 se puede apreciar más en detalle el flujo de la información de las peticiones que se realizan con el objetivo de obtener los datos de la cuenta del usuario en el juego en línea.

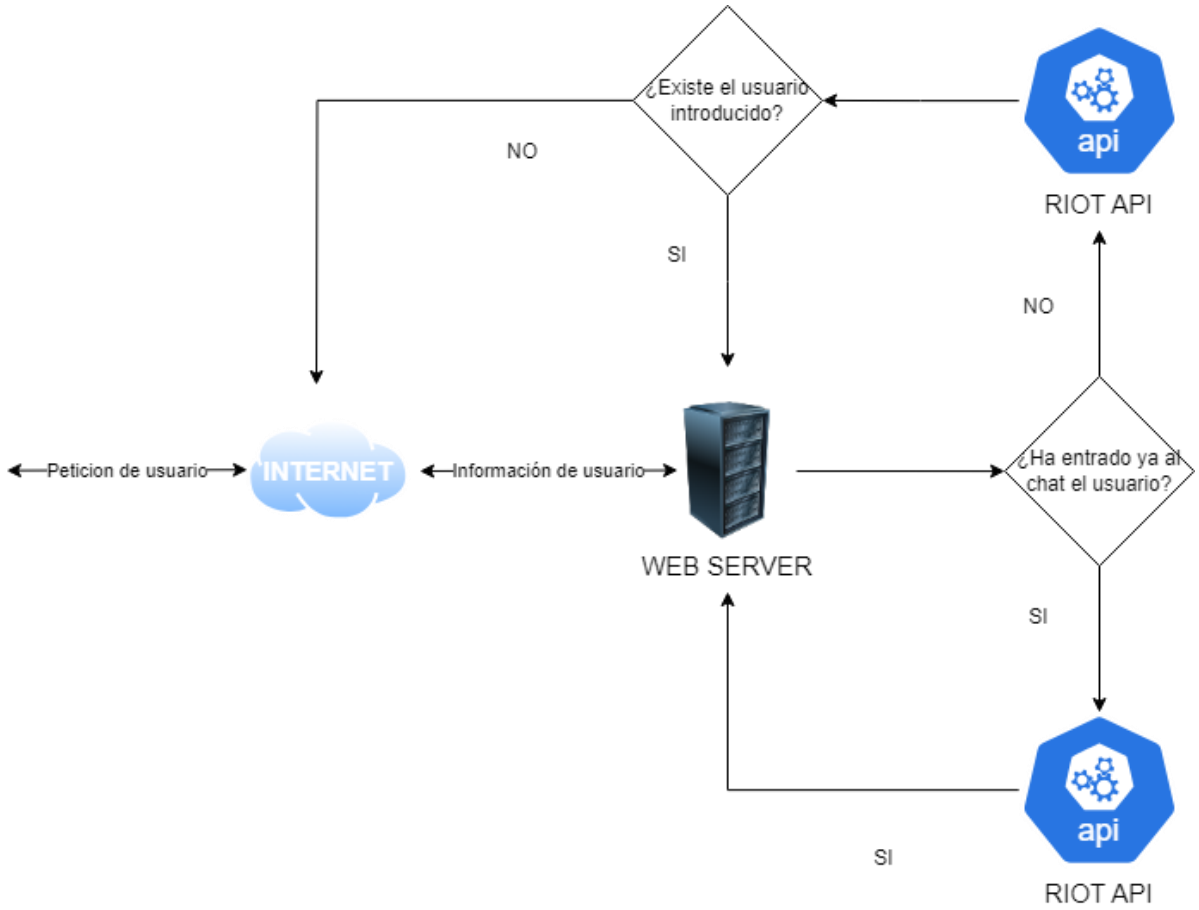


Ilustración 7: Diseño del diagrama de flujo de la información enviada a la API de Riot Games y recibida de esta [Recurso propio]

Para comenzar, la petición del usuario al introducir información en la web se envía al servidor y en este se genera una dicotomía:

- El usuario ya ha entrado a la página del chat: se recoge la información relacionada con la cuenta, como estadísticas de las últimas partidas o los personajes más jugados
- El usuario no ha entrado en el chat: se consulta a Riot Games si existe tal usuario introducido y si existe se avanza a la página de chat, en caso contrario se redireccionaría otra vez a la página principal para que reintroduzca el nombre de invocador

3.3.3 Diagrama del servicio de Voiceflow

En cuanto al diagrama que describe el servicio del chat, podemos observar el recorrido que tienen las entradas del usuario en la parte de chat de la aplicación web en la figura 8, ya sea si acaba de ingresar en la plataforma o si está realizando alguna pregunta al chatbot.

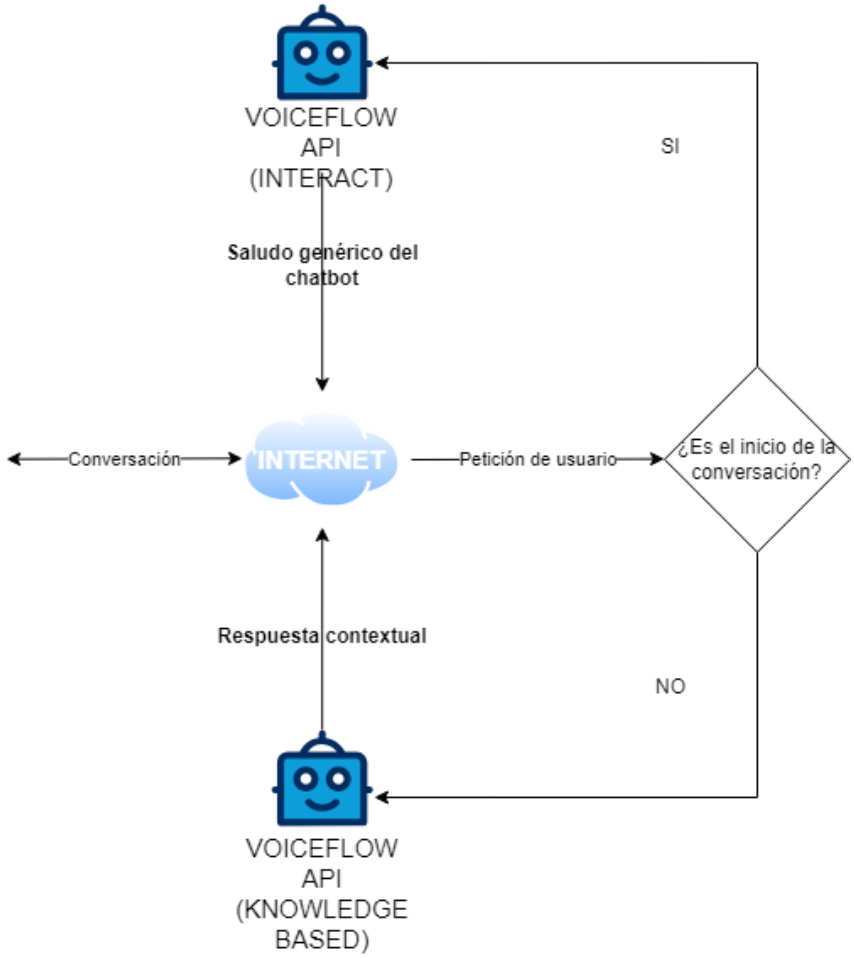


Ilustración 8: Diseño del diagrama de flujo de la información enviada a la API de Voiceflow y recibida de esta [Recurso propio]

Estas dos posibilidades hacen uso de dos servicios diferentes ofrecidos por la compañia de Voiceflow:

- Interact: la funcionalidad de “interact” o el “interaction model” no es más que un modelo que debe ayudar al desarrollador a contextualizar la experiencia que va a tener el usuario.
    - En el caso de este trabajo se encarga de, cuando carga la página, recibir al usuario con un mensaje del estilo “¡Hola! Soy tu asistente para cualquier pregunta que tengas sobre el LoL”, además de recibir la “query” o pregunta que tenga el usuario para pasarla al modelo con la información.
    - Una vez recibida la pregunta se comunicará con el agente de IA que se encargará de generar una respuesta acorde a la entrada para devolverla al usuario
- Knowledge Base: es un servicio que se encarga de almacenar y gestionar la información concreta con la que se quiera especializar el modelo para las interacciones que vaya a tener, para poder dirigir o aconsejar de manera más precisa al usuario final.

