TRABAJOS FIN DE GRADO

Universidad de Jaén

Escuela Politécnica Superior de Jaén

Selección inteligente de personajes en League of Legends

Autor: Álvaro Martínez Morales

Grado: Ingeniería Informática

Director: Álvaro Labella Romero
Departamento del director: Informática

Fecha: 08/09/2024

CREA

UNIVERSIDAD DE JAÉN

D./Dª Álvaro Labella Romero y D./Dª Diego García Zamora, tutor(es) del Trabajo Fin de Grado titulado: **Selección inteligente de personajes en League of Legends**, que presenta Álvaro Martínez Morales, autoriza(n) su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Septiembre de 2024

El estudiante Los tutores

Álvaro Martínez Morales Álvaro Labella Romero Diego García Zamora

Agradecimientos

Quiero expresar mi agradecimiento a mis padres, familia y pareja, quienes han sido y continúan siendo un pilar fundamental en mi vida. Su apoyo y paciencia, incluso en los momentos más difíciles, han sido esenciales para poder seguir adelante.

Dedico este proyecto a todos aquellos que creyeron en mí, porque a base de trabajo y constancia he sido capaz de culminar esta etapa.

Por último, me gustaría agradecer a los profesores de la universidad. Su enseñanza ha allanado el camino que he recorrido junto a mis compañeros durante todos estos años.

Tabla de contenidos

Lista de figuras

Lista de tablas

Lista de listados de código

Capítulo 1

INTRODUCCIÓN

1.1. Planteamiento del problema

En la era digital actual, el mundo de los videojuegos se ha convertido en un fenómeno mundial que une a distintos jugadores de todos los rincones del planeta. Los videojuegos han evolucionado de simples pasatiempos a competiciones con una comunidad apasionada y dedicada [14]. Ante este panorama, League of Legends (LoL) destaca como una de las cabezas en el sector de los deportes electrónicos *(e-Sports)*, logrando así una apreciación global y unos registros que lo convierten en uno de los juegos con más jugadores registrados del mundo.

Desarrollado por Riot Games, League of Legends fue lanzado al mercado en octubre de 2009. Desde entonces, ha mantenido su popularidad y ha crecido exponencialmente. Se trata de un juego gratuito, lo que facilita su accesibilidad a una amplia audiencia. Su modelo de negocio se basa en la compra de elementos estéticos, conocidos como *skins*, que permiten la personalización de los personajes, lo cual no afecta a la jugabilidad pero agrega un nivel de personalización muy apreciado por los jugadores.

League of Legends es un juego en línea que pertenece al género MOBA *(Multiplayer Online Battle Arena)* [15] en el que dos equipos de cinco jugadores compiten por destruir el nexo enemigo, situado en la base adversaria, mientras defienden su propio nexo. Este género de juego combina elementos de estrategia en tiempo real, habilidad individual y cooperación en equipo, requiriendo que los jugadores trabajen juntos para lograr objetivos y derrotar al equipo contrario. Cada jugador controla un campeón con habilidades únicas y la selección de estos campeones es crucial para el desarrollo de

estrategias efectivas durante la partida.

Ante este escenario competitivo, la selección de personajes va más allá de la preferencia individual; es un componente crítico que influye directamente en el desarrollo y resultado de cada partida. La capacidad de seleccionar sabiamente un campeón no solo demuestra el entendimiento profundo de las habilidades individuales de cada personaje, sino que también se convierte en una pieza clave para la construcción de estrategias efectivas y la sinergia del equipo. La elección de campeones debe tener en cuenta factores como la composición del equipo propio y enemigo, las fortalezas y debilidades de cada campeón, y como se complementan entre sí para maximizar las posibilidades de éxito.

1.2. Objetivos

Este apartado se dedica a presentar los objetivos del proyecto, dividiéndolos en un objetivo general y varios específicos. Definir claramente estos objetivos es fundamental para orientar el desarrollo del proyecto y asegurar que se abordan todas las áreas de manera efectiva. Los objetivos proporcionan una hoja de ruta estructurada que guían cada fase del trabajo, desde la concepción inicial hasta la implementación y evaluación final.

1.2.1. Objetivo general

El objetivo general del proyecto es desarrollar una aplicación web que permita a los usuarios seleccionar campeones de League of Legends de manera estratégica, informada y optimizada. Esta herramienta está diseñada para facilitar la toma de decisiones durante la fase de selección de campeones, proporcionando datos detallados y actualizados sobre diversos aspectos del juego.

1.2.2. Objetivos específicos

Mediante la consecución de los objetivos específicos aseguraremos el éxito del objetivo general. Estos objetivos detallan los pasos necesarios y las metas intermedias que guiarán el desarrollo del proyecto. Los objetivos específicos son:

Investigación previa

Antes de comenzar la implementación, es necesario realizar una investigación exhaustiva sobre los temas que se abordarán en el proyecto. Esto implica revisar bibliografía, tutoriales, articulos y otros recursos. La investigación se centrará en temas como el web scraping, los sistemas de toma de decisiones, lenguajes de programación y tecnologías que se utilizarán en la implementación de la aplicación web. Esta fase es importante para construir una base sólida de conocimiento y asegurar que las decisiones técnicas sean correctas.

Adquisición de datos

Para poder mostrar información en la página web, primero es necesario obtener los datos relevantes. Utilizaremos técnicas de web scraping para recolectar datos de fuentes confiables como la página oficial de League of Legends y OP.GG. Una vez obtenidos, los datos se transforman a formato JSON para organizarlos de forma estructurada y facilitar su posterior utilización.

Creación de la aplicación web

Desarrollar una aplicación web con una interfaz de usuario intuitiva y accesible es uno de los objetivos clave. La aplicación debe ser fácil de usar y permitir a los usuarios localizar la información de manera rápida. Esto implica diseñar y desarrollar un frontend atractivo y funcional utilizando tecnologías como HTML, CSS y JavaScript.

Implementación de las funcionalidades y herramientas

Añadir las funcionalidades necesarias a la aplicación web para que cumpla con el objetivo general. Incluye la implementación de herramientas específicas como el listado de campeones, la búsqueda de campeones, la tier list y la herramienta de selección inteligente. La integración de estas herramientas debe ser fluida y coherente, asegurando que todas las partes de la aplicación funcionen en conjunto.

Análisis de los resultados obtenidos

Por último, tenemos como objetivo la evaluación de los resultados obtenidos a lo largo del proyecto. Incluye la realización de un análisis detallado de las funcionalidades implementadas, comentando las limitaciones y ventajas de la aplicación. Se comentarán los aspectos que han funcionado correctamente al igual que las posibles mejoras del proyecto para futuras iteraciones.

1.3. Estructura del proyecto

En esta sección se explica brevemente la estructura del proyecto, la cual nos da una visión general del contenido del documento.

- **Introducción**: La introducción proporciona una visión general del problema que se abordará a lo largo del proyecto. En este apartado se explica la motivación detrás de la realización del proyecto y cómo se va a resolver. Además, se incluye un esquema con los capítulos del trabajo y una breve descripción de cada uno de ellos.
- **Antecedentes**: En este capítulo se exponen todos los antecedentes necesarios, así como las técnicas y metodologías que se emplearán para la resolución del problema. Se presentan estudios previos, teorías relevantes y trabajos relacionados que fundamentan el proyecto.
- **Diseño inicial**: Este capítulo aborda el diseño inicial de la aplicación. Se detalla la estructura de la aplicación web, técnicas de web scraping utilizadas para extraer información relevante sobre campeones, los casos de uso y diagramas de secuencia, proporcionando una visión clara de cómo los usuarios interactúan con el sistema.
- **Desarrollo de la aplicación web**: Este capítulo describe todos los aspectos relacionados con la obtención, preparación y uso de los datos necesarios para alcanzar los resultados del proyecto. Se detallan los materiales utilizados y los métodos aplicados para garantizar la validez de los datos recopilados.
- **Conclusiones**: Por útlimo, este capítulo presenta un análisis de los resultados obtenidos, explicación detallada del estudio realizado a lo largo del proyecto, y posibles mejoras y recomendaciones para futuros trabajos. También se ofrecen conclusiones finales sobre el alcance y el éxito del proyecto.

1.4. Planificación temporal

A continuación, se presenta la planificación temporal del proyecto, que detalla las fases y los hitos alcanzados durante su desarrollo.

Tarea	Duración	Fecha de inicio	Descripción
Investigación inicial	3 semanas	20/05/2024	Se realiza una investigación previa sobre los temas que se abordarán
Primeras pruebas de web scraping	2 semanas	10/06/2024	Se llevan a cabo pruebas de web scraping para familiarizarse con la técnica
Diseño de la aplicación web	4 semanas	24/06/2024	Se implementa la aplicación web
Desarrollo de las funcionalidades	4 semanas	8/07/2024	Se implementan las funcionalidades descritas
Implementación de mejoras	1 semana	5/08/2024	Se realizan mejoras en la interfaz y diseño de la aplicación
Pruebas y validaciones	1 semana	12/08/2024	Se realizan múltiples pruebas para comprobar el correcto funcionamiento de la aplicación
Desarrollo de la memoria	3 semanas	19/08/2024	Elaboración de la memoria

Tabla. 1.1: Tabla de hitos para el desarrollo del proyecto

Esta planificación se ha estructurado mediante un diagrama de Gantt, el cual permite visualizar de manera clara las tareas y distribución a lo largo del tiempo. El diagrama facilita el seguimiento del progreso y asegura que cada etapa se complete en los plazos establecidos.

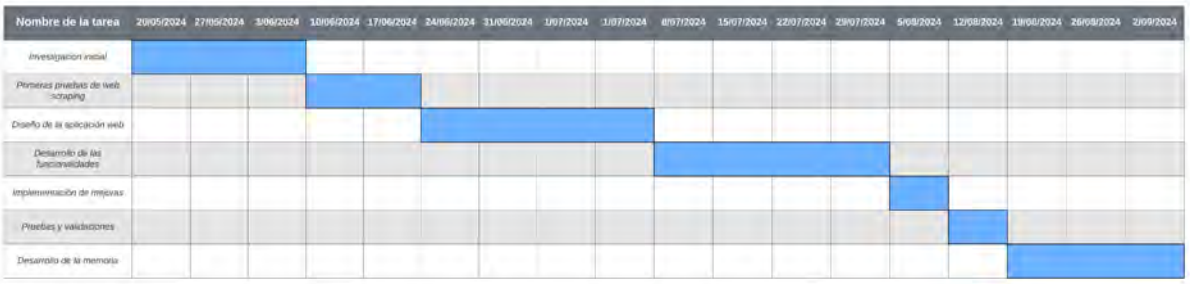

Figura 1.1: Diagrama de Gantt de la planificiación. Elaboración propia

1.5. Presupuesto

Todo proyecto tiene un coste y es fundamental realizar una estimación detallada de los recursos utilizados para poder planificar correctamente su desarrollo. En esta estimación debemos tener en cuenta diferentes factores como el coste de personal, la amortización del equipamiento informático o los costes indirectos, entre otros.

En cuanto al coste de los trabajadores, en este caso específico, estará concentrado en un único trabajador, ya que el desarrollo del proyecto ha sido realizado por una sola persona. Esto simplifica la estructura de costos en lo referente a la mano de obra, aunque es importante destacar el valor del tiempo invertido, calculado en función de la cantidad de semanas trabajadas y el valor promedio de un profesional en este ámbito.

Puesto del trabajador	**Sueldo semanal**	**Duración**	**Sueldo total**
Analista	580€	4 semanas	2.320€
Programador	625€	10 semanas	6.250€

Tabla. 1.2: Tabla de costes de personal

En lo que respecta a los costes de software, se han utilizado herramientas gratuitas, lo que reduce considerablemente el presupuesto en este apartado.

Por otro lado, el hardware es una parte fundamental del proyecto, ya que debe ser adecuado para ejecutar las herramientas necesarias y el entrono de desarrollo. En este sentido, la vida útil de un ordenador portátil se suele estimar en unos 5 años. Los costes de amortización se calcularán en función de este periodo de vida útil, distribuyendo el coste del hardware de manera proporcional.

Hardware	**Coste**	**Vida útil**	**Amortización**
Ordenador MSI GE72 6QD Apache Pro	1.185€	60 meses	19,75€

Tabla. 1.3: Tabla de costes hardware

La duración del proyecto ha sido de aproximadamente 18 semanas, lo que equivale a unos 4 meses y medio. Para facilitar los cálculos relacionados con la amortización, se ha decidido redondear este tiempo a 5 meses de uso estimado.

Finalmente, para calcular los costes indirectos, se ha establecido un porcentaje del 10 % del coste total. De este modo podemos concluir:

Tipo de coste	**Coste**
Costes de personal	8.570€
Costes de hardware	98,75€
Costes indirectos	866,87€
Total	9.535,62€

Tabla. 1.4: Tabla de coste total del proyecto

1.6. Glosario de términos

En esta sección se muestran los acrónimos usados a lo largo del proyecto. Se incluye el acrónimo, su significado en inglés y la traducción española.

Acrónimo	Descripción(Inglés)	Descripción(Español)
LOL	League of Legends	-
MOBA	Multiplayer Online Battle Arena	Campo de batalla multijugador en línea
ARAM	All Random All Mid	Todos al azar todos medio
e-Sport	Electronic sport	Deporte electrónico
TFT	Teamfight Tactics	Tácticas de pelea en equipo
META	Most efficient tactic available	Táctica disponible más eficiente
API	Application Programing Interface	Interfaz de programación de aplicaciones
HTML	HyperText Markup Language	Lenguaje de marcado de hipertexto
CSS	Cascading Style Sheets	Hojas de estilo en cascada
JS	JavaScript	-
UI	User Interface	Interfaz de usuario
JSON	JavaScript Object Notation	Notación de objeto de JavaScript
WWW	World Wide Web	Web
W3C	World Wide Web Consortium	-
DOM	Document Object Model	-
IDE	Integrated development environment	Entorno de desarrollo integrado
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution	-
MCDM	Multi-attribute decision making	Toma de decisión multicriterio
PIS	Ideal Positive Solution	Solución ideal
NIS	Ideal Negative Solution	Solución no ideal

Tabla. 1.5: Tabla de acrónimos usados en el proyecto

Capítulo 2

ANTECEDENTES

En este capítulo se presentan los antecedentes necesarios para contextualizar el proyecto y comprender su relevancia. Se revisan teorías relevantes que sustentan la investigación, proporcionando una base sobre la cual se desarrollará el trabajo. Además, se describen las técnicas y metodologías que se emplearán para la resolución del problema planteado.

A continuación, se explican los conceptos básicos de forma general y se profundiza en los conceptos específicos de las técnicas a desarrollar. Esta revisión no solo clarifica el contexto del proyecto, sino que también resalta las herramientas y enfoques que serán fundamentales para su ejecución.

2.1. League of Legends

En esta sección se presenta League of Legends y los elementos más determinantes para el desarrollo del proyecto. Se ofrece una visión general de su historia, roles, campeones, metajuego y mapa, proporcionando el contexto necesario para comprender el impacto de estos componentes en el proyecto.

2.1.1. Historia

Desde su lanzamiento en 2009, League of Legends ha experimentado una notable evolución que lo ha llevado de ser un juego relativamente desconocido a convertirse en uno de los títulos más influyentes en la industria de los videojuegos y deportes elec-

trónicos. Este crecimiento ha sido impulsado por una combinación de innovaciones constantes y una comunidad apasionada.

La historia de League of Legends se remonta a los orígenes del género MOBA, inspirado en un mod del juego Warcraft III conocido como *Defense of the Ancients* (DotA). Este mod creó un nuevo tipo de juego que combinaba estrategia en tiempo real con elementos de rol y acción, y sirvió como base para el desarrollo de League of Legends. Desde sus primeros días, Riot Games ha demostrado un compromiso continuo con la mejora del juego mediante actualizaciones regulares, la introducción de nuevos campeones y la implementación de nuevos modos de juego.

League of Legends cuenta con dos modos de juego adicionales al clásico: ARAM *(All Random All Mid)* y TFT *(Teamfight Tactics)*. ARAM ofrece una experiencia más rápida y caótica donde todos los jugadores luchan en una única línea con campeones asignados al azar. Este modo de juego es idóneo para aquellos jugadores que buscan partidas más cortas y un cambio de ritmo respecto al juego estándar. Por otro lado, TFT es un juego de estrategia en el que los jugadores compiten utilizando campeones de League of Legends como piezas en un tablero de juego. Este modo de juego ha atraído a una audiencia que disfruta de juegos de estrategia y planificación a largo plazo.

Figura 2.1: Logo de League of Legends [1]

La popularidad de LoL ha ido en aumento a lo largo de los años, catapultando el juego a la cima de la escena de los e-sports [16]. Torneos como el Campeonato Mundial de LoL, también conocido como *Worlds*, han atraído a millones de espectadores de todo el mundo. La final del Mundial de 2023 hizo historia al convertirse en la más vista hasta la actualidad con un pico de 6,4 millones de espectadores simultáneos [17]. Estos eventos no solo destacan por sus producciones de alta calidad, sino también por los sustanciales premios y prestigio asociado con ganar en el escenario mundial.

