UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Desarrollo de una plataforma web de análisis y asistencia para el videojuego League of Legends

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Sancho Ángel, Vicente

Tutor/a: Valderas Aranda, Pedro José

CURSO ACADÉMICO: 2022/2023

Resumen

El trabajo consiste en el desarrollo de una página web que proporciona información sobre distintos aspectos del videojuego League of Legends, de este modo pretende ser una guía del mismo que los jugadores pueden utilizar para analizar su rendimiento y aprender. Por ello se quiere dar un análisis de estadísticas que pueda resultar útil a los jugadores, así como recomendaciones basadas en el desempeño de los jugadores más experimentados y un historial de partidas propio en el que pueda ver su rendimiento individual entre otras.

Para ello se hará uso de la API que nos proporciona el desarrollador del videojuego, de este modo podremos interactuar con la base de datos del juego y podremos dotar a la web de distintas funcionalidades, eligiendo las funcionalidades más útiles de las webs del mismo tipo ya existentes pero además añadiendo otras nuevas.

Palabras clave: League of Legends, análisis de estadísticas, API.

Abstract

This work consists of the development of a website that provides information on different aspects of the League of Legends videogame, pretending to be a guide to it, that players can use to analyze their performance and learn. For this reason, we want to provide an analysis of statistics that can be useful to players, as well as recommendations based on the performance of the most experienced players and their own game history where they can see their individual performance.

In order to achieve this we will use the API provided by the videogame developer, thanks to this we will be able to interact with the game's database and we will be able to provide the website with different functionalities, choosing the most useful functionalities of the existing websites of the same type, but also adding some new ones.

Keywords : League of Legends, analysis of statistics, API.

Tabla de contenidos

Glosario

Este glosario contiene definiciones de ciertos conceptos que van a aparecer a lo largo del trabajo y que se entiende que no son de saber popular. Principalmente contiene conceptos acerca del videojuego League of Legends del cual trata el trabajo, pero también se pueden encontrar definiciones sobre conceptos de servicios web.

1. **API:** acrónimo de Application Programming Interface, es un conjunto de reglas y protocolos que permite la comunicación y compartición de datos entre diferentes aplicaciones o sistemas.
2. **API Key:** o clave de una API, es un identificador que sirve para la autenticación de un usuario para el uso de un servicio.
3. **Backend:** es la parte de un sistema que se encarga de la lógica y el procesamiento de los datos, siendo invisible para los usuarios.
4. **Frontend:** es la parte de un sistema que se comunica con los usuarios del mismo, permitiendo que interactúen con el sistema.
5. **REST:** acrónimo de Representational State Transfer, es un estilo de arquitectura utilizado en el diseño de servicios web basado en un conjunto de principios y restricciones que permiten una arquitectura simple y escalable.
6. **LoL:** acrónimo para referirse al videojuego League of Legends muy usado por la comunidad.
7. **Campeón:** nombre que se le da a los personajes del juego LoL.
8. **Ban:** traduciendo del inglés “prohibición”, hace referencia a la fase de elección de personajes del juego LoL, dónde los jugadores pueden prohibir qué personajes se van a jugar en una partida.
9. **Cola:** se refiere al modo de emparejamiento de jugadores dentro del juego. Actualmente existen dos tipos de colas competitivas: “Solo Q”, dónde puedes jugar solo o con un amigo; y “Flex Q” dónde pueden jugar hasta 5 amigos simultáneamente.
10. **Rango:** está estrictamente relacionado con tu nivel como jugador y es la motivación principal para jugar partidas competitivas. Cuanto más alto sea tu rango, más alto debería ser tu nivel por lo que los jugadores buscan llegar al rango más alto posible. Actualmente existen 9 rangos y cada uno de ellos se divide en 4 ligas.
11. **Puntos de Liga:** o también abreviados como LPs, son el sistema de puntos utilizados para subir o bajar de rangos. Van de 0 a 100 y si sales de este rango cambiaras consecuentemente de liga.

12. **Challenger:** es el más alto rango que existe en el videojuego y lo forman los 300 jugadores con más puntos de liga del servidor.
13. **Regiones mayoritarias:** es un concepto utilizado únicamente en el ambiente competitivo del videojuego y se emplea para nombrar a las regiones del mundo que contienen un mayor número de jugadores y dónde supuestamente el nivel de los servidores es mayor que el resto. Estas regiones son: China, Corea, Norteamérica y Europa.

1. Introducción

Todos somos conscientes del constante crecimiento que está viviendo la industria de los videojuegos, a pesar de que en sus inicios fuera un mercado bastante pequeño, en la actualidad empresas multimillonarias son las que se mueven por este mercado cuyo crecimiento no parece tener límite. Este crecimiento de popularidad fue lo que despertó el interés de muchos, y por lo que cada vez podemos ver más y más productos relacionados con los propios videojuegos. Estamos hablando no solo de los videojuegos como producto en sí mismo, sino todo lo que tiene relación con ellos, como por ejemplo Software.

Como se ha comentado anteriormente, con el aumento de la popularidad de la industria comenzaron a aumentar los lanzamientos de nuevos videojuegos, y fue en 2009 cuando el videojuego League of Legends (LoL) fue lanzado. Estamos hablando de uno de los juegos más exitosos de la historia, siendo capaz de mantenerse hoy en día después de más de 13 años en activo. Debido a esta popularidad, no tardó en sumarse al resto de juegos que tenían competiciones oficiales que hoy en día conocemos como e-Sports, siendo en la actualidad probablemente el videojuego más influyente en cuanto a e-Sports se refiere.

La existencia de estas competiciones es la atracción principal por la que los jugadores desean llegar a lo más alto dentro del juego, pero hay una principal problemática, mejorar por tu propia cuenta es realmente difícil y muy pocas personas tienen esta capacidad. Podemos ver el símil en el deporte tradicional, donde es indispensable un entrenador, por lo que es necesario buscar un sustituto del mismo si queremos mejorar realmente. Es en este contexto es en el que entran las herramientas software que proporcionan soporte a los jugadores, a pesar de que existen otras muchas opciones que ayudan a los jugadores a mejorar, una herramienta de este tipo es indispensable en cualquier caso, podemos confirmarlo ya que absolutamente todos los jugadores profesionales hacen uso de al menos una de las herramientas existentes.

Tanta es la importancia de estas herramientas para los jugadores que la propia desarrolladora del juego pone facilidades a todo aquel que quiera desarrollar una, dejando a disposición de los programadores una API[1] con la que se pueden realizar consultas a la base de datos del juego y un programa de ayuda para desarrolladores en el que los propios empleados de la empresa pueden ver tu producto y ayudarte con el mismo.

1.1 Objetivos

El objetivo de este trabajo es desarrollar una herramienta que sea capaz de otorgar a sus usuarios las capacidades básicas para analizar su rendimiento dentro del videojuego, además de suponer una ayuda significativa para el aumento del rendimiento y expandir el conocimiento del propio juego. Además, el producto debe ser amigable para los nuevos usuarios, disponiendo de una interfaz sencilla de comprender para aquellas personas que no disponen de altos conocimientos

1.2 Metodología

La metodología empleada en el desarrollo de este proyecto se trata de Desarrollo Centrado en el Usuario (DCU). Esta metodología viene recogida y explicada en la ISO 9241-210[2] publicada en el 2019, que es una norma internacional que establece los requisitos y recomendaciones para los principios y actividades de diseño centrado en el operador humano para los sistemas interactivos. El objetivo principal de esta norma es mejorar la calidad de la interacción entre el usuario y el sistema.

Para llevar a cabo esta forma de trabajo, la ISO especifica que el proceso debe ser iterativo y se debe involucrar a los usuarios finales del producto para así desarrollar un resultado que sea acorde a las necesidades de los mismos.

Además, esta norma ISO define siete fases que deberían formar los procesos DCU, estas son las siguientes:

1. Especificación del contexto de uso: Esta fase se centra en comprender el entorno en el que se utilizará el sistema y las necesidades de los usuarios. Incluye la identificación de los usuarios, sus características y las tareas que realizarán con el sistema.
2. Especificación de los requisitos: En esta fase, se definen los requisitos del sistema en función del contexto de uso establecido en la fase anterior. Esto implica identificar los objetivos del sistema, las funciones que debe tener y las restricciones asociadas.
3. Diseño de soluciones: Aquí se crean y se evalúan diferentes soluciones de diseño para el sistema. Se generan conceptos de diseño y se desarrollan prototipos que luego se evalúan con los usuarios. El objetivo es encontrar la solución óptima que cumpla con los requisitos establecidos.
4. Desarrollo: En esta fase, se lleva a cabo la implementación técnica del sistema en base al diseño seleccionado. Se desarrolla el software, se crea la interfaz de usuario y se integran todas las funcionalidades del sistema.

5. Evaluación: Durante esta fase, se evalúa el sistema con usuarios reales para determinar si cumple con los requisitos y si es fácil de usar. Se pueden utilizar diferentes técnicas de evaluación, como pruebas de usabilidad, observación de usuarios y recopilación de retroalimentación.
6. Despliegue: En esta fase, el sistema se implementa y se pone en uso en el entorno real. Se realiza el despliegue completo del sistema y se realiza un seguimiento para asegurarse de que funcione correctamente y se ajuste a las necesidades de los usuarios.
7. Uso continuo: Después del despliegue, el sistema entra en la fase de uso continuo, donde se monitorea su funcionamiento, se realizan ajustes y actualizaciones según sea necesario. Esta fase es importante para mantener y mejorar la calidad y la usabilidad del sistema a lo largo del tiempo.

Dadas estas etapas, vamos a adaptarlas para poder usarlas en nuestro proyecto, ya que en nuestro caso no tendría sentido emplearlas tal cual están definidas. Esto es debido a que, por ejemplo, puede que nuestro producto no necesite un despliegue y mucho menos un uso continuo, son fases que podrían tener sentido si tenemos en cuenta trabajos futuros pero que no sabemos con seguridad si necesitaremos emplearlas para este trabajo, por lo que simplificando las etapas el resultado podría ser el siguiente:

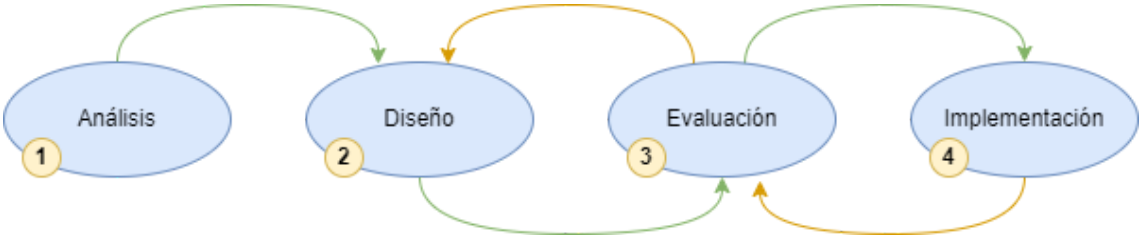


Ilustración 1: Etapas DCU

Dicho esto, la metodología DCU se ajusta perfectamente al tipo de producto que queremos desarrollar y propiciará el desarrollo de un resultado final que sea de agrado para los usuarios, lo cual es necesario para el éxito del proyecto.

1.3 Estructura

El contenido de este documento muestra el desarrollo de un producto software y está dividido en diferentes apartados. A continuación se presentará la estructura que va a seguir este documento y qué se puede encontrar en cada uno de los apartados.

El primer apartado que encontramos es el de la introducción, en él podemos encontrar la puesta en contexto del proyecto junto con una breve explicación de la historia e importancia del producto que vamos a desarrollar.

A continuación encontramos el estado del arte, dónde podemos ver un análisis del ecosistema en el que se encuentra la aplicación que queremos desarrollar, teniendo en cuenta la industria de los videojuegos y los productos software; y analizando la competencia existente.

El tercer apartado es el análisis de necesidades. Aquí podemos encontrar un proceso de obtención de necesidades de nuestro proyecto llevado a cabo con la metodología que hemos elegido previamente. Este proceso terminará con la obtención de unos casos de uso que serán vitales para el desarrollo de la aplicación.

Seguidamente encontramos el análisis conceptual y diseño de la solución, en él se hace uso de la información obtenida en el apartado anterior para proceder con el diseño de cada una de las partes del proyecto, obteniendo bocetos o modelos.

El quinto apartado es el desarrollo de la solución, dónde se profundizará acerca de la implementación de nuestro proyecto. En él podemos encontrar la arquitectura de nuestra aplicación, además de ciertos aspectos importantes relacionados con la codificación y desarrollo del trabajo.

El sexto apartado es la implantación del proyecto, dónde podemos ver la puesta en funcionamiento del servicio desarrollado.

El siguiente apartado es el de producto desarrollado, en él se hará una comprobación de que los casos de prueba que se diseñaron en un principio se están cubriendo con la implementación desarrollada y que estamos cumpliendo con las necesidades que nos planteamos.

El octavo apartado son las conclusiones, en él se reflexionará acerca del proyecto en su conjunto, así como en la solución final que se ha obtenido. También se hablará acerca del aprendizaje que ha supuesto la realización del trabajo y la relación del mismo con los estudios cursados.

A continuación encontramos los trabajos futuros, dónde se aborda acerca de todo aquello que se desearía implementar pero se ha obviado por falta de tiempo o de aquello que se implementaría en el caso de seguir desarrollando el proyecto más en profundidad.

Los últimos apartados son las referencias, dónde se encontrarán todas las fuentes que se han nombrado en alguna parte del documento; y el anexo, en el que se reflexiona acerca de la relación del proyecto con los objetivos de desarrollo sostenible.

2. Estado del arte

En este apartado se presentará el contexto en el que se pretende desarrollar el proyecto. Para ello se presentará información acerca de los distintos competidores que presentan productos similares así como el estado actual de la economía en la industria de los videojuegos.

Para tener una visión amplia sobre el producto que se quiere desarrollar es necesario conocer la industria en la que se desenvuelve el proyecto, esta junta dos mundos que conviven dentro de la industria de la tecnología, estos son las Tecnologías de la Información y la Comunicación o TIC, y los videojuegos.

Para comenzar, el mercado que abarcan los productos Software está en constante crecimiento, según las estadísticas que pone a disposición la Unión Internacional de Telecomunicaciones (UIT) el número de usuarios individuales que hubo en internet el año 2022 fue de 5.3 billones, superando en un 8,16% el año anterior, y como podemos ver esta cifra no ha dejado de aumentar desde que se tiene registro[3]. Esto hace que el número de clientes potenciales que tienen los productos software sea cada vez mayor, aumentando la viabilidad de los mismos.

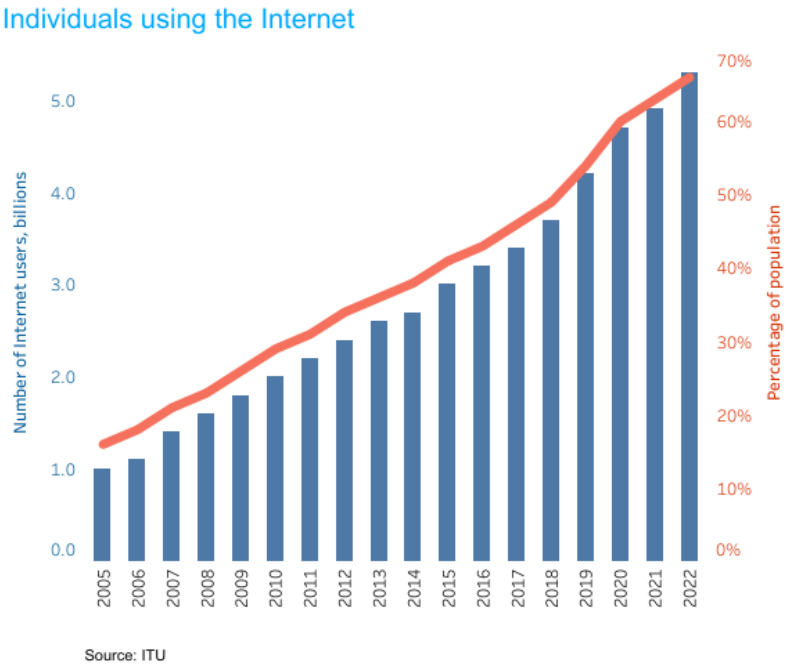


Ilustración 2: Gráfico de usuarios de Internet

Hablando sobre el contexto de las tecnologías de la información es inevitable tener en cuenta el impacto que ha tenido la pandemia del COVID-19 sobre la industria. Según un informe de McKinsey & Company[4], la pandemia ha acelerado la adopción de tecnologías digitales en todas las industrias y ha llevado a un aumento en la demanda de servicios en línea. Respecto a la adopción de tecnologías digitales, aunque podríamos pensar que esto supone un claro beneficio para el contexto de nuestra aplicación, la realidad es que no está directamente relacionado con el contexto de nuestra aplicación, esto se debe principalmente a que los nuevos servicios que se han desarrollado a partir del COVID-19 forman parte de un proceso de digitalización de unos servicios que ya existían de manera física , es decir estamos hablando de una transformación. En cambio, nuestro proyecto es un tipo de producto que no puede ofrecerse de manera física, los asistentes de videojuegos solo existen de manera digital.

