



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

F1nalLap



Presentado por Alberto García Alcolado
en Universidad de Burgos — 11 de junio
de 2025

Tutor: César Represa Pérez

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	iv
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Estudio de viabilidad	8
Apéndice B Especificación de Requisitos	11
B.1. Introducción	11
B.2. Objetivos generales	11
B.3. Catálogo de requisitos	12
B.4. Especificación de requisitos	14
Apéndice C Especificación de diseño	21
C.1. Introducción	21
C.2. Diseño de datos	21
C.3. Diseño arquitectónico	22
C.4. Diseño procedimental	23
Apéndice D Documentación técnica de programación	27
D.1. Introducción	27
D.2. Estructura de directorios	27
D.3. Manual del programador	28

Apéndice E Documentación de usuario	33
E.1. Introducción	33
E.2. Requisitos de usuarios	33
E.3. Instalación	34
E.4. Manual del usuario	34
Apéndice F Anexo de sostenibilización curricular	47
F.1. Introducción	47
F.2. Conclusión	49
Bibliografía	51

Índice de figuras

E.1. Vista principal.	35
E.2. Barra de navegación.	36
E.3. Barra de navegación en móvil.	36
E.4. Pie de página.	36
E.5. Selección de circuitos.	37
E.6. Selección de estrategia.	37
E.7. Resultado de la estrategia.	38
E.8. Selección de carrera.	39
E.9. Resultado de carrera seleccionada.	39
E.10. Selección de temporada.	40
E.11. Carreras de temporada seleccionada.	40
E.12. Resultado de la carrera seleccionada.	41
E.13. Tabla de clasificación de pilotos.	42
E.14. Tabla de clasificación de equipos.	42
E.15. Selección de temporada.	43
E.16. Ranking de pilotos de la temporada seleccionada.	43
E.17. Ranking de equipos de la temporada seleccionada.	44
E.18. Comparación de victorias de pilotos.	44
E.19. Comparación de puntos de pilotos.	45
E.20. Comparación de puntos por carrera de pilotos.	45
E.21. Comparación de victorias de equipos.	45
E.22. Comparación de puntos de equipos.	46
E.23. Comparación de puntos por carrera de equipos.	46
E.24. Método de contacto.	46

Índice de tablas

A.1. Tareas y objetivos del Sprint 1 (20/04/2025 - 26/04/2025)	3
A.2. Tareas y objetivos del Sprint 2 (27/04/2025 - 03/05/2025)	4
A.3. Tareas y objetivos del Sprint 3 (04/05/2025 - 10/05/2025)	5
A.4. Tareas y objetivos del Sprint 4 (11/05/2025 - 17/05/2025)	6
A.5. Tareas y objetivos del Sprint 5 (18/05/2025 - 24/05/2025)	6
A.6. Tareas y objetivos del Sprint 6 (25/05/2025 - 31/05/2025)	7
A.7. Tareas y objetivos del Sprint 7 (01/06/2025 - 07/06/2025)	8
B.1. CU-1 Consultar resultados de la carrera.	14
B.2. CU-2 Ver clasificaciones de pilotos y equipos.	15
B.3. CU-3 Simular estrategia de carrera.	16
B.4. CU-4 Análisis de pilotos y equipos.	17
B.5. CU-5 Consultar calendario de carreras.	18
B.6. CU-6 Ver noticias y actualizaciones.	19

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apéndice de los anexos, plan de proyecto software, se presenta la planificación del proyecto F1nalLap [1], detallando tanto la planificación temporal como el estudio de viabilidad económica y legal.

El objetivo de este anexo es ofrecer una visión general detallada sobre cómo se ha gestionado, ejecutado y controlado el proyecto a lo largo de su desarrollo. A través de este documento, se describen las fases de planificación y las decisiones clave tomadas, permitiendo una comprensión clara de la estructura y los objetivos alcanzados en cada etapa del proceso.

A.2. Planificación temporal

El proyecto F1nalLap ha sido gestionado mediante una metodología ágil, específicamente SCRUM, que ha permitido organizar el desarrollo de manera flexible y orientada a resultados. Esta metodología se caracteriza por su enfoque iterativo, donde el trabajo se divide en ciclos cortos o sprints, cada uno de aproximadamente una semana.

Cada sprint ha tenido como objetivo principal entregar funcionalidades completas y operativas de la aplicación, asegurando que se puedan evaluar y ajustar en función de los avances y las necesidades emergentes. La estructura de sprints no solo facilita el seguimiento continuo del progreso, sino que también permite priorizar tareas y adaptarse rápidamente a cualquier cambio, optimizando así la calidad final del proyecto.

Para garantizar un rumbo claro y un adecuado seguimiento de las tareas realizadas, se ha utilizado Trello, una plataforma de gestión visual de tareas. Esta herramienta ha permitido coordinar y monitorear el avance de cada sprint de forma eficiente, asegurando que las tareas se realicen en el tiempo estipulado y que el progreso del proyecto sea constantemente evaluado.

A continuación, se detallan las tareas y objetivos asignados a cada uno de los sprints, los cuales están diseñados para alcanzar un avance incremental y continuo.

Sprint 1 (20/04/2025 - 26/04/2025)

Configuración inicial y establecimiento del entorno de desarrollo. En este primer sprint, se llevaron a cabo las actividades iniciales para establecer el entorno de trabajo necesario para el desarrollo del proyecto. Las tareas incluyeron:

- Instalación y configuración del entorno: Instalación de Angular, TypeScript y demás herramientas necesarias para la creación de la aplicación.
- Creación del repositorio en GitHub: Se configuró el repositorio donde se gestionaría el código fuente del proyecto, asegurando el control de versiones adecuado.
- Integración inicial de APIs: Investigación y conexión inicial con APIs externas para la obtención de datos sobre las temporadas de Fórmula 1, como resultados de carreras y clasificaciones de pilotos.
- Diseño preliminar de la interfaz de usuario (UI): Creación de los primeros bocetos de la interfaz para tener una idea general del diseño de la aplicación.

Tareas/Objetivos	Estado
Instalación y configuración del entorno	Completado
Creación del repositorio en GitHub	Completado
Integración inicial de APIs externas	Completado
Diseño preliminar de la interfaz de usuario (UI)	Completado

Tabla A.1: Tareas y objetivos del Sprint 1 (20/04/2025 - 26/04/2025)

Objetivo alcanzado: Se logró configurar el entorno de desarrollo y se estableció la conexión básica con las APIs, así como un esquema preliminar de la UI.

Sprint 2 (27/04/2025 - 03/05/2025)

Desarrollo de la interfaz de usuario (UI) Durante este sprint, el enfoque principal fue la creación de la interfaz de usuario, que debería ser interactiva y responsive para adaptarse a distintos dispositivos. Las tareas realizadas fueron:

- Diseño y desarrollo de la UI: Implementación de las vistas básicas para mostrar la información de las temporadas de F1, resultados de carreras y clasificaciones.
- Integración de gráficos: Uso de bibliotecas como Ngx-charts para representar gráficamente los resultados de las carreras y las estadísticas de pilotos y equipos.
- Adaptación responsive: Se aplicó CSS y SCSS para hacer la aplicación compatible con dispositivos móviles y de escritorio.
- Primeras pruebas de interfaz: Se realizaron pruebas iniciales de usabilidad para asegurar que la interfaz fuera clara y accesible para los usuarios.

Tareas/Objetivos	Estado
Diseño y desarrollo de la UI	Completado
Integración de gráficos con Chart.js	Completado
Adaptación responsive	Completado
Primeras pruebas de interfaz	Completado

Tabla A.2: Tareas y objetivos del Sprint 2 (27/04/2025 - 03/05/2025)

Objetivo alcanzado: Se completó la interfaz de usuario básica y se integraron los primeros gráficos con los datos de las APIs externas.

Sprint 3 (04/05/2025 - 10/05/2025)

Implementación del simulador de estrategias de carrera (Parte 1) En este sprint, se comenzó con el desarrollo del simulador de estrategias de carrera, una de las funcionalidades más complejas y fundamentales del proyecto. Las tareas fueron:

- Desarrollo de la lógica del simulador: Implementación de las primeras funciones para simular la degradación de neumáticos y calcular el impacto en los tiempos de vuelta.
- Integración de la simulación con los datos de la carrera: El simulador comenzó a interactuar con los resultados de las carreras y las clasificaciones para calcular las mejores estrategias de parada.
- Desarrollo de la interfaz del simulador: Diseño y creación de las vistas necesarias para que el usuario pueda seleccionar los neumáticos y los tiempos de parada.