El éxito de estos torneos ha contribuido significativamente a la consolidación de los deportes electrónicos como una forma reconocida y popular de entretenimiento competitivo. Riot Games ha invertido en la creación de infraestructuras y plataformas para soportar la creciente demanda de e-sports, asegurando que League of Legends permanezca en el centro de esta industria en expansión.

2.1.2. Campeones

En este contexto, es esencial comprender no solo la diversidad de campeones disponibles, sino también los roles que desempeñan y las estrategias asociadas a cada uno. En esta sección, exploraremos la amplia variedad de campeones, sus roles y funciones, porporcionando una visión detallada que permitirá a los jugadores adaptarse a las necesidades del equipo.

Los campeones son personajes jugables en LoL. Actualmente, el juego cuenta con más de 150 campeones, cada uno con habilidades únicas. Las habilidades se dividien en tres básicas, una definitiva y una pasiva. Las habilidades básicas y definitivas se activan durante el juego, mientras que las pasivas proporcionan efectos continuos sin necesidad de activación. Esta diversidad permite a los jugadores experimentar y encontrar campeones que se adapten a su estilo de juego preferido.

Cada campeón pertenece a un rol específico, y cada rol tiene funciones y responsabilidades únicas que son cruciales para el éxito del equipo. Los principales roles son los siguientes:

- **Tanque**: Los tanques son campeones con gran resistencia, mucha vida y capacidad de absorber daño. Su principal función es proteger a su equipo y buscar escaramuzas. Estos campeones suelen iniciar las peleas y absorber el daño enemigo, permitiendo que los compañeros más frágiles pero con más daño hagan su función.
- **Luchador**: Son una combinación de resistencia y daño. Están diseñados para enfrentarse cara a cara con los enemigos, ofreciendo un equilibrio entre capacidad de soportar daño y realizarlo a los enemigos.
- **Asesino**: Los asesinos son especialistas en eliminar rápidamente a los oponentes con menos vida. Suelen tener gran movilidad y un daño elevado, lo que les permite entrar y salir de las peleas rápidamente. Se enfocan en objetivos prioritarios como los tiradores o magos enemigos, buscando eliminarlos antes de que puedan pelear.

- **Tirador**: Son campeones a distancia con poca vida pero que inflingen una gran cantidad de daño físico. Son esenciales en las peleas de equipo debido a su capacidad para causar daño continuo desde la distancia. Los tiradores necesitan protección por parte de su equipo debido a su fragilidad pero pueden decidir el resultado de una pelea con su daño sostenido.

- **Soporte**: Brindan utilidad y apoyo al resto del equipo. Pueden curar, dar escudos a sus aliados y facilitar oportunidades para realizar emboscadas. Su papel es esencial en fase de líneas y las peleas de equipo, ya que ofrecen protección a los campeones más vulnerables.

- **Mago**: Los magos son campeones con un elevado daño mágico a distancia que suelen carecer de movilidad. Utilizan sus habilidades para inflingir grandes cantidades de daño a ráfagas desde la retaguardia.

Figura 2.2: Algunos de los campeones disponibles en League of Legends [2]

2.1.3. Posiciones

La posición que ocupa un campeón durante la partida depende principalmente de sus habilidades y rol. Es importante destacar que, aunque hay roles comunes para cada posición, la flexibilidad e innovación son una poderosa herramienta con la que sorprender al equipo rival. La habilidad para adaptarse y experimentar con diferentes campeones en diversas posiciones puede ser clave para ganar ventaja sobre los oponentes [18]. Las posiciones en el LoL son las siguientes:

- **Top**: El campeón encargado de ir a la calle superior es conocido como el *top laner*. Los roles más comunes en esta posición son los tanques, luchadores y

asesinos, debido a la necesidad de resistencia y capacidad de daño en enfrentamientos prolongados. Los jugadores de esta posición se enfrentan en combates uno contra uno, donde la habilidad individual y el conocimiento del enfrentamiento juegan un papel crucial.

- **Jungla**: Es el campeón encargado de jugar en la jungla. Basa su juego en moverse entre las líneas ayudando a sus compañeros y haciendo objetivos neutrales como el Dragón o Heraldo. Los junglas juegan un papel esencial en el control del mapa y creación de ventajas para su equipo. Este rol requiere un conocimiento de la ruta a seguir en la jungla y una comunicación constante con el resto del equipo.

- **Medio**: El campeón encargado de ir a la línea central, conocido como *mid laner*, juega un papel crucial en el control del centro del mapa y contribuyendo a las emboscadas en otras partes del mismo. Principalmente se juegan magos o asesinos debido a su capacidad para inflingir gran daño. Deben de ser versátiles, capaces de dominar la línea, influir en el resto del mapa a través de rotaciones y ser decisivos en las peleas de equipo.

- **Tirador**: El campeón encargado de ir a la calle inferior, también conocido como *AD Carry* o *ADC*, es fundamentalmente un tirador. Los jugadores de esta posición deben estar preparados para continuos enfrentamientos dos contra dos y para escalar a medida que avanza la partida. Su función es inflingir daño constantemente en las peleas de equipo, por lo que el resto del equipo debe protegerlo.

- **Apoyo**: Campeón encargado de jugar en la línea inferior. Como su nombre indica, su función es apoyar al equipo y principalmente al tirador, ya que ambos juegan en la misma línea. Proporcionan utilidad como escudos, curaciones y tienen gran capacidad para iniciar peleas de equipo.

2.1.4. Metajuego

La selección de campeones en League of Legends no solo se ve afectada por las características y preferencias individuales de cada jugador, sino que también es influida por el metajuego actual. El meta *(most efficient tactic available)* [19] es un concepto en constante evolución que refleja la respuesta colectiva de la comunidad de jugadores a las actualizaciones que recibe el juego.

Cada dos semanas, Riot Games lanza un parche con actualizaciones en las que

algunos campeones sufren debilitaciones en sus estadísticas, conocidas como *nerfs*, mientras que otros reciben potenciaciones, conocidas como *buffs*. De esta forma, los desarrolladores consiguen equilibrar los campeones para que los jugadores tengan disponible una mayor variedad donde elegir. Estas actualizaciones no solo buscan mantener el equilibrio entre campeones, sino que también fomentar la diversidad e innovación en las estrategias del juego.

La llegada de los parches no solo implica modificaciones a las características individuales de los campeones, sino que también tiene un impacto signifiativo en las estrategias globales y las composiciones de equipo. Estrategias que alguna vez dominaron pueden quedar obsoletas, dando paso a nuevas tácticas que podían ser infravaloradas previamente. Por ejemplo, un campeón que recibe un *buff* puede pasar de ser una elección poco común a convertirse en una opción dominante, alterando las estrategias de selección o baneo en las partidas.

El impacto de estas actualizaciones se extiende desde las partidas casuales hasta la escena competitiva. En el ámbito profesional, los equipos deben adaptarse rápidamente a los cambios del meta para mantenerse competitivos. Esto puede implicar priorizar campeones distintos en la fase de selección de campeones, ajustar las estrategias de equipo o desarrollar nuevas sinergias entre jugadores. La capacidad de adaptación de un equipo al meta es a menudo un factor decisivo en su éxito en torneos y competiciones.

Figura 2.3: Actualización del parche 13.1 [3]

2.1.5. Mapa

El mapa en League of Legends es el escenario donde tienen lugar las partidas. Es simétrico y se divide en dos mitades, una para cada equipo, y a su vez, en tres calles conectadas por la jungla. La simetría del mapa asegura que ambos equipos tengan igualdad de condiciones al inicio de la partida.

Cada mitad del mapa tiene tres calles, *top, mid* y *bot*, conocidas como líneas, que conectan las bases de los dos equipos. En las líneas encontraremos torres defensivas, que son estructuras que atacan automáticamente a los rivales al acercarse. Las torres son una parte fundamental del mapa ya que proporcionan defensa contra los enemigos. La destrucción de las torres es crucial para abrir caminos hacia la base y nexo rival.

Además de las líneas, el mapa cuenta con la jungla. Es un área neutral entre las calles donde se encuentran monstruos neutrales que otorgan experiencia y oro al ser derrotados. Aparte de los recursos, también ofrece caminos alternativos a las líneas para moverse por el mapa, permitiendo emboscadas y estrategias que pueden cambiar el curso de la partida.

Figura 2.4: Mapa del League of Legends [4]

En la jungla se encuentran los objetivos neutrales más importantes del juego, el Heraldo, el Barón Nashor y los dragones. El Heraldo aparece en la parte superior de la jungla y, al ser derrotado, proporciona un beneficio que permite invocarlo para asediar torres enemigas. Una vez pasados 20 minutos, el Heraldo evoluciona al Barón Nashor, que otorga una poderosa potenciación al equipo que lo derrota, aumentando significativamente su poder y capacidad de asedio. En la zona inferior de la jungla se

encuentra el foso del dragón, donde aparecen varios tipos de dragones a lo largo de la partida. Cada dragón derrotado porporciona potenciaciones únicas al equipo, como mayor daño, curación, velocidad de movimiento, entre otras.

Figura 2.5: Objetivos neutrales de League of Legends. Elaboración propia

2.2. Selección inteligente

La selección inteligente es un proceso crítico en la toma de decisiones que permite evaluar y elegir la mejor alternativa entre varias opciones disponibles, utilizando criterios correctamente definidos. Esta sección se desglosa en varios apartados que abordan diferentes aspectos de la toma de decisiones, incluyendo enfoques y métodos específicos como el análisis multicriterio y el método TOPSIS.

2.2.1. Toma de decisiones

La toma de decisiones [20] es un proceso fundamental en diversos campos del conocimiento y en la vida cotidiana. Consiste en elegir una opción entre varias alter-

nativas para alcanzar un objetivo específico. La complejidad de este proceso puede variar desde decisiones simples a complejas. Las fases del proceso de una toma de decisión son las siguientes:

1. **Identificación del problema:** El primer paso en la toma de decisiones es comprender el contexto y factores que influyen en la situación que requiere una decisión.
2. **Recopilación de información:** Una vez identificado el problema, se debe recopilar la información necesaria para tomar una decisión informada.
3. **Identificación de alternativas:** Con la información recopilada, se generan múltiples alternativas entre las que poder decidir.
4. **Evaluación de alternativas:** Las alternativas identificadas se evalúan en función de criterios específicos. Esta evaluación puede involucrar técnicas y herramientas de apoyo a la decisión como los árboles de decisión.
5. **Selección de la mejor alternativa:** Basada en la evaluación, se selecciona la alternativa que se adapta mejor a los objetivos establecidos.

2.2.2. Toma de decisión multicriterio

La toma de decisiones multicriterio (MCDM) [21] es un enfoque utilizado para evaluar y elegir entre alternativas cuando se consideran mútliples criterios. Este método es especialmente útil en situaciones complejas donde las decisiones no pueden basarse en un solo factor. Cada alternativa se evalúa en función de varios criterios, sean cualitativos o cuantitativos. La finalidad es la misma que una toma de decisión simple, encontrar la mejor alternativa que cumpla los criterios considerados. El proceso de la toma de decisiones multicriterio es:

1. **Definición del problema:** El primer paso es identificar el problema de decisión y los objetivos que se quieren alcanzar. Esto incluye definir qué aspectos son importantes y qué criterios se utilizarán para evaluar las alternativas.
2. **Identificación de criterios y altenativas:** Se establecen los criterios y alternativas relevantes para la decisión.
3. **Asignación de pesos [22]:** Se asignan pesos a cada criterio para reflejar su relevancia en el contexto de la decisión. Estos pesos pueden determinarse mediante diversas técnicas.

4. **Suma de resultados:** Las puntuaciones obtenidas por cada alternativa en los diferentes criterios se combinan utilizando los pesos asignados a cada criterio. Esto se puede realizar mediante métodos como la suma ponderada, el método PROMETHEE [23] o el método TOPSIS entre otros.

5. **Selección de la mejor alternativa:** La alternativa con la puntuación más alta tras la suma de los resultados se selecciona como mejor.

2.2.3. TOPSIS

El método TOPSIS [24] es una técnica utilizada en la toma de decisiones multicriterio. Se basa en que la mejor alternativa tenga la mayor distancia posible a la solución no ideal y la menor distancia a la ideal. Este método se sustenta en la idea de que en un problema multicriterio existen dos soluciones ideales:

- **Solución ideal (PIS):** Representa el escenario más deseable, donde cada criterio alcanza su valor óptimo.
- **Solución no ideal (NIS):** Representa el escenario menos deseable, donde cada crierio alcanza el peor valor posible.

Los pasos del método TOPSIS son los siguientes [25]:

1. **Construcción de la matriz de decisión:** Se recopilan los datos correspondientes a cada criterio y alternativa, formando así una matriz de decisión. Las filas representan las alternativas y las columnas los criterios.

Decision Matrix		Criterion				
		C_1	C_2	C_3	...	C_n
Alternatives	A_1	x_{11}	x_{12}	x_{13}	...	x_{1n}
	A_2	x_{21}	x_{22}	x_{23}	...	x_{2n}
	A_3	x_{31}	x_{32}	x_{33}	...	x_{3n}
	⋮	⋮	⋮	⋮	⋱	⋮
	A_n	x_{m1}	x_{m2}	x_{m3}	...	x_{mn}

Figura 2.6: Matriz de decisión [5]

2. **Normalización de la matriz de decisión:** Se normalizan los valores de la matriz para hacer comparables los criterios. Se suele realizar utilizando la siguiente fórmula:

$$R_{ij} = \frac{X_{ij}}{\sqrt{\sum_{k=1}^{m} x_{kj}^2}} \tag{2.1}$$

3. **Construcción de la matriz de decisión ponderada:** Se multiplican los valores normalizados por los pesos asignados a cada criterio, obteniendo la matriz ponderada:

$$v_{ij} = w_j \cdot r_{ij} \tag{2.2}$$

donde w es el peso del j-ésimo criterio.

4. **Determinación de PIS y NIS:** Se identifica la solución ideal y no ideal:

   - **PIS:**

$$A^+ = \{v_1^+, v_2^+, ..., v_n^+\} \tag{2.3}$$

   donde $v_j^+ = max(v_{ij})$ para criterios de beneficio y $v_j^+ = min(v_{ij})$ para criterios de costo.

   - **NIS:**

$$A^- = \{v_1^-, v_2^-, ..., v_n^-\} \tag{2.4}$$

   donde $v_j^- = min(v_{ij})$ para criterios de beneficio y $v_j^- = max(v_{ij})$ para criterios de costo.

5. **Cálculo de las distancias:** Se calcula la distancia de cada alternativa a las soluciones ideal y no ideal:

$$D_i^+ = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^+)^2} \tag{2.5}$$

$$D_i^- = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^-)^2} \tag{2.6}$$

6. **Cálculo de los coeficientes de proximidad:** Se calcula el coeficiente de proximidad de cada alternativa con respecto a la solución ideal:

$$C_i^* - = \frac{D_i^-}{D_i^+ + D_i^-} \tag{2.7}$$

donde $0 \leq C_i^* \leq 1$. Un valor más cercano a 1 indica una mayor proximidad a la solución ideal.

7. **Ordenación:** Las alternativas se ordenan en función de sus coeficientes de proximidad, seleccionando la alternativa con el valor más alto como la mejor opción.

Capítulo 3

Diseño inicial

En este capítulo se presentan aspectos clave que estructuran la aplicación web desde sus primeras fases, abordando tanto la arquitectura técnica como los flujos de interacción. Primero, se detalla la arquitectura de la aplicación web, incluyendo tanto el diseño del frontend como del backend. También se describe el proceso de web scraping, una de las técnicas clave utilizadas para la recolección de datos de fuentes externas.

A continuación, se definen los principales casos de uso de la aplcación, modelando los diferentes escenarios en los que el usuario interactúa con el sistema. Asimismo, se complementa el diseño con la representación gráfica de los diagramas de secuencia y diagramas de clase.

3.1. Aplicación web

Se denomina aplicación web [26] a aquella herramienta que los usuarios pueden utilizar accediendo a un servidor web a través de internet mediante un navegador. Estas aplicaciones no requieren instalaciones locales en el dispositivo del usuario, lo que facilita el acceso y uso desde cualquier lugar con conexión a internet. La flexibilidad y accesibilidad de las aplicaciones web las han convertido en una solución popular para una amplia gama de necesidades, desde gestión de proyectos hasta entretenimiento y comercio electrónico.

La aplicación desarrollada en este proyecto tiene como objetivo principal proporcionar una herramienta funcional y eficiente que permita a los usuarios interactuar en tiempo real para la selección inteligente de campeones en League of Legends. Esta

herramienta no solo ayuda a los jugadores a optimizar sus elecciones sino que también les proporciona una ventaja estratégica al evaluar las composiciones de los equipos.

Para brindar una visión completa, esta sección se subdivide en varios apartados que cubren desde la arquitectura hasta las funcionalidades específicas.

3.1.1. Arquitectura de la aplicación

En esta subsección se describe la estructura de la aplicación web, incluyendo el diseño de frontend y backend, así como la interacción entre ambos componentes y las tecnologías empleadas, aunque posteriormente en el apartado 4.2 las analizaremos a fondo.

