

ELABORACIÓN DE INFORMES
AUTOMATIZADOS SOBRE DATOS HISTÓRICOS:
LA LIGA ESPAÑOLA DE FÚTBOL
AUTOMATED REPORTING ON HISTORICAL
DATA: THE SPANISH FOOTBALL LEAGUE



TRABAJO FIN DE GRADO
CURSO 2023-2024

AUTORES
ADRIANA FERNÁNDEZ DE MATOS
ALBA GORRINDO MARTÍNEZ
DANIEL CANO SEVILLA
PABLO REGIDOR GARCÍA

DIRECTOR
RUBÉN FUENTES FERNÁNDEZ

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

AUTOMATED REPORTING ON HISTORICAL DATA: THE SPANISH FOOTBALL LEAGUE

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTORES

ADRIANA FERNÁNDEZ DE MATOS

ALBA GORRINDO MARTÍNEZ

DANIEL CANO SEVILLA

PABLO REGIDOR GARCÍA

DIRECTOR

RUBÉN FUENTES FERNÁNDEZ

CONVOCATORIA: SEPTIEMBRE 2024

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

13 DE SEPTIEMBRE DE 2024

RESUMEN

La elaboración de informes históricos sobre un tema en el periodismo demanda un enorme esfuerzo. En primer lugar, se invierte mucho tiempo consultando fuentes de diversa índole, como noticias, informes y otros documentos que, además, suelen encontrarse en distintas localizaciones, ej. bases de datos o páginas web, y con distintas estructuras, ej. texto y tablas. Tras recabar toda esa información, hay que filtrarla para extraer la más relevante para cada caso concreto. Después, hay que procesarla y elaborarla para generar los formatos y el texto requeridos, mejorar su legibilidad y extraer y destacar la información más importante.

El presente Trabajo de Fin de Grado (TFG) tiene como objetivo simplificar la elaboración de estos informes usando los desarrollos recientes en Inteligencia Artificial Generativa (IAG), una subcategoría dentro de la inteligencia Artificial (IA) capaz de generar texto, imágenes y, en definitiva, contenido nuevo, a partir de modelos de aprendizaje profundo entrenados con la suficiente cantidad de datos de calidad. En el contexto de la generación de informes, la IAG está emergiendo como una herramienta muy valiosa para optimizar el flujo de trabajo y proporcionar análisis más detallados. Los profesionales ahora pueden aprovechar estas capacidades para centrarse en otras tareas mientras la IA se encarga del procesamiento y síntesis de información.

Este trabajo prueba la aproximación con IAG mediante la creación de una aplicación para la generación de informes deportivos. La aplicación extrae información web utilizando web scraping, almacena parte de ella en una base de datos y usa una IAG para elaborar los informes conforme a ciertas pautas (algunas que están embebidas en la aplicación y otras proporcionadas por el usuario). Además, se incluye una interfaz web de usuario para facilitar la interacción con el sistema, fundamentalmente especificar los informes a generar y los datos relevantes sobre ellos y examinar los resultados.

El dominio elegido para los informes es la liga española de fútbol. Los informes que se pueden generar son relativos a un partido, jornada o temporada, y el resultado es un informe textual sobre los datos seleccionados a partir de los datos disponibles en las fuentes web que usa la aplicación.

Palabras clave

Liga de fútbol, informes deportivos, generación automática de texto, Inteligencia Artificial Generativa, Modelo de lenguaje de gran tamaño.

ABSTRACT

Historical reporting in journalism requires a lot of effort. First, a lot of time is spent consulting sources of various kinds, such as, news, reports and other documents, which are often found in different locations, e.g. databases or websites, and with different structures, e.g. text and tables. Once all this information has been gathered, it needs to be filtered to extract the most relevant information for each specific case. It then needs to be processed and edited to generate the required formats, improve its readability and extract the most relevant information.

The aim of this Final Degree Project (Spanish acronym TFG) is to facilitate the production of these reports using the latest developments in Generative Artificial Intelligence (Generative AI or GAI), a subcategory of AI capable of generating text, images and, in short, new content from deep learning models trained with sufficient amount of quality data. In recent years, we have seen a significant growth in the application of GAI in various domains. In the context of reporting, GAI is proving to be a valuable tool for workflow optimization and more detailed analysis. Users can now use these capabilities to focus on other tasks, while AI takes care of information processing and synthesis.

This paper tests this approach with GAI by creating an application for generating sports reports. The application extracts web information using web scraping, stores part of it in a database and uses GAI to produce the reports according to certain guidelines (some hard-coded as parts of the application and others provided by the user). In addition, a web user interface is included to facilitate the interaction with the system, essentially specifying the reports to be generated and the relevant data about them and examining the results.

The chosen domain for the reports is the Spanish football league. The reports that can be generated are related to a match, a game week, or a season, and the result is a textual report on the selected data based on the information available from the web sources used by the application.

Keywords

Web scraping, Artificial Intelligence, Large Language Model (LLM), Database, Web interface, Generative Artificial Intelligence

ÍNDICE DE CONTENIDOS

Contenido

| | |
|--|-----|
| Resumen | III |
| Abstract..... | V |
| Índice de contenidos | VI |
| Índice de figuras..... | XI |
| 1. Introducción | 1 |
| 1.1. Motivación..... | 1 |
| 1.2. Objetivos | 1 |
| 1.3. Plan de trabajo | 3 |
| 1.3.1. Aproximación general..... | 3 |
| 1.3.2. Planificación temporal | 4 |
| 2. Estado de la cuestión..... | 11 |
| 2.1. Power BI aplicado al fútbol | 11 |
| 2.2. BeSoccer Pro | 14 |
| 2.3. Tagg-maker..... | 16 |
| 2.4. Anafut | 18 |
| 2.5. Conclusiones | 18 |
| 3. Requisitos de la aplicación..... | 21 |
| 3.1. Requisitos funcionales | 21 |
| 3.2. Requisitos no funcionales | 22 |
| 4. Tecnologías utilizadas..... | 23 |
| 4.1. Tecnologías y lenguajes de programación | 23 |

| | |
|--|----|
| 4.2. Herramientas | 24 |
| 4.3. IAG..... | 25 |
| 4.3.1. GAN | 26 |
| 4.3.2. VAE..... | 27 |
| 4.3.3. Modelos basados en transformadores..... | 28 |
| 4.3.4. Casos de uso | 29 |
| 4.3.5. Nuestra elección | 30 |
| 4.3.6. Conclusiones..... | 33 |
| 4.4. Web scraping | 34 |
| 4.5. Aplicación web | 36 |
| 4.5.1. Elección del framework backend | 36 |
| 4.5.2. Elección del framework frontend | 37 |
| 5. Arquitectura del proyecto | 39 |
| 5.1. Arquitectura general (alto nivel)..... | 39 |
| 5.2. Flujo de información | 39 |
| 5.3. Diseño detallado (bajo nivel) | 41 |
| 5.3.1. Diseño base de datos..... | 41 |
| 5.3.2. Diseño de recopilación, tratamiento y almacenamiento de datos | 43 |
| 5.3.3. Construcción de la interfaz web..... | 45 |
| 5.3.3.1. Organización general | 45 |
| 5.3.3.2. Estructura de archivos..... | 45 |
| 5.3.3.3. Patrones de la aplicación | 47 |
| 5.3.3.4. Gestión de usuarios | 48 |
| 5.4. Integración con modelo IA generativa para generación de crónicas..... | 49 |

| | |
|---|----|
| 6. Manual de uso | 51 |
| 6.1. Manual de uso: Cliente..... | 51 |
| 6.2. Manual de uso: Administrador | 55 |
| 7. Pruebas de ejecución..... | 59 |
| 7.1. Resumen de un partido | 59 |
| 7.1.1. Input..... | 59 |
| 7.1.2. Output | 60 |
| 7.2. Resumen de una jornada | 64 |
| 7.2.1. Input..... | 65 |
| 7.2.2. Output | 66 |
| 7.3. Resumen de una temporada..... | 70 |
| 7.3.1. Input..... | 71 |
| 7.3.2. Output | 72 |
| 8. Conclusiones y trabajo futuro..... | 77 |
| 8.1. Conclusiones | 77 |
| 8.2. Trabajo a futuro..... | 78 |
| 9. Introduction | 81 |
| 9.1. Motivation..... | 81 |
| 9.2. Objectives | 81 |
| 9.3. Work plan..... | 83 |
| 9.3.1. General approach | 83 |
| 9.3.2. Time planning..... | 84 |
| 10. Conclusions and future work | 89 |
| 10.1. Conclusions | 89 |

| | |
|--|-----|
| 10.2. Future Work | 90 |
| Introducción al despliegue..... | 91 |
| Instalación de herramientas | 92 |
| Preparación de la base de datos..... | 92 |
| Archivos de configuración | 93 |
| Configuración archivo de configuración del script de obtención/tratamiento/almacenamiento de datos | 93 |
| Configuración archivo de configuración para la funcionalidad de la aplicación web..... | 94 |
| Proceso de obtención, tratamiento y almacenamiento de datos..... | 95 |
| Despliegue de la aplicación web | 96 |
| Apéndices | 105 |

ÍNDICE DE FIGURAS

| | |
|--|----|
| Figura 1. Diagrama de Gantt para detallar los sprints del proyecto | 4 |
| Figura 2. Cuadro de scouting Power BI | 12 |
| Figura 3. Panel de estadísticas de la Real Sociedad en Power BI | 13 |
| Figura 4. Panel estadísticas Vinicius en Power BI | 13 |
| Figura 5. Tiros a favor y en contra por equipos en BeSoccer Pro | 15 |
| Figura 6. Ficha jugador en BeSoccer Pro | 15 |
| Figura 7. Informe generado por Tagg Maker | 17 |
| Figura 8. Arquitectura tipo GAN..... | 27 |
| Figura 9. Arquitectura tipo VAE | 28 |
| Figura 10. Arquitectura modelo basado en transformadores | 29 |
| Figura 11. Flujo de información del proyecto | 40 |
| Figura 12. Datos almacenados de un partido..... | 42 |
| Figura 13. Diagrama de flujo de script de recopilado/tratado/almacenado de datos | 44 |
| Figura 14. Estructura de archivos del proyecto | 46 |
| Figura 15. Modelo-Vista-Template | 48 |
| Figura 16. URL acceso a la aplicación web..... | 51 |
| Figura 17. Panel de inicio de sesión | 51 |
| Figura 18. Panel de registro..... | 52 |
| Figura 19. Página temporadas..... | 53 |
| Figura 20. Página jornadas | 53 |
| Figura 21. Página partidos | 54 |
| Figura 22. Crónicas de un partido generadas..... | 55 |

| | |
|---|----|
| Figura 23. Página del administrador | 57 |
| Figura 24. Input datos estadísticos resumen de partido..... | 60 |
| Figura 25. Eventos relevantes de la jornada | 60 |
| Figura 26. Crónica 1 de un partido | 61 |
| Figura 27. Crónica 2 de un partido | 63 |
| Figura 28. Prompt input para generar crónica de una jornada | 65 |
| Figura 29. Input datos para crónica de una jornada (parte 1) | 65 |
| Figura 30. Input datos para crónica de una jornada (parte 2) | 66 |
| Figura 31. Crónica 1 de una jornada..... | 67 |
| Figura 32. Crónica 2 de una jornada..... | 69 |
| Figura 33. Prompt del input para generar la crónica de una temporada..... | 71 |
| Figura 34. Input para crónica de temporada (parte 1) | 71 |
| Figura 35. Input para crónica de temporada (parte 2) | 71 |
| Figura 36. Input para crónica de una temporada (parte3) | 72 |
| Figura 37. Input para crónica de una temporada (parte 4) | 72 |
| Figura 38. Crónica 1 de una temporada | 73 |
| Figura 39. Crónica 2 de una temporada | 75 |
| Figura 40. Gantt chart to detail project sprints | 84 |
| Figura 41. Contenido del archivo config.json | 93 |
| Figura 42. Contenido del fichero .env recopilación de datos | 94 |
| Figura 43. Contenido del fichero .env de la parte app web | 95 |

1.Introducción

1.1. Motivación

El fútbol es uno de los deportes más populares en el mundo y la Liga Española es una de las más famosas, seguidas y competitivas a nivel mundial. Los periodistas emplean mucho tiempo y esfuerzo en examinar un gran número de fuentes tratando de recabar todo tipo de información, como los resultados de los partidos, las clasificaciones, estadísticas o los jugadores que mejor rinden.

En una sociedad donde se premia la inmediatez con la que se obtiene la información, un sistema que proporcione la información necesaria para crear los informes y los genere bajo demanda y en un tiempo muy reducido sería de mucha ayuda. Este tipo de sistema podría tener un impacto muy relevante en ámbitos como el periodismo deportivo o bursátil donde se demandan las noticias con gran inmediatez, ya que minimizaría los tiempos de producción de los informes básicos y permitiría a los periodistas centrarse en analizar dichos informes y su personalización, al tiempo que adelantarse a su competencia a la hora de publicar sus trabajos.

1.2. Objetivos

El objetivo principal de este Trabajo Fin de Grado (TFG) es crear un sistema, con una interfaz web intuitiva y accesible capaz de elaborar informes automatizados utilizando información extraída automáticamente de la web e Inteligencia Artificial Generativa (IAG) para producir el texto. El dominio específico será el de la información deportiva sobre la liga de fútbol española de primera división. Los informes de la aplicación abarcarán crónicas deportivas que describan lo sucedido en un partido concreto, el resumen de cada jornada de la temporada o el resumen de una temporada al completo. Además, se podrá consultar la evolución de un equipo a lo largo de la temporada y quiénes han sido los jugadores más destacados de cada temporada, es decir, goleadores, asistentes o jugadores con más tarjetas rojas y amarillas.

Para conseguir este objetivo global, se plantean los siguientes objetivos específicos desde el punto de vista del desarrollo del sistema:

- Investigación de diversas fuentes de información de diferentes tipos sobre las temporadas de la liga de fútbol con el fin de obtener la información a partir de la cual se generarán los informes. Su extracción se realizará mediante el uso de web scraping y un LLM (*Large Language Model*). El web scraping permitirá extraer información de la web (tanto estructurada, ej. estadísticas, como no, ej. texto) y el LLM se usará en esta fase para obtener de los textos (noticias y otras crónicas ya existentes) eventos no recogidos en las estadísticas de los diferentes partidos. Tras la obtención de todos estos datos se procederá a su exportación a un formato manejable por el sistema.
- Almacenamiento de los datos obtenidos a una base de datos para optimizar el tiempo de acceso a información que no cambiará una vez extraída, como pueden ser resultados de partidos, goles o tarjetas.
- Elaboración de los informes mediante una IAG de texto que, a través de un prompt (mecanismo para suministrarle entradas) recibirá la información que necesite para la elaboración del informe que el usuario le solicite. El informe devuelto contendrá el relato de un partido a partir de los eventos de éste, la descripción de la clasificación de la liga, el resumen de una jornada concreta de la temporada o el resumen de la propia temporada al completo. También se podrá obtener la evolución de un equipo a lo largo de una temporada al completo.
- Creación de una interfaz web que permita al usuario final (ej. periodista deportivo) interactuar con el sistema de manera sencilla e intuitiva y obtener los informes deseados.

1.3. Plan de trabajo

1.3.1. Aproximación general

Para el desarrollo de este proyecto hemos optado por el uso de Scrum (Scrum.org, s.f.), una metodología ágil que creemos que es la que mejor se adapta a nuestras circunstancias actuales y a nuestra forma de trabajar. Scrum se basa en hacer sprints cada cierto tiempo para ir avanzando sobre un prototipo inicial funcional que, en cada iteración (o sprint), va ganando nuevas funcionalidades hasta alcanzar la versión final.

Como nuestra idea era reunirnos con el tutor cada dos/tres semanas, vimos como una opción viable el hacer coincidir estas reuniones con cada una de las iteraciones del desarrollo del proyecto.

El grueso del trabajo se planteó a partir del final del segundo cuatrimestre, ya que la carga de trabajo entonces nos permitía dedicarnos de manera constante y consistente al proyecto. Sin embargo, durante el primer cuatrimestre avanzamos en la parte del web scraping con el fin de familiarizarnos con la extracción de datos y durante el segundo progresamos con la creación de la aplicación web e hicimos pruebas con los diferentes modelos y apis para conseguir el más idóneo con nuestros recursos para el desarrollo proyecto.

Las reuniones con el tutor se fueron ajustando según los avances realizados y la disponibilidad de los miembros y del tutor para tener un seguimiento continuo del estado del proyecto. Así garantizamos trabajar de manera constante y pudimos plantear objetivos a corto y medio plazo.

Para organizar el reparto de tareas dentro del grupo se optó por el uso de una herramienta muy utilizada en las metodologías ágiles, Trello. Gracias al uso de esta herramienta todos los miembros del equipo tenían claro cuáles eran sus tareas y las de sus compañeros, el estado pasado y actual de las mismas y las tareas pendientes.

Pasando a actividades más concretas, el proyecto se planteó de la siguiente forma. Se llevaría a cabo un proceso de investigación que involucrara diversas fuentes para recopilar información sobre las temporadas de la liga de fútbol. El objetivo era disponer de un conjunto rico

de datos para la aplicación. Después, esta información se exportaría a una base de datos con el fin de optimizar el acceso a la misma y garantizar la disponibilidad constante de datos como resultados de partidos y estadísticas relevantes. Para generar informes, se utilizaría una IAG de texto que trabajaría a partir de prompts proporcionados por los usuarios. Los informes que generaría abarcarían desde la descripción detallada de eventos de un partido hasta resúmenes de jornadas o temporadas completas, así como la evolución de los equipos a lo largo de una temporada. Finalmente, se desarrollaría una interfaz web intuitiva para que los usuarios pudieran interactuar directamente con el sistema y obtener los informes deseados.

1.3.2. Planificación temporal

En este apartado detallaremos un poco más cómo se ha dividido el trabajo realizado (explicado en el apartado anterior) en los diferentes sprints del proyecto. Esto es aproximado y las fechas indican la semana en la que se llevó a cabo cada reunión.

En el siguiente diagrama de Gantt (Figura 1. Diagrama de Gantt para detallar los sprints del proyecto) se pueden apreciar los diferentes sprints con los que hemos trabajado:

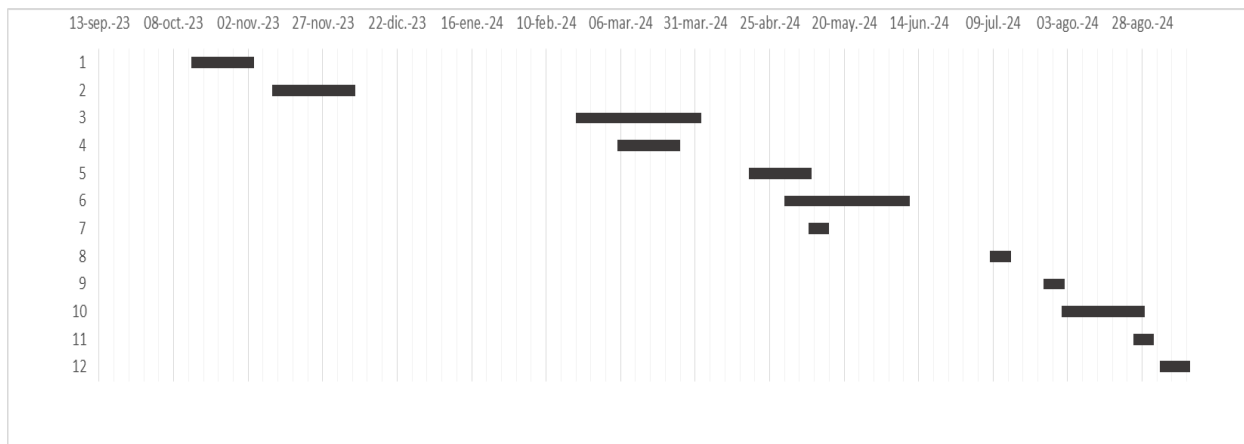


Figura 1. Diagrama de Gantt para detallar los sprints del proyecto

En el diagrama se pueden ver sprints solapados ya que, mientras unos se centraban una parte del proyecto y trabajaban en uno de los sprints, otros estaban con otros aspectos del proyecto y se centraban en otros. En determinados sprints vimos más útil esta aproximación que

estar todos trabajando en lo mismo, un ejemplo es que mientras Alba y Adriana estaban trabajando en las primeras pruebas con los diferentes modelos de IAG, Daniel estaba con el prototipo de interfaz web y Pablo con la introducción de la memoria. Cabe destacar también que, por experiencia previa/intereses a cada uno nos gustaba más una parte del proyecto, esto propició también que en muchos momentos del desarrollo trabajásemos en paralelo (memoria, interfaz web y modelos de IAG/almacenamiento de datos). De esta forma, juntábamos las “piezas” de la aplicación, probábamos su correcto funcionamiento y corregíamos los errores que no íbamos encontrando antes de continuar incrementando el modelo inicial (en nuestro caso este modelo fue el que era capaz de crear una crónica de un solo partido sin incluir eventos extradeportivos).

Aunque las reuniones se llevaron a cabo durante el curso para dejar claras las ideas, el grueso del proyecto se dejó para desarrollar en verano, ya que durante el curso todos cursamos asignaturas del grado y tuvimos prácticas en empresa. Esto junto a los exámenes nos consumió la mayoría del tiempo.

Reunión inicial - 13 septiembre 2023

Reunión inicial para hablar sobre la idea de proyecto a desarrollar y los pasos a seguir para lograrlo.

Sprint 1 - 14 octubre 2023 (3 semanas)

El objetivo fue investigar qué era el web scraping y las herramientas que existen para ello. Seleccionamos una y empezamos a familiarizarnos con ella y a hacer pruebas para descubrir todas sus posibilidades.

Sprint 2 -10 noviembre 2023 (4 semanas)

El objetivo fue ser capaces de extraer los datos sobre todos los partidos que se habían jugado durante una jornada (ej. número de la jornada, equipos enfrentados, resultado, tarjetas

recibidas, autores de goles o cambios) con la herramienta que ya habíamos elegido en el sprint anterior. En este punto, gran parte del web scraping estaba acabado, de forma que contábamos con la información estadística necesaria para poder generar crónicas de partidos y jornadas en el desarrollo posterior.

Recuperación del proyecto segundo cuatrimestre - 14 febrero 2024

Una vez finalizados los exámenes intentamos seguir avanzando con el proyecto. El objetivo era revisar lo anteriormente hecho (la parte del web scraping) y organizarnos para volver a retomar el proyecto.

Sprint 3 - 20 febrero 2024 (6 semanas)

Hicimos una búsqueda de APIs (*Application Programming Interface*) gratuitas para usar la IAG en nuestra elaboración de informes. Hicimos pruebas y analizamos varias para ver cuál era la que más se adaptaba a lo que queríamos.

Empezamos a hacer pruebas con varios modelos y APIs que parecían que íbamos a utilizar y empezamos a familiarizarnos con ella. Queríamos ser capaces de generar crónicas preliminares de partidos. También analizamos si veíamos necesario guardar información en una base de datos o si lo queríamos hacer todo en tiempo real.

Sprint 4 - 5 marzo 2024 (3 semanas)

Mientras seguíamos con pruebas con los modelos y las APIs, decidimos que era preferible usar una base de datos para almacenar cierta información y evitar depender de su obtención bajo demanda.

En este punto se abordó unificar una interfaz web sencilla, con la base de datos, el web scraping y la IAG para poder realizar todo el flujo de trabajo de generar una crónica completa.

En este punto, la memoria estaba empezada también.

Sprint 5 - 18 abril 2024 (3 semanas) Ya estaba la base de datos creada la base de datos en la que guardamos los datos de los partidos, seguíamos haciendo pruebas con los modelos.

En este momento estas pruebas se ampliaron para poder generar crónicas de jornadas completas y resúmenes de temporadas enteras, además de las de los partidos que ya deberían funcionar.

Así mismo, montamos la parte del proyecto de extracción de datos con selenium en un servidor ahora que disponíamos de las credenciales.

La introducción de la memoria estaba finalizada.

Sprint 6 - 30 abril 2024 (6 semana)

Haber revisado y corregido la memoria con las correcciones dadas por el tutor. Además, empezamos a desarrollar la que sería la interfaz web definitiva (ya que antes teníamos una versión muy sencilla para hacer las primeras pruebas de interacción entre el usuario y el modelo de IAG).

Sprint 7 – 8 mayo 2024 (1 semana)

La memoria se avanzó para dejar explicado lo que teníamos hecho hasta el momento antes de empezar exámenes.

Recuperación del proyecto tras exámenes del segundo cuatrimestre - 28 junio 2024

Una vez finalizados los exámenes intentamos avanzar con lo que nos faltaba del proyecto.

El objetivo era revisar los resultados obtenidos de las pruebas con los modelos (fijándonos en los resultados de interacciones con diferentes IAGs obtenidos con las pruebas en

Google Colab) e ir decantándonos por el que mejor respondiera nuestras consultas y menos recursos utilizase para poder ejecutar en local y no depender de un servidor.

Sprint 8 – 8 julio 2024 (3 semanas)

Quisimos recoger además de los datos estadísticos, otros datos no estadísticos que enriquecieran las crónicas generadas, para lo que usamos selenium y esos datos los almacenamos en la base de datos que ya teníamos creada.