Tareas/Objetivos	Estado
Desarrollo de la lógica del simulador	Completado
Integración de la simulación con los datos de la carrera	Completado
Desarrollo de la interfaz del simulador	Completado

Tabla A.3: Tareas y objetivos del Sprint 3 (04/05/2025 - 10/05/2025)

Objetivo alcanzado: Se logró establecer la base del simulador, con la capacidad de simular la degradación de neumáticos y calcular tiempos de carrera.

Sprint 4 (11/05/2025 - 17/05/2025)

Optimización del simulador y pruebas iniciales Este sprint se dedicó a la optimización y las primeras pruebas de funcionamiento del simulador de estrategias de carrera. Las tareas realizadas fueron:

- Optimización de la lógica del simulador: Se ajustaron las fórmulas para hacer los cálculos más precisos y realistas en cuanto a la degradación de neumáticos y los tiempos de parada.
- Pruebas del simulador: Se realizaron pruebas internas para asegurar que el simulador generara resultados coherentes y lógicos, y se ajustaron las variables de entrada.
- Revisión de la interfaz: Se realizaron ajustes en la interfaz para mejorar la usabilidad y la interactividad del simulador.
- Pruebas de integración con otras funcionalidades: Se integraron los resultados de la simulación con los datos de la temporada y se aseguraron los flujos de trabajo entre las diferentes partes de la aplicación.

Tareas/Objetivos	Estado
Optimización de la lógica del simulador	Completado
Pruebas del simulador	Completado
Revisión de la interfaz	Completado
Pruebas de integración con otras funcionalidades	Completado

Tabla A.4: Tareas y objetivos del Sprint 4 (11/05/2025 - 17/05/2025)

Objetivo alcanzado: El simulador fue optimizado y probado, y se validaron las funcionalidades de la estrategia de neumáticos.

Sprint 5 (18/05/2025 - 24/05/2025)

Desarrollo de pruebas de carga. En este sprint, el enfoque estuvo en las pruebas de carga para garantizar el rendimiento de la aplicación. Las tareas realizadas fueron:

- Pruebas de carga y rendimiento: Se realizaron pruebas de carga para verificar que la aplicación pudiera manejar grandes volúmenes de datos en tiempo real sin perder rendimiento.
- Revisión de la usabilidad: Se realizaron ajustes finales en la interfaz para mejorar la experiencia del usuario y garantizar que la aplicación fuera fácil de usar.

Tareas/Objetivos	Estado
Pruebas de carga y rendimiento	Completado
Revisión de la usabilidad	Completado

Tabla A.5: Tareas y objetivos del Sprint 5 (18/05/2025 - 24/05/2025)

Objetivo alcanzado: Se completó la funcionalidad y se aseguraron los requisitos de rendimiento para un uso fluido.

Sprint 6 (25/05/2025 - 31/05/2025)

Pruebas finales y ajustes de la interfaz Este sprint se dedicó a la finalización de la aplicación, realizando las pruebas finales y los últimos ajustes en la interfaz de usuario. Las tareas fueron:

- Pruebas finales: Se realizaron pruebas de integración entre todas las partes de la aplicación para garantizar que todo funcionara correctamente.
- Ajustes en el diseño: Se mejoró la estética y la interactividad de la interfaz, asegurando que fuera intuitiva y atractiva para los usuarios.
- Documentación técnica: Se comenzó a documentar el proceso de desarrollo y las funcionalidades de la aplicación, lo que facilitará su mantenimiento futuro.

Tareas/Objetivos	Estado
Pruebas finales de integración	Completado
Ajustes en el diseño	Completado
Documentación técnica inicial	Completado

Tabla A.6: Tareas y objetivos del Sprint 6 (25/05/2025 - 31/05/2025)

Objetivo alcanzado: Se completaron las pruebas finales y se ajustaron los detalles de la interfaz.

Sprint 7 (01/06/2025 - 10/06/2025)

Despliegue y documentación final Este último sprint se dedicó a la puesta en producción de la aplicación y a la finalización de la documentación. Las tareas realizadas fueron:

- Despliegue en Netlify: Se completó el despliegue de la aplicación en la plataforma de Netlify para su publicación en línea.
- Documentación final: Se completó la documentación técnica y el manual de usuario, asegurando que la aplicación estuviera completamente documentada.
- Revisión final y cierre del proyecto: Se hizo una revisión final de todas las funcionalidades y se aseguraron que todos los aspectos del proyecto estuvieran cerrados y listos para su entrega.

Tareas/Objetivos	Estado
Despliegue en Netlify	Completado
Documentación final	Completado
Revisión final y cierre del proyecto	Completado

Tabla A.7: Tareas y objetivos del Sprint 7 (01/06/2025 - 07/06/2025)

Objetivo alcanzado: El proyecto fue desplegado en Netlify y se completó la documentación final.

A.3. Estudio de viabilidad

En este apartado se realiza un análisis de la viabilidad económica y legal del proyecto F1nalLap. El objetivo es evaluar los costos asociados al desarrollo y mantenimiento de la aplicación, así como garantizar el cumplimiento de las normativas legales relacionadas con el uso de herramientas y APIs externas.

Viabilidad económica

El desarrollo de F1nalLap está basado en un modelo de código abierto y gratuito, lo que implica que no se prevé la monetización directa del proyecto. Sin embargo, se ha realizado un análisis de los costos involucrados en el proceso de desarrollo y mantenimiento para asegurar la viabilidad del proyecto a largo plazo.

Costes asociados

El principal coste asociado al proyecto es el coste de mano de obra, ya que se requiere un desarrollo especializado en tecnologías como Angular, TypeScript, y la integración de APIs externas.

- Mano de obra: El proyecto será desarrollado por una persona, con un total estimado de 400 horas de trabajo distribuidas a lo largo de 7 semanas. El coste estimado por hora de desarrollo es de 20€/hora.
- Coste total de mano de obra:

$$400 \text{ horas} \times 20 \text{ €/hora} = 8,000 \text{ €}$$

- Herramientas y software: La mayoría de las herramientas utilizadas, como Angular y Netlify, son gratuitas. Sin embargo, algunas APIs externas utilizadas en la aplicación podrían requerir una suscripción premium para garantizar la escalabilidad y funcionalidad completa, lo que implicaría un costo aproximado de 100€/mes.
- Hardware: No se incurre en costos adicionales de hardware, ya que se utilizará un ordenador portátil ya disponible, completamente amortizado.

Coste de funcionamiento

Además de los costos de desarrollo, también se deben considerar los costos de funcionamiento, que incluyen el mantenimiento de la infraestructura y posibles tarifas por servicios premium en APIs externas. Para el mantenimiento básico del servidor en Netlify y el uso de las APIs externas (si se opta por planes premium), el coste mensual estimado es de:

$$100 \text{ €/mes (APIs externas)} + \text{costes de servidores}$$

Total de costes

El cálculo total de los costes es el siguiente:

$$\text{Costes de desarrollo} = 8,000 \text{ € (mano de obra)}$$

$$\text{Costes mensuales de funcionamiento} = 100 \text{ €/mes (APIs externas)} + \text{costes de servidores}$$

Viabilidad legal

El proyecto F1nalLap utilizará una serie de APIs externas para obtener datos de la Fórmula 1, tales como resultados de carreras, estadísticas de pilotos y clasificaciones de equipos. Estas APIs se utilizarán de acuerdo con las licencias y condiciones de uso de cada una.

Licencias de las herramientas utilizadas

A continuación se presenta un resumen de las principales herramientas y APIs utilizadas, junto con sus respectivas licencias:

- Angular: Licencia MIT. Framework de desarrollo de aplicaciones web que facilita la creación de interfaces interactivas y modulares.
- Netlify: Licencia gratuita para el alojamiento continuo y la implementación del proyecto. Netlify proporciona un plan gratuito con características básicas para proyectos pequeños.
- APIs externas: Se utilizarán APIs públicas y gratuitas para obtener datos de la Fórmula 1. Algunas de las APIs utilizadas incluyen la API de Jolpi para acceder a resultados de carreras y clasificaciones. Las licencias de estas APIs permiten su uso público para proyectos de investigación y desarrollo de aplicaciones web.
- Ngx-charts: Licencia MIT. Librería de gráficos que facilita la representación visual de los datos.