Sin embargo, si hablamos del aumento de la demanda de los servicios en línea y en especial de los servicios de entretenimiento, vemos que está estrictamente relacionado con nuestro contexto y que supone un claro beneficio. La pandemia propició un aumento en la cantidad de jugadores de videojuegos en línea y aunque es posible que al finalizar el confinamiento se redujera significativamente, la realidad es que ha acabado siendo un factor muy beneficioso para la industria. Podemos confirmar esto si observamos los datos que nos ofrece la desarrolladora del videojuego que nos concierne, League of Legends. Y es que Riot Games, la desarrolladora del videojuego, permite contabilizar la cantidad de jugadores mensuales que hay en sus servidores, lo que da pie a herramientas como “activeplayer.io” que nos pone a disposición estos datos de forma sencilla, en ella podemos comprobar que el número de jugadores activos el pasado mes de Abril fue de aproximadamente 153 Millones, lo cual lo sitúa entre los más jugados a nivel global.

Para conocer exactamente nuestro contexto económico debemos también tener en cuenta el estado de la industria de los videojuegos. Para ello, vamos a fijarnos en una de las principales potencias desarrolladoras de videojuegos y país de origen de la empresa Riot Games, Estados Unidos. Según un artículo publicado por Robert W. Crandall y J. Gregory Sidak “En 2005, los ingresos por productos de software de entretenimiento y accesorios directamente relacionados fueron de $10.5 mil millones[5]. Cada dólar gastado en software de entretenimiento en los Estados Unidos contribuye directamente al Producto Interior Bruto (PIB)”. Estamos hablando de que las ventas producidas en la industria de los videojuegos juegan un papel muy beneficioso en la economía del país, pero esto se trata de un dato anticuado, ya que se aproxima que el tamaño del mercado de la industria de los videojuegos en Estados Unidos en 2022 fue de $97.6 mil millones[6].

Por último, para desarrollar nuestra aplicación es muy importante tener en cuenta las opciones ya existentes que ofrecen un servicio similar. Comenzando por el que es un claro referente, "op.gg" ha sido el claro líder en el servicio de análisis de datos del videojuego. Se trata de una empresa Coreana que lanzó la primera versión de este producto en 2012, haciendo de este uno de los productos pioneros en el análisis de estadísticas para el videojuego League of Legends. Su largo recorrido ha sido uno de los factores clave para el éxito de esta plataforma, habiendo evolucionado su producto hacía una herramienta que los jugadores encuentran realmente útil. A esto lo respaldan los datos que podemos encontrar directamente desde su página web, dónde afirman tener actualmente 55 millones de usuarios mensuales únicos, que con respecto a los 153 millones de usuarios mensuales de los cuales dispone el propio videojuego, hace que más de la tercera parte de la base de jugadores usen su plataforma, haciéndola la plataforma líder en la actualidad. Por estos motivos el trabajo realizado en este proyecto va a estar altamente inspirado en la herramienta op.gg, ya que se ha demostrado que ofrece un servicio que de verdad resulta de utilidad para los jugadores y que ha supuesto la imitación de muchos otros productos posteriores.

Restando la anterior página web, también podemos encontrar otras plataformas que ofrecen un servicio similar y que también han logrado triunfar entre los jugadores, estas son Porofessor.gg y Champion.gg. Ante la imposibilidad de competir en el mismo servicio que otorga op.gg, estas páginas buscaron otra funcionalidad en la que focalizar sus herramientas. Aunque al principio champion.gg se enfocaba en dar estadísticas sobre aspectos generales del juego, como puede ser la tasa de victorias de los personajes, cuando op.gg comenzó a dar este mismo servicio en su plataforma necesitaban algo más que les diferenciara del que era el producto estrella entre los jugadores. Fué ahí donde entraron las aplicaciones que permitían analizar estadísticas a tiempo real, así como el estado actual de las partidas. Porofessor es una herramienta que permite visualizar información en tiempo real acerca de una partida que se está produciendo, poniendo a disposición de sus usuarios algunas partidas de los mejores jugadores del servidor en su página de inicio, además también dispone de una aplicación de escritorio que proporciona distintas ayudas a los jugadores interactuando con el propio juego a tiempo real. Del mismo modo Champion.gg tiene una aplicación similar a la anteriormente nombrada, llamada Blitz.gg. Esta destaca principalmente por analizar tus partidas y dar consejos acorde a tus resultados, además de interactuar directamente con el juego añadiendo "overlays", estos son ciertos aspectos visuales que aparecen dentro del juego pero que no forman parte de él, sino que provienen de la aplicación en sí misma, y que nos proporcionan distintas ayudas como algunas estadísticas o consejos basados en las acciones que toman los mejores jugadores.

2.1 Propuesta

Teniendo en cuenta las características anteriormente nombradas de las herramientas ya existentes en el mercado, se pretende realizar una propuesta de trabajo que tenga un aporte significativo para los usuarios.

Ya que los servicios actuales se encuentran en un estado maduro debido a la cantidad de tiempo que llevan en funcionamiento, hay realmente pocos aspectos que puedan suponer un avance o mejora sobre la información que aportan a los usuarios del producto. Realmente, estos servicios proporcionan toda la información que un jugador experimentado puede necesitar, pero precisamente ahí reside el problema, estas webs se enfocan en los jugadores que tienen experiencia con el juego y saben entender los datos que les brindan, en cambio, un jugador nuevo no es capaz de comprender aquello que le están ofreciendo.

Por estos motivos, la propuesta de trabajo presentada consiste en un servicio de análisis de partidas y estadísticas, muy similar a los de la competencia, pero que se enfoque en los jugadores nuevos, ofreciendo una interfaz más minimalista y acogedora para aquellos usuarios que no tienen un gran entendimiento del juego, incluso llegando a mostrar explicaciones de los elementos que puedan llegar a ser motivo de confusión.

3. Análisis de necesidades

En este apartado vamos a realizar el análisis de necesidades utilizando la metodología DCU. Esto tiene el objetivo de crear una persona modelo en la cual centrarnos a la hora de desarrollar la aplicación.

3.1 Cuestionarios hechos y resultados

Existen distintas maneras de crear un modelo de persona, en este caso se va a llevar a cabo la realización de una encuesta a través de Google Forms, que nos ayudará a hacernos una idea del tipo de persona que va a utilizar nuestra web.

Para ello, primero debemos llevar a cabo las preguntas de este cuestionario. Estas preguntas deben de ser de dos tipos: unas personales, que nos aporten información acerca de la persona que está respondiendo el cuestionario, y otras funcionales, que nos permitan saber qué tipo de funcionalidades busca el usuario en nuestra aplicación.

Las preguntas personales serán las que nos ayuden a crear nuestro modelo de persona, por lo cual es necesario hacer desde preguntas triviales, como la edad o el género, que nos servirán para crear un modelo que se sienta realista; hasta preguntas acerca de su objetivo con el videojuego o el tiempo que le dedica, de este modo tendremos más contexto sobre qué busca el usuario en nuestra aplicación.

Estas son las preguntas personales que contiene el cuestionario y las respuestas recibidas a las mismas:

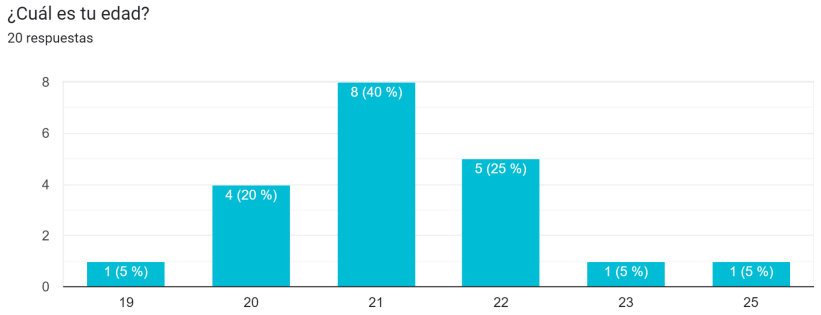


Ilustración 3: Respuestas a cuestionario 1 - Edad

La edad que más se ha obtenido como respuesta ha sido 21 años, por esto la persona modelo que se creará tendrá esta misma edad.

Del mismo modo, haremos lo mismo para el género de nuestro modelo, según los resultados el género más obtenido ha sido “hombre” por lo que la persona que creemos será del mismo género.

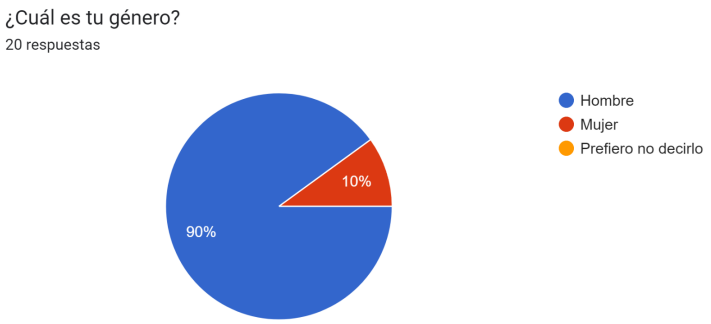


Ilustración 4: Respuestas a cuestionario 2 - Género

También nos interesa saber la cantidad de horas que nuestro modelo le va a dedicar a el juego, ya que con esto podemos saber el grado de dedicación que tiene hacia el mismo y el posible conocimiento del juego que puede llegar a tener sobre sus conceptos. En este caso, se ha obtenido dos respuestas por igual por lo que seleccionaremos un valor neutral entre estas dos respuestas, es decir, 10 horas.

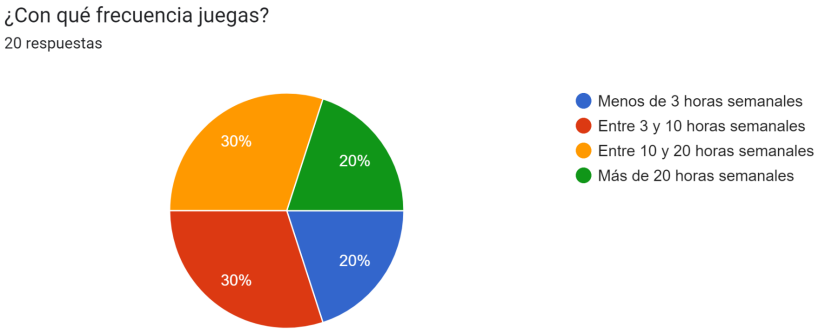


Ilustración 5: Respuestas a cuestionario 3 - Frecuencia de juego

Con la siguiente pregunta podemos saber también cual es el objetivo principal de los usuarios con el videojuego y podemos ver que aunque puede que tengan como objetivo mejorar, el objetivo principal es el de divertirse. Esto debemos tenerlo en cuenta ya que una aplicación que resulte difícil de usar, requeriría demasiado tiempo para un usuario que busca divertirse principalmente, por lo que es más que probable que sea un tiempo que no esté dispuesto a emplear en tu aplicación.

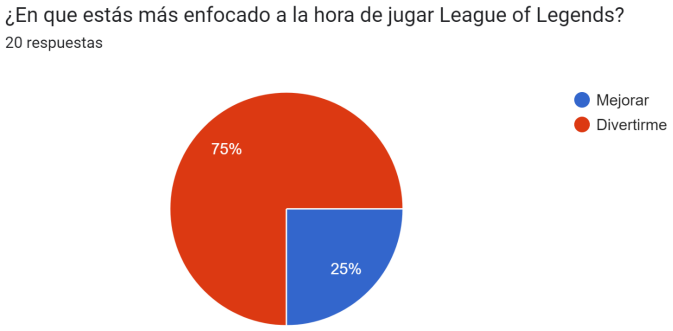


Ilustración 6: Respuestas a cuestionario 4 - Enfoque

A continuación pasamos a las preguntas funcionales y en esta primera podemos observar cuales son las herramientas de la competencia más utilizadas por los usuarios. Podemos confirmar aquí las afirmaciones hechas en apartados anteriores sobre el liderazgo de op.gg, siendo claramente la aplicación más utilizada. También podemos ver que blitz.gg y porofessor.gg son bastante utilizadas. Esto nos servirá para saber qué aplicaciones deberemos tener en cuenta a la hora de desarrollar la nuestra propia.

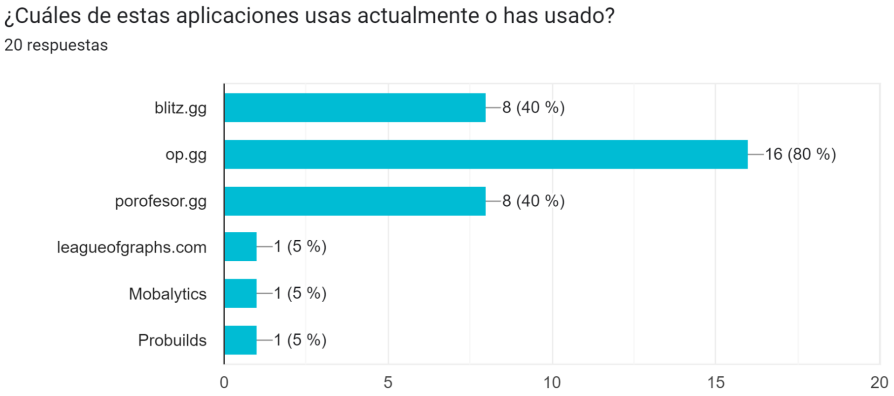


Ilustración 7: Respuestas a cuestionario 5 - Aplicaciones usadas

En la siguiente pregunta también podemos confirmar afirmaciones hechas anteriormente, ya que todas las personas que han realizado la encuesta han afirmado utilizar este tipo de herramientas al menos de vez en cuando. Esto nos da a entender la importancia de estas aplicaciones y lo muy instauradas que se encuentran entre los jugadores. El resultado también nos sirve para crear los hábitos de uso de nuestro modelo de persona.

Ilustración 8: Respuestas a cuestionario 6 - Frecuencia de uso

Seguidamente podemos ver los resultados de las funciones más utilizadas, con ello podemos ver qué funcionalidades podríamos añadir a nuestra aplicación a parte de la función principal, que es el historial. Hemos obtenido que la segunda opción más usada es el análisis de campeones y que las otras dos opciones son relativamente usadas también por lo que sería interesante generar casos de uso para estas funciones más adelante.

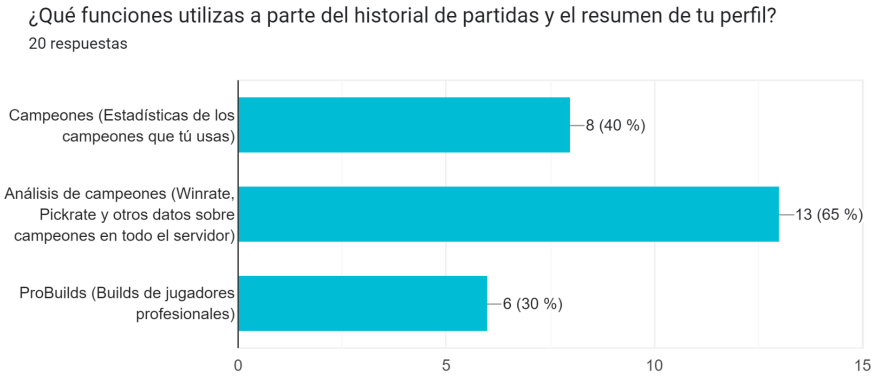


Ilustración 9: Respuestas a cuestionario 7 - Funciones más utilizadas

A continuación tenemos varias preguntas acerca de la interfaz de la aplicación más utilizada, op.gg. Con las respuestas podemos ver que una gran parte de los encuestados encuentran la herramienta sobrecargada de información o difícil de entender y que la mayoría de ellos cree que hay algunos datos que son irrelevantes o de poca importancia. Esto podemos interpretarlo como que los usuarios buscan una interfaz más simple y con menos carga de información en la que los elementos sean fácilmente reconocibles y debemos tenerlo en cuenta a la hora de desarrollar nuestra solución.

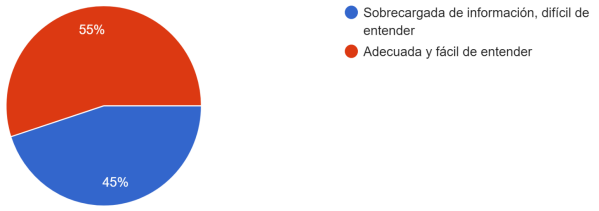


Ilustración 10: Respuestas a cuestionario 8 - Cantidad de información

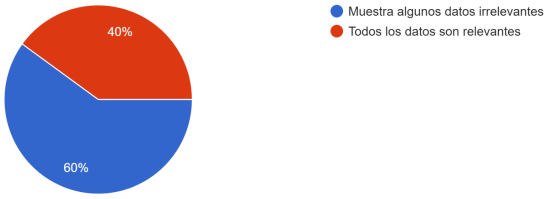


Ilustración 11: Respuestas a cuestionario 9 - Relevancia de los datos

Por último podemos encontrar dos preguntas sobre funcionalidades que no están presentes en la aplicación op.gg tomada como base. En la primera podemos ver que la mayoría de usuarios encontrarían útil que se les proporcionaran consejos personalizados en base a sus partidas, por ello, esta funcionalidad es algo muy a tener en cuenta a la hora del desarrollo y podríamos emplearla en los casos de uso.

