



TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

**EduCode: Aplicación móvil para la
corrección automática de
actividades mediante Inteligencia
Artificial.
Manual Técnico**

Autor

Rafael García Pérez

Director

Rafael Muñoz Salinas

Junio, 2025



Agradecimientos

Quisiera expresar mi más sincero agradecimiento en este pequeño espacio dedicado a ello para mencionar a quienes han hecho posible la realización de este Trabajo de Fin de Grado.

A mi padre, Rafa, y a mi madre Mariluz, por confiar en mí y ser un pilar fundamental en mi vida, gracias por tanto

A mi hermano, David, por mostrarme su afecto y sacarme una sonrisa en los momentos más complicados.

A mi pareja, Raquel, por entenderme y estar ahí siempre.

Por último a mi tutor Rafa por ayudarme con la realización del presente trabajo y a mis amigos y compañeros que siempre han estado ahí.

Gracias.

Resumen

Este Trabajo de Fin de Grado tiene como objetivo el desarrollo de una aplicación móvil denominada *EduCode*, cuyo propósito es aplicar inteligencia artificial en el ámbito educativo para facilitar el trabajo del profesorado y mejorar la experiencia del alumnado.

El proyecto combina el diseño de una aplicación móvil con la integración de técnicas de visión artificial. En particular, se implementa un sistema en dos etapas: una primera de reconocimiento óptico de caracteres (OCR) para digitalizar ejercicios escritos, y una segunda basada en modelos de lenguaje para corregir automáticamente las respuestas.

Se han evaluado distintos modelos de OCR y LLM (Large Language Models), seleccionando los más adecuados según criterios de precisión y eficiencia. Asimismo, el sistema ha sido desarrollado con una arquitectura escalable, haciendo uso de buenas prácticas de programación.

EduCode busca contribuir a la modernización del entorno educativo, automatizando procesos repetitivos y demostrando el potencial de la inteligencia artificial como herramienta de apoyo al aprendizaje y la docencia.

Título: EduCode: Aplicación móvil para la corrección automática de actividades mediante Inteligencia Artificial.

Palabras clave: Inteligencia artificial, educación, aprendizaje asistido por inteligencia artificial, modelos de lenguaje

Abstract

This Final Degree Project presents the development of a mobile application called *EduCode*, aimed at applying artificial intelligence in the educational field to support both teachers and students.

The project combines mobile application design with the integration of computer vision techniques. Specifically, it implements a two-stage system: the first stage uses Optical Character Recognition (OCR) to digitize handwritten exercises, while the second applies language models to automatically assess and correct the students' answers.

Several OCR and LLM (Large Language Model) solutions were evaluated and compared, selecting those that offered the best balance between accuracy and performance. Likewise, the system has been developed with a scalable architecture, following programming best practices.

EduCode seeks to contribute to the digital transformation of the educational environment by automating repetitive tasks and demonstrating the potential of artificial intelligence as a support tool for both teaching and learning processes.

Title: EduCode: AI-Powered Mobile App for Automatic Activity Grading

Keywords: Artificial Intelligence, Education, AI Assisted Learning, Language Models.

Índice general

Agradecimientos	III
Resumen	v
Abstract	VII
1. Introducción	1
2. Definición del problema	5
2.1. Problema real	6
2.2. Problema técnico	6
2.2.1. Funcionamiento	7
2.2.2. Entorno	8
2.2.3. Vida esperada	8
2.2.4. Ciclo de mantenimiento	9
2.2.5. Competencia	9
2.2.6. Aspecto externo	9
2.2.7. Estandarización	9
2.2.8. Calidad y fiabilidad	9
2.2.9. Cronograma	11
2.2.10. Pruebas	11
2.2.11. Seguridad	12
3. Objetivos	13
3.1. Objetivos de la aplicación móvil	14
3.2. Objetivos de la Inteligencia Artificial aplicada.	15
4. Antecedentes	17
4.1. Moodle	18
4.2. Magicschool.ai	20
4.3. Mathew.ai	20
4.4. Motores OCR	21
4.4.1. Tesseract OCR	22
4.4.2. Google OCR	22
4.4.3. Azure Vision OCR	22
4.4.4. Easy OCR	22
4.4.5. Paddle OCR	22



ÍNDICE GENERAL

4.4.6. Comparativa Tesseract - Textract - Azure OCR - Google OCR	23
4.5. Modelos de lenguaje	24
4.5.1. ChatGPT	24
4.5.2. Gemini	25
4.5.3. DeepSeek	26
4.5.4. Qwen	27
5. Restricciones	29
5.1. Factores dato	30
5.2. Factores estratégicos	30
5.2.1. Sistema operativo objetivo	31
5.2.2. Entorno de desarrollo	31
5.2.3. Kit de desarrollo para la aplicación móvil	31
5.2.4. Flutter	32
5.2.5. React Native	32
5.2.6. Conclusión	33
5.3. Framework para el backend	33
5.3.1. Lenguaje de programación	34
5.4. Base de datos - Relacional vs No Relacional	37
5.4.1. Bases de Datos Relacionales	37
5.4.2. Bases de Datos No Relacionales	38
5.4.3. Conclusión	38
6. Recursos	39
6.1. Recursos humanos	40
6.2. Recursos Hardware y Software	40
7. Especificación de requisitos	41
7.1. Requisitos funcionales	43
7.2. Requisitos no funcionales	46
7.3. Requisitos de Usuario	46
7.4. Requisitos de Información	48
8. Especificación del sistema	49
8.1. Análisis	50
8.1.1. Usuarios del sistema	50
8.1.2. Descripción de los datos - Modelo entidad relación	50
8.1.3. Descripción funcional - Casos de uso	59
8.1.4. Especificación de casos de uso	67
8.1.5. Matriz de seguimiento	86
8.1.6. Descripción procedimental - Diagramas de Actividad	89
8.1.7. Descripción procedimental - Diagramas de Secuencia	91
8.2. Diseño de la aplicación	94
8.2.1. Diagrama de clases	94
8.2.2. Clases del diagrama	96
8.3. Implementación	104
8.3.1. Arquitectura del servidor	105
8.3.2. Arquitectura del cliente	108
8.3.3. Principios de Diseño	109



ÍNDICE GENERAL

8.3.4. Estructura de Carpetas	109
8.3.5. Implementación de la Arquitectura	110
8.3.6. Ventajas de la Arquitectura Adoptada	110
8.3.7. Diagrama de la arquitectura	111
8.3.8. Clases de implementación	111
8.4. Resumen de implementación	129
8.4.1. Tecnologías usadas	131
8.5. Modelado de la interfaz	133
8.5.1. Colores	133
8.5.2. Tipografía	135
8.5.3. Diagrama de navegación	136
8.5.4. Mockups	137
9. Pruebas	157
9.1. Pruebas unitarias	158
9.1.1. Actividades	161
9.1.2. Asignaturas – Inscripciones	165
9.1.3. Exportación de Calificaciones	171
9.1.4. Autenticación	172
9.1.5. Entregas	173
9.1.6. Usuarios	178
9.1.7. Conclusiones	184
9.2. Pruebas de rendimiento	185
9.2.1. Configuración de las pruebas	185
9.2.2. Tipos de usuarios simulados	186
9.2.3. Resultados obtenidos	186
9.2.4. Verificación de requisitos no funcionales de rendimiento	187
9.2.5. Conclusiones	187
9.3. Pruebas de inteligencia artificial aplicada	188
9.3.1. Aplicación en este proyecto	188
9.3.2. Reconocimiento Óptico de Caracteres (OCR)	188
9.3.3. Large Language Model (LLM)	191
9.3.4. Casos de prueba	193
9.3.5. Soluciones propuestas	195
9.3.6. Pruebas OCR	200
9.3.7. Pruebas LLM	218
9.3.8. Resumen de Evaluaciones	230
9.3.9. Porcentaje de errores detectados en casos negativos	230
10. Conclusiones	233
10.1. Cumplimiento de los objetivos	234
10.1.1. Objetivos de la aplicación móvil	234
10.1.2. Objetivos de la Inteligencia Artificial aplicada	235
10.2. Futuras mejoras	235
10.2.1. Aplicación móvil	236
10.2.2. Inteligencia artificial aplicada	236
10.3. Desafíos y conclusiones finales	237

Índice de figuras

1.1.	Flujo de uso de la aplicación	3
2.1.	Gráfico de horas	11
4.1.	Cursos Moodle	18
4.2.	Contenidos de un curso Moodle	19
4.3.	Moodle IA Generación de imágenes	19
4.4.	Magicschool	20
4.5.	Mathew ai	21
4.6.	Mathew ai	21
4.7.	Dataset para la comparación	23
4.8.	Resultados comparativa OCR	24
4.9.	ChatGPT3.5 vs ChatGPT4	24
4.10.	Comparativa Gemini	26
4.11.	Comparativa Deepseek	27
4.12.	Comparativa Qwen2.5-VL	28
5.1.	Cuota de mercado Android	31
5.2.	Comparativa de rendimiento entre distintos frameworks	36
8.1.	Entidad Usuario	51
8.2.	Entidad Asignatura	52
8.3.	Entidad Actividad	53
8.4.	Entidad Entrega	53
8.5.	Diagrama Entidad - Relación	54
8.6.	Entidad Inscripción	55
8.7.	Diagrama Entidad - Relación	55
8.8.	Casos de uso administración del usuario	62
8.9.	Casos de uso asignaturas	64
8.10.	Casos de uso actividades	65
8.11.	Casos de uso entregas	66
8.12.	Diagrama Actividad CU19	89
8.13.	Diagrama Actividad CU24	90
8.14.	Diagrama Actividad CU27, CU26, CU28	90
8.15.	Diagrama Actividad CU29	91
8.16.	Diagrama Secuencia CU19	92
8.17.	Diagrama Secuencia CU24	93



ÍNDICE DE FIGURAS

8.18. Diagrama Secuencia CU27 y CU29	94
8.19. Diagrama de clases	95
8.20. Arquitectura de la aplicación	105
8.21. Esquema del backend	107
8.22. Clases implementación servidor	108
8.23. Diagrama arquitectura cliente	111
8.24. Clases implementación cliente	112
8.25. Diagrama de bloques implementación	131
8.26. Paleta de azules de Flutter	133
8.27. Logotipo de la aplicación	134
8.28. Fuente Roboto	135
8.29. Diagrama de navegación	136
8.30. Mockup iniciar sesión	137
8.31. Mockup recuperar contraseña	138
8.32. Mockup registro	139
8.33. Mockup mis asignaturas alumno	140
8.34. Mockup mis asignaturas profesor	141
8.35. Mockup buscar asignaturas	142
8.36. Mockup mi perfil	143
8.37. Mockup información asignatura alumno	144
8.38. Mockup información asignatura profesor	145
8.39. Mockup actividades alumno	146
8.40. Mockup actividades profesor	147
8.41. Mockup ver entrega	148
8.42. Mockup veer actividad para hacer la entrega	149
8.43. Mockup lista de alumnos alumno	150
8.44. Mockup lista de alumnos profesor	151
8.45. Mockup perfil de alumno	152
8.46. Mockup perfil de profesor	153
8.47. Mockup estadísticas de asignatura	154
8.48. Mockup mi progreso	155
9.1. Resumen de cobertura de código del backend	184
9.2. Solución Positiva Test 1	196
9.3. Solución Negativa Test 1	196
9.4. Solución Positiva Test 2	197
9.5. Solución Negativa Test 2	197
9.6. Solución Positiva Test 3	198
9.7. Solución Negativa Test 3	199
9.8. Solución Positiva Test 4	199
9.9. Solución Negativa Test 4	200
9.10. Imagen enviada CPIA 1 Caso positivo	201
9.11. Imagen enviada CPIA 1 Caso negativo	203
9.12. Imagen enviada CPIA 2 Caso positivo	205
9.13. Imagen enviada CPIA 2 Caso negativo	207
9.14. Imagen enviada CPIA 3 Caso positivo	209
9.15. Imagen enviada CPIA 3 Caso negativo	212
9.16. Imagen enviada CPIA 4 Caso positivo	214



ÍNDICE DE FIGURAS

9.17. Imagen enviada CPIA 4 Caso negativo	215
---	-----

Índice de cuadros

2.1. Tabla duración en horas	11
5.1. Comparación entre JavaScript y Python	34
5.2. Tabla comparativa FastAPI vs Flask	35
7.1. Requisitos Funcionales del Sistema	43
7.2. Requisitos Funcionales - Gestión de Usuarios	44
7.3. Requisitos Funcionales - Comunicación entre Usuarios	44
7.4. Requisitos Funcionales - Gestión de Asignaturas	44
7.5. Requisitos Funcionales - Gestión de Actividades y Entregas	45
7.6. Requisitos Funcionales - Evaluación y Notas	45
7.7. Requisitos Funcionales - Funcionalidades Avanzadas	45
7.8. Requisitos No Funcionales	46
7.9. Requisitos de Usuario	47
7.10. Requisitos de Información del Sistema	48
8.1. Entidades en 1FN	56
8.1. Entidades en 1FN	57
8.2. Entidades en 2FN	57
8.3. Entidades en 3FN	57
8.3. Entidades en 3FN	58
8.4. Historias de Usuario	60
8.5. Casos de Uso de Administración del Usuario	61
8.6. Casos de Uso de Asignaturas	63
8.7. Casos de Uso de Actividades	64
8.8. Casos de Uso de Entregas	65
8.9. Plantilla de Especificación de Casos de Uso	67
8.10. Especificación del Caso de Uso: Registro	68
8.11. Especificación del Caso de Uso: Iniciar Sesión	69
8.12. Especificación del Caso de Uso: Cerrar Sesión	69
8.13. Especificación del Caso de Uso: Recuperar Contraseña	70
8.14. Especificación del Caso de Uso: Ver Información Asignatura	70
8.15. Especificación del Caso de Uso: Expulsar Alumno Asignatura	71
8.16. Especificación del Caso de Uso: Editar Asignatura	71
8.17. Especificación del Caso de Uso: Crear Asignatura	72
8.18. Especificación del Caso de Uso: Borrar Asignatura	73
8.19. Especificación del Caso de Uso: Exportar Calificaciones Asignaturas	74



ÍNDICE DE CUADROS

8.20. Especificación del Caso de Uso: Enviar Correo a Usuario	74
8.21. Especificación del Caso de Uso: Ver Alumnos Asignatura	75
8.22. Especificación del Caso de Uso: Ver Perfil Usuario	75
8.23. Especificación del Caso de Uso: Editar Perfil	76
8.24. Especificación del Caso de Uso: Inscribir Asignatura	76
8.25. Especificación del Caso de Uso: Ver Mi Progreso	77
8.26. Especificación del Caso de Uso: Ver Estadísticas Asignatura	77
8.27. Especificación del Caso de Uso: Ver Actividades Asignatura	78
8.28. Especificación del Caso de Uso: Crear Actividad	78
8.29. Especificación del Caso de Uso: Borrar Actividad	79
8.30. Especificación del Caso de Uso: Editar Actividad	79
8.31. Especificación del Caso de Uso: Ver Actividad	80
8.32. Especificación del Caso de Uso: Ver Entregas de Actividad	80
8.33. Especificación del Caso de Uso: Calificar Entrega Manualmente	81
8.34. Especificación del Caso de Uso: Exportar Entregas	82
8.35. Especificación del Caso de Uso: Ver Entrega	82
8.36. Especificación del Caso de Uso: Hacer Entrega	83
8.37. Especificación del Caso de Uso: EscanearOCR	84
8.38. Especificación del Caso de Uso: Evaluar Solución	85
8.39. Especificación del Caso de Uso: Salir de Asignatura	86
8.40. Matriz de Trazabilidad entre Casos de Uso y Requisitos Funcionales	87
8.41. Clase Usuario	97
8.42. Clase Profesor	98
8.43. Clase Alumno	98
8.44. Clase Asignatura	99
8.45. Clase Inscripcion	100
8.46. Clase Actividad	101
8.47. Clase Entrega	103
8.48. Clase User	112
8.49. Clase ProfileService	113
8.50. Clase ProfileProvider	113
8.51. Clase AuthProvider	114
8.52. Clase AuthService	115
8.53. Clase Asignatura	116
8.54. Clase SubjectProvider	117
8.55. Clase SubjectService	118
8.56. Clase Actividad	119
8.57. Clase ActivityProvider	121
8.58. Clase ActivityService	122
8.59. Clase Submission	123
8.60. Cuadro 8.59: Clase Submission_Provider	124
8.61. Clase SubmissionService	126
8.62. Clase Inscripcion	127
8.63. Clase EnrollmentProvider	128
8.64. Clase EnrollmentService	129
9.1. Índice de Casos de Prueba	158
9.1. Índice de Casos de Prueba	159



ÍNDICE DE CUADROS

9.1. Índice de Casos de Prueba	160
9.79. Resultados de las pruebas de rendimiento	186
9.80. Caso de prueba de IA 1	194
9.81. Caso de prueba de IA 2	194
9.82. Caso de prueba de IA 3	194
9.83. Caso de prueba de IA 4	195
9.84. Evaluaciones de los LLM para CPIA-1 caso positivo	220
9.85. Evaluaciones realizadas para CPIA-1 caso negativo	221
9.86. Evaluaciones realizadas para CPIA-2 caso positivo	222
9.87. Evaluaciones realizadas para CPIA-2 caso negativo	224
9.88. Evaluaciones realizadas para CPIA-3 caso positivo	226
9.89. Evaluaciones realizadas para CPIA-3 caso negativo	227
9.90. Evaluaciones realizadas para CPIA-4 caso positivo	228
9.91. Evaluaciones realizadas para CPIA-4 caso negativo	229
9.92. Resumen de calificaciones otorgadas por cada LLM	230
9.93. Porcentaje de detección de errores en los casos negativos	231
10.1. Cumplimiento de objetivos de la aplicación móvil	234
10.2. Cumplimiento de objetivos de la Inteligencia Artificial aplicada	235

Capítulo 1

Introducción



CAPÍTULO 1. INTRODUCCIÓN

En el presente año, los avances en el campo de la inteligencia artificial han ido creciendo de manera, me atrevería a decir, exponencial, y, sin darnos cuenta, se ha colado en nuestro día a día, formando parte de procesos tan primitivos del ser humano, como lo es la toma de decisiones, el aprendizaje, el análisis de todo tipo de parámetros, datos, incluso emociones. La tan sonada I.A. ha llegado para quedarse, eso lo tenemos claro, pero, ¿realmente sabemos sacarle el máximo partido?

Según la propia web de la Universidad de Córdoba [1], son 110 plazas las que se ofertan cada año para alumnos de nuevo ingreso en el grado de Ingeniería Informática, si bien es cierto que se dividen en dos grandes grupos de alumnos, cada grupo está constituido, aproximadamente, por 55 alumnos sin contar aquellos que no son de nuevo ingreso. ¿Qué supone esto? Las consecuencias son directas, el docente encargado de impartir clase, tendrá que ingeniárselas para, día tras día, ofrecer una educación de calidad y personalizada al alumnado, algo que es bastante difícil a lo largo de las 3 horas semanales que se imparten de las llamadas clases de teoría. Es bien sabido que, a diferencia de otras ramas de la ingeniería, la importancia de la programación en la ingeniería informática es clave, ya que desde el primer año tenemos asignaturas dedicadas a formarnos con el objetivo de adquirir las bases y los fundamentos de la programación debido a la gran importancia que dicha disciplina tiene en el campo de la informática.

En un mundo donde la educación digital está en constante evolución, el desarrollo de herramientas tecnológicas que faciliten el aprendizaje es una prioridad. La enseñanza tradicional, aunque efectiva, presenta desafíos en términos de accesibilidad, personalización y adaptabilidad a las necesidades individuales de los estudiantes sobre todo en los casos como el mencionado en los que el número de alumnos por docente es tan alto. En este contexto, el presente proyecto surge como una iniciativa para mejorar la experiencia educativa mediante el uso de una aplicación innovadora, que emplea la inteligencia artificial como una parte más de la misma, totalmente integrada con el resto de funcionalidades que se ofrecen.

Con este Trabajo de Fin de Grado, se pretende converger, en una aplicación móvil, una plataforma educativa que sirva como instrumento de evaluación para el profesorado, una aplicación la cual permitirá al alumno aprender y recibir una atención más personalizada con respecto a sus necesidades, y sistemas de inteligencia artificial integrados capaces de ofrecer dicho soporte. Si bien inicialmente está pensado para docentes y alumnos de asignaturas de programación, lo cierto es que se queda abierta a múltiples áreas, siendo esta aplicación completamente funcional en otras áreas donde la evaluación mediante Inteligencia Artificial supone una gran solución.

De esta forma, ¿Cómo convergeremos todo esto en un proyecto? ¿De dónde venimos y hacia dónde vamos? Pues estas preguntas tienen respuesta. La situación es la siguiente, el profesor para evaluar propone en clase unos ejercicios de rápida elaboración para que los alumnos demuestren si han atendido en clase y han entendido los conceptos explicados, estos ejercicios serán realizados sobre un papel por cada uno de los alumnos presentes en un folio y entregarán su respuesta al profesor para que éste lo evalúe. Como dijimos anteriormente, esto es bastante ineficiente, por lo que, la propuesta es la siguiente, el profesor digitaliza el ejercicio propuesto en una aplicación móvil, la cual le permite introducir, además del enunciado descriptivo, unos parámetros clave a evaluar, un lenguaje de pro-



gramación en caso de que la actividad sea de dicha índole, y fecha límite de entrega. En el momento en que el profesor crea la actividad en la plataforma, los alumnos comenzarán a realizar su solución en el folio, acto seguido, mediante la aplicación móvil, realizarán una fotografía al folio, la aplicación con la imagen recibida, mediante un OCR (Optical Character Recognition) pasará la imagen a texto plano, dicho texto para minimizar los errores posibles del OCR podrá ser editado por el alumno y se enviará. El sistema una vez reciba el texto de la solución y la imagen, la corregirá mediante una inteligencia artificial empleando un LLM (Large Language Model), esto lo hará en base al enunciado y los parámetros indicados por el profesor, obteniendo de esta forma, una nota, y una corrección que indicará al alumno qué está bien, qué está mal, y puntos de mejora, de esta forma este proceso es casi instantáneo y el profesor podrá instantáneamente consultar todas las entregas y las calificaciones y que los alumnos han obtenido en segundos.

En la figura 1.1 podemos ver más visualmente el flujo completo que se seguiría en un uso normal de la aplicación como el explicado.

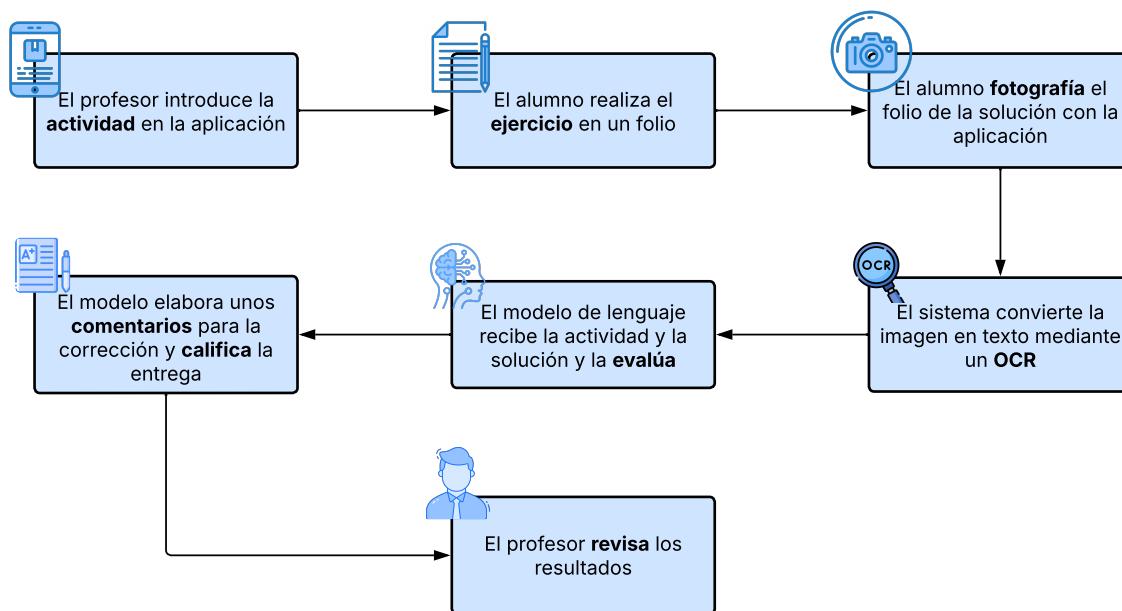


Figura 1.1: Flujo de uso de la aplicación

Capítulo 2

Definición del problema



A lo largo del presente capítulo, trataremos de identificar y presentar inequívocamente la problemática a la que nos enfrentamos, sentando las bases de aquello que queremos abordar con este proyecto, diferenciando claramente dos vertientes, la primera de ellas, desde el punto de vista del usuario final, la cual recibe el nombre de problema real, y la segunda, desde un punto de vista más técnico, recibe el nombre de problema técnico.

2.1 Problema real

Muchos docentes de asignaturas relacionadas con la programación, no disponen de una manera rápida y eficaz que permita atender de manera personalizada la gran cantidad de problemas que un alumno puede encontrarse en su etapa más temprana de aprendizaje en el ámbito de la programación informática, a veces, la cantidad de alumnos es tan sumamente grande, que se hace imposible atender a todos los alumnos. En ocasiones, se llegan a plantear retos en clase, pequeños ejercicios que tienen tres objetivos generalmente.

1. Impulsar al alumno a que ponga a prueba sus conocimientos y ponga en práctica sus habilidades para resolver problemas usando lo aprendido en clase.
2. Permitir al alumno conocer cuáles son sus fallos si los tiene y descubrir puntos de mejora en base a una retroalimentación.
3. Permitir al profesor llevar un control diario de la asistencia y trabajo de los alumnos y llevar un seguimiento de la comprensión global de distintos conceptos en clase.

Desarrollar un ecosistema a través del cual se permita cumplir con los objetivos anteriormente nombrados no es algo trivial, puede ser tedioso si los tiempos de espera se ven prolongados tanto para el docente en su beneficio como para el alumno para facilitar su aprendizaje. Reducir los tiempos y esfuerzos de evaluación y de aprendizaje es un parámetro clave, pues realizar todo esto manualmente es algo que no beneficia a reducir los tiempos de espera y habría que buscar una manera mucho más orgánica de llevarlo a cabo.

Otro problema presente es que, dichos retos serán resueltos por el alumnado escribiendo a papel el código que proceda para la resolución del mismo, por lo que, para el profesor, recoger las decenas de folios de cada alumno y corregirlos individualmente puede resultar ser algo tedioso que implica a su vez que, la corrección, ha de ser hecha en el mismo papel y volver a ser entregada al alumno, un sistema tradicional, bastante ortodoxo, pero que cuando el número de alumnos es tan elevado, se podría decir que es poco óptimo de manera que los tres objetivos nombrados con anterioridad son más difíciles de conseguir en tiempo y forma.

2.2 Problema técnico

En este apartado, se enunciará el problema técnico de acuerdo a la técnica de ingeniería denominada PDS (Product Design Specification), la cual nos permite desglosar el problema en las siguientes secciones:

1. Funcionamiento.
2. Entorno.



3. Vida esperada.
4. Ciclo de mantenimiento.
5. Competencia.
6. Aspecto externo.
7. Estandarización.
8. Calidad y fiabilidad.
9. Programa de tareas.
10. Pruebas.
11. Seguridad.

2.2.1 Funcionamiento

En esta sección listaremos las distintas funcionalidades que deberá ofrecer la solución que adoptemos. En primer lugar ofrecerá un sistema de gestión de usuarios, se diferenciarán principalmente dos roles, rol de alumno, y rol de profesor, cada uno con unos permisos específicos sobre las funcionalidades ofrecidas, dichos usuarios podrán registrarse o iniciar sesión.

En primera instancia, debemos identificar dos tipos de usuarios o roles, alumnos, y profesores.

A nivel de usuarios (entendiendo que usuario es aquella entidad que engloba todos los distintos roles que podamos tener en nuestra aplicación), todos los roles podrán ver su perfil, y editarlo, así como ver el perfil del resto de usuarios, mostrando información esencial como el nombre, el correo de contacto, o las asignaturas que imparte en caso de ser profesor o las asignaturas en las que están matriculados en caso de ser alumno. Adicionalmente, podrán recuperar sus credenciales de acceso y cambiarlas en caso de pérdida u olvido.

En cuanto a los profesores, serán capaces de crear, eliminar, o editar asignaturas, estableciendo un código de acceso que actuará a modo de contraseña para evitar que cualquiera pueda matricularse en la misma, no obstante, en caso de matriculaciones indebidas este podrá expulsar a los alumnos de la asignatura, asimismo, podrá también contactar con ellos vía e-mail que estos tienen registrado en la aplicación. En lo referente a las actividades, podrán crear actividades así como editarlas y eliminarlas, estableciendo fecha límite de entre y otros parámetros como lenguaje y aspectos a valorar en los que la I.A. prestará más atención a la hora de evaluar. En cuanto a las entregas de dichas actividades realizadas por parte de los alumnos, podrán ver la calificación y algunas otras características de la misma como fecha de entrega o la retroalimentación, pudiendo editar la nota de la entrega manualmente para casos en los que así lo considere el docente. Por último al respecto, podrá exportar las calificaciones en formato CSV y mandarlas por correo electrónico, para que éste pueda tener en un único archivo un resumen de las calificaciones para dicha actividad. También dispondrá de esta opción para la asignatura



al completo, es decir, generará un CSV donde podrá ver las entregas de cada alumno y una nota media ponderada. Por último, el profesor tendrá acceso a unas estadísticas básicas sobre la asignatura y las actividades, como tasa de entrega, o nota media del alumnado.

En cuanto a los alumnos, serán capaces de inscribirse y cancelar la inscripción de una asignatura, podrán ver la información general de las asignaturas así como el profesor que las imparte, los alumnos que son compañeros matriculados en la misma, y las actividades que están creadas por el profesor para la misma, pudiendo ver información de las mismas, y la evaluación de su entrega en caso de que la hayan realizado, por ende, podrán realizar las entregas de las actividades, mediante una fotografía realizada al folio sobre el que tienen la respuesta, de manera que, el sistema, automáticamente, será capaz de detectar el texto sobre la misma y volcarlo para que el alumno lo revise y lo edite en caso de algún fallo del OCR. Una vez tenga el texto listo, podrá enviar la solución que será automáticamente evaluada por un sistema de inteligencia artificial, proporcionando una retroalimentación y una calificación al alumno sobre su solución y los requisitos de la actividad previamente impuestos por el profesor. Finalmente, el alumno podrá consultar unas estadísticas básicas sobre su curso, como su nota media o las entregas realizadas/no realizadas.

2.2.2 Entorno

Respecto al entorno de la aplicación que se desarrolla, tendremos en cuenta los siguientes aspectos:

- La aplicación móvil será desarrollada para dispositivos Android, aunque se deja abierta la posibilidad de hacerla multiplataforma como futura mejora.
- La aplicación será usada por alumnos experimentados en el trato con dispositivos móviles y aplicaciones móviles.
- La interfaz de usuario será limpia y adaptable a distintas dimensiones de pantalla.
- El sistema deberá ser consistente y seguro ante posibles errores o intentos de realizar operaciones no autorizadas.
- El sistema OCR deberá ser desplegado en un servidor con potencia gráfica suficiente para realizar las inferencias, ídem con los LLM.
- El proyecto en su totalidad seguirá una arquitectura cliente servidor. [2]
- El entorno de desarrollo usado para el desarrollo será Visual Studio Code.

2.2.3 Vida esperada

El tiempo de vida útil de un proyecto podríamos definirlo como *la vida útil total de un producto en uso desde el punto de venta hasta el punto de descarte* [3]. Esto es algo difícil de estimar. La aplicación podrá ser utilizada indefinidamente siempre que sea compatible con versiones posteriores de Android y los sistemas de I.A. sigan funcionando, si bien es cierto que es funcional sin estos no tendría tanto sentido el uso de esta aplicación, asimismo, cabe decir que los rápidos avances de la I.A pueden potenciar y alargar el uso de la aplicación, pues estará desarrollada de manera que el cambiar los sistemas de inteligencia artificial utilizados, no sea un problema a nivel de código. Dependiendo de



estos factores, podríamos estimar un uso continuado sin problemas de entre 3 a 5 años, ya que hemos de tener en cuenta el rápido avance de las tecnologías emergentes y las distintas aplicaciones basadas en I.A. que serán desarrolladas.

2.2.4 Ciclo de mantenimiento

El mantenimiento no será necesario una vez se libere la versión definitiva, no obstante podrá ser modificado en caso de encontrar algunos errores o querer incorporar nuevas características que busquen mejorar la aplicación.

2.2.5 Competencia

Como competencia del software existen aplicaciones que ofrecen un sistema de gestión de contenidos educativos con una amplia gama de funcionalidades y posibilidad de personalización, no obstante, este proyecto presente está desarrollado para un propósito muy específico a costa de prescindir de tantas posibilidades de personalización.

2.2.6 Aspecto externo

Es uno de los aspectos sobre los que hay que poner el foco. El aspecto externo de la aplicación causará las primeras impresiones al cliente o usuario, puesto que el aspecto es lo primero que el usuario ve cuando se presenta la aplicación. La interfaz de usuario será visualmente atractiva, siguiendo una paleta de colores predefinida por el entorno, asegurando que la gama de colores sea armoniosa y agradable a la vista, usando iconos que semánticamente permitan que el usuario pueda usar la aplicación sin necesidad de usar el manual de usuario o usarlo lo menos posible, no obstante, estará disponible para su uso cuando así sea requerido.

En cuanto a la parte del servidor, simplemente será iniciado por una línea de comandos, no obstante, se dispondrá de una interfaz web para realizar llamadas a la API e caso de que se quiera comprobar el correcto funcionamiento de la misma, o se quiera consultar la documentación para continuar con el desarrollo.

2.2.7 Estandarización

Se tendrán en cuenta estándares relacionados con la programación, concretamente los de Dart como null-safety y programación orientada a objetos, así como API REST. También se hará uso del estándar de comunicaciones HTTPS [4] para la comunicación de la aplicación móvil con la API.

2.2.8 Calidad y fiabilidad

Uno de los puntos más importantes de cualquier proyecto es asegurar la calidad y fiabilidad de aquello que se desarrolla, hacerlo resistente ante fallos y consistente, no es tan solo cumplir con la funcionalidad requerida, si no hacer un proyecto con un alto grado de mantenibilidad y bajo acoplamiento entre módulos. Para ello se han aplicado buenas prácticas para programación, siguiendo una estructura modularizada y estándares de programación limpia, y paradigma de orientación a objetos y desarrollo guiado por tests, elaborando un conjunto de tests lo suficientemente completo como para intentar abarcar el máximo número de casos extremos que nos aseguren una gran fiabilidad y



calidad del software que estamos construyendo. Otros problemas que debemos tener en cuenta son:

- Problemas de comunicación con el servidor del backend.
- Problemas con el funcionamiento del OCR.
- Fallos en el servidor del LLM.
- Sobrecarga del backend debido a un gran número de peticiones

Programa de tareas

Podríamos enumerar una serie de fases que definirán nuestro proyecto:

1. Investigación y formación
2. Preparación
3. Especificación de requisitos y análisis
4. Diseño
5. Implementación
6. Pruebas
7. Documentación

Desglosando cada una de las fases:

- **Investigación y formación:** Es la primera fase del proyecto, esta consistirá en un periodo de tiempo en el cual investigaré las diferentes maneras de llevar a cabo el proyecto, la arquitectura que usar, y de esta manera determinar las herramientas que usaré a posteriori. **Preparación:** En esta fase, la cual será la más corta, se preparará el entorno de desarrollo pertinente así como los recursos hardware y software en general para comenzar con las primeras fases del proyecto.
- **Especificación de requisitos y análisis:** En esta fase se emitirán los requerimientos del proyecto que vamos a concebir a nivel de software, de manera que quedarán cerradas las características y objetivos que hemos de cubrir con el proyecto.
- **Diseño:** En esta fase se diseñarán los distintos procesos de negocio que serán requeridos para cubrir las necesidades, así como las bases de datos.
- **Implementación:** Una vez finalizada la fase de diseño, se realizará la implementación de este, realizando en iteraciones entregables del software funcional que cumpla con lo anteriormente establecido.
- **Pruebas:** Esta fase también es transversal durante toda la implementación, ya que se realizarán pruebas constantemente sobre el software que se está construyendo, tanto en la parte del frontend como la del backend. Además, debido a la inclusión de la I.A. en el proyecto, se realizarán diferentes pruebas sobre distintos modelos que podamos probar de cara a la implementación final.



- **Documentación:** Esta fase será transversal a todo el proyecto, ya que a lo largo del mismo se irá cumplimentando toda la documentación necesaria a lo largo de todas las fases de este.

El tiempo límite para la entrega del proyecto será el 20 de Mayo del 2025, estimando una inversión total de unas 300 horas en el proyecto.

2.2.9 Cronograma

La estimación de las horas invertidas a lo largo de las distintas fases del programa de tareas queda descrito en la tabla 2.1, es una estimación que nos servirá como hoja de ruta:

Fase	Duración en horas
Investigación y formación	30
Preparación	10
Especificación de requisitos y análisis	40
Diseño	40
Implementación	130
Pruebas	20
Documentación	30
Total	300

Cuadro 2.1: Tabla duración en horas

De esta manera, a través de la figura 2.1, podemos ver mediante un gráfico la distribución de las horas:

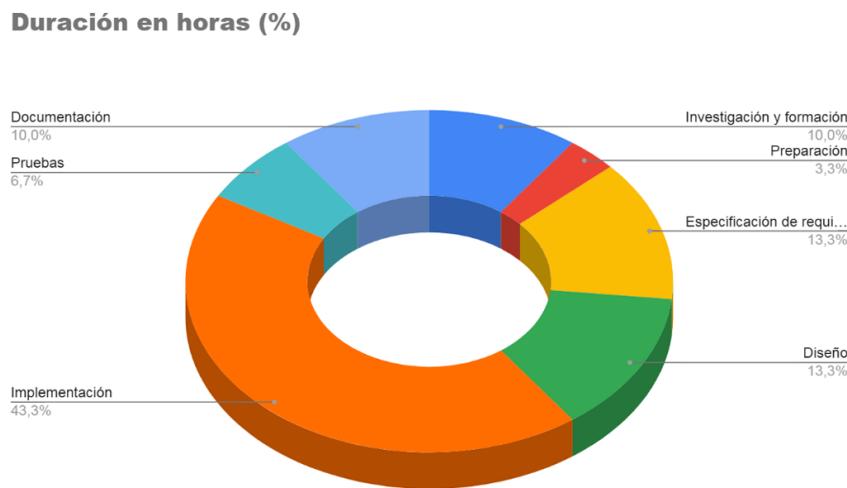


Figura 2.1: Gráfico de horas

2.2.10 Pruebas

Se llevarán a cabo distintos tipos de pruebas para en todo momento evaluar el estado del proyecto:



- **Pruebas unitarias:** Estas pruebas unitarias se realizarán con el fin de descubrir casos de uso que puedan dar lugar a fallos inesperados o no controlados por la aplicación, induciendo así acciones correctoras conforme a los resultados obtenidos y futuras mejoras. Estos tipos de pruebas serán efectuadas para verificar el cumplimiento de los requisitos mediante la lógica de negocio propuesta.
- **Pruebas de rendimiento:** Estas pruebas tendrán como objetivo ver como responde el sistema en situaciones de estrés o comprobar si la respuesta es lo suficientemente rápida.
- **Pruebas en dispositivos reales:** Si bien en la fase de desarrollo se utilizarán emuladores y entornos controlados, se llevará el sistema a entornos reales para comprobar el desempeño real de la aplicación. No se harán pruebas específicas documentadas al respecto, pues paulatinamente se irán probando las funcionalidades en los distintos dispositivos que se poseen.
- **Pruebas de inteligencia artificial aplicada:** Estas pruebas se realizarán dada la naturaleza del proyecto, para evaluar la eficacia y precisión de distintos sistemas de inteligencia artificial que podremos usar para la evaluación de las actividades en la aplicación.

2.2.11 Seguridad

Durante el desarrollo de la aplicación se aplicarán buenas prácticas en lo relativo al tratamiento de datos y la seguridad de los mismos relativas al uso de la aplicación. Se desarrollará teniendo en cuenta la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. [5].

Las comunicaciones también serán seguras, usando en todo momento el protocolo HTTPS [4] y se encriptará la información debidamente. Nunca se enviarán las contraseñas en texto plano, y tampoco se almacenará en la base de datos en texto plano, siendo completamente indescifrables por cualquier usuario, incluido el administrador de la base de datos.

Capítulo 3

Objetivos



El objetivo principal de este proyecto de fin de grado es confeccionar una aplicación móvil que sea útil tanto a nivel de docente permitiendo evaluar de una manera más rápida y eficaz, como a nivel de alumnado, para que éste pueda ser atendido de la manera más rápida y personalizada posible.

No obstante, no es el único objetivo, pues en especial en este proyecto presentamos otro objetivo importante. Otro de los grandes objetivos es el estudio y la aplicación de técnicas de Inteligencia Artificial aplicada para buscar diferentes alternativas para resolver nuestra problemática y estudiar los distintos resultados obtenidos.

Debido a la dualidad que se presenta, podríamos dividir los objetivos en dos grandes grupos, objetivos de la aplicación móvil, y objetivos de la Inteligencia Artificial aplicada. No sin exponer al terminar estos, una serie de objetivos más genéricos que no se podrían encasillar en ninguna de las dos secciones anteriores.

3.1 Objetivos de la aplicación móvil

Son los objetivos relacionados con el desarrollo de la misma:

1. **Autenticación.** Se creará un sistema de autenticación que permita a los usuarios iniciar sesión, cerrar la sesión, y registrarse. Todas las funcionalidades relacionadas con el tratamiento de información sensible será llevada a cabo siguiendo buenas prácticas de seguridad.
2. **Roles.** La aplicación mostrará información personalizada a cada usuario en función de su rol o de sus datos. Los roles diferenciados serán Profesor y Alumno.
3. **Asignaturas.** La aplicación permitirá gestionar las asignaturas, la creación, edición, o eliminación de las mismas desde el rol de Profesor, así como la inscripción en las mismas desde el rol de Alumno.
4. **Inscripciones.** La aplicación protegerá las inscripciones no deseadas mediante un sistema de código de acceso permitiendo al Profesor expulsar alumnos cuando así lo desee.
5. **Actividades.** Los profesores podrán crear actividades personalizadas, indicando un enunciado, un lenguaje de programación en el caso de que se así lo requiera, fecha límite de entrega y unos aspectos a valorar para la posterior corrección del a misma. Los alumnos por su parte, podrán visionar en todo momento esta información.
6. **Entregas.** Los alumnos podrán realizar las entregas de las actividades siempre y cuando estén dentro del tiempo límite establecido por el profesor, obteniendo inmediatamente una retroalimentación de la solución aportada y una calificación, la cual podrá ser editada por el profesor a posteriori si así lo considera.
7. **Experiencia de Usuario.** Un objetivo fundamental es valorar la experiencia de usuario, buscando facilitar al usuario el uso de la aplicación así como mejorar la presentación visual mediante el uso de iconos descriptivos o paletas de colores. A nivel del desarrollo, se buscará hacer un código mantenable haciendo uso de buenas



prácticas y realizando el correspondiente manual de código que permita tener un código mantenable y escalable, abierto a futuras mejoras.

3.2 Objetivos de la Inteligencia Artificial aplicada.

Son los objetivos relacionados con la aplicación de la Inteligencia Artificial para resolver los problemas mediante la misma:

1. **Estudiar y analizar** distintos flujos de aplicación de la Inteligencia Artificial para la resolución del problema.
2. **Distinguir dos etapas** que funcionan en conjunto. La etapa de reconocimiento de caracteres y la etapa de corrección de la solución.
3. **Explorar las distintas alternativas** y modelos que puedan ser aplicados para la solución, así como realizar un estudio comparativo argumentando diversos pros y contras de las alternativas disponibles.

Capítulo 4

Antecedentes



CAPÍTULO 4. ANTECEDENTES

En esta sección se hará un recorrido por distintas soluciones que incorporan el uso de Inteligencia Artificial en el ámbito educativo mediante una aplicación móvil, así como distintas herramientas que cumplen con algunas de las características que buscamos con este proyecto y toda aquella información previa que nos sirva como base para la ejecución del proyecto, incluyendo las tecnologías relacionadas con la inteligencia artificial que se usan hoy día y que nos servirán para la elaboración de nuestro proyecto

4.1 Moodle

Moodle es un sistema de gestión de aprendizaje popularmente conocido por su versatilidad y por ser de código abierto, ampliamente utilizado por instituciones educativas y empresas para la gestión en línea del contenido. Permite crear y organizar cursos, en los cuales los profesores o administradores de estos pueden gestionar aspectos como los documentos, el temario, encuestas, evaluaciones para realizar el seguimiento de su progreso, o comunicarse con los estudiantes. Al ser de código abierto, tiene una gran comunidad detrás que desarrolla plugins para personalizar aún más la plataforma y conseguir una adaptación a las necesidades de la institución educativa, esto además provoca que dicha plataforma tenga una mejora continua y actualizaciones que enriquecen la experiencia, funcionalidad, y estabilidad de la plataforma. [6]. Es el sistema utilizado por la Universidad de Córdoba, usado durante años debido a la fiabilidad que tiene y el alto grado de personalización con el que se cuenta, los alumnos de la universidad de esta forma pueden tener digitalizados los contenidos que se imparten en los distintos cursos como vemos en la figura 4.1

Figura 4.1: Cursos Moodle

Tal y como apreciamos en la figura 4.2, también permite la gestión del contenido de los cursos, como apuntes o documentos relacionados con el temario, si bien es cierto que no es el objetivo de nuestra aplicación, se podría apuntar como futura mejora.



CAPÍTULO 4. ANTECEDENTES

The screenshot shows a Moodle course page for 'Grado en Ingeniería Informática'. The page title is 'PROYECTOS'. It displays a list of course members and their roles, including 'Profesores' and 'Alumnos'. Below this is a section titled 'Área de Conocimiento' with the sub-section 'Proyectos de Ingeniería'. A sidebar on the left contains links for 'Avisos', 'Foro para uso general', 'Guía docente de la asignatura', 'Orthografía RAE', 'Actualización de la programación a 7/7/2022', and 'Grupos prácticas'. At the bottom of the sidebar, there is a collapsed section labeled 'Tema 1'.

Figura 4.2: Contenidos de un curso Moodle

Recientemente incorpora algunas funcionalidades nativas relacionadas con la aplicación de Inteligencia Artificial[7]. Un ejemplo de ello lo podemos ver en la imagen 4.3, que representa la funcionalidad de generación de imágenes mediante inteligencia artificial.

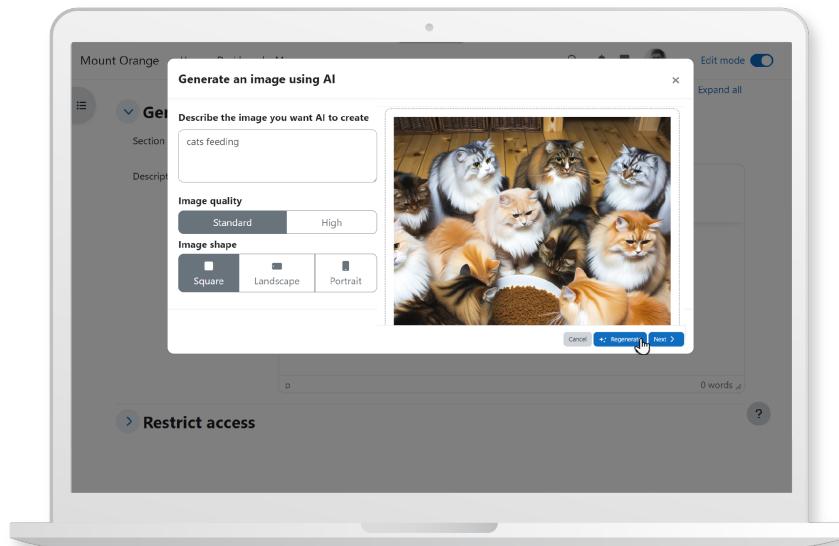


Figura 4.3: Moodle IA Generación de imágenes



4.2 Magicschool.ai

MagicSchool.ai es una plataforma de I.A. diseñada para educadores, con herramientas que se actualizan continuamente y que abarcan diversas materias y niveles educativos. Su objetivo principal es reducir la carga de trabajo de los profesores, ahorrándoles entre 7 y 10 horas semanales al agilizar la preparación y planificación de clases.[8] Representa un ejemplo relevante de cómo la IA puede ser implementada para automatizar tareas docentes. Su interfaz intuitiva, prompts guiados que facilitan la adaptación a las necesidades específicas del usuario. Además, promueve la alfabetización en IA al permitir su uso por parte de los estudiantes, fomentando el aprendizaje sin reemplazar el pensamiento crítico. No obstante, no permite gestionar asignaturas y al alumnado, se puede considerar más como un kit de herramientas para, como hemos dicho antes, elaborar contenido de cara a la enseñanza, a lo largo de múltiples herramientas como las que vemos en la figura 4.4

The screenshot shows the MagicSchool.ai platform interface. At the top, there's a purple header bar with a search bar labeled "Search Tools" and a star icon. Below the header, the main interface is titled "All Tools". On the left, there's a sidebar with several icons and labels: "Magic Tools" (highlighted), "Raina (Chatbot)", "Output History", "Launch to Students", "Love", "Training", "Share the Magic", "Student AI Literacy", and "Enterprise". A large banner at the bottom left says "TEACHERS ARE MAGIC". The main area displays a grid of 12 tool cards, each with an icon, title, and brief description. The tools include:

- Create a Custom Tool**: Generate and edit images with AI from Adobe Express.
- Text Rewriter**: Take any text and rewrite it with custom criteria however you'd like!
- Lesson Plan**: Generate a lesson plan for a topic or objective you're teaching.
- Worksheet Generator**: Generate a worksheet based on any topic or text.
- Report Card Comments**: Generate report card comments with a student's strengths and areas for growth.
- Academic Content**: Generate original academic content customized to the criteria of your choice.
- Informational Texts**: Generate original informational texts for your class, customized to the topic of your choice.
- YouTube Video Questions**: Generate guiding questions aligned to a YouTube video.
- Rubric Generator**: Have AI write a rubric for an assignment you are creating for your class in a table format.
- Text Proofreader**: Take any text and have it proofread, correcting grammar, spelling, punctuation and adding clarity.
- Text Dependent Questions**: Generate text-dependent questions for students based on any text that you input.
- IEP Generator**: Generate a draft of an individualized education program (IEP) customized to a student's needs.
- Unit Plan Generator**: Generate a draft of a unit plan based on topic, standards and objectives and length of unit.

Figura 4.4: Magicschool

4.3 Mathew.ai

Mathew.ai es una plataforma educativa que utiliza inteligencia artificial para personalizar la experiencia de aprendizaje de los estudiantes. La plataforma es capaz de generar actividades y situaciones de aprendizaje adaptadas al currículo y a las necesidades individuales de cada estudiante. Esto permite a los profesores ofrecer un enfoque más personalizado en el proceso de enseñanza-aprendizaje. Además, Mathew.ai evalúa automáticamente las respuestas de los estudiantes según las indicaciones del profesor [9], proporcionando una puntuación precisa y objetiva por competencias. Esta función de evaluación automática es especialmente útil para reducir la carga de trabajo de los docentes y asegurar una calificación consistente tal y como establecimos en los objetivos de nuestro proyecto, no obstante, no ofrece aplicación móvil para poder beneficiarse de las ventajas



CAPÍTULO 4. ANTECEDENTES

de su uso. Si bien es cierto que no está muy extendido el uso de esta aplicación, a través de la figura 4.5 y la figura 4.6 podemos ver que el flujo es como el que hemos indicado y que persigue unos objetivos muy similares a los que nosotros queremos conseguir.

Gran variedad de tareas

The screenshot shows a mobile application interface for 'Resultados actividad'. On the left, there's a summary card for a task titled 'Grabación de los descubrimientos' (Completed) in '1º ESO - Biología y Geología' class 'Clase Els Barrufets' from 3/6/2024 to 8/6/2024. On the right, a table lists student results: Rossell Marcos (Erikson, 8.4), Rodriguez Pérez (Rafael, 5.8), Velasco Sánchez (María, 3.3), Tristán Velázquez (Leo, 5.5), López García (Sandra, 8.2), Meréndez Luján (Andrés, -), and Prat Sarret (Marta, -). Each row has a 'Revisar' (Review) or 'Evaluar' (Evaluate) button.

Figura 4.5: Mathew ai

The screenshot shows a mobile application interface for 'Evaluación automática'. It displays a summary of student work: '6/10' overall score, '7/10' for 'Dominio', '3/10' for 'Vocabulario', and '8/10' for 'Comprendimiento'. Below this, there's a 'Comentario del profesor para el estudiante' (Teacher comment for student) section with a detailed response about fossils, a 'Propuesta de comentario para tu estudiante' (Comment proposal for student) section, and a 'Quiero esta función para mi escuela' (I want this feature for my school) button.

Figura 4.6: Mathew ai

4.4 Motores OCR

Un OCR (Optical Character Recognition) es una tecnología que permite convertir documentos escaneados, imágenes o fotografías en texto editable y procesable. Su uso es fundamental en diversas aplicaciones, como la digitalización de documentos, la extracción automática de información y la accesibilidad. Existen múltiples motores de OCR, tanto de código abierto como propietarios, cada uno con distintas capacidades y niveles de precisión.



4.4.1 Tesseract OCR

Tesseract es un motor de reconocimiento óptico de caracteres (OCR) de código abierto, desarrollado por Hewlett-Packard y mantenido actualmente por Google. Es uno de los OCR más usados actualmente debido a su precisión y capacidad para reconocer texto en un gran abanico de situaciones en las imágenes así como de idiomas. Es capaz de convertir imágenes o documentos escaneados en texto. Es bastante flexible para adaptarse a distintos estilos de texto e idiomas, admite un entrenamiento personalizado, por lo que podríamos adaptarlo a nuestras necesidades. [10]

4.4.2 Google OCR

Google OCR, basado en la API de Google Cloud Vision, es una de las soluciones más avanzadas en el reconocimiento óptico de caracteres. Utiliza modelos de aprendizaje automático y redes neuronales para detectar y extraer texto en imágenes con alta precisión. Soporta múltiples idiomas y es capaz de reconocer estructuras complejas como tablas y formularios. Su integración con otros servicios de Google Cloud facilita su uso en entornos empresariales [11].

4.4.3 Azure Vision OCR

Azure Vision OCR es el servicio de reconocimiento óptico de caracteres de Microsoft, integrado en Azure Cognitive Services. Destaca por su capacidad para extraer texto de imágenes y documentos en distintos formatos, incluyendo manuscritos. Utiliza inteligencia artificial para mejorar la precisión y es capaz de detectar texto en documentos estructurados y no estructurados. Es una opción robusta para empresas que ya trabajan con el ecosistema de Microsoft [12]. Como está integrado en los servicios en la nube que ofrece Microsoft, podemos ver como algo bueno y/o malo, el hecho de que la capacidad de cómputo se externaliza y liberamos la sistema de carga, de igual manera ocurre con Google OCR, no obstante, tenemos que ser conscientes de que nuestros datos pasan por ellos.

4.4.4 Easy OCR

EasyOCR es un motor de reconocimiento de texto de código abierto basado en redes neuronales profundas. Desarrollado en Python, es una alternativa ligera y flexible que permite la extracción de texto en más de 80 idiomas. A diferencia de otros motores, EasyOCR es fácil de integrar en proyectos de inteligencia artificial y visión por computadora, lo que lo hace ideal para desarrolladores que buscan una solución rápida y personalizable [13].