3.2.4. Diseño del “front-end”

En lo referente a la interfaz con la que el usuario interactuará para hacer servir las funcionalidades de la aplicación web, se plantearon unos diseños base sobre los que construir la fachada de la plataforma web.

Ilustración 9: Diseño base de la “landing page” [Recurso propio]

Para empezar, comentaremos el diseño inicial de la “landing page”, la cual se puede apreciar en la figura 9. Este es un diseño minimalista, centrado en aportar la información estrictamente necesaria para el usuario final, con el único objetivo de que introduzca el nombre de invocador y nada más.

A continuación, seguiremos la sección del diseño del “front-end” con el diseño inicial de la página que contiene el chat, el cual se puede apreciar en la figura 10.

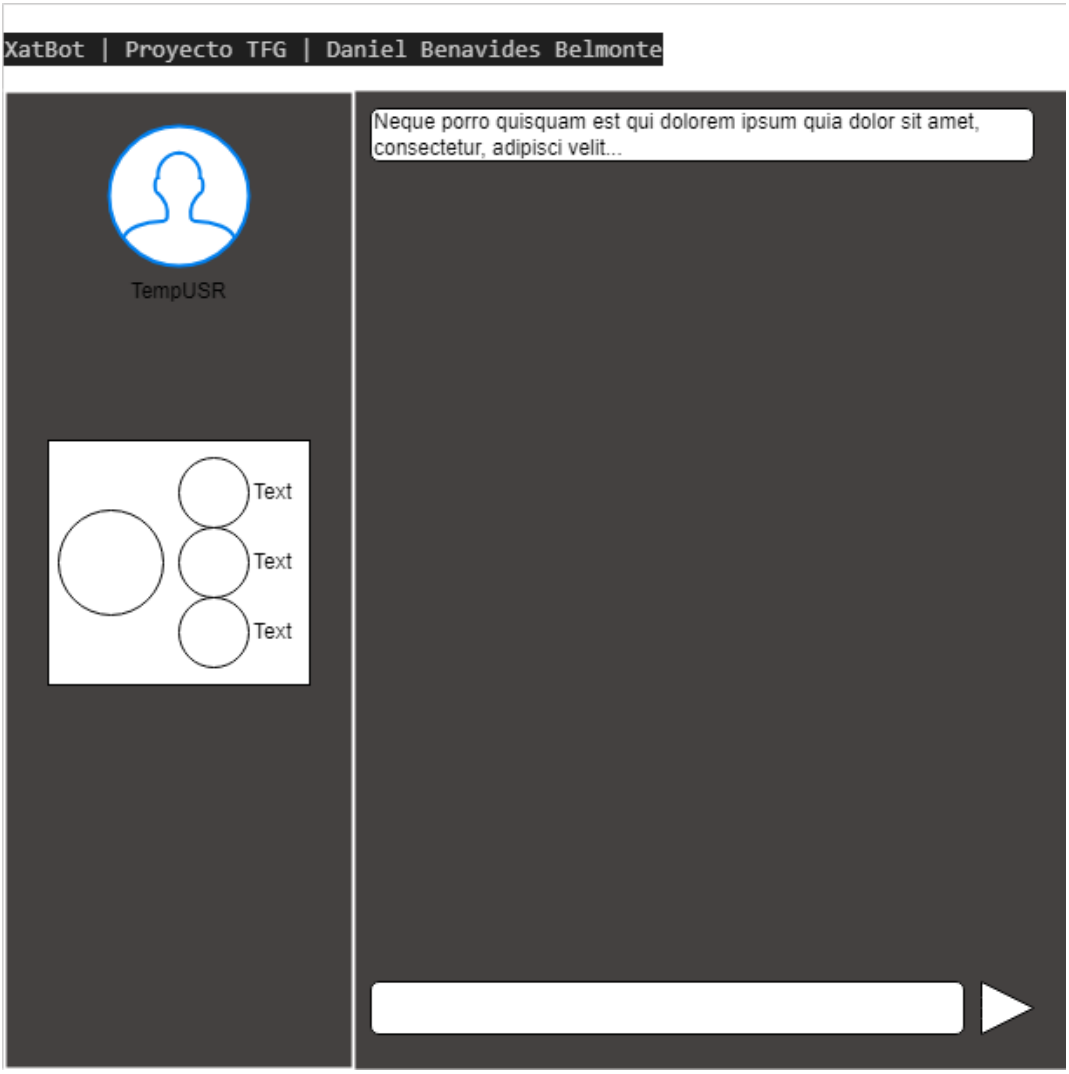


Ilustración 10: Diseño base de la página de chat [Recurso propio]

Esta página está destinada a albergar la funcionalidad principal de la aplicación que se está desarrollando, el espacio donde conversar con el chatbot para resolver las dudas que el usuario pueda tener.

El objetivo de esta página es exponer cierta información base relacionada con la cuenta de invocador del League of Legends que haya introducido el usuario, con la idea de darle cierta sensación de personalización, en una de las secciones.

La otra sección, aquella que por uso e importancia tiene más peso en la aplicación web, está destinada para el chat. Este tiene un diseño sencillo con cuadros de texto con bordes redondeados y un cuadro de texto donde introducir las preguntas.

3.3 Implementación del chatbot

En lo referente a la implementación de la funcionalidad del chatbot, esta se ha llevado a cabo en varias fases diferenciadas entre si, en base a los diferentes objetivos que se intentaban cumplir.

La primera fase constituye la creación del flujo por el que van a pasar el usuario al interactuar con el "bot" o, dicho de otra manera, la contextualización de la funcionalidad completa que el chatbot va a brindar a la aplicación, el cual se ve representado en la figura 11.

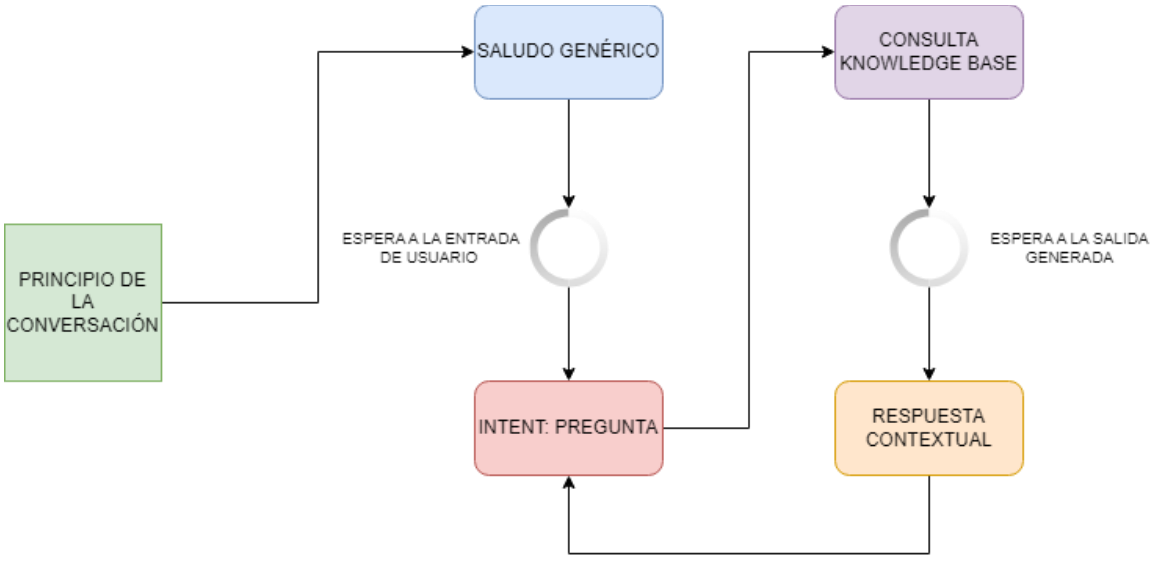


Ilustración 11: Diagrama de flujo del funcionamiento del chatbot de la página web [Recurso propio]

Como se ha descrito anteriormente, el “interact model” sirve para acotar un marco para el “bot” y que este sepa manejarse en las interacciones que vaya a tener con el usuario,

orientándolo en una dirección, de esta manera podemos ver el itinerario por el que pasará cada vez que se inicie una instancia, única por usuario.