Ilustración 12: Respuestas a cuestionario 10 - Consejos

La última pregunta es de respuesta abierta, en ella se pregunta por funcionalidades que no estén implementadas en otras aplicaciones y que ellos crean que podría resultar útil. Algunas de las respuestas más interesantes han sido:

- "Que indique cuando estoy en una racha de derrotas"
- "Poder ver cuánto acierto mis habilidades"
- "Que dé más detalles sobre qué significa cada concepto del juego"

Estas recomendaciones serán tenidas en cuenta a la hora del desarrollo y se deliberará si es posible implementarlas o no.

3.2 Definición de persona

A continuación vamos a generar la persona modelo a partir de las respuestas obtenidas en el apartado anterior. Como hemos dicho anteriormente, esta persona será ficticia e intentará recrear un tipo de usuario que representa a la mayoría de personas que van a utilizar nuestra aplicación. De este modo, todo nuestro desarrollo estará focalizado en este modelo, intentando cubrir todas sus necesidades. El resultado obtenido es el siguiente:

Nombre: Adrián García

Sexo: Masculino

Edad: 21

Ilustración 13: Persona modelo

Tecnología:

- Utiliza constantemente dispositivos electrónicos como el teléfono móvil o ordenador de sobremesa
- Suele jugar unas 10 horas semanales
- Es un nuevo jugador por lo que no tiene muchos conocimientos sobre el juego
- Está acostumbrado a usar las redes sociales, en especial Twitter
- Utiliza de vez en cuando la famosa web de estadísticas op.gg
- Encuentra innecesarios algunos datos que muestran las páginas más utilizadas, prefiere un diseño más simple

Objetivos:

- Quiere gastar el mínimo tiempo posible en la aplicación
- Le gustaría recibir consejos personalizados en base a su experiencia
- Quiere disponer de las mismas funcionalidades básicas que dan las apps más utilizadas, dentro de estas podemos identificar: Estadísticas de los personajes utilizados, análisis de los personajes y historial de partidas.
- Recibir recomendaciones en la fase previa de juego

3.3 Escenarios de uso

Escenario 1: Acceder al historial de partidas de su cuenta

Adrián acaba de terminar una partida de LoL, lleva varias horas jugando y quiere saber qué rendimiento ha tenido, por ello se dirige a su navegador en el ordenador y entra en nuestra página web. Al acceder a la página principal, tan solo tiene que introducir su nombre de usuario en el buscador y seleccionar el servidor en el que juega. Al introducir los datos y realizar la búsqueda, la web le muestra directamente su historial en el que podrá ver su tasa de victorias, sus Puntos de Liga (LP) y el resultado de sus últimas partidas.

Escenario 2: Ver detalles de una partida

Adrián acaba de terminar una partida de LoL y quiere analizar su rendimiento con datos más detallados que los que ofrece el propio juego por lo que nuevamente se dirige a el apartado “historial” (escenario 1). Aquí puede ver la lista de sus partidas, entre las que visualiza la que acaba de terminar al comienzo de la lista. Como quiere conocer más datos sobre la partida, Adrián hace click sobre la partida que le interesa y ve como la aplicación despliega una serie de datos que antes no se mostraban entre los que puede ver más detalles sobre el resto de participantes de la partida.

Escenario 3: Análisis de los personajes

Adrián quiere comprobar que personajes tienen mejor rendimiento en el servidor ya que saber esta información le supone una ventaja con respecto al resto de jugadores, para ello desde la ventana de inicio de la página web tiene un botón que lo lleva directamente a una vista con los personajes más utilizados por la comunidad y aquellos que tienen más tasa de victorias actualmente.

Escenario 4: Ranking de jugadores

A Adrián le gustaría ver el ranking de los mejores jugadores del servidor. Como es muy fan de un equipo europeo, quiere ver en qué posición se encuentran los jugadores de su equipo favorito. Para ello se dirige a nuestra web, y en la página de inicio se dirige a la pestaña “Ranking”, en ella puede seleccionar el servidor del cual quiere ver el ranking junto con la cola que desea pero por defecto se encuentran los valores que busca por lo que no necesita hacer ningún cambio. Entonces, solo le queda visualizar la lista de los 300 mejores jugadores del servidor que le ofrece esta pestaña. En ella puede ver la posición en la que se encuentra cada jugador, sus Puntos de Liga (LP), tasa de victorias y demás datos. Dentro de este ranking Adrián visualiza a su jugador favorito y cómo quiere saber más acerca de él, hace click en el nombre del jugador, esto hace que la web le redirija hacía el historial de partidas del jugador en el que puede ver todo tipo de información sobre el mismo.

Escenario 5: Visualizar consejos

Adrián ha terminado una partida pero no se siente satisfecho con su rendimiento, para poder mejorarlo quiere saber qué ha hecho mal o que podría haber hecho mejor. Para ello se dirige a la pestaña de “Consejos”, en ella puede visualizar una serie de vídeos pensados según el nivel de cada jugador en los que se explican conceptos que se consideran útiles para cada nivel. Adrián puede hacer click en el vídeo y reproducirlo directamente desde la web sin tener que redirigirse a la plataforma original donde reside el contenido.

4. Análisis conceptual y diseño de la solución

Una vez terminado el análisis de necesidades podemos dar paso al siguiente apartado, en este se llevará a cabo el diseño de la solución que vamos a implementar en nuestro proyecto. Para ello, partiendo de la información que hemos obtenido en el análisis anterior, se van a utilizar distintas técnicas que nos ayudarán a dar con un diseño óptimo.

4.1- Modelo de la BD

Para el modelo de la Base de Datos vamos a emplear el modelo relacional que fue introducido por Edgar F. Codd en 1970. Siendo el modelo más utilizado, este ordena los datos en tablas en las que cada una de sus columnas representa un atributo de la entidad siendo uno de ellos la Clave Primaria (CP), que nos servirá para establecer relaciones entre tablas. Del mismo modo, cada fila representa una instancia de la entidad en cuestión, en este caso hemos generado unas instancias de ejemplo que nos permiten hacernos una idea de qué forma pueden tener las tablas siendo estos datos ficticios.

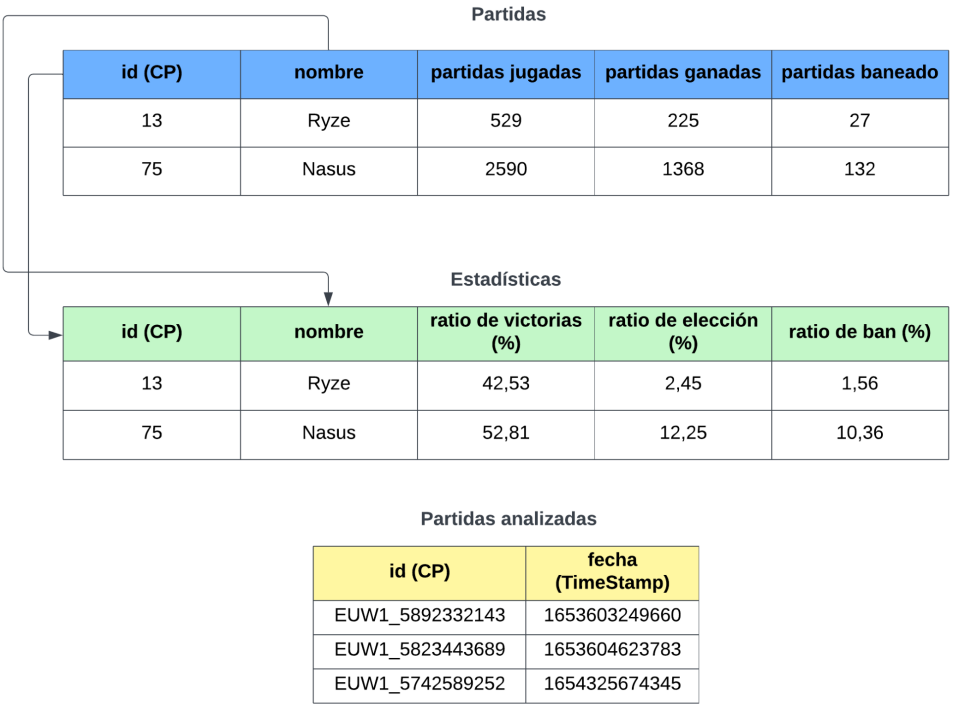


Ilustración 14: Modelo relacional BD

Estas son todas las tablas necesarias para nuestra Base de Datos, aunque puedan parecer pocas, no se necesitan más tablas para cumplir nuestro cometido. Esto se debe a que el videojuego sobre el que trata nuestra aplicación tiene su propia Base de Datos que almacena mucha información acerca de las partidas que se llevan a cabo y esta información es accesible, por lo que no necesitamos almacenar muchos de los datos que generan las partidas jugadas. En cambio, las estadísticas que queremos generar a partir de los datos no se guardan en la BD del juego por lo que es ahí donde entran las tablas que hemos mostrado, estas serán suficientes para generar las estadísticas que queremos mostrar a los usuarios.

Además, con este modelo podemos ver las relaciones que existen entre las diferentes tablas. Como podemos ver, existe una relación entre la primera tabla y la segunda, esta será una relación 1 a 1, ya que por cada columna que tengamos en la primera tabla, tendremos una columna en la segunda. El número de columnas o instancias que encontraremos en las dos primeras tablas variará muy poco a lo largo del tiempo, pues habrá una instancia por cada personaje que tenga el juego. Actualmente, el último personaje fue lanzado en Marzo de 2023, haciendo un total de 163. Estos personajes los podremos identificar de dos maneras, la primera con su identificador dentro del juego, que hemos establecido como CP, o por su nombre que también es único y lo hemos fijado como Clave Ajena (CA), de este modo podremos buscar instancias de una tabla en la otra.

Por último, podemos ver que la tabla de “Partidas analizadas” no tiene ninguna relación con el resto. Aunque no tenga ninguna relación directa en la BD ni ninguna CA, esta tabla es necesaria para calcular las estadísticas. Esta tabla será la encargada de guardar el ID de las partidas que hemos analizado y guardará tantas instancias como partidas queramos analizar, evitando así analizar más de una vez la misma partida, generando datos duplicados y por tanto erróneos. Además nos servirá para contabilizar el número total de partidas analizadas que es necesario para calcular los ratios.

4.2- Bocetos de las interfaces

El siguiente paso consiste en realizar los bocetos que vamos a utilizar como base para el desarrollo de nuestra interfaz. Para ello debemos tener en cuenta los escenarios de uso que hemos planteado anteriormente, ya que estos bocetos deben de cumplir todos los casos planteados, en caso contrario se estarían dejando funcionalidades sin cubrir. También deberemos de tener en cuenta los aspectos abordados en el análisis de necesidades que influyen en la interfaz de la web, entre ellos podemos encontrar por ejemplo el minimalismo que recalcábamos con anterioridad para ofrecer una mejor experiencia de usuario a los jugadores nuevos.

De este modo comenzamos con la página principal de nuestra aplicación ya que será lo primero a lo que accederán nuestros usuarios. Lo primero que podemos encontrar en la parte superior de la página es un menú que nos servirá para navegar entre las distintas pestañas de nuestra aplicación. Este menú se mantendrá en todas las pestañas de la web, mejorando así la navegabilidad de la misma.

A excepción del logo, que es un apartado meramente estético que permite generar una imagen de tu aplicación que los usuarios puedan recordar, el resto de elementos que hay incluidos se limitan a cumplir con las funcionalidades que se propusieron en los casos de uso. Entre estos elementos encontramos un buscador que permite seleccionar el servidor en el que juega el usuario e introducir su nombre, este buscador llevará al usuario hacía el historial de partidas que se considera como el elemento principal de la aplicación. Los elementos restantes son dos extractos de las tablas que podemos encontrar en otras pestañas, estos contienen los elementos situados en las primeras posiciones de las tablas originales que pueden ser considerados como los elementos de mayor interés.

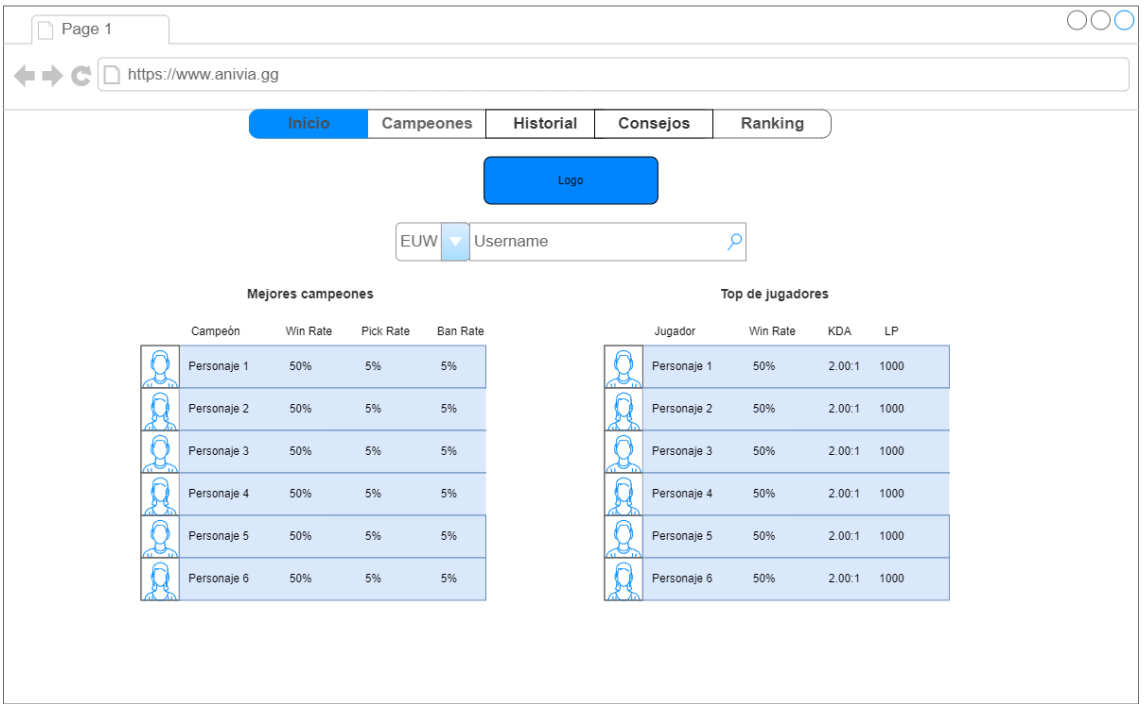


Ilustración 15: Boceto 1 - Página de inicio

A continuación tenemos el historial de partidas, el cuál se considera que es el foco principal de los usuarios de nuestra aplicación. Nótese que ahora el buscador se sitúa en la parte superior de la pantalla, este lo podremos encontrar en el resto de pestañas de la web con el motivo de tener un acceso rápido al historial.

En cuanto a los elementos exclusivos de esta pestaña encontramos cierta información acerca de la cuenta del jugador que se haya buscado, como puede ser el nivel de la cuenta, su nombre o su rango clasificatorio y una lista de partidas que corresponde con su historial de partidas jugadas. Cada elemento de la lista contará con cierta información trivial de la partida que además podremos ampliar haciendo click sobre la partida, desplegándose información más detallada sobre el resto de participantes.

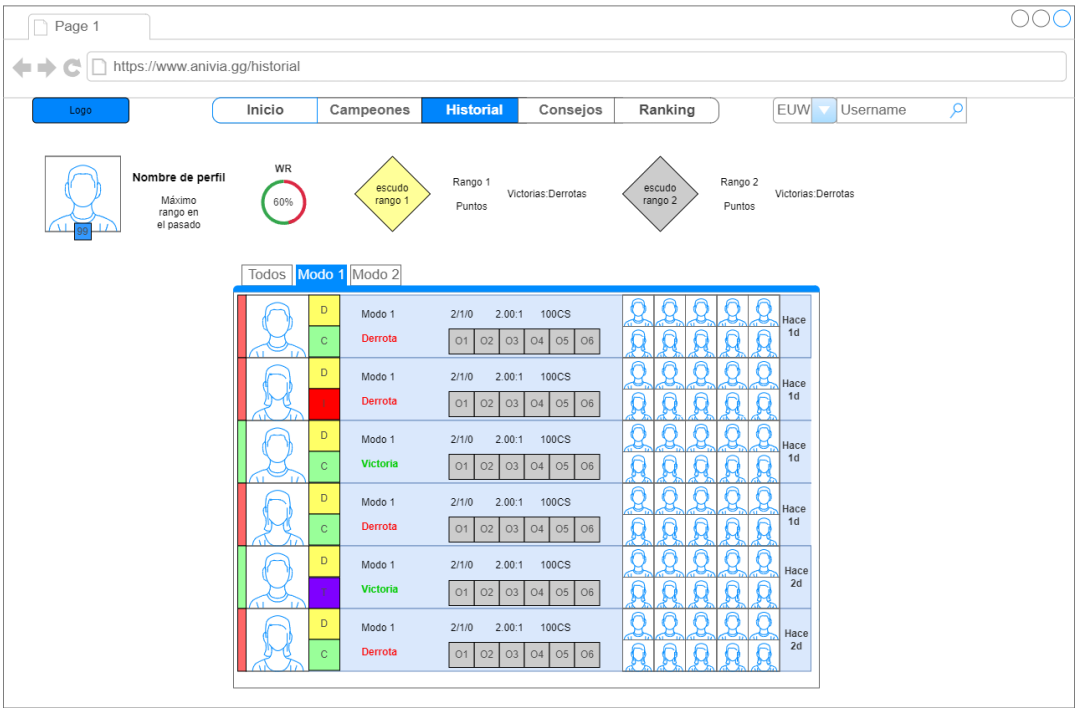


Ilustración 15: Boceto 2 - Historial

Siguiendo con la pestaña de "Campeones", encontramos una tabla que contiene estadísticas de los personajes del juego. Esta tabla tendrá una entrada por cada personaje del juego y se encargará de mostrar las estadísticas que se guardan en la Base de Datos que hemos modelado anteriormente. La tabla estará ordenada por el porcentaje de victorias de cada personaje, de modo que los jugadores podrán ver en la parte superior los personajes con los que serán más propensos a ganar partidas. Además también podremos encontrar otras estadísticas interesantes, así como un buscador que permita introducir el nombre de un personaje, filtrando los resultados que muestra la tabla.