Frontend

Se refiere a la parte de la aplicación web que interactúa directamente con el usuario. Incluye todo lo que el usuario puede ver y con lo que puede interactuar. Las principales tecnologías usadas en el frontend son:

- **HTML *(HyperText Markup Language)***: Utilizado para estructurar el contenido de la página web. Es la base sobre la que se construyen todas las páginas web, definiendo elementos básicos como párrafos, encabezados, imágenes y enlaces.
- **CSS *(Cascading Style Sheets)***: Estiliza y da formato al contenido HTML. Permite diseñar visualmente la página web, controlando aspectos como colores, fuentes y disposición general de los elementos.
- **JavaScript**: Agrega interactividad y dinamismo a la página web. Permite responder a eventos del usuario, como clicks o desplazamientos y actualizar el contenido de la página web sin necesidad de recargarla.

Estas tecnologías trabajan en conjunto para crear una interfaz de usuario *(UI)* atractiva e interactiva.

Backend

Hace referencia a la parte de la aplicación web que se ejecuta en el servidor. Es responsable de la lógica, gestión de bases de datos, autenticación de usuarios y otras

funciones no visibles para el usuario final. Las tecnologías usadas en el backend son:

- **Node.js**: Entorno de ejecución de JavaScript del lado del servidor que permite ejecutar código en el servidor. Es conocido por su capacidad de manejar múltiples conexiones simultáneas con eficiencia, lo que lo hace ideal para aplicaciones en tiempo real.
- **Puppeteer [12]**: Biblioteca de Node.js que proporciona un API *(Application Programing Interface)* para controlar el navegador Chrome para realizar tareas de automatización. Es especialmente útil para tareas de web scraping, pruebas automatizadas y otras operaciones que requieren interacción con una página web.

Ambas tecnologías funcionan de forma coordinada para proporcionar funcionalidades y datos al frontend de la aplicación, creando una experiencia completa y dinámica para el usuario.

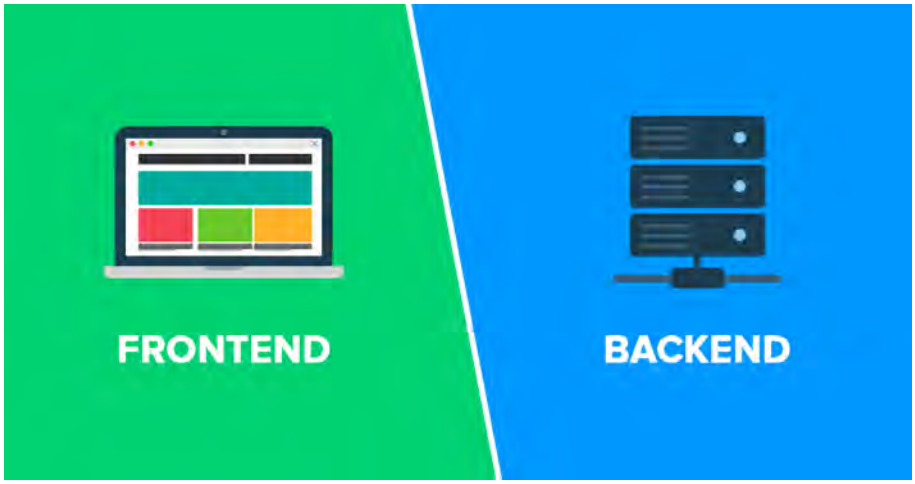

Figura 3.1: Comparación entre frontend y backend [6]

3.1.2. Funcionalidades

En esta subsección se detallan las funcionalidades clave de la aplicación web desarrollada. Cuenta con varias secciones clave, organizadas en distintas páginas, cada una con funcionalidades que permiten al usuario interactuar de manera eficiente con la plataforma. Entre estas secciones se incluyen el listado de campeones, la tier list, la búsqueda de campeones, la página específica de cada campeón y la herramienta de selección de campeones. Estas funcionalidades están diseñadas para ofrecer una experiencia de usuario completa, facilitando el acceso a la información de forma sencilla.

- **Listado de campeones**: Proporciona un listado de todos los campeones junto con las posiciones en las que se juegan y su rol. Además, muestra los porcentajes de victoria tanto positivos como negativos contra sus principales enfrentamientos, ofreciendo una visión detallada de su rendimiento, y los objetos que se compran de forma frecuente con dicho campeón.

- **Búsqueda de campeones**: Permite a los usuarios buscar campeones por su nombre, facilitando el acceso a la información de una forma más rápida.

- **Tier list**: Lista jerarquizada que muestra todos los campeones, posición en la que se juega y tasa de victoria. Permite a los usuarios identificar de una forma sencilla qué campeones son más efectivos en el metajuego actual.

- **Herramienta de selección inteligente**: Permite a los usuarios seleccionar campeones de forma inteligente basándose en la composición enemiga, los enfrentamientos directos en línea, roles, etc., optimizando la estrategia y aumentando las posibilidades de victoria.

A pesar de la diversidad de funciones que cada página ofrece, todas ellas comparten ciertos elementos comunes que forman parte de la navegación e interacción dentro de la aplicación. Estos elementos son la cabecera, el pie de página, el slider y el campo de búsqueda. Estos componentes aseguran que los usuarios puedan moverse fácilmente entre las distintas secciones de la aplicación.

3.2. Web scraping

El web scraping [27] es una técnica utilizada para extraer información de sitios web de manera automática. Esta metodología es especialmente útil cuando la cantidad de datos que necesitamos es elevada y sería inviable o extremadamente laborioso obtenerlos manualmente. Empresas como aquellas que comparan precios de productos, servicios de noticias y otras muchas utilizan esta técnica para obtener datos actualizados.

En el contexto de este proyecto, el web scraping es esencial para obtener datos actualizados sobre los campeones de League of Legends. Estos datos son indispensables para alimentar la aplicación web desarrollada, proporcionando información precisa y actualizada a los usuarios. Los cambios en las estadísticas de los enfrentamientos entre campeones y tendencias del metajuego cambian frecuentemente por lo

que hacen del web scraping una herramienta crucial para mantener la exactitud de la información mostrada.

El proceso de web scraping para obtener datos en este proyecto es el siguiente:

1. **Análisis del código HTML:** En primer lugar, se realiza un análisis manual del código HTML de las páginas web objetivo. En este paso se identifican los selectores y elementos HTML que contienen la información deseada como etiquetas, clases, identificadores o selectores.
2. **Extracción de datos:** Utilizando herramientas y bibliotecas de scraping como Puppeteer, se crean scripts para acceder a las páginas web y extraer la información relevante. Estos scripts recorren el código HTML, seleccionan los elementos identificados en el paso anterior y extraen los datos contenidos en ellos.
3. **Transformación de los datos:** Los datos extraídos son transformados a un formato estructurado como JSON (*JavaScript Object Notation*). Este formato es ideal para la manipulación y almacenamiento de datos, ya que es legible por máquinas y humanos.
4. **Almacenamiento y uso:** Los datos en formato JSON son almacenados para su posterior uso en la aplicación.

Es crucial considerar las implicaciones éticas y legales que tiene el uso de web scraping [28]. Es fundamental asegurar que esta práctica no incumple los términos de servicio de las páginas web objetivo. Además, se debe evitar la recolección de datos personales sin consentimiento y no sobrecargar los servidores de las páginas web con demasiadas solicitudes en un corto periodo de tiempo.

3.3. Casos de uso

En esta sección se presentarán los casos de uso de la aplicación web, proporcionando ejemplos prácticos de cómo los usuarios pueden interactuar con las distintas funcionalidades del sistema. Los casos de uso describen situaciones concretas en las que el usuario realiza una serie de acciones con el fin de cumplir un objetivo específico dentro de la aplicación.

Caso de uso general

En el caso de uso general, el usuario puede interactuar con la página actual o navegar a otra sección de la aplicación. Dependiendo de las opciones que se le presenten dentro de la página, podrá realizar acciones como consultar información, acceder a funcionalidades específicas o cambiar de página. Ante este panorama, obtenemos el siguiente diagrama:

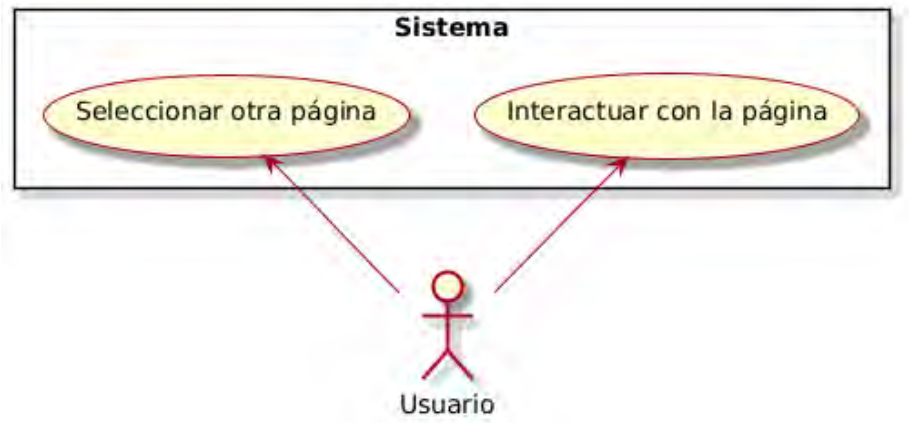

Figura 3.2: Diagrama de casos de uso general. Elaboración propia

Listado de campeones

En este diagrama de casos de uso se ilustra cómo el usuario interactúa con el listado y la búsqueda de campeones. El usuario tiene varias opciones, puede buscar un campeón por su nombre, visualizar las sugerencias que coincidan con su búsqueda y, si lo desea, seleccionar uno de los campeones sugeridos. Además, también puede navegar hacia otras secciones de la aplicación en cualquier momento o seleccionar un campeón de los mostrados en el listado.

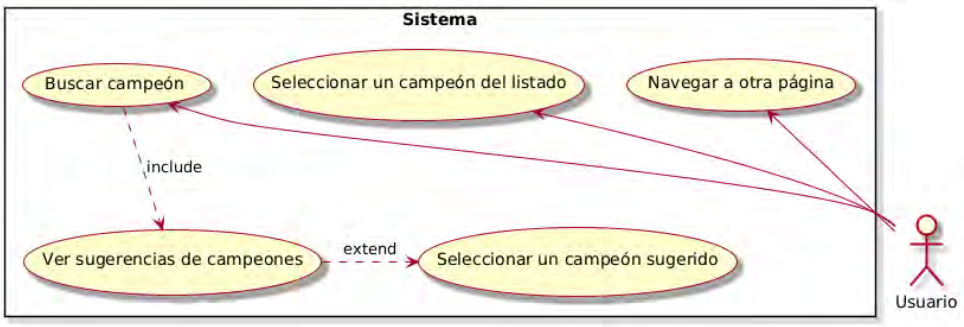

Figura 3.3: Diagrama de casos de uso del listado de campeones. Elaboración propia

Tier list

En este diagrama de casos de uso se refleja cómo el usuario interactúa con la tier list. Al igual que en el diagrama anterior, el usuario puede buscar campeones por su nombre, visualizar las sugerencias coincidentes y seleccionar uno de los campeones sugeridos. Asimismo, el usuario puede visualizar la información que le proporciona la tier list o navegar libremente a otras páginas de la aplicación.

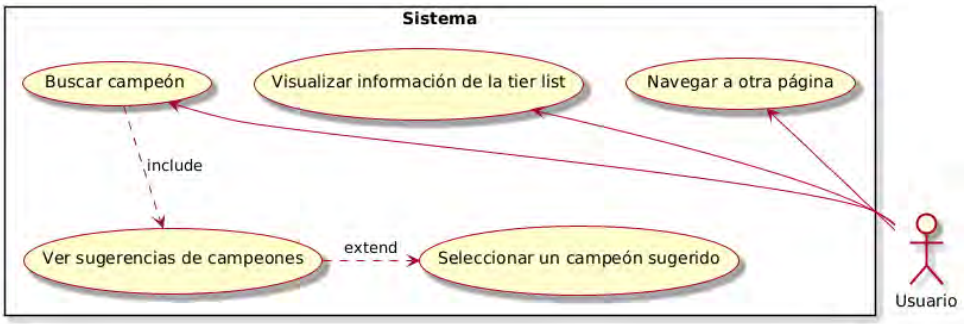

Figura 3.4: Diagrama de casos de uso de la tier list. Elaboración propia

Herramienta de selección de campeones

Este diagrama de casos de uso representa la interacción del usuario con la herramienta de selección inteligente de campeones. La herramienta permite al usuario buscar campeones por su nombre y navegar libremente por las otras secciones de la aplicación web, al igual que el resto de diagramas.

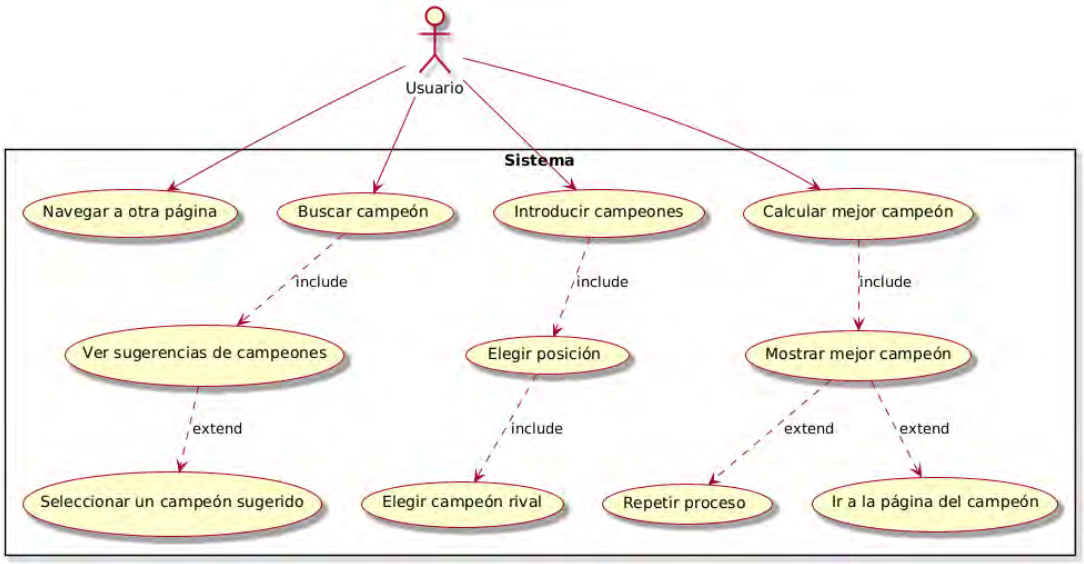

Figura 3.5: Diagrama de casos de uso de la herramienta de selección inteligente de campeones. Elaboración propia

Además, el usuario puede introducir campeones que desea jugar, seleccionar una posición para ellos y elegir un rival. Una vez introducidos los campeones, el usuario puede calcular cuál es el mejor de ellos y el sistema mostrará este campeón. Después de ver el campeón seleccionado, el usuario tiene la opción de repetir el proceso de selección o visitar la página específica del campeón mostrado.

3.4. Diagramas de secuencia

En esta sección, se explorarán los diagramas de secuencia que detallan el flujo de interacción entre los componentes del sistema para cada caso de uso presentado anteriormente. Mientras que los casos de uso ofrecen una descripción general de cómo los usuarios interactúan con la aplicación web, los diagramas de secuencia proporcionan una visión más detallada del orden en que se producen las interacciones entre actores y objetos del sistema.

Listado de campeones

El siguiente diagrama de secuencia ilustra el proceso de interacción del usuario con el sistema de búsqueda de campeones en la aplicación web. Este diagrama detalla cómo el usuario puede realizar una búsqueda de campeones, ver las sugerencias en tiempo real, seleccionar un campeón de las sugerencias o del listado y navegar a otras secciones de la aplicación.