El “intent” que podemos observar declara un caso de uso con el cual el bot sabrá que ha de consultar la base de conocimiento y brindar una respuesta contextual acorde a la información proporcionada por el desarrollador.

Para finalizar con esta parte del desarrollo, se generó una variedad de saludos genéricos para darle un toque aleatorio y evitar hacer cada interacción redundante.

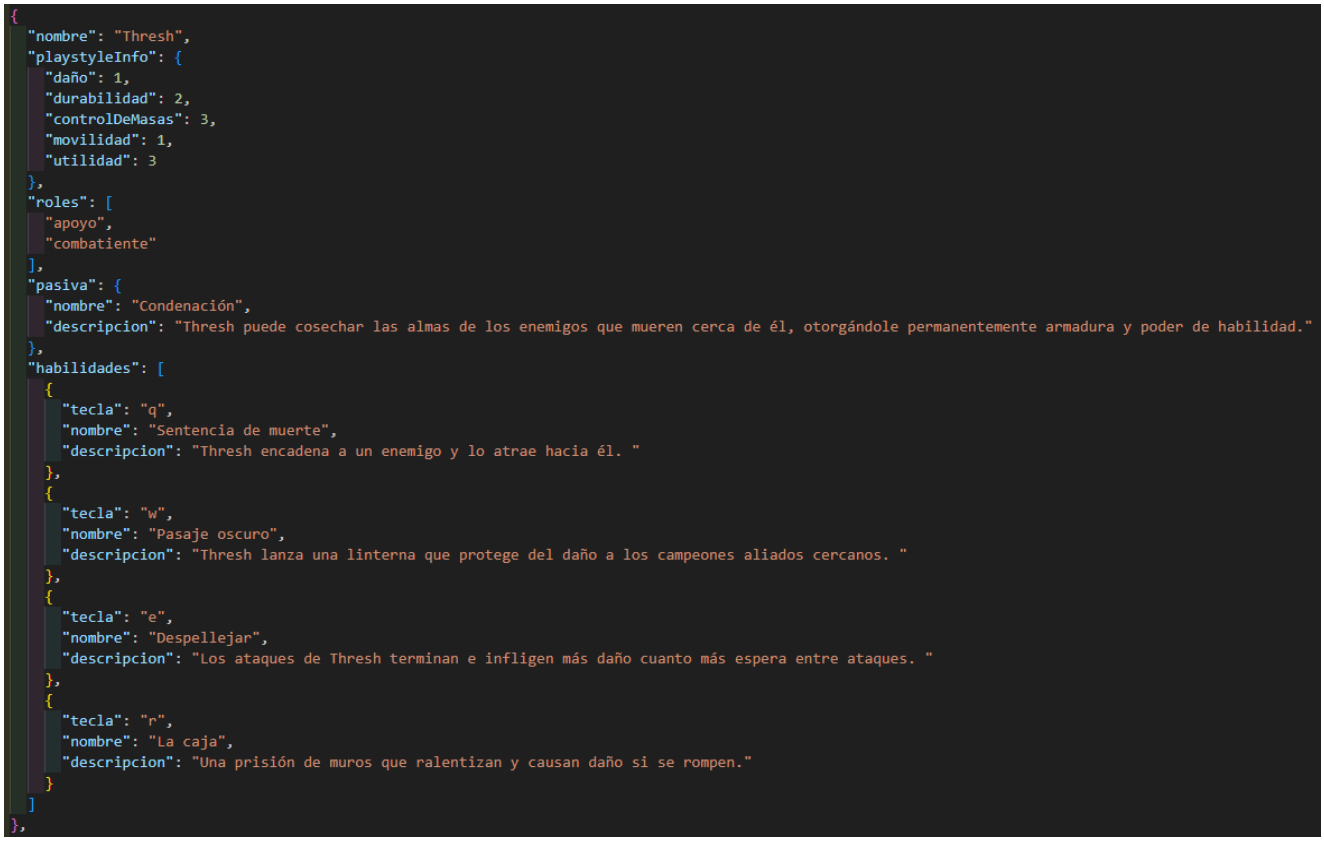
La segunda fase se centra en la fuente del conocimiento y la estructuración de los datos para poder brindar información actualizada y curada al usuario, ya que, en el caso de nuestro modelo ChatGPT[2], lo que limita en cierta manera algunos conceptos o campeones que hayan surgido desde esa fecha hasta ahora.

Para los datos de los campeones del videojuego, estos se obtuvieron de la API oficial de Riot Games [3] la cual nos ofrece una descripción bastante detallada de los campeones y sus estadísticas, conjuntamente sus habilidades y sus estadísticas, entre otras.

Con una cierta cantidad de testeo, se hace evidente que la información cruda que se puede obtener del enlace anterior, aunque bastante específica, se hace demasiado grande para la división en “chunks” de la misma, por lo que inducia a errores constantes, como no encontrar la información o “alucinaciones de la IA” que mezclaba campeones y habilidades por igual.

Se acabó optando por reducir la información con la que se entrenaba el chatbot, así como curar el texto que aparecia en ella, eliminando diversos “tags” como el de <active> </active> que plagaban las descripciones de las habilidades, obteniendo el resultado final mostrado en la figura 12.

[2] En el momento de desarrollo estaba actualizado con fecha 23/04/2023
[3] Enlace a la información de campeones



```
{
  "nombre": "Thresh",
  "playstyleInfo": {
    "daño": 1,
    "durabilidad": 2,
    "controlDeMasas": 3,
    "movilidad": 1,
    "utilidad": 3
  },
  "roles": [
    "apoyo",
    "combatiente"
  ],
  "pasiva": {
    "nombre": "Condenación",
    "descripcion": "Thresh puede cosechar las almas de los enemigos que mueren cerca de él, otorgándole permanentemente armadura y poder de habilidad."
  },
  "habilidades": [
    {
      "tecla": "q",
      "nombre": "Sentencia de muerte",
      "descripcion": "Thresh encadena a un enemigo y lo atrae hacia él. "
    },
    {
      "tecla": "w",
      "nombre": "Pasaje oscuro",
      "descripcion": "Thresh lanza una linterna que protege del daño a los campeones aliados cercanos. "
    },
    {
      "tecla": "e",
      "nombre": "Despellejar",
      "descripcion": "Los ataques de Thresh terminan e infligen más daño cuanto más espera entre ataques. "
    },
    {
      "tecla": "r",
      "nombre": "La caja",
      "descripcion": "Una prisión de muros que ralentizan y causan daño si se rompen."
    }
  ]
},
```


Ilustración 12: Captura del "dataset" con la información de los campeones

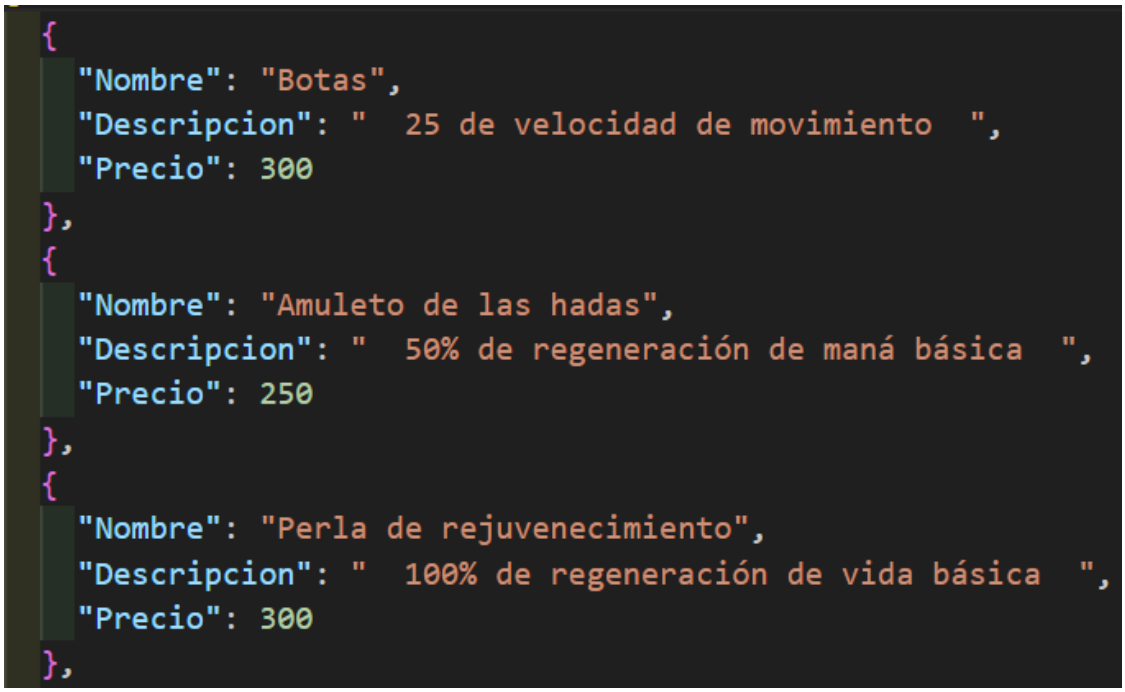
Con relación a los campeones, estos están divididos en varias categorías definidas por los desarrolladores dependiendo de qué rol tomen en una partida o las características de las habilidades de este, con lo que se decidió plantear un "dataset" para qué el chatbot pudiese informar al usuario de las características de cada una de ellas.