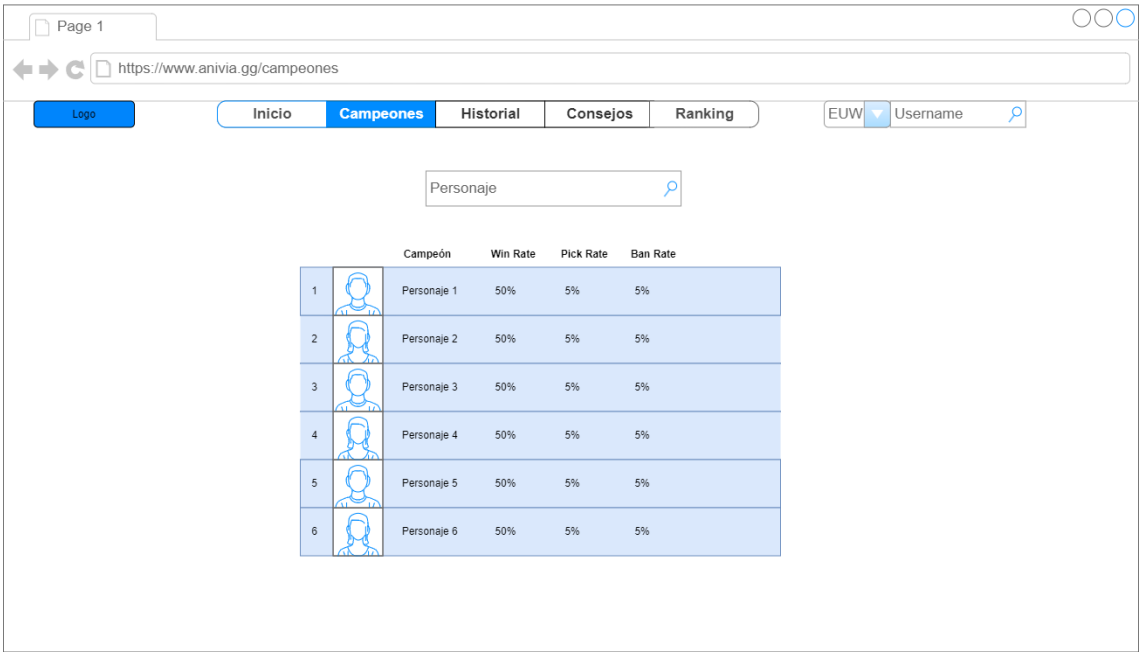


Ilustración 15: Boceto 3 - Campeones

Una de las funcionalidades más diferenciales de la aplicación es el apartado de "Consejos", en este se pueden encontrar diferentes consejos y conocimientos del juego separados por niveles. Principalmente encontramos vídeos que se consideran de gran utilidad para el aprendizaje de un aspecto del juego según el nivel en el que te encuentres, estos vendrán acompañados de un pequeño resumen que le dará una pequeña idea a los usuarios de qué va a tratar el vídeo en cuestión. Todo el contenido de interés está alojado en Youtube por lo que se debe incorporar un reproductor de esta plataforma para que no sea necesario redirigir a los usuarios a otra web.

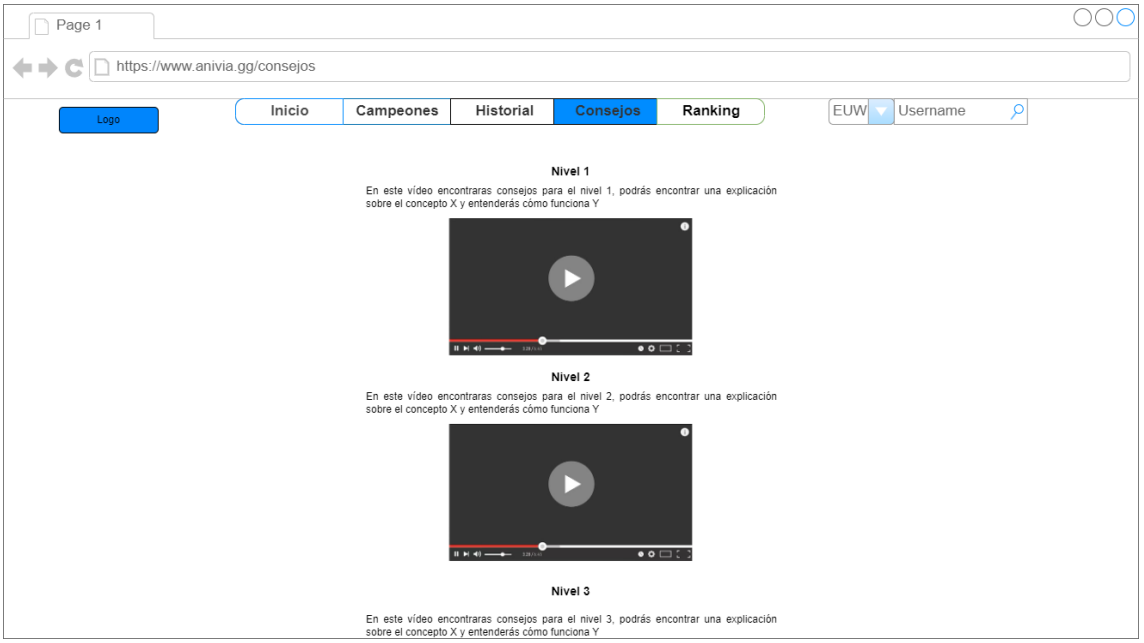


Ilustración 15: Boceto 4 - Consejos

Por último encontramos la pestaña “Ranking”, en ella podemos ver unos botones que nos permiten seleccionar el servidor y la cola que queremos consultar. Una vez seleccionadas, o simplemente tomando los valores por defecto, se muestra una tabla con el ranking de jugadores para ese servidor y cola, estos son los 300 mejores jugadores. Se muestra una entrada por cada jugador en la que podremos ver algunos datos acerca del mismo, así como hacer click en un nombre, lo cual nos redirigirá hacia el historial del jugador seleccionado.

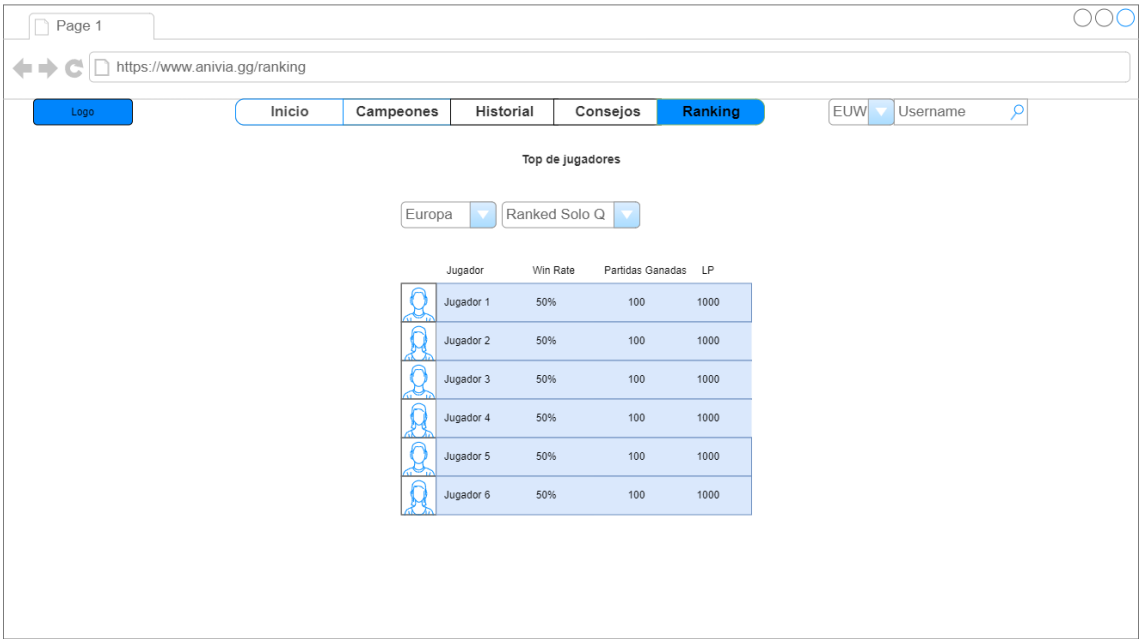


Ilustración 16: Boceto 5 - Ranking

5. Desarrollo de la solución

5.1 Problemas previos al desarrollo

Antes de comenzar con el desarrollo del proyecto debemos de tener en cuenta algunas consideraciones que pueden afectar en el transcurso del mismo. Estas son algunas necesidades que no pueden obtenerse a partir de la metodología DCU, ya que son aspectos relacionados estrictamente con el funcionamiento interno de la aplicación, del cual los usuarios no forman parte.

La primera necesidad que debemos tener en cuenta, ya que es obligatoria, es que debemos buscar una manera de obtener datos del videojuego. No es posible mostrar estadísticas de un videojuego si no tenemos acceso a la información del mismo por lo que debemos encontrar una forma de acceder a la base de datos del videojuego para obtener sus datos.

Así mismo, debemos tener en cuenta que el juego no está desplegado de forma globalmente unificado, es decir, el juego está desplegado por regiones y estas son independientes las unas de las otras. Esto conlleva algunas implicaciones como por ejemplo que la cuenta con nombre “Player123” en Norte América no es la misma que la cuenta con el mismo nombre pero localizada en Sudamérica, tanto los dueños como los datos generados por esa cuenta serán distintos según la región en la que se encuentren. Esto genera la necesidad de separar ciertas funcionalidades según la región en la que queramos obtener esa información.

Del mismo modo, nos interesa que las estadísticas que brindemos a los usuarios sean calculadas de manera global, esto añadirá cierta complejidad ya que, como hemos dicho anteriormente, las regiones son independientes, por lo que sus datos están separados.

También debemos tener en cuenta otra consideración acerca de las estadísticas, esto es que calcular unas estadísticas que tengan cierto valor requiere utilizar una gran cantidad de datos, es decir, en un entorno en el que se llevan a cabo millones de partidas al mes, no podemos limitarnos a analizar 50 partidas, la muestra sería demasiado pequeña y los datos mostrados no serían ni relevantes ni acordes con el resultado real. Teniendo en cuenta que tenemos que analizar una gran cantidad de datos, debemos encontrar una manera de que los datos estadísticos sean accesibles en un tiempo razonable, sería inviable que un usuario debiera esperar 30 minutos en una pantalla de carga para poder ver las estadísticas que busca.

Para satisfacer nuestro concepto diferenciador elegido anteriormente, que es el de enfocarnos en los jugadores más nuevos, debemos ofrecer a los usuarios herramientas

para comprender los elementos que no entiendan, ofrecer consejos que sean realmente útiles para mejorar, así como ofrecer una interfaz que no esté sobrecargada de información pero que siga siendo útil para el resto de usuarios más experimentados en el juego.

5.2 Soluciones propuestas

Dados todos los problemas y consideraciones planteados en apartado anterior, se van a exponer las diferentes soluciones tomadas.

Ante la necesidad de obtener los datos de la BD del videojuego solo existe una posible solución y esta reside en el uso de la API que pone a disposición la desarrolladora para ser utilizada por los programadores. Esta API no es de uso público pero se puede solicitar su acceso realizando una petición de la cual hablaremos más adelante.

Con respecto a los servidores en los que está dividido el juego, hemos optado por dar soporte a hasta 10 servidores distintos. Para ello hemos elegido los que creemos más importantes o con mayor número de jugadores. Esto implica, como ya hemos implementado en los diseños previos, tener ciertos botones de selección de servidor en distintas pestañas de la aplicación, que permitirán a los usuarios ser capaces de consultar sus datos sin importar la región a la que pertenezcan.

Siguiendo con el problema de los tiempos de carga, la solución que se ha considerado más eficaz es la de almacenar los datos de las estadísticas en una BD que se actualizará regularmente desde el servidor. Para ello se desarrollará un método que se encargará del análisis de los datos, este método se deberá ejecutar con una frecuencia que se considere adecuada desde el servidor que ejecute la aplicación, ya que puede llegar a tardar horas en finalizar, debido a la gran cantidad de partidas que se pretende abordar.

Por último, para ofrecer un producto que sea de utilidad a los jugadores nuevos, ya hemos introducido en los bocetos de la interfaz un apartado exclusivamente para estos usuarios. Además de esto, se pretende añadir explicaciones sobre los conceptos más complejos del juego que se desplegarán cuando los usuarios posicionen el cursor del ratón encima de uno de estos elementos.

5.3 Arquitectura

La arquitectura de nuestra aplicación va a venir definida por cierta limitación que nos ofrece uno de los elementos de la misma, esta es la API web. Esto es debido a que existen dos tipos de claves que permiten realizar consultas al servicio REST:

- Una clave personal, que tiene la posibilidad de hacer un número no muy elevado de consultas y que no tiene permitido ser ejecutada en la web por distintos usuarios.
- Una clave de producción, está pensada para proyectos grandes que se pretenden desplegar en la web y que van a realizar un número considerable de peticiones.

Para nuestro caso, la clave que necesitamos para nuestro tipo de proyecto es una clave de producción pero los requisitos para obtener una clave de este tipo son mucho más estrictos que para conseguir una personal. Entre ellos se encuentra tener un prototipo funcional de la aplicación, cosa de la cual no disponemos en estos momentos por lo que nos vamos a ver restringidos a utilizar una clave personal con la posibilidad de cambiar a una de producción en el futuro.

Otra de las limitaciones que nos da el uso de la API es que a la hora de construir las peticiones nos encontramos con un problema. Este es el protocolo CORS que nos exige el navegador a la hora de realizar la petición y que nos impide lanzar peticiones a la API directamente desde el frontend.

¿Por qué ocurre esto? Esto es debido a que el servidor que atiende a nuestras peticiones está configurado para que rechace las peticiones que se lancen directamente desde el frontend de la aplicación con el objetivo de evitar que los programadores expongan su API Key a los usuarios, no permitiendo así que la clave sea robada.

Esto elimina la posibilidad de desarrollar nuestro proyecto en su totalidad desde un frontend, ya que existen tecnologías que no necesitan de un backend en funcionamiento y pueden ser autosuficientes haciendo uso de scripts o otras técnicas.

De este modo, estas dos limitaciones hacen que nuestra aplicación no pueda ser desplegada en la web por el momento, ni pueda estar desarrollada únicamente en un frontend, por lo que debemos plantear una arquitectura que se adecue a nuestras necesidades y desplegar la misma de manera local, al menos por el momento. El resultado es el siguiente:

Ilustración 17: Arquitectura

Como podemos ver, se ha separado la aplicación en tres componentes principales: un frontend, que se encargará de llevar la interfaz con la que interactuarán los usuarios; un backend, que responderá a las peticiones del frontend y se encargará de ejecutar la mayor parte de la lógica de la aplicación, este tendrá una API REST al cual el frontend lanzará sus peticiones para que él mismo interactúe con el último componente; una Base de Datos que contendrá toda la información que hemos comentado en apartados previos.

5.4 Tecnología empleada

Ahora pasaremos a explicar todas las tecnologías y diferentes aplicaciones que se van a emplear en el desarrollo de la aplicación. Vamos a comenzar entonces por los entornos de desarrollo (IDEs) que vamos a emplear para la codificación del proyecto y continuaremos con las tecnologías empleadas en cada uno de ellos.

IntelliJ IDEA

IntelliJ IDEA es un entorno de desarrollo integrado diseñado principalmente para el desarrollo de proyectos Java. Desarrollado por JetBrains, es conocido por ser uno de los entornos que más herramientas ofrece a los programadores, además de su capacidad para trabajar en proyectos complejos de manera estable y eficiente.

