



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



## TFG del Grado en Ingeniería Informática

**Herramienta para la  
evaluación automática de  
entregas de programación con  
rúbricas y LLMs: IAGScore  
Documentación Técnica**



Presentado por Pedro Antonio Abellaneda  
Canales  
en Universidad de Burgos — 10 de junio  
de 2025  
Tutor: Raúl Marticorena Sánchez



---

# Índice general

---

<b>Índice general</b>	i
<b>Índice de figuras</b>	iii
<b>Índice de tablas</b>	v
<b>Apéndice A Plan de proyecto software</b>	1
A.1. Introducción . . . . .	1
A.2. Planificación temporal . . . . .	1
A.3. Estudio de viabilidad . . . . .	11
<b>Apéndice B Especificación de Requisitos</b>	17
B.1. Introducción . . . . .	17
B.2. Objetivos generales . . . . .	17
B.3. Catálogo de requisitos . . . . .	18
B.4. Especificación de requisitos . . . . .	19
<b>Apéndice C Especificación de diseño</b>	37
C.1. Introducción . . . . .	37
C.2. Diseño de datos . . . . .	37
C.3. Diseño arquitectónico . . . . .	39
C.4. Diseño procedimental . . . . .	47
<b>Apéndice D Documentación técnica de programación</b>	51
D.1. Introducción . . . . .	51
D.2. Estructura de directorios . . . . .	51
D.3. Manual del programador . . . . .	58

D.4. Compilación, instalación y ejecución del proyecto . . . . .	60
D.5. Pruebas del sistema . . . . .	67
<b>Apéndice E Documentación de usuario</b>	<b>71</b>
E.1. Introducción . . . . .	71
E.2. Requisitos de usuarios . . . . .	71
E.3. Instalación . . . . .	71
E.4. Manual del usuario . . . . .	72
<b>Apéndice F Anexo de sostenibilización curricular</b>	<b>83</b>
<b>Bibliografía</b>	<b>87</b>

---

# Índice de figuras

---

A.1. Gráfico <i>burndown</i> del <i>sprint 0</i> . . . . .	3
A.2. Gráfico <i>burndown</i> del <i>sprint 1</i> . . . . .	4
A.3. Gráfico <i>burndown</i> del <i>sprint 2</i> . . . . .	5
A.4. Gráfico <i>burndown</i> del <i>sprint 3</i> . . . . .	6
A.5. Gráfico <i>burndown</i> del <i>sprint 4</i> . . . . .	7
A.6. Gráfico <i>burndown</i> del <i>sprint 5</i> . . . . .	8
A.7. Gráfico <i>burndown</i> del <i>sprint 6</i> . . . . .	9
A.8. Gráfico <i>burndown</i> del <i>sprint 7</i> . . . . .	10
A.9. Gráfico <i>burndown</i> del <i>sprint 8</i> . . . . .	11
 B.1. Diagrama de casos de uso . . . . .	20
C.1. Diagrama E/R . . . . .	38
C.2. Diagrama Relacional . . . . .	39
C.3. Diagrama <i>MVT</i> de <i>Django</i> . . . . .	40
C.4. Diagrama de despliegue de la arquitectura Docker . . . . .	41
C.5. Diagrama de paquetes <i>accounts</i> . . . . .	42
C.6. Diagrama de paquetes <i>core</i> . . . . .	43
C.7. Diagrama de paquetes <i>prompts</i> . . . . .	44
C.8. Diagrama de paquetes <i>rubrics</i> . . . . .	44
C.9. Diagrama de paquetes <i>corrections</i> . . . . .	45
C.10. Diagrama de paquetes completo . . . . .	46
C.11. Diagrama de secuencias: crear nuevo <i>prompt</i> . . . . .	48
C.12. Diagrama de secuencia: crear nueva rúbrica . . . . .	48
C.13. Diagrama de secuencias: crear nueva corrección . . . . .	49
C.14. Diagrama de secuencias: ejecutar corrección . . . . .	50
 D.1. Imagen configuración de recursos Docker Desktop . . . . .	59

D.2. Imagen configuración unidad externa Docker Desktop . . . . .	60
D.3. Imagen clonación de repositorio . . . . .	61
D.4. Imagen copia variables de entorno . . . . .	62
D.5. Imagen construcción de contenedores . . . . .	63
D.6. Imagen levantando proyecto . . . . .	64
D.7. Imagen haciendo pull modelo Llama3.1 . . . . .	65
D.8. Imagen haciendo pull modelo Qwen3 . . . . .	65
D.9. Imagen mostrando lista de modelos cargados . . . . .	66
D.10.Imagen eliminando modelo . . . . .	66
D.11.Imagen mostrando ejecución de <i>test</i> . . . . .	69
D.12.Imagen mostrando resultado ejecución <i>test</i> . . . . .	69
E.1. Captura del botón de registro de usuario . . . . .	73
E.2. Captura de la ventana de registro de usuario . . . . .	74
E.3. Captura del mensaje de registro correcto . . . . .	74
E.4. Captura de la página de bienvenida . . . . .	75
E.5. Captura de la creación de un nuevo <i>prompt</i> . . . . .	76
E.6. Captura del <i>prompt</i> creado correctamente . . . . .	77
E.7. Captura del formulario de importación de una rúbrica . . . . .	78
E.8. Captura de la rúbrica creada correctamente . . . . .	79
E.9. Captura mostrando rúbrica formateada en <i>HTML</i> . . . . .	79
E.10.Captura mostrando el botón <b>Nueva corrección</b> . . . . .	80
E.11.Captura mostrando el formulario de configuración de corrección	80
E.12.Captura mostrando una corrección configurada . . . . .	81
E.13.Captura mostrando una corrección en ejecución . . . . .	81
E.14.Captura mostrando la opción de descarga . . . . .	82

---

# Índice de tablas

---

A.1. Costes de personal . . . . .	12
A.2. Costes de <i>hardware y software</i> . . . . .	12
A.3. Costes varios . . . . .	12
A.4. Costes totales . . . . .	13
A.5. Licencias de terceros . . . . .	14
B.1. CU-1 Registrar usuario. . . . .	21
B.2. CU-2 Login de usuario. . . . .	22
B.3. CU-3 Crear Prompts. . . . .	23
B.4. CU-4 Listar Prompts. . . . .	24
B.5. CU-5 Mostrar <i>Prompts</i> . . . . .	25
B.6. CU-6 Eliminar <i>Prompts</i> . . . . .	26
B.7. CU-7 Importar rúbrica. . . . .	27
B.8. CU-8 Listar rúbricas. . . . .	28
B.9. CU-9 Mostrar rúbrica. . . . .	29
B.10.CU-10 Eliminar rúbrica. . . . .	30
B.11.CU-11 Configurar corrección. . . . .	31
B.12.CU-12 Listar correcciones. . . . .	32
B.13.CU-13 Mostrar configuración. . . . .	33
B.14.CU-14 Eliminar corrección. . . . .	34
B.15.CU-15 Ejecutar corrección. . . . .	35
B.16.CU-16 Descargar corrección. . . . .	36
D.1. Relación entre pruebas y requisitos funcionales (RF) . . . . .	68



## *Apéndice A*

---

# **Plan de proyecto software**

---

## **A.1. Introducción**

En este apartado se detalla la planificación del proyecto, una etapa clave que implica definir los objetivos y el alcance, identificar los requisitos y recursos necesarios, establecer un cronograma con hitos clave, asignar tareas al equipo y prever posibles riesgos. Además, se analiza el coste en términos de tiempo, recursos y, aunque se trate de un proyecto académico, el coste económico y sus posibles beneficios. Con los resultados obtenidos, se elabora la planificación temporal y se evalúa la viabilidad del proyecto.

## **A.2. Planificación temporal**

En la gestión del proyecto se ha optado por seguir una metodología de desarrollo ágil *Scrum* con algunas salvedades, al participar una persona con la supervisión del tutor y según la guía *Scrum*: “*Scrum emplea un enfoque iterativo e incremental para optimizar la previsibilidad y controlar el riesgo. Scrum involucra a grupos de personas que colectivamente tienen todas las habilidades y experiencia para hacer el trabajo y compartir o adquirir tales habilidades según sea necesario.*” [4]

Para el desarrollo incremental del proyecto se han planificado una serie de *sprints* de dos semanas de duración, planificando al inicio de cada uno de ellos las tareas a desarrollar y revisando estas tareas a la finalización del mismo.

Para la gestión de los *sprints*, planificación y asignación de tareas se usó la herramienta *Zube* [3].

A continuación se muestran los *sprints* llevados a cabo durante las distintas etapas de este proyecto.

### **Sprint 0 (27/02/2025 hasta 6/03/2025)**

Se define un *sprint* inicial de una semana de duración en el que se realiza una primera aproximación a la propuesta de proyecto. En este *sprint* se realizan las siguientes tareas:

- Entrevista con tutor.
- Preparación de los entornos *Github*, *Zube*.
- Investigación sobre herramientas.
- Investigación sobre *LLMs*.
- Inicio de la documentación.
- Configuración del entorno de desarrollo.
- Definición de objetivos generales.
- Diseño de caso de uso genérico.

A continuación se muestra el gráfico *burndown* del *sprint*.

Figura A.1: Gráfico *burndown* del *sprint 0*

## Sprint 1 (7/03/2025 hasta 20/03/2025)

Segundo *sprint* en el que se realiza una planificación más específica de las tareas:

- Implementación de *Login*.
- Creación de un *mockup* del esqueleto de la página *Home*.
- Realización de pruebas con distintos *LLMs*.
- Revisión y actualización del caso de uso general.
- Diseño de diagramas ER.
- Personalización del modelo de usuario.
- Implementación del registro de usuarios.

Gráfico *burndown* del *sprint*.



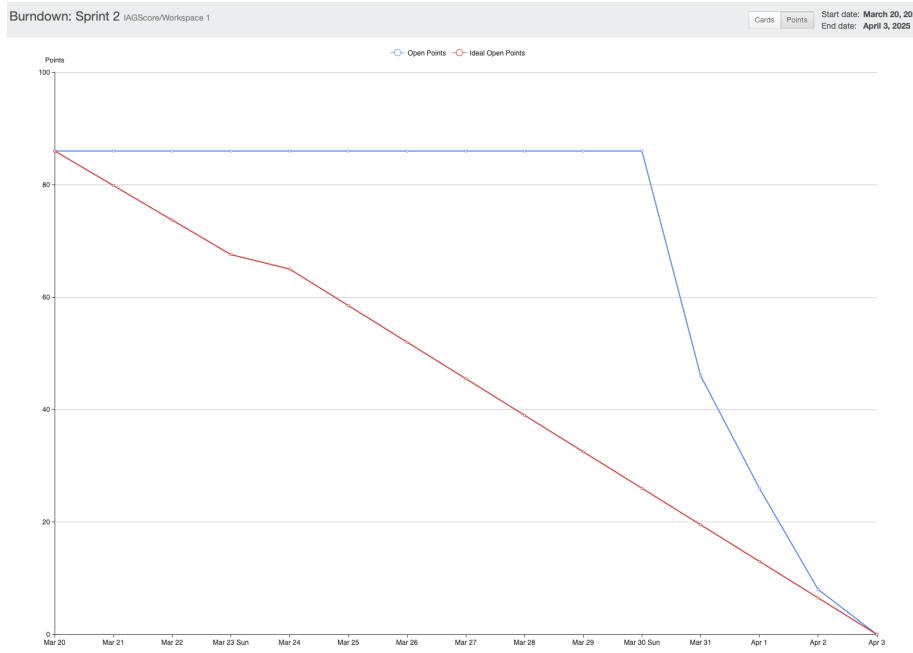
Figura A.2: Gráfico *burndown* del *sprint 1*

## **Sprint 2 (20/03/2025 hasta 3/04/2025)**

A continuación se muestran las tareas planificadas:

- Implementación de *Home page*.
- Implementación inicial de la sección rúbricas.
- Implementación inicial de la sección *prompts*.
- Integración de *SonarQube cloud*.
- Revisión de *Login* y Registro de usuarios.

Gráfico *burndown* del *sprint*.

Figura A.3: Gráfico *burndown* del *sprint* 2

### Sprint 3 (03/04/2025 hasta 16/04/2025)

A continuación se muestran las tareas planificadas:

- Implementación de modelo Correcciones.
- Implementación de formulario para el modelo.
- Implementación inicial de la carga de ficheros.
- Implementación de *test*.
- Integración de las vistas.
- Revisión de Rúbricas y *Prompts*.

Gráfico *burndown* del *sprint*.



Figura A.4: Gráfico *burndown* del *sprint 3*

### **Sprint 4 (16/04/2025 hasta 30/04/2025)**

Tareas planificadas:

- Investigación sobre integración de modelo en la aplicación.
- Integración de modelo en la aplicación.
- Investigación sobre implementación de respuesta asíncrona.
- Implementación de respuesta asíncrona.

Gráfico *burndown* del *sprint*.

Figura A.5: Gráfico *burndown* del *sprint 4*

## Sprint 5 (1/05/2025 hasta 15/05/2025)

Tareas planificadas para este *sprint*:

- Comprobación y generación de documentación.
- Lanzar *Release*.
- Modificación de orden del menú.
- Parametrización del modelo.
- Completar sección *LLM*.

Gráfico *burndown* del *sprint*.



Figura A.6: Gráfico *burndown* del *sprint* 5

## **Sprint 6 (16/05/2025 hasta 29/05/2025)**

A continuación se muestran las tareas planificadas:

- Internacionalización de la aplicación.
- Añadir cabecera con campo de búsqueda en las tablas.
- Revisar documentación generada con *Sphinx*.
- Avanzar documentación de la memoria.

Gráfico *burndown* del *sprint*.

Figura A.7: Gráfico *burndown* del *sprint 6*

### Sprint 7 (30/05/2025 hasta 05/06/2025)

Tareas planificadas:

- Revisión de proyecto.
- Avance de anexos.
- Avanzar memoria.

Gráfico *burndown* del *sprint*.

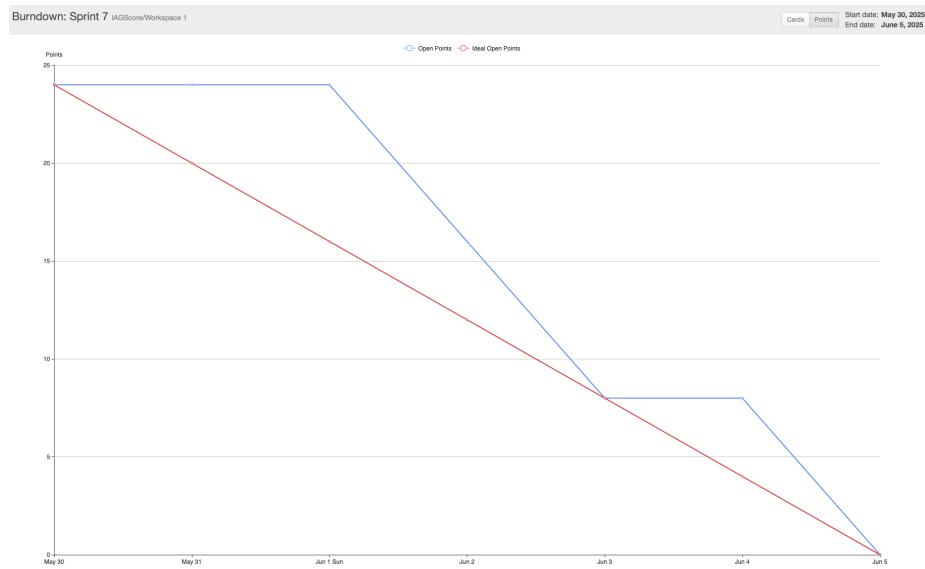


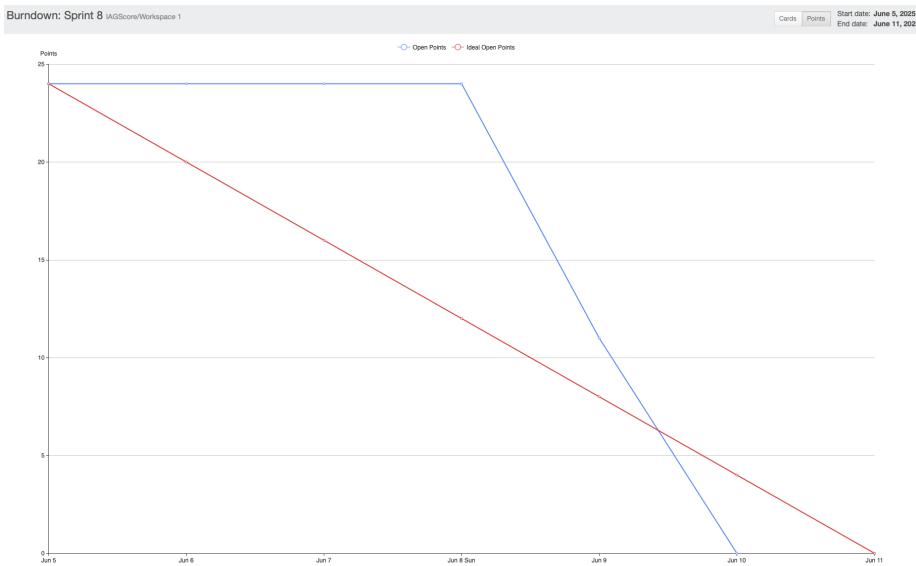
Figura A.8: Gráfico *burndown* del *sprint 7*

### **Sprint 8 (5/06/2025 hasta 11/06/2025)**

Tareas planificadas:

- Preparar presentación.
- Actualizar proyecto.
- Grabar videos.
- Revisar documentación.