La información que contiene el archivo JSON se ha obtenido de la wiki hecha por la comunidad del juego en base a la información oficial y se ha ido actualizando con el paso del tiempo [4].

Por otra parte, para obtener los datos de los diferentes objetos del LoL se utilizó el mismo mecanismo, realizando una llamada a la API [5] para recogerlos en formato JSON.

[4] Enlace a la página de la Wiki del LoL con la información de cada clase
[5] Enlace a la dirección de la API

Esta información también paso por un proceso de curado con el cual se redujo la cantidad de información, a un volumen que contuviese los datos estrictamente necesarios, y se modificó las descripciones para dar un contexto más amplio de los beneficios de cada uno de los objetos, resultando en la siguiente estructura:



```
{
  "Nombre": "Botas",
  "Descripcion": "  25 de velocidad de movimiento  ",
  "Precio": 300
},
{
  "Nombre": "Amuleto de las hadas",
  "Descripcion": "  50% de regeneración de maná básica  ",
  "Precio": 250
},
{
  "Nombre": "Perla de rejuvenecimiento",
  "Descripcion": "  100% de regeneración de vida básica  ",
  "Precio": 300
},
```


Ilustración 13: Captura de la información de los objetos [Recurso propio]

Finalmente se obtuvieron los diferentes términos que se usan en el chat del juego y los conceptos clave de este de la “wiki” del videojuego que, aunque esta no es una fuente oficial de información, tras revisar todos y cada uno de los conceptos se catalogaron como válidos [6]. Cabe destacar que, la información que se encuentra alojada en la página, debido a la gran influencia anglosajona, está en su totalidad en inglés.

Como se puede comprobar, las diferentes entradas en la web tienen una o varias entradas o descripciones, con lo que se procedió a examinar y elegir aquellas que fuesen de mayor relevancia y a escoger sus descripciones que mejor se adaptasen al objetivo de este trabajo, teniendo como resultado un archivo con una llave, siendo esta el concepto o término, y su descripción como valor.

[6] Enlace a la página de la Wiki que contiene la terminologia y conceptos del videojuego

```
{
  "10 death powerspike": "Se refiere a un meme de la comunidad donde alguien proclama en br
},
{
  "1v1": "Un desafío a un duelo."
},
{
  "200 years": "Se refiere a un campeón cuyo kit tiene una cantidad complicada o superflua
},
{
  "AA": "Ataque básico, que recibe el sobrenombre de Ataque automático ya que el campeón pu
},
{
  "Auto": "Ataque básico, que recibe el sobrenombre de Ataque automático ya que el campeón
},
```

Ilustración 14: Captura con la estructura final de los términos y conceptos [Recurso propio]

Para finalizar el proceso de implementación del chatbot en la web, queda la comunicación entre esta y el servicio ofrecido por Voiceflow, la cual funciona mediante llamadas a la API de la compañía, incluyendo, en el caso de que lo haya introducido el usuario, el mensaje con la pregunta en una petición POST (Voiceflow, 2024).

```
const interact = (request) =>
  // call the voiceflow api with the user's name & request, get back a response
  fetch(
    `https://general-runtime.voiceflow.com/state/user/${nombre_usuario_temp}/interact`,
    {
      method: "POST",
      headers: {
        accept: "application/json",
        Authorization: process.env.NEXT_PUBLIC_VOICEFLOW_API_KEY,
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        action: { type: "launch" },
        config: {
          tts: false,
          stripSSML: true,
          stopAll: true,
          excludeTypes: ["block", "debug", "flow"],
        },
        state: { variables: { x_var: 2 } },
      }),
    }
  )
    .then((res) => res.json())
    .then((trace) => {
      console.log("API RESPONSE BODY:", trace);
      trace.forEach((trace) => {
        if (trace.type === "speak" || trace.type === "text") {
          setMsg((lastMsg) => [...lastMsg, { id: nextid++, msg: trace.payload.message }])
        }
      });
    });
```

Ilustración 15: Captura con la llamada a la API de Voiceflow para iniciar una conversación [Recurso propio]

En esta figura 15 podemos ver la primera parte de la implementación de las llamadas a la API, en concreto aquella que se llamará cada vez que se entre a la página o se refresque.

Como se puede apreciar, se hace uso de la “built-in function” de JS “fetch”, para que se ocupe de realizar la petición HTTP POST a la API, con una variable ${nombre_usuario_temp}, que es interesante ya que el servicio crea una conversación por cada userID (la variable ${nombre_usuario_temp} en este caso) con la información asociada a él.

Por otro lado, en el cuerpo de la petición declaramos tres objetos como parte de la configuración del servicio al que queremos acceder:

- Action: este objeto es el que declara a que funcionalidad se pretende acceder dentro del “endpoint” de Interact al que estamos consultando, en este caso “launch” ya que se pretende comenzar una conversación
- Config: estas son declaraciones opcionales con las que estructurar la respuesta del agente IA
    - tts: “text to speech”, o en el castellano, texto a habla, define si está activada el envio de un archivo mp3 con la respuesta en reproducida por el chatbot
    - stripSSML: opción con la que activas o desactivas la posibilidad de utilizar un sintetizador para mejorar la pronunciación del texto por el chatbot
    - stopALL: stopALL prohibe la llegada de “traces” -las respuestas generadas- que no tengan la estructura marcada por el servicio
    - excludeTypes: declara los tipos de respuestas que no se quieren recibir en la parte que realiza la petición
- State: declara el número de variables para la petición

Por último, una vez se ha realizado la petición y se ha recibido una respuesta, se convierte el texto en un objeto JSON y se itera para encontrar el tipo que contiene la respuesta generada que estamos esperando, en concreto el objeto “text”. Seguidamente, se añade a la lista de mensajes de la conversación, que en este caso será el único.

```
async function callAPIVoiceFlowInteract() {
    // call the voiceflow api with the user's name & request, get back a response
    await fetch(
      `https://general-runtime.voiceflow.com/state/user/${nombre_usuario_temp}/interact`,
      {
        method: "POST",
        headers: {
          Authorization: process.env.NEXT_PUBLIC_VOICEFLOW_API_KEY,
          "Content-Type": "application/json",
          accept: "application/json",
          versionID:process.env.NEXT_PUBLIC_VOICEFLOW_VERSIONID,
        },
        body: JSON.stringify({ action: { type: "text", payload: document.getElementById("chat").value} }),
      }
    )
      .then((res) => res.json())
      .then((trace) => {
        for(var output of trace)
        {
          if(output["type"] == "text")
          {
            setMsg((lastMsg) => [...lastMsg, {id: nextid++, msg:output.payload.message}]);
          }
        }

      })
      .catch((error) => console.error(error));

      document.getElementById("chat").value = ""
}
```

Ilustración 16: Captura de la función con la que enviar los mensajes del usuario para mantener la conversación [Recurso propio]

Si el usuario presenta una pregunta para el agente inteligente, la petición cambia ligeramente. Como se puede observar en la figura 16, la cabecera cambia mínimamente con la introducción del “versionID”, necesaria para poder acceder al modelo que hayamos creado en el servicio de Voiceflow.

