

GRADO EN INGENIERÍA INFORMÁTICA DE
GESTIÓN Y SISTEMAS DE INFORMACIÓN

TRABAJO FIN DE GRADO

MOODEX: APLICACIÓN DE PLANIFICACIÓN Y ESTADÍSTICAS PARA EGELA



Estudiante: Rodríguez Cermeño, Jon

Director/Directora: Villamañe Gironés, Mikel

Curso: 2022-2023

Fecha: Barakaldo, 26, julio, 2023

RESUMEN

Con la llegada de Internet han nacido plataformas web dedicadas al soporte de la docencia. ¿Qué utilidad le puede sacar el profesorado a este tipo de plataformas? Por medio de estas plataformas el profesorado puede enviar tareas, mensajes, información, incluso realizar pruebas con el alumnado sin necesidad de estar reunidos en un mismo lugar, así también, si el alumnado tiene dudas acerca de un tema puede preguntar por medio de esta plataforma.

eGela¹ es el nombre otorgado por la UPV/EHU², a la plataforma Moodle³ que se utiliza como soporte a la docencia. Moodle almacena mucha información que, bien usada, podría servir, por ejemplo, para planificar la fecha de una entrega (en base a las fechas de entregas de otras asignaturas) o para comprobar el uso que se está haciendo del aula (por ejemplo, qué tanto por ciento del alumnado accede a cada curso al menos una vez a la semana). En este proyecto se pretende desarrollar una aplicación (moodex: moodle data exploitation) que explote toda esa información y permita realizar una planificación de los cursos y realizar informes de uso.

¹ eGela: <https://egela.ehu.eus/>

² UPV/EHU (Universidad del País Vasco): <https://www.ehu.eus/es/>

³ Moodle: <https://moodle.org/?lang=es>

ÍNDICE DE CONTENIDOS

INTRODUCCIÓN	10
PLANTEAMIENTO INICIAL	11
Objetivos	11
Alcance	12
Ciclo de vida	12
Estructura de Descomposición del Trabajo	13
Descripción de los Paquetes de Trabajo	16
Planificación temporal	54
Diagrama de Gantt	54
Herramientas	60
Herramientas hardware	60
Herramientas software	60
Gestión de riesgos	63
Análisis	66
Evaluación económica	83
Coste de desarrollo	83
Coste del hardware empleado	83
Coste de las licencias del software	84
Costes indirectos	84
Coste total	85
Modelos de negocio	85
ANTECEDENTES	86
Situación actual	86
Estudio de las diferentes alternativas existentes	86
UBUMonitor	86
Plugins	87
Estudio de los distintos enfoques del problema a solucionar	87
Tecnologías a utilizar	87
Tipo de aplicación a desarrollar	87
ARQUITECTURA	89
CAPTURA DE REQUISITOS	90
Requisitos del proyecto	90
Jerarquía de actores	91
Modelo de casos de uso	91
Modelo de dominio	92
ANÁLISIS Y DISEÑO	93
Estructura del backend	93
Capa de seguridad	94
Capa del núcleo	95
Estructura del frontend	97
Diagrama de clases	101
DESARROLLO	104

Desarrollo del backend	104
Estructura de la aplicación	104
Rutas y proceso de comunicación	106
Planificación para los docentes de Moodle	112
Seguridad en la parte servidor	115
Desarrollo del frontend	116
Estructura de la aplicación	116
Funcionamiento y proceso de comunicación	120
Planificación para los docentes de Moodle	123
Seguridad en la parte cliente	125
VERIFICACIÓN Y EVALUACIÓN	126
Pruebas del backend	126
Pruebas del frontend	135
CONCLUSIONES Y TRABAJO FUTURO	142
Objetivos cumplidos	142
Planificación final	144
Riesgos y sus apariciones	145
Evaluación económica final	145
Líneas futuras	146
Reflexión personal	147
BIBLIOGRAFÍA	149
ANEXO I.- CASOS DE USO EXTENDIDOS	151
Iniciar sesión	151
Obtener cursos	154
Ver calendario	155
Cerrar sesión	157
ANEXO II.- DIAGRAMAS DE SECUENCIA	158
Iniciar sesión	158
Obtener cursos	159
Ver calendario	159
Cerrar sesión	160

ÍNDICE DE FIGURAS

Fig. 1.- Ciclo de vida incremental (1)	12
Fig. 2.- EDT de Moodex	14
Fig. 3.- EDT del Planteamiento inicial	15
Fig. 4.- EDT de cada prototipo	16
Fig. 5.- Diagrama de Gantt de las funciones básicas	55
Fig. 6.- Diagrama de Gantt del proyecto completo	56
Fig. 7.- Arquitectura de la aplicación	89
Fig. 8.- Jerarquía de actores	91
Fig. 9.- Modelo de casos de uso	92
Fig. 10.- Estructura del backend	94
Fig. 11.- Capa de seguridad	95
Fig. 12.- Capa del núcleo	97
Fig. 13.- Estructura de unos componentes de Angular	98
Fig. 14.- Estructura de un módulo de Angular	99
Fig. 15.- Esquema general de la aplicación web	100
Fig. 16.- Diagrama de clases	101
Fig. 17.- Estructura básica del backend	105
Fig. 18.- Carpeta modules	105
Fig. 19.- Carpeta work_tools	106
Fig. 20.- Estructura completa del backend	106
Fig. 21.- Esqueleto base de una aplicación Angular	117
Fig. 22.- Estructura básica de la carpeta app	118
Fig. 23.- Estructura del frontend en Angular	120
Fig. 24.- Servicios en Angular	122
Fig. 25.- Cursos del docente	124
Fig. 26.- Calendario del curso	125
Fig. 27.- Bloques de pruebas en Postman	126
Fig. 28.- Ejemplo petición en Postman	127
Fig. 29.- Pantalla de inicio	152
Fig. 30.- Mensaje de error	152
Fig. 31.- Pantalla principal del administrador	153
Fig. 32.- Pantalla principal del docente	153
Fig. 33.- Calendario del curso	156
Fig. 34.- Botón Cerrar sesión	157

ÍNDICE DE TABLAS

Tabla 1.- Descripción del Paquete de Trabajo 1 - Introducción	17
Tabla 2.- Descripción del Paquete de Trabajo 2.1 - Objetivos	17
Tabla 3.- Descripción del Paquete de Trabajo 2.2 - Alcance	18
Tabla 4.- Descripción del Paquete de Trabajo 2.3 - Planificación temporal	18
Tabla 5.- Descripción del Paquete de Trabajo 2.4 - Herramientas	19
Tabla 6.- Descripción del Paquete de Trabajo 2.5 - Gestión de riesgos	19
Tabla 7.- Descripción del Paquete de Trabajo 2.6 - Evaluación económica	20
Tabla 8.- Descripción del Paquete de Trabajo 3 - Antecedentes	20
Tabla 9.- Descripción del Paquete de Trabajo 4.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo I - Iniciar sesión	21
Tabla 10.- Descripción del Paquete de Trabajo 4.1.2 - Casos de uso extendidos del Prototipo I - Iniciar sesión	21
Tabla 11.- Descripción del Paquete de Trabajo 4.1.3 - Captura de requisitos del Prototipo I - Iniciar sesión	22
Tabla 12.- Descripción del Paquete de Trabajo 4.2.1 - Diagrama de clases del Prototipo I - Iniciar sesión	22
Tabla 13.- Descripción del Paquete de Trabajo 4.2.2 - Diagramas de secuencia del Prototipo I - Iniciar sesión	23
Tabla 14.- Descripción del Paquete de Trabajo 4.2.3 - Análisis y diseño del Prototipo I - Iniciar sesión	23
Tabla 15.- Descripción del Paquete de Trabajo 4.3.1 - Implementación del backend del Prototipo I - Iniciar sesión	24
Tabla 16.- Descripción del Paquete de Trabajo 4.3.2 - Implementación del frontend del Prototipo I - Iniciar sesión	24
Tabla 17.- Descripción del Paquete de Trabajo 4.3.3 - Desarrollo del Prototipo I - Iniciar sesión	25
Tabla 18.- Descripción del Paquete de Trabajo 4.4.1 - Pruebas del backend del Prototipo I - Iniciar sesión	25
Tabla 19.- Descripción del Paquete de Trabajo 4.4.2 - Pruebas del frontend del Prototipo I - Iniciar sesión	26
Tabla 20.- Descripción del Paquete de Trabajo 4.4.3 - Verificación y evaluación del Prototipo I - Iniciar sesión	26
Tabla 21.- Descripción del Paquete de Trabajo 5.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo II - Listar cursos	27
Tabla 22.- Descripción del Paquete de Trabajo 5.1.2 - Casos de uso extendidos del Prototipo II - Listar cursos	27
Tabla 23.- Descripción del Paquete de Trabajo 5.1.3 - Captura de requisitos del Prototipo II - Listar cursos	28
Tabla 24.- Descripción del Paquete de Trabajo 5.2.1 - Diagrama de clases del Prototipo II - Listar cursos	28
Tabla 25.- Descripción del Paquete de Trabajo 5.2.2 - Diagramas de secuencia del Prototipo II - Listar cursos	29
Tabla 26.- Descripción del Paquete de Trabajo 5.2.3 - Análisis y diseño del Prototipo II - Listar cursos	29
Tabla 27.- Descripción del Paquete de Trabajo 5.3.1 - Implementación del backend del Prototipo II - Listar cursos	30

Tabla 28.- Descripción del Paquete de Trabajo 5.3.2 - Implementación del frontend del Prototipo II - Listar cursos	30
Tabla 29.- Descripción del Paquete de Trabajo 5.3.3 - Desarrollo del Prototipo II - Listar cursos	31
Tabla 30.- Descripción del Paquete de Trabajo 5.4.1 - Pruebas del backend del Prototipo II - Listar cursos	31
Tabla 31.- Descripción del Paquete de Trabajo 5.4.2 - Pruebas del frontend del Prototipo II - Listar cursos	32
Tabla 32.- Descripción del Paquete de Trabajo 5.4.3 - Verificación y evaluación del Prototipo II - Listar cursos	32
Tabla 33.- Descripción del Paquete de Trabajo 6.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo III - Mostrar calendario	33
Tabla 34.- Descripción del Paquete de Trabajo 6.1.2 - Casos de uso extendidos del Prototipo III - Mostrar calendario	33
Tabla 35.- Descripción del Paquete de Trabajo 6.1.3 - Captura de requisitos del Prototipo III - Mostrar calendario	34
Tabla 36.- Descripción del Paquete de Trabajo 6.2.1 - Diagrama de clases del Prototipo III - Mostrar calendario	34
Tabla 37.- Descripción del Paquete de Trabajo 6.2.2 - Diagramas de secuencia del Prototipo III - Mostrar calendario	35
Tabla 38.- Descripción del Paquete de Trabajo 6.2.3 - Análisis y diseño del Prototipo III - Mostrar calendario	35
Tabla 39.- Descripción del Paquete de Trabajo 6.3.1 - Implementación del backend del Prototipo III - Mostrar calendario	36
Tabla 40.- Descripción del Paquete de Trabajo 6.3.2 - Implementación del frontend del Prototipo III - Mostrar calendario	36
Tabla 41.- Descripción del Paquete de Trabajo 6.3.3 - Desarrollo del Prototipo III - Mostrar calendario	37
Tabla 42.- Descripción del Paquete de Trabajo 6.4.1 - Pruebas del backend del Prototipo III - Mostrar calendario	37
Tabla 43.- Descripción del Paquete de Trabajo 6.4.2 - Pruebas del frontend del Prototipo III - Mostrar calendario	38
Tabla 44.- Descripción del Paquete de Trabajo 6.4.3 - Verificación y evaluación del Prototipo III - Mostrar calendario	38
Tabla 45.- Descripción del Paquete de Trabajo 7.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo IV - Consulta	39
Tabla 46.- Descripción del Paquete de Trabajo 7.1.2 - Casos de uso extendidos del Prototipo IV - Consulta	39
Tabla 47.- Descripción del Paquete de Trabajo 7.1.3 - Captura de requisitos del Prototipo IV - Consulta	40
Tabla 48.- Descripción del Paquete de Trabajo 7.2.1 - Diagrama de clases del Prototipo IV - Consulta	40
Tabla 49.- Descripción del Paquete de Trabajo 7.2.2 - Diagramas de secuencia del Prototipo IV - Consulta	41
Tabla 50.- Descripción del Paquete de Trabajo 7.2.3 - Análisis y diseño del Prototipo IV - Consulta	41
Tabla 51.- Descripción del Paquete de Trabajo 7.3.1 - Implementación del backend del Prototipo IV - Consulta	42
Tabla 52.- Descripción del Paquete de Trabajo 7.3.2 - Implementación del frontend del	

Prototipo IV - Consulta	42
Tabla 53.- Descripción del Paquete de Trabajo 7.3.3 - Desarrollo del Prototipo IV - Consulta	43
Tabla 54.- Descripción del Paquete de Trabajo 7.4.1 - Pruebas del backend del Prototipo IV - Consulta	43
Tabla 55.- Descripción del Paquete de Trabajo 7.4.2 - Pruebas del frontend del Prototipo IV - Consulta	44
Tabla 56.- Descripción del Paquete de Trabajo 7.4.3 - Verificación y evaluación del Prototipo IV - Consulta	44
Tabla 57.- Descripción del Paquete de Trabajo 8.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo V - Planificar tarea	45
Tabla 58.- Descripción del Paquete de Trabajo 8.1.2 - Casos de uso extendidos del Prototipo V - Planificar tarea	45
Tabla 59.- Descripción del Paquete de Trabajo 8.1.3 - Captura de requisitos del Prototipo V - Planificar tarea	46
Tabla 60.- Descripción del Paquete de Trabajo 8.2.1 - Diagrama de clases del Prototipo V - Planificar tarea	46
Tabla 61.- Descripción del Paquete de Trabajo 8.2.2 - Diagramas de secuencia del Prototipo V - Planificar tarea	47
Tabla 62.- Descripción del Paquete de Trabajo 8.2.3 - Análisis y diseño del Prototipo V - Planificar tarea	47
Tabla 63.- Descripción del Paquete de Trabajo 8.3.1 - Implementación del backend del Prototipo V - Planificar tarea	48
Tabla 64.- Descripción del Paquete de Trabajo 8.3.2 - Implementación del frontend del Prototipo V - Planificar tarea	48
Tabla 65.- Descripción del Paquete de Trabajo 8.3.3 - Desarrollo del Prototipo V - Planificar tarea	49
Tabla 66.- Descripción del Paquete de Trabajo 8.4.1 - Pruebas del backend del Prototipo V - Planificar tarea	49
Tabla 67.- Descripción del Paquete de Trabajo 8.4.2 - Pruebas del frontend del Prototipo V - Planificar tarea	50
Tabla 68.- Descripción del Paquete de Trabajo 8.4.3 - Verificación y evaluación del Prototipo V - Planificar tarea	50
Tabla 69.- Descripción del Paquete de Trabajo 9 - Conclusiones y trabajo futuro	51
Tabla 70.- Descripción del Paquete de Trabajo 10 - Resumen	51
Tabla 71.- Resumen de la duración estimada de los paquetes de trabajo con más peso del proyecto	52
Tabla 72.- Resumen de la duración estimada del proyecto	53
Tabla 73.- Resumen de la duración y del periodo de realización estimado del proyecto	57
Tabla 74.- Probabilidad y porcentaje correspondiente	63
Tabla 75.- Impacto y retraso correspondiente	63
Tabla 76.- Valoración y revisión correspondiente	64
Tabla 77.- Análisis del riesgo - Planificación temporal incorrecta	66
Tabla 78.- Análisis del riesgo - Variación en los requisitos o especificaciones del proyecto	67
Tabla 79.- Análisis del riesgo - Problemas de análisis y diseño	67
Tabla 80.- Análisis del riesgo - Desaparición de alguna librería utilizada	68

Tabla 81.- Análisis del riesgo - Sobrecargar el servidor de peticiones	69
Tabla 82.- Análisis del riesgo - Enfermedad / lesión	70
Tabla 83.- Análisis del riesgo - Cúmulo de trabajos / exámenes	71
Tabla 84.- Análisis del riesgo - Cambio de país de residencia	71
Tabla 85.- Análisis del riesgo - Problemas personales	72
Tabla 86.- Análisis del riesgo - Baja moral y/o desmotivación	72
Tabla 87.- Análisis del riesgo - Estancamiento	73
Tabla 88.- Análisis del riesgo - Carencia de una API necesaria	74
Tabla 89.- Análisis del riesgo - Pérdida de la conexión a Internet	75
Tabla 90.- Análisis del riesgo - Apagones	76
Tabla 91.- Análisis del riesgo - Falta de conocimiento del desarrollador en alguna tecnología	76
Tabla 92.- Análisis del riesgo - Infección de un virus	77
Tabla 93.- Análisis del riesgo - Rotura / Pérdida / Robo del equipo	78
Tabla 94.- Análisis del riesgo - Fallo en el disco duro	79
Tabla 95.- Análisis del riesgo - Cambios / Actualizaciones en las librerías utilizadas	80
Tabla 96.- Análisis del riesgo - Cambios / Actualizaciones en APIs utilizadas	81
Tabla 97.- Análisis del riesgo - Problemas de instalación de software	82
Tabla 98.- Coste de desarrollo	83
Tabla 99.- Coste del portátil	84
Tabla 100.- Costes indirectos y coste total	84
Tabla 101.- Pruebas del bloque Login (I)	127
Tabla 102.- Pruebas del bloque Middleware	129
Tabla 103.- Pruebas del bloque Courses (I)	130
Tabla 104.- Pruebas del bloque Events	131
Tabla 105.- Pruebas del bloque NotFound (I)	134
Tabla 106.- Pruebas del bloque Login (II)	135
Tabla 107.- Pruebas del bloque Session	137
Tabla 108.- Pruebas del bloque Courses (II)	137
Tabla 109.- Pruebas del bloque Calendar	138
Tabla 110.- Pruebas del bloque NotFound (II)	140
Tabla 111.- Resumen de la duración estimada y real de los paquetes de trabajo con más peso del proyecto	144

INTRODUCCIÓN

Con la llegada de Internet han nacido plataformas web dedicadas al soporte de la docencia. ¿Qué utilidad le puede sacar el profesorado a este tipo de plataformas? Por medio de estas plataformas el profesorado puede enviar tareas, mensajes, información, incluso realizar pruebas con el alumnado sin necesidad de estar reunidos en un mismo lugar, así también, si el alumnado tiene dudas acerca de un tema puede preguntar por medio de esta plataforma.

eGela⁴ es el nombre otorgado por la UPV/EHU⁵, a la plataforma Moodle⁶ que se utiliza como soporte a la docencia. Moodle almacena mucha información que, bien usada, podría servir, por ejemplo, para planificar la fecha de una entrega (en base a las fechas de entregas de otras asignaturas) o para comprobar el uso que se está haciendo del aula (por ejemplo, qué tanto por ciento del alumnado accede a cada curso al menos una vez a la semana). En este proyecto se pretende desarrollar una aplicación (moodex: moodle data exploitation) que explote toda esa información y permita realizar una planificación de los cursos y realizar informes de uso.

Para llevar a cabo la aplicación web que implemente las operaciones mencionadas se va a hacer uso de distintos tipos de tecnologías. Entre ellas se encuentran Node.js⁷ que se va a encargar del desarrollo de la parte servidor de la aplicación, que implementa la lógica, y Angular⁸ que se va a encargar del desarrollo de la parte cliente de la aplicación web. Estas dos herramientas están basadas en JavaScript⁹ y TypeScript¹⁰ respectivamente, por lo tanto, también se va a hacer uso de estos lenguajes de programación. Además se van a utilizar técnicas de analítica de aprendizaje para medir, recopilar, analizar e informar sobre datos de los usuarios de la plataforma de docencia.

⁴ eGela: <https://egela.ehu.eus/>

⁵ UPV/EHU (Universidad del País Vasco): <https://www.ehu.eus/es/>

⁶ Moodle: <https://moodle.org/?lang=es>

⁷ Node.js: <https://nodejs.org/es/>

⁸ Angular: <https://angular.io/>

⁹ JavaScript: <https://www.javascript.com/>

¹⁰ TypeScript: <https://www.typescriptlang.org/>

PLANTEAMIENTO INICIAL

Objetivos

Este apartado explica tanto los objetivos funcionales del proyecto, como los objetivos personales y de aprendizaje del desarrollador. Además, cada uno de los objetivos del proyecto pueden ser o no principales, es decir, el proyecto consta de objetivos secundarios cuyo desarrollo se decidirá a lo largo del proyecto.

Los objetivos principales del proyecto son los siguientes:

- Desarrollar una herramienta que proporcione información al profesorado y que pueda servir de ayuda para planificar la docencia y la entrega de tareas. Esto le puede llegar a beneficiar al alumnado porque es más difícil que le coincidan entregas.
- Realizar un trabajo que, al menos, cumpla los requisitos y objetivos mínimos necesarios para aprobar el TFG y que cuente con el visto bueno del tutor y del desarrollador.
- Aplicar los conocimientos obtenidos a lo largo de la carrera con el fin de plasmar sobre el trabajo todo lo aprendido.
- Aprender a utilizar herramientas novedosas para el desarrollador.

Como objetivo secundario se pretenden añadir funcionalidades extra, como pueden ser ampliar las estadísticas disponibles o permitir estadísticas más complejas y facilitar la comunicación entre eGela y la aplicación desarrollada para reducir lo máximo posible la carga de trabajo del profesorado evitándole tener que andar alternando las plataformas.

Alcance

Este apartado explica el tipo de ciclo de vida con el que se va a trabajar y las razones de su elección, la estructura de descomposición del trabajo diseñado y una breve descripción de cada uno de los paquetes de trabajo que lo forman.

Ciclo de vida

A lo largo del desarrollo del proyecto se va a hacer uso del ciclo de vida incremental que se caracteriza por dividir la aplicación en distintos prototipos cuyas funcionalidades se van complementando. Cada prototipo pasa a su vez por distintas fases: análisis, diseño, implementación y pruebas. Una vez el prototipo ha completado con éxito cada una de las fases se da por finalizado y se pasa al siguiente. En la Fig. 1 se muestra el ciclo de vida incremental.

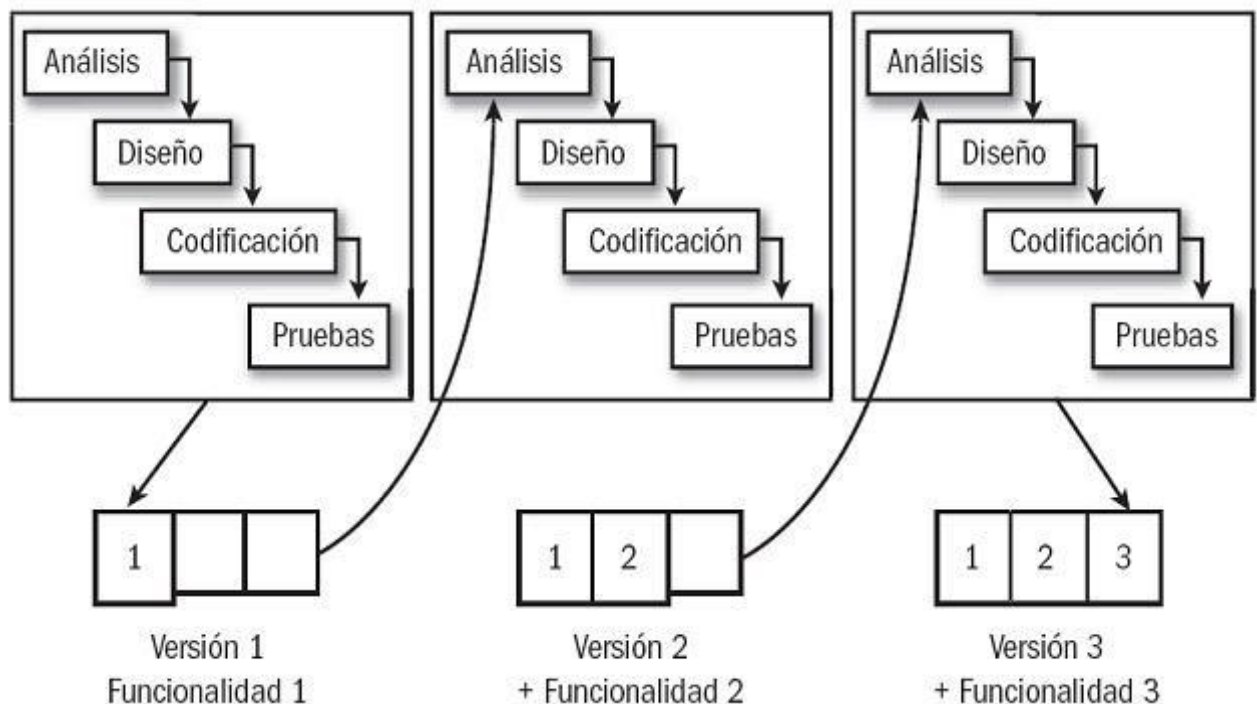


Fig. 1.- Ciclo de vida incremental (1)

La mayoría de los prototipos de la aplicación están relacionados con el objetivo principal y se van a desarrollar con total seguridad. Estos prototipos son los siguientes:

- **Prototipo I - Iniciar sesión:** El prototipo que se va a encargar de proporcionar una herramienta a través de la cual el profesorado y la administración de eGela pueda introducir sus credenciales para iniciar sesión.
- **Prototipo II - Listar cursos:** El prototipo que se va a encargar de mostrar la lista de cursos que imparte al profesorado de eGela

- **Prototipo III - Mostrar calendario:** El prototipo que, una vez el docente haya pulsado sobre un curso de la lista, se va a encargar de mostrarle un calendario donde podrá apreciar mediante distintos colores el porcentaje de alumnos que tienen al menos la entrega de una tarea.

El resto de prototipos están relacionados con los objetivos secundarios, y por tanto, no se tiene la certeza de que se vayan a realizar. Por lo que, al finalizar el desarrollo de los prototipos que son seguros, se analizará si se han cumplido los objetivos y se tendrá en cuenta el tiempo restante para decidir si finalmente se desarrollan o no. Por el momento se incluyen en la planificación del proyecto porque se cuenta con llevarlos a cabo. Los prototipos que no son seguros se describen a continuación:

- **Prototipo IV - Consulta:** El prototipo que se va a encargar de proporcionar una herramienta a través de la cual la administración de eGela pueda realizar una consulta, como puede ser escoger dos fechas y recibir un listado de cursos en los que no ha entrado nadie entre dichas fechas. La herramienta dispondrá de varios filtros que le pueden servir a la administración para diferenciar el uso que le da a la plataforma el profesorado, el alumnado o ambos.
- **Prototipo V - Planificar tarea:** El prototipo que se va a encargar de dar la opción al profesorado de planificar la tarea directamente desde la aplicación.

Hay varias razones por las que se ha elegido este tipo de ciclo de vida respecto a un ciclo de vida clásico. Dividir la aplicación en distintos prototipos simplifica bastante el desarrollo dado que es más sencillo llevar una funcionalidad a cabo que todas a la vez. Asimismo, desarrollar la funcionalidad de identificación del prototipo I servirá para familiarizarse con la tecnología y los lenguajes de programación a utilizar. Una vez desarrollada esta funcionalidad se podrá realizar una mejor y más realista planificación temporal del resto de prototipos, por lo que también se podrá saber con mayor exactitud si se está cumpliendo con la planificación al ritmo deseado. Además, en el caso de cometer algún error es más fácil identificar en qué fase y prototipo se ha cometido.