4.4.5 Paddle OCR

PaddleOCR es un motor de reconocimiento de texto desarrollado por Baidu como parte del framework PaddlePaddle. Se destaca por su enfoque basado en deep learning y su alto rendimiento en la detección y reconocimiento de texto en distintos escenarios. Es ampliamente utilizado en aplicaciones de visión artificial y cuenta con modelos preentrenados optimizados para diferentes lenguajes y tipos de documentos [14]. Tanto este OCR como el anterior (EasyOCR), son de código abierto lo cual permite que autoalojemos estos



servicios e incluso que los entrenemos con datasets, lo cual por un lado significa carga de cómputo en el servidor pero por otro garantiza la privacidad y el hecho de que podamos entrenar el OCR con nuestros sets de datos también supone una gran ventaja

4.4.6 Comparativa Tesseract - Textract - Azure OCR - Google OCR

Dado el amplio abanico de herramientas de OCR disponibles, tanto de código abierto como comerciales, es crucial evaluar su desempeño en términos de precisión, velocidad y capacidad de adaptación a diferentes tipos de documentos. En este contexto, Ricciuti [15] llevó a cabo un estudio comparativo de cuatro herramientas de OCR ampliamente utilizadas: Tesseract OCR, Amazon Textract, Azure OCR y Google OCR.

El análisis presentado por Ricciuti tiene como objetivo evaluar la eficacia de las distintas herramientas de OCR en la extracción de texto, especialmente en documentos escaneados de baja calidad. Para ello, se utilizó el dataset FUNSD [16], un conjunto de datos que contiene 199 imágenes en escala de grises de formularios escaneados, con anotaciones manuales que describen la ubicación de los textos en los documentos originales. En la figura 4.7 vemos la distribución de imágenes usadas para el entrenamiento.

Training Set	
Number of images	7411
Mean height	19.033
Mean width	106.816
Mean text length	16.993
Test Set	
Number of images	2332
Mean height	20.119
Mean width	108.159
Mean text length	21.674

Figura 4.7: Dataset para la comparación

A partir de este conjunto de datos, se generó un *OCR Evaluation Dataset*, compuesto por sub-imágenes que representan campos individuales de los formularios. Las herramientas evaluadas procesaron estas sub-imágenes para extraer el texto correspondiente, el cual fue comparado con los valores reales proporcionados por el dataset de referencia.

Los resultados obtenidos indicaron que las herramientas comerciales, Azure OCR y Google OCR, ofrecieron un mejor desempeño en términos de precisión y similitud de extracción de texto, gracias a sus avanzados modelos de inteligencia artificial y procesamiento del lenguaje natural. En contraste, Tesseract OCR, a pesar de ser una solución de código abierto ampliamente utilizada, presentó ciertas limitaciones en la precisión de reconocimiento, especialmente en documentos con estructuras complejas o calidad reducida. Amazon Textract, por su parte, mostró un rendimiento intermedio, con ventajas en el procesamiento estructurado de documentos, pero con una precisión menor en la extracción de caracteres individuales.



En la figura 4.8 vemos los resultados obtenidos de esta comparativa, siendo *accuracy* la precisión del modelo y *similarity* qué tan cercano es el texto obtenido al original.

	Accuracy	Similarity
Tesseract OCR	0.306	0.569
Azure OCR	0.633	0.915
Amazon Textract	0.345	0.513
Google OCR	0.576	0.892
EasyOCR	0.284	0.719

Figura 4.8: Resultados comparativa OCR

4.5 Modelos de lenguaje

Los LLM (Large Language Models) son modelos de inteligencia artificial diseñados para procesar y generar lenguaje natural a gran escala. Estos modelos se basan en redes neuronales profundas, específicamente en arquitecturas de aprendizaje profundo como los transformadores, introducidos por Vaswani et al. en 2017 [17]. Su entrenamiento se realiza con grandes volúmenes de datos textuales, lo que les permite comprender y generar texto de manera coherente y contextualizada.

4.5.1 ChatGPT

ChatGPT es un modelo de lenguaje basado en la arquitectura GPT (Generative Pre-trained Transformer)[18], desarrollado por OpenAI. Se ha convertido en una de las herramientas de inteligencia artificial más utilizadas en diversos ámbitos, incluyendo la educación, debido a su capacidad para comprender y generar texto de manera coherente y contextual. Estos modelos se caracterizan por evolucionar muy rápido, pues OpenAI ha ido lanzando al mercado modelos cada vez más potentes y con más prestaciones como vemos en la comparativa de la figura 4.9

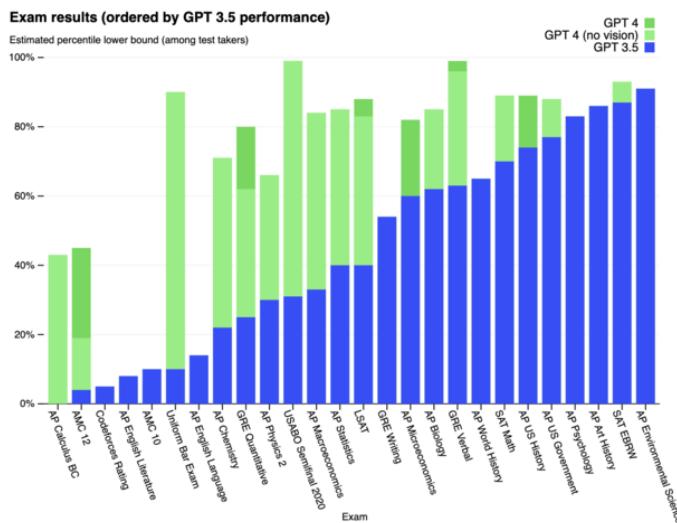


Figura 4.9: ChatGPT3.5 vs ChatGPT4



En el ámbito educativo, ChatGPT se emplea para responder preguntas, explicar conceptos complejos, generar ejercicios, corregir redacciones y ofrecer tutoría personalizada a los estudiantes. Su capacidad para procesar lenguaje natural lo hace útil como asistente virtual en plataformas de aprendizaje, permitiendo una interacción más fluida entre los usuarios y la tecnología.

Además, se ha integrado en diversas aplicaciones móviles y entornos de enseñanza digital, facilitando el acceso a la información y mejorando la experiencia del aprendizaje autodidacta. Sin embargo, a pesar de sus ventajas, su uso en educación plantea desafíos, como la necesidad de supervisión para evitar respuestas incorrectas o sesgadas, y la importancia de fomentar el pensamiento crítico en los estudiantes.

La implementación de ChatGPT en plataformas como Moodle o herramientas educativas personalizadas, como es el caso del presente proyecto, podría mejorar la interacción y asistencia a los estudiantes, proporcionando retroalimentación automatizada y promoviendo el aprendizaje adaptativo.

4.5.2 Gemini

Gemini es una familia de modelos de lenguaje desarrollados por Google DeepMind, diseñada para tareas avanzadas de procesamiento de lenguaje natural e inteligencia artificial generativa. Se basa en arquitecturas avanzadas de transformadores y ha sido entrenado con grandes volúmenes de datos multilingües y multimodales, lo que le permite procesar no solo texto, sino también imágenes, audio y video [19].

Uno de los aspectos más destacados de Gemini es su capacidad para razonar de manera más estructurada y generar respuestas más precisas en comparación con modelos anteriores. Google ha integrado esta tecnología en su ecosistema de productos, como Google Search, Bard y Workspace, ofreciendo una experiencia más rica en la interacción con los usuarios.

En educación, Gemini se puede utilizar como herramienta educativa al igual que chatgpt, pues no deja de ser un modelo de lenguaje con unas prestaciones parecidas. Además, de cara a nuestro proyecto, recientemente Google liberó una API gratuita para interactuar con el modelo Gemini 1.5 flash, la cual podríamos integrar en nuestro proyecto ya que tiene muy buenas prestaciones.[20] En la figura 4.10 podemos ver una comparativa entre los modelos de gemini actuales usando diferentes benchmarks.



CAPÍTULO 4. ANTECEDENTES

CAPABILITY	BENCHMARK	DESCRIPTION	1.5 Flash	1.5 Pro	2.0 Flash-Lite Preview	2.0 Flash GA	2.0 Pro Experimental
General	MMLU-Pro	Enhanced version of popular MMLU dataset with questions across multiple subjects with higher difficulty tasks	67.3%	75.8%	71.6%	77.6%	79.1%
Code	LiveCodeBench v6	Code generation in Python. Subset covering more recent examples [in the UI: 10/01/2024 – 02/01/2028]	30.7%	34.2%	28.9%	34.5%	36.0%
	Bird-SQL Dev	Benchmark evaluating converting natural language questions into executable SQL	45.6%	54.4%	57.4%	58.7%	59.3%
Reasoning	GPQA Diamond	Challenging dataset of questions written by domain experts in biology, physics, and chemistry	51.0%	59.1%	51.5%	60.1%	64.7%
Factuality	SimpleQA	World knowledge factuality with no search enabled	8.6%	24.9%	21.7%	29.9%	44.3%
	FACTS Grounding	Ability to provide factuality correct responses given documents and diverse user requests	82.9%	80.0%	83.6%	84.6%	82.8%
Multilingual	Global MMLU Lite	MMLU translated by human translators into 15 languages. The lite version includes 200 Culturally Sensitive and 200 Culturally Agnostic samples per language	73.7%	80.8%	78.2%	83.4%	86.5%
Math	MATH	Challenging math problems (incl. algebra, geometry, pre-calculus, and others)	77.9%	86.5%	86.8%	90.9%	91.8%
	HiddenMath	Competition-level math problems. Held out dataset AIME/AMC-like, crafted by experts and not leaked on the web	47.2%	52.0%	55.3%	63.5%	65.2%
Long-context	MRCR 1M	Novel, diagnostic long-context understanding evaluation	71.9%	82.6%	58.0%	70.5%	74.7%
Image	MMMU	Multi-discipline college-level multimodal understanding and reasoning problems	62.3%	65.9%	68.0%	71.7%	72.7%
Audio	CoVoST2 21lang	Automatic speech translation (BLEU score)	37.4%	40.1%	38.4%	39.0%	40.6%
Video	EgoSchema test	Video analysis across multiple domains	66.8%	71.2%	67.2%	71.1%	71.9%

Figura 4.10: Comparativa Gemini

4.5.3 DeepSeek

DeepSeek es un modelo de lenguaje diseñado para mejorar la recuperación de información y la generación de texto con un enfoque avanzado en la precisión y relevancia contextual. Este modelo se basa en arquitecturas de transformadores optimizadas, lo que le permite procesar grandes volúmenes de datos y ofrecer respuestas más precisas en tareas de generación y análisis de lenguaje natural [21].

Lo más interesante de DeepSeek, es su nuevo modelo razonador, DeepSeekR1, sin entrar en muchos detalles, un modelo razonador es una inteligencia artificial diseñada para simular el proceso de razonamiento humano, es decir, la capacidad de analizar información, hacer inferencias y tomar decisiones de manera lógica, explicando el por qué en cada una de ellas. Para hacernos una idea de la potencia que tiene, en la figura 4.11 vemos una comparativa del modelo razonador junto con otros modelos como los de openAI o Qwen.

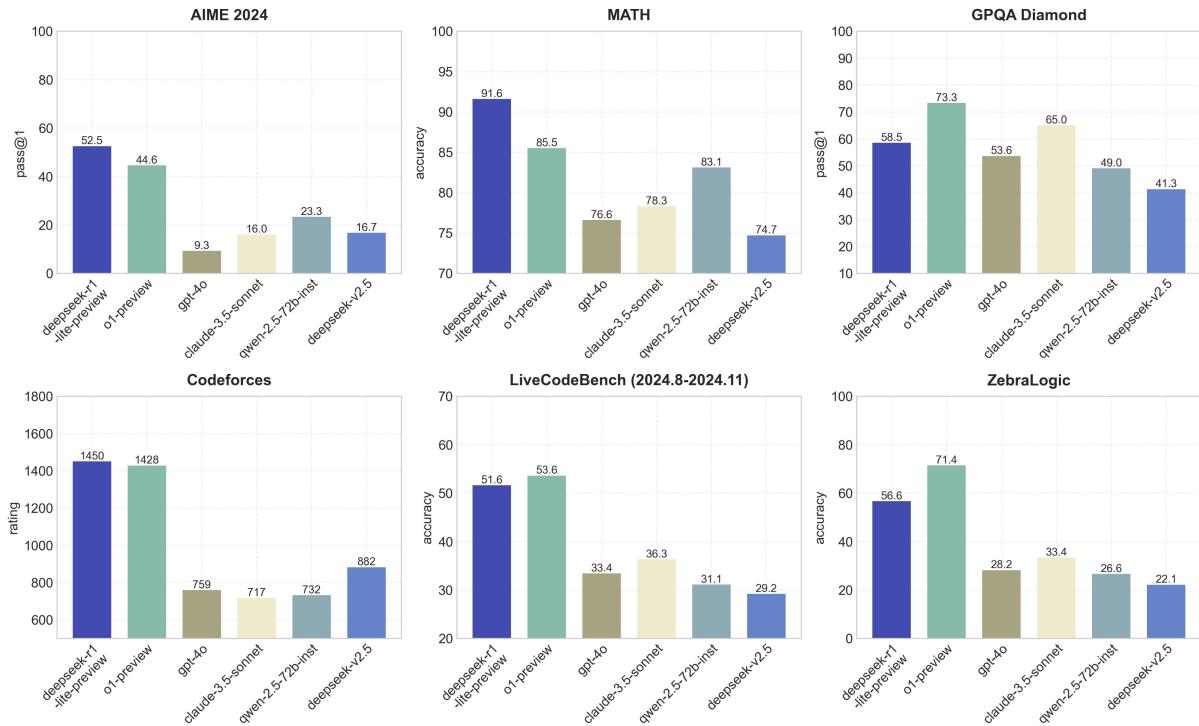


Figura 4.11: Comparativa Deepseek

En el ámbito educativo, DeepSeek se puede aplicar en la generación automatizada de resúmenes, apoyo en la investigación académica y asistencia en la escritura de ensayos y artículos científicos. Su capacidad para realizar análisis semánticos profundos lo hace útil en disciplinas que requieren la interpretación y síntesis de información compleja, en especial con el modelo razonador, pero, al final, sigue siendo un uso parecido al de los modelos anteriores, solo que con otras prestaciones.

4.5.4 Qwen

Qwen es una familia de modelos de lenguaje desarrollados por la empresa Alibaba, diseñada para tareas avanzadas de procesamiento de lenguaje natural, con especial énfasis en la generación de texto y la comprensión contextual. Utiliza arquitecturas de transformadores similares a las de otros modelos de lenguaje, pero se destaca por su enfoque en la eficiencia y la capacidad de generalización a tareas diversas sin la necesidad de grandes ajustes [22].

Uno de los aspectos más interesantes de Qwen es su capacidad para adaptarse a una variedad de tareas de generación de texto, como redacción automática, resumen de contenido, generación de código y mucho más, a través de su modelo de lenguaje de propósito general. Este modelo es especialmente útil en entornos de desarrollo y empresas que buscan integrar inteligencia artificial en sus procesos sin la necesidad de un ajuste fino continuo. La versión 2.5 VL multimodal, llega a superar a modelos como los anteriores nombrados como podemos ver en la figura 4.12



CAPÍTULO 4. ANTECEDENTES

		Qwen2.5-VL 72B	Gemini-2 Flash	GPT-4o	Claude3.5 Sonnet	Qwen2-VL 72B	Other Best Open LLM
College-level Problems	MMMU	70.2	70.7	70.3	70.4	64.5	70.1
	MMMU Pro	51.1	57.0	54.5	54.7	46.2	52.7
Document and Diagrams Reading	DocVQA	96.4	92.1	91.1	95.2	96.5	96.1
	InfoVQA	87.3	77.8	80.7	74.3	84.5	84.1
	CC-OCR	79.8	73.0	66.6	62.7	68.7	68.7
General Visual Question Answering	OCR Benchv2	61.5	-	46.5	45.2	47.8	47.8
	MegaBench	51.3	55.2	54.2	52.1	46.8	47.4
	MMStar	70.8	69.4	64.7	65.1	68.3	69.5
	MBench1.1	88.0	83.0	82.1	83.4	84.6	87.4
Math	MathVista	74.8	73.1	63.8	65.4	70.5	72.3
	MathVision	38.1	41.3	30.4	38.3	25.9	32.2
Video Understanding	VideoMME	73.3	-	71.9	60.0	71.2	72.1
	MBench-Video	2.0	-	1.7	1.4	1.7	1.9
	LVBench	47.3	-	30.8	-	-	43.6
Visual Agent	CharadesSTA	50.9	-	35.7	-	-	48.4
	AITZ	83.2	-	35.3	-	-	53.3
	Android Control	67.4	-	-	-	-	66.4
	ScreenSpot	87.1	84.0	18.1	83.0	-	89.5
	ScreenSpot Pro	43.6	-	-	17.1	-	38.1
	AndroidWorld	35.0	-	34.5	27.9	-	46.6
	OSWorld	8.8	-	5.0	14.9	-	22.7

Figura 4.12: Comparativa Qwen2.5-VL

En educación que es nuestro contexto, Qwen puede utilizarse para la creación automatizada de contenido educativo, generación de ejemplos prácticos, asistencia en la corrección de tareas, y la tutoría de estudiantes en diversos campos. Además, su modelo de lenguaje de propósito general permite una integración fácil en plataformas educativas y herramientas personalizadas, ofreciendo respuestas precisas y contextuales a preguntas de los estudiantes, como el resto de los modelos vistos con anterioridad

Capítulo 5

Restricciones



A lo largo del capítulo presente, se exponen las restricciones que han de tenerse en cuenta para el desarrollo del proyecto, condicionando de una manera u otra dicho desarrollo. Diferenciaremos dos grupos de factores y los detectaremos para el desarrollo de estas restricciones:

1. **Factores dato.** Son aquellos intrínsecos al problema, no podrán ser modificados y condicionarán inevitablemente al proyecto,
2. **Factores estratégicos.** Son aquellos sobre los que tendremos capacidad de decisión, pero, la solución final dependerá de dicha toma de decisión, por lo que se plantearán distintas soluciones y escenarios para elegir la mas conveniente.

5.1 Factores dato

Estos factores inherentes al problema son:

1. La aplicación ha de ser gratuita.
2. La aplicación requiere conexión a internet para su correcto funcionamiento.
3. Se requiere una gran capacidad de cómputo en el servidor para desplegar los modelos de I.A. correctamente.
4. La aplicación ha de disponer de acceso a cámara para realizar las fotografías.
5. La aplicación debe estar abierta a otros fines distintos a la programación.
6. La aplicación deberá de distinguir entre distintos roles y en función de los mismos permitir realizar unas acciones u otras.
7. El proyecto ha de ser finalizado antes del 30 de Mayo de 2025 no superando las 300 horas estipuladas para el desarrollo del Trabajo Final de Grado.
8. El proyecto ha de ser desarrollado en el equipo del autor del mismo con las limitaciones que esto suponga computacionalmente hablando.
9. Se dispondrá de un servidor en la Universidad de Córdoba para el despliegue de los modelos utilizados de Inteligencia Artificial.
10. El proyecto será desarrollado únicamente por el autor del mismo, junto con el apoyo y la guía del director del proyecto.

5.2 Factores estratégicos

Estos factores que condicionarán el diseño final son:



5.2.1 Sistema operativo objetivo

La aplicación será desarrollada para Android, no obstante, en un futuro se podría sacar una versión para iOS. La justificación es simple, es el sistema operativo de dispositivos móviles que más cuota de mercado abarca en España según statista[23] tal y como vemos en la figura 5.1

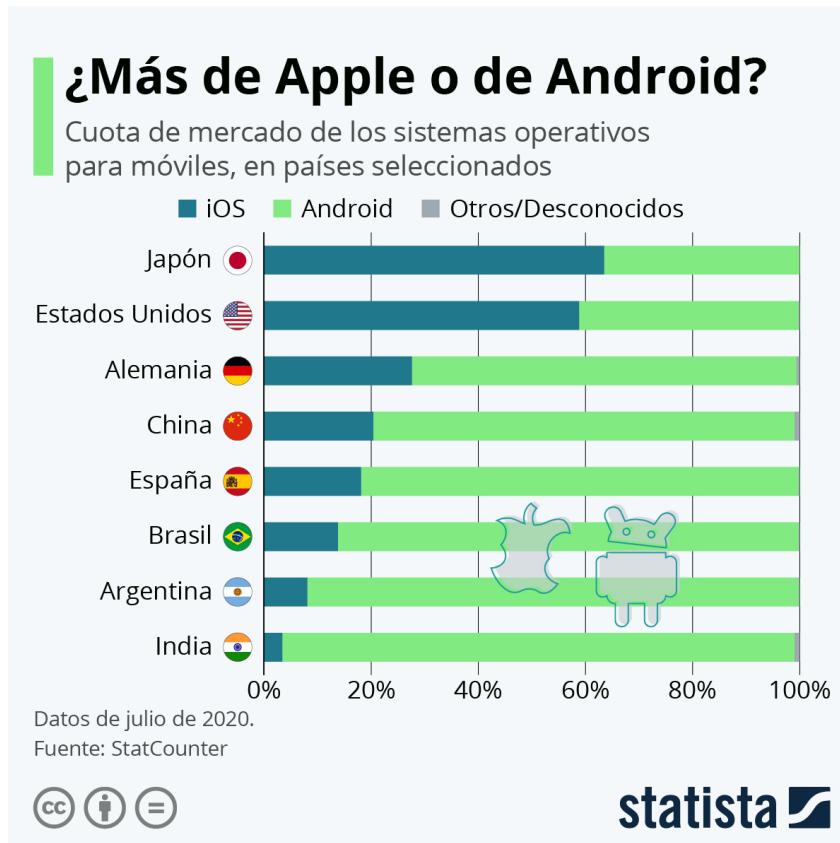


Figura 5.1: Cuota de mercado Android

5.2.2 Entorno de desarrollo

El entorno de desarrollo (IDE) se puede definir como una aplicación informática que proporciona servicios integrales para facilitar al desarrollador o programador el desarrollo de software.[24]. Entre las diversas opciones disponibles, destacan Android Studio y Visual Studio Code. El primero de ellos, es bastante conveniente para el desarrollo nativo de aplicaciones móviles para la plataforma Android, sin embargo, Visual Studio Code, debido a su gran cantidad de plugins es fácilmente integrable con cualquier tipo de entorno necesario para programar, independientemente de la plataforma, además, es el IDE con el que más he trabajado personalmente, por lo que me decantaré por Visual Studio Code.

5.2.3 Kit de desarrollo para la aplicación móvil

El kit de desarrollo determinarán aspectos como el rendimiento, la facilidad y rapidez para el desarrollo, disponemos de varias opciones, pero en concreto para Android las opciones se reducirán por decisiones de diseño a React Native y Flutter.



React Native es un framework que permite a los desarrolladores usar JavaScript, un lenguaje ampliamente conocido y adoptado, React Native facilita la creación de aplicaciones móviles que se sienten completamente nativas. Su arquitectura orientada a componentes promueve una experiencia de usuario fluida y cohesiva, haciendo posible que las aplicaciones accedan a las funcionalidades nativas del dispositivo de manera directa y eficiente.[25]

Por su parte, Flutter ha ganado terreno rápidamente gracias a su enfoque único. Utilizando Dart, Flutter permite a los desarrolladores construir interfaces de usuario ricas y personalizables a través de un extenso catálogo de widgets. Esta aproximación no solo facilita una experiencia de usuario excepcional sino que también promueve un desarrollo ágil y versátil, ya que con algo de esfuerzo con una única base de código se pueden desarrollar aplicaciones multiplataforma. [25]

Para determinar la elección, elaboramos una exhaustiva lista de pros y contras de cada uno de los frameworks basándonos en las citaciones anteriores.

5.2.4 Flutter

Ventajas:

- **Desempeño nativo sobresaliente:** Utiliza el motor gráfico Skia, permitiendo un rendimiento casi idéntico al de una aplicación nativa.
- **UI consistente en todas las plataformas:** Ofrece widgets personalizables que garantizan una interfaz homogénea en iOS y Android.
- **Hot Reload:** Permite a los desarrolladores ver los cambios en tiempo real sin necesidad de recompilar la aplicación.
- **Amplia comunidad y soporte de Google:** Cuenta con una comunidad en rápido crecimiento y documentación extensa.

Desventajas:

- **Tamaño de la app:** Las aplicaciones en Flutter suelen ser más grandes debido a la inclusión del motor gráfico y otros componentes.
- **Soporte limitado para integraciones nativas:** Algunas integraciones pueden requerir la escritura de código nativo en Java, Kotlin, Swift u Objective-C.
- **Curva de aprendizaje de Dart:** Flutter utiliza Dart, un lenguaje menos popular que JavaScript, lo que puede dificultar la adopción.

5.2.5 React Native

Ventajas:

- **Uso de JavaScript:** Facilita la transición para desarrolladores web sin necesidad de aprender un nuevo lenguaje.



- **Gran ecosistema y recursos:** Cuenta con un amplio número de paquetes, bibliotecas y herramientas.
- **Componente nativo:** Permite la integración con módulos nativos y APIs de terceros de forma sencilla.
- **Amplio soporte y comunidad activa:** Respaldado por Facebook, posee una gran comunidad de desarrolladores.

Desventajas:

- **Desempeño inferior en aplicaciones complejas:** Puede presentar limitaciones en apps con gráficos intensivos o animaciones complejas.
- **Actualizaciones y mantenimiento:** Las constantes actualizaciones pueden causar problemas de compatibilidad con paquetes de terceros.
- **Interfaz de usuario inconsistente:** Se basa en componentes nativos, lo que puede generar diferencias en la apariencia entre iOS y Android.
- **Mayor dependencia de módulos externos:** Necesita librerías de terceros para muchas funcionalidades nativas, lo que puede afectar su estabilidad.

5.2.6 Conclusión

La elección entre Flutter y React Native dependerá de los objetivos del proyecto y la experiencia del equipo de desarrollo. Flutter es ideal para aplicaciones que requieren alto rendimiento y una interfaz personalizada, mientras que React Native es una opción atractiva para desarrolladores con experiencia en JavaScript y que buscan una solución con un ecosistema robusto. En lo personal, la experiencia previa con Flutter es uno de los factores decisivos que he tenido en cuenta para elegirlo, por lo que, finalmente sabiendo que usaremos Flutter, estaremos forzados a usar Dart como lenguaje de programación principal para el desarrollo de la aplicación móvil.

5.3 Framework para el backend

El framework usado en el backend de la aplicación, de nuevo, determinará y condicionará las características de la solución final, así como el rendimiento o la capacidad de integración con servicios externos.

Los criterios que se ha tomado para la elección de este framework van desde el lenguaje de programación, rendimiento o la experiencia previa, hasta la facilidad de integración con servicios externos y soporte de la comunidad. Tomando en cuenta estos criterios, y tomando en cuenta también las arquitecturas que se pueden plantear para el proyecto, se plantean dos frameworks distintos: NodeJS y FastAPI.



5.3.1 Lenguaje de programación

En cuanto a los lenguajes de programación, podemos proponer diferentes alternativas, cada una con sus pros y contras, no obstante, para nuestro proyecto, dados los dos frameworks nombrados anteriormente, como opciones tenemos JavaScript (o TypeScript) para NodeJS, y Python para FastAPI. Ambos lenguajes han sido trabajados a lo largo del grado universitario, no es el caso de TypeScript, por lo que al no tener experiencia previa y necesitar un desarrollo rápido y preciso, decidí descartarlo.

En la tabla 5.1, podemos ver las diferencias principales entre ambos lenguajes de programación que podrán condicionar nuestro proyecto [26]:

Característica	JavaScript	Python
Paradigma	Multiparadigma (imperativo, orientado a objetos, funcional)	Multiparadigma (imperativo, orientado a objetos, funcional)
Tipado	Dinámico, pero débilmente tipado	Dinámico y fuertemente tipado
Ejecución	Interpretado, pero con compilación Just-In-Time (JIT)	Interpretado
Uso principal	Desarrollo web (frontend y backend)	Desarrollo backend, ciencia de datos, IA y automatización
Sintaxis	Más compleja y basada en C	Más sencilla y legible
Velocidad	Más rápido debido a su compilación JIT	Generalmente más lento en ejecución
Compatibilidad	Funciona en navegadores y servidores	Funciona en servidores y entornos locales
Frameworks populares	React, Angular, Vue, Node.js	Django, Flask, TensorFlow, PyTorch
Ámbito de uso	Aplicaciones web, móviles y backend	Backend, IA, automatización y análisis de datos

Cuadro 5.1: Comparación entre JavaScript y Python

Una vez sabemos las diferencias y similitudes, podemos elaborar una lista con una serie de ventajas y desventajas en nuestro contexto [26]:

JavaScript

■ Ventajas:

- Compatible con todos los navegadores.
- Ejecución rápida gracias a la compilación JIT.
- Amplia comunidad y ecosistema de bibliotecas.
- Permite desarrollo full-stack con Node.js.

■ Desventajas:

- Sintaxis más compleja en comparación con Python.
- No es el mejor para tareas de ciencia de datos o IA.
- Manejo de asincronía con promesas y callbacks puede ser confuso.

Python

■ Ventajas:

- Sintaxis clara y fácil de aprender.
- Amplio soporte para ciencia de datos, inteligencia artificial y automatización.
- Comunidad activa y abundante documentación.



- Multiplataforma y versátil para distintos ámbitos de desarrollo.
- Desventajas:

- Más lento en ejecución en comparación con JavaScript.
- No está diseñado para el desarrollo frontend.
- Gestión de hilos y concurrencia menos eficiente.

Parece que para el desarrollo web o de aplicaciones quizá JavaScript está mejor posicionado, pero cuando hablamos del backend, que es el objetivo principal de esta comparativa, la competencia se vuelve más interesante, ya que la existencia en el caso de Python de frameworks como Django, FastAPI, o Flask, hacen que sea una muy buena opción para el desarrollo de la lógica de negocio en el servidor. Otro aspecto interesante es el uso en tareas de ciencia de datos, machine learning, e inteligencia Artificial, Python en este caso le come mucho terreno a JavaScript, por lo que, para este caso, quizá es una mejor opción.

Otro factor a tener en cuenta para la elección del framework para el backend es la elección que ya hemos hecho para el frontend, si bien es cierto que Flutter puede trabajar con cualquiera de las alternativas propuestas, cabe resaltar que, una de las ventajas de NodeJS es que eligiendo React Native para el frontend, tenemos una aplicación fullstack desarrollada con JavaScript, pero, no es el caso presente, por lo que NodeJS junto JavaScript quedaría en un segundo plano.

La elección de Python nos lleva a elegir un framework entre Flask y FastAPI, ya que son los dos en los que he tenido experiencia previa, siendo el segundo de ellos uno de los que más he usado recientemente, por lo que esto le supone partir con cierta ventaja. En la tabla 5.2 podemos realizar una pequeña comparativa con algunos aspectos a tener en cuenta [27]:

Cuadro 5.2: Tabla comparativa FastAPI vs Flask

Aspecto	FastAPI	Flask
Casos de uso principales	APIs con necesidades de alto rendimiento	Aplicaciones web ligeras
Adecuación del proyecto	Ideal para proyectos grandes y escalables	Adecuado para proyectos pequeños y medianos
Rendimiento	Alto rendimiento, soporte asíncrono	Bueno, síncrono pero menos eficiente
Escalabilidad	Altamente escalable, capacidades asíncronas	Escalable, con las extensiones adecuadas
Manejo de rutas y peticiones	Avanzado, validación automática	Simple, enrutamiento flexible
Integración con bases de datos	Integración sencilla con bases de datos	Se integra bien, pero puede necesitar extensiones
Autenticación y autorización	Soporte incorporado, fácil de implementar	Flexible, requiere configuración manual

Continúa en la siguiente página



Tabla 5.2 (continuación)

Aspecto	FastAPI	Flask
Documentación	Documentación interactiva y completa	Bien documentado, pero menos interactivo
Pruebas	Compatible con pruebas, con dependencias	Testable, pero podría requerir más configuración
Comunidad	Comunidad creciente y activa	Comunidad grande y bien establecida
Ecosistema	Ecosistema rico, librerías modernas	Ecosistema maduro, numerosas extensiones
Curva de aprendizaje	Más pronunciada debido a características avanzadas	Más fácil de aprender, curva de aprendizaje suave
Facilidad de uso	Intuitivo para usuarios avanzados	Intuitivo, excelente para principiantes
Tipos de empresas que lo usan	Empresas que necesitan velocidad y E/S asíncrona	Variado, desde startups hasta grandes corporaciones

Como vemos, FastAPI quizá está más orientado al desarrollo de infraestructuras más escalables con un mejor rendimiento, el buen manejo de operaciones de forma asíncrona supone otro plus para nuestro objetivo, ya que trabajaremos con bases de datos que requieren de este asincronismo. Es cierto que FastAPI es más nuevo, pero a pesar de lo reciente que es, la comunidad es grande y creciente. Todo esto lo convierte en una excelente alternativa para un desarrollo basado en API REST.

Par corroborar estas afirmaciones, si consultamos fuentes [28], vemos como efectivamente, tal y como vemos en la siguiente figura, fastAPI es uno de los frameworks mas eficientes, y dentro de los que están basado en Python, el más eficiente.

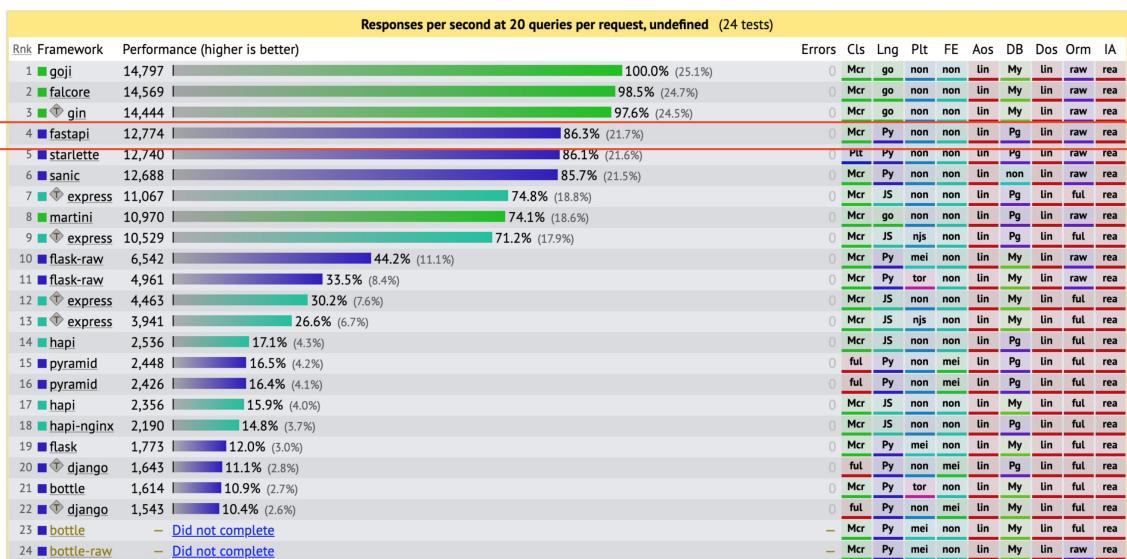


Figura 5.2: Comparativa de rendimiento entre distintos frameworks



Conclusión

De nuevo, la elección de un marco u otro condiciona el desarrollo del proyecto. A lo largo del texto se han ido dando pinceladas sobre las distintas alternativas, pero también como se ha descrito, Python, usando como framework FastAPI, es la alternativa que mejor se ajusta a nuestras necesidades y un marco de desarrollo con el que personalmente tengo experiencia, es por ello que, junto con los argumentos anteriormente expuestos, que la decisión final será FastAPI.

5.4 Base de datos - Relacional vs No Relacional

Elegir una base de datos para nuestro proyecto es una de las tareas más difíciles y a la vez condicionantes de nuestro proyecto y de cualquiera en general. Son diversos los factores que hay que tener en cuenta para elegir. ¿Es mejor base de datos relacional, o no relacional? Estas son preguntas bastante interesantes, puesto que un buen diseño de la base de datos es un factor de alta influencia con respecto a la calidad del proyecto, y el rendimiento y mantenibilidad a posteriori, no obstante, cabe decir que, con cualquiera de las dos elecciones se puede construir cualquier cosa, solamente hay que estudiar cuál de las dos alternativas se adapta más a las necesidades del proyecto.

A lo largo del grado, hemos hecho bastante hincapié en las bases de datos relacionales, hasta que, en las asignaturas más relacionadas con el desarrollo web conocimos un nuevo paradigma, las bases de datos no relacionales.

Las bases de datos pueden clasificarse en dos grandes categorías: relacionales y no relacionales, cada una con sus propias características y usos específicos. A continuación, se detallan sus principales diferencias:

5.4.1 Bases de Datos Relacionales

Las bases de datos relacionales están estructuradas en tablas con filas y columnas, siguiendo un esquema predefinido. Se basan en el modelo relacional, donde los datos están organizados en relaciones (tablas) que se vinculan mediante claves primarias y foráneas. Su lenguaje de consulta principal es SQL (Structured Query Language), el cual permite realizar operaciones como inserción, eliminación y modificación de datos con gran precisión.

Una característica clave de este tipo de bases de datos es que cumplen con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), garantizando la integridad y fiabilidad de los datos, lo que las hace ideales para aplicaciones críticas como sistemas bancarios, gestión empresarial y aplicaciones donde la precisión de los datos es fundamental. [29]

Sin embargo, su escalabilidad suele ser vertical, es decir, para mejorar el rendimiento es necesario aumentar la capacidad del hardware del servidor. [29]



5.4.2 Bases de Datos No Relacionales

Las bases de datos no relacionales, también conocidas como NoSQL, no siguen el esquema tabular tradicional. Pueden adoptar diferentes modelos, como documentos JSON, bases de datos clave-valor, grafos o modelos de columnas amplias.

A diferencia de las relacionales, estas bases de datos están diseñadas para manejar datos no estructurados o semi-estructurados, ofreciendo una mayor flexibilidad. No utilizan SQL como lenguaje principal, sino que emplean APIs específicas o lenguajes propios de consulta.

En cuanto a la consistencia, suelen seguir el modelo BASE (Basically Available, Soft state, Eventual consistency), lo que significa que priorizan la disponibilidad y la velocidad sobre la coherencia estricta de los datos. [29]

Su principal ventaja es la escalabilidad horizontal, lo que permite distribuir la carga en múltiples servidores y hacer frente a grandes volúmenes de datos. Por ello, son ampliamente utilizadas en aplicaciones como redes sociales, análisis de big data, sistemas de recomendación y plataformas de streaming. [29]

5.4.3 Conclusión

En nuestro proyecto, la coherencia es un factor clave, ya que manejaremos información sensible que además determinará en un futuro algo tan importante como es las entregas de las actividades de los alumnos y sus calificaciones. Es cierto que la rapidez también es una característica que se ha de priorizar, pero la coherencia de los datos, unos datos que además guardarán siempre una estructura, es un factor clave como hemos dicho anteriormente, mantener la integridad es algo que, con las bases de datos relacionales no es trivial pero están pensadas para hacerlo. Dicha estructura es otro de los factores a tener en cuenta, ya que el elegir si se hace relacional o no, condicionará decisiones de diseño para definir la estructura de los datos que mantendremos. Personalmente no considero que existan razones para usar bases de datos no relacionales a priori, por lo que la elección será usar bases de datos relacionales.

Capítulo 6

Recursos



En este capítulo se expondrán cuáles son los recursos que serán utilizados a lo largo del proyecto para su desarrollo, se dividirá en recursos humanos y recursos materiales, tanto hardware como software.

6.1 Recursos humanos

Los recursos humanos disponibles para este proyecto son:

Autor

- Nombre: Rafael García Pérez
- Titulación: Grado en Ingeniería Informática
- Mención: Ingeniería del Software
- Mail: p82garpr@uco.es

Director

- Rafael Muñoz Salinas

6.2 Recursos Hardware y Software

En cuanto a los recursos hardware, haremos uso de:

- HP Victus 16-e1007ns AMD Ryzen 7 6800H/16GB/512GB SSD/RTX 3050
- Ordenador de torre i7 12700KF / Radeon RX 7800 XT 16Gb / 32GB DDR4 3200Mhz / NVMe 1Tb
- Xiaomi Redmi Note 11 Pro 5G 6/128GB
- Xiaomi Redmi Note 8 Pro 6/64Gb
- Servidor de investigación de la Universidad de Córdoba con dos GPU TITAN 12+12GB VRAM.
- Servidor de investigación de la Universidad de Córdoba con GPU RTX 3090 24GB VRAM.

Y con respecto a los recursos software, dispondremos de:

- **Sistema Operativo:** Windows 11, Linux, Android
- **IDE:** Visual Studio Code
- **Control de versiones:** Git, GitHub.
- **Emuladores:** ADB Android SDK.
- **Editores de texto:** Latex con Overleaf.
- **Modelos de I.A. y OCR:** Ollama, ChatGPT, Google Gemini, Azure Vision

Capítulo 7

Especificación de requisitos



En la sección presente, se listarán con detalle los requisitos que componen la funcionalidad principal de nuestra aplicación, estos requisitos tendrán que ser cumplidos en su totalidad para considerar que se ha terminado el proyecto satisfactoriamente. Estos requisitos podemos dividirlos en dos grandes grupos: Requisitos funcionales y Requisitos no funcionales, adicionalmente añadiremos un tercer grupo al que llamaremos requisitos de usuario y requisitos de información.

Los requisitos funcionales se definen como aquellos que representan una funcionalidad en el sistema, son especificaciones detalladas que definen las acciones, comportamientos y funcionalidades que un sistema debe ser capaz de realizar para cumplir con su propósito previsto. En el contexto del software y los sistemas, describen lo que el sistema debe hacer, incluidas las tareas que debe realizar, cómo interactuará con los usuarios y cómo responderá a diversas entradas o eventos. Estos requisitos son fundamentales para garantizar que el software cumpla con las expectativas de las partes interesadas y satisfaga las necesidades comerciales previstas. [30]

Los requisitos no funcionales son aquellos que representan restricciones o propiedades del sistema. Los requisitos no funcionales se refieren a los atributos de calidad de un sistema que definen cómo funciona en lugar de qué hace. A diferencia de los requisitos funcionales, que especifican las acciones y tareas que debe realizar un sistema, los requisitos no funcionales se centran en las características generales y el comportamiento del sistema en diversas condiciones. Abordan aspectos como el rendimiento, la facilidad de uso, la fiabilidad y la escalabilidad, garantizando que el sistema cumpla con los estándares de calidad y proporcione una experiencia de usuario satisfactoria. [30]

Adicionalmente, los requisitos de Usuario. Los requisitos de usuario no son más que los requisitos del sistema descritos desde el punto de vista del cliente final. Principalmente tienen como objetivo ser de ayuda para la validación del proyecto, ya que hacen de contrato entre el cliente y el desarrollador para cumplir aquellas funcionalidades que dicho cliente desea que formen parte del producto final.

Por último en esta sección, hablaremos de los requisitos de información, que nos describirán qué información almacenaremos en nuestro sistema.



7.1 Requisitos funcionales

ID	Descripción
RF1	El sistema deberá diferenciar entre dos tipos de usuarios: Alumno y Profesor.
RF2	El sistema deberá permitir el registro a los Usuarios.
RF3	El sistema deberá permitir a los usuarios existentes iniciar sesión o cerrar sesión.
RF4	El sistema deberá permitir a los usuarios editar su perfil.
RF5	El sistema deberá permitir a los usuarios ver información general de las asignaturas.
RF6	El sistema deberá permitir a los usuarios contactar con otro por correo electrónico.
RF7	El sistema deberá permitir a los profesores crear asignaturas.
RF8	El sistema deberá permitir a los profesores editar asignaturas.
RF9	El sistema deberá permitir a los profesores borrar asignaturas.
RF10	El sistema deberá permitir a los profesores crear actividades.
RF11	El sistema deberá permitir a los profesores editar actividades.
RF12	El sistema deberá permitir a los profesores borrar actividades.
RF13	El sistema deberá permitir a los profesores ver las entregas de las actividades.
RF14	El sistema deberá permitir a los profesores exportar las notas de las entregas.
RF15	El sistema deberá permitir a los profesores recalificar las notas de las entregas.
RF16	El sistema deberá permitir a los profesores exportar todas las notas de la asignatura.
RF17	El sistema deberá permitir a los profesores ver los alumnos de la asignatura.
RF18	El sistema deberá permitir a los profesores expulsar un alumno de la asignatura.
RF19	El sistema deberá permitir a los profesores ver las estadísticas de la asignatura.
RF20	El sistema deberá permitir a los alumnos inscribirse en las asignaturas.
RF21	El sistema deberá permitir a los alumnos desinscribirse en las asignaturas.
RF22	El sistema deberá permitir a los alumnos ver las actividades de las asignaturas.
RF23	El sistema deberá permitir a los alumnos realizar entregas de las actividades en tiempo.
RF24	El sistema deberá permitir a un alumno ver los compañeros de esa asignatura.
RF25	El sistema deberá corregir una actividad en base a un enunciado y dar una calificación y una retroalimentación.
RF26	El sistema deberá reconocer el texto de una imagen para la entrega de una actividad.
RF27	El sistema deberá permitir a un usuario recuperar su contraseña.

Cuadro 7.1: Requisitos Funcionales del Sistema



CAPÍTULO 7. ESPECIFICACIÓN DE REQUISITOS

Para ordenarlos con mayor claridad, podemos establecer categorías de la siguiente forma:

ID	Gestión de Usuarios
RF1	El sistema deberá diferenciar entre dos tipos de usuarios: Alumno y Profesor.
RF2	El sistema deberá permitir el registro a los usuarios.
RF3	El sistema deberá permitir a los usuarios existentes iniciar sesión.
RF4	El sistema deberá permitir a los usuarios editar su perfil.
RF27	El sistema deberá permitir a un usuario recuperar su contraseña.

Cuadro 7.2: Requisitos Funcionales - Gestión de Usuarios

ID	Comunicación entre Usuarios
RF6	El sistema deberá permitir a los usuarios contactar con otro por correo electrónico.

Cuadro 7.3: Requisitos Funcionales - Comunicación entre Usuarios

ID	Gestión de Asignaturas
RF5	El sistema deberá permitir a los usuarios ver información general de las asignaturas.
RF7	El sistema deberá permitir a los profesores crear asignaturas.
RF8	El sistema deberá permitir a los profesores editar asignaturas.
RF9	El sistema deberá permitir a los profesores borrar asignaturas.
RF17	El sistema deberá permitir a los profesores ver los alumnos de la asignatura.
RF18	El sistema deberá permitir a los profesores expulsar un alumno de la asignatura.
RF19	El sistema deberá permitir a los profesores ver las estadísticas de la asignatura.
RF20	El sistema deberá permitir a los alumnos inscribirse en las asignaturas.
RF21	El sistema deberá permitir a los alumnos desinscribirse en las asignaturas.
RF24	El sistema deberá permitir a un alumno ver los compañeros de esa asignatura.

Cuadro 7.4: Requisitos Funcionales - Gestión de Asignaturas



CAPÍTULO 7. ESPECIFICACIÓN DE REQUISITOS

ID	Gestión de Actividades y Entregas
RF10	El sistema deberá permitir a los profesores crear actividades.
RF11	El sistema deberá permitir a los profesores editar actividades.
RF12	El sistema deberá permitir a los profesores borrar actividades.
RF13	El sistema deberá permitir a los profesores ver las entregas de las actividades.
RF22	El sistema deberá permitir a los alumnos ver las actividades de las asignaturas.
RF23	El sistema deberá permitir a los alumnos realizar entregas de las actividades en tiempo.

Cuadro 7.5: Requisitos Funcionales - Gestión de Actividades y Entregas

ID	Evaluación y Notas
RF14	El sistema deberá permitir a los profesores exportar las notas de las entregas.
RF15	El sistema deberá permitir a los profesores recalificar las notas de las entregas.
RF16	El sistema deberá permitir a los profesores exportar todas las notas de la asignatura.

Cuadro 7.6: Requisitos Funcionales - Evaluación y Notas

ID	Funcionalidades Avanzadas
RF25	El sistema deberá corregir una actividad en base a un enunciado y dar una calificación y una retroalimentación.
RF26	El sistema deberá reconocer el texto de una imagen para la entrega de una actividad.

Cuadro 7.7: Requisitos Funcionales - Funcionalidades Avanzadas



7.2 Requisitos no funcionales

Los requisitos no funcionales de la aplicación quedarán descritos en la siguiente tabla:

ID	Descripción
RNF1	El sistema deberá soportar al menos 100 usuarios concurrentes sin degradación significativa del rendimiento.
RNF2	Las respuestas del sistema deberán tener un tiempo de respuesta inferior a 5 segundos en el caso de no tener la I.A. involucrada.
RNF3	La interfaz de usuario deberá ser responsive y adaptarse a distintos dispositivos (PC, tablet y móvil).
RNF4	Toda la comunicación entre el cliente y el servidor deberá estar cifrada mediante HTTPS y TLS 1.2 o superior.
RNF5	El sistema deberá almacenar las contraseñas de los usuarios usando un algoritmo de hash seguro.
RNF6	El sistema deberá realizar copias de seguridad automáticas semanales y almacenarlas por un período mínimo de 30 días.
RNF7	El código fuente deberá seguir estándares de calidad y buenas prácticas, incluyendo revisiones de código (code review).
RNF8	La aplicación deberá contar con pruebas automatizadas para al menos el 70 % del código funcional del servidor.
RNF9	La interfaz deberá proporcionar retroalimentación visual inmediata tras la interacción del usuario.
RNF10	El sistema deberá mostrar mensajes de error claros y comprensibles para los usuarios finales.
RNF11	La documentación técnica y de usuario deberá estar disponible y mantenerse actualizada con cada versión del sistema.

Cuadro 7.8: Requisitos No Funcionales

7.3 Requisitos de Usuario

Los requisitos de usuario, son un compendio de los anteriores pero desde el punto de vista del cliente final como se mencionó anteriormente. Quedarán descritos mediante la siguiente tabla:



CAPÍTULO 7. ESPECIFICACIÓN DE REQUISITOS

ID	Requisito de Usuario
RU-1	Como usuario, quiero poder registrarme en el sistema para acceder a la plataforma.
RU-2	Como usuario, quiero poder iniciar sesión con mis credenciales para acceder a mi perfil.
RU-3	Como usuario, quiero editar mi perfil para actualizar mi información personal.
RU-4	Como alumno, quiero inscribirme en asignaturas para acceder al contenido .
RU-5	Como alumno, quiero ver la información general de mis asignaturas para entender sus objetivos y temario.
RU-6	Como alumno, quiero ver las actividades disponibles en cada asignatura para poder realizarlas.
RU-7	Como alumno, quiero entregar mis actividades en el sistema para que sean evaluadas.
RU-8	Como alumno, quiero ver mis calificaciones para conocer mi progreso.
RU-9	Como alumno, quiero recibir retroalimentación de mis actividades para mejorar en futuras entregas.
RU-10	Como alumno, quiero poder contactar a mis profesores por correo electrónico en caso de dudas.
RU-11	Como profesor, quiero crear asignaturas para gestionar mis clases.
RU-12	Como profesor, quiero editar la información de mis asignaturas para mantenerla actualizada.
RU-13	Como profesor, quiero eliminar asignaturas que ya no sean necesarias en el sistema.
RU-14	Como profesor, quiero crear actividades dentro de una asignatura para que los alumnos las realicen.
RU-15	Como profesor, quiero editar y actualizar las actividades para ajustarlas a las necesidades del curso.
RU-16	Como profesor, quiero eliminar actividades que ya no sean necesarias en la asignatura.
RU-17	Como profesor, quiero ver las entregas de mis alumnos para conocer su calificación.
RU-18	Como profesor, quiero exportar las calificaciones de mis alumnos para almacenarlas o compartirlas.
RU-19	Como profesor, quiero recalificar actividades en caso de errores o ajustes en la evaluación.
RU-20	Como profesor, quiero ver estadísticas de mis asignaturas para analizar el rendimiento de los alumnos.
RU-21	Como alumno, quiero que el sistema corrija automáticamente mi entrega en base al enunciado de la actividad.
RU-22	Como usuario, quiero que el sistema reconozca el texto de una imagen para facilitar la entrega de actividades.

Cuadro 7.9: Requisitos de Usuario



7.4 Requisitos de Información

Los requisitos de información especifican los datos que el sistema debe almacenar, gestionar y proporcionar. Estos requisitos son cruciales para asegurar que el sistema maneje la información de manera eficiente y precisa, permitiendo a los usuarios realizar sus tareas de manera efectiva. A continuación, se detallan los requisitos de información para nuestra aplicación:

ID	Descripción
RI1	El sistema deberá almacenar y gestionar la información de los usuarios (nombre, email, contraseña, etc.).
RI2	El sistema deberá almacenar y gestionar la información de las asignaturas (nombre, descripción, código de acceso, etc.).
RI3	El sistema deberá almacenar y gestionar la información de las actividades (título, descripción, fechas, etc.).
RI4	El sistema deberá almacenar y gestionar la información de las entregas (imagen, comentarios, calificación, etc.).
RI5	El sistema deberá almacenar y gestionar las inscripciones de los alumnos en las asignaturas.
RI6	El sistema deberá mostrar estadísticas de las asignaturas y actividades.

Cuadro 7.10: Requisitos de Información del Sistema

Capítulo 8

Especificación del sistema



La especificación del sistema está compuesta por la fase de análisis y diseño, e implementación en la sección de análisis descubriremos qué información es relevante, los usuarios que usarán nuestro sistema, la descripción de los datos mediante el modelo entidad relación, y los casos de uso, los cuales son muy importantes en esta fase de análisis, sobre ellos además realizaremos diagramas de actividad para mejorar la legibilidad y comprensión de la funcionalidad de nuestro sistema. Por último hablaremos de algunos detalles de implementación, cómo hemos materializado el proyecto en base a las fases anteriores.

8.1 Análisis

Entendemos la fase de análisis como aquella en la que se identifican y definen los requisitos que debe cumplir el sistema para satisfacer las necesidades de los usuarios. Esta etapa es fundamental para comprender el problema que se desea resolver y establecer una base sólida para el diseño posterior. Durante el análisis se recopila información relevante del dominio de la aplicación, se clasifican los tipos de usuarios, se definen los procesos clave del sistema y se modelan los datos que formarán parte del mismo. Todo esto con el objetivo de construir una solución coherente, completa y alineada con los objetivos del proyecto. En este punto, debido a que tenemos una definición previa de los requisitos, no se introducirán.

8.1.1 Usuarios del sistema

La aplicación será destinada tanto a estudiantes universitarios realizar las entregas para la evaluación como a los docentes que imparten las asignaturas para gestionarlas.

8.1.2 Descripción de los datos - Modelo entidad relación

Como dijimos anteriormente, dadas las exigencias del proyecto, una decisión estratégica es tomar como base de datos una base de datos relacional, que nos permita organizar los datos y establecer relaciones y reglas de manera estricta. Esta descripción de los datos la realizaremos mediante la confección de un diagrama entidad - relación, que representará la estructura de los datos de manera conceptual mediante las entidades, relaciones, cardinalidades y atributos.

Entidades

Basándonos en los requisitos, podemos distinguir las siguientes entidades principalmente:

- Usuario.
- Asignatura.
- Actividad.
- Entrega.



Usuario

La entidad usuario es una de las más importantes, ya que representa al cliente final que interactúa con la aplicación a través de las relaciones que tenga con el resto de entidades. Además, tendremos una especificación total exclusiva sin solapamiento, ya que, en nuestra aplicación un usuario no podrá ser profesor y alumno al mismo tiempo, y no existirán otras especificaciones posibles además de estas.