Gráfico *burndown* del *sprint*.

Figura A.9: Gráfico *burndown* del *sprint 8*

## A.3. Estudio de viabilidad

### Viabilidad económica

En este apartado se realiza una estimación de los costes del proyecto, lo más aproximada posible a un entorno empresarial, y se analizan los posibles beneficios en caso de que el proyecto se comercialice.

Los costes principales son de personal, junto con *hardware*, *software* y, si aplica, infraestructura.

### Estudio de costes

El proyecto ha tenido una duración de cuatro meses y se ha desarrollado por una persona, incluye formación, implementación y documentación.

Teniendo en cuenta el salario medio de un programador junior en España [5] y considerando ausencia de deducciones tributarias.

Concepto	Coste anual	Prorratoe (4 meses)
<i>Salario bruto</i>	21.000,00€	7.000,00€
<i>Retención IRPF (17 %)</i>	3.570,00€	1.190,00€
<i>Seguridad Social (7 %)</i>	1.220,10€	406,70€
<i>Salario neto</i>	16.209,90€	5.403,30€

Tabla A.1: Costes de personal

A continuación, se detallan los costes asociados al *hardware* y *software* utilizados en el proyecto. Para su desarrollo, se empleó un *MacBook Pro*, cuyo sistema operativo (macOS) cuenta con una licencia incluida, y como entorno de desarrollo integrado (*IDE*) se utilizó *Visual Studio Code*, el cual es gratuito.

Dado que el proyecto tiene una duración de 4 meses, se ha aplicado un criterio de amortización proporcional del *hardware*. En concreto, se ha considerado una vida útil del equipo de 3 años, imputando únicamente el coste correspondiente a los 4 meses de uso.

Concepto	Coste total	Coste amortizado
<i>Ordenador personal</i>	1.500,00 €	166,67 €
<i>macOS</i>	0,00 €	0,00 €
<i>Visual Studio Code</i>	0,00 €	0,00 €

Tabla A.2: Costes de *hardware* y *software*

También se consideran otros costes, como el consumo eléctrico y otros gastos operativos derivados del desarrollo y ejecución del proyecto, tales como servicios de conectividad y material de oficina, entre otros.

Concepto	Coste mensual	Prorratoe (4 meses)
<i>Consumo eléctrico</i>	100,00 €	400,00 €
<i>Internet</i>	30,00 €	120,00 €
<i>Material de oficina</i>	30,00 €	30,00€

Tabla A.3: Costes varios

Los costes estimados totales son:

Concepto	Coste
<i>Costes de personal</i>	7.000,00 €
<i>Costes hardware y software</i>	166,67 €
<i>Costes varios</i>	550,00 €
<b>TOTAL</b>	7.716,67 €

Tabla A.4: Costes totales

## Análisis de beneficios

Este proyecto no se plantea inicialmente como un mecanismo para generar beneficios económicos, sino como una herramienta gratuita destinada a facilitar la corrección y evaluación automática de tareas, especialmente en el ámbito de la programación.

No obstante, en caso de considerar su monetización, existen diversas vías viables. Una opción sería incluir publicidad contextual no intrusiva en la interfaz web, lo cual permitiría generar ingresos suficientes para cubrir los costes operativos del proyecto. Otra posibilidad consiste en ofrecer servicios personalizados bajo demanda, como adaptaciones específicas del sistema para centros educativos o empresas de formación técnica. Asimismo, se podría licenciar el uso de la plataforma a instituciones interesadas en integrar esta herramienta en sus propios entornos virtuales de aprendizaje, ya sea mediante *APIs* o despliegues locales.

A continuación se presenta una estimación aproximada de cuánto podría cobrarse en cada caso:

1. **Publicidad contextual no intrusiva:** la integración de anuncios discretos en la interfaz web podría generar ingresos complementarios. Estos ingresos pueden variar ampliamente según el tráfico.
2. **Servicios personalizados bajo demanda:** la adaptación del sistema a necesidades específicas de centros educativos o entidades formativas podría suponer ingresos entre **500 y 2.000€ por proyecto**. Con apenas **2 o 3 contrataciones anuales**, se cubriría una parte significativa de los costes de desarrollo y soporte técnico.

3. **Licencias institucionales:** ofrecer el sistema como un servicio licenciado a instituciones interesadas (con acceso vía *API* o instalación local) permitiría ingresos más estables. Una tarifa anual por uso institucional podría oscilar entre **1.000 y 3.000€**, dependiendo del grado de personalización y soporte requerido.

## Viabilidad legal

### Licencias de *software*

En este apartado se detallan las licencias de las herramientas y bibliotecas de software utilizadas durante el desarrollo del proyecto. Asimismo, se asignará una licencia al propio proyecto.

En la tabla siguiente se enumeran las herramientas y bibliotecas de software empleadas, junto con sus respectivas licencias. Solo se incluyen aquellas que forman parte del entorno de ejecución del sistema o que desempeñan un papel técnico relevante.

Herramienta	Licencia
<i>Django</i>	BSD
<i>djangorestframework</i>	BSD
<i>python-decouple</i>	MIT
<i>psycopg2-binary</i>	BSD-style
<i>sqlparse</i>	BSD
<i>langchain</i>	MIT
<i>langchain-community</i>	MIT
<i>langchain-ollama</i>	MIT
<i>ollama</i>	MIT
<i>celery</i>	BSD
<i>redis</i>	BSD (v5.2.1)
<i>sphinx</i>	BSD
<i>sphinx-book-theme</i>	MIT
<i>Tailwind CSS</i>	MIT
<i>Flowbite</i>	MIT

Tabla A.5: Licencias de terceros

Todas las licencias listadas son de tipo permisivo (BSD o MIT) [7, 8], lo que garantiza su compatibilidad con la licencia **MIT** seleccionada para este proyecto.

Para la licencia de este proyecto se ha optado por la *MIT License*, una licencia de *software* libre y permisiva que permite a los usuarios utilizar, copiar, modificar y distribuir el código de manera sencilla y flexible. Esta elección facilita la colaboración y la adopción del *software* en diferentes entornos educativos y profesionales.

Además, la *MIT License* es compatible con todas las licencias de las herramientas y bibliotecas empleadas en el desarrollo del proyecto, lo que garantiza la coherencia y legalidad en el uso conjunto de estas tecnologías.



## *Apéndice B*

---

# Especificación de Requisitos

---

## B.1. Introducción

En este anexo se describen los servicios que debe ofrecer la aplicación, para la configuración de correcciones y la evaluación de tareas, principalmente de programación. También se detallan las restricciones asociadas a su funcionamiento. La función principal de esta especificación de requisitos es servir como medio de comunicación claro y preciso entre clientes, usuarios y desarrolladores.

Se incluirán tanto los requisitos funcionales como los no funcionales del sistema.

## B.2. Objetivos generales

Este trabajo se ha realizado persiguiendo el cumplimiento de los siguientes **objetivos:**

- Desarrollar un sistema *software* que permita a los usuarios la corrección de ejercicios de programación.
- El usuario debe estar *registrado* para acceder al sistema iniciando sesión.
- Los usuarios *importarán* sus propias rúbricas en formato *Markdown*.
- Las rúbricas importadas en *Markdown* serán visibles en formato *HTML*.

- Los usuarios crearán sus propios *prompts*.
- La evaluación de los ejercicios se realizará mediante modelos de lenguaje *LLM*.
- Los usuarios podrán configurar aspectos determinados del *LLM* usado.

### B.3. Catálogo de requisitos

Requisitos funcionales y no funcionales para cumplir con los objetivos generales.

#### Requisitos funcionales

- **RF-1 - Registro de usuarios.** Los usuarios podrán registrarse mediante nombre, correo electrónico y contraseña.
- **RF-2 - Login de usuarios.** Los usuarios deberán iniciar sesión para usar la aplicación.
- **RF-3 Crear *prompts*.** Los usuarios crearán sus propios *prompts*.
  - **RF-3.1 - Listar *prompts*.** Los usuarios podrán consultar sus *prompts* creados.
  - **RF-3.2 - Mostrar *prompt*.** Los usuarios podrán visualizar un *prompt* concreto.
  - **RF-3.3 - Eliminar *prompt*.** Los usuarios podrán eliminar un *prompt*.
- **RF-4 - Importar rúbricas.** Los usuarios podrán importar sus propias rúbricas en formato *Markdown*.
  - **RF-4.1 - Listar rúbricas.** Los usuarios podrán consultar sus rúbricas importadas.
  - **RF-4.2 - Mostrar rúbrica.** Los usuarios podrán visualizar los detalles de una rúbrica concreta en formato *HTML*.
  - **RF-4.3 - Eliminar rúbrica.** Los usuarios podrán eliminar una rúbrica importada.
- **RF-5 Configurar correcciones.** Los usuarios configurarán las correcciones.

- **RF-5.1 - Listar correcciones.** Los usuarios podrán consultar sus correcciones configuradas.
- **RF-5.2 - Mostrar configuración.** Los usuarios podrán visualizar los detalles de configuración de una corrección.
- **RF-5.3 - Eliminar corrección.** Los usuarios podrán eliminar una corrección.
- **RF-5.4 - Ejecutar corrección.** Los usuarios podrán ejecutar una corrección.
- **RF-5.5 - Descargar corrección.** Los usuarios podrán descargar el resultado de la corrección ejecutada.

## B.4. Especificación de requisitos

A continuación se muestran los casos de uso contemplados para el desarrollo del proyecto

## Casos de uso

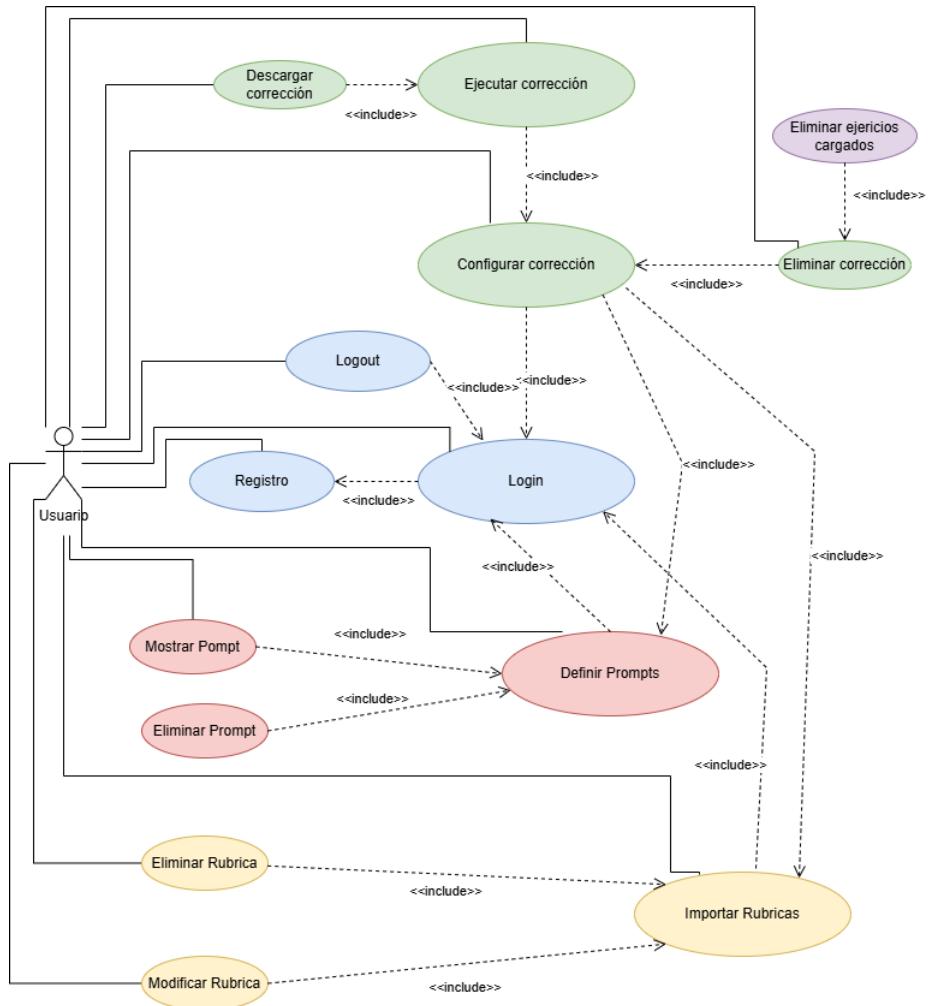


Figura B.1: Diagrama de casos de uso

<b>CU-1</b>	<b>Registro de usuario</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-1
<b>Descripción</b>	Permite registrarse a un usuario
<b>Precondición</b>	El usuario no puede estar <i>logeado</i>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario entra en la aplicación y accede a la página de registro</li> <li>2. El usuario introduce su email y una contraseña</li> <li>3. El usuario confirma el registro con el botón registrar</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la página de <i>login</i>.</li> <li>■ Mensaje: Usuario creado correctamente.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Mensaje: Ya existe Usuario con este Email.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.1: CU-1 Registrar usuario.

---

CU-2	<i>Login</i> de usuario
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-2
<b>Descripción</b>	Permite <i>logearse</i> a un usuario
<b>Precondición</b>	El usuario debe estar registrado
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario entra en la aplicación y accede a la página de <i>login</i>.</li> <li>2. El usuario introduce su <i>email</i> y una contraseña.</li> <li>3. El usuario confirma pulsando el botón entrar.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la página <i>home</i>.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Mensaje: Usuario o contraseña incorrectos</li> </ul>
<b>Importancia</b>	Alta

---

Tabla B.2: CU-2 Login de usuario.

<b>CU-3</b>	<b>Crear <i>Prompts</i></b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-3
<b>Descripción</b>	Permite crear un <i>prompt</i> a un usuario
<b>Precondición</b>	El usuario debe estar <i>logeado</i>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de <i>prompts</i> y pulsa botón nuevo <i>prompt</i>.</li> <li>2. El usuario introduce un nombre.</li> <li>3. El usuario introduce el texto del <i>prompt</i>.</li> <li>4. El usuario pulsa botón importar.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirecciona a la página de <i>prompts</i></li> <li>■ Mensaje: <i>Prompt</i> creado correctamente</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Mensaje: Error al guardar el prompt: Prompt ya existente</li> </ul>
<b>Importancia</b>	Alta

Tabla B.3: CU-3 Crear Prompts.

---

CU-4	Listar <i>Prompts</i>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-3.1
<b>Descripción</b>	Permite listar los <i>prompts</i> de un usuario
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar logeado.</li> <li>■ El usuario debe haber creado algún <i>prompt</i>.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de prompts.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se muestra la tabla con nombre y fecha de creación y acciones disponibles de cada una de los prompts creados.</li> </ul>
<b>Excepciones</b>	
<b>Importancia</b>	Alta

---

Tabla B.4: CU-4 Listar Prompts.

CU-5	Mostrar <i>Prompt</i>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-3.2
<b>Descripción</b>	Permite mostrar un <i>prompt</i> concreto
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar logeado.</li> <li>■ El usuario debe haber creado el <i>prompt</i>.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de <i>prompts</i>.</li> <li>2. El usuario busca en la tabla el <i>prompt</i> que deseé.</li> <li>3. El usuario pulsa el botón Mostrar del <i>prompt</i> que desea visualizar.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se muestra el texto del <i>prompt</i>.</li> <li>■ Se muestran botones para volver o eliminar.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Error 404 – Prompt no encontrado.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.5: CU-5 Mostrar *Prompts*.