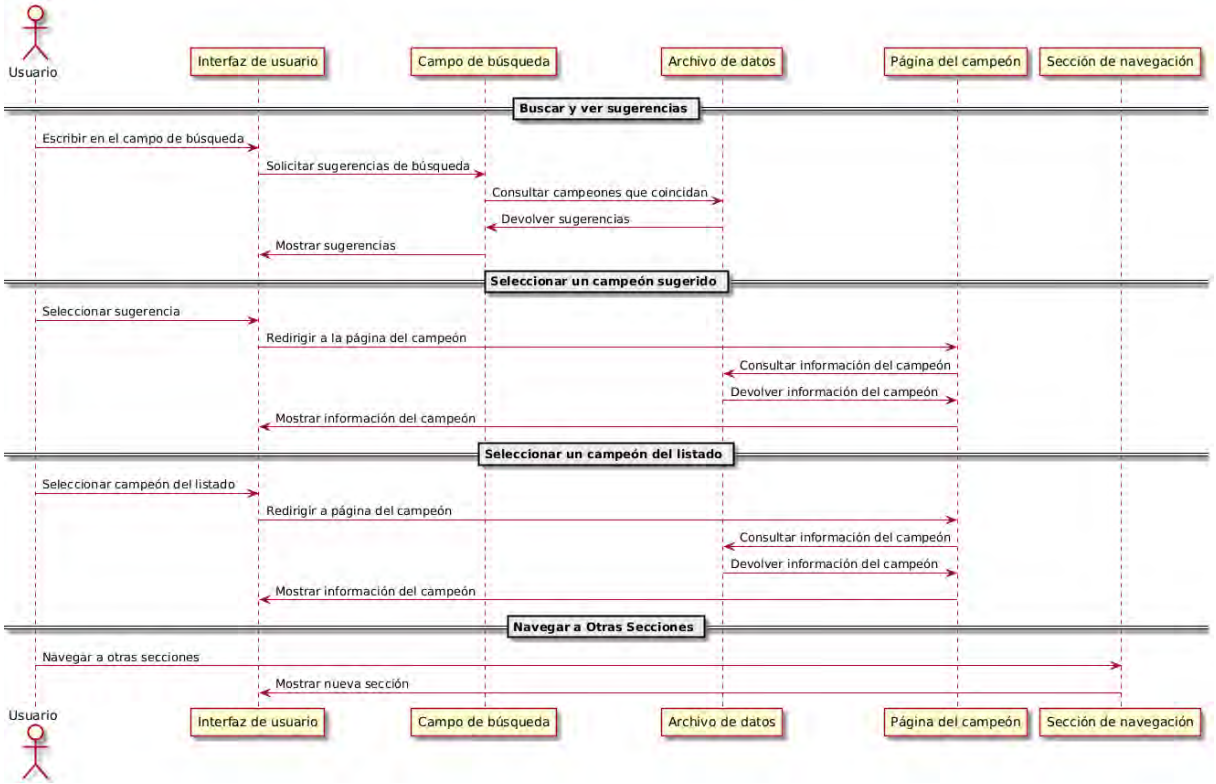

Figura 3.6: Diagrama de secuencia del listado de campeones. Elaboración propia

Tier list

El siguiente diagrama de secuencia ilustra cómo el usuario interactúa con la tier list de la aplicación web. Al igual que en el diagrama anterior, también se refleja la posibilidad de que el usuario navegue libremente por el resto de la página y realice acciones como la búsqueda de campeones.

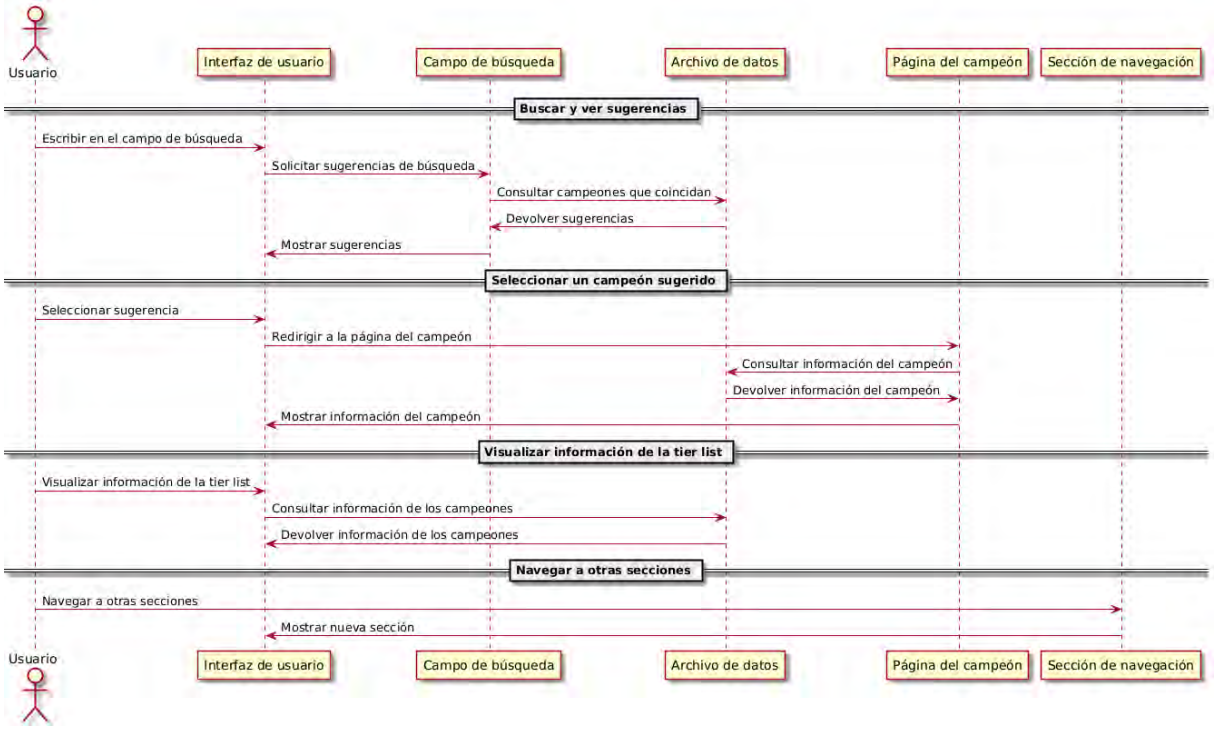

Figura 3.7: Diagrama de secuencia de la tier list. Elaboración propia

Herramienta de selección de campeones

El siguiente diagrama de secuencia detalla el flujo de interacción entre el usuario y la herramienta de selección de campeones de la aplicación web. El usuario puede introducir varios campeones, seleccionar una posición o campeón rival, y solicitar al sistema que calcule cuál es el mejor campeón para ese contexto. Luego, puede optar por repetir el proceso o visitar la página del que ha sido considerado como mejor campeón.

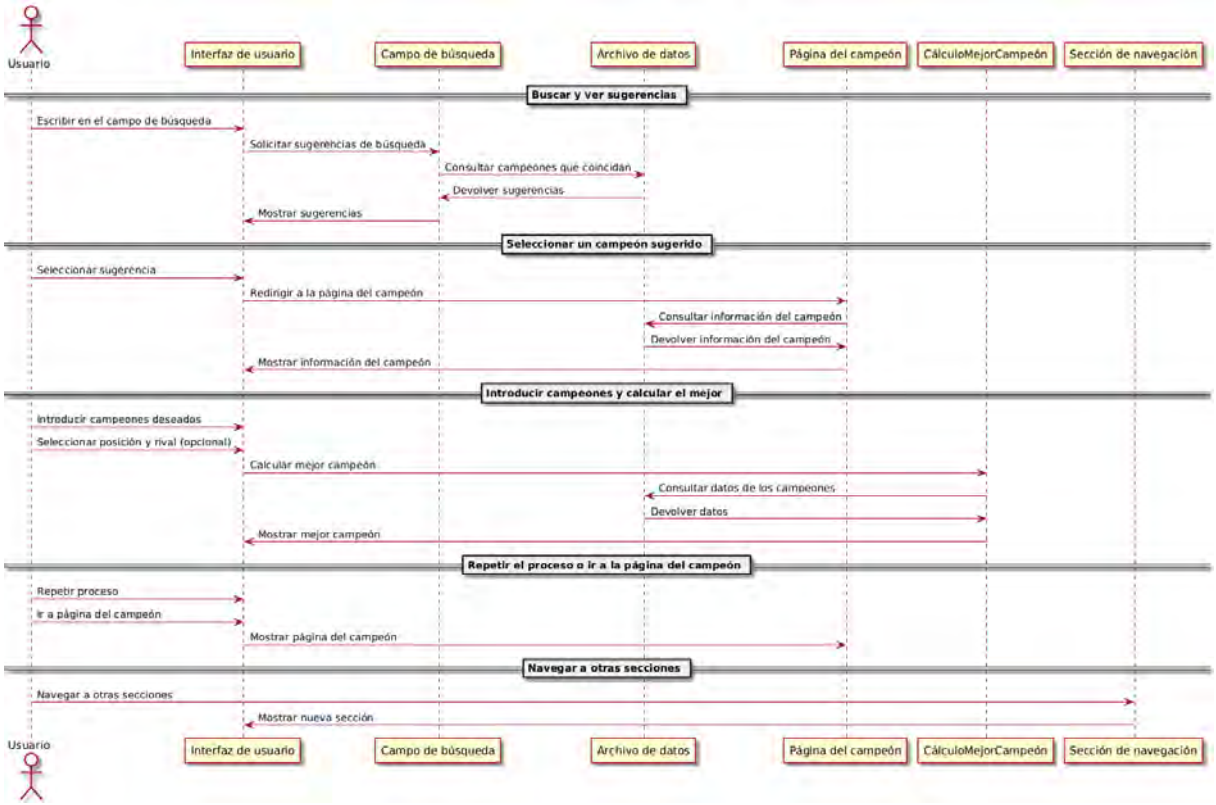

Figura 3.8: Diagrama de secuencia de la herramienta de selección de campeones. Elaboración propia

Capítulo 4

Desarrollo de la aplicación web

En este capítulo, se aborda de manera detallada el proceso de desarrollo de la aplicación web, comenzando con una descripción completa de los recursos empleados. Este desarrollo incluye tanto los datos utilizados como las herramientas que hicieron posible la implementación de las diversas funcionalidades. Uno de los componentes esenciales del desarrollo fue el conjunto de datos. A lo largo del capítulo, se explica cómo fueron obtenidos estos datos, qué fuentes se utilizaron y cómo se procesaron para ser integrados adecuadamente en la plataforma.

4.1. Conjunto de datos

Según se ha descrito en la subsección 1.2.2, el conjunto de datos con el que trabajaremos se obtiene de la página oficial de League of Legends y de OP.GG. La recolección y procesamiento de estos datos es sumamente importante para el desarrollo del proyecto, dado que la precisión de la información impacta de forma directa en la funcionalidad y fiabilidad de la aplicación web.

La página oficial de League of Legends proporciona información sobre los campeones como sus habilidades, roles y estadísticas básicas. Por otro lado, OP.GG es una plataforma reconocida entre la comunidad de jugadores de LoL que ofrece una extensa gama de datos sobre campeones, jugadores, historial de partidas o búsqueda de partidas en vivo, entre otros. Esta página web es extensamente utilizada para el análisis de partidas y enfrentamientos entre campeones, proporcionando una visión detallada sobre las tendencias y estadísticas actuales del juego.

A pesar de la gran cantidad de información disponible en OP.GG, no toda resulta

relevante para el proyecto. Por ello, se realiza un proceso de filtrado y selección de datos que sean realmente útiles. Aunque la página muestra los datos de una forma sencilla y visual, obtenerlos para usarlos en nuestra aplicación web es un proceso complejo que se explicará a fondo en la sección 4.3.

4.2. Tecnologías usadas

En este proyecto se han empleado diversas tecnologías para desarrollar una aplicación web robusta y eficiente. Estas tecnologías permiten la creación de una interfaz de usuario atractiva y la gestión de datos complejos. A continuación se presenta una breve descripción de cada una de las tecnolgías utilizadas.

4.2.1. IntelliJ IDEA

IntelliJ IDEA es un IDE *(Integrated development environment)* desarrollado por JetBrains. Este IDE es utilizado para escribir, probar y depurar el código. IntelliJ ofrece herramientas avanzadas para el desarrollo web, facilitando la implementación del proyecto.

Beneficios de utilizar IntelliJ IDEA

- **Soporte para múltiples lenguajes:** Es compatible con una amplia gama de lenguajes de programación, incluyendo Java, JavaScript, HTML, y más. Esto lo convierte en una herramienta versátil ya que en el proyecto se utilizan varios lenguajes.

- **Depuración y pruebas:** IntelliJ ofrece potentes herramientas de depuración y pruebas. Se pueden establecer puntos de interrupción, inspeccionar variables y evaluar expresiones durante la ejecución del programa.

- **Facilidad de uso:** La interfaz intuitiva y su capacidad para personalizar atajos y configuraciones permiten adaptarse rápidamente y trabajar de manera más cómoda.

Figura 4.1: Logo de IntelliJ [7]

4.2.2. HTML

HTML *(HyperText Markup Language)* [29] es el lenguaje estándar utilizado mundialmente para crear y estructurar el contenido de las páginas web. Utiliza una serie de elementos o etiquetas como `<head>`, `<title>`, `<body>`, `<header>`, `<footer>`, `<section>`, `<div>`, `<span>`, `<img>`, `<nav>`, `<li>` entre otras, para definir diferentes partes del contenido. HTML es considerado el lenguaje web más importante y su invención fue crucial para el surgimiento, desarrollo y expansión de la *World Wide Web* (WWW).

Beneficios de utilizar HTML

- **Simplicidad:** Es fácil de aprender y usar. Su sintaxis sencilla permite crear páginas web básicas rápidamente. Las etiquetas y atributos son intuitivos y descriptivos.
- **Compatibilidad y estándares:** HTML es un estándar mantenido por el *World Wide Web Consortium* (W3C), lo que asegura su compatibilidad con todos los navegadores y plataformas. Esto significa que las páginas web creadas con este lenguaje serán accesibles desde cualquier dispositivo.
- **Integración con CSS y JavaScript:** HTML se integra perfectamente con estos lenguajes. HTML se encarga de la estructura y el contenido, CSS del diseño y presentación, y JavaScript añade interactividad y comportamiento dinámico. Esta separación de responsabilidades facilita el desarrollo web modular y escalable.

Figura 4.2: Logo de HTML [8]

4.2.3. JavaScript

JavaScript [30] es un lenguaje de programación de alto nivel, dinámico e interpretado que se utiliza principalmente para el desarrollo web y creación de páginas interactivas. Es conocido por su capacidad de responder a eventos del usuario como clicks del ratón, entradas de teclado u otras interacciones.

Beneficios de utilizar JavaScript

- **Dinamismo:** Es un lenguaje dinámico que permite cambios en la estructura y comportamiento del contenido web en tiempo real. Esto incluye la modificación del DOM *(Document Object Model)*, lo que facilita la actualización dinámica del contenido de una página web sin necesidad de recargarla.
- **Interpretado:** Es interpretado por el navegador del usuario, lo que significa que no requiere una compilación previa. Esto permite que la ejecución sea más rápida.
- **Desarrollo full stack:** JavaScript se ha convertido en un lenguaje que puede ser utilizado tanto para el desarrollo frontend como backend. Esto reduce la necesidad de aprender varios lenguajes y facilita la integración entre el cliente y el servidor.

Figura 4.3: Logo de JavaScript [9]

4.2.4. CSS

CSS *(Cascading Style Sheets)* es un lenguaje utilizado para manejar el diseño y presentación de las páginas web que utilizan un lenguaje de marcado como HTML. Permite aplicar estilos como colores, fuentes, márgenes, líneas, alturas, anchos, entre otros. Utiliza selectores para apuntar a los elementos HTML a los que aplicar un estilo.

Beneficios de utilizar CSS

- **Responsive design:** Permite crear diseños adaptativos mediante el uso de media queries, que permiten aplicar diferentes estilos según el tamaño y las características del dispositivo del usuario. Esto asegura que las páginas web se vean bien en cualquier dispositivo, desde teléfonos móviles hasta monitores.
- **Consistencia en el diseño:** Permite aplicar estilos uniformes a mútliples elementos, asegurando un diseño coherente en toda la página web que mejore la experiencia del usuario.
- **Compatibilidad y estándares:** CSS, al igual que HTML, es compatible con todos los navegadores y es un estándar reconocido por el W3C. De esta forma se garantiza que los estilos aplicados mediante CSS funcionarán de manera consistente en diferentes navegadores y dispositivos.

Figura 4.4: Logo de CSS [10]

4.2.5. Node.js

Es un entorno de ejecución de JavaScript del lado del servidor que permite ejecutar código fuera del navegador. En este proyecto, Node.js [31] se utiliza para gestionar el backend, incluyendo la lógica y la interacción con los datos.

Beneficios de utilizar Node.js

- **Asincronía y event-driven:** Está diseñado para ser asíncrono y basado en eventos, lo que significa que puede manejar varias operaciones al mismo tiempo sin bloquear el hilo principal.
- **Soporte JSON:** Node.js se integra perfectamente con JSON, lo que facilita la gestión y manipulación de datos.

- **Desarrollo full stack con JavaScript:** Permite utilizar JavaScript tanto en el frontend como en el backend, unificando el lenguaje de programación.

Figura 4.5: Logo de Node.js [11]

4.2.6. Puppeteer

Puppeteer [32] es una librería de Node.js que proporciona una API de alto nivel para controlar navegadores web basados en Chromium como Google Chrome. Fue desarrollada por el equipo de Google Chrome y se utiliza principalmente para automatizar tareas del navegador o para hacer web scraping.

Beneficios de utilizar Puppeteer

- **Interacción con el DOM:** Facilita la manipulación del DOM, permitiendo seleccionar y modificar elementos, disparar eventos y recuperar datos directamente desde las páginas web.
- **Automatización del navegador:** Puppeteer permite la automatización de casi cualquier acción que un usuario pueda realizar en un navegador, desde la navegación y e interacción con formularios hasta la captura de la pantalla y la generación de archivos PDF.
- **Control de recursos:** Puede interceptar y modificar solicitudes de red, bloquear recursos como imágenes o anuncios para acelerar la carga de la página.

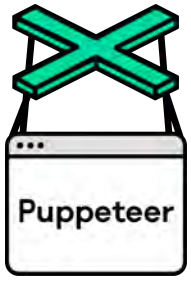

Figura 4.6: Logo de Puppeteer [12]

4.3. Obtención de datos

En este apartado se detalla el proceso seguido para obtener los datos necesarios de las páginas web mediante web scraping. Este proceso es esencial para conseguir información precisa y actualizada para utilizarla en la aplicación web.