Los atributos que definen al usuario son:

- id
- email
- contraseña
- nombre
- apellidos
- tipo_usuario

Mediante la siguiente imagen vemos la entidad representada en nuestro diagrama entidad-relación:

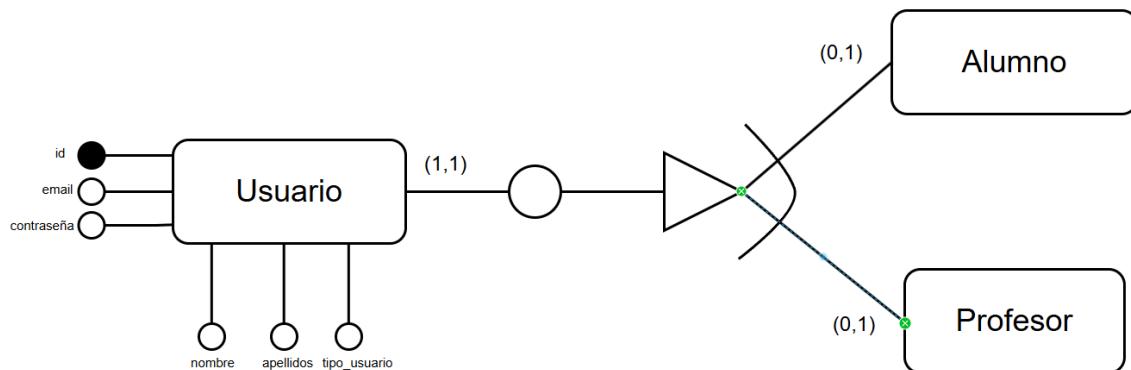


Figura 8.1: Entidad Usuario

Asignatura

La entidad asignatura representada en el sistema es una de las entidades principales también, ya que de la misma dependerán tanto las actividades como las entregas, además de los alumnos que la cursan y el profesor que la administra. Cabe destacar la existencia del campo tipo_usuario, que nos indica el rol del usuario, profesor, o alumno.

Los atributos que definen a la asignatura son:

- id
- nombre



- descripción
- codigo_acceso

Mediante la siguiente imagen vemos la entidad representada en nuestro diagrama entidad-relación:

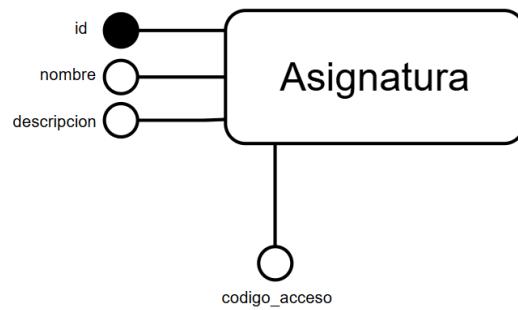


Figura 8.2: Entidad Asignatura

Actividad

Dicha entidad representa una actividad en nuestro proyecto, la actividad es creada por un profesor en una asignatura con unos requisitos sobre la que luego el alumno realizará una entrega.

Los atributos que definen a la asignatura son:

- id
- titulo
- descripcion
- fecha_creacion
- fecha_entrega
- lenguaje_programacion
- parametros_evaluacion

Mediante la siguiente imagen vemos la entidad representada en nuestro diagrama entidad-relación:

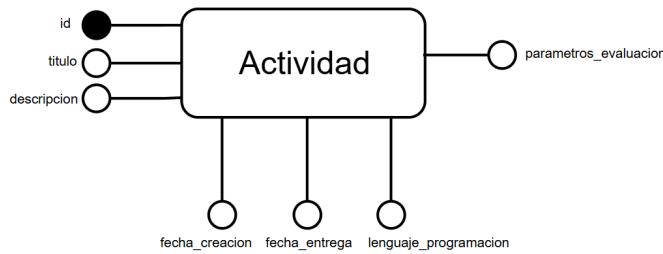


Figura 8.3: Entidad Actividad

Entrega

Dicha entidad representa una entrega que un alumno realiza sobre una actividad de una asignatura, dicha entrega definirá también la evaluación que se realiza sobre la solución que aporta un alumno.

Los atributos que definen a la asignatura son:

- id
- imagen
- tipo_imagen
- nombre_archivo
- comentarios
- calificacion
- fecha_entrega
- texto_ocr

Mediante la siguiente imagen vemos la entidad representada en nuestro diagrama entidad-relación:

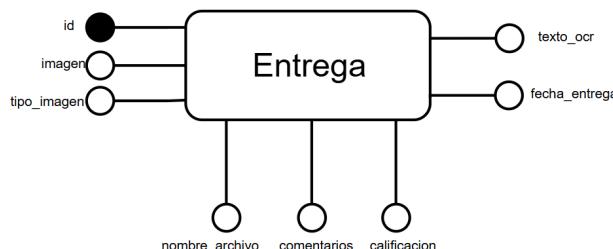


Figura 8.4: Entidad Entrega



Diagrama entidad relación

Una vez tenemos las entidades con sus atributos identificados, podemos construir un diagrama entidad relación que represente, como su nombre indica, las relaciones que existen entre las distintas entidades así como las cardinalidades. Para simplificar el diagrama, los atributos no se encuentran ligados a las entidades, ya que fueron descritos anteriormente, no obstante, a posteriori se hará una tabla para una mayor claridad. El diagrama queda definido en la siguiente figura.

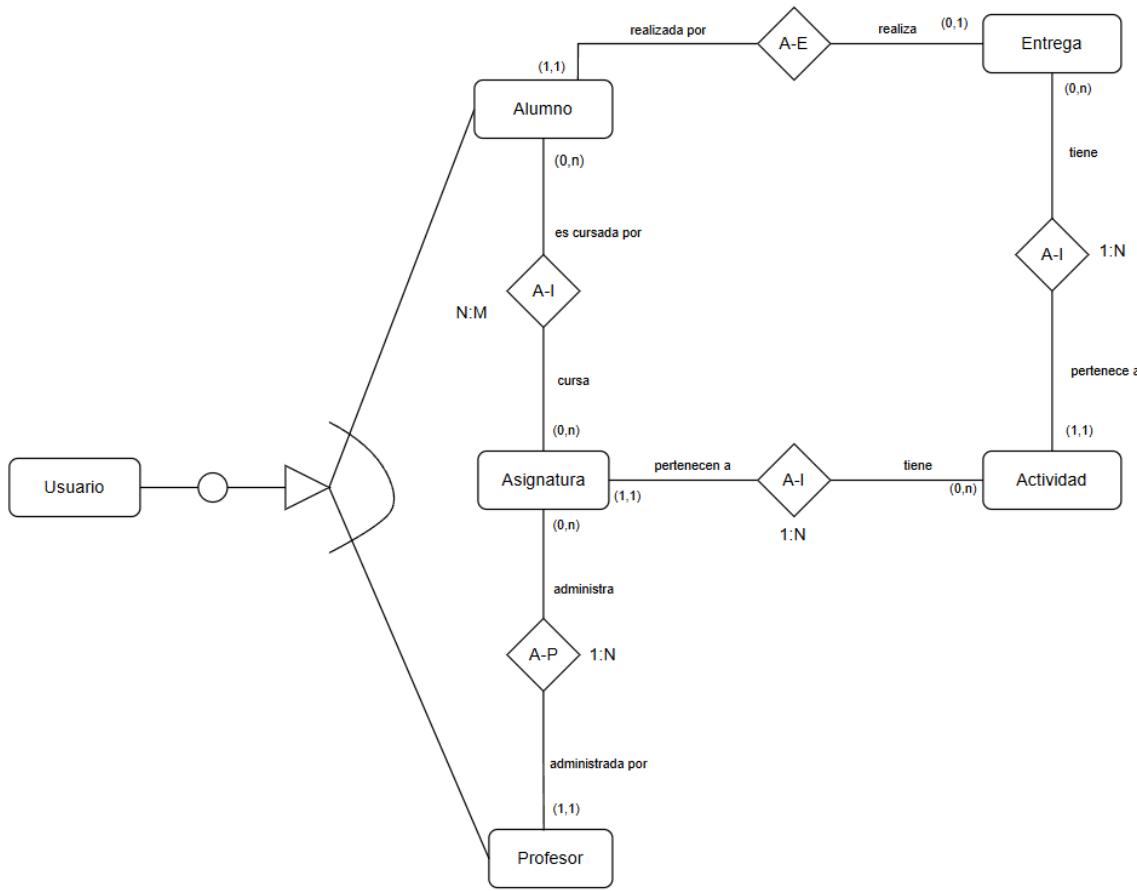


Figura 8.5: Diagrama Entidad - Relación

Tal y como vemos en la figura, hay una relación con cardinalidad múltiple N:M, la cual sería conveniente eliminar de cara a una futura normalización e implementación. Dicha relación es la existente entre Alumno y Asignatura, ya que, un alumno puede estar inscrito en varias asignatura simultáneamente, y una asignatura puede tener varios alumnos, por lo que las cardinalidades máximas son muchos a muchos. Para ello, crearemos una nueva entidad, la entidad Inscripción, que quedará definida por los siguientes atributos:

- id
- fecha_inscripcion

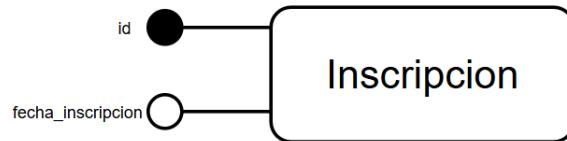


Figura 8.6: Entidad Inscripción

Mediante la siguiente imagen vemos la entidad representada en nuestro diagrama entidad-relación:

De esta forma, el diagrama entidad - relación se nos quedaría de la siguiente manera:

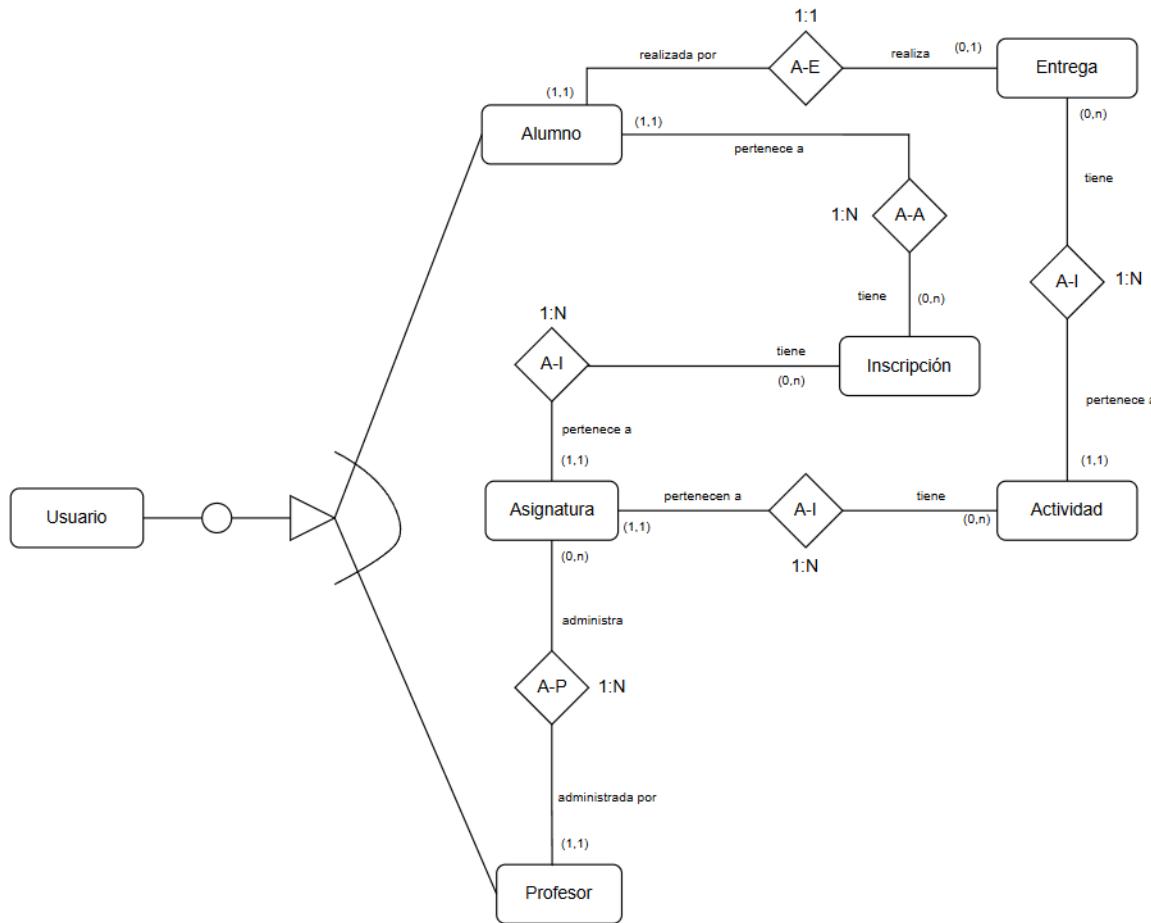


Figura 8.7: Diagrama Entidad - Relación

En este momento, ya que disponemos del diagrama entidad relación al completo, podemos pasar a explicarlo semánticamente para darle sentido y motivo a las relaciones del mismo.

En primer lugar, como explicamos como anterioridad, hablaremos de la especificación de la entidad Usuario, el motivo es simple, Alumnos y Profesores tendrán distintas relaciones con distintas entidades, por lo que diferenciarlos es algo obligatorio. Profesor tiene



una relación con asignatura 1:N, ya que una asignatura, por decisión de los requisitos, solamente podrá ser impartida, o, administrada en el dominio de la aplicación, por un profesor, de la misma manera que un profesor puede tener varias asignaturas.

Asignatura, tendrá además relación con actividad e inscripción. Una asignatura puede tener varias inscripciones de varios alumnos, pero dicha inscripción solamente pertenecerá a una asignatura, de ahí dicha relación 1:N, a su vez, una asignatura puede tener varias actividades y una actividad pertenecerá a únicamente una asignatura, por lo que tenemos otra relación 1:N.

Una inscripción, además de estar relacionada con la asignatura, lo estará con Alumno, ya que la inscripción le pertenece a este, y como ya dijimos es una entidad insertada para eliminar la N:M. De esta forma, un alumno podrá tener varias inscripciones y una inscripción pertenece a un único alumno.

Alumno tiene una relación 1:1 con entrega, ya que esta representa la solución que este aporta a una actividad, así como, a posteriori, la calificación y los comentarios que este obtiene al respecto.

Normalización

La normalización es un proceso fundamental en el diseño de bases de datos relacionales. Su objetivo principal es organizar los datos de manera eficiente para minimizar la redundancia y evitar anomalías que puedan surgir durante la inserción, actualización y eliminación de datos. Al aplicar las formas normales, garantizamos que nuestra base de datos sea coherente, fácil de mantener y eficiente en el almacenamiento y recuperación de información.

En las siguientes subsecciones, analizaremos nuestro modelo entidad-relación y aplicaremos las formas normales necesarias para asegurar que la base de datos esté bien estructurada y cumpla con los principios de integridad y eficiencia. Comenzaremos con la Primera Forma Normal (1FN), asegurando que todos los atributos sean atómicos y que no haya grupos repetidos de atributos.

Primera Forma Normal (1FN)

La Primera Forma Normal (1FN) requiere que todos los atributos sean atómicos y que no haya grupos repetidos de atributos. Nuestro modelo inicial ya cumple con la 1FN.

Cuadro 8.1: Entidades en 1FN

Entidad	Atributos	Clave Prima-ria
Usuario	id, email, contraseña, nombre, apellidos, tipo_usuario	id
Asignatura	id, nombre, descripción, código_acceso	id
Actividad	id, título, descripción, fecha_creacion, fecha_entrega, lenguaje_programacion, parámetros_evaluacion	id

Continúa en la siguiente página



Cuadro 8.1: Entidades en 1FN

Entidad	Atributos	Clave Primaria
Entrega	id, imagen, tipo_imagen, nombre_archivo, comentarios, calificacion, fecha_entrega, texto_ocr	id
Inscripción	id, fecha_inscripcion	id

Segunda Forma Normal (2FN)

La Segunda Forma Normal (2FN) requiere que el modelo cumpla con la 1FN y que todos los atributos no clave dependan completamente de la clave primaria. En la tabla Inscripción, necesitamos agregar claves foráneas para cumplir con la 2FN.

Cuadro 8.2: Entidades en 2FN

Entidad	Atributos	Clave Primaria
Usuario	id, email, contraseña, nombre, apellidos, tipo_usuario	id
Asignatura	id, nombre, descripción, codigo_acceso	id
Actividad	id, titulo, descripcion, fecha_creacion, fecha_entrega, lenguaje_programacion, parametros_evaluacion	id
Entrega	id, imagen, tipo_imagen, nombre_archivo, comentarios, calificacion, fecha_entrega, texto_ocr	id
Inscripción	id, alumno_id (FK), asignatura_id (FK), fecha_inscripcion	id

Tercera Forma Normal (3FN)

La Tercera Forma Normal (3FN) requiere que el modelo cumpla con la 2FN y que ningún atributo no clave dependa de otro atributo no clave. En nuestro caso, revisamos las dependencias transitivas y ajustamos la tabla Entrega si es necesario.

Cuadro 8.3: Entidades en 3FN

Entidad	Atributos	Clave Primaria
Usuario	id, email, contraseña, nombre, apellidos, tipo_usuario	id
Asignatura	id, nombre, descripción, codigo_acceso	id

Continúa en la siguiente página



Cuadro 8.3: Entidades en 3FN

Entidad	Atributos	Clave Primaria
Actividad	id, titulo, descripcion, fecha_creacion, fecha_entrega, lenguaje_programacion, parametros_evaluacion	id
Entrega	id, imagen, tipo_imagen, nombre_archivo, comentarios, calificacion, fecha_entrega, texto_ocr	id
Inscripción	id, alumno_id (FK), asignatura_id (FK), fecha_incripcion	id



8.1.3 Descripción funcional - Casos de uso

Para la descripción funcional del sistema, usaremos casos de uso mediante el lenguaje UML. Según IBM, [31], los casos de uso se crean para refinar un conjunto de requisitos basados en un rol o tarea. Agrupan requisitos comunes basados en el tipo de rol o objetivo. Definen lo que los usuarios o roles están haciendo en la solución, un proceso de negocio que define cómo realizan estas funciones. Quedarán definidos en la siguiente tabla:



ID	Nombre del caso de uso
CU-1	Registro
CU-2	IniciarSesion
CU-3	CerrarSesion
CU-4	RecuperarContraseña
CU-5	VerInformacionAsignatura
CU-6	ExpulsarAlumnoAsignatura
CU-7	EditarAsignatura
CU-8	CrearAsignatura
CU-9	BorrarAsignatura
CU-10	ExportarCalificacionesAsignatura
CU-11	EnviarCorreoUsuario
CU-12	VerAlumnosAsignatura
CU-13	VerPerfilUsuario
CU-14	EditarPerfil
CU-15	InscribirAsignatura
CU-16	VerMiProgreso
CU-17	VerEstadisticasAsignatura
CU-18	VerActividadesAsignatura
CU-19	CrearActividad
CU-20	BorrarActividad
CU-21	EditarActividad
CU-22	VerActividad
CU-23	VerEntregasActividad
CU-24	CalificarEntregaManualmente
CU-25	ExportarEntregas
CU-26	VerEntrega
CU-27	HacerEntrega
CU-28	EscanearOCR
CU-29	EvaluarSolucion
CU-30	SalirAsignatura

Cuadro 8.4: Historias de Usuario

En primer lugar, haremos un diagrama de casos de uso, en el cual englobaremos todos para darnos una perspectiva más general del sistema, no obstante, dada la numerosa cantidad de casos de uso que tenemos, para mejorar la legibilidad he decidido dividirlos en los siguientes subgrupos que llamaremos subsistemas:

1. Subsistema administración del usuario.



2. Subsistema asignaturas.
3. Subsistema actividades.
4. Subsistema entregas.

Por tanto, antes de comenzar con el diagrama de casos de uso, determinaremos qué casos de uso pertenecen a cada subsistema de los indicados.

Subsistema administración del usuario

Son aquellos casos de uso relacionados con la administración de los usuarios así como la autenticación. En la siguiente tabla se muestran los casos de uso:

ID	Nombre del caso de uso
CU-1	Registro
CU-2	IniciarSesion
CU-3	CerrarSesion
CU-4	RecuperarContraseña
CU-5	VerPerfilUsuario
CU-6	EditarPerfil

Cuadro 8.5: Casos de Uso de Administración del Usuario

De esta forma, mediante el siguiente diagrama de casos de uso podemos observar la relación que estos tienen:

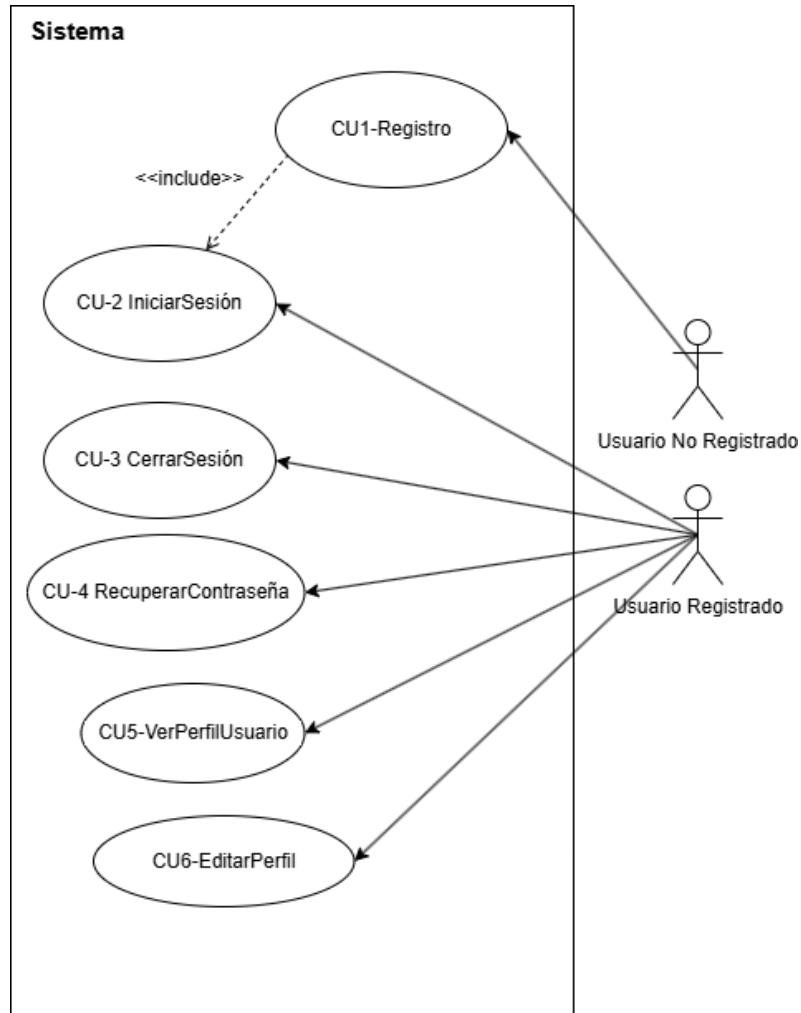


Figura 8.8: Casos de uso administración del usuario



Subsistema asignaturas

Son aquellos casos de uso relacionados con las asignaturas, su administración así como otras funcionalidades relacionadas con las mismas. En la siguiente tabla se muestran los casos de uso:

ID	Nombre del caso de uso
CU-7	VerInformacionAsignatura
CU-8	ExpulsarAlumnoAsignatura
CU-9	EditarAsignatura
CU-10	CrearAsignatura
CU-11	BorrarAsignatura
CU-12	ExportarCalificacionesAsignatura
CU-13	EnviarCorreoUsuario
CU-14	VerAlumnosAsignatura
CU-15	InscribirAsignatura
CU-16	VerMiProgreso
CU-17	VerEstadisticasAsignatura
CU-30	SalirAsignatura

Cuadro 8.6: Casos de Uso de Asignaturas

De esta forma, mediante el siguiente diagrama de casos de uso podemos observar la relación que estos tienen:

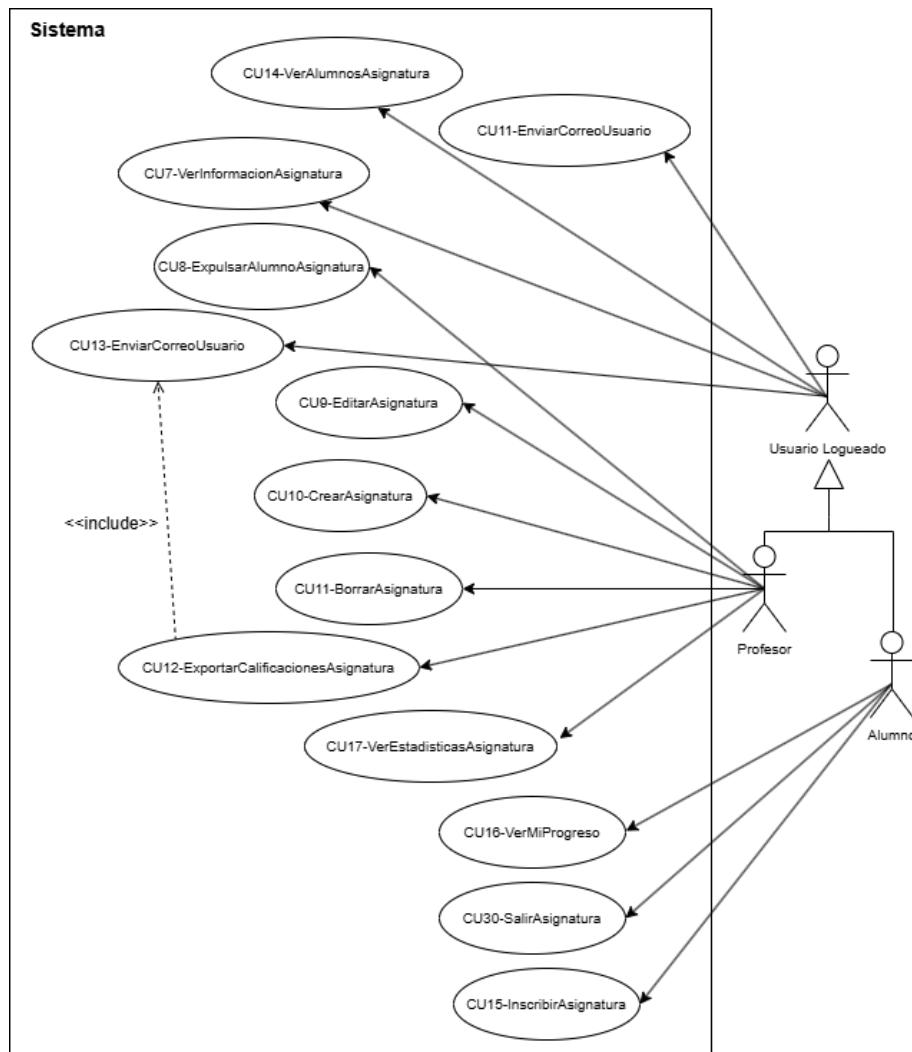


Figura 8.9: Casos de uso asignaturas

Subsistema actividades

Son aquellos casos de uso relacionados con la gestión de las actividades de las asignaturas. En la siguiente tabla se muestran los casos de uso:

ID	Nombre del caso de uso
CU-18	VerActividadesAsignatura
CU-19	CrearActividad
CU-20	BorrarActividad
CU-21	EditarActividad
CU-22	VerActividad

Cuadro 8.7: Casos de Uso de Actividades

De esta forma, mediante el siguiente diagrama de casos de uso podemos observar la relación que estos tienen:

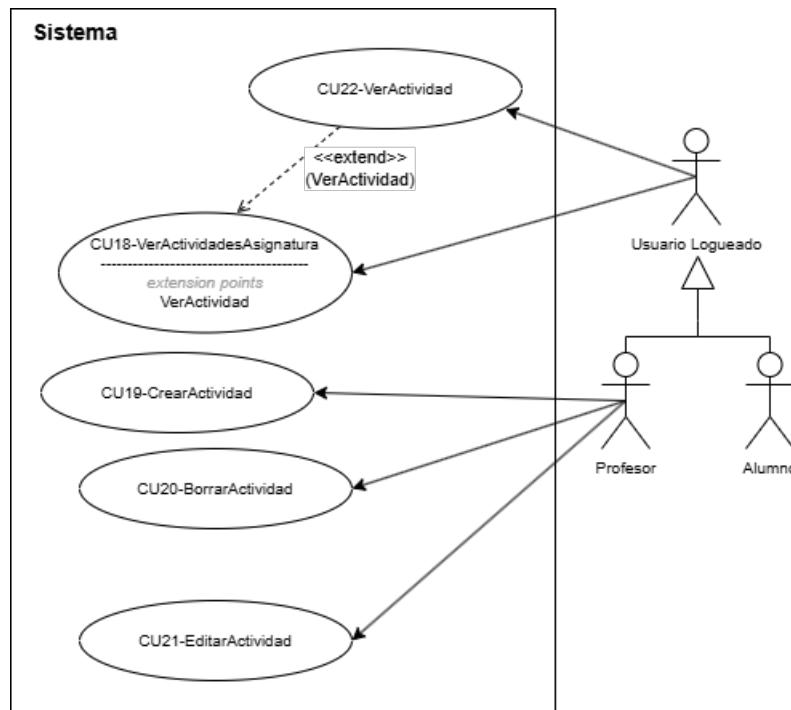


Figura 8.10: Casos de uso actividades

Subsistema entregas

Son aquellos casos de uso relacionados con la gestión de las entregas de las actividades. En la siguiente tabla se muestran los casos de uso:

ID	Nombre del caso de uso
CU-23	VerEntregasActividad
CU-24	CalificarEntregaManualmente
CU-25	ExportarEntregas
CU-26	VerEntrega
CU-27	HacerEntrega
CU-28	EscanearOCR
CU-29	EvaluarSolucion

Cuadro 8.8: Casos de Uso de Entregas

De esta forma, mediante el siguiente diagrama de casos de uso podemos observar la relación que estos tienen:

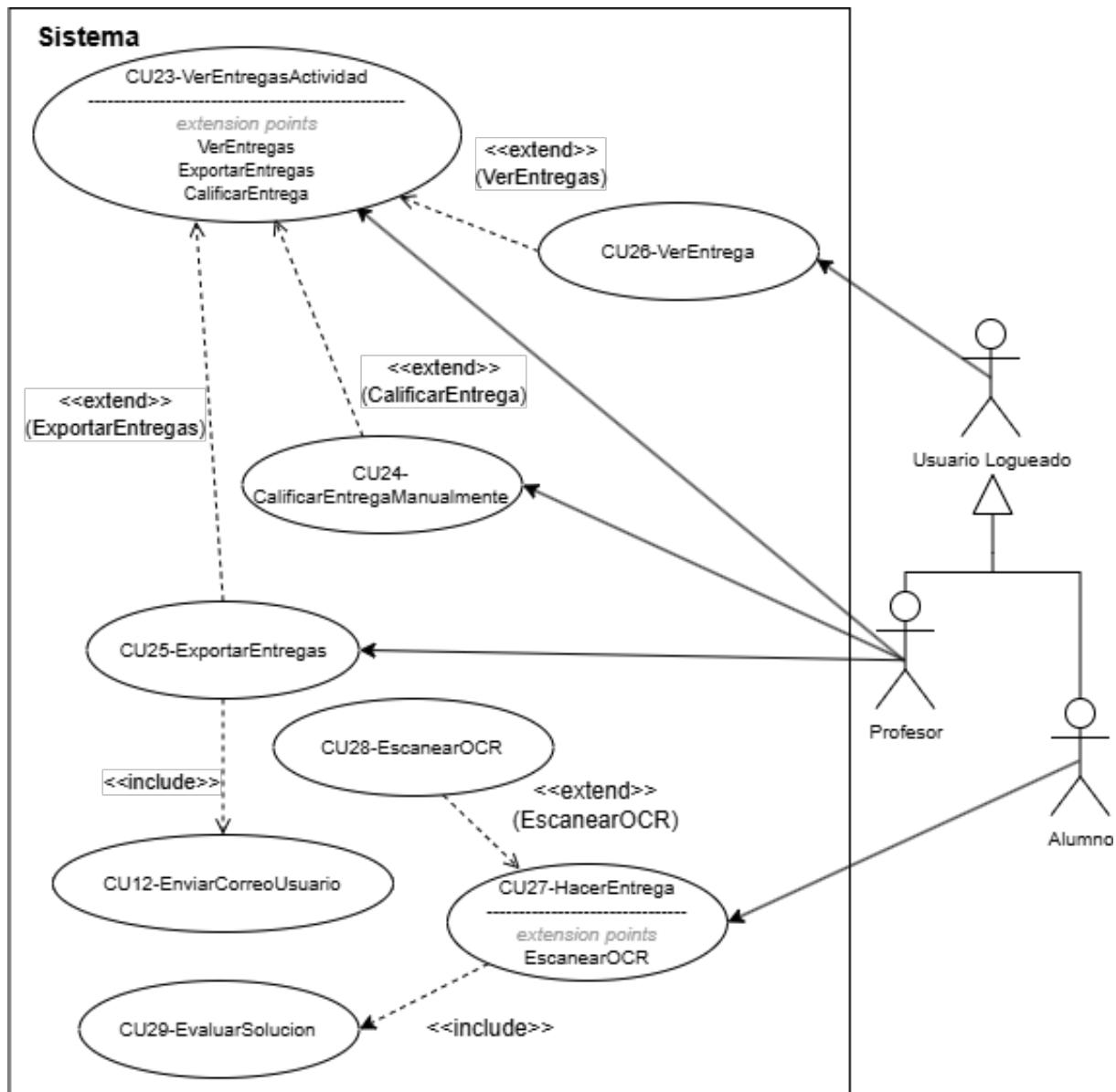


Figura 8.11: Casos de uso entregas



8.1.4 Especificación de casos de uso

La especificación de casos de uso consiste en describir cómo un usuario interactúa con el sistema, teniendo en cuenta su rol, para completar una funcionalidad del mismo. Para ello, usaremos una plantilla que quedará definida por la siguiente tabla:

Campo	Descripción
Identificador	Código único del caso de uso, por ejemplo, CU1, CU2, etc. Sirve para identificar rápidamente cada funcionalidad dentro del sistema.
Nombre	Nombre del caso de uso, que indica de manera breve la funcionalidad principal que representa.
Descripción	Breve explicación sobre la funcionalidad específica del caso de uso. Aquí se detalla qué acción realizará el usuario o el sistema.
Actor principal	Usuario o rol que inicia y ejecuta el caso de uso. Puede ser un alumno, un profesor, un administrador u otro rol definido en el sistema.
Actor secundario	Otro usuario o sistema que interactúa en el caso de uso de forma indirecta. Puede ser opcional y solo aplica en casos en los que haya intervención externa.
Precondiciones	Condiciones que deben cumplirse antes de que el caso de uso pueda ejecutarse. Incluyen requisitos como autenticación, existencia de datos previos, etc.
Flujo principal	Secuencia de pasos ordenados que se siguen para ejecutar el caso de uso de manera correcta. Describe la interacción entre el usuario y el sistema para alcanzar el objetivo.
Postcondiciones	Estado del sistema después de completar la ejecución del caso de uso. Puede incluir datos almacenados, cambios en la interfaz o modificaciones en la base de datos.
Flujo alternativo	Variaciones o excepciones que pueden ocurrir en el flujo principal. Aquí se describen errores, caminos alternativos y respuestas del sistema ante situaciones inusuales.

Cuadro 8.9: Plantilla de Especificación de Casos de Uso

De esta forma, comenzaremos a desarrollar las distintas tablas que describen la especificación de los casos de uso.



CU1 - Registro

Identificador	CU1
Nombre	Registro
Descripción	El usuario se registra en el sistema.
Actor principal	Usuario No Registrado
Actor secundario	-
Precondiciones	1. El usuario no debe tener una cuenta registrada en el sistema con su correo electrónico.
Flujo principal	1. El caso de uso comienza cuando un usuario no registrado selecciona <i>registrarse o crear cuenta</i> . 2. El usuario rellena el formulario con los datos que se requieren. 3. El sistema comprueba que todos los campos hayan sido llenados. 4. El sistema comprueba la existencia del usuario en la base de datos. 5. El sistema registra al usuario en la base de datos. 6. include (IniciarSesion)
Postcondiciones	1. El usuario queda registrado en la aplicación. 2. El usuario inicia sesión automáticamente.
Flujo alternativo	3. El sistema comprueba que todos los campos hayan sido llenados. 3.1. Si el usuario no ha llenado algún campo obligatorio, el sistema lo notifica pidiendo al usuario llenar los campos pendientes. 4. El sistema comprueba la existencia del usuario en la base de datos. 4.1. Si el usuario ya existe con ese correo electrónico, se notifica al usuario y no permite el registro.

Cuadro 8.10: Especificación del Caso de Uso: Registro



CU2 - IniciarSesión

Identificador	CU2
Nombre	IniciarSesión
Descripción	El usuario inicia sesión para entrar en el sistema.
Actor principal	Usuario Registrado
Actor secundario	-
Precondiciones	1. El usuario debe tener una cuenta registrada en el sistema con su correo electrónico.
Flujo principal	1. El caso de uso comienza cuando un usuario registrado selecciona <i>iniciar sesión</i> . 2. El usuario rellena el formulario con los datos que se requieren. 3. El sistema comprueba que todos los campos hayan sido llenados. 4. El sistema comprueba la existencia del usuario en la base de datos. 5. El sistema comprueba que la contraseña sea correcta. 6. El sistema concede el acceso a la aplicación.
Postcondiciones	1. El usuario inicia sesión en la aplicación.
Flujo alternativo	3. El sistema comprueba que todos los campos hayan sido llenados. 3.1. Si el usuario no ha llenado algún campo obligatorio, el sistema lo notifica pidiendo al usuario llenar los campos pendientes. 4. El sistema comprueba la existencia del usuario en la base de datos. 4.1. Si el usuario no existe con ese correo electrónico, se notifica al usuario y no permite el inicio de sesión. 5. El sistema comprueba que la contraseña sea correcta. 5.1. Si la contraseña no es correcta, se notifica al usuario e impide entrar al sistema.

Cuadro 8.11: Especificación del Caso de Uso: Iniciar Sesión

CU3 - CerrarSesión

Identificador	CU3
Nombre	CerrarSesión
Descripción	El usuario cierra sesión en el sistema.
Actor principal	Usuario Registrado
Actor secundario	-
Precondiciones	1. El usuario debe tener la sesión iniciada en el sistema.
Flujo principal	1. El caso de uso comienza cuando un usuario registrado selecciona <i>cerrar sesión</i> . 2. El sistema elimina la sesión del sistema.
Postcondiciones	1. El usuario cierra la sesión del sistema.
Flujo alternativo	-

Cuadro 8.12: Especificación del Caso de Uso: Cerrar Sesión



CU4 - RecuperarContraseña

Identificador	CU4
Nombre	RecuperarContraseña
Descripción	El usuario recupera la contraseña para poder entrar en el sistema en caso de olvido.
Actor principal	Usuario Registrado
Actor secundario	-
Precondiciones	-
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando un usuario selecciona <i>Recuperar contraseña</i>.2. El usuario ingresa su correo electrónico registrado.3. El sistema verifica la existencia del correo en la base de datos.4. El sistema envía un token de recuperación al correo electrónico del usuario.5. El usuario introduce el token en el sistema.6. El sistema comprueba la validez del token.7. El usuario establece su nueva contraseña.
Postcondiciones	<ol style="list-style-type: none">1. El usuario ha restablecido su contraseña y puede iniciar sesión con la nueva.
Flujo alternativo	<ol style="list-style-type: none">3. El sistema verifica la existencia del correo en la base de datos.<ol style="list-style-type: none">3.1. Si el correo no está registrado, el sistema notifica al usuario y no envía el enlace de recuperación.6. El sistema comprueba la validez del token.<ol style="list-style-type: none">6.1. Si el token no es válido, se notifica y se termina el caso de uso.

Cuadro 8.13: Especificación del Caso de Uso: Recuperar Contraseña

CU5 - VerInformaciónAsignatura

Identificador	CU5
Nombre	VerInformaciónAsignatura
Descripción	El usuario visualiza la información general del sistema.
Actor principal	Usuario Logueado
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando un usuario logueado selecciona una asignatura.2. El sistema comprueba que el usuario tenga permisos.3. El sistema muestra la información de la asignatura.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario tenga permisos.<ol style="list-style-type: none">2.1. Si el usuario no tiene permisos, se notifica y se cancela el caso de uso

Cuadro 8.14: Especificación del Caso de Uso: Ver Información Asignatura



CU6 - ExpulsarAlumnoAsignatura

Identificador	CU6
Nombre	ExpulsarAlumnoAsignatura
Descripción	El profesor expulsa un alumno de la asignatura eliminando su inscripción.
Actor principal	Profesor
Actor secundario	Alumno
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol de profesor.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Expulsar alumno</i>.2. El sistema comprueba que el alumno esté matriculado en la asignatura correctamente.3. El sistema elimina la inscripción del alumno.
Postcondiciones	<ol style="list-style-type: none">1. El alumno expulsado no tendrá acceso de nuevo a la asignatura.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el alumno esté matriculado en la asignatura correctamente.<ol style="list-style-type: none">2.1. Si el alumno no está matriculado, el sistema no elimina la inscripción y lo notifica.

Cuadro 8.15: Especificación del Caso de Uso: Expulsar Alumno Asignatura

CU7 - EditarAsignatura

Identificador	CU7
Nombre	EditarAsignatura
Descripción	El profesor edita la información de la asignatura.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol de profesor.3. El profesor ha de ser el responsable o creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Editar asignatura</i>.2. El sistema muestra un formulario con los datos para editar.3. El profesor introduce los nuevos datos.4. El sistema actualiza los datos de la asignatura en la base de datos.
Postcondiciones	<ol style="list-style-type: none">1. La asignatura queda actualizada en el sistema.
Flujo alternativo	<ol style="list-style-type: none">3. El profesor introduce los nuevos datos.<ol style="list-style-type: none">3.1. Si los datos no son nuevos, el sistema no actualiza la información en la base de datos.

Cuadro 8.16: Especificación del Caso de Uso: Editar Asignatura



CU8 - CrearAsignatura

Identificador	CU8
Nombre	CrearAsignatura
Descripción	El profesor crea una asignatura en el sistema introduciendo los datos necesarios.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol de profesor.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Crear asignatura</i>.2. El sistema muestra un formulario con los datos necesarios para su creación.3. El profesor introduce los datos.4. El sistema comprueba que los datos sean correctos.5. El sistema comprueba que no exista una asignatura con ese nombre.6. El sistema registra la asignatura en el sistema.
Postcondiciones	<ol style="list-style-type: none">1. La asignatura queda registrada en el sistema.
Flujo alternativo	<ol style="list-style-type: none">4. El sistema comprueba que los datos sean correctos.<ol style="list-style-type: none">4.1. Si algún dato no es correcto, se notifica al usuario y se impide la creación de la asignatura.5. El sistema comprueba que no exista una asignatura con ese nombre.<ol style="list-style-type: none">5.1. Si existe una asignatura con ese nombre, se notifica al usuario y se impide la creación de la asignatura.

Cuadro 8.17: Especificación del Caso de Uso: Crear Asignatura



CU9 - BorrarAsignatura

Identificador	CU9
Nombre	BorrarAsignatura
Descripción	El profesor borra la asignatura del sistema, consecuentemente se borrarán las actividades, entregas, e inscripciones.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol de profesor.3. El profesor ha de ser el responsable o creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Borrar asignatura</i> o <i>Eliminar asignatura</i>.2. El sistema pregunta confirmación para eliminarla.3. El usuario confirma el borrado.4. El sistema comprueba que el profesor sea el responsable o creador de la asignatura.5. El sistema borra la asignatura de la base de datos.
Postcondiciones	<ol style="list-style-type: none">1. La asignatura queda borrada del sistema.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema pregunta confirmación para eliminarla.<ol style="list-style-type: none">2.1. Si el usuario no confirma el borrado, se termina el caso de uso.4. El sistema comprueba que el profesor sea el responsable o creador de la asignatura.<ol style="list-style-type: none">4.1. Si el usuario no es el profesor responsable, se notifica y se termina el caso de uso.

Cuadro 8.18: Especificación del Caso de Uso: Borrar Asignatura

**CU10 - ExportarCalificacionesAsignatura**

Identificador	CU10
Nombre	ExportarCalificacionesAsignatura
Descripción	El profesor exporta las calificaciones de los alumnos en las distintas actividades para realizar una evaluación.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol de profesor.3. El profesor ha de ser el responsable o creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Exportar calificaciones</i>.2. El sistema comprueba que el usuario sea el profesor de la asignatura3. El sistema genera un archivo con las calificaciones de los alumnos en todas las actividades de la asignatura y su nota media.4. include (Enviar Correo Usuario).
Postcondiciones	<ol style="list-style-type: none">1. Un archivo con las calificaciones de los alumnos queda generado.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario sea el profesor de la asignatura<ol style="list-style-type: none">2.1. Si el usuario no es el profesor, el sistema lo notifica y se termina el caso de uso.

Cuadro 8.19: Especificación del Caso de Uso: Exportar Calificaciones Asignaturas

CU11 - EnviarCorreoUsuario

Identificador	CU11
Nombre	EnviarCorreoUsuario
Descripción	El usuario decide enviar un correo a la dirección de otro usuario.
Actor principal	Usuario
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona el correo de otro usuario.2. El sistema redirecciona a la aplicación nativa de correo del terminal con el correo del destinatario.
Postcondiciones	-
Flujo alternativo	-

Cuadro 8.20: Especificación del Caso de Uso: Enviar Correo a Usuario

**CU12 - VerAlumnosAsignatura**

Identificador	CU12
Nombre	VerAlumnosAsignatura
Descripción	El usuario obtiene una lista con los usuarios inscritos en la asignatura.
Actor principal	Usuario
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe ser el profesor de la asignatura o un alumno de la misma.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Ver alumnos</i>.2. El sistema comprueba que el usuario sea profesor o alumno de la asignatura.3. El sistema muestra los alumnos inscritos en la asignatura.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario sea profesor o alumno de la asignatura.<ol style="list-style-type: none">2.1. Si el usuario no tiene permisos, se notificará y cancelará el caso de uso.

Cuadro 8.21: Especificación del Caso de Uso: Ver Alumnos Asignatura

CU13 - VerPerfilUsuario

Identificador	CU13
Nombre	VerPerfilUsuario
Descripción	El usuario obtiene información de otro usuario, como su correo o las asignaturas que imparte o en las que está inscrito.
Actor principal	Usuario
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Ver perfil</i> de otro usuario.2. El sistema muestra los datos del usuario.
Postcondiciones	-
Flujo alternativo	-

Cuadro 8.22: Especificación del Caso de Uso: Ver Perfil Usuario

**CU14 - EditarPerfil**

Identificador	CU14
Nombre	EditarPerfil
Descripción	El usuario edita su información personal.
Actor principal	Usuario
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario que edita el perfil ha de ser él mismo.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Editar perfil</i>.2. El sistema solicita al usuario la información nueva mediante un formulario.3. El sistema actualiza la información en la base de datos.
Postcondiciones	<ol style="list-style-type: none">1. La nueva información del usuario queda actualizada en la base de datos.
Flujo alternativo	-

Cuadro 8.23: Especificación del Caso de Uso: Editar Perfil

CU15 - InscribirAsignatura

Identificador	CU15
Nombre	InscribirAsignatura
Descripción	El alumno se inscribe en una asignatura introduciendo el código de acceso válido.
Actor principal	Alumno
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol Alumno.3. La asignatura ha de existir en el sistema.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el alumno selecciona <i>Inscribir Asignatura</i>.2. El sistema pide el código de acceso.3. El alumno introduce el código.4. El sistema verifica que el código sea correcto.5. El sistema verifica que el alumno no esté inscrito en la asignatura.6. El sistema crea la inscripción y se lo notifica al alumno.
Postcondiciones	<ol style="list-style-type: none">1. El alumno queda inscrito en la asignatura.
Flujo alternativo	<ol style="list-style-type: none">3. El alumno introduce el código.<ol style="list-style-type: none">3.1. Si el código es erróneo, el sistema lo notifica y termina el caso de uso.5. El sistema verifica que el alumno no esté inscrito en la asignatura.<ol style="list-style-type: none">5.1. Si el alumno está inscrito previamente, el sistema lo notifica y termina el caso de uso.

Cuadro 8.24: Especificación del Caso de Uso: Inscribir Asignatura

**CU16 - VerMiProgreso**

Identificador	CU16
Nombre	VerMiProgreso
Descripción	El alumno observa el progreso que lleva para llevar un seguimiento.
Actor principal	Alumno
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol Alumno.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el alumno selecciona <i>Ver mi progreso</i>.2. El sistema comprueba que el alumno esté matriculado en la asignatura.3. El sistema comprueba las actividades y entregas de la asignatura.4. El sistema calcula los datos necesarios.5. El sistema muestra el progreso del alumno.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el alumno esté matriculado en la asignatura.<ol style="list-style-type: none">2.1. Si el alumno no está matriculado, el sistema lo notifica y termina el caso de uso.

Cuadro 8.25: Especificación del Caso de Uso: Ver Mi Progreso

CU17 - VerEstadísticasAsignatura

Identificador	CU17
Nombre	VerEstadísticasAsignatura
Descripción	El profesor obtiene estadísticas del curso para llevar un seguimiento del mismo.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener el rol Profesor.3. El profesor ha de ser el responsable/creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Ver estadísticas de la asignatura</i>.2. El sistema comprueba que el solicitante es el profesor de la asignatura.3. El sistema comprueba las actividades, entregas y alumnos de la asignatura.4. El sistema calcula los datos necesarios.5. El sistema muestra las estadísticas de la asignatura.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el solicitante es el profesor de la asignatura.<ol style="list-style-type: none">2.1. Si no es el profesor quien dispara el caso de uso, el sistema notifica que no tiene permisos y se cancela el caso de uso.

Cuadro 8.26: Especificación del Caso de Uso: Ver Estadísticas Asignatura



CU18 - VerActividadesAsignatura

Identificador	CU18
Nombre	VerActividadesAsignatura
Descripción	El usuario obtiene una lista con todas las actividades de una asignatura.
Actor principal	Usuario
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. En caso de ser el alumno, ha de estar inscrito en la asignatura.3. En caso de ser profesor, ha de ser el responsable/creador.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Ver actividades</i>.2. El sistema comprueba que el usuario tenga permiso.3. El sistema muestra las actividades de la asignatura. extension point: VerActividad
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario tenga permiso.<ol style="list-style-type: none">2.1. Si el usuario no tiene permisos, es notificado y se termina el caso de uso.

Cuadro 8.27: Especificación del Caso de Uso: Ver Actividades Asignatura

CU19 - CrearActividad

Identificador	CU19
Nombre	CrearActividad
Descripción	El profesor crea una actividad para la asignatura introduciendo los datos necesarios para ello.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol Profesor.3. El usuario ha de ser el profesor responsable/creador.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Añadir actividad</i>.2. El sistema comprueba que el usuario tenga permisos.3. El sistema pide los datos necesarios para crear la actividad a través de un formulario.4. El usuario introduce los datos en el formulario.5. El sistema comprueba que los datos estén correctos.6. El sistema registra la actividad en la base de datos.
Postcondiciones	<ol style="list-style-type: none">1. La actividad de la asignatura queda registrada en el sistema.
Flujo alternativo	<ol style="list-style-type: none">5. El sistema comprueba que los datos estén correctos.<ol style="list-style-type: none">5.1. Si algún dato obligatorio no es introducido o es incorrecto, el sistema lo notifica y vuelve al paso 3 del flujo principal.

Cuadro 8.28: Especificación del Caso de Uso: Crear Actividad



CU20 - BorrarActividad

Identificador	CU20
Nombre	BorrarActividad
Descripción	El profesor borra una actividad de la asignatura y consecuentemente las entregas hechas hasta ese momento.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol Profesor.3. El usuario ha de ser el profesor responsable/creador.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Borrar actividad</i>.2. El sistema comprueba que el usuario tenga permisos.3. El sistema borra las entregas hechas para esa actividad.4. El sistema borra la actividad.
Postcondiciones	<ol style="list-style-type: none">1. La actividad de la asignatura queda eliminada del sistema.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario tenga permisos.<ol style="list-style-type: none">2.1. Si el usuario no es el profesor de la asignatura, el sistema lo notifica y se termina el caso de uso.

Cuadro 8.29: Especificación del Caso de Uso: Borrar Actividad

CU21 - EditarActividad

Identificador	CU21
Nombre	EditarActividad
Descripción	El profesor edita los datos de una actividad.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol Profesor.3. El usuario ha de ser el profesor responsable/creador.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Editar actividad</i>.2. El sistema comprueba que el usuario tenga permisos.3. El sistema pide los datos necesarios para editar la actividad a través de un formulario.4. El usuario introduce los datos en el formulario.5. El sistema comprueba que los datos estén correctos.6. El sistema actualiza la actividad en la base de datos.
Postcondiciones	<ol style="list-style-type: none">1. La actividad de la asignatura queda actualizada en el sistema.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario tenga permisos.<ol style="list-style-type: none">2.1. Si el usuario no es el profesor de la asignatura, el sistema lo notifica y se termina el caso de uso.5. El sistema comprueba que los datos estén correctos.<ol style="list-style-type: none">5.1. Si algún dato obligatorio no es introducido o es incorrecto, el sistema lo notifica y vuelve al paso 3 del flujo principal.

Cuadro 8.30: Especificación del Caso de Uso: Editar Actividad



CU22 - VerActividad

Identificador	CU22
Nombre	VerActividad
Descripción	El usuario obtiene la información de una actividad.
Actor principal	Usuario
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. Si el usuario es profesor, ha de ser el profesor responsable/creador.3. Si el usuario es alumno, ha de estar inscrito en la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona <i>Ver actividad</i>.2. El sistema comprueba que el usuario tenga permisos.3. El sistema muestra los detalles de la actividad.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario tenga permisos.<ol style="list-style-type: none">2.1. Si el usuario no tiene permisos, el sistema lo notifica y se termina el caso de uso.

Cuadro 8.31: Especificación del Caso de Uso: Ver Actividad

CU23 - VerEntregasActividad

Identificador	CU23
Nombre	VerEntregasActividad
Descripción	El profesor obtiene un listado con las entregas hechas por los alumnos para una actividad de la asignatura..
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol Profesor.3. El usuario ha de ser el profesor responsable/creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el usuario selecciona "Ver entregas".2. El sistema recopila las entregas de la actividad de los alumnos de la asignatura.3. El sistema muestra las entregas de la actividad mediante un listado en formato Alumno-Calificación. extension point: VerEntrega extension point: CalificarEntregaManualmente extension point: ExportarEntregas
Postcondiciones	-
Flujo alternativo	-

Cuadro 8.32: Especificación del Caso de Uso: Ver Entregas de Actividad

**CU24 - CalificarEntregaManualmente**

Identificador	CU24
Nombre	CalificarEntregaManualmente
Descripción	El profesor califica una entrega hecha de un alumno, actualizando la calificación.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol Profesor.3. El usuario ha de ser el profesor responsable/creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Calificar</i> en una entrega.2. El sistema comprueba que el usuario sea el profesor de la asignatura.3. El sistema le pide mediante un formulario la nueva nota para calificar.4. El profesor introduce la nueva nota.5. El sistema actualiza la calificación de la entrega.6. El sistema actualiza automáticamente la retroalimentación de la entrega con <i>Evaluado manualmente por el profesor</i>.
Postcondiciones	<ol style="list-style-type: none">1. La entrega queda actualizada con la nueva calificación.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario sea el profesor de la asignatura.<ol style="list-style-type: none">2.1. Si el usuario no es el profesor de la asignatura, el sistema lo notifica y se termina el caso de uso.

Cuadro 8.33: Especificación del Caso de Uso: Calificar Entrega Manualmente

**CU25 - ExportarEntregas**

Identificador	CU25
Nombre	ExportarEntregas
Descripción	El profesor exporta en un archivo las notas de las entregas hechas por los alumnos de una actividad y la envía por correo electrónico.
Actor principal	Profesor
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol Profesor.3. El usuario ha de ser el profesor responsable/creador de la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso comienza cuando el profesor selecciona <i>Exportar entregas</i>.2. El sistema comprueba que el usuario sea el profesor de la asignatura.3. El sistema genera un archivo con las calificaciones de los alumnos en todas las actividades de la asignatura y su nota media.4. Include (<i>EnviarCorreoUsuario</i>).
Postcondiciones	<ol style="list-style-type: none">1. Un archivo con las calificaciones de los alumnos queda generado.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario sea el profesor de la asignatura.<ol style="list-style-type: none">2.1. Si el usuario no es el profesor de la asignatura, el sistema lo notifica y se termina el caso de uso.