Otra parte de la declaración anterior que cambia acorde a la necesidad de la petición es el cuerpo de esta, en el que cambiamos el objeto acción para configurar el tipo “text”, que declara la necesidad de una respuesta generada por la IA, así como un “payload” que contendrá el mensaje con la pregunta que haya introducido el usuario.

Para terminar la petición, cuando se recibe se procede de la misma manera que con la respuesta de la anterior figura 16 “launch”, transformando, iterando y guardando el mensaje del objeto JSON recibido, además de borrar el campo de texto en el que se ha introducido el mensaje para prepararlo para el siguiente.

3.3 Implementación de la aplicación web

Por otro lado, la implementación de la aplicación web con la que el usuario interactúa y obtiene la información se compone de dos partes bien diferenciadas: el “front-end” -la interfaz de usuario- y el “back-end” -el servidor que hace posible las funcionalidades de la aplicación-.

Comenzaremos desarrollando la implementación de la interfaz de usuario, aprovechando que es lo primero que se realizó en este proyecto, con el objetivo de tener una base fundacional sobre la que poder implementar y probar, como usuario final, las funcionalidades que se llevarían a cabo en el “back-end”.

En primer lugar, se optó por desarrollar lo que sería la página inicial de la aplicación, con la que interactuaría el usuario en primera instancia para busca el nombre de invocador asociado a su cuenta y poder obtener la información relevante asociada a su cuenta del League of Legends

En el caso de la implementación, esta idea inicial se ha mantenido casi inalterada, añadiendo únicamente la funcionalidad de poder cambiar a placer el servidor en el que se busca el nombre de invocador -el juego tiene centros desde los que ofrece el servicio en cada una de las principales zonas del mundo, como en Norteamérica, Europa este o Rusia, entre otros-.

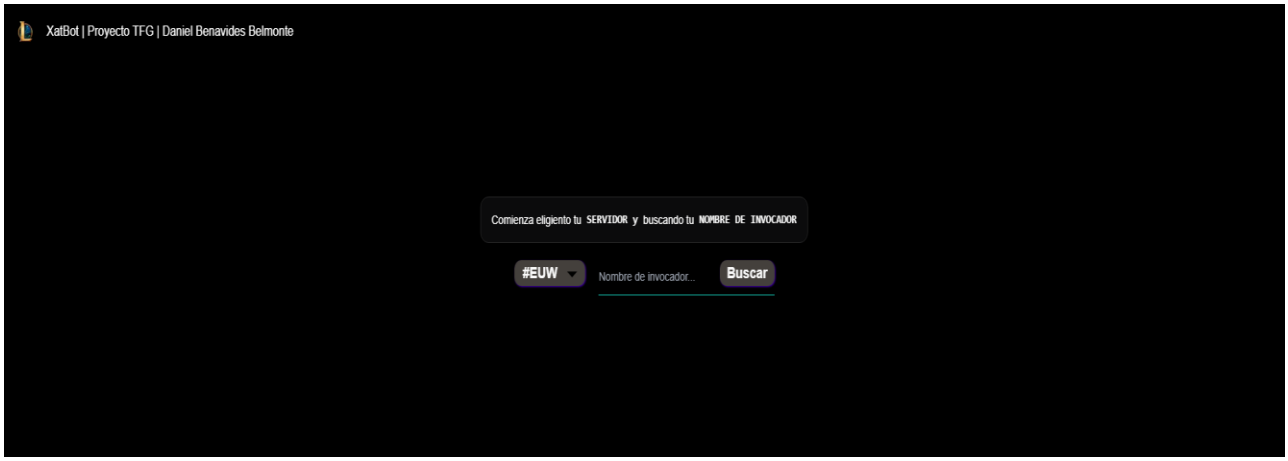


Ilustración 17: “Landing page” implementada [Recurso propio]

El producto final es el que se puede ver en la figura 17, en esta se puede apreciar varias pistas -en la burbuja de encima del cuadro de texto y en el mismo cuadro de texto- de que debe introducir el nombre de invocador. Además, se ha añadido el contraste de color de ambos botones en pantalla para resaltarlos y aportar información visual, juntamente con su texto, para destacar sus funcionalidades

Seguidamente se implementó la que sería la página del chat, donde el usuario pasaría la gran mayoría del tiempo que utilizase la aplicación web. En este caso se ha tenido que balancear el peso, por importancia y uso, de cada una de las partes que componen dicha aplicación.

El diseño inicial de la página que contiene el chat ha sufrido varios cambios en el proceso de implementación en la web, ya que en el transcurso del desarrollo se realizó varias veces el ciclo de diseño e implementación de funcionalidades para esta sección de la aplicación, con nuevas propuestas que añadiesen valor al producto, otorgando al usuario más información o cambiando la presentación de los componentes ya desarrollados.

Conceptualmente se dividió la interfaz en dos partes diferenciadas:

- Componente de usuario: esta sección se dedicaría íntegramente a mostrar la información recogida mediante el nombre de invocador que se ha introducido, así como para el componente de sugerencias con diferentes componentes del videojuego
- Componente de chat: esta otra sección contendría las utilidades necesarias para poder mantener una conversación con el chat

En primera instancia desgranaremos la sección de usuario, que en un inicio tenía como único fin personalizar de manera adecuada la plataforma para ofrecer un servicio más personal. Esta estaba compuesta por el icono de invocador que el usuario tuviese en el juego, seguido de un componente con la estadística de victorias-derrotas de este en las últimas 20 partidas y los tres campeones más usados por el jugador.

En una iteración de la idea inicial se decidió añadir un nuevo componente con varias sugerencias relacionadas con el juego para dar una experiencia al usuario más liviana y que hiciese la interacción con el chatbot más intuitiva, ya que se llegó a la conclusión que entrar a un chatbot sin mucha idea de lo que preguntar puede llegar puede llegar a ser abrumador, con lo que se incluyeron sugerencias sobre:

- Campeones
- Objetos
- Términos y conceptos
- Clases de campeones

Estos conceptos están implementados como botones que, una vez pulsados, generan un “prompt”, confeccionado previamente en base a las pruebas que se realizaron en la implementación del chatbot para obtener la cantidad y calidad óptima de la información del tema que se quisiese consultar. Estos cuentan con imágenes relacionadas con el tema de la sección en la que se encuentren -si es una sugerencia de campeones, el icono de este, así como si es una sugerencia de objetos aparece el icono correspondiente-. La sección de conceptos es una excepción ya que no existen iconos para cada uno de ellos.

Una vez el usuario considere que la entrada es válida y pulse el botón para enviar el mensaje, el cuadro de texto donde había aparecido el “prompt” se vaciará dando paso a la siguiente cuestión que este quiera plantear.

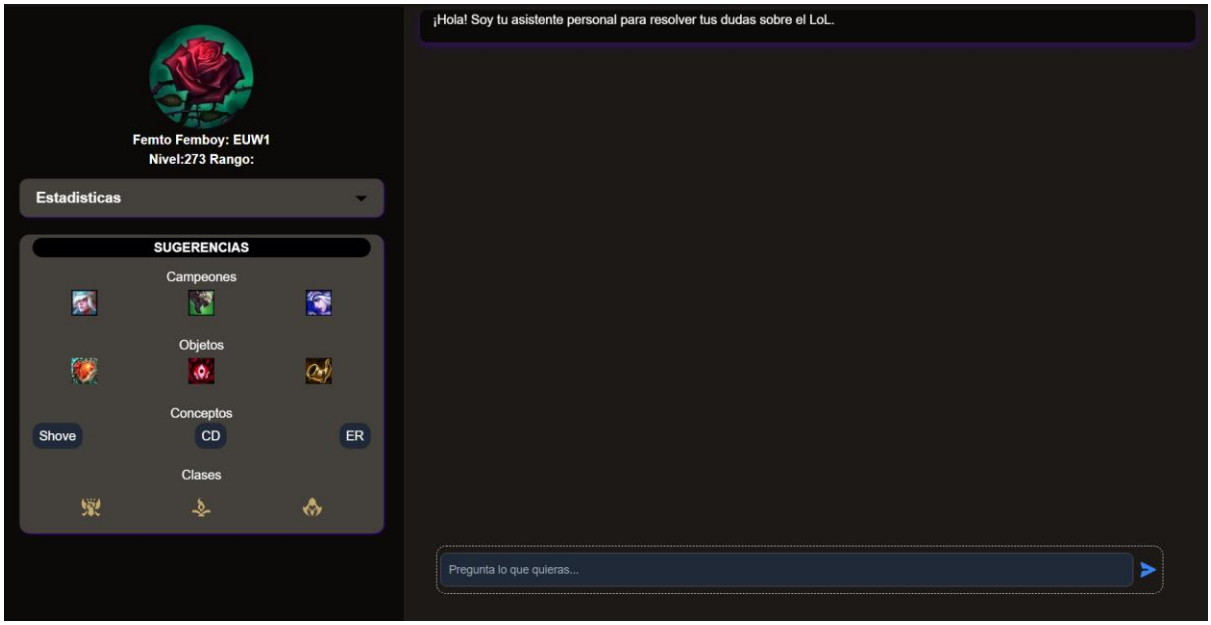


Ilustración 18: Interfaz de usuario de la página del chatbot [Recurso propio]