Apéndice B

Especificación de Requisitos

B.1. Introducción

Este apartado tiene como objetivo describir los requisitos funcionales y no funcionales del proyecto F1nalLap. A continuación, se detallan los casos de uso y su respectiva descripción, considerando las funcionalidades necesarias para que la aplicación cumpla con los objetivos establecidos. Además, se definen las precondiciones, acciones, postcondiciones y excepciones asociadas a cada uno de los casos de uso.

B.2. Objetivos generales

Los objetivos generales del proyecto F1nalLap son los siguientes:

- Desarrollar una aplicación web que permita a los usuarios consultar información sobre temporadas pasadas y actuales de Fórmula 1, incluyendo resultados, clasificaciones, estadísticas de pilotos y equipos, y más.
- Implementar un simulador de estrategias de carrera que permita a los usuarios experimentar con diferentes configuraciones de neumáticos, tiempos de pit stop y duración de los stints para optimizar la estrategia de carrera.
- Ofrecer una interfaz de usuario responsiva y modular, compatible con dispositivos móviles, tabletas y de escritorio.

- Mantener la aplicación actualizada en tiempo real utilizando APIs externas para obtener los últimos datos de las temporadas de Fórmula 1.

B.3. Catálogo de requisitos

A continuación se describen los requisitos funcionales y no funcionales que se deben cumplir para que la aplicación sea completamente funcional y eficiente.

Requisitos funcionales

- **RF-01:** La aplicación debe permitir al usuario consultar los resultados de las carreras de la temporada actual y de temporadas pasadas.
- **RF-02:** Los usuarios podrán ver las clasificaciones de pilotos y equipos actualizadas en tiempo real.
- **RF-03:** El simulador de estrategias de carrera debe permitir a los usuarios elegir neumáticos (blandos, medios, duros) y simular diferentes combinaciones de paradas en boxes.
- **RF-04 - Análisis de pilotos y equipos:**
 - **RF-04.1 - Mostrar pilotos o equipos:** La aplicación debe permitir al usuario elegir y visualizar pilotos o equipos de la temporada elegida mediante un selector. El selector debe permitir elegir de manera simultánea dos pilotos o dos equipos para mostrar su rendimiento conjunto.
 - **RF-04.2 - Gráficas y estadísticas:** La aplicación debe ser capaz de mostrar mediante gráficos interactivos las estadísticas de los pilotos y equipos. Estas estadísticas deben incluir puntos, victorias y puntos por carrera..
- **RF-05:** El sistema debe mostrar el calendario de las carreras de la temporada actual y las fechas de cada Gran Premio.
- **RF-06:** La aplicación debe contar con una sección de noticias y actualizaciones relacionadas con la Fórmula 1, donde los usuarios puedan leer las últimas noticias sobre pilotos, equipos y eventos.

- **RF-07 - Menú de la aplicación:** La aplicación debe contener un menú en el que puedas navegar por las distintas funcionalidades que tiene la aplicación.
- **RF-08 - Apartado de temporadas:**
 - **RF-08.1 - Temporada actual:** La aplicación debe permitir al usuario acceder a la temporada actual, mostrando información básica como el líder del campeonato, la clasificación de pilotos y equipos, y los próximos eventos de la temporada en curso.
 - **RF-08.2 - Temporadas pasadas:** La aplicación debe contar con una sección de temporadas pasadas, donde los usuarios puedan elegir una temporada específica y consultar los resultados del campeonato de pilotos o constructores de esa temporada, así como los resultados de las carreras correspondientes.

Requisitos no funcionales

- **RNF-01:** La aplicación debe ser responsiva, adaptándose a diferentes tamaños de pantalla (móviles, tabletas y ordenadores de escritorio).
- **RNF-02:** La aplicación debe ser capaz de manejar grandes volúmenes de datos en tiempo real sin perder rendimiento.
- **RNF-03:** La aplicación debe tener un tiempo de respuesta rápido, asegurando que las interacciones con el usuario sean fluidas y sin retrasos significativos.
- **RNF-04:** La aplicación debe ser segura, protegiendo la información del usuario, especialmente en el caso de registro y autenticación.
- **RNF-05:** La aplicación debe estar disponible 24/7, minimizando el tiempo de inactividad al máximo.
- **RNF-06:** La aplicación debe ser fácil de mantener y actualizar, con un código modular y bien documentado.
- **RNF-07:** La aplicación debe tener una interfaz visual e intuitiva, garantizando que los usuarios puedan interactuar con la plataforma de forma sencilla y eficiente.

B.4. Especificación de requisitos

La especificación de requisitos describe los casos de uso asociados a la aplicación, detallando cómo el sistema debe comportarse en diferentes situaciones. A continuación, se presenta un ejemplo de cómo se define un caso de uso básico de la aplicación.

CU-1	Consultar resultados de la carrera
Requisitos asociados	RF-01, RF-05
Descripción	El usuario podrá consultar los resultados de las carreras de la temporada actual y de temporadas pasadas. Los resultados estarán actualizados en tiempo real.
Precondición	El usuario debe tener acceso a la aplicación.
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la sección de Carreras en Clasificación en la interfaz principal (temporada actual) o a la sección Temporadas Anteriores en Formula 1 en el menú principal. 2. Elige la temporada y la carrera de interés. 3. La aplicación muestra los resultados detallados de la carrera seleccionada.
Postcondición	Los resultados de la carrera seleccionada son mostrados correctamente.
Excepciones	Si no hay datos disponibles, se muestra un mensaje de error indicando que no se pueden obtener los resultados.
Importancia	Alta

Tabla B.1: CU-1 Consultar resultados de la carrera.

CU-2	Ver clasificaciones de pilotos y equipos
Requisitos asociados	RF-02
Descripción	Los usuarios podrán ver las clasificaciones de pilotos y equipos actualizadas en tiempo real.
Precondición	El usuario debe tener acceso a la aplicación.
Acciones	<ol style="list-style-type: none">1. El usuario accede a la sección de Clasificaciones en la interfaz principal.2. La aplicación muestra la clasificación actualizada de pilotos y equipos.
Postcondición	La clasificación de pilotos y equipos es mostrada correctamente.
Excepciones	Si no hay datos disponibles, se muestra un mensaje de error indicando que no se pueden obtener las clasificaciones.
Importancia	Alta

Tabla B.2: CU-2 Ver clasificaciones de pilotos y equipos.

CU-3	Simular estrategia de carrera
Requisitos asociados	RF-03
Descripción	El simulador de estrategias de carrera debe permitir a los usuarios elegir neumáticos (blandos, medios, duros) y simular diferentes combinaciones de paradas en boxes.
Precondición	El usuario debe estar autenticado.
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la sección de Simulador de estrategias. 2. Elige los neumáticos y los tiempos de parada en boxes. 3. La aplicación muestra la simulación de la estrategia de carrera.
Postcondición	La estrategia de carrera es mostrada correctamente, incluyendo el impacto de los neumáticos y las paradas en el tiempo total de la carrera.
Excepciones	Si los datos ingresados no son válidos, se muestra un mensaje de error indicando que no se puede realizar la simulación.
Importancia	Alta

Tabla B.3: CU-3 Simular estrategia de carrera.

CU-4	Análisis de pilotos y equipos
Requisitos asociados	RF-04.1, RF-04.2, RF-04.3
Descripción	La aplicación debe permitir al usuario elegir y comparar pilotos o equipos, mostrando sus estadísticas en gráficos interactivos.
Precondición	El usuario debe tener acceso a la aplicación.
Acciones	<ol style="list-style-type: none">1. El usuario accede a la sección de Análisis de pilotos y equipos.2. Elige los pilotos o equipos a comparar.3. La aplicación muestra las estadísticas en gráficos interactivos, permitiendo la comparación entre ellos.
Postcondición	Los gráficos comparativos de pilotos y equipos son mostrados correctamente.
Excepciones	Si no hay datos disponibles, se muestra un mensaje de error indicando que no se pueden obtener las estadísticas.
Importancia	Alta

Tabla B.4: CU-4 Análisis de pilotos y equipos.

CU-5	Consultar calendario de carreras
Requisitos asociados	RF-05
Descripción	El sistema debe mostrar el calendario de las carreras de la temporada actual, incluyendo las fechas de cada Gran Premio.
Precondición	El usuario debe tener acceso a la aplicación.
Acciones	<ol style="list-style-type: none"> 1. El usuario accede a la sección de Clasificación y selecciona Carreras. 2. La aplicación muestra el calendario con las fechas y ubicaciones de los próximos Grandes Premios y los resultados de los gran premios ya realizados.
Postcondición	El calendario de las carreras de la temporada actual es mostrado correctamente.
Excepciones	Si no se pueden obtener las fechas, se muestra un mensaje de error indicando que no se puede cargar el calendario.
Importancia	Media

Tabla B.5: CU-5 Consultar calendario de carreras.