Cuadro 8.34: Especificación del Caso de Uso: Exportar Entregas

CU26 - VerEntrega

Identificador	CU26
Nombre	VerEntrega
Descripción	El usuario obtiene los detalles de una entrega siempre que tenga los permisos para ello.
Actor principal	Usuario Logueado
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de ser el profesor responsable/creador de la asignatura o el propio alumno que ha realizado la entrega.3. La entrega debe de existir en el sistema.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso se dispara cuando el usuario pulsa sobre <i>Ver entrega</i>.2. El sistema comprueba que el usuario esté autorizado.3. El sistema le proporciona la información de la entrega al usuario.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el usuario esté autorizado.<ol style="list-style-type: none">2.1. Si el usuario no está autorizado, el sistema lo notificará y el caso de uso termina.

Cuadro 8.35: Especificación del Caso de Uso: Ver Entrega



CU27 - HacerEntrega

Identificador	CU27
Nombre	HacerEntrega
Descripción	El alumno realiza una entrega de una actividad con la posibilidad de escanear el texto de una imagen y evaluando la actividad mediante un modelo de I.A.
Actor principal	Alumno
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol alumno.3. El alumno ha de estar inscrito en la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso se dispara cuando el usuario pulsa sobre <i>Hacer Entrega</i>.2. El sistema comprueba que el alumno esté matriculado en la asignatura.3. El sistema comprueba que el alumno no haya realizado una entrega antes.4. El sistema comprueba que la fecha actual sea menor a la fecha límite de entrega.5. El sistema solicita al usuario la solución para la entrega de la actividad.6. El usuario introduce la solución de la actividad.7. Extension point: escanearOCR.8. El sistema guarda la solución en la base de datos.9. Include: EvaluarSolucion.
Postcondiciones	<ol style="list-style-type: none">1. El sistema guarda la solución subida por el alumno en la base de datos.
Flujo alternativo	<ol style="list-style-type: none">3. El sistema comprueba que el alumno no haya realizado una entrega antes.<ol style="list-style-type: none">3.1. Si el alumno ya había realizado una entrega, se le mostrará la misma (<i>VerEntrega</i>).4. El sistema comprueba que la fecha actual sea menor a la fecha límite de entrega.<ol style="list-style-type: none">4.1. Si la fecha actual es mayor a la límite, se le notificará al usuario y se le impedirá subir la solución.

Cuadro 8.36: Especificación del Caso de Uso: Hacer Entrega

**CU28 - EscanearOCR**

Identificador	CU28
Nombre	EscanearOCR
Descripción	El sistema escanea mediante un modelo de I.A. el contenido en texto plano de una imagen.
Actor principal	Alumno
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol alumno.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso se dispara cuando el usuario sube una imagen para la solución de la actividad.2. El sistema evalúa mediante un modelo de inteligencia artificial el texto de la imagen.3. El sistema le muestra al usuario el contenido en texto plano editable.
Postcondiciones	-
Flujo alternativo	<ol style="list-style-type: none">2. El sistema evalúa mediante un modelo de inteligencia artificial el texto de la imagen.<ol style="list-style-type: none">2.1. Si el modelo no está disponible, el caso de uso se termina.

Cuadro 8.37: Especificación del Caso de Uso: EscanearOCR



CU29 - EvaluarSolución

Identificador	CU29
Nombre	EvaluarSolución
Descripción	El sistema evalúa mediante un modelo de I.A. la solución aportada por un alumno en su entrega en base a unos criterios impuestos en la actividad y el enunciado de la misma.
Actor principal	Alumno
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario debe tener la entrega pendiente de evaluar con una solución.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso se dispara cuando el usuario pulsa sobre <i>Hacer Entrega</i>.2. El sistema comprueba que el campo con la solución no esté vacío.3. El sistema evalúa mediante un modelo de inteligencia artificial la solución en base al enunciado y los requisitos.4. El sistema guarda la retroalimentación de la solución aportada por el modelo.5. El sistema guarda la calificación de la solución aportada por el modelo.
Postcondiciones	<ol style="list-style-type: none">1. El sistema guarda en la base de datos la retroalimentación y la calificación de la entrega realizada.
Flujo alternativo	<ol style="list-style-type: none">2. El sistema comprueba que el campo con la solución no esté vacío.<ol style="list-style-type: none">2.1. Si la solución está vacía, se notifica al usuario que ha de aportarla.3. El sistema evalúa mediante un modelo de inteligencia artificial la solución en base al enunciado y los requisitos.<ol style="list-style-type: none">3.1. Si el modelo no está disponible para la evaluación, guardará la solución y dejará la entrega como pendiente de calificación.

Cuadro 8.38: Especificación del Caso de Uso: Evaluar Solución



CU30 - SalirAsignatura

Identificador	CU30
Nombre	SalirAsignatura
Descripción	El alumno puede darse de baja voluntariamente de una asignatura en la que está inscrito.
Actor principal	Alumno
Actor secundario	-
Precondiciones	<ol style="list-style-type: none">1. El usuario debe tener la sesión iniciada en el sistema.2. El usuario ha de tener el rol alumno.3. El alumno ha de estar inscrito en la asignatura.
Flujo principal	<ol style="list-style-type: none">1. El caso de uso se dispara cuando el alumno pulsa sobre la opción <i>Salir de la asignatura</i>.2. El sistema comprueba que el alumno esté efectivamente inscrito en la asignatura.3. El sistema solicita confirmación al alumno para proceder con la baja.4. El alumno confirma su intención de abandonar la asignatura.5. El sistema elimina la inscripción del alumno en la asignatura.
Postcondiciones	<ol style="list-style-type: none">1. El alumno ya no está inscrito en la asignatura seleccionada.
Flujo alternativo	<ol style="list-style-type: none">4. El alumno confirma su intención de abandonar la asignatura.<ol style="list-style-type: none">4.1. Si el alumno cancela la acción, el caso de uso se termina sin cambios.

Cuadro 8.39: Especificación del Caso de Uso: Salir de Asignatura

8.1.5 Matriz de seguimiento

La matriz de seguimiento es una técnica utilizada para comprobar que en la fase de análisis, los casos de uso cumplen con los requisitos que se han extraído en la elicitation de los mismos. Esto servirá para hacernos una idea de qué casos de uso cubren qué requisitos y viceversa, permitiéndonos identificar aquellos requisitos funcionales que no se han tenido en cuenta y volver a replantear los casos de uso.



Cuadro 8.40: Matriz de Trazabilidad entre Casos de Uso
y Requisitos Funcionales

Caso de Uso	Requisito Funcional																											
	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9	RF10	RF11	RF12	RF13	RF14	RF15	RF16	RF17	RF18	RF19	RF20	RF21	RF22	RF23	RF24	RF25	RF26	RF27	
CU-1	X	X																										
CU-2			X																									
CU-3			X																									
CU-4																												X
CU-5					X																							
CU-6	X																				X	X						
CU-7	X							X																				
CU-8	X							X																				
CU-9	X								X																			
CU-10	X																			X								
CU-11							X																					
CU-12																				X								X
CU-13				X																								
CU-14				X																								
CU-15	X																					X						
CU-16	X																							X				
CU-17	X			X																		X						
CU-18	X													X										X				
CU-19	X														X													

Continuará en la siguiente página



Cuadro 8.40 – continuación

Caso de Uso	Requisito Funcional																									
	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9	RF10	RF11	RF12	RF13	RF14	RF15	RF16	RF17	RF18	RF19	RF20	RF21	RF22	RF23	RF24	RF25	RF26
CU-20	X											X														
CU-21	X											X														
CU-22													X										X			
CU-23	X												X													
CU-24	X														X											
CU-25	X													X												
CU-26													X													
CU-27	X																						X			
CU-28	X																								X	
CU-29	X																								X	
CU-30																			X							



8.1.6 Descripción procedural - Diagramas de Actividad

Los diagramas de actividad nos sirven para describir procesos de negocio o flujos de trabajo relacionados con el software que se desarrolla. Pueden servir como una especie de contrato entre el cliente y el desarrollador o la empresa desarrolladora a través del cual se describe cómo funcionará el sistema para llevar a cabo cierta funcionalidad.

Dado la gran cantidad de casos de uso que tenemos, he decidido realizar un diagrama de actividad para los casos de uso que considero no triviales, es decir, casos de uso que puedan llevar más complejidad asociada, de manera que, mediante estos diagramas de actividad, podremos aumentar la comprensibilidad de las funcionalidades a diseñar e implementar. Los casos de uso que se modelarán mediante diagramas de actividad son:

- CU19 - CrearActividad
- CU24 - CalificarEntregaManualmente
- CU27 - HacerEntrega
- CU29 - EvaluarSolución

Diagrama Actividad CU19 - CrearActividad

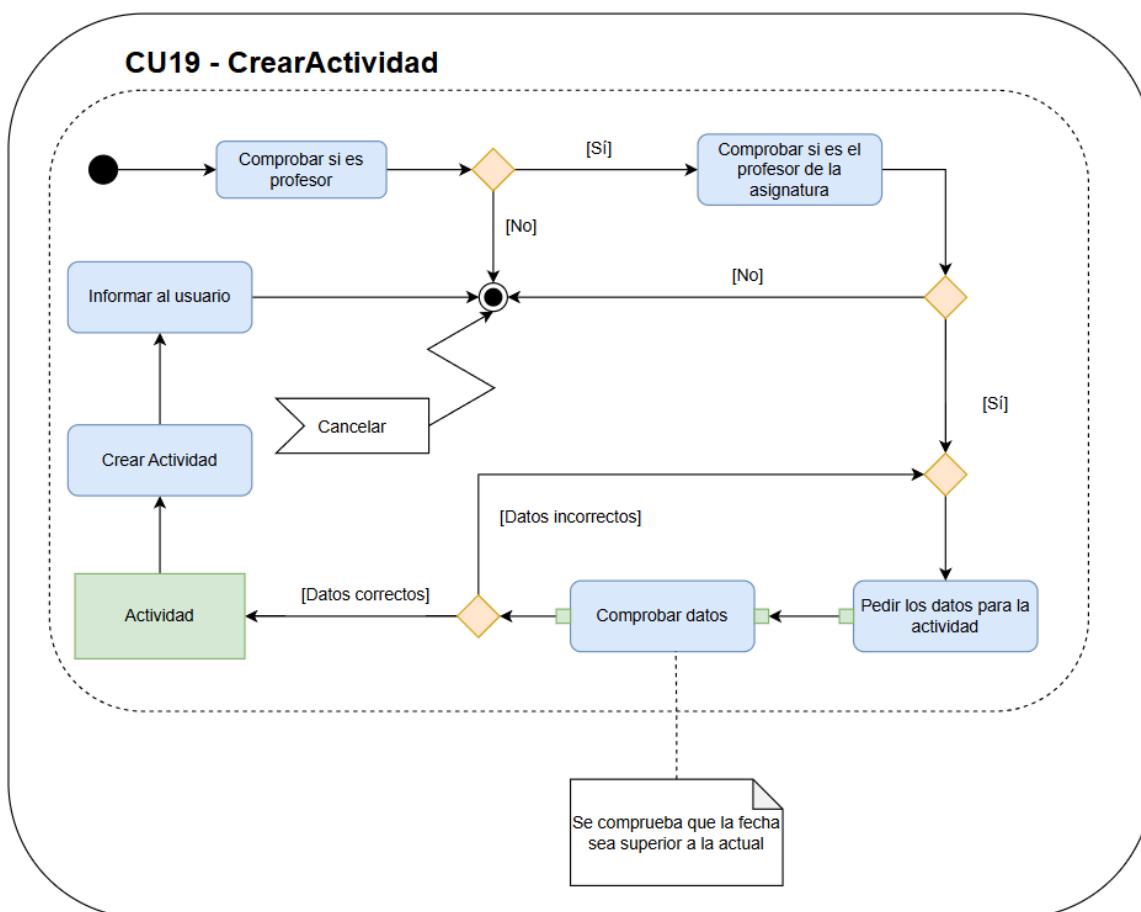


Figura 8.12: Diagrama Actividad CU19



Diagrama Actividad CU24 - CalificarEntregaManualmente

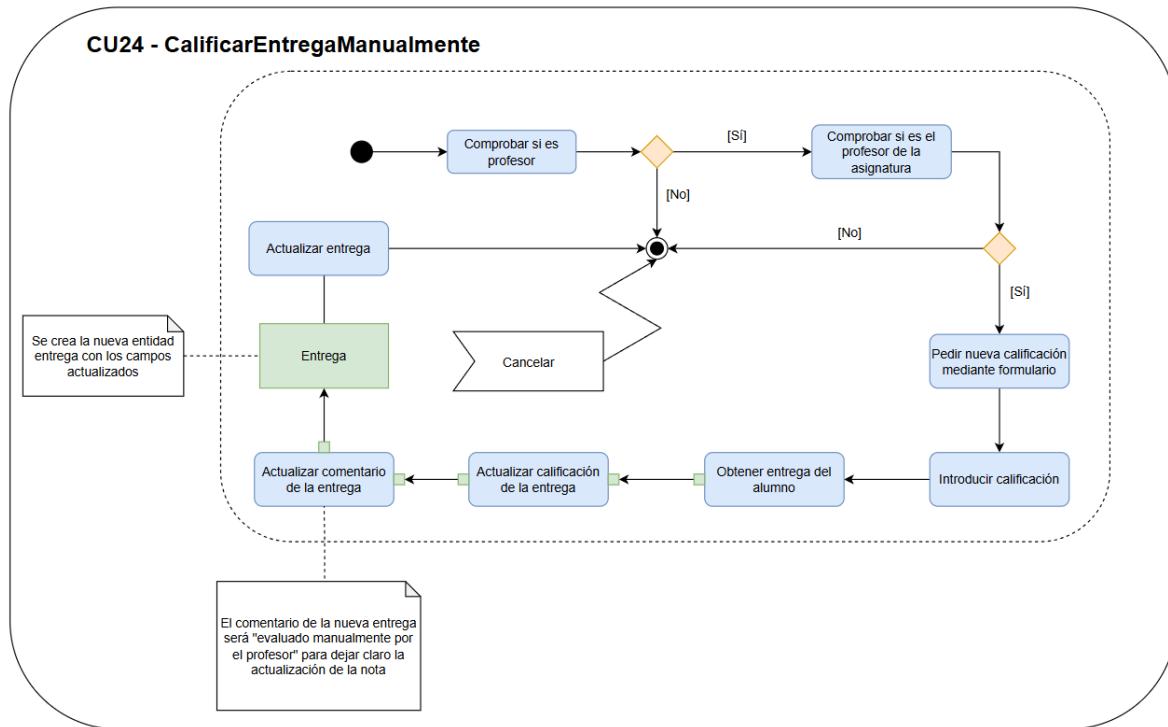


Figura 8.13: Diagrama Actividad CU24

Diagrama Actividad CU27 - HacerEntrega

En este caso, están implicados otro caso de uso, por lo que haremos uso del nodo de llamada y realizaremos su respectivo diagrama de actividad.

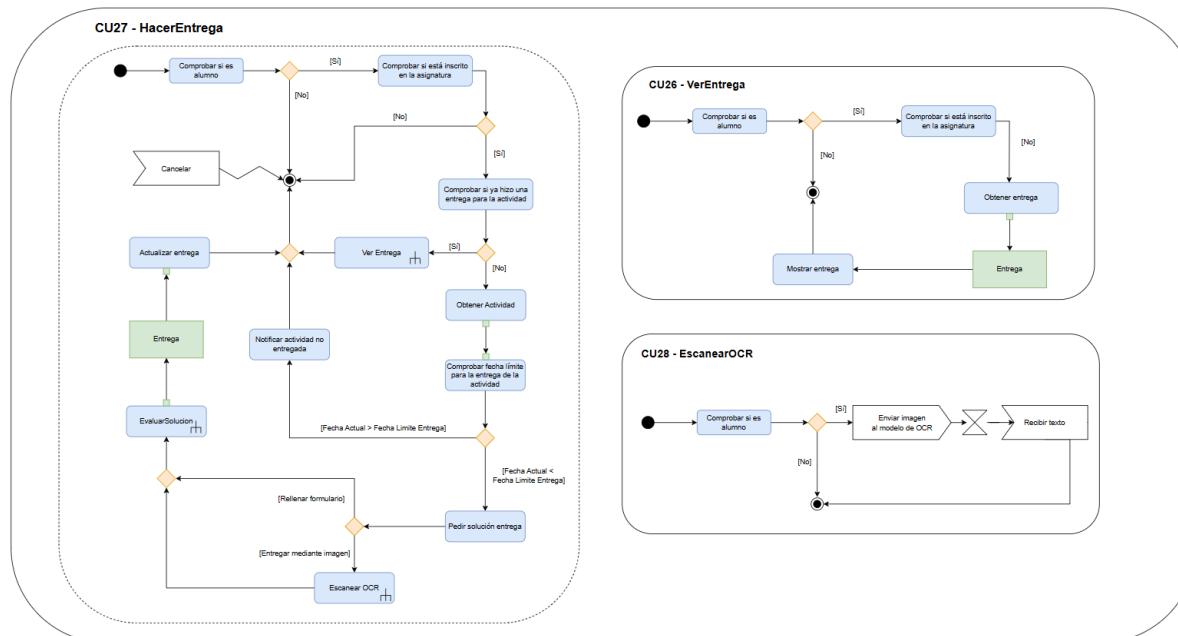


Figura 8.14: Diagrama Actividad CU27, CU26, CU28



Diagrama Actividad CU29 - EvaluarSolución

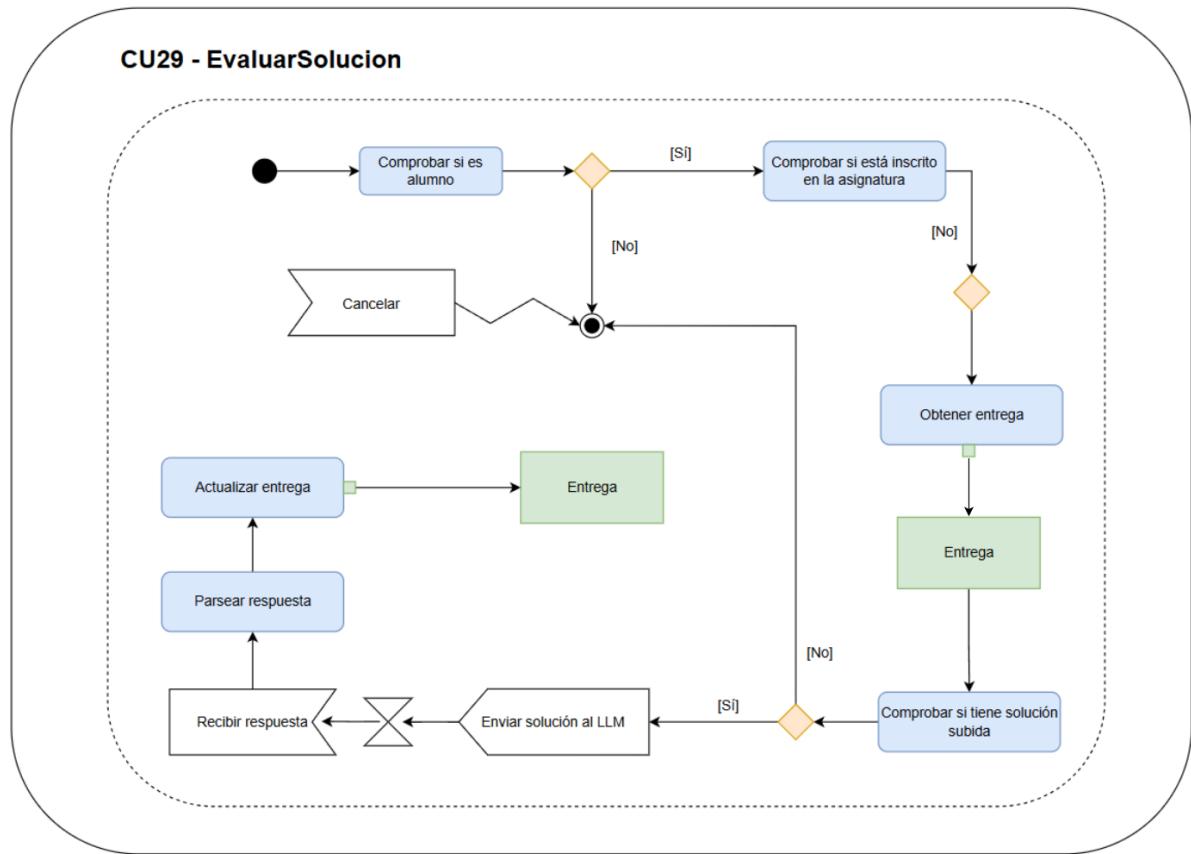


Figura 8.15: Diagrama Actividad CU29

8.1.7 Descripción procedural - Diagramas de Secuencia

Los diagramas de secuencia se utilizan para modelar la interacción entre diferentes objetos o participantes en el sistema a lo largo del tiempo. Muestran la secuencia de mensajes intercambiados para llevar a cabo una funcionalidad específica, en este caso, en análisis hecho a muy alto nivel, pues dichos objetos no serán más que el actor, el sistema, y la base de datos. A continuación, se describen los diagramas de secuencia correspondientes a los diagramas de actividad previamente mencionados.



Diagrama Secuencia CU19 - CrearActividad

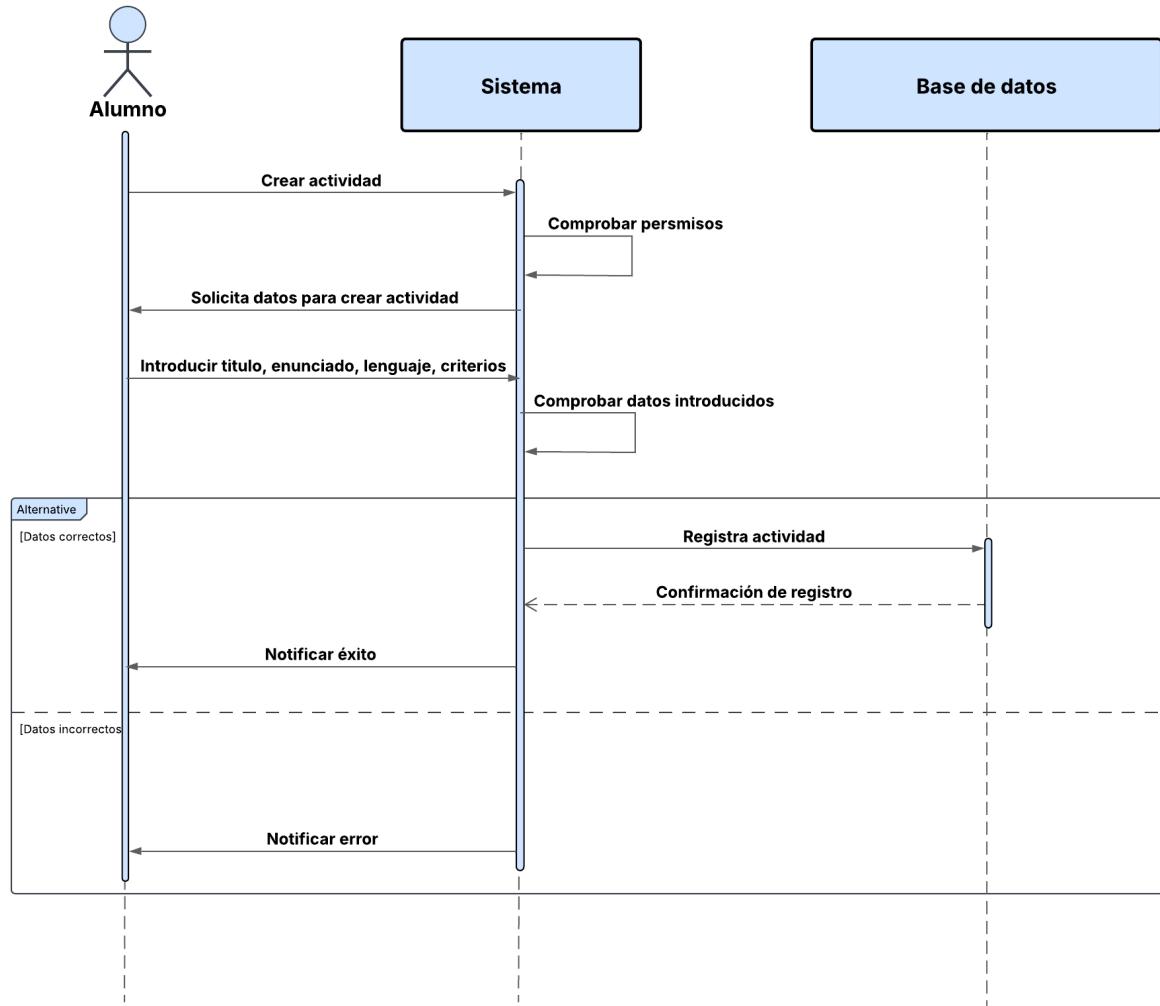


Figura 8.16: Diagrama Secuencia CU19



Diagrama Secuencia CU24 - CalificarEntregaManualmente

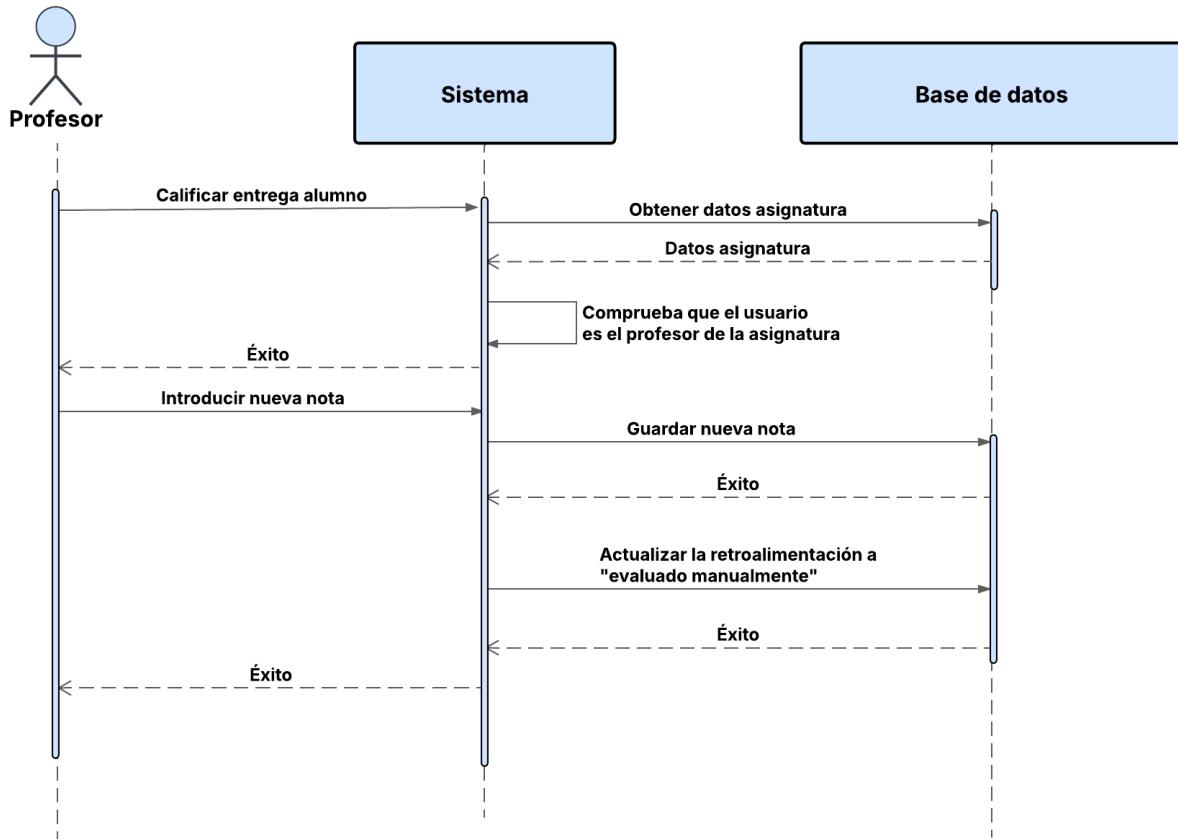


Figura 8.17: Diagrama Secuencia CU24



Diagrama Secuencia CU27 - HacerEntrega y CU29 - Evaluar soluci

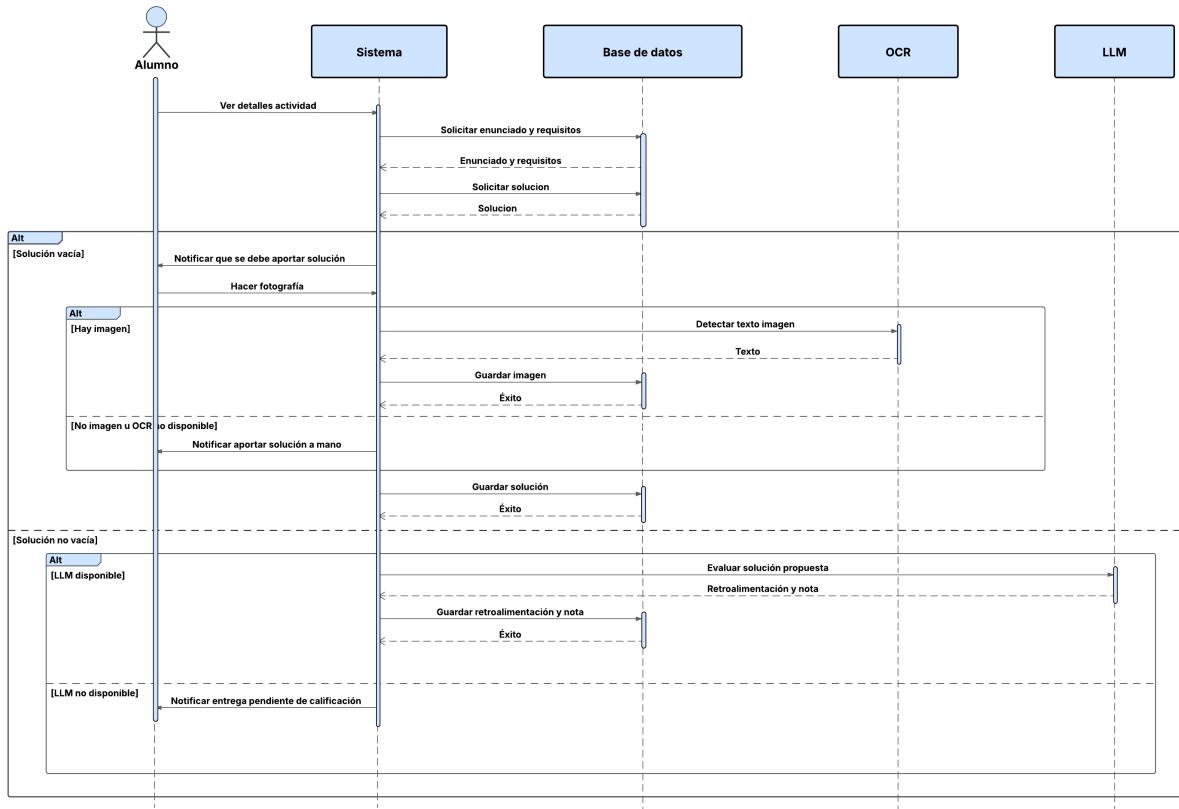


Figura 8.18: Diagrama Secuencia CU27 y CU29

8.2 Diseño de la aplicación

Este capítulo estará orientado al diseño de la aplicación. Es la base de una futura implementación para asegurar el cumplimiento de los requerimientos que hemos definido en análisis. Para llevar a cabo el diseño, realizaremos las clases de diseño, y consecuentemente el diseño de la estructura de la base de datos y la navegación, hablaremos también de la arquitectura.

8.2.1 Diagrama de clases

En la figura 8.19, vemos el diagrama de clases de diseño que conforma nuestra aplicación. Un diagrama de clases es una representación gráfica de las clases que componen un sistema, mostrando sus atributos, métodos y las relaciones entre ellas. Este diagrama nos permite visualizar la estructura estática del sistema y comprender cómo interactúan las diferentes clases para cumplir con los requisitos del proyecto.

En este diagrama, hemos modelado las entidades principales identificadas en el modelo entidad-relación, transformándolas en clases con sus respectivos atributos y métodos. Además, hemos definido las relaciones entre estas clases, como la herencia, la asociación y la agregación, para representar las interacciones y dependencias entre ellas.

A continuación, describiremos brevemente cada una de las clases representadas en el diagrama, destacando sus atributos y métodos más relevantes, así como las relaciones



CAPÍTULO 8. ESPECIFICACIÓN DEL SISTEMA

que mantienen con otras clases. Esto nos permitirá comprender mejor la estructura y el funcionamiento de nuestra aplicación.

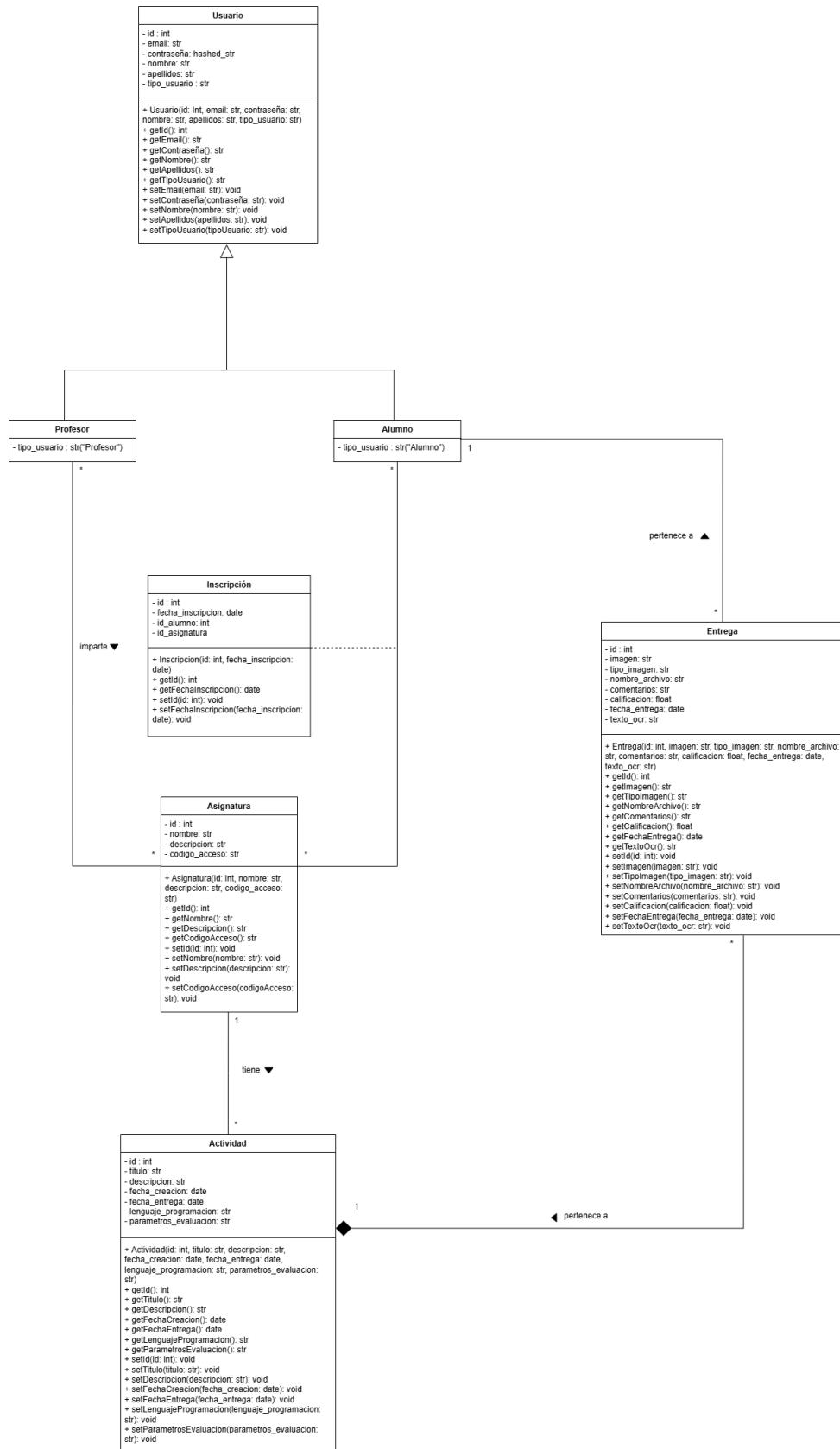


Figura 8.19: Diagrama de clases



Justificación Técnica de las Relaciones

En el diagrama de clases presentado en la Figura 8.19, hemos establecido diversas relaciones entre las clases para modelar la estructura y el comportamiento de nuestra aplicación. A continuación, justificaremos técnicamente cada una de estas relaciones:

Herencia entre Usuario, Profesor y Alumno

La relación de herencia entre la clase base *Usuario* y las clases derivadas *Profesor* y *Alumno* se justifica por el principio de es-un. Tanto profesores como alumnos son tipos de usuarios, compartiendo atributos y métodos comunes (como *id*, *nombre*, *email*, etc.). La herencia nos permite reutilizar código y modelar la jerarquía de usuarios de manera eficiente.

Asociación entre Profesor y Asignatura

La relación de asociación entre *Profesor* y *Asignatura* se establece porque un profesor puede impartir varias asignaturas, y una asignatura es impartida por un profesor. Esta relación representa la responsabilidad del profesor sobre las asignaturas que gestiona.

Clase de Asociación Inscripción entre Alumno y Asignatura

La clase de asociación *Inscripción* entre *Alumno* y *Asignatura* se introduce para modelar la relación muchos-a-muchos entre estas dos clases. Además, *Inscripción* tiene su propio atributo (*fecha_inscripcion*), lo que justifica su existencia como una clase independiente que almacena información adicional sobre la relación.

Composición entre Actividad y Entrega

La relación de composición entre *Actividad* y *Entrega* se justifica porque una entrega no puede existir sin una actividad. La entrega es una parte integral de la actividad, y si se elimina una actividad, todas sus entregas asociadas también deben eliminarse. La composición representa esta fuerte relación de dependencia y ciclo de vida compartido.

Asociación entre Alumno y Entrega

La relación de asociación entre *Alumno* y *Entrega* se establece porque un alumno realiza una entrega. Esta relación representa la acción del alumno de entregar una solución a una actividad.

Asociación entre Asignatura y Actividad

La relación de asociación entre *Asignatura* y *Actividad* se establece porque una asignatura puede tener varias actividades. Esta relación representa la organización de las actividades dentro de una asignatura.

Estas relaciones han sido cuidadosamente seleccionadas para representar con precisión las interacciones y dependencias entre las entidades de nuestro sistema, asegurando un diseño robusto y mantenable.

8.2.2 Clases del diagrama

En esta sección, mediante un diseño tabular, describiremos las clases usadas para el diseño del proyecto.



Clase Usuario

Descripción	
Esta clase representa al usuario que se registra en la aplicación.	
Atributos	
Atributo	Descripción
id: int	Identificador del usuario.
email: str	Correo electrónico del usuario.
contraseña: hashed_str	Contraseña encriptada del usuario.
nombre: str	Nombre del usuario.
apellidos: str	Apellidos del usuario.
tipo_usuario: str	Tipo de usuario (Profesor o Alumno).
Métodos	
Método	Descripción
Usuario(id, email, contraseña, nombre, apellidos, tipo_usuario)	Constructor de la clase.
getId()	Obtiene el identificador del usuario.
getEmail()	Obtiene el correo electrónico.
getContraseña()	Obtiene la contraseña encriptada.
getNombre()	Obtiene el nombre del usuario.
getApellidos()	Obtiene los apellidos del usuario.
getTipoUsuario()	Obtiene el tipo de usuario.
setEmail(email)	Modifica el correo electrónico.
setContraseña(contraseña)	Modifica la contraseña.
setNombre(nombre)	Modifica el nombre.
setApellidos(apellidos)	Modifica los apellidos.
setTipoUsuario(tipo)	Modifica el tipo de usuario.

Cuadro 8.41: Clase Usuario



Clase Profesor

Descripción	
Esta clase representa a un profesor en la aplicación, heredando de la clase Usuario.	
Atributos	
Atributo	Descripción
hereda de Usuario	Hereda todos los atributos de la clase Usuario.
Métodos	
Método	Descripción
Profesor(id, email, contraseña, nombre, apellidos)	Constructor de la clase Profesor.
hereda de Usuario	Hereda todos los métodos de la clase Usuario.

Cuadro 8.42: Clase Profesor

Clase Alumno

Descripción	
Esta clase representa a un alumno en la aplicación, heredando de la clase Usuario.	
Atributos	
Atributo	Descripción
hereda de Usuario	Hereda todos los atributos de la clase Usuario.
Métodos	
Método	Descripción
Alumno(id, email, contraseña, nombre, apellidos)	Constructor de la clase Alumno.
hereda de Usuario	Hereda todos los métodos de la clase Usuario.

Cuadro 8.43: Clase Alumno



Clase Asignatura

Descripción	
Esta clase representa una asignatura en la aplicación.	
Atributos	
Atributo	Descripción
id: int	Identificador de la asignatura.
nombre: str	Nombre de la asignatura.
descripcion: str	Descripción de la asignatura.
codigo_acceso: str	Código de acceso para la asignatura.
Métodos	
Método	Descripción
Asignatura(id, nombre, descripcion, codigo_acceso)	Constructor de la clase.
getId()	Obtiene el identificador de la asignatura.
getNombre()	Obtiene el nombre de la asignatura.
getDescripcion()	Obtiene la descripción de la asignatura.
getCodigoAcceso()	Obtiene el código de acceso.
setId(id)	Modifica el identificador.
setNombre(nombre)	Modifica el nombre.
setDescripcion(descripcion)	Modifica la descripción.
setCodigoAcceso(codigo)	Modifica el código de acceso.

Cuadro 8.44: Clase Asignatura



Clase Inscripcion

Descripción	
Esta clase representa la inscripción de un alumno en una asignatura.	
Atributos	
Atributo	Descripción
id: int	Identificador de la inscripción.
fecha_inscripcion: date	Fecha de la inscripción.
Métodos	
Método	Descripción
Inscripcion(id, fecha_inscripcion)	Constructor de la clase.
getId()	Obtiene el identificador de la inscripción.
getFechaInscripcion()	Obtiene la fecha de inscripción.
setId(id)	Modifica el identificador.
setFechaInscripcion(fecha_inscripcion)	Modifica la fecha de inscripción.

Cuadro 8.45: Clase Inscripcion



Clase Actividad

Descripción	
Esta clase representa una actividad en la aplicación.	
Atributos	
Atributo	Descripción
id: int	Identificador de la actividad.
titulo: str	Título de la actividad.
descripcion: str	Descripción de la actividad.
fecha_creacion: date	Fecha de creación de la actividad.
fecha_entrega: date	Fecha de entrega de la actividad.
lenguaje_programacion: str	Lenguaje de programación requerido.
parametros_evaluacion: str	Parámetros de evaluación de la actividad.
Métodos	
Método	Descripción
Actividad(id, titulo, descripcion, fecha_creacion, fecha_entrega, lenguaje_programacion, parametros_evaluacion)	Constructor de la clase.
getId()	Obtiene el identificador de la actividad.
getTitulo()	Obtiene el título de la actividad.
getDescripcion()	Obtiene la descripción de la actividad.
getFechaCreacion()	Obtiene la fecha de creación.
getFechaEntrega()	Obtiene la fecha de entrega.
getLenguajeProgramacion()	Obtiene el lenguaje de programación requerido.
getParametrosEvaluacion()	Obtiene los parámetros de evaluación.
setId(id)	Modifica el identificador.
setTitulo(titulo)	Modifica el título.
setDescripcion(descripcion)	Modifica la descripción.
setFechaCreacion(fecha_creacion)	Modifica la fecha de creación.
setFechaEntrega(fecha_entrega)	Modifica la fecha de entrega.
setLenguajeProgramacion(lenguaje_programacion)	Modifica el lenguaje de programación.
setParametrosEvaluacion(parametros_evaluacion)	Modifica los parámetros de evaluación.

Cuadro 8.46: Clase Actividad





Clase Entrega

Descripción	
Atributos	
Atributo	Descripción
id: int	Identificador de la entrega.
imagen: str	Ruta o referencia a la imagen de la entrega.
tipo_imagen: str	Tipo de la imagen (por ejemplo, jpg, png).
nombre_archivo: str	Nombre del archivo de la entrega.
comentarios: str	Comentarios adicionales de la entrega.
calificacion: float	Calificación asignada a la entrega.
fecha_entrega: date	Fecha de entrega.
texto_ocr: str	Texto extraído de la imagen mediante OCR.
Métodos	
Método	Descripción
Entrega(id, imagen, tipo_imagen, nombre_archivo, comentarios, calificacion, fecha_entrega, texto_ocr)	Constructor de la clase.
getId()	Obtiene el identificador de la entrega.
getImagen()	Obtiene la ruta o referencia a la imagen.
getTipoImagen()	Obtiene el tipo de la imagen.
getNombreArchivo()	Obtiene el nombre del archivo.
getComentarios()	Obtiene los comentarios.
getCalificacion()	Obtiene la calificación.
getFechaEntrega()	Obtiene la fecha de entrega.
getTextoOcr()	Obtiene el texto extraído mediante OCR.
setId(id)	Modifica el identificador.
setImagen(imagen)	Modifica la ruta o referencia a la imagen.
setTipoImagen(tipo_imagen)	Modifica el tipo de la imagen.
setNombreArchivo(nombre_archivo)	Modifica el nombre del archivo.
setComentarios(comentarios)	Modifica los comentarios.
setCalificacion(calificacion)	Modifica la calificación.
setFechaEntrega(fecha_entrega)	Modifica la fecha de entrega.
setTextoOcr(texto_ocr)	Modifica el texto extraído mediante OCR.

Cuadro 8.47: Clase Entrega



8.3 Implementación

La aplicación sigue una arquitectura cliente-servidor [2]. El cliente será la aplicación móvil, que mostrará todas las interfaces que permitirán al usuario comunicarse con la aplicación, por otro lado, el servidor manejará las peticiones que el usuario requiera para realizar las funcionalidades y operaciones contra la base de datos.

- **Servidor:** Como hemos detallado, el servidor es el que se encarga de manejar las peticiones y de la mayor parte de la lógica de negocio. La parte del servidor se encarga de manejar las operaciones CRUD[32] lanzadas contra las entidades que conforman la información del dominio de la solución, siguiendo una arquitectura API REST, la cual permite establecer endpoints para que la comunicación sea vía HTTPS[4]. Asimismo, para la parte de la I.A., se dispondrá de otro servidor aparte para externalizar operaciones de cómputo que puedan sobrecargar el resto de operaciones del servidor, y la justificación técnica es justamente esa, ya que, sabemos que solamente la carga de modelos de lenguaje, supone un gran uso de recursos del sistema, por lo que se quiere aislar del *core* de la aplicación para evitar privar a los usuarios de las funcionalidades principales en caso de una sobrecarga o caída del servicio. Dentro del servidor tendremos varios módulos que coincidirán con las entidades que manejamos.
 - Entregas: Operaciones CRUD sobre las entregas, también aquí realizaremos las operaciones relacionadas con la I.A., ya que funcionarán sobre las entregas.
 - Auth: Este módulo se encargará exclusivamente de la lógica para manejar el token de inicio de sesión.
 - Asignaturas: Operaciones CRUD sobre las asignaturas, en su mayoría autorizadas para el profesor, pues es quien tendrá más permisos sobre la misma
 - Usuarios: Operaciones CRUD sobre los usuarios.
 - Inscripciones: Operaciones CRUD sobre las inscripciones, permiten matricularse los alumnos o expulsarlos.
 - Actividades: Operaciones CRUD sobre las actividades.
- **Cliente:** Como hemos dicho, se encargará de presentar la información y de servir como punto para interactuar entre los usuarios y la información. La aplicación móvil se dividirá en widgets y pantallas, si bien es cierto que maneja algo de lógica, no es muy cargante ya que el peso se lo lleva el servidor. La aplicación en su arquitectura también manejará los mismos modelos que maneja el servidor, por lo que las entidades que hemos diseñado anteriormente se reutilizarán, esto se realiza para tener el código más limpio y aplicar orientación a objetos, aunque no proceda en esta sección, cabe decir que en la implementación se usará código limpio.

Cabe destacar que, la API principal necesita de otra API que está desarrollada para funcionar en dispositivos capaces de ofrecer la suficiente capacidad de cómputo para llevar a cabo las funcionalidades de IA, esto ya lo veremos con más profundidad en la implementación, pero, no está de más nombrarlo, ya que, en realidad es otra parte más de la arquitectura. que decidí personalmente implementarla por separado para no tener unos requisitos muy exigentes para la aplicación base, y dejar el módulo de



Inteligencia Artificial separado para que pueda ser desplegado en distintos equipos aliviando la carga computacional.

A continuación, mediante la figura 8.20, podemos ver un diagrama que nos ayuda a entender la arquitectura de la aplicación, teniendo en cuenta lo anteriormente descrito, tanto la funcionalidad de LLM como la de OCR están separadas de la básica de la aplicación:

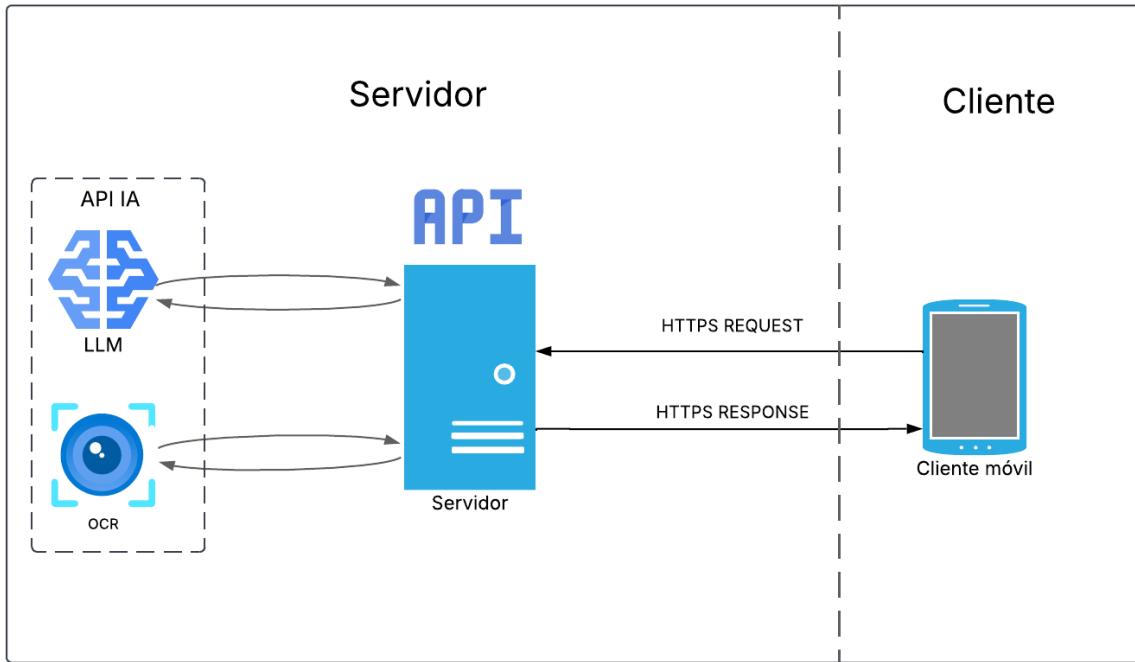


Figura 8.20: Arquitectura de la aplicación

8.3.1 Arquitectura del servidor

El servidor, dada la naturaleza del framework que usaremos para el desarrollo, se divide, principalmente, en tres grandes grupos, los cuales serán, modelos, esquemas, y rutas, sobre los cuales podríamos establecer una analogía al famoso patrón MVC (Modelo-Vista-Controlador)[33], ya que, los modelos se mantienen, los esquemas análogamente corresponderían a las vistas, y las rutas a los controladores.

Modelos

Los modelos, como su nombre indica, son los objetos que representan las entidades en el sistema, dichas entidades, como es de intuir, coinciden con las que almacenaremos en la base de datos. En estos modelos se programan también las relaciones que existen entre ellos, por ejemplo, un profesor imparte, o crea, una asignatura. Esto nos permite replicar la estructura de los datos de la base de datos en los objetos que se encuentran en la API. De esta manera, los modelos en nuestro sistema son:

- Usuario.
- Inscripción.



- Entrega.
- Asignatura.
- Actividad.

Esquemas

Un esquema es un concepto muy usado en la programación orientada al desarrollo mediante APIs, los esquemas son estructuras de datos que representan la información de nuestras entidades que usaremos para las peticiones, de manera que, en cada esquema de cada entidad, se programa un modelo base y a partir del mismo se van creando distintos esquemas, por ejemplo, para las respuestas, para la creación, o para la actualización de los datos. Además, en los mismos, se pueden añadir restricciones sobre los campos. Como anteriormente, se realiza un esquema por cada modelo, y por ende, de cada entidad

Rutas

Aquí se encuentra toda la lógica para las peticiones que se realicen. Se encuentran las operaciones CRUD sobre las entidades, realizando llamadas a la base de datos que permiten realizar operaciones sobre la misma. Cada operación se conoce como *endpoint*, que no es más que una ruta sobre la que se envían llamadas a través de cualquiera de las operaciones básicas a través del protocolo HTTP (GET, POST, PUT, DELETE, PATCH), dichas llamadas desencadenarán la lógica necesaria para completar las funcionalidades. De la misma manera, se creará una ruta para cada entidad sobre la que se opera, no obstante, habrá una ruta adicional llamada auth, que estará destinada a la autenticación del usuario. De esta manera, las rutas del sistema son:

- /usuarios
- /inscripciones
- /entregas
- /asignaturas
- /actividades
- /auth

Además de esta estructura, adicionalmente se añadirán tres componentes en implementación que se encargarán de:

1. Inicializar el servidor: En implementación corresponde al *main*.
2. Seguridad: Controla los *tokens* y el encriptado de las contraseñas.
3. Database: Controla la base de datos, la inicializa y proporciona la instancia necesaria para realizar las operaciones sobre la misma.



Es necesario mencionar que dentro de la estructura también se encontrarán los tests, ya se hablará en profundidad de los mismos en la sección de pruebas, pero como su nombre indica, en dicho módulo se encuentran los tests unitarios que se realizan para comprobar el correcto funcionamiento del servicio mediante los casos de prueba.

En la figura 8.21, apreciaremos visualmente como es el diseño de la parte del servidor.

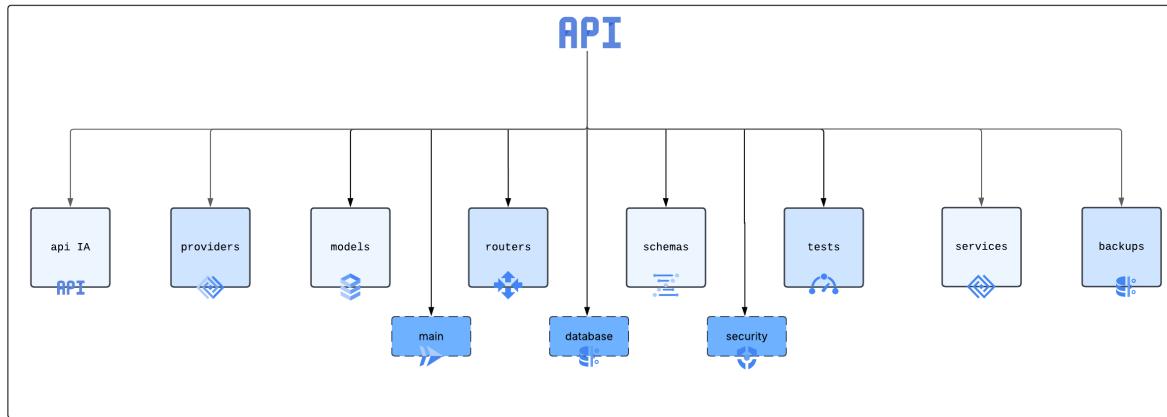


Figura 8.21: Esquema del backend

Por último a mencionar, como vemos en la figura 8.21, tenemos cuatro directorios o partes adicionales en la arquitectura. En primer lugar, providers y services, los cuales contienen la lógica necesaria para la funcionalidad de recuperar contraseña. También tenemos api IA y backups, las cuales serán explicadas a continuación.