Estructura de Descomposición del Trabajo

A continuación, se presenta la Estructura de Descomposición del Trabajo (EDT) que se ha diseñado y con la que se va a trabajar de ahora en adelante.

La Fig. 2 muestra el EDT general de Moodex. Las tareas en las que se divide este EDT siguen un orden cronológico, es decir, siguen una sucesión lógica que empieza con la introducción de la memoria y termina con el resumen de esta misma. Entre estas dos tareas también se realiza el planteamiento inicial del proyecto, se hace un estudio de las diferentes alternativas y de los distintos enfoques del problema, se desarrolla cada uno de los prototipos y, finalmente, se sacan conclusiones y se plantean mejoras y ampliaciones.



Fig. 2.- EDT de Moodex

La Fig. 3 muestra el EDT del Planteamiento inicial del proyecto. Este está compuesto por cada uno de los siguientes apartados: objetivos, alcance, planificación temporal, herramientas, gestión de riesgos y evaluación económica.

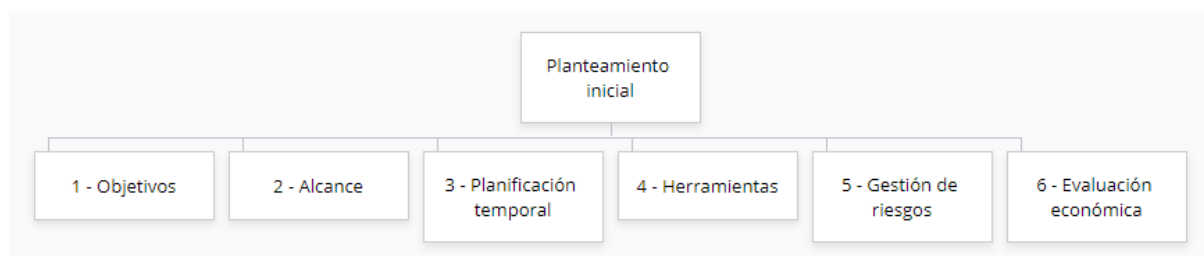


Fig. 3.- EDT del Planteamiento inicial

La Fig. 4 muestra el EDT de cada uno de los prototipos antes mencionados. Todos los prototipos cumplen la misma estructura y pasan por las mismas fases, es decir, las tareas de la figura se repiten por cada prototipo:

- Se empieza con la fase de análisis, en la que se realizan el modelo de casos de uso y la jerarquía de actores y los casos de uso extendidos, y se documenta la captura de requisitos de la memoria.
- En la fase de diseño, se llevan a cabo el diagrama de clases y los diagramas de secuencia, y se documenta el análisis y diseño de la memoria.
- La fase de implementación consiste en desarrollar tanto la parte cliente como la parte servidor de la aplicación, además de documentar en la memoria todo lo realizado.
- Se finaliza con la fase de pruebas, en la que, como su propio nombre indica, se prueban tanto la parte cliente como la parte servidor de la aplicación de manera exhaustiva, asimismo, los resultados se recogen en la memoria.

Cabe decir, que las tareas de documentación, como pueden ser los casos de uso extendidos, la captura de requisitos, los diagramas de secuencia, el análisis y diseño, el desarrollo y la verificación y evaluación, no estarán organizadas por prototipos, es decir, conforme se vayan desarrollando los prototipos se irá avanzando y se irán completando dichos apartados de la memoria.

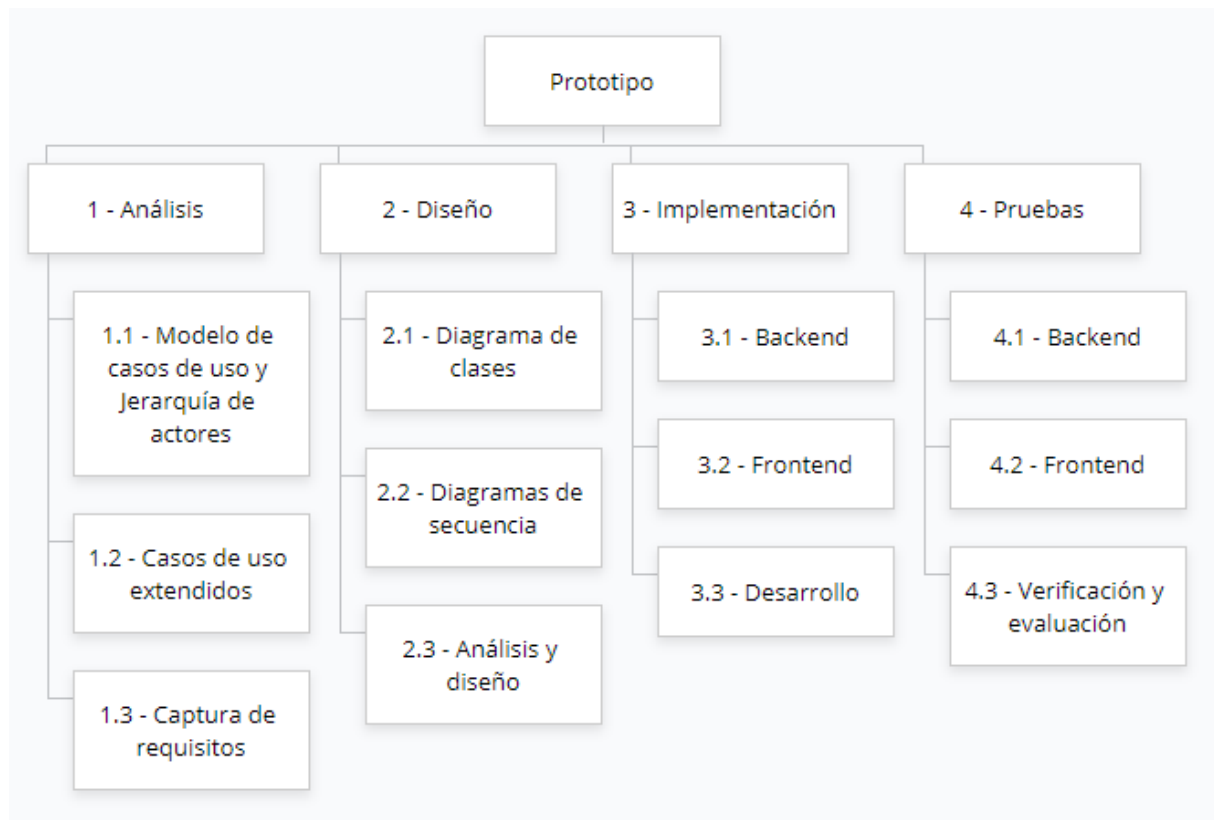


Fig. 4.- EDT de cada prototipo

Descripción de los Paquetes de Trabajo

Una vez identificados los diversos paquetes de trabajo de los que está compuesto el proyecto, hay que describir cada uno de ellos de manera que quede plasmado en qué consiste, cuál es su duración estimada, qué elementos requiere para poder realizarse, qué salidas o entregables genera, qué recursos o herramientas son necesarias para llevarlo a cabo y qué otros paquetes de trabajo le preceden.

Tabla 1.- Descripción del Paquete de Trabajo 1 - Introducción

Moodex	
Paquete de trabajo	1 - Introducción
Descripción	Este paquete de trabajo consiste en redactar el apartado de introducción de la memoria.
Duración estimada	10 horas
Entradas	
Salidas / Entregables	Apartado de introducción de la memoria
Recursos necesarios	Documentos de Google
Precedencias	

Tabla 2.- Descripción del Paquete de Trabajo 2.1 - Objetivos

2 - Planteamiento inicial	
Paquete de trabajo	2.1 - Objetivos
Descripción	Este paquete de trabajo consiste en redactar el apartado de objetivos de la memoria.
Duración estimada	10 horas
Entradas	Apartado de introducción de la memoria
Salidas / Entregables	Apartado de objetivos de la memoria
Recursos necesarios	Documentos de Google
Precedencias	1 - Introducción

Tabla 3.- Descripción del Paquete de Trabajo 2.2 - Alcance

2 - Planteamiento inicial	
Paquete de trabajo	2.2 - Alcance
Descripción	Este paquete de trabajo consiste en redactar el apartado de alcance de la memoria.
Duración estimada	20 horas
Entradas	Apartado de objetivos de la memoria
Salidas / Entregables	Apartado de alcance de la memoria
Recursos necesarios	Documentos de Google y GlocMaps
Precedencias	2.1 - Objetivos

Tabla 4.- Descripción del Paquete de Trabajo 2.3 - Planificación temporal

2 - Planteamiento inicial	
Paquete de trabajo	2.3 - Planificación temporal
Descripción	Este paquete de trabajo consiste en redactar el apartado de planificación temporal de la memoria.
Duración estimada	20 horas
Entradas	Apartado de alcance de la memoria
Salidas / Entregables	Apartado de planificación temporal de la memoria
Recursos necesarios	Documentos de Google y GanttProject
Precedencias	2.2 - Alcance

Tabla 5.- Descripción del Paquete de Trabajo 2.4 - Herramientas

2 - Planteamiento inicial	
Paquete de trabajo	2.4 - Herramientas
Descripción	Este paquete de trabajo consiste en redactar el apartado de herramientas de la memoria.
Duración estimada	10 horas
Entradas	Apartado de planificación temporal de la memoria
Salidas / Entregables	Apartado de herramientas de la memoria
Recursos necesarios	Documentos de Google
Precedencias	2.3 - Planificación temporal

Tabla 6.- Descripción del Paquete de Trabajo 2.5 - Gestión de riesgos

2 - Planteamiento inicial	
Paquete de trabajo	2.5 - Gestión de riesgos
Descripción	Este paquete de trabajo consiste en redactar el apartado de gestión de riesgos de la memoria.
Duración estimada	20 horas
Entradas	Apartado de herramientas de la memoria
Salidas / Entregables	Apartado de gestión de riesgos de la memoria
Recursos necesarios	Documentos de Google
Precedencias	2.4 - Herramientas

Tabla 7.- Descripción del Paquete de Trabajo 2.6 - Evaluación económica

2 - Planteamiento inicial	
Paquete de trabajo	2.6 - Evaluación económica
Descripción	Este paquete de trabajo consiste en redactar el apartado de evaluación económica de la memoria.
Duración estimada	20 horas
Entradas	Apartado de gestión de riesgos de la memoria
Salidas / Entregables	Apartado de evaluación económica de la memoria
Recursos necesarios	Documentos de Google
Precedencias	2.5 - Gestión de riesgos

Tabla 8.- Descripción del Paquete de Trabajo 3 - Antecedentes

Moodex	
Paquete de trabajo	3 - Antecedentes
Descripción	Este paquete de trabajo consiste en redactar el apartado de antecedentes de la memoria.
Duración estimada	10 horas
Entradas	Apartado de evaluación económica de la memoria
Salidas / Entregables	Apartado de antecedentes de la memoria
Recursos necesarios	Documentos de Google
Precedencias	2.6 - Evaluación económica

Tabla 9.- Descripción del Paquete de Trabajo 4.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo I - Iniciar sesión

4.1 - Análisis del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en diseñar el Modelo de casos de uso y la Jerarquía de actores del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Apartado de antecedentes de la memoria
Salidas / Entregables	Modelo de casos de uso y Jerarquía de actores del Prototipo I - Iniciar sesión
Recursos necesarios	Visual Paradigm
Precedencias	3 - Antecedentes

Tabla 10.- Descripción del Paquete de Trabajo 4.1.2 - Casos de uso extendidos del Prototipo I - Iniciar sesión

4.1 - Análisis del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.1.2 - Casos de uso extendidos del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en redactar los Casos de uso extendidos del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Modelo de casos de uso y Jerarquía de actores del Prototipo I - Iniciar sesión
Salidas / Entregables	Casos de uso extendidos del Prototipo I - Iniciar sesión
Recursos necesarios	Documentos de Google
Precedencias	4.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo I - Iniciar sesión

Tabla 11.- Descripción del Paquete de Trabajo 4.1.3 - Captura de requisitos del Prototipo I - Iniciar sesión

4.1 - Análisis del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.1.3 - Captura de requisitos del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en redactar la Captura de requisitos del Prototipo I - Iniciar sesión.
Duración estimada	10 horas
Entradas	Casos de uso extendidos del Prototipo I - Iniciar sesión
Salidas / Entregables	Captura de requisitos del Prototipo I - Iniciar sesión
Recursos necesarios	Documentos de Google
Precedencias	4.1.2 - Casos de uso extendidos del Prototipo I - Iniciar sesión

Tabla 12.- Descripción del Paquete de Trabajo 4.2.1 - Diagrama de clases del Prototipo I - Iniciar sesión

4.2 - Diseño del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.2.1 - Diagrama de clases del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en diseñar el Diagrama de clases del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Captura de requisitos del Prototipo I - Iniciar sesión
Salidas / Entregables	Diagrama de clases del Prototipo I - Iniciar sesión
Recursos necesarios	Visual Paradigm
Precedencias	4.1.3 - Captura de requisitos del Prototipo I - Iniciar sesión

Tabla 13.- Descripción del Paquete de Trabajo 4.2.2 - Diagramas de secuencia del Prototipo I - Iniciar sesión

4.2 - Diseño del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.2.2 - Diagramas de secuencia del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en diseñar los Diagramas de secuencia del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Diagrama de clases del Prototipo I - Iniciar sesión
Salidas / Entregables	Diagramas de secuencia del Prototipo I - Iniciar sesión
Recursos necesarios	Visual Paradigm
Precedencias	4.2.1 - Diagrama de clases del Prototipo I - Iniciar sesión

Tabla 14.- Descripción del Paquete de Trabajo 4.2.3 - Análisis y diseño del Prototipo I - Iniciar sesión

4.2 - Diseño del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.2.3 - Análisis y diseño del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en redactar el Análisis y diseño del Prototipo I - Iniciar sesión.
Duración estimada	10 horas
Entradas	Diagramas de secuencia del Prototipo I - Iniciar sesión
Salidas / Entregables	Análisis y diseño del Prototipo I - Iniciar sesión
Recursos necesarios	Documentos de Google
Precedencias	4.2.2 - Diagramas de secuencia del Prototipo I - Iniciar sesión

Tabla 15.- Descripción del Paquete de Trabajo 4.3.1 - Implementación del backend del Prototipo I - Iniciar sesión

4.3 - Implementación del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.3.1 - Implementación del backend del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en implementar la parte servidor del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Análisis y diseño del Prototipo I - Iniciar sesión
Salidas / Entregables	Parte servidor del Prototipo I - Iniciar sesión
Recursos necesarios	Visual Studio Code, Node.js, Express y JavaScript
Precedencias	4.2.3 - Análisis y diseño del Prototipo I - Iniciar sesión

Tabla 16.- Descripción del Paquete de Trabajo 4.3.2 - Implementación del frontend del Prototipo I - Iniciar sesión

4.3 - Implementación del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.3.2 - Implementación del frontend del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en implementar la parte cliente del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Parte servidor del Prototipo I - Iniciar sesión
Salidas / Entregables	Parte cliente del Prototipo I - Iniciar sesión
Recursos necesarios	Visual Studio Code, Angular, HTML y TypeScript
Precedencias	4.3.1 - Implementación del backend del Prototipo I - Iniciar sesión

Tabla 17.- Descripción del Paquete de Trabajo 4.3.3 - Desarrollo del Prototipo I - Iniciar sesión

4.3 - Implementación del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.3.3 - Desarrollo del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en redactar el Desarrollo del Prototipo I - Iniciar sesión.
Duración estimada	10 horas
Entradas	Parte cliente del Prototipo I - Iniciar sesión
Salidas / Entregables	Desarrollo del Prototipo I - Iniciar sesión
Recursos necesarios	Documentos de Google
Precedencias	4.3.2 - Implementación del frontend del Prototipo I - Iniciar sesión

Tabla 18.- Descripción del Paquete de Trabajo 4.4.1 - Pruebas del backend del Prototipo I - Iniciar sesión

4.4 - Pruebas del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.4.1 - Pruebas del backend del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en probar la parte servidor del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Desarrollo del Prototipo I - Iniciar sesión
Salidas / Entregables	Pruebas de la parte servidor del Prototipo I - Iniciar sesión
Recursos necesarios	Postman y Moodle
Precedencias	4.3.3 - Desarrollo del Prototipo I - Iniciar sesión

Tabla 19.- Descripción del Paquete de Trabajo 4.4.2 - Pruebas del frontend del Prototipo I - Iniciar sesión

4.4 - Pruebas del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.4.2 - Pruebas del frontend del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en probar la parte cliente del Prototipo I - Iniciar sesión.
Duración estimada	4 horas
Entradas	Pruebas de la parte servidor del Prototipo I - Iniciar sesión
Salidas / Entregables	Pruebas de la parte cliente del Prototipo I - Iniciar sesión
Recursos necesarios	Moodle
Precedencias	4.4.1 - Pruebas del backend del Prototipo I - Iniciar sesión

Tabla 20.- Descripción del Paquete de Trabajo 4.4.3 - Verificación y evaluación del Prototipo I - Iniciar sesión

4.4 - Pruebas del Prototipo I - Iniciar sesión	
Paquete de trabajo	4.4.3 - Verificación y evaluación del Prototipo I - Iniciar sesión
Descripción	Este paquete de trabajo consiste en redactar la Verificación y evaluación del Prototipo I - Iniciar sesión.
Duración estimada	10 horas
Entradas	Pruebas de la parte cliente del Prototipo I - Iniciar sesión
Salidas / Entregables	Verificación y evaluación del Prototipo I - Iniciar sesión
Recursos necesarios	Documentos de Google
Precedencias	4.4.2 - Pruebas del frontend del Prototipo I - Iniciar sesión

Tabla 21.- Descripción del Paquete de Trabajo 5.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo II - Listar cursos

5.1 - Análisis del Prototipo II - Listar cursos	
Paquete de trabajo	5.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en diseñar el Modelo de casos de uso y la Jerarquía de actores del Prototipo II - Listar cursos.
Duración estimada	4 horas
Entradas	Verificación y evaluación del Prototipo I - Iniciar sesión
Salidas / Entregables	Modelo de casos de uso y Jerarquía de actores del Prototipo II - Listar cursos
Recursos necesarios	Visual Paradigm
Precedencias	4.4.3 - Verificación y evaluación del Prototipo I - Iniciar sesión

Tabla 22.- Descripción del Paquete de Trabajo 5.1.2 - Casos de uso extendidos del Prototipo II - Listar cursos

5.1 - Análisis del Prototipo II - Listar cursos	
Paquete de trabajo	5.1.2 - Casos de uso extendidos del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en redactar los Casos de uso extendidos del Prototipo II - Listar cursos.
Duración estimada	4 horas
Entradas	Modelo de casos de uso y Jerarquía de actores del Prototipo II - Listar cursos
Salidas / Entregables	Casos de uso extendidos del Prototipo II - Listar cursos
Recursos necesarios	Documentos de Google
Precedencias	5.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo II - Listar cursos

Tabla 23.- Descripción del Paquete de Trabajo 5.1.3 - Captura de requisitos del Prototipo II - Listar cursos

5.1 - Análisis del Prototipo II - Listar cursos	
Paquete de trabajo	5.1.3 - Captura de requisitos del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en redactar la Captura de requisitos del Prototipo II - Listar cursos.
Duración estimada	10 horas
Entradas	Casos de uso extendidos del Prototipo II - Listar cursos
Salidas / Entregables	Captura de requisitos del Prototipo II - Listar cursos
Recursos necesarios	Documentos de Google
Precedencias	5.1.2 - Casos de uso extendidos del Prototipo II - Listar cursos

Tabla 24.- Descripción del Paquete de Trabajo 5.2.1 - Diagrama de clases del Prototipo II - Listar cursos

5.2 - Diseño del Prototipo II - Listar cursos	
Paquete de trabajo	5.2.1 - Diagrama de clases del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en diseñar el Diagrama de clases del Prototipo II - Listar cursos.
Duración estimada	4 horas
Entradas	Captura de requisitos del Prototipo II - Listar cursos
Salidas / Entregables	Diagrama de clases del Prototipo II - Listar cursos
Recursos necesarios	Visual Paradigm
Precedencias	5.1.3 - Captura de requisitos del Prototipo II - Listar cursos

Tabla 25.- Descripción del Paquete de Trabajo 5.2.2 - Diagramas de secuencia del Prototipo II - Listar cursos

5.2 - Diseño del Prototipo II - Listar cursos	
Paquete de trabajo	5.2.2 - Diagramas de secuencia del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en diseñar los Diagramas de secuencia del Prototipo II - Listar cursos.
Duración estimada	4 horas
Entradas	Diagrama de clases del Prototipo II - Listar cursos
Salidas / Entregables	Diagramas de secuencia del Prototipo II - Listar cursos
Recursos necesarios	Visual Paradigm
Precedencias	5.2.1 - Diagrama de clases del Prototipo II - Listar cursos

Tabla 26.- Descripción del Paquete de Trabajo 5.2.3 - Análisis y diseño del Prototipo II - Listar cursos

5.2 - Diseño del Prototipo II - Listar cursos	
Paquete de trabajo	5.2.3 - Análisis y diseño del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en redactar el Análisis y diseño del Prototipo II - Listar cursos.
Duración estimada	10 horas
Entradas	Diagramas de secuencia del Prototipo II - Listar cursos
Salidas / Entregables	Análisis y diseño del Prototipo II - Listar cursos
Recursos necesarios	Documentos de Google
Precedencias	5.2.2 - Diagramas de secuencia del Prototipo II - Listar cursos

Tabla 27.- Descripción del Paquete de Trabajo 5.3.1 - Implementación del backend del Prototipo II - Listar cursos

5.3 - Implementación del Prototipo II - Listar cursos	
Paquete de trabajo	5.3.1 - Implementación del backend del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en implementar la parte servidor del Prototipo II - Listar cursos.
Duración estimada	8 horas
Entradas	Análisis y diseño del Prototipo II - Listar cursos
Salidas / Entregables	Parte servidor del Prototipo II - Listar cursos
Recursos necesarios	Visual Studio Code, Node.js, Express y JavaScript
Precedencias	5.2.3 - Análisis y diseño del Prototipo II - Listar cursos

Tabla 28.- Descripción del Paquete de Trabajo 5.3.2 - Implementación del frontend del Prototipo II - Listar cursos

5.3 - Implementación del Prototipo II - Listar cursos	
Paquete de trabajo	5.3.2 - Implementación del frontend del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en implementar la parte cliente del Prototipo II - Listar cursos.
Duración estimada	8 horas
Entradas	Parte servidor del Prototipo II - Listar cursos
Salidas / Entregables	Parte cliente del Prototipo II - Listar cursos
Recursos necesarios	Visual Studio Code, Angular, HTML y TypeScript
Precedencias	5.3.1 - Implementación del backend del Prototipo II - Listar cursos

Tabla 29.- Descripción del Paquete de Trabajo 5.3.3 - Desarrollo del Prototipo II - Listar cursos

5.3 - Implementación del Prototipo II - Listar cursos	
Paquete de trabajo	5.3.3 - Desarrollo del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en redactar el Desarrollo del Prototipo II - Listar cursos.
Duración estimada	10 horas
Entradas	Parte cliente del Prototipo II - Listar cursos
Salidas / Entregables	Desarrollo del Prototipo II - Listar cursos
Recursos necesarios	Documentos de Google
Precedencias	5.3.2 - Implementación del frontend del Prototipo II - Listar cursos

Tabla 30.- Descripción del Paquete de Trabajo 5.4.1 - Pruebas del backend del Prototipo II - Listar cursos

5.4 - Pruebas del Prototipo II - Listar cursos	
Paquete de trabajo	5.4.1 - Pruebas del backend del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en probar la parte servidor del Prototipo II - Listar cursos.
Duración estimada	4 horas
Entradas	Desarrollo del Prototipo II - Listar cursos
Salidas / Entregables	Pruebas de la parte servidor del Prototipo II - Listar cursos
Recursos necesarios	Postman y Moodle
Precedencias	5.3.3 - Desarrollo del Prototipo II - Listar cursos

Tabla 31.- Descripción del Paquete de Trabajo 5.4.2 - Pruebas del frontend del Prototipo II - Listar cursos

5.4 - Pruebas del Prototipo II - Listar cursos	
Paquete de trabajo	5.4.2 - Pruebas del frontend del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en probar la parte cliente del Prototipo II - Listar cursos.
Duración estimada	4 horas
Entradas	Pruebas de la parte servidor del Prototipo II - Listar cursos
Salidas / Entregables	Pruebas de la parte cliente del Prototipo II - Listar cursos
Recursos necesarios	Moodle
Precedencias	5.4.1 - Pruebas del backend del Prototipo II - Listar cursos

Tabla 32.- Descripción del Paquete de Trabajo 5.4.3 - Verificación y evaluación del Prototipo II - Listar cursos

5.4 - Pruebas del Prototipo II - Listar cursos	
Paquete de trabajo	5.4.3 - Verificación y evaluación del Prototipo II - Listar cursos
Descripción	Este paquete de trabajo consiste en redactar la Verificación y evaluación del Prototipo II - Listar cursos
Duración estimada	10 horas
Entradas	Pruebas de la parte cliente del Prototipo II - Listar cursos
Salidas / Entregables	Verificación y evaluación del Prototipo II - Listar cursos
Recursos necesarios	Documentos de Google
Precedencias	5.4.2 - Pruebas del frontend del Prototipo II - Listar cursos

Tabla 33.- Descripción del Paquete de Trabajo 6.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo III - Mostrar calendario

6.1 - Análisis del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en diseñar el Modelo de casos de uso y la Jerarquía de actores del Prototipo III - Mostrar calendario.
Duración estimada	4 horas
Entradas	Verificación y evaluación del Prototipo II - Listar cursos
Salidas / Entregables	Modelo de casos de uso y Jerarquía de actores del Prototipo III - Mostrar calendario
Recursos necesarios	Visual Paradigm
Precedencias	5.4.3 - Verificación y evaluación del Prototipo II - Listar cursos

Tabla 34.- Descripción del Paquete de Trabajo 6.1.2 - Casos de uso extendidos del Prototipo III - Mostrar calendario

6.1 - Análisis del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.1.2 - Casos de uso extendidos del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en redactar los Casos de uso extendidos del Prototipo III - Mostrar calendario.
Duración estimada	4 horas
Entradas	Modelo de casos de uso y Jerarquía de actores del Prototipo III - Mostrar calendario
Salidas / Entregables	Casos de uso extendidos del Prototipo III - Mostrar calendario
Recursos necesarios	Documentos de Google
Precedencias	6.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo III - Mostrar calendario

Tabla 35.- Descripción del Paquete de Trabajo 6.1.3 - Captura de requisitos del Prototipo III - Mostrar calendario

6.1 - Análisis del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.1.3 - Captura de requisitos del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en redactar la Captura de requisitos del Prototipo III - Mostrar calendario.
Duración estimada	10 horas
Entradas	Casos de uso extendidos del Prototipo III - Mostrar calendario
Salidas / Entregables	Captura de requisitos del Prototipo III - Mostrar calendario
Recursos necesarios	Documentos de Google
Precedencias	6.1.2 - Casos de uso extendidos del Prototipo III - Mostrar calendario