CU-6	Eliminar <i>Prompt</i>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-3.3
<b>Descripción</b>	Permite eliminar un <i>prompt</i> concreto
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe haber creado el <i>prompt</i>.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. Opción 1 - Sin visualizar contenido:             <ol style="list-style-type: none"> <li>a) El usuario accede a la sección de <i>prompts</i>.</li> <li>b) El usuario busca en la tabla el <i>prompt</i> que desee.</li> <li>c) El usuario pulsa el botón eliminar.</li> <li>d) El usuario confirma la opción eliminar.</li> </ol> </li> <li>2. Opción 2 - Visualizando contenido:             <ol style="list-style-type: none"> <li>a) El usuario accede a la sección de <i>prompts</i>.</li> <li>b) El usuario busca en la tabla el <i>prompt</i> que desee.</li> <li>c) El usuario pulsa el botón mostrar.</li> <li>d) El usuario visualiza el contenido del <i>prompt</i>.</li> <li>e) El usuario pulsa el botón eliminar.</li> <li>f) El usuario confirma que desea eliminar.</li> </ol> </li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ El <i>prompt</i> es eliminado de la base de datos.</li> <li>■ La aplicación redirige a la vista de <i>prompts</i>.</li> <li>■ Mensaje: Prompt eliminado correctamente.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Error 404 - Prompt no encontrado</li> </ul>
<b>Importancia</b>	Alta

Tabla B.6: CU-6 Eliminar *Prompts*.

<b>CU-7</b>	<b>Importar rúbrica</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-4
<b>Descripción</b>	Permite importar una rúbrica a un usuario
<b>Precondición</b>	El usuario debe estar autenticado
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de rúbricas y pulsa botón nueva rúbrica.</li> <li>2. El usuario introduce un nombre para la rúbrica.</li> <li>3. El usuario importa una rúbrica en formato Markdown.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la página de vista de rúbricas importadas.</li> <li>■ Mensaje: Rúbrica importada correctamente.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Mensaje: Error al guardar la rúbrica: Rúbrica ya existente.</li> <li>■ Mensaje: El archivo debe estar codificado en UTF-8.</li> <li>■ Mensaje: El archivo subido no es un archivo Markdown.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.7: CU-7 Importar rúbrica.

<b>CU-8</b>	<b>Listar rúbricas</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-4.1
<b>Descripción</b>	Permite ver las rúbricas del usuario
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe haber importado una rúbrica.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de rúbricas</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se muestra la tabla con nombre, fecha de creación y acciones para cada una de las rúbricas importadas.</li> </ul>
<b>Excepciones</b>	
<b>Importancia</b>	Alta

Tabla B.8: CU-8 Listar rúbricas.

<b>CU-9</b>	<b>Mostrar rúbrica</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-4.2
<b>Descripción</b>	Permite mostrar una rúbrica a un usuario
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe haber importado una rúbrica.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de rúbricas.</li> <li>2. El usuario busca en la tabla la rúbrica que deseé.</li> <li>3. El usuario pulsa el botón Mostrar de la rúbrica que deseé.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la página de vista de una rúbrica concreta.</li> <li>■ se muestra el contenido de la rúbrica en formato <i>HTML</i>.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Error 404 - Rúbrica no encontrada.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.9: CU-9 Mostrar rúbrica.

CU-10	Eliminar rúbrica
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-4.3
<b>Descripción</b>	Permite eliminar una rúbrica a un usuario
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe haber importado una rúbrica.</li> </ul>
<b>Acciones</b>	
<p>1. Opción 1 - Sin visualizar contenido:</p> <ol style="list-style-type: none"> <li>a) El usuario accede a la sección de rúbricas.</li> <li>b) El usuario busca en la tabla la rúbrica que desee.</li> <li>c) El usuario pulsa el botón eliminar.</li> <li>d) El usuario confirma la opción eliminar.</li> </ol> <p>2. Opción 2 - Visualizando contenido:</p> <ol style="list-style-type: none"> <li>a) El usuario accede a la sección de rúbricas.</li> <li>b) El usuario busca en la tabla la rúbrica que desee.</li> <li>c) El usuario pulsa el botón mostrar.</li> <li>d) El usuario visualiza el contenido de la rúbrica.</li> <li>e) El usuario pulsa el botón eliminar.</li> <li>f) El usuario confirma que desea eliminar.</li> </ol>	
<b>Postcondición</b>	
<ul style="list-style-type: none"> <li>■ La rúbrica es eliminada de la base de datos.</li> <li>■ La aplicación redirige a la vista de rúbricas.</li> <li>■ Mensaje: Rúbrica eliminada correctamente.</li> </ul>	
<b>Excepciones</b>	
<ul style="list-style-type: none"> <li>■ Error 404 - Rúbrica no encontrada.</li> </ul>	
<b>Importancia</b>	Alta

Tabla B.10: CU-10 Eliminar rúbrica.

CU-11	Configurar corrección
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-5
<b>Descripción</b>	Permite configurar una corrección a un usuario
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe haber creado un <i>prompt</i>.</li> <li>■ El usuario debe haber importado una rúbrica.</li> <li>■ Debe de existir un modelo cargado en local.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de Correcciones.</li> <li>2. El usuario pulsa botón Nueva corrección.</li> <li>3. El usuario selecciona un <i>prompt</i> de la lista.</li> <li>4. El usuario selecciona una rúbrica de la lista.</li> <li>5. El usuario carga un fichero comprimido con las tareas.</li> <li>6. El usuario introduce un nombre.</li> <li>7. El usuario selecciona un modelo de la lista.</li> <li>8. El usuario configura los parámetros del modelo.</li> <li>9. El usuario pulsa el botón Guardar configuración.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la vista de correcciones.</li> <li>■ Mensaje: Corrección creada correctamente.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la vista de nueva corrección.</li> <li>■ Mensaje: Error al crear la corrección.</li> <li>■ Mensaje: Error en el campo Prompt: Este campo es requerido.</li> <li>■ Mensaje: Error en el campo Rúbrica: Este campo es requerido.</li> <li>■ Mensaje: Error en el campo Modelo: Este campo es requerido.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.11: CU-11 Configurar corrección.

<b>CU-12</b>	<b>Listar correcciones</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-5.1
<b>Descripción</b>	Permite visualizar las correcciones configuradas de un usuario
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe de haber configurado una dirección.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de Correcciones.</li> <li>2. El usuario pulsa botón Ver correcciones.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se muestra la tabla con los campos y acciones disponibles para cada corrección.</li> </ul>
<b>Excepciones</b>	
<b>Importancia</b>	Alta

Tabla B.12: CU-12 Listar correcciones.

<b>CU-12</b>	<b>Mostrar configuración</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-5.2
<b>Descripción</b>	Permite visualizar los datos de configuración de una corrección
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe de haber configurado una corrección.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de Correcciones.</li> <li>2. El usuario pulsa botón Ver correcciones.</li> <li>3. El usuario busca la corrección que desee.</li> <li>4. El usuario pulsa botón Mostrar configuración, de la corrección que desee.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se muestran los datos relacionados con la corrección seleccionada.</li> </ul>
<b>Excepciones</b>	
<b>Importancia</b>	Alta

Tabla B.13: CU-13 Mostrar configuración.

<b>CU-14</b>	<b>Eliminar corrección</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-5.3
<b>Descripción</b>	Permite eliminar una corrección.
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe de haber configurado una corrección.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de Correcciones.</li> <li>2. El usuario pulsa botón Ver correcciones.</li> <li>3. El usuario busca la corrección que desee.</li> <li>4. El usuario pulsa botón Eliminar, de la corrección que desee.</li> <li>5. El usuario confirma que desea eliminar.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ La aplicación redirige a la vista de correcciones.</li> <li>■ Se eliminan los archivos relacionados con la corrección.</li> <li>■ Mensaje: Corrección: &lt;descripción&gt;, eliminada correctamente.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Error 404 - Corrección no encontrada.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.14: CU-14 Eliminar corrección.

<b>CU-15</b>	<b>Ejecutar corrección</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-5.4
<b>Descripción</b>	Permite ejecutar una corrección.
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe de haber configurado una corrección.</li> <li>■ El modelo seleccionado para la corrección debe estar cargado en local.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de Correcciones.</li> <li>2. El usuario pulsa botón Ver correcciones.</li> <li>3. El usuario busca la corrección que desee.</li> <li>4. El usuario pulsa botón Ejecutar, de la corrección que desee.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se envían los datos y las tareas de la corrección ejecutada al modelo</li> <li>■ Mensaje: Corrección en proceso. Revisa el resultado más tarde.</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Mensaje: El modelo seleccionado para esta corrección no está cargado en Ollama.</li> </ul>
<b>Importancia</b>	Alta

Tabla B.15: CU-15 Ejecutar corrección.

<b>CU-16</b>	<b>Descargar resultado</b>
<b>Versión</b>	1.0
<b>Autor</b>	Pedro Antonio Abellaneda Canales
<b>Requisitos asociados</b>	RF-5.5
<b>Descripción</b>	Permite ejecutar una corrección.
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ El usuario debe estar autenticado.</li> <li>■ El usuario debe de haber configurado una corrección.</li> <li>■ El usuario debe de haber ejecutado una corrección.</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la sección de Correcciones.</li> <li>2. El usuario pulsa botón Ver correcciones.</li> <li>3. El usuario busca la corrección que desee.</li> <li>4. El usuario pulsa botón Descargar último resultado, de la corrección que desee.</li> </ol>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Se descarga el resultado de la última corrección ejecutada.</li> </ul>
<b>Excepciones</b>	
<b>Importancia</b>	Alta

Tabla B.16: CU-16 Descargar corrección.

## *Apéndice C*

---

# Especificación de diseño

---

## C.1. Introducción

En esta sección se describe el proceso de diseño de la aplicación *IAGScore*, orientado a dar cumplimiento a los objetivos y requisitos definidos con anterioridad. Se especifican los datos que utilizará la aplicación, así como su estructura procedimental y la arquitectura del sistema.

## C.2. Diseño de datos

En este proyecto se han desarrollado seis aplicaciones, organizadas con el objetivo de separar de forma coherente la lógica y las funcionalidades del sistema, de acuerdo con los resultados esperados. Existe una aplicación principal denominada *iagscore*, encargada de la gestión y configuración general del proyecto, y otras cinco aplicaciones denominadas *accounts*, *core*, *prompts*, *rubrics* y *corrections*.

El diagrama entidad-relación que se presenta a continuación refleja las entidades y relaciones que forman parte del modelo de datos persistente de la aplicación. En este sentido, se han incluido únicamente los modelos propios definidos en las aplicaciones que generan tablas en la base de datos: *accounts*, *prompts*, *rubrics* y *corrections*.

Las aplicaciones *iagscore* y *core*, aunque esenciales para la configuración, organización y funcionamiento general del sistema, no aportan entidades directamente al esquema relacional, ya que no definen modelos. Por este motivo, no se representan en el diagrama, aunque su funcionalidad es fundamental para la arquitectura global del proyecto.

## Diagrama E/R

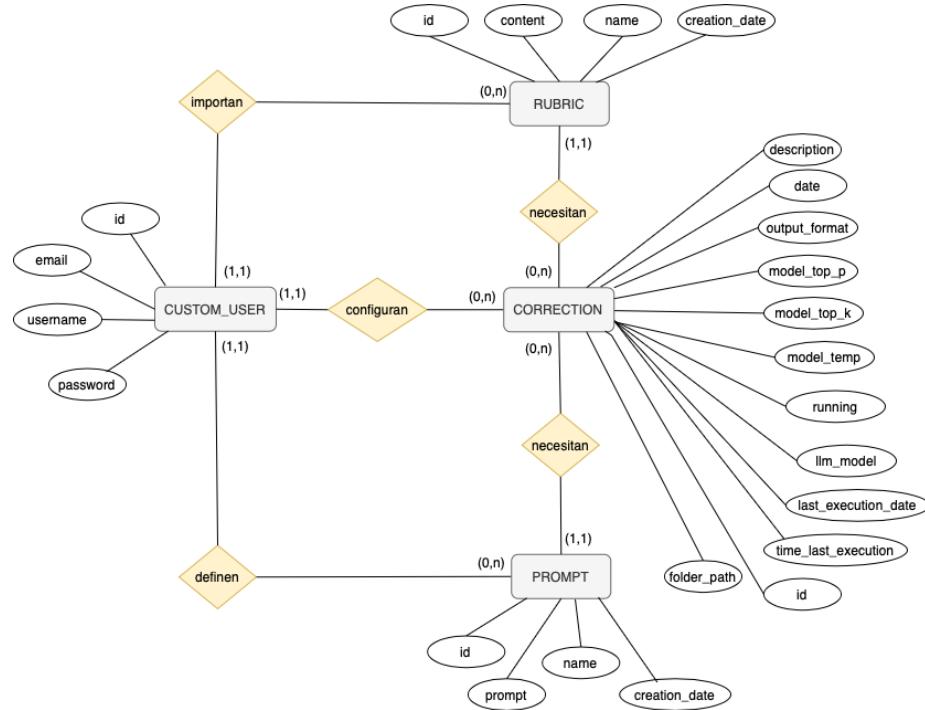


Figura C.1: Diagrama E/R

## Diagrama relacional

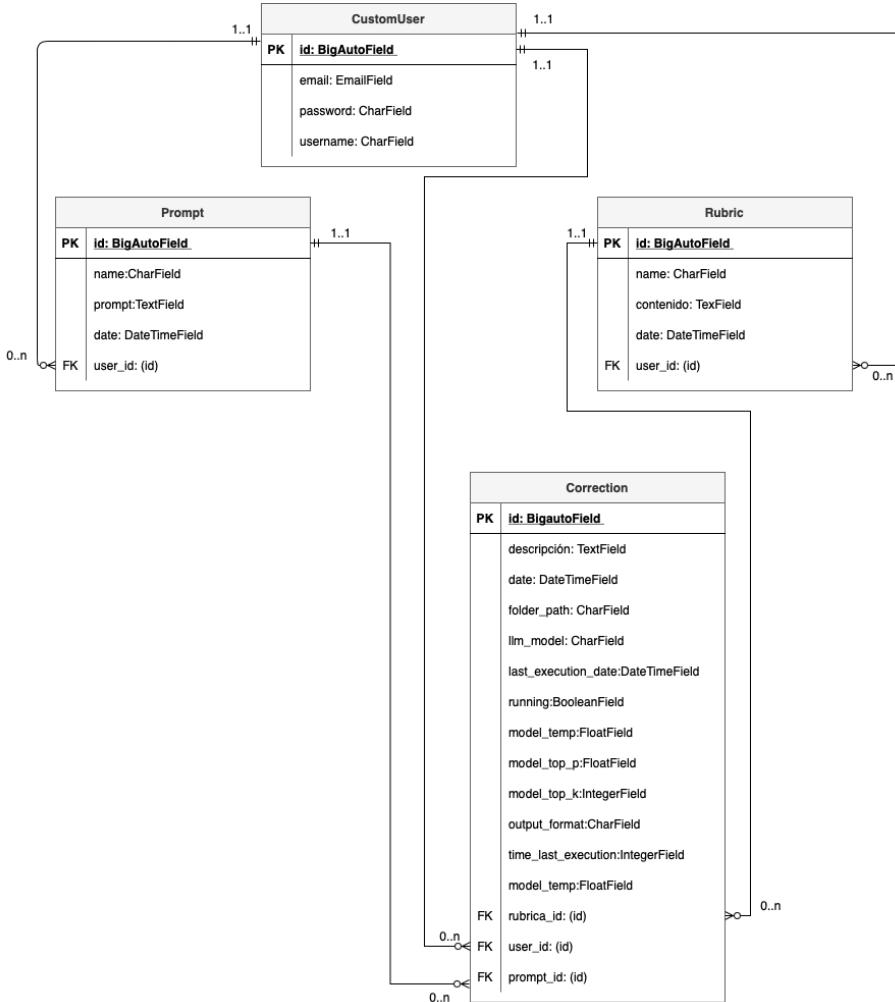


Figura C.2: Diagrama Relacional

## C.3. Diseño arquitectónico

*Django* permite acelerar el desarrollo de aplicaciones web seguras y escalables gracias a su enfoque *batteries-included* [2], que incluye componentes como un *ORM*, sistema de autenticación, administración automática y soporte nativo para pruebas.

El sistema está construido siguiendo el patrón arquitectónico *Modelo-Vista-Template (MVT)*, propio de *Django*, que permite una separación

clara de responsabilidades y facilita el mantenimiento y escalabilidad de la aplicación.

En este patrón:

- **Modelo (Model)**: define la estructura de datos y la lógica para acceder y manipular la base de datos.
- **Vista (View)**: gestiona la lógica de negocio, procesa las peticiones del usuario y devuelve respuestas, normalmente utilizando plantillas para generar la presentación.
- **Plantilla (Template)**: define cómo se presenta la información, generando el contenido *HTML* que la vista devuelve al usuario.