CU-6	Ver noticias y actualizaciones
Requisitos asociados	RF-06
Descripción	La aplicación debe mostrar una sección de noticias donde los usuarios puedan leer las últimas actualizaciones e innovaciones.
Precondición	El usuario debe tener acceso a la aplicación.
Acciones	<ol style="list-style-type: none">1. El usuario accede a la sección de Noticias.2. La aplicación muestra las últimas noticias relacionadas con la Fórmula 1.
Postcondición	Las noticias más recientes son mostradas correctamente.
Excepciones	Si no hay noticias disponibles, se muestra un mensaje de error indicando que no se pueden obtener las noticias.
Importancia	Baja

Tabla B.6: CU-6 Ver noticias y actualizaciones.

Apéndice C

Especificación de diseño

C.1. Introducción

En este apartado se detallan los aspectos clave del diseño de la aplicación F1nalLap. Aquí se describe cómo se organizan los datos, la arquitectura del sistema y los procedimientos que guían el funcionamiento interno de la aplicación. Este diseño está orientado a garantizar que la aplicación sea eficiente, escalable y fácil de mantener.

C.2. Diseño de datos

El diseño de datos de la aplicación se enfoca en cómo se almacenan y gestionan los datos dentro del sistema. La aplicación utiliza una base de datos para almacenar información sobre las temporadas de la Fórmula 1, resultados de carreras, clasificaciones de pilotos y equipos, y estrategias de carrera simuladas.

Modelo de base de datos

Para la gestión de los datos, se utiliza una base de datos NoSQL, lo que permite flexibilidad y escalabilidad en el almacenamiento de datos de diferentes tipos, como información de temporadas, estadísticas de carreras y datos históricos. A continuación se describe el esquema de las colecciones principales de la base de datos:

- Temporadas: Almacena información sobre cada temporada, incluyendo los eventos de la temporada (fechas, circuitos, etc.).

- Carreras: Contiene los resultados de cada carrera, incluyendo la posición final de cada piloto y los puntos obtenidos.
- Pilotos y equipos: Guarda las estadísticas detalladas de los pilotos y equipos, como victorias, puntos acumulados, posiciones por carrera, entre otros.
- Simulaciones: Registra los resultados de las simulaciones de estrategias de carrera realizadas por los usuarios, incluyendo los datos sobre las combinaciones de neumáticos y tiempos de pit stop.

Relaciones entre entidades

Las relaciones entre las entidades del modelo de datos se gestionan mediante claves de referencia. Por ejemplo, cada carrera se refiere a una temporada específica, y cada piloto o equipo está asociado a los resultados de diversas carreras a lo largo de la temporada.

C.3. Diseño arquitectónico

El diseño arquitectónico de F1nalLap se basa en una estructura cliente-servidor, donde la aplicación interactúa con el backend a través de APIs RESTful para obtener y enviar datos. A continuación se detallan los componentes principales de la arquitectura:

Arquitectura cliente-servidor

- Frontend (Cliente): La interfaz de usuario de la aplicación se desarrolla utilizando Angular, un framework de JavaScript, que permite construir aplicaciones web interactivas y escalables. El cliente realiza solicitudes al servidor para obtener datos, mostrar resultados y simular estrategias de carrera.

- Backend (Servidor): El backend se encarga de procesar las solicitudes provenientes del cliente y de interactuar con las APIs externas para gestionar la información. Las interacciones se realizan utilizando TypeScript y APIs RESTful. El servidor maneja los datos, realizando cálculos como los de la simulación de estrategias de carrera, y devuelve la información al cliente de manera eficiente.

- APIs externas: La aplicación se conecta con APIs externas para obtener datos en tiempo real sobre los resultados de las carreras, clasificaciones

y estadísticas de pilotos y equipos. Estas APIs proporcionan información actualizada sobre la Fórmula 1 sin necesidad de almacenar grandes volúmenes de datos en el backend.

Flujo de datos

El flujo de datos en la aplicación sigue el siguiente patrón:

1. El cliente envía una solicitud para obtener información (por ejemplo, resultados de la carrera o clasificaciones).
2. El backend procesa la solicitud, interactuando con las APIs externas para obtener los datos requeridos.
3. Los datos son enviados de vuelta al cliente, donde se procesan y se muestran al usuario en la interfaz.

Este flujo de datos asegura una comunicación eficiente entre el cliente y el servidor, optimizando la experiencia del usuario.

C.4. Diseño procedimental

El diseño procedimental describe los algoritmos y procedimientos utilizados en la aplicación para cumplir con los requisitos funcionales. A continuación, se describen algunos de los procedimientos más importantes utilizados en F1naLLap.

Simulación de estrategias de carrera

El simulador de estrategias de carrera es uno de los componentes clave de la aplicación. La simulación permite a los usuarios experimentar con diferentes configuraciones de neumáticos, tiempos de parada y número de stints para optimizar su estrategia de carrera. El procedimiento para calcular y simular una estrategia de carrera es el siguiente:

1. Selección de neumáticos y paradas: El usuario selecciona los neumáticos (blandos, medios o duros) y el número de paradas que desea realizar durante la carrera. Además, el sistema ofrece la opción de definir la duración de cada stint.

2. Cálculo de la degradación de neumáticos: El sistema calcula la degradación de los neumáticos en función del tipo de compuesto elegido (blando, medio, duro) y las condiciones de la pista. Se utiliza una fórmula no lineal que ajusta la degradación a medida que avanza la carrera, con un factor de variabilidad aleatorio para simular las fluctuaciones en el rendimiento.
3. Simulación de paradas en boxes: Se simulan las paradas en boxes, teniendo en cuenta el tiempo de cada pit stop (tiempo que el coche pasa en la parada). El algoritmo calcula el momento óptimo para cada parada, considerando la degradación de los neumáticos.
4. Cálculo del tiempo total de la carrera: El sistema calcula el tiempo total de la carrera, sumando el tiempo de vuelta (ajustado por la degradación de neumáticos) y el tiempo de pit stop. Este cálculo incluye los tiempos de cambio de neumáticos, así como la pérdida de tiempo en las paradas.
5. Comparación de estrategias: Una vez completado el cálculo de la estrategia seleccionada, el sistema compara el tiempo total estimado con otras estrategias posibles, mostrando cuál sería la opción más óptima en función de las variables de entrada (tipo de neumático, número de paradas, condiciones de la pista).
6. Visualización de los resultados: El sistema presenta los resultados de la simulación en un formato visual, utilizando gráficos interactivos para mostrar la evolución de la degradación de neumáticos y los tiempos por vuelta. También se muestra una comparación entre las estrategias simuladas.

Este procedimiento permite a los usuarios experimentar con diferentes configuraciones de neumáticos y paradas, ofreciendo una herramienta valiosa para analizar las decisiones estratégicas durante una carrera de Fórmula 1.

Obtención de datos en tiempo real

Para mostrar los resultados de las carreras y las clasificaciones en tiempo real, la aplicación interactúa con las APIs externas. El procedimiento para obtener y mostrar esta información es el siguiente:

1. Solicitud de datos a la API externa: El cliente realiza una solicitud a la API externa correspondiente para obtener los datos más recientes

sobre los resultados de las carreras, las clasificaciones de pilotos y equipos, o cualquier otra información relevante de la Fórmula 1.

2. Recepción de datos en formato JSON: La API externa responde con los datos solicitados en formato JSON. Este formato es ligero, fácil de procesar y ampliamente utilizado para la transmisión de datos entre el cliente y el servidor.
3. Procesamiento de los datos: El backend de la aplicación procesa los datos recibidos, extrayendo la información relevante (resultados de las carreras, posiciones de los pilotos, puntos obtenidos, etc.). En este paso, se realiza cualquier ajuste necesario, como la conversión de unidades o la validación de los datos.
4. Envío de datos al cliente: Una vez procesados, los datos son enviados al cliente en formato estructurado para su visualización. Esto puede incluir la clasificación general de los pilotos, los resultados de una carrera específica o estadísticas detalladas de los equipos.
5. Visualización interactiva: Los datos se muestran en la interfaz de usuario a través de gráficos interactivos, tablas y otros elementos visuales. Estos gráficos se actualizan en tiempo real, ofreciendo una experiencia dinámica para el usuario.
6. Actualización continua: La aplicación realiza actualizaciones periódicas para mantener la información al día. La frecuencia de las actualizaciones depende de la naturaleza de los datos y la API utilizada, asegurando que los resultados de las carreras o clasificaciones estén siempre disponibles en tiempo real.