El primer paso fue realizar un análisis detallado del código HTML de las páginas web objetivo. Esta tarea se realiza de forma manual, permitiendo identificar y seleccionar los elementos HTML que contienen la información relevante. Para facilitar el proceso, se utiliza la herramienta Google DevTool, que es parte de las Chrome Developer Tools. Esta herramienta permite inspeccionar y explorar las estructuras HTML y CSS de manera interactiva, ayudando a localizar los selectores e identificadores de los elementos de interés.

De la primera página web, que corresponde a OP.GG, se extraen elementos visuales y estadísticas. En estos, se incluyen las imágenes de los campeones, porcentajes de victoria, selección y baneo. Estos datos se obtienen analizando las secciones correspondientes de la página y posteriormente accediendo a las páginas específicas de cada campeón para analizarlas a fondo.

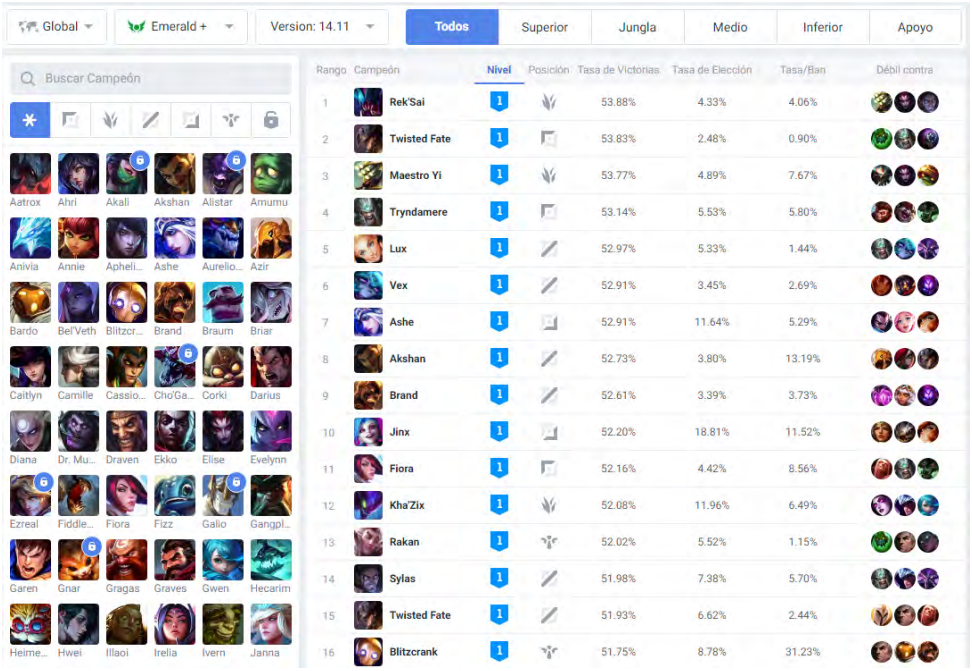

Figura 4.7: Tier list de OPGG. Elaboración propia

En la página de cada campeón se obtiene información clave sobre los enfrentamientos directos entre campeones, tanto favorables como desfavorables, y los objetos que los jugadores suelen comprar para optimizar el rendimiento del campeón, permitiendo así conocer las caracteríticas del campeón.

La segunda página de interés es el sitio oficial de League of Legends. De esta página obtenemos el rol de cada campeón.

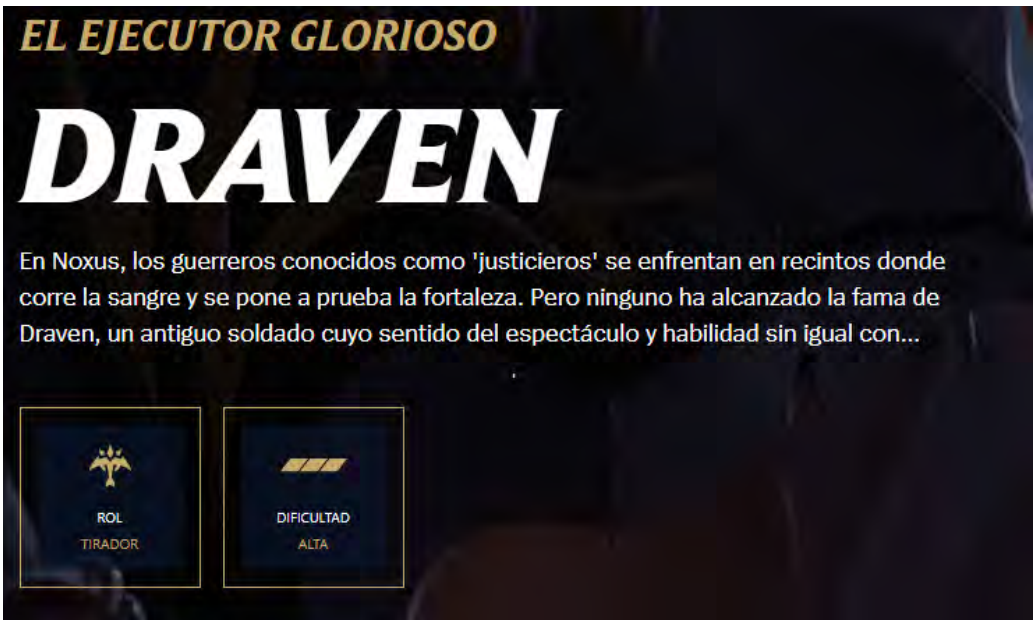

Figura 4.8: Ejemplo de rol de un campeón. Elaboración propia

Una vez identificados los elementos HTML relevantes, el siguiente paso es automatizar el proceso de extracción de datos utilizando Puppeteer.

Para ilustrar cómo utilizar Puppeteer en este contexto, se presenta un ejemplo básico que muestra los pasos iniciales de la automatización. Este ejemplo cubre la inicialización del navegador, la navegación a una página web específica y la realización de una tarea simple como una captura de pantalla. Este proceso inicial proporciona una base para comprender cómo utilizar Puppeteer en proyectos más complejos como la extracción de datos que abordaremos a continuación.

```
1  const puppeteer = require('puppeteer');
2
3  (async () => {
4      const navegador = await puppeteer.launch();
5      const pagina = await navegador.newPage();
6
7      await pagina.goto('https://ejemplo.com');
8      await pagina.screenshot({path: 'ejemplo.com'});
9      console.log('Captura de pantalla realizada con éxito');
10
11     await navegador.close();
12 })();
```

Listado 4.1: Ejemplo de uso de Puppeteer

En nuestro caso, el proceso es considerablemente más complejo. Aunque la inicialización del navegador y la navegación a la página de OP.GG siguen los pasos del ejemplo básico, a partir de ahí se añaden varias capas de complejidad. Una vez en la página de OP.GG, se configura Puppeteer para esperar hasta que los selectores necesarios estén presentes y la página esté completamente cargada. A continuación, se procede a recorrer la tabla que contiene los datos de los campeones, extrayendo los atributos y almacenándolos en un vector para su posterior procesamiento.

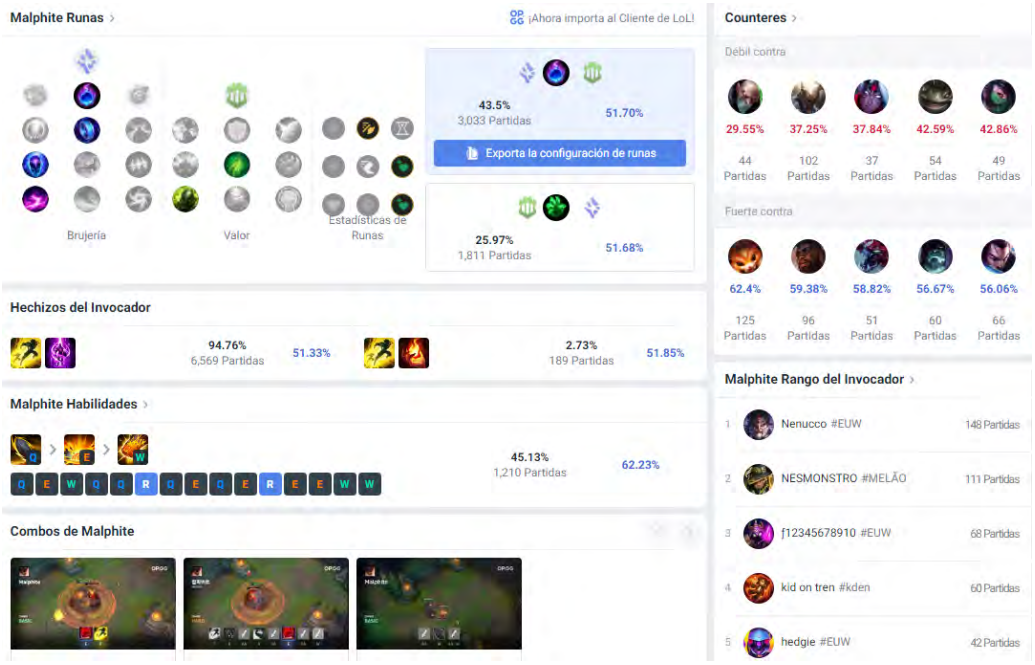

Figura 4.9: Página con los datos del campeón. Elaboración propia

Para utilizar los datos obtenidos de manera efectiva dentro de la aplicación web, es necesario convertirlos a un formato estructurado y fácilmente manipulable como JSON. Una vez que los atributos necesarios están almacenados en un vector, se procede a transformar este vector a un archivo JSON.

```
1 const fs = require('fs');
2
3 const datosJSON = JSON.stringfy(datosCampeones, null, 2);
4 fs.writeFile('datosCampeones.json', datosJSON);
```

Listado 4.2: Transformación a JSON

Para acceder a las páginas específicas de cada campeón y obtener los porcentajes de victoria frente a sus principales enfrentamientos, seguimos un proceso similar al empleado en la extracción de datos de la tier list. Al cargar la página del campeón, esperamos a que los selectores necesarios estén presentes. Posteriormente, procedemos a recorrer la tabla que contiene la información que necesitamos, obteniendo los datos necesarios. Una vez recopilados, se almacenan temporalmente en un vector para posteriormente convertirlos a un archivo JSON.

```
for(let i=0; i<campeones.length; i++){
    const url = campeones[i].url.replace('build', 'counters');
    await page.goto( url: `https://www.op.gg${url}`);

    await page.waitForSelector( selector: '[class^=" css-117bph8"]');
    const datosCounter = await page.evaluate( pageFunction: () => {
        const filas = document.querySelectorAll( selectors: '[class^=" css-117bph8"] tr');
        const campeones = [];

        for(let fila of filas){
            const campeon = {};
            const nombreCampeon = fila.querySelector( selectors: 'td:nth-child(2) div div');
            const winRateCampeon = fila.querySelector( selectors: 'td:nth-child(3) span');

            if(nombreCampeon && winRateCampeon) {
                campeon.nombre = nombreCampeon.innerText;
                campeon.win = winRateCampeon.innerText;

                campeones.push(campeon);
            }
        }
        return campeones;
    });
```

Figura 4.10: Web scraping de los enfrentamientos de OP.GG. Elaboración propia

Para obtener la información acerca de los objetos de cada campeón en la página de OP.GG, debemos afrontar varios inconvenientes que dificultan el proceso de scraping. El primer obstáculo que encontramos es la presencia de una ventana emergente relacionada con la información de cookies del sitio web. Esta ventana interfiere con nuestra capacidad para interactuar con la página, por lo que resulta necesario evitarla antes de empezar con la extracción de datos. Para resolver este problema, implementamos un método que cierra la ventana emergente de forma automática.

```
1    await page.evaluate(() => {
2        document.querySelector('[id="id-objetivo"]').remove();
3    });
```

Listado 4.3: Método para cerrar ventanas emergentes

El segundo problema que surge es que las características de los objetos no aparecen a simple vista; en lugar de ello, requieren que el usuario pase el ratón sobre la imagen del objeto para que se despliegue la información detallada. Este comportamiento complica la extracción de datos, ya que, durante el scraping, necesitamos simular la acción de pasar el ratón por encima de cada imagen de los objetos de forma automática. Para ello, se recorren todas las imágenes de objetos presentes en la página y se simula el paso del ratón por encima, lo que nos permite acceder a las estadísticas.

```
1    const elementos = await page.$$('[id="id-objetivo"]');
2    for(const elemento of elementos){
3        await elemento.hover();
4        await new Promise(resolve => setTimeout(resolve, 200));
5    }
```

Listado 4.4: Método para simular el paso del ratón

Una vez solventados estos problemas, procedemos a recorrer la lista de objetos, seleccionando la información relevante para nuestro análisis. Los datos extraídos incluyen el nombre del objeto, su precio y sus estadísticas específicas. Estos datos se almacenan temporalmente en un vector aunque posteriormente, los transformamos en un archivo JSON para mantener la información estructurada.

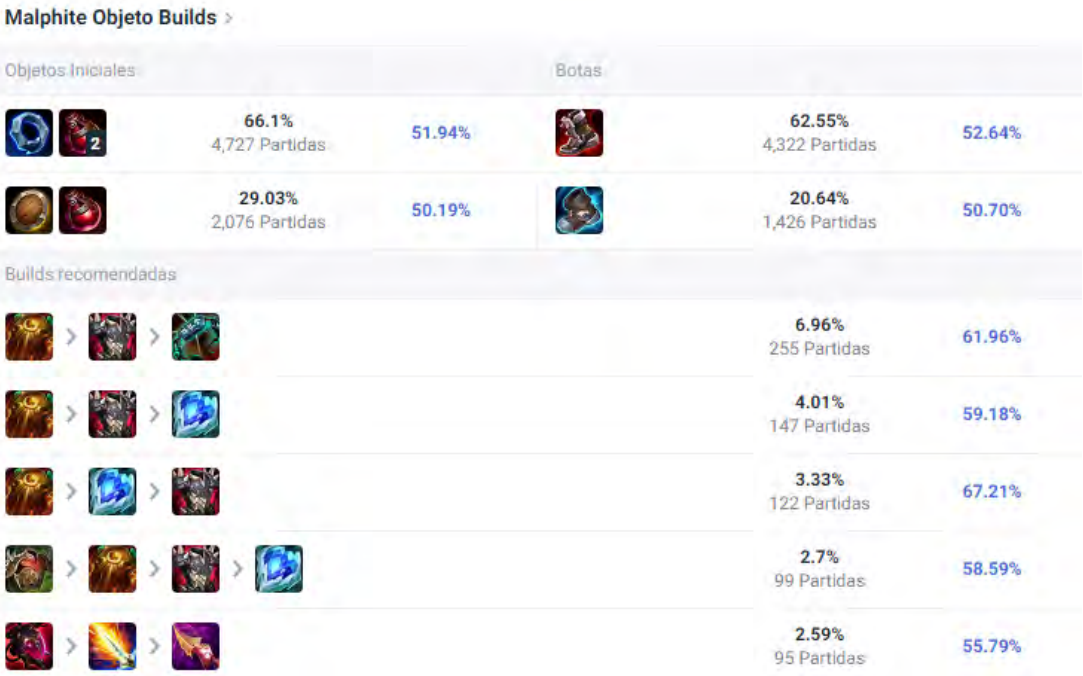

Figura 4.11: Página con los objetos del campeón. Elaboración propia

Finalmente, el proceso concluye con la generación de varios archivos JSON que tienen la siguiente estructura:

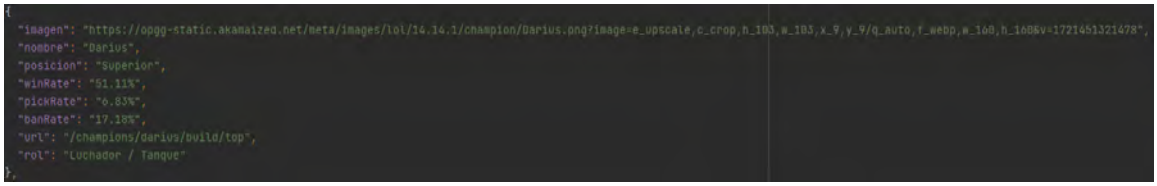

Figura 4.12: Estructura del archivo JSON de datos de campeones. Elaboración propia

Este archivo JSON contiene una lista de objetos, donde cada objeto representa un campeón con sus respectivos atributos. Este formato permite que los datos sean fácilmente accesibles y manipulables dentro de nuestra aplicación web.

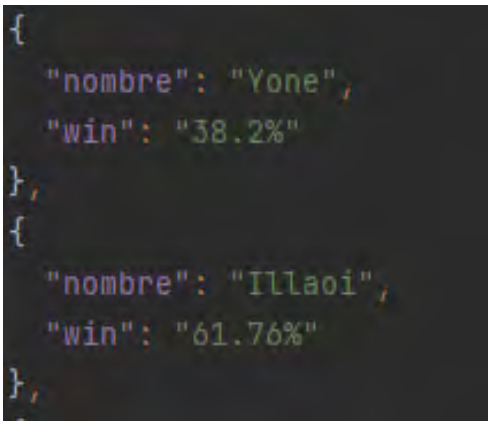

Figura 4.13: Estructura del archivo JSON de enfrentamientos. Elaboración propia

Este archivo JSON contiene una lista de objetos, donde cada objeto representa uno de los principales enfrentamientos del campeón. Cada enfrentamiento tiene el nombre del campeón y el porcentaje de victoria. Si el porcentaje es inferior al 50 % implica que es un enfrentamiento negativo. Si es superior al 50 %, es un enfrentamiento en el que tendremos ventaja.