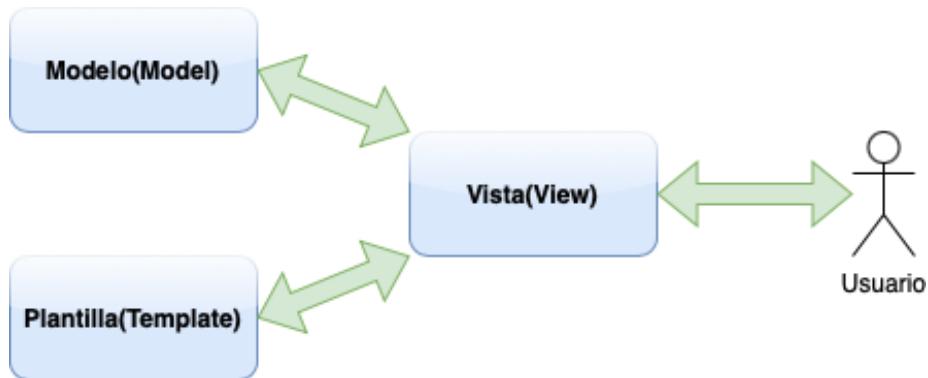


Figura C.3: Diagrama *MVT* de *Django*

Sobre esta base, el sistema se despliega en una arquitectura de microservicios contenerizados usando *Docker*. Esta estructura permite el despliegue y escalado sencillo de los distintos componentes que conforman la aplicación, así como su separación lógica y técnica.

A continuación, se describen los principales servicios definidos en el archivo `docker-compose.yaml`:

- ***web***: contenedor principal que ejecuta el servidor de *Django*. Es el punto de entrada del sistema y gestiona tanto las peticiones de los usuarios como la lógica del negocio.
- ***postgres\_db***: servicio de base de datos relacional utilizando *PostgreSQL*. Gestiona el almacenamiento persistente de la información del sistema, incluyendo usuarios, correcciones, rúbricas, *prompts*, etc.

- **redis**: sistema utilizado como *backend* para la gestión de la cola de tareas asíncronas con *Celery*.
- **celery**: servicio de ejecución en segundo plano para tareas intensivas o que requieren tiempo de procesamiento prolongado. Utiliza *Redis* como intermediario y permite delegar tareas como la corrección con *LLM*.
- **ollama**: servicio de modelos de lenguaje basado en la imagen oficial de *Ollama*. Permite interactuar con modelos *LLM* (*Large Language Models*) directamente dentro de la infraestructura del sistema, sin necesidad de servicios externos.

Todos los servicios están conectados mediante una red interna personalizada (app-network), que garantiza la comunicación segura y eficiente entre ellos. Se utilizan volúmenes para asegurar la persistencia de datos relevantes tareas cargadas por los usuarios o las respuestas del modelo *LLM*.

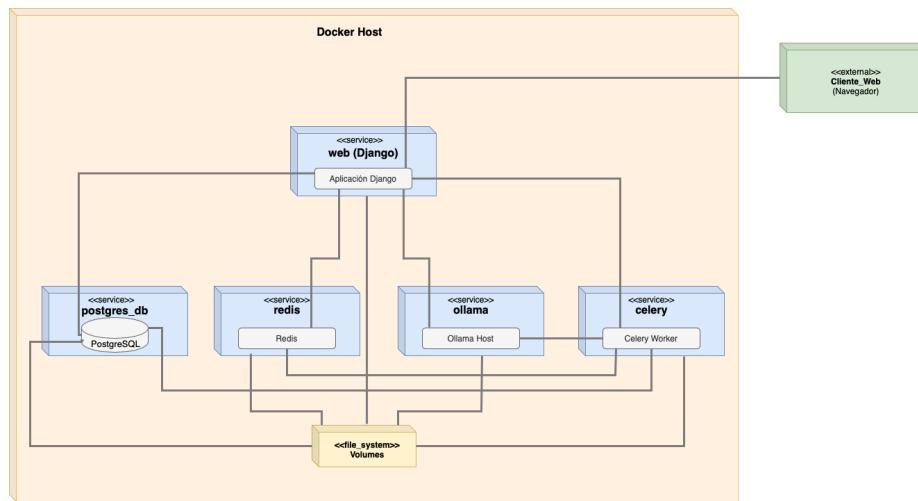


Figura C.4: Diagrama de despliegue de la arquitectura Docker

## Estructura de Paquetes del Sistema

El sistema ha sido desarrollado como una única aplicación web utilizando el framework **Django**.

Se ha seguido un **enfoque basado en características funcionales**, organizando el sistema en paquetes que agrupan todos los elementos rela-

cionados con una misma funcionalidad (como autenticación, correcciones, rúbricas, etc.).

Esta elección arquitectónica promueve una **alta cohesión** dentro de cada paquete y un **bajo acoplamiento** entre ellos, en línea con los principios del diseño modular.

Los distintos diagramas de paquetes incluidos a continuación reflejan esta organización, permitiendo visualizar la estructura jerárquica interna y las dependencias funcionales de cada una de las aplicaciones que componen el sistema.

### Aplicación *accounts*

La aplicación *accounts* gestiona el modelo de usuario personalizado. Su diagrama de paquetes muestra la estructura interna de los módulos relacionados con la gestión de usuarios.

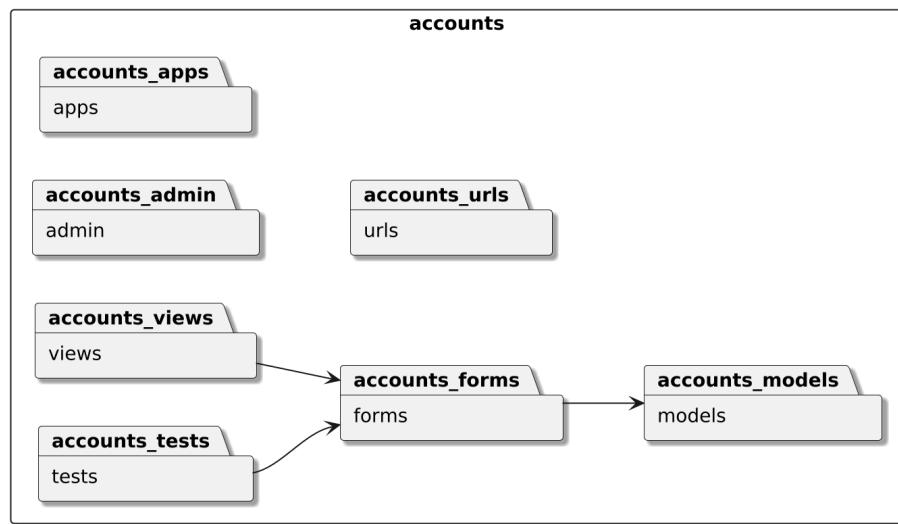
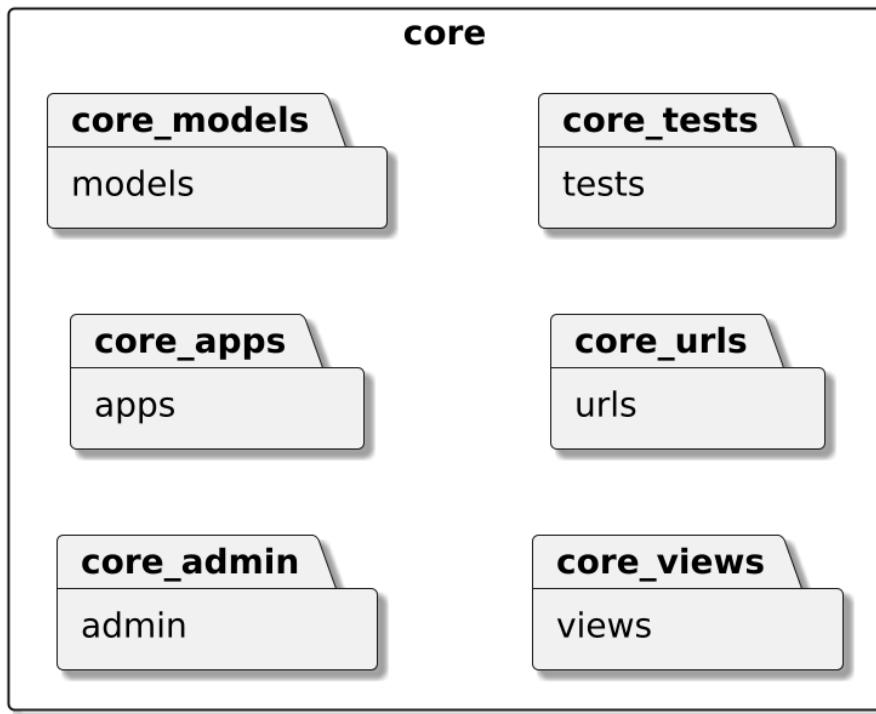


Figura C.5: Diagrama de paquetes *accounts*

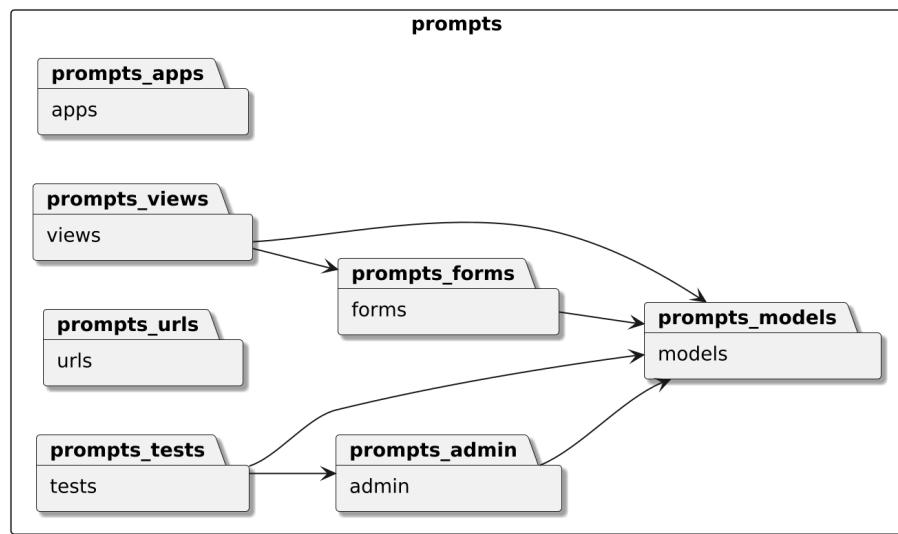
### Aplicación *core*

La aplicación *core* contiene componentes compartidos por el resto del sistema y la configuración del sistema de autenticación.

Figura C.6: Diagrama de paquetes *core*

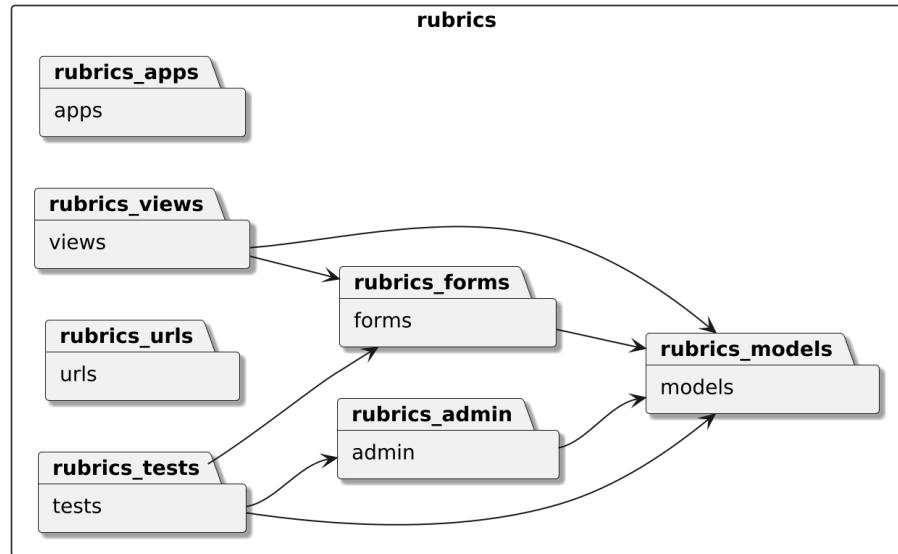
### Aplicación *prompts*

La aplicación *prompts* se encarga de la creación, edición y visualización de los *prompts*. El siguiente diagrama muestra su estructura modular y los componentes principales implicados.

Figura C.7: Diagrama de paquetes *prompts*

### Aplicación *rubrics*

La aplicación *rubrics* gestiona las rúbricas utilizadas en los procesos de corrección. Su organización interna se detalla en el siguiente diagrama de paquetes.

Figura C.8: Diagrama de paquetes *rubrics*

### Aplicación *corrections*

La aplicación *corrections* es responsable de la configuración y ejecución de las correcciones automáticas, integrando *prompts*, rúbricas, señales y tareas asíncronas. El siguiente diagrama muestra la disposición de sus módulos funcionales.

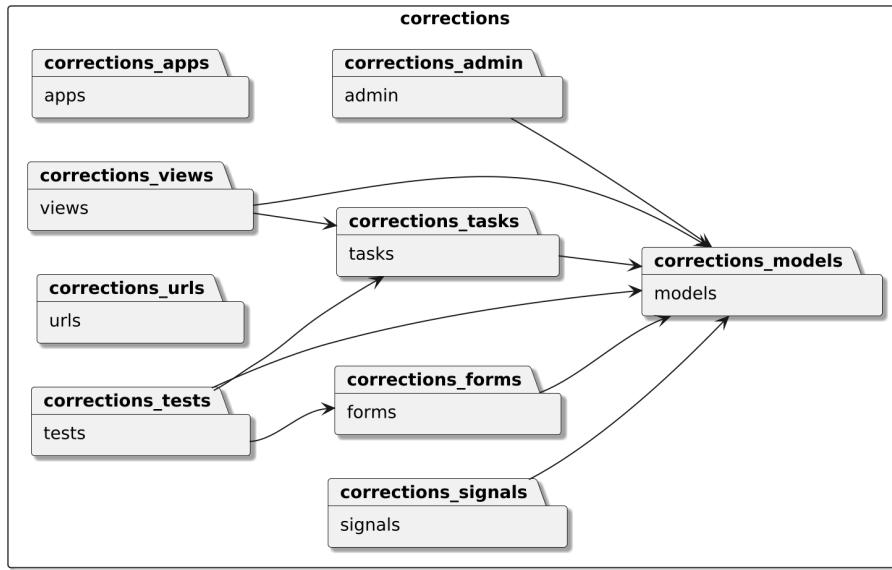


Figura C.9: Diagrama de paquetes *corrections*

El siguiente diagrama proporciona una visión general de la estructura de paquetes del sistema, mostrando las aplicaciones principales y sus interacciones.

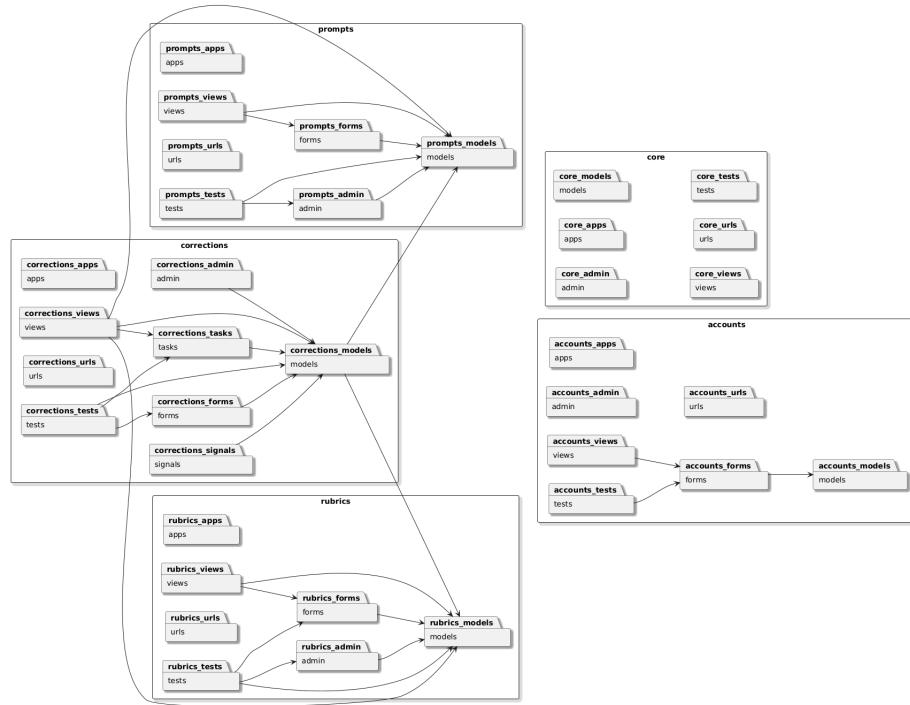


Figura C.10: Diagrama de paquetes completo

## Gestión de la Persistencia con el ORM de Django

Una de las piezas clave del sistema es el *Object-Relational Mapper* (ORM) que proporciona *Django*, utilizado como capa de abstracción entre los modelos del dominio y la base de datos relacional (*PostgreSQL*).