Este procedimiento asegura que los usuarios reciban información actualizada sin retrasos, mejorando la experiencia en la aplicación.

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este apartado se describe la documentación técnica de programación de la aplicación F1nalLap. Esta sección está dirigida principalmente a los programadores que deseen comprender cómo está estructurado el código, cómo instalar y ejecutar el proyecto, y cómo realizar pruebas del sistema. El objetivo es proporcionar una guía detallada para facilitar el mantenimiento y la expansión del proyecto en el futuro.

D.2. Estructura de directorios

La estructura de directorios del proyecto F1nalLap está organizada de manera modular para facilitar el mantenimiento y la escalabilidad. A continuación se describe la estructura básica de directorios:

- /src: Contiene todos los archivos fuente de la aplicación.
 - /app: Aquí se encuentran los componentes, servicios y módulos principales de la aplicación.
 - /assets: Contiene los recursos estáticos como imágenes, estilos CSS, fuentes y otros archivos estáticos.
- /node-modules: Carpeta generada automáticamente por npm, que contiene las dependencias del proyecto.

- `/dist`: Carpeta generada después de la compilación del proyecto, que contiene los archivos listos para producción.
- `/angular.json`: Archivo de configuración principal para el proyecto Angular.
- `/package.json`: Contiene las dependencias del proyecto y scripts de construcción.

La estructura de directorios está diseñada para facilitar la organización y el acceso rápido a los diferentes módulos y componentes de la aplicación.

D.3. Manual del programador

Este manual está diseñado para guiar al programador en el proceso de instalación, configuración y ejecución del proyecto F1nalLap. Aquí se detallan los pasos para instalar las herramientas necesarias, clonar el repositorio, instalar las dependencias y ejecutar la aplicación tanto en el entorno local como en producción.

Instalación de Visual Studio Code

Para instalar Visual Studio Code (VS Code), sigue estos pasos:

1. Descarga Visual Studio Code desde su sitio web oficial: <https://code.visualstudio.com/>.
2. Ejecuta el instalador y sigue las instrucciones en pantalla para completar la instalación.
3. Una vez instalado, puedes abrir VS Code desde el menú de inicio o desde una terminal escribiendo:

```
code .
```

Esto abrirá VS Code en el directorio en el que te encuentres.

Instalación de Node.js y npm

Node.js es un entorno de ejecución para JavaScript en el lado del servidor, y npm es el gestor de paquetes de Node.js. Aunque npm no se usa directamente en este proyecto, es necesario para gestionar las dependencias de Angular.

Para instalar Node.js y npm:

1. Descarga el instalador de Node.js desde su sitio web oficial: <https://nodejs.org/en/download/>.
2. Ejecuta el instalador y sigue las instrucciones. Este instalador también instalará npm.
3. Abre una terminal y ejecuta los siguientes comandos para verificar la instalación:

```
node -v  
npm -v
```

Estos comandos deberían devolver las versiones de Node.js y npm instaladas.

Clonar el repositorio de GitHub

Para montar el proyecto en tu entorno local, el primer paso es clonar el repositorio del proyecto desde GitHub. Abre la terminal y sigue estos pasos:

1. Navega al directorio donde deseas clonar el repositorio.
2. Ejecuta el siguiente comando para clonar el repositorio:

```
git clone https://github.com/Ixo22/F1nalLap
```

3. Cambia al directorio del proyecto:

```
cd F1nalLap
```

Instalar las dependencias de Angular

Dentro del directorio del proyecto, deberás instalar las dependencias de Angular utilizando npm. Para ello, ejecuta los siguientes comandos en la terminal:

1. Navega al directorio principal del proyecto:

```
cd F1nalLap
```

2. Instala las dependencias del proyecto:

```
npm install
```

Esto descargará todas las dependencias necesarias para que la aplicación funcione correctamente.

3. Asegúrate de tener la última versión de Angular CLI ejecutando:

```
npm install -g @angular/cli
```

Inicializar Netlify

Si deseas desplegar la aplicación en Netlify para pruebas o producción, sigue estos pasos:

1. Crea una cuenta en Netlify si aún no tienes una:

<https://www.netlify.com/>.

2. Inicia sesión en tu cuenta y haz clic en "New site from Git".
3. Conecta tu cuenta de GitHub y selecciona el repositorio F1nalLap.
4. Configura el comando de construcción:

```
npm run build --prod
```

5. Luego, configura el directorio de salida para Netlify como /dist.
6. Haz clic en Deploy site y Netlify comenzará a construir y desplegar tu aplicación.

Ejecución del proyecto en local

Una vez realizado el proceso de instalación y configuración, puedes ejecutar el proyecto en tu máquina local para observar los cambios realizados y comprobar su funcionamiento. Para ello, utiliza uno de los siguientes comandos desde el directorio principal del proyecto:

1. Ejecutar el servidor de desarrollo con Angular CLI:

```
ng serve
```

2. Alternativamente, ejecuta:

```
npm run start
```

Esto iniciará un servidor local y podrás ver la aplicación corriendo en <http://localhost:4200/>.

Con estos pasos, tendrás el proyecto en funcionamiento localmente y podrás comenzar a desarrollar y realizar pruebas.

Despliegue de la aplicación con Netlify

Para desplegar la aplicación en un entorno de producción en Netlify, sigue estos pasos:

1. Después de realizar las pruebas locales y asegurarte de que todo funcione correctamente, realiza una compilación para producción ejecutando el siguiente comando:

```
ng build --prod
```

2. Una vez completada la compilación, sube los archivos generados en el directorio `/dist` a Netlify siguiendo los pasos mencionados anteriormente.
3. Netlify proporcionará una URL donde la aplicación estará disponible para su uso en producción.

Apéndice E

Documentación de usuario

E.1. Introducción

La Documentación de usuario está destinada a los usuarios finales de la aplicación F1nalLap. El objetivo de esta sección es proporcionar una guía detallada sobre cómo usar la aplicación, incluyendo la instalación, los requisitos del sistema y un manual de uso que facilite la interacción con todas las funcionalidades de la plataforma. F1nalLap es una aplicación web interactiva dedicada a la Fórmula 1, que permite consultar resultados de carreras, clasificaciones, simular estrategias de carrera y acceder a información sobre temporadas pasadas y actuales.

E.2. Requisitos de usuarios

Para usar la aplicación F1nalLap, los usuarios deben cumplir con los siguientes requisitos:

- Navegador web compatible: La aplicación es compatible con los navegadores más populares, como Google Chrome, Mozilla Firefox, Microsoft Edge, y Safari.
- Conexión a Internet: Para acceder a los resultados de las carreras y las clasificaciones en tiempo real, es necesario contar con una conexión estable a Internet.
- Dispositivo con acceso a la web: La aplicación es responsiva, por lo que puede usarse tanto en dispositivos de escritorio como en dispositivos móviles y tabletas.

No es necesario registrar una cuenta o iniciar sesión para utilizar la aplicación, ya que el acceso es abierto. Sin embargo, se recomienda que los usuarios tengan una conexión a Internet activa para aprovechar las funcionalidades de la aplicación que dependen de datos en tiempo real.

E.3. Instalación

La aplicación F1nalLap es una aplicación web y no requiere instalación en el sistema local del usuario. A continuación, se describen los pasos para acceder a la aplicación desde un navegador web:

1. Abre tu navegador web preferido (Google Chrome, Mozilla Firefox, etc.).
2. En la barra de direcciones, ingresa la URL de la aplicación: **http://localhost:4200/** si la estás ejecutando localmente o la URL de producción proporcionada después del despliegue en Netlify: **https://f1nallap.netlify.app/**.
3. La aplicación debería cargar automáticamente y estar lista para usar.

No es necesario realizar ninguna instalación en el sistema, ya que toda la funcionalidad se gestiona a través del navegador.

E.4. Manual del usuario

Este documento describe detalladamente cómo utilizar la aplicación F1nalLap, con el fin de ayudar al usuario a comprender y manejar todas sus funcionalidades.

Para empezar, accede a la aplicación en tu navegador mediante la URL **https://f1nallap.netlify.app/** y sigue los pasos que se indican a continuación.

Pantalla de inicio

Al entrar, verás la barra de navegación en la parte superior, el contenido principal (main) y el pie de página (footer) al final de la página.



Figura E.1: Vista principal.