Tabla 36.- Descripción del Paquete de Trabajo 6.2.1 - Diagrama de clases del Prototipo III - Mostrar calendario

6.2 - Diseño del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.2.1 - Diagrama de clases del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en diseñar el Diagrama de clases del Prototipo III - Mostrar calendario.
Duración estimada	4 horas
Entradas	Captura de requisitos del Prototipo III - Mostrar calendario
Salidas / Entregables	Diagrama de clases del Prototipo III - Mostrar calendario
Recursos necesarios	Visual Paradigm
Precedencias	6.1.3 - Captura de requisitos del Prototipo III - Mostrar calendario

Tabla 37.- Descripción del Paquete de Trabajo 6.2.2 - Diagramas de secuencia del Prototipo III - Mostrar calendario

6.2 - Diseño del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.2.2 - Diagramas de secuencia del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en diseñar los Diagramas de secuencia del Prototipo III - Mostrar calendario.
Duración estimada	4 horas
Entradas	Diagrama de clases del Prototipo III - Mostrar calendario
Salidas / Entregables	Diagramas de secuencia del Prototipo III - Mostrar calendario
Recursos necesarios	Visual Paradigm
Precedencias	6.2.1 - Diagrama de clases del Prototipo III - Mostrar calendario

Tabla 38.- Descripción del Paquete de Trabajo 6.2.3 - Análisis y diseño del Prototipo III - Mostrar calendario

6.2 - Diseño del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.2.3 - Análisis y diseño del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en redactar el Análisis y diseño del Prototipo III - Mostrar calendario.
Duración estimada	10 horas
Entradas	Diagramas de secuencia del Prototipo III - Mostrar calendario
Salidas / Entregables	Análisis y diseño del Prototipo III - Mostrar calendario
Recursos necesarios	Documentos de Google
Precedencias	6.2.2 - Diagramas de secuencia del Prototipo III - Mostrar calendario

Tabla 39.- Descripción del Paquete de Trabajo 6.3.1 - Implementación del backend del Prototipo III - Mostrar calendario

6.3 - Implementación del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.3.1 - Implementación del backend del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en implementar la parte servidor del Prototipo III - Mostrar calendario.
Duración estimada	16 horas
Entradas	Análisis y diseño del Prototipo III - Mostrar calendario
Salidas / Entregables	Parte servidor del Prototipo III - Mostrar calendario
Recursos necesarios	Visual Studio Code, Node.js, Express y JavaScript
Precedencias	6.2.3 - Análisis y diseño del Prototipo III - Mostrar calendario

Tabla 40.- Descripción del Paquete de Trabajo 6.3.2 - Implementación del frontend del Prototipo III - Mostrar calendario

6.3 - Implementación del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.3.2 - Implementación del frontend del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en implementar la parte cliente del Prototipo III - Mostrar calendario.
Duración estimada	16 horas
Entradas	Parte servidor del Prototipo III - Mostrar calendario
Salidas / Entregables	Parte cliente del Prototipo III - Mostrar calendario
Recursos necesarios	Visual Studio Code, Angular, HTML y TypeScript
Precedencias	6.3.1 - Implementación del backend del Prototipo III - Mostrar calendario

**Tabla 41.- Descripción del Paquete de Trabajo 6.3.3 - Desarrollo del Prototipo III -
Mostrar calendario**

6.3 - Implementación del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.3.3 - Desarrollo del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en redactar el Desarrollo del Prototipo III - Mostrar calendario.
Duración estimada	10 horas
Entradas	Parte cliente del Prototipo III - Mostrar calendario
Salidas / Entregables	Desarrollo del Prototipo III - Mostrar calendario
Recursos necesarios	Documentos de Google
Precedencias	6.3.2 - Implementación del frontend del Prototipo III - Mostrar calendario

**Tabla 42.- Descripción del Paquete de Trabajo 6.4.1 - Pruebas del backend del
Prototipo III - Mostrar calendario**

6.4 - Pruebas del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.4.1 - Pruebas del backend del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en probar la parte servidor del Prototipo III - Mostrar calendario.
Duración estimada	4 horas
Entradas	Desarrollo del Prototipo III - Mostrar calendario
Salidas / Entregables	Pruebas de la parte servidor del Prototipo III - Mostrar calendario
Recursos necesarios	Postman y Moodle
Precedencias	6.3.3 - Desarrollo del Prototipo III - Mostrar calendario

Tabla 43.- Descripción del Paquete de Trabajo 6.4.2 - Pruebas del frontend del Prototipo III - Mostrar calendario

6.4 - Pruebas del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.4.2 - Pruebas del frontend del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en probar la parte cliente del Prototipo III - Mostrar calendario.
Duración estimada	4 horas
Entradas	Pruebas de la parte servidor del Prototipo III - Mostrar calendario
Salidas / Entregables	Pruebas de la parte cliente del Prototipo III - Mostrar calendario
Recursos necesarios	Moodle
Precedencias	6.4.1 - Pruebas del backend del Prototipo III - Mostrar calendario

Tabla 44.- Descripción del Paquete de Trabajo 6.4.3 - Verificación y evaluación del Prototipo III - Mostrar calendario

6.4 - Pruebas del Prototipo III - Mostrar calendario	
Paquete de trabajo	6.4.3 - Verificación y evaluación del Prototipo III - Mostrar calendario
Descripción	Este paquete de trabajo consiste en redactar la Verificación y evaluación del Prototipo III - Mostrar calendario.
Duración estimada	10 horas
Entradas	Pruebas de la parte cliente del Prototipo III - Mostrar calendario
Salidas / Entregables	Verificación y evaluación del Prototipo III - Mostrar calendario
Recursos necesarios	Documentos de Google
Precedencias	6.4.2 - Pruebas del frontend del Prototipo III - Mostrar calendario

Tabla 45.- Descripción del Paquete de Trabajo 7.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo IV - Consulta

7.1 - Análisis del Prototipo IV - Consulta	
Paquete de trabajo	7.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en diseñar el Modelo de casos de uso y la Jerarquía de actores del Prototipo IV - Consulta.
Duración estimada	4 horas
Entradas	Verificación y evaluación del Prototipo III - Mostrar calendario
Salidas / Entregables	Modelo de casos de uso y Jerarquía de actores del Prototipo IV - Consulta
Recursos necesarios	Visual Paradigm
Precedencias	6.4.3 - Verificación y evaluación del Prototipo III - Mostrar calendario

Tabla 46.- Descripción del Paquete de Trabajo 7.1.2 - Casos de uso extendidos del Prototipo IV - Consulta

7.1 - Análisis del Prototipo IV - Consulta	
Paquete de trabajo	7.1.2 - Casos de uso extendidos del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en redactar los Casos de uso extendidos del Prototipo IV - Consulta.
Duración estimada	4 horas
Entradas	Modelo de casos de uso y Jerarquía de actores del Prototipo IV - Consulta
Salidas / Entregables	Casos de uso extendidos del Prototipo IV - Consulta
Recursos necesarios	Documentos de Google
Precedencias	7.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo IV - Consulta

Tabla 47.- Descripción del Paquete de Trabajo 7.1.3 - Captura de requisitos del Prototipo IV - Consulta

7.1 - Análisis del Prototipo IV - Consulta	
Paquete de trabajo	7.1.3 - Captura de requisitos del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en redactar la Captura de requisitos del Prototipo IV - Consulta.
Duración estimada	10 horas
Entradas	Casos de uso extendidos del Prototipo IV - Consulta
Salidas / Entregables	Captura de requisitos del Prototipo IV - Consulta
Recursos necesarios	Documentos de Google
Precedencias	7.1.2 - Casos de uso extendidos del Prototipo IV - Consulta

Tabla 48.- Descripción del Paquete de Trabajo 7.2.1 - Diagrama de clases del Prototipo IV - Consulta

7.2 - Diseño del Prototipo IV - Consulta	
Paquete de trabajo	7.2.1 - Diagrama de clases del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en diseñar el Diagrama de clases del Prototipo IV - Consulta.
Duración estimada	4 horas
Entradas	Captura de requisitos del Prototipo IV - Consulta
Salidas / Entregables	Diagrama de clases del Prototipo IV - Consulta
Recursos necesarios	Visual Paradigm
Precedencias	7.1.3 - Captura de requisitos del Prototipo IV - Consulta

Tabla 49.- Descripción del Paquete de Trabajo 7.2.2 - Diagramas de secuencia del Prototipo IV - Consulta

7.2 - Diseño del Prototipo IV - Consulta	
Paquete de trabajo	7.2.2 - Diagramas de secuencia del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en diseñar los Diagramas de secuencia del Prototipo IV - Consulta.
Duración estimada	4 horas
Entradas	Diagrama de clases del Prototipo IV - Consulta
Salidas / Entregables	Diagramas de secuencia del Prototipo IV - Consulta
Recursos necesarios	Visual Paradigm
Precedencias	7.2.1 - Diagrama de clases del Prototipo IV - Consulta

Tabla 50.- Descripción del Paquete de Trabajo 7.2.3 - Análisis y diseño del Prototipo IV - Consulta

7.2 - Diseño del Prototipo IV - Consulta	
Paquete de trabajo	7.2.3 - Análisis y diseño del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en redactar el Análisis y diseño del Prototipo IV - Consulta.
Duración estimada	10 horas
Entradas	Diagramas de secuencia del Prototipo IV - Consulta
Salidas / Entregables	Análisis y diseño del Prototipo IV - Consulta
Recursos necesarios	Documentos de Google
Precedencias	7.2.2 - Diagramas de secuencia del Prototipo IV - Consulta

Tabla 51.- Descripción del Paquete de Trabajo 7.3.1 - Implementación del backend del Prototipo IV - Consulta

7.3 - Implementación del Prototipo IV - Consulta	
Paquete de trabajo	7.3.1 - Implementación del backend del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en implementar la parte servidor del Prototipo IV - Consulta.
Duración estimada	16 horas
Entradas	Análisis y diseño del Prototipo IV - Consulta
Salidas / Entregables	Parte servidor del Prototipo IV - Consulta
Recursos necesarios	Visual Studio Code, Node.js, Express y JavaScript
Precedencias	7.2.3 - Análisis y diseño del Prototipo IV - Consulta

Tabla 52.- Descripción del Paquete de Trabajo 7.3.2 - Implementación del frontend del Prototipo IV - Consulta

7.3 - Implementación del Prototipo IV - Consulta	
Paquete de trabajo	7.3.2 - Implementación del frontend del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en implementar la parte cliente del Prototipo IV - Consulta.
Duración estimada	16 horas
Entradas	Parte servidor del Prototipo IV - Consulta
Salidas / Entregables	Parte cliente del Prototipo IV - Consulta
Recursos necesarios	Visual Studio Code, Angular, HTML y TypeScript
Precedencias	7.3.1 - Implementación del backend del Prototipo IV - Consulta

Tabla 53.- Descripción del Paquete de Trabajo 7.3.3 - Desarrollo del Prototipo IV - Consulta

7.3 - Implementación del Prototipo IV - Consulta	
Paquete de trabajo	7.3.3 - Desarrollo del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en redactar el Desarrollo del Prototipo IV - Consulta.
Duración estimada	10 horas
Entradas	Parte cliente del Prototipo IV - Consulta
Salidas / Entregables	Desarrollo del Prototipo IV - Consulta
Recursos necesarios	Documentos de Google
Precedencias	7.3.2 - Implementación del frontend del Prototipo IV - Consulta

Tabla 54.- Descripción del Paquete de Trabajo 7.4.1 - Pruebas del backend del Prototipo IV - Consulta

7.4 - Pruebas del Prototipo IV - Consulta	
Paquete de trabajo	7.4.1 - Pruebas del backend del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en probar la parte servidor del Prototipo IV - Consulta.
Duración estimada	4 horas
Entradas	Desarrollo del Prototipo IV - Consulta
Salidas / Entregables	Pruebas de la parte servidor del Prototipo IV - Consulta
Recursos necesarios	Postman y Moodle
Precedencias	7.3.3 - Desarrollo del Prototipo IV - Consulta

Tabla 55.- Descripción del Paquete de Trabajo 7.4.2 - Pruebas del frontend del Prototipo IV - Consulta

7.4 - Pruebas del Prototipo IV - Consulta	
Paquete de trabajo	7.4.2 - Pruebas del frontend del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en probar la parte cliente del Prototipo IV - Consulta.
Duración estimada	4 horas
Entradas	Pruebas de la parte servidor del Prototipo IV - Consulta
Salidas / Entregables	Pruebas de la parte cliente del Prototipo IV - Consulta
Recursos necesarios	Moodle
Precedencias	7.4.1 - Pruebas del backend del Prototipo IV - Consulta

Tabla 56.- Descripción del Paquete de Trabajo 7.4.3 - Verificación y evaluación del Prototipo IV - Consulta

7.4 - Pruebas del Prototipo IV - Consulta	
Paquete de trabajo	7.4.3 - Verificación y evaluación del Prototipo IV - Consulta
Descripción	Este paquete de trabajo consiste en redactar la Verificación y evaluación del Prototipo IV - Consulta.
Duración estimada	10 horas
Entradas	Pruebas de la parte cliente del Prototipo IV - Consulta
Salidas / Entregables	Verificación y evaluación del Prototipo IV - Consulta
Recursos necesarios	Documentos de Google
Precedencias	7.4.2 - Pruebas del frontend del Prototipo IV - Consulta

Tabla 57.- Descripción del Paquete de Trabajo 8.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo V - Planificar tarea

8.1 - Análisis del Prototipo V - Planificar tarea	
Paquete de trabajo	8.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en diseñar el Modelo de casos de uso y la Jerarquía de actores del Prototipo V - Planificar tarea.
Duración estimada	4 horas
Entradas	Verificación y evaluación del Prototipo IV - Consulta
Salidas / Entregables	Modelo de casos de uso y Jerarquía de actores del Prototipo V - Planificar tarea
Recursos necesarios	Visual Paradigm
Precedencias	7.4.3 - Verificación y evaluación del Prototipo IV - Consulta

Tabla 58.- Descripción del Paquete de Trabajo 8.1.2 - Casos de uso extendidos del Prototipo V - Planificar tarea

8.1 - Análisis del Prototipo V - Planificar tarea	
Paquete de trabajo	8.1.2 - Casos de uso extendidos del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en redactar los Casos de uso extendidos del Prototipo V - Planificar tarea.
Duración estimada	4 horas
Entradas	Modelo de casos de uso y Jerarquía de actores del Prototipo V - Planificar tarea
Salidas / Entregables	Casos de uso extendidos del Prototipo V - Planificar tarea
Recursos necesarios	Documentos de Google
Precedencias	8.1.1 - Modelo de casos de uso y Jerarquía de actores del Prototipo V - Planificar tarea

Tabla 59.- Descripción del Paquete de Trabajo 8.1.3 - Captura de requisitos del Prototipo V - Planificar tarea

8.1 - Análisis del Prototipo V - Planificar tarea	
Paquete de trabajo	8.1.3 - Captura de requisitos del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en redactar la Captura de requisitos del Prototipo V - Planificar tarea.
Duración estimada	10 horas
Entradas	Casos de uso extendidos del Prototipo V - Planificar tarea
Salidas / Entregables	Captura de requisitos del Prototipo V - Planificar tarea
Recursos necesarios	Documentos de Google
Precedencias	8.1.2 - Casos de uso extendidos del Prototipo V - Planificar tarea

Tabla 60.- Descripción del Paquete de Trabajo 8.2.1 - Diagrama de clases del Prototipo V - Planificar tarea

8.2 - Diseño del Prototipo V - Planificar tarea	
Paquete de trabajo	8.2.1 - Diagrama de clases del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en diseñar el Diagrama de clases del Prototipo V - Planificar tarea.
Duración estimada	4 horas
Entradas	Captura de requisitos del Prototipo V - Planificar tarea
Salidas / Entregables	Diagrama de clases del Prototipo V - Planificar tarea
Recursos necesarios	Visual Paradigm
Precedencias	8.1.3 - Captura de requisitos del Prototipo V - Planificar tarea

Tabla 61.- Descripción del Paquete de Trabajo 8.2.2 - Diagramas de secuencia del Prototipo V - Planificar tarea

8.2 - Diseño del Prototipo V - Planificar tarea	
Paquete de trabajo	8.2.2 - Diagramas de secuencia del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en diseñar los Diagramas de secuencia del Prototipo V - Planificar tarea.
Duración estimada	4 horas
Entradas	Diagrama de clases del Prototipo V - Planificar tarea
Salidas / Entregables	Diagramas de secuencia del Prototipo V - Planificar tarea
Recursos necesarios	Visual Paradigm
Precedencias	8.2.1 - Diagrama de clases del Prototipo V - Planificar tarea

Tabla 62.- Descripción del Paquete de Trabajo 8.2.3 - Análisis y diseño del Prototipo V - Planificar tarea

8.2 - Diseño del Prototipo V - Planificar tarea	
Paquete de trabajo	8.2.3 - Análisis y diseño del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en redactar el Análisis y diseño del Prototipo V - Planificar tarea.
Duración estimada	10 horas
Entradas	Diagramas de secuencia del Prototipo V - Planificar tarea
Salidas / Entregables	Análisis y diseño del Prototipo V - Planificar tarea
Recursos necesarios	Documentos de Google
Precedencias	8.2.2 - Diagramas de secuencia del Prototipo V - Planificar tarea

Tabla 63.- Descripción del Paquete de Trabajo 8.3.1 - Implementación del backend del Prototipo V - Planificar tarea

8.3 - Implementación del Prototipo V - Planificar tarea	
Paquete de trabajo	8.3.1 - Implementación del backend del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en implementar la parte servidor del Prototipo V - Planificar tarea.
Duración estimada	8 horas
Entradas	Análisis y diseño del Prototipo V - Planificar tarea
Salidas / Entregables	Parte servidor del Prototipo V - Planificar tarea
Recursos necesarios	Visual Studio Code, Node.js, Express y JavaScript
Precedencias	8.2.3 - Análisis y diseño del Prototipo V - Planificar tarea

Tabla 64.- Descripción del Paquete de Trabajo 8.3.2 - Implementación del frontend del Prototipo V - Planificar tarea

8.3 - Implementación del Prototipo V - Planificar tarea	
Paquete de trabajo	8.3.2 - Implementación del frontend del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en implementar la parte cliente del Prototipo V - Planificar tarea.
Duración estimada	8 horas
Entradas	Parte servidor del Prototipo V - Planificar tarea
Salidas / Entregables	Parte cliente del Prototipo V - Planificar tarea
Recursos necesarios	Visual Studio Code, Angular, HTML y TypeScript
Precedencias	8.3.1 - Implementación del backend del Prototipo V - Planificar tarea

Tabla 65.- Descripción del Paquete de Trabajo 8.3.3 - Desarrollo del Prototipo V - Planificar tarea

8.3 - Implementación del Prototipo V - Planificar tarea	
Paquete de trabajo	8.3.3 - Desarrollo del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en redactar el Desarrollo del Prototipo V - Planificar tarea.
Duración estimada	10 horas
Entradas	Parte cliente del Prototipo V - Planificar tarea
Salidas / Entregables	Desarrollo del Prototipo V - Planificar tarea
Recursos necesarios	Documentos de Google
Precedencias	8.3.2 - Implementación del frontend del Prototipo V - Planificar tarea

Tabla 66.- Descripción del Paquete de Trabajo 8.4.1 - Pruebas del backend del Prototipo V - Planificar tarea

8.4 - Pruebas del Prototipo V - Planificar tarea	
Paquete de trabajo	8.4.1 - Pruebas del backend del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en probar la parte servidor del Prototipo V - Planificar tarea.
Duración estimada	4 horas
Entradas	Desarrollo del Prototipo V - Planificar tarea
Salidas / Entregables	Pruebas de la parte servidor del Prototipo V - Planificar tarea
Recursos necesarios	Postman y Moodle
Precedencias	8.3.3 - Desarrollo del Prototipo V - Planificar tarea

Tabla 67.- Descripción del Paquete de Trabajo 8.4.2 - Pruebas del frontend del Prototipo V - Planificar tarea

8.4 - Pruebas del Prototipo V - Planificar tarea	
Paquete de trabajo	8.4.2 - Pruebas del frontend del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en probar la parte cliente del Prototipo V - Planificar tarea.
Duración estimada	4 horas
Entradas	Pruebas de la parte servidor del Prototipo V - Planificar tarea
Salidas / Entregables	Pruebas de la parte cliente del Prototipo V - Planificar tarea
Recursos necesarios	Moodle
Precedencias	8.4.1 - Pruebas del backend del Prototipo V - Planificar tarea

Tabla 68.- Descripción del Paquete de Trabajo 8.4.3 - Verificación y evaluación del Prototipo V - Planificar tarea

8.4 - Pruebas del Prototipo V - Planificar tarea	
Paquete de trabajo	8.4.3 - Verificación y evaluación del Prototipo V - Planificar tarea
Descripción	Este paquete de trabajo consiste en redactar la Verificación y evaluación del Prototipo V - Planificar tarea.
Duración estimada	10 horas
Entradas	Pruebas de la parte cliente del Prototipo V - Planificar tarea
Salidas / Entregables	Verificación y evaluación del Prototipo V - Planificar tarea
Recursos necesarios	Documentos de Google
Precedencias	8.4.2 - Pruebas del frontend del Prototipo V - Planificar tarea

Tabla 69.- Descripción del Paquete de Trabajo 9 - Conclusiones y trabajo futuro

Moodex	
Paquete de trabajo	9 - Conclusiones y trabajo futuro
Descripción	Este paquete de trabajo consiste en redactar el apartado de conclusiones y trabajo futuro de la memoria.
Duración estimada	20 horas
Entradas	Verificación y evaluación del Prototipo V - Planificar tarea
Salidas / Entregables	Apartado de conclusiones y trabajo futuro de la memoria
Recursos necesarios	Documentos de Google
Precedencias	8.4.3 - Verificación y evaluación del Prototipo V - Planificar tarea

Tabla 70.- Descripción del Paquete de Trabajo 10 - Resumen

Moodex	
Paquete de trabajo	10 - Resumen
Descripción	Este paquete de trabajo consiste en redactar el apartado de resumen de la memoria.
Duración estimada	10 horas
Entradas	Apartado de conclusiones y trabajo futuro de la memoria
Salidas / Entregables	Apartado de resumen de la memoria
Recursos necesarios	Documentos de Google
Precedencias	9 - Conclusiones y trabajo futuro

La Tabla 71 recoge un resumen de la duración estimada de los paquetes de trabajo con más peso del proyecto.

Tabla 71.- Resumen de la duración estimada de los paquetes de trabajo con más peso del proyecto

Paquete de trabajo	Duración estimada
1 - Introducción	10 horas
2 - Planteamiento inicial	100 horas
3 - Antecedentes	10 horas
4 - Prototipo I - Iniciar sesión	72 horas
4.1 - Análisis del Prototipo I - Iniciar sesión	18 horas
4.2 - Diseño del Prototipo I - Iniciar sesión	18 horas
4.3 - Implementación del Prototipo I - Iniciar sesión	18 horas
4.4 - Pruebas del Prototipo I - Iniciar sesión	18 horas
5 - Prototipo II - Listar cursos	80 horas
5.1 - Análisis del Prototipo II - Listar cursos	18 horas
5.2 - Diseño del Prototipo II - Listar cursos	18 horas
5.3 - Implementación del Prototipo II - Listar cursos	26 horas
5.4 - Pruebas del Prototipo II - Listar cursos	18 horas
6 - Prototipo III - Mostrar calendario	96 horas
6.1 - Análisis del Prototipo III - Mostrar calendario	18 horas
6.2 - Diseño del Prototipo III - Mostrar calendario	18 horas
6.3 - Implementación del Prototipo III - Mostrar calendario	42 horas
6.4 - Pruebas del Prototipo III - Mostrar calendario	18 horas
7 - Prototipo IV - Consulta	96 horas
7.1 - Análisis del Prototipo IV - Consulta	18 horas
7.2 - Diseño del Prototipo IV - Consulta	18 horas
7.3 - Implementación del Prototipo IV - Consulta	42 horas
7.4 - Pruebas del Prototipo IV - Consulta	18 horas
8 - Prototipo V - Planificar tarea	80 horas
8.1 - Análisis del Prototipo V - Planificar tarea	18 horas
8.2 - Diseño del Prototipo V - Planificar tarea	18 horas

8.3 - Implementación del Prototipo V - Planificar tarea	26 horas
8.4 - Pruebas del Prototipo V - Planificar tarea	18 horas
9 - Conclusiones y trabajo futuro	20 horas
10 - Resumen	10 horas
Moodex	574 horas

Ahora que se conoce la duración estimada de los paquetes de trabajo con más peso del proyecto se puede hacer una estimación del tiempo que llevará desarrollar tanto las funcionalidades básicas, las de los 3 primeros prototipos, como el proyecto al completo, los 5 prototipos (Tabla 72).

Tabla 72.- Resumen de la duración estimada del proyecto

Número de prototipos desarrollados	3	5
Duración estimada	398 horas	574 horas

Planificación temporal

Este apartado explica la planificación temporal que se va a seguir a lo largo del desarrollo del proyecto, es decir, va a determinar la distribución de la carga de trabajo semanal, también va a estimar la fecha de finalización del proyecto y finalmente va a establecer el periodo de realización estimado de cada paquete de trabajo de una manera gráfica mediante el Diagrama de Gantt.

En el apartado anterior se ha estimado tanto el tiempo que llevará desarrollar las funciones básicas de la aplicación como el proyecto al completo, 398 y 574 horas respectivamente. Se va a establecer una carga de trabajo diaria de 4 horas, lo que viene a ser media jornada laboral, y se ha planteado trabajar y hacer avances continuos día sí y día también, incluso domingos y días festivos, por lo que el cómputo de carga de trabajo semanal será de 28 horas.

La intención del desarrollador es comenzar a trabajar en el proyecto una vez concluido el cuatrimestre, incluido el periodo de trabajos y exámenes finales, exactamente el miércoles día 1 de junio de 2022. Se hará uso de esta fecha para estimar la fecha de finalización del proyecto. Como se ha mencionado anteriormente, el tiempo de desarrollo de las funciones básicas es de 398 horas y del proyecto completo de 574. Si se trabajan 4 horas diarias entonces se tardarán 100 y 144 días respectivamente. Por lo que en el caso de que se cumpla la planificación a rajatabla la fecha de finalización ronda entre el día 9 de septiembre y el día 23 de octubre de 2022.

Diagrama de Gantt

Las figuras que se presentan a continuación corresponden al Diagrama de Gantt de las funciones básicas (Fig. 5) y al Diagrama de Gantt del proyecto completo (Fig. 6), que se han desarrollado con el fin de llevar una planificación temporal lo más precisa posible. En la parte izquierda de los diagramas se pueden visualizar varias propiedades de cada tarea como son el nombre y la fecha de inicio y la fecha de fin estimadas. En la parte derecha únicamente se pueden ver las precedencias de las tareas.