Figura 4.14: Estructura del archivo JSON de los objetos. Elaboración propia

Este archivo JSON contiene una lista de objetos, donde cada elemento representa un objeto que se compra habitualmente con el campeón. Cada objeto tiene su nombre, precio, imagen y una lista con las estadísticas.

4.4. Decisiones inteligentes

En este apartado se describe el enfoque adoptado para optimizar la selección de campeones dentro de la aplicación web. Aquí se explica el planteamiento del problema, la información utilizada para fundamentar las decisiones y la aplicación del método TOPSIS para generar recomendaciones efectivas.

4.4.1. Planteamiento

Tomar una decisión en cualquier contexto puede ser un proceso que oscile entre lo arbitrario o basado en azar, o bien, fundamentarse en preferencias personales sin un análisis previo. Sin embargo, cuando disponemos de datos e información relevante, podemos transformar este proceso en una decisión inteligente e informada. En el ámbito de los videojuegos, especialmente en LoL, la capacidad de tomar decisiones informadas puede ser decisiva para el éxito en una partida.

Durante una partida de LoL, los jugadores se enfrentan a la necesidad de tomar miles de decisiones de forma instantánea. Estas decisiones, en su mayoría, están influenciadas por la experiencia, intuición y habilidad individual del jugador. No obstante, existe una fase en el juego en la que los jugadores disponen de un valioso tiempo adicional para decidir: la fase de selección de campeones. Este es un momento clave en el que, en lugar de optar por un campeón al azar o basado únicamente en preferencias personales, el jugador puede tomar una decisión más estratégica fundamentada en criterios específicos.

Figura 4.15: Fase de selección de campeones [13]

Con este propósito en mente, se ha diseñado una herramienta que permite a los jugadores identificar el mejor campeón posible para cada situación. Esta herramienta no solo tiene en cuenta las preferencias del usuario, sino que también evalúa una serie de criterios que pueden influir en el resultado final de la partida. En un juego donde cada detalle cuenta, una selección adecuada de campeones puede ser la diferencia entre la victoria y la derrota.

4.4.2. Información utilizada

Para asegurar una selección de campeones óptima, la herramienta se apoya en un amplio conjunto de datos, entre los que se incluyen porcentajes de victoria, tasas de selección y baneo, enfrentamientos favorables y desfavorables, así como las posiciones en las que cada campeón suele jugarse. Este conjunto de datos permite realizar un análisis de las opciones disponibles, ofreciendo al usuario la mejor recomendación posible en función de su situación.

El funcionamiento de la herramienta es sencillo e intuitivo. El usuario debe proporcionar una lista de campeones que desea jugar. Además, tiene la opción de introducir información adicional como el campeón rival al que se enfrentará y la posición en la que jugará, si dispone de esa información. Según los datos introducidos por el usuario, la herramienta calculará la mejor opción de manera personalizada.

En ambos casos, el mejor campeón se determina utilizando el método TOPSIS. La diferencia es que, si el usuario introduce tanto el campeón rival como la posición en la que jugará, en lugar de emplear el porcentaje de victoria general del campeón, se puede utilizar el porcentaje de victoria específico del enfrentamiento contra ese rival. Esto permite una decisión más precisa, ya que no solo se toma en cuenta el porcentaje de victoria en el enfrentamiento, sino que también criterios adicionales como el porcentaje de selección y la tasa de baneo.

Por otro lado, si el usuario únicamente proporciona la lista de campeones sin especificar el rival ni la posición, el método TOPSIS recurre al porcentaje de victoria general del campeón. En este caso, el método de decisión multicriterio evalúa las distintas alternativas en función de varios criterios como el porcentaje de victoria global del campeón, el porcentaje de selección y el de baneo, para determinar cuál de ellas es la opción más equilibrada.

4.4.3. Aplicación del método TOPSIS

Para aplicar el método TOPSIS en este proyecto, seguimos los pasos previamente descritos en el subapartado 2.2.3, adaptándolos a las necesidades del contexto de la selección de campeones. La matriz de decisión que utilizamos está estructurada con los campeones que el usuario introduce como alternativas, dispuestos en las filas, mientras que en las columnas se encuentran los criterios que consideramos claves para la evaluación, como el porcentaje de victoria, el porcentaje de selección y el porcentaje de baneo.

Dado que los tres criterios son porcentajes que varían entre 0 y 100, es necesario normalizar estos valores para que oscilen entre 0 y 1. Este proceso de normalización es fundamental para asegurar que todos los criterios sean comparables. Además, otro aspecto importante en la aplicación del método TOPSIS es la asignación de un peso a cada criterio, reflejando así su importancia en la toma de decisiones.

Una vez normalizados los valores, se procede a ponderar la matriz de decisión. Esto se realiza multiplicando cada valor normalizado por el peso asignado al criterio correspondiente, lo que ajusta la influencia de cada criterio según su relevancia. Con la matriz ya ponderada, se calcula la solución ideal y no ideal. Estas soluciones representan, respectivamente, el escenario más favorable y el menos favorable.

El siguiente paso en el proceso es calcular la distancia de cada campeón a la solución ideal y no ideal. Esta distancia determina qué tan cerca o lejos está cada alternativa de ambas opciones teóricas. Finalmente, se calcula el coeficiente de proximidad para cada alternativa. Este coeficiente es un número entre 0 y 1 que indica el grado de cercanía de cada campeón a la solución ideal. Un valor más cercano a 1 sugiere que la alternativa está más alineada con la solución óptima, haciendo de ese campeón la elección recomendada.

Ejemplo de aplicación del método TOPSIS

Para ilustrar de manera práctica cómo se aplica el método TOPSIS en el contexto de la selección de campeones, se presenta un ejemplo concreto utilizando datos reales. Este ejemplo permitirá visualizar cada uno de los pasos descritos previamente. Supongamos que un usuario desea encontrar el campeón más equilibrado para la posición de jungla. Para ello, introduce 5 campeones que suelen desempeñar ese rol: Amumu, Nocturne, Zyra, Lee Sin y Sejuani. A continuación, se muestran los datos y los pasos seguidos en la aplicación de este método:

1. **Construcción de la matriz de decisión:** Las filas representan las alternativas y las columnas los criterios.

Campeón	Win rate	Pick rate	Ban rate
Amumu	52.18 %	6.45 %	3.24 %
Zyra	47.76 %	1.04 %	0.97 %
Nocturne	51.94 %	5.23 %	6.51 %
Sejuani	48.73 %	3.52 %	0.52 %
Lee Sin	49.63 %	20.23 %	14.93 %

Tabla. 4.1: Matriz de decisión con campeones en la posición de jungla

2. **Normalización de la matriz de decisión:** Se normaliza la matriz usando la fórmula vista en la ecuación 2.1.

Campeón	Win rate	Pick rate	Ban rate
Amumu	0.465	0.29	0.194
Zyra	0.426	0.046	0.058
Nocturne	0.463	0.235	0.391
Sejuani	0.435	0.158	0.031
Lee Sin	0.443	0.912	0.897

Tabla. 4.2: Matriz normalizada con campeones en la posición de jungla

3. **Construcción de la matriz de decisión ponderada:** Se multiplica el valor normalizado por el peso del criterio. En este ejemplo se asigna un peso de 0.7 al porcentaje de victoria, un 0.15 al porcentaje de selección y un 0.15 a la tasa de baneo.

Campeón	Win rate	Pick rate	Ban rate
Amumu	0.326	0.043	0.029
Zyra	0.298	0.007	0.008
Nocturne	0.324	0.035	0.058
Sejuani	0.304	0.023	0.004
Lee Sin	0.310	0.136	0.134

Tabla. 4.3: Matriz ponderada con campeones en la posición de jungla

4. **Determinación de PIS y NIS:**
   - **PIS:** Se selecciona el valor máximo para el porcentaje de victoria y de selección. Para la tasa de baneo se elige el menor valor.
     - **Win rate:** 0.326 (Amumu)
     - **Pick rate:** 0.136 (Lee Sin)
     - **Ban rate:** 0.004 (Sejuani)
   - **NIS:** Se selecciona el valor mínimo para el porcentaje de victoria y de selección. Para la tasa de baneo se elige el mayor valor.
     - **Win rate:** 0.298 (Zyra)
     - **Pick rate:** 0.007 (Zyra)
     - **Ban rate:** 0.134 (Lee Sin)
5. **Cálculo de las distancias:** Se calcula la distancia de cada alternativa a la solución ideal y no ideal siguiendo la ecuación 2.5 y 2.6.
   - **Amumu:**
     - **Distancia PIS:** 0.00929
     - **Distancia NIS:** 0.0132
   - **Zyra:**
     - **Distancia PIS:** 0.01763
     - **Distancia NIS:** 0.01583
   - **Nocturne:**
     - **Distancia PIS:** 0.01321
     - **Distancia NIS:** 0.00724
   - **Sejuani:**
     - **Distancia PIS:** 0.01324
     - **Distancia NIS:** 0.01718
   - **Lee Sin:**
     - **Distancia PIS:** 0.01712
     - **Distancia NIS:** 0.01698

6. **Cálculo de los coeficientes de proximidad:** Se calculan los coeficientes de proximidad utilizando la ecuación 2.7.

Campeón	Coeficientes de proximidad
Amumu	0.5438
Zyra	0.4865
Nocturne	0.4254
Sejuani	0.5325
Lee Sin	0.4990

Tabla. 4.4: Matriz con los coeficientes de proximidad de los campeones

7. **Ordenación:** Se ordenan los campeones basados en su coeficiente de proximidad. En este caso, el mejor campeón es Amumu, seguido de Sejuani.

Campeón	Coeficientes de proximidad
Amumu	0.5438
Sejuani	0.5325
Lee Sin	0.4990
Zyra	0.4865
Nocturne	0.4254

Tabla. 4.5: Matriz ordenada con los resultados

Capítulo 5

CONCLUSIONES

En este capítulo de conclusiones, se realizará un análisis crítico de los resultados obtenidos a lo largo del proyecto. Se discutirán los principales desafíos y problemas encontrados durante el proceso, evaluando cómo afectaron al desarrollo de la aplicación y las soluciones implementadas para superarlos. Además, se identificarán las limitaciones del trabajo realizado, proponiendo posibles mejoras para futuros desarrollos que podrían mejorar la herramienta o añadir funcionalidades.

5.1. Trabajo realizado

Tras la conclusión del desarrollo de este proyecto, se puede afirmar que la creación de la aplicación web ha cumplido satisfactoriamente los objetivos inicialmente propuestos. Desde el inicio, la idea principal de este proyecto fue ofrecer a los jugadores de League of Legends una herramienta útil y eficiente que pudieran emplear durante sus partidas. Gracias a las funcionalidades implementadas, se ha logrado no solo crear una herramienta que facilita la selección inteligente de campeones, sino también añadir otras características que porporcionan un valor añadido a la experiencia del usuario.

El desarrollo de la aplicación comenzó con una fase de investigación preliminar, donde fue necesario familiarizarse con diversas tecnologías y metodologías fundamentales para el proyecto. Por ejemplo, se dedicó tiempo a explorar y entender tecnologías inicialmente desconocidas como el web scraping, así como métodos de tomas de decisiones como el TOPSIS. Sin embargo, no fue necesario realizar investigación sobre el juego en sí dado que ya poseía un conocimiento sólido sobre su mecánica y

estructura.

Tras esta fase de investigación, se llevaron a cabo pruebas enfocadas en la adquisición de datos [33]. Gracias a estas pruebas se logró una base para desarrollar una versión funcional para la obtención de datos para la aplicación web. Posteriormente, se procedió a la implementación de la aplicación y al diseño de la interfaz de usuario. Se hizo hincapié en crear una interfaz accesible y fácil de usar, con el objetivo de que cualquier jugador pudiera navegar de manera sencilla independientemente de su nivel de experiencia.

Una vez que la interfaz de la aplicación y el proceso de obtención de datos estuvieron listos, se procedió a implementar las funcionalidades descritas en el subapartado 1.2.2. Estas funcionalidades han sido diseñadas para ofrecer una experiencia de usuario completa, asegurando que cumpla las expectativas y necesidades de los usuarios.

En resumen, este proyecto ha representado una valiosa oportunidad para consolidar y aplicar los conocimientos adquiridos a lo largo del Grado en Ingeniería Informática. Lo que inicialmente era una idea abstracta ha culminado en la satisfacción de ver cómo esa idea se ha ido materializando día a día hasta convertirse en una herramienta completamente funcional.

5.2. Dificultades encontradas

A lo largo del desarrollo del proyecto, las principales dificultades encontradas han estado más relacionadas con el aspecto teórico que con la parte práctica. Si bien la implementación técnica ha presentado ciertos retos, el verdadero desafío ha sido comprender en profundidad los conceptos que sustentan el proyecto.

En primer lugar, fue necesario familiarizarse con las diversas tecnologías involucradas. Aunque algunas de ellas eran ya conocidas, otras eran completamente nuevas. En particular, la teoría detrás del método TOPSIS y la toma de decisiones requirió un estudio intenso para entender sus fundamentos.

A nivel de código, uno de los mayores retos estuvo relacionado con el web scraping. La recolección de datos de páginas web implicó enfrentar dificultades con el reconocimiento de las estructuras de los sitios web, la identificación de los elementos correctos y la manipulación eficiente de la información extraída.

La implementación del método TOPSIS también presentó dificultades ya que, aunque la teoría había sido comprendida, llevarla a cabo mediante código resultó un proceso complejo. Las principales trabas estuvieron relacionadas con los cálculos necesarios, la normalización de datos y la adecuación a la estructura de la información disponible.

5.3. Pruebas en usuarios

Para validar la usabilidad de la aplicación web desarrollada, se realizaron pruebas con un grupo diverso de usuarios. Estos participantes fueron seleccionados para representar una variedad de perfiles, lo que permitió evaluar la eficacia de la aplicación desde distintas perspectivas. El grupo de usuarios incluía personas que conocían League of Legends aunque no jugasen activamente, jugadores frecuentes del juego y usuarios que no tenían nigún conocimiento previo sobre él.

Las pruebas consistieron en dos fases. En la primera fase, los usuarios navegaron libremente por la aplicación. Se les permitió explorar por todas las secciones disponibles y probar distintas funcionalidades. Esta fase tenía como objetivo familiarizar a los usuarios con la interfaz. En la segunda fase, se llevó a cabo una simulación de la fase de selección de campeones, replicando el límite de tiempo de 30 segundos que tienen los jugadores en League of Legends. Se utilizó un cronómetro para simular esta presión temporal y se pidió a los usuarios que introdujeran los campeones que quisieran en la herramienta de selección inteligente de campeones.

Para reflejar la satisfacción de los usuarios, se realizó una encuesta con varias preguntas sobre la aplicación. Las respuestas de la encuesta se midieron en una escala de 0 a 5, donde 0 indica el nivel más bajo de satisfacción y 5 representa el nivel más alto.

Pregunta	U1	U2	U3	U4	U5
¿Cómo valorarías la interfaz de la aplicación?	4	3	4	2	4
¿Te ha parecido útil la información mostrada?	5	4	5	5	4
¿Te ha resultado sencillo navegar por la aplicación?	5	4	5	5	5
¿Te ha parecido útil la herramienta de selección de campeones?	3	5	5	4	5
¿Te ha dado tiempo a introducir todos los datos necesarios?	2	5	5	5	3
¿Usarías la apliación web?	3	5	4	4	5

Tabla. 5.1: Encuesta realizada a los usuarios

Los resultados de las pruebas fueron en general positivos, incluso entre los usuarios que no estaban familiarizados con el juego. Las pruebas indican que la aplicación web es usable y cumple con los objetivos establecidos.

5.4. Trabajos futuros

A pesar de que el desarrollo del proyecto ha sido un éxito, siempre existe margen para seguir perfeccionando y ampliando sus funcionalidades. Estas mejoras no solo optimizarían la experiencia del usuario, sino que también mejorarían la eficiencia de la aplicación.

Por lo tanto, a continuación se presenta un listado con posibles mejoras y desarrollos futuros que podrían ser abordados en fases posteriores del proyecto.

Obtención independiente de los datos

Como se describió en el apartado 4.3, la obtención de datos en el proyecto depende actualmente de la página de OP.GG. Aunque esta dependencia no presenta un problema significativo en la actualidad, es importante considerar las posibles limitaciones y riesgos asociados a depender de un tercero para acceder a la información que alimenta la aplicación web. Si en algún momento OP.GG modifica su interfaz, sufre caídas o restringe el acceso a ciertos datos podría afectar negativamente al rendimiento de nuestra aplicación.

Una opción para superar esta dependencia es la obtención directa de datos desde Riot Games, lo que ofrecería una mayor autonomía y control sobre la información. Riot Games proporciona una cuenta de desarrollador a aquellos usuarios que deseen acceder a los datos oficiales proporcionados por la empresa. Al aprovechar esta oportunidad, se podrían obtener los datos desde una fuente directa, asegurando la consistencia y precisión de la información.