Visual Studio Code

Visual Studio Code (VS Code) es un editor de código fuente rápido y liviano que se puede usar para ver, editar, ejecutar y depurar código fuente para aplicaciones. VS Code es un editor potente en gran parte por las extensiones que nos permiten personalizar y agregar funcionalidad adicional de forma modular y aislada. Está desarrollado por Microsoft y se puede obtener de forma gratuita.

MySQL

MySQL es un sistema de administración de bases de datos relacionales de código abierto con un modelo cliente-servidor que ha sido desarrollado por Oracle. Se ha elegido MySQL debido a que las experiencias previas han sido positivas, siendo este un sistema fácil de manejar y que no es propenso a errores.

Tomcat

Apache Tomcat (o simplemente Tomcat) es un servidor web de código abierto y contenedor de servlets desarrollado por la Apache Software Foundation (ASF). Tomcat implementa las especificaciones de los servlets y JavaServer Pages (JSP) de Oracle Corporation y proporciona un entorno para ejecutar código Java en un servidor web.

Se ha elegido Tomcat como servidor ya que permite poner en marcha aplicaciones de forma sencilla y por tener experiencia con la herramienta.

XAMPP

XAMPP es un paquete de software libre que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. Teniendo en cuenta que queremos utilizar MySQL y un servidor Tomcat, esta aplicación es perfecta para ello, ya que permite gestionar una BD MySQL de forma sencilla y además incorpora un servidor Apache Tomcat que nos permite ejecutar la BD para que sea accesible por los demás componentes de la aplicación.

Spring[7]

Es un framework para el desarrollo de aplicaciones Java que permite crear aplicaciones autónomas de forma sencilla y rápida, ofreciendo una configuración mínima en el momento de creación del proyecto y aumentando la productividad del lenguaje Java.

Destacamos la librería RestController, que nos permite construir APIs Rest de manera muy sencilla, de la cual haremos uso para agilizar y facilitar el proceso de desarrollo de nuestro backend.

Maven

Es una herramienta de gestión de proyectos software utilizada principalmente en el desarrollo de aplicaciones Java. Proporciona un sistema de construcción y administración de dependencias que simplifica procesos de desarrollo, como la compilación o el empaquetado. Maven utiliza un archivo de configuración llamado "pom.xml" en el que se define la estructura y las dependencias del proyecto, esto facilita mucho la gestión de las dependencias, entre otras cosas.

React[8]

React es una biblioteca de JavaScript desarrollada por Facebook que se utiliza para construir interfaces de usuario interactivas y reactivas. Se caracteriza por el uso de componentes, que son piezas de código que podemos extraer para manejar de manera independiente, facilitando su reutilización.

React utiliza una técnica llamada "renderizado virtual" que optimiza el renderizado eficiente de los componentes en respuesta a cambios de estado. Esto permite la creación de aplicaciones de una sola página (SPA) rápidas y fluidas, en las que la navegabilidad de la aplicación no incluye cargas en el navegador.

Además, React dispone de más bibliotecas, como puede ser React Router, que completan la experiencia de uso y la dotan de todas las funcionalidades necesarias para el desarrollo de aplicaciones web.

Advanced REST Client

Advanced REST Client (ARC) es una extensión de navegador que permite a los desarrolladores probar y depurar sus aplicaciones web RESTful. ARC proporciona una interfaz gráfica de usuario para enviar solicitudes HTTP y HTTPS a un servidor web y ver la respuesta del servidor.

Esta extensión es perfecta para testear que nuestro servicio REST está funcionando correctamente, por lo que le daremos un buen uso en el apartado de testeo o como herramienta para identificar errores cuando el servicio no esté funcionando como debería.

5.5 Código desarrollado

5.5.1 Proceso de solicitud de la clave para la API

El primer paso para poder desarrollar un producto funcional es la obtención de la clave de la API que hemos comentado con anterioridad. Para ello debemos dirigirnos al portal de la desarrolladora que contiene toda la información para el uso de la API así como el proceso de solicitud de la clave. https://developer.riotgames.com/ . Para acceder al proceso de solicitud necesitaremos una cuenta, aunque podemos utilizar la misma que empleamos para acceder al juego por lo que esto no será un problema.

Una vez dentro del proceso, se nos piden algunos requisitos como aceptar los términos y condiciones, añadir una descripción detallada de nuestro producto o darle un nombre a nuestro producto entre otras.

Dentro de la descripción del producto, se pide ser lo más precisos posible, debiendo añadir que servicios vamos a utilizar explícitamente y explicando de qué trata el producto así como las tecnologías que se van a emplear. A continuación podemos ver una imagen del proceso:

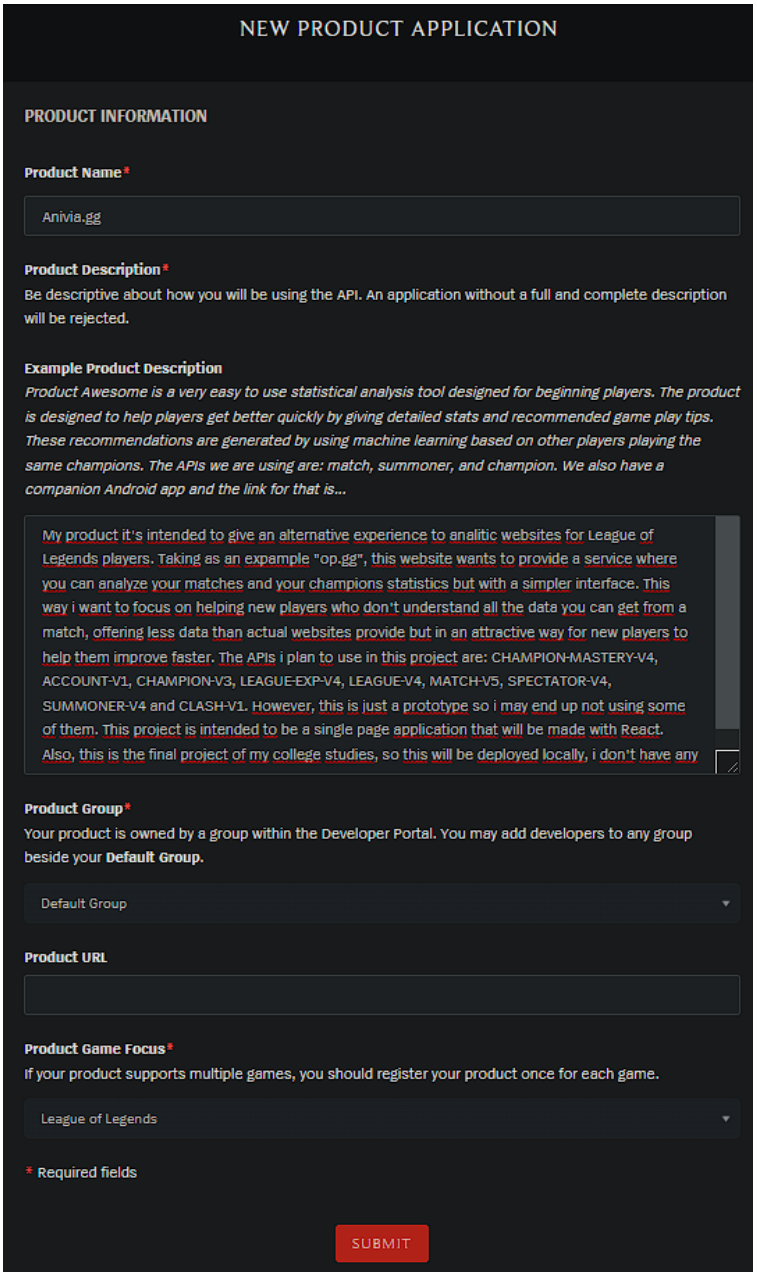


Ilustración 18: Proceso de obtención de clave

Al cabo de unos días mi solicitud fue aprobada por el equipo de desarrollo de Riot, dándome un ID para mi producto y una API Key personalizada que no caduca.

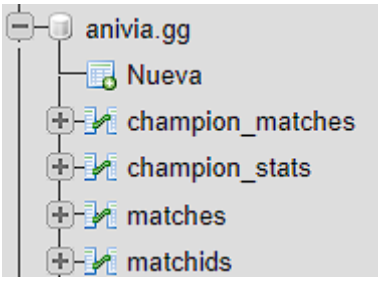
5.5.2 Base de Datos

Con motivo de entender cómo funcionan y se relacionan los 3 componentes que forman nuestra arquitectura vamos a explicarlos desde el elemento final hasta el primero, por lo que comenzaremos con la Base de Datos.

Para poner en marcha la base de datos MySQL haremos uso de la aplicación XAMPP que tiene integrado un servidor Tomcat y un MySQL con la posibilidad de trabajar con interfaz gráfica de forma cómoda y rápida desde el navegador.

Una vez estemos en el navegador, tendremos a nuestra disposición la interfaz de phpMyAdmin que nos permite gestionar nuestra BD con total libertad, pero para ello primero debemos crear nuestra BD y sus respectivas tablas.

En este punto tan solo debemos seguir el modelo de BD que construimos en el análisis y diseño de la solución. Crearemos una nueva BD con el nombre de la aplicación que vamos a desarrollar e iremos creando las tablas una a una siguiendo los esquemas que habíamos desarrollado. Tan solo tendremos que indicar el tipo de cada variable y asignar las Claves Primarias y Claves Ajenas correspondientes, el resultado es el siguiente:



Además de las tablas propuestas originalmente, se ha añadido una nueva llamada “macthes” que pretende ser un contador del número de elementos de la tabla “matchids”.

Ilustración 19: Base de Datos

Este es un ejemplo de una de las tablas que contiene un par de instancias que hemos generado para hacer pruebas de funcionamiento:

Ilustración 20: Instancias de la BD

Una vez generadas las tablas el trabajo en la BD está acabado, a no ser que queramos hacer una modificación en las tablas lo único que resta es asegurarnos de que el servidor esté levantado cuando queramos hacer pruebas de funcionamiento.

5.5.3 Backend

El siguiente elemento en la arquitectura es el backend, cómo hemos mencionado anteriormente este va a ser el encargado de ejecutar una API REST a la cual hará peticiones nuestro frontend. Este va a ser un proyecto JAVA que va a utilizar el framework Spring, el cual nos dará muchas facilidades para crear el servicio.

El primer paso para crear un proyecto en Spring es dirigirse a la web Spring Initializr, esta nos permite crear un proyecto de forma automatizada, tan solo deberemos indicarle algunos parámetros y las dependencias que queremos que tenga nuestro proyecto.

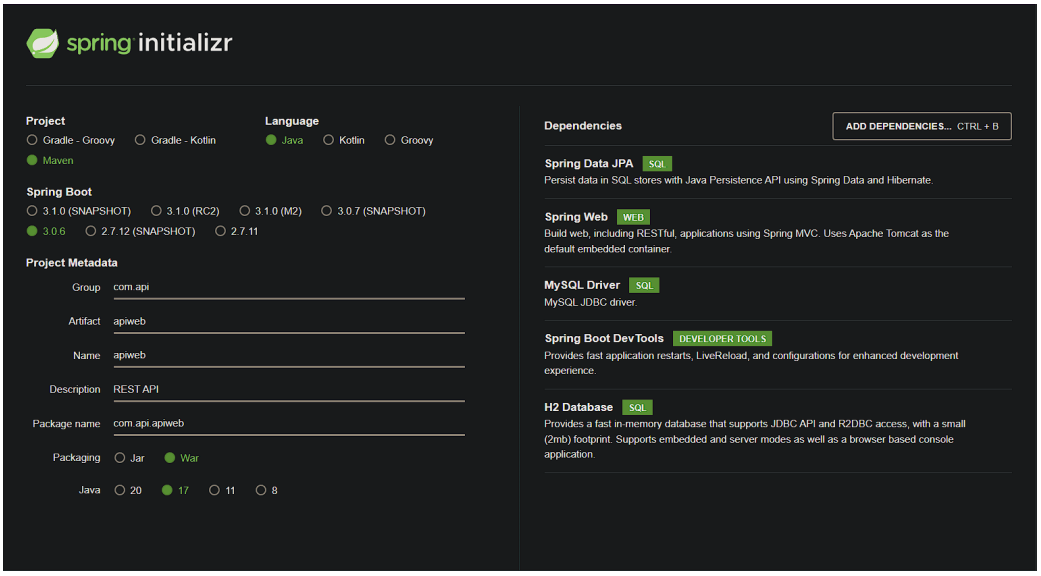


Ilustración 21: Spring Initializr

Para estar seguros de que nuestro proyecto se mantenga bien a lo largo del tiempo, es necesario hacer uso de las versiones más nuevas posibles en las tecnologías empleadas. Por ello vamos a seleccionar la última versión estable de Spring (3.0.6) y Java 17, que a pesar de no ser la última versión, es relativamente reciente. Además de esto deberemos indicar que vamos a hacer uso de Maven, que el empaquetado es de tipo “war”, esto es necesario en nuestro caso para desarrollar una API web, y añadir algunas dependencias que nos ayudarán con el manejo de la BD y con el desarrollo de nuestra API.

Una vez seleccionados todos los parámetros sólo queda generar el proyecto, esto nos devolverá un ZIP como resultado que deberemos descomprimir para obtener nuestro proyecto con su estructura base lista.

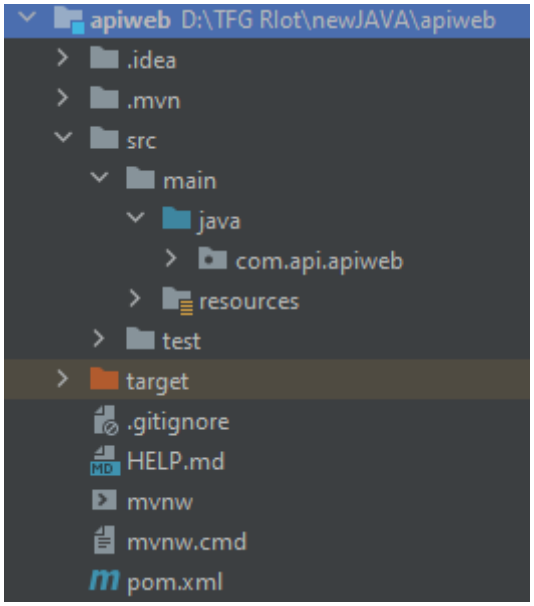


Ilustración 21: Estructura inicial Backend

Con la estructura base del proyecto creada ya podemos comenzar a generar las clases y el código de nuestra aplicación. Para ello vamos a definir la estructura en la que vamos a dividir las distintas clases Java que va a contener nuestro proyecto, estas son las siguientes:

- Carpeta "domain" que contendrá todas las entidades del proyecto, tendremos una entidad por cada tabla que hayamos definido en la BD y las propiedades de estas entidades se corresponderán con las columnas de cada tabla.
- Carpeta "repository" que contendrá todos los repositorios, estos son interfaces que harán uso de JPA Repository que es una herramienta que nos ayuda a realizar operaciones CRUD a nuestra BD de manera sencilla.
- Carpeta "service" que tendrá los servicios que hacen uso de los repositorios anteriormente nombrados. En ellos tendremos diferentes métodos que corresponden a una determinada acción sobre la BD.
- Carpeta "controller", en ella se alojarán los controladores REST que dan servicio a nuestra API. Son los encargados de exponer las direcciones URL a los que se conectará el frontend. Estos harán uso de los servicios que hemos configurado previamente para realizar ciertas acciones sobre la DB.

En la siguiente imagen podemos comprobar el aspecto que tiene la estructura final del proyecto, como podemos ver se han creado una clase de cada tipo por cada tabla que hemos definido en la BD, a excepción de los controladores dónde podemos ver que hay uno extra. Esto es debido a que uno de los controladores es el que se encarga de hacer las peticiones a la API del videojuego.

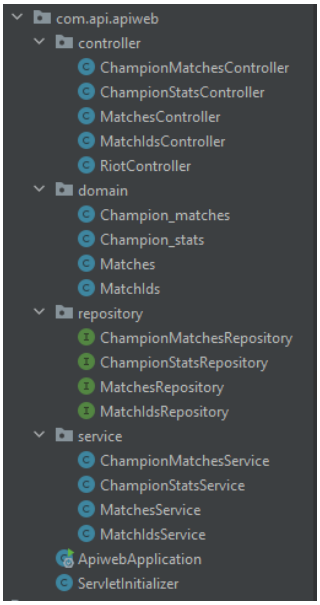


Ilustración 22: Estructura final Backend

Ya que esta es una parte importante del servicio, usaremos una de las clases para mostrar un pequeño ejemplo de cómo funciona la API haciendo uso de las anotaciones de Spring.

```
@RestController
@RequestMapping("/matchids")
@CrossOrigin(origins = "http://localhost:3000")
public class MatchIdsController {

    @Autowired
    private MatchIdsService service;

    @GetMapping("/")
    public ResponseEntity<List<MatchIds>> getMatchIds() {
        return ResponseEntity.status(HttpStatus.OK).body(service.getMatchIds());
    }

    @GetMapping("/ordered")
    public ResponseEntity<List<MatchIds>> getMatchIdsOrderedByDate() {
        return ResponseEntity.status(HttpStatus.OK).body(service.getMatchIdsOrderedByDate());
    }

    @GetMapping("/{id}")
    public ResponseEntity<MatchIds> alreadyExists(@PathVariable String id) {
        return ResponseEntity.status(HttpStatus.OK).body(service.alreadyExists(id));
    }
```

Ilustración 23: Controlador API REST

Este es un pequeño extracto de la clase, tan solo cabe destacar el uso que hace las anotaciones de Spring. La anotación @RestController le indica a Spring que la clase es un controlador, @RequestMapping sirve para indicar el prefijo que tendrán las URLs correspondientes a esta clase y del mismo modo @GetMapping conforma el resto de la URL para cada método, pudiendo incluso añadir variables al path como es el caso del último método de la imagen. De este modo si quisiéramos acceder al método getMatchIdsOrderedByDate() tendremos que hacer una petición HTTP de tipo GET a la dirección "/matchids/ordered" precedido por el servidor en el que se está ejecutando la aplicación.