Seguimos probando modelos con el objetivo de ser capaces de generar crónicas deportivas con estos eventos extradeportivos recogidos.

Sprint 9 – 26 julio 2024 (1 semana)

Memoria actualizada con las nuevas incorporaciones del proyecto.

Sprint 10 – 1 agosto 2024 (4 semanas)

Elección final de la API de Gemma para la generación de crónicas tanto de partido y jornada como de temporada.

Implementación de la conexión entre web y script de generación de crónicas y finalización de la web para que funcione correctamente.

Sprint 11 – 25 agosto 2024 (1 semana)

Finalización de la memoria con las nuevas actualizaciones. Entrega del borrador del proyecto y memoria con las primeras correcciones ya implementadas.

Sprint 12 – 3 septiembre 2024 (10 días)

El objetivo es haber implementado todas las mejoras dadas después de la entrega parcial del proyecto y de la memoria y hacer la entrega final del trabajo de fin de grado

2. Estado de la cuestión

Si se quiere desarrollar un sistema capaz de elaborar informes automatizados relacionados con el fútbol, se debe empezar estudiando cómo se realiza el proceso actualmente (en buena parte de forma manual), las aplicaciones o servicios existentes que ofrecen servicios similares a los que se plantean en este trabajo, cómo estos elaboran los informes, y observar qué tipos de datos e información son los más utilizados por dichos sistemas. De este modo, al trasladar este proceso a un sistema automatizado se podrán incorporar las características deseables, limitar los problemas y ver qué partes se pueden automatizar.

Aunque los informes sobre fútbol son comunes, no siempre es fácil encontrar información detallada sobre los procedimientos exactos utilizados para su elaboración. Sin embargo, se pueden identificar ciertos patrones y metodologías comunes entre muchos de estos informes, así como en sistemas similares de generación automática de informes deportivos. A continuación, se discuten algunos ejemplos significativos.

2.1. Power BI aplicado al fútbol

Power BI (Davidiseminger, 2024) es una herramienta integral, desarrollada por Microsoft, que combina software, aplicaciones y conexiones para transformar datos dispersos en información visualmente atractiva y coherente. Puede unir datos de diversas fuentes, como hojas de cálculo de Excel o almacenes de datos locales y en la nube.

Power BI es una herramienta que permite hacer Business Intelligence (tableau, s.f.) en diversos ámbitos como las finanzas, marketing, ventas, recursos humanos o incluso consultoría, gracias a su capacidad de análisis de datos y creación de informes. A pesar de que no existen versiones específicas, Power BI es muy utilizado en los ámbitos periodístico y futbolístico por todo tipo de usuarios, desde analistas de fútbol, ojeadores y entrenadores, hasta periodistas y medios de comunicación (Analista, 2023). A partir de unos datos bien estructurados, Power BI permite generar tablas específicas sobre diferentes aspectos. Por ejemplo, de jugadores, mostrando su país de origen, equipo actual, posición, valor de mercado, etc (Figura 2) Power BI

también ofrece la posibilidad de obtener un informe del desempeño de los equipos en distintas áreas, como progresiones al área, goles, tarjetas amarillas o tiros a puerta, asimismo desglosadas por cada jugador del equipo. También permite obtener informes personalizados de equipos, mostrando su posición actual en la clasificación de la liga, promedios de goles a favor y goles en contra, así como los totales, junto con breve información de cada jugador de la plantilla (ver Figura 3 y Figura 4). Power BI ofrece también la posibilidad de generar informes más específicos sobre los jugadores, como informes de análisis de comportamientos tácticos (por ejemplo, con el número de pases entre cada jugador) o médicos (el informe de lesiones de cada jugador en el que se muestra la severidad de la lesión, una descripción de esta y la fecha prevista de la vuelta del jugador).

| Finalización | Bandera | Foto | Nombre | Edad | Peso | Posición | Pie preferido | Club | Equipo | Valoración | Sueldo semanal | Valor de mercado | Fin de contrato |
|--------------|---------|------|-------------------|------|------|----------|---------------|---------------------|---------------------|------------|----------------|------------------|-----------------|
| 50 | FR | | L. Messi | 34 | 72 | RW | Left | Paris Saint-Germain | Paris Saint-Germain | 92 | 320 | 78 | 2023 |
| | FR | | K. Mbappé | 22 | 73 | ST | Right | Paris Saint-Germain | Paris Saint-Germain | 92 | 230 | 194 | 2022 |
| | PL | | R. Lewandowski | 32 | 81 | ST | Right | FC Bayern München | FC Bayern München | 92 | 270 | 120 | 2023 |
| | ES | | Cristiano Ronaldo | 34 | 83 | ST | Right | Manchester United | Manchester United | 91 | 270 | 45 | 2023 |
| | HR | | J. Doku | 28 | 87 | OK | Right | Atlético de Madrid | Atlético de Madrid | 91 | 130 | 112 | 2023 |
| | BE | | K. De Bruyne | 30 | 78 | CM | Right | Manchester City | Manchester City | 91 | 350 | 126 | 2025 |
| | BR | | Neymar Jr | 29 | 68 | LW | Right | Paris Saint-Germain | Paris Saint-Germain | 91 | 270 | 129 | 2025 |
| | NO | | E. Haaland | 20 | 94 | ST | Left | Borussia Dortmund | Borussia Dortmund | 90 | 110 | 138 | 2024 |
| | GB | | H. Kane | 27 | 89 | ST | Right | Tottenham Hotspur | Tottenham Hotspur | 90 | 240 | 130 | 2024 |
| | DE | | M. Neuer | 35 | 93 | OK | Right | FC Bayern München | FC Bayern München | 90 | 86 | 14 | 2023 |
| | DE | | M. ter Stegen | 29 | 85 | OK | Right | FC Barcelona | FC Barcelona | 90 | 250 | 99 | 2025 |
| | FR | | N. Kanté | 30 | 78 | CDM | Right | Chelsea | Chelsea | 90 | 230 | 100 | 2023 |
| | BR | | Casemiro | 29 | 84 | CDM | Right | Real Madrid CF | Real Madrid CF | 89 | 310 | 88 | 2025 |
| | BR | | Ederson | 27 | 84 | OK | Left | Manchester City | Manchester City | 89 | 200 | 94 | 2024 |
| | ES | | F. de Jong | 24 | 74 | CM | Right | FC Barcelona | FC Barcelona | 89 | 210 | 120 | 2026 |
| | IT | | G. Donnarumma | 22 | 98 | OK | Right | Paris Saint-Germain | Paris Saint-Germain | 89 | 110 | 120 | 2026 |
| | KR | | H. Son | 28 | 78 | LM | Right | Tottenham Hotspur | Tottenham Hotspur | 89 | 220 | 104 | 2025 |
| | DE | | J. Kimmich | 26 | 75 | CDM | Right | FC Bayern München | FC Bayern München | 89 | 160 | 108 | 2025 |
| | ES | | K. Benzema | 33 | 81 | CF | Right | Real Madrid CF | Real Madrid CF | 89 | 250 | 44 | 2023 |
| | EG | | M. Salah | 29 | 71 | RW | Left | Liverpool | Liverpool | 89 | 270 | 101 | 2023 |
| | BR | | Marquinhos | 27 | 75 | CB | Right | Paris Saint-Germain | Paris Saint-Germain | 89 | 135 | 90 | 2024 |
| | ES | | P. Dybala | 27 | 75 | CAM | Left | Juventus | Juventus | 89 | 160 | 93 | 2022 |
| | PT | | Rúben Dias | 26 | 82 | CB | Right | Manchester City | Manchester City | 89 | 170 | 102 | 2027 |
| | ES | | S. Mané | 29 | 69 | LW | Right | Liverpool | Liverpool | 89 | 270 | 101 | 2023 |

Figura 2. Cuadro de scouting Power BI

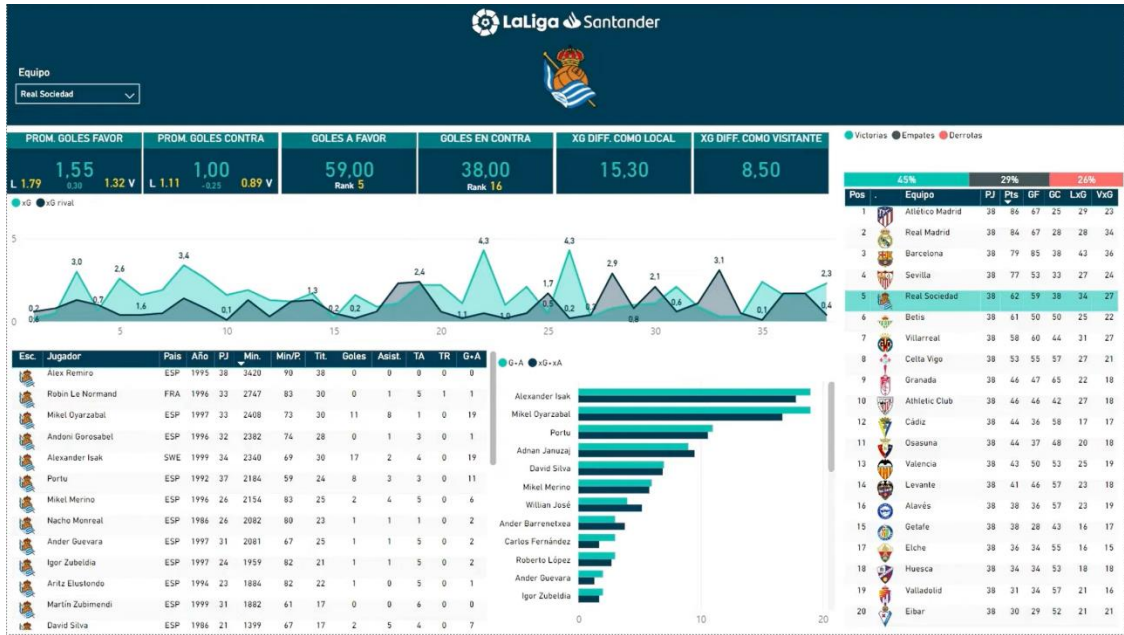


Figura 3. Panel de estadísticas de la Real Sociedad en Power BI

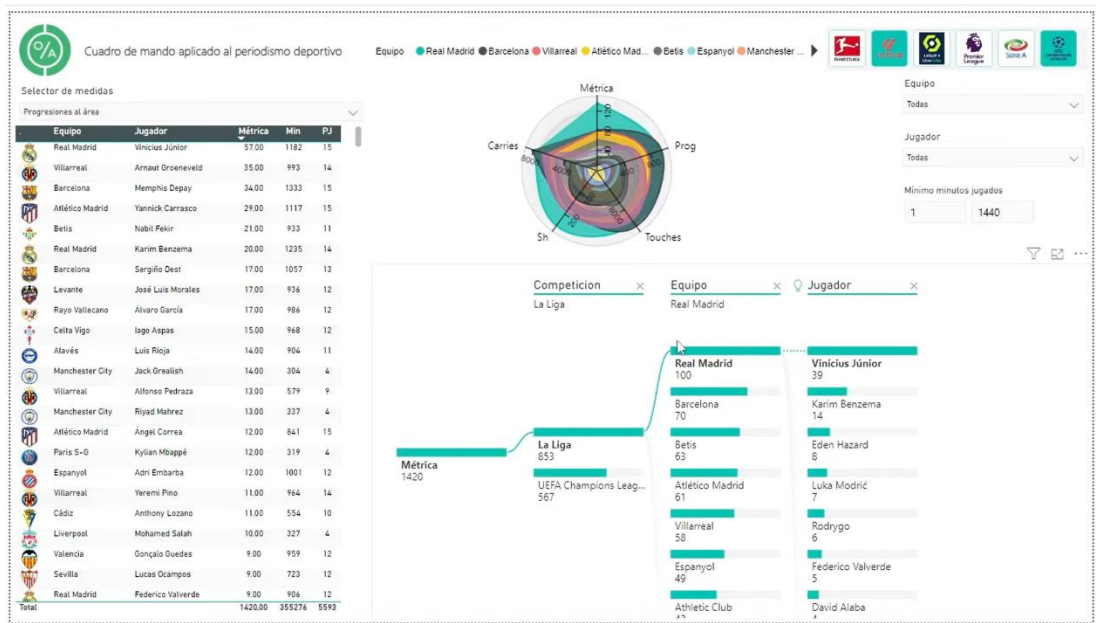


Figura 4. Panel estadísticas Vinicius en Power BI

2.2. BeSoccer Pro

BeSoccer Pro¹ es una herramienta de análisis de datos, rendimiento y scouting respaldada por una base de datos con información de más de 270.000 jugadores, 190.000 equipos y 1.000 competiciones. La mayor parte de estas funciones son de pago. BeSoccer es una herramienta altamente utilizada por clubes profesionales y semiprofesionales, agencias de representación, jugadores y hasta medios de comunicación y creadores de contenido.

BeSoccer Pro permite obtener informes específicos de jugadores, equipos y partidos. A nivel de jugador, se ofrecen estadísticas de la última temporada, goles, asistencias y partidos y minutos jugados, así como estadísticas por cada 90 minutos como pueden ser los regates con éxito, los duelos ganados e incluso un mapa de calor del jugador (Figura 5 Figura 6). BeSoccer Pro también ofrece un informe postpartido con todas las acciones llevadas a cabo por ambos equipos, tanto en defensa (ej. duelos defensivos ganados, recuperaciones o intercepciones), como en ataque (incluyendo tiros, tiros a puerta o regates con éxito).

Además de los informes ya mencionados, destaca el de adaptabilidad individual de los jugadores. Aquí se ofrece información de todo tipo acerca del jugador, incluyendo personal, trayectoria por los distintos equipos y selecciones en los que ha jugado e incluso un informe con todas sus lesiones, las zonas afectadas y los años en los que las sufrió. Lo más novedoso e importante de este informe es el propio apartado de adaptabilidad. Aquí se puede observar la similitud del equipo actual del jugador con el posible nuevo equipo y una comparación de los jugadores con mayor porcentaje de similitud en dichos equipos con el jugador del informe.

¹ <https://pro.besoccer.com/>

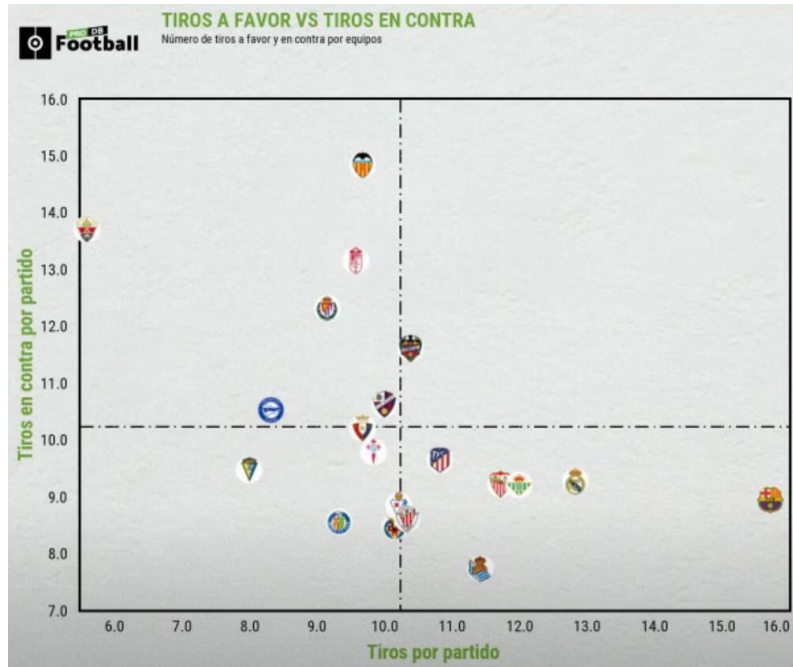


Figura 5. Tiros a favor y en contra por equipos en BeSoccer Pro

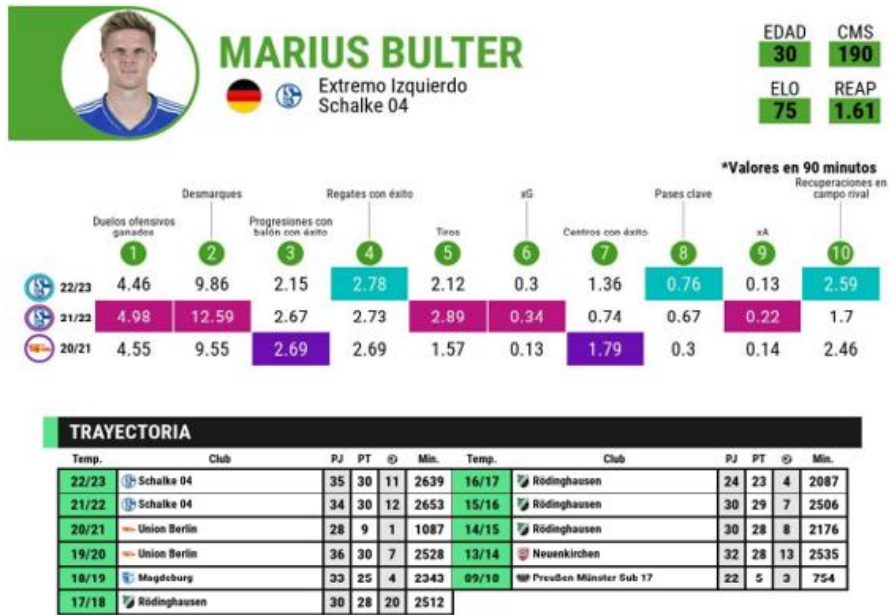


Figura 6. Ficha jugador en BeSoccer Pro

2.3. Tagg-maker

Tagg-maker² es una plataforma digital, principalmente de pago, diseñada para la recopilación, etiquetado y análisis de datos de partidos de fútbol. Se dirige fundamentalmente a entrenadores, analistas, ojeadores y cuerpo técnico de equipos y entidades deportivas. Tagg-maker combina una aplicación móvil intuitiva y un programa on-line de gestión, permitiendo así a los usuarios recopilar de manera fácil y sencilla los eventos y acciones de juego de cualquier partido de fútbol en tiempo real.

Tagg-maker aplica IA al análisis de datos. Utiliza algoritmos propios para identificar patrones de juego y calcular goles esperados o goles proyectados, lo que ayuda a evaluar el rendimiento de los equipos y jugadores. Tras el final del partido marcado, Tagg-maker ofrece tablas y gráficos a modo resumen/informe de lo acontecido en dicho partido, tanto del equipo local como del equipo visitante. La información devuelta por la aplicación incluye entre otros los goles a favor y en contra, llegadas al área, y porcentajes de los goles según el tipo de juego, a balón parado o juego abierto.

A continuación, se muestran un ejemplo de informe generado por esta herramienta (ver Figura 7).

² <https://tagg-maker.com/#caracteristicas>



| | | |
|----------------------|-----------|------------------------|
| Número de informes | 23 | Puntos por partido |
| Sistema más empleado | 3-4-3 B | Partidos ganados |
| Sistema secundario | 3-4-3 B | Partidos perdidos |
| Goles favor | 1.52 / 35 | Partidos empatados |
| Goles contra | 1.22 / 28 | Salidas balón centrado |
| Valor gol | 0.91 | Salidas balón largo |
| Llegadas | 70 | Duelos aéreos |
| Llegadas favor | 1.50 / 36 | Duelos aéreos |
| Llegadas contra | 1.42 / 34 | Recuperaciones |

Figura 7. Informe generado por Tagg Maker

2.4. Anafut

Anafut (Torrijos & Bran, 2019) es un bot de periodismo deportivo automatizado creado por el medio español El Confidencial en septiembre de 2017. Este bot se comenzó a utilizar a principios de 2018 para producir crónicas de partidos de fútbol de Segunda División B y Tercera División (Titania Cía. Editorial, S.L., s.f.). Anafut es capaz de generar informes objetivos de los partidos y jugadores participantes en el mismo, proporcionando análisis y descripciones detalladas de todos los eventos acontecidos en dichos partidos. El funcionamiento de este bot se basa en el uso de la IA, encargada de procesar los datos y generar las crónicas, combinada con la programación de rutinas informativas a partir de datos estadísticos. Este bot proporciona información de todo tipo, incluyendo posición en la liga de cada equipo, información relativa a los últimos partidos jugados por cada equipo, eventos importantes en cada mitad, los puntos de ambos equipos antes y después del partido o los enfrentamientos más esperados de la próxima jornada.

2.5. Conclusiones

La generación automatizada de informes en el ámbito del fútbol ha experimentado un crecimiento significativo recientemente como consecuencia del avance de tecnologías como la IA y el análisis de datos. A través de una variedad de herramientas y servicios, los analistas, entrenadores, ojeadores y medios de comunicación pueden acceder a informes detallados y visualmente atractivos sobre el desempeño de jugadores, equipos y partidos.

En este contexto, se han identificado varias similitudes entre las herramientas investigadas. En primer lugar, todas ellas ofrecen la capacidad de recopilar datos de diversas fuentes y generar informes personalizados para satisfacer las necesidades específicas de los usuarios. Además, todas integran elementos de visualización de datos, como tablas, gráficos y mapas de calor, para facilitar la interpretación de la información. Estas funciones se ofrecen con un soporte semiautomatizado.

Por otro lado, se observan diferencias significativas entre las herramientas en términos de características y funcionalidades. Algunas herramientas, como Power BI y BeSoccer Pro, se

centran en el análisis del rendimiento de jugadores y equipos, proporcionando estadísticas detalladas sobre goles, asistencias y tiros a puerta, entre otros. Otras, como Tagg-maker, se enfocan en la recopilación y etiquetado de datos en tiempo real durante los partidos, utilizando inteligencia artificial para analizar patrones de juego.

Respecto a la forma en que estas herramientas abordan la generación de informes de fútbol, se han identificado patrones comunes. A pesar de que no hay mucha información concreta en las fuentes consultadas acerca de cómo las aplicaciones mencionadas usan la IA, sí que se menciona que muchas de ellas utilizan este tipo de algoritmos para procesar grandes volúmenes de datos y generar conclusiones automatizadas sobre el desempeño de jugadores y equipos (ver ejemplos de informes). Asimismo, la mayoría de las herramientas ofrecen informes personalizados que permiten a los usuarios analizar aspectos específicos del juego, como el rendimiento táctico o la adaptabilidad de los jugadores.

En resumen, este dominio cuenta con una amplia gama de herramientas y servicios que simplifican la recopilación, el análisis y la visualización de datos, además de la creación de textos relacionados con el deporte. La diversidad de resultados que ofrecen estas herramientas impulsa el progreso del análisis deportivo en la era digital.

3.Requisitos de la aplicación

3.1. Requisitos funcionales

Los requisitos funcionales definen las funcionalidades específicas que la aplicación será capaz de realizar para satisfacer las necesidades de los usuarios. Los requisitos funcionales identificados son los siguientes:

- Acceso a los resultados de los partidos correspondientes a una jornada de una temporada específica.
- Acceso a la crónica correspondiente a un partido concreto.
- Acceso a la crónica correspondiente a una jornada de una temporada específica.
- Opción de descargar en formato TXT alguna de las crónicas generadas: Los usuarios tendrán la posibilidad de descargar en este formato en su equipo local la crónica que se haya generado si así lo desean.
- Elegir entre dos crónicas o resúmenes: Cuando el usuario pida una crónica, el sistema generará dos opciones diferentes, de forma que el usuario tendrá la opción de escoger la que más se acerque a lo que busca.
- Generación de informes automatizados: El modelo de IAG utilizado en la aplicación debe ser capaz de recibir los datos de la petición del usuario y generar el informe o crónica correspondiente. La crónica consiste en un relato textual describiendo todo lo acontecido durante la jornada seleccionada, mientras que el informe resume la temporada elegida en base a los criterios seleccionados, como, por ejemplo, la clasificación de la liga o los cruces entre todos los equipos.
- Interfaz gráfica de usuario: Se proporcionará una interfaz web de usuario para manejar las funcionalidades anteriores.

- Integración con una base de datos: Almacenamiento y gestión eficiente de los datos de los partidos de diferentes temporadas mediante una base de datos, permitiendo un acceso rápido y ordenado a la información.
- El usuario "Admin" deberá poder gestionar todos los usuarios, editando su información o pudiendo eliminarlo del sistema. (Existen dos tipos de usuarios, los normales que son los que podrán descargar las crónicas desde la aplicación web sin necesidad de iniciar sesión, y el admin que, con previo inicio de sesión, deberá tener acceso a lo anteriormente mencionado)

3.2. Requisitos no funcionales

Los requisitos no funcionales son los criterios que describen cómo debería comportarse y desempeñar la aplicación desarrollada. Los requisitos no funcionales identificados son los siguientes:

- Usabilidad: La interfaz web debe de ser fácil de navegar, permitiendo a los usuarios utilizar la aplicación sin ningún tipo de ayuda externa y obtener los datos que quieran.
- Rendimiento: Las crónicas y resúmenes generados por la aplicación y las consultas a la base de datos se ejecutarán de forma rápida y eficiente. Es decir, un tiempo inferior a un minuto para la generación de crónica y/o resumen sería aceptable y por parte de las consultas en el rango de 5 a 10 segundos.
- Mantenibilidad: El código de la aplicación debería diseñarse para ser mantenible y escalable, facilitando así la implementación de actualizaciones y la realización de correcciones de manera sencilla.
- Documentación completa: La aplicación debe estar acompañada de una documentación clara y completa que facilite tanto su instalación como su uso y mantenimiento.