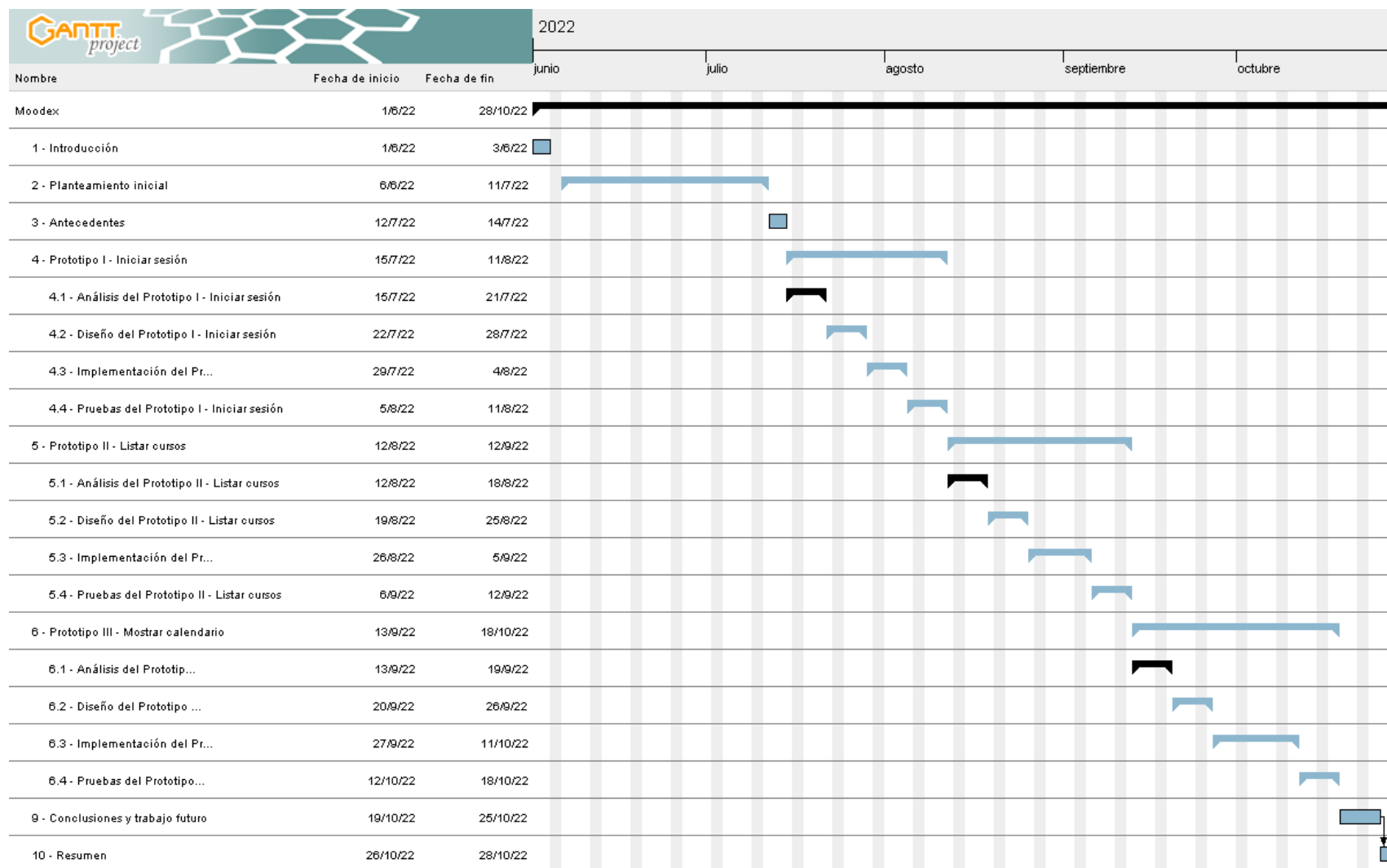


Fig. 5.- Diagrama de Gantt de las funciones básicas

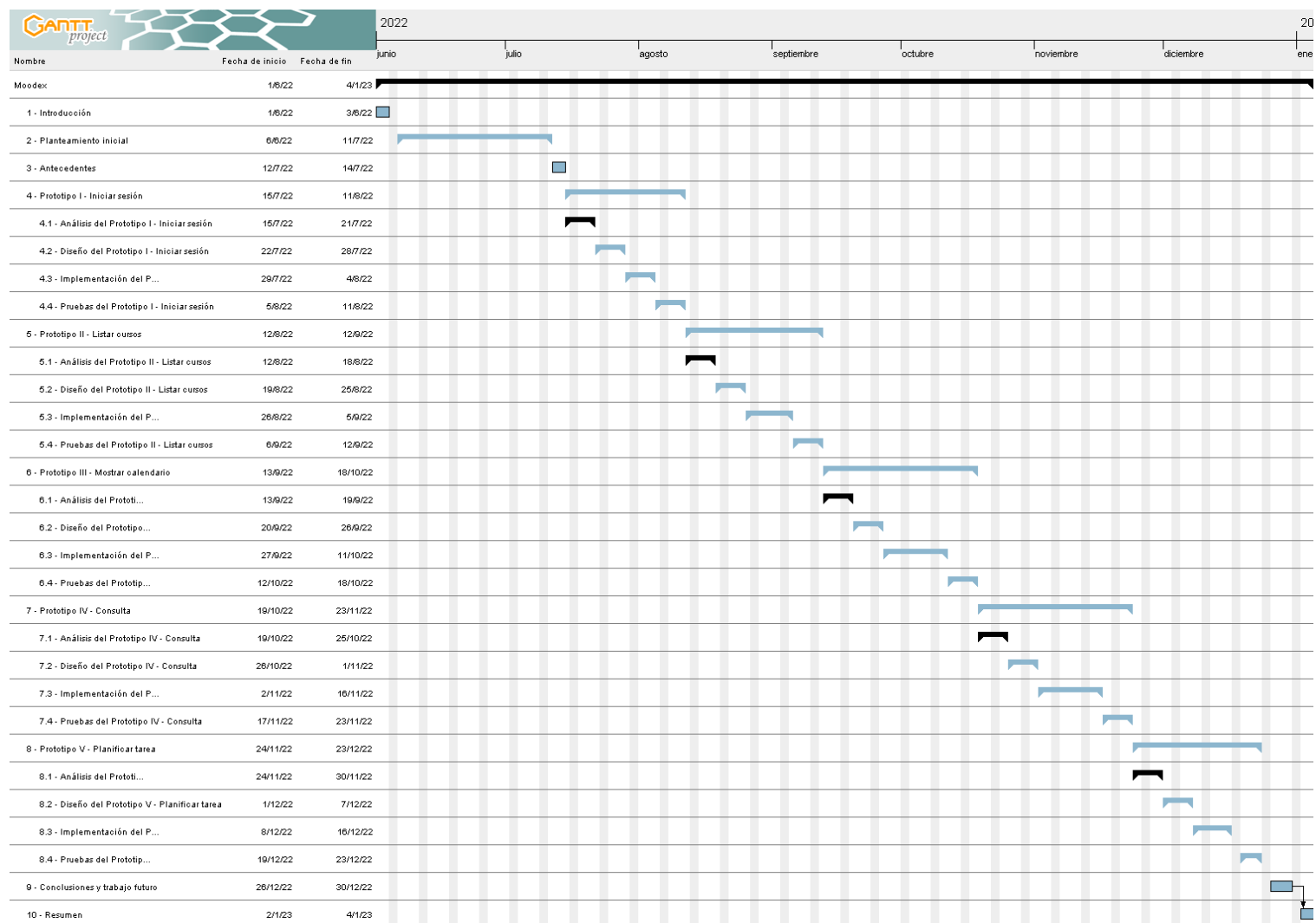


Fig. 6.- Diagrama de Gantt del proyecto completo

La Tabla 73 resume la parte izquierda de los diagramas anteriores, además indica la duración de cada tarea tanto en horas como en días laborales.

Tabla 73.- Resumen de la duración y del periodo de realización estimado del proyecto

Tarea	Duración estimada	Fecha de inicio estimada	Fecha de fin estimada
1 - Introducción	10 horas - 3 días laborales	1/6/22	3/6/22
2 - Planteamiento inicial	100 horas - 25 días laborales	6/6/22	11/7/22
3 - Antecedentes	10 horas - 3 días laborales	12/7/22	14/7/22
4 - Prototipo I - Iniciar sesión	72 horas - 18 días laborales	15/7/22	11/8/22
4.1 - Análisis del Prototipo I - Iniciar sesión	18 horas - 5 días laborales	15/7/22	21/7/22
4.2 - Diseño del Prototipo I - Iniciar sesión	18 horas - 5 días laborales	22/7/22	28/7/22
4.3 - Implementación del Prototipo I - Iniciar sesión	18 horas - 5 días laborales	29/7/22	4/8/22
4.4 - Pruebas del Prototipo I - Iniciar sesión	18 horas - 5 días laborales	5/8/22	11/8/22
5 - Prototipo II - Listar cursos	80 horas - 20 días laborales	12/8/22	12/9/22
5.1 - Análisis del Prototipo II - Listar cursos	18 horas - 5 días laborales	12/8/22	18/8/22
5.2 - Diseño del Prototipo II - Listar cursos	18 horas - 5 días laborales	19/8/22	25/8/22
5.3 - Implementación del Prototipo II - Listar cursos	26 horas - 7 días laborales	26/8/22	5/9/22
5.4 - Pruebas del Prototipo II - Listar	18 horas - 5 días laborales	6/9/22	12/9/22

cursos			
6 - Prototipo III - Mostrar calendario	96 horas - 24 días laborales	13/9/22	18/10/22
6.1 - Análisis del Prototipo III - Mostrar calendario	18 horas - 5 días laborales	13/9/22	19/9/22
6.2 - Diseño del Prototipo III - Mostrar calendario	18 horas - 5 días laborales	20/9/22	26/9/22
6.3 - Implementación del Prototipo III - Mostrar calendario	42 horas - 11 días laborales	27/9/22	11/10/22
6.4 - Pruebas del Prototipo III - Mostrar calendario	18 horas - 5 días laborales	12/10/22	18/10/22
7 - Prototipo IV - Consulta	96 horas - 24 días laborales	19/10/22	23/11/22
7.1 - Análisis del Prototipo IV - Consulta	18 horas - 5 días laborales	19/10/22	25/10/22
7.2 - Diseño del Prototipo IV - Consulta	18 horas - 5 días laborales	26/10/22	1/11/22
7.3 - Implementación del Prototipo IV - Consulta	42 horas - 11 días laborales	2/11/22	16/11/22
7.4 - Pruebas del Prototipo IV - Consulta	18 horas - 5 días laborales	17/11/22	23/11/22
8 - Prototipo V - Planificar tarea	80 horas - 20 días laborales	24/11/22	23/12/22
8.1 - Análisis del Prototipo V - Planificar tarea	18 horas - 5 días laborales	24/11/22	30/11/22
8.2 - Diseño del Prototipo V - Planificar tarea	18 horas - 5 días laborales	1/12/22	7/12/22
8.3 - Implementación del Prototipo V - Planificar tarea	26 horas - 7 días laborales	8/12/22	16/12/22

8.4 - Pruebas del Prototipo V - Planificar tarea	18 horas - 5 días laborales	19/12/22	23/12/22
En caso de desarrollar las funciones básicas (los 3 primeros prototipos)			
9 - Conclusiones y trabajo futuro	20 horas - 5 días laborales	19/10/22	25/10/22
10 - Resumen	10 horas - 3 días laborales	26/10/22	28/10/22
Moodex	398 horas - 100 días laborales	1/6/22	28/10/22
En caso de desarrollar el proyecto completo (los 5 prototipos)			
9 - Conclusiones y trabajo futuro	20 horas - 5 días laborales	26/12/22	30/12/22
10 - Resumen	10 horas - 3 días laborales	2/1/23	4/1/23
Moodex	574 horas - 144 días laborales	1/6/22	4/1/23

Herramientas

Este apartado cita cada una de las herramientas que se van a utilizar a lo largo del proyecto. Explica brevemente qué es, para qué sirve y qué uso se le va a dar a cada herramienta. Cabe mencionar que las herramientas se dividen en herramientas hardware y en herramientas software. Este último conjunto de herramientas se compone de herramientas dedicadas a la documentación, al diseño, al desarrollo, a las pruebas y otras que no se pueden catalogar en las anteriores.

Herramientas hardware

Las herramientas hardware hacen referencia a las herramientas físicas. En este caso la única herramienta hardware que se va a utilizar para llevar a cabo el proyecto es un portátil Acer Aspire 3 A315-53G-5947.

Herramientas software

Herramientas para documentación

Documentos de Google¹¹: Esta herramienta se va a utilizar para realizar todas las tareas de documentación como puede ser la propia memoria.

Herramientas para diseño

GlooMaps¹²: *“Gloomaps es una herramienta para hacer organigramas de forma sencilla. El gráfico es más que aceptable y se descarga en PDF o PNG”* (2). Esta herramienta se va a utilizar para diseñar la Estructura de Descomposición del Trabajo del apartado de alcance del planteamiento inicial de la memoria.

Gantt Project¹³: Esta herramienta se va a utilizar para hacer el diseño de los diagramas de Gantt tanto de la planificación temporal inicial como de la final.

Visual Paradigm¹⁴: *“Visual Paradigm ayuda a los equipos de desarrollo de software a capturar los requisitos correctos y transformarlos en diseños precisos, lo que ayuda a los desarrolladores a crear el software adecuado según los requisitos”* (3). Esta herramienta se va a utilizar para hacer el diseño de los distintos diagramas, como el diagrama de los casos de uso y de la jerarquía de actores.

¹¹ Documentos de Google: <https://www.google.es/intl/es/docs/about/>

¹² GlooMaps: <https://www.gloomaps.com/>

¹³ Gantt Project: <https://www.ganttproject.biz/>

¹⁴ Visual Paradigm: <https://www.visual-paradigm.com/>

Paint¹⁵: Esta herramienta se va a utilizar para crear varias figuras de la documentación.

Herramientas para desarrollo

Visual Studio Code¹⁶: *“Visual Studio Code es un editor de código fuente independiente que se ejecuta en Windows, macOS y Linux. La elección principal para desarrolladores web y JavaScript, con extensiones para admitir casi cualquier lenguaje de programación.”* (4). Esta herramienta se va a utilizar para desarrollar tanto la parte lógica y funcional como la parte que interactúa con los usuarios de la aplicación.

Node.js: *“Node.js es un entorno JavaScript que nos permite ejecutar en el servidor, de manera asíncrona, con una arquitectura orientada a eventos y basado en el motor V8 de Google”* (5). Esta herramienta se va a utilizar para desarrollar la parte lógica y funcional de la aplicación.

Express¹⁷: *“Express es el framework web más popular de Node, y es la librería subyacente para un gran número de otros frameworks web de Node populares. Proporciona mecanismos para: escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas), integración con motores de renderización de “vistas” para generar respuestas mediante la introducción de datos en plantillas, establecer ajustes de aplicaciones web como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta y añadir procesamiento de peticiones “middleware” adicional en cualquier punto dentro de la tubería de manejo de la petición”* (6). Esta herramienta se va a utilizar como framework de Node.js con el que trabajar.

JavaScript: Este lenguaje es el que se va a utilizar para escribir el código de la parte lógica y funcional de la aplicación.

Angular: *“Angular es un framework open source desarrollado por Google para facilitar la creación y programación de aplicaciones web de una sola página, las webs SPA (Single Page Application)”* (7). Esta herramienta se va a utilizar para desarrollar la parte de la aplicación que interactúa con los usuarios.

HTML¹⁸: Este lenguaje es el que se va a utilizar para estructurar y desplegar las distintas páginas de la aplicación web.

TypeScript: *“TypeScript es un superconjunto de JavaScript que permite escribir y generar código de JavaScript que opera de manera fuertemente tipada y orientada a objetos, pero que conserva esa flexibilidad que los desarrolladores adoran (o, en algunos casos, detestan) de JavaScript”* (8). Este lenguaje se va a utilizar para escribir el código de la parte de la aplicación que interactúa con los usuarios.

¹⁵ Paint: <https://apps.microsoft.com/store/detail/paint/9PCFS5B6T72H?hl=es-es&gl=ES>

¹⁶ Visual Studio Code: <https://code.visualstudio.com/>

¹⁷ Express: <http://expressjs.com/es/>

¹⁸ HTML: <https://developer.mozilla.org/es/docs/Web/HTML>

Herramientas para pruebas

Postman¹⁹: *“Postman es una aplicación que nos permite testear APIs a través de una interfaz gráfica de usuario. Entre las ventajas que tiene Postman encontramos la capacidad de crear colecciones y distintos ambientes de pruebas. Postman es una herramienta fácil de usar que nos ayuda a optimizar el tiempo de ejecución de pruebas.”* (9). Esta herramienta se va a utilizar para realizar las pruebas correspondientes a la parte lógica y funcional de la aplicación.

Moodle: *“Moodle es una plataforma de aprendizaje diseñada para proporcionar a educadores, administradores y estudiantes un sistema integrado único, robusto y seguro para crear ambientes de aprendizaje personalizados”* (10). Esta herramienta se va a utilizar como plataforma contra la que realizar las pruebas correspondientes a las operaciones que va a implementar la aplicación web. Es obligatorio utilizar esta herramienta dado que es la plataforma que se usa en la UPV/EHU.

Otras herramientas

Windows²⁰: Este es el sistema operativo del que dispone el portátil que se va a utilizar a lo largo del proyecto.

Google Chrome²¹: Este es el navegador que se va a utilizar para hacer todas las operaciones y consultas en línea a lo largo del proyecto.

Google Drive²²: Esta herramienta se va a utilizar para almacenar archivos de la parte de documentación y de diseño y para crear todos los archivos relacionados con la parte de documentación.

GitHub²³: Esta herramienta se va a utilizar para almacenar archivos de la parte de desarrollo.

¹⁹ Postman: <https://www.postman.com/>

²⁰ Windows: <https://www.microsoft.com/es-es/windows?r=1>

²¹ Google Chrome: https://www.google.com/intl/es_es/chrome/

²² Google Drive: https://www.google.com/intl/es_es/drive/

²³ GitHub: <https://github.com/>

Gestión de riesgos

Este apartado cita cada uno de los riesgos existentes a la hora de realizar el proyecto. Explica brevemente en qué consiste el riesgo y cuales son sus consecuencias y la probabilidad de que este suceda y el impacto que tendría dado el caso. También detalla el plan de prevención y el plan de contingencia que se llevará a cabo en cada caso. Finalmente se valora y se le asigna el respectivo seguimiento a cada riesgo.

Calcular el valor exacto de la probabilidad de que un riesgo ocurra es difícil y es más comprensible explicar la probabilidad mediante un valor nominal que por un valor numérico, por lo que, se va a trabajar con las franjas que aparecen en la Tabla 74.

Tabla 74.- Probabilidad y porcentaje correspondiente

Probabilidad	Porcentaje
Improbable	0 - 25
Poco probable	25 - 50
Probable	50 - 75
Muy probable	75 - 100

Lo mismo pasa con el impacto de un riesgo, es complejo calcular el retraso exacto que va a provocar y es más comprensible explicar el impacto mediante un valor nominal que por un valor numérico, por lo que, se va a trabajar con las franjas que se muestran en la Tabla 75.

Tabla 75.- Impacto y retraso correspondiente

Impacto	Retraso
Muy pequeño	0 - 60 minutos
Pequeño	1 - 24 horas
Medio	1 - 3 días
Grande	3 - 7 días
Muy grande	Más de 1 semana

La valoración del riesgo se hace en base a las dos métricas anteriores y puede tener tres distintos valores: baja, media y alta. En cada caso se hará un seguimiento periódico distinto al riesgo, concretamente como se detalla en la Tabla 76.

Tabla 76.- Valoración y revisión correspondiente

Valoración	Revisión del riesgo
Baja	Mensualmente
Media	Cada dos semanas
Alta	Semanalmente

Antes de comenzar a analizar cada uno de los riesgos cabe recalcar que cada riesgo tiene relación con una parte del proyecto, es decir, hay distintos tipos de riesgos como pueden ser los riesgos de diseño, los riesgos del producto generado y los riesgos de índole personal del desarrollador, entre otros. A continuación se detalla la distribución que siguen los riesgos que se han logrado detectar:

- **Riesgos de diseño y planificación**
 - Planificación temporal incorrecta
 - Variación en los requisitos o especificaciones del proyecto
 - Problemas de análisis y diseño
- **Riesgos del producto generado**
 - Desaparición de alguna librería utilizada
 - **Riesgos de que haya más usuarios de lo planificado**
 - Sobrecargar el servidor de peticiones
- **Riesgos de índole personal del desarrollador**
 - Enfermedad / lesión
 - Cúmulo de trabajos / exámenes
 - Cambio de país de residencia
 - Problemas personales
 - Baja moral y/o desmotivación
 - Estancamiento
- **Riesgos de hardware y software**
 - **Inutilización del equipo informático**
 - Infección de un virus
 - Rotura / Pérdida / Robo del equipo
 - Fallo en el disco duro
 - Carencia de una API necesaria
 - **Cambio / Actualización de alguna herramienta de desarrollo**
 - Cambios / Actualizaciones en las librerías utilizadas
 - Cambios / Actualizaciones en APIs utilizadas
 - Pérdida de la conexión a Internet
 - Apagones
 - Falta de conocimiento del desarrollador en alguna tecnología
 - **Problemas con alguna herramienta de desarrollo**
 - Problemas de instalación de software

Por último, comentar que el hecho de trabajar en la nube reduce el riesgo de pérdida de información de manera considerable, por lo que se ha decidido almacenar y trabajar con los

archivos relacionados con la documentación mediante Google Drive y almacenar la aplicación en un repositorio de GitHub.

Análisis

Riesgos de diseño y planificación

La Tabla 77, la Tabla 78 y la Tabla 79 muestran los riesgos asociados al apartado de diseño y planificación, es decir, los riesgos que conlleva realizar un mal diseño o una mala planificación.

Tabla 77.- Análisis del riesgo - Planificación temporal incorrecta

Planificación temporal incorrecta	
Descripción	Este riesgo implica realizar una planificación temporal incorrecta lo que puede acarrear un gran desajuste en el tiempo dedicado a cada tarea y al proyecto en general.
Plan de prevención	Consiste en contrastar la planificación con alguien experto.
Plan de contingencia	Consiste en reajustar la planificación en caso de que haya un gran retraso respecto a las tareas programadas.
Probabilidad	Muy probable
Impacto	Grande
Valoración	Alta

Tabla 78.- Análisis del riesgo - Variación en los requisitos o especificaciones del proyecto

Variación en los requisitos o especificaciones del proyecto	
Descripción	Este riesgo implica una variación en los requisitos o especificaciones del proyecto y dependiendo la fase en la que se encuentre el proyecto puede llegar a acarrear una edición enorme.
Plan de prevención	Consiste en hacer una primera aproximación de la captura de requisitos y, tras esto, acordar con el tutor cuáles serán los requisitos exactos y obligatorios.
Plan de contingencia	Consiste en adaptar todo los avances del proyecto a los nuevos requisitos.
Probabilidad	Improbable
Impacto	Muy grande
Valoración	Baja

Tabla 79.- Análisis del riesgo - Problemas de análisis y diseño

Problemas de análisis y diseño	
Descripción	Este riesgo implica realizar un diseño pobre e incluso inadecuado, lo que puede complicar la fase de desarrollo futura.
Plan de prevención	Consiste en hacer una primera aproximación al análisis y diseño y contrastarlo con el tutor antes de seguir avanzando.
Plan de contingencia	Consiste en volver a la fase de análisis y diseño y plantear cambios o mejoras para paliar la situación.
Probabilidad	Poco probable
Impacto	Grande
Valoración	Alta

Riesgos del producto generado

La Tabla 80 muestra los riesgos asociados al producto generado, es decir, los riesgos que surgen una vez el producto está en la fase de lanzamiento o mantenimiento.

Tabla 80.- Análisis del riesgo - Desaparición de alguna librería utilizada

Desaparición de alguna librería utilizada	
Descripción	Este riesgo implica la desaparición de alguna librería utilizada en el código de la aplicación y puede acarrear que quede inhabilitada.
Plan de prevención	Consiste en buscar librerías que tengan un largo recorrido y soporte de una amplia comunidad.
Plan de contingencia	Consiste en sustituir la librería por otra que implemente las mismas funciones y en caso de que no haya ninguna librería similar implementar las funciones a mano siempre y cuando sea factible, es decir, cuando no requiera demasiado esfuerzo.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Riesgos de que haya más usuarios de lo planificado

La Tabla 81 muestra los riesgos asociados al sobreuso del producto, es decir, los riesgos que conlleva que los usuarios saturen el producto.

Tabla 81.- Análisis del riesgo - Sobrecargar el servidor de peticiones

Sobrecargar el servidor de peticiones	
Descripción	Este riesgo implica una sobrecarga de peticiones en el servidor lo que puede inhabilitar la aplicación web temporalmente.
Plan de prevención	Consiste en implementar una funcionalidad capaz de gestionar una gran cantidad de peticiones simultáneas rápidamente.
Plan de contingencia	Consiste en programar una gestión distinta de las peticiones.
Probabilidad	Improbable
Impacto	Muy pequeño
Valoración	Baja

Riesgos de índole personal del desarrollador

La Tabla 82, la Tabla 83, la Tabla 84, la Tabla 85, la Tabla 86 y la Tabla 87 muestran los riesgos asociados al desarrollador, es decir, los riesgos e impedimentos que le pueden surgir al desarrollador.

Tabla 82.- Análisis del riesgo - Enfermedad / lesión

Enfermedad / lesión	
Descripción	Este riesgo implica que el desarrollador sufra una enfermedad o una lesión, lo que le puede inhabilitar temporalmente.
Plan de prevención	Consiste en cumplir las medidas necesarias para no contraer ninguna enfermedad ni lesionarse de gravedad.
Plan de contingencia	Consiste en cumplir las instrucciones impuestas por el profesional en cuestión para recuperarse cuanto antes y reajustar la planificación temporal.
Probabilidad	Poco probable
Impacto	Grande
Valoración	Alta

Tabla 83.- Análisis del riesgo - Cúmulo de trabajos / exámenes

Cúmulo de trabajos / exámenes	
Descripción	Este riesgo implica una sobrecarga de trabajos y/o exámenes, lo que puede impedir que el desarrollador avance temporalmente.
Plan de prevención	Consiste en llevar una buena organización y tener un determinado tiempo dedicado a la realización del proyecto a lo largo de la semana.
Plan de contingencia	Consiste en reajustar la planificación temporal.
Probabilidad	Muy probable
Impacto	Pequeño
Valoración	Alta

Tabla 84.- Análisis del riesgo - Cambio de país de residencia

Cambio de país de residencia	
Descripción	Este riesgo implica cambiar el país de residencia, lo que puede suponer un impedimento para el desarrollador temporalmente.
Plan de prevención	Consiste en dedicar el tiempo necesario en reflexionar si merece la pena cambiar de país de residencia.
Plan de contingencia	Consiste en reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Tabla 85.- Análisis del riesgo - Problemas personales

Problemas personales	
Descripción	Este riesgo implica que el desarrollador padece problemas personales, lo que puede suponer un impedimento para el desarrollador temporalmente.
Plan de prevención	Consiste en evitar todo lo posible la causa de los problemas.
Plan de contingencia	Consiste en reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Tabla 86.- Análisis del riesgo - Baja moral y/o desmotivación

Baja moral y/o desmotivación	
Descripción	Este riesgo implica que el desarrollador tenga baja moral o sufra de desmotivación, lo que puede suponer un impedimento para el desarrollador temporalmente.
Plan de prevención	Consiste en implantar unos hábitos saludables en los que se prime la disciplina.
Plan de contingencia	Consiste en reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Tabla 87.- Análisis del riesgo - Estancamiento

Estancamiento	
Descripción	Este riesgo implica que el desarrollador se estanque y se quede sin ideas con las que proseguir, lo que puede suponer un impedimento para el desarrollador temporalmente.
Plan de prevención	Consiste en acudir al tutor en busca de consejo.
Plan de contingencia	Consiste en reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Riesgos de hardware y software

La Tabla 88, la Tabla 89, la Tabla 90 y la Tabla 91 muestran los riesgos asociados al apartado de herramientas, es decir, los riesgos que surgen a las herramientas tanto hardware como software de las que se disponen para realizar el proyecto.