También debemos destacar el archivo "application.properties", este es el necesario para establecer la conexión entre el backend y la BD. En él deberemos añadir la propiedad "datasource.url" con la dirección de nuestra BD además de otras configuraciones extras que podemos ver en la siguiente imagen.

```
1  spring.datasource.url=jdbc:mysql://localhost:3306/anivia.gg
2  spring.datasource.username=root
3  spring.datasource.password=
4  spring.jpa.hibernate.ddl-auto=update
5  spring.datasource.driver-class-name= com.mysql.cj.jdbc.Driver
6  spring.jpa.show-sql=true
```

Ilustración 24: application.properties

Por último debemos hablar del método updatestats() que se encuentra en la clase RiotController, este es el método más complejo del backend y probablemente de todo el proyecto. Este es el encargado de llevar a cabo el análisis de las partidas y el volcado de las estadísticas sobre la BD. Debido a que el método lo conforman casi 100 líneas de código generaremos un diagrama de flujo que siga la ejecución del método para que sea más visual y entendible.

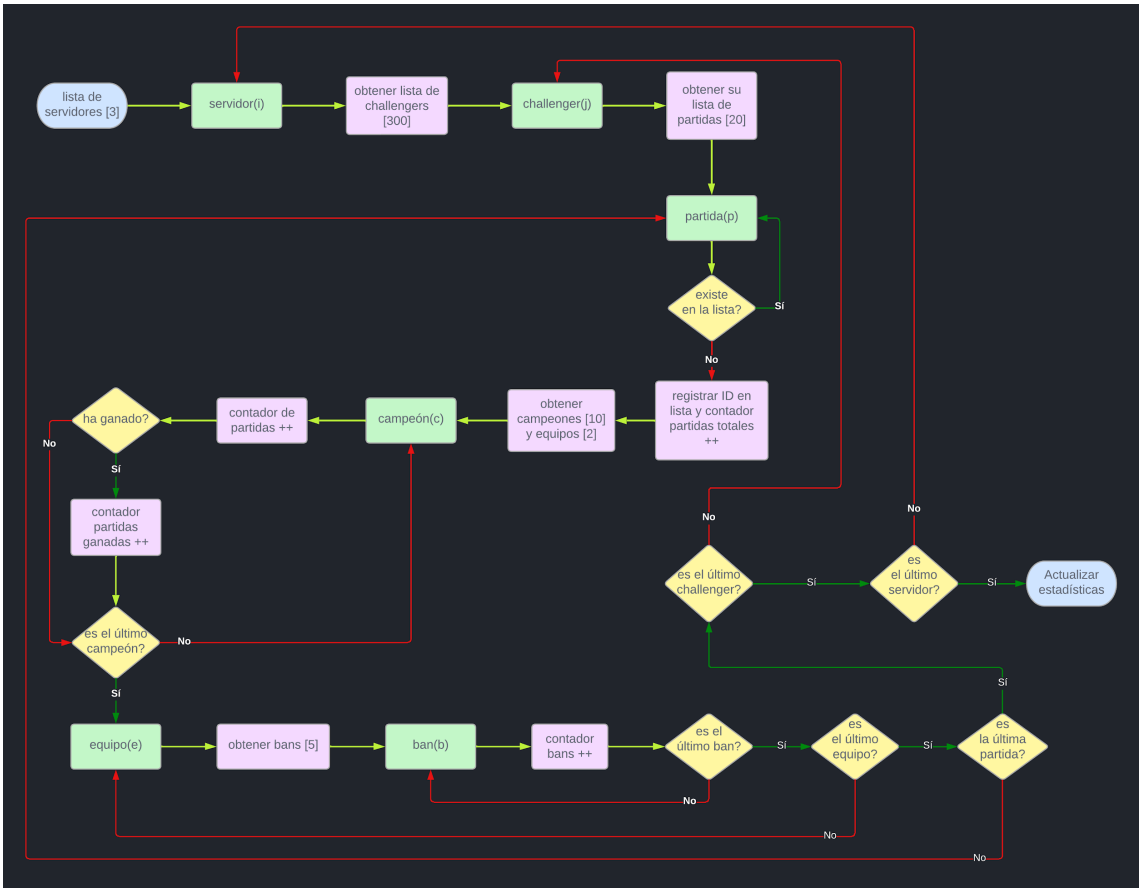


Ilustración 25: Diagrama de flujo updatestats()

Vamos a seguir entonces el diagrama paso por paso, nótese que el diagrama tiene un patrón de colores, mostrando los bucles en color verde, las condiciones en amarillo y las acciones en violeta. Se da por supuesto que cuando se comprueba que un elemento es el último de la lista y la respuesta es no, se pasa al siguiente elemento de la lista (i++). Partimos de una lista de servidores que es fija y no pretende ser cambiada, esta la forman tres de los cuatro servidores de las regiones mayoritarias ya que la información del servidor chino no es pública, con lo que tendremos que conformarnos con Europa, Norteamérica y Corea.

Comenzamos pues recorriendo la lista de servidores, donde de cada servidor obtendremos la lista de jugadores Challenger, estos son los trescientos mejores jugadores del servidor. Para esta nueva lista realizaremos otro bucle, donde obtendremos las últimas veinte partidas de cada jugador. Para cada partida tendremos que comprobar que esta no existe en la lista de partidas de nuestra BD, esto sirve para evitar duplicar resultados ya que los jugadores juegan entre ellos y podemos encontrar partidas repetidas, si la partida existe se pasa a la siguiente.

Una vez comprobamos que no existe la partida procedemos a analizarla, para ello el primer paso es registrarla en la lista de partidas de la BD y sumar en uno la lista de partidas totales. Seguidamente, para esa partida tendremos que obtener su lista de campeones y de equipos. Primero analizaremos los campeones, para ello haremos un bucle recorriendo los personajes donde para cada uno de ellos sumaremos en uno el contador de partidas que tiene ese campeón en específico, también deberemos comprobar si ha ganado y si es el caso, deberemos sumar en uno el contador de partidas ganadas de ese campeón.

Una vez analizamos todos los campeones pasamos a analizar los equipos, en ellos vamos a buscar la lista de bans que contiene cada equipo. A continuación recorreremos los bans sumando uno a los contadores de "partidas baneado" de cada campeón baneado. Una vez acabado este bucle se puede dar por finalizado el análisis de esa partida, por lo que deberemos pasar a la siguiente sucesivamente hasta acabar con todas las del jugador que se esté analizando. Del mismo modo se recorren todos los jugadores hasta terminar los trescientos de cada servidor y al terminar los tres servidores podríamos dar por finalizado el análisis por lo que solo queda volcar los resultados en la BD.

Por último, debemos recalcar cómo funciona internamente este método, donde destacamos el uso de RestTemplate, una clase de la biblioteca Spring que se utiliza para hacer solicitudes HTTP, con ella se hacen consultas a la API del videojuego, obteniendo datos en formato JSON. De este modo hemos podido manejar los datos de forma simple para poder trabajar con ellos y realizar las comprobaciones necesarias que hemos comentado en el diagrama de flujo.

5.5.4 Frontend

El último elemento en la arquitectura de la aplicación es el frontend, este es el encargado de ejecutar una aplicación React que contendrá la interfaz que los usuarios van a utilizar para interactuar con la web. El principal motivo para elegir React es la capacidad que otorga para crear Single Page Applications (SPA), estas son aplicaciones que están compuestas por una sola página lo que evita que el navegador cargue nuevas páginas para cada acción del usuario, aumentando mucho la sensación de fluidez de la aplicación. En la actualidad, cualquier web profesional es una SPA por lo que es un requisito que debíamos de cumplir.

Una vez nos disponemos a crear la aplicación React nos surge el problema de cómo crear el proyecto, ya que requiere crear una estructura base con los archivos mínimos que hacen funcionar la web. Para solucionar este problema existe la herramienta Create React App, que a través de órdenes de comandos nos proporciona un entorno base y funcional con el que podemos empezar a trabajar.

El siguiente paso es comenzar con la codificación pero para ello necesitamos entender cómo funciona React ya que va a condicionar la estructura de nuestro proyecto. Si reducimos a lo más básico el funcionamiento de React, se podría decir que las aplicaciones que crea están formadas por páginas y cada una de esas páginas contiene componentes. Los componentes son el elemento principal de React, ya que todo puede ser un componente, desde un texto o un botón hasta una página completa, esto es así ya que los componentes pueden estar compuestos por otros componentes formando unos más complejos. Además de las páginas y los componentes hay otro elemento característico de React que necesitamos conocer, estos son los hooks, que son funciones especiales que se ejecutan en los componentes y que nos permiten modificar los estados de los componentes de una manera sencilla.

A continuación podemos observar el resultado final de la estructura del proyecto. Vemos que se separa principalmente en tres: "node_modules" contiene todas las dependencias del proyecto, "public" contiene todos los archivos que van a ser accesibles para los usuarios, se trata principalmente de imágenes, y por último "src" contiene todo el código del proyecto.

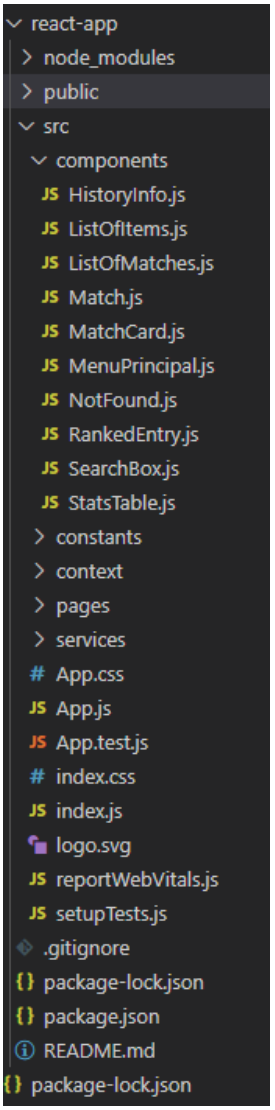


Ilustración 26: Estructura Frontend

Dentro de "src" podemos ver la estructura que hemos diseñado para nuestro proyecto:

- Components: contiene todos los componentes que forman la aplicación
- Constants: contiene las variables constantes
- Context: contiene las clases necesarias para establecer variables globales
- Pages: contiene las distintas páginas de la web
- Services: contiene los servicios o funciones que se utilizan en más de un componente

Se debe destacar también el "App.js" que es el archivo principal que contiene la aplicación y los "package*.json" dónde va incluida la configuración del proyecto.

Otro aspecto importante que hay que tener en cuenta en el frontend de una aplicación son los estilos. Existen varias formas de dar estilos a una aplicación, la más común es a través de las hojas de estilos CSS pero nosotros vamos a optar por una opción diferente, estas son las bibliotecas Ant Design. Estas bibliotecas nos proporcionan componentes ya montados y estilizados para que nosotros podamos importarlos de una manera sencilla y tenerlos disponibles para su uso. Esto además supone una ventaja a la hora de desarrollar componentes ya que no tenemos que crearlos desde cero, aunque tampoco van a cumplir el cometido que queremos tan sólo importándolos, en la mayoría de ellos deberemos hacer modificaciones para que se adecuen a nuestro contexto. Sin embargo, esta biblioteca nos va a permitir despreocuparnos de los estilos, ya que los componentes vienen estilizados de forma muy profesional, por lo que tan solo deberemos hacer alguna pequeña modificación de tanto en tanto.

Por último, queda mencionar otra biblioteca que nos ha facilitado el desarrollo de uno de los casos de uso, esta es ReactPlayer. Con anterioridad definimos un caso de uso que incluía un reproductor de Youtube que permitiera visualizar vídeos sin tener que redirigir a los usuarios a su plataforma, para añadir este reproductor se ha empleado ReactPlayer, que permite añadir reproductores de forma sencilla. Tan solo necesitaremos proporcionar la URL del enlace al vídeo y tendremos disponible un reproductor sobre ese vídeo. Además, esta biblioteca nos permite modificar el reproductor a nuestro gusto con diferentes opciones como por ejemplo añadir controles de tiempo y sonido.

Una vez profundizado de forma teórica sobre cada aspecto que forma el frontend es conveniente contemplar algún ejemplo que nos ayude a comprender cómo funciona React y como está estructurado el proyecto.

Comenzando por la anteriormente nombrada clase “App.js” que conforma la clase principal sobre la que se ejecuta la aplicación y contiene todos los elementos que queremos que aparezcan en todas las pestañas de la aplicación. Entre estos elementos se suele incluir siempre un React Router, que se encarga de gestionar todos los cambios de pestañas dentro de la web, como podemos ver nosotros hemos incluido uno que engloba toda la aplicación (<Router>) y en el que hemos añadido las rutas que conforman nuestro servicio. Adicionalmente también hemos añadido un “<MenuPrincipal>”, que es un componente que hemos diseñado para que actúe como menú que nos permita navegar entre las diferentes pestañas y que va a estar presente en todas las pestañas de la aplicación.

```
class App extends Component {

  render() {
    return (
      <Router>
        <div className="App">
          <Layout style={{ height: "100vh", background:"#EEEEEE" }}>
            <Provider>
              <MenuPrincipal modo="horizontal" tema="dark" />
              <Routes>
                <Route path="/" element={<Home />} />
                <Route path="/history/:summoner" element={<History />} />
                <Route path="/champions" element={<ChampionStats />} />
                <Route path="/tips" element={<Tips/>}/>
                <Route path="/ranking" element={<Ranking/>}/>
                <Route path="*" element={<NotFound/>} />
              </Routes>
            </Provider>
          </Layout>
        </div>
      </Router>

    );
  }
}

export default App;
```

Ilustración 27: App.js

Los elementos que hemos indicado para cada ruta del Router son las diferentes pestañas que podemos navegar, estas son componentes que tienen la misma forma que la propia "App" que acabamos de ver, pero a diferencia de ésta el contenido será mucho más complejo y extenso.

Un ejemplo de esto lo podemos encontrar en la pestaña "Historial", que es la más compleja de nuestra aplicación, llegando a formar una jerarquía de hasta 5 niveles de componentes.

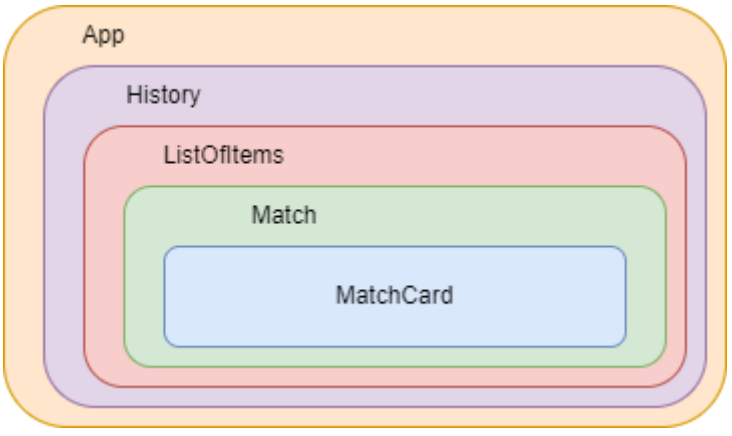


Ilustración 28: Jerarquía MatchCard

Todos los componentes han sido desarrollados de forma similar y explicarlos todos sería una tarea tediosa por lo que nos limitaremos a mostrar las partes más relevantes que no se han mostrado todavía. Esto es el uso de los hooks dentro de los componentes.

Con anterioridad hemos explicado qué son y cómo funcionan los hooks por lo que vamos a ver un ejemplo de su uso en nuestro código. Para ello debemos diferenciar entre los dos tipos de hook que existen: los hooks normales, que son hooks que vienen definidos en React por defecto y que podemos usarlos a nuestro gusto, y los "Custom hooks", estos son hooks que creamos nosotros especificamente para nuestra aplicación ya que no existe ninguno de React que nos ofrezca la funcionalidad que deseamos.

```
componentDidMount() {
    this.getPlayerData((this.state.summoner))
}

componentDidUpdate(prevProps, prevState) {
    if (prevState.playerData !== this.state.playerData) {
        this.setState({ loading: false })
    }
}
```

Ilustración 29: Hooks de React

Estas dos funciones que hemos utilizado en el componente que forma el historial son dos hooks de React. El primero ejecuta el código que introduzcamos dentro justo después de que React monte el componente en el que estamos ejecutando el hook, es decir, ejecutará la función "getPlayerData()" cuando el historial esté montado.