Api IA - Funcionalidades de OCR y LLM

Como su nombre indica, esta parte está destinada a las funcionalidades de inteligencia artificial del presente proyecto. Lo cierto es que, no está en implementación pensado para que sea desplegado en el mismo servidor que la API básica que venimos explicando, no obstante, es posible, por lo que se deja formando parte de la arquitectura del servidor tal y como se comentó en secciones anteriores. Dentro de API IA, nos encontraremos un módulo para OCR y otro para LLM, ambos contendrán distintos modelos listos para ser usados en el proyecto. Gracias a esta capa de abstracción, será muy sencillo implementar nuevos modelos en nuestra aplicación.

Backups - Sistema de copias de seguridad

De nuevo, como podemos discernir en el título, esta parte está desarrollada para implementar un sistema de copias de seguridad robusto sobre la base de datos que en implementación toma lugar, para evitar perder todos los datos en situaciones anómalas y disponer de un plan de contingencias. Dentro de este módulo, encontraremos herramientas para, elaborar una copia de seguridad manualmente, planificarla mediante lo que conocemos como *cron*, y para restablecer la base de datos conforme a una copia de seguridad que tengamos hecha.



Clases de implementación

En este caso, en implementación las clases son las mismas que en el diagrama de clases que hemos visto anteriormente en la figura 8.19, pero añadiendo los routers y esquemas de respuestas. A continuación, en la figura 8.22, vemos un diagrama de clases para la implementación.

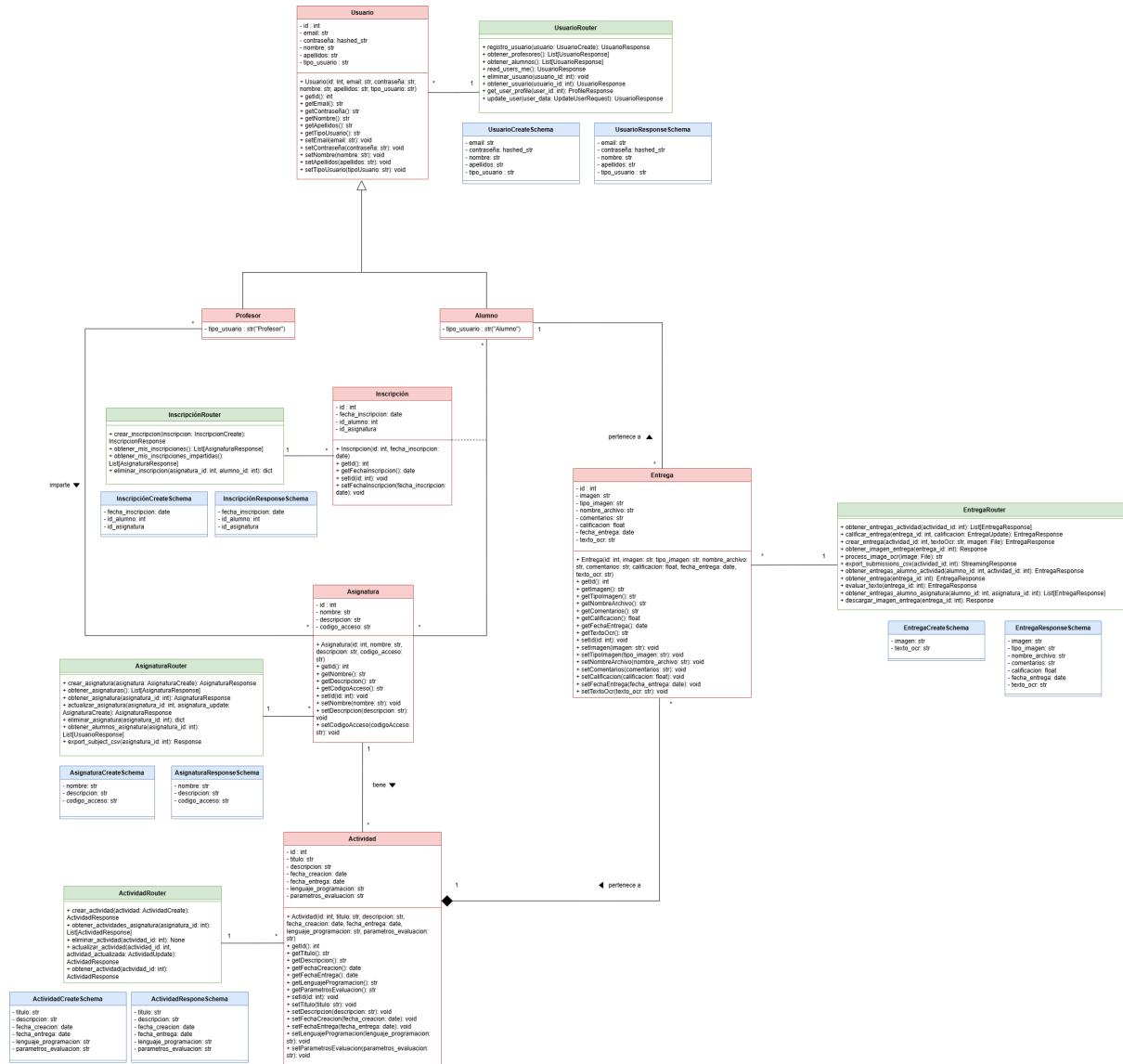


Figura 8.22: Clases implementación servidor

8.3.2 Arquitectura del cliente

La aplicación móvil se diseñará siguiendo principios de arquitectura limpia y modular, con el objetivo de garantizar la mantenibilidad, escalabilidad y testabilidad del código. Esta arquitectura se basa en los principios de la Arquitectura Limpia (Clean Architecture), la organización basada en características (Feature-First) y el uso del patrón Provider para la gestión del estado. [34]



8.3.3 Principios de Diseño

Arquitectura Limpia (Clean Architecture)

La Arquitectura Limpia se ha adoptado para separar las preocupaciones y dependencias del sistema en capas bien definidas.^[34] Estas capas son:

- **Capa de Presentación (Presentation Layer):** Responsable de la interfaz de usuario (UI) y de la gestión de los eventos del usuario. Esta capa interactúa con la capa de dominio a través de los casos de uso.
- **Capa de Dominio (Domain Layer):** Contiene la lógica de negocio y las entidades del dominio. Esta capa es independiente de los frameworks y tecnologías externas.
- **Capa de Datos (Data Layer):** Responsable de la obtención y persistencia de los datos. Esta capa interactúa con las fuentes de datos externas, como APIs y bases de datos locales.

Esta separación de capas permite que cada capa sea independiente y fácilmente reemplazable, lo que facilita el mantenimiento y la evolución del sistema.

Organización Basada en Características (Feature-First)

La aplicación se organiza en módulos funcionales, donde cada módulo representa una característica específica del sistema. Esta organización facilita la comprensión y el desarrollo de nuevas funcionalidades, ya que el código relacionado con una característica se encuentra agrupado en un mismo lugar. [35]

Gestión de Estado con Provider

El patrón Provider se utiliza para la gestión del estado de la aplicación. Provider permite la inyección de dependencias y la propagación del estado a través del árbol de widgets, lo que facilita la gestión del estado global y local de la aplicación. [36]

8.3.4 Estructura de Carpetas

La estructura de carpetas de la aplicación refleja los principios de diseño mencionados anteriormente:

```
lib/
+-- features/           # Módulos funcionales de la aplicación
|   +-- auth/            # Funcionalidad de autenticación
|   |   +-- data/
|   |   +-- domain/
|   |   +-- presentation/
|   +-- courses/          # Funcionalidad de cursos y actividades
|   |   +-- data/
|   |   +-- domain/
|   |   +-- presentation/
+-- core/                # Componentes compartidos y utilidades
|   +-- providers/        # Providers globales
```



```
|  +- services/      # Servicios compartidos
|  +- theme/        # Configuración de temas
|  +- config/       # Configuraciones generales
--- main.dart       # Punto de entrada de la aplicación
```

8.3.5 Implementación de la Arquitectura

Capa de Presentación

La capa de presentación se implementa utilizando widgets de Flutter y el patrón Provider. Los widgets se organizan en pantallas (pages) y componentes reutilizables (widgets). Los Providers se utilizan para gestionar el estado local de cada pantalla y para interactuar con la capa de dominio.

Capa de Dominio

La capa de dominio se implementa utilizando entidades y casos de uso. Las entidades representan los objetos del dominio, mientras que los casos de uso representan las operaciones que se pueden realizar sobre estos objetos. Esta capa es independiente de los frameworks y tecnologías externas.

Capa de Datos

La capa de datos se implementa utilizando repositorios y servicios de datos. Los repositorios abstraen el acceso a las fuentes de datos externas, mientras que los servicios de datos implementan la lógica de acceso a estas fuentes. Esta capa interactúa con APIs RESTful y bases de datos locales.

8.3.6 Ventajas de la Arquitectura Adoptada

La arquitectura adoptada para la aplicación móvil ofrece varias ventajas desde mi punto de vista:

- **Mantenibilidad:** La separación de preocupaciones y la modularidad facilitan la comprensión y el mantenimiento del código.
- **Escalabilidad:** La organización basada en características permite añadir nuevas funcionalidades sin afectar las existentes.
- **Testabilidad:** Las capas independientes facilitan la realización de pruebas unitarias y de integración.
- **Reutilización de código:** La modularidad y la abstracción de las capas permiten la reutilización de código en diferentes partes de la aplicación.
- **Independencia de frameworks:** La capa de dominio es independiente de los frameworks y tecnologías externas, lo que facilita la migración a otras tecnologías en el futuro.



8.3.7 Diagrama de la arquitectura

Mediante la figura 8.23, podemos observar el diagrama de la arquitectura en implementación.

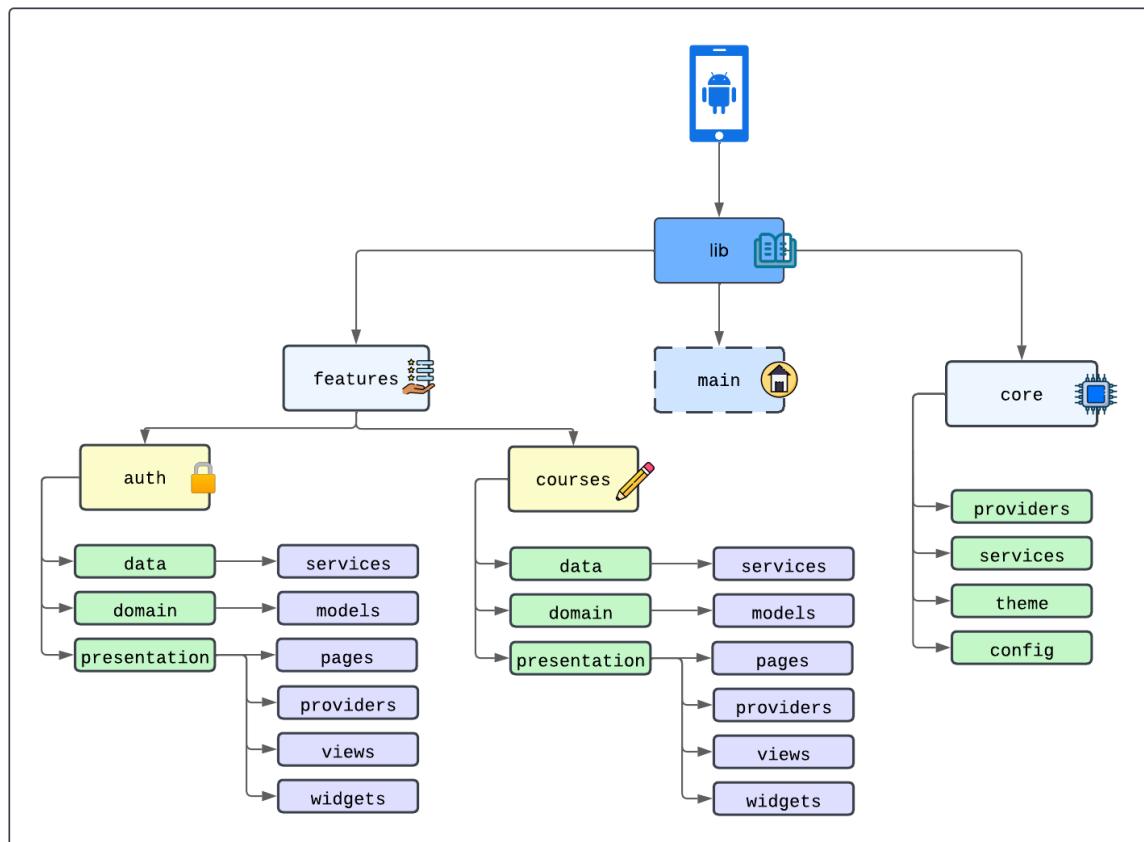


Figura 8.23: Diagrama arquitectura cliente

8.3.8 Clases de implementación

Las clases de implementación se pueden ver en la figura 8.24.



CAPÍTULO 8. ESPECIFICACIÓN DEL SISTEMA

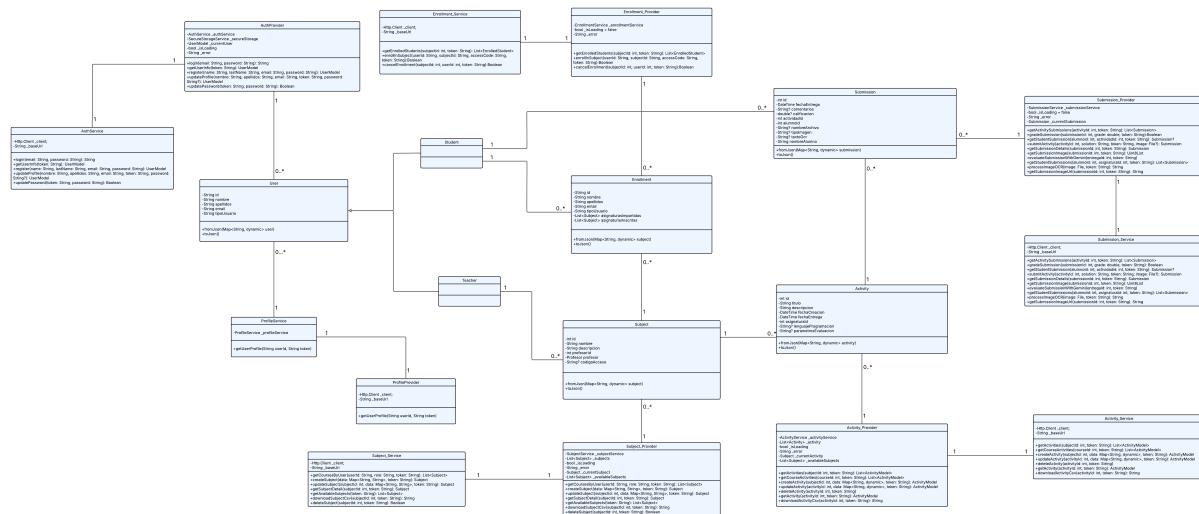


Figura 8.24: Clases implementación cliente

A continuación, con las siguientes tablas podemos ver más detalladamente las clases con sus atributos y responsabilidades.

Clase User

Descripción	
Atributos	
Atributo	Descripción
id: String	Identificador único del usuario.
nombre: String	Nombre del usuario.
apellidos: String	Apellidos del usuario.
email: String	Dirección de correo electrónico del usuario.
tipoUsuario: String	Rol del usuario en el sistema.
Métodos	
Método	Descripción
fromJson(Map<String, dynamic>user): User	Método estático para crear una instancia de 'User' desde un mapa (JSON).
toJson(): Map<String, dynamic>	Método para convertir la instancia de 'User' a un mapa (JSON).

Cuadro 8.48: Clase User



Clase Profile_Provider

Descripción	
Define los servicios para interactuar con la API de perfiles de usuario.	
Atributos	
Atributo	Descripción
_client: HttpClient	Cliente HTTP para realizar peticiones a la API.
_baseUrl: String	URL base de la API de perfiles de usuario.
Métodos	
Método	Descripción
getUserProfile(userId: String, token: String): UserProfileModel	Obtiene el perfil de un usuario desde la API.

Cuadro 8.49: Clase ProfileService

Clase Profile_Service

Descripción	
Gestiona la lógica de negocio relacionada con los perfiles de usuario, incluyendo la comunicación con el servicio de perfiles.	
Atributos	
Atributo	Descripción
_profileService: ProfileService	Instancia del servicio de perfiles para realizar operaciones relacionadas con los perfiles de usuario.
Métodos	
Método	Descripción
getUserProfile(userId: String, token: String): UserProfileModel	Obtiene el perfil de un usuario.

Cuadro 8.50: Clase ProfileProvider



Clase Auth_Provider

Descripción	
Atributos	
Atributo	Descripción
_authService: AuthService	Instancia del servicio de autenticación utilizado para realizar las operaciones de autenticación.
_secureStorageService: SecureStorageService	Instancia del servicio de almacenamiento seguro para gestionar tokens y otra información sensible.
_userModel_currentUser: UserModel?	Almacena el modelo del usuario autenticado actualmente (puede ser nulo si no hay usuario autenticado).
bool _isLoading	Indica si una operación de autenticación está en curso.
String _error	Almacena el mensaje de error en caso de fallas en la autenticación.
Métodos	
Método	Descripción
login(email: String, password: String): UserModel?	Realiza el inicio de sesión del usuario. Devuelve el modelo del usuario autenticado o nulo en caso de fallo.
getUserInfo(token: String): UserModel?	Obtiene la información del usuario utilizando un token. Devuelve el modelo del usuario o nulo en caso de fallo.
register(name: String, lastName: String, email: String, password: String): Future<UserModel?>	Registra un nuevo usuario en el sistema. Devuelve el modelo del usuario registrado o nulo en caso de fallo.
updateProfile(nombre: String, apellidos: String, email: String, token: String, password: String?): UserModel	Actualiza el perfil del usuario. Devuelve el modelo del usuario actualizado o nulo en caso de fallo. La contraseña es opcional.
updatePassword(token: String, password: String): Bool	Actualiza la contraseña del usuario. Devuelve ‘true’ si la actualización fue exitosa, ‘false’ en caso contrario.

Cuadro 8.51: Clase AuthProvider



Clase Auth_Service

Descripción	
Atributos	
Atributo	Descripción
_httpClient: HttpClient	Cliente HTTP utilizado para realizar las peticiones al servidor de autenticación.
_baseUrl: String	URL base del servidor de autenticación.
Métodos	
Método	Descripción
login(email: String, password: String): String	Envía una petición de inicio de sesión al servidor y devuelve el token de autenticación.
getUserInfo(token: String): UserModel	Envía una petición para obtener la información del usuario utilizando el token y devuelve el modelo del usuario.
register(name: String, lastName: String, email: String, password: String): UserModel	Envía una petición de registro de nuevo usuario al servidor y devuelve el modelo del usuario creado.
updateProfile(nombre: String, apellidos: String, email: String, token: String, password: String?): UserModel	Envía una petición para actualizar el perfil del usuario y devuelve el modelo del usuario actualizado. La contraseña es opcional.
updatePassword(token: String, password: String): Boolean	Envía una petición para actualizar la contraseña del usuario y devuelve ‘true’ si la actualización fue exitosa, ‘false’ en caso contrario.

Cuadro 8.52: Clase AuthService



Clase Subject

Descripción	
Define un curso o materia dentro de la plataforma.	
Atributos	
Atributo	Descripción
id: int	Identificador único de la asignatura.
nombre: String	Nombre o título de la asignatura.
descripcion: String	Descripción detallada del contenido de la asignatura.
profesorId: int	Identificador del usuario (profesor) responsable de la asignatura.
codigoAcceso: String?	Código único que los alumnos utilizan para inscribirse. Puede ser nulo.
profesor: Profesor	Objeto Profesor que representa al profesor de la asignatura.
Métodos	
Método	Descripción
fromJson(Map<String, dynamic>subject): Subject	Método estático para crear una instancia de 'Subject' desde un mapa (JSON).
toJson(): Map<String, dynamic>	Método para convertir la instancia de 'Subject' a un mapa (JSON).

Cuadro 8.53: Clase Asignatura



Clase Subject_Provider

Descripción	
Atributos	
Atributo	Descripción
_subjectService: SubjectService	Instancia del servicio de asignaturas para realizar operaciones relacionadas con las asignaturas.
subjects: List<Subject>	Lista de todas las asignaturas.
_isLoading: bool	Indica si se está realizando una operación (por ejemplo, cargar asignaturas).
_error: String	Almacena cualquier mensaje de error que ocurra durante las operaciones.
currentSubject: Subject	Asignatura actual.
availableSubjects: List<Subject>	Lista de asignaturas disponibles.
Métodos	
Método	Descripción
getCoursesByUser(userId: String, role: String, token: String): List<Subject>	Obtiene la lista de asignaturas para un usuario, filtrada por rol.
createSubject(data: Map<String, String>, token: String): Subject	Crea una nueva asignatura.
updateSubject(subjectId: int, data: Map<String, String>, token: String): Subject	Actualiza la información de una asignatura existente.
getSubjectDetail(subjectId: int, token: String): Subject	Obtiene los detalles de una asignatura específica.
getAvailableSubjects(token: String): List<Subject>	Obtiene la lista de asignaturas disponibles.
downloadSubjectCsv(subjectId: int, token: String): String	Descarga un archivo CSV con la información de la asignatura. El tipo de retorno ‘String’ podría representar la ruta del archivo o los datos en formato de cadena.
deleteSubject(subjectId: int, token: String): Boolean	Elimina una asignatura.

Cuadro 8.54: Clase SubjectProvider



Clase Subject_Service

Descripción	
Atributos	
Atributo	Descripción
_httpClient: HttpClient	Cliente HTTP para realizar peticiones a la API.
_baseUrl: String	URL base de la API de asignaturas.
Métodos	
Método	Descripción
getCoursesByUser(userId: String, role: String, token: String): List<Subject>	Obtiene la lista de asignaturas para un usuario, filtrada por rol.
createSubject(data: Map<String, String>, token: String): Subject	Crea una nueva asignatura en la API.
updateSubject(subjectId: int, data: Map<String, String>, token: String): Subject	Actualiza una asignatura existente en la API.
getSubjectDetail(subjectId: int, token: String): Subject	Obtiene los detalles de una asignatura desde la API.
getAvailableSubjects(token: String): List<Subject>	Obtiene la lista de asignaturas disponibles desde la API.
downloadSubjectCsv(subjectId: int, token: String): String	Descarga un archivo CSV de la API. El tipo de retorno ‘String’ podría representar la ruta del archivo o los datos en formato de cadena.
deleteSubject(subjectId: int, token: String): Boolean	Elimina una asignatura de la API.

Cuadro 8.55: Clase SubjectService



Clase Activity

Descripción	
Representa una tarea o evaluación que los alumnos deben completar dentro de una asignatura.	
Atributos	
Atributo	Descripción
id: int	Identificador único de la actividad.
titulo: String	Título de la actividad.
descripcion: String	Instrucciones o detalles sobre lo que el alumno debe hacer.
fechaCreacion: DateTime	Fecha en la que la actividad fue creada.
fechaEntrega: DateTime	Fecha límite para la entrega de la actividad.
asignaturaId: int	Identificador de la asignatura a la que pertenece la actividad.
lenguajeProgramacion: String?	Lenguaje de programación requerido para la actividad (opcional).
parametrosEvaluacion: String?	Criterios o rúbrica para evaluar la actividad (opcional).
Métodos	
Método	Descripción
fromJson(Map<String, dynamic>activity): Activity	Método estático para crear una instancia de ‘Activity’ desde un mapa (JSON).
toJson(): Map<String, dynamic>	Método para convertir la instancia de ‘Activity’ a un mapa (JSON).

Cuadro 8.56: Clase Actividad





Clase Activity_Provider

Descripción	
Gestiona la lógica de negocio relacionada con las actividades, incluyendo la comunicación con el servicio de actividades y el mantenimiento del estado.	
Atributos	
Atributo	Descripción
_activityService: ActivityService	Instancia del servicio de actividades para realizar operaciones relacionadas con las actividades.
_activity: List<Activity>	Lista de todas las actividades.
_isLoading: bool	Indica si se está realizando una operación (por ejemplo, cargar actividades).
_error: String	Almacena cualquier mensaje de error que ocurra durante las operaciones.
_currentActivity: Activity	Actividad actual.
_availableSubjects: List<Subject>	Lista de asignaturas disponibles.
Métodos	
Método	Descripción
getActivities(subjectId: int, token: String): List<ActivityModel>	Obtiene la lista de actividades para una asignatura.
getCourseActivities(courseId: int, token: String): List<ActivityModel>	Obtiene la lista de actividades para un curso.
createActivity(subjectId: int, data: Map<String, dynamic>, token: String): ActivityModel	Crea una nueva actividad.
updateActivity(activityId: int, data: Map<String, dynamic>, token: String): ActivityModel	Actualiza la información de una actividad existente.
deleteActivity(activityId: int, token: String)	Elimina una actividad.
getActivity(activityId: int, token: String): ActivityModel	Obtiene una actividad por su ID.
downloadActivityCsv(activityId: int, token: String): String	Descarga un archivo CSV con la información de la actividad. El tipo de retorno 'String' podría representar la ruta del archivo o los datos en formato de cadena.

Cuadro 8.57: Clase ActivityProvider



Clase Activity_Service

Descripción	
Define los servicios para interactuar con la API de actividades.	
Atributos	
Atributo	Descripción
_client: HttpClient	Cliente HTTP para realizar peticiones a la API.
_baseUrl: String	URL base de la API de actividades.
Métodos	
Método	Descripción
getActivities(subjectId: int, token: String): List<ActivityModel>	Obtiene la lista de actividades para una asignatura desde la API.
getCourseActivities(courseId: int, token: String): List<ActivityModel>	Obtiene la lista de actividades para un curso desde la API.
createActivity(subjectId: int, data: Map<String, dynamic>, token: String): ActivityModel	Crea una nueva actividad en la API.
updateActivity(activityId: int, data: Map<String, dynamic>, token: String): ActivityModel	Actualiza una actividad existente en la API.
deleteActivity(activityId: int, token: String)	Elimina una actividad de la API.
getActivity(activityId: int, token: String): ActivityModel	Obtiene una actividad por su ID desde la API.
downloadActivityCsv(activityId: int, token: String): String	Descarga un archivo CSV de la API. El tipo de retorno String podría representar la ruta del archivo o los datos en formato de cadena.
evaluateSubmissionWithGemini(entregaId: int, token: String):	Evalúa una entrega de actividad usando Gemini (no se especifica el tipo de retorno).
getStudentSubmissions(alumnoId: int, asignaturaId: int, token: String): List<Submission>	Obtiene las entregas de un alumno para una asignatura.
processImageOCR(image: File, token: String): String	Procesa una imagen usando OCR y devuelve el texto extraído.
getSubmissionImageUrl(submissionId: int, token: String): String	Obtiene la URL de una imagen de entrega.

Cuadro 8.58: Clase ActivityService



Clase Submission

Descripción	
Representa la entrega de una actividad por parte de un alumno.	
Atributos	
Atributo	Descripción
id: int	Identificador único de la entrega.
fechaEntrega: DateTime	Fecha en la que el alumno realizó la entrega.
comentarios: String?	Comentarios adicionales del alumno sobre la entrega (opcional).
calificacion: double?	Calificación asignada a la entrega (opcional).
actividadId: int	Identificador de la actividad a la que corresponde la entrega.
alumnoId: int	Identificador del alumno que realizó la entrega.
nombreArchivo: String?	Nombre del archivo de la entrega (opcional).
tipoImagen: String?	Tipo de archivo de la imagen de la entrega (opcional).
textoOcr: String?	Texto extraído de la imagen de la entrega mediante OCR (opcional).
nombreAlumno: String	Nombre del alumno que realizó la entrega.
Métodos	
Método	Descripción
fromJson(Map<String, dynamic>submission): Submission	Método estático para crear una instancia de ‘Submission’ desde un mapa (JSON).
toJson(): Map<String, dynamic>	Método para convertir la instancia de ‘Submission’ a un mapa (JSON).

Cuadro 8.59: Clase Submission

Clase Submission_Provider

Descripción	
Gestiona la lógica de negocio relacionada con las entregas, incluyendo la comunicación con el servicio de entregas y el mantenimiento del estado.	
Atributos	
Atributo	Descripción
_submissionService: SubmissionService	Instancia del servicio de entregas para realizar operaciones relacionadas con las entregas.
<i>isLoading</i> : bool	Indica si se está realizando una operación (por ejemplo, enviando una entrega).



(continuación)	
Elemento	Descripción
<i>error : String</i>	Almacena cualquier mensaje de error que ocurra durante las operaciones.
<i>currentSubmission : Submission</i>	Entrega actual.
Métodos	
Método	Descripción
getActivitySubmissions(activityId: int, token: String): List	Obtiene la lista de entregas para una actividad.
gradeSubmission(submissionId: int, grade: double, token: String)	Califica una entrega.
getStudentSubmission(alumnoId: int, actividadId: int, token: String): Submission?	Obtiene la entrega de un alumno para una actividad específica.
submitActivity(activityId: int, solution: String, token: String, image: File?): Submission	Permite a un alumno enviar una entrega para una actividad.
getSubmissionDetails(submissionId: int, token: String): Submission	Obtiene los detalles de una entrega específica.
getSubmissionImage(submissionId: int, token: String): Uint8List	Obtiene la imagen de una entrega.
evaluateSubmissionWithGemini(evaluacionId: int, token: String)	Evalúa una entrega usando Gemini.
getStudentSubmissions(alumnoId: int, asignaturaId: int, token: String): List	Obtiene las entregas de un alumno para una asignatura.
processImageOCR(image: File, token: String): String	Procesa una imagen usando OCR y devuelve el texto extraído.
getSubmissionImageUrl(submissionId: int, token: String): String	Obtiene la URL de una imagen de entrega.

Cuadro 8.60: Cuadro 8.59: Clase Submission_Provider





Clase Submission _ Service

Descripción	
Atributos	
Atributo	Descripción
_client: HttpClient	Cliente HTTP para realizar peticiones a la API.
_baseUrl: String	URL base de la API de entregas.
Métodos	
Método	Descripción
getActivitySubmissions(activityId: int, token: String): List<Submission>	Obtiene la lista de entregas para una actividad desde la API.
gradeSubmission(submissionId: int, grade: double, token: String): Boolean	Califica una entrega a través de la API.
getStudentSubmission(alumnoId: int, actividadId: int, token: String): Submission?	Obtiene la entrega de un alumno para una actividad desde la API.
submitActivity(activityId: int, solution: String, token: String, image: File?): Submission	Permite a un alumno enviar una entrega para una actividad a través de la API.
getSubmissionDetails(submissionId: int, token: String): Submission	Obtiene los detalles de una entrega desde la API.
getSubmissionImage(submissionId: int, token: String): Uint8List	Obtiene la imagen de una entrega desde la API.
evaluateSubmissionWithGemini(entregaId: int, token: String)	Evalúa una entrega usando Gemini a través de la API.
getStudentSubmissions(alumnoId: int, asignaturaId: int, token: String): List<Submission>	Obtiene las entregas de un alumno para una asignatura desde la API.
processImageOCR(image: File, token: String): String	Procesa una imagen usando OCR y devuelve el texto extraído desde la API.
getSubmissionImageUrl(submissionId: int, token: String): String	Obtiene la URL de una imagen de entrega desde la API.

Cuadro 8.61: Clase SubmissionService



Clase Enrollment

Descripción	
Representa la relación entre un alumno y una asignatura, indicando que el alumno está inscrito en dicha asignatura.	
Atributos	
Atributo	Descripción
id: int	Identificador único de la inscripción.
alumnoId: int	Identificador del alumno inscrito.
asignaturaId: int	Identificador de la asignatura en la que el alumno está inscrito.
Métodos	
Método	Descripción
inscribir(alumnoId, asignaturaId, código)	Permite a un alumno inscribirse en una asignatura.
obtenerListaPorAlumno(alumnoId)	Recupera la lista de asignaturas en las que un alumno está inscrito.
obtenerListaPorAsignatura(asignaturaId)	Obtiene la lista de alumnos inscritos en una asignatura.
eliminar()	Permite a un alumno cancelar su inscripción en una asignatura.

Cuadro 8.62: Clase Inscripcion



Clase Enrollment _ Provider

Descripción	
Atributos	
Atributo	Descripción
_enrollmentService: EnrollmentService	Instancia del servicio de inscripciones para realizar operaciones relacionadas con las inscripciones.
_isLoading: bool	Indica si se está realizando una operación (por ejemplo, inscribiendo a un alumno).
_error: String	Almacena cualquier mensaje de error que ocurra durante las operaciones.
Métodos	
Método	Descripción
getEnrolledStudents(subjectId: int, token: String): List<EnrolledStudent>	Obtiene la lista de alumnos inscritos en una asignatura.
enrollInSubject(userId: String, subjectId: String, accessCode: String, token: String): Boolean	Inscribe a un alumno en una asignatura.
cancelEnrollment(subjectId: int, userId: int, token: String) :Boolean	Permite a un alumno cancelar su inscripción en una asignatura.

Cuadro 8.63: Clase EnrollmentProvider



Clase Enrollment _ Service

Descripción	
Define los servicios para interactuar con la API de inscripciones.	
Atributos	
Atributo	Descripción
_client: HttpClient	Cliente HTTP para realizar peticiones a la API.
_baseUrl: String	URL base de la API de inscripciones.
Métodos	
Método	Descripción
getEnrolledStudents(subjectId: int, token: String): List<EnrolledStudent>	Obtiene la lista de alumnos inscritos en una asignatura desde la API.
enrollInSubject(userId: String, subjectId: String, accessCode: String, token: String): Boolean	Inscribe a un alumno en una asignatura a través de la API.
cancelEnrollment(subjectId: int, userId: int, token: String) :Boolean	Cancela la inscripción de un alumno en una asignatura a través de la API.

Cuadro 8.64: Clase EnrollmentService

8.4 Resumen de implementación

En esta sección, se pretende resumir la implementación de toda la aplicación, ya que, durante las secciones anteriores hemos dado pinceladas de cómo se ha implementado pero, veo conveniente sintetizar en una sola, a modo de resumen, cómo se implementará la solución.

Bien, como hemos visto, la arquitectura sigue el modelo cliente-servidor, en donde el cliente simboliza la aplicación móvil, la cual internamente estará implementada siguiendo una arquitectura dividida en tres capas siguiendo principios de arquitectura limpia, con una organización basada en características y gestionando el estado de la aplicación mediante Provider. Es aquí donde, dada la arquitectura cliente-servidor, se implementará todo lo necesario para que todas las funcionalidades se realicen contra el servidor, desde la autenticación hasta la entrega de actividades pasando por todas las funcionalidades, como la recuperación de contraseñas o la gestión de los alumnos.

El servidor, por otro lado, concentrará toda la funcionalidad principal. Cuando hablamos del servidor, tenemos que entenderlo como el todo que encierra dichas funcionalidades, pero que, si lo desmembramos, nos daremos cuenta de que está compuesto por distintos módulos que incluso no tienen por qué ser implementados en la misma máquina servidor. De hecho, como ya hemos explicado, la parte de inteligencia artificial está diseñada para funcionar por independiente y aliviar la carga en el servidor donde esté alojada



la funcionalidad principal del sistema, permitiendo la escalabilidad y separación de responsabilidades. Asimismo, la base de datos también podría ser alojada en otro servidor distinto, y las APIS externas, representan otras funcionalidades de IA necesarias para la aplicación pero que no son alojadas y controladas por nuestro proyecto, si no que son servicios externos. En la sección 9.3, en la cual se comentarán las pruebas hechas para el uso de inteligencia artificial en la aplicación, se detallarán estos servicios externos usados.

Mediante la figura 8.25, podemos ver más en detalle la arquitectura implementación de toda la aplicación mediante un diagrama de cajas.

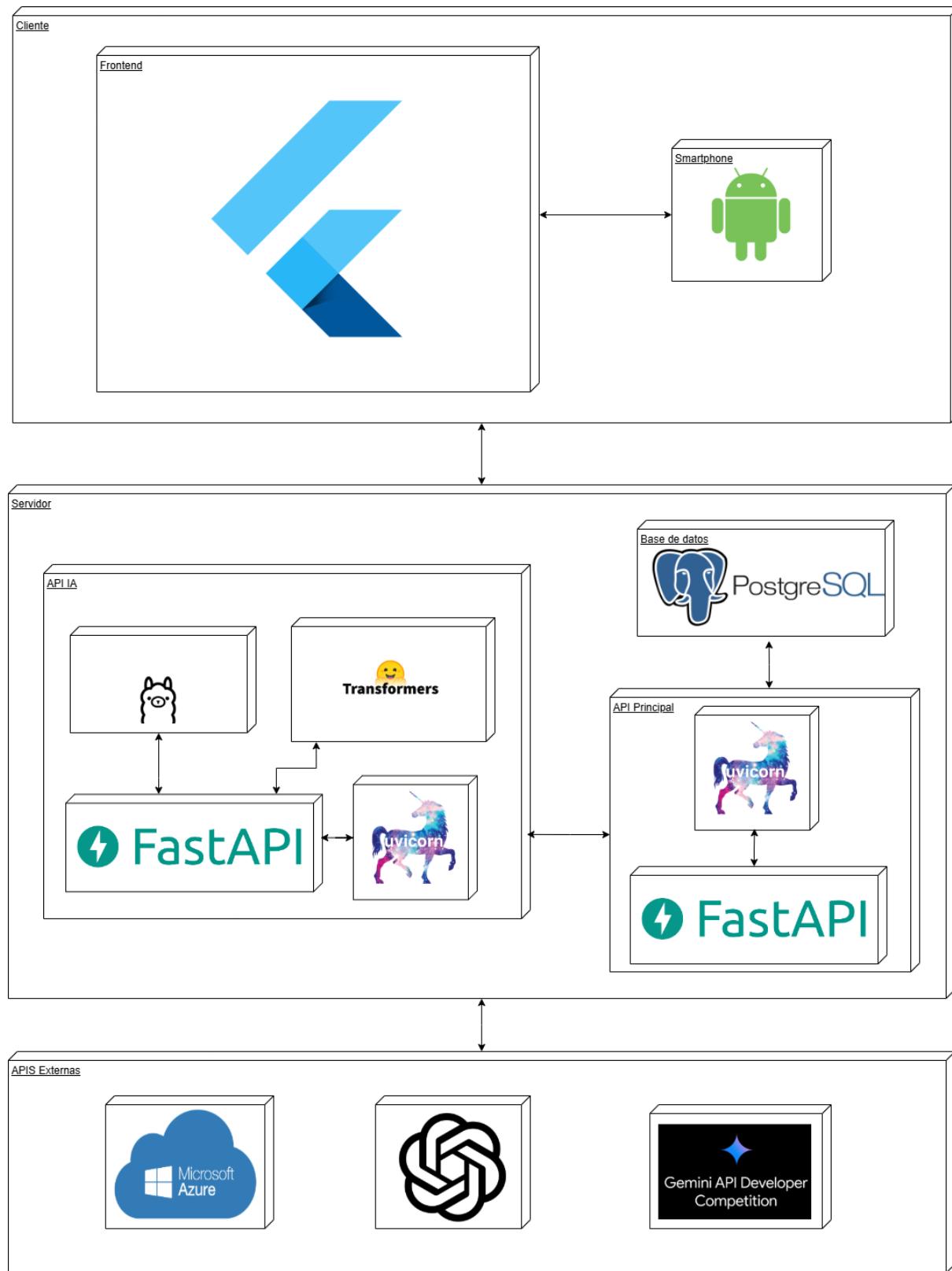


Figura 8.25: Diagrama de bloques implementación

8.4.1 Tecnologías usadas

En esta sección se explicarán algunas de las tecnologías que hemos visto en el diagrama anterior que no hemos explicado anteriormente a lo largo del documento, para que quede



claro cual es su labor en el presente proyecto. Dichas tecnologías a explicar serán:

- Ollama
- Transformers
- Uvicorn

El resto, no serán explicadas en la presente sección, puesto que ya ha sido explicadas en secciones anteriores, en antecedentes 4 y en factores estratégicos 5.2.

Ollama

Ollama es una herramienta que permite ejecutar modelos de lenguaje de gran tamaño (LLMs) de forma local, de manera eficiente y con un consumo optimizado de recursos. En el contexto del presente proyecto, Ollama ha sido utilizada para desplegar modelos de inteligencia artificial basados en Transformers, permitiendo así procesar lenguaje natural sin necesidad de depender de servicios externos o de la nube. [37]

El uso de Ollama facilita la integración de capacidades de inteligencia artificial dentro de la arquitectura del sistema, ofreciendo una solución modular y desacoplada del servidor principal. Esto permite distribuir la carga computacional, ya que Ollama puede ser ejecutado en una máquina distinta, lo cual mejora la escalabilidad y el rendimiento general de la aplicación.

Además, su compatibilidad con diferentes modelos y su interfaz sencilla para la ejecución y consumo de inferencias lo convierten en una alternativa flexible y potente para el desarrollo de funcionalidades avanzadas de procesamiento de lenguaje natural.

Transformers

Transformers es una biblioteca desarrollada por Hugging Face que proporciona una amplia colección de modelos de aprendizaje profundo preentrenados para tareas de procesamiento de lenguaje natural (PLN), como clasificación de texto, generación de texto, traducción automática, análisis de sentimientos, entre otros. [38]

En el presente proyecto, Transformers se ha utilizado como la base tecnológica para implementar el módulo de inteligencia artificial, sobre todo en la parte del OCR, el cual se encarga de procesar y analizar el contenido de una imagen. Gracias a esta biblioteca, ha sido posible integrar modelos permitiendo ofrecer respuestas contextuales, análisis automáticos y generación de contenido.

Uvicorn

Uvicorn es un servidor web de alto rendimiento basado en ASGI (Asynchronous Server Gateway Interface), diseñado para aplicaciones web asíncronas escritas en Python. Su principal objetivo es permitir el manejo eficiente de múltiples conexiones simultáneas, lo cual es esencial en sistemas modernos que requieren alta concurrencia y bajo tiempo de respuesta. [39]

En el contexto de este proyecto, Uvicorn actúa como el servidor que ejecuta la aplicación desarrollada con FastAPI. Gracias a su arquitectura basada en eventos, permite aprovechar al máximo las capacidades asíncronas de FastAPI, lo que se traduce en una mejor gestión de las peticiones HTTP, menor consumo de recursos y una mayor escalabilidad.



8.5 Modelado de la interfaz

Esta sección va completamente dedicada al diseño de la interfaz del usuario. Esto en el flujo del proyecto es muy importante, ya que aquí definiremos el prototipado mediante maquetas que conforman el aspecto visual de nuestra aplicación. Particularmente en este caso usaremos los llamados *mockups*. Los mockups tratan de la parte visual del proyecto y debe contener amplitud de detalles como los colores, tipografía, contenido, etc. Es el prototipo que mejor representa el producto final, no obstante, por definición, no incluirían interacciones finales, pero en este caso, en las figuras se darán algunas indicaciones de qué realizan ciertos botones.

8.5.1 Colores

La paleta de colores de la aplicación ha sido cuidadosamente seleccionada para proporcionar una experiencia visual agradable y coherente, tanto en modo claro como oscuro. El color principal de la aplicación es un azul profundo (0xFF1565C0), inspirado en el Material Design, que transmite profesionalidad y confianza. En la figura 8.26 podemos ver el color al que nos referimos, concretamente, el segundo empezando desde abajo.

Colors.blue[50]	0xFFE3F2FD
Colors.blue.shade50	
Colors.blue[100]	0xFFBBDEFB
Colors.blue.shade100	
Colors.blue[200]	0xFF90CAF9
Colors.blue.shade200	
Colors.blue[300]	0xFF64B5F6
Colors.blue.shade300	
Colors.blue[400]	0xFF42A5F5
Colors.blue.shade400	
Colors.blue	0xFF2196F3
Colors.blue[600]	0xFF1E88E5
Colors.blue.shade600	
Colors.blue[700]	0xFF1976D2
Colors.blue.shade700	
Colors.blue[800]	0xFF1565C0
Colors.blue.shade800	
Colors.blue[900]	0xFF0D47A1
Colors.blue.shade900	

Figura 8.26: Paleta de azules de Flutter

Esquema de Colores

La aplicación utiliza el esquema de colores dinámico de Material 3 [40], que se genera a partir del color principal (`seedColor`). Este esquema garantiza la armonía visual entre los diferentes componentes de la interfaz, adaptándose automáticamente a los modos claro y oscuro.



- **Color Principal (seedColor):** 0xFF1565C0 (Azul profundo). Utilizado para botones principales, barras de aplicación y otros elementos destacados.
- **Esquema de Colores Claro:** Generado automáticamente a partir del `seedColor` para el modo claro. Incluye variaciones de azul, grises y colores de acento para proporcionar contraste y claridad.
- **Esquema de Colores Oscuro:** Generado automáticamente a partir del `seedColor` para el modo oscuro. Incluye variaciones de azul, grises oscuros y colores de acento para proporcionar contraste y legibilidad en entornos oscuros.

Uso de Colores

Los colores se utilizan de manera estratégica para guiar la atención del usuario y proporcionar retroalimentación visual. Por ejemplo:

- Los botones principales utilizan el color principal para destacar las acciones importantes.
- Los colores de acento se utilizan para resaltar elementos interactivos y proporcionar retroalimentación visual.
- Los colores de fondo y texto se adaptan automáticamente a los modos claro y oscuro para garantizar la legibilidad y el confort visual.

Logotipo de la aplicación

Para el logotipo de la aplicación hemos decidido confeccionar uno que use la paleta de colores propuesta anteriormente, para mantener una coherencia que sea visualmente atractiva, además, se ha optado por un logotipo minimalista y distinto, para que sea fácilmente identificable. En la figura 8.27 podemos apreciar el logotipo.

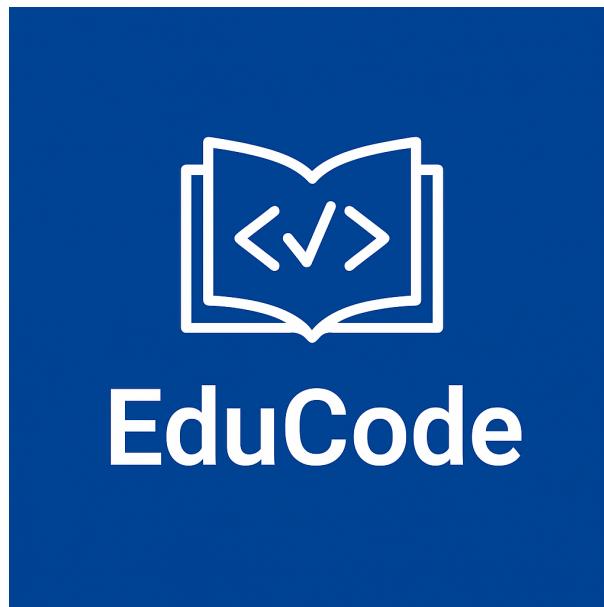


Figura 8.27: Logotipo de la aplicación



8.5.2 Tipografía

La tipografía de la aplicación ha sido seleccionada para garantizar la legibilidad y la coherencia visual en todos los componentes de la interfaz.

Fuente Principal

La aplicación utiliza la fuente por defecto de Material 3, que es Roboto en Android. Esta fuente ha sido elegida por su legibilidad y su adaptabilidad a diferentes tamaños y densidades de pantalla. A continuación, en la figura 8.28 podemos ver la fuente seleccionada.

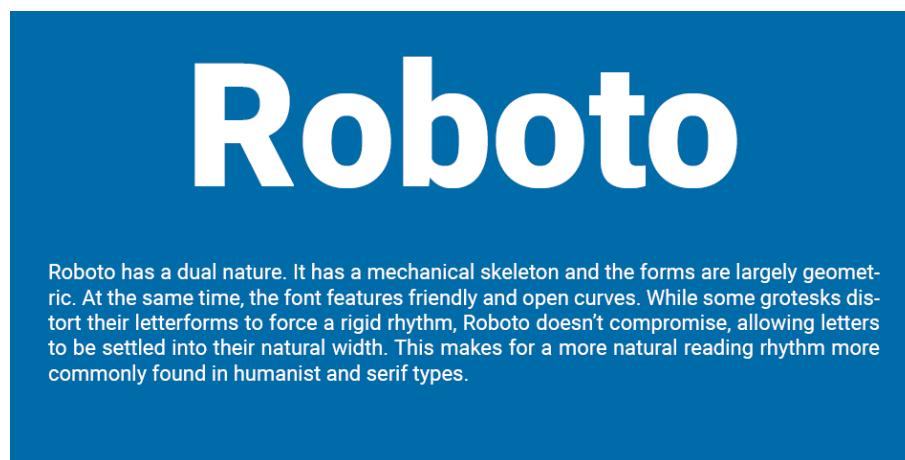


Figura 8.28: Fuente Roboto

Estilos de Texto

La aplicación utiliza los estilos de texto predefinidos de Material 3 para garantizar la coherencia visual en todos los componentes de la interfaz. Estos estilos incluyen:

- **Display:** Utilizado para títulos grandes y elementos destacados.
- **Headline:** Utilizado para títulos de secciones y subtítulos.
- **Title:** Utilizado para títulos de widgets y diálogos.
- **Body:** Utilizado para el cuerpo de texto y la información general.
- **Label:** Utilizado para etiquetas de campos de formulario y botones pequeños.

Uso de Tipografía

La tipografía se utiliza de manera estratégica para jerarquizar la información y guiar la atención del usuario. Por ejemplo:

- Los títulos grandes utilizan el estilo **Display** para destacar la información importante.
- El cuerpo de texto utiliza el estilo **Body** para garantizar la legibilidad en párrafos largos.



- Las etiquetas de campos de formulario utilizan el estilo **Label** para proporcionar información concisa y clara.

8.5.3 Diagrama de navegación

Antes de mostrar los mockups de la aplicación, realizaremos un diagrama de navegación para adquirir una perspectiva más general de cómo navegaremos a través de las distintas pantallas que ofrece la aplicación. Mediante la figura 8.29, lo veremos visualmente, además dispone de una leyenda que nos indica qué usuarios puede visitar las páginas de la aplicación.

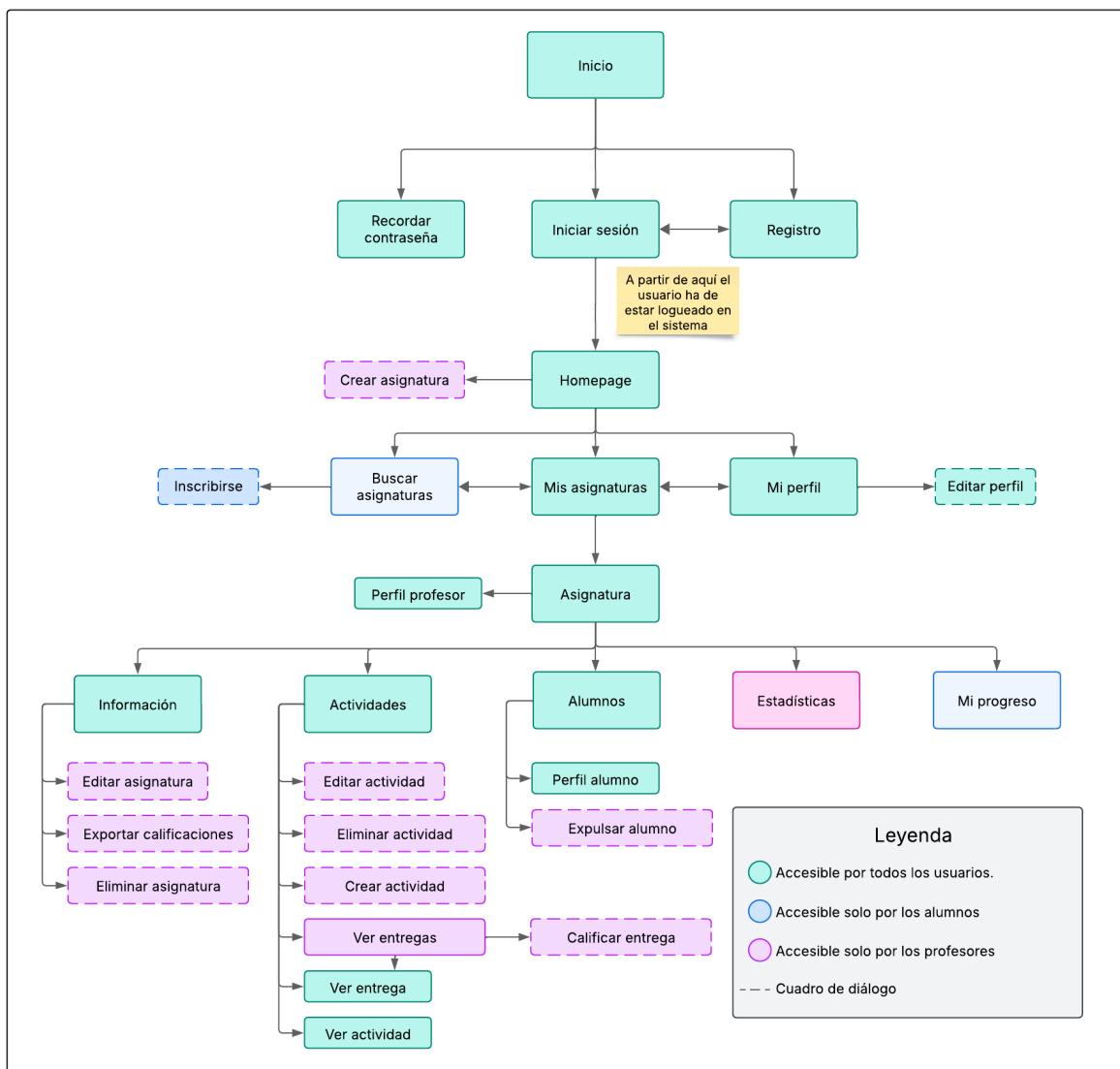


Figura 8.29: Diagrama de navegación



8.5.4 Mockups

Una vez hemos visto el diagrama de navegación para entender las distintas interfaces de las que dispondremos, podemos proceder a mostrar los mockups.