Cada modelo definido en el sistema representa una tabla, y las relaciones entre modelos se traducen directamente en claves foráneas. Esta representación se define de forma estructurada mediante clases en *Python*, y permite una gestión completamente tipada, validada y controlada del esquema relacional.

El *ORM* permite:

- Crear y modificar el esquema de base de datos utilizando migraciones (`makemigrations` y `migrate`).
- Definir restricciones, relaciones y validaciones desde el propio modelo.
- Interactuar con los datos mediante una API de alto nivel sin necesidad de escribir SQL.

- Integrar de forma directa el modelo de datos con componentes del sistema como vistas, serializadores o la administración web.

Gracias a este enfoque, los modelos del sistema no sólo representan la estructura de la base de datos, sino que encapsulan también la lógica de negocio básica asociada a cada entidad (como métodos personalizados o validaciones), fomentando un diseño centrado en el dominio [1].

A continuación, se muestra un ejemplo representativo del modelo que se implementa en la aplicación `corrections` (`Correction`):

```
class Correction(models.Model):  
    prompt = models.ForeignKey(Prompt, on_delete=models.CASCADE)  
    rubric = models.ForeignKey(Rubric, on_delete=models.CASCADE)  
    user = models.ForeignKey(User, on_delete=models.CASCADE)  
    date = models.DateTimeField(auto_now_add=True)  
    description = models.TextField()
```

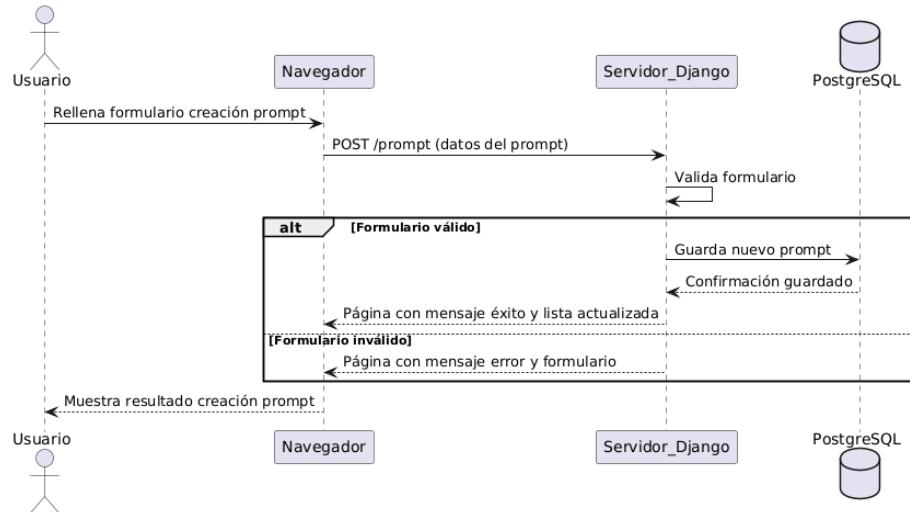
Esta arquitectura favorece una clara separación de responsabilidades entre la lógica de negocio, las operaciones de persistencia y la presentación, en línea con el patrón MVT.

## C.4. Diseño procedimental

En esta sección se presentan los diagramas de secuencia que describen el flujo de interacción entre los distintos componentes del sistema durante la ejecución de operaciones clave. Estos diagramas ayudan a visualizar el comportamiento dinámico de la aplicación, mostrando el orden de los mensajes intercambiados entre actores, vistas, modelos, tareas en segundo plano y bases de datos.

### Diagrama de secuencias: crear nuevo *prompt*

Este diagrama representa el proceso seguido por el sistema cuando un usuario crea un nuevo *prompt*. Incluye la validación del formulario, la persistencia en base de datos y la respuesta adecuada al cliente.

Figura C.11: Diagrama de secuencias: crear nuevo *prompt*

## Diagrama de secuencias: crear nueva rúbrica

Este diagrama detalla los pasos realizados al importar una nueva rúbrica desde un archivo `.md`. Se ilustra cómo se valida el archivo, se comprueba su unicidad, y se guarda en la base de datos si es válido.

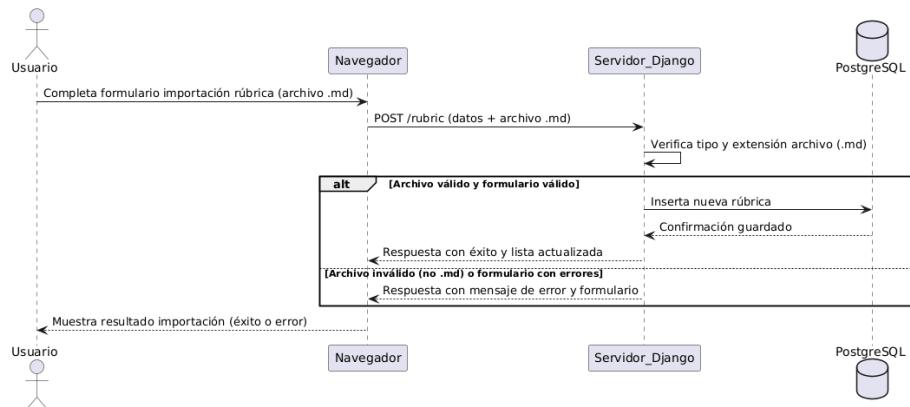


Figura C.12: Diagrama de secuencia: crear nueva rúbrica

## Diagrama de secuencias: crear nueva corrección

Este diagrama muestra la creación de una nueva corrección, asociando los elementos necesarios como rúbrica, *prompt* y tareas. También se realiza la validación de los datos antes de su almacenamiento.

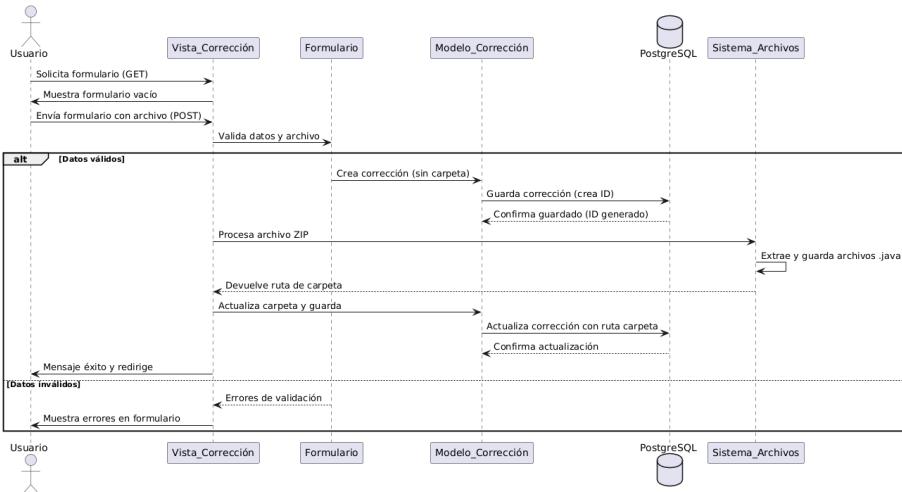


Figura C.13: Diagrama de secuencias: crear nueva corrección

## Diagrama de secuencias: Ejecutar corrección

Este diagrama describe el proceso de ejecución de una corrección utilizando un modelo LLM. Incluye la preparación de datos, el uso de una tarea en segundo plano mediante Celery, y la interacción con el sistema de archivos y el modelo *Ollama*.

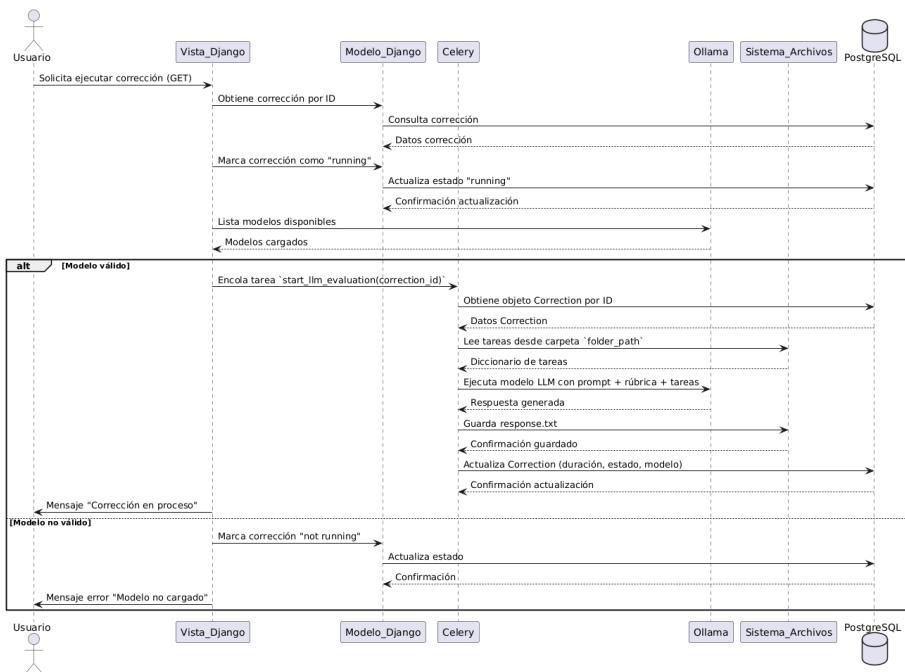


Figura C.14: Diagrama de secuencias: ejecutar corrección

## *Apéndice D*

---

# **Documentación técnica de programación**

---

## **D.1. Introducción**

En este anexo se presenta la documentación técnica del proyecto, la cual proporciona una guía completa para cualquier programador que desee continuar con su desarrollo. Se incluyen recomendaciones sobre el entorno de desarrollo, la organización y estructura de la aplicación, los procesos de compilación, la configuración para la integración, la instalación de dependencias y las baterías de pruebas realizadas.

## **D.2. Estructura de directorios**

### **Directorio Raíz (./)**

El directorio raíz contiene los archivos y carpetas principales del proyecto, incluyendo configuraciones globales y documentación. Los archivos más relevantes son:

- **.dockerignore**: especifica los archivos y directorios que Docker debe ignorar al construir la imagen.
- **.gitattributes**: define atributos de Git, como reglas de normalización de fin de línea o configuraciones para exportar archivos.

- **.gitignore**: especifica los archivos y directorios excluidos del control de versiones (por ejemplo, `venv`, `.git`, archivos de entorno, compilados, etc.).
- **CHANGELOG.md**: registro de los cambios realizados en cada versión del proyecto.
- **Dockerfile**: define el entorno de ejecución del proyecto dentro de un contenedor, incluyendo dependencias, comandos de instalación y configuración.
- **LICENSE**: indica la licencia del proyecto (*MIT*).
- **README.md**: documentación inicial con instrucciones de instalación, descripción del proyecto, tecnologías utilizadas y licencia.
- **TROUBLESHOOTING.md**: documento de resolución de problemas frecuentes que pueden surgir durante la instalación, configuración o ejecución del proyecto.
- **docker-compose.ci.yml**: configuración específica para la integración continua, se desactivan servicios innecesarios para acelerar el flujo de pruebas.
- **docker-compose.yaml**: archivo principal para definir y ejecutar servicios multi-contenedor del proyecto, como Django, PostgreSQL, Redis, etc.
- **entrypoint.sh**: script que se ejecuta al iniciar el contenedor.
- **env.example**: plantilla de ejemplo para configuración de variables de entorno (`.env`).
- **manage.py**: utilidad de línea de comandos para tareas administrativas de Django.
- **pytest.ini**: archivo de configuración para `pytest`.
- **requirements.txt**: lista de dependencias de Python requeridas para el proyecto.

### iagscore

Directorio que contiene la configuración global del proyecto Django. Aquí se encuentran los archivos principales para la configuración y enrutamiento:

- `settings.py`: configuración del proyecto, incluyendo la base de datos, aplicaciones instaladas, claves secretas y ajustes de internacionalización.
- `urls.py`: enrutamiento principal de URLs, que conecta las rutas de las aplicaciones con las vistas correspondientes.
- `celery.py`: configuración para la integración de Celery, utilizado para manejar tareas asíncronas y programadas.
- `asgi.py`: configuración para el despliegue en servidores ASGI, compatible con aplicaciones asíncronas y WebSockets.
- `wsgi.py`: configuración para el despliegue en servidores WSGI.

### accounts

Aplicación para la gestión de cuentas de usuarios (registro de usuarios, etc).

- `models.py`: modelos personalizados para usuarios.
- `views.py`: lógica para registro y gestión de perfiles.
- `urls.py`: enrutamiento para URLs de registro y gestión de perfiles.
- `forms.py`: formularios Django para la creación de nuevos usuarios.
- `admin.py`: configuración de la interfaz de administración para los modelos de usuarios.
- `apps.py`: configuración de la aplicación.
- `tests.py`: pruebas unitarias para los modelos, forms, vistas y otras funcionalidades de la aplicación.
- `migrations/`: migraciones de la base de datos.
- `static/accounts/`: archivos estáticos específicos.
- `templates/accounts/`: plantillas *HTML* para registro y gestión de perfil.

### core

Aplicación que gestiona la lógica básica del proyecto como login , logout, etc.

- `models.py`: definición de modelos de datos para la base de datos.
- `admin.py`: configuración de la interfaz de administración de Django.
- `apps.py`: configuración de la aplicación, incluyendo su nombre.
- `views.py`: lógica de las vistas para procesar solicitudes HTTP.
- `urls.py`: enrutamiento específico de la aplicación.
- `tests.py`: pruebas unitarias para los modelos, forms, vistas y otras funcionalidades de la aplicación.
- `migrations/`: archivos de migraciones para cambios en la base de datos.
- `static/core/`: archivos estáticos específicos de la aplicación.
  - `static/core/`:
    - `docs/`: documentación estática.
    - `img/`: imágenes usadas en la aplicación.
    - `sphinx_docs/`: copia de la documentación generada por Sphinx en *HTML*.
- `templates/core/`: plantillas *HTML* para la aplicación.
  - `templates/core/partials/`: plantillas reutilizables para componentes modulares.

### rubrics

Aplicación para gestionar rúbricas.

- `models.py`: modelos de datos para rúbricas.
- `admin.py`: configuración de la interfaz de administración para los modelos de rúbricas.
- `apps.py`: configuración de la aplicación.

- `views.py`: lógica para mostrar o gestionar rúbricas.
- `urls.py`: enrutamiento de URLs específicas.
- `forms.py`: formularios Django para la importación de rúbricas.
- `tests.py`: pruebas unitarias para los modelos, forms, vistas y otras funcionalidades de la aplicación.
- `migrations/`: migraciones de la base de datos.
- `templates/rubrics/`: plantillas *HTML* para mostrar o importar rúbricas.

## **prompts**

Aplicación para gestionar *prompts*, entradas de texto para modelos de IA.

- `models.py`: modelos para almacenar *prompt*.
- `views.py`: lógica para gestionar *prompt*.
- `admin.py`: configuración de la interfaz de administración para los modelos de *prompt*.
- `apps.py`: configuración de la aplicación.
- `urls.py`: enrutamiento de URLs.
- `forms.py`: formularios Django para la creación de *prompt*.
- `tests.py`: pruebas unitarias para los modelos, forms, vistas y otras funcionalidades de la aplicación.
- `migrations/`: migraciones de la base de datos.
- `templates/prompts/`: plantillas *HTML* para mostrar o crear *prompt*.

## corrections

Aplicación para gestionar, configurar y ejecutar correcciones.

- `admin.py`: configuración de la interfaz de administración para los modelos de correcciones.
- `apps.py`: configuración de la aplicación.
- `forms.py`: formularios Django para la gestión de correcciones.
- `models.py`: modelos para correcciones.
- `signals.py`: configuración de señales Django para ejecutar acciones automáticas tras ciertos eventos (ej. eliminar un modelo).
- `tasks.py`: definición de tareas asíncronas, gestionadas con Celery.
- `tests.py`: pruebas unitarias para los modelos, forms, vistas y otras funcionalidades de la aplicación.
- `urls.py`: enrutamiento de URLs.
- `views.py`: lógica para procesar correcciones.
- `migrations/`: migraciones de la base de datos.
- `templates/corrections/`: plantillas *HTML* para mostrar o configurar correcciones.