Del mismo modo, el segundo hook ejecuta el código que tiene en su interior después de que el componente se haya actualizado, para ello comprueba que el estado anterior sea diferente del actual para no crear un bucle infinito de actualizaciones.

Hemos nombrado los estados de los componentes pero no hemos profundizado sobre ellos. El estado de un componente son un conjunto de variables que forman parte de él y que pueden contener información de todo tipo. Estos son muy usados en React comúnmente para disparar renderizados, ya que si el estado del componente cambia, la información que muestra el componente al usuario debe cambiar consecuentemente a la del nuevo estado.

```
state = {
    summoner: this.props.params.summoner,
    matchCount: 0,
    server: 'euw1',
    region: 'europe',
    playerData: null,
    loading: true,
}
```

Ilustración 30: ejemplo de estado en React

Por último podemos observar la implementación de un Custom hook que hemos usado en nuestro proyecto, este se encarga de realizar peticiones HTTP usando Fetch, para después convertir la respuesta en formato JSON.

```
export default function getFetchResponse ( APIurl ) {
    return fetch(APIurl)
            .then(res => res.json())
            .then(response => {
                return response
            }).catch(err => { console.log(err) });
}
```

Ilustración 31: Ejemplo Custom Hook

6. Implantación

En este apartado se va a explicar de forma breve cómo poner en funcionamiento el servicio, para ello deberemos arrancar los tres componentes que forman nuestra aplicación.

En primer lugar debemos iniciar la BD, para ello tan solo es necesario encender el servidor Apache y MySQL a través de la aplicación XAMPP, la cual se mencionó anteriormente. En tan solo unos segundos la estará todo en funcionamiento y tendremos disponible la BD para hacer las consultas que necesitemos.

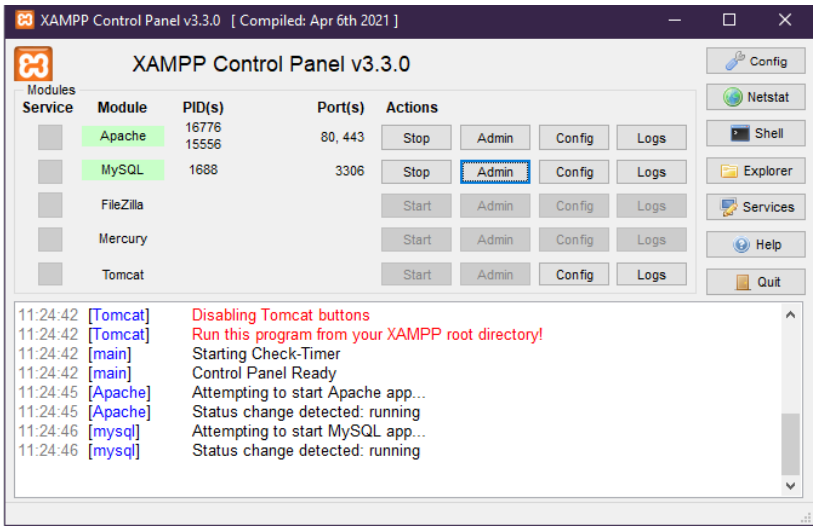


Ilustración 32: XAMPP

El siguiente elemento que debemos arrancar es el backend, para ello debemos dirigirnos a IntelliJ y abrir nuestro proyecto JAVA. El proyecto tiene una clase encargada de ejecutar la aplicación por lo que deberemos ejecutarla dando click derecho sobre la misma y seguidamente hacer click en "Run Application". Cómo resultado podremos ver en la consola que la aplicación se está ejecutando en el puerto 8080.

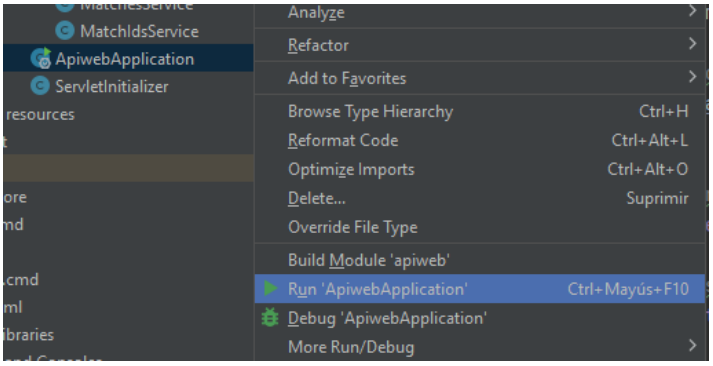


Ilustración 33: Ejecución Backend

Por último queda poner en marcha en frontend, para ello debemos dirigirnos a la consola de Visual Studio dónde estamos editando nuestro proyecto, ya que es necesario usar líneas de comando para encender el proyecto. Desde el terminal debemos dirigirnos al directorio en el que se encuentra la aplicación y escribir la orden "npm start".

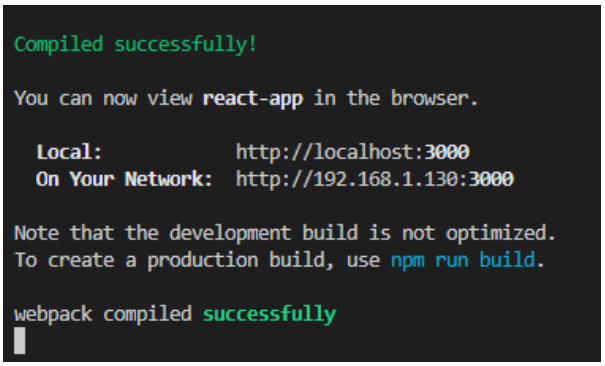


Ilustración 34: Ejecución Frontend

Cómo consecuencia, el IDE pone en funcionamiento nuestro frontend en el puerto 3000, con lo que ya tendríamos todos los elementos de nuestra aplicación en funcionamiento. De este modo solo debemos dirigirnos al navegador e introducir la URL de la página de inicio, si tuviéramos un dominio y hubiéramos desplegado el servicio en un servidor podríamos acceder a ella a través de su nombre "Anivia.gg", como de momento el proyecto está desplegado de forma local tendremos que acceder a él a través de la URL que nos indica React: "http://localhost:3000".

7. Producto desarrollado

Una vez habiendo acabado la codificación y habiendo puesto en marcha todo el sistema, el último paso es comprobar que nuestra aplicación se adecúa a los casos de uso que hemos definido. Para ello iremos uno por uno comprobando que podemos cumplir la funcionalidad descrita en cada caso y además lo compararemos con los bocetos diseñados.

En primer lugar encontramos la página de inicio de la aplicación, en comparación con el boceto diseñado podemos ver que no se ha incluído el extracto de las tablas de otras pestañas, pero esto no afecta a la funcionalidad que se había diseñado. En los casos de uso solo se especifica como necesario un buscador y la capacidad de seleccionar un servidor y como podemos ver en el resultado final, encontramos un selector al cual podemos hacer click para seleccionar el servidor y un buscador dónde introducir un nombre.

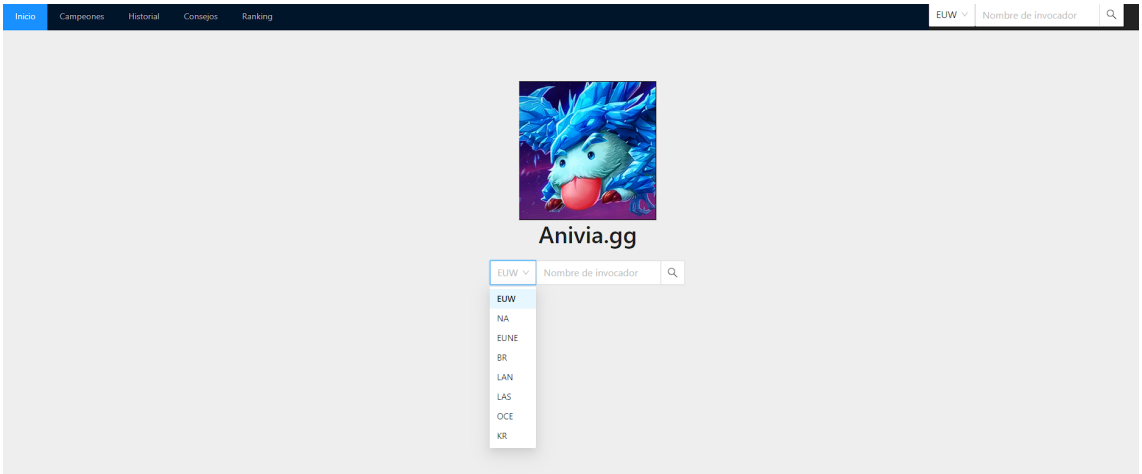


Ilustración 35: Producto final 1 - Página de Inicio

Para completar los requisitos del escenario uno no basta con la página principal sino que el buscador nos debe llevar al historial del jugador. De este modo introducimos un nombre de un jugador que existe y la página nos dirige hasta la pestaña de historial del jugador que hemos introducido. Una vez aquí podemos hacer todo lo especificado en los casos de uso, es decir, podemos ver de forma sencilla su rango, tasa de victorias, Puntos de Liga y el resultado de sus últimas partidas.

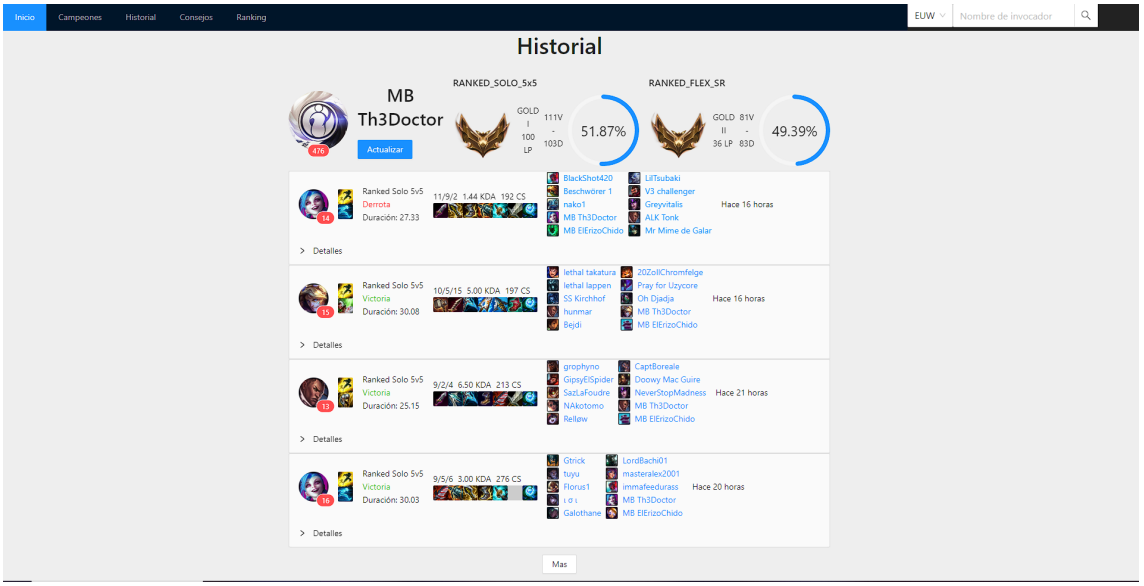


Ilustración 36: Producto final 2 - Historial 1

Siguiendo con el segundo escenario planteado y partiendo de la imagen anterior, tan solo debemos de hacer click en "Detalles" para que se nos despliegue más información acerca de los demás integrantes de la partida. Además tenemos la posibilidad de acceder a descripciones acerca de algunos conceptos poniendo el indicador del ratón sobre ellos. De este modo el segundo escenario queda completamente cubierto.

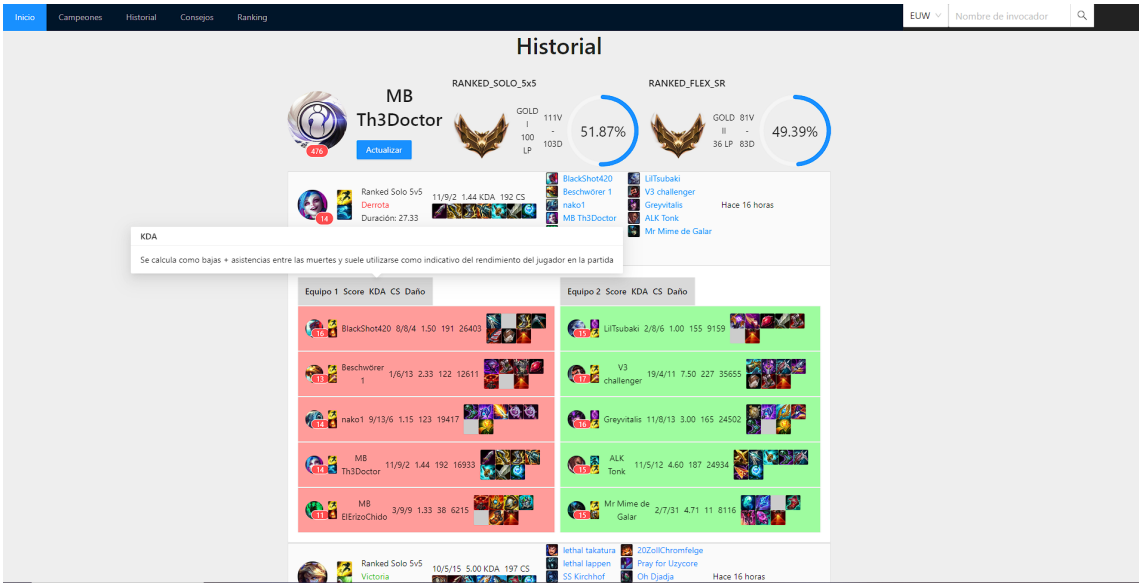


Ilustración 37: Producto final 3 - Historial 2

A continuación comprobaremos que el escenario tres de los casos de uso ha sido cubierto. Para ello podemos acceder desde cualquier parte de nuestra aplicación a la pestaña "Campeones" a través del menú superior que se encuentra fijo en toda la aplicación. Una vez allí se nos muestra directamente una tabla con la información relevante acerca de cada personaje dónde podemos buscar el personaje por su nombre o ordenar la tabla según la estadística que creamos más relevante.

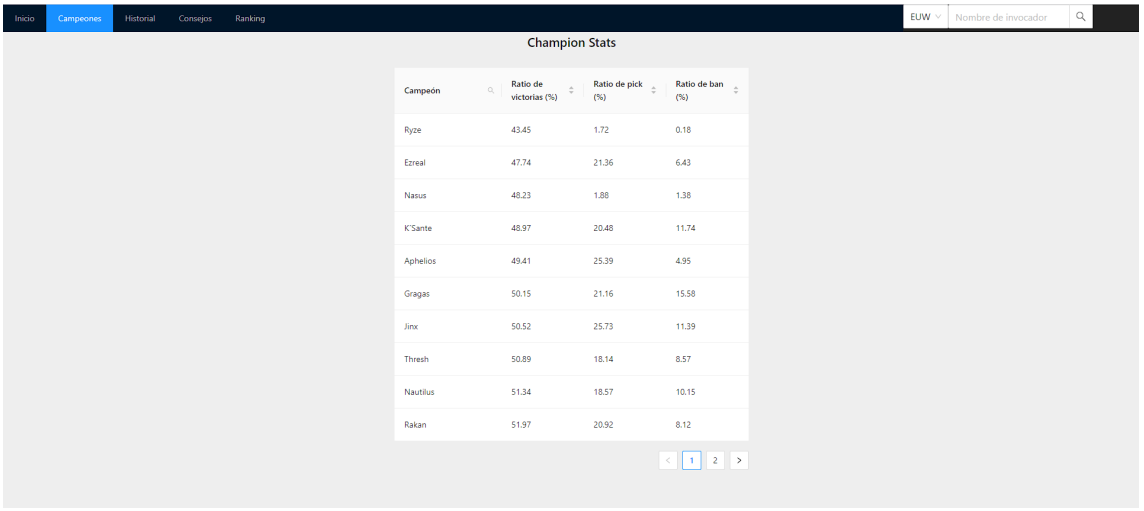


Ilustración 35: Producto final 4 - Campeones

El cuarto escenario habla de la pestaña de ranking de jugadores, para acceder a ella tan solo debemos de hacer click en el botón "Ranking" que aparece en el menú superior y la página nos redirigirá hasta la pestaña. En ella podemos ver una tabla con los 300 mejores jugadores del servidor y cola por defecto ordenados por Puntos de Liga (LPs). También tenemos la posibilidad de cambiar el servidor y la cola por medio de los selectores que se encuentran encima de la tabla que cambiarán automáticamente el contenido de la misma con los nuevos resultados.