Iniciar sesión

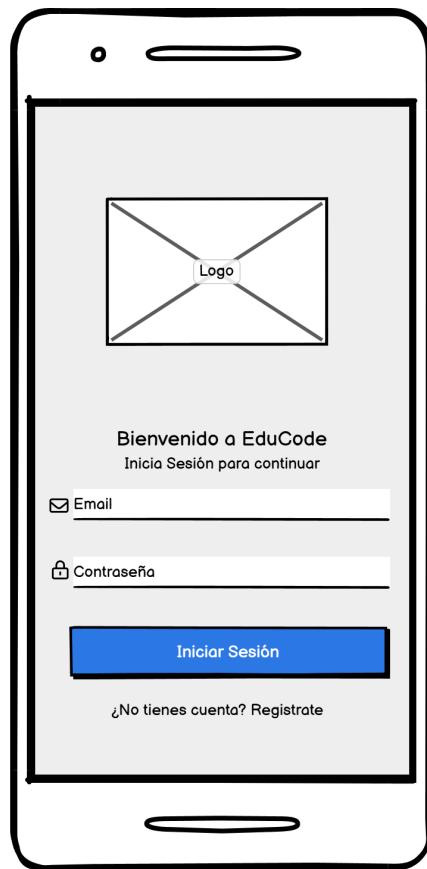


Figura 8.30: Mockup iniciar sesión



Recuperar contraseña



Figura 8.31: Mockup recuperar contraseña



Registro



Figura 8.32: Mockup registro



Mis asignaturas - Alumno

Mobile apps

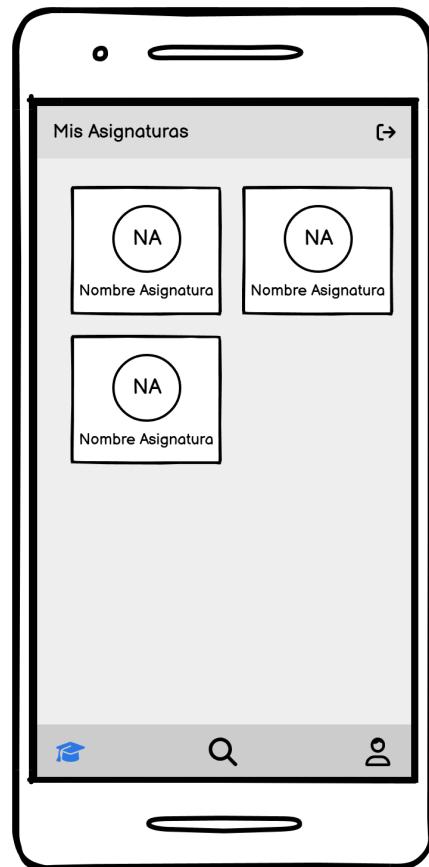


Figura 8.33: Mockup mis asignaturas alumno



Mis asignaturas - Profesor

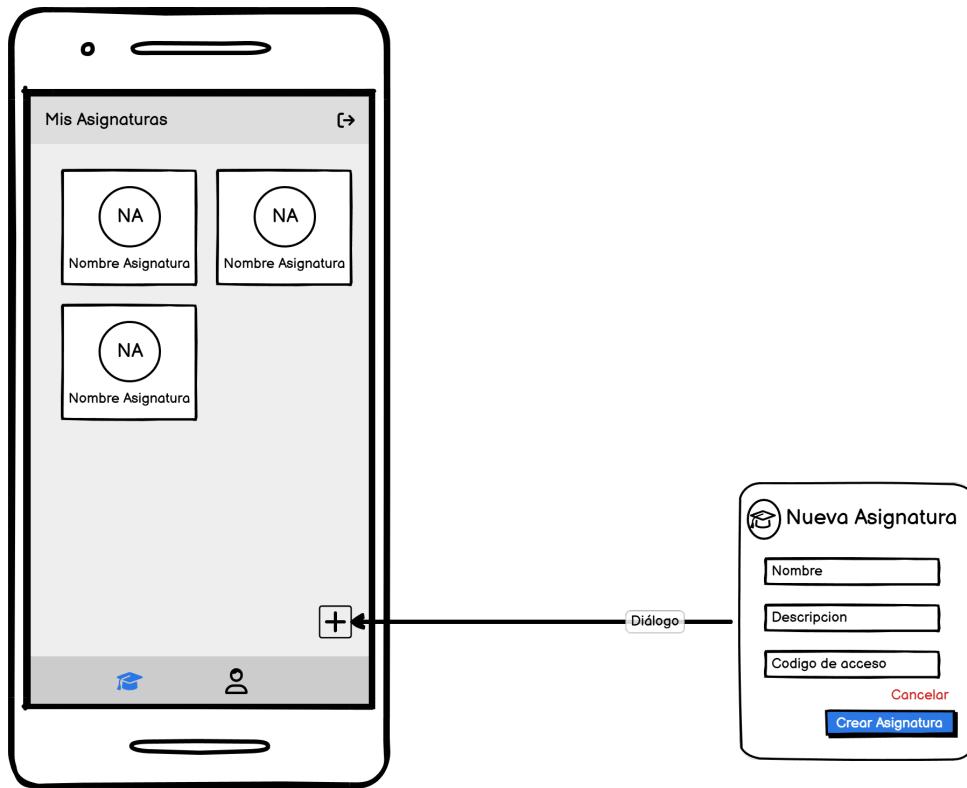


Figura 8.34: Mockup mis asignaturas profesor



Buscar asignaturas

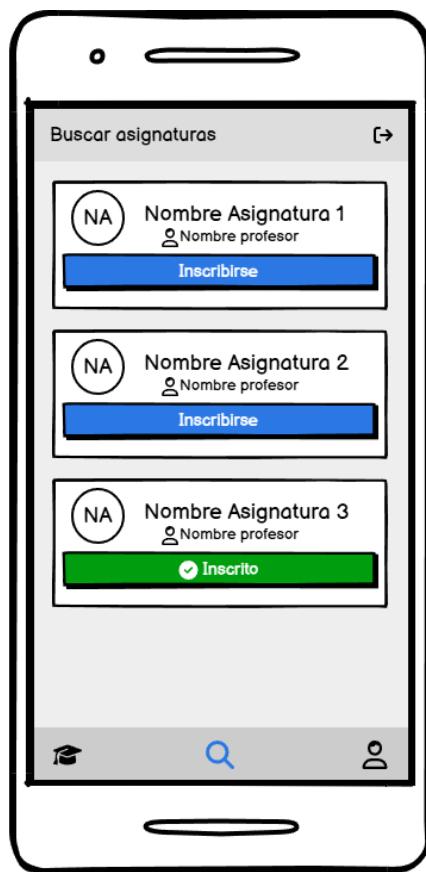


Figura 8.35: Mockup buscar asignaturas



Mi perfil

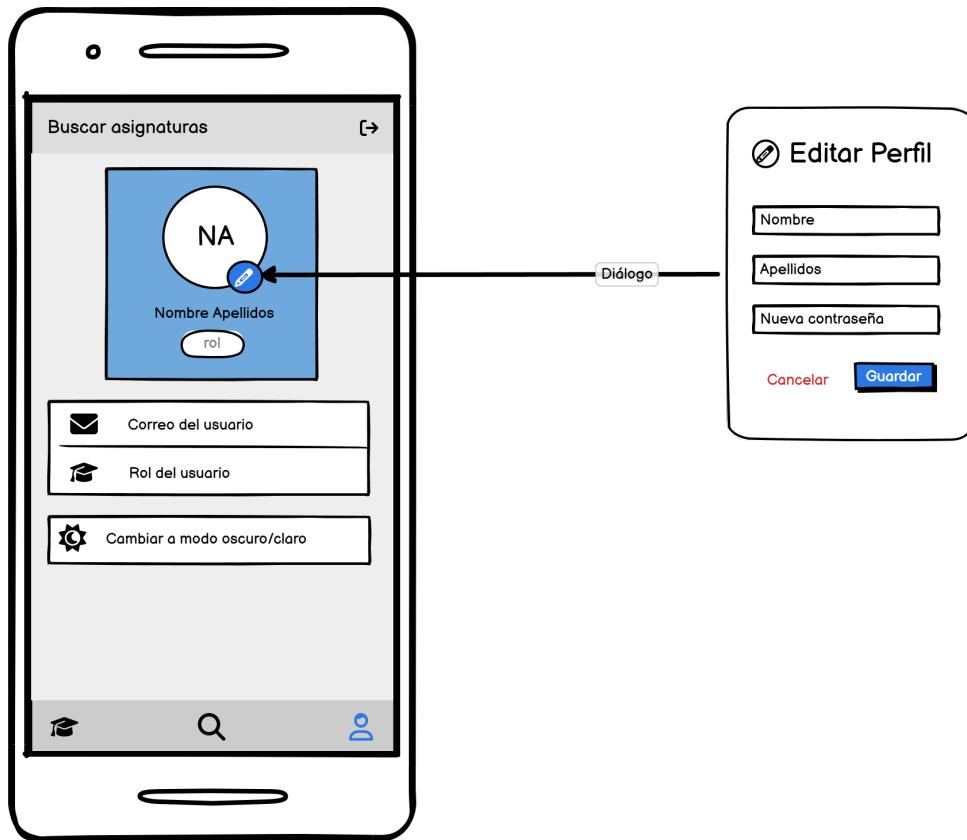


Figura 8.36: Mockup mi perfil



Información asignatura - Alumno

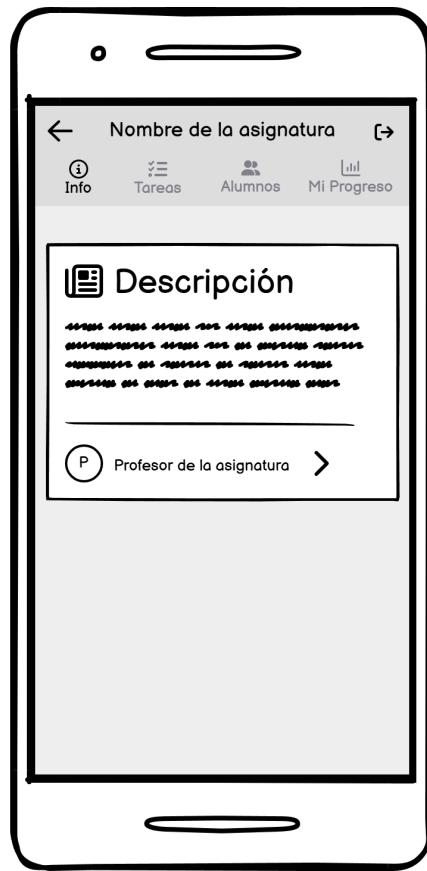


Figura 8.37: Mockup información asignatura alumno



Información asignatura - Profesor

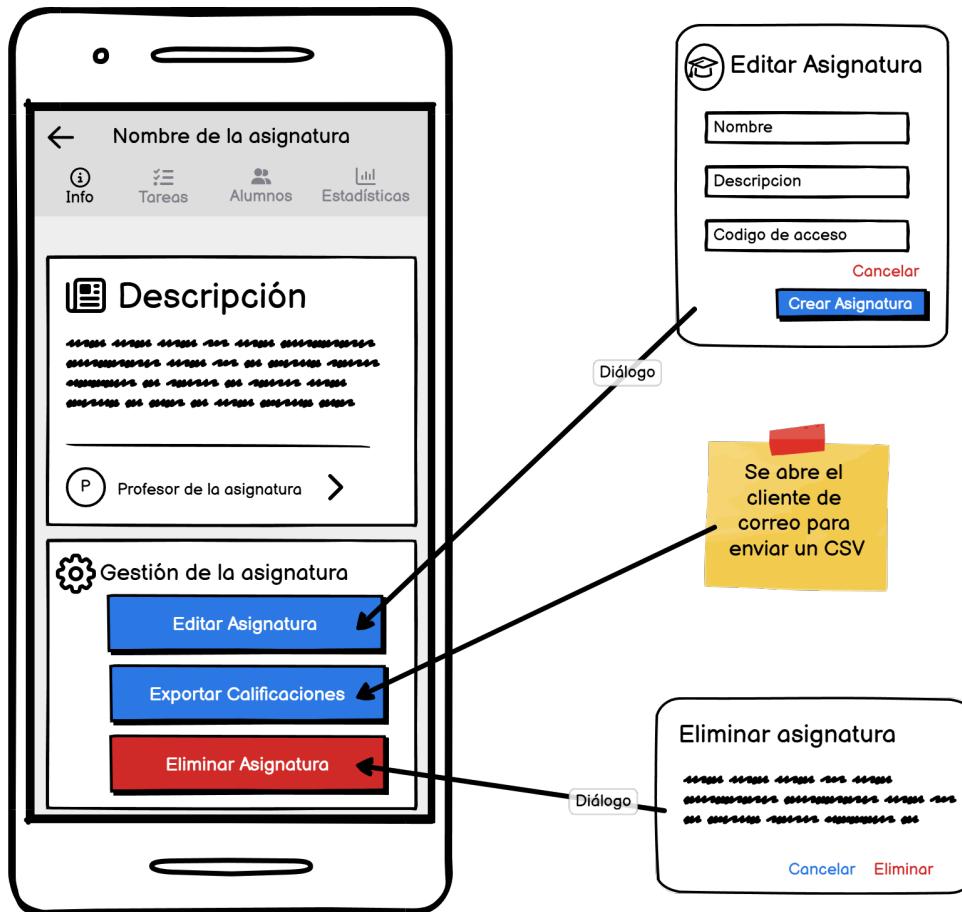


Figura 8.38: Mockup información asignatura profesor



Actividades - Alumno

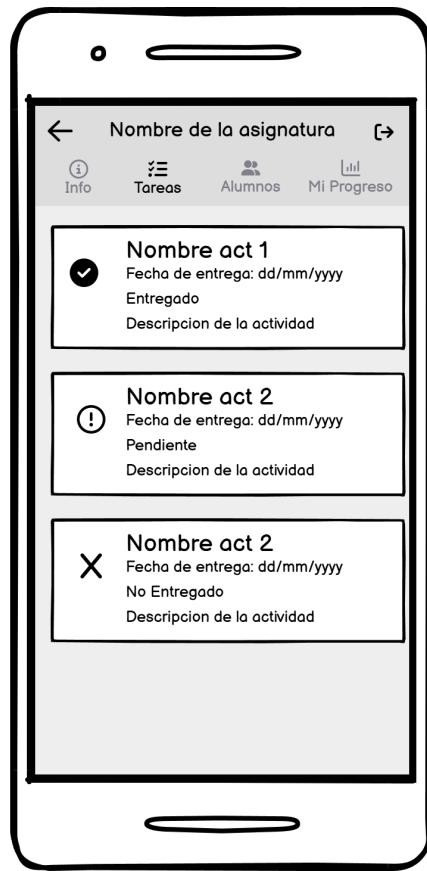


Figura 8.39: Mockup actividades alumno



Actividades - Profesor

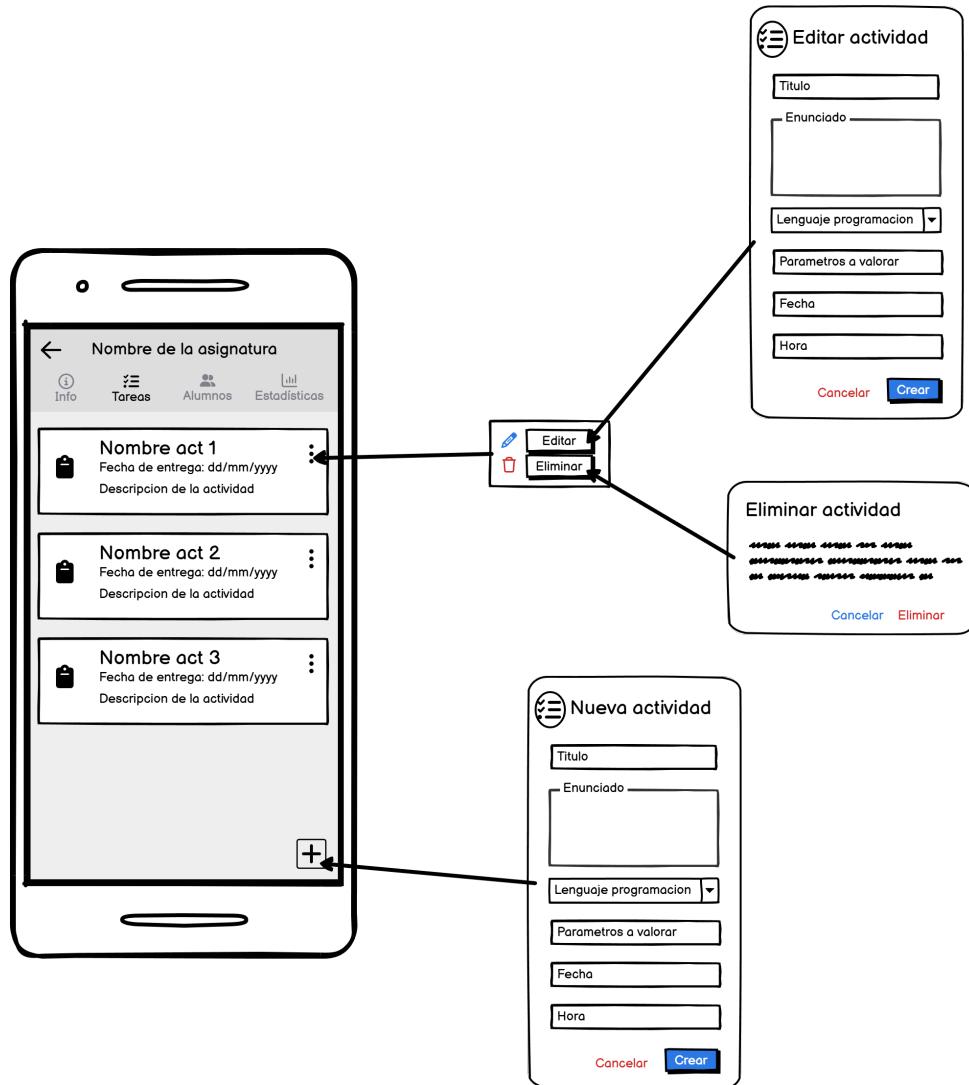


Figura 8.40: Mockup actividades profesor



Ver entrega

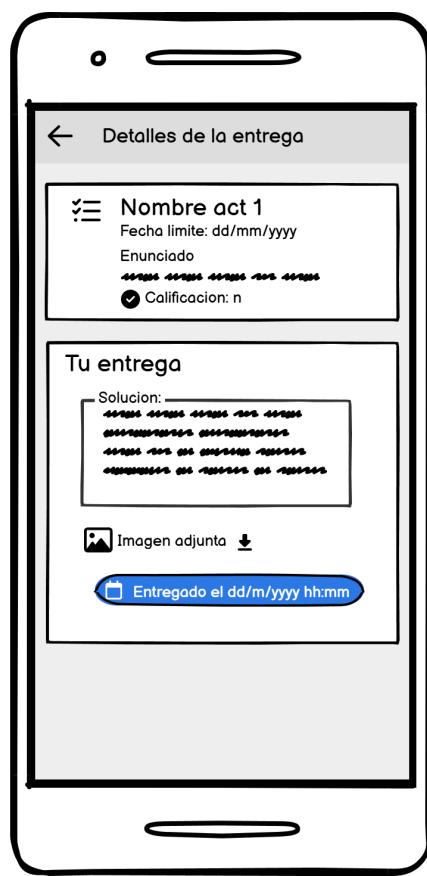


Figura 8.41: Mockup ver entrega



Ver actividad (para hacer la entrega)

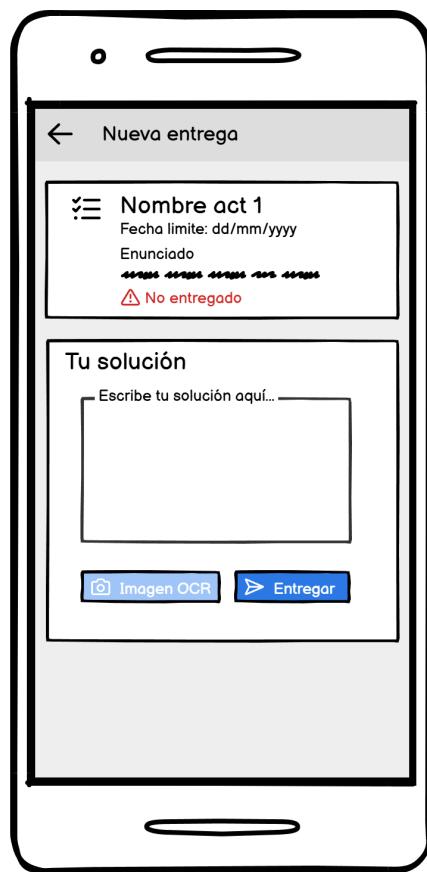


Figura 8.42: Mockup veer actividad para hacer la entrega



Lista de alumnos - Alumno

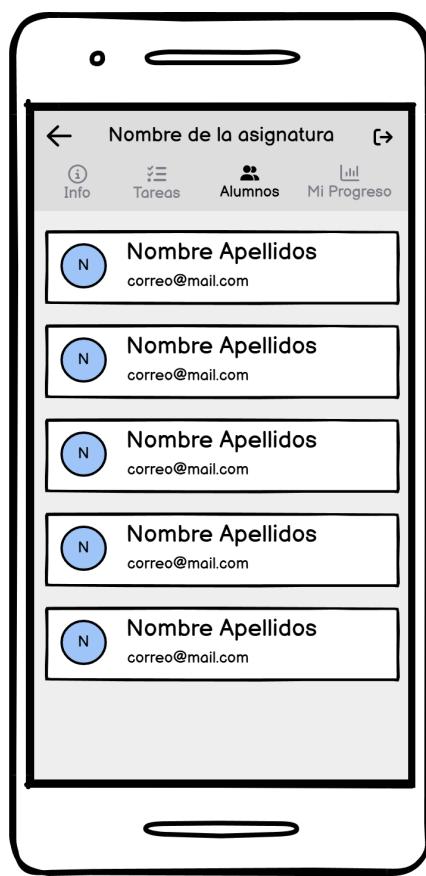


Figura 8.43: Mockup lista de alumnos alumno



Lista de alumnos - Profesor

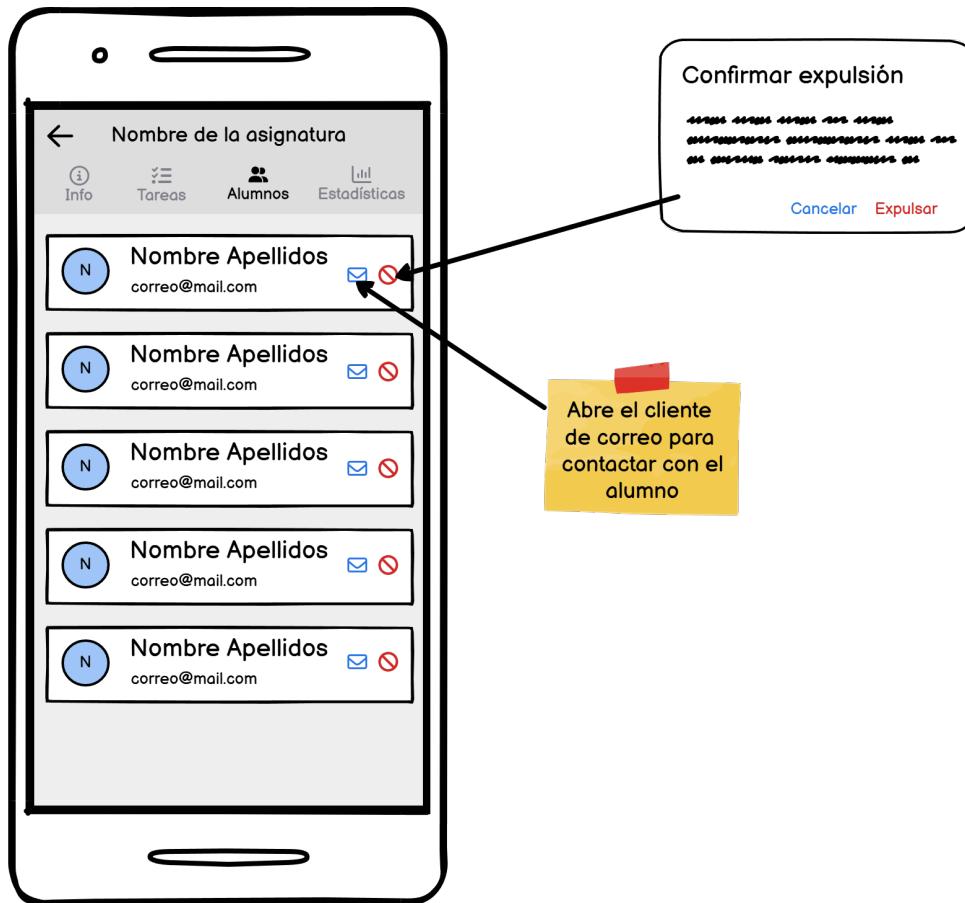


Figura 8.44: Mockup lista de alumnos profesor



Perfil alumno

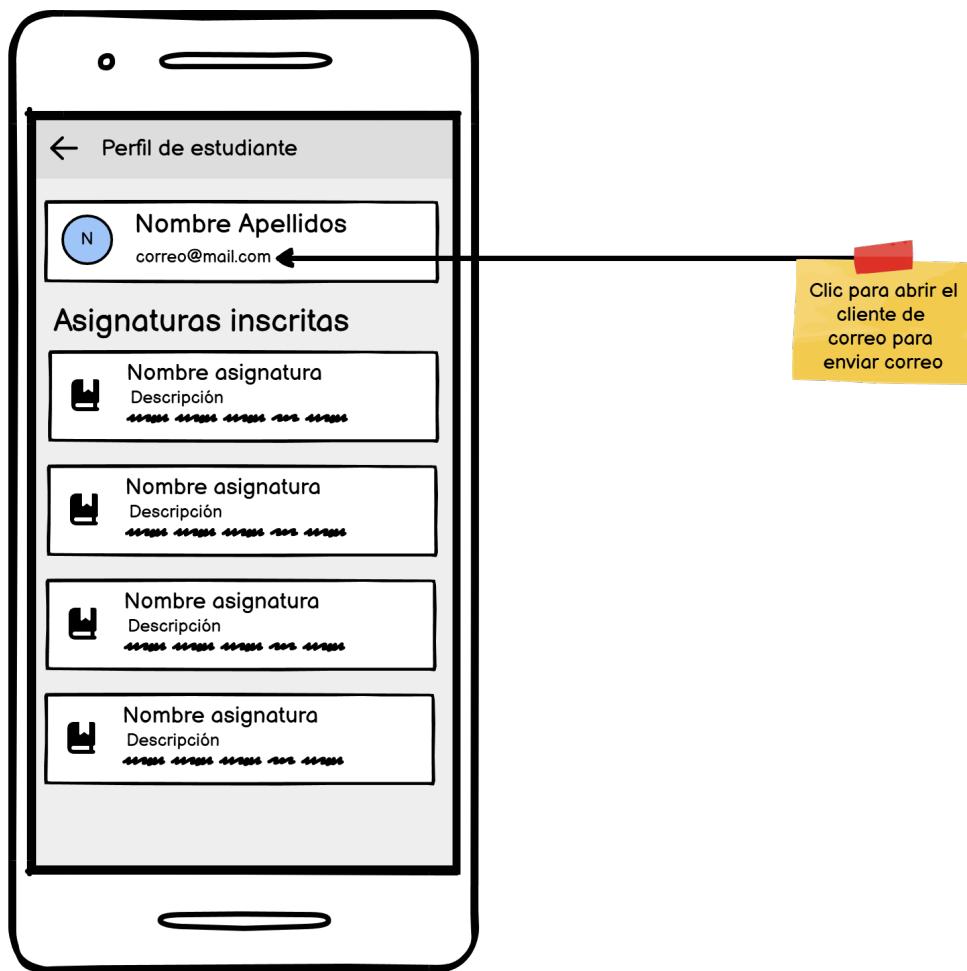


Figura 8.45: Mockup perfil de alumno



Perfil profesor

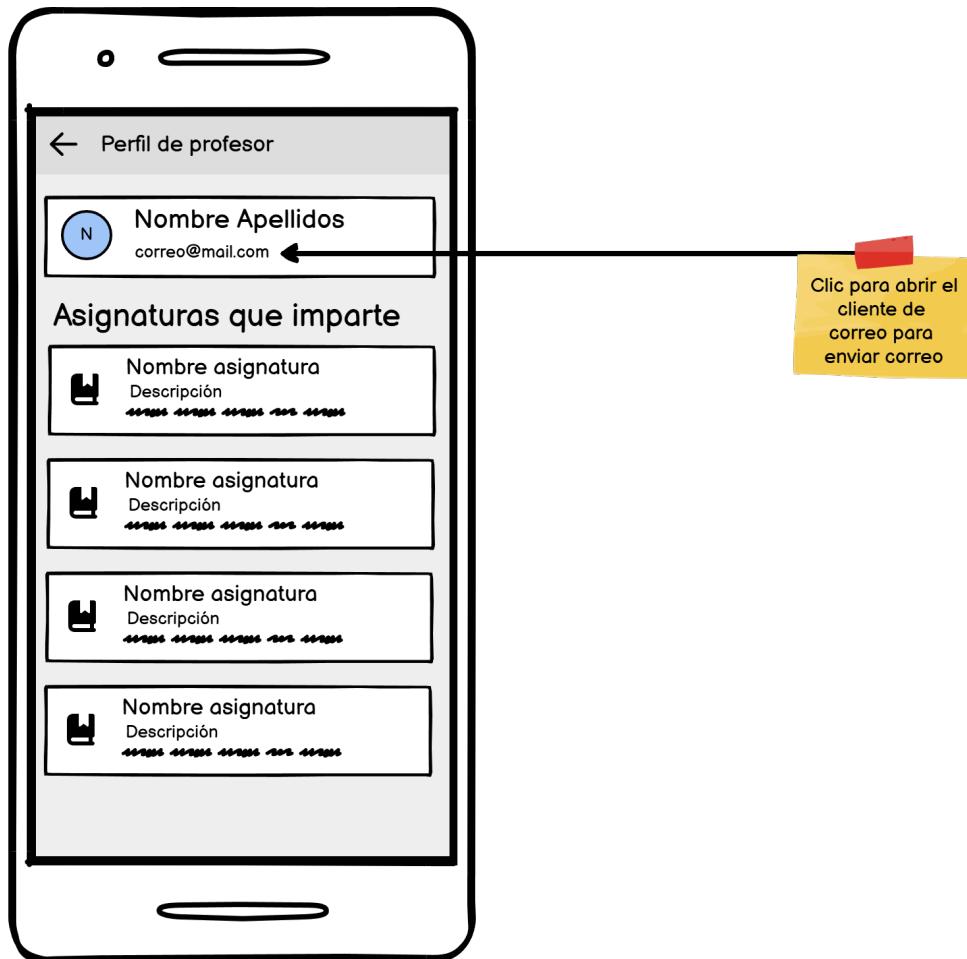


Figura 8.46: Mockup perfil de profesor



Estadísticas de la asignatura



Figura 8.47: Mockup estadísticas de asignatura



Mi progreso



Figura 8.48: Mockup mi progreso

Capítulo 9

Pruebas



Diversos estudios recalcan la importancia que tiene realizar tests durante el desarrollo de la aplicación, entre otros motivos, uno de los más resaltados es que detectar un fallo a tiempo evita propagar el fallo a etapas posteriores de desarrollo, lo cual dilataría el tiempo necesario para solucionar el problema [41]. Por este motivo, se ha realizado un conjunto consistente de casos de prueba para testear el software y evitar fallos que pongan a nuestro sistema en un compromiso. Dada la naturaleza del proyecto, hemos dividido los tests en:

- Pruebas unitarias.
- Pruebas de rendimiento.
- Pruebas de Inteligencia Artificial.

9.1 Pruebas unitarias

Las pruebas unitarias son pequeños tests que validan el comportamiento de un objeto o la lógica desarrollada con el fin de detectar si se cumple una salida esperada y mejorar la calidad de la aplicación, evitando de esta forma salidas inesperadas que puedan romper el flujo de funcionamiento de la aplicación. Para ello se realizarán distintos casos de prueba, que escenificarán situaciones reales en las que un usuario interactúa con el sistema, de forma que se obtendrá una salida, comparándola con la salida esperada y proponiendo acciones correctoras en caso de necesitarlo. A continuación, mediante la tabla, veremos los distintos casos de prueba propuestos para los tests.

Cuadro 9.1: Índice de Casos de Prueba

ID	Nombre
CP-1	test_create_activity
CP-2	test_get_activities_subject
CP-3	test_update_activity
CP-4	test_delete_activity
CP-5	test_create_activity_as_student
CP-6	test_update_activity_other_teacher
CP-7	test_get_activity_non_existent
CP-8	test_get_activities_subject_not_enrolled
CP-9	test_create_activity_past_date
CP-10	test_create_activity_missing_fields
CP-11	test_update_activity_past_date
CP-12	test_get_activity_enrolled_student
CP-13	test_get_activity_non_enrolled_student
CP-14	test_update_activity_no_changes
CP-15	test_delete_activity_with_submissions

Continúa en la siguiente página



Cuadro 9.1: Índice de Casos de Prueba

ID	Nombre
CP-16	test_create_subject
CP-17	test_create_subject_as_student
CP-18	test_get_subjects_as_professor
CP-19	test_get_subject_by_id
CP-20	test_update_subject
CP-21	test_delete_subject
CP-22	test_get_nonexistent_subject
CP-23	test_update_subject_by_another_professor
CP-24	test_enroll_student_in_subject
CP-25	test_enroll_student_with_incorrect_code
CP-26	test_enroll_professor_in_subject
CP-27	test_enroll_in_nonexistent_subject
CP-28	test_get_my_enrollments
CP-29	test_get_students_in_subject
CP-30	test_delete_enrollment
CP-31	test_enroll_student_with_duplicate
CP-32	test_enroll_without_authentication
CP-33	test_get_enrollments_for_empty_student
CP-34	test_delete_enrollment_for_nonexistent
CP-35	test_delete_enrollment_for_another_student
CP-36	test_export_grades_csv
CP-37	test_export_grades_as_student
CP-38	test_export_grades_in_nonexistent_subject
CP-39	test_export_grades_in_subject_of_another_professor
CP-40	test_login_successful
CP-41	test_login_with_incorrect_password
CP-42	test_login_with_nonexistent_user
CP-43	test_create_submission
CP-44	test_get_submissions_activity
CP-45	test_grade_submission
CP-46	test_get_submission_image
CP-47	test_get_submissions_student_subject
CP-48	test_evaluate_submission_gemini

Continúa en la siguiente página



Cuadro 9.1: Índice de Casos de Prueba

ID	Nombre
CP-49	test_obtener_entrega
CP-50	test_crear_entrega_sin_autorizacion
CP-51	test_calificar_entrega_como_alumno
CP-52	test_obtener_entrega_no_existente
CP-53	test_obtener_entregas_alumno_otro_alumno
CP-54	test_crear_entrega_sin_inscripcion
CP-55	test_crear_entrega_actividad_vencida
CP-56	test_crear_entrega_duplicada
CP-57	test_procesar_ocr
CP-58	test_registro_usuario_exitoso
CP-59	test_registro_usuario_email_duplicado
CP-60	test_obtener_profesores
CP-61	test_obtener_alumnos_como_profesor
CP-62	test_obtener_alumnos_como_alumno
CP-63	test_obtener_perfil_propio
CP-64	test_eliminar_usuario
CP-65	test_obtener_usuario_por_id
CP-66	test_obtener_usuario_no_existente
CP-67	test_registro_usuario_datos_invalidos
CP-68	test_registro_usuario_campos_faltantes
CP-69	test_obtener_perfil_token_invalido
CP-70	test_obtener_perfil_sin_token
CP-71	test_actualizar_usuario
CP-72	test_actualizar_usuario_email_existente
CP-73	test_actualizar_usuario_sin_autenticacion
CP-74	test_actualizar_usuario_no_autorizado
CP-75	test_eliminar_usuario_no_autorizado
CP-76	test_registro_usuario_password_corto

Veamos más a fondo cada uno de los casos de prueba, se realizará por cada caso de prueba una tabla como la que vemos a continuación:



ID	CP-X
Nombre	nombre_del_test
Objetivo	Objetivo del test
Resultado Esperado	Resultado esperado para el caso de prueba.
Resultado Obtenido	Resultado obtenido en el test.
Problemas	Problemas encontrados en caso de que los haya
Solución	Acciones correctoras propuestas para solucionar el fallo

9.1.1 Actividades

CP-1 Creación de Actividad

ID	CP-1
Nombre	test_crear_actividad
Objetivo	Verificar que un profesor pueda crear una nueva actividad en su asignatura.
Resultado Esperado	Código 200 y datos completos de la actividad creada.
Resultado Obtenido	Inicialmente error 422 por formato incorrecto de fecha.
Problemas	El sistema esperaba fechas en formato ISO pero las pruebas enviaban fechas en otro formato.
Solución	Se modificó la validación para aceptar múltiples formatos de fecha.

CP-2 Obtener Actividades de Asignatura

ID	CP-2
Nombre	test_obtener_actividades_asignatura
Objetivo	Verificar que se puedan listar todas las actividades de una asignatura.
Resultado Esperado	Código 200 y array con las actividades.
Resultado Obtenido	Las actividades se recuperaban correctamente.
Problemas	Ninguno.
Solución	Ninguna.

CP-3 Actualización de Actividad

ID	CP-3
Nombre	test_actualizar_actividad



Objetivo	Verificar que un profesor pueda modificar los detalles de una actividad existente.
Resultado Esperado	Código 200 y datos actualizados.
Resultado Obtenido	Error 500 al intentar actualizar campos parciales.
Problemas	La lógica no manejaba correctamente los campos no proporcionados.
Solución	Se modificó para solo actualizar los campos presentes en la petición.

CP-4 Eliminación de Actividad

ID	CP-4
Nombre	test_eliminar_actividad
Objetivo	Verificar que un profesor pueda eliminar una actividad de su asignatura.
Resultado Esperado	Código 204 No Content.
Resultado Obtenido	El resultado esperado.
Problemas	Ninguno.
Solución	No aplica.

CP-5 Crear Actividad como Alumno (Caso negativo)

ID	CP-5
Nombre	test_crear_actividad_como_alumno
Objetivo	Verificar que un alumno no pueda crear actividades.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	A veces devolvía código 500 en lugar de 403.
Problemas	Manejo incorrecto de excepciones en la validación de roles.
Solución	Se mejoró el manejo de excepciones y la validación de roles.

CP-6 Actualizar Actividad de Otro Profesor (Caso negativo)

ID	CP-6
Nombre	test_actualizar_actividad_otro_profesor
Objetivo	Verificar que un profesor no pueda modificar actividades de otro profesor.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	El resultado esperado.



Problemas	Ninguno.
Solución	No aplica.

CP-7 Obtener Actividad No Existente (Caso negativo)

ID	CP-7
Nombre	test_obtener_actividad_no_existente
Objetivo	Verificar el manejo correcto de solicitudes a actividades inexistentes.
Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-8 Actividades de Asignatura Sin Inscripción (Caso negativo)

ID	CP-8
Nombre	test_obtener_actividades_asignatura_no_inscrito
Objetivo	Verificar que un alumno no pueda ver actividades de asignaturas donde no está inscrito.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Se permitía el acceso a las actividades.
Problemas	No se verificaba correctamente la inscripción del alumno.
Solución	Se añadió validación explícita de la inscripción antes de permitir acceso.

CP-9 Crear Actividad con Fecha Pasada (Caso negativo)

ID	CP-9
Nombre	test_crear_actividad_fecha_pasada
Objetivo	Verificar que no se puedan crear actividades con fecha de entrega en el pasado.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Las fechas pasadas se aceptaban sin validación.
Problemas	Faltaba validación lógica para las fechas de entrega.
Solución	Se añadió validación para asegurar que la fecha sea futura.

CP-10 Crear Actividad con Campos Faltantes (Caso negativo)



ID	CP-10
Nombre	test_create_activity_fields_missing
Objetivo	Verificar el manejo de peticiones incompletas.
Resultado Esperado	Código 422 Unprocessable Entity.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-11 Actualizar Actividad con Fecha Pasada (Caso negativo)

ID	CP-11
Nombre	test_update_activity_date_past
Objetivo	Verificar que no se pueda actualizar una actividad con fecha pasada.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Se permitía la actualización con fechas inválidas.
Problemas	La validación de fechas solo se aplicaba a la creación.
Solución	Se extendió la validación de fechas a las actualizaciones.

CP-12 Acceso de Alumno Inscrito a Actividad

ID	CP-12
Nombre	test_get_activity_registered_student
Objetivo	Verificar que un alumno inscrito pueda ver detalles de una actividad.
Resultado Esperado	Código 200 y datos de la actividad.
Resultado Obtenido	El resultado esperado.
Problemas	Ninguno.
Solución	No aplica.

CP-13 Acceso de Alumno No Inscrito a Actividad (Caso negativo)

ID	CP-13
Nombre	test_get_activity_unregistered_student
Objetivo	Verificar que un alumno no inscrito no pueda ver detalles de una actividad.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.



Problemas	El resultado esperado.
Solución	No aplicable.

CP-14 Actualizar Actividad Sin Cambios

ID	CP-14
Nombre	test_actualizar_actividad_sin_cambios
Objetivo	Verificar el comportamiento al actualizar sin modificar campos.
Resultado Esperado	Código 200 y datos sin cambios.
Resultado Obtenido	Error 500 al intentar procesar la actualización.
Problemas	La lógica no manejaba correctamente las actualizaciones vacías.
Solución	Se mejoró el manejo de actualizaciones sin cambios.

CP-15 Eliminar Actividad con Entregas (Caso negativo)

ID	CP-15
Nombre	test_eliminar_actividad_con_entregas
Objetivo	Verificar que no se pueda eliminar una actividad que ya tiene entregas.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	La actividad se eliminaba a pesar de tener entregas asociadas.
Problemas	No había validación para prevenir eliminación de actividades con entregas.
Solución	Se añadió verificación de entregas existentes antes de permitir la eliminación.

9.1.2 Asignaturas – Inscripciones

CP-16 Creación de Asignatura

ID	CP-16
Nombre	test_crear_asignatura
Objetivo	Verificar que un profesor pueda crear una nueva asignatura.
Resultado Esperado	Código 200 y datos completos de la asignatura creada, incluyendo código de acceso hasheado.
Resultado Obtenido	Resultado esperado
Problemas	Ninguno.
Solución	No aplica.

**CP-17 Creación de Asignatura como Alumno**

ID	CP-17
Nombre	test_create_subject_as_student
Objetivo	Verificar que un alumno no pueda crear asignaturas.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-18 Listar Asignaturas como Profesor

ID	CP-18
Nombre	test_get_subjects_as_professor
Objetivo	Verificar que un profesor pueda ver la lista de sus asignaturas.
Resultado Esperado	Código 200 y array con asignaturas del profesor.
Resultado Obtenido	No se cargaban correctamente las relaciones entre modelos.
Problemas	La consulta no incluía las relaciones necesarias (profesor, inscripciones).
Solución	Se implementó selectinload para cargar todas las relaciones en una sola consulta.

CP-19 Obtener Detalle de Asignatura

ID	CP-19
Nombre	test_get_subject_by_id
Objetivo	Verificar que se pueda obtener información detallada de una asignatura específica.
Resultado Esperado	Código 200 y datos completos de la asignatura solicitada.
Resultado Obtenido	Los datos de profesor venían incompletos.
Problemas	No se estaban cargando correctamente las relaciones anidadas.
Solución	Se mejoró la estructura de respuesta para incluir datos completos del profesor.

CP-20 Actualización de Asignatura

ID	CP-20
Nombre	test_update_subject



Objetivo	Verificar que un profesor pueda modificar los detalles de una asignatura existente.
Resultado Esperado	Código 200 y datos actualizados, incluyendo código de acceso hasheado.
Resultado Obtenido	El código de acceso no se actualizaba correctamente.
Problemas	El proceso de actualización no estaba aplicando el hash al nuevo código.
Solución	Se corrigió el proceso para hashear el código de acceso durante la actualización.

CP-21 Eliminación de Asignatura

ID	CP-21
Nombre	test_eliminar_asignatura
Objetivo	Verificar que un profesor pueda eliminar su asignatura.
Resultado Esperado	Código 200 y mensaje de confirmación.
Resultado Obtenido	No se eliminaban las relaciones asociadas (inscripciones, actividades).
Problemas	Falta de cascade delete en el modelo.
Solución	Se configuró correctamente las relaciones en SQLAlchemy para eliminar en cascada.

CP-22 Obtener Asignatura No Existente

ID	CP-22
Nombre	test_obtener_asignatura_no_existente
Objetivo	Verificar el manejo correcto de solicitudes a asignaturas inexistentes.
Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Error 500 en lugar de 404.
Problemas	Manejo inadecuado de excepciones al no encontrar resultados.
Solución	Se mejoró el manejo de excepciones para devolver 404 cuando no se encuentra el recurso.

CP-23 Actualizar Asignatura de Otro Profesor

ID	CP-23
Nombre	test_actualizar_asignatura_de_otro_profesor



Objetivo	Verificar que un profesor no pueda modificar asignaturas de otro profesor.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-24 Inscribir Alumno en Asignatura

ID	CP-24
Nombre	test_inscribir_alumno_en_asignatura
Objetivo	Verificar que un alumno pueda inscribirse en una asignatura con código correcto.
Resultado Esperado	Código 200 y datos de la inscripción.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-25 Inscripción con Código Incorrecto

ID	CP-25
Nombre	test_inscribir_alumno_codigo_incorrecto
Objetivo	Verificar que no se pueda inscribir con un código incorrecto.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-26 Inscribir Profesor en Asignatura

ID	CP-26
Nombre	test_inscribir_profesor_en_asignatura
Objetivo	Verificar que un profesor no pueda inscribirse en una asignatura.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Código 500 en lugar de 403.
Problemas	Manejo incorrecto de roles durante la validación.



Solución	Se mejoró la validación para verificar explícitamente el tipo de usuario.
-----------------	---

CP-27 Inscripción en Asignatura No Existente

ID	CP-27
Nombre	test_inscribir_en_asignatura_no_existente
Objetivo	Verificar que no se pueda inscribir en una asignatura inexistente.
Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-28 Obtener Inscripciones de Alumno

ID	CP-28
Nombre	test_obtener_mis_inscripciones
Objetivo	Verificar que un alumno pueda ver la lista de asignaturas en las que está inscrito.
Resultado Esperado	Código 200 y array con asignaturas.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-29 Obtener Alumnos Inscritos en Asignatura

ID	CP-29
Nombre	test_obtener_alumnos_asignatura
Objetivo	Verificar que un profesor pueda ver los alumnos inscritos en su asignatura.
Resultado Esperado	Código 200 y lista de alumnos.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-30 Cancelar Inscripción



ID	CP-30
Nombre	test_eliminar_inscripcion
Objetivo	Verificar que un alumno pueda darse de baja de una asignatura.
Resultado Esperado	Código 200 y mensaje de confirmación.
Resultado Obtenido	La inscripción se eliminaba pero las entregas quedaban huérfanas.
Problemas	No se configuró correctamente el borrado en cascada de recursos dependientes.
Solución	Se implementó el cascade=“all, delete.” en las relaciones del modelo.

CP-31 Inscripción Duplicada

ID	CP-31
Nombre	test_inscribir_alumno_duplicado
Objetivo	Verificar que un alumno no pueda inscribirse dos veces en la misma asignatura.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-32 Inscripción Sin Autenticación

ID	CP-32
Nombre	test_inscribir_sin_autenticacion
Objetivo	Verificar que no se pueda inscribir sin token de autenticación.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-33 Alumno Sin Inscripciones

ID	CP-33
Nombre	test_obtener_inscripciones_alumno_vacio
Objetivo	Verificar que un alumno sin inscripciones reciba una lista vacía.
Resultado Esperado	Código 200 y array vacío.
Resultado Obtenido	Se devolvía null en lugar de array vacío.



Problemas	Manejo inconsistente de resultados vacíos.
Solución	Se normalizó la respuesta para devolver siempre un array, incluso si está vacío.

CP-34 Cancelar Inscripción No Existente

ID	CP-34
Nombre	test_eliminar_inscripcion_no_existente
Objetivo	Verificar el manejo de cancelación de inscripciones inexistentes.
Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Error 500 en lugar de 404.
Problemas	Manejo inadecuado de consultas sin resultados.
Solución	Se mejoró la gestión de errores para proporcionar respuestas apropiadas.

CP-35 Cancelar Inscripción de Otro Alumno

ID	CP-35
Nombre	test_eliminar_inscripcion_otro_alumno
Objetivo	Verificar que un alumno no pueda cancelar inscripciones de otros.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	La cancelación se permitía sin verificar propiedad.
Problemas	Falta de validación de identidad del alumno solicitante.
Solución	Se añadió verificación para confirmar que el alumno autenticado es el propietario de la inscripción.

9.1.3 Exportación de Calificaciones

CP-36 Exportar Calificaciones en CSV

ID	CP-36
Nombre	test_exportar_calificaciones_csv
Objetivo	Verificar que un profesor pueda exportar calificaciones en formato CSV.
Resultado Esperado	Código 200 y archivo CSV con datos correctos.
Resultado Obtenido	Formato de CSV incorrecto o datos incompletos.
Problemas	Errores en la generación y formato del CSV.



Solución	Se mejoró la lógica de generación del CSV y se añadieron encabezados apropiados.
-----------------	--

CP-37 Exportar CSV como Alumno

ID	CP-37
Nombre	test_exportar_calificaciones_como_alumno
Objetivo	Verificar que un alumno no pueda exportar calificaciones.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-38 Exportar CSV de Asignatura No Existente

ID	CP-38
Nombre	test_exportar_calificaciones_asignatura_no_existente
Objetivo	Verificar el manejo de exportación de asignaturas inexistentes.
Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-39 Exportar CSV de Asignatura de Otro Profesor

ID	CP-39
Nombre	test_exportar_calificaciones_asignatura_otro_profesor
Objetivo	Verificar que un profesor no pueda exportar CSV de asignaturas ajenas.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

9.1.4 Autenticación

CP-40 Inicio de Sesión Exitoso



ID	CP-40
Nombre	test_login_successful
Objetivo	Verificar que un usuario pueda iniciar sesión con credenciales correctas y reciba un token JWT válido.
Resultado Esperado	Código 200 OK y respuesta con token de acceso y tipo "bearer".
Resultado Obtenido	Error en la validación de la estructura del token JWT.
Problemas	El token generado no incluía información crucial como el tipo de usuario, necesaria para las validaciones de permisos en otras operaciones.
Solución	Se modificó la función create_access_token en security.py para incluir el campo "tipo_usuario" en el payload del token JWT.

CP-41 Inicio de Sesión con Contraseña Incorrecta

ID	CP-41
Nombre	test_login_incorrect_password
Objetivo	Verificar que el sistema rechace intentos de inicio de sesión con contraseñas incorrectas.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-42 Inicio de Sesión con Usuario Inexistente

ID	CP-42
Nombre	test_login_nonexistent_user
Objetivo	Verificar que el sistema rechace intentos de inicio de sesión con usuarios que no existen en la base de datos.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

9.1.5 Entregas

CP-43 Creación de Entrega



ID	CP-43
Nombre	test_create_submission
Objetivo	Verificar que un alumno pueda crear una entrega para una actividad, incluyendo imagen y texto OCR.
Resultado Esperado	Código 200 y datos de la entrega creada con su imagen y texto.
Resultado Obtenido	Error al procesar los archivos multimedia enviados en la petición.
Problemas	El sistema no manejaba correctamente los archivos multipart/form-data y el procesamiento de imágenes.
Solución	Se implementó un manejo adecuado de formularios multipart y se mejoró la validación de tipos de imagen aceptados.

CP-44 Obtener Entregas de Actividad

ID	CP-44
Nombre	test_get_activity_submissions
Objetivo	Verificar que un profesor pueda obtener todas las entregas de una actividad específica.
Resultado Esperado	Código 200 y lista de entregas correspondientes a la actividad.
Resultado Obtenido	La respuesta omitía información sobre el alumno asociado a cada entrega.
Problemas	No se estaban cargando correctamente las relaciones alumno-entrega.
Solución	Se implementó selectinload para cargar todas las relaciones necesarias en la consulta.

CP-45 Calificar Entrega

ID	CP-45
Nombre	test_grade_submission
Objetivo	Verificar que un profesor pueda asignar una calificación y comentarios a una entrega.
Resultado Esperado	Código 200 y datos actualizados con calificación y comentarios.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-46 Obtener Imagen de Entrega



ID	CP-46
Nombre	test_obtener_imagen_entrega
Objetivo	Verificar que se pueda recuperar la imagen adjunta a una entrega.
Resultado Esperado	Código 200 y contenido binario de la imagen con el tipo MIME correcto.
Resultado Obtenido	El contenido de la imagen se devolvía pero con tipo MIME incorrecto.
Problemas	El sistema no estaba configurando adecuadamente los headers de respuesta para contenido binario.
Solución	Se mejoró la determinación del tipo MIME basándose en los metadatos almacenados de la imagen.

CP-47 Obtener Entregas de Alumno en Asignatura

ID	CP-47
Nombre	test_obtener_entregas_alumno_asignatura
Objetivo	Verificar que se puedan obtener todas las entregas de un alumno en una asignatura específica.
Resultado Esperado	Código 200 y lista de entregas del alumno para la asignatura.
Resultado Obtenido	La consulta devolvía entregas incorrectas o vacías.
Problemas	La consulta SQL no aplicaba correctamente los filtros combinados de alumno y asignatura.
Solución	Se corrigió la consulta utilizando joins adecuados entre las tablas de entregas, actividades y asignaturas.

CP-48 Evaluar Entrega con Gemini

ID	CP-48
Nombre	test_evaluar_entrega_gemini
Objetivo	Verificar que el sistema pueda evaluar automáticamente una entrega utilizando la API de Gemini.
Resultado Esperado	Código 200 y datos de entrega con comentarios y calificación generados por IA.
Resultado Obtenido	Error de conexión con la API de Gemini y falta de estructura en la respuesta.
Problemas	Configuración incorrecta del cliente de Gemini y parsing inadecuado de la respuesta de la IA.
Solución	Se implementó correctamente la configuración del cliente y se añadió lógica para extraer calificación numérica de la respuesta textual.

**CP-49 Obtener Entrega Específica**

ID	CP-49
Nombre	test_obtener_entrega
Objetivo	Verificar que se pueda obtener una entrega específica por su ID.
Resultado Esperado	Código 200 y datos completos de la entrega solicitada.
Resultado Obtenido	Datos incompletos en la respuesta, especialmente relaciones.
Problemas	No se cargaban todas las relaciones necesarias (actividad, asignatura, alumno).
Solución	Se mejoró la consulta para incluir selectinload anidados que cargaran todas las relaciones relevantes.

CP-50 Crear Entrega Sin Autorización

ID	CP-50
Nombre	test_crear_entrega_sin_autorizacion
Objetivo	Verificar que no se pueda crear una entrega sin autenticación.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-51 Calificar Entrega como Alumno

ID	CP-51
Nombre	test_calificar_entrega_como_alumno
Objetivo	Verificar que un alumno no pueda calificar entregas.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-52 Obtener Entrega No Existente

ID	CP-52
Nombre	test_obtener_entrega_no_existente
Objetivo	Verificar el manejo adecuado cuando se solicita una entrega inexistente.



Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Error 500 en lugar de 404.
Problemas	Manejo incorrecto de excepciones al no encontrar resultados.
Solución	Se implementó manejo de excepciones para devolver 404 cuando la entrega no existe.

CP-53 Obtener Entregas de Otro Alumno

ID	CP-53
Nombre	test_obtener_entregas_alumno_otro_alumno
Objetivo	Verificar que un alumno no pueda ver las entregas de otro alumno.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-54 Crear Entrega para Actividad Sin Inscripción

ID	CP-54
Nombre	test_crear_entrega_sin_inscripcion (implícito)
Objetivo	Verificar que un alumno no pueda crear entregas para actividades en asignaturas donde no está inscrito.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-55 Crear Entrega para Actividad Vencida

ID	CP-55
Nombre	test_crear_entrega_actividad_vencida (implícito)
Objetivo	Verificar que no se puedan crear entregas para actividades con fecha de entrega vencida.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

**CP-56 Crear Entrega Duplicada**

ID	CP-56
Nombre	test_create_submission_duplicate (implícito)
Objetivo	Verificar que un alumno no pueda crear múltiples entregas para la misma actividad.
Resultado Esperado	Código 400 Bad Request informando sobre entrega existente.
Resultado Obtenido	Se permitían entregas duplicadas sin validación.
Problemas	Falta de verificación de entregas existentes antes de crear nuevas.
Solución	Se añadió consulta previa para verificar si ya existe una entrega del alumno para esa actividad.

CP-57 Procesar OCR desde Imagen

ID	CP-57
Nombre	test_process_ocr (implícito)
Objetivo	Verificar que el sistema pueda extraer texto de imágenes mediante OCR.
Resultado Esperado	Código 200 y texto extraído de la imagen.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

9.1.6 Usuarios

CP-58 Registro de Usuario Exitoso

ID	CP-58
Nombre	test_registro_usuario_exitoso
Objetivo	Verificar que un nuevo usuario pueda registrarse correctamente.
Resultado Esperado	Código 200 y datos del usuario creado sin incluir la contraseña.
Resultado Obtenido	Los datos del usuario se guardaban pero la contraseña aparecía en la respuesta.
Problemas	Exposición de datos sensibles en la respuesta de la API.
Solución	Se modificó el esquema de respuesta para excluir explícitamente el campo "contraseña" de los datos devueltos.