4. Tecnologías utilizadas

En este capítulo, se presenta un análisis detallado de las tecnologías y herramientas empleadas en el desarrollo de la aplicación. Se discuten las consideraciones a la hora de escoger estas herramientas y tecnologías, las características de los diferentes frameworks utilizados y las necesidades del proyecto exploradas.

4.1. Tecnologías y lenguajes de programación

Python³: Lenguaje de programación principal del proyecto y con el que se fusionan todas las funcionalidades de éste. Se ha escogido Python por ser relativamente fácil de aprender (lo que permite a cualquier integrante del grupo aportar al desarrollo de la aplicación), su amplia variedad de bibliotecas y frameworks, su variedad de opciones para hacer web scraping, así como para transformar de manera fácil los datos y hacerlos más legibles, y por su compatibilidad con MongoDB, la base de datos utilizada en el proyecto para almacenar los datos de la aplicación.

MongoDB⁴: Base de datos no SQL (Structured Query Language) utilizada para almacenar los datos extraídos de las páginas webs. Se ha escogido MongoDB por su alta compatibilidad con Python, su flexibilidad en cuanto a esquemas, lo que permite almacenar distintos tipos de datos con facilidad, y el alto rendimiento que ofrece a la hora de realizar operaciones de lectura y escritura muy rápidas.

Django⁵: Framework utilizado en el backend de la aplicación web. Hemos escogido Django por estar escrito en Python, lo que facilita la implementación en la aplicación web de las

³ <https://www.python.org/>

⁴ <https://www.mongodb.com/es/company/what-is-mongodb>

⁵ <https://docs.djangoproject.com/en/5.0/>

funcionalidades de web scraping, conexión con la base de datos y con el modelo de IAG, y su escalabilidad con vistas al futuro si se aumenta la cantidad de datos a manejar y funciones.

HTML⁶/CSS⁷: Utilizado en el frontend de la aplicación web.

JSON (*JavaScript Object Notation*)⁸: Formato en el que se recoge la información de la interfaz web y se comparte con el modelo de IAG.

Selenium⁹: Librería escogida para el desarrollo de la parte de web scraping de nuestro proyecto por ser una de las más completas en este ámbito. Otras opciones probadas fueron Beautiful Soup (Beautiful Soup Documentation — Beautiful Soup 4.12.0 documentation, s.f.), y Requets (requests, 2024), que no es propiamente una herramienta de web scraping, pero se utiliza comúnmente para realizar solicitudes HTTP y descargar el contenido web antes de procesarlo con otras herramientas de scraping. Las alternativas consideradas frente a Selenium fueron descartadas al no permitirnos navegar de manera tan fácil y visual como sí ofrece Selenium.

4.2. Herramientas

Spyder¹⁰: Programa utilizado como entorno de desarrollo principal para la parte de web scraping y realización de pruebas a lo largo del desarrollo del proyecto.

⁶ <https://developer.mozilla.org/es/docs/Web/HTML>

⁷ <https://developer.mozilla.org/es/docs/Web/CSS>

⁸ <https://www.json.org/json-es.html>

⁹ <https://www.selenium.dev/>

¹⁰ <https://www.spyder-ide.org/>

Visual Studio Code¹¹: Editor de texto utilizado en la parte de desarrollo de la interfaz web. Cuenta con ayudas para la generación de código y detección de errores.

Google Colab¹²: Programa utilizado durante las pruebas en el desarrollo del código que interactúa con la base de datos y el que ejecuta el modelo de IAG. En este entorno hicimos muchas de las pruebas con los diferentes modelos de IAG hasta encontrar el que más adecuado para el proyecto.

MongoDB Atlas¹³: Servicio de base de datos en la nube desarrollado por MongoDB que permite administrar bases de datos Mongo desde cualquier lugar a través de su plataforma.

Github¹⁴: Servicio basado en la nube utilizado como método de control de versiones para mantener el código actualizado y consistente en todo momento.

Trello¹⁵: Software de administración de proyectos utilizado para organizar y repartir tareas en la construcción del proyecto.

4.3. IAG

La IAG (Talin, 2023) es un campo dentro de la IA que tiene como objetivo generar contenido (texto, imágenes, audio...) nuevo a partir de una entrada de datos y que se asemeje lo máximo posible a los datos utilizados durante su entrenamiento.

El aprendizaje profundo es la base sobre la que está construida la IAG reciente. El aprendizaje profundo utiliza numerosas capas de elementos básicos de cómputo (las neuronas)

¹¹ <https://code.visualstudio.com/>

¹² <https://colab.google/>

¹³ <https://www.mongodb.com/products/platform/atlas-database>

¹⁴ <https://docs.github.com/es/get-started/start-your-journey/about-github-and-git>

¹⁵ <https://trello.com/es/tour>

interconectadas mediante conexiones que tienen pesos asignados, lo que permite darles más o menos importancia. Estos pesos son la información que se calcula durante el entrenamiento. Normalmente, cuantos más datos de entrenamiento reciba la red neuronal, más información podrá aprender y más precisos y contextualizados serán sus resultados. De forma muy aproximada, se puede decir que la forma de trabajar de una red que genera texto es decidir qué palabra (o palabras) sigue a las anteriores (las que forman parte de la "ventana" de contexto) basándose en la probabilidad que haya aprendido para esa secuencia (o similar) durante su entrenamiento.

Existen múltiples arquitecturas o modelos en el campo de la IAG, cada una con sus propias ventajas e inconvenientes, así como ámbito de aplicación (IBM, 2024) (Amazon Web Services, Inc., s.f.) (Chernandez, 2023). Aquí se discuten tres de las más populares: las redes generativas adversariales (GAN) (de la Torre, Redes Generativas Adversarias (GAN) Fundamentos Teóricos y Aplicaciones, 2023), los autocodificadores variacionales (VAE) (de la Torre, Autocodificadores Variacionales (VAE) Fundamentos Teóricos y Aplicaciones, 2023) y los modelos basados en transformadores.

4.3.1. GAN

La arquitectura de una IAG de este tipo (ver Figura 8) consta de dos redes neuronales, una de ellas conocida como generadora y la otra como discriminadora.

Su funcionamiento a grandes rasgos consiste en que la red generadora produzca datos que el discriminador no sea capaz de distinguir de los reales, a la vez que la red discriminadora trata de detectar con la mayor exactitud posible si los datos que recibe del generador son reales o creados por él.

Con el paso del tiempo ambas redes son entrenadas y aprenderán de sus errores, consiguiendo desenvolver sus respectivos trabajos de una forma más efectiva. De esta forma, el generador llegará a producir datos tan realistas que el discriminador no sea capaz de diferenciarlos de los reales (dando una probabilidad del 50% a que esos datos sean reales o generados).

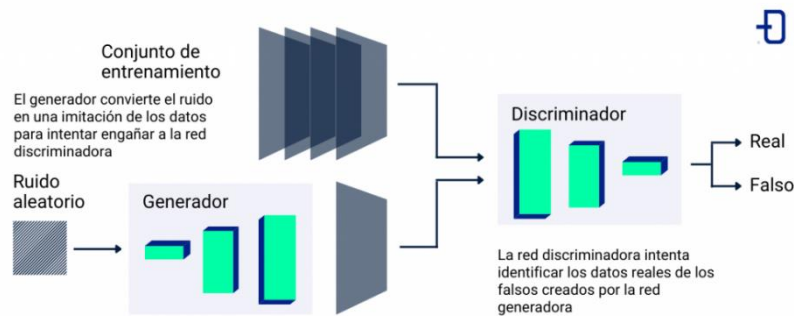


Figura 8. Arquitectura tipo GAN

4.3.2. VAE

Estos modelos basados en redes neuronales utilizan la probabilidad y la estadística para generar datos nuevos similares a los de la distribución original. Su estructura consta de un codificador y un decodificador (ver Figura 9).

El codificador es el encargado de comprimir o representar los datos originales de una manera más compacta. Lo hace resumiendo la información en un formato reducido. Esta compresión no es arbitraria, sino que se basa en distribuciones probabilísticas que toman en cuenta entre otros la media y la varianza de los datos. En otras palabras, el codificador aprende a crear una "firma" de los datos en base a patrones probabilísticos.

Una vez que el codificador ha generado esta representación reducida de los datos, es el turno del decodificador. El decodificador toma esas representaciones reducidas de los datos y trata de reconstruir los datos originales lo mejor posible. Esto significa que el decodificador actúa como un generador que intenta recrear los datos originales a partir de esa representación interna que el codificador le proporcionó. Es decir, el codificador (que actúa como un modelo de reconocimiento) debe aproximarse a/aprender la inversa del proceso del decodificador (que actúa como un modelo generativo).

El objetivo del entrenamiento del sistema completo es por tanto que la diferencia entre los datos de entrada y de salida sea mínima.

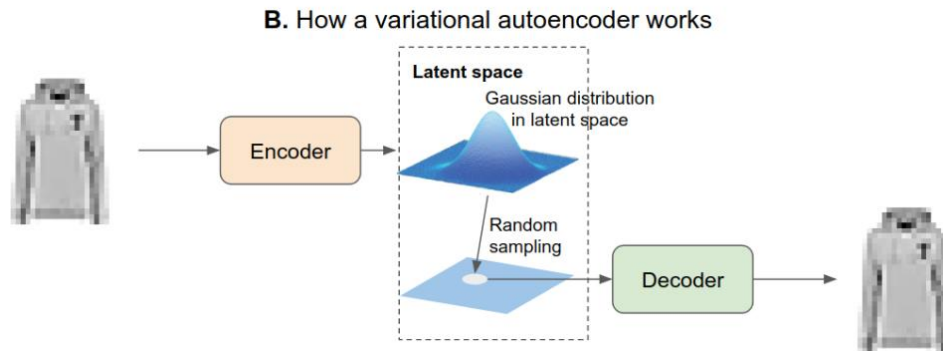


Figura 9. Arquitectura tipo VAE

4.3.3. Modelos basados en transformadores

Los modelos basados en transformadores (de la Torre, Transformadores: Fundamentos teóricos y Aplicaciones, 2023) son hoy en día muy populares al estar en la base de herramientas de éxito como GPT-3. A diferencia de los dos anteriores, su mayor potencial aparece con tareas que implican trabajar con datos secuenciales (ej. procesamiento de lenguaje natural).

Su arquitectura (ver Figura 10) está basada en mecanismos de atención que asignan un determinado nivel de importancia a partes concretas de los datos recibidos como entrada. De esta forma logran seguir varios puntos relevantes del discurso, mejorando la extracción de información y ponderando el significado de partes del enunciado (ej. referencias).

A pesar de que gracias a la inmensa cantidad y calidad de datos con los que son entrenados son capaces de generar datos muy coherentes y similares a los que podría generar una persona, no hay que olvidar que son modelos basados en la probabilidad, lo que tiene varias implicaciones importantes. Primero, y aunque en ocasiones pueda parecerlo, no comprenden realmente la información de entrada (ej. el texto). Segundo, pueden inventar contenidos erróneos e incoherentes por detectar como probable la secuencia generada, es decir, el texto de continuación probable no tiene por qué existir en los datos reales con los que se entrenó al sistema ni derivarse de ellos.

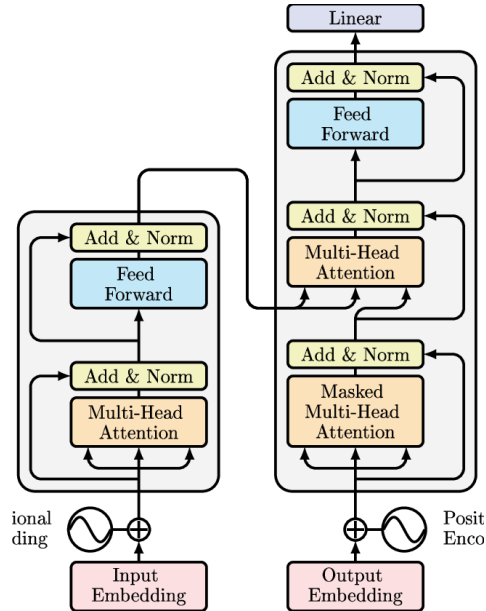


Figura 10. Arquitectura modelo basado en transformadores

4.3.4. Casos de uso

Actualmente hay numerosas y muy diversas actividades en las que se está aplicando la IAG. Hay que destacar las tareas en las que se quiere generar algo muy parecido a los datos con los que se ha entrenado al modelo y las tareas que requieren salidas probables (como la generación de texto o transcripciones).

En el ámbito del arte y del diseño, los modelos GAN son los más utilizados. Entre otras aplicaciones, pueden generar o mejorar la calidad de imágenes de todo tipo o componer y generar música. Ejemplos de estos sistemas son Dall-E¹⁶ de OpenAI o MusicLM¹⁷ de Google respectivamente para imágenes y música.

¹⁶ <https://openai.com/index/dall-e-3/>

¹⁷ <https://arxiv.org/abs/2301.11325>

La medicina también se beneficia de la ayuda que puede prestar esta tecnología. Por ejemplo, los GAN se utilizan para generar imágenes médicas para el entrenamiento de modelos sin comprometer la privacidad de los clientes, además de para estudiar la síntesis de moléculas y compuestos químicos nuevos que puedan ayudar al descubrimiento de nuevos fármacos. Los VAE, por su parte, son empleados en la comprensión de datos genómicos, ayudando a mapear relaciones en grandes conjuntos de datos. (Chernandez, 2023)

Por último, y siendo el caso que nos ocupa, hay que hablar del Procesamiento del Lenguaje Natural (PLN), para la creación de diálogos, traducciones o nuevo contenido. Los mayores exponentes en este ámbito son la familia de GPT, por ser los más conocidos por el público general y, en consecuencia, uno de los modelos más utilizados en la actualidad por la población para tareas rutinarias como escribir artículos, obtener información sobre ciertos temas, entablar conversaciones u obtener ayuda a la hora de escribir código.

4.3.5. Nuestra elección

Existen numerosos modelos que pueden ser útiles a la hora de realizar las tareas previstas en este proyecto, entre ellos GPT-2, Mixtral-8x7B-v0.1, Gemma o T5, que son con los que hemos estado haciendo pruebas para elegir el más adecuado para nuestro objetivo teniendo en cuenta los recursos de que disponemos.

Para la realización de estas pruebas de funcionamiento hemos utilizado, principalmente, modelos presentados en Hugging Face Transformers¹⁸, un marco de código abierto para el desarrollo de aplicaciones que usan aprendizaje automático. Allí se pueden encontrar de forma gratuita muchos modelos preentrenados e incluso entrenar los propios.

GPT-2¹⁹: Los modelos de la familia GPT son en general de pago, mientras que este proyecto se planteaba buscar una alternativa gratuita que permitiese realizar la tarea de generar

¹⁸ <https://huggingface.co/docs/transformers/es/index>

¹⁹ <https://huggingface.co/openai-community/gpt2>

texto. En esta búsqueda se encontró un modelo GPT-2 en Hugging Face que se evaluó para el proyecto.

Tras diferentes pruebas (tanto con el modelo preentrenado como con entrenamiento adicional), se vio que los resultados obtenidos para esta tarea no eran apropiados, generalmente por falta de coherencia en la respuesta. Un ejemplo de esto puede ser el alimentar al modelo con información sobre los jugadores del equipo del club Real Madrid y que, al pedirle un informe sobre ese equipo, al detectar la palabra Bélgica (como país del que procede su portero Courtois), se pusiese a hablar de Bélgica como país. Consideramos que el problema no radicaba en un prompt erróneo, ya que los resultados de las diferentes pruebas carecían de cualquier relación discernible entre sí. En la salida, apenas se encontraban palabras relacionadas con los datos de entrada, como nombres de países, equipos o jugadores, y cuando se encontraban no guardaban conexión alguna entre ellas (¿por qué se habían seleccionado entre las palabras de entrada?) ni con el contexto general del problema (no parecían particularmente destacables en relación con la entrada). En resumen, la única 'relación' entre los datos de entrada y salida se limitaba a la mera presencia de algunas palabras, sin ninguna conexión significativa.

T5²⁰: En el caso del modelo T5, usando entrenamiento se podían generar resúmenes de partidos coherentes a partir de datos estadísticos como goles, asistencias o tarjetas mostradas. No obstante, a pesar de usar de entrenamiento más de 100 casos (todos con una estructura muy marcada y de una longitud no mayor de 3 o 4 líneas), se observaban algunas alucinaciones e incoherencias. Por ejemplo, se inventaba los equipos que habían disputado un partido. Como el objetivo era trabajar con una cantidad y complejidad de datos muchísimo mayor para cada caso, se decidió buscar otras alternativas.

²⁰ <https://huggingface.co/google-t5/t5-large>

Mixtral-8x7B-v0.1²¹: Mixtral, también de Hugging Face permitió, sin necesidad de entrenamiento adicional, generar texto a partir de las entradas que se le proporcionaban de forma sencilla y coherente desde el principio.

En un principio, la ejecución de pruebas con los distintos modelos la hicimos en local, lo que derivó en tiempos de espera muy altos por las altas demandas de hardware que tienen este tipo de modelos tan pesados. Una vez estudiada esta opción, y pese a obtener buenos resultados, decidimos que no podría ser nuestra opción definitiva para la generación de crónicas. Al ser un modelo tan pesado y requerir de tantos recursos, el lanzarlo en local era imposible y todas las pruebas y ejecuciones las teníamos que hacer en Google Colab, un entorno que está lejos de ser el ideal para producción y que complicaba el ejecutar el proyecto en conjunto en un solo entorno.

Gemma²²: Investigamos un poco sus fortalezas y limitaciones y concluimos que podría servir para nuestro propósito. De nuevo era de pago, pero encontramos en Hugging Face un modelo gratuito. Investigando descubrimos la existencia de una API gratuita disponible en Hugging Face. Con ella podíamos trabajar con el modelo lanzado desde nuestra aplicación web y obteníamos resultados satisfactorios para nuestro objetivo. Por haber obtenido resultados coherentes y similares a una crónica deportiva al uso, y poder lanzar la aplicación conjunta gracias al uso de su API desde nuestra aplicación, esta fue nuestra elección de modelo para la parte de generación de crónicas.

Su única debilidad es que, al ser de uso gratuito, la api permite un número de peticiones limitado cada cierto tiempo. Por eso optamos por en lugar de utilizar la API usar el modelo para la parte de recopilación, tratamiento y almacenamiento de datos que se lanza de forma independiente y previa al despliegue de la aplicación web (la que genera las crónicas e interacciona con el cliente). No cuenta con un número limitado de consultas gratuitas, contamos

²¹ <https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1>

²² <https://huggingface.co/google/gemma-2-9b-it>

con un servidor con las características necesarias para ejecutarlo y da resultados afines a lo que queremos para esta fase.

4.3.6. Conclusiones

Los modelos de IAG están revolucionando diversas industrias y experiencias de clientes al acelerar la investigación, mejorar la personalización del servicio al usuario, optimizar procesos empresariales y aumentar la productividad de los empleados. Al permitir la exploración de datos complejos y ofrecer respuestas naturales en interacciones humanas, la IAG promete un aumento significativo en la eficiencia y la innovación a nivel mundial.

Con un entrenamiento previo de suficiente calidad (variedad y cantidad de datos, datos de comprobación suficientes...), la IAG puede hacer que un periodista deportivo centre su atención en aspectos de las crónicas como conseguir una narrativa más rica, además de ajustar la respuesta del modelo, desempeñando así de una forma más eficiente su trabajo.

En el caso de este TFG, Gemma es el modelo de IAG seleccionado para generar crónicas de partidos de fútbol ya que es capaz de procesar una gran cantidad de información, generar contenido coherente, es mucho más rápida que las otras opciones probadas y logra todo esto sin la necesidad de recibir un entrenamiento adicional especializado en lo que le vamos a pedir. A pesar de todos los puntos mencionados de este modelo que explican su elección para ser el encargado de generar nuestras crónicas, su api gratuita de Hugging Face (que es la que usaremos por los recursos de los que disponemos) tiene un número de usos limitado cada cierto tiempo. Todo ello hace que no sea viable su uso en la fase de extracción de eventos destacados dentro de cada uno de los partidos de los que recabamos información en la fase de recopilación, tratamiento y almacenamiento de datos.

Para esa parte del proyecto hemos optado por utilizar el modelo de Gemma en sí en vez de utilizar la api ya comentada, siempre obtuvimos buenos resultados para esta tarea con este método. Su único problema es el propio tamaño del modelo, que imposibilitaba el lanzamiento conjunto con la aplicación web de forma simultánea debido a los recursos de los que disponíamos pero, para esta parte del proyecto, que por ahora se lanza de forma independiente a la aplicación

web (ver último punto del capítulo [Trabajo a futuro](#) para entender cómo queremos que esto cambie) y por lo anteriormente mencionado, fue la opción escogida.

4.4. Web scraping

Para la realización de nuestro proyecto necesitamos una gran cantidad y variedad de datos. Esta información es fácil de encontrar en la web en diarios deportivos, blogs, páginas oficiales de las diferentes competiciones... No obstante, existe un problema que es el cómo recolectar toda esa información y almacenarla de forma coherente para su posterior interpretación por herramientas informáticas (como la IAG).

El proceso anterior se conoce como web scraping (Equipo editorial de IONOS, 2023) y es una forma eficiente de acceder a todos estos datos mediante el escaneo de diferentes sitios web para posteriormente tratarlos y almacenarlos si fuera necesario.

En nuestro caso hemos accedido y trabajado con la información disponible en:

- Resultados fútbol: <https://www.resultados-futbol.com/>
- Página oficial periódico deportivo as: <https://as.com/>

Existen numerosas librerías y frameworks para hacer web scraping. Ya que el proyecto está basado en Python3, buscamos herramientas de web scraping compatibles. Algunas de las más conocidas son Scrapy, Selenium o BeautifulSoup.

Para la realización de estas pruebas hemos utilizado principalmente herramientas basadas o escritas en Python, lenguaje principal del proyecto, para facilitar el desarrollo del proyecto y la unión del scraping con el resto de las funcionalidades.

Scrapy: Scrapy (Scrapy | A Fast and Powerful Scraping and Web Crawling Framework, s.f.) es especialmente útil para proyectos que tengan que procesar grandes volúmenes de páginas web. Esto es debido a que realiza la apertura de webs de forma asíncrona. Está basada en spiders (clases propias de Python), cada una de ellas encargada de procesar una web concreta.

BeautifulSoup: BeautifulSoup (Beautiful Soup Documentation — Beautiful Soup 4.12.0 documentation, s.f.) es la más antigua de las tres herramientas y, a diferencia de lo que sucedía con Scrapy, no es necesaria la programación orientada a objetos para utilizarla. Funciona con scripts y es mucho más fácil de aprender en un principio gracias a su método basado en “pescar información de la sopa de tags”, como su propio nombre indica.

Selenium: Selenium (Selenium, s.f.) es un framework que en un principio fue creado para testear webs, pero con el WebDriver de Selenium (Gazala, 2023), herramienta de automatización de pruebas que permite simular acciones humanas en una aplicación web como hacer clic en botones o rellenar formularios, puede utilizarse para hacer web scraping.

A diferencia de los dos casos anteriores, Selenium no trabaja con el texto fuente en HTML de la web, sino que carga la página en un navegador sin interfaz de usuario. De esta forma, el navegador interpreta el código fuente y crea un DOM (*Document Object Model*). Esta interfaz permite simular así interacciones del usuario con la web. Como el DOM se genera de forma dinámica, Selenium puede analizar webs con comportamientos dinámicos. Esta es la ventaja clave de esta herramienta, ya que permite la interacción más sofisticada con los sitios web.

A pesar de la corta curva de aprendizaje que tiene BeautifulSoup, como ya contamos con conocimientos de programación, aprender a utilizar Python no era un problema para nosotros. Además, el desarrollo del proyecto no necesitaba ser rápido, ya que contábamos con un curso académico completo para acabarlo. Por todo esto decidimos buscar herramientas con una funcionalidad más rica y descartar BeautifulSoup.

Por su parte, Scrapy nos permite realizar proyectos de web scraping mucho más grandes y complejos de lo que necesitamos. Además, este proyecto necesita extraer información de webs dinámicas, lo que hizo que nos decantásemos por Selenium para esta parte del proyecto.

Selenium es una de las herramientas más potentes de web scraping en la actualidad, además de contar con una extensa documentación y comunidad a sus espaldas. Esto hace que cualquier desarrollo con esta herramienta sea más sencillo, al menos desde el punto de vista del soporte disponible.

4.5. Aplicación web

Para mostrar las crónicas generadas a partir de la IA y que la interacción entre ésta y el usuario sea lo más intuitiva posible, construiremos una aplicación web. Desde la elección de tecnologías hasta la implementación de la arquitectura, cada paso se ha tomado considerando cuidadosamente las necesidades de los futuros usuarios y la eficiencia del proceso. Un aspecto clave ha sido la elección de las tecnologías para implementar las dos grandes partes de la aplicación web: el frontend, que incluye los elementos que permiten a los usuarios interactuar con la aplicación, tales como elementos visuales (ej. botones, casillas de verificación, gráficos y mensajes de texto) y elementos de soporte; el back-end que almacena y procesa los datos de las aplicaciones, es decir, la infraestructura que permiten que la aplicación procese datos y funcione.