## ejemplos

Directorio que contiene recursos esenciales para facilitar la interacción con la herramienta.

- `Prompt/promt_1.txt`: *prompt* predefinido para copiar.
- `Rúbrica/rubrica_1.md`: rúbrica en formato *markdown* lista para importar.
- `Tareas/fibo_2_tareas_java.zip`: fichero comprimido con dos tareas java.
- `Tareas/fibo_3_tareas_java.zip`: fichero comprimido con tres tareas java.

**static**

Carpeta para archivos estáticos globales del proyecto.

- `js/`: archivos JavaScript para funcionalidad frontend.

**tailwind**

Directorio para la configuración y compilación de Tailwind CSS.

- `tailwind.config.js`: configuración personalizada de Tailwind CSS, donde se definen temas, colores y plugins.
- `package.json`: dependencias de Node.js y scripts para compilar Tailwind CSS.
- `static/css/`: archivos CSS generados por Tailwind CSS.

**docs**

Carpeta para documentación del proyecto.

- `LaTeX/`: documentación escrita en LaTeX.
  - `tex/`: archivos fuente de LaTeX.
  - `img/`: imágenes usadas en la documentación.
- `Sphinx/`: módulo para generar la documentación técnica.
  - `_static/`: archivos estáticos (CSS, imágenes, etc.) utilizados por Sphinx.
  - `_templates/`: plantillas personalizadas de *HTML* para la documentación.
  - `accounts.rst`,  
`core.rst`,  
`corrections.rst`,  
`prompts.rst`,  
`rubrics.rst`: documentación específica de cada módulo de la aplicación.
  - `conf.py`: archivo de configuración de Sphinx.

- `index.rst`: página principal (tabla de contenidos) que enlaza a los distintos ficheros `.rst`.
- `intro.md`: archivo de introducción o guía general en formato Markdown.
- `Makefile`, `make.bat`: scripts para compilar la documentación en *HTML*.

## D.3. Manual del programador

Este documento describe las etapas clave que se han seguido durante el desarrollo de este proyecto y puede servir de guía para nuevos colaboradores. En sus páginas se detalla cómo configurar el entorno de trabajo y cuáles son las dependencias necesarias. Asimismo, se describen con detalle los procedimientos principales de compilación, instalación y puesta en marcha.

En la portada del [repositorio](#) de este proyecto se encuentra un resumen de este proceso.

### Requisitos Previos

Para poder ejecutar el proyecto, es necesario contar con las siguientes herramientas instaladas:

- *Git*
- *Docker*
- *Docker Compose*

Adicionalmente, se debe configurar Docker Desktop para asignar recursos mínimos:

- Asignar al menos 8–12 GB de RAM.
- Asignar al menos 4 CPUs.
- Hacer clic en **Apply & Restart** para aplicar los cambios.

Puede verse una imagen de la ventana de configuración en Fig. [D.1](#)

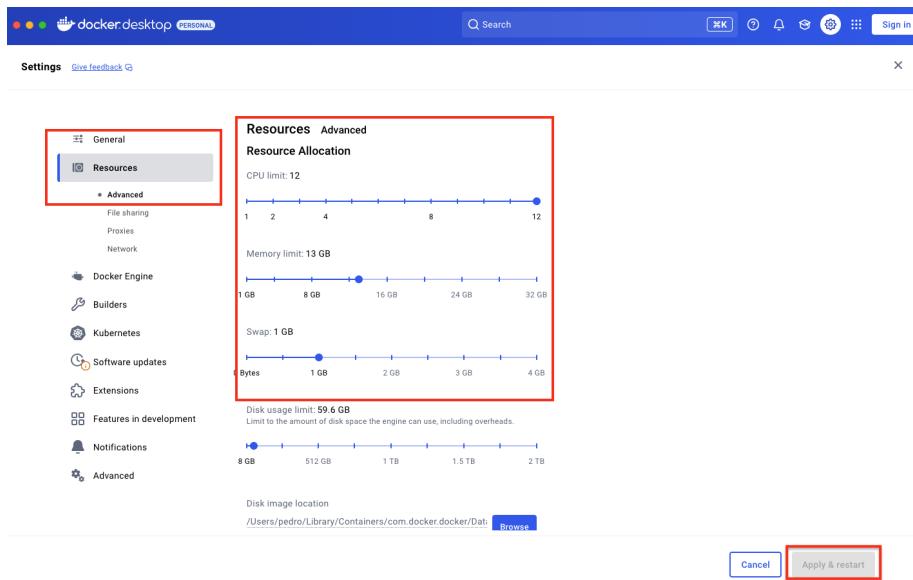


Figura D.1: Imagen configuración de recursos Docker Desktop

### Montaje desde una unidad externa

Para ejecutar el repositorio desde una unidad externa, siga estos pasos en Docker Desktop:

1. Abra Docker Desktop.
2. Vaya a **Settings** → **Resources** → **File Sharing**.
3. Añada la ruta del repositorio que está intentando montar (por ejemplo, F:\path\to\repo).
4. Haga clic en **Apply & Restart** para aplicar los cambios.

Se muestra una imagen en Fig. D.2

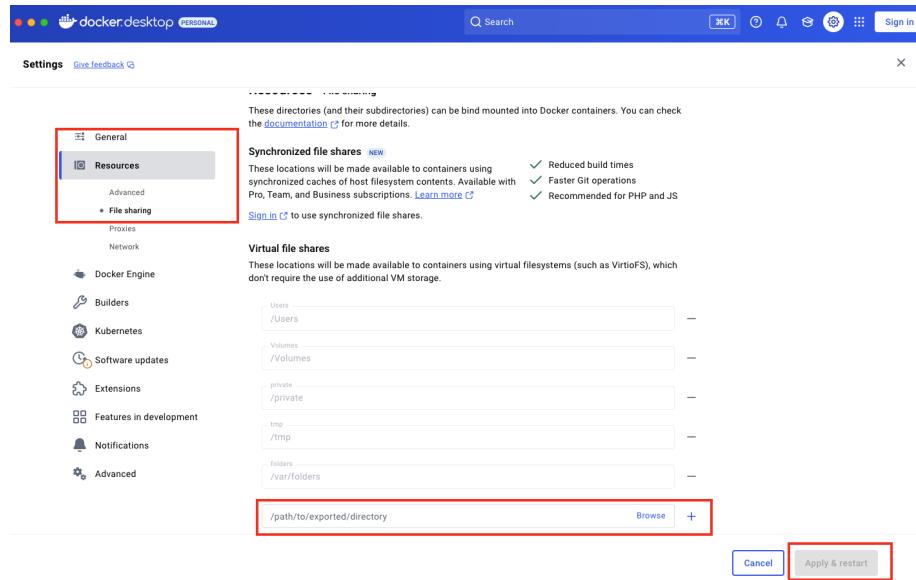


Figura D.2: Imagen configuración unidad externa Docker Desktop

Este proyecto puede ejecutarse tanto desde un *IDE* (por ejemplo, el empleado durante el desarrollo) como desde cualquier terminal.

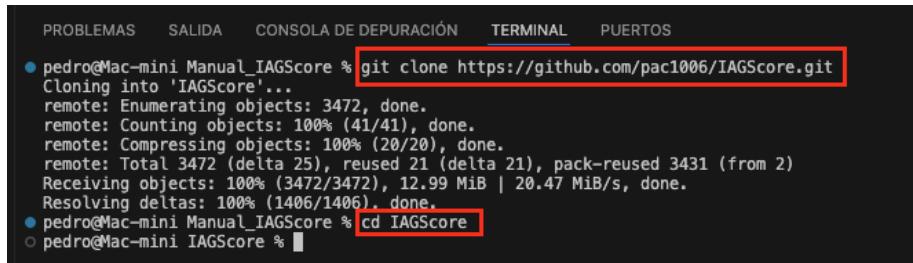
En este manual se describen los pasos de instalación y ejecución usando la terminal integrada de *Visual Studio Code*; sin embargo, los mismos comandos pueden llevarse a cabo en la terminal de cualquier otro sistema operativo.

## D.4. Compilación, instalación y ejecución del proyecto

### Clonar el repositorio

Desde *Visual Studio Code* y usando git: En primer lugar realizaremos la clonación del proyecto, como puede verse en Fig D.3.

```
git clone https://github.com/pac1006/IAGScore.git
cd IAGScore
```



The screenshot shows a terminal window with several tabs at the top: PROBLEMAS, SALIDA, CONSOLA DE DEPURACIÓN, TERMINAL (which is selected), and PUERTOS. The terminal content is as follows:

```
pedro@Mac-mini Manual_IAGScore % git clone https://github.com/pac1006/IAGScore.git
Cloning into 'IAGScore'...
remote: Enumerating objects: 3472, done.
remote: Counting objects: 100% (41/41), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 3472 (delta 25), reused 21 (delta 21), pack-reused 3431 (from 2)
Receiving objects: 100% (3472/3472), 12.99 MiB | 20.47 MiB/s, done.
Resolving deltas: 100% (1406/1406), done.
pedro@Mac-mini Manual_IAGScore % cd IAGScore
pedro@Mac-mini IAGScore %
```

Figura D.3: Imagen clonación de repositorio

## Crear el archivo .env

Para que el proyecto levante correctamente es necesario configurar el fichero con las variables de entorno. En el repositorio se facilita un fichero `env.example` que se puede copiar directamente o replicar usando otros valores para cada variable, Fig. D.4.

En este caso se realiza la copia del archivo de ejemplo:

- **Linux y macOS**

```
cp env.example .env
```

- **Windows**

```
copy env.example .env
```

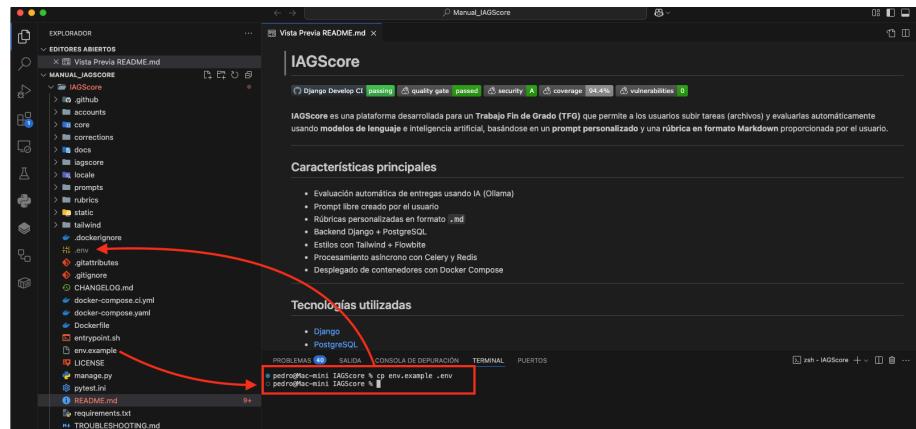


Figura D.4: Imagen copia variables de entorno

## Build con Docker Compose

**Importante:** antes de ejecutar este comando, asegúrate de que *Docker* esté en funcionamiento. Para ello, abre *Docker Desktop* y verifica que se encuentra activo.

Una vez verificado, puedes proceder a construir los contenedores con el siguiente comando:

```
docker compose build
```

Como se muestra en Fig. D.5.

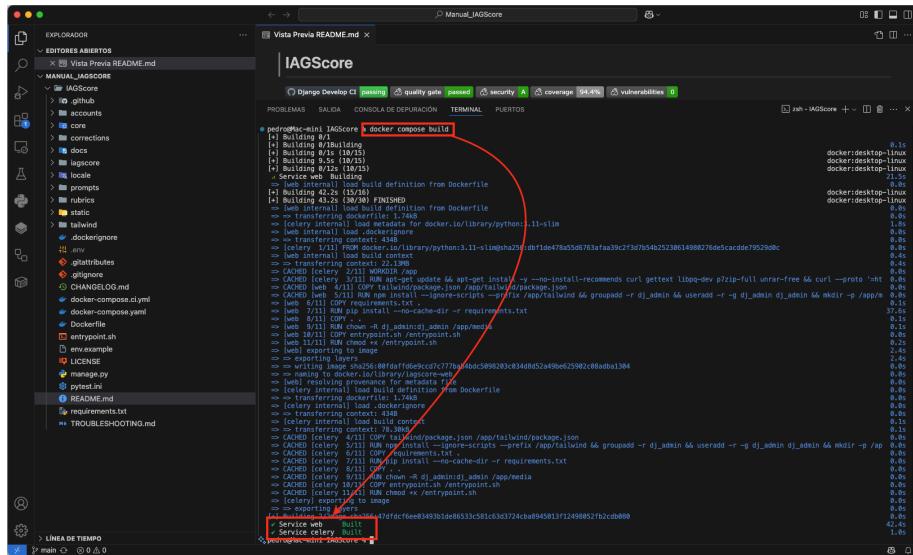


Figura D.5: Imagen construcción de contenedores

Este proceso puede tardar algunos minutos, especialmente la primera vez, ya que se descargan imágenes base y se instalan las dependencias del proyecto dentro de los contenedores.

## Levantar el proyecto

Una vez construidos los contenedores, se puede iniciar el proyecto con uno de los siguientes comandos:

```
docker compose up -d
```

```
docker compose up
```

Este comando levanta los servicios definidos en `docker-compose.yaml`, incluyendo la base de datos, el *backend* de *Django*, *Redis*, *Celery* y *Ollama*, si están habilitados.

Es posible que la primera vez que se ejecute tarde unos minutos, ya que *Docker* debe crear las redes internas, iniciar cada contenedor y realizar configuraciones iniciales. Una vez levantado, se debería ver mensajes en la terminal que indican que los servicios están activos y escuchando.

Se recomienda ejecutar el proyecto en segundo plano con la opción `-d` como se muestra en Fig. D.6. Esto permitirá seguir usando la terminal sin cerrar los servicios, que continuarán ejecutándose en segundo plano.