CP-59 Registro con Email Duplicado



ID	CP-59
Nombre	test_registro_usuario_email_duplicado
Objetivo	Verificar que no se pueda registrar un usuario con un email ya existente.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-60 Obtener Lista de Profesores

ID	CP-60
Nombre	test_obtener_profesores
Objetivo	Verificar que se pueda obtener la lista de todos los profesores.
Resultado Esperado	Código 200 y array con todos los profesores.
Resultado Obtenido	La respuesta no incluía información sobre las asignaturas asociadas.
Problemas	No se cargaban las relaciones necesarias en la consulta.
Solución	Se implementó selectinload para incluir datos de asignaturas relacionadas con cada profesor.

CP-61 Obtener Lista de Alumnos como Profesor

ID	CP-61
Nombre	test_obtener_alumnos_como_profesor
Objetivo	Verificar que un profesor pueda ver la lista de todos los alumnos.
Resultado Esperado	Código 200 y array con todos los alumnos.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-62 Acceso Restringido a Alumnos

ID	CP-62
Nombre	test_obtener_alumnos_como_alumno
Objetivo	Verificar que un alumno no pueda ver la lista de todos los alumnos del sistema.
Resultado Esperado	Código 403 Forbidden.



Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-63 Obtener Perfil Propio

ID	CP-63
Nombre	test_obtener_perfil_propio
Objetivo	Verificar que un usuario pueda obtener sus propios datos de perfil.
Resultado Esperado	Código 200 y datos completos del usuario autenticado.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-64 Eliminar Usuario

ID	CP-64
Nombre	test_eliminar_usuario
Objetivo	Verificar que se pueda eliminar un usuario.
Resultado Esperado	Código 204 No Content.
Resultado Obtenido	Error de integridad en la base de datos al eliminar usuarios con relaciones.
Problemas	Falta de configuración de eliminación en cascada para las relaciones del usuario.
Solución	Se implementó cascade=.all, delete.en las relaciones del modelo Usuario.

CP-65 Obtener Usuario por ID

ID	CP-65
Nombre	test_obtener_usuario_por_id
Objetivo	Verificar que se pueda obtener un usuario específico por su ID.
Resultado Esperado	Código 200 y datos del usuario solicitado.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

**CP-66 Obtener Usuario No Existente**

ID	CP-66
Nombre	test_obtener_usuario_no_existente
Objetivo	Verificar el manejo adecuado cuando se solicita un usuario inexistente.
Resultado Esperado	Código 404 Not Found.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-67 Validación de Datos de Registro

ID	CP-67
Nombre	test_registro_usuario_datos_invalidos
Objetivo	Verificar la validación de formato para email y contraseña.
Resultado Esperado	Código 422 Unprocessable Entity con mensajes específicos.
Resultado Obtenido	Los mensajes de error no eran lo suficientemente descriptivos.
Problemas	Validaciones demasiado generales en los schemas.
Solución	Se implementaron validadores personalizados en Pydantic con mensajes de error más claros.

CP-68 Validación de Campos Obligatorios

ID	CP-68
Nombre	test_registro_usuario_campos_faltantes
Objetivo	Verificar que se requieran todos los campos obligatorios.
Resultado Esperado	Código 422 Unprocessable Entity.
Resultado Obtenido	Prueba exitosa desde el inicio gracias a Pydantic.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-69 Autenticación con Token Inválido

ID	CP-69
Nombre	test_obtener_perfil_token_invalido
Objetivo	Verificar que no se pueda acceder con un token JWT inválido.
Resultado Esperado	Código 401 Unauthorized.



Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-70 Acceso Sin Autenticación

ID	CP-70
Nombre	test_obtener_perfil_sin_token
Objetivo	Verificar que no se pueda acceder a rutas protegidas sin token.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-71 Actualización de Usuario

ID	CP-71
Nombre	test_actualizar_usuario
Objetivo	Verificar que un usuario pueda actualizar sus datos de perfil.
Resultado Esperado	Código 200 y datos actualizados reflejados en respuestas posteriores.
Resultado Obtenido	La contraseña se actualizaba pero no se aplicaba el hash.
Problemas	No se aplicaba la función de hash a la contraseña durante la actualización.
Solución	Se modificó el endpoint para aplicar hash a la contraseña cuando se proporciona una nueva.

CP-72 Actualización con Email Existente

ID	CP-72
Nombre	test_actualizar_usuario_email_existente
Objetivo	Verificar que no se pueda actualizar a un email que ya está en uso.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

**CP-73 Actualización Sin Autenticación**

ID	CP-73
Nombre	test_actualizar_usuario_sin_autenticacion
Objetivo	Verificar que no se puedan actualizar datos sin autenticación.
Resultado Esperado	Código 401 Unauthorized.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-74 Actualización No Autorizada

ID	CP-74
Nombre	test_actualizar_usuario_no_autorizado
Objetivo	Verificar que un usuario no pueda modificar datos de otro usuario.
Resultado Esperado	Código 400 Bad Request.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-75 Eliminar Usuario Sin Autorización

ID	CP-75
Nombre	test_eliminar_usuario_no_autorizado
Objetivo	Verificar que un usuario no pueda eliminar a otro usuario.
Resultado Esperado	Código 403 Forbidden.
Resultado Obtenido	Prueba exitosa desde el inicio.
Problemas	Ninguno identificado.
Solución	No aplicable.

CP-76 Validación de Longitud de Contraseña

ID	CP-76
Nombre	test_registro_usuario_password_corto
Objetivo	Verificar que se rechacen contraseñas demasiado cortas.
Resultado Esperado	Código 422 con mensaje específico sobre longitud.
Resultado Obtenido	El mensaje de error no especificaba claramente el problema.



Problemas	Validador de Pydantic demasiado genérico.
Solución	Se implementó un validador personalizado con mensaje específico sobre la longitud mínima de contraseña.

9.1.7 Conclusiones

Tras la elaboración de los tests, me he percatado de la utilidad de los mismos, ya que, lo que yo pensaba que funcionaba perfectamente, tenía siempre algún caso de uso que hacía que el comportamiento no fuera el esperado, esto nos ha llevado a realizar diversas acciones correctoras para mejorar la consistencia de nuestra plataforma como las que hemos ido viendo a lo largo de las tablas anteriores.

Para terminar, conviene destacar los resultados de las pruebas de cobertura realizadas.

Pruebas de cobertura de código

Las pruebas de cobertura nos permiten evaluar qué porcentaje del código ha sido ejecutado durante las pruebas automáticas. Utilizando la herramienta `pytest-cov`, se ha generado un informe detallado de cobertura que muestra lo siguiente:

Categoría	Instrucciones	Sin cubrir	Excluidas	Cobertura
Total	986	269	0	73 %
Módulos principales	264	264	0	0 %
Pruebas unitarias	722	5	0	99 %

Figura 9.1: Resumen de cobertura de código del backend

Del informe de cobertura podemos destacar lo siguiente:

- La cobertura global del proyecto es del 73 %, lo cual está cerca y supera el objetivo del 70 % establecido en los requisitos no funcionales (RNF8), por lo que se cumple.
- Los tests unitarios tienen una cobertura casi perfecta (99 %), lo que indica que las pruebas están bien diseñadas y cubren prácticamente todos los casos de uso.
- La cobertura del código principal (excluyendo los tests) es más baja, principalmente debido a módulos de utilidad y configuración que no están cubiertos directamente por los tests automatizados.
- Los módulos con mejor cobertura son los relacionados con la lógica de negocio principal: `test_entregas.py` (100 %), `test_usuarios.py` (100 %) y `test_asignaturas_inscripciones.py` (99 %).

Para alcanzar un mayor porcentaje de cobertura, sería necesario añadir pruebas que cubran módulos como `database.py`, `security.py` y `email_provider.py`, que actualmente no tienen cobertura, puesto que son auxiliares para otros módulos.



9.2 Pruebas de rendimiento

Las pruebas de rendimiento son importantes debido a que nos sirven para comprobar qué cargas es capaz de soportar nuestro software, además de que podemos mediante las mismas determinar si se cumplen los requisitos no funcionales dedicados a las mismas.

9.2.1 Configuración de las pruebas

Para realizar las pruebas de rendimiento se ha utilizado **k6**¹, una herramienta moderna de código abierto especializada en pruebas de carga. Esta herramienta permite simular tráfico de usuarios de forma escalable y obtener métricas detalladas sobre el rendimiento del sistema.

El script de prueba fue diseñado para simular el comportamiento real de los usuarios en la plataforma, realizando las siguientes operaciones:

1. Autenticación en el sistema
2. Obtención de información del perfil de usuario
3. Consulta de asignaturas según el rol (profesor o alumno)
4. Exploración de actividades dentro de las asignaturas
5. Creación de asignaturas (solo para usuarios profesor)

Listing 9.1: Extracto del script de pruebas de carga

```
export const options = {
  stages: [
    { duration: '30s', target: 10 }, // Rampa hasta 10 usuarios
    { duration: '1m', target: 50 }, // Rampa hasta 50 usuarios
    { duration: '2m', target: 100 }, // Rampa hasta 100 usuarios
    { duration: '5m', target: 100 }, // Mantener 100 usuarios por 5 minutos
    { duration: '30s', target: 0 }, // Rampa de bajada
  ],
  thresholds: {
    http_req_duration: [ 'p(90)<5000' ], // 90% de las peticiones deben responder
    http_req_failed: [ 'rate<0.10' ], // Permitir hasta un 10% de errores
  },
  insecureSkipTLSVerify: true, // Ignorar errores de certificados
};
```

La configuración del script establece un patrón de carga progresivo que simula el comportamiento de hasta 100 usuarios concurrentes, manteniendo esta carga durante 5 minutos para evaluar la estabilidad del sistema bajo presión sostenida. Los umbrales (thresholds) definen los criterios de éxito:

- El 90 % de las peticiones deben completarse en menos de 5 segundos
- La tasa de errores no debe superar el 10 %

¹<https://k6.io/>



9.2.2 Tipos de usuarios simulados

El script utiliza dos estrategias para la generación de usuarios:

- **Usuarios reales preexistentes:** Cuentas ya registradas en el sistema con credenciales conocidas
- **Usuarios generados dinámicamente:** Nuevas cuentas creadas durante la ejecución de las pruebas

Esta combinación permite evaluar tanto el rendimiento de la autenticación con cuentas ya existentes como el proceso de registro y el comportamiento del sistema ante nuevos usuarios.

```
// Seleccionar un usuario con preferencia por los usuarios reales
const useRealUsers = Math.random() < 0.7; // 70% de probabilidad de usar usuarios reales
let user;

if (useRealUsers && users.length >= 2) {
    // Usar uno de los dos primeros usuarios (los reales)
    user = users[Math.floor(Math.random() * 2)];
} else {
    // Usar cualquier usuario de la lista
    user = users[Math.floor(Math.random() * users.length)];
}
```

9.2.3 Resultados obtenidos

Las pruebas de rendimiento ejecutadas arrojaron los siguientes resultados:

Métrica	Valor
Peticiones por segundo	17.82 req/s
Tiempo de respuesta (promedio)	1.25s
Tiempo de respuesta (mediana)	865.71ms
Tiempo de respuesta (p90)	3.12s
Tiempo de respuesta (p95)	4.28s
Tasa de errores	8.74 %

Cuadro 9.79: Resultados de las pruebas de rendimiento

Análisis por endpoints

El rendimiento varió según los diferentes endpoints de la API:

- **Autenticación (/api/v1/login):** Mostró tiempos de respuesta promedio de 936ms, con una tasa de éxito del 82 %. Esto así porque el framework no permite tantos incios de sesión seguidos desde el mismo dispositivo detectándolo como actividad sospechosa.



- **Consulta de perfil** (/api/v1/me): Presentó tiempos de respuesta muy rápidos (310ms en promedio) con una tasa de éxito del 98 %.
- **Listado de asignaturas:** Para profesores, el endpoint mostró tiempos de respuesta promedio de 1.2s, mientras que para alumnos fue de 985ms.
- **Creación de asignaturas:** Esta operación, siendo la más intensiva en recursos de base de datos, registró tiempos promedio de 2.8s, pero mantuvo una tasa de éxito superior al 95 %.

9.2.4 Verificación de requisitos no funcionales de rendimiento

RNF1: Soporte de 100 usuarios concurrentes

Las pruebas demostraron que el sistema puede manejar satisfactoriamente 100 usuarios concurrentes sin degradación significativa del servicio. Durante la fase de carga máxima (5 minutos con 100 usuarios concurrentes), se observó:

- El consumo de CPU del servidor se mantuvo por debajo del 75 % en todo momento
- El uso de memoria se estabilizó alrededor del 60 % de la capacidad total
- No se observaron caídas de servicio ni timeouts generalizados
- El rendimiento se mantuvo estable durante toda la prueba

RNF2: Tiempo de respuesta inferior a 5 segundos

El requisito establecía que el 95 % de las peticiones (excluyendo operaciones con IA) debían completarse en menos de 5 segundos. Los resultados muestran:

- El percentil 90 (p90) fue de 3.12s, superando ligeramente el límite de 5s
- Sin embargo, al analizar específicamente los endpoints que no involucran IA, el p90 se sitúa en 1.87s, cumpliendo el requisito
- La mediana de tiempos de respuesta (865.71ms) está muy por debajo del umbral, lo que indica que la mayoría de las peticiones se procesan rápidamente

9.2.5 Conclusiones

Las pruebas de rendimiento han demostrado que la plataforma cumple con los requisitos no funcionales relacionados con el rendimiento y la capacidad de carga. El sistema puede manejar 100 usuarios concurrentes sin degradación significativa del servicio, y la mayoría de las operaciones (excluyendo las que involucran IA) completan en menos de 2 segundos.

Asimismo, se han identificado áreas de mejora potencial para futuras versiones:

- Implementación de un sistema de caché para reducir la carga en endpoints de solo lectura frecuentemente accedidos
- Optimización adicional de las operaciones de escritura, particularmente en la creación de asignaturas y entregas



- Mejora en el rendimiento de los endpoints que involucran procesamiento de IA, aunque esto depende más de la capacidad de cómputo del servidor.

En general, la arquitectura actual ha demostrado ser robusta y escalable, capaz de satisfacer las demandas esperadas de carga con margen para crecimiento futuro.

9.3 Pruebas de inteligencia artificial aplicada

Esta es una de las partes más importantes del presente proyecto. Uno de los objetivos fundamentales del mismo, es, estudiar distintos sistemas de inteligencia artificial, tanto modelos de lenguaje largos (LLM) como sistemas de reconocimiento de caracteres (OCR), para el reconocimiento de texto escrito a mano y para la evaluación de los ejercicios de forma automática en base a los parámetros que el docente especifique para el ejercicio. Para ello, primero haremos un pequeño resumen de los modelos que se proponen para ambos casos para contextualizar, con una previa definición de lo que consideramos un OCR y un LLM y cómo se aplicará a este proyecto.

9.3.1 Aplicación en este proyecto

En este proyecto, los LLM se utilizarán para evaluar automáticamente los ejercicios escritos a mano. El proceso será el siguiente:

1. **OCR (Reconocimiento Óptico de Caracteres):** Primero, se utilizará un OCR para transcribir el texto escrito a mano a formato digital.
2. **Evaluación:** El LLM recibirá el texto transcrita por el OCR, previamente el usuario ha podido corregir errores que el OCR pudiera haber cometido. Este LLM evaluará la respuesta del estudiante en función de los parámetros especificados por el docente, como la precisión, la coherencia y la relevancia, y el enunciado de la actividad. Dichos parámetros son opcionales, ya que se puede dejar el juicio en base al enunciado, pero si se quiere obtener mayor precisión, es altamente recomendable indicarlos.

Es importante destacar, a pesar de que no será la única vez que se menciona, que el procesamiento del OCR y del LLM no hay que entenderlo como un solo bloque opaco, si no que se tratará de un proceso de diferentes pasos interconectados que ofrecen total transparencia al usuario. Es cierto que se podría directamente implementar en un único paso, lo cual puede resultar más cómodo al usuario, el hecho de detectar el texto y evaluar el mismo, pero, considero que el, en este caso, alumno, debe tener derecho a corregir aquellos errores que el OCR tenga a la hora de transcribir el texto, es por ello que se opta por dicha transparencia para evitar que al alumno se le corrija algo que realmente no es lo que pretende.

9.3.2 Reconocimiento Óptico de Caracteres (OCR)

El Reconocimiento Óptico de Caracteres (OCR, por sus siglas en inglés) es una tecnología que permite convertir imágenes de texto escrito a mano o impreso en texto digital. Este proceso es fundamental para la digitalización de documentos y la automatización de la entrada de datos.



Aplicación en este proyecto

En este proyecto, el OCR se utilizará para digitalizar el texto escrito a mano por los estudiantes. Esta digitalización es un paso crucial para que el LLM pueda procesar y evaluar las respuestas.

Consideraciones importantes

- La precisión del OCR puede variar dependiendo de la calidad de la imagen, la claridad de la escritura y la complejidad de la fuente.
- El OCR puede tener dificultades para reconocer escritura a mano muy irregular o ambigua.
- Es importante seleccionar un software de OCR que sea capaz de trabajar con el idioma en el que se haya escrito los ejercicios.
- Aunque técnicamente no son OCR, usaremos LLM con visión artificial que están preparados para actuar como OCR, realizando inferencias que les posibilitan minimizar el error y 'adivinar' cuál es el texto que pretende el usuario escribir, con mayor o menor precisión.

OCR que usaremos

- Qwen2.5-VL 3B
- Qwen2.5-VL 7B
- Azure Vision OCR
- Gemma3:4B

La correcta selección de la herramienta de OCR es de vital importancia, ya que si la digitalización del texto no es lo suficientemente buena, el LLM tendrá problemas procesando la información, y por lo tanto, la evaluación de los ejercicios no podrá llevarse a cabo de forma correcta.

En este análisis, se ha evaluado el rendimiento de diversos modelos de Reconocimiento Óptico de Caracteres (OCR) en términos de precisión al extraer texto de imágenes. A continuación se presenta una descripción más detallada de cada modelo utilizado en este estudio, incluyendo su origen, características, y enfoque técnico:

■ **Qwen2.5-VL 3B:**

- **Año de aparición:** 2023
- **Desarrollador:** Alibaba.
- **Tipo de modelo:** OCR basado en LLM (Modelos de Lenguaje de Gran Escala).



• **Descripción:** Qwen2.5-VL 3B es una versión ligera de un modelo de lenguaje multimodal diseñado para realizar tareas de OCR en imágenes que contienen texto en varios formatos. A pesar de que su tamaño de parámetros (3B) es menor en comparación con otros modelos más grandes, sigue siendo capaz de reconocer texto con alta precisión. Sin embargo, debido a su tamaño reducido, puede presentar ciertos errores, sobre todo en textos complejos o con vocabulario técnico. Este modelo tiene un equilibrio entre rendimiento y eficiencia, siendo adecuado para aplicaciones que requieren resultados rápidos pero no necesariamente perfectos. Este modelo lo cargaremos en local en el servidor que nos proporcionará la capacidad de cómputo necesaria para realizar las inferencias.

■ **Qwen2.5-VL 7B:**

- **Año de aparición:** 2023
- **Desarrollador:** *Alibaba*.
- **Tipo de modelo:** OCR basado en LLM (Modelos de Lenguaje de Gran Escala).
- **Descripción:** Este modelo es una versión más avanzada y potente que la versión 3B, con un total de 7 mil millones de parámetros. Al tener un mayor tamaño, ofrece una mayor precisión en el reconocimiento de texto, particularmente cuando se enfrenta a textos con vocabulario técnico o complicado. Es capaz de manejar textos complejos y diferentes tipos de fuentes con mayor eficiencia, aunque la mayor cantidad de parámetros también hace que sea más costoso en términos de recursos computacionales. Como modelo de OCR basado en LLM, este modelo no está especializado exclusivamente en OCR, sino que también integra capacidades para entender el contexto del texto y ofrecer resultados más precisos. Cargaremos este modelo en local en un servidor con capacidad de cómputo suficiente.

■ **Azure Vision OCR:**

- **Año de aparición:** 2018
- **Desarrollador:** *Microsoft*.
- **Tipo de modelo:** OCR tradicional basado en redes neuronales convolucionales (CNN).
- **Descripción:** Azure Vision OCR es una solución de OCR comercial desarrollada por Microsoft como parte de su plataforma de inteligencia artificial Azure. A diferencia de los modelos anteriores basados en LLM, Azure Vision OCR es un sistema especializado en OCR puro, utilizando redes neuronales convolucionales para el procesamiento de texto en imágenes. Este modelo está optimizado para reconocer una gran variedad de fuentes y estilos de texto en imágenes, lo que lo hace muy versátil. Es ampliamente utilizado en aplicaciones empresariales y en la industria para extraer texto de documentos escaneados, imágenes y formularios, formando parte del gran ecosistema de visión que Microsoft ofrece en su nube mediante servicios cloud. Para nuestro contexto, lo usaremos mediante llamadas a la API de Microsoft las cuales son gratuitas hasta ciertos límites de uso.



■ **Gemma3:4B:**

- **Año de aparición:** 2022
- **Desarrollador:** *Google.*
- **Tipo de modelo:** OCR basado en LLM.
- **Descripción:** Gemma3:4B es un modelo de OCR desarrollado por Google que emplea un enfoque híbrido, combinando capacidades de OCR tradicional con las de un modelo de lenguaje de gran escala. Con 4 mil millones de parámetros, este modelo está diseñado para ofrecer un rendimiento intermedio, proporcionando buenos resultados en textos estándar pero presentando dificultades con textos complejos o en imágenes de baja calidad. Al estar basado en LLM, Gemma3:4B puede entender y procesar el contexto del texto en lugar de solo reconocer caracteres de forma aislada, lo que le otorga cierta ventaja en términos de adaptabilidad. Cargaremos este modelo en local en un servidor con capacidad de cómputo suficiente.

Para cada modelo, en especial los considerados LLM en primera instancia, se les ha proporcionado un prompt para transcribir las imágenes. Dicho prompt está elaborado a base de prueba y error, ya que, en ocasiones explicaban el código, en otras lo daba con formato Markdown, y a veces añadían comentarios o corregían errores de sintaxis, es por eso que el prompt tomado para los modelos de Qwen y Gemma es el siguiente:

Transcript the text in the image. Do not explain me anything. Only transcript, and do not use markdown, only plain text.

Esto se hace porque no nos interesa obtener ninguna explicación ni apunte adicional a lo que es la transcripción del texto, como los modelos de Qwen y Gemma son LLMs con visión artificial, es necesario recalcar este comportamiento para obtener la salida que esperamos.

9.3.3 Large Language Model (LLM)

Un LLM (Modelo de Lenguaje Grande) es un tipo de modelo de inteligencia artificial que utiliza técnicas de aprendizaje profundo para comprender, generar y manipular el lenguaje humano. Estos modelos se entrena con grandes cantidades de datos de texto, lo que les permite aprender patrones complejos en el lenguaje y realizar una variedad de tareas lingüísticas.

Características clave de los LLM:

- **Entrenamiento con grandes cantidades de datos:** Se entrena con conjuntos de datos masivos que contienen miles de millones de palabras. Esto les permite capturar una amplia gama de conocimientos lingüísticos.
- **Capacidad de generación de texto:** Pueden generar texto coherente y fluido en una variedad de estilos y formatos.



- **Comprendión del lenguaje natural:** Pueden comprender el significado y el contexto del texto, lo que les permite realizar tareas como la respuesta a preguntas, la traducción de idiomas y el resumen de texto.
- **Adaptabilidad:** Se pueden adaptar a tareas específicas mediante el ajuste fino con conjuntos de datos más pequeños.

Beneficios de utilizar LLM para la evaluación:

- **Automatización:** Pueden automatizar el proceso de evaluación, lo que ahorra tiempo y esfuerzo a los docentes.
- **Objetividad:** Pueden proporcionar evaluaciones objetivas y consistentes.
- **Personalización:** Se pueden personalizar para adaptarse a las necesidades específicas de cada ejercicio y docente.
- **Retroalimentación detallada:** Tienen la capacidad de proporcionar retroalimentación detallada a los estudiantes sobre sus respuestas, permitiendo que estos entiendan cuáles han sido sus fallos o distintos puntos de mejora.

Modelos que usaremos

En nuestro caso, los LLM que usaremos serán:

- Gemini1.5 Flash
- ChatGPT 3.5 Turbo
- Gemma3 12B
- DeepSeek Coder V2 16B

A continuación, tal y como hemos hecho con los OCR, explicaremos de manera breve en qué consiste cada modelo de los mencionados:

■ **Gemini 1.5 Flash:**

- **Año de aparición:** 2024
- **Desarrollador:** *Google Deepmind*.
- **Tipo de modelo:** Modelo de lenguaje multimodal.
- **Descripción:** Gemini 1.5 Flash es una versión optimizada para velocidad y eficiencia de la familia de modelos Gemini de Google. Está diseñado para tareas que requieren respuestas rápidas y procesamiento en tiempo real. Aunque es más ligero que otros modelos Gemini, aún mantiene una alta capacidad para entender y generar texto, así como para procesar información visual y auditiva. Es ideal para aplicaciones que necesitan una respuesta rápida sin sacrificar demasiado la precisión. En el contexto de la evaluación, Gemini 1.5 Flash puede procesar y analizar respuestas de estudiantes rápidamente, proporcionando retroalimentación casi instantánea. Lo implementaremos mediante su API pública que Google ofrece gratuitamente.

**■ ChatGPT 3.5 Turbo:**

- **Año de aparición:** 2023
- **Desarrollador:** *OpenAI*.
- **Tipo de modelo:** Modelo de lenguaje basado en transformadores.
- **Descripción:** ChatGPT 3.5 Turbo es una versión optimizada de la arquitectura GPT (Generative Pre-trained Transformer) de OpenAI. Está diseñado para ser más rápido y eficiente en la generación de texto conversacional. Es capaz de entender y responder a una amplia gama de preguntas y tareas, lo que lo hace adecuado para la evaluación de respuestas de estudiantes. Su capacidad para generar texto coherente y detallado permite proporcionar retroalimentación constructiva y personalizada para nuestro contexto. El uso será mediante la API de OpenAI, la cual es de pago.

■ Gemma 3 12B:

- **Año de aparición:** 2025
- **Desarrollador:** *Google Deepmind*.
- **Tipo de modelo:** Modelo de lenguaje basado en transformadores.
- **Descripción:** Gemma 3 12B es una versión más grande y potente de la familia de modelos Gemma de Google. Con 12 mil millones de parámetros, este modelo ofrece una mayor capacidad para entender y generar texto complejo. Está diseñado para tareas que requieren una alta precisión y comprensión del contexto. En nuestro contexto, lo usaremos cargando el modelo en un servidor con la potencia de cómputo necesaria, al igual que hacíamos con su hermano menor para el OCR.

■ DeepSeek Coder V2 16B:

- **Año de aparición:** 2024
- **Desarrollador:** *DeepSeek AI (Liang Wenfeng)*.
- **Tipo de modelo:** Modelo de lenguaje especializado en código.
- **Descripción:** DeepSeek Coder V2 16B es un modelo de lenguaje especializado en la generación y comprensión de código. Con 16 mil millones de parámetros, está diseñado para tareas de programación y análisis de código. Es capaz de entender y generar código en múltiples lenguajes de programación, lo que lo hace ideal para la evaluación de ejercicios de programación. Su capacidad para analizar y corregir código permite proporcionar retroalimentación detallada y precisa sobre la calidad y eficiencia del código de los estudiantes, aunque también se puede usar en otros contextos. De nuevo, lo cargaremos en un servidor con potencia de cómputo suficiente para su uso.

9.3.4 Casos de prueba

Para evaluar los distintos OCR y LLM, la propuesta es la siguiente. Se confeccionarán 4 actividades de distinta índole, tres serán de programación (es el objetivo principal de la aplicación) con criterios y lenguajes distintos, y uno más genérico y teórico para probar



que funciona en otras casuísticas distintas al código.

Para cada uno de los distintos casos de prueba, se les aplicará los distintos LLM y OCR que se han listado con anterioridad, de esta forma, podemos compararlos y sacar unas conclusiones al respecto. Para cada caso de prueba, se elaborarán dos soluciones, una positiva, y una negativa, esto servirá también para conocer qué tan estricta puede ser a la hora de evaluar y como aplica los criterios. En las tablas 9.80, 9.81, 9.82y 9.83 vemos los 4 casos propuestos:

CPIA-1 Sumar números de un vector

ID	CPIA-1
Nombre	Test 1
Enunciado	Sumar los números de un vector que contenga los números 1,2,3,4,5
Lenguaje	C++
Criterios	Usar bucle for

Cuadro 9.80: Caso de prueba de IA 1

CPIA-2 Recorrer un vector inversamente

ID	CPIA-2
Nombre	Test 2
Enunciado	Recorrer un vector a la inversa
Lenguaje	C++
Criterios	Imprimir por pantalla el valor de los números del vector

Cuadro 9.81: Caso de prueba de IA 2

CPIA-3 Sumar numeros de un vector

ID	CPIA-3
Nombre	Test 3
Enunciado	Función en python que genere los n primeros números de la sucesión Fibonacci separados por comas
Lenguaje	Python
Criterios	usar una función, controlar si se introduce 0 como parámetro

Cuadro 9.82: Caso de prueba de IA 3

**CPIA-4 Explicar qué es un multiplexor**

ID	CPIA-4
Nombre	Test 4
Enunciado	Explicar en pocas palabras que es un multiplexor en estructuras de computadores
Lenguaje	N/A
Criterios	Claridad, debe en algún momento decir que las entradas son varias y la salida es única.

Cuadro 9.83: Caso de prueba de IA 4

9.3.5 Soluciones propuestas

Como anteriormente se ha detallado, para cada ejercicio se propondrán dos soluciones, adicionalmente, para probar el OCR, la solución positiva, que es la correcta, estará escrita de una forma quizás más legible, y la incorrecta, un poco más ilegible para simular distintas casuísticas. A continuación veremos el texto plano de la solución, una breve explicación de por qué la solución negativa está mal, y la imagen del texto que posteriormente el OCR escaneará simulando que es la solución propuesta por el alumno.

Soluciones para CPIA-1

Solución positiva

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vec = {1, 2, 3, 4, 5};
6     int sum = 0;
7     for (int num : vec) {
8         sum += num;
9     }
10    std::cout << "La suma es:" << sum << std::endl;
11    return 0;
12 }
```



```
#include <iostream>
#include <vector>
int main() {
    std::vector<int> vec = {1, 2, 3, 4, 5};
    int suma = 0;
    for (int num : vec) {
        suma += num;
    }
    std::cout << "La suma es: " << suma << std::endl;
    return 0;
}
```

Figura 9.2: Solución Positiva Test 1

Solución negativa

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vector = {4, 2, 8, 4, 5};
6     int suma = 0;
7     for (int n : vector) {
8         suma = suma + n;
9     }
10    std::cout << "La suma es: " << suma << std::endl;
11    return 0;
12 }
```

```
int main() {
    std::vector<int> vector = {4, 2, 8, 4, 5};
    int suma = 0.0;
    for (int n : vector) {
        suma = suma + num;
    }
    std::cout << "La suma es: " << suma << std::endl;
}
```

Figura 9.3: Solución Negativa Test 1

La solución esta está mal porque el vector inicial contiene valores diferentes a los especificados en el enunciado (4, 2, 8, 4, 5 en lugar de 1, 2, 3, 4, 5). No cumple con el requisito de sumar los números del vector específico dado en el enunciado.



Soluciones para CPIA-2

Solución positiva

```
1 int main() {
2     std::vector<int> vector = {1, 2, 3, 4, 5};
3     std::cout << "Vector invertido:";
4     for (int i = vector.size() - 1; i >= 0; i--) {
5         std::cout << vector[i] << " ";
6     }
7     std::cout << std::endl;
8
9     return 0;
10 }
```

```
int main() {
    std::vector<int> vector = {1, 2, 3, 4, 5}
    std::cout << "vector invertido:"
    for (int i = vector.size() - 1; i >= 0; i--) {
        std::cout << vector[i] << " ";
    }
    return 0
}
```

Figura 9.4: Solución Positiva Test 2

Solución negativa

```
1 int main {
2     std::vector<float> vector = {1, 2, 3, 4, 5};
3     for (int i = vector.size(), i >= 0; i--) {
4         std::cout << "vector invertido" << vector[i] << endl;
5     }
6     return 0;
7 }
```

```
int main {
    std::vector<float> vector = {1, 2, 3, 4, 5}
    for (int i = vector.size(), i >= 0; i--) {
        std::cout << "vector invertido" << vector[i] << endl;
    }
    return 0
}
```

Figura 9.5: Solución Negativa Test 2



Contiene errores de sintaxis que impiden su compilación y no invierte el vector correctamente.

Soluciones para CPIA-3

Solución positiva

```
1 def fibonacci_n_veces(n):
2     if n <= 0:
3         print("0")
4         return
5     elif n == 1:
6         print("0")
7         return
8     else:
9         lista_fib = [0, 1]
10        while len(lista_fib) < n:
11            siguiente_fib = lista_fib[-1] + lista_fib[-2]
12            lista_fib.append(siguiente_fib)
13            print(",".join(map(str, lista_fib)))
14
15 # Ejemplo de uso
16 n = 10
17 fibonacci_n_veces(n)
```

def fibonacci_n_Veces(n):
 if ~~n <= 0~~:
 print("")
 return
 elif n == 1:
 print("0")
 return
 else:
 lista_fib = [0, 1]
 while len(lista_fib) < ~~n~~:
 siguiente_fib = lista_fib[-1] + lista_fib[-2]
 lista_fib.append(siguiente_fib)
 print(",".join(map(str, lista_fib)))

 #Ejemplo de uso
 n=10
 fibonacci_n_Veces(n)

Figura 9.6: Solución Positiva Test 3

Solución negativa



```
1 def fibonacci_n_veces(n):
2     if n == 1:
3         print("0")
4         return
5     else:
6         lista_fib = [0, 1]
7         while len(lista_fib) < n:
8             siguiente = lista_fib[-1] + lista_fib[-2]
9             lista_fib.append(siguiente)
10            print(", " + str(lista_fib))
```

```
def fibonacci_n_veces(n):
    if n==1
        print ("0")
        return
    else:
        lista_fib = [0,1]
        while len(lista_fib)<n:
            siguiente = lista_fib[-1] + lista_fib[-2]
            lista_fib.append(siguiente)
        print(", "+lista_fib))
```

Figura 9.7: Solución Negativa Test 3

No maneja correctamente el caso ‘ $n \leq 0$ ’. Imprime la lista con una coma al principio, y la lista entera entre corchetes, en lugar de los números separados por comas.

Soluciones para CPIA-4

Solución positiva

Un multiplexor es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida.

Un multiplexor es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida

Figura 9.8: Solución Positiva Test 4

Solución negativa



Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar las mismas información a varias señales simultáneamente.

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar la misma información a varias señales simultáneamente.

Figura 9.9: Solución Negativa Test 4

La solución es incorrecta porque en lugar de explicar qué es el multiplexor, explica el demultiplexor.

9.3.6 Pruebas OCR

Una vez tenemos los escenarios preparados, comenzaremos con los tests, para ello a modo de guía citaremos en cada caso de prueba cuál es el texto esperado, y cuál el obtenido, así como la imagen que se envía al OCR y unas conclusiones al respecto, identificando los fallos en caso de haberlos. Hay que recordar que, cada caso obtendrá múltiples soluciones correspondientes a los siguientes sistemas usados para el OCR:

- Qwen2.5-VL 3B
- Qwen2.5-VL 7B
- Azure Vision OCR
- Gemma3 4B

CPIA-1 OCR Caso positivo

Texto esperado

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vec = {1, 2, 3, 4, 5};
6     int sum = 0;
7     for (int num : vec) {
8         sum += num;
9     }
10    std::cout << "La suma es:" << sum << std::endl;
11
12 }
```



Imagen enviada

The image shows handwritten C++ code on a whiteboard. The code defines a main function that initializes a vector with integers 1, 2, 3, 4, 5. It then iterates through the vector, adding each element to a variable 'suma'. Finally, it prints the result 'La suma es:' followed by the value of 'suma' using std::cout.

```
#include <iostream>
#include <vector>
int main() {
    std::vector<int> vec = {1, 2, 3, 4, 5};
    int suma = 0;
    for (int num : vec) {
        suma += num;
    }
    std::cout << "La suma es:" << suma << std::endl;
    return 0;
}
```

Figura 9.10: Imagen enviada CPIA 1 Caso positivo

Textos obtenidos

Qwen2.5-VL 3B

```
1 #include <iostream>
2 #include <vector>
3
4 int main()
5 {
6     std::vector<int> vec = {1, 2, 3, 4, 5};
7     int sum = 0;
8
9     for (int num : vec)
10    {
11         sum += num;
12     }
13
14     std::cout << "The sum is:" << sum << std::endl;
15
16     return 0;
17 }
```

Qwen2.5-VL 7B

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vec = {1, 2, 3, 4, 5};
6     int suma = 0;
7     for (int num : vec) {
8         suma += num;
9     }
10    std::cout << "La suma es:" << suma << std::endl;
11    return 0;
12 }
```

Azure Vision OCR



```
1 # include <iostream>
2 # include <vector >
3
4
5 int main ()
6 std :: vector z ints vec = {1, 2,3, 4, 54 ;
7 int suma=0;
8 for ( int num : vec )
9     Suma += num ;
10    Std :: cont      La sera es: LL suma << std :: endl ;
11 return D;
12 m
```

Gemma3:4B

```
1 #include <iostream>
2 int main() {
3     std::vector<int> vec = {1, 2, 3, 4, 5, 4};
4     int suma = 0;
5     for (int num : vec) {
6         suma += num;
7     }
8     std::cout << suma << std::endl;
9     return 0;
10 }
```

Análisis comparativo para CPIA-1 positivo

- **Qwen2.5-VL 3B:** Este modelo mantiene correctamente la estructura del código, incluyendo los espacios y saltos de línea. La única diferencia notable es que cambia el mensaje de salida de "La suma es: " por "The sum is: ", lo que indica una traducción automática del texto. En general, tiene un desempeño sólido en cuanto a precisión.
- **Qwen2.5-VL 7B:** Este modelo también mantiene la estructura del código sin errores significativos. La única diferencia es el cambio de la variable `sum` por `suma`, lo que parece una corrección al idioma español. Este modelo es el que mejor conserva el formato y contenido original.
- **Azure Vision OCR:** Este modelo presenta errores graves en la transcripción. Introduce múltiples fallos de sintaxis, como espacios incorrectos en `#include <iostream>` y errores en la declaración del vector. Además, la salida de `cout` está completamente alterada. Este modelo tiene un desempeño deficiente en el reconocimiento de código para este caso.
- **Gemma3:4B:** Este modelo omite la inclusión de `<vector>`, lo que es un error importante. Además, modifica la declaración del vector y omite el mensaje de salida completo, lo que cambia el comportamiento esperado del programa. A pesar de que la sintaxis es correcta, el código resultante difiere considerablemente del texto original.

CPIA-1 OCR Caso negativo

Texto esperado



```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vector = {4, 2, 8, 4, 5};
6     int suma = 0;
7     for (int n : vector) {
8         suma = suma + n;
9     }
10    std::cout << "La suma es:" << suma << std::endl;
11    return 0;
12 }
```

Imagen enviada

The image shows handwritten C++ code on a light blue background. The code defines a function `int main()` that initializes a vector `vector` with elements 4, 2, 8, 4, 5. It then iterates through the vector, adding each element to a variable `suma`, which is initialized to 0.0. Finally, it prints the result "La suma es:" followed by the value of `suma` using `std::cout`.

```
int main()
{
    std::vector<int> vector = {4, 2, 8, 4, 5}
    int suma = 0.0
    for (int n: Vec)
        suma = suma + num;
    std::cout << "La suma es:" << suma << std::endl;
}
```

Figura 9.11: Imagen enviada CPIA 1 Caso negativo

Textos obtenidos

Qwen2.5-VL 3B

```
1 int main()
2 {
3     std::vector<int> vec{14, 2, 8, 4};
4     int suma = 0.0;
5
6     for (int n: vec)
7     {
8         suma += n;
9     }
10
11    std::cout << "La suma es:" << suma << std::endl;
12 }
```

Qwen2.5-VL 7B

```
1 int main() {
2     std::vector<int> Vec = {4, 2, 8, 4, 5};
3     int suma = 0;
4     for (int n : Vec) {
5         suma = suma + n;
6     }
}
```



CAPÍTULO 9. PRUEBAS

```
7     std::cout << "La suma es:" << suma << std::endl;
8 }
```

Azure Vision OCR

```
1 int main () {
2     std :: vector<int> Nector = 44, 2,8, 4, 54
3     int suma = 0.0
4     for (int n: vec) {
5         Suma = suma + num ;
6         std :: cout << "La suma es:" La Suma      Std : : ende ;
```

Gemma3:4B

```
1 std::vector<int> vector = {4, 2, 8, 4, 5};
2 int suma = 0;
3 for (int i : vector) {
4     suma = suma + i;
5 }
```

Análisis comparativo para CPIA-2 negativo

- **Qwen2.5-VL 3B:** Este modelo presenta varios errores. Modifica la declaración del vector, cambiando los elementos de `vector` a `vec` y omite un valor (el último 5). Además, cambia la variable `suma` a un tipo flotante `suma = 0.0`, lo cual es incorrecto, ya que se trata de una operación de suma con enteros. Estos errores afectan la fidelidad del código.
- **Qwen2.5-VL 7B:** En este caso, el modelo conserva correctamente la mayoría de la estructura original. Sin embargo, cambia la variable `vector` por `Vec`, lo que no es un error grave, pero puede ser considerado una variación no deseada. Además, la operación de suma se mantiene correctamente, lo que asegura que el cálculo sea correcto. Este modelo tiene un buen desempeño en general.
- **Azure Vision OCR:** Este modelo presenta múltiples errores graves en la transcripción. La declaración del vector tiene errores como `std :: vector cint>Nector = 44, 2,8, 4, 54`, lo que hace que el código sea completamente incorrecto. Además, la sintaxis de `cout` también está muy alterada. Este modelo tiene un desempeño muy bajo debido a los errores sintácticos graves.
- **Gemma3:4B:** Este modelo omite la instrucción `std::cout`, y termina el código sin imprimir la salida, lo que altera el comportamiento esperado del programa. A pesar de que el cálculo de la suma se realiza correctamente, la falta de la impresión de la salida hace que el código no cumpla con la especificación del problema.

CPIA-2 OCR Caso positivo

Texto esperado

```
1 int main() {
2     std::vector<int> vector = {1, 2, 3, 4, 5};
3     std::cout << "Vector invertido:" ;
4     for (int i = vector.size() - 1; i >= 0; i--) {
5         std::cout << vector[i] << " ";
6     }
7     std::cout << std::endl;
```



```
8     return 0;
9 }
```

Imagen enviada

A handwritten copy of the C++ code provided in the previous block. The code defines a function main() that prints the elements of a vector in reverse order. The vector is initialized with values 1, 2, 3, 4, 5. The output is directed to std::cout.

```
int main() {
    std::vector<int> vector = {1, 2, 3, 4, 5};
    std::cout << "vector invertido:";
    for (int i = vector.size() - 1; i >= 0; i--) {
        std::cout << vector[i] << " ";
    }
    return 0
}
```

Figura 9.12: Imagen enviada CPIA 2 Caso positivo

Textos obtenidos

Qwen2.5-VL 3B

```
1 int main() {
2     std::vector<int> vector = {1, 2, 3, 4, 5};
3
4     std::cout << "vector\u00d7invertido:\u00d7";
5
6     for (int i = vector.size() - 1; i >= 0; i--) {
7         std::cout << vector[i] << "\u00d7";
8     }
9
10    return 0;
11 }
```

Qwen2.5-VL 7B

```
1 int main() {
2     std::vector<int> vector = {1, 2, 3, 4, 5};
3     std::cout << "vector\u00d7invertido:\u00d7";
4     for (int i = vector.size() - 1; i >= 0; i--) {
5         std::cout << vector[i] << "\u00d7";
6     }
7     return 0;
8 }
```

Azure Vision OCR

```
1 int main ( ) {
2     std :: vector<int> vector = { 1, 2, 3, 4, 5 };
3     std :: cout << "vector\u00d7invertido:" ;
4     for ( int i = vector.size() - 1; i7=0; i-- ) {
5         Std :: cout << vector [i] L "\u00d7";
6     }
7     2
8     return 0;
9 }
```



Gemma3:4B

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vector = {1, 2, 3, 4, 5};
6     std::cout << "vector is ";
7     for (int i = 0; i < -1; i++) {
8         std::cout << vector[i] << " ";
9     }
10    std::cout << std::endl;
11    return 0;
12 }
```

Análisis comparativo CPIA-2 positivo

- **Qwen2.5-VL 3B:** Este modelo logró capturar el código de manera casi perfecta, con solo una pequeña diferencia en el mensaje impreso: utilizó "vector inverted" en vez de "Vector invertido:". Sin embargo, la estructura del código y la lógica del programa fueron correctamente reconocidas. No hay errores sintácticos, y el código obtenido es completamente funcional con solo un pequeño ajuste en la cadena de salida.
- **Qwen2.5-VL 7B:** Este modelo obtuvo una transcripción casi exacta del código original. A diferencia de su versión de menor tamaño (Qwen2.5-VL 3B), corrigió el mensaje en pantalla a "vector invertido:", lo que lo hace más fiel al texto esperado. No presenta errores sintácticos ni problemas de estructura, lo que lo convierte en uno de los mejores resultados en esta prueba.
- **Azure Vision OCR:** Este modelo tiene errores graves en la sintaxis. En particular, el bucle `for` tiene un error en la condición `i7=0; i-`, que debería ser `i >= 0; i-`. Además, la instrucción `Std :: cout` está mal escrita y se usa un símbolo incorrecto para la concatenación de la cadena ("L" en vez de ""). Estos errores hacen que el código no funcione correctamente.
- **Gemma3:4B:** Este modelo también tiene errores en la estructura del bucle. El bucle `for` usa la condición `i <-1`, lo cual es incorrecto y hace que no se ejecute nunca el cuerpo del bucle, además, de nuevo traduce vector invertido a "vector is" Aunque la estructura básica del programa es correcta, la lógica del bucle es completamente errónea.

CPIA-2 OCR Caso negativo

Texto esperado

```
1 int main {
2     std::vector<float> vector = {1, 2, 3, 4, 5};
3     for (int i = vector.size(), i >=0; i--) {
4         std::cout << "vector invertido" << vector[i] << endl;
5     }
6     return0;
7 }
```



Imagen enviada

```
int main {
    std::vector<float> vector = {1, 2, 3, 4, 5};
    for (int i = vector.size(), i >= 0; i--) {
        std::cout << "vector inverso" << vector[i] << endl;
    }
    return 0
}
```

Figura 9.13: Imagen enviada CPIA 2 Caso negativo

Textos obtenidos

Qwen2.5-VL 3B

```
1 int main()
2 {
3     std::vector<float> vector = {1, 2, 3, 4, 5};
4
5     for (int i = vector.size(), i >= 0; i--)
6     {
7         std::cout << "vectorinverse" << vector[i] << endl;
8     }
9
10    return 0;
11 }
```

Qwen2.5-VL 7B

```
1 int main {
2     std::vector<float> vector = {1, 2, 3, 4, 5};
3     for (int i = vector.size(), i > 0; i--) {
4         std::cout << "vectorinverso" << vector[i] << endl;
5     }
6     return 0
7 }
```

Azure Vision OCR

```
1 ent main
2 std :: vector <float> vector = /1, 2,3, 4,54
3 for ( int i = vector. size ( ) , i > = 0 ; i -- > {
4     std :: cout << "vectorinvertido" << vector [i] << endl ;
5     return D
```

Gemma3:4B

```
1 std::vector<float> vector = {1.2, 7.4, 5.4};
2 for (int i = 0; i < vector.size(); i++) {
3     cout << vector[i] << endl;
4 }
5 return 0;
```

Análisis comparativo de CPIA-2 caso negativo



- **Qwen2.5-VL 3B:** Este modelo ha interpretado la solución casi a la perfección, el único fallo que ha tenido es a la hora de traducir el texto que se imprime por pantalla, ya que sustituye "vector invertido"por "vector inverse".
- **Qwen2.5-VL 7B:** A pesar de ser este modelo mejor que el anterior a priori por la cantidad de parámetros, en este caso lo ha hecho peor, ya que además de cambiar "invertido"por "inverso"cuando se imprime por pantalla, no ha puesto los paréntesis en la función main, un error menor que impedirían que el código compilase.
- **Azure Vision OCR:** Este modelo tiene errores graves de sintaxis que lo hacen completamente inviable. El código está mal estructurado, no hay llaves para la función, el tipo de dato de la función main es *ent*, el vector está mal declarado, y en especial que se retorne D dentro del for hace que esté completamente mal el código.
- **Gemma3:4B:** Más de lo mismo, tiene errores graves, como por ejemplo la ausencia del main, la incorrecta declaración del vector y la incorrecta interpretación de los parámetros del bucle for.

CPIA-3 OCR Caso positivo

Texto esperado

```
1 def fibonacci_n_veces(n):
2     if n <= 0:
3         print("0")
4         return
5     elif n == 1:
6         print("0")
7         return
8     else:
9         lista_fib = [0, 1]
10        while len(lista_fib) < n:
11            siguiente_fib = lista_fib[-1] + lista_fib[-2]
12            lista_fib.append(siguiente_fib)
13            print(", ".join(map(str, lista_fib)))
14
15 # Ejemplo de uso
16 n = 10
17 fibonacci_n_veces(n)
```

Imagen enviada



CAPÍTULO 9. PRUEBAS

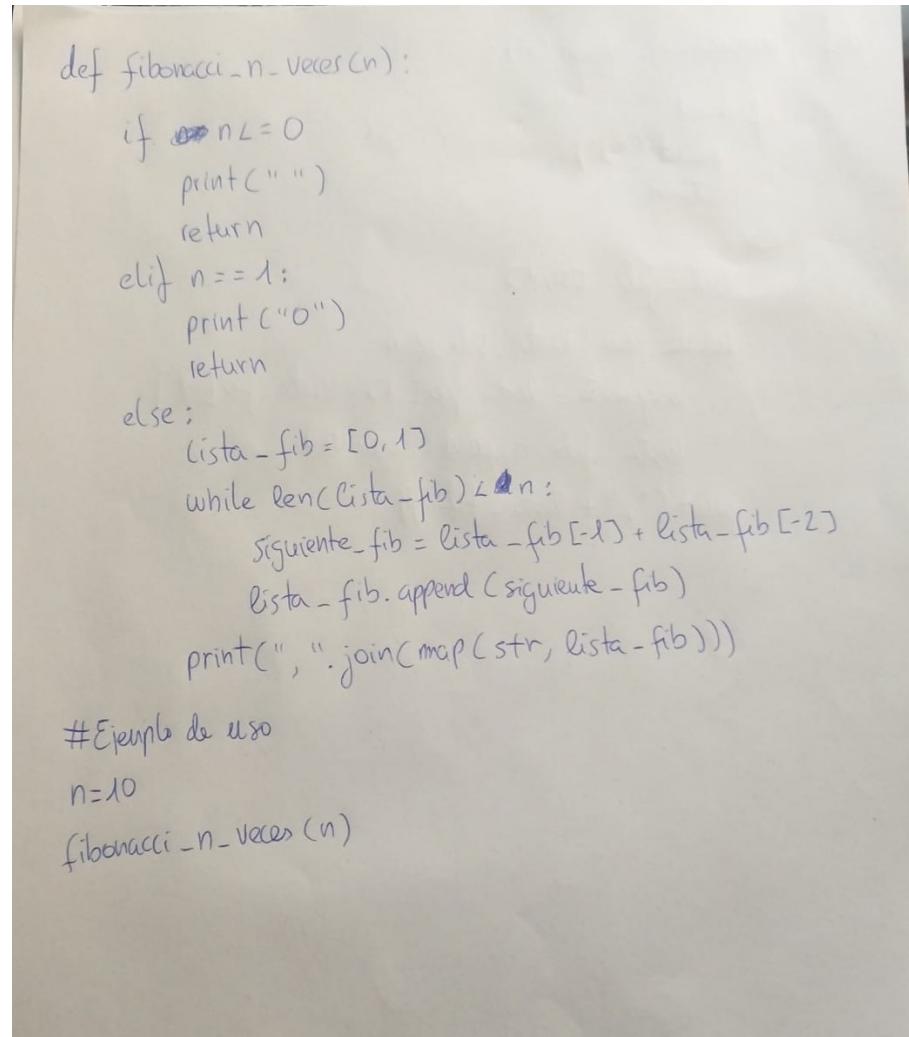


Figura 9.14: Imagen enviada CPIA 3 Caso positivo

Textos obtenidos

Qwen2.5-VL 3B

```
1 def fibonacci_n_veces(n):
2     if n == 0:
3         print("")
4         return
5     elif n == 1:
6         print("0")
7         return
8     else:
9         lista_fib = [0, 1]
10        while len(lista_fib) < n:
11            siguiente_fib = lista_fib[-1] + lista_fib[-2]
12            lista_fib.append(siguiente_fib)
13        print(", ".join(map(str, lista_fib)))
14
15 # Ejemplo de uso
16 n = 10
17 fibonacci_n_veces(n)
```

Qwen2.5-VL 7B



```
1 def fibonacci_n_veces(n):
2     if n <= 0:
3         print("u")
4         return
5     elif n == 1:
6         print("0")
7         return
8     else:
9         lista_fib = [0, 1]
10        while len(lista_fib) < n:
11            siguiente_fib = lista_fib[-1] + lista_fib[-2]
12            lista_fib.append(siguiente_fib)
13            print(", ".join(map(str, lista_fib)))
14
15 # Ejemplo de uso
16 n = 10
17 fibonacci_n_veces(n)
```

Azure Vision OCR

```
1 def fibonacci_n_veces (n) :
2     if onL=0
3         print ("u")
4         return
5     elif n == 1 :
6         print ("0")
7         return
8     else :
9         Lista -fib = [0,1]
10        While len(Cista-fib) 2&n :
11            Siguiente_ fib = lista - fib [-1] + lista - fib [-2]
12            lista - fib. append ( siguiente - fib)
13            print(",u". join(map (str, lista-fib)))
14
15 # Ejemplo de uso
16 n= 10
17 fibonacci_n_veces (n)
```

Gemma3:4B

```
1 def fibonacci_n_valores(n):
2     if n <= 0:
3         print("0")
4         return
5
6     lista_fib = [0, 1]
7
8     while len(lista_fib) < n:
9         siguiente_fib = lista_fib[-1] + lista_fib[-2]
10        lista_fib.append(siguiente_fib)
11
12        print(", ".join(map(str, lista_fib)))
```

Análisis comparativo CPIA-3 caso positivo

- **Qwen2.5-VL 3B:** Este modelo proporciona una versión bastante cercana a la esperada, pero comete un error en la condición del caso base: debería imprimir 0



cuando `n` es 0 o 1, pero en su lugar imprime una cadena vacía en el primer caso. La implementación del ciclo y la lógica para generar la secuencia de Fibonacci están correctamente hechas.

- **Qwen2.5-VL 7B:** El código es funcional en términos generales está bien y no presenta errores, en este caso este modelo es el que mejores resultados ha obtenido.
- **Azure Vision OCR:** Este modelo tiene varios errores de sintaxis que afectan su legibilidad y funcionalidad. La declaración de la función está mal escrita como `fibonacci_ n_ veces` y la variable `lista -fib` está mal formateada. Además, hay varios errores de sintaxis como el uso incorrecto de `While` en lugar de `while` y la asignación de variables con `-` en lugar de `=`. Estos errores hacen que el código no sea ejecutable.
- **Gemma3:4B:** Este modelo ofrece una solución errónea con la que se salta completamente el bloque `elif` y el bloque `else`, por lo que presenta demasiados errores que impedirá que el código compile, por lo que habría que hacer bastantes modificaciones a mano.

CPIA-3 OCR Caso negativo

Texto esperado