Tabla 88.- Análisis del riesgo - Carencia de una API necesaria

Carencia de una API necesaria	
Descripción	Este riesgo implica que no exista una API necesaria para implementar una función del proyecto, lo que puede suponer una complicación en la fase de desarrollo.
Plan de prevención	Consiste en hacer una breve búsqueda de la API necesaria.
Plan de contingencia	Consiste en replantear las funcionalidades afectadas y buscar alternativas a implementar en caso de que sea factible.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Tabla 89.- Análisis del riesgo - Pérdida de la conexión a Internet

Pérdida de la conexión a Internet	
Descripción	Este riesgo implica la pérdida de la conexión a Internet, lo que puede suponer un impedimento para el desarrollo temporalmente.
Plan de prevención	Consiste en cerciorarse de que la conexión es estable antes de empezar a trabajar y trabajar de manera que los avances se guarden automáticamente y queden almacenados en la nube.
Plan de contingencia	Consiste en establecer una conexión alternativa y recuperar la información perdida, almacenada en el último backup realizado, o volver a hacerla, lo que supone reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Muy pequeño
Valoración	Baja

Tabla 90.- Análisis del riesgo - Apagones

Apagones	
Descripción	Este riesgo implica que haya un apagón, lo que puede suponer la pérdida de los avances hechos en dicho momento y un impedimento para el desarrollo temporalmente.
Plan de prevención	Consiste en trabajar de manera que los avances se guarden automáticamente y queden almacenados en la nube.
Plan de contingencia	Consiste en recuperar la información perdida, almacenada en el último backup realizado, o volver a hacerla, lo que supone reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Muy pequeño
Valoración	Baja

Tabla 91.- Análisis del riesgo - Falta de conocimiento del desarrollador en alguna tecnología

Falta de conocimiento del desarrollador en alguna tecnología	
Descripción	Este riesgo implica que el desarrollador no tenga los conocimientos necesarios para hacer un uso adecuado de alguna tecnología, lo que puede suponer un impedimento para el desarrollador temporalmente.
Plan de prevención	Consiste en dedicar el tiempo necesario para aprender a utilizar la tecnología.
Plan de contingencia	Consiste en hacer una breve búsqueda de la información necesaria para comprender la tecnología y reajustar la planificación temporal.
Probabilidad	Muy probable
Impacto	Medio
Valoración	Alta

Inutilización del equipo informático

La Tabla 92, la Tabla 93 y la Tabla 94 muestran los riesgos asociados a la inutilización del equipo, es decir, los riesgos que conlleva un periodo de inutilización del ordenador que es una herramienta indispensable para desarrollar el proyecto.

Tabla 92.- Análisis del riesgo - Infección de un virus

Infección de un virus	
Descripción	Este riesgo implica que el equipo sea infectado por un virus, lo que puede suponer una pérdida de datos y un impedimento para el desarrollo temporalmente.
Plan de prevención	Consiste en evitar todo lo posible que el equipo sea infectado por un virus, esto se lleva a cabo cumpliendo con las medidas de seguridad pertinentes, y trabajar de manera que los avances se guarden automáticamente y queden almacenados en la nube.
Plan de contingencia	Consiste en deshacerse del virus lo antes posible, mientras tanto intentar conseguir otro equipo con el que seguir trabajando y recuperar la información perdida, almacenada en el último backup realizado, o volver a hacerla, lo que supone reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Medio
Valoración	Baja

Tabla 93.- Análisis del riesgo - Rotura / Pérdida / Robo del equipo

Rotura / Pérdida / Robo del equipo	
Descripción	Este riesgo implica que el equipo se rompa, se pierda o que alguien lo robe, lo que puede suponer una pérdida de datos y un impedimento para el desarrollo temporalmente.
Plan de prevención	Consiste en evitar todo lo posible que el equipo se rompa, se pierda o alguien lo robe, esto se lleva a cabo cumpliendo con las medidas de seguridad pertinentes, y trabajar de manera que los avances se guarden automáticamente y queden almacenados en la nube.
Plan de contingencia	Consiste en arreglar, encontrar o recuperar el equipo lo antes posible, mientras tanto intentar conseguir otro equipo con el que seguir trabajando y recuperar la información perdida, almacenada en el último backup realizado, o volver a hacerla, lo que supone reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Medio
Valoración	Baja

Tabla 94.- Análisis del riesgo - Fallo en el disco duro

Fallo en el disco duro	
Descripción	Este riesgo implica que haya un fallo en el disco duro, lo que puede suponer una pérdida de datos y un impedimento para el desarrollo temporalmente.
Plan de prevención	Consiste en evitar todo lo posible que falle el disco duro, esto se lleva a cabo cumpliendo con las medidas de seguridad pertinentes, y trabajar de manera que los avances se guarden automáticamente y queden almacenados en la nube.
Plan de contingencia	Consiste en arreglar el disco duro lo antes posible, mientras tanto intentar conseguir otro equipo con el que seguir trabajando y recuperar la información perdida, almacenada en el último backup realizado, o volver a hacerla, lo que supone reajustar la planificación temporal.
Probabilidad	Improbable
Impacto	Medio
Valoración	Baja

Cambio / Actualización de alguna herramienta de desarrollo

La Tabla 95 y la Tabla 96 muestran los riesgos asociados al cambio o actualización de alguna herramienta de desarrollo, es decir, los riesgos que conlleva que una herramienta sufra un cambio o una actualización.

Tabla 95.- Análisis del riesgo - Cambios / Actualizaciones en las librerías utilizadas

Cambios / Actualizaciones en las librerías utilizadas	
Descripción	Este riesgo implica que una de las librerías utilizadas sufra un cambio o una actualización, lo que puede suponer una complicación en la fase de desarrollo o de mantenimiento de la aplicación.
Plan de prevención	Consiste en hacer un breve estudio de las librerías para saber si van a sufrir un cambio o una actualización en un futuro próximo y si es necesario contar con otras librerías que implementen las mismas funciones para sustituirlas llegado el momento.
Plan de contingencia	Consiste en revisar en qué consiste el cambio o la actualización y si es necesario sustituir las librerías por otras que implementen las mismas funciones y en caso de que no haya ninguna librería similar implementar las funciones a mano siempre y cuando sea factible, es decir, cuando no requiera demasiado esfuerzo.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Tabla 96.- Análisis del riesgo - Cambios / Actualizaciones en APIs utilizadas

Cambios / Actualizaciones en APIs utilizadas	
Descripción	Este riesgo implica que una de las APIs utilizadas sufra un cambio o una actualización, lo que puede suponer una complicación en la fase de desarrollo o de mantenimiento de la aplicación.
Plan de prevención	Consiste en hacer un breve estudio de las APIs para saber si van a sufrir un cambio o una actualización en un futuro próximo y si es necesario contar con otras APIs que implementen las mismas funciones para sustituirlas llegado el momento.
Plan de contingencia	Consiste en revisar en qué consiste el cambio o la actualización y si es necesario sustituir las APIs por otras que implementen las mismas funciones y en caso de que no haya ninguna API similar implementar las funciones a mano siempre y cuando sea factible, es decir, cuando no requiera demasiado esfuerzo.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Problemas con alguna herramienta de desarrollo

La Tabla 97 muestra los riesgos asociados a los problemas con las herramientas de desarrollo, es decir, los riesgos que pueden surgir al utilizar una herramienta software.

Tabla 97.- Análisis del riesgo - Problemas de instalación de software

Problemas de instalación de software	
Descripción	Este riesgo implica que haya problemas con la instalación de alguna herramienta software, lo que puede suponer una complicación en la fase de desarrollo.
Plan de prevención	Consiste en informarse de antemano sobre los pasos a seguir para instalar el software.
Plan de contingencia	Consiste en hacer una breve búsqueda de la información necesaria para comprender los pasos a seguir para instalar el software.
Probabilidad	Improbable
Impacto	Pequeño
Valoración	Baja

Evaluación económica

Este apartado calcula el coste que genera llevar a cabo el proyecto, para ello hay que tener en cuenta los distintos tipos de coste, como son el coste del desarrollo, el coste del hardware empleado, el coste de las licencias del software y los costes indirectos.

Coste de desarrollo

El coste de desarrollo es el coste que conlleva realizar la aplicación web y se calcula mediante el producto entre el número de horas que le llevará al desarrollador llevar a cabo la aplicación y el salario medio de un desarrollador web por hora. Por lo que se ha visto en el alcance realizar las funciones básicas y el proyecto por completo llevará 398 y 574 horas respectivamente y el sueldo medio de un desarrollador web en España a principios de agosto de 2022 es de 12'95 € por hora según la página web talent.com²⁴. La Tabla 98 muestra los resultados:

Tabla 98.- Coste de desarrollo

Desarrollo	Horas	Coste / Hora (€ / hora)	Coste total (€)
Funciones básicas (3 prototipos)	398	12'95	5154'1
Proyecto completo (5 prototipos)	574	12'95	7433'3

Coste del hardware empleado

El coste del hardware empleado es la amortización lograda a lo largo del desarrollo del proyecto de los dispositivos hardware utilizados y se calcula mediante el producto entre la división entre el valor actual y la vida útil del dispositivo y el tiempo necesario para llevar a cabo la aplicación. En este caso el único dispositivo que se va a emplear es un portátil Acer Aspire 3 A315-53G-5947 valorado en torno a 500 € actualmente y con una vida útil de 4 años. La Tabla 99 muestra los resultados:

²⁴ Sueldo medio de un desarrollador web en España según talent.com:
<https://es.talent.com/salary?job=desarrollador+web>

Tabla 99.- Coste del portátil

Desarrollo	Horas	Coste del portátil (€)	Vida útil (horas)	Amortización (€)
Funciones básicas (3 prototipos)	398	500	35040	5'68
Proyecto completo (5 prototipos)	574	500	35040	8'19

Coste de las licencias del software

El coste de las licencias del software es el coste que genera tener que pagar las distintas cuotas de las herramientas software que se utilizan a lo largo del proyecto. En este caso, las herramientas utilizadas son gratuitas o se dispone de licencia educativa, como es el caso de Visual Paradigm, por lo que no se ha originado este tipo de gasto.

Costes indirectos

Los costes indirectos son aquellos que no son directamente imputables al desarrollo de la aplicación. En este caso se atribuyen a los costes de mantenimiento producidos durante el proceso de elaboración del proyecto; luz, agua, electricidad, etc. Se considera que este tipo de costes pueden llegar a suponer un 5% de la suma del resto de costes. La Tabla 100 muestra los resultados:

Tabla 100.- Costes indirectos y coste total

Desarrollo	Coste de desarrollo (€)	Coste del hardware empleado (€)	Coste de las licencias del software (€)	Costes indirectos (€)	Coste total (€)
Funciones básicas (3 prototipos)	5154'1	5'68	0	257'99	5417'77
Proyecto completo (5 prototipos)	7433'3	8'19	0	372'07	7813'56

Coste total

El coste total es la suma de todos los costes anteriores y como se puede ver en la Tabla 100 ronda en torno a los 5420 y a los 7815 €, dependiendo de la cantidad de prototipos que se lleguen a completar.

Modelos de negocio

Por último, decir que el propósito de este proyecto no es la obtención de beneficios económicos. Su finalidad, como ya se ha comentado anteriormente, es desarrollar una herramienta que proporcione información al profesorado y que pueda servir de ayuda para planificar la docencia y la entrega de tareas. Por ello, no se va a hacer una estimación de los beneficios que se podrían obtener una vez lanzada la versión final de la aplicación.

ANTECEDENTES

Este apartado expone la situación actual del proyecto, para ello se hace un estudio de las diferentes alternativas existentes y otro de los distintos enfoques que se le pueden dar al desarrollo del mismo.

Situación actual

La situación actual del proyecto es que hay herramientas que tienen funcionalidades que actúan de una forma similar a la de Moodex, básicamente comparten la manera de acceder y extraer la información desde Moodle. Pero no se encuentra ninguna aplicación que lleve a cabo todas las funciones de Moodex y tenga sus mismos objetivos.

Estudio de las diferentes alternativas existentes

Para llevar a cabo el estudio de las distintas alternativas existentes se comienza buscando herramientas similares y se indican tanto las similitudes como las diferencias respecto a la herramienta en desarrollo.

Esta búsqueda se hace tanto de manera genérica, indagando por internet mirando herramientas que tengan alguna funcionalidad parecida, como de manera específica, revisando la sección de plugins de la página web del propio Moodle²⁵, comprobando si ya hay una herramienta que realiza la misma función.

UBUMonitor

Tras realizar un estudio superficial a través del buscador de Google se ha ido a parar al repositorio de GitHub del proyecto UBUMonitor²⁶.

El objetivo primordial de esta herramienta es hacer un seguimiento del alumnado de la plataforma Moodle con el fin de que el profesorado pueda visualizar de una manera gráfica distintas estadísticas, como son las calificaciones y los registros que se generan en los distintos cursos.

La semejanza principal es que la herramienta accede a Moodle y extrae toda la información de las asignaturas impartidas por el docente que haya iniciado sesión. Sin embargo, el uso

²⁵ Plugins de Moodle: <https://moodle.org/plugins/?q=>

²⁶ UBUMonitor: <https://github.com/yjx0003/UBUMonitor>

que se le da a esta información es diferente, Moodex la utiliza para realizar una planificación de los cursos y realizar informes de uso.

UBUMonitor está programada en Java y Python y Moodex va a ser desarrollada en JavaScript principalmente, así que será difícil reutilizar código. De todos modos puede ser útil para hacerse una idea general y ver cómo funciona.

Plugins

Tras buscar a fondo entre los distintos plugins de los que consta Moodle no se ha encontrado una herramienta que implemente exactamente las mismas funciones. Se ha topado una funcionalidad similar que es el cálculo del tiempo estimado dedicado de los participantes de un curso, que puede ser útil a la hora de realizar los informes de uso. Esta funcionalidad la implementa el plugin Course dedication²⁷.

Estudio de los distintos enfoques del problema a solucionar

Para llevar a cabo el estudio de los distintos enfoques que se le pueden dar al problema a solucionar se comienza buscando los aspectos del proyecto que tienen diferentes maneras de realizarse. Entre dichos aspectos se encuentran las tecnologías a utilizar y el tipo de aplicación a desarrollar. Se finaliza tomando una decisión y justificando la elección.

Tecnologías a utilizar

Una gran mayoría de las herramientas han sido seleccionadas por el desarrollador basándose en TFGs anteriores y en sus preferencias. Sin embargo, varias herramientas de desarrollo y pruebas vienen dadas de antemano: Node.js, Angular y Moodle.

Tipo de aplicación a desarrollar

Debido a que las herramientas de desarrollo que nos vienen dadas, Node.js y Angular, sirven específicamente para crear sitios web dinámicos muy eficientes, la mejor opción es desarrollar una aplicación web. Además, una aplicación web puede funcionar en cualquier dispositivo que disponga de conexión a internet.

²⁷ Course dedication: https://moodle.org/plugins/block_dedication

Ahora es el momento de decidir si se va a desarrollar un plugin o una herramienta externa a Moodle. En este caso, es preferible llevar a cabo una herramienta externa a Moodle dado que un plugin tiene una estructura concreta y esto supone mayor tiempo de aprendizaje para el desarrollador.

ARQUITECTURA

La Fig. 7 muestra la arquitectura de la aplicación. La arquitectura está compuesta por Moodex, que es la aplicación web que se va a desarrollar, y Moodle, la plataforma de docencia. Moodex dispone de un frontend sobre el que los usuarios realizan distintas acciones. Ciertas acciones necesitan que el backend se encargue de solicitar información a Moodle. Para estas situaciones Moodle cuenta con determinados servicios web que recopilan la información necesaria de la base de datos y se la devuelven al solicitante, el backend en este caso. El backend realiza lo oportuno con esa información y le devuelve el resultado al frontend y al usuario, en última instancia.

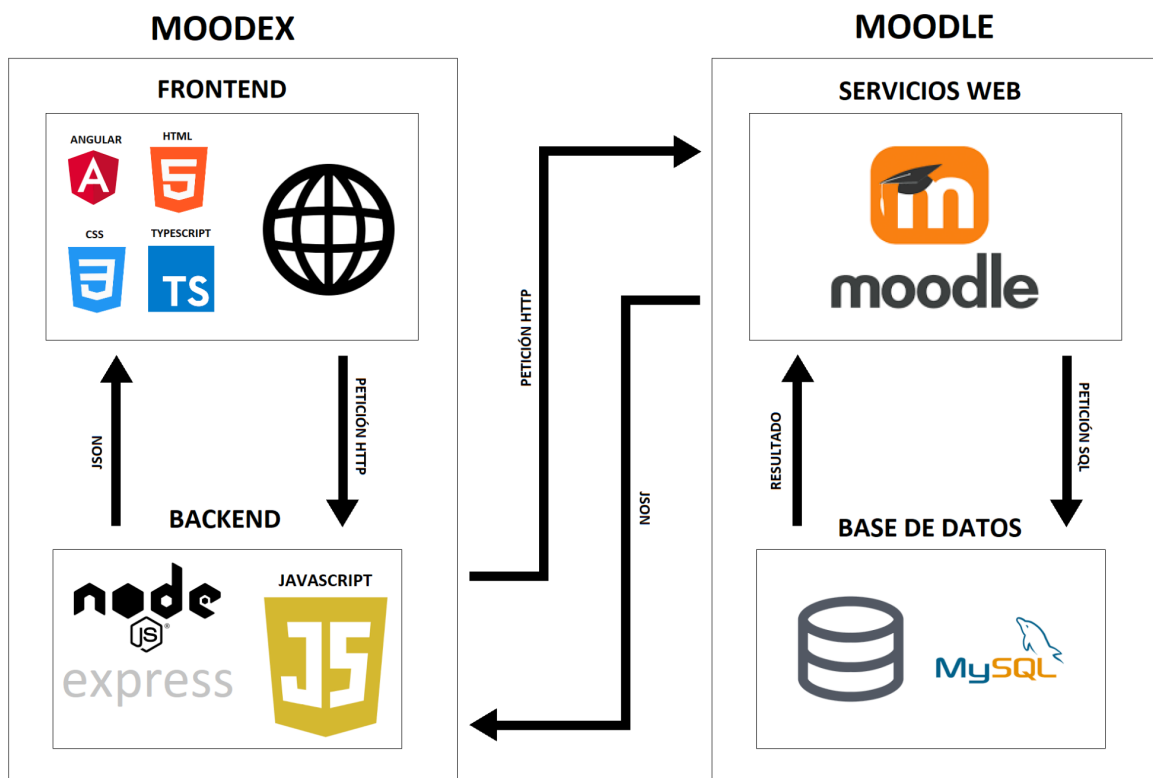


Fig. 7.- Arquitectura de la aplicación

CAPTURA DE REQUISITOS

Este apartado explica qué tiene que cumplir el trabajo que se va a desarrollar, qué necesidades tiene que satisfacer, quienes van a ser sus usuarios, etc. En otras palabras, expone los requisitos del proyecto. Para ayudar a comprender dichos requisitos figuran el Modelo de Casos de Uso y la Jerarquía de Actores, acompañando cada caso de uso y cada actor de una pequeña definición. En caso de querer una descripción más detallada de cada caso de uso, al final de esta memoria se encuentra el **Anexo I.- Casos de uso extendidos**.

Requisitos del proyecto

Los requisitos del proyecto están compuestos de requisitos funcionales y requisitos no funcionales. Los requisitos funcionales se dividen a su vez en requisitos principales y requisitos secundarios u opcionales.

Los requisitos funcionales son aquellos requisitos que declaran los servicios que proveerá el sistema y de qué manera reaccionará ante entradas particulares. En este caso, los requisitos funcionales principales son los siguientes:

- El sistema debe permitir iniciar sesión con las mismas credenciales con las que acceden a eGela al profesorado y a la administración. De igual modo, el sistema debe permitirles cerrar sesión.
- El sistema debe mostrar los cursos que imparte al profesorado una vez haya iniciado sesión.
- El sistema debe permitir escoger un curso al profesorado, entonces el sistema debe mostrar un calendario que le indique al profesorado los días apropiados para asignar una tarea.

Y los secundarios u opcionales estos otros:

- El sistema debe permitir realizar consultas referentes al uso que se le da a eGela a la administración una vez haya iniciado sesión. Además, el sistema debe permitir añadir filtros a dichas consultas.
- El sistema debe permitir escoger una fecha del calendario al profesorado para planificar una tarea.

Los requisitos no funcionales hacen referencia a las cualidades, restricciones y características del software. Estos son los requisitos no funcionales de los que dispondrá el sistema:

- El sistema debe ser estable y accesible.

Jerarquía de actores

La Jerarquía de actores es la estructura que se establece en orden a su criterio de subordinación entre los actores. La Fig. 8 representa la Jerarquía de actores del sistema:

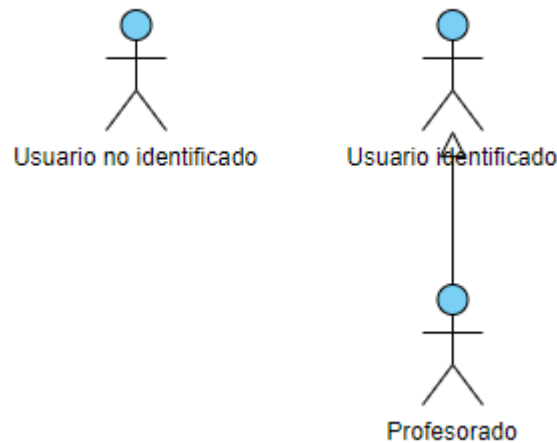


Fig. 8.- Jerarquía de actores

Los actores que la componen son los siguientes:

- **Usuario no identificado:** Representa a todo usuario que accede al sistema y no se identifica.
- **Usuario identificado:** Representa a todo usuario que accede al sistema y logra identificarse, en este caso, solo pueden identificarse el profesorado y la administración de eGela.
- **Profesorado:** Representa a todo usuario que accede al sistema, logra identificarse y el sistema le asigna rol de docente, es decir, forma parte del profesorado de eGela.

Modelo de casos de uso

El Modelo de casos de uso sirve para especificar la comunicación y el comportamiento del sistema mediante su interacción con los usuarios y/u otros sistemas. Es decir, muestra la relación entre los actores y los casos de uso en el sistema. La Fig. 9 representa el Modelo de casos de uso del sistema:

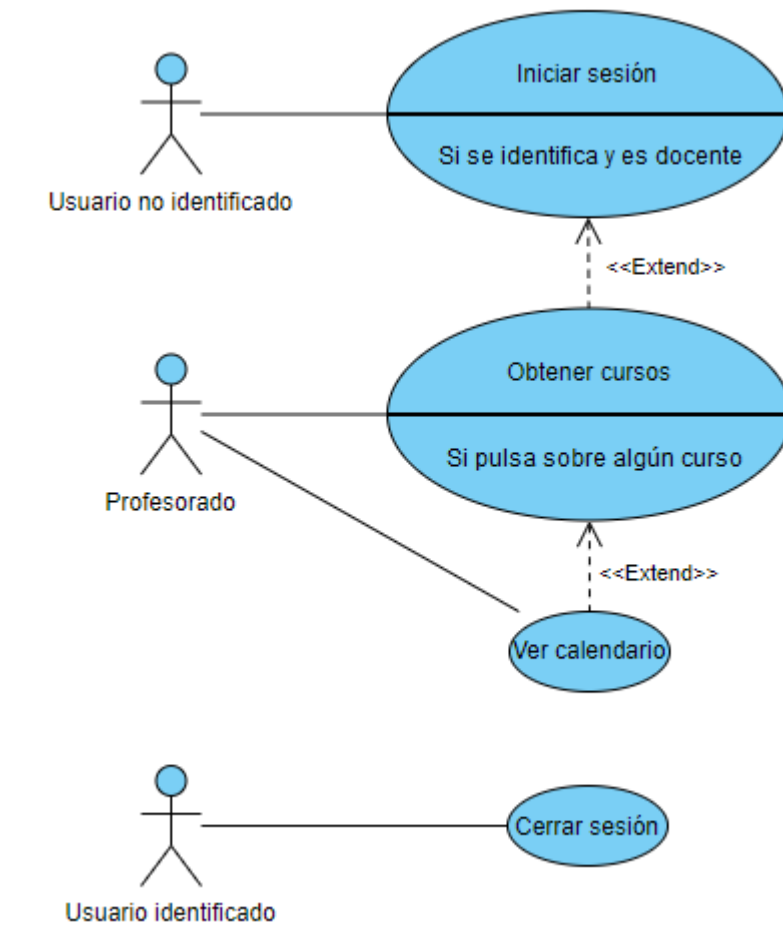


Fig. 9.- Modelo de casos de uso

Los casos de uso que lo componen son los siguientes:

- **Iniciar sesión:** Permite identificarse introduciendo sus credenciales al profesorado y a la administración de eGela.
- **Obtener cursos:** Permite obtener los cursos que imparte al profesorado una vez ha logrado iniciar sesión.
- **Ver calendario:** Permite ver un calendario al profesorado una vez ha pulsado sobre un curso. Este calendario le indica los días apropiados para asignar una tarea.
- **Cerrar sesión:** Permite cerrar sesión a los usuarios identificados.

Modelo de dominio

La aplicación no dispone de un Modelo de dominio dado que no almacena ningún tipo de información como tal. La aplicación cada vez que necesita determinada información de Moodle se la solicita al servicio web pertinente.

ANÁLISIS Y DISEÑO

Este apartado explica cómo se plantea el desarrollo del trabajo, es decir, describe y justifica las distintas partes en las que se divide la solución y como están estructuradas. Para facilitar su comprensión también se documenta el Diagrama de Clases. En caso de querer una descripción más detallada de cada funcionalidad, al final de esta memoria se encuentra el **Anexo II.- Diagramas de secuencia**.

La aplicación web se divide básicamente en dos partes: el backend y el frontend. El backend es la parte de la app que no es visible para el usuario final y su función es acceder a la información que él solicita, a través de la parte visible, para luego combinarla y devolvérsela. El frontend es la parte que conecta e interactúa con los usuarios que la visitan, es la parte visible, la que muestra el diseño, los contenidos y la que permite a los visitantes navegar por las diferentes páginas.

Estructura del backend

Moodex se trata de una API REST por lo que sigue la arquitectura REST, en la que se encuentra un canal de comunicación a través del cual el servidor envía los datos solicitados a los clientes.

Se implementan ciertos aspectos de seguridad, como son la autenticación mediante tokens e incorporación y comprobaciones de roles, dado que los datos se envían utilizando redes al alcance de cualquiera.