Este apartado está dividida en varias secciones clave. Destaca la Introducción, que explica el propósito de F1nalLap, y una serie de tarjetas con las funcionalidades principales: acceso a la temporada en curso, temporadas pasadas, clasificación de pilotos y equipos, comparador de rendimiento y simulador de estrategias. En la última tarjeta hay varias formas de contactar con nosotros para colaborar en el proyecto o mandar sugerencias.

Barra de navegación

La barra de navegación incluye el logotipo de la aplicación, que al hacer clic devuelve al inicio. Además, dispone de un menú desplegable con accesos directos a las secciones más importantes: Temporadas de F1, Clasificación, Análisis y Simulador.



Figura E.2: Barra de navegación.

Esta barra permanece en todas las vistas de la aplicación. En dispositivos móviles se compacta para mantener las secciones sin que se solapen o se salga de la vista del dispositivo.



Figura E.3: Barra de navegación en móvil.

Pie de página

En el pie de página aparece “© AGA. Todos los derechos reservados.”, que enlaza a mi perfil de LinkedIn.

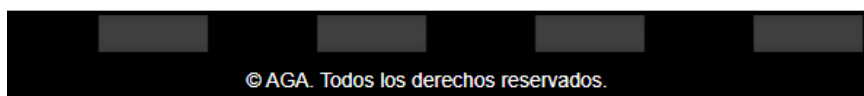


Figura E.4: Pie de página.

Simulador de estrategias

El simulador permite probar distintas combinaciones de neumáticos y número de paradas en boxes. Elige entre blandos, medios o duros y define cuántas paradas quieres simular.

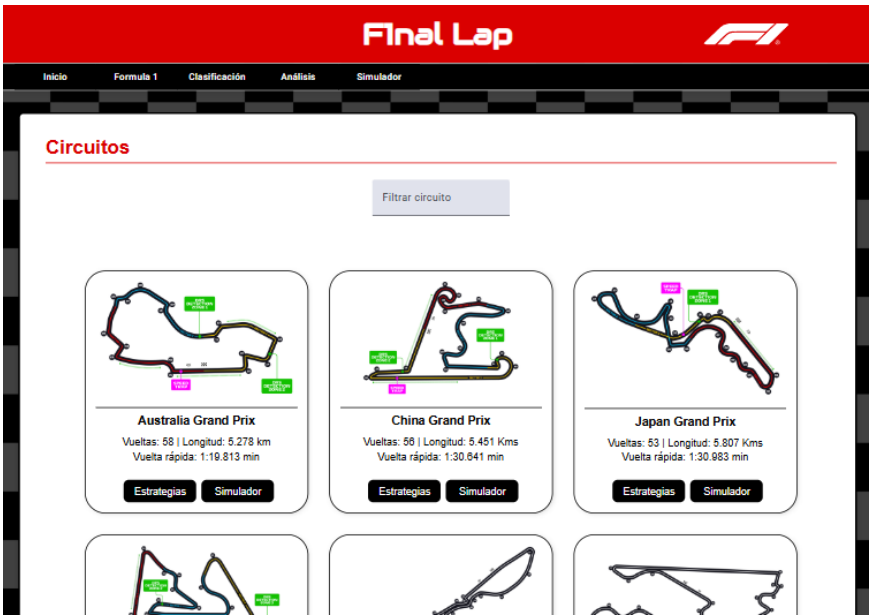


Figura E.5: Selección de circuitos.

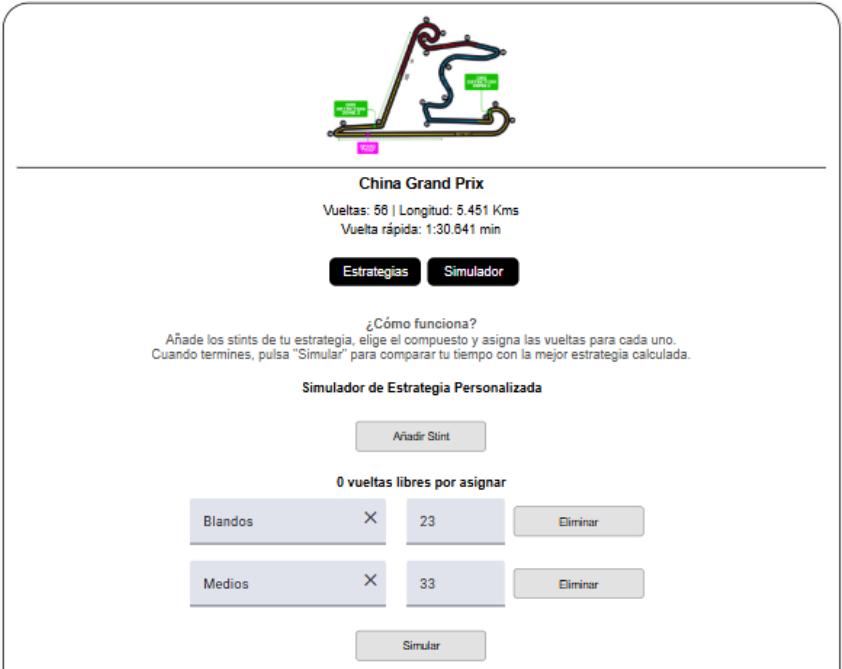


Figura E.6: Selección de estrategia.

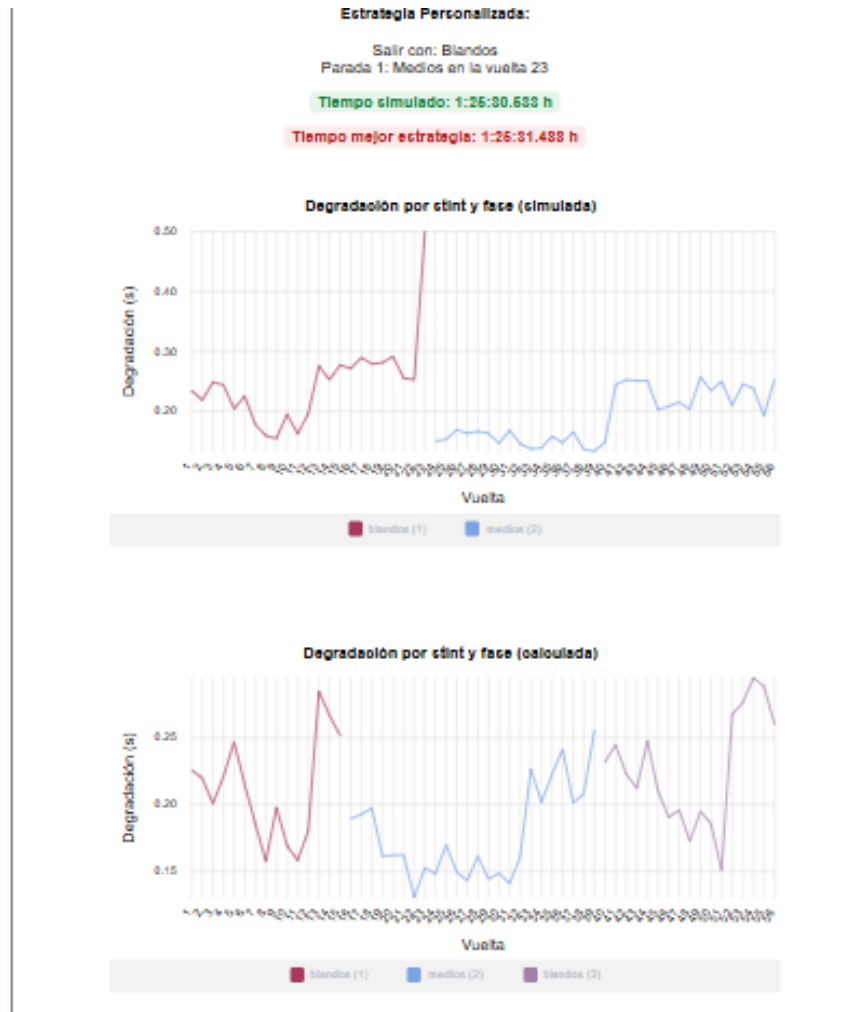


Figura E.7: Resultado de la estrategia.

El sistema estima el tiempo total de carrera considerando el desgaste de los neumáticos y la duración de cada parada, y luego presenta los resultados y la comparación entre las estrategias.

Resultados de carreras

Temporada actual

Para ver los resultados de un Gran Premio de la temporada actual, ve a la sección Clasificación -> Carreras desde el menú. Selecciona la prueba que te interese y mostrar resultados. Si aún no se ha corrido ese gran premio saldrá como "Sin resultados"



Figura E.8: Selección de carrera.