4.5.1. Elección del framework backend

La elección de Django como framework para el backend se basa en su eficiencia y robustez. Django es conocido por su capacidad para facilitar el desarrollo rápido de aplicaciones web, proporcionando una estructura sólida y una gran cantidad de características integradas, como la administración de bases de datos y el manejo de URL. Su adopción del patrón Modelo-Vista-Template también facilita la organización y modularidad del código. Al incorporar todas estas características para realizar la aplicación web y estar escrito en Python facilita la integración del resto de los componentes de la aplicación.

Si bien Django es una opción popular y sólida para el desarrollo de aplicaciones web en Python, consideramos otras alternativas. Flask, por ejemplo, es un microframework que ofrece una gran flexibilidad y simplicidad, siendo mejor para proyectos que requieran menos recursos. Por otro lado, FastAPI es un framework moderno que se centra en la velocidad y el rendimiento, especialmente para aplicaciones que requieren una alta concurrencia y escalabilidad, pero optamos por una framework más fácil de usar y más sólido.

4.5.2. Elección del framework frontend

HTML, CSS y JavaScript son estándares abiertos ampliamente aceptados en la web, garantizando la compatibilidad entre navegadores y dispositivos. Con ellos se puede personalizar y adaptar la interfaz de usuario, creando experiencias web únicas y atractivas. Además, cuentan con una comunidad de desarrolladores activa y una gran cantidad de recursos disponibles, desde documentación hasta frameworks, facilitando el aprendizaje y el desarrollo de aplicaciones web. En conjunto, estas tecnologías proporcionan una base sólida y versátil para crear aplicaciones web modernas y eficientes.

HTML, CSS y JavaScript son la opción más popular, sencilla e intuitiva. No obstante, existe frameworks y API que pueden facilitar más el desarrollo del frontend, algunos de los cuales también se han estudiado. En concreto, probamos a usar Vue.js, un framework progresivo de JavaScript que se utiliza para construir interfaces de usuario interactivas y reactivas, pero para su uso, necesitábamos lanzar una instancia más a parte del Django, cosa que nos daba problemas.

5.Arquitectura del proyecto

5.1. Arquitectura general (alto nivel)

El sistema de generación de crónicas deportivas cuenta con un frontend que tiene una interfaz web construida con HTML y CSS. Su propósito es ofrecer una interfaz gráfica amigable para que los usuarios puedan seleccionar partidos, jornadas o temporadas, y obtener crónicas o resúmenes deportivos, mostrando así los resultados generados por el sistema.

Las solicitudes provenientes del frontend se procesan en el backend, que incluye una parte de servidor implementado con Django. Éste actúa como intermediario entre el usuario y la lógica interna de la aplicación, encargándose de recibir las solicitudes de generación de crónicas, interactuar con otras partes del backend, y enviar las respuestas generadas. Es por tanto el puente entre la interfaz de usuario, el modelo de IAG y la base de datos.

Además de especificar y generar las crónicas, el sistema necesita recopilar la información que sirve de base para generar esos informes. Esto incluye un proceso de web scraping para recolectar los datos que luego se procesan y filtran para asegurar su formato óptimo, permitiendo que el sistema los utilice eficazmente.

Toda la información recopilada y procesada se almacena en una base de datos MongoDB en el proveedor de servicio MongoDB Atlas. Este almacenamiento asegura que los datos estén disponibles y sean consistentes cuando otros subsistemas de la aplicación los requieran.

Finalmente, el modelo de IAG utilizado es Gemma, que recibe los datos de entrada y genera crónicas a partir de ellos. Este componente es esencial para el proyecto, ya que se encarga de generar contenido dinámico basado en las solicitudes realizadas por el usuario a través de la interfaz web.

5.2. Flujo de información

El flujo de información (ver Figura 11) comienza con la extracción de datos deportivos desde diversas páginas web mediante técnicas de web scraping. Estos datos pueden incluir

detalles como resultados de partidos, estadísticas de los jugadores, comentarios sobre los encuentros y cualquier otro tipo de información relevante. Una vez obtenidos, los datos se almacenan en la base de datos NoSQL MongoDB que permite estructurar esta información de manera eficiente en colecciones que agrupan los datos por categorías como temporadas, jornadas o partidos individuales.

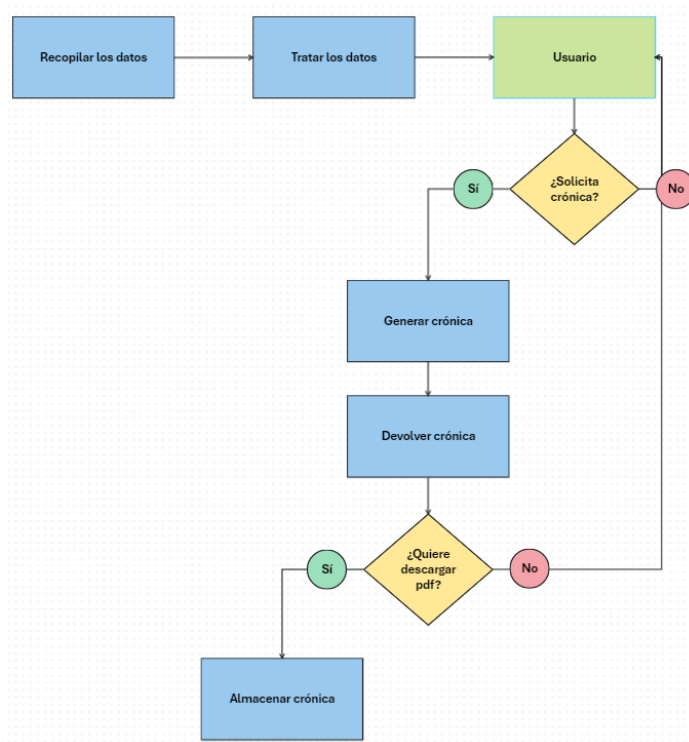


Figura 11. Flujo de información del proyecto

Cuando un usuario accede a la página web y solicita una crónica deportiva, ya sea sobre una temporada completa, una jornada en particular o un partido específico, el sistema procesa esta solicitud en el backend. El backend recupera los datos pertinentes desde MongoDB y los organiza en un formato que pueda ser comprendido por un modelo de IAG. Este modelo generativo, utilizando los datos deportivos como base, crea un texto narrativo que refleja la

crónica solicitada por el usuario, asegurando que el contenido esté adaptado a los detalles específicos del evento seleccionado.

Una vez que la IAG ha generado la crónica, el backend la envía a la interfaz web (frontend), donde se muestra de manera clara y accesible para que el usuario pueda leerla.

Además de visualizarla en la página, el usuario tiene la opción de descargar la crónica en un archivo de texto (.txt), lo que le permite guardar o compartir el contenido fácilmente, hemos decidido que el formato de descarga sea este para permitirle al usuario modificar a su gusto la crónica si así lo desea. Así, el flujo de información abarca desde la extracción inicial de datos en la web hasta la presentación final de una crónica personalizada, ya sea en pantalla o como archivo descargable, proporcionando una solución completa y automatizada al usuario final.

5.3. Diseño detallado (bajo nivel)

5.3.1. Diseño base de datos

La función de la base de datos es almacenar los datos de partidos, temporadas, equipos etc. de forma estructurada. Como ya comentamos anteriormente, hemos optado por trabajar con Mongo Atlas, un servicio de Mongo DB en la nube. Al ser una base de datos no relacional, esto permite escalar en un futuro el proyecto y facilita ampliar los datos utilizados, como por ejemplo más estadísticas para generar otros tipos de informes.

En este servicio de almacenamiento en la nube, contamos con una base de datos llamada TFG, que contiene las diferentes colecciones que necesitamos. Muchas de ellas son generadas automáticamente por el propio Django, así que nos centraremos en las que almacenan la información recopilada con el módulo de web scraping. Las crónicas generadas por la aplicación las podrá descargar el usuario directamente desde la web después de haberlas generado, por lo que no se almacenarán en la base de datos.

cruces, *datos_partidos* y *estadisticas Equipos* son colecciones que almacenan los documentos relacionados con los datos necesarios para la generación de las crónicas.

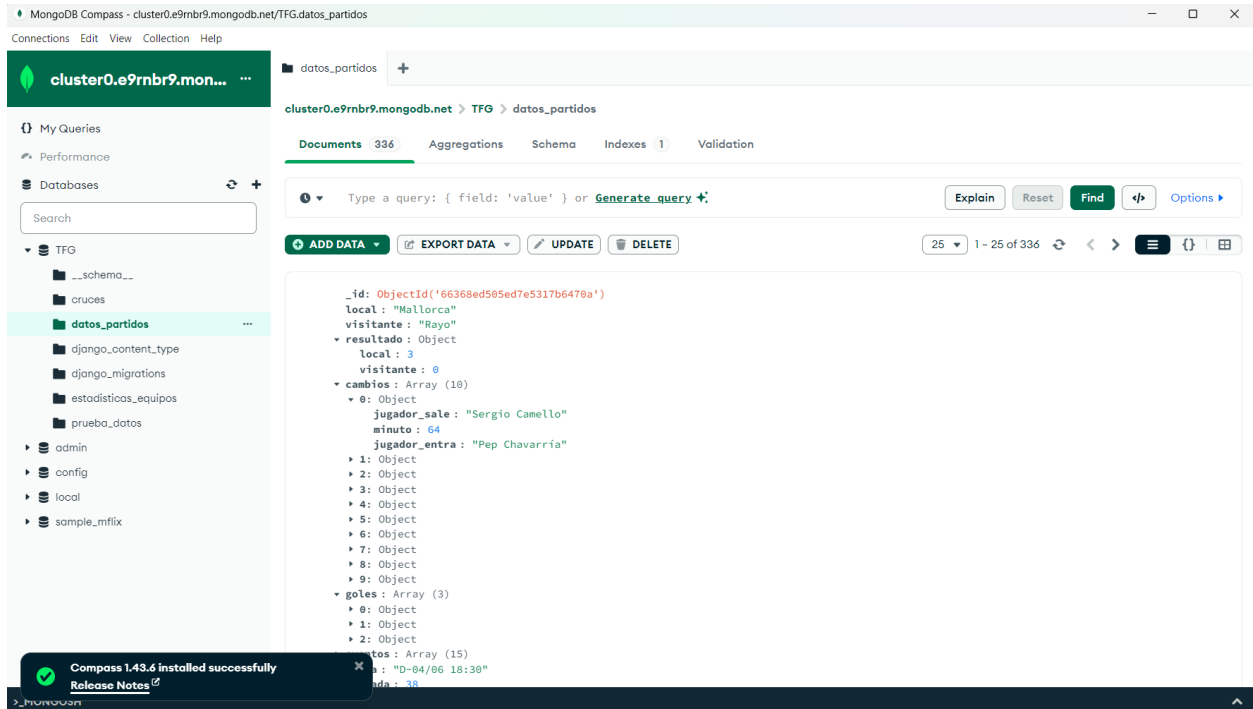


Figura 12. Datos almacenados de un partido

La colección de datos_partidos son registros de partidos de fútbol entre en diferentes temporadas. Contiene el identificador único del partido, los equipos local y visitante, el resultado del partido, los cambios de jugadores realizados, los goles con sus anotadores y minutos, eventos destacados del partido, la fecha y hora del encuentro, la jornada y la temporada en la que se jugó.

La colección estadísticas_equipos describe la estructura de un registro estadístico de un equipo de fútbol para una temporada específica. Incluye información clave como el nombre del equipo, su ranking, puntos totales y el número de partidos jugados, ganados, empatados y perdidos. También se divide en estadísticas de rendimiento en casa (local) y fuera de casa (visitante), mostrando cuántos puntos obtuvieron en cada condición, cuántos partidos jugaron, ganaron, empataron o perdieron. Además, se detallan los goles a favor y en contra. Esta

estructura es útil para hacer un seguimiento del desempeño global de un equipo a lo largo de una temporada.

La colección cruces representa un registro básico de un partido de fútbol. Contiene información sobre el equipo local y visitante, los goles anotados por cada equipo, la fecha y la temporada en la que se jugó el partido. Este tipo de estructura se utiliza para almacenar y consultar los resultados de encuentros de manera más generalizada en una base de datos, permitiendo hacer un seguimiento más general de los partidos jugados durante una temporada

Scripts utilizados:

- **ConexiónMongo.py:** establece la conexión con nuestra base de datos en Mongo Atlas, más concretamente con la base de datos llamada TFG.

5.3.2. Diseño de recopilación, tratamiento y almacenamiento de datos

Estos procesos se han automatizado para que, únicamente con la ejecución del archivo *main.py*, que podrá hacer el admin desde su panel de control dentro de la propia aplicación web en un futuro (ver último apartado del capítulo Trabajo a futuro), se hagan las llamadas necesarias para la recopilación, tratamiento y almacenamiento de información necesaria.

Actualmente la ejecución de *main.py* se ha de hacer en un servidor con los recursos hardware necesarios y, por buenas prácticas, en un entorno virtual. Es una ejecución independiente y que se hace previa al despliegue de la propia aplicación web en local (para que esta ya cuente con toda la información que necesita para funcionar almacenada en la base de datos de forma correcta).

A continuación, mostramos un diagrama (ver Figura 13) para entender mejor todo lo que sucede al ejecutar *main.py*. Los colores son para facilitar su comprensión (el color verde corresponde a tareas de web scraping y el azul a tratamiento de esa información y su almacenamiento en la base de datos).

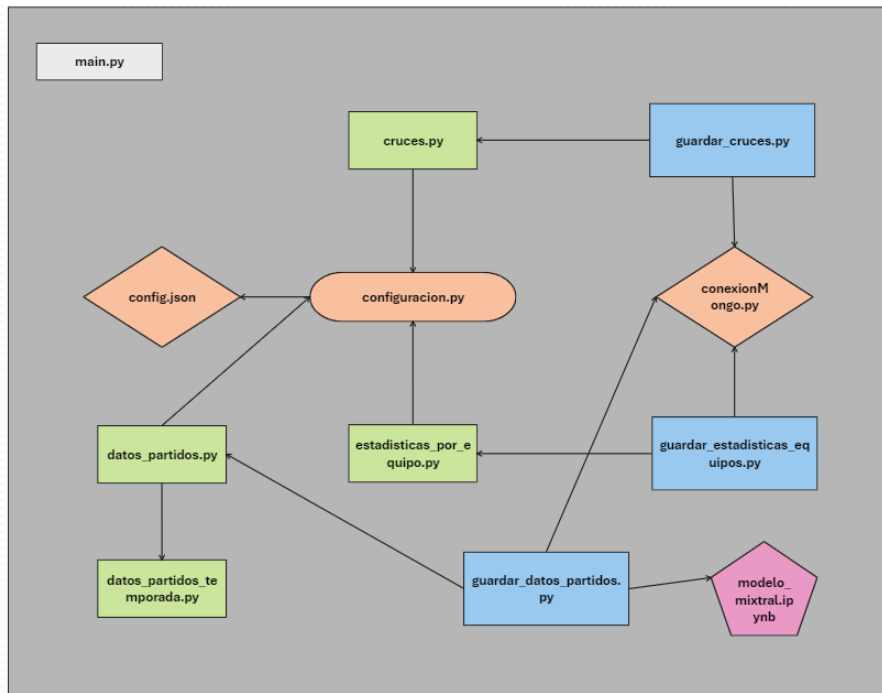


Figura 13. Diagrama de flujo de script de recopilado/tratado/almacenado de datos

Scripts utilizados:

- **Main.py:** es el archivo encargado de lanzar los scripts con los que se han automatizado todos estos procesos.
- **Config.json:** Contiene un JSON que es el que indica las temporadas con las que vamos a querer trabajar y los scripts que se deben ejecutar para recopilar y guardar la información con la que trataremos.
- **Configuracion.py:** Es el encargado de leer los datos del config.json y contiene funciones que devuelven las temporadas a analizar y la lista de scripts a ejecutar por main.py.
- **Cruces.py, estadisticas_por_equipo.py, datos_partidos.py, datos_partidos_temporada.py:** Son los encargados de hacer el web scraping de los

datos a los que hace referencia su nombre usando Selenium. Además, guardan toda esta información ordenada en archivos .csv

- **Guardar_cruces.py**, **guardar_estadisticasEquipos.py**, **guardar_datos_partidos.py**: Leen los csv generados por los scripts anteriormente mencionados y, tras tratarla, la almacena en la base de datos. En el caso de *guardar_datos_partidos.py*, hay una interacción con el modelo Gemma para extraer los eventos no estadísticos de un partido dado a partir de su crónica. Esos son los Datos almacenados en la base de datos y no las crónicas al completo. Estos archivos, como es lógico, interaccionan con *ConexiónMongo.py*, que es el que establece la conexión a la base de datos donde se almacenará toda la información.

5.3.3. Construcción de la interfaz web

5.3.3.1. Organización general

La aplicación web pretende facilitar a los usuarios, en este caso a los periodistas deportivos, la obtención de crónicas de forma rápida y precisa. Para ello, la interfaz de usuario debe ser intuitiva y fácil de usar, permitiendo a los usuarios seleccionar una temporada, jornada o partido específico y obtener crónicas generadas automáticamente por un sistema de IA. La aplicación también necesita integrarse con una base de datos eficiente, como MongoDB, para gestionar grandes volúmenes de datos sobre los partidos de distintas temporadas y asegurar un acceso rápido y eficaz a la información.

5.3.3.2. Estructura de archivos

El proyecto está organizado siguiendo una estructura típica de un framework de desarrollo web. En la raíz del proyecto se encuentra la carpeta principal llamada “LigaFutbol”, que contiene los subdirectorios y archivos fundamentales para el funcionamiento del sistema. Dentro de esta

carpeta principal, encontramos una subcarpeta llamada "Jornada", que agrupa la lógica específica relacionada con la jornada de la liga (ver Figura 14).

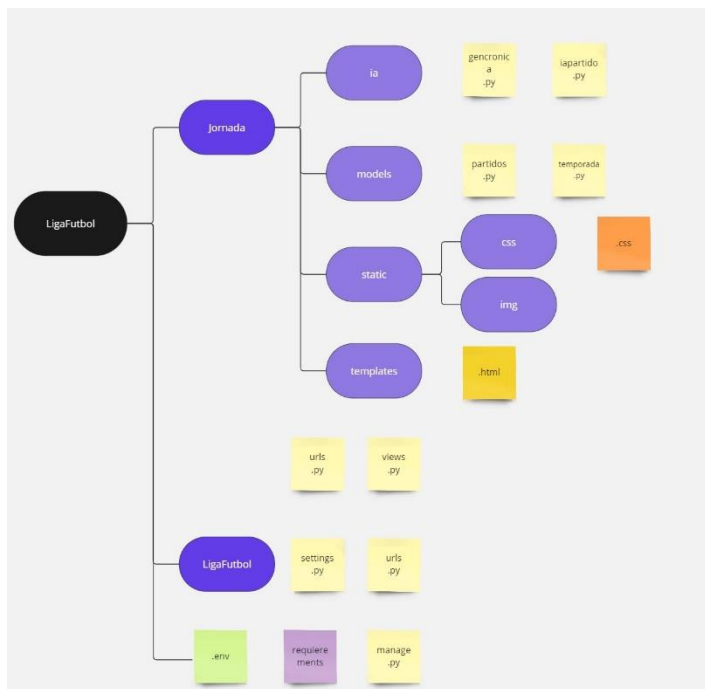


Figura 14. Estructura de archivos del proyecto

Dentro de "Jornada", hay varias subcarpetas que organizan diferentes aspectos del proyecto. La carpeta "ia" contiene la inclusión de los algoritmos de ia, relacionados con la generación automática de las crónicas. La carpeta "models" alberga las definiciones de las estructuras de datos utilizadas en el proyecto, como cruces o partidos, siguiendo el patrón de Django para los modelos de datos. La carpeta "static" contiene archivos estáticos como imágenes y hojas de estilo, organizados a su vez en subdirectorios llamados "css" y "img", lo que facilita el manejo de los recursos visuales del proyecto. En paralelo, la carpeta "templates" almacena las plantillas HTML del proyecto, cruciales para la construcción de la interfaz de usuario de la aplicación.

Además de la carpeta "Jornada", encontramos una segunda subcarpeta llamada "LigaFutbol" que contiene otros archivos esenciales para la funcionalidad del sistema. Entre ellos,

se destacan `urls.py`, que define las rutas de acceso dentro del sitio web, y `views.py`, que contiene la lógica que gestiona las peticiones del usuario y retorna las respuestas correspondientes, normalmente en forma de plantillas HTML. También está presente el archivo `settings.py`, encargado de gestionar la configuración global del proyecto, como la conexión a la base de datos o las aplicaciones instaladas.

A nivel de la raíz del proyecto, también se encuentran los archivos `“.env”` y `“requirements”`. El archivo `“.env”` es utilizado para almacenar variables de entorno importantes, como claves y configuraciones sensibles, mientras que `“requirements”` lista las dependencias del proyecto, facilitando la instalación de los paquetes necesarios.

5.3.3.3. Patrones de la aplicación

La aplicación sigue el patrón de diseño Modelo-Vista-Template (ver Figura 15). Uno de los principales factores para utilizar Django, es su modelo predefinido, ya que nos facilita la comunicación con el Mongo DB y la organización de la estructura del backend de la aplicación.

Este patrón divide la aplicación en tres componentes principales: el Modelo, que representa la estructura de datos y la lógica de la aplicación; la Vista, que se encarga de mostrar la información al usuario final de manera comprensible; y el Template, que define la presentación visual de la aplicación web.

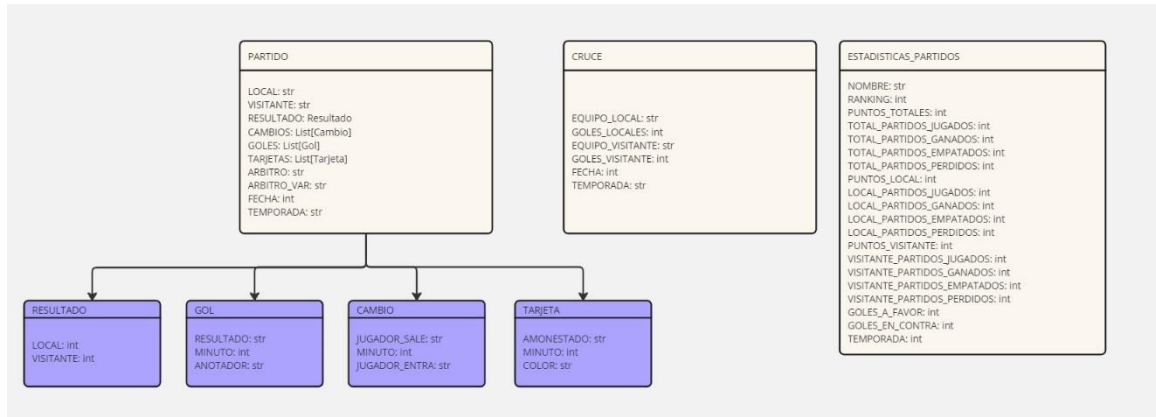


Figura 15. Modelo-Vista-Template

La aplicación web cuenta con tres elementos modelo: Cruce y Estadísticas Partidos, que se usan para generar después el input de la crónica de temporadas y Partidos, que además de generar el input de la crónica de partidos y jornadas se utiliza para el flujo de la web. Todos se relacionan entre sí por el número de la temporada que se está tratando.

5.3.3.4. Gestión de usuarios

Dentro de la aplicación existen dos tipos de usuarios con diferentes niveles de acceso: los usuarios estándar y los administradores.

Los usuarios estándar tienen permisos limitados, lo que les permite interactuar con las funcionalidades básicas de la aplicación, como navegar a través de las diferentes temporadas y sus respectivas jornadas, además de obtener las crónicas de dichas jornadas. Los usuarios estándar podrán ver también los partidos jugados en cada jornada, fecha, equipo local, equipo visitante y resultado de cada partido, así como obtener las crónicas de cada partido individual.

Los administradores, además de tener acceso a todas las funcionalidades de los usuarios estándar, podrán gestionar los usuarios dentro de la aplicación. La gestión de usuarios incluye la edición del nombre de un usuario y de su contraseña, cambiarle el rol (administrador o usuario) e inhabilitar o borrar un usuario de la aplicación. Por otro lado, el administrador tendrá la

posibilidad de ejecutar los scripts encargados de obtener los datos de diferentes temporadas, sus jornadas y sus respectivos partidos.

5.4. Integración con modelo IA generativa para generación de crónicas

La IAG es la clave del proyecto al tratarse del núcleo de su funcionamiento. Como ya se comentó en apartados anteriores, este proyecto usa el modelo Gemma en esta parte.

La forma de trabajar con el modelo se centra en el archivo *gencronica.py*. En él se llama a una API de este modelo ubicado en Hugging Face²³, pasándole una entrada de texto como argumento. Esta entrada varía dependiendo de la función que queremos que desempeñe la IAG, pero siempre será una combinación entre un enunciado que le detalle la tarea que debe realizar y la información en la que deberá basar su respuesta.