La estructura que se sigue a la hora de desarrollar el backend es una estructura multicapa, en la que cada capa cumple su función. Esta división por capas le otorga mayor modularidad y escalabilidad al software. Las capas mencionadas son la capa de seguridad y la capa del núcleo. La Fig. 10 representa la estructura del backend del sistema:

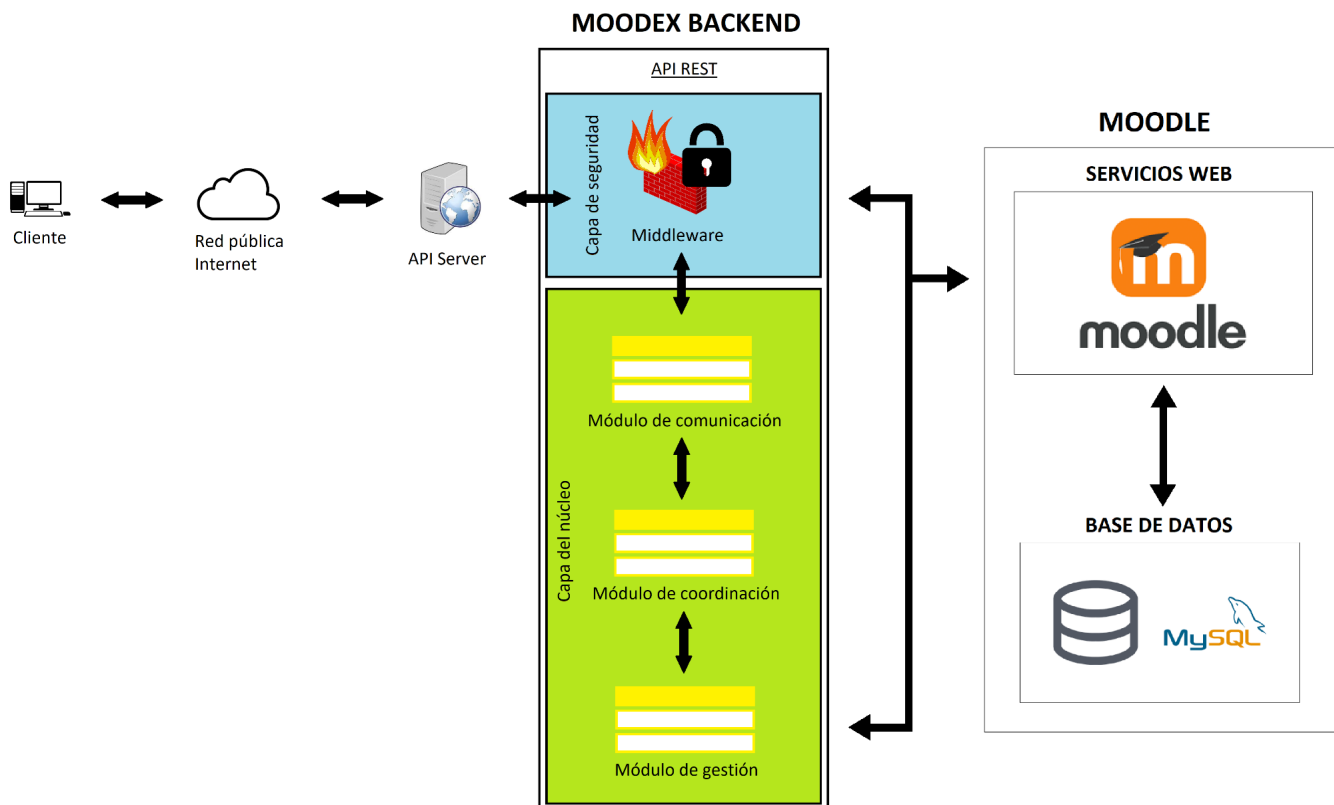


Fig. 10.- Estructura del backend

A continuación, se da paso a la descripción de cada capa explicando tanto su función como los elementos que la componen.

Capa de seguridad

La capa de seguridad es la primera capa, por la que entran las peticiones. Se encarga de comprobar las credenciales de cada petición y deniega o permite el acceso, en cada caso.

En esta primera capa se aloja el Middleware, que se encarga de validar el token de las peticiones, en cuyo caso permite el acceso.

El token es una secuencia de valores alfanuméricos cuya función es identificar a cada usuario, por lo que cada usuario dispone de un único token. El token se genera mezclando información relevante del usuario, una palabra secreta proporcionada por el sistema y que solo conoce el desarrollador, y un tiempo de expiración. Después se pasa a cifrar el token, lo que le otorga mayor seguridad. Finalmente, se le entrega el token al usuario que inicia sesión y este lo utiliza a modo de identificación cada vez que lleve a cabo alguna petición.

En este caso, el token se obtiene a través de los servicios web de Moodle, para ello es necesario proporcionar el nombre y la contraseña del usuario.

Cada vez que el Middleware recibe una petición, se comprueba la validez y la caducidad del token recibido y se determina si se debe procesar. En caso de que el token no sea válido o haya expirado, el Middleware rechaza la petición y devuelve al usuario el código de error estándar 401 UNAUTHORIZED de HTTP. En el caso contrario, si el token supera las pruebas de verificación, entonces la petición pasa a la capa del núcleo.

La Fig. 11 representa el comportamiento de la capa de seguridad del sistema:

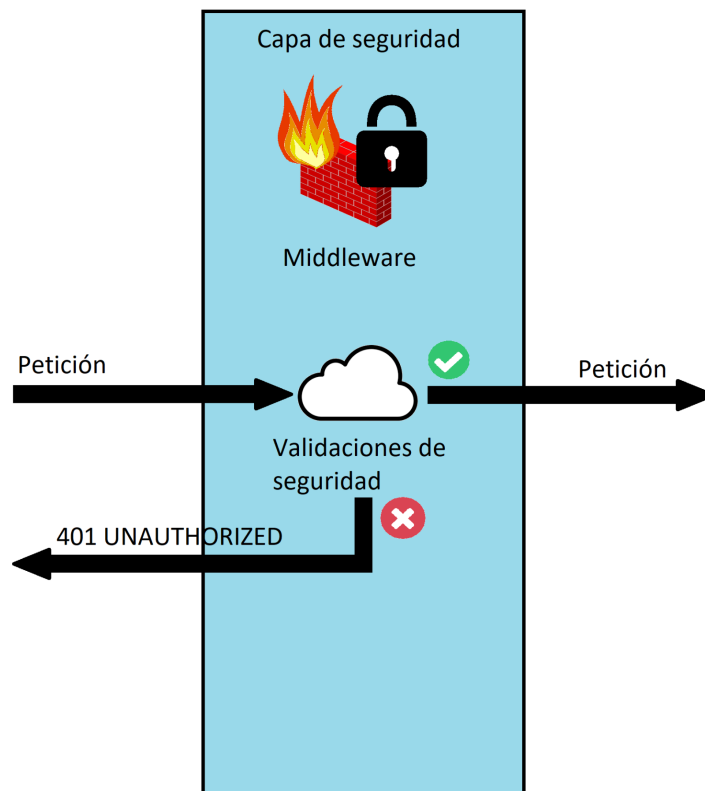


Fig. 11.- Capa de seguridad

Capa del núcleo

La capa del núcleo aloja una gran parte de la lógica de negocio referente al tratamiento de peticiones.

Esta capa está dividida en tres módulos independientes que trabajan de manera conjunta y se comunican entre ellos, dando así respuesta a las peticiones recibidas. Los módulos son los siguientes:

1. **Módulo de comunicación:** Este módulo es el punto de entrada de las peticiones, que confirma que el recurso solicitado existe y se encarga de dar validez a los parámetros presentes en la cabecera o en el cuerpo de la petición con el fin de comprobar que estos cumplen con la estructura requerida, en cuyo caso continúa el tratamiento de la petición. Una vez obtenida la respuesta a la petición, devuelve el resultado junto con el estado.
2. **Módulo de coordinación:** Este módulo se encarga de tratar los datos y recopilar la información necesaria para dar respuesta a la petición.
3. **Módulo de gestión:** Este módulo es el único que puede utilizar los servicios web de Moodle con el objetivo de resolver la petición.

Tras analizar los distintos módulos de la capa del núcleo se da paso a detallar más en profundidad el proceso de comunicación entre dichos módulos y los ficheros que contienen. La Fig. 12 representa el comportamiento de la capa del núcleo del sistema.

Cuando una petición accede a la capa del núcleo se analiza la ruta que posee en el fichero base de la aplicación, que dispone de una lista de todas las rutas entrantes junto con el nombre del fichero donde se tratan, para comprobar que existe la misma, en cuyo caso se redirige la petición al fichero correspondiente.

Este fichero se sitúa en el Módulo de comunicación y comprueba que la petición dispone de los parámetros establecidos. En caso de no recibir dichos parámetros, se genera un error y la petición no se termina de resolver.

Cuando el Módulo de comunicación valida una petición se redirige al nivel inferior, es decir, al fichero del Módulo de coordinación que se encarga de tratar el recurso solicitado, de modo que la petición se envía al método que realiza la acción solicitada. Dicha acción consiste en gestionar los ficheros del Módulo de gestión necesarios para resolver la petición y finalmente, obtener una respuesta.

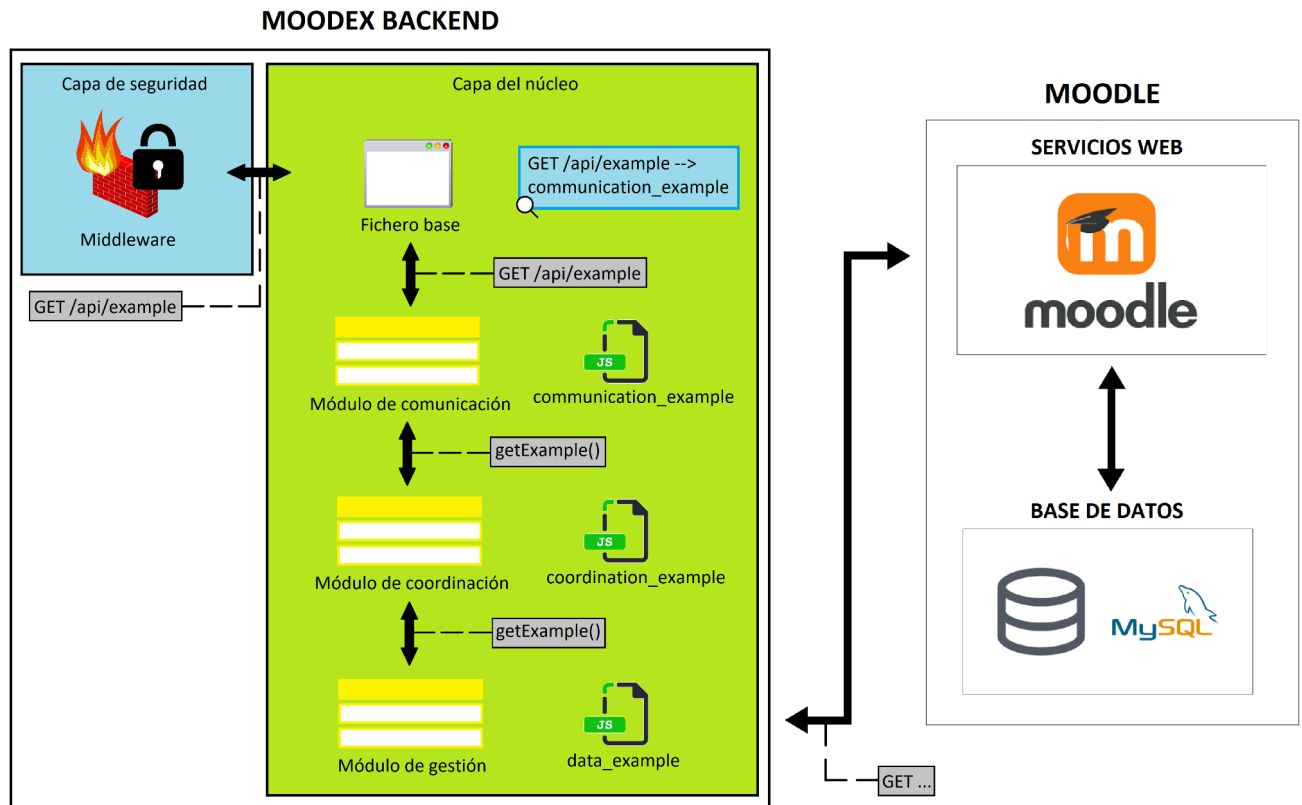


Fig. 12.- Capa del núcleo

Estructura del frontend

Para poder entender la estructura del frontend, debemos conocer la forma de trabajar de la herramienta de desarrollo, en este caso, el framework Angular. Al empezar un nuevo proyecto, Angular proporciona la estructura básica del código, lo que facilita construir una arquitectura que los navegadores puedan interpretar.

Angular funciona mediante componentes y como es un framework de desarrollo de webs SPA la aplicación dispone de un único punto de entrada, por lo que empieza por un solo componente, `app.component`. A partir de este se van añadiendo más formando una estructura con forma de árbol, y de este modo, se va generando la aplicación web.

Se pueden crear tantos componentes como funciones requiera el sistema. Un componente es como una etiqueta HTML nueva y puede ser cualquier porción o elemento de la vista, desde un formulario hasta un simple botón.

La Fig. 13 representa el ejemplo gráfico de una estructura de unos componentes de Angular, que acaban formando una interfaz gráfica completa:

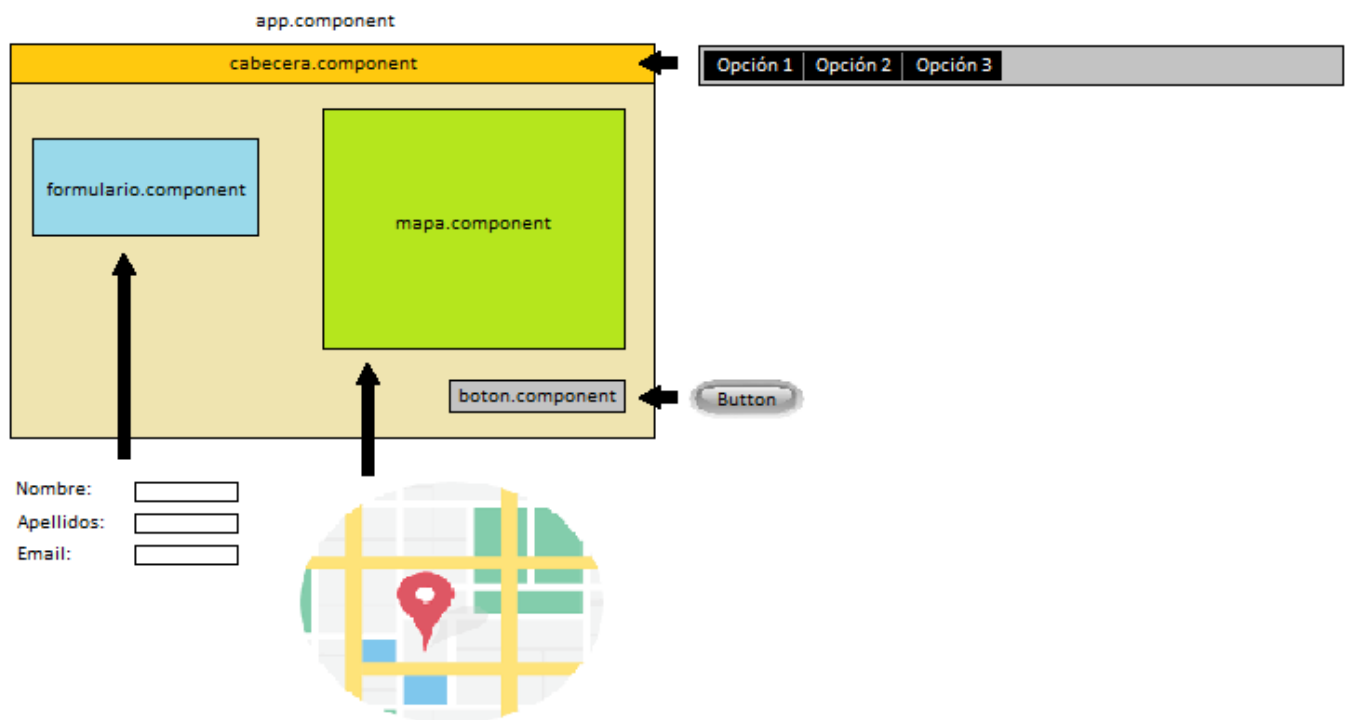


Fig. 13.- Estructura de unos componentes de Angular

Cada componente está compuesto a su vez de tres ficheros que definen su estructura, diseño y funcionalidad. Los ficheros son los siguientes:

- **.html:** Define la vista del componente.
- **.css:** Define el estilo de la vista.
- **.ts:** Define la funcionalidad del componente.

Trabajar con componentes mejora sensiblemente la organización, el mantenimiento y la reutilización, por ello, se consideran una pieza clave de las aplicaciones en Angular.

Tras entender qué son y para qué sirven los componentes en Angular, se puede llegar a la conclusión de que las aplicaciones web utilizan una gran cantidad de estos. A mayor complejidad y más funcionalidades tiene la aplicación entonces una mayor cantidad de componentes se utilizan. Para gestionar tales cantidades de componentes Angular proporciona módulos.

Un módulo puede encapsular o empaquetar tantos componentes como se quiera. En un módulo se suelen alojar componentes de una temática o lógica similar con el fin de mejorar la escalabilidad, usabilidad y modularidad de la aplicación.

Cada módulo dispone de un fichero `.module`, el cual define qué componentes encapsula e importa las librerías o módulos externos que se van a utilizar en dicho módulo.

La Fig. 14 representa el ejemplo gráfico de una estructura de un módulo de Angular, donde el módulo `module_A` está formado por los componentes `component_1` y `component_2`; el módulo `module_B`, que es otro módulo de la aplicación; y dos módulos de librerías externas ajenas a la aplicación, representados por los iconos con símbolo de interrogación:

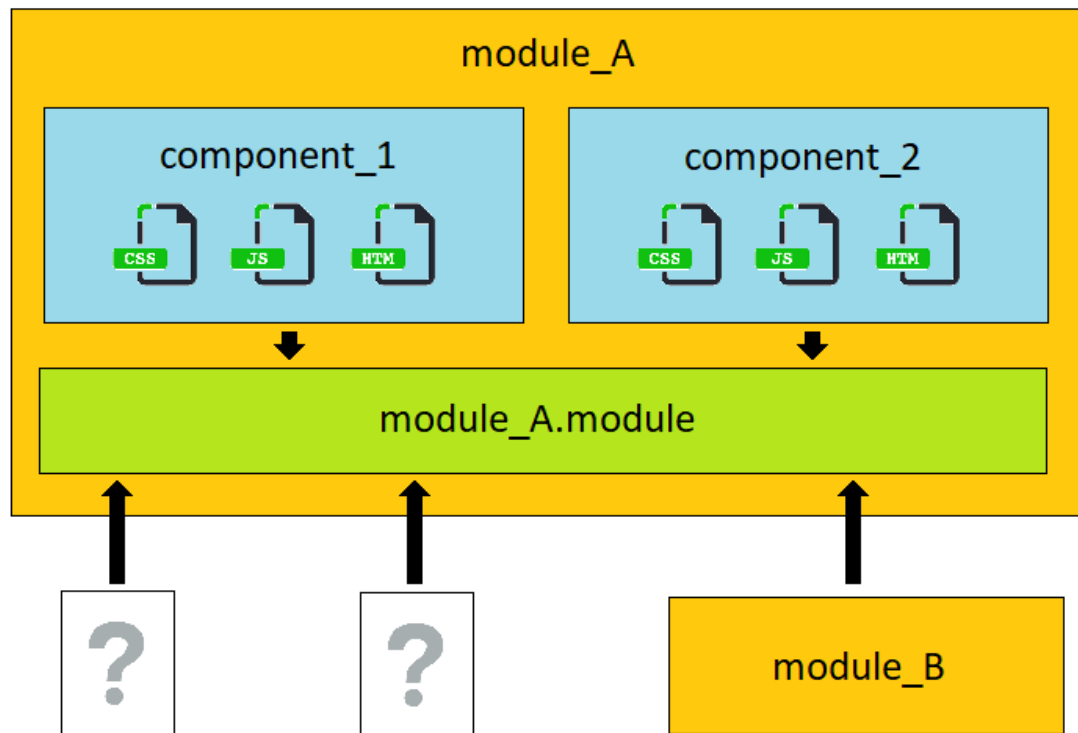


Fig. 14.- Estructura de un módulo de Angular

Para finalizar se explica en qué consisten los servicios, otro elemento importante en una estructura Angular.

Un servicio es una clase que contiene lógica de negocio o funciones de acceso a datos y que se invoca desde cualquier componente que lo tenga importado. Varios ejemplos de servicio son una clase de utilidades y una clase que realiza peticiones a una API REST.

Una vez descritos los elementos principales de un proyecto Angular y entendido cómo se relacionan entre ellos, es hora de explicar el diseño de este proyecto en concreto.

Se empieza con `app.component`, que es el componente principal en el que se van introduciendo el resto.

A su vez, se crean las carpetas `modules` y `services`. La carpeta `modules` contiene los distintos módulos y la carpeta `services` los diferentes servicios de los que dispondrá la aplicación web.

Se genera un módulo por cada vista o página de la web y un módulo adicional llamado shared, encargado de compartir aquellos componentes que sean necesarios en varias vistas. Ejemplos de dichos componentes son la cabecera del menú de la web, ventanas de avisos y mensajes de alerta.

Se ha mencionado anteriormente que los servicios pueden ser clases de utilidades o clases que realizan peticiones a la API REST, por lo que los servicios se separan en estos dos grupos, dividiendo la carpeta services en las subcarpetas generalServices, que contiene las clases de utilidades y clases de lógica de negocio, y webServices, que contiene aquellas clases que realizan peticiones a la API REST.

También se genera un router, que se encarga de cambiar las vistas cuando es necesario, es decir, cuando se produce un cambio en la URL.

La Fig. 15 representa el esquema general de la aplicación web:



Fig. 15.- Esquema general de la aplicación web

Diagrama de clases

El Diagrama de clases es uno de los tipos de diagramas más útiles en UML, ya que traza claramente la estructura de un sistema concreto al modelar sus clases, atributos, operaciones y relaciones entre objetos. La Fig. 16 representa el Diagrama de clases del sistema:

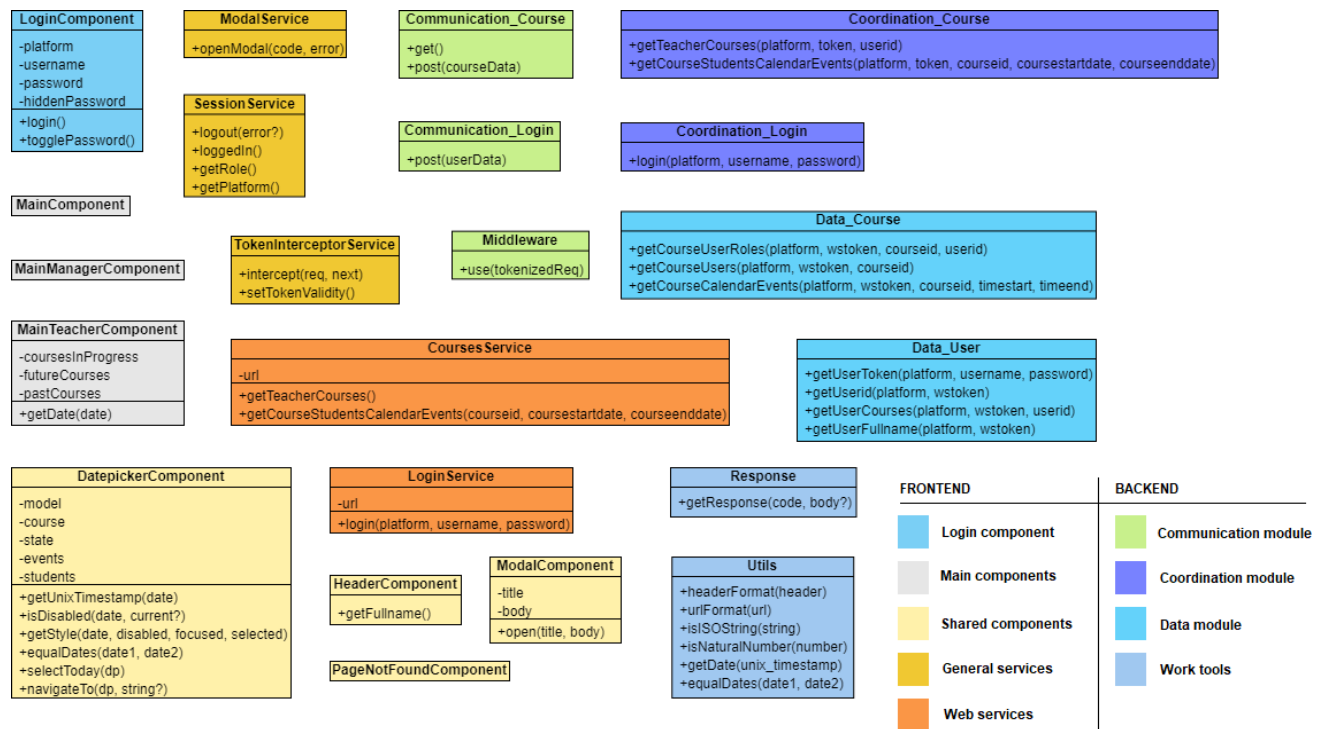


Fig. 16.- Diagrama de clases

Las clases que lo componen son las siguientes:

- **LoginComponent:** Este componente se trata del formulario de la página de inicio que permite al usuario introducir la plataforma, el nombre de usuario y la contraseña. En caso de que los datos introducidos sean correctos, le permite iniciar sesión.
- **MainComponent:** Este componente comprueba el rol del usuario y le redirige a la página principal que le corresponde.
- **MainManagerComponent:** Este componente se trata de la página principal del administrador. Actualmente no tiene nada especial.
- **MainTeacherComponent:** Este componente se trata de la página principal del profesor, que obtiene los cursos del profesor a través de determinado servicio web y los agrupa en base a su estado: en curso, futuro o pasado.
- **DatepickerComponent:** Este componente se trata del calendario que muestra al profesor mediante un código de colores los días más apropiados para asignar una tarea a los estudiantes del curso correspondiente. Para realizar esta recomendación el componente se basa en el porcentaje del alumnado que tiene al menos una tarea asignada determinado día.