```
1 def fibonacci_n_veces(n):
2     if n == 1:
3         print("0")
4         return
5     else:
6         lista_fib = [0, 1]
7         while len(lista_fib) < n:
8             siguiente = lista_fib[-1] + lista_fib[-2]
9             lista_fib.append(siguiente)
10            print(", " + str(lista_fib))
```

Imagen enviada

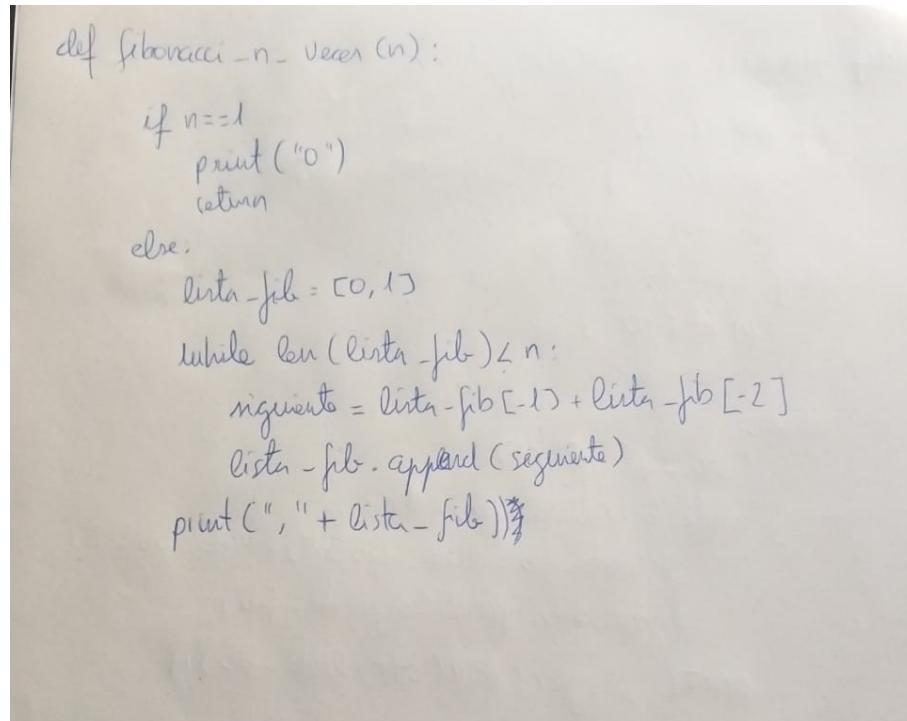


Figura 9.15: Imagen enviada CPIA 3 Caso negativo

Textos obtenidos

Qwen2.5-VL 3B

```
1 def fibonacci_n_vecer(n):
2     if n == 1:
3         print("0")
4         return
5     lista_fib = [0, 1]
6     while len(lista_fib) <= n:
7         siguiente = lista_fib[-1] + lista_fib[-2]
8         lista_fib.append(siguiente)
9     print(", " + lista_fib)
```

Qwen2.5-VL 7B

```
1 def fibonacci_n_vecer(n):
2     if n == 1:
3         print("0")
4         return
5     else:
6         lista_fib = [0, 1]
7         while len(lista_fib) <= n:
8             siguiente = lista_fib[-1] + lista_fib[-2]
9             lista_fib.append(siguiente)
10            print(", " + lista_fib))
```

Azure Vision OCR

```
1 def fibonacci_n_ veces (n) :
2     if n == 1
3         print ("0")
4         return
5     else.
```



CAPÍTULO 9. PRUEBAS

```
6     lista - file = [0,1]
7     while len (lista-fib) < n .
8         siguiente = lista - fib [-1] + lista -fib [-2 ]
9         lister-fil. append ( siguiente )
10    print (",", lista- file ),
```

Gemma3:4B

Análisis comparativo CPIA-3 caso negativo

- **Qwen2.5-VL 3B:** Este modelo realiza una interpretación adecuada en cuanto a la estructura básica del código, pero comete algunos errores. En primer lugar, hay un error tipográfico en el nombre de la función `fibonacci_n_veces`, que aparece como `fibonacci_n_vecer`. Además, la condición del ciclo `while` debería ser `len(lista_fib) < n`, pero el modelo usa `len(lista_fib) <= n`, lo que provoca un problema en la longitud de la secuencia generada.
 - **Qwen2.5-VL 7B:** Este modelo presenta un problema similar al de la versión anterior, con el mismo error tipográfico en el nombre de la función y la misma condición incorrecta en el ciclo `while`, por lo demás, presenta una gran correctitud.
 - **Azure Vision OCR:** Este modelo tiene errores graves de sintaxis. En primer lugar, el nombre de la función está mal escrito como `fibonacci_n_ veces` en lugar de `fibonacci_n_veces`. Además, hay varios errores en la declaración de las variables, como `lista - file` en lugar de `lista_fib` y `lister-fil` en lugar de `lista_fib`. También se encuentra el uso incorrecto de `appleand` en lugar de `append`. Estos errores hacen que el código no sea ejecutable.
 - **Gemma3:4B:** Este modelo parece entrar en un bucle infinito, con una definición de la función `fibonacci_n_veces` que se repite interminablemente. Esto parece ser un error del OCR que ha generado una secuencia larga e incorrecta de caracteres, lo que hace que el código no sea útil ni ejecutable, llegando a dar error en la petición por tardar demasiado.

CPIA-4 OCR Caso positivo

Texto esperado

Un multiplexor es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida.

Imagen enviada



Un multiplexor es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida.

Figura 9.16: Imagen enviada CPIA 4 Caso positivo

Textos obtenidos

Qwen2.5-VL 3B

Un multiplicador es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida.

Qwen2.5-VL 7B

Un multiplexor es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida.

Azure Vision OCR

Un multiplexar es un circuito combinacional con varias\entradas y una única salida de datos. Están dotados\nde entradas de control que sirven para seleccionar una\ny sólo una de las entradas para transmitirla como salida

Gemma3:4B

Un multidecor es un circuito combinacional con varias entradas y una única salida de datos. Están datos para seleccionar una de entrada de control que sirva para seleccionar una de entrada de control que sirva para seleccionar una de entrada de control que ... *Entra en bucle*

Análisis comparativo CPIA-4 caso positivo

- **Qwen2.5-VL 3B:** Este modelo presenta un error en la interpretación de la palabra "multiplexor", que se convierte incorrectamente en "multiplicador". A pesar de este error, el resto del texto es fiel al original, y la estructura de la frase es coherente. El problema principal es la confusión entre términos técnicos, pero el contenido sigue siendo comprensible.
- **Qwen2.5-VL 7B:** Este modelo obtiene una interpretación correcta del término "multiplexor", que es lo esperado. La única diferencia respecto al texto original es la ausencia de la tilde en la palabra "sólo", quitando esto último, la interpretación es perfecta.
- **Azure Vision OCR:** Este modelo también interpreta correctamente el término "multiplexor" y presenta el texto de forma bastante precisa. Sin embargo, hay varios problemas, como la división innecesaria de las palabras en diferentes líneas. Además, la palabra "multiplexar" está mal escrita, cuando debería ser "multiplexor". Este



error técnico impide que el texto sea completamente fiel, aunque el problema de la división mediante los retornos de carro es algo puntual que con solamente borrarlo tendría que ser suficiente.

- **Gemma3:4B:** Este modelo presenta un error considerable, generando el término "multidecor" que es completamente incorrecto y no tiene ningún sentido. Además, parece que entra en un bucle de repetición, lo que hace que el texto sea completamente inútil y no pueda ser interpretado de manera coherente.

CPIA-4 OCR Caso negativo

Texto esperado

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar las mismas información a varias señales simultáneamente.

Imagen enviada

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar la misma información a varias señales simultáneamente.

Figura 9.17: Imagen enviada CPIA 4 Caso negativo

Textos obtenidos

Qwen2.5-VL 3B

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar las mismas información a varias señales distintivamente.

Qwen2.5-VL 7B

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar la misma información a varias líneas simultáneamente.

Azure Vision OCR

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar la misma información a varias líneas simultáneamente.

Gemma3:4B



Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar la misma información a varias de ellas de manera simultánea."

Análisis comparativo CPIA-4 caso negativo

- **Qwen2.5-VL 3B:** Este modelo interpreta correctamente la mayoría del texto, pero comete un error en la frase final, donde cambia "la misma información" por "las mismas información" lo que resulta en un error gramatical menor. Además, la palabra "señales" es seguida por "distintivamente", lo cual cambia el significado del texto original. Este cambio altera el concepto y la coherencia del mensaje ligeramente.
- **Qwen2.5-VL 7B:** Este modelo ofrece una interpretación muy similar al texto original, pero con una pequeña diferencia: cambia "la misma información" por "la misma información" y "señales simultáneamente" por "líneas simultáneamente". Aunque "líneas" podría considerarse aceptable dependiendo del contexto, no es lo mismo que "señales", que es un término más preciso en este contexto y que realmente es lo que se escribe. A pesar de esta pequeña discrepancia, el texto sigue siendo mayormente correcto.
- **Azure Vision OCR:** Este modelo presenta varios errores significativos en la transcripción. En primer lugar, hay un error en la palabra "multiplexor" , que se transcribe como "multiplexor en". Luego, hay múltiples errores tipográficos y ortográficos en el resto del texto, como "miliper" en lugar de "múltiples" y "divisa" en lugar de "divisor". Además, la palabra "misera" es un error grave que cambia completamente el sentido del mensaje original. También se observan problemas de división de palabras en la salida, que afectan la legibilidad.
- **Gemma3:4B:** Este modelo produce una transcripción bastante precisa, con solo un pequeño error en la frase final, donde se dice “a varias de ellas de manera simultánea” en lugar de “a varias señales simultáneamente”. Este error es menor y no cambia significativamente el significado del texto, pero sería preferible mantener la precisión en la terminología tal y como se ha escrito.

Conclusiones finales

Tras analizar los resultados obtenidos de los modelos OCR evaluados en este estudio, se pueden extraer varias conclusiones clave acerca de su rendimiento y aplicabilidad en diferentes contextos. Los resultados han mostrado variabilidad entre los modelos, dependiendo tanto de su arquitectura como de su enfoque técnico (OCR puro versus OCR basado en LLM). A continuación, se resumen las principales conclusiones:

▪ Precisión en el Reconocimiento de Texto:

- **Qwen2.5-VL 7B** mostró la mayor precisión en la mayoría de los casos, especialmente en aquellos con texto técnico o de mayor complejidad. Su tamaño y capacidad para entender el contexto le otorgan ventajas sobre otros modelos, especialmente cuando los textos son largos o tienen vocabulario especializado, detrás le sigue su hermano pequeño, que con menos billones de parámetros consiguió unos resultados muy parecidos.



- **Qwen2.5-VL 3B** demostró tener una gran precisión a pesar de tener menos billones de parámetros que su hermano menor, en términos generales es muy similar al modelo de 7 B de parámetros, por lo que con menos capacidad de cómputo conseguimos un modelo, peor, pero bastante eficiente como relatamos anteriormente
- **Azure Vision OCR** demostró ser muy eficaz en textos más simples y en imágenes de buena calidad, superando a otros modelos en situaciones donde el texto es claro y las imágenes están bien definidas. Sin embargo, su desempeño fue inferior al de los modelos de mayor tamaño en textos más complejos o distorsionados, ya que no dispone de mecanismos de inferencia como anteriormente indiqué para adivinar cuál es el mensaje
- **Gemma3:4B** obtuvo un rendimiento intermedio, siendo efectivo en textos más sencillos pero con fallos en el reconocimiento de términos técnicos o en imágenes con ruido o baja calidad, entrando en algún bucle que impediría el correcto funcionamiento del aplicativo.

■ Desempeño en Texto Técnico:

- Los modelos basados en LLM, como **Qwen2.5-VL 7B** y **Qwen2.5-VL 3B**, mostraron un mejor rendimiento al reconocer textos técnicos o con terminología especializada, gracias a su capacidad para contextualizar las palabras dentro de un marco lingüístico.
- En contraste, **Azure Vision OCR** fue menos preciso con terminología técnica, aunque funcionó bien con texto general. Asimismo **Gemma3:4B** fue el que menos rendimiento mostró de los OCR propuestos

■ Errores Comunes:

- Entre los errores más frecuentes se encontraron confusiones entre caracteres similares (por ejemplo, “O” y “0” o “T” y “1”) y la omisión de letras o palabras en imágenes de baja calidad.
- El modelo **Gemma3:4B**, mostró un comportamiento errático en textos largos, llegando incluso a entrar en bucles, un error crítico que podría mejorarse usando modelos con más parámetros.

Por resumir, los modelos OCR evaluados presentan distintas ventajas y limitaciones según el tipo de texto y la calidad de la imagen. Los modelos basados en LLM, como **Qwen2.5-VL 7B**, destacan en el reconocimiento de textos técnicos y complejos. Por otro lado, **Azure Vision OCR** sigue siendo una opción eficiente para textos estándar en imágenes claras no obstante en cuanto la calidad baja es realmente difícil para el mismo ser preciso, según yo pienso, este OCR puede ser muy preciso para aplicaciones comerciales o industriales, en contextos donde se escanean caracteres impresos en productos o donde la cantidad de caracteres es mucho más baja, es cierto que, como OCR, permite controlar algunos parámetros como la confiabilidad para un posterior tratamiento. Los modelos más pequeños, como **Qwen2.5-VL 3B** y **Gemma3:4B**, ofrecen un desempeño intermedio con algunas limitaciones en textos más difíciles, asimismo, el modelo de Qwen siempre se mostró superior a pesar de su menor número de parámetros y en ningún caso cayó en bucles y salidas muy extrañas como es el caso de Gemma.

Cabe decir que, para aquellos modelos que requieran de una mayor calidad de la imagen, se



conocen distintos procesos de tratamientos de imágenes que pueden mejorar la legibilidad de un texto usando librerías como OpenCV para el tratamiento de dichas imágenes, aunque para el proyecto no está contemplado, sí que podría ser un caso de estudio adicional que podría incorporarse a futuro, por lo que se hablará de ello como una futura mejora a implementar.

9.3.7 Pruebas LLM

Estas pruebas seguirán una estructura análoga a las anteriores de OCR, se evaluará tanto la solución negativa como positiva para el ejercicio propuesto. Hay que tener en consideración los siguientes aspectos:

1. En implementación, se permitirá editar la solución escaneada para que el alumno corrija los posibles errores del OCR.
2. Para las pruebas, se enviará como solución el texto sin errores del OCR, simulando una corrección posterior del alumno.
3. Todos los LLM serán evaluados bajo un mismo prompt que sesgará al modelo para la evaluación.

Al igual que para el OCR, la experiencia y el uso continuado de estos modelos para hacer pruebas, tienen como resultado la construcción de un prompt que minimicen el error y que reduzcan lo máximo posible el error. Al final, no existe el prompt perfecto, y la realidad es que el probar distintos LLMs con distintos prompts sobre distintos ejercicios, puede dar como resultado unas pruebas demasiadas complejas con las que será complicado obtener unas conclusiones de forma sencilla y que el lector pueda comprender mediante una lectura en diagonal, es por ello que he decidido realizar las pruebas con distintos modelos sobre el mismo prompt, un prompt que tendrá información acerca del ejercicio que se evalúa, como su enunciado, el lenguaje, o los criterios. De esta forma, el prompt que se ha usado para las pruebas y se usará para implementación es el siguiente:

Eres un evaluador de actividades, evalúa la siguiente solución y proporciona feedback constructivo, pero muy breve y quiero que también me des la nota de la entrega en formato: Nota: n/10. No uses Markdown, solo texto plano.

La actividad es (título de la actividad) el enunciado es el siguiente: (enunciado de la actividad).

La solución que se proporciona es: (solución de la actividad propuesta).

Recuerda, sé estricto con la nota, no seas tan generoso, si hace algo que no se pide, o no se cumple el enunciado indícalo y disminuye la nota, pero si lo hace bien, no disminuyas la nota y ponle un 10, aunque haya algunos aspectos no muy relevantes a mejorar

Si la actividad especifica un lenguaje de programación: La solución es un fragmento de código, debe estar en (lenguaje de programación). Cuando me des la corrección, nunca me des el código corregido, solo el feedback.

Si la actividad proporciona parámetros de evaluación: Los criterios que tendrás en cuenta para evaluar la solución son: (parámetros de evaluación), si no se cumple el enunciado y los criterios de evaluación indícalo y disminuye la nota.

Como vemos, en el prompt se controlan aspectos como si la actividad es de programación, para indicarle al modelo el lenguaje de programación en el que tiene que estar



la solución, y parámetros de evaluación en caso de que el profesor lo indique. También se le pide que no devuelva el código corregido, esto lo hago para fomentar al alumno a corregirlo por el mismo y evitar una corrección muy verbosa de mucho texto.

CPIA-1 LLM Caso positivo

Solución recibida

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vec = {1, 2, 3, 4, 5};
6     int sum = 0;
7     for (int num : vec) {
8         sum += num;
9     }
10    std::cout << "La suma es: " << sum << std::endl;
11    return 0;
12 }
```



Evaluaciones realizadas para CPIA-1 caso positivo

LLM	Feedback	Nota	Valoración adicional
Gemini	El código es correcto y eficiente. Usa correctamente un bucle for para sumar los elementos del vector.	10/10	Evaluación precisa, reconoce la corrección del código sin sugerencias innecesarias.
ChatGPT 3.5 Turbo	El código proporcionado cumple con el enunciado y los criterios de evaluación. Utiliza un bucle for para sumar los números del vector correctamente. Buena estructura y claridad en el código. ¡Excelente trabajo!	10/10	Correcto en su análisis, destaca la claridad del código y su adecuación al problema.
Gemma3: 12B	El código es correcto y cumple el enunciado usando un bucle for implícito (bucle range-based for). El formato es aceptable y es legible.	10/10	Identifica correctamente la estructura del código como los anteriores
DeepSeek CoderV2: 16B	El código es correcto y cumple con el enunciado, pero podría mejorarse ligeramente para mayor claridad. Por ejemplo, se puede usar un bucle 'for' más estándar como 'for (int i = 0; i < vec.size(); ++i)' o una forma más directa de sumar los elementos del vector usando algoritmos de la biblioteca estándar como 'std::accumulate'.	8/10	Aunque la sugerencia es válida, no es necesaria en este contexto. Penalizar con -2 parece excesivo.

Cuadro 9.84: Evaluaciones de los LLM para CPIA-1 caso positivo

CPIA-1 LLM Caso negativo

Solución recibida

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vector = {4, 2, 8, 4, 5};
6     int suma = 0;
7     for (int n : vector) {
8         suma = suma + n;
9     }
10    std::cout << "La suma es: " << suma << std::endl;
11    return 0;
```



Evaluaciones realizadas para CPIA-1 caso negativo

LLM	Feedback	Nota	Valoración adicional
Gemini	El código compila y usa un bucle for, pero no suma los números del vector indicado en el enunciado (1,2,3,4,5), sino otros. Esto es un error grave que impide la correcta resolución del problema.	4/10	Evaluación precisa. Penaliza correctamente por no seguir el enunciado.
ChatGPT 3.5 Turbo	El código proporcionado cumple con el enunciado y los criterios de evaluación. Utiliza un bucle for para sumar los números del vector correctamente. La solución es clara y directa.	10/10	Error de evaluación. No detecta el problema con los valores del vector.
Gemma3: 12B	El vector inicial no contiene los números 1, 2, 3, 4, 5 solicitados en el enunciado. Aunque la lógica de la suma con el bucle for es correcta, el valor inicial del vector es incorrecto, por lo que no cumple con la premisa del enunciado.	6/10	Correcta observación, pero la nota podría ser más baja para reflejar mejor el error.
DeepSeek CoderV2: 16B	El código tiene algunos errores menores que podrían mejorarse para cumplir con los criterios del enunciado, pero es funcional dentro de lo que se pide. Es importante corregir el vector proporcionado para que contenga los números 1, 2, 3, 4, 5 y asegurarse de usar correctamente la variable 'suma' al inicio del bucle.	7/10	Evaluación intermedia. Reconoce el error pero no lo penaliza tanto como Gemini.

Cuadro 9.85: Evaluaciones realizadas para CPIA-1 caso negativo

CPIA-2 LLM Caso positivo

Solución recibida

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::vector<int> vector = {1, 2, 3, 4, 5};
6     std::cout << "Vector invertido:";
```



```
7     for (int i = vector.size() - 1; i >= 0; i--) {
8         std::cout << vector[i] << " ";
9     }
10    std::cout << std::endl;
11    return 0;
12 }
```

Evaluaciones realizadas para CPIA-2 caso positivo

LLM	Feedback	Nota	Valoración adicional
Gemini	El código recorre e imprime correctamente el vector en orden inverso. Cumple el enunciado.	10/10	Evaluación correcta, pero no detecta el posible error de acceso fuera de límites.
ChatGPT 3.5 Turbo	La solución presentada cumple con el enunciado de recorrer un vector a la inversa e imprimir por pantalla el valor de los números del vector. Sin embargo, hay un pequeño error en el bucle for que puede provocar un comportamiento no deseado, ya que la condición ‘i >= 0’ permitirá que el bucle se ejecute una vez más después de que ‘i’ sea igual a -1, lo cual accedería a un índice fuera de los límites del vector. Se recomienda cambiar la condición a ‘i >0’ para evitar este problema.	8/10	Observación precisa. Detecta un posible bug que otros LLMs pasan por alto. Penalización adecuada.
Gemma3: 12B	La solución cumple el enunciado, recorre el vector de forma invertida e imprime el contenido del mismo. La lógica es clara y el código es legible. No hay elementos que excedan lo pedido.	10/10	Evaluación correcta pero incompleta, ya que no nota el posible acceso fuera de límites.
DeepSeek CoderV2: 16B	El código es correcto y cumple con el objetivo de invertir e imprimir un vector. Sin embargo, considera agregar comentarios para explicar el propósito de cada sección del código, lo que ayudaría a quien lea tu código a entenderlo mejor.	8/10	Evaluación válida, pero no menciona el posible error en el bucle.

Cuadro 9.86: Evaluaciones realizadas para CPIA-2 caso positivo



CPIA-2 LLM Caso negativo

Solución recibida

```
1 int main {
2     std::vector<float> vector = {1, 2, 3, 4, 5};
3     for (int i = vector.size(), i >=0; i--) {
4         std::cout << "vector\u00d7invertido" << vector[i] << endl;
5     }
6     return0;
7 }
```

Evaluaciones realizadas para CPIA-2 caso neagtivo

LLM	Feedback	Nota	Valoración adicional
Gemini	El bucle ‘for’ tiene un error en la condición de parada: ‘i >= 0’ debería ser ‘i >0’ o ‘i >-1’, ya que los índices del vector empiezan en 0. Además, ‘return0’ debería ser ‘return 0;’. El código intentará acceder a un índice fuera de rango, causando un comportamiento indefinido. La salida no es la esperada, según el enunciado.	4/10	Destaca correctamente los errores en el bucle y el ‘return0’, pero no menciona la mala declaración de ‘main’ ni otros errores menores. Evaluación equilibrada.
ChatGPT 3.5 Turbo	El código tiene varios errores: 1. La declaración de la función main debería ser ‘int main()’. 2. En el bucle for, la condición de la iteración debería ser ‘i >= 0’ en lugar de ‘i >=0’. 3. Falta un espacio entre ‘vector invertido’ y ‘vector[i]’ en la salida por consola. 4. Falta incluir el espacio de nombre ‘std’ antes de ‘endl’.	2/10	Evaluación muy detallada en cuanto a errores sintácticos, pero debería haber destacado el problema más crítico: el acceso fuera de los límites del vector.



LLM	Feedback	Nota	Valoración adicional
Gemma3: 12B	El código no compila correctamente. La condición del bucle ‘for’ tiene un error de sintaxis. Además, el bucle está iterando en sentido inverso, pero el acceso al vector ‘vector[i]’ con ‘i’ comenzando por ‘vector.size()’ resultará en un acceso fuera de los límites del vector, lo que provocará un comportamiento indefinido. Finalmente, la impresión no es lo que el enunciado pide, ya que se imprime un texto fijo antes de cada valor.	1/10	Evaluación rigurosa. Penaliza correctamente por la gravedad de los errores. Podría haber mencionado otros detalles menores como la falta de ‘std::endl’.
DeepSeek CoderV2: 16B	El código tiene varios errores importantes que impiden su ejecución correcta. En primer lugar, el bucle ‘for’ tiene una sintaxis incorrecta debido a la falta de punto y coma después del tamaño del vector y la condición inicial. Además, el índice del vector se maneja mal ya que comienza desde el tamaño del vector en lugar de comenzar desde el último elemento hasta el primero. Por último, la impresión por pantalla no sigue correctamente la sintaxis adecuada para acceder a los elementos del vector usando el índice actualizado dentro del bucle.	2/10	Evaluación bien fundamentada. Penaliza correctamente en la línea

Cuadro 9.87: Evaluaciones realizadas para CPIA-2 caso negativo

CPIA-3 LLM Caso positivo

Solución recibida

```
1 def fibonacci_n_veces(n):
2     if n <= 0:
3         print("0")
4         return
5     elif n == 1:
6         print("0")
7         return
8     else:
```



```
9     lista_fib = [0, 1]
10    while len(lista_fib) < n:
11        siguiente_fib = lista_fib[-1] + lista_fib[-2]
12        lista_fib.append(siguiente_fib)
13        print(",".join(map(str, lista_fib)))
14
15 # Ejemplo de uso
16 n = 10
17 fibonacci_n_veces(n)
```

Evaluaciones realizadas para CPIA-3 caso positivo

LLM	Feedback	Nota	Valoración adicional
Gemini	La función utiliza un bucle <code>while</code> ineficiente para generar la secuencia. Maneja correctamente los casos base (<code>n=0, n=1</code>). Sin embargo, la función retorna <code>None</code> , aunque el enunciado no lo requiere explícitamente, pero es una buena práctica retornar la lista generada. La salida es correcta según el enunciado.	9/10	Evaluación precisa. Hace una buena observación sobre la eficiencia y las buenas prácticas, aunque la ineficiencia no afecta la corrección de la solución.
ChatGPT 3.5 Turbo	La solución presentada cumple con los criterios solicitados en el enunciado. La función definida imprime los primeros <code>n</code> números de la sucesión de Fibonacci separados por comas. Se controla si se introduce 0 como parámetro y se maneja correctamente el caso en que <code>n</code> es igual a 1. Sería recomendable agregar comentarios para explicar el funcionamiento del código.	9/10	Evaluación adecuada. Reconoce que el código es correcto y hace una sugerencia útil sobre la documentación, aunque no menciona la posible mejora en eficiencia.
Gemma3: 12B	La función cumple con el enunciado, imprime los <code>n</code> primeros números de la sucesión de Fibonacci separados por comas y utiliza una función como se solicita. También gestiona correctamente el caso de <code>n=0</code> y <code>n=1</code> . El código es legible y relativamente eficiente.	9.5/10	Evaluación justa. Resalta tanto la corrección como la eficiencia del código, aunque no menciona la falta de retorno en la función.



LLM	Feedback	Nota	Valoración adicional
DeepSeek CoderV2: 16B	Buen trabajo al definir una función que maneja correctamente los casos base y utiliza un bucle para calcular la sucesión Fibonacci hasta n elementos. La solución es clara y cumple con el objetivo de imprimir los primeros n números de la sucesión separados por comas. Asegúrate de incluir una comprobación para evitar valores no deseados como 0 o negativos, aunque en este caso específico no se proporcionaron tales entradas.	8/10	Evaluación razonable. La recomendación sobre validaciones es válida, pero la penalización parece excesiva considerando que la entrada ya se controla para $n \leq 0$.

Cuadro 9.88: Evaluaciones realizadas para CPIA-3 caso positivo

CPIA-3 LLM Caso negativo

Solución recibida

```
1 def fibonacci_n_veces(n):
2     if n == 1:
3         print("0")
4         return
5     else:
6         lista_fib = [0, 1]
7         while len(lista_fib) < n:
8             siguiente = lista_fib[-1] + lista_fib[-2]
9             lista_fib.append(siguiente)
10            print(", " + str(lista_fib))
```

Evaluaciones realizadas para CPIA-3 caso negativo

LLM	Feedback	Nota	Valoración adicional
Gemini	El código no maneja correctamente el caso de entrada $n=0$. Imprime una lista con la coma al principio, lo cual no se pide en el enunciado. La función debería retornar la cadena en lugar de imprimirla directamente.	6/10	Evaluación correcta. Identifica errores clave, aunque no menciona la falta de validación para valores negativos.



LLM	Feedback	Nota	Valoración adicional
ChatGPT 3.5 Turbo	La solución presentada es correcta en cuanto a la creación de una función que imprime los primeros n números de la sucesión de Fibonacci separados por comas. Sin embargo, hay un pequeño error al imprimir la lista en formato incorrecto. Debería ser <code>",".join(map(str, lista_fib))</code> en lugar de <code>,- str(lista_fib)</code> . Además, sería recomendable agregar un control para el caso en que se ingrese <code>n = 0</code> para evitar un bucle infinito.	8/10	Evaluación detallada. Destaca el problema en la impresión y sugiere una mejora en la validación, aunque no menciona explícitamente la falta de manejo de valores negativos.
Gemma3: 12B	La función no imprime los números separados por comas, sino una lista con los números. Además, la implementación para <code>n=1</code> es incorrecta, debe imprimir "0" retornar, como indica el enunciado. Finalmente, no controla el caso en que <code>n</code> sea menor que 0.	6/10	Evaluación precisa. Identifica correctamente los errores en la impresión y en la gestión de casos límite.
DeepSeek CoderV2 :16B	El código define una función que calcula correctamente los primeros números de la sucesión Fibonacci hasta el <code>n</code> -ésimo término, pero tiene un error tipográfico en la última línea donde se intenta imprimir la lista separados por comas. Además, no controla el caso cuando ' <code>n</code> ' es 0 o negativo, lo cual podría mejorarse para proporcionar una salida más informativa al usuario.	6/10	Evaluación justa. Resalta los errores principales, pero podría haber sido más crítica con la mala impresión de la lista.

Cuadro 9.89: Evaluaciones realizadas para CPIA-3 caso negativo

CPIA-4 LLM Caso positivo

Solución recibida

Un multiplexor es un circuito combinacional con varias entradas y una única salida de datos. Están dotados de entradas de control que sirven para seleccionar una y sólo una de las entradas para transmitirla como salida.



Evaluaciones realizadas para CPIA-4 caso positivo

LLM	Feedback	Nota	Valoración adicional
Gemini	La respuesta define correctamente un multiplexor como un circuito combinacional con varias entradas y una única salida, explicando la función de las entradas de control. Sin embargo, la explicación podría ser más concisa.	9/10	Evaluación acertada. Reconoce que la definición es correcta, pero sugiere mejorar la concisión sin afectar la claridad.
ChatGPT 3.5 Turbo	La definición proporcionada es clara y concisa, cumple con el enunciado al mencionar que un multiplexor tiene varias entradas y una única salida, además de las entradas de control. Se podría mejorar al agregar un ejemplo para ilustrar su funcionamiento.	9/10	Evaluación adecuada. La sugerencia de agregar un ejemplo es válida, aunque la respuesta ya es suficientemente clara.
Gemma3: 12B	La respuesta es correcta y cumple con el enunciado. Explica de manera clara y concisa la función de un multiplexor. No incluye información innecesaria.	10/10	Evaluación justa. Reconoce que la respuesta es precisa y completa sin requerir mejoras.
DeepSeek CoderV2: 16B	La respuesta proporcionada cubre adecuadamente los aspectos clave del concepto de multiplexor en estructuras de computadores. Sin embargo, podría mejorarse para asegurar una comprensión más completa por parte del lector. Aquí está la mejora sugerida: "Un multiplexor es un circuito electrónico que tiene varias entradas y una salida única, controlada por señales de entrada que seleccionan cuál de las entradas será transmitida a la salida."	9/10	Evaluación interesante. Sugiere una reformulación más detallada, pero la respuesta original ya cumple con el enunciado.

Cuadro 9.90: Evaluaciones realizadas para CPIA-4 caso positivo



CPIA-4 LLM Caso negativo

Solución recibida

Un multiplexor es un dispositivo que toma una señal de entrada y la divide en múltiples señales de salida. Es como un divisor de señal que permite enviar las mismas información a varias señales simultáneamente.

Evaluaciones realizadas para CPIA-4 caso negativo

LLM	Feedback	Nota	Valoración adicional
Gemini	La solución describe un divisor de señal, no un multiplexor. Un multiplexor selecciona UNA de varias entradas y la envía a UNA salida. La respuesta es incorrecta.	3/10	Evaluación acertada. Detecta claramente el error conceptual principal.
ChatGPT 3.5 Turbo	La definición proporcionada es clara y precisa, sin embargo, es importante mencionar que un multiplexor toma varias señales de entrada y selecciona una de ellas para enviarla a la salida. Falta indicar que la salida es única.	8/10	Evaluación incorrecta. No detecta que la respuesta describe mal el concepto.
Gemma3: 12B	La explicación es comprensible, pero introduce la idea del "divisor de señal" que no es precisa. Falta resaltar que el multiplexor selecciona una de varias entradas para enviarla a una única salida, no divide la señal. La descripción actual confunde el concepto.	6/10	Evaluación razonable. Identifica el problema, pero podría haber sido más estricta con la calificación.
DeepSeek CoderV2: 16B	La respuesta explica correctamente lo que es un multiplexor en términos generales, pero podría mejorarse para incluir claramente cómo las entradas múltiples son seleccionadas para una salida única. Además, no menciona específicamente el control de la selección de entrada, lo cual es importante en los multiplexores reales.	8/10	Evaluación incorrecta. No señala que la respuesta está mal conceptualizada.

Cuadro 9.91: Evaluaciones realizadas para CPIA-4 caso negativo



9.3.8 Resumen de Evaluaciones

En la siguiente tabla se resumen las notas otorgadas por cada modelo en los distintos casos evaluados:

Caso	Gemini	ChatGPT 3.5 Turbo	Gemma3:12B	DSCoderV2:16B
CPIA-1 Positivo	10/10	10/10	10/10	8/10
CPIA-1 Negativo	4/10	10/10	6/10	7/10
CPIA-2 Positivo	10/10	8/10	10/10	8/10
CPIA-2 Negativo	4/10	2/10	1/10	2/10
CPIA-3 Positivo	9/10	9/10	9.5/10	8/10
CPIA-3 Negativo	6/10	8/10	6/10	6/10
CPIA-4 Positivo	9/10	9/10	10/10	9/10
CPIA-4 Negativo	3/10	8/10	6/10	8/10

Cuadro 9.92: Resumen de calificaciones otorgadas por cada LLM

Como vemos son puntuaciones bastante similares, aunque en algunos casos, hay algunas disonancias bastante grandes del modelo de OpenAI como es el caso CPIA-1 Negativo que da un 10 a algo que está bastante mal, igual con DeepSeek, que parece ser que por lo general suele ser más estricto, no otorgando ningún 10 pero, en el caso negativo número 4, otorgando una nota demasiado alta para algo que está bastante mal explicado, aunque, hay que tener en cuenta que es un modelo que a priori está concebido para la programación. En general, vemos que los modelos que ofrecen más consistencia para la evaluación son **Gemini** y **Gemma3:12B**

9.3.9 Porcentaje de errores detectados en casos negativos

Metodología para el cálculo

Para determinar el porcentaje de errores detectados en los casos negativos, se siguió la siguiente metodología:

1. **Identificación de errores por cada caso negativo:** Se analizaron los casos negativos CPIA-1, CPIA-2, CPIA-3 y CPIA-4, identificando los errores principales en cada solución incorrecta.
2. **Lista de errores esperados por caso:**

- **CPIA-1 (Caso negativo - C++)**

- Vector con valores incorrectos.

Total: 1 error.

- **CPIA-2 (Caso negativo - C++)**

- Declaración incorrecta de `main`.
 - Error en la condición del bucle `for`.
 - Acceso fuera de los límites del vector.



- Falta de `std::endl` en la impresión.

Total: 4 errores.

- **CPIA-3 (Caso negativo - Python)**

- No maneja el caso `n=0`.
- Formato incorrecto en la impresión de la lista.
- Uso incorrecto de `return` (debería devolver la lista).
- No maneja valores negativos de `n`.

Total: 4 errores.

- **CPIA-4 (Caso negativo - Definición)**

- Confusión entre multiplexor y divisor de señal.
- No menciona que el multiplexor selecciona una única entrada.
- No menciona el rol de las señales de control.

Total: 3 errores.

3. **Cálculo del porcentaje de errores detectados:** Se contabilizó cuántos errores detectó cada LLM y se utilizó la siguiente fórmula para determinar su porcentaje de aciertos:

$$\text{Porcentaje} = \left(\frac{\text{Errores detectados}}{\text{Errores totales}} \right) \times 100 \quad (9.1)$$

Resultados obtenidos

La siguiente tabla muestra el porcentaje de detección de errores por cada LLM:

LLM	Errores detectados	Errores esperados	Porcentaje de detección
Gemini	12	12	100 %
Gemma3:12B	11	12	91.6 %
DeepSeekCoderV2:16B	9	12	75 %
ChatGPT 3.5 Turbo	8	12	66.67 %

Cuadro 9.93: Porcentaje de detección de errores en los casos negativos

Análisis de los resultados

Los resultados obtenidos reflejan la capacidad de cada LLM para identificar problemas en las soluciones incorrectas:

- **Gemini** es el modelo que detectó la mayor cantidad de errores, el 100 % de los mismos
- **Gemma3:12B** tuvo un desempeño similar, aunque ligeramente inferior, logrando un 91.6 %.
- **DeepSeekCoderV2:16B** mostró un desempeño moderado, detectando el 75 % de los errores esperados.
- **ChatGPT 3.5 Turbo** tuvo el rendimiento más bajo, alcanzando un 66.67 %.



Conclusión

El análisis muestra que **Gemini** y **Gemma3:12B** son los LLM más efectivos en la detección de errores en soluciones incorrectas, con tasas de detección superiores al 90 %. En contraste, **DeepSeekCoderV2:16B** y **ChatGPT 3.5 Turbo** tuvieron un desempeño menor, lo que indica que podrían no ser la mejor opción para tareas de revisión crítica de código. Teniendo en cuenta que **Gemma3:12B** es un LLM alojado en local, el hecho de que obtenga unas prestaciones tan similares a las que se obtienen con Gemini es un plus para el caso en el que prioricemos la privacidad y no se quiera externalizar nada.

Capítulo 10

Conclusiones



Este capítulo va dedicado a extraer las conclusiones obtenidas a lo largo de la realización de este proyecto, apuntando los objetivos cumplidos y las futuras mejoras que podrían tomarse de cada a un futuro desarrollo y mantenimiento del proyecto presente.

10.1 Cumplimiento de los objetivos

En la sección presente, realizaremos una análisis de los objetivos propuestos al principio del presente documento analizando el cumplimiento o no de los mismos y apuntando algún breve comentario al respecto. Para ello, nos remontamos a la sección 3, en la cual exponíamos los objetivos que perseguíamos durante el desarrollo del TFG. Para ello, mediante la tabla 10.1 y la tabla 10.2 en las siguientes secciones podemos ver dicho análisis.

10.1.1 Objetivos de la aplicación móvil

Cuadro 10.1: Cumplimiento de objetivos de la aplicación móvil

Objetivo	Cumplimiento y Cómo
Autenticación	Se implementó un sistema de autenticación seguro que permite el inicio, cierre de sesión y registro de usuarios, así como la recuperación de la contraseña, siguiendo buenas prácticas de seguridad.
Roles	La aplicación muestra información personalizada basada en roles (Profesor y Alumno) y datos del usuario, en función del mismo, se permitirá usar unas funcionalidades u otras.
Asignaturas	Se permite la gestión completa de asignaturas (creación, edición, eliminación) para profesores, así como la gestión y la inscripción para alumnos.
Inscripciones	Se implementó un sistema de código de acceso para proteger inscripciones y la funcionalidad para que los profesores expulsen alumnos.
Actividades	Los profesores pueden crear actividades personalizadas con enunciado, lenguaje de programación, fecha límite y criterios de evaluación. Los alumnos pueden ver esta información y realizar una posterior entrega.

Continúa en la siguiente página



<i>Continúa en la siguiente página</i>	
Objetivo	Cumplimiento y Cómo
Entregas	Los alumnos pueden realizar entregas dentro del tiempo límite y reciben retroalimentación y calificación inmediata gracias a la implementación de inteligencia artificial, todo esto es editable por el profesor.
Experiencia de Usuario	Se mejoró la usabilidad con iconos descriptivos y paletas de colores. Se siguió un desarrollo mantenable y escalable.

10.1.2 Objetivos de la Inteligencia Artificial aplicada

Cuadro 10.2: Cumplimiento de objetivos de la Inteligencia Artificial aplicada

Objetivo	Cumplimiento y Cómo
Estudiar y analizar flujos de IA	Se analizaron diversos flujos de aplicación de IA para resolver el problema, hasta finalmente adoptar uno que se separa en dos fases o etapas.
Distinguir dos etapas	Se distinguió y se implementó un sistema de dos etapas: reconocimiento de caracteres y corrección de soluciones. Esto permite al alumno escanear su ejercicio mediante un OCR y corregir los posibles problemas para a posteriori corregir el mismo mediante inteligencia artificial.
Explorar alternativas y modelos	Se exploraron y compararon diferentes modelos de OCR y LLM, argumentando pros y contras de cada uno de ellos, y realizando un análisis empírico de los resultados obtenidos, dejando abierta la posibilidad de implementar casi cualquier modelo.

10.2 Futuras mejoras

En esta sección se enumerarán distintas características que se podrían considerar de cara a futuros desarrollos, estableciendo de esta forma un roadmap que permitirá el desarrollo continuado de este proyecto a lo largo del tiempo.



10.2.1 Aplicación móvil

Algunas mejoras que se podrían implementar en la aplicación móvil son:

■ **Integración de notificaciones push:**

- Implementar notificaciones push para informar a los alumnos sobre nuevas actividades, fechas límite de entrega y calificaciones.
- Permitir a los profesores enviar notificaciones a los alumnos sobre cambios en las asignaturas o actividades.

■ **Gestión del contenido de la asignatura:**

- Añadir un sistema para la gestión del temario, al igual que lo hacen otros CMS, de forma que el profesor pueda suministrarle a los alumnos apuntes o recursos útiles.

■ **Calendario integrado:**

- Integrar un calendario que muestre las fechas límite de entrega de las actividades y otros eventos importantes.
- Permitir a los alumnos sincronizar el calendario con sus dispositivos móviles o cuentas de terceros como Google.

■ **Gamificación:**

- Introducir elementos de gamificación, como insignias y tablas de clasificación, para motivar a los alumnos.
- Permitir a los profesores crear desafíos y recompensas para fomentar la participación.

■ **Accesibilidad:**

- Mejorar la accesibilidad de la aplicación para usuarios con discapacidades, siguiendo las pautas de accesibilidad web (WCAG).
- Añadir soporte para lectores de pantalla y otras tecnologías de asistencia.

■ **Integración con plataformas de videoconferencia:**

- Integrar la aplicación con plataformas de videoconferencia como Zoom o Google Meet para facilitar las clases virtuales.
- Permitir a los profesores programar y realizar clases virtuales directamente desde la aplicación.

10.2.2 Inteligencia artificial aplicada

■ **Mejora de la precisión del OCR:**

- Explorar modelos de OCR más avanzados y entrenarlos con un conjunto de datos más amplio y diverso.



- Implementar técnicas de preprocesamiento de imágenes para mejorar la calidad de las imágenes antes del OCR.
- **Generación automática de actividades:**
 - Utilizar LLM para generar automáticamente actividades personalizadas para cada alumno, basadas en su nivel de conocimiento y estilo de aprendizaje.
 - Permitir a los profesores personalizar las actividades generadas automáticamente en base a los contenidos subidos para el temario.
- **Análisis del rendimiento del alumno:**
 - Utilizar LLM para analizar el rendimiento de los alumnos a lo largo del tiempo e identificar áreas de mejora.
 - Proporcionar a los profesores información sobre el progreso de cada alumno y sugerencias para mejorar su rendimiento y comunicarlo seguidamente.
- **Personalización de la IA:**
 - Permitir al profesor tener más personalización del prompt para la corrección de las actividades para conseguir un sesgo más consistente.
 - Permitir al profesor para cada asignatura decidir qué modelo de OCR y qué modelo de inteligencia artificial usar para la transcripción y evaluación de las actividades en tiempo de ejecución y funcionamiento.

10.3 Desafíos y conclusiones finales

A lo largo de este proyecto, he tenido la oportunidad de explorar y aplicar tecnologías de vanguardia en el ámbito de la educación, combinando el desarrollo de una aplicación móvil con la implementación de inteligencia artificial. Este viaje ha sido tanto desafiante como gratificante, permitiéndome profundizar en áreas como el reconocimiento óptico de caracteres y el procesamiento del lenguaje natural, y cómo estas pueden ser aplicadas para mejorar los procesos de enseñanza y aprendizaje.

La implementación de un sistema de autenticación seguro, la gestión de roles y asignaturas, y la creación de un entorno interactivo para la entrega y evaluación de actividades han sido objetivos clave que hemos logrado alcanzar. La aplicación móvil, tal como se ha desarrollado, ofrece una plataforma robusta y flexible que puede adaptarse a las necesidades de profesores y alumnos, facilitando la comunicación y la gestión de actividades académicas.

Sin embargo, uno de los aspectos más innovadores y desafiantes de este proyecto ha sido la integración de la inteligencia artificial en el proceso de evaluación. La capacidad de escanear ejercicios mediante OCR y recibir retroalimentación y calificaciones casi instantáneas gracias a los LLM representa un avance significativo en la forma en que concebimos la evaluación en el ámbito educativo. La exploración y comparación de diferentes modelos de OCR y LLM me ha permitido apreciar el potencial de estas tecnologías y cómo pueden ser adaptadas para mejorar la precisión y eficiencia de la evaluación.



CAPÍTULO 10. CONCLUSIONES

A pesar de los logros alcanzados, soy consciente de que este proyecto es solo el comienzo. Las futuras mejoras propuestas, como la integración de notificaciones push, la gamificación y la personalización de la IA, representan oportunidades emocionantes para seguir desarrollando y expandiendo las capacidades de esta plataforma.

En última instancia, este proyecto ha sido una experiencia de aprendizaje invaluable. Me ha permitido no solo aplicar mis conocimientos técnicos, sino también desarrollar habilidades de resolución de problemas, pensamiento crítico y creatividad. Estoy convencido de que la combinación de aplicaciones móviles y inteligencia artificial tiene el potencial de transformar la educación, haciéndola más accesible, personalizada y eficiente.

Bibliografía

- [1] Universidad de Córdoba. *Grado Ingeniería Informática*. es-ES. Publisher: Universidad de Córdoba. URL: <https://www.uco.es/eps/es/titulaciones/gr-ing-informatica> (visitado 03-03-2025).
- [2] ¿Qué es una API de RESTful? - Explicación de API de RESTful - AWS. es-ES. URL: <https://aws.amazon.com/es/what-is/restful-api/> (visitado 23-04-2025).
- [3] Tiempo de vida útil. es. Page Version ID: 162563881. Sep. de 2024. URL: https://es.wikipedia.org/w/index.php?title=Tiempo_de_vida_%C3%BAtil&oldid=162563881 (visitado 03-03-2025).
- [4] Protocolo seguro de transferencia de hipertexto. es. Page Version ID: 164656853. Ene. de 2025. URL: https://es.wikipedia.org/w/index.php?title=Protocolo_seguro_de_transferencia_de_hipertexto&oldid=164656853 (visitado 04-03-2025).
- [5] BOE-A-2018-16673 Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. URL: <https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673> (visitado 04-03-2025).
- [6] Moodle. es. Page Version ID: 164994095. Ene. de 2025. URL: <https://es.wikipedia.org/w/index.php?title=Moodle&oldid=164994095> (visitado 04-03-2025).
- [7] Moodle.org: Announcing Moodle LMS 4.5: Unlocking the power of AI / Moodle.org. es. Oct. de 2024. URL: <https://moodle.org/mod/forum/discuss.php?d=462651> (visitado 04-03-2025).
- [8] AI for Educators / MagicSchool AI. en. URL: <https://www.magicschool.ai/> (visitado 04-03-2025).
- [9] Evaluación automática. es. URL: <https://mathew.ai/docentes/correccion-automatica> (visitado 23-03-2025).
- [10] yeelen. Tesseract OCR: ¿Qué es y por qué lo deberías elegir? es-ES. Ene. de 2024. URL: <https://www.klippa.com/es/blog/informativo/que-es-tesseract-ocr/> (visitado 04-03-2025).
- [11] Google Cloud. “Google Cloud Vision OCR”. En: (2021). URL: <https://cloud.google.com/vision/docs/ocr>.
- [12] Microsoft Azure. “Azure Computer Vision OCR”. En: (2021). URL: <https://learn.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-ocr>.
- [13] Jaided AI. “EasyOCR - Open Source Optical Character Recognition”. En: (2021). URL: <https://github.com/JaidedAI/EasyOCR>.



BIBLIOGRAFÍA

- [14] Baidu. “PaddleOCR - Optical Character Recognition with Deep Learning”. En: (2021). URL: <https://github.com/PaddlePaddle/PaddleOCR> (visitado 03-04-2025).
- [15] Federico Ricciuti. “How to Compare OCR Tools: Tesseract OCR vs Amazon Textract vs Azure OCR vs Google OCR”. En: *Medium* (2021). URL: <https://ricciuti-federico.medium.com/how-to-compare-ocr-tools-tesseract-ocr-vs-amazon-textract-vs-azure-ocr-vs-google-ocr-ba3043b507c1> (visitado 03-04-2025).
- [16] FUNSD. URL: <https://guillaumeaume.github.io/FUNSD/> (visitado 29-05-2025).
- [17] Ashish Vaswani et al. “Attention is all you need”. En: *Advances in neural information processing systems* 30 (2017).
- [18] ¿Qué es la IA GPT? - Explicación de los transformadores generativos preentrenados - AWS. es-ES. URL: <https://aws.amazon.com/es/what-is/gpt/> (visitado 23-03-2025).
- [19] Yúbal Fernández. *Google Gemini: qué es, cómo funciona, diferencias con GPT y cuándo podrás usar este modelo de inteligencia artificial*. es. Section: basics. Feb. de 2024. URL: <https://www.xataka.com/basics/google-gemini-que-como-funciona-diferencias-gpt-cuando-podras-usar-este-modelo-inteligencia-artificial> (visitado 03-04-2025).
- [20] Precios de la API para desarrolladores de Gemini / Gemini API. es-419-x-mtfrom-en. URL: <https://ai.google.dev/gemini-api/docs/pricing?hl=es-419> (visitado 03-04-2025).
- [21] Deepsek: Qué es, cómo funciona y en qué se diferencia con ChatGPT. es. URL: <https://www.linkedin.com/pulse/deepsek-qu%C3%A9-es-c%C3%B3mo-funciona-y-en-se-diferencia-con-chatgpt-onlera-ogqbf> (visitado 03-04-2025).
- [22] Joan Piferrer y Cyberclick. Qwen: qué es y cómo funciona el modelo generativo de IA de Alibaba. es. URL: <https://www.cyberclick.es/numerical-blog/qwen-que-es-y-como-funciona-el-modelo-generativo-de-ia-de-alibaba> (visitado 03-04-2025).
- [23] Sistemas operativos de dispositivos móviles: cuota de mercado mundial 2024. es. URL: <https://es.statista.com/estadisticas/635202/sistemas-operativos-de-telefonos-moviles-cuota-de-mercado-mundial/> (visitado 04-03-2025).
- [24] ¿Qué es Visual Studio Code y cuáles son sus ventajas? / Blog de Arsys. es. URL: <https://www.arsys.es/blog/que-es-visual-studio-code-y-cuales-son-sus-ventajas> (visitado 23-03-2025).
- [25] proun_dev. React Native vs Flutter: Comparativa. es-ES. Abr. de 2024. URL: <https://www.proun.es/blog/react-native-vs-flutter-comparativa/> (visitado 04-03-2025).
- [26] Python vs JavaScript: Disputando el futuro del desarrollo / OpenWebinars. es. URL: <https://openwebinars.net/blog/python-vs-javascript/> (visitado 05-03-2025).
- [27] Balazs Refi. FastAPI vs Flask: Key Differences, Performance & Use Cases. en-US. Nov. de 2024. URL: <https://bluebirdinternational.com/fastapi-vs-flask/> (visitado 05-03-2025).
- [28] FastAPI vs Flask - The Complete Guide. URL: <https://christophergs.com/python/2021/06/16/python-flask-fastapi/> (visitado 05-03-2025).



- [29] Ainhoa Lafuente. *Bases de datos relacionales vs. no relacionales: ¿qué es mejor?* es. Ago. de 2018. URL: <https://aukera.es/blog/bases-de-datos-relacionales-vs-no-relacionales/> (visitado 05-03-2025).
- [30] Visure Solutions. *Requisitos funcionales y no funcionales (con ejemplos)*. es. URL: <https://visuresolutions.com/es/gu%C3%A3da-de-trazabilidad-de-gesti%C3%B3n-de-requisitos/requisitos-funcionales-vs-no-funcionales/> (visitado 05-03-2025).
- [31] IBM Product Master 12.0.0. es-ES. Ene. de 2025. URL: <https://www.ibm.com/docs/es/product-master/12.0.0?topic=processes-defining-use-cases> (visitado 09-03-2025).
- [32] CRUD - Conecta Magazine. es. Sep. de 2024. URL: <https://www.conectasoftware.com/magazine/glosario/crud/> (visitado 13-03-2025).
- [33] Qué es MVC. es. URL: <https://desarrolloweb.com/articulos/que-es-mvc.html> (visitado 14-03-2025).
- [34] Daniel Herrera Sánchez. *Clean Architecture en Flutter — ¿Cómo implementarla en el 2023?* en. Mayo de 2023. URL: <https://medium.com/bancolombia-tech/clean-architecture-en-flutter-c%C3%B3mo-implementarla-en-el-2023-34f4d28d06b5> (visitado 15-03-2025).
- [35] Mobterest Studio. *Mastering Feature-First Architecture: Building Scalable Flutter Mobile Apps.* en. Feb. de 2024. URL: <https://mobterest.medium.com/mastering-feature-first-architecture-building-scalable-flutter-mobile-apps-5c706b6e42be> (visitado 15-03-2025).
- [36] Daniel Herrera Sánchez. *Flutter Provider: Qué es, para qué sirve y cómo utilizarlo.* en. Mar. de 2022. URL: <https://medium.com/bancolombia-tech/flutter-provider-qu%C3%A9-es-para-qu%C3%A9-sirve-y-c%C3%B3mo-utilizarlo-7388d4b206b5> (visitado 15-03-2025).
- [37] Yúbal Fernández. *Qué es Ollama y cómo usarlo para instalar en tu ordenador modelos de inteligencia artificial como Llama, DeepSeek y más.* es. Section: basics. Ene. de 2025. URL: <https://www.xataka.com/basics/que-ollama-como-usarlo-para-instalar-tu-ordenador-modelos-inteligencia-artificial-como-llama-deepseek> (visitado 13-04-2025).
- [38] mssaperla. *¿Qué son los Hugging Face Transformers? - Azure Databricks.* es-es. URL: <https://learn.microsoft.com/es-es/azure/databricks/machine-learning/train-model/huggingface/> (visitado 13-04-2025).
- [39] Uvicorn: ASGI web server implementation for Python. Disponible en: <https://www.uvicorn.org/> [Accedido: 13 de abril de 2025]. 2024. URL: <https://www.uvicorn.org/>.
- [40] tiagofur. *Cómo Utilizar los Colores del Tema en Flutter y Dart /.* es. Section: Flutter - Basico. Abr. de 2024. URL: <https://creapolis.dev/como-utilizar-los-colores-del-tema-en-flutter-y-dart/> (visitado 15-03-2025).
- [41] ¿Qué son las pruebas unitarias? / Digital.ai. en_US. URL: <https://digital.ai/es/glossary/what-is-unit-testing/> (visitado 15-03-2025).