Scripts utilizados:

La interfaz Django se comunica con el apartado de la IAG mediante el apartado de las vistas, dentro de las funciones utilizadas para mostrar la crónica, donde se utilizan las funciones *cronicaTemporada*, *cronicaJornada* y *cronicaPartido*, ubicadas en *gencronica.py*.

Dentro de las funciones anteriores, se utilizan los modelos de Django mencionados anteriormente para generar la entrada que se usará para la IAG. Una vez generada dicha entrada, se llama a la función generar ubicada en *iapartidos.py*.

En la función generar hacemos la llamada a la API y ajustamos varios parámetros como la temperatura, el número máximo de tokens o la penalización por repeticiones para obtener los resultados más satisfactorios posibles.

²³ <https://huggingface.co/>

Una vez generada devolvemos el output a las funciones que muestran las crónicas e imprimimos los resultados en la interfaz.

Todos los archivos, utilizados para el manejo de la IAG anteriormente mencionados se encuentran dentro del proyecto Django en una carpeta llamada IA, para poder acceder fácilmente a sus funcionalidades.

6.Manual de uso

6.1. Manual de uso: Cliente

1. Lanzar aplicación mediante el comando **python manage.py runserver**
2. Acceder a la URL del sitio web (ver Figura 16). Como la lanzamos en local, la URL será <http://127.0.0.1:8000/>

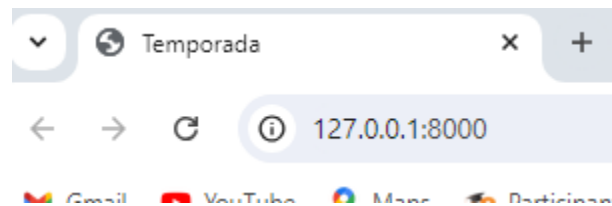


Figura 16. URL acceso a la aplicación web

3. Iniciar sesión en la aplicación con usuario y contraseña. En caso de no disponer de un usuario válido, realizar el registro en la aplicación.

A login form titled 'Iniciar Sesión'. It has two input fields: 'Nombre de usuario' and 'Contraseña'. Below the fields is a red button labeled 'Iniciar Sesión'. At the bottom, there is a link that says '¿No tienes una cuenta? Regístrate aquí'.

Figura 17. Panel de inicio de sesión

Crea tu cuenta

Nombre de usuario:

Nombre:

Apellido:

Correo Electrónico:

Contraseña:

Confirmar Contraseña:

Registrarse

¿Ya tienes una cuenta? [Inicia sesión aquí.](#)

Figura 18. Panel de registro

4. Seleccionar una de las temporadas disponibles, haciendo clic en una de ellas, para ver el listado de jornadas de dicha temporada (ver Figura 19).

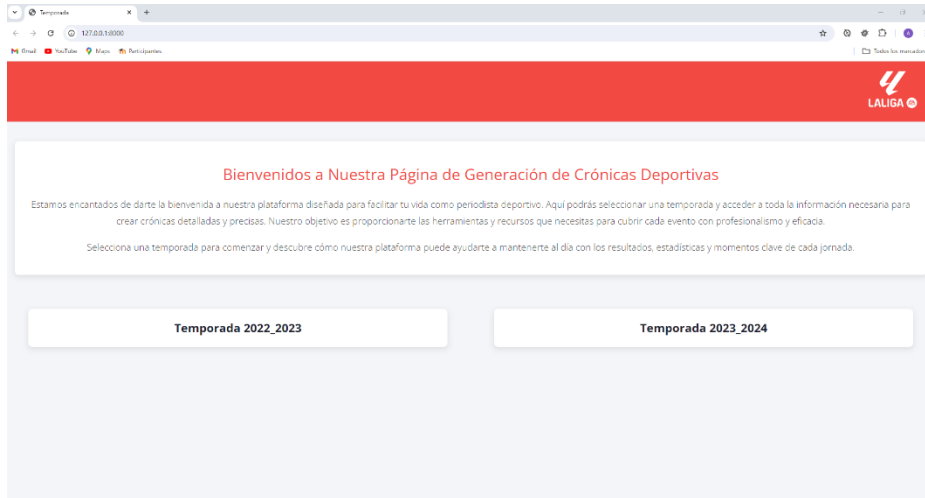


Figura 19. Página temporadas

5. Una vez hemos accedido a una de las temporadas tendremos dos opciones. Presionar el botón superior que dice “Ver Crónica de la Temporada 2022_2023” si queremos obtener el resumen de esa temporada o seleccionar una de las jornadas de la temporada elegida anteriormente haciendo clic en ella (ver Figura 20).

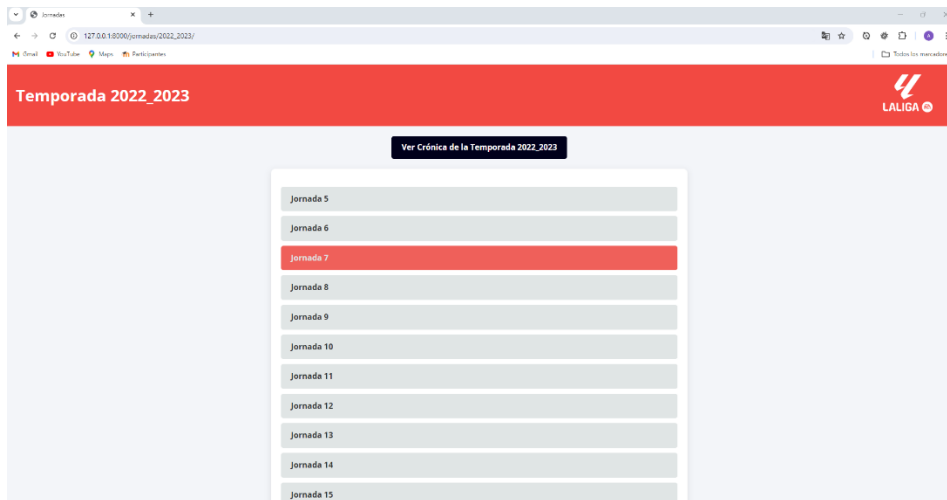
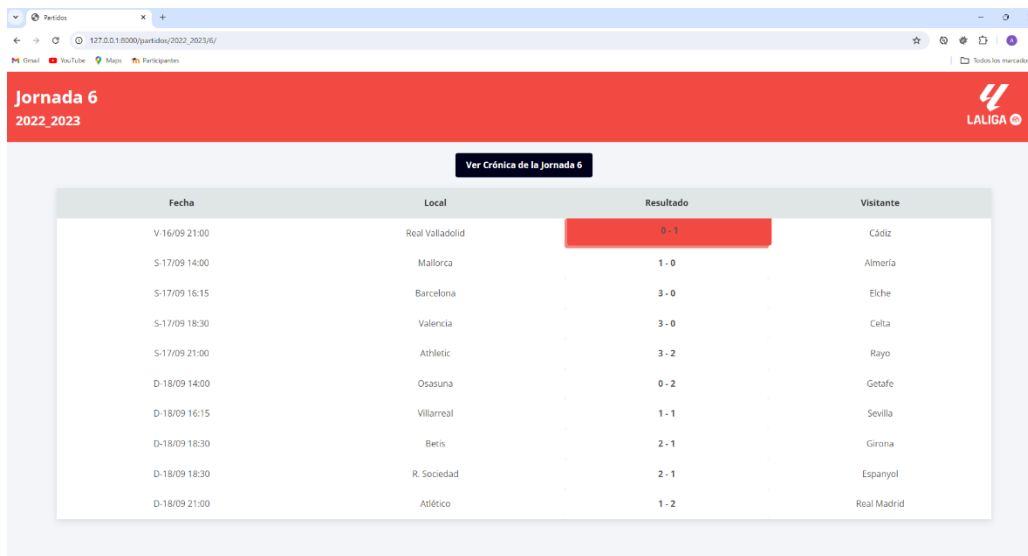


Figura 20. Página jornadas

6. En la página de la jornada se muestra un botón “Ver crónica de la jornada 6” con el que podemos obtener la crónica de esa jornada al completo. Debajo podemos ver una

tabla con información como fecha, equipo local, equipo visitante y resultado, de todos los partidos de esa jornada (ver Figura 21). Si hacemos clic en el resultado de alguno de los partidos (al pasar el ratón por encima de ellos se pondrá de color rojo para que sepamos qué partido vamos a escoger) obtendremos la crónica deportiva de ese partido en concreto.



| Fecha | Local | Resultado | Visitante |
|---------------|-----------------|-----------|-------------|
| V-16/09 21:00 | Real Valladolid | 0 - 1 | Cádiz |
| S-17/09 14:00 | Mallorca | 1 - 0 | Almería |
| S-17/09 16:15 | Barcelona | 3 - 0 | Elche |
| S-17/09 18:30 | Valencia | 3 - 0 | Celta |
| S-17/09 21:00 | Athletic | 3 - 2 | Rayo |
| D-18/09 14:00 | Osasuna | 0 - 2 | Getafe |
| D-18/09 16:15 | Villarreal | 1 - 1 | Sevilla |
| D-18/09 18:30 | Betis | 2 - 1 | Girona |
| D-18/09 18:30 | R. Sociedad | 2 - 1 | Espanyol |
| D-18/09 21:00 | Atlético | 1 - 2 | Real Madrid |

Figura 21. Página partidos

7. El sistema siempre genera dos versiones alternativas de una crónica con diferentes matices en su escritura de forma que el usuario puede elegir la que prefiera. Para descargar una de las dos opciones, deberemos seleccionar la elegida haciendo clic y presionar el botón azul de debajo que pone “DESCARGAR TXT” (ver Figura 22).

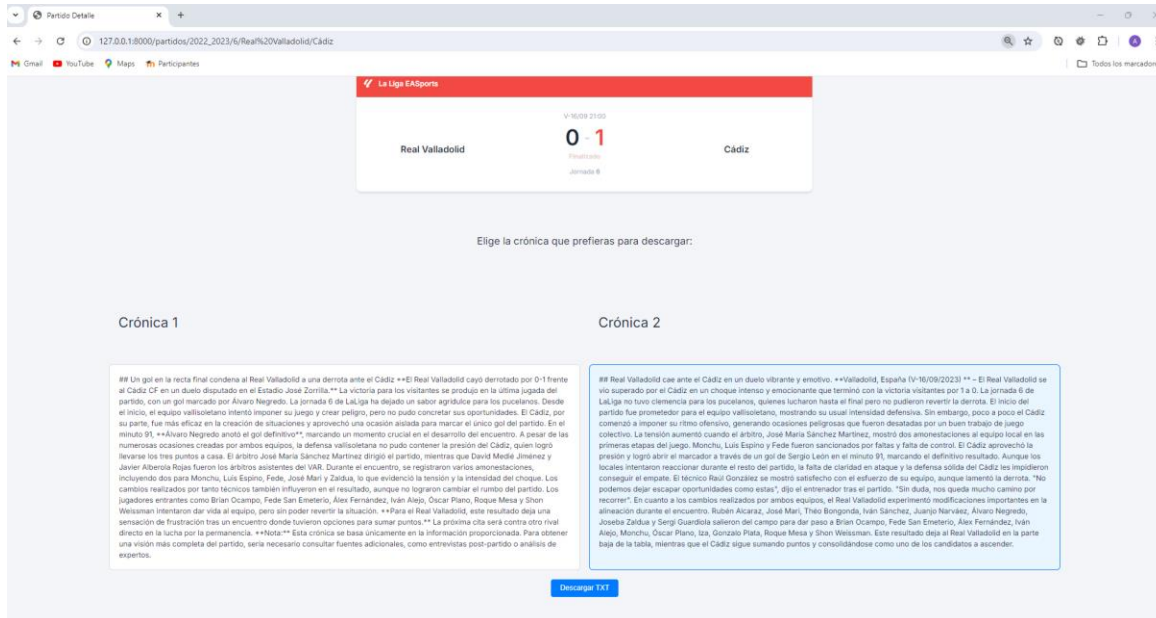


Figura 22. Crónicas de un partido generadas

6.2. Manual de uso: Administrador

El administrador cuenta con la misma funcionalidad que un usuario normal / cliente, además de poder gestionar todos los usuarios registrados en la aplicación, pudiendo cambiar sus permisos, y tener la posibilidad de ejecutar los scripts que realizan el web scraping y actualizan la base de datos.

Funcionalidad específica del administrador:

Como se ha explicado en apartados anteriores, por falta de recursos no se ha podido programar el lanzamiento del script de la fase de recopilación/tratamiento/almacenamiento para hacerlo en local con el resto de la aplicación web.

Por ello, el admin tendrá que ejecutar dicho script en un servidor para cargar la base de datos y que la aplicación cuente con la información que necesita para su correcto funcionamiento.

Una vez en el servidor, deberá ejecutar los siguientes comandos (a poder ser dentro de un entorno virtual para mantener una ejecución independiente al resto de su entorno):

```
>> pip install -r requirements.txt
```

```
>> python main.py
```

Funcionalidad específica del administrador en un futuro aprovechando el panel de control de usuarios ya creado en la aplicación web y preparado con ese fin (ver capítulo [“Trabajo a futuro”](#)):

1. Una vez se ha iniciado sesión con una cuenta válida de administrador, en la página principal aparecerá un botón “Configuración”. Hacer clic en dicho botón para ir a la página del administrador.
2. En la página del administrador se muestra una tabla con todos los usuarios de la aplicación, se muestra nombre, email, nombre, apellido y rol, además de un botón de edición para cada usuario y una opción para eliminarlo. A parte de la tabla de usuarios, se muestra un botón “Actualizar base de datos utilizando Web Scrapping”.
3. Al hacer clic en el botón “Actualizar base de datos utilizando Web Scrapping”, aparecerá un formulario para definir los distintos parámetros con los que se va a lanzar el script.

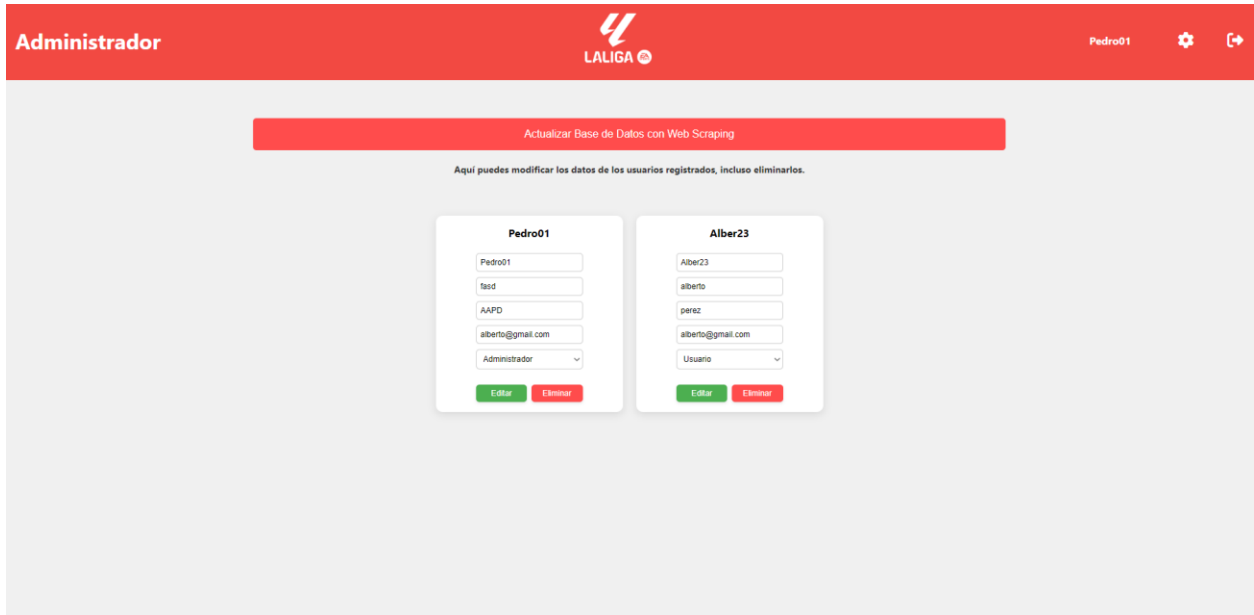


Figura 23. Página del administrador

7. Pruebas de ejecución

Para la realización de las diferentes pruebas de ejecución tenemos dos aspectos claves: el input que le pasamos al modelo y el análisis del output de éste para hacer las correcciones necesarias para afinar cada vez más la salida del modelo y hacer que se parezca lo máximo posible a lo que queremos conseguir.

A continuación, se describirá el input que recibe Gemma para cada tipo de petición, qué archivos son los encargados de estas tareas y las crónicas/resúmenes obtenidos.

7.1. Resumen de un partido

Al seleccionar el partido del que queremos extraer la crónica se mostrarán dos opciones para que el usuario escoja y descargue la que más le guste. Para el análisis del output recibido de un partido utilizaremos como ejemplo el enfrentamiento entre el Girona y el Real Valladolid de la Jornada 5 de la temporada 2022-2023.

7.1.1. Input

Para esta tarea el archivo *input_resumen_partido.py* es el encargado de pasarle los datos de entrada que utilizará Gemma para generar la crónica del partido que se le haya solicitado. Además de incluir los datos típicos de un partido, como pueden ser los goles, tarjetas y árbitros, hemos querido enriquecer la generación de este tipo de crónicas utilizando eventos destacados del encuentro, escogiendo los que pueden enriquecer más la narración. Ejemplos de estos eventos pueden ser altercados que hayan ocurrido durante su disputa o ceremonias de despedida de jugadores/árbitros.

- Prompt en el que se le pide que haga un resumen de una jornada teniendo en cuenta los datos que se le están pasando (ver Figura 24).

```

Imaginate que eres un periodista deportivo, genera una crónica deportiva del partido de fútbol con los siguientes datos:

Fecha del partido: V-09/09 21:00
Equipo local: Girona
Resultado del partido: 2 - 1
Goles equipo local: 2
Goles equipo visitante: 1
Equipo visitante: Real Valladolid
Árbitro del encuentro: César Soto Grado
Árbitros del VAR: Santiago Jaime Latre, Íñigo Prieto López de Ceraín
Tarjetas: [{'amonestado': 'Anuar', 'minuto': 8, 'color': 'Amarilla'}, {'amonestado': 'Roque Mesa', 'minuto': 78, 'color': 'Amarilla'}]
Jornada: 5
Cambios hechos: [{'jugador_sale': 'Shon Weissman', 'minuto': 5, 'jugador_entra': 'Sergi Guardiola'}, {'jugador_sale': 'Iván Fresneda', 'minuto': 27, 'jugador_entra': 'Luis Pérez'}, {'jugador_sale': 'Óscar Plano', 'minuto': 45, 'jugador_entra': 'Anuar Tuhami'}, {'jugador_sale': 'Samu Sáiz', 'minuto': 58, 'jugador_entra': 'Rodrigo Riquelme'}, {'jugador_sale': 'Manu Vallejo', 'minuto': 58, 'jugador_entra': 'Bernardo Espinosa'}, {'jugador_sale': 'Toni Villa', 'minuto': 73, 'jugador_entra': 'Reinier'}, {'jugador_sale': 'Yan Couto', 'minuto': 73, 'jugador_entra': 'Arnau Martínez'}, {'jugador_sale': 'Álvaro Aguado', 'minuto': 75, 'jugador_entra': 'Kike Pérez'}, {'jugador_sale': 'Kenedy', 'minuto': 75, 'jugador_entra': 'Gonzalo Plata'}, {'jugador_sale': 'Javi Hernández', 'minuto': 91, 'jugador_entra': 'Valentín Castellanos'}]
Goles anotados: [{'resultado': '1-0', 'minuto': 20, 'anotador': 'Reinier Carvalho'}, {'resultado': '1-1', 'minuto': 37, 'anotador': 'Monchu'}, {'resultado': '2-1', 'minuto': 87, 'anotador': 'Oriol Romeu'}]

```

Figura 24. Input datos estadísticos resumen de partido

- También se le proporcionan los eventos que se hayan considerado más relevantes.

```

Eventos: ['1. El Valladolid no pudo mantener su racha de victorias consecutivas y cayó 1-2 frente al Girona.', '2. El mediocampista Oriol Romeu, en su segundo partido con el equipo catalán, anotó el gol de la victoria en el minuto 88, lo que aumenta su simpatía entre la afición.', '3. El partido estuvo lleno de momentos de tensión y emoción, con varias lesiones en ambos equipos que afectaron el ritmo del juego.', '4. Los jugadores lesionados fueron David López (Girona) en el calentamiento, y Sergi Guardiola, Luis Pérez y Kike (Valladolid).', '5. El Girona abrió el marcador gracias a un gol de Reinier, quien hizo historia como el goleador más joven del equipo en Primera División.', '6. Reinier agradeció el gol a Aleix García, quien realizó una jugada individual de gran calidad para asistir al brasileño.', '7. Anuar del Valladolid recibió una tarjeta amarilla en el minuto 30, y el árbitro optó por no mostrarle la segunda pese a su conducta peligrosa durante el juego.', '8. Monchu anotó el gol del empate y pidió perdón a la afición del Girona por celebrar el gol, debido a su pasado en ese equipo.', '9. En la segunda mitad, Gonzalo Plata reclamó un penalti que el árbitro y el VAR rechazaron.', '10. A pesar de ser inferiores en el juego, el Valladolid presionó por el segundo gol y lo consiguió gracias a Oriol Romeu.', 'En resumen, el partido estuvo lleno de acción y tensión, con múltiples oportunidades de gol y momentos de drama para ambos equipos. El Valladolid no pudo mantener la ventaja y el Girona logró la victoria en el último minuto con un gol de Oriol Romeu.']
Temporada del encuentro: 2022_2023

```

Figura 25. Eventos relevantes de la jornada

7.1.2. Output

A continuación, mostramos las dos crónicas que ha ofrecido la aplicación como respuesta a nuestra petición (ver Figura 26 y Figura 27).

Crónica 1:

Girona se impone al Valladolid en una emocionante jornada **Girona, 09/09/2023** - La exigente Liga Española ofreció una batalla épica en el estadio Municipal de Girona, donde el conjunto catalán superó al Real Valladolid por 2-1 en una noche llena de emociones. Un triunfo que le da una nueva inyección de confianza al equipo dirigido por Michel González. El inicio fue explosivo. A solo 20 minutos, **Reinier Carvalho**, el nuevo ídolo del Girona, marcó el primer gol del partido, marcando su nombre en la historia del club como el jugador más joven en anotar para el equipo en la máxima categoría. Su asistencia llegó gracias a una jugada brillante de Aleix García. Sin embargo, el Valladolid no desistió. Con el paso del tiempo, lograron igualar las acciones. **Monchu**, exjugador del Girona, convirtió el gol del empate a 37 minutos. Este momento de tensión provocó un cambio en el ritmo del juego, y el partido se mantuvo envuelto en una atmósfera electrizante hasta el final. La segunda parte fue un espectáculo de goles, pero también de tensiones. La lesión de David López en el calentamiento del Girona, y luego de Sergi Guardiola y Luis Pérez en el campo, obligó a ambos técnicos a hacer cambios constantes. Finalmente, **Oriol Romeu**, quien ingresó en el minuto 67, encontró el gol definitivo en el minuto 87, sentenciando la victoria del Girona. Esta anotación consolidó la figura del futbolista español, quien demostró su talento y liderazgo en este partido crucial. Los dos equipos se enfrentaron en un duelo intenso con varios incidentes. El defensa del Valladolid, Roque Mesa, recibió una amarilla tras una falta peligrosa. Y aunque el partido estuvo marcado por diversos incidentes, tanto el Girona como el Valladolid destacaron por su intensidad y compromiso. Este resultado coloca al Girona en una posición favorable dentro del campeonato, mientras que el Real Valladolid deberá analizar sus errores para retomar la línea ascendente. **Un resumen de los eventos clave:**

- Debut exitoso de Reinier:** El brasileño firmó su nombre en la historia del Girona como el goleador más joven en la liga.
- Reconocimiento por el gol:** Reinier mostró agradecimiento a Aleix García por la asistencia.
- Controversia sobre la actuación del árbitro:** Se discutió la decisión de no otorgar un penalti a Gonzalo Plata.
- Eficiencia del Girona:** El equipo catalán aprovechó bien las oportunidades y supo gestionar el partido hasta el final. Esta victoria es un hito importante para el Girona, que consolida su presencia en la élite del fútbol español. El futuro parece prometedor para el equipo catalán, especialmente si continúan mostrando esta misma actitud y determinación.

Figura 26. Crónica 1 de un partido

Análisis de la crónica 1:

Puntos positivos:

- Interpreta correctamente equipos enfrentados, resultado, entrenadores, equipo local y visitante y fecha.
- Menciona el hecho de que Reinier hizo historia en su equipo, el Girona, al convertirse en el jugador más joven en marcar en la Liga con este equipo. También se hace referencia correctamente a que fue el encargado de abrir el marcador y de sumar el primer tanto para en conjunto catalán en el minuto 20 (con asistencia de Aleix García).
- Empate del encuentro a manos de Monchu, exjugador del Girona, en el minuto 37 que igualó el partido y le dio más tensión.
- Hace mención de las diferentes lesiones producidas durante el encuentro y de su repercusión al obligar a los entrenadores a hacer cambios.
- Oriol le dio la victoria al Girona en el minuto 87 con un gol que sentenció el partido, como bien se comenta. "Sentenciar un partido" es una expresión muy típica en una narración deportiva y está bien usada en este contexto.
- Mención de eventos destacados correctos, entre ellos el hecho de la no concesión de un penalti a Gonzalo Plata.
- Hincapié en que hubo incidentes, ya que el penalti no pitado y la segunda tarjeta amarilla perdonada a Anuar pueden tomarse como tal.