Los datos se actualizan en tiempo real, mostrando posiciones, tiempos por vuelta y puntos.

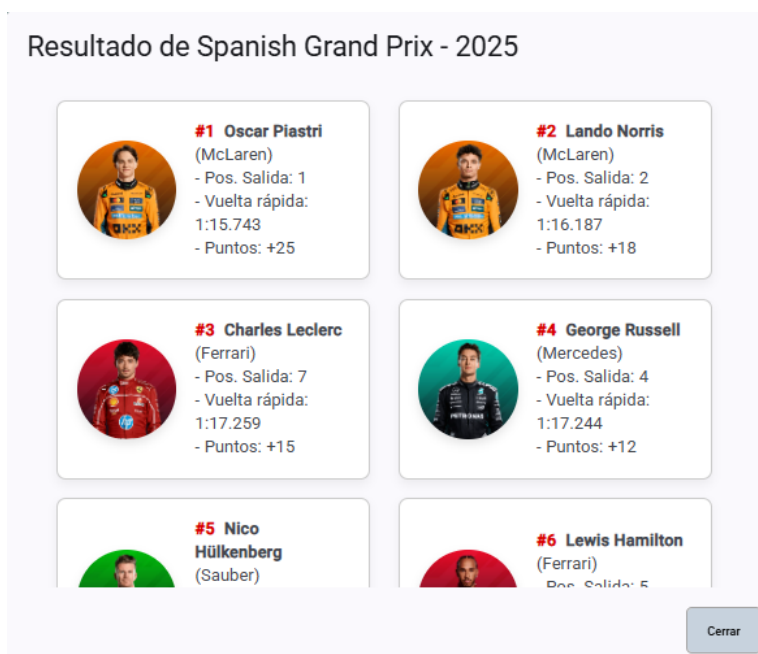


Figura E.9: Resultado de carrera seleccionada.

Temporadas pasadas

Para ver los resultados de cualquier Gran Premio, ve a la sección Fórmula 1 -> Temporadas Anteriores desde el menú. Selecciona la temporada que te interese y pulsar sobre el botón Carreras.



Figura E.10: Selección de temporada.

Resultados Temporada 2012			
	Carrera	Fecha	Resultado
1	Australian Grand Prix	18/03/2012	1. Jenson Button... mostrar todo
2	Malaysian Grand Prix	25/03/2012	1. Fernando Alonso... mostrar todo
3	Chinese Grand Prix	15/04/2012	1. Nico Rosberg... mostrar todo
4	Bahrain Grand Prix	22/04/2012	1. Sebastian Vettel... mostrar todo
5	Spanish Grand Prix	13/05/2012	1. Pastor Maldonado... mostrar todo
6	Monaco Grand Prix	27/05/2012	1. Mark Webber... mostrar todo
7	Canadian Grand Prix	10/06/2012	1. Lewis Hamilton... mostrar todo
8	European Grand Prix	24/06/2012	1. Fernando Alonso... mostrar todo

Figura E.11: Carreras de temporada seleccionada.

Resultado de Malaysian Grand Prix - 2012

Pos.	Piloto	Equipo	Vuelta rápida	Pts.
1	Fernando Alonso	Ferrari	1:41.680	+25
2	Sergio Pérez	Sauber	1:41.021	+18
3	Lewis Hamilton	McLaren	1:41.539	+15
4	Mark Webber	Red Bull	1:41.017	+12
5	Kimi Räikkönen	Lotus F1	1:40.722	+10
6	Bruno Senna	Williams	1:41.404	+8
7	Paul di Resta	Force India	1:41.819	+6

Cerrar

Figura E.12: Resultado de la carrera seleccionada.

Clasificación

En la sección “Clasificación” verás la tabla de pilotos y equipos para la temporada actual, con victorias, puntos y posiciones.

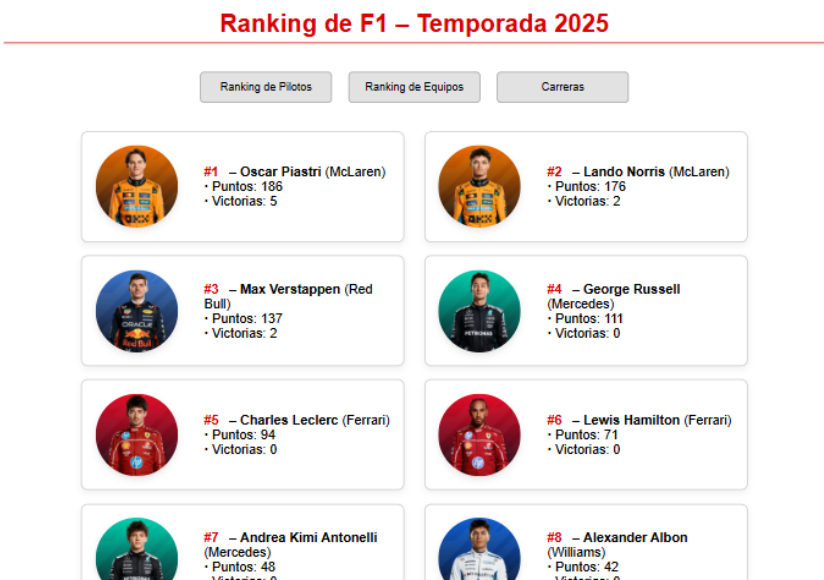


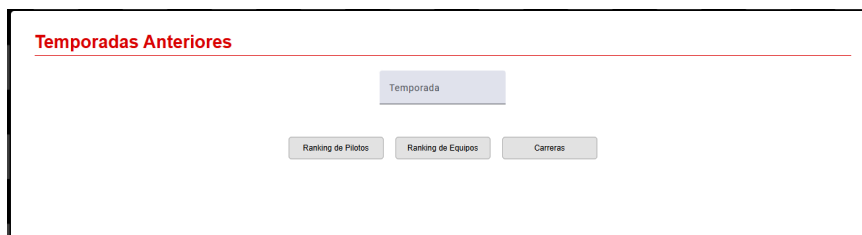
Figura E.13: Tabla de clasificación de pilotos.



Figura E.14: Tabla de clasificación de equipos.

Temporadas anteriores

Desde “Temporadas anteriores” puedes acceder a los datos de años anteriores: campeonatos, resultados de cada GP, etc.



Temporadas Anteriores

Temporada

Ranking de Pilotos Ranking de Equipos Carreras

Figura E.15: Selección de temporada.

Resultados Temporada 2012			
Pos.	Piloto	Escudería	Puntos
1	Sebastian Vettel	Red Bull	281
2	Fernando Alonso	Ferrari	278
3	Kimi Räikkönen	Lotus F1	207
4	Lewis Hamilton	McLaren	190
5	Jenson Button	McLaren	188
6	Mark Webber	Red Bull	179
7	Felipe Massa	Ferrari	122
8	Romain Grosjean	Lotus F1	96
9	Nico Rosberg	Mercedes	93
10	Sergio Pérez	Sauber	66
11	Nico Hülkenberg	Force India	63

Figura E.16: Ranking de pilotos de la temporada seleccionada.

Resultados Temporada 2012		
Pos.	Escudería	Puntos
1	Red Bull	460
2	Ferrari	400
3	McLaren	378
4	Lotus F1	303
5	Mercedes	142
6	Sauber	126
7	Force India	109
8	Williams	76

Figura E.17: Ranking de equipos de la temporada seleccionada.

Análisis comparativo

La sección Análisis ofrece gráficos interactivos para comparar dos pilotos o equipos a lo largo de la temporada, permitiendo estudiar su rendimiento de forma visual.

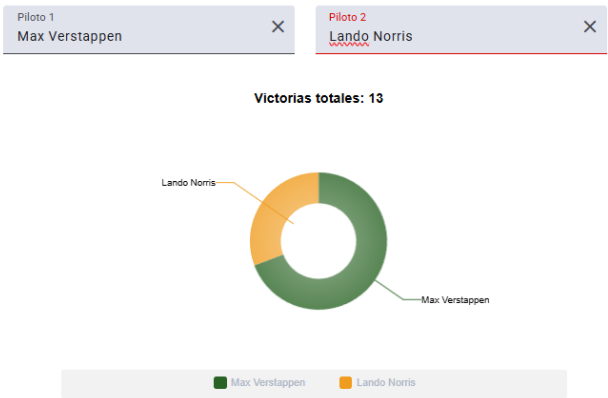


Figura E.18: Comparación de victorias de pilotos.