Por otra parte, la introducción de este último componente para añadir la funcionalidad de las sugerencias surgió un problema de espacio en la sección del usuario. Por lo anterior, se decidió editar el componente de estadísticas para que formase parte de un botón con un texto que informase sobre que sección contenía y al clicarlo desencadenase una animación que desplegase o retrajese la sección al completo.

4. Pruebas

En el apartado de pruebas se revisará el proceso de escrutinio al que se ha sometido a los endpoints que conforman el servidor de la aplicación web, así como a la base de

conocimiento con la que se ha entrenado al chatbot. De igual manera se explicará los resultados que se esperaban y se pondrán ejemplos representados en figuras de respuestas exitosas.

4.1. Comprobación de endpoints

Para asegurar de cierta manera la calidad y usabilidad de la solución tecnológica desarrollada se prepararon y realizaron varias pruebas que pusiesen a prueba el funcionamiento de estos componentes del trabajo.

El primer punto que sobre el que se realizaron pruebas haciendo uso de la aplicación para el diseño, desarrollo y prueba de APIs “Postman”. Se ha escogido esta plataforma por la razón de que otorga una herramienta que facilita probar el flujo de datos de una API, de una manera casi instantánea y sin la necesidad de escribir código adicional.

Utilizando el cliente de “Postman” para hacer llamadas a los “endpoints” del servidor se ha podido probar la accesibilidad de estos, así como comprobar la veracidad de las respuestas de las diferentes llamadas a la API del trabajo.

```
""" Ruta de api que devuelve un dict() con la ···
@app.route('/api/info_usr', methods=['GET'])
async def info_usr():···

""" Ruta de api que devuelve un dict() con la ···
@app.route('/api/info_champs', methods=['GET'])
async def info_champs():···

""" Ruta de api que devuelve un dict() con un bool según si el ···
@app.route('/esUsuario', methods=['GET'])
async def existe_usr():···

""" Ruta de api que devuelve un dict() con un string conteniendo el nombre···
@app.route('/sugerencia', methods=['GET'])
async def sugerencia_rng():···
```

Ilustración 19: Captura con los diferentes endpoints con los que cuenta el servidor [Recurso propio]

Las rutas estáticas que podemos ver en la figura 24 son las que conforman los “endpoints” de nuestro servidor y se ocupan de parte de las funcionalidades de la aplicación web, principalmente aquellas que tienen que ver con la información del usuario.

Estas están divididas dependiendo si se hace un preprocesado de los datos para devolver un objeto curado con la información necesaria para la página o aquellas que son llamadas simples, denotadas por el directorio /api o directamente en la raíz respectivamente

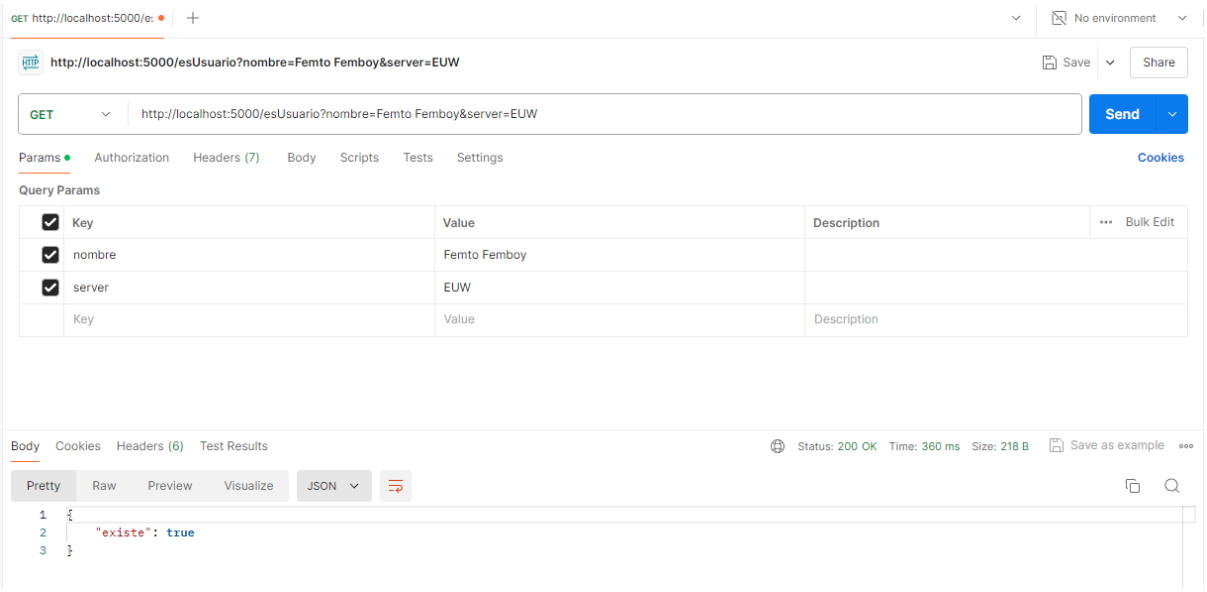


Ilustración 20: Captura de la prueba de la ruta "/esUsuario" [Recurso propio]

Gracias a la intuitiva interfaz de la aplicación de "Postman" podemos hacer pruebas en segundos y comprobar la usabilidad del servidor. En la figura 20 podemos ver como se realiza la prueba a una ruta estática como puede ser /esUsuario -la comprobación de la existencia de un usuario en las bases de datos de Riot Games-.

Estas pruebas se componen de tres secciones diferenciadas:

- Dirección: en este campo se introduce la ruta que se quiere comprobar y, en caso de necesitarlos, los parámetros -en este caso /esUsuario?nombre={}&server{}-
- Parámetros: en esta tabla podemos ver representados, o en su defecto podemos introducir, los parámetros u objetos en el cuerpo de la petición principalmente
- Respuesta: en esta última sección podemos ver un desglose de la totalidad de la respuesta, cuya pestaña principal es el cuerpo con el objeto JSON resultante de la llamada

Se ha realizado el procedimiento previo para cada una de las diferentes direcciones que abarca el "back-end", con la finalidad de comprobar que son capaces de comunicarse

ambas partes de la aplicación y que la información que se comunica es la que se busca obtener:

- /sugerencia: con la llamada desde el Postman se pretendía que se recibiesen los diccionarios con los campeones, objetos y clases con sus respectivas direcciones para los iconos, así como los conceptos. También se pretendía comprobar que ninguno de estos tuviera valores repetidos. Un ejemplo exitoso sería la figura 21.

Ilustración 21: Captura de la respuesta del servidor a la llamada /sugerencias [Recurso propio]

- /api/info_usr: con la llamada a esta dirección se pretendía comprobar que se podía recoger la información correcta de la cuenta que haya introducido el usuario en la “landing page”, como el id, nombre o icono de perfil, entre otros. Un ejemplo exitoso sería la figura 22.