Puntos a mejorar:

- "La segunda parte fue un espectáculo de goles, pero también de tensiones" es una afirmación un poco exagerada, ya que solo hubo una anotación en la segunda mitad del partido.
- Se afirma que Oriol ingresó al terreno de juego en el minuto 67, cuando ese dato no es real.

Crónica 2:

Girona se impone al Real Valladolid en un duelo emocionante **El Girona logra una importante victoria ante el Real Valladolid (2-1)** en un partido disputado en el estadio Municipal de Montilivi. La afición local disfrutó de un partido vibrante, marcado por ocasiones claras de gol, tensiones constantes y momentos cruciales. La primera parte del partido fue dominada por el Girona, quien abrió el marcador a través de **Reinier Carvalho**, quien aprovechó una asistencia magistral de Aleix García para convertir su primer gol en la Liga española. Esta anotación le dio a Reinier el récord de ser el **goleador más joven del Girona en Primera División**. Sin embargo, el Real Valladolid reaccionó rápidamente al inicio de la segunda parte. Un error defensivo permitió a **Monchu** igualar el marcador a falta de tiempo para llegar al descanso. Tras un intenso choque, la lucha por la victoria continuó hasta el final. Finalmente, **Oriol Romeu** marcó el gol decisivo para el Girona a los 87 minutos. Su potente disparo tras una buena intervención de Aleix García sentenció la victoria del equipo catalán. Este triunfo coloca al Girona en una posición favorable en la clasificación, mientras que el Real Valladolid deberá trabajar para recuperar su dinámica después de esta derrota.

****Destacados:****

- * **Reinier Carvalho**: Debutante en la liga española, demostrando su talento con un golazo que marca un hito histórico para el club.
- * **Oriol Romeu**: Contundente actuación en la última etapa del partido, marcando el gol definitivo para el Girona.
- * **Aleix García**: Se destacó por su asistencia clave para Reinier, generando diversas oportunidades de ataque.

****Aspectos negativos:****

- * **Real Valladolid**: No pudieron mantener la ventaja obtenida en la primera parte y sufrieron un duro golpe por la derrota.
- * **Lesiones**: El partido estuvo marcado por las lesiones de varios jugadores de ambos equipos.

****Resumen:**** Un partido apasionante y memorable donde el Girona superó la presión y conquistó tres puntos importantes. Con este resultado, el Girona demuestra su potencial y aspiraciones en la temporada. Mientras tanto, el Real Valladolid tendrá que analizar sus errores para buscar una respuesta inmediata en su siguiente compromiso.

Figura 27. Crónica 2 de un partido

Análisis de la crónica 2:

Puntos positivos:

- Equipos, resultado y demás datos mencionados a lo largo de la crónica son correctos.
- Al saber que el equipo local es el Girona, el modelo menciona al elaborar la crónica que el encuentro se llevó a cabo en Montilivi.
- Mención correcta de que Reinier hizo historia en el Girona explicando bien el porqué. Además, se dice que abre el marcador y pone por delante al Girona con una asistencia magistral de Aleix, como así fue.
- Monchu empata el encuentro.
- Victoria gracias al gol de Oriol “al final del partido”. Este gol lo anotó en el 87, como un partido dura 90 más añadido está bien utilizada la expresión.

Puntos a mejorar:

- Sección de eventos destacados puede ser demasiado extensa en comparación a la crónica en sí, aunque sean eventos interesantes y bien mencionados.
- Se dice que Monchu empata el encuentro en la segunda parte, cuando fue en el minuto 37 y, como cada parte son 45 minutos, la afirmación temporal es incorrecta.

7.2. Resumen de una jornada

Se va a utilizar como ejemplo la ejecución de una petición de resumen de la Jornada número 6 de la temporada 2022-2023.

Como resumir lo ocurrido durante una jornada completa requiere filtrar los datos de todos los partidos de esta para solo mencionar los hechos más destacados, hemos trabajado los datos de todos los partidos para quedarnos con la información más importante.

7.2.1. Input

- Prompt en el que se le pide que haga un resumen de una jornada teniendo en cuenta los datos que va a recibir (ver Figura 28).

Imagínate que eres un periodista deportivo, escribe un resumen de la jornada de la que forman parte los 10 partidos con los siguientes datos:

Figura 28. Prompt input para generar crónica de una jornada

- Se mostrará por cada uno de los 10 partidos disputados durante esa jornada: la fecha y hora del encuentro, equipo local, visitante y resultado y las tarjetas rojas y los amonestados. Además, se mencionarán los goles anotados en los primeros 10 minutos del encuentro y los anotados a partir del minuto 80 y los cambios en los que el jugador que ingresa al terreno de juego haya marcado algún gol (ver Figura 29 y Figura 30).

```
Fecha del partido 1: V-16/09 21:00
Equipo local del partido 1: Real Valladolid
Resultado del partido 1: 0 - 1
Equipo visitante del partido 1: Cádiz
Tarjetas rojas del partido 1: []
Cambios hechos durante el partido 1: []
Goles anotados en el partido 1: [{"resultado": "0-1", "minuto": 91, "anotador": "Negredo", "id": ""}]
Temporada del encuentro 1: 2022_2023

Fecha del partido 2: S-17/09 14:00
Equipo local del partido 2: Mallorca
Resultado del partido 2: 1 - 0
Equipo visitante del partido 2: Almería
Tarjetas rojas del partido 2: []
Cambios hechos durante el partido 2: []
Goles anotados en el partido 2: []
Temporada del encuentro 2: 2022_2023

Fecha del partido 3: S-17/09 16:15
Equipo local del partido 3: Barcelona
Resultado del partido 3: 3 - 0
Equipo visitante del partido 3: Elche
Tarjetas rojas del partido 3: [{"amonestado": "Gonzalo Verdú", "minuto": 13, "color": "Roja", "id": ""}, {"amonestado": "Rodríguez", "minuto": 44, "color": "Roja", "id": ""}]
Cambios hechos durante el partido 3: []
Goles anotados en el partido 3: []
Temporada del encuentro 3: 2022_2023

Fecha del partido 4: S-17/09 18:30
Equipo local del partido 4: Valencia
Resultado del partido 4: 3 - 0
Equipo visitante del partido 4: Celta
Tarjetas rojas del partido 4: [{"amonestado": "Franco Cerúdi", "minuto": 59, "color": "Roja", "id": ""}]
Cambios hechos durante el partido 4: [{"jugador_sala": "Justín Klüivert", "minuto": 70, "jugador_entra": "Samu Castillejo", "id": ""}]
Goles anotados en el partido 4: [{"resultado": "2-0", "minuto": 81, "anotador": "Marcos André", "id": ""}, {"resultado": "3-0", "minuto": 92, "anotador": "André Almeida", "id": ""}]
Temporada del encuentro 4: 2022_2023

Fecha del partido 5: S-17/09 21:00
Equipo local del partido 5: Athletic
Resultado del partido 5: 3 - 2
Equipo visitante del partido 5: Rayo
Tarjetas rojas del partido 5: []
Cambios hechos durante el partido 5: [{"jugador_sala": "Oier Zarraga", "minuto": 79, "jugador_entra": "Hico Williams", "id": ""}]
Goles anotados en el partido 5: [{"resultado": "0-1", "minuto": 4, "anotador": "Trejo", "id": ""}]
Temporada del encuentro 5: 2022_2023
```

Figura 29. Input datos para crónica de una jornada (parte 1)

```

Fecha del partido 6: D-18/09 14:00
Equipo local del partido 6: Osasuna
Resultado del partido 6: 0 - 2
Equipo visitante del partido 6: Getafe
Tarjetas rojas del partido 6: [{"amonestado": "Chimy Ávila", "minuto": 42, "color": "Roja", "id": ""}]
Cambios hechos durante el partido 6: [{"jugador_sale": "Álex Revuelta", "minuto": 93, "jugador_entra": "Gastón Álvarez", "id": ""}]
Goles anotados en el partido 6: []
Temporada del encuentro 6: 2022_2023

Fecha del partido 7: D-18/09 16:15
Equipo local del partido 7: Villarreal
Resultado del partido 7: 1 - 1
Equipo visitante del partido 7: Sevilla
Tarjetas rojas del partido 7: []
Cambios hechos durante el partido 7: []
Goles anotados en el partido 7: [{"resultado": "0-1", "minuto": 7, "anotador": "Óliver Torres", "id": ""}]
Temporada del encuentro 7: 2022_2023

Fecha del partido 8: D-18/09 18:30
Equipo local del partido 8: Betis
Resultado del partido 8: 2 - 1
Equipo visitante del partido 8: Girona
Tarjetas rojas del partido 8: [{"amonestado": "Sánchez", "minuto": 20, "color": "Roja", "id": ""}]
Cambios hechos durante el partido 8: []
Goles anotados en el partido 8: [{"resultado": "0-1", "minuto": 6, "anotador": "Arnau Martínez", "id": ""}]
Temporada del encuentro 8: 2022_2023

Fecha del partido 9: D-18/09 18:30
Equipo local del partido 9: R. Sociedad
Resultado del partido 9: 2 - 1
Equipo visitante del partido 9: Espanyol
Tarjetas rojas del partido 9: []
Cambios hechos durante el partido 9: [{"jugador_sale": "Dani Gómez", "minuto": 74, "jugador_entra": "Edu Expósito", "id": ""}]
Goles anotados en el partido 9: []
Temporada del encuentro 9: 2022_2023

Fecha del partido 10: D-18/09 21:00
Equipo local del partido 10: Atlético
Resultado del partido 10: 1 - 2
Equipo visitante del partido 10: Real Madrid
Tarjetas rojas del partido 10: []
Cambios hechos durante el partido 10: []
Goles anotados en el partido 10: [{"resultado": "1-2", "minuto": 82, "anotador": "Mario Hermoso", "id": ""}]
Temporada del encuentro 10: 2022_2023

```

Figura 30. Input datos para crónica de una jornada (parte 2)

7.2.2. Output

El siguiente es un ejemplo de resumen que daría el modelo al recibir el input mencionado anteriormente (ver Figura 31 y Figura 32):

Crónica 1:

Resumen Jornada 🏆 ¡El fútbol se vive! La jornada deportiva nos ha regalado una serie de emociones intensas, desde victorias contundentes hasta duelos apretados. Aquí te presentamos un resumen de lo sucedido: ****Partido 1:**** Real Valladolid cae ante un sólido Cádiz (1-0). Negredo marcó el único gol para los visitantes. La acción estuvo marcada por una buena defensa y estrategia táctica del equipo gaditano. ****Partido 2:**** Mallorca logra imponerse a Almería con un solitario tanto (1-0), mostrando su buen ritmo ofensivo. ****Partido 3:**** Un Barcelona imponente domina al Elche (3-0) gracias a goles de Lewandowski, Dembélé y Pedri. Los jugadores catalanes demostraron una excelente conexión y dominio sobre el campo. ****Partido 4:**** Un Valencia explosivo vence a Celta Vigo (3-0) con dos goles de Marcos André y uno de Andre Almeida. Una actuación brillante de los valencianos con una gran intensidad y control del balón. ****Partido 5:**** El Athletic Club lidera el duelo contra el Rayo Vallecano (3-2) tras una emocionante lucha. Trejo abrió el marcador para los rojiblancos. ****Partido 6:**** Osasuna no pudo superar la superioridad de Getafe (0-2), cayendo derrotada en casa. Chimy Ávila fue expulsado por un fallo en la jugada. ****Partido 7:**** Un empate sin goles entre Villarreal y Sevilla deja la disputa abierta. Ambos equipos mostraron solidez defensiva. ****Partido 8:**** Un Betis vibrante derrota a Girona (2-1) con un gol de Sánchez y otro de Arnau Martínez. Un triunfo importante para el conjunto verdiblanco. ****Partido 9:**** La Real Sociedad se impone en tierras peruanas ante Espanyol (2-1) con goles clave. Dani Gómez salió en el minuto 74, siendo reemplazado por Edu Expósito. ****Partido 10:**** Un Atlético Madrid desafiante cae ante un formidable Real Madrid (1-2) en una batalla épica. Mario Hermoso anotó el gol del descuento para los colchoneros. Esta jornada ha sido un claro ejemplo de la pasión y emoción que caracteriza el deporte. Esperemos seguir disfrutando de estos momentos deportivos en las próximas jornadas.

Figura 31. Crónica 1 de una jornada

Análisis de la crónica 1:

Puntos positivos:

- Enfrentamientos, cambios, tarjetas y resultados mencionados son correctos.
- Trejo abrió el marcador es correcto, interpreta bien que es el primer gol del equipo y del partido.
- Chimy Ávila expulsado.
- Saca como conclusión de un empate a 0 entre Villarreal y Sevilla que ambos contaron con una buena defensa, lo que tiene sentido por el resultado.
- Expresión “verdiblanco”, “rojiblanco” o “colchoneros” para referirse a aficiones, expresiones que no se le han dado como parte del input al modelo y que interpreta y utiliza correctamente.
- Cambios de jugadores mencionados son correctos.
- Frase final muy típica en los diarios deportivos.

Puntos a mejorar:

- Goleadores del enfrentamiento FC Barcelona y Elche mal dados.

Crónica 2:

Resumen Jornada Fútbol España: La Liga **Una jornada llena de emociones y resultados inesperados.** Los equipos lucharon por cada punto, dejando una serie de encuentros apasionantes para recordar. Aquí te presento un resumen de lo ocurrido: * **Real Valladolid cae ante un sólido Cádiz:** Los pucelanos se vieron superados por un equipo como Cádiz, que logró imponer su juego y ganar con un solitario gol. * **Mallorca vence a Almería:** Un triunfo contundente para el conjunto mallorquín, quien demostró su solidez defensiva y anotó un gol decisivo. * **Barcelona domina al Elche:** Una victoria clara para el Barça, mostrando su superioridad técnica y táctica. Se notaron las tensiones dentro del equipo debido a las tarjetas amarillas recibidas por Gonzalo Verdú y Rodríguez. * **Valencia lidera con contundencia:** El Valencia aprovechó sus oportunidades para derrotar al Celta con dos goles marcados en el segundo tiempo. Franco Cervi fue expulsado tras un penalti cometido. * **Athletic logra un triunfo ajustado frente al Rayo Vallecano:** A pesar de estar perdiendo desde el inicio, el Athletic supo remontar y lograr una victoria ajustada. Oier Zarraga fue sustituido por Nico Williams. * **Osasuna cede terreno ante Getafe:** El Getafe se impuso con facilidad sobre Osasuna, marcando dos goles antes del descanso. Chimy Ávila fue expulsado por una falta peligrosa. * **Villarreal empatata con un Sevilla sin brillo:** Dos equipos sin mucha inspiración se enfrentaron en un encuentro parejo. No hubo goles hasta el final del partido, cuando Oliver Torres marcó para el Sevilla. * **Betis sorprende a Girona:** Tras una primera parte complicada, los verdiblanco lograron vencer a Girona con un gol de Arnau Martínez. Sánchez recibió una tarjeta roja por una falta innecesaria. * **R.Sociedad derrota a Espanyol:** La R.Sociedad se mostró más dinámico y efectivo, consiguiendo una victoria importante. Dani Gómez fue sustituido por Edu Expósito. * **Atlético sufre una derrota contra el Real Madrid:** En un partido muy disputado, el Real Madrid logró imponerse con un gol de Mario Hermoso. Esta jornada ha dejado claro que la lucha por la cima continúa siendo intensa. ¡Esperamos ver qué sorpresas nos deparan las próximas jornadas!

Figura 32. Crónica 2 de una jornada

Análisis de la crónica 2:

Puntos positivos:

- Enfrentamientos, cambios, tarjetas y resultados mencionados son correctos.
- Expresión “pucelanos” para referirse a aficionados son expresiones que no se le han dado como parte del input al modelo y que interpreta y utiliza correctamente.
- “Ganar con un solitario gol”, se parece mucho a la típica frase de periodista deportivo “gol solitario” cuando hablan de un 0-1 o 1-0.
- “Goles marcados en el segundo tiempo”, interpreta correctamente que a partir del minuto 45’ es la segunda mitad y lo utiliza como recurso para enriquecer la narración del encuentro.
- Remontada del Athletic que, efectivamente, empezó perdiendo contra el Rayo y acabó llevándose el partido.
- Frase final muy típica en los diarios deportivos.

Puntos a mejorar:

- Interpreta de forma errónea la expulsión en el partido del Celta contra el Valencia como un penalti. Los goles y la expulsión fueron independientes, la expulsión no derivó en penalti.
- “Sánchez recibió una tarjeta por una falta innecesaria” no sabemos si es una afirmación cierta del todo, es algo subjetivo.

7.3. Resumen de una temporada

Se va a utilizar como ejemplo una ejecución de petición de resumen de la temporada 2022-2023.

Como resumir lo ocurrido durante una temporada completa requiere de un trabajo de síntesis importante de la información de la que disponemos, hemos llevado a cabo varios pasos para conseguir los datos más relevantes y significativos para esta tarea.

7.3.1. Input

El input consta de un string que incluye varios aspectos claves para el resumen de una temporada:

- Prompt en el que se le pide que haga un resumen de una temporada teniendo en cuenta los datos que va a recibir.

```
Imagínate que eres uno de los mejores periodistas deportivos, escribe un resumen de la temporada 2022_2023 explicando como se ha ido desarrollando si cuentas con los siguientes datos sobre ella:
```

Figura 33. Prompt del input para generar la crónica de una temporada

- Equipos con más victorias y con más derrotas por la mínima (1-0 o 0-1) y cuántas han sido.

```
El equipo con más derrotas por 1-0 o 0-1 es Valencia, con un total de 9 derrotas.  
El equipo con más victorias por 1-0 o 0-1 es Barcelona, con un total de 11 victorias.  
Resumen de la temporada 2022_2023:
```

Figura 34. Input para crónica de temporada (parte 1)

- Equipos con más victorias y derrotas por una diferencia de cuatro goles o más y sus detalles.

```
Equipos con las mayores victorias o derrotas por diferencia de 4 goles o más:
```

```
Real Valladolid:
```

```
Derrota por 6 goles: Real Madrid vs Real Valladolid 6 - 0  
Derrota por 4 goles: Barcelona vs Real Valladolid 4 - 0
```

```
Real Madrid:
```

```
Victoria por 6 goles: Real Madrid vs Real Valladolid 6 - 0  
Victoria por 4 goles: Real Madrid vs Elche 4 - 0
```

```
Atlético:
```

```
Victoria por 5 goles: Atlético vs Sevilla 6 - 1  
Victoria por 4 goles: Atlético vs Cádiz 5 - 1
```

Figura 35. Input para crónica de temporada (parte 2)

- Clasificación final de esa temporada concreta.

```
Puntos totales del Barcelona: 88
Puntos totales del Real Madrid: 78
Puntos totales del Atlético: 77
Puntos totales del Real Sociedad: 71
Puntos totales del Villarreal: 64
Puntos totales del Real Betis: 60
Puntos totales del Osasuna: 53
Puntos totales del Athletic: 51
Puntos totales del Mallorca: 50
Puntos totales del Girona: 49
Puntos totales del Rayo Vallecano: 49
Puntos totales del Sevilla: 49
Puntos totales del Celta: 43
Puntos totales del Cádiz: 42
Puntos totales del Getafe: 42
Puntos totales del Valencia: 42
Puntos totales del Almería: 41
Puntos totales del Real Valladolid: 40
Puntos totales del Espanyol: 37
Puntos totales del Elche: 25
```

Figura 36. Input para crónica de una temporada (parte3)

- Los equipos con más victorias, con más empates y con más derrotas.

```
Estadísticas destacadas de los equipos durante la temporada 2022_2023:
Barcelona está entre los dos equipos con más victorias: 28.
Real Madrid está entre los dos equipos con más victorias: 24.
Real Valladolid está entre los dos equipos con más derrotas: 20.
Espanyol tuvo el mayor número de empates: 13.
Elche está entre los dos equipos con más derrotas: 23.
```

Figura 37. Input para crónica de una temporada (parte 4)

7.3.2. Output

El siguiente es un ejemplo de resumen que daría el modelo al recibir el input mencionado anteriormente (ver Figura 38 y Figura 39):

Crónica 1:

La Liga 2022/2023: Un año marcado por la hegemonía azulgrana y la lucha constante por los puestos europeos La temporada 2022/2023 en La Liga fue una apasionante batalla por el título, donde el **FC Barcelona** se impuso como el rey indiscutible, demostrando su dominio absoluto al sumar un récord histórico de **44 victorias**. Sin embargo, no todo estuvo a favor del Barça, ya que la presión por mantener la cima llevó a algunos altibajos en su rendimiento. En cuanto a la lucha por los puestos europeos, **el Real Madrid**, liderado por **Carlo Ancelotti**, mantuvo su posición dominante, culminando la temporada con un sólido **78 puntos**. Sin embargo, el conjunto blanco no pudo evitar enfrentarse a una serie de desafíos, especialmente en el tramo final de la competición. Un elemento clave para entender la temporada fue la dinámica del **Atletico de Madrid**. Los colchoneros lograron un impresionante **77 puntos**, consolidándose como un rival directo para ambos titanes. Su capacidad para mantener una consistencia notable durante toda la campaña les permitió luchar por los puestos de Champions League hasta el último minuto. Otro punto destacado fue el desempeño de **Real Sociedad**, que logró un brillante **71 puntos** y disputó la pelea por los puestos europeos con intensidad. Sus actuaciones fueron cruciales en la búsqueda de un lugar dentro de la élite europea. Por otro lado, el **Valencia CF** sufrió una temporada marcada por una abrumadora cantidad de derrotas por 1-0 o 0-1 (con **36 derrotas**) lo que afectó significativamente su clasificación. Esta tendencia se repitió en otras posiciones de la tabla, con equipos como el **Rayo Vallecano** y el **Cádiz**, quienes también tuvieron dificultades para alcanzar la seguridad en sus puestos. **Algunos momentos destacados:** * **El triunfo de Xavi Hernández**: Con una gestión impecable, Xavi transformó al FC Barcelona en un equipo dinámico y ofensivo. La llegada de jugadores jóvenes como Pedri y Gavi le dio un nuevo aire a la plantilla. * **El regreso del fútbol español a la máxima competencia**: El Real Madrid demostró su poderío tras una temporada complicada, mientras que el Atleti mantuvo su forma consistente y desafió constantemente al líder. * **Una temporada llena de emociones**: Las lesiones y las bajas jugadas por los equipos, así como los resultados inesperados, dieron un toque de incertidumbre a la liga. A pesar de las diferencias en puntos, todos los equipos mostraron gran competitividad y pasión por el juego. La temporada 2022/2023 dejó una huella imborrable en la historia de La Liga, marcando un antes y un después en la carrera hacia la gloria.

Figura 38. Crónica 1 de una temporada

Análisis de la crónica 1:

Puntos positivos:

- La crónica comienza destacando la hegemonía del FC Barcelona, esto tiene sentido ya que, si nos fijamos en la clasificación final, vemos que es el equipo que ha ganado la competición esa temporada.

- De igual forma, si vemos la clasificación general podemos dar por sentado que los puestos clasificatorios para competiciones europeas estuvieron también disputados al fijarnos en las diferencias de puntos.
- Siempre que se comenta los puntos conseguidos por algún equipo al final de la liga son correctos.
- Entrenadores y equipos que se nombran a lo largo del escrito son acordes también a la realidad.
- Utiliza expresiones como “colchoneros” para referirse a la afición del Atlético de Madrid de forma correcta a pesar de no haber alimentado al modelo con ellas a través del input. De la misma forma habla de competiciones europeas con sentido teniendo en cuenta la clasificación final de la liga y los puestos en los que los equipos deben quedar para clasificarse para jugarlas. Esto denota que el modelo ha “comprendido” correctamente el contexto y lo que debe comentar.
- Menciona a Xavi Hernández como clave en el FC Barcelona esta temporada. Es cierto que en su segunda temporada (primera completa), consiguió el título de liga con el equipo, siendo un hecho destacable.
- Pedri y Gavi fueron piezas claves durante la temporada.
- FC Barcelona, Real Madrid y Atleti luchando por los primeros puestos.

Puntos a mejorar:

- Récord histórico de 44 victorias para el FC Barcelona es falso.
- Menciona al Rayo Vallecano, Valencia y al Cádiz como equipos que han tenido problemas por mantener la permanencia en la categoría... Valencia y Cádiz, efectivamente, han estado muy cerca de descender “peleando” con el Real Valladolid (que es el que finalmente descendió con más puntos) con tan solo 2 puntos de diferencia, el Rayo Vallecano por su parte, con 49 puntos, no destacaría como un candidato al descenso como así afirma la crónica de esta temporada.

- Frase final demasiado “de película” para una crónica deportiva, no ha sido una temporada tan determinante ni especial como se expone.