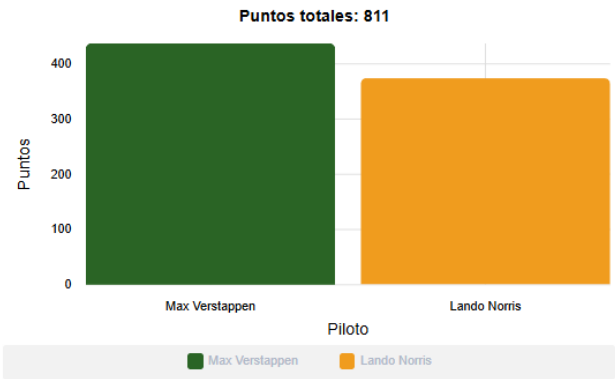


Figura E.19: Comparación de puntos de pilotos.

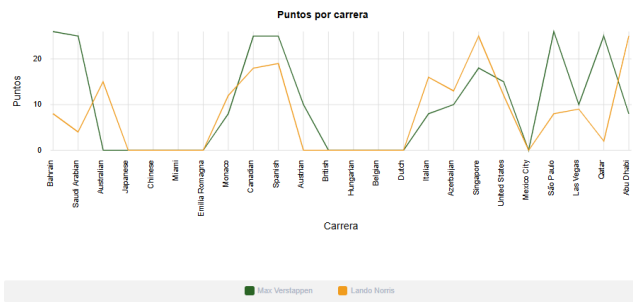


Figura E.20: Comparación de puntos por carrera de pilotos.

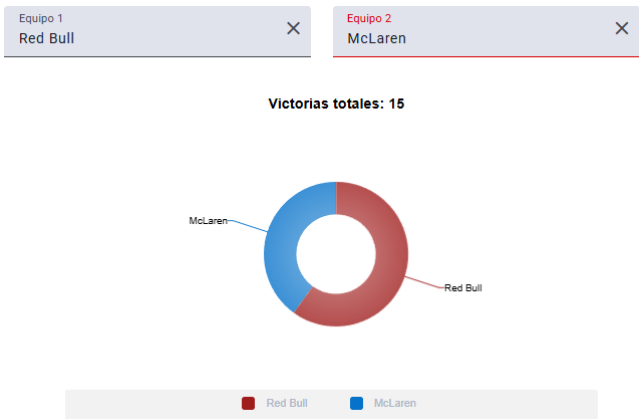


Figura E.21: Comparación de victorias de equipos.

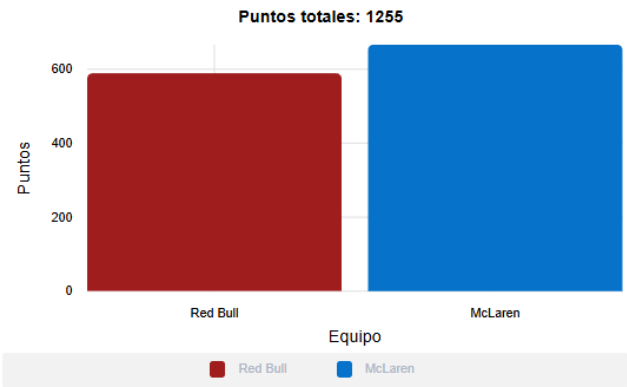


Figura E.22: Comparación de puntos de equipos.

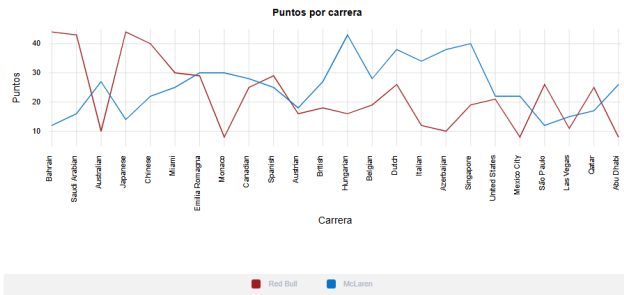


Figura E.23: Comparación de puntos por carrera de equipos.

Contacto

Si deseas enviar sugerencias o comentarios, utiliza el formulario de la sección Contacto en la pestaña de Inicio, usa el enlace del footer o visita el repositorio en GitHub para colaborar.

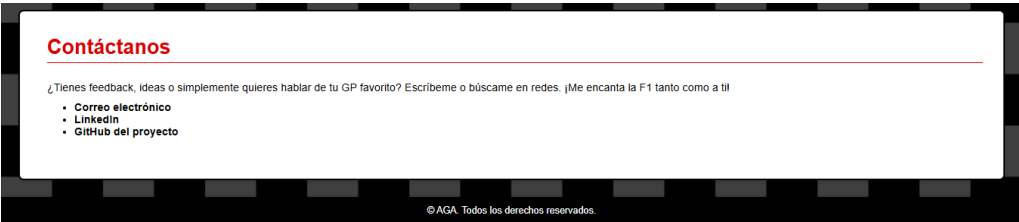


Figura E.24: Método de contacto.

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

En el presente anexo reflexiono sobre mi aprendizaje y competencias en sostenibilidad adquiridas durante el desarrollo de mi Trabajo de Fin de Grado (TFG) en F1nalLap. Parto de las directrices establecidas por la CRUE para integrar criterios de sostenibilidad en la enseñanza superior (CRUE, 2012), y analizo cómo estas buenas prácticas se han aplicado a lo largo del proyecto. El objetivo es mostrar de forma estructurada los aprendizajes, valores y herramientas que he incorporado para fomentar un enfoque sostenible en el desarrollo de software y en mi formación como ingeniero.

Comprensión de los criterios de sostenibilidad

Como punto de partida, revisé el documento de la CRUE [2], identificando tres ejes principales:

- **Ambiental:** optimizar recursos y minimizar el impacto ecológico de las infraestructuras y el código.
- **Social:** favorecer la accesibilidad, la usabilidad y la inclusión de todos los perfiles de usuario.
- **Económico:** aplicar principios de eficiencia presupuestaria y durabilidad de las soluciones.

Esta clasificación me permitió enfocar el TFG no solo como un proyecto técnico, sino también como una oportunidad para desarrollar una conciencia crítica sobre el ciclo de vida del software.

Diseño y desarrollo con enfoque ambiental

En la fase de diseño, prioricé prácticas de programación eficientes:

- **Minimización de peticiones:** integré lazy loading y cache en Angular 19 para reducir el consumo de ancho de banda y energía en el cliente.
- **Optimización de imágenes:** utilicé técnicas de compresión sin pérdida para disminuir el peso de los recursos estáticos.
- **Despliegue sostenible:** seleccioné Netlify, que utiliza CDN con políticas de baja latencia y energías renovables compartidas por el proveedor.

Estas medidas ayudan a reducir el consumo energético tanto en el servidor como en el dispositivo del usuario, alineándose con el criterio ambiental de la CRUE.

Eficiencia económica y mantenimiento

En términos económicos, adopté un enfoque de desarrollo que favorece la sostenibilidad del proyecto a largo plazo:

- **Arquitectura modular:** con componentes standalone y servicios inyectables para facilitar reutilización y pruebas.

Este planteamiento contribuye a un ciclo de vida de software más durable y eficiente en costes.

Reflexión personal y competencias adquiridas

A lo largo del TFG, he desarrollado competencias clave relacionadas con sostenibilidad:

1. **Visión holística:** comprendo el impacto del software más allá de su funcionalidad inmediata.
2. **Toma de decisiones informadas:** soporto cada decisión técnica con criterios medioambientales, sociales y económicos.

3. **Comunicación y divulgación:** soy capaz de justificar ante partes interesadas la relevancia de prácticas sostenibles.

Considero que estos aprendizajes trascienden el proyecto y formarán parte esencial de mi perfil profesional.

F.2. Conclusión

Este anexo ha expuesto cómo he integrado la sostenibilidad en cada fase del TFG, desde el diseño hasta el despliegue, siguiendo las directrices de la CRUE. La experiencia me ha permitido interiorizar valores y metodologías que aplicaré en futuros desarrollos, contribuyendo a la creación de soluciones tecnológicas responsables y perdurables en el tiempo.

Bibliografía

- [1] Alberto García Alcolado. Repositorio f1nallap. <https://github.com/Ixo22/F1nalLap>, 2024.
- [2] CRUE. Directrices para la introducción de la sostenibilidad en el curriculum. https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sostenibilidad_Crue2012.pdf, 2024.