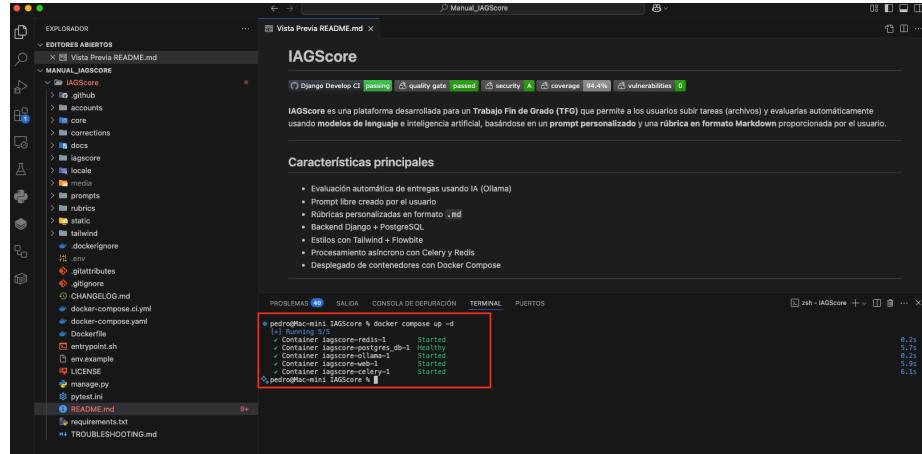


Figura D.6: Imagen levantando proyecto

## Pull del modelo llama3.1

Una vez levantado el proyecto, es necesario hacer *pull* del modelo que se desea utilizar por primera vez Fig. D.7. Por ejemplo, para descargar el modelo `llama3.1`, ejecuta el siguiente comando desde la carpeta raíz del proyecto (`IAGScore`):

Si no has usado la opción `-d` al levantar los contenedores, necesitarás abrir una nueva terminal para ejecutar este comando:

```
docker compose exec ollama ollama pull llama3.1:8b
```

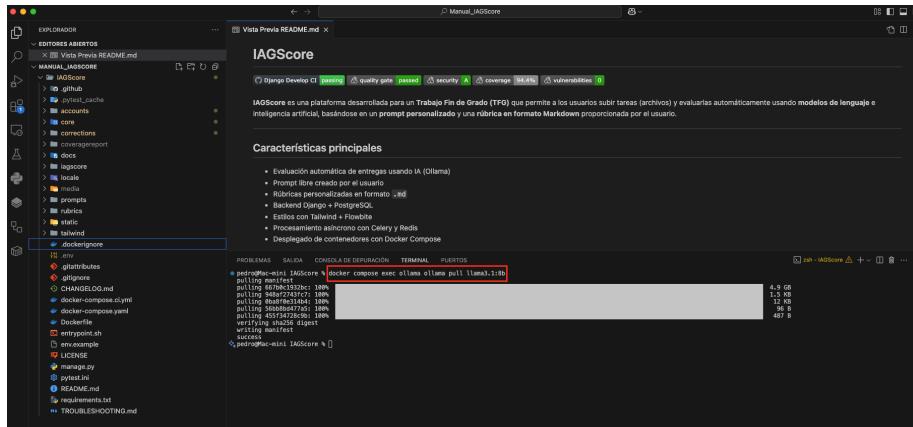


Figura D.7: Imagen haciendo pull modelo Llama3.1

*(Este paso únicamente es necesario hacerlo en la primera ejecución ya que una vez hecho el pull, el modelo permanece en Ollama.)*

### Otros modelos disponibles

Puedes hacer *pull* de cualquier otro modelo soportado por *Ollama*:

```
docker compose exec ollama ollama pull <nombre-del-modelo>
```

Por ejemplo, puedes hacer *pull* de un modelo mas ligero como *Qwen3* (0.6b) como se muestra en Fig. D.8

```
docker compose exec ollama ollama pull qwen3:0.6b
```

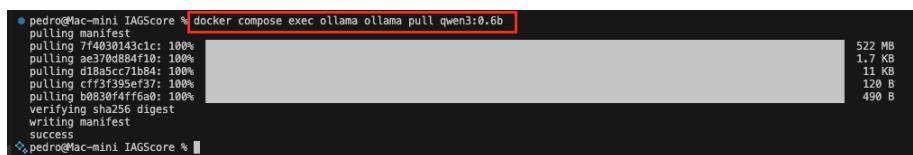


Figura D.8: Imagen haciendo pull modelo Qwen3

### Modelos cargados

Puedes consultar tus modelos cargados en *Ollama*, Fig. D.9, ejecutando el comando:

```
docker compose exec ollama ollama list
```

NAME	ID	SIZE	MODIFIED
qwen3:0.6b	7df6b6e09427	522 MB	3 minutes ago
llama3.1:latest	46e0c10c039e	4.9 GB	11 minutes ago

Figura D.9: Imagen mostrando lista de modelos cargados

### Eliminar modelos

Para eliminar un modelo cargado en *Ollama*, ejecuta:

```
docker compose exec ollama ollama rm <nombre-del-modelo>
```

Se puede ver en Fig. D.10.

```
deleted 'qwen3:0.6b'
```

Figura D.10: Imagen eliminando modelo

## Configuración opcional

### Crear superusuario *Django*

Opcionalmente, se puede crear un *superusuario* para acceder al panel de administración de *Django*:

```
docker compose exec web python3 manage.py createsuperuser
```

Deberás introducir:

- *Email*
- *Username*
- *Password*

Una vez realizados estos pasos, la aplicación estará disponible en el navegador a través de la dirección:

- `http://localhost:8000`

También es posible acceder utilizando la dirección *IP* de la máquina anfitriona: `http://<IP-de-la-máquina>:8000`

## D.5. Pruebas del sistema

A lo largo del desarrollo del proyecto se ha realizado una batería significativa de pruebas, utilizando principalmente `pytest` como herramienta de testeo y `coverage` para el análisis de cobertura de código. En total, se han ejecutado 74 pruebas automatizadas que abarcan distintas áreas de la aplicación, incluyendo modelos, formularios, vistas y rutas de las aplicaciones *accounts*, *core*, *corrections*, *prompts* y *rubrics*.

A continuación se presenta una tabla que relaciona las pruebas automatizadas más representativas con los requisitos funcionales (RF) que verifican. Estas pruebas aseguran que las funcionalidades clave de la aplicación cumplen con las especificaciones definidas. Cabe destacar que, además de estas, se han desarrollado muchas otras pruebas orientadas a validar aspectos internos, casos límite, rendimiento y seguridad del sistema.

Test	Requerimiento
<code>test_register_success</code>	Los usuarios podrán registrarse mediante nombre, correo electrónico y contraseña. (RF-1)
<code>test_login_success</code>	Los usuarios deberán iniciar sesión para usar la aplicación. (RF-2)
<code>test_valid_prompt_form</code>	Los usuarios crearán sus propios <i>prompts</i> . (RF-3)
<code>test_list_prompts</code>	Los usuarios podrán consultar sus <i>prompts</i> creados. (RF-3.1)
<code>test_show_prompt_view</code>	Los usuarios podrán visualizar un <i>prompt</i> concreto. (RF-3.2)
<code>test_delete_prompt_view</code>	Los usuarios podrán eliminar un <i>prompt</i> . (RF-3.3)
<code>test_rubric_page_view_post</code>	Los usuarios podrán importar sus propias rúbricas en formato <i>Markdown</i> . (RF-4)
<code>test_rubric_page_view_get</code>	Los usuarios podrán consultar sus rúbricas importadas. (RF-4.1)
<code>test_show_rubric_view</code>	Los usuarios podrán visualizar los detalles de una rúbrica concreta en formato <i>HTML</i> . (RF-4.2)
<code>test_delete_rubric_view</code>	Los usuarios podrán eliminar una rúbrica importada. (RF-4.3)
<code>test_valid_correction_form</code>	Los usuarios configurarán las correcciones. (RF-5)
<code>test_show_view_correction_view</code>	Los usuarios podrán consultar sus correcciones configuradas. (RF-5.1)
<code>test_show_view_correction_view</code>	Los usuarios podrán visualizar los detalles de configuración de una corrección. (RF-5.2)
<code>test_delete_correction_view</code>	Los usuarios podrán eliminar una corrección. (RF-5.3)
<code>test_run_model_and_download</code>	Los usuarios podrán ejecutar una corrección. (RF-5.4)
<code>test_run_model_and_download</code>	Los usuarios podrán descargar el resultado de la corrección ejecutada. (RF-5.5)

Tabla D.1: Relación entre pruebas y requisitos funcionales (RF)

## Cobertura y enfoque de pruebas

Las pruebas pueden ser ejecutadas con el siguiente comando:

```
docker compose exec web pytest .
```

En Fig. D.11 y Fig. D.12 se muestra el resultado de la ejecución de este comando.

```
(venv) pedro@Mac-mini IAGScore % docker compose exec web pytest .
platform linux -- Python 3.11.13, pytest-8.3.5, pluggy-1.6.0 -- /usr/local/bin/python3.11
cachedir: ./pytest_cache
django: version: 5.1.6, settings: iagscore.settings (from ini)
rootpath: /app
configfile: pytest.ini
plugins: lansmith-0.3.44, django-4.11.1, anyio-4.9.0, cov-6.1.1
collected 74 items
```

Figura D.11: Imagen mostrando ejecución de *test*

```
TOTAL 1439 29 98%
Coverage HTML written to dir coverage/report/html
Coverage XML written to file coverage/report/coverage.xml
74 passed in 64.53s (0:01:04)
```

Figura D.12: Imagen mostrando resultado ejecución *test*

La aplicación debe estar en funcionamiento para poder ejecutarlas.

La cobertura total alcanzada en la base de código es del **98%**, con algunos módulos como *manage.py* y los puntos de entrada *WSGI/ASGI* excluidos del *testeo* por no representar lógica del negocio.

Se han **moqueado servicios externos** como *Ollama*, debido al tiempo que requiere su ejecución real. Esto ha permitido mantener un tiempo de *testeo* razonable sin sacrificar realismo en los *tests* relevantes. También se ha generado una carpeta temporal **media** durante los *tests* para evitar sobrescribir archivos corregidos por los usuarios, asegurando así un entorno de pruebas seguro y aislado.

## Pruebas unitarias y de integración

- **Pruebas unitarias**, con foco en:

- Lógica de validación en formularios.
- Creación y relaciones de modelos.
- Vistas y controladores.

- **Pruebas de caja blanca**, evaluando:
  - Interfaces entre modelos y métodos.
  - Pruebas de estructuras internas y flujos.
  - Comprobación de condiciones límite (*login*, registros).
- **Pruebas de caja negra**, enfocadas en:
  - Renderizado de vistas.
  - Comprobación de comportamientos esperados a partir de distintas entradas.
  - Partición de equivalencia y análisis de valores frontera.
- **Pruebas de integración**, centradas en:
  - Interacción con la base de datos.
  - Flujo completo en funcionalidades como correcciones automáticas o interacción con *LLMs*.

## Cobertura de código

La cobertura de código ha sido medida con `coverage` y el informe *HTML* se puede consultar en `coveragereport/html/index.html`. El resumen general es:

Módulos	Sentencias	Missing	Cobertura
Total	1439	29	98 %

## *Apéndice E*

---

# **Documentación de usuario**

---

## **E.1. Introducción**

En este apartado se ofrece una guía básica para que el usuario pueda interactuar de forma adecuada con la herramienta. Se describen los formularios principales y las distintas opciones disponibles para el usuario final al utilizar este entorno.

## **E.2. Requisitos de usuarios**

Este proyecto ha sido diseñado para ejecutarse en entornos locales, con el objetivo de proteger la privacidad y seguridad de los datos personales generados a partir de las tareas del alumnado. Debido a la naturaleza de los servicios que integra (*Ollama*, *Celery*, *Redis*, *PostgreSQL* y *Django*) no ha sido posible desplegarlo en plataformas gratuitas como *Render* o *Heroku*, ya que presentan incompatibilidades técnicas. No obstante, el usuario puede instalar y ejecutar la aplicación en su propio equipo, siempre que cuente con *Docker* y *Docker Compose* previamente instalados.

## **E.3. Instalación**

Para realizar la instalación del proyecto, basta con seguir las instrucciones de detalladas en la sección [Compilación, instalación y ejecución del proyecto](#)

## E.4. Manual del usuario

En esta sección se documenta el funcionamiento y la interacción con las principales características de la aplicación.

### Recursos de ejemplo

El repositorio incluye una carpeta denominada `ejemplos`, que contiene recursos esenciales para facilitar la interacción con la herramienta. En esta carpeta se encuentran archivos que permiten: Rellenar automáticamente un *prompt* de ejemplo. Importar una rúbrica predefinida. Cargar tareas de muestra para realizar pruebas funcionales. Estos recursos están diseñados para ayudar al usuario a familiarizarse rápidamente con el entorno y comprender su funcionamiento sin necesidad de generar contenido desde cero.

### Acceso a la aplicación

Para acceder a la aplicación desde el navegador, simplemente introduce en la barra de direcciones la siguiente URL:

- `http://localhost:8000`

Al ingresar, se mostrará la ventana principal desde la cual podrás iniciar sesión (*login*). Para ello, deberás proporcionar tu *email* y contraseña.

Si has creado un *superusuario* siguiendo las indicaciones en la sección **Crear *superusuario***, podrás acceder con esas credenciales y obtener acceso al panel de administración de *Django*.

En caso de no disponer de credenciales, será necesario registrarse en la aplicación. Para ello, haz clic en la opción **Regístrate aquí** como se muestra en la Fig. E.1.



Figura E.1: Captura del botón de registro de usuario

Se abrirá la ventana de registro Fig. E.2, donde deberás introducir un nombre, una dirección de *email* y una contraseña. Además, deberás aceptar los Términos y condiciones y pulsar el botón **Crear una cuenta**.



Figura E.2: Captura de la ventana de registro de usuario

Si el registro se ha completado correctamente, aparecerá un mensaje de confirmación Fig. E.3, como el siguiente:

Usuario creado correctamente

Figura E.3: Captura del mensaje de registro correcto

A continuación, podrás acceder a la página de bienvenida de la aplicación, Fig. E.4. En esta página encontrarás una barra de navegación con las distintas secciones disponibles, botones de acceso rápido, un selector de idioma y un botón para cerrar sesión.

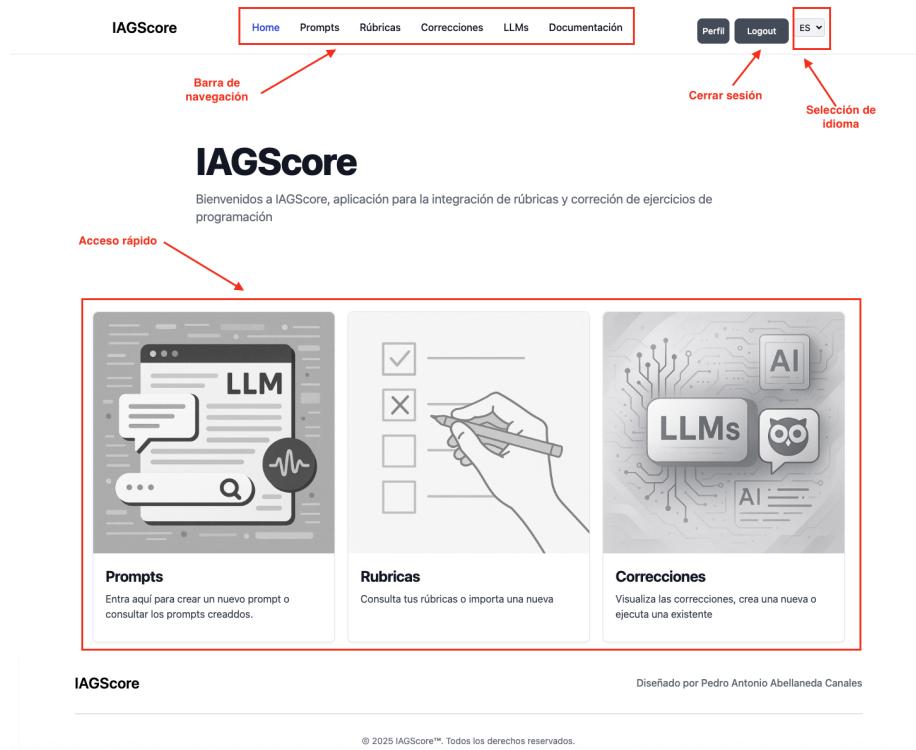


Figura E.4: Captura de la página de bienvenida

Desde esta página se puede seguir el flujo lógico propuesto a través de los menús, comenzando por la creación de un *prompt*, la importación de una rúbrica, la configuración de una corrección mediante la importación de tareas y, finalmente, la ejecución del proceso de corrección.

Para realizar el primer paso, accede a la sección **Prompts**, bien a través del menú de navegación o mediante el botón de acceso rápido. En esta sección se mostrarán los *prompts* ya creados (en caso de que existan) y se ofrecerá la opción de crear uno nuevo. Para ello, pulsa el botón **Nuevo Prompt**, Fig. E.5, introduce un nombre identificativo y el contenido del *prompt*, donde se debe indicar al modelo que se le proporcionará una rúbrica y un conjunto de tareas para realizar la corrección.

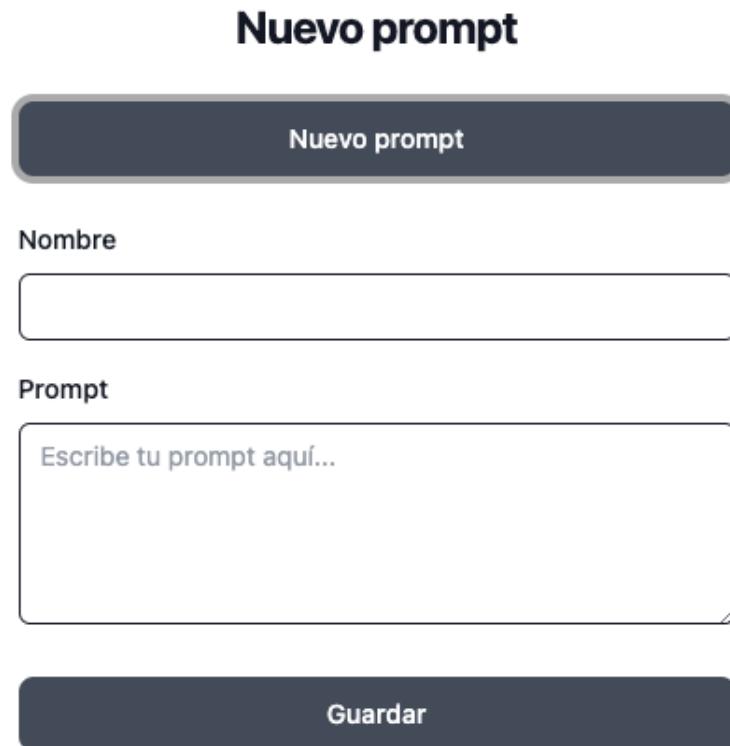


Figura E.5: Captura de la creación de un nuevo *prompt*

Tras pulsar el botón **Guardar**, si todo ha funcionado correctamente, se mostrará un mensaje de confirmación indicando que el *prompt* se ha creado con éxito y aparecerá listado en la tabla correspondiente, junto con la fecha de creación y las acciones disponibles, como se puede ver en Fig. E.6.