Crónica 2:

La Liga 2022/2023: Un Año De Alto Nivel y Una Nueva Era para Barcelona La temporada 2022/2023 en La Liga fue una verdadera batalla por la cima, donde el dominio habitual de Real Madrid y Atlético se vio desafiado por un nuevo contendiente: **el brillante Barcelona**. A pesar de no haber logrado el título, el Barça demostró su capacidad para competir a nivel internacional, dejando claro que están listos para volver a ser protagonistas. **Un Barça Revolucionario**: Barça lideró la tabla con una impresionante cantidad de **44 victorias**, incluyendo varias goleadas contundentes. Esta victoria se basaba en una ofensiva dinámica y agresiva, liderada por jugadores como Gavi y Pedri, quienes sentaron las bases de un juego moderno y vistoso. Su entrenador Xavi Hernández implementó un estilo de juego basado en la posesión del balón y la creación de oportunidades, lo cual le dio resultados positivos. **Los Campeones de la Fortaleza**: Mientras tanto, Real Madrid mantuvo su posición dominante, pero sin alcanzar la misma intensidad que en temporadas anteriores. Sin embargo, sus victorias fueron numerosas, con **75 victorias** y una solidez defensiva que les permitió mantener la presión constante. **Atlético Entre las Potencias**: Los rojiblancos también mostraron su fuerza, logrando una gran temporada con **33 victorias** y mostrando una defensa sólida que les permitía controlar partidos. **Una Temporada Con Variedades**: No obstante, la liga no estuvo exenta de sorpresas. El Real Sociedad, con **35 victorias**, se posicionó como un rival serio al acercarse al top. También hubo momentos brillantes de otros equipos, como el **Valencia**, que se enfrentó a grandes desafíos con un rendimiento irregular. El Girona, con un sistema táctico innovador y un enfoque agresivo, logró destacarse con un buen balance de victorias y empates. **Las Estadísticas Destacadas**: El éxito del Barcelona se refleja en su superioridad en cuanto a victorias, mientras que el Real Madrid se destacó por su poderío en ataque. En términos generales, la temporada fue marcada por la competitividad y la innovación, con varios equipos experimentando cambios significativos en su estrategia y plantilla. En definitiva, la temporada 2022/2023 dejó un legado memorable en La Liga, con un nuevo campeón emergiendo y otros equipos mostrando un potencial aún incierto. La lucha por el título será siempre un espectáculo emocionante, y este año nos mostró que cualquier equipo puede aspirar a ganar.

Figura 39. Crónica 2 de una temporada

Análisis de la crónica 2:

Puntos positivos:

- Utiliza expresiones como “Nueva Era para Barcelona” y “juego de posesión del balón como estilo de Xavi Hernández” de forma correcta a pesar de no haber alimentado al modelo con ellas a través del input. De la misma forma habla de competiciones europeas con sentido teniendo en cuenta la clasificación final de la liga y los puestos en los que los equipos deben quedar para clasificarse para jugarlas. Esto denota que el modelo ha “comprendido” correctamente el contexto y lo que debe comentar.
- Pedri y Gavi como jugadores claves en el Barcelona.
- Xavi Hernández como entrenador del FC Barcelona.
- Real Sociedad como equipo luchando por las primeras posiciones.
- Buen balance del Girona.

Puntos a mejorar:

- Llamar “nuevo contendiente” al FC Barcelona para la victoria de la competición como si estuviesen normalmente alejados del Real Madrid y Atlético de Madrid en esta lucha no es una afirmación real.
- 44 victorias del FC Barcelona es una afirmación falsa.
- La forma de narrar los acontecimientos no queda en algunos casos natural o no suena realista.
- La última afirmación del Barcelona como “cualquier equipo” sorprendiendo al ganar la competición cuando suele estar entre los tres primeros luchando por la competición es extraña.

8. Conclusiones y trabajo futuro

En este capítulo se indican las conclusiones del trabajo realizado y se describen aspectos de la aplicación que podrían mejorarse y ampliarse en un futuro.

8.1. Conclusiones

Uno de los aspectos claves en el trabajo de un periodista es la rapidez con la que publica sus trabajos, además de su calidad. Acabado un partido, por ejemplo, el periódico que cuente con las primeras crónicas tendrá más posibilidades de recibir lectores, ya que habrá menos ofertas de crónicas y, por tanto, estará mejor posicionado en los buscadores y será más fácil encontrarlo. Uno de los obstáculos que encuentran los periodistas para generar sus crónicas es la gran cantidad de tiempo a emplear para recopilar todos los datos necesarios y organizarlos de alguna manera legible para su posterior uso.

Parar reducir este problema hemos realizado el desarrollo de una aplicación capaz de extraer información relevante de las temporadas de la liga de fútbol española, almacenarla en una base de datos para consultas posteriores más rápidas y eficientes, y devolver al usuario en un lenguaje legible y propio de un humano la información que desee.

Con nuestra aplicación, el periodista dispondrá de dos versiones de la crónica deportiva sobre, por ejemplo, un partido concreto, lo que le permitirá ahorrar mucho tiempo de escritura y centrar más sus esfuerzos en perfeccionar la crónica inicial que recibe de la aplicación. Esto da una clara ventaja frente a sus competidores a quien lo use, tanto en la velocidad de trabajo como en la riqueza del texto que será capaz de generar, ya que podrá focalizar sus esfuerzos en este aspecto más de lo que podría haberlo hecho de tener que empezar a desarrollar sus publicaciones desde cero.

Cabe destacar el gran potencial de mejora y ampliación que tiene nuestro sistema de cara a un futuro. Con una arquitectura pensada para facilitar el cambio, se podrían hacer más tipos de crónicas, con más detalles y más variantes, consiguiendo que el periodista sólo tenga que revisarlas y cada vez necesite menos trabajo previo a su publicación.

Por otra parte, el desarrollo de este trabajo ha sido la oportunidad perfecta de aplicar los conocimientos adquiridos durante el grado, así como la oportunidad de descubrir y trabajar con tecnologías que se usan muy habitualmente como el web scraping, y otras recién llegadas y en rápido crecimiento como lo son las IAG. El desarrollo de este proyecto nos ha aportado nuevos conocimientos sobre el análisis de datos, programación con IAG, desarrollo web y experiencia gestionando el proyecto.

8.2. Trabajo a futuro

El desarrollo de un proyecto se define en base a sus requisitos y queda limitado por su duración y la disponibilidad de recursos técnicos y humanos. En este sentido, siempre quedan pendientes ideas de mejoras que se podrían implementar para aumentar así las prestaciones de la aplicación.

En el presente proyecto han surgido algunas ideas de mejora no incluidas en la versión actual que se discuten a continuación:

- Automatizar la recogida de datos sobre partidos de la temporada actual de forma que, con el uso de un temporizador, cada semana (ej. lunes a las 10:00 am) se carguen los datos de los partidos que se hayan disputado durante esa jornada.
- Recopilar más datos como pueden ser los fichajes y modificar el input con el que trabaja el modelo para generar las crónicas para enriquecerlas todavía más.
- Expandir el uso de nuestra aplicación a otras competiciones, tanto españolas como europeas en un principio, y con posibilidad de, a la larga, trabajar también con competiciones a nivel mundial.
- Añadir la posibilidad de generar, a parte de las crónicas ya establecidas, nuevos informes. Por ejemplo, informes que expliquen el progreso de un equipo a lo largo de la temporada o que hablen de los jugadores más y menos destacados de una temporada o partido.
- Se podría implementar también algún tipo de predictor para los resultados (o al menos ganador y perdedor) de un partido que esté pendiente de jugarse. De la misma forma, con

los datos disponibles, se podría tratar de analizar la posible evolución de los diferentes equipos y hacer una predicción de la clasificación final de la competición después de cada jornada.

- En caso de contar con los recursos hardware necesarios o de planes de pago que no pongan limitaciones de llamadas a la api Gemma, cambiar el modelo de Gemma que utilizamos actualmente para la fase de recogida/tratamiento/almacenamiento de datos por la api y unificar el proyecto para que toda la aplicación funcione interaccionando de la misma forma con y se pueda ejecutar en el mismo servidor o lugar que se escoja para su despliegue.
- Añadir el siguiente requisito funcional: el usuario "Admin" deberá poder lanzar el script de recopilación, tratamiento y almacenamiento de los datos que usará la aplicación desde la propia interfaz web, este panel de administrador ya está creado para facilitar la implementación de este requisito. (Existen dos tipos de usuarios, los normales que son los que podrán descargar las crónicas desde la aplicación web sin necesidad de iniciar sesión, y el admin que, con previo inicio de sesión, deberá tener acceso a lo anteriormente mencionado).

9. Introduction

9.1. Motivation

Soccer is one of the most popular sports in the world and the Spanish League is one of the most famous, followed and competitive worldwide. Journalists spend a lot of time and effort consulting a large number of sources trying to gather all kinds of information, such as match results or which players are performing the best.

In a society that values the immediacy with which information is obtained, it would be of great help (especially for sports journalism professionals, for example) to have a system that provides the information necessary to create reports and features on demand in a very short time. Developing a system that automates the preparation of reports on historical data of all kinds and that also presents an intuitive and accessible interface that facilitates interaction and use of the system in an efficient and effective way, could greatly reduce the effort required to obtain them. It could have a particularly relevant impact in the field of sports journalism, minimizing production times and thus allowing newspapers whose journalists rely on this tool to help them get ahead of their competitors when publishing their work.

9.2. Objectives

The main objective of this Final Degree Project (Spanish acronym TFG) is to create a system with an intuitive and accessible web interface capable of producing automated reports using information automatically extracted from the Web and Generative Artificial Intelligence (GAI) to produce the text. The specific domain here will be sports information about the first division of the Spanish football league. The application reports will include sports chronicles describing what happened in a specific match, the summary of each matchday of the season or the summary of a complete season. In addition, it will be possible to consult the evolution of a team throughout

the season and who have been the most outstanding players of each season, i.e., scorers, assistants or players with more red and yellow cards.

To achieve this overall objective, the following specific objectives are proposed from the point of view of system development:

- Research of various sources of information of different types about the soccer league seasons to obtain the information from which the reports will be generated. Its extraction will be done by using web scraping and a Large Language Model (LLM). The web scraping will allow to extract information from the web (both structured, e.g. statistics in tables, and unstructured, e.g. text) and the LLM will be used in this phase to obtain events not included in the statistics of the different matches from the texts (news and other existing chronicles). After obtaining all these data, they will be exported to a suitable format to be used in the system.
- The elaboration of the reports by means of a text GAI. The GAI will receive the information it needs for the elaboration of the report requested by the user through a prompt (mechanism to supply inputs). The report returned will contain the story of a match from the events of the match, the description of the league data, the summary of a specific day of the season or the summary of the entire season itself. It will also be possible to obtain the evolution of a team over the course of a whole season.
- Creation of a web interface that allows the end user (e.g. sports journalist) to interact with the system in a simple and intuitive way and obtain the desired reports.

9.3. Work plan

9.3.1. General approach

For the development of this project, we chose Scrum, an agile methodology that we believe is the one that best suits our features and way of working. Scrum is based on doing sprints (iterations) to make progress on an initial functional prototype that gains new features in each of those iterations until getting the final version.

As our idea was to meet with the tutor every two/three weeks, we saw as a viable option to make these meetings at the end of each sprint of the project development.

The bulk of the work was planned from the beginning of the second quarter, since the workload at that time allowed us to dedicate ourselves fully to the project. Nevertheless, in the first term, we advanced in the web scraping part to familiarize ourselves with the data extraction and to design a clear path to follow during the second term.

The meetings with the tutor were adjusted according to the progress made and the availability of the members and the tutor in order to have a continuous monitoring of the project status. In this way, we were able to work constantly and set short and medium-term objectives.

To organize the distribution of tasks within the group, we used a tool widely used in agile methodologies, Trello. Thanks to the use of this tool, all team members were clear about their tasks and those of their colleagues, their past and current status and pending tasks.

In summary, the project was approached as follows. A research process involving a variety of sources would be conducted to gather information on seasons of the Spanish football league. The goal was to have a rich set of data for the application. This information would then be exported to a database to optimize its access and ensure constant availability of data such as match results and relevant statistics. The generation of reports will rely on a text GAI using prompts provided by users to guide the tasks. The reports it would generate would range from detailed descriptions of match events to summaries of match days or complete seasons, as well as the evolution of teams over a season. Finally, an intuitive web interface would be developed so that users could interact directly with the system and obtain the desired reports.

9.3.2. Time planning

In this section we will detail a little more how the work carried out (explained in the previous section) has been divided into the different sprints of the project. This is approximate and the dates indicate the week each meeting took place.

In the following Gantt (see Figura 40) chart you can see the different sprints with which we have worked:

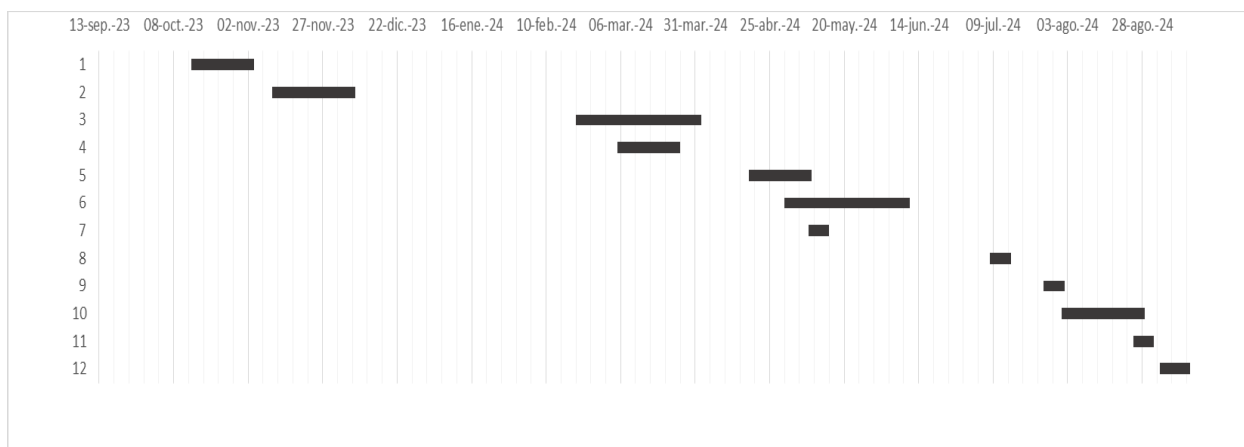


Figura 40. Gantt chart to detail project sprints

In the diagram you can see overlapping sprints since, while some focused on one part of the project and worked on one of the sprints, others were with other aspects of the project and focused on others. In certain sprints we saw this approach as more useful than everyone working on the same thing. An example is that while Alba and Adriana were working on the first tests with the different IAG models, Daniel was with the web interface prototype and Pablo with the introduction. of memory. It should also be noted that, due to previous experience/interests, each of us liked one part of the project more, this also meant that in many moments of the development we worked in parallel (memory, web interface and IAG models/data storage). In this way, we put together the “pieces” of the application, tested its correct operation and corrected the errors that we were not finding before continuing to increase the initial model (in our case this model was the

one that was capable of creating a chronicle of a single match without including extra-sports events).

Although the meetings were held during the course to clarify the ideas, the bulk of the project was left to be developed in the summer, since during the course we all took degree subjects and had company internships. This, along with the exams, consumed most of our time.

Initial meeting - September 13, 2023

Initial meeting to talk about the project idea to be developed and the steps to follow to achieve it.

Sprint 1 - October 14, 2023 (3 weeks)

The objective was to investigate what web scraping was and the tools that exist for it. We select one and begin to familiarize ourselves with it and do tests to discover all its possibilities.

Sprint 2 - November 10, 2023 (4 weeks)

The objective was to be able to extract data on all the matches that had been played during a matchday (e.g. matchday number, teams faced, result, cards received, scorers or substitutions) with the tool that we had already chosen in the previous sprint. At this point, much of the web scraping was finished, so that we had the necessary statistical information to be able to generate chronicles of matches and matchdays in subsequent development.

Sprint 3 - February 20, 2024 (6 weeks)

We did a search for free APIs (Application Programming Interface) to use the IAG in our reporting. We did tests and analyzed several to see which one best suited what we wanted.

We started testing various models and APIs that seemed like we were going to use and we started to get familiar with it. We wanted to be able to generate preliminary match reports. We also analyzed whether we saw it necessary to save information in a database or if we wanted to do everything in real time.

Sprint 4 - March 5, 2024 (3 weeks)

While we continued testing the models and APIs, we decided that it was preferable to use a database to store certain information and avoid relying on obtaining it on demand.

At this point, we addressed unifying a simple web interface, with the database, web scraping and the IAG to be able to carry out the entire workflow of generating a complete chronicle.

At this point, the memory was also beginning.

Sprint 5 - April 18, 2024 (3 weeks)

The database in which we store the match data was already created, we continued testing the models.

At this time these tests were expanded to be able to generate chronicles of full days and summaries of entire seasons, in addition to those of the games that should already work.

Recovery of the second quarter project - February 14, 2024

Once the exams are finished we try to continue moving forward with the project. The objective was to review what was previously done (the web scraping part) and organize ourselves to resume the project.

Likewise, we set up the data extraction part of the project with selenium on a server now that we had the credentials.

The introduction of the memory was completed.

Sprint 6 - April 30, 2024 (6 week)

Have reviewed and corrected the memory with the corrections given by the tutor. In addition, we began to develop what would be the definitive web interface (since before we had a very simple version to do the first interaction tests between the user and the IAG model).

Sprint 7 - May 8, 2024 (1 week)

The memory advanced to explain what we had done so far before starting exams.

Project recovery after second semester exams - June 28, 2024

Once the exams were finished we tried to move forward with what we were missing from the project.

The objective was to review the results obtained from the tests with the models (looking at the results of interactions with different IAGs obtained with the tests in Google Colab) and opt for the one that best answered our queries and used the least resources to be able to execute locally. and not depend on a server.

Sprint 8 – July 8, 2024 (3 weeks)

In addition to the statistical data, we wanted to collect other non-statistical data that would enrich the chronicles generated, for which we used selenium and stored those data in the database that we had already created.

We continue testing models with the aim of being able to generate sports chronicles with these non-sports events collected.

Sprint 9 – July 26, 2024 (1 week)

Updated report with new additions to the project.

Sprint 10 – August 1, 2024 (4 weeks)

Final choice of the Gemma API for the generation of match, matchday and season reports.

Implementation of the connection between the website and the chronicle generation script and completion of the website so that it works correctly. [OBJ]

Sprint 11 – August 25, 2024 (1 week)

Finalization of the memory with the new updates. Delivery of the project draft and report with the first corrections already implemented.

Sprint 12 – September 3, 2024 (10 days)

The objective is to have implemented all the improvements given after the partial delivery of the project and the report and to make the final delivery of the final degree project.

10. Conclusions and future work

This chapter presents the conclusions of the work carried out and describes aspects of the application that could be improved and expanded in the future.

10.1. Conclusions

One of the key aspects of the journalists' work is the speed with which they publish their works, as well as their quality. After a match, for example, the newspaper that has the first reports will have more chances of receiving readers, since there will be few offers of reports about it and, therefore, it will be better positioned in the search engines results. The main problem for journalists here is the amount of time they have to spend to collect all the data and to organize it in a readable way for its later use in reports or for their readers.

To address this issue, we have developed an application able to extract relevant information about the seasons of the Spanish football league, store it in a database for faster and more efficient queries, and provide the user in the desired information based on those data in a human readable text language.

With our application, journalists will have two versions of a sports report on a specific match, competition day or season with a couple of clicks. This will allow them to save a lot of writing time and focus more on improving the initial report they receive from the application. This gives a clear advantage over a more manual approach both in the speed of work and in the richness of the text journalists, since they will be able to focus on these aspects more than they could do if they had to start developing their reports from scratch.

It is worth mentioning the great potential for improvement and expansion that our system has for the future. With an architecture designed to facilitate change, more types of reports could be made, with more details and more variants, making possible that journalists only would have to review them.

On the other hand, the development of this work has been the perfect opportunity for us to apply the knowledge acquired during the degree, as well as the opportunity to discover and work with technologies that are widely used, such as web scraping, and other new and fast-moving technologies. growth such as GAI. Thus, the development of this project has provided us with new knowledge about data analysis, GAI programming, web development and experience in project management.

10.2. Future Work

The development of a project is defined on the basis of its requirements and limited by its duration and the availability of technical and human resources. In this sense, there are always pending ideas for improvements. Some of them not included in the current version are discussed next:

- Automate the collection of data about matches of the current season so that, with the use of a timer, every week (e.g. Monday at 10:00 am) the data of the matches that have been played during the previous week are collected.
- Collect more data such as player transfers.
- Expand the use of our application to other competitions, both Spanish and European at first, and with the possibility of eventually also working with competitions worldwide.
- Add the possibility of generating new reports. For instance, reports that explain the progress of a team throughout the season or about the most and least outstanding players of a season or match.
- The system could also implement some kind of prediction about results (or at least winner and loser) of matches not yet played. In the same way, with the available data, we could try to analyze the possible evolution of the different teams and make a prediction of the final classification of the competition after each matchday.

Apéndice A - Despliegue del sistema

Introducción al despliegue

La idea es conseguir el mayor grado de automatización de todo el proceso necesario para generar las crónicas deportivas.

Para hacerlo, como ya explicamos en el apartado de ["Arquitectura del proyecto"](#), contamos con scripts que ejecutan y automatizan las principales funciones de la aplicación (recabar datos, trabajarlos, almacenarlos, lanzar la interfaz web y generar crónicas).

El despliegue se divide en 5 partes:

- Instalación de herramientas y librerías necesarias para el correcto funcionamiento del proyecto.
- Preparación de la base de datos, necesaria para el almacenamiento de los datos que usará la propia aplicación.
- Archivos de configuración. Tanto de parámetros que usará la aplicación web, como el modelo de IAG, como el script de obtención/tratamiento/almacenamiento de datos.
- Proceso de obtención, tratamiento y almacenamiento de datos para recopilar todos los datos que necesitará la aplicación para funcionar.
- Despliegue de la aplicación web. La interfaz es una parte esencial para que el usuario obtenga las crónicas que desee interaccionando con el modelo de IAG de una forma intuitiva.

Instalación de herramientas

Desde la máquina en la que vamos a querer lanzar la aplicación deberemos abrir una consola de comandos. Procederemos a continuación a instalar todas las herramientas y librerías que vamos a necesitar para el despliegue del proyecto.

Esto lo hacemos con un archivo denominado *requirements*.

Preparación de la base de datos

Como ya hemos explicado en capítulos anteriores, nuestra base de datos se encuentra en Mongo Atlas, que proporciona una instancia de base de datos de Mongo en la nube.

Como el proyecto está preparado para utilizar esta base de datos, lo único que tendremos que hacer en este paso del despliegue es ir a la página de Cloud Mongo DB²⁴ y crearnos una cuenta para poder usar Mongo Atlas.

Nos crearemos un clúster. Cluster0 es el que viene por defecto y es gratuito (dejando la configuración por defecto que nos trae) y, después, nos crearemos una base de datos (nosotros la llamamos TFG).

Además de los pasos ya mencionados, será necesario tener localizado nuestro token de acceso a la base de datos, imprescindible para poder trabajar con ella, como se explicará en la sección de [Archivos de configuración](#) y permitir el acceso a nuestra base de datos desde cualquier IP.

²⁴ <https://www.mongodb.com/products/platform/cloud>

Archivos de configuración

Configuración archivo de configuración del script de obtención/tratamiento/almacenamiento de datos

Para facilitar el proceso de automatización del proyecto, trabajamos con un archivo de configuración que nos permite detallar con qué temporadas vamos a querer trabajar y, por tanto, de cuales de ellas vamos a tener que almacenar todos los datos en una primera ejecución.

Este archivo es "config.json", tiene tres variables. Las dos primeras son la temporada inicial y final con las que queremos trabajar. En el caso de ejemplo (ver Figura 41) trataremos con las temporadas que empiezan en los años 2021, 2022 y 2023. La última variable llamada "scripts" detalla los scripts que vamos a querer ejecutar desde el "main.py". De nuevo en el ejemplo, los tres scripts indicados se encargan de recoger y almacenar información sobre los cruces, las estadísticas de los equipos a lo largo de las diferentes temporadas y los datos de cada uno de los partidos disputados en liga durante esas temporadas.

A no ser que se desarrollen más scripts para recopilar nueva información (que se añadirían a la lista de scripts), lo único que deberá modificar el administrador son las temporadas que quiere estudiar.

```
{
  "temporada_inicial": 2021,
  "temporada_final": 2023,
  "scripts": ["guardar_cruces.py", "guardar_estadisticas Equipos.py", "guardar_datos_partidos.py"]
}
```

Figura 41. Contenido del archivo config.json

Contamos también con un .env al igual que en la parte de la aplicación web (en el siguiente apartado se explica más en detalle en qué consiste su uso), esto es para facilitar el mantenimiento del código:

```
##Configuracion base de datos
DB= 'nombre_base_datos'
MONGODB_URI= "mongodb_uri"
CRUCES= 'coleccion_cruces'
ESTADISTICAS_EQUIPO= 'coleccion_estadisticas'
DATOS_PARTIDOS= 'coleccion_partidos'

##Configuracion para leer y escribir los eventos extraidos por la IA
PATH_SENTENCES= 'path_sentences.json'

##Llamada a ia para extraer eventos
INPUT_NOTEBOOK = 'path_modelo.ipynb'
OUTPUT_NOTEBOOK = 'path_modelo_salida.ipynb'

##IA
TOKEN= "tu_token_modelo"
```

Figura 42. Contenido del fichero .env recopilación de datos

Configuración archivo de configuración para la funcionalidad de la aplicación web.