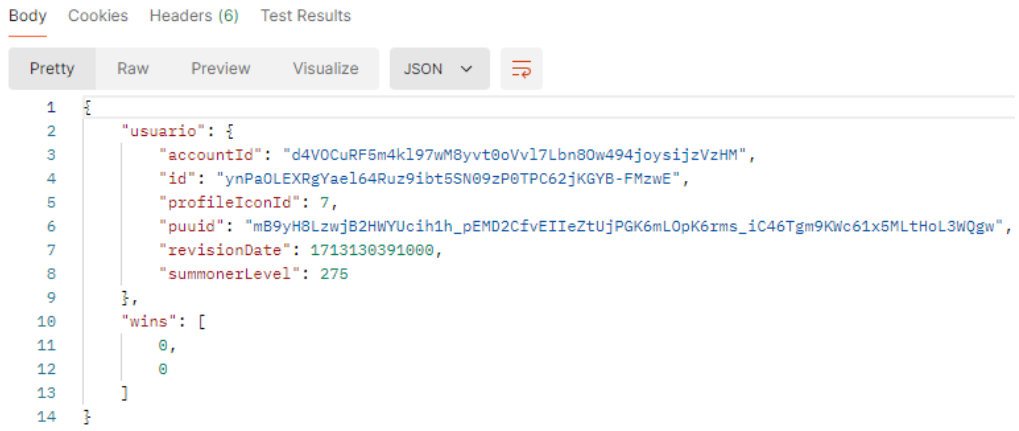


Ilustración 22. Captura de la respuesta del servidor a la llamada /api/info_usr [Recurso propio][7]

- /api/info_champs: con la llamada a este punto del servidor se pretendía recoger la información correcta de los 3 campeones más jugados por el usuario de la aplicación, así como las victorias en las últimas 20 partidas. Un ejemplo exitoso sería la figura 23.

[7] En este ejemplo hay un total de 0 victorias y derrotas por el tiempo que se lleva sin jugar en la cuenta

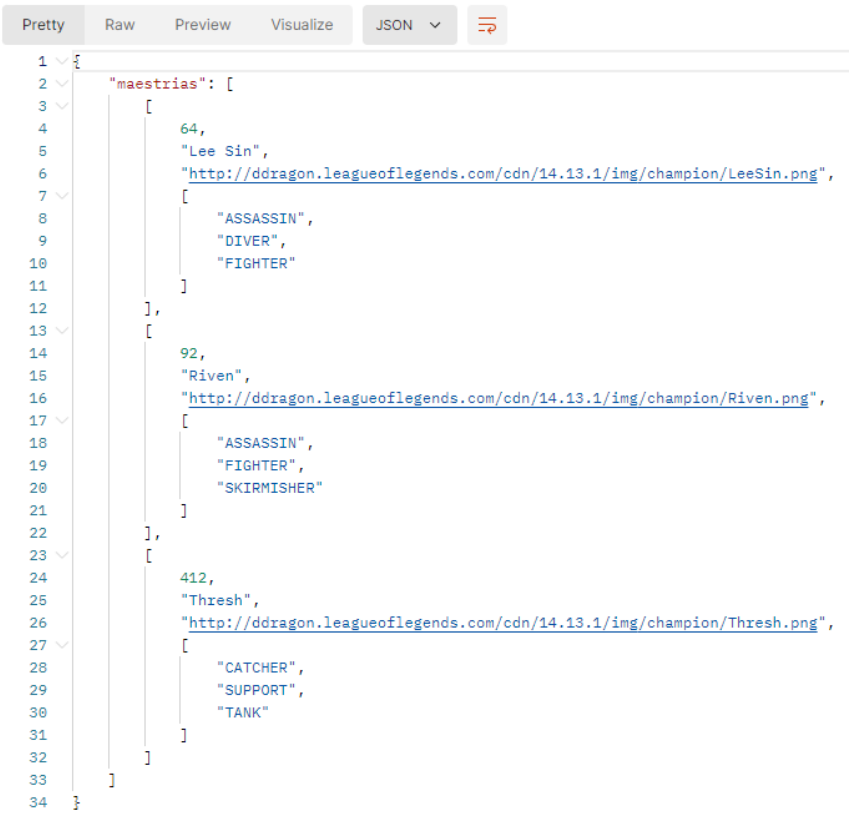


Ilustración 23: Captura de la respuesta del servidor a la llamada /api/info_champs [Recurso propio]

4.2. Revisión del conocimiento del chatbot

Por otro lado, se hicieron una batería de preguntas para probar el funcionamiento de la generación de texto acorde al tema que se plantea por parte del chatbot. Para esta tarea se utilizó la herramienta de “preview” implementada dentro de la interfaz de la plataforma de “Voiceflow”.

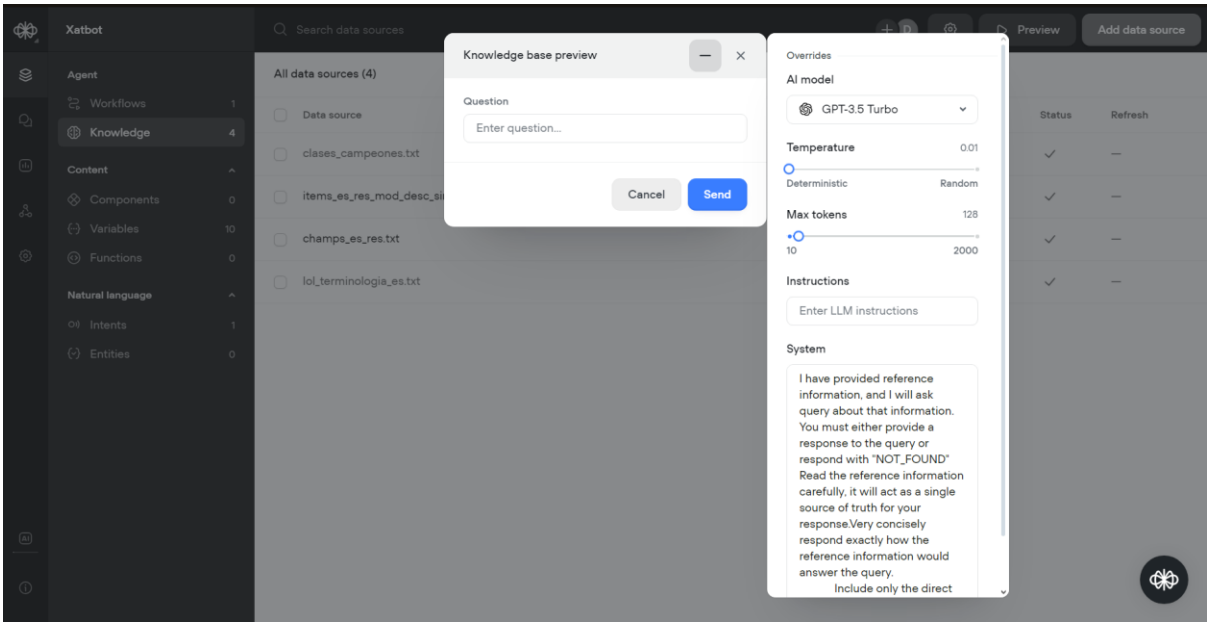


Ilustración 24: Captura de la herramienta preview de la sección Knowledge Base

La manera de proceder para esta prueba de usabilidad fue como está listada a continuación:

- Redacción de una serie de preguntas para cada uno de los temas en los que se ha entrenado al chatbot
- Consulta con la herramienta de “preview” al chatbot
- Evaluación de la veracidad y sentido del texto generado y comparación de este con la información de entrenamiento

Para la parte de campeones se ha usado “prompts” como “¿Quién es {Thresh}[8]?” o “Explícame todo lo que sepas sobre {Thresh}” y se ha utilizado ChatGPT para crear variaciones de estos y probar las diferentes respuestas con un acercamiento ligeramente diferente.

Para las siguientes secciones se procedió de la misma manera con preguntas como las siguientes:

- “¿Qué me da la {Sanguinaria}?” y variaciones generadas por IA para probar el conocimiento sobre los objetos

[8] Las palabras entre {} son aquellas que variaron a la hora de hacer las pruebas, utilizando diferentes campeones, objetos, terminología o conceptos y clases para cada uno de los “prompts”

- “¿Qué significa {KYS}?” para comprobar las respuestas sobre la terminología y los conceptos
- “¿Qué es un {mago} en el LoL?” para probar el conocimiento sobre las clases en las que se categorizan los campeones

En la primera tanda de pruebas se pudo observar un mal funcionamiento del chatbot, con respuestas directamente erróneas o con alucinaciones de la IA, en la que confundía información y la combinaba en el texto generado como respuesta.