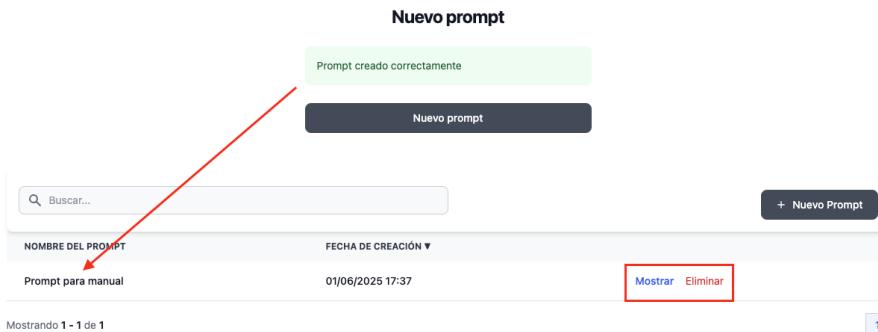


Figura E.6: Captura del *prompt* creado correctamente

El siguiente paso consiste en importar una rúbrica en formato Markdown. Para ello, debemos acceder a la sección **Rúbricas**.

Una vez en esta sección, se debe pulsar el botón **Nueva rúbrica**, lo que abrirá un formulario, como el que se muestra en Fig. E.7, donde podremos introducir un nombre identificativo e importar un archivo en formato .md que contenga el contenido de la rúbrica.

## Nueva rúbrica

**Nueva rúbrica**

**Nombre**

**Archivo Markdown**

Ningún archivo seleccionado

Sube un archivo Markdown con la rúbrica

**Importar**

The screenshot shows a user interface for creating a new rubric. At the top, a dark header bar contains the text "Nueva rúbrica". Below this, there are two main input fields. The first field is labeled "Nombre" (Name) and contains a large, empty text input box. The second field is labeled "Archivo Markdown" (Markdown file) and includes a "Seleccionar archivo" (Select file) button and a message indicating "Ningún archivo seleccionado" (No file selected). Below these fields is a placeholder text "Sube un archivo Markdown con la rúbrica" (Upload a Markdown file with the rubric). At the bottom of the form is a dark button labeled "Importar" (Import).

Figura E.7: Captura del formulario de importación de una rúbrica

Al pulsar el botón **Guardar**, si el proceso se realiza correctamente, se mostrará un mensaje de confirmación, Fig. E.8, indicando que la rúbrica ha sido creada con éxito. Esta aparecerá listada en la tabla correspondiente, junto con la fecha de creación y las acciones disponibles.



Figura E.8: Captura de la rúbrica creada correctamente

Importar la rúbrica en formato *Markdown* resulta especialmente útil, ya que permite una mejor visualización del contenido estructurado, Fig. E.9. Esta visualización puede comprobarse pulsando el botón **Mostrar**.

La captura de pantalla muestra la rúbrica 'Rúbrica de Evaluación - Ejercicios de Programación en Java'. Se observa la cabecera 'Criterios de Evaluación' y una tabla que enumera los criterios con sus descripciones y puntuaciones. Al final de la tabla, se indica 'Total: 10 puntos'. Hay botones 'Volver' y 'Eliminar' en la parte superior y otra vez en la parte inferior.

Criterio	Descripción	Puntos
<b>Funcionamiento</b>	El programa compila y ejecuta correctamente, resolviendo el problema planteado sin errores.	3 pts
<b>Estructura</b>	Se sigue una estructura clara y organizada (uso adecuado de clases, métodos y funciones). Se evita código redundante.	2 pts
<b>Conceptos</b>	Se emplean correctamente estructuras de control (bucles, condicionales), tipos de datos, y otros elementos del lenguaje Java.	2 pts
<b>Legibilidad</b>	Código bien indentado, uso adecuado de nombres de variables y métodos, y comentarios que explican partes clave del código.	2 pts
<b>Excepciones</b>	Se manejan adecuadamente posibles errores o entradas inválidas mediante excepciones o validaciones.	1 pt

**Total: 10 puntos**

Figura E.9: Captura mostrando rúbrica formateada en *HTML*

Una vez disponemos de un *prompt* y de una rúbrica, ya podemos proceder a configurar una corrección. Para ello, accedemos a la sección **Correcciones**, seleccionando la opción correspondiente en la barra de navegación y, a continuación, haciendo clic en **Nueva corrección**, como se puede ver en Fig. E.10.

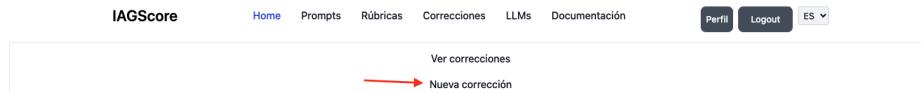


Figura E.10: Captura mostrando el botón **Nueva corrección**

Se abrirá el formulario de configuración, Fig. E.11, donde será necesario seleccionar el *prompt* y la rúbrica previamente creados. Además, se deberá cargar un archivo comprimido (en formato .zip, .rar, .7z o .tar) que contenga las tareas a corregir. También será preciso introducir un nombre o una breve descripción que identifique la corrección y escoger un modelo disponible en la lista (el cual debe haber sido previamente **cargado**).

A continuación, se mostrarán una serie de parámetros avanzados, tales como **Temperatura**, **Top\_p**, **Top\_k**, **Ventana de contexto** y **Formato de salida**, que vienen configurados por defecto, aunque el usuario podrá modificarlos según sus necesidades.

Criterio	Descripción	Puntos
<b>Funcionamiento</b>	El programa compila y ejecuta correctamente, resolviendo el problema planteado sin errores.	3 pts
<b>Estructura</b>	Se sigue una estructura clara y organizada (uso adecuado de clases, métodos y funciones). Se evita código redundante.	2 pts
<b>Conceptos</b>	Se emplean correctamente estructuras de control (bucles, condicionales), tipos de datos, y otros elementos del lenguaje Java.	2 pts
<b>Legibilidad</b>	Código bien indentado, uso adecuado de nombres de variables y métodos, y comentarios que explican partes clave del código.	2 pts
<b>Excepciones</b>	Se manejan adecuadamente posibles errores o entradas inválidas mediante excepciones o validaciones.	1 pt

Figura E.11: Captura mostrando el formulario de configuración de corrección

Una vez completados los datos, al pulsar el botón **Guardar configuración** la aplicación redirigirá automáticamente a la página principal de correcciones. Allí se listarán todas las correcciones configuradas, junto con

información relevante y las acciones disponibles para cada una, como se muestra en Fig. E.12.



Ver correcciones  
Nueva corrección

Corrección creada correctamente

Buscar... + Nueva corrección

DESCRIPCIÓN DE LA CORRECCIÓN	FECHA DE CREACIÓN	ÚLTIMA EJECUCIÓN	VISUALIZACIÓN	EJECUCIÓN	RESULTADO ÚLTIMA EJECUCIÓN	TIEMPO RESPUESTA	ACCIÓN
Corrección para manual	10/06/2025 13:17	Nunca ejecutada	Configuración Tareas	Ejecutar	Descargar último resultado		Clonar Eliminar

Figura E.12: Captura mostrando una corrección configurada

Con la corrección configurada, estará lista para ejecutarse pulsando la opción **Ejecutar**. Al hacerlo, se mostrará un mensaje indicando que la corrección está en proceso y que el resultado podrá consultarse más adelante. Esta operación se realiza en segundo plano, por lo que no bloquea la aplicación. Mientras la corrección se ejecuta, su estado aparecerá como *Ejecutando...* y se deshabilitarán las acciones sobre ella para garantizar la consistencia del sistema, como puede verse en Fig. E.13.



Ver correcciones  
Nueva corrección

Corrección en proceso. Revisa el resultado más tarde.

Buscar... + Nueva corrección

DESCRIPCIÓN DE LA CORRECCIÓN	FECHA DE CREACIÓN	ÚLTIMA EJECUCIÓN	VISUALIZACIÓN	EJECUCIÓN	RESULTADO ÚLTIMA EJECUCIÓN	TIEMPO RESPUESTA	ACCIÓN
Corrección para manual	10/06/2025 13:17	Nunca ejecutada	Configuración Tareas	Ejecutando...	Descargar último resultado		Clonar Eliminar

Figura E.13: Captura mostrando una corrección en ejecución

Al finalizar el proceso, estará disponible la opción para descargar la corrección, Fig. E.14, en formato .txt.

VISUALIZACIÓN	EJECUCIÓN	RESULTADO ÚLTIMA EJECUCIÓN	TIEMPO RESPUESTA	ACCIÓN
Configuracion Tareas	Ejecutar	Descargar último resultado	63 s	Clonar Eliminar

Figura E.14: Captura mostrando la opción de descarga

De este modo, se completa un ciclo completo de la funcionalidad principal de la aplicación.

La sección **LLM** ofrece detalles sobre el modelo utilizado en las diversas pruebas realizadas, mientras que la sección **Documentación** contiene la documentación técnica del proyecto, junto con el manual de instalación.

## *Apéndice F*

---

# **Anexo de sostenibilización curricular**

---

## **F.1. Introducción**

Este anexo expone cómo el desarrollo de la herramienta *IAGScore*, presentada en este Trabajo de Fin de Grado, se alinea con los Objetivos de Desarrollo Sostenible (*ODS*) propuestos por la Agenda 2030 de las Naciones Unidas [6]. La automatización de la evaluación de tareas de programación mediante modelos de lenguaje (*LLMs*) no solo supone un avance técnico, sino que también permite avanzar hacia una educación más justa, eficiente, privada y accesible.

## **F.2. Objetivos de Desarrollo Sostenible relacionados**

***ODS 4 – Educación de calidad:*** Este objetivo [10] busca garantizar una educación inclusiva, equitativa y de calidad, promoviendo el aprendizaje a lo largo de la vida. *IAGScore* contribuye a este fin al ofrecer una evaluación más justa y objetiva, reduciendo los sesgos de la corrección manual. Al funcionar localmente, protege los datos personales del alumnado, asegurando la confidencialidad y el cumplimiento de la normativa de protección de datos. Esto permite un uso ético y respetuoso de los *LLMs* en el ámbito educativo.

***ODS 10 – Reducción de las desigualdades:*** [9] La evaluación automatizada con criterios unificados y trazables promueve la equidad educativa, garantizando una valoración coherente del esfuerzo estudiantil,

sin importar el contexto institucional. Al no depender de servicios en la nube, *IAGScore* evita sesgos externos y barreras tecnológicas, facilitando su adopción en centros con distintos niveles de recursos.

**ODS 9 – Industria, innovación e infraestructura:** este *ODS* [11] promueve la construcción de infraestructuras resilientes, la industrialización sostenible y la innovación. *IAGScore* representa una innovación aplicada al sector educativo al integrar tecnologías emergentes como los *LLMs* en procesos institucionales clave. La posibilidad de desplegar estos modelos en entornos locales es una ventaja estratégica, ya que permite mantener el control completo sobre la infraestructura, reducir la dependencia de servicios externos y evitar la externalización de datos sensibles. Además, el uso de tecnologías abiertas (licencias *MIT* y *BSD*) fomenta la creación de una infraestructura tecnológica accesible, reproducible y escalable.

### F.3. Accesibilidad y adherencia tecnológica

Desde el diseño inicial del proyecto se ha priorizado la usabilidad de la herramienta, con una interfaz sencilla que permite al profesorado sin conocimientos técnicos integrar rúbricas personalizadas y evaluar tareas mediante simples interacciones. La facilidad de uso, la documentación técnica incluida y la transparencia del proceso de corrección aumentan la adopción y fidelización del usuario (profesorado), lo cual es fundamental para que su aplicación tenga impacto real.

La posibilidad de ejecutar la herramienta en máquinas locales o servidores institucionales garantiza una accesibilidad adecuada, incluso en entornos con limitaciones de conectividad o políticas estrictas de protección de datos. Además, se evitan riesgos asociados a la transferencia de datos personales fuera del entorno académico, lo cual refuerza la confianza en su adopción y cumplimiento normativo.

### F.4. Futuras líneas de trabajo

Aunque el enfoque actual de la herramienta se centra en mejorar la eficiencia y objetividad de la evaluación, se prevé incorporar mecanismos explícitos que informen a los usuarios sobre el impacto de su uso en términos de sostenibilidad. Por ejemplo:

- Incorporar estadísticas de uso y ahorro de tiempo, evidenciando el impacto positivo en la gestión docente.

- Integrar análisis de sesgo para garantizar que los modelos empleados mantengan criterios justos y no discriminatorios.
- Ampliar las opciones de despliegue local con soporte para arquitecturas ligeras y entornos desconectados de internet.
- Implementar una API REST que permita integrar la herramienta con plataformas educativas existentes como Moodle, lo que facilitaría su adopción en contextos institucionales amplios, automatizando aún más los flujos de evaluación.

## F.5. Conclusión

Este TFG demuestra que es posible alinear el desarrollo tecnológico con los principios de sostenibilidad. El diseño y aplicación de *IAGScore* no solo ha implicado retos técnicos, sino también decisiones conscientes orientadas a promover una educación más equitativa, eficiente, privada y responsable. En un contexto donde el acceso a la tecnología puede profundizar o mitigar desigualdades, herramientas como esta representan un paso hacia una transformación digital verdaderamente inclusiva y respetuosa con los derechos delumnado.

No obstante, es importante reconocer que el uso de modelos de lenguaje de gran escala (LLMs) conlleva un elevado consumo energético, lo que plantea desafíos en términos de sostenibilidad ambiental. Aunque *IAGScore* cumple con varios Objetivos de Desarrollo Sostenible, este aspecto debe ser considerado en futuras versiones, buscando un equilibrio entre innovación educativa y responsabilidad ecológica.



---

## Bibliografía

---

- [1] Django Software Foundation. Django 5.2: Consultas de bases de datos (orm). <https://docs.djangoproject.com/en/5.2/topics/db/queries/>, 2025. [Internet; Accedido el 1 de junio de 2025].
- [2] Django Software Foundation. Paquetes de contribución en django. <https://docs.djangoproject.com/en/5.2/ref/contrib/>, 2025. [Internet; Accedido el 1 de junio de 2025].
- [3] Zube Inc. Zube | project management for github issues. <https://zube.io>, 2025. [Internet; Accedido el 19 de mayo de 2025].
- [4] Ken Schwaber and Jeff Sutherland. Guía de scrum: Las reglas del juego, 2020. [Internet; descargado 27-marzo-2025].
- [5] Talent.com. Salario: Programador junior. <https://es.talent.com/salary?job=programador+junior>, 2025. [Internet; Accedido el 30 de mayo de 2025].
- [6] Naciones Unidas. La asamblea general adopta la agenda 2030 para el desarrollo sostenible. <https://www.un.org/sustainabledevelopment/es/2015/09/la-asamblea-general-adopta-la-agenda-2030-para-el-desarrollo-sostenible/>, 2015. [Internet; Accedido el 19 de mayo de 2025].
- [7] Wikipedia. Licencia bsd — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/wiki/Licencia\\_BSD](https://es.wikipedia.org/wiki/Licencia_BSD), 2025. [Internet; Accedido el 19 de mayo de 2025].

- [8] Wikipedia. Licencia mit — wikipedia, la enciclopedia libre. <https://es.wikipedia.org/wiki/Licencia/MIT>, 2025. [Internet; Accedido el 19 de mayo de 2025].
- [9] Wikipedia. Objetivo de desarrollo sostenible 10 — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/wiki/Objetivo\\_de\\_Desarrollo\\_Sostenible\\_10](https://es.wikipedia.org/wiki/Objetivo_de_Desarrollo_Sostenible_10), 2025. [Internet; Accedido el 19 de mayo de 2025].
- [10] Wikipedia. Objetivo de desarrollo sostenible 4 — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/wiki/Objetivo\\_de\\_Desarrollo\\_Sostenible\\_4](https://es.wikipedia.org/wiki/Objetivo_de_Desarrollo_Sostenible_4), 2025. [Internet; Accedido el 19 de mayo de 2025].
- [11] Wikipedia. Objetivo de desarrollo sostenible 9 — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/wiki/Objetivo\\_de\\_Desarrollo\\_Sostenible\\_9](https://es.wikipedia.org/wiki/Objetivo_de_Desarrollo_Sostenible_9), 2025. [Internet; Accedido el 19 de mayo de 2025].