En el desarrollo de aplicaciones modernas es común la necesidad de gestionar diferentes configuraciones y credenciales que varían entre entornos de trabajo, como desarrollo, pruebas y producción. Para abordar esta problemática, se utilizan archivos de configuración denominados .env (abreviatura de "environment" o "entorno").

Un archivo .env permite almacenar de manera segura y accesible las variables de entorno que la aplicación necesita para funcionar. Estas variables pueden incluir datos sensibles, como claves API, credenciales de bases de datos, tokens de autenticación, y otros parámetros de configuración específicos del entorno. La utilización de un archivo .env contribuye a mejorar la seguridad, la flexibilidad y la mantenibilidad de la aplicación.

En nuestro caso utilizamos el fichero .env para guardar variables utilizadas para la gestión de la IAG y de Django como se indica en la siguiente figura (ver Figura 43).

```
##Modelo IA
CLIENT= "API utilizada para la ia"
# Parametros ia
MAX_TOKENS=1024
TEMPERATURE=0.6
TOP_P=0.9
TOP_K=50
REPETITION_PENALTY=1.2

# Configuración de la base de datos
KEY="Clave secreta django"
DB_ENGINE='tipo de base de datos'
DB_NAME='nombre base de datos'
DB_ENFORCE_SCHEMA=false
DB_HOST='enlace conexion base de datos'
DB_AUTH_MECHANISM='tipo de encriptacion de datos'
DB_AUTH_SOURCE='autoridad para el manejo de datos'

# Configuración de idioma y zona horaria
LANGUAGE_CODE='en-us'
TIME_ZONE='UTC'
```

Figura 43. Contenido del fichero .env de la parte app web

Proceso de obtención, tratamiento y almacenamiento de datos

Para lanzar esta parte del proyecto ver apartado "[Manual de uso: Administrador](#)". En él se explica cómo y por qué se debe ejecutar esta fase de recopilación de datos desde un servidor.

En un futuro, desde la interfaz, el usuario tendrá la opción de iniciar sesión como *admin*, que es el único usuario que podrá ejecutar el script necesario para que la fase de obtención, tratamiento y almacenamiento de datos se complete correctamente.

Para ello, una vez en el panel de control de usuario podrá ejecutar el *main.py*, script que automatiza el proceso de recogida de datos, su tratamiento y su almacenamiento en la base de

datos. Tras este paso ya contaremos con toda la información necesaria para que nuestra web trabaje.

Cabe destacar que este proceso se realizará una única vez, la primera que se vaya a utilizar la aplicación. Esto es porque está construida de forma que cargue toda la información necesaria de todas las crónicas que estén detalladas en el archivo de configuración “config.json”, esto puede requerir de muchos recursos hardware y, por tanto, de mucho tiempo de ejecución.

Es importante entender que, aunque la primera vez tarde horas en completarse este proceso, lo provoca la enorme cantidad de datos que se manejan. Por eso mismo contamos con una base de datos en la que almacenar todos los partidos que ya se haya pedido recopilar.

La idea para un futuro sería que, una vez se hayan cargado todos los datos de partidos ya finalizados que se deseen, exista un script que semanalmente (o diariamente, como se prefiera), se encargue de actualizar la base de datos con la información de los nuevos encuentros disputados. Este proceso, al trabajar con muchos menos datos que temporadas al completo, tardará menos tiempo e incluso se podrá programar para que lo haga automáticamente cada lunes a las 4 de la mañana, por poner un ejemplo (esto está detallado en el capítulo de “[Trabajo a futuro](#)”).

Despliegue de la aplicación web

Con todos los datos ya cargados en la base de datos, lo siguiente será el lanzamiento de la interfaz web, en este punto es cuando el usuario ya podrá acceder a la URL para solicitar las crónicas que desee.

Para ello lo único que debemos hacer es ejecutar el siguiente comando desde una terminal en el equipo que queremos que lance la aplicación, ya que es una ejecución en local:

```
>>python manage.py runserver
```

Apéndice B - Contribuciones personales

Adriana Fernández de Matos

Al principio del desarrollo del proyecto todos colaboramos a la hora de investigar y escoger las herramientas que mejor se adaptaban a nuestro TFG. Decidimos usar Selenium para web scraping y las webs que íbamos a estudiar.

Decidimos qué tareas había que completar y con qué prioridades y las organizamos en un panel de Trello para llevar un control de los sprints.

Una vez tuvimos la información necesaria recolectada gracias a los scripts de web scraping, nos pusimos a desarrollar la forma de almacenar parte de esa información más relevante en una base de datos. La elegida fue Mongo DB en la nube, en Mongo Atlas.

Para este proceso trabajamos en scripts que filtrasen y ordenasen la información para que se almacenase de la forma más eficiente posible. Una vez conseguimos que la fase de recogida, tratamiento y almacenamiento de datos diesen los resultados esperados en un entorno de pruebas como es el Colab, mientras Daniel avanzaba con la parte web del proyecto, me dediqué a acabar de afinar su funcionalidad puliendo algún detalle que no acababa de funcionar como queríamos, unificar estos procesos y automatizar su ejecución por medio de un solo archivo .py.

Junto a Alba y Daniel, nos pusimos a probar diferentes modelos de IA generativa para la parte de generación de crónicas del proyecto. Con esto nos pasamos gran parte del tiempo de desarrollo ya que, por falta de recursos hardware, outputs incoherentes, intentos de entrenar modelos para afinar su funcionamiento fallidos por falta de recursos o insuficientes... nos resultó muy difícil encontrar un modelo ligero con el que poder trabajar.

Hicimos pruebas con modelos de Hugging Face como gpt2, Gemma, Mixtral o T5, entre otros. Al ver que con un prompt estándar en el que le solicitábamos una crónica no daba los

resultados esperados, intentamos entrenar al modelo en vez de usarlo sin entrenamiento. Este proceso de pruebas de entrenamiento llevó también varias semanas y, por los recursos de hardware que necesitaba el modelo, decidimos descartarlo y buscar otras alternativas.

Al conseguir los primeros resultados positivos con Mixtral, Daniel se centró más en la parte web, Pablo en la memoria y nosotras en la creación de scripts para construir el input que le pasaremos al modelo según la interacción que haga el usuario en la web.

Para enriquecer la generación de crónicas de los partidos buscamos la forma de obtener eventos extradeportivos relacionados con los mismos. De esta forma, modificamos los scripts de Selenium para que recogiesen las crónicas de partidos ya jugados en el diario as y modificamos los scripts de almacenamiento en Mongo para guardar esta información también. De la misma manera, modificamos los scripts para generar el input que recibirá el modelo para generar el resumen de un partido, para esta parte utilizamos la api de Hugging Face que interacciona con el modelo de Gemma, Daniel descubrió esta opción y resultó ser mucho más ligera y, por tanto, facilitó el despliegue de la interfaz web y de la interacción con el modelo al hacerlo todo en local.

Una vez automatizada toda la parte de recogida de información en el servidor y con la parte de la web ya casi finalizada, mi trabajo estuvo centrado en la elaboración y corrección de la memoria.

Alba Gorrindo Martínez

Al igual que el resto de los compañeros para comenzar empecé a familiarizarme con Selenium, herramienta para automatizar la extracción de datos con los que trabajar en la siguiente fase del proyecto. Además de ir haciendo pruebas con Selenium, iba investigando diferentes páginas para ver qué datos aportaba cada una y si había gran diferencia entre una y otra web. Cada uno hizo su script y de ahí llegamos a una versión conjunta en la que se refactorizó lo que había y se eligieron algunas de las webs exploradas para que la base de datos fuera lo más completa posible pero que no hubiera datos duplicados.

Tras disponer de los datos, se transitó a la parte de generación de texto con IAG. A esta parte se le dedicó bastante tiempo explorando las diferentes posibilidades.

Primero se miró la api de CHAT-GPT, pero requería de pago, por lo que se descartó y se pasó a mirar diferentes modelos de Hugging Face en paralelo con Adriana y Daniel.

Comenzamos a experimentar con GPT-2 para nuestro proyecto, pero nos encontramos con resultados inesperados y vimos que planteaban desafíos técnicos que superaban nuestro nivel de conocimiento y requerían de un tiempo que no disponíamos para llegar a objetivo deseado. A pesar de los contratiempos, esta experiencia nos ayudó a conocer en profundidad el proceso de un LLM.

Al ver que esto no daba resultado, continuó la investigación llegando a hacer un curso de Udemy de Inteligencia Artificial Generativa para ver si con lo que en este enseñaban conseguíamos el objetivo. Tras el intento con GPT2, probamos con T5. Lo malo de este era que necesitábamos reentrenarlo con lo que queríamos que hiciera. Al principio probamos a coger con Gnews y/o Google search distintas crónicas disponibles en Internet para después guardarlas, pero no fue posible ya que con esta herramienta y la estructuración de las webs no era posible capturar bien las crónicas. Tras eso, montamos un script que, usando Selenium, obtenía las crónicas del periódico as para darle ejemplos al modelo de lo que queríamos. Nos dimos cuenta de que hacer este proceso de entrenamiento requería mucho tiempo del que no disponíamos, por lo que mientras las pruebas continuaban seguía la exploración de los modelos de Hugging

Face. No obstante, fue enriquecedor, ya que permitió conocer otro tipo de IA que nunca había visto.

Descubrimos Mixtral. Al principio fue un problema porque requería de mucha RAM ya que era un LLM muy pesado, pero se llegó a Google Colab que permitió que el modelo se ejecutara sin problema, y tras una serie de pruebas en pequeña escala vi que hacía lo que buscábamos. Se puso en común con los compañeros y comenzamos a hacer pruebas con esta herramienta y ver las posibilidades que presentaba.

Teníamos gran parte resuelta, por lo que mientras otros compañeros iban adelantando con la memoria y la web, Adriana y yo nos dedicamos a perfeccionar el script y a hacer mejoras como el pasar los datos a una base de datos para que no hubiera que recoger los datos para cada consulta y hacer más eficiente el proyecto.

Mientras se iba haciendo la plataforma web, se intentó ampliar el contenido que esta iba a disponer haciendo también crónicas además de un partido, que hubiera también crónicas de una jornada y de una temporada.

Tras disponer de lo anterior, se quiso ampliar y recopilar también datos no estadísticos con procesamiento de lenguaje natural. Sin embargo, con un poco de información vimos que no era factible pero que podíamos usar Mixtral para que nos extrajese de crónicas ya publicadas en otros medios esos datos. Tras una reestructuración al código que ya se había hecho para usar como entrenamiento de T5 con las crónicas del as conseguimos estos datos y los añadimos a la base de datos.

Dado que Mixtral requería de muchos recursos, seguimos investigando distintas opciones para llegar a ejecutar la aplicación web en local. Llegamos a la API de Gemma en hugging face que permitió ejecutar el modelo en local por lo que nos quedamos con esta para el desarrollo del proyecto.

Por último, integramos el modelo en la aplicación en web e incluimos las últimas modificaciones en la memoria.

Daniel Cano Sevilla

Primero, realicé una investigación en la web oficial de la Liga para recopilar datos relevantes para nuestro proyecto. Para llevar a cabo esta tarea de manera eficiente, utilicé herramientas avanzadas como Selenium, que me permitieron automatizar el proceso de extracción de datos.

Desarrollé un script en Python denominado "laligaoficial.py", diseñado específicamente para extraer información detallada sobre los partidos, incluyendo datos como goles, tarjetas y resultados. Implementé el script de manera que la información obtenida se organizara de forma clara y estructurada, facilitando así su posterior análisis y utilización en el proyecto.

Mediante esta metodología, logré recopilar una amplia gama de datos relevantes directamente desde la fuente oficial, lo que garantizó la precisión y fiabilidad de la información. Este enfoque me proporcionó una base sólida de datos para el proyecto y sentó las bases para el análisis y desarrollo posteriores.

Luego, colaboré brevemente con Alba y Adriana en la investigación de IAG proporcionadas por Hugging Face. Nos enfocamos específicamente en experimentar con GPT-2 para evaluar su idoneidad para nuestras necesidades en el proyecto. A pesar de los desafíos técnicos y las limitaciones en la adaptación de esta tecnología a nuestro contexto específico, esta experiencia me brindó conocimientos valiosos sobre las capacidades y limitaciones de las IAG disponibles, lo que me permitió tomar decisiones fundamentadas para avanzar en el proyecto.

Una vez que obtuvimos resultados satisfactorios con Mixtral, asumí la responsabilidad del desarrollo de la plataforma web utilizando Django y HTML. Mi objetivo era crear una interfaz intuitiva y atractiva que permitiera mostrar de manera efectiva las crónicas generadas por Mixtral.

Busqué realizar una página web para mostrar las crónicas de manera clara y legible, para el uso posterior de periodistas deportivos, asegurando que la información se presentara de manera ordenada y coherente.

Dado que me encargué del desarrollo de la página web, también asumí parte del Capítulo 6 de esta memoria, explicando por qué utilizamos Django y HTML y el porqué de las estructuras

y clases utilizadas. Investigué en las páginas oficiales de Django para esta tarea, aprendiendo la funcionalidad de Django, la web y cómo su patrón encajaba a la perfección con el proyecto.

Intentamos subir la parte de Mixtral al servidor con éxito esta vez. Sin embargo, al intentar establecer un túnel utilizando Gunicorn y Nginx para conectar Mixtral con la interfaz web, descubrimos que el dominio de la web no funcionaba correctamente, lo que provocó que la comunicación entre el servidor SSH y la web no pudiera establecerse. A pesar de muchos intentos, no tuvimos éxito en resolver este problema.

Por ello finalmente seguí investigando distintas páginas relacionadas con IAG, en concreto la de Hugging Faces hasta que descubrí la manera de conectar la IAG con la Web Django, esta solución fue utilizar una API ubicada en la página web de Hugging Faces, la cual nos permitía interactuar con ella consumiendo sus recursos y permitiéndonos generar e imprimir los resultados en la Web sin muchos tiempos de espera y con poco consumo de recursos locales.

Pablo Regidor García

En primera instancia, me encargué de investigar diferentes sitios web de los que extraer los datos a utilizar en el proyecto, como lo son los cruces de una temporada, los jugadores que más han destacado por su rendimiento o la clasificación en una temporada, y de investigar distintas herramientas de web scraping para la extracción eficiente de dichos datos. Tras encontrar una página web de la que sacar bastante información, <https://www.resultados-futbol.com/>, y elegir Selenium como herramienta de web scraping me dispuse a desarrollar varios scripts con los que extraer los datos.

Tras un tiempo probando varios enfoques y depurando código, logré desarrollar los scripts, `cruces.py`, `estadísticas_por_equipo.py` y `jugadores_destacados.py`. Además, conseguí hacer que los scripts obtuvieran la información de las temporadas pasadas por parámetros, e incluso pensando ya en el trabajo a futuro, los datos de todas las temporadas disponibles en la web. En una primera iteración del web scraping, estos scripts guardaban los datos en un txt para poder ver de manera legible si los datos eran coherentes.

Cuando ya se estaba en la fase de guardar los datos en la base de datos y pasarlo al modelo de IAG, nos dimos cuenta de que había que hacer un trabajo excesivo e inútil procesando los datos desde un txt y luego pasarlos a formato json o csv. Por eso me encargué de realizar una segunda iteración del web scraping, modificando los tres scripts mencionados anteriormente para que los datos extraídos se guardaran directamente en un archivo csv en vez de txt. Asimismo, me dispuse a probar la inserción correcta de los datos en la base de datos, realizando pequeñas modificaciones en los scripts, que ya hacían la inserción en la base de datos y transferían los datos al modelo de IAG, realizados por Adriana.

Una vez los scripts funcionaban perfectamente, Adriana y yo decidimos limpiar los scripts que realizaban el web scraping y aquellos que realizaban todo el procesamiento de los datos, eliminando código antiguo, añadiendo comentarios para la legibilidad del código y eliminando también aquellos que no aportaban nada a dicha legibilidad.

Mientras el resto del equipo se centraba en conectar el modelo de IAG y la aplicación web, me dispuse a empezar la memoria. Empecé redactando el resumen así, como la lista de

palabras clave del TFG, que tras tener una versión casi definitiva tradujo al inglés. Tras haber completado el resumen y su traducción, a falta de las posibles correcciones del tutor, elaboré y tradujo las secciones de motivación y objetivos del capítulo 1, el correspondiente a la introducción, y empecé la sección del plan de trabajo, que más tarde terminaría Adriana.

Para seguir con el desarrollo de la memoria me encargué de realizar el capítulo 2 en su totalidad. Para ello tuve que realizar una profunda investigación de aplicaciones o páginas web cuya finalidad era semejante a la del TFG. Tras dos, tres días de investigación encontré cuatro herramientas, Power BI, BeSoccer Pro, Tagg-maker y Anafut, que generaban informes o resúmenes relacionados con las ligas de fútbol y sus jugadores. Redacté la descripción de cada una de las herramientas, además de pegar pruebas descriptivas acerca de lo que eran capaces dichas herramientas. Además, me encargué de formular los requisitos funcionales y no funcionales de la aplicación, descritos en el capítulo 3 de la memoria.

En las etapas finales del proyecto a falta de la puesta a punto de la aplicación web junto con el modelo de IAG, dediqué gran parte de su esfuerzo a realizar las correcciones pertinentes y sugerencias realizadas por el tutor del TFG, las secciones en las que formé parte, así como en cuestiones de formato, puntualización y referencias en varias de las figuras incluidas en la memoria.

APÉNDICES

Apéndice C - Glosario

TFG: Trabajo Fin de Grado

API: Application programming interface

PLN: Procesamiento de lenguaje natural

JSON: JavaScript Object Notation

URL: Uniform Source Location

PDF: Portable Document Format

CSV: Comma Separated Values

IAG: Inteligencia Artificial Generativa

IA: Inteligencia Artificial

LLM: Large Language Model

SQL: Structured Query Language

HTML: Hypertext Markup Language

CSS: Cascading Style Sheets

HTTP: Hypertext Transfer Protocol

GPT: Generative Pre-trained Transformer

DOM: Document Object Model


Bibliografía

¿Qué es Business Intelligence o inteligencia de negocios? (s.f.). Obtenido de tableau:
<https://www.tableau.com/es-es/learn/articles/business-intelligence>

Acerca de GitHub y Git - documentación de GitHub. (s.f.). Obtenido de GitHub Docs:
<https://docs.github.com/es/get-started/start-your-journey/about-github-and-git>

Agostinelli, A., Denk, T., Borsos, Z., Engel, J., Verzetti, M., Caillon, A., . . . Frank, C. (26 de enero de 2023). *MusicLM: Generating Music from text*. Obtenido de arXiv.org:
<https://arxiv.org/abs/2301.11325>

Amazon Web Services, Inc. (s.f.). Obtenido de ¿Qué es la IA generativa? - Explicación de la inteligencia artificial generativa - AWS: <https://aws.amazon.com/es/what-is/generative-ai/>

Analista, O. (18 de octubre de 2023). *7 Casos de Uso de Power BI en Fútbol Profesional [Ejemplos Reales]* . Obtenido de YouTube:
<https://www.youtube.com/watch?v=bkG4K4hwxB4>

Beautiful Soup Documentation — Beautiful Soup 4.12.0 documentation. (s.f.). Obtenido de <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

beautifulsoup. (s.f.). Obtenido de Aprende Python:
<https://aprendepython.es/pypi/scraping/beautifulsoup/>

Besoccer.com. (s.f.). *La herramienta definitiva de análisis de datos y scouting - BeSoccerPro*. Obtenido de <https://pro.besoccer.com/>

Chernandez. (18 de octubre de 2023). *¿Cuál es el mejor modelo IA generativa? - OpenSistemas*. Obtenido de OpenSistemas: <https://opensistemas.com/cual-es-el-mejor-modelo-de-ia-generativa/>

Chernandez. (18 de Octubre de 2023). *OpenSistemas*. Obtenido de ¿Cuál es el mejor modelo IA generativa? - OpenSistemas: <https://opensistemas.com/cual-es-el-mejor-modelo-de-ia-generativa/>

CSS | MDN. (6 de julio de 2024). Obtenido de MDN Web Docs: <https://developer.mozilla.org/es/docs/Web/CSS>

Davidiseminger. (22 de marzo de 2024). *¿Qué es Power BI? - Power BI*. Obtenido de Microsoft Learn: <https://learn.microsoft.com/es-es/power-bi/fundamentals/power-bi-overview>

de la Torre, J. (Febrero de 2023). *Autocodificadores Variacionales (VAE) Fundamentos Teóricos y Aplicaciones*. Obtenido de ResearchGate: https://www.researchgate.net/publication/368664523_Autocodificadores_Variacionales_VAE_Fundamentos_Te'oricos_y_Aplicaciones

de la Torre, J. (18 de Febrero de 2023). *Redes Generativas Adversarias (GAN) Fundamentos Teóricos y Aplicaciones*. Obtenido de arxiv.org: <https://arxiv.org/abs/2302.09346>

de la Torre, J. (18 de Febrero de 2023). *Transformadores: Fundamentos teoricos y Aplicaciones*. Obtenido de arxiv.org: <https://arxiv.org/abs/2302.09327>

Django documentation | Django documentation. (s.f.). Obtenido de Django Project: <https://docs.djangoproject.com/en/5.0/>

Equipo editorial de IONOS. (22 de febrero de 2023). *Web scraping con Python: introducción y tutorial*. Obtenido de IONOS Digital Guide: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/web-scraping-con-python/>

Gazala, S. (19 de diciembre de 2023). *Selenium WebDriver: Detail Guide with use cases*. Obtenido de <https://www.lambdatest.com/learning-hub/webdriver>

Google Colab. (s.f.). *Colab.google*. Obtenido de colab.google: [https://colab.google/](https://colab.google/Home)

Home. (s.f.). Obtenido de Tagg-maker: <https://tagg-maker.com/#caracteristicas>

HTML: Lenguaje de etiquetas de hipertexto | MDN. (28 de julio de 2024). Obtenido de MDN Web Docs: <https://developer.mozilla.org/es/docs/Web/HTML>

IBM. (22 de Marzo de 2024). Obtenido de ¿Qué es la IA generativa?

JSON. (s.f.). Obtenido de <https://www.json.org/json-es.html>

mistralai/Mixtral-8x7B-Instruct-v0.1 · *Hugging Face*. (s.f.). Obtenido de <https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1>

MongoDB. (s.f.). ¿Qué es MongoDB? Obtenido de MongoDB: <https://www.mongodb.com/es/company/what-is-mongodb>

MonGoDB Atlas Database | Multi-Cloud Database Service. (s.f.). Obtenido de MongoDB: <https://www.mongodb.com/products/platform/atlas-database>

Qué es Trello: descubre sus funciones, usos y todo lo que ofrece | Trello. (s.f.). Obtenido de <https://trello.com/es/tour>

requests. (29 de Mayo de 2024). Obtenido de PyPI: <https://pypi.org/project/requests/>

Scrapy | A Fast and Powerful Scraping and Web Crawling Framework. (s.f.). Obtenido de <https://scrapy.org/>

Scrum.org. (s.f.). Obtenido de <https://www.scrum.org/learning-series/what-is-scrum/>

Selenium. (s.f.). Obtenido de Selenium: <https://www.selenium.dev/>

Spyder (S.) Team. (s.f.). *Home — Spyder IDE*. Obtenido de © 2023 Spyder Website Contributors: <https://www.spyder-ide.org/>

tableau. (s.f.). ¿Qué es Business Intelligence o inteligencia de negocios? Obtenido de tableau: <https://www.tableau.com/es-es/learn/articles/business-intelligence>

Talin, B. (11 de Julio de 2023). *¿Qué es la IA Generativa? – Todo lo que necesitas saber: significado, modelos y ejemplos.* Obtenido de morethandigital: https://morethandigital.info/es/que-es-la-ia-generativa-significado-modelos-ia-y-ejemplos/#%C2%BFQue_es_la_IA_Generativa

Titania Cía. Editorial, S.L. (s.f.). *El Confidencial*. Obtenido de Ana Futbot. El Cartagena y el Villanovense empatan (1-1): <https://www.elconfidencial.com/autores/ana-futbot-1824/>

Torrijos, J., & Bran, C. (1 de Diciembre de 2019). Automated sports journalism. The AnaFut case study, the bot developed by El Confidencial for writing football match reports. *Doxa. Comunicación/Doxa. Comunicación.* doi:10.31921/doxacom.n29a12

Visual Studio Code - Code editing. Redefined. (3 de noviembre de 2021). Obtenido de <https://code.visualstudio.com/>

Welcome to Python.org. (29 de agosto de 2024). Obtenido de Python.org: <https://www.python.org/>

🤖 *Transformers.* (s.f.). Obtenido de <https://huggingface.co/docs/transformers/es/index>