Una vez reestructurada la información para que quedase como el producto final que se puede observar en las figuras 12, 13 y 14 de la sección de “implementación del chatbot”, las respuestas de este agente conversacional mejoraron hasta el punto de no encontrar error en 50 preguntas que se le plantearon sobre los diferentes temas en los que se le había entrrenado.

5. Conclusiones

Al finalizar este proyecto de final de grado se ha conseguido como resultado del trabajo una aplicación web que permite a un usuario interactuar con un agente inteligente y nutrirse de información esencial sobre el videojuego LoL, cumpliendo los objetivos y requisitos marcados en la fase de concepción.

Se ha logrado desarrollar una primera versión de la aplicación web en base a NextJS la cual funciona de manera intuitiva y refuerza el uso del chatbot en el usuario. El “back-end” desarrollado en Flask permite conectar al usuario con la información que desea gracias a la definición de rutas estáticas y el procesado de datos en Python, así como a las respuestas generadas por IA brindadas por la plataforma de Voiceflow que hospeda y se encarga de la ejecución del entrenamiento de nuestro chatbot conversacional.

En cuanto a propuesta de mejora, podría plantearse la implementación de la recolección de datos de las partidas de los usuarios en tiempo real, con el objetivo de aconsejar al usuario en partidas y realizar comentarios sobre las jugadas realizadas. Adicionalmente podría ampliarse la base de conocimiento con el objetivo de responder preguntas como los objetos que debería comprar uno en la partida o el campeón que debería escoger en un encuentro.

Por otro lado, podría ser interesante implementar que la funcionalidad del bot fuese compatible con comandos de voz, lo cual es una opción en la plataforma de Voiceflow pero no se consideró suficientemente relevante como para introducir en el proyecto.

En cuanto a la usabilidad de la aplicación, ha de decirse que ha sido una lástima no poder brindar la aplicación a posibles usuarios debido a la incapacidad de montar dicha aplicación en un servicio de hosting. Esto hubiese aportado una información bastante relevante para poder mejorar la experiencia de usuario.

Para acabar con las conclusiones, el diseño e implementación de la aplicación web y la IA generativa ha supuesto de igual manera un reto personal y una ampliación a los conocimientos adquiridos en la formación recibida a lo largo del Grado de Tecnologías Interactivas.

En lo referente a la implementación de la plataforma web, se ha ampliado mi conocimiento dado que, en cuanto al desarrollo del “front-end”, se ha utilizado el “framework” de React -Nextjs- el cual introducía nuevos conceptos como los componentes o el SSR, herramientas las cuales hacen mas sencillas y ágiles para el desarrollo, en contraste con el Javascript base que se ha aprendido en el grado.

Por otro lado, se ha agilizado de igual manera el desarrollo del “back-end” utilizando un lenguaje y “framework” que no se han empleado para esta tarea en las clases de la titulación, utilizando en esta Nodejs y Express.

Por otro lado, el entrenamiento del chatbot ha sido indispensable para implementar una solución competente, lo cual no se ha visto en clase y he tenido que leer documentación sobre el tema -tanto en el preprocesado de la información como en el formateo de esta para el entrenamiento-.

Agradecimientos

Como inciso final, me gustaría dedicar una última sección a los agradecimientos a las personas que han estado presentes en el proceso creativo y de desarrollo.

Primero, agradezco a Juan Miguel Arbeloa, Victor Sanchez y Carles Canut por el apoyo, ayuda y guía tanto en el proceso de concepción de la idea, dando ideas y recursos, como en el proceso de desarrollo aconsejando.

En segunda instancia, me gustaría agradecer a mis padres Mirian Belmonte y Julio Benavides por el apoyo y financiación durante todos estos años estudiando el grado, ya que sin ellos este momento no podría haber sido posible

Finalmente, aprovecho para agradecer a mi mujer por el apoyo incondicional, soportando monólogos aburridos sobre las tecnologías y procesos que conforman esta aplicación, y la ayuda en la edición y corrección de este trabajo en sus momentos libres compaginándolo con sus responsabilidades.

Bibliografía

Adhithya, B. R. (2023). The Impact, Advancements and Applications. *Research gate*, 1-8.

Anthropic. (22 de Julio de 2023). *Claude AI*. Obtenido de https://claude.ai

Backseat GG. (2024). *Backseat AI x Tyler1*. Obtenido de https://www.backseat.gg

Epic Games. (2024). *Most played* . Obtenido de https://store.epicgames.com/en-US/collection/most-played

Esports earnings. (20 de junio de 2011). *Esports earnings - Summer 2011 LoL World championship*. Obtenido de https://www.esportsearnings.com/tournaments/2673-dreamhack-summer-2011-lol-world-championship

Esports Earnings. (2016). *DreamHack Summer 2011 - League of Legends Season One Championship*. Obtenido de https://www.esportsearnings.com/tournaments/2673-dreamhack-summer-2011-lol-world-championship

Filipović, A. (2023). The Role of Artificial Intelligence in Video Game Developmen. *Kultura Polisa*, 50-67.

Gackenheimer, C. (2015). *Introduction to React.* Apress.

iann838. (2024). *Pulsefire Reference*. Obtenido de https://pulsefire.iann838.com/reference/

League of Legends Wiki. (2024). *Terminology (League of Legends)*. Obtenido de https://leagueoflegends.fandom.com/wiki/Terminology_(League_of_Legends)

Lehnert, K. W. (2022). The booming eSports market: a field day for fans. *Journal of Buisness*, 2-3.

Liquipedia. (2024). *Liquipedia - B Tier Tournaments*. Obtenido de https://liquipedia.net/leagueoflegends/B-Tier_Tournaments

Liquipedia. (2024). *Liquipedia - C Tier tournaments*. Obtenido de https://liquipedia.net/leagueoflegends/LVP/Superliga/2nd_Division/2024/Spring

Liquipedia. (2024). *Liquipedia - S Tier tournaments*. Obtenido de https://liquipedia.net/leagueoflegends/World_Championship/2024

Meta. (23 de Febrero de 2023). *Llama Meta*. Obtenido de https://llama.meta.com

Microsoft. (Febrero de 2023). *Bing*. Obtenido de https://www.bing.com/chat

Microsoft. (junio de 2024). *Most played games*. Obtenido de https://www.microsoft.com/en-us/store/most-played/games/xbox

Nalli, J. (2019). *Effect of esports and future development of esports.* Obtenido de https://www.theseus.fi/bitstream/handle/10024/168883/Nalli_Joni.pdf.pdf?sequence=2

OP.GG. (2024). *OP.GG*. Obtenido de https://www.op.gg

OpenAI. (22 de Noviembre de 2022). *ChatGPT OpenAI*. Obtenido de https://chatgpt.com

Riot Games. (2021). *Developer portal - League of Legends*. Obtenido de https://developer.riotgames.com/docs/lol

Shirazi, N. H. (2021). Social Bots and the Spread of Disinformation in Social Media: The Challenges of Artificial Intelligence. *British journal of management*.

Valve. (junio de 2024). *Steam Charts*. Obtenido de https://steamcharts.com/top

Vardal, O. B. (2022). Mind the gap: Distributed practice enhances performance in a MOBA game. *Plosone*.

Vercel. (2024). *Nextjs Docs*. Obtenido de https://nextjs.org/docs

Voiceflow. (2024). *Knowledge Base*. Obtenido de https://learn.voiceflow.com/hc/en-us/articles/22213924437517-Knowledge-Base

Voiceflow. (2024). *Voiceflow API reference*. Obtenido de https://developer.voiceflow.com/reference/stateinteract-1

Voiceflow. (s.f.). *What is an interaction model in conversational AI?* Obtenido de https://www.voiceflow.com/blog/what-is-an-interaction-model-in-conversational-ai

Wikipedia. (2024). *Eliza*. Obtenido de https://es.wikipedia.org/wiki/ELIZA

Williams, L. J. (8 de marzo de 2024). *Games Hub*. Obtenido de https://www.gameshub.com/news/news/ea-ceo-andrew-wilson-generative-ai-2637802/