- **HeaderComponent:** Este componente se trata de la cabecera que comparten todas las páginas de la aplicación web.
- **ModalComponent:** Este componente se trata de la ventana emergente que se abre en la aplicación web. Para ello debe recibir un título y un texto.
- **PageNotFoundComponent:** Este componente se muestra cuando se visita una ruta que el sistema no reconoce.
- **ModalService:** Este servicio de utilidades se encarga de abrir las ventanas emergentes.
- **SessionService:** Este servicio de utilidades permite ejecutar métodos relacionados con la sesión, como pueden ser cerrar la sesión del usuario, ver si el usuario está identificado, obtener el rol del mismo u obtener la plataforma a la que está conectado.
- **TokenInterceptorService:** Este servicio de utilidades se encarga de añadir la plataforma, el token, el rol y la fecha de caducidad de la sesión a las peticiones y de establecer la fecha de expiración de la sesión.
- **CoursesService:** Este servicio web se encarga tanto de obtener los cursos del profesor como de obtener los eventos de calendario de los estudiantes de determinado curso. En el primer caso realiza una petición GET al backend de la aplicación, en cambio, en el segundo caso le envía el identificador del curso junto a su fecha de inicio y a su fecha de fin a través de una petición POST.
- **LoginService:** Este servicio web envía la plataforma, el nombre de usuario y la contraseña al backend de la aplicación a través de una petición POST, para saber si el usuario puede iniciar sesión.
- **Communication_Course:** En caso de solicitar los cursos del profesor, este fichero del Módulo de comunicación comprueba que la petición posea el rol de docente. Si lo tiene entonces redirige la petición al fichero del Módulo de coordinación correspondiente, si no devuelve un error. En caso de solicitar los eventos de calendario de los estudiantes del curso, el fichero debe recibir el identificador del curso junto a su fecha de inicio y fecha de fin, además la petición debe poseer el rol de docente. Si los recibe y la petición tiene el rol de docente entonces se los envía al fichero del Módulo de coordinación correspondiente, si no devuelve un error.
- **Communication_Login:** Este fichero del Módulo de comunicación debe recibir la plataforma, el nombre de usuario y la contraseña. Si los recibe se los envía al fichero del Módulo de coordinación correspondiente, si no devuelve un error.
- **Middleware:** Este fichero del Módulo de comunicación forma parte de la capa de seguridad del backend y se encarga de validar la plataforma, el token y el rol del usuario y la caducidad de la sesión. Para ello extrae estos datos de las cabeceras de la petición y comprueba su validez. Si todos los datos son válidos la petición continúa, si no, se rechaza.
- **Coordination_Course:** Este fichero del Módulo de coordinación se encarga tanto de obtener los cursos del profesor como de obtener los eventos de calendario de los alumnos de determinado curso. Para ello, hace uso de los ficheros del Módulo de gestión.
- **Coordination_Login:** Este fichero del Módulo de coordinación se encarga de comprobar si las credenciales introducidas por el usuario corresponden a algún profesor o administrador de la plataforma. Para ello, hace uso de los ficheros del Módulo de gestión.
- **Data_Course:** Este fichero del Módulo de gestión se encarga de enviar peticiones a

determinados servicios web de Moodle para obtener información de los cursos, como puede ser los roles de un usuario en el curso, los usuarios del curso o los eventos de calendario del curso.

- **Data_User:** Este fichero del Módulo de gestión se encarga de enviar peticiones a determinados servicios web de Moodle para obtener información de los usuarios, como puede ser el token del usuario, el identificador del usuario, los cursos del usuario y el nombre completo del usuario.
- **Response:** Esta clase se encarga de formar y estandarizar las respuestas.
- **Utils:** Esta clase recoge todas las funciones genéricas que se suelen reutilizar.

DESARROLLO

Este apartado explica cómo se desarrolla el backend y el frontend del proyecto usando un lenguaje natural.

Desarrollo del backend

A continuación, se explica cómo se crea la estructura y los módulos que forman el backend del proyecto, la funcionalidad de los mismos, los problemas que surgen y las medidas de seguridad que se aplican.

Estructura de la aplicación

Las herramientas de desarrollo del backend son NodeJS, el framework Express y el entorno de desarrollo Visual Studio Code.

En este caso, se han utilizado las siguientes versiones:

- **NodeJS:** 16.17.0
- **Express:** 4.18.1
- **Visual Studio Code:** 1.76.2

Una vez instaladas las herramientas mencionadas, construido el esqueleto principal de la aplicación, instaladas las dependencias necesarias para comenzar el desarrollo y asegurado el correcto funcionamiento de la aplicación se puede dar por definida la estructura del proyecto. Sin embargo, dicha estructura contiene una serie de carpetas que son prescindibles e irrelevantes dado que no se les da ningún uso a lo largo del desarrollo del proyecto y que, por tanto, son eliminadas. Por ejemplo, la carpeta `views` almacena las vistas del sistema y un motor de representación de datos, por lo que, dado que la aplicación tiene un frontend independiente, esta carpeta no se utiliza. Por este mismo motivo, la carpeta `public` también se puede eliminar, puesto que se encarga de definir y almacenar las imágenes utilizadas por las vistas y las hojas de estilo, entre otras. La carpeta `routes`, donde se encuentran las rutas del sistema, también se elimina, dado que las rutas y los elementos de configuración del servidor y del API REST se definen en un único archivo, el `app.js`. Finalmente, el contenido de la carpeta `bin`, configuraciones para poner en marcha y realizar distintas pruebas con el servidor, también se traspasa al fichero `app.js`. A continuación, la Fig. 17 muestra la estructura mínima necesaria para el proyecto:

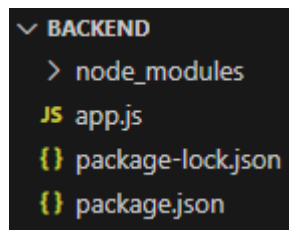


Fig. 17.- Estructura básica del backend

Una vez eliminadas las carpetas y archivos innecesarios, quedan varios elementos importantes que hay que analizar. Por un lado, el fichero base de configuración `app.js` es semejante a la clase principal de la aplicación, y como tal, se encarga de configurar y arrancar el servidor, de definir las rutas y asociarlas con las clases correspondientes, etc. Por otro lado, el fichero `package.json` contiene información importante de la aplicación, como son el nombre y la versión, además gestiona las dependencias del proyecto, como son los módulos o las librerías externas, que se encuentran en la carpeta `node_modules`. Por último, el archivo `package-lock.json` sirve para llevar un control de las versiones con las que se trabaja.

A continuación, se crea la carpeta `src` en la carpeta raíz del proyecto y se introduce el fichero base de configuración `app.js` en su interior. A partir de este momento, el desarrollo del backend se desenvuelve dentro de esta nueva carpeta.

A estas alturas, la estructura inicial ya está creada y todas las configuraciones están indicadas en el fichero base, por lo que es hora de aplicar la estructura de capas o módulos descrita en el apartado de Análisis y diseño de la forma más modular posible. Para ello, se crea una carpeta por cada uno de los módulos (Módulo de comunicación, Módulo de coordinación y Módulo de gestión). En concreto, la carpeta relacionada con el Módulo de gestión, al ser la que solicita los datos a Moodle, se llama `data`.

De la misma manera que la carpeta `node_modules` contiene los módulos y las librerías externas, se crea la carpeta `modules` para que almacene los módulos generados (Fig. 18).

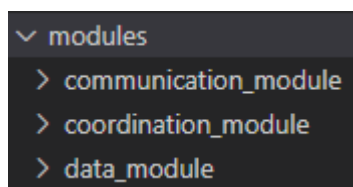


Fig. 18.- Carpeta modules

También se crea la carpeta `work_tools` (Fig. 19) con el propósito de almacenar los ficheros con funcionalidades o métodos comunes a toda la aplicación, lo que aporta una mayor reutilización de código al proyecto. A continuación, se describen brevemente cada uno de los ficheros:

- **response.js:** Este fichero es una factoría de respuestas, es decir, se encarga de preparar y estructurar las respuestas que devuelve la API REST.
- **utils.js:** Este fichero recoge todas las funciones genéricas que se suelen reutilizar.

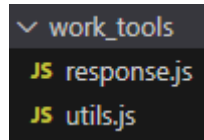


Fig. 19.- Carpeta work_tools

A continuación, la Fig. 20 muestra la estructura completa del backend:

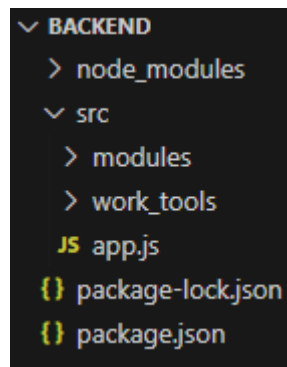


Fig. 20.- Estructura completa del backend

Rutas y proceso de comunicación

Como ya se ha explicado antes, el backend dispone de un proceso de comunicación entre ficheros que tramita las solicitudes entrantes con la mayor eficiencia posible. Para ejecutar este proceso satisfactoriamente es preciso definir las rutas del sistema en el fichero que primero se ejecuta en el momento en que entra una petición al sistema. Esto debe ser así puesto que, en caso de configurar las rutas en otro fichero, las peticiones no son redirigidas y esto produce un error en la ejecución del proceso. En este caso, dicho fichero es app.js. El siguiente fragmento muestra las rutas declaradas en el fichero app.js.

```
// Módulos de comunicación
const communication_login = require('./modules/communication_module/communication_login');
const communication_course = require('./modules/communication_module/communication_course');

// Middleware
const middleware = require('./modules/communication_module/middleware');

// Rutas
app.use('/auth', communication_login);
app.use('/api', middleware);
app.use('/api/course', communication_course);
```

Como se puede observar en el fragmento anterior, en primer lugar se crean una serie de constantes que actúan como controladores y se encargan de acceder a los recursos del sistema. El contenido de estas constantes se trata de un path, o ruta, que apunta a un fichero. Las tres apuntan a un fichero situado en el Módulo de comunicación, la primera capa del sistema donde se tratan peticiones entrantes. La tercera de ellas apunta al fichero `middleware.js`, en concreto. En segundo lugar, se define cada ruta del sistema junto con el fichero al que debe redirigir las peticiones, es decir, junto a la constante que le corresponda.

Analizando más en profundidad se puede apreciar que las rutas definidas se agrupan por funcionalidad, por ejemplo, todas las peticiones relacionadas con los cursos tienen, como inicio de ruta, `/api/course`. Asimismo, todas las rutas, excepto una, poseen la parte `/api` en común. Además, hay una ruta que solamente tiene esta parte como ruta, y es precisamente en esta donde se encuentra el Middleware de la aplicación.

Middleware

El Middleware es un programa que está situado entre el sistema y los módulos. Principalmente se trata de una capa oculta que administra y comprueba distintos tipos de datos, entre otras cosas. En este caso, el Middleware de la aplicación se encuentra en el fichero `middleware` y se encarga de comprobar que el token recibido junto a la cabecera de la petición entrante es válido. Si el token es válido, el Middleware cede el control a la próxima parte de la ruta. En caso contrario, el Middleware califica la petición como inválida y el sistema se encarga de indicar que la petición es errónea y de negar el acceso a los recursos solicitados.

La única ruta que no pasa por el Middleware es la ruta `/auth` puesto que corresponde a la petición de identificación del sistema, y en ese momento aún no se almacena el token de ningún usuario en la cabecera de la petición, lógicamente.

Por ejemplo, en el caso de la ruta `/api/course`, una vez que la petición pasa el control del fichero que se encuentra en el path correspondiente a la ruta `/api`, esta pasa al path de la ruta completa `/api/course`, es decir, al fichero `communication_course`. En otras palabras, cuando la petición pasa el control del Middleware satisfactoriamente, se redirecciona a la primera capa del sistema, el Módulo de comunicación.

Módulo de comunicación

El Módulo de comunicación se comunica de manera directa con las peticiones entrantes, por lo que construye las funciones de tal forma que pueda trabajar con las rutas que contienen dichas peticiones, esto es, crea métodos de ruta. Este tipo de método es derivado de los métodos HTTP y se adjunta a una instancia de la clase express. Solo puede haber un método para cada tipo de petición HTTP a cada ruta, en otras palabras, no pueden existir dos peticiones GET a la misma ruta con el mismo número de parámetros. Por esta razón, para acceder a un método determinado es necesario utilizar la ruta y el tipo de petición HTTP preciso. El siguiente fragmento de código muestra un ejemplo de método de ruta.

```
// Obtener ejemplo
communication_example.get('/', (req, res) => {
  coordination_example.getExample()
    .then((data) => {})
    .catch((error) => {});
});
```

El método de ruta utiliza la clase Router de la librería express, que permite los redireccionamientos de ruta, junto con el tipo de petición HTTP que se trata en esa función. A continuación, se aplican la ruta, que es la parte restante y no analizada de la ruta analizada en el fichero app.js, y la función. Por ejemplo, si el fichero app.js declara que para llegar al fichero communication_example es necesario que la ruta empiece por /api/example, entonces si la ruta original de la petición es /api/example/example1, la ruta que aplica el método debe ser /example1. Si la ruta es analizada por completo al pasar por el fichero app.js, la ruta que aplica el método es una barra /.

Uno de los problemas encontrados a lo largo del desarrollo del backend consiste en la asincronía, que es parte del proceso de comunicación de NodeJS y dificulta el tratamiento de datos. Las funciones básicas no sufren este problema, no así como las funciones que buscan obtener una serie de datos, para luego efectuar distintas acciones con ellos. Esto sucede ya que el código no se queda pendiente a la respuesta al realizar la petición, sino que procede a ejecutar las siguientes acciones sin los datos necesarios, lo que genera errores. Para solventar este problema se barajan dos posibles soluciones: utilizar Promises o utilizar el método async-await.

En definitiva, las dos alternativas devuelven los datos solicitados a través de una promesa (return new Promise). En caso de que se obtengan los datos satisfactoriamente, la promesa los devuelve mediante el método resolve(datos), en cambio, en caso de que se produzca algún error, los devuelve mediante el método reject(datos). A partir de este momento es cuando empiezan las diferencias entre los métodos mencionados.

Por un lado, el método Promises utiliza un then y un catch para recoger los datos devueltos junto a la promesa. Estos funcionan de la misma manera que las excepciones, es decir, si la promesa se resuelve con un resolve va por el then de la función que la invoca, por el contrario, si se resuelve con un reject la promesa entra por el catch. Esta distinción entre las

peticiones exitosas y las fallidas facilita la construcción de respuestas y aporta un mayor control de los errores, en consecuencia, el sistema logra estabilidad. El mayor inconveniente de esta alternativa es que en caso de solicitar datos mediante distintas funciones se tienen que anidar las promesas, lo que genera una gran cantidad de código. Este método se ve aplicado en el fragmento de código anterior, donde la función `getExample()` es la encargada de devolver la promesa junto con los datos.

En este caso, el problema de asincronía del Módulo de comunicación se resuelve mediante el método Promises, puesto que el Módulo de comunicación llama a una única función del módulo inferior, con lo que el código que se genera no resulta tan voluminoso.

Por otro lado, el método `async-await` no tiene el mismo inconveniente que el método Promises, no genera gran cantidad de código, ya que recoge la promesa utilizando la función `async` delante de la función principal y la función `await` delante del método que devuelve la promesa. De este modo, se declara que la función es asíncrona y que el código debe esperar hasta que se obtengan los datos solicitados, y no se utiliza una estructura determinada. Este método se puede ver aplicado en el Módulo de coordinación, que se analiza a continuación.

Módulo de coordinación

El Módulo de coordinación recibe la petición desde el Módulo de comunicación, y se encarga de obtener los datos o de efectuar las acciones solicitadas. Para ello, a veces, debe emplear varias divisiones del Módulo de gestión en un mismo método. Por ejemplo, es posible que para resolver una petición tenga que obtener un dato intermedio necesario para obtener el dato o efectuar la acción solicitada.

Por ello, este módulo también presenta el problema de la asincronía. Sin embargo, esta vez se opta por utilizar el método `async-await` dado que se pueden llegar a tratar distintos datos y usar el método Promises puede suponer aumentar las líneas de código en gran medida y dificultar su comprensión. El siguiente fragmento de código muestra un ejemplo del uso del método `async-await`.

```
// Obtener ejemplo
coordination_example.getExample = () => {
  return new Promise(async (resolve, reject) => {
    try {
      const example = await data_example.getExample();
      resolve(example);
    }
    catch (error) {
      reject(50000);
    }
  });
};
```

Módulo de gestión

El Módulo de gestión es el nivel más bajo del proceso de comunicación, y se encarga de interactuar directamente con los servicios web de Moodle para obtener información o efectuar acciones solicitadas por el Módulo de coordinación. Una vez que el Módulo de gestión cumple las demandas del módulo superior, le devuelve el resultado y finaliza su labor hasta la siguiente petición.

En definitiva, la finalidad de esta capa consiste en realizar las peticiones HTTP correspondientes a los servicios web de Moodle, para luego devolver el resultado a la capa superior.

Respuestas y códigos de error

Una vez finalizado el proceso de comunicación entre las distintas capas, los datos solicitados llegan al Módulo de comunicación, por lo que, es hora de formar y devolver una respuesta a la petición entrante. En este punto se origina un problema respecto a cómo devolver las respuestas debido a la gran variedad de solicitudes que se pueden realizar. No es viable que cada respuesta adopte una forma distinta, por lo que, se opta por estandarizar el formato de todas las respuestas, tanto para las que devuelven datos y son correctas como para las incorrectas que devuelven algún tipo de error.

Para que todos los métodos de ruta puedan enviar las respuestas de forma modular, se crea el fichero `response.js` dentro de la carpeta `work_tools`, directorio que sirve para almacenar herramientas comunes y para reutilizar código. Este nuevo fichero se encarga de tratar y de dar el formato adecuado a todas las respuestas y códigos de error existentes. Su uso proporciona centralización del código, lo que facilita la labor de añadir o modificar alguna respuesta dado que todas se encuentran en este mismo fichero.

El siguiente fragmento de código muestra cómo el Módulo de comunicación llama a un método del fichero `response` para formatear las respuestas. Cuando la petición se gestiona correctamente, el método recibe un código y los datos obtenidos como parámetros. En cambio, cuando la solicitud va mal, solamente recibe el código de error. Esto es así porque el método analiza el número de parámetros recibidos para saber si tiene que formatear y mandar algún dato en el body de la respuesta.

```

.then((data) => {
    const json = response.getResponse(20000, data);
    const status = json.status;
    res.status(status).json(json);
})
.catch((error) => {
    const json = response.getResponse(error);
    const status = json.status;
    res.status(status).json(json);
});

```

Una vez desarrollada la parte del método que determina cuándo enviar datos en la respuesta, es necesario saber gestionar los códigos internos de la aplicación para dar con el status y el mensaje correspondiente. Estos se obtienen a partir del código interno que genera el sistema y con ellos se forma la respuesta. A continuación se describen las tres partes mencionadas:

- **status:** El status o código genérico es conocido por ser la respuesta al servidor. En concreto, en este proyecto se trabaja con seis códigos genéricos. Uno de ellos indica que la operación se realiza correctamente, el resto determinan el motivo por el que falla.
- **code:** El code se trata del código interno de la aplicación y corresponde a un único status. El código interno indica que operación se ha efectuado y determina su éxito o el motivo de su fracaso.
- **msg:** El mensaje que indica el resultado de la operación y que se envía en la respuesta junto al código genérico y al código interno a los que corresponde.

```

{"status": generateStatus, "code": generateCode, "msg": generateMsg, "body": data}
{status:200, code:20000, msg:'Token y rol obtenidos'},
{status:400, code:40000, msg:'¡No se han recibido la plataforma, el nombre de
usuario y la contraseña, o la plataforma no dispone del formato de URL adecuado, o
el nombre de usuario o la contraseña no es una cadena!'}

```

Por otra parte, al momento de crear un código interno en la aplicación se tiene en cuenta el status al que pertenece. Esto se ve reflejado en el fragmento anterior, donde se ve que para el status 200 el primer código interno es el 20000, de manera que el segundo es el 20001, y así sucesivamente. Los seis códigos genéricos con los que se trabaja para desarrollar las respuestas, ya mencionados, son los siguientes:

- **200 (OK):** Indica que todo se realiza correctamente y el servidor responde con un OK.
- **400 (Bad Request):** Los datos enviados junto a la solicitud no son correctos o no están completos.

- **401 (Unauthorized):** Fallos de autorización relacionados con la validez o la caducidad del token, con el inicio de sesión o con la petición de datos.
- **403 (Forbidden):** Error a la hora de ejecutar una operación sin autorización.
- **404 (Not Found):** No se encuentra la URL o el método solicitado.
- **500 (Internal Server Error):** Error interno del servidor.

Por último, una vez formada la respuesta es devuelta al cliente y concluye el proceso de comunicación hasta la próxima solicitud.

Planificación para los docentes de Moodle

Para que Moodex proporcione a los docentes de Moodle la información que les pueda ser útil para planificar las tareas de los cursos que imparten, se han desarrollado una serie de funciones: la de identificación, otra para obtener los cursos que imparten y la última para obtener los eventos de calendario de los estudiantes de un determinado curso.

Antes de empezar a describir cada una de las funciones, cabe de decir que las llamadas a los servicios web de Moodle que se van a mencionar a continuación se realizan a través de la librería Axios²⁸. *“Axios es una librería JavaScript que puede ejecutarse en el navegador y que nos permite hacer sencillas las operaciones como cliente HTTP, por lo que podremos configurar y realizar solicitudes a un servidor y recibiremos respuestas fáciles de procesar.”* (11). Además, para que la aplicación funcione correctamente la plataforma Moodle que quiera hacer uso del sistema debe otorgar los siguientes permisos al rol de Usuario identificado:

- **webservice/rest:use:** Permiso necesario para poder utilizar el protocolo REST al realizar llamadas al servicio web.
- **moodle/course:view:** Permiso necesario para que el docente pueda acceder a la información de los cursos de los que no forma parte.
- **moodle/user:viewdetails:** Permiso necesario para que el docente pueda ver el perfil de sus alumnos en el resto de cursos.
- **moodle/calendar:manageentries:** Permiso necesario para que el docente pueda obtener los eventos de calendario del resto de cursos.

Identificación

El requisito principal para iniciar sesión en Moodex es formar parte del profesorado o de la administración de la plataforma de Moodle a la que se quiere conectar y de la que quiere obtener la información determinado usuario. Para ello el usuario debe proporcionar tanto la plataforma, como las credenciales utilizadas para conectarse a ella.

²⁸ Axios: <https://axios-http.com/es/docs/intro>

Una vez la aplicación recibe los parámetros mencionados, intenta obtener el token del usuario en la plataforma para asegurarse de que es usuario de la misma. Para ello solicita el token al servicio web que incluye Moodle **Moodle mobile web service (moodle_mobile_app)**, al que le pasa la plataforma y las credenciales como parámetros y devuelve el token como respuesta. Si no se obtiene respuesta no es usuario de la plataforma, por lo que se detiene el proceso de identificación.

Si se obtiene respuesta quiere decir que es usuario de la plataforma y a partir de este momento, a la hora de solicitar información a los servicios web de Moodle se le facilitará la plataforma y el token del usuario, entre otros datos. El token reemplaza a las credenciales.

A continuación, se obtiene el identificador del usuario a través del servicio web **core_webservice_get_site_info**, que devuelve el identificador del usuario como respuesta. Este dato intermedio es necesario pasárselo como parámetro al servicio web **core_enrol_get_users_courses** para obtener una lista con los cursos del usuario. Si no se recibe ningún curso significa que el usuario no imparte ni administra ningún curso, por lo que no puede iniciar sesión.

En el caso contrario, se escoge uno de los cursos y se llama al servicio web **core_user_get_course_user_profiles**, que requiere el identificador del curso y el identificador del usuario y devuelve los roles del usuario en el curso como respuesta. Tras obtener los roles, se comprueba a ver si entre ellos se encuentra el de administrador o el de profesor. Si el usuario no dispone de ninguno, se pasa al siguiente curso.

Si dispone de alguno, entonces significa que el usuario es profesor o administrador, por lo que se devuelve el token y el rol correspondiente. Con estos datos se puede decir que el usuario ha logrado iniciar sesión en Moodex.

Cursos del docente

En este caso, el requisito básico para obtener los cursos del profesor es ser profesor, es decir, que en la identificación el sistema haya asignado rol de profesor al usuario. El usuario también tiene que proporcionar la plataforma de la que obtener la información, además de su token y su identificador.

Este algoritmo es bastante similar al de la identificación, sin embargo, en este caso ya se cuenta con el token y el identificador del usuario y se sabe que el usuario es profesor de al menos un curso. Por lo que se comienza obteniendo los cursos del usuario directamente.

Cabe decir que en este caso se recorren todos los cursos, no como en el algoritmo anterior, que se recorren hasta encontrar el rol de administrador o el de profesor en alguno de ellos.

Por cada curso del usuario se obtienen los roles del usuario en los mismos y se mira a ver si entre ellos se encuentra el de profesor, en cuyo caso se recopila la información del curso necesaria (identificador, nombre, fecha de comienzo, fecha de fin y si el profesor tiene permiso de edición) y se almacena en un objeto, y este objeto a su vez, se almacena en una lista.

Al finalizar de procesar todos los cursos del usuario, se devuelve la lista mencionada.

Eventos de calendario de los estudiantes del curso

Por último, el requisito para obtener los eventos de calendario de los estudiantes de un determinado curso es el mismo que el de la función anterior, ser profesor, en este caso, ser profesor de este curso. El usuario que solicite esta información debe proporcionar la plataforma y su token, además del identificador, la fecha de inicio y la fecha de fin del curso.

Para empezar hay que dar con los alumnos del curso, por lo que se llama al servicio web **core_enrol_get_enrolled_users**, al que se le pasa el identificador del curso mencionado y devuelve los usuarios del curso como respuesta. Luego, se miran los usuarios uno a uno para ver si son alumnos.

Por cada alumno se obtienen los cursos y por cada uno de los cursos se mira a ver si también dispone del rol de alumno. Lo lógico es pensar que si es alumno de un curso es alumno del resto, pero se comprueba por si acaso.

Si lo es, entonces se llama al servicio web **core_calendar_get_calendar_events**. A este servicio hay que facilitarle tanto el identificador del curso del alumno, como la fecha de inicio y la fecha de fin del curso mencionado al comienzo, y devuelve los eventos de calendario del curso como respuesta.

Si el curso del alumno tiene eventos de calendario, entonces por cada uno de ellos se comprueba si es visible y si es un evento de curso. Esta comprobación se hace porque el profesor puede tener sin visibilizar varios eventos por alguna razón y porque hay otros tipos de eventos que no son de curso, y ninguno de ellos se debe tener en cuenta.

Una vez realizada la comprobación, si todo es correcto, se procede a obtener las fechas en las que sucede el evento de calendario. Un evento de calendario se puede alargar en el tiempo, es decir, puede durar más de un día y hay que tener en cuenta cada uno de ellos.

Hay que almacenar cada una de las fechas del evento junto al alumno que lo tiene asignado. Para ello disponemos de una lista de eventos. Pero se actúa de manera distinta en los siguientes casos:

- Si la fecha todavía no forma parte de la lista, entonces se añade junto a una lista de alumnos compuesta únicamente por el alumno.
- Si la fecha ya forma parte de la lista y el alumno ya le acompaña, entonces no se hace nada.
- Si la fecha ya forma parte de la lista, pero el alumno no le acompaña, entonces se añade el alumno a la lista de alumnos que tienen asignado al menos un evento de calendario en dicha fecha.

Finalmente, se devuelve la lista de eventos y el número total de alumnos del curso.

Seguridad en la parte servidor

La seguridad es uno de los aspectos más importantes a tener en cuenta a la hora de desarrollar una aplicación ya que, en cierta medida, puede evitar que el sistema se vea afectado por ataques informáticos. Por esta razón, Moodex pone en práctica diferentes y robustas medidas preventivas que aportan seguridad al sistema, entre ellas, el uso de tokens, la validación de datos en el Módulo de comunicación y la presencia de roles en el sistema.