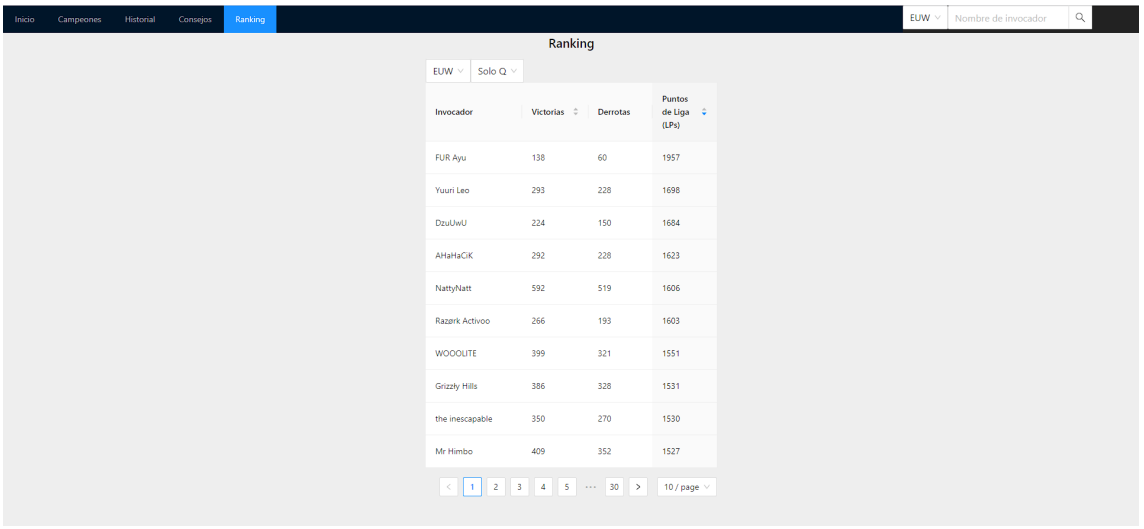


Ilustración 35: Producto final 5 - Ranking

El último caso de uso especifica sobre la pestaña "Consejos", para acceder a ella tenemos que seguir el mismo proceso que los dos casos anteriores, es decir, acceder a través del menú superior. Una vez en la pestaña podemos ver que se incluye todo lo que habíamos especificado, conteniendo consejos separados en niveles. Estos consejos vienen en forma de vídeo que podemos visualizar directamente desde nuestra web a través de un reproductor de Youtube integrado que nos evita tener que dirigirnos hacía la plataforma donde reside el vídeo.

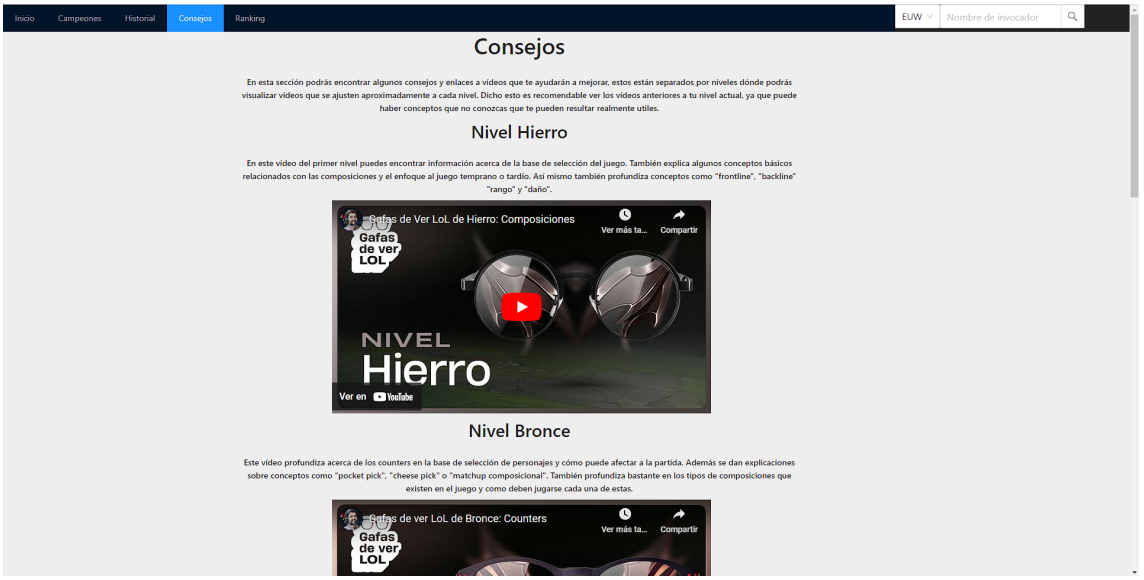


Ilustración 35: Producto final 6 - Consejos

Con esto habríamos cubierto todos los casos de uso que se especificaron en el tercer apartado de la memoria y podemos ver que el resultado obtenido satisface completamente todas las funcionalidades previstas en un inicio.

Además podemos ver que el producto final tiene muchas similitudes con los bocetos diseñados, esto es así ya que la implementación se ha hecho a partir de los mismos, intentando seguir los patrones de diseño que habíamos propuesto.

8. Conclusiones

Durante este trabajo hemos conocido más acerca de la importancia de la industria de los videojuegos y como los productos de su entorno son una oportunidad de negocio muy a tener en cuenta, sobre todo si hablamos de software.

El resultado final del proyecto ha sido una aplicación web SPA que proporciona un servicio de análisis a los jugadores que pretenden mejorar en el videojuego. Hemos cumplido con todos los requisitos que se plantearon al comienzo del trabajo, dando como resultado una herramienta que pueda ser de utilidad a los jugadores más novatos.

Para conseguir nuestro objetivo hemos hecho uso de la metodología DCU y hemos comprobado de primera mano su efectividad. Puede parecer una forma de trabajo tediosa debido a la cantidad de tiempo que requiere para la planificación, pero es muy beneficioso para obtener resultados que los usuarios encuentren atractivos. Además, el hecho de tener siempre en cuenta a los usuarios ayuda a que el resultado final sea probablemente de su gusto, lo que aumenta la probabilidad de éxito del proyecto.

Cabe resaltar que este proyecto ha sido de gran ayuda para el aprendizaje de nuevas tecnologías que se utilizan para el desarrollo de páginas web profesionales, estas son concretamente React y Springboot. Partiendo del total desconocimiento de su uso, al finalizar el proyecto hemos conseguido tener un dominio bastante alto de estas tecnologías, que va a resultar muy beneficioso en el entorno laboral.

Si debiéramos resaltar un error en este proyecto sería haber planteado un trabajo demasiado complicado en un inicio. La idea principal del proyecto era realizar una aplicación que pudiera interactuar a tiempo real con el videojuego, debido al desconocimiento de las tecnologías empleadas para ello, se empleó demasiado tiempo en un proyecto que no era abordable en un lapso de tiempo razonable.

8.1 Relación del trabajo desarrollado con los estudios cursados

Reflexionando acerca de la relación del proyecto con el grado cursado, cabe decir que todos los elementos que forman el trabajo tienen relación con al menos una asignatura.

La única excepción que encontramos a esto son los videojuegos, ya que a pesar de que hay asignaturas optativas que tratan el tema del desarrollo de videojuegos, personalmente nosotros no hemos cursado ninguna de ellas. Dicho esto, dentro del proyecto no se lleva a cabo ninguna actividad relacionada con el desarrollo de videojuegos por lo que no ha supuesto ningún problema, tan solo cabe mencionar que es un tema muy relacionado a la Ingeniería Informática y que existen asignaturas que lo respaldan.

Dentro de las asignaturas que se han cursado durante el grado y que han sido de gran utilidad a la hora del desarrollo del proyecto podemos encontrar las siguientes:

- La asignatura de Bases de Datos y Sistemas de la Información (BDA) y Tecnología de Bases de Datos (TBD) que nos han ayudado a conocer como crear y gestionar una Base de Datos, por lo que hemos sido capaces de gestionar nuestra propia Base de Datos para nuestra aplicación sin ningún tipo de problema, habiendo empleado además las herramientas que aprendimos a usar dentro de estas asignaturas.
- Integración de Aplicaciones (IAP), que fue la que nos introdujo los servicios REST y nos enseñó a integrar diferentes sistemas, dentro del proyecto no solo hemos consumido una API REST externa sino que además hemos creado la nuestra propia y hemos integrado tres servicios distintos en uno.
- Desarrollo Centrado en el Usuario (DCU), fue la asignatura que nos enseñó todo acerca de la metodología con el mismo nombre, que ha sido la escogida para desarrollar el proyecto.
- En Introducción a la Programación (IP) y Programación (PRG) aprendimos a desarrollar código en Java que ha sido uno de los lenguajes escogidos para el desarrollo del proyecto. Además si sumamos a estas la asignatura de Ingeniería del Software (ISW), estas tres nos han enseñado a organizar e implementar aplicaciones.
- Por último encontramos Desarrollo Web (DEW), que nos enseñó a desarrollar servicios web que es principalmente lo que hemos llevado a cabo.

9. Trabajos futuros

Una vez finalizado el proyecto cabe mencionar las posibilidades de ampliación que tiene el mismo y vamos a centrarnos en la más obvia, ya que el siguiente paso lógico dentro del desarrollo del servicio sería ejecutarlo en un servidor para que fuera accesible por todos los usuarios de internet.

Para conseguir esto el primer paso sería cambiar la clave de nuestra API, cómo hemos comentado con anterioridad, la clave que poseemos para realizar consultas a la API del videojuego es una clave de uso personal, pensada para desarrolladores que quieren realizar un proyecto pequeño o para proyectos en sus primeras fases de desarrollo. Como el número de consultas que nuestro servicio crece en función de los usuarios que usan la aplicación, necesitamos una clave que nos permita realizar un número elevado de peticiones al servicio, esta es una clave de producción. La manera de conseguirlo es dirigirse al portal del desarrollador del juego dónde solicitamos la clave personal en el pasado y esta vez solicitar una nueva de producción. Para ello deberemos presentar el prototipo de nuestro servicio y los administradores deberán revisarlo y dar el visto bueno. Si nuestro prototipo cumple con sus requisitos, pasaremos a obtener una clave de producción.

A continuación, antes de subir la aplicación a la red debemos de obtener un dominio para nuestra web, esta es la dirección URL que identificará a nuestra web. Para ello deberemos buscar un dominio que esté disponible y proceder a comprarlo. Existen muchas webs que permiten registrar de forma rápida y sencilla un dominio, haciendo una búsqueda rápida en una de estas webs elegidas al azar podemos comprobar que dominio que buscábamos obtener desde un principio “anivia.gg” se encuentra disponible y si comparamos un poco los precios podemos ver que ronda entre los 50 y 100 € anuales. Aunque no es un precio desconmensurado, tenemos la posibilidad de cambiar el dominio a .com o .es dónde podemos encontrar ofertas mucho más económicas pero esto significaría no acoplarse al estándar que existe entre la competencia.

Una vez dispongamos del dominio deseado solo queda buscar un servicio que nos ofrezca servidores en los cuales alojar los distintos componentes que forman el servicio. Los principales proveedores de servicios dónde podríamos alojar la web son Amazon Web Services (AWS) o Azure pero podríamos buscar otras opciones si fuera necesario. Ejecutando nuestros servicios en un servidor ya tendríamos la aplicación lista para ser accedida por cualquier usuario.

Otra característica que no se ha podido abordar por falta de tiempo es la implicación de los usuarios en el proyecto final desarrollado. Siguiendo la metodología DCU al pie de la letra, la fase final de aceptación del proyecto debería incluir de algún tipo a los usuarios finales, de este modo comprobamos si están satisfechos con el resultado y hemos cumplido sus expectativas. Del mismo modo, la metodología DCU se trata de un proceso iterativo por lo que se debería recoger el feedback de los usuarios para volver a iniciar un proceso de desarrollo en el que implementar todo aquello que se nos indique.

Referencias

1. Richardson, L., & Amundsen, M. (2013). RESTful Web APIs. Sebastopol, CA: O'Reilly Media

2. International Organization for Standardization. (2019). ISO 9241-210:2019 Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems. Geneva, Switzerland: ISO

3. Unidad Internacional de Telecomunicaciones (2023). Informe sobre la medición de la sociedad de la información. URL: https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx Último acceso: Junio 2023

4. Baig A. Hall B. Jenkins P. Lamarre E. and McCarthy B. (2020). The COVID-19 recovery will be digital: A plan for the first 90 days. URL: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-covid-19-recovery-will-be-digital-a-plan-for-the-first-90-days#/ Último acceso: Junio 2023

5. Robert W. Crandall and J. Gregory Sidak (2006). Video Games: Serious Business for America's Economy. Entertainment Software Association Report

6. Statista (2023). Tamaño del mercado de la industria de los videojuegos en Estados Unidos desde 2010 hasta 2023. URL: https://es.statista.com/estadisticas/1320510/tamano-del-mercado-de-la-industria-de-los-videojuegos-en-los-ee-uu/ Último acceso: Junio 2023

7. Walls, C. (2021). Spring Boot in Action. Shelter Island, NY: Manning Publications

8. Banks, A., & Porcello, E. (2019). Learning React: Functional Web Development with React and Redux. Sebastopol, CA: O'Reilly Media

Anexo

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1.**Fin de la pobreza.**				X
ODS 2.**Hambre cero.**				X
ODS 3.**Salud y bienestar.**				X
ODS 4.**Educación de calidad.**				X
ODS 5.**Igualdad de género.**				X
ODS 6.**Agua limpia y saneamiento.**				X
ODS 7.**Energía asequible y no contaminante.**				X
ODS 8.**Trabajo decente y crecimiento económico.**		X		
ODS 9.**Industria, innovación e infraestructuras.**		X		
ODS 10.**Reducción de las desigualdades**.				X
ODS 11.**Ciudades y comunidades sostenibles**.				X
ODS 12.**Producción y consumo responsables.**				X
ODS 13.**Acción por el clima.**				X
ODS 14.**Vida submarina.**				X
ODS 15.**Vida de ecosistemas terrestres.**				X
ODS 16.**Paz, justicia e instituciones sólidas.**				X
ODS 17.**Alianzas para lograr objetivos.**				X

A continuación se reflexionará acerca de la relación del trabajo realizado con los ODS, para ello vamos a nombrar aquellos objetivos con los que se piensa que están relacionados y el por qué.

Comenzando por el octavo objetivo, el **trabajo decente y crecimiento económico**, podemos encontrar una relación considerable con el trabajo que se ha desarrollado. Haciendo referencia al objetivo 8.3, que busca aumentar los puestos de trabajo decentes y el crecimiento de las pequeñas y medianas empresas, nuestro producto encaja directamente con este objetivo. Esto es así ya que el producto que se ha desarrollado es solo una pequeña parte de lo que son capaces de hacer los demás productos de la competencia, que están formados por un equipo de tamaño medio y que han demostrado que este tipo de proyecto es realmente sostenible, habiendo estado en funcionamiento por más de diez años. De este modo, si se pretendiera continuar con nuestro servicio de manera profesional, hay evidencias que demuestran que el proyecto es viable y sostenible, teniendo la posibilidad de crear un equipo de trabajo que generaría así nuevos puestos de empleo.

Siguiendo con el mismo ODS, tenemos los puntos 8.b y 8.6, que ambos están destinados al aumento de empleo en los jóvenes. Continuando con la propuesta expuesta anteriormente, si procediéramos a la creación de una empresa que llevara a cabo el desarrollo y soporte de una aplicación más profesional, cumpliríamos fácilmente estos dos objetivos de la misma forma. Esto es debido a que los profesionales que más se sienten atraídos por este tipo de proyecto son los jóvenes. Podemos mirar directamente a la empresa que hemos tomado como referencia principal, op.gg, que afirma estar formada por desarrolladores jóvenes que disfrutan de los videojuegos, de esta forma afirman conocer cómo desarrollar productos para jugadores, ya que ellos mismos lo son. Tener empleados que conocen las necesidades de los usuarios a los cuales están destinados los productos que desarrollan es esencial para el éxito del proyecto, por lo que tener una plantilla joven es un requisito que aumentaría las posibilidades del éxito de la empresa.

Otro de los objetivos que podemos encontrar relacionado con nuestro trabajo es el noveno, **Industria, innovación e infraestructuras,** dónde encontramos varios de sus objetivos alineados con la visión de nuestro producto.

Comenzando por el objetivo 9.2, el cual busca aumentar el empleo y el producto interno bruto; podemos tomar la misma premisa que se ha comentado para el punto anterior. Siendo nuestro proyecto un producto el cual se podría profesionalizar, ya hemos comentado anteriormente como puede generar empleos. Con respecto al producto interno bruto, al ser un producto software, existe la posibilidad de trabajar con un equipo remotamente, por lo que no hay necesidad de que los empleados fueran de un mismo país, pero si el objetivo es contribuir al PIB de nuestro país, la plantilla de la empresa puede estar fácilmente formada por empleados residentes en España.

Por último tenemos el objetivo 9.5, el cual busca mejorar la capacidad tecnológica fomentando la innovación. Podemos relacionar este objetivo con nuestro proyecto gracias a las tecnologías que estamos utilizando, ya que hemos buscado usar herramientas más novedosas y que más éxito están teniendo en la actualidad en los entornos profesionales. Destacamos Spring y React, que son las dos tecnologías principales en las que se ha desarrollado el proyecto y las cuales son altamente demandadas en el entorno laboral.