Aumentar la eficiencia de la obtención de datos

Aunque el proceso de obtención de datos es invisible para el usuario final, quien siempre tiene acceso a la información, la realidad es que el proceso de web scraping utilizado para recopilar estos datos es complejo y requiere un tiempo considerable. Este tiempo de procesamiento no afecta a la experiencia del usuario directamente, sin embargo, para el desarrollador de la aplicación, el tiempo necesario para ejecutar este código puede resultar molesto.

La necesidad de actualizar los datos con frecuencia es sumamente importante pero este proceso se complica por la gran cantidad de datos que maneja la aplicación. Dado que la cantidad de campeones y detalles que deben ser extraídos son factores que no podemos modificar, una posible solución para mejorar la eficiencia de este proceso sería paralelizar [34] la ejecución de código encargado de obtener los datos. Al dividir la tarea en múltiples hilos que se ejecutan simultáneamente, podríamos reducir el tiempo necesario para completar la recolección de datos.

Adición de criterios adicionales

En el método TOPSIS, los criterios son fundamentales para tomar decisiones informadas y precisas. En el proyecto actual se implementan tres criterios como el porcentaje de victoria, el de selección y el de baneo. Estos criterios ofrecen una base para evaluar y seleccionar el campeón más adecuado en función de las estadísticas disponibles, sin embargo, hay margen para ampliar y enriquecer este método mediante la incorporación de criterios adicionales.

Una posible mejora sería la adición de un criterio que evalúe la composición del equipo propio. Este criterio podría tener en cuenta la necesidad de equilibrar el daño físico y mágico y equilibrar los roles específicos del equipo para que tenga la diversidad necesaria.

Otra mejora sería incluir una valoración del equipo rival. Este criterio podría analizar la composición del equipo rival y sugerir campeones que contrarresten las fortalezas del oponente.

La adición de estos criterios no solo haría que nuestra herramienta sea más completa, sino que también permitiría a los usuarios tomar decisiones más informadas, considerando un mayor número de variables.

Añadir un registro de usuarios

Una mejora para el proyecto sería implementar un sistema de registro y autenticación de usuarios. Esta funcionalidad permitiría a los usuarios crear cuentas, iniciar sesión y gestionar su información personal dentro de la aplicación.

Cada cuenta de usuario podría almacenar un historial de campeones buscados, lo cual permitiría a los usuarios acceder rápidamente a los campeones que han buscado previamente, evitando la necesidad de repetir búsquedas. Además, podría incluir detalles adicionales como un historial de los cálculos de los mejores campeones anteriores para evitar tener que repetir el proceso.

Apéndice A

Manual de instalación

El repositorio que contiene el código de la aplicación web está disponible en el siguiente enlace: https://github.com/amm00337/TFG.git

A continuación, se muestran los pasos necesarios para ejecutar el proyecto de manera local.

A.1. IntelliJ

1. Acceder a la página oficial de descargas: https://www.jetbrains.com/es-es/idea/
2. Descargar el instalador y seguir los pasos que indica.

Figura A.1: Página de descarga de IntelliJ. Elaboración propia

A.2. Node.js

1. Acceder a la página oficial de descargas: https://nodejs.org/en/
2. Descargar el instalador y seguir los pasos que indica.

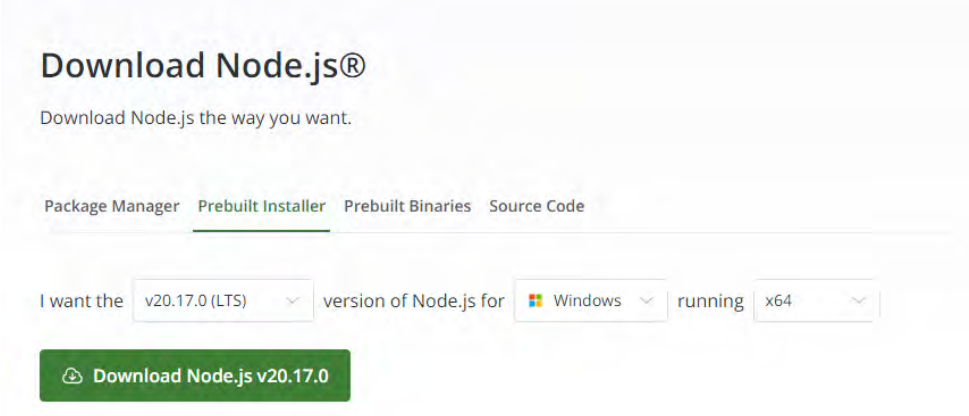

Figura A.2: Página de descarga de Node.js. Elaboración propia

A.3. Puppeteer

1. Acceder a la página oficial de descargas: https://pptr.dev/guides/installation
2. Seguir los pasos que indica, ejecutando el código en la consola.

A.4. GitHub

1. Acceder al repositorio: https://github.com/amm00337/TFG.git
2. Descargar el código en un ZIP comprimido.

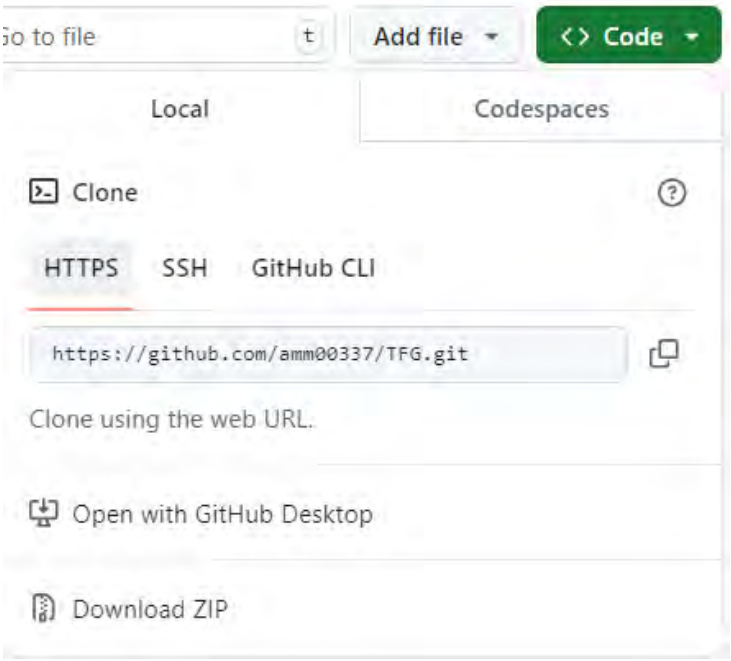

Figura A.3: Descarga del código en GitHub. Elaboración propia

3. Una vez descargado, abrir el proyecto en IntelliJ y ejecutar el archivo `index.html` en el navegador de preferencia del usuario.

Apéndice B

Manual de usuario

La aplicación web cuenta con varias secciones clave que permiten al usuario interactuar de manera eficiente con la plataforma. Entre estas secciones se incluyen la página principal, la tier list, el listado de campeones, la página específica de cada campeón y la herramienta de selección de campeones. Estas funcionalidades están diseñadas para ofrecer una experiencia de usuario completa, facilitando el acceso a la información de forma sencilla.

Además, cuenta con un campo de búsqueda intuitivo, donde el usuario puede escribir el nombre del campeón y visualizar tanto su nombre como su imagen para facilitar la identificación.

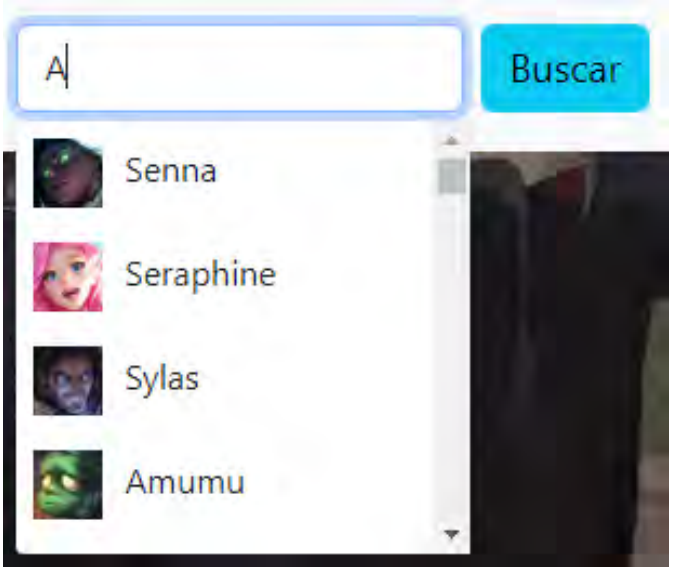

Figura B.1: Interfaz del campo de búsqueda. Elaboración propia

B.1. Página principal

La página principal de la aplicación web tiene un diseño sencillo que actúa como punto de partida. Desde aquí, los usuarios pueden navegar libremente por las distintas secciones disponibles.

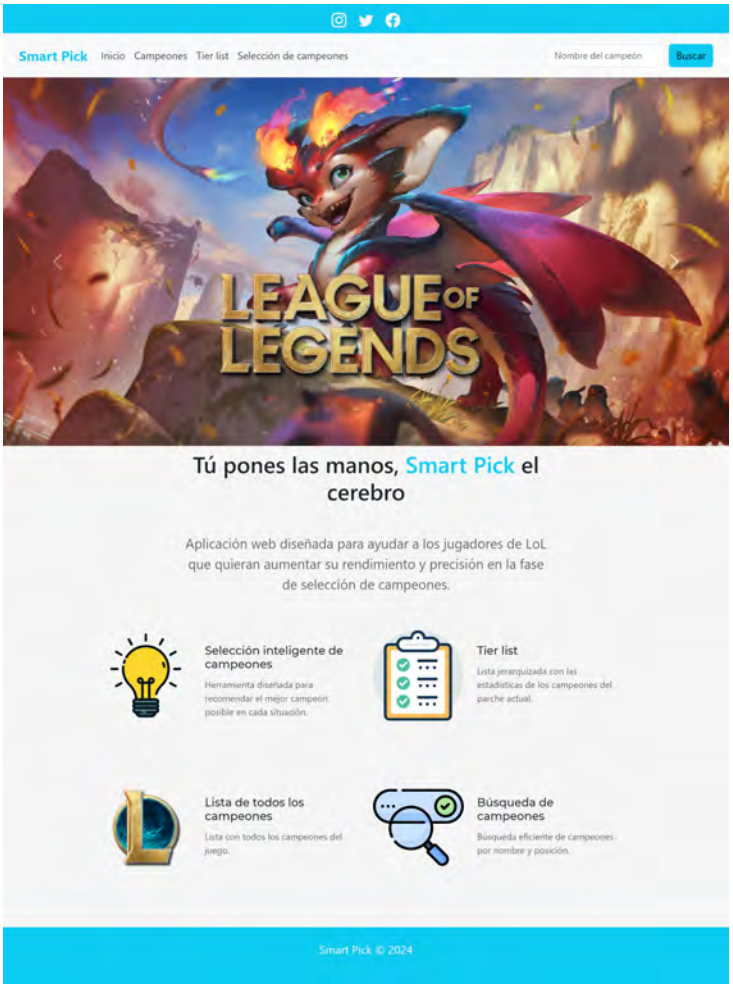

Figura B.2: Interfaz de la página principal. Elaboración propia

En la parte central de la página, se incluye una breve descripción que resume las herramientas y funcionalidades que contiene la aplicación.

B.2. Listado de campeones

La sección del listado de campenes ofrece a los usuarios una visión completa de todos los personajes disponibles en el juego, presentando sus nombres e imágenes en una lista organizada. Este diseño permite identificar rápidamente el campeón que el usuario está buscando, ahorrando tiempo y mejorando la experiencia de navegación.

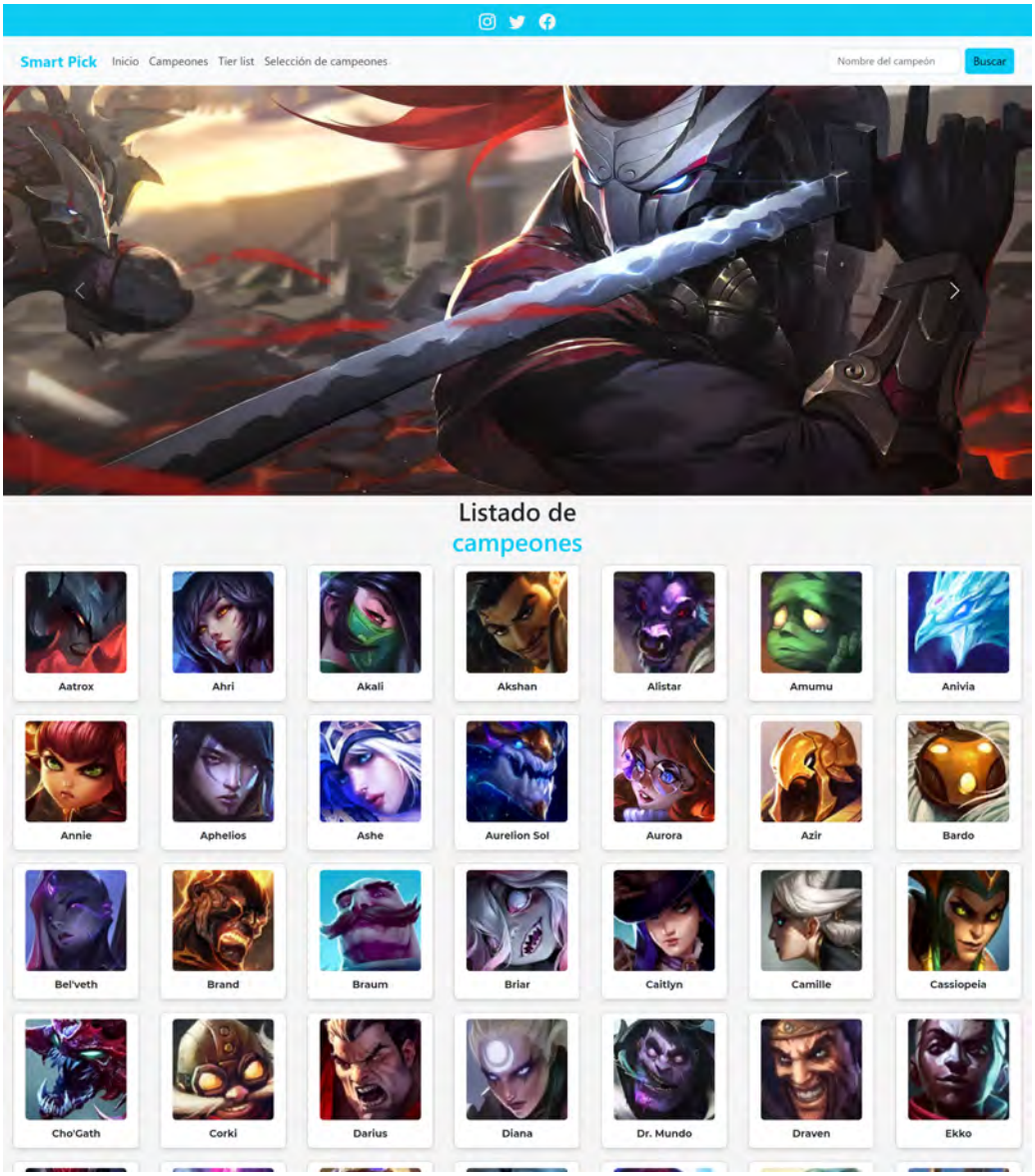

Figura B.3: Interfaz del listado de campeones. Elaboración propia

Al seleccionar un campeón, el usuario es redirigido a la página específica del campeón, donde puede consultar estadísticas detalladas y relevantes.

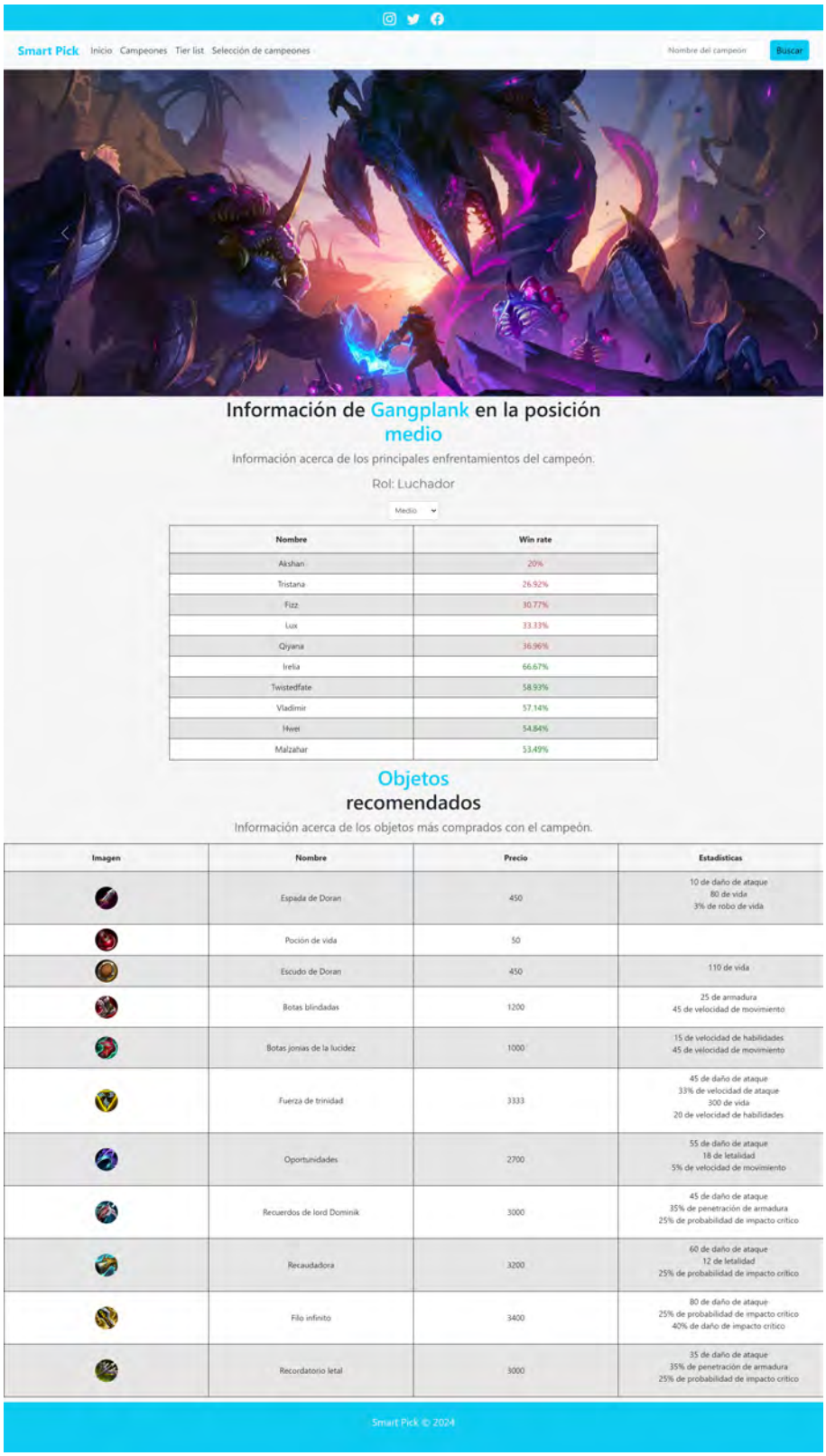

Figura B.4: Interfaz de la página específica de cada campeón. Elaboración propia

Otra funcionalidad importante de esta página es la presentación de los objetos que se suelen comprar con el campeón seleccionado. De esta manera, el usuario puede resolver dudas sobre el equipamiento ideal y los enfrentamientos más beneficiosos o

perjudiciales para su campeón.

B.3. Tier list

La tier list es una lista jerarquizada de los campeones en función de su rendimiento y popularidad en el parche actual. Cada fila de la lista incluye el nombre, imagen y estadísticas del campeón, como porcentaje de victoria, porcentaje de selección y de baneo.

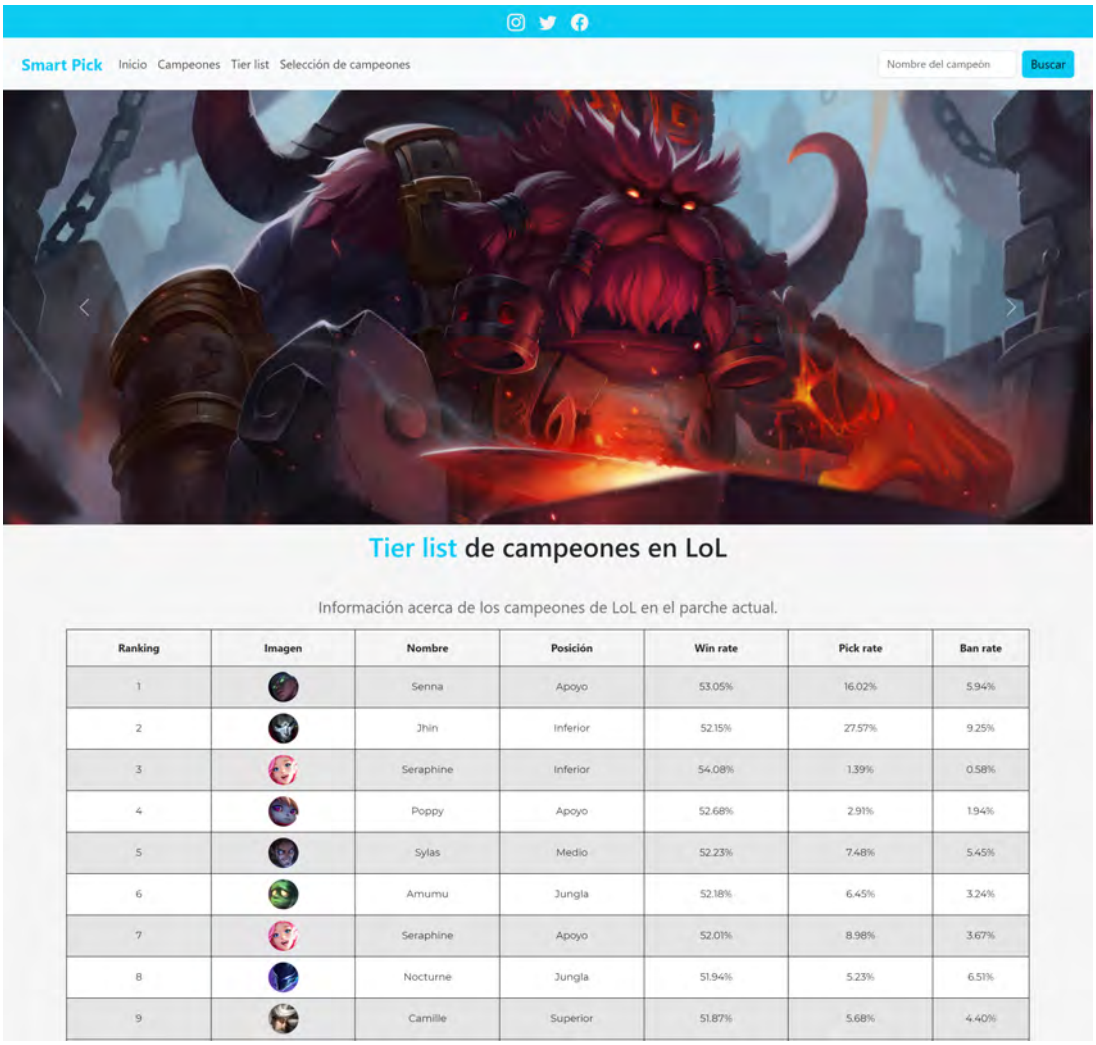

Figura B.5: Interfaz de la tier list. Elaboración propia

El orden de la lista está estructurado de forma descendente, lo que significa que los campeones en las posiciones más elevadas son considerados como los más fuertes o prioritarios en el parche actual.

B.4. Herramienta de selección de campeones

La herramienta de selección inteligente de campeones permite a los usuarios tomar decisiones estratégicas informadas al seleccionar campeones para sus partidas. El usuario puede introducir los campeones que desea jugar, seleccionándolos por posición. Opcionalmente, puede introducir información sobre el campeón rival con el que se enfrenta. Con estos datos, la herramienta utiliza métodos de decisión para recomendar el mejor campeón.

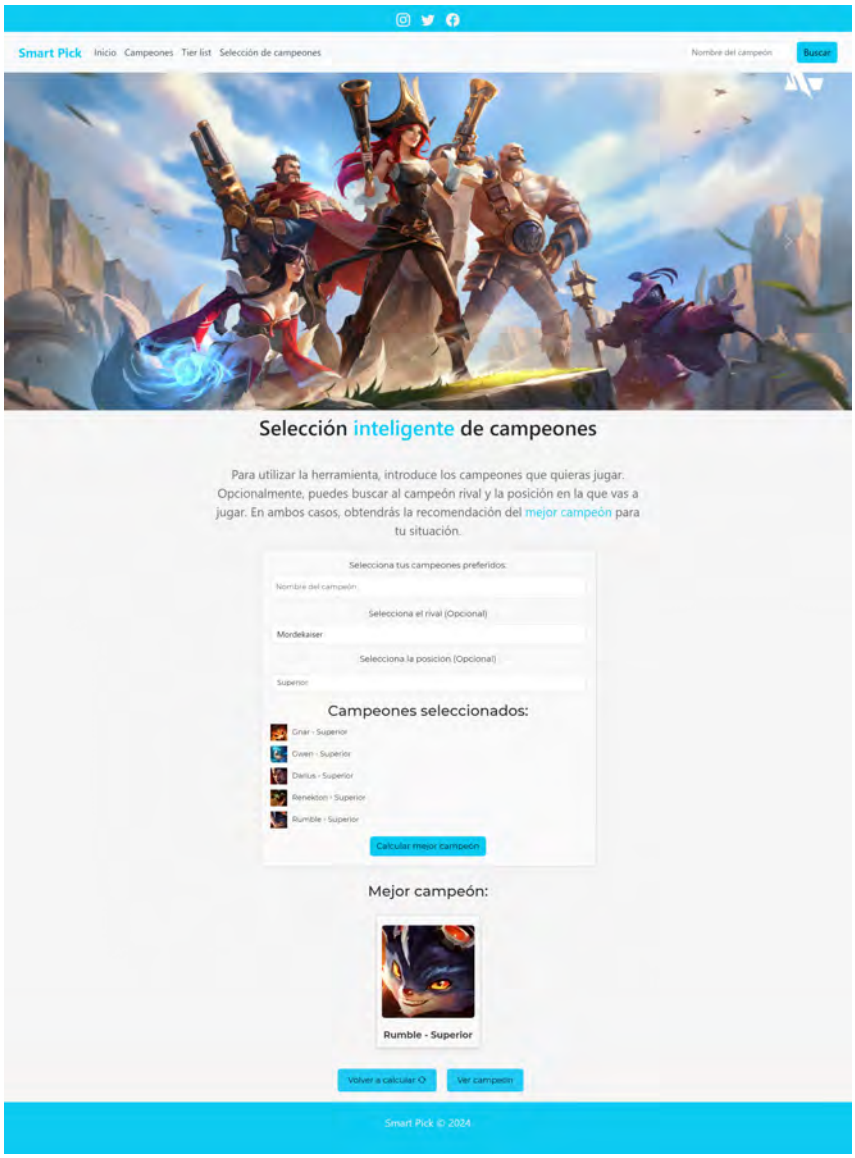

Figura B.6: Interfaz de la herramienta de selección de campeones. Elaboración propia

Esta herramienta es sencilla e intuitiva, diseñada para ser usada en tiempo real durante la fase de selección de campeones.

Bibliografía

[1] Esportmaniacos. Imagen de league of legends, 2024. URL https://www.esportmaniacos.com/wp-content/uploads/2022/03/Leage-of-Legends-780x470.webp. Comprobado en 2024-09-03.

[2] 3D Juegos. Imagen de campeones en league of legends, 2024. URL https://i.blogs.es/f75074/league-legends-seguira-constante-actualizacion_1464463551_673787_1440x810/840_560.jpeg. Comprobado en 2024-09-03.

[3] 3D Juegos. Imagen del parche 13.1, 2024. URL https://images.contentstack.io/v3/assets/blt731acb42bb3d1659/blt4e7caa12ca1ac28b/63bd4c0b7a6762171d82184c/LOL_13.1_Patch-Highlights_TW_1920x1080_TTan_v01_ES.jpg?disposition=inline. Comprobado en 2024-09-03.

[4] League of Legends Wiki. Imagen del mapa de lol, 2024. URL https://static.wikia.nocookie.net/leagueoflegends/images/5/53/Summoner%27s_Rift_Update_Map.png/revision/latest?cb=20170223053555. Comprobado en 2024-09-03.

[5] Git Hub. Matriz de decisión, 2024. URL https://user-images.githubusercontent.com/44522286/131339802-c16bb37c-6479-40e5-b28f-b79471b9bb26.png. Comprobado en 2024-09-03.

[6] Angular JS India. Comparativa entre frontend y backend, 2024. URL https://www.angularjsindia.com/blog/wp-content/uploads/2021/06/backend-front-end.png. Comprobado en 2024-09-03.

[7] Icon Duck. Logo de intellij, 2024. URL https://static-00.iconduck.com/assets.00/intellij-idea-icon-2048x2048-hsyna1mi.png. Comprobado en 2024-09-03.

[8] PNGEgg. Logo de html, 2024. URL https://e7.pngegg.com/pngimages/913/851/png-clipart-responsive-web-design-html-world-wide-web-consortium-world-wide-w
png. Comprobado en 2024-09-03.

[9] Logos World. Logo de javascript, 2024. URL https://logos-world.net/wp-content/uploads/2023/02/JavaScript-Symbol.png. Comprobado en 2024-09-03.

[10] PNGEgg. Logo de css, 2024. URL https://e7.pngegg.com/pngimages/603/759/png-clipart-css3-cascading-style-sheets-logo-html-world-wide-web-blue-angle-thum
png. Comprobado en 2024-09-03.

[11] Icon Duck. Logo de node.js, 2024. URL https://static-00.iconduck.com/assets.00/puppeteer-icon-1371x2048-otngklvq.png. Comprobado en 2024-09-03.

[12] Puppeteer. Documentación de puppeteer, 2024. URL https://pptr.dev/guides/getting-started. Comprobado en 2024-09-03.

[13] Millenium. Fase de selección de campeones, 2024. URL https://static1-es.millenium.gg/articles/2/34/13/2/@/158391-seleccion-campeon-lol-article_image_bd-1.jpg. Comprobado en 2024-09-03.

[14] Diego Martín Muñoz and Luis Miguel Pedrero Esteban. Deporte y espectáculo en la narrativa de los 'e-sports': El caso de 'league of legends'. *Index. comunicación: Revista científica en el ámbito de la Comunicación Aplicada*, 11(2):59–79, 2021.

[15] Marçal Mora-Cantallops and Miguel-Ángel Sicilia. Moba games: A literature review. *Entertainment computing*, 26:128–138, 2018.

[16] Yaoyao Sun. Motivation to play esports: case of league of legends. Master's thesis, University of South Carolina, 2017.

[17] Tina Lynn Taylor. *Raising the stakes: E-sports and the professionalization of computer gaming*. Mit Press, 2012. ISBN 978-0-262-01737-4.

[18] Choong-Soo Lee and Ivan Ramler. Identifying and evaluating successful non-meta strategies in league of legends. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–6, 2017.

[19] Athanasios Kokkinakis, Peter York, Moni Sagarika Patra, Justus Robertson, Ben Kirman, Alistair Coates, Alan Pedrassoli Pedrassoli Chitayat, Simon Demediuk, Anders Drachen, Jonathan Hook, et al. Metagaming and metagames in esports. *International Journal of Esports*, 1(1), 2021.

[20] Kathleen M Eisenhardt and Mark J Zbaracki. Strategic decision making. *Strategic management journal*, 13(S2):17–37, 1992.

[21] K Paul Yoon and Ching-Lai Hwang. *Multiple attribute decision making: an introduction*. Sage publications, 1995. ISBN 0-8039-5486-7.

[22] David L Olson. Comparison of weights in topsis models. *Mathematical and Computer Modelling*, 40(7-8):721–727, 2004.

[23] Majid Behzadian, Reza Baradaran Kazemzadeh, Amir Albadvi, and Mohammad Aghdasi. Promethee: A comprehensive literature review on methodologies and applications. *European journal of Operational research*, 200(1):198–215, 2010.

[24] C. L. Hwang and K. Yoon. *Multiple Attribute Decision Making: Methods and Applications*. CRC press, 2011. ISBN 978-1-4398-6157-8.

[25] RM Zulqarnain, M Saeed, N Ahmad, F Dayan, and B Ahmad. Application of topsis method for decision making. *Int. J. Sci. Res. in Mathematical and Statistical Sciences*, 7(2), 2020.

[26] Sergio Luján-Mora. *Programación de aplicaciones web: historia, principios básicos y clientes web*. Editorial Club Universitario, 2002. ISBN 978-84-8454-206-3.

[27] Bo Zhao. Web scraping. *Encyclopedia of big data*, 1, 2017.

[28] Vlad Krotov, Leigh Johnson, and Leiser Silva. Tutorial: Legality and ethics of web scraping. *Communications of the Association for Information Systems*, 2020.

[29] Equipo Vértice et al. *Diseño básico de páginas web en HTML*. Editorial Vértice, 2009. ISBN 978-84-9931-034-3.

[30] Juan Diego Gauchat. *El gran libro de HTML5, CSS3 y Javascript*. Marcombo, 2012. ISBN 978-84-267-1770-2.

[31] Basarat Syed. *Beginning Node. js*. Apress, 2014. ISBN 978-1-4842-0188-6.

[32] Kazuaki Matsuo. Testing with puppeteer, a complete guide, 2022. URL https://www.headspin.io/blog/testing-with-puppeteer-a-complete-guide. Comprobado en 2024-09-03.

[33] To Scrape. To scrape, 2024. URL https://toscrape.com/. Comprobado en 2024-09-03.

[34] Tamás Sallai. How to speed up puppeteer scraping with parallelization, 2021. URL https://advancedweb.hu/how-to-speed-up-puppeteer-scraping-with-parallelization/. Comprobado en 2024-09-03.