Utilización de tokens

La primera medida de seguridad que se aplica es el uso de tokens. Mediante su uso el sistema puede comprobar, entre otras cosas, si el usuario que solicita el recurso tiene acceso o si su sesión continúa activa. De este modo, el intercambio de datos a través de la red se vuelve más seguro y se hace más difícil que un ataque informático logre acceder al sistema.

Para llevar un control minucioso de las acciones relacionadas con los tokens se dispone de los ficheros `data_user.js` y `middleware.js`. El primero se encuentra en la carpeta `data_module`, directorio de la última capa del sistema. El segundo, en cambio, se encuentra en la carpeta `communication_module`, directorio de la primera capa del sistema. Cada uno de estos ficheros tiene su cometido: el fichero `data_user.js` se encarga de obtener a través de los servicios web de Moodle información de los usuarios, como puede ser el token, y el fichero `middleware.js` se encarga de comprobar la validez del token que viaja en la cabecera de las peticiones entrantes.

Respecto al primer cometido, el token se obtiene a través de un servicio web de la plataforma Moodle y se almacena en el `sessionStorage` del navegador junto con la plataforma, el rol y el nombre completo del usuario y una fecha y hora de validez. El tiempo de validez de un token en Moodex se ha establecido en 30 minutos, por lo que, si un usuario está esa cantidad de tiempo inactivo, el token caduca. Si el usuario desea realizar una nueva operación debe volver a iniciar sesión en el sistema.

En cuanto al segundo, el Middleware es la capa encargada de aplicar la función que valida las peticiones entrantes, para ello, las intercepta, accede a las cabeceras, obtiene los datos de autorización necesarios y, finalmente, comprueba la validez de estos datos. Para empezar, extrae la plataforma de la cabecera `Platform` y comprueba que tenga formato de URL. Luego, extrae el token de la cabecera `Authorization` y comprueba su validez a través de otro servicio web de Moodle. También extrae el rol del usuario de la cabecera `Role` y comprueba que este exista. Por último, comprueba la caducidad del token, la cual se encuentra en la cabecera `Expiration`. En caso de que la plataforma no tenga formato de URL, el token no sea válido o expire o el rol no exista, se rechaza la petición, evitando así posibles ataques. En caso contrario, la petición sigue adelante.

Seguridad en el Módulo de comunicación

El Módulo de comunicación también aplica varias medidas de seguridad. La primera de ellas consiste en comprobar el nivel de acceso del usuario para saber si tramitar su solicitud o no. Para ello, cada función tiene una serie de roles permitidos, así, cuando llega una petición hasta este punto del sistema, se comprueba el rol que va junto al token, y si este tiene permiso, la petición sigue adelante. En cambio, si el usuario no dispone de los permisos necesarios, se rechaza la petición y se indica el error de autorización. Moodex tiene dos roles definidos: el rol de docente y el rol de administrador.

La segunda, por su parte, trata de revisar los datos que viajan en la cabecera o en el cuerpo de la petición para confirmar que cumplen la estructura correcta y son del tipo de datos esperado. Por ejemplo, cuando una petición es válida y llega al Módulo de comunicación, si esta tiene que llevar tres parámetros con un nombre y un tipo concreto, este módulo comprueba que esto se cumpla, y si no es así, la petición no sigue adelante. Para llevar a cabo esta comprobación, se analiza si están presentes en la petición todos los parámetros previstos, en cuyo caso, se verifica que sean del tipo esperado. Esto es, si el parámetro nombre tiene que ser una cadena de texto, se confirma que realmente lo sea, y si en cambio se trata de un número, se deniega la petición. De este modo se evita que lleguen caracteres no controlados al sistema.

Gestión de las contraseñas de usuario

Por último, cabe mencionar que las contraseñas de los usuarios de la web no se almacenan en ningún sitio. Las contraseñas introducidas por los usuarios al intentar iniciar sesión únicamente se utilizan para obtener el token, por lo que no es necesario aplicar un proceso de cifrado sobre ellas ya que no hay manera de robar esta información, y por tanto, de saber cuál es la contraseña de cada usuario. En caso de obtener el token, la contraseña introducida es correcta, en caso contrario, la contraseña es incorrecta.

Desarrollo del frontend

A continuación, se explica cómo se crea la estructura, los módulos y los componentes que forman el frontend del proyecto, la funcionalidad de los mismos, los problemas que surgen y las medidas de seguridad que se aplican.

Estructura de la aplicación

Las herramientas de desarrollo del frontend son el framework Angular y el entorno de desarrollo Visual Studio Code.

En este caso, se han utilizado las siguientes versiones:

- **Angular:** 15.1.2
- **Angular CLI:** 15.1.2
- **Visual Studio Code:** 1.76.2

Una vez instaladas las herramientas mencionadas, construido el esqueleto principal de la aplicación, instaladas las dependencias necesarias para comenzar el desarrollo y asegurado el correcto funcionamiento de la aplicación se puede dar por definida la estructura del proyecto. La Fig. 21 muestra el esqueleto base.

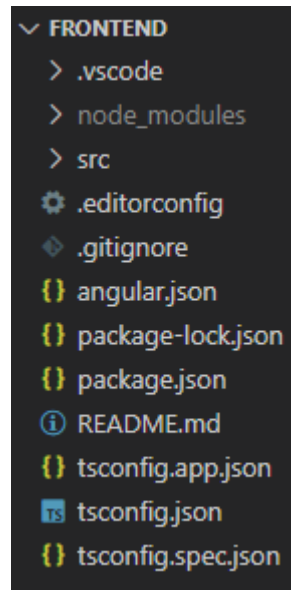


Fig. 21.- Esqueleto base de una aplicación Angular

El fichero `angular.json` es uno de los más importantes de la aplicación dado que define información relevante del proyecto, como el nombre, la dirección de destino una vez la aplicación está construida y empaquetada, el fichero `.html` principal (`index.html`), los ficheros de estilo `.css` y los ficheros JavaScript de librerías externas que se desean importar, etc.

En el fichero `package.json` se indican las dependencias del proyecto y se pueden definir acciones personalizadas que se pueden utilizar más adelante por consola, por lo que también se considera un archivo de gran importancia.

El resto de ficheros de la figura anterior, excepto el archivo `README.md`, son propios de la configuración de Angular y contribuyen en el proceso de montar y compilar la aplicación para que el navegador logre interpretarla, por lo que son indispensables para desarrollar la interfaz gráfica de la aplicación y no se pueden eliminar.

También aparece la carpeta `node_modules` en la figura. Esta carpeta tiene el mismo propósito que la carpeta con el mismo nombre de la parte del backend, almacenar los módulos y librerías externas que utiliza la aplicación.

El directorio src contiene diversos ficheros, pero los más importantes son el fichero main.ts y el fichero index.html.

El fichero main.ts se encarga de elegir el módulo que debe ser inicializado al poner en marcha la aplicación, también se ocupa de esto último. En este caso, el módulo de arranque es el AppModule.

Entre las etiquetas <body> del fichero index.html se indica el componente que debe ser inyectado en el cuerpo del fichero, que es el padre del resto de componentes, que son a su vez inyectados en su interior. En este caso, dicho componente, que es el punto de entrada a partir del cual comienza toda la aplicación, es el app.component.

El directorio src además de contener ficheros, también contiene dos carpetas, concretamente la carpeta app y la carpeta assets, las cuales se utilizan a lo largo del desarrollo del proyecto.

Por un lado, la carpeta app se considera de gran importancia, ya que es donde se estructura y desarrolla la aplicación y se crean los distintos componentes, módulos y servicios que la componen. Por otro lado, la carpeta assets almacena los recursos que usa la página web, como puede ser una imagen. La Fig. 22 muestra la estructura básica de la carpeta app.

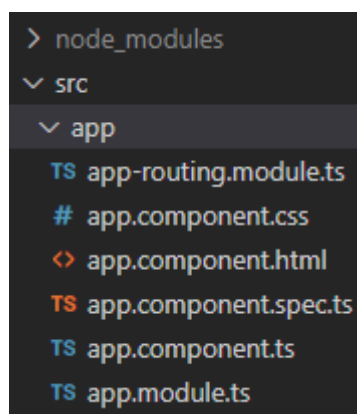


Fig. 22.- Estructura básica de la carpeta app

Dentro de la carpeta app se encuentra el fichero app-routing.module.ts, el fichero app.module.ts y los cuatro ficheros que forman el componente principal. El fichero app-routing.module.ts se trata del router que se encarga de mostrar la interfaz gráfica correspondiente a cada ruta de la aplicación, es decir, dependiendo de la ruta en la que se encuentra la aplicación muestra una vista u otra. El fichero app.module.ts recoge todos los componentes que se crean en la aplicación para que el navegador pueda encontrarlos y consiga interpretar el código correctamente, por esta razón, se considera uno de los ficheros más importantes de la aplicación. Los cuatro ficheros que conforman tanto el componente principal como cualquier otro son los siguientes:

- **Fichero component.css:** Este fichero permite aplicar estilos gráficos sobre el contenido del componente.
- **Fichero component.html:** Este fichero define la estructura del componente, es decir, diseña la vista que visualizan los clientes de la aplicación.
- **Fichero component.spec.ts:** Este fichero es un archivo TypeScript destinado a realizar pruebas con los componentes, sin embargo, en este caso no se le va a dar ningún uso, por lo que se puede prescindir de él.
- **Fichero component.ts:** Este fichero en formato TypeScript es el fichero principal del componente. En su interior se encuentra la lógica del componente, es decir, se definen las funciones que se ejecutan al realizar acciones sobre la vista, de las cuales varias se valen de servicios.

Por otro lado, la aplicación emplea los módulos de los que dispone Angular para organizar el código. Estos se utilizan para estructurar el código en fragmentos, evitando así generar los componentes sin orden alguno. Un módulo está compuesto al menos por un componente, y del mismo modo que estos, los módulos también deben declararse en el fichero app.module.ts. Cada módulo posee un fichero modulenombre.module.ts donde se declaran los módulos y librerías externas utilizadas por los componentes que forman parte del módulo, tal y como hace la aplicación con el fichero app.module.ts.

Como se explica en el apartado de Análisis y Diseño, el frontend del sistema está dividido en dos carpetas principales: modules y services. La carpeta modules contiene los módulos de cada vista y el módulo compartido, y la carpeta services los servicios de utilidades y los servicios web de la aplicación. La Fig. 23 muestra la estructura del frontend en Angular.

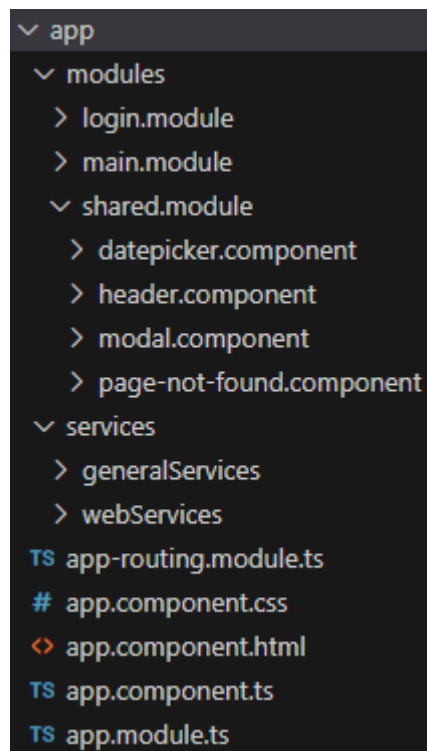


Fig. 23.- Estructura del frontend en Angular

La estructura de la figura anterior aporta modularidad a la aplicación, gracias a su división de componentes y a que evita en gran medida la repetición de código.

Funcionamiento y proceso de comunicación

A continuación, para lograr comprender más en profundidad la manera de trabajar de Angular, se destacan ciertos aspectos relevantes e interesantes del desarrollo del frontend de la aplicación, incluyendo el modo en el que se gestiona el inicio de sesión y se realizan las peticiones al backend.

Inicio de sesión en el frontend

Como ya se ha mencionado, Angular dispone de un router que accede al componente correspondiente a la ruta activa y muestra la interfaz gráfica definida en dicho componente. El fichero `app-routing.module.ts` se trata de este router, y para cumplir su cometido tiene que ser importado en el fichero `app.module.ts`, además requiere de un array de JSON denominado `Routes`. Cada JSON consta del `path`, o ruta, y de el componente al que hay que redirigir el sistema cuando aparece dicho `path`, en otras palabras, el array `Routes` recoge todas las rutas existentes y relaciona cada una de ellas con el componente que define la vista que debe mostrar el sistema cuando la ruta está activa. Al ejecutar la aplicación web Angular hace uso de este router para poder mostrar el formulario de inicio de sesión. El siguiente fragmento de código muestra la manera en que se declara la ruta de la pantalla de inicio.

```
const routes: Routes = [
  {
    path: 'login',
    component: LoginComponent
  }
];
```

Tal y como muestra el fragmento de código anterior, el `path` de la pantalla de inicio de sesión es `login`. Lo que quiere decir que el componente `LoginComponent` se ejecuta cuando el usuario accede a la web a través de la dirección `http://servidor/proyecto/login`.

Sin embargo, el usuario, por lo general, accede a la web a través de la dirección `http://servidor/proyecto`. Es irrelevante a través de qué dirección se accede a la web, el caso es que nada más acceder esta debe mostrar la pantalla de inicio. Para resolver esta cuestión el router es capaz de configurar una dirección de forma que se redirija a otra URL automáticamente.


```
const routes: Routes = [
  {
    path: '',
    redirectTo: 'login',
    pathMatch: 'full'
  },
  {
    path: 'login',
    component: LoginComponent
  }
];
```

Del mismo modo, se han controlado aquellas direcciones que no corresponden a ningún componente de la aplicación, para que, en el caso de que el usuario termine en alguna de ellas se le muestre una vista que le explique la situación.

```
{
  path: '**',
  component: PageNotFoundComponent
}
```

Una vez se muestra la pantalla de inicio de sesión, el usuario puede iniciar sesión en el sistema introduciendo sus credenciales en el formulario y, a continuación, pulsando el botón de Iniciar sesión. Entonces el fichero login.component.ts, que se encarga de la lógica del componente, recoge los valores introducidos en cada campo del formulario y los envía al servicio web login.service.ts. Este último envía los datos al backend a través de una petición para comprobar si el inicio de sesión es correcto o no.

Para enviar los datos a un servicio y esperar la respuesta a la petición enviada mediante tal servicio es necesario suscribirse al mismo, y no llamarlo del mismo modo que a una función cualquiera. Esto debe ser así debido a la asincronía, de igual forma que en el backend. Dicha suscripción establece que debe hacer el frontend tras recibir la respuesta a la petición que realiza el servicio al backend. Es decir, una vez el servicio recibe una respuesta, esta es devuelta al método suscrito al servicio, entonces este método analiza la respuesta y en función del código o status que tenga ejecuta una acción u otra. La Fig. 24 muestra el proceso de comunicación de un servicio en Angular.

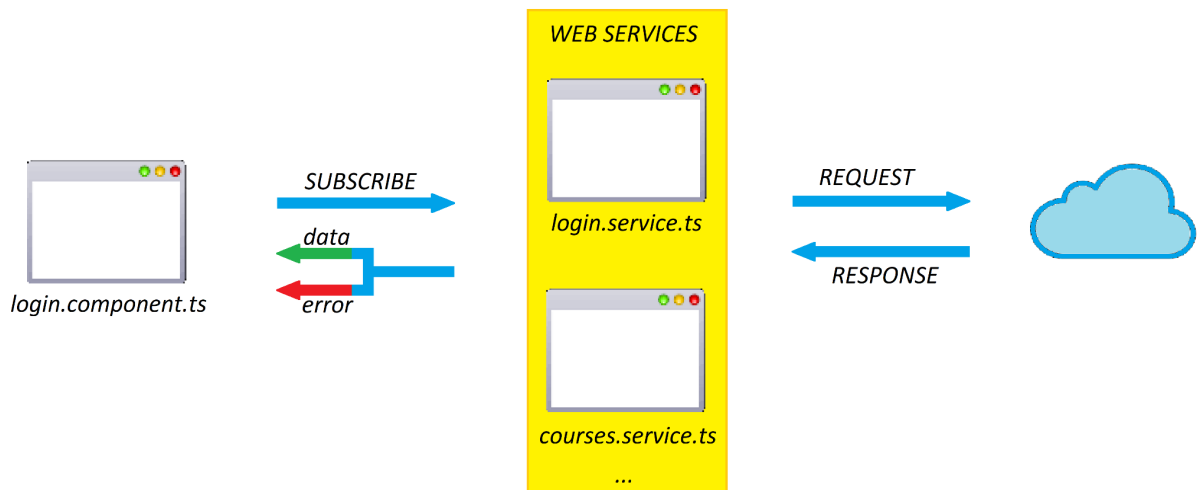


Fig. 24.- Servicios en Angular

Por defecto, al suscribirse a un servicio se deben crear un data y un error, al igual que en el método Promises, de modo que las respuestas que tienen un código correcto, como el 200, se analizan en el data, en cambio, las que tienen algún tipo de error se redirigen al error.

En este caso, cuando un usuario intenta iniciar sesión se ejecuta la suscripción sobre el servicio al que se le envían los datos introducidos en el formulario. Entonces el mismo servicio se encarga de definir la solicitud con el formato indicado para que sea aceptada por el backend, de señalar el tipo de petición del que se trata, por ejemplo, un GET o un POST, y de crear y enviar el objeto con los datos en el body o en la cabecera de la petición.

En caso de que la suscripción se gestione correctamente, se reciben los datos junto con la respuesta. Estos datos, que incluyen el token, el rol y el nombre completo del usuario identificado, son almacenados al instante en la sesión del usuario. Entonces, el sistema redirige la URL al path correspondiente a la pantalla principal de la aplicación, así se logra acceder a esta última. La sesión del usuario, también conocida como sessionStorage, es un contexto que está presente en toda la aplicación Angular, por lo que cualquier componente o servicio puede acceder a las variables que almacena. De este modo, se puede enviar el token en la cabecera de las peticiones de los servicios de los que dispone la aplicación, para que el Middleware del backend compruebe su validez. Por otro lado, en el momento en el que el usuario cierra sesión en el sistema, las variables almacenadas en el sessionStorage son eliminadas en favor de la seguridad.

Por el contrario, si no se consigue iniciar sesión correctamente se le muestra una ventana emergente al usuario que le describe el error. Este tipo de ventanas emergentes hacen uso de Bootstrap.

Utilización de ng-Bootstrap

ng-Bootstrap²⁹ es una librería externa que facilita en gran medida el uso de Angular ya que proporciona un mejor aspecto visual y permite incluir funcionalidades en la aplicación, entre otras cosas. En este caso, este módulo es utilizado para crear y mostrar distintos tipos de ventanas emergentes, básicamente.

Estas ventanas emergentes requieren de la suscripción a algún servicio o de la realización de algún proceso, y se crean y muestran una vez se reciben las respuestas a las solicitudes que realizan los servicios al backend. Estas ventanas disponen de botones, por lo que pueden ser cerradas por el propio usuario, y su función principal, en este caso, es mostrar mensajes de error.

Las ventanas emergentes también son conocidas como modals, o modales. Estos modales se llaman desde el fichero donde se encuentra la lógica del componente, y del mismo modo que con los servicios, también existe la posibilidad de suscribirse a ellos. Esto es así dado que el método que lanza el modal puede esperar a que el usuario tome una decisión y, en función de esta, realizar una acción u otra. Por ejemplo, si el usuario pulsa Aceptar se puede realizar una determinada acción, y si cierra la ventana puede realizarse otra distinta.

Los modales se crean del mismo modo que los componentes, por lo que están compuestos por los mismos ficheros. Por un lado, la vista del modal define su estructura y los botones de los que dispone, por otro lado, la lógica del modal define los métodos asociados a los botones, que posibilitan que el usuario devuelva una respuesta a la función que llama al modal.

Planificación para los docentes de Moodle

Para que Moodex muestre a los docentes de Moodle la información que les pueda ser útil para planificar las tareas de los cursos que imparten, se han desarrollado dos componentes: el primero muestra una lista con los cursos que imparten los docentes y el segundo muestra el calendario que indica los días apropiados para asignar una tarea a los alumnos de un determinado curso.

Cursos del docente

La lógica del componente main-teacher.component es la encargada de solicitar los cursos del profesor identificado al backend. Esta solicitud se hace mediante una suscripción a cierto servicio web del frontend. En caso de no recibir ningún dato, se cierra sesión y se informa del error. En caso contrario, se reciben los cursos y se agrupan por estado: en progreso, futuros y pasados.

²⁹ ng-Bootstrap: <https://ng-bootstrap.github.io>

Tras esto, la vista muestra por pantalla una lista con los cursos agrupados por los estados mencionados (Fig. 25). Esta lista se trata de una tablist, es decir, al pulsar sobre un curso aparece en la pantalla el calendario correspondiente.

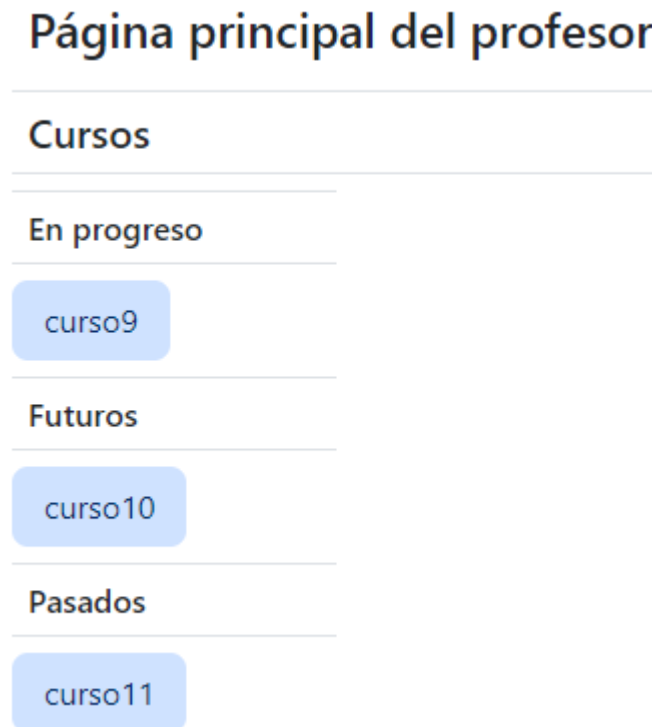


Fig. 25.- Cursos del docente

Calendario del curso

La lógica del componente `datepicker.component` es la encargada de solicitar los eventos de calendario de los alumnos del curso que escoge el profesor de la lista anterior. Esta solicitud se hace mediante una suscripción a cierto servicio web del frontend. En caso de no recibir ningún dato, se cierra sesión y se informa del error. En caso contrario, se reciben los eventos y el número total de alumnos, y a la hora de asignar un estilo (color) a cada fecha del calendario se tienen estos en cuenta. Si la fecha se encuentra entre los eventos, entonces se usa la cantidad de alumnos que tienen asignado al menos un evento en dicha fecha y el número total de alumnos del curso para obtener el porcentaje. Dependiendo del porcentaje se le asigna un estilo u otro a la fecha:

- Si el porcentaje se encuentra entre el 0 y el 25, se le asigna el color gris.
- Si el porcentaje se encuentra entre el 25 y el 50, se le asigna el color naranja.
- Si el porcentaje se encuentra entre el 50 y el 100, se le asigna el color rojo.

Lógicamente cuanto menor sea el porcentaje más recomendable es asignar una tarea.

Tras esto, la vista muestra por pantalla el calendario del curso con el código de colores descrito aplicado en cada fecha (Fig. 26). Este calendario se trata de un datepicker, es decir, al pulsar sobre una fecha está aparece seleccionada. Sin embargo, esta selección no sirve de nada por el momento.



Fig. 26.- Calendario del curso

Seguridad en la parte cliente

Del mismo modo que en la parte servidor de la aplicación, en la parte cliente también se aplican robustas medidas de seguridad para evitar todo lo posible ataques malintencionados.

Por un lado, como ya se ha mencionado, al iniciar sesión en el sistema, el token y el rol correspondientes al usuario se almacenan en el sessionStorage, y así, antes de enviar una petición al backend se puede acceder al sessionStorage, obtener el token e introducir el mismo en la cabecera de la petición.

Por otro lado, al igual que el backend no permite acceder a los recursos sin autorización, el frontend debe hacer lo mismo, por ello, al usuario no se le muestran aquellas acciones que no puede realizar. De esta manera, además de reforzar la seguridad, el sistema se vuelve más intuitivo. Para ello, se dispone del rol, que se encuentra en el sessionStorage e indica los permisos del usuario, en función de los cuales se le autorizan determinadas acciones.

VERIFICACIÓN Y EVALUACIÓN

El objetivo de este apartado consiste en definir y realizar una serie de pruebas sobre el backend y el frontend de la aplicación, y exponer e interpretar los resultados obtenidos.

Pruebas del backend

Dada la cantidad de servicios de los que dispone el backend, se opta por utilizar la herramienta Postman para automatizar y, de este modo, agilizar el proceso de pruebas.

Postman es una plataforma que permite crear y enviar peticiones a servicios para comprobar que funcionan correctamente. También permite crear variables que pueden ser usadas en pruebas posteriores, por ejemplo, el token puede ser almacenado en una variable en la prueba de inicio de sesión para posteriormente ser añadido automáticamente a las pruebas que lo necesiten.

Las pruebas se distribuyen por temática o funcionalidad, formando así distintos bloques de pruebas y proporcionando cierto orden. Además, de este modo, se pueden ejecutar las pruebas de un bloque en específico para comprobar su comportamiento.

A continuación, la Fig. 27 muestra la distribución de los bloques de pruebas en Postman:

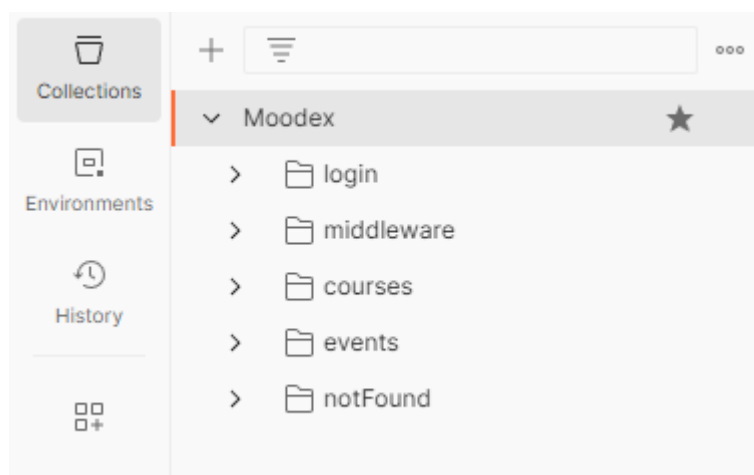


Fig. 27.- Bloques de pruebas en Postman

Para enviar una solicitud a través de Postman, antes se debe elegir el método HTTP e introducir la llamada al servicio. Además, dependiendo la solicitud, se tienen que añadir ciertas cabeceras al apartado Headers, o el cuerpo de la petición al apartado Body. El apartado Test permite realizar pruebas automáticas una vez se recibe la respuesta. A continuación, la Fig. 28 muestra un ejemplo de petición en Postman, que utiliza el método GET y envía el token en la cabecera Authorization para obtener los cursos de determinado

usuario, que es profesor. En la parte inferior de la figura se puede observar la respuesta a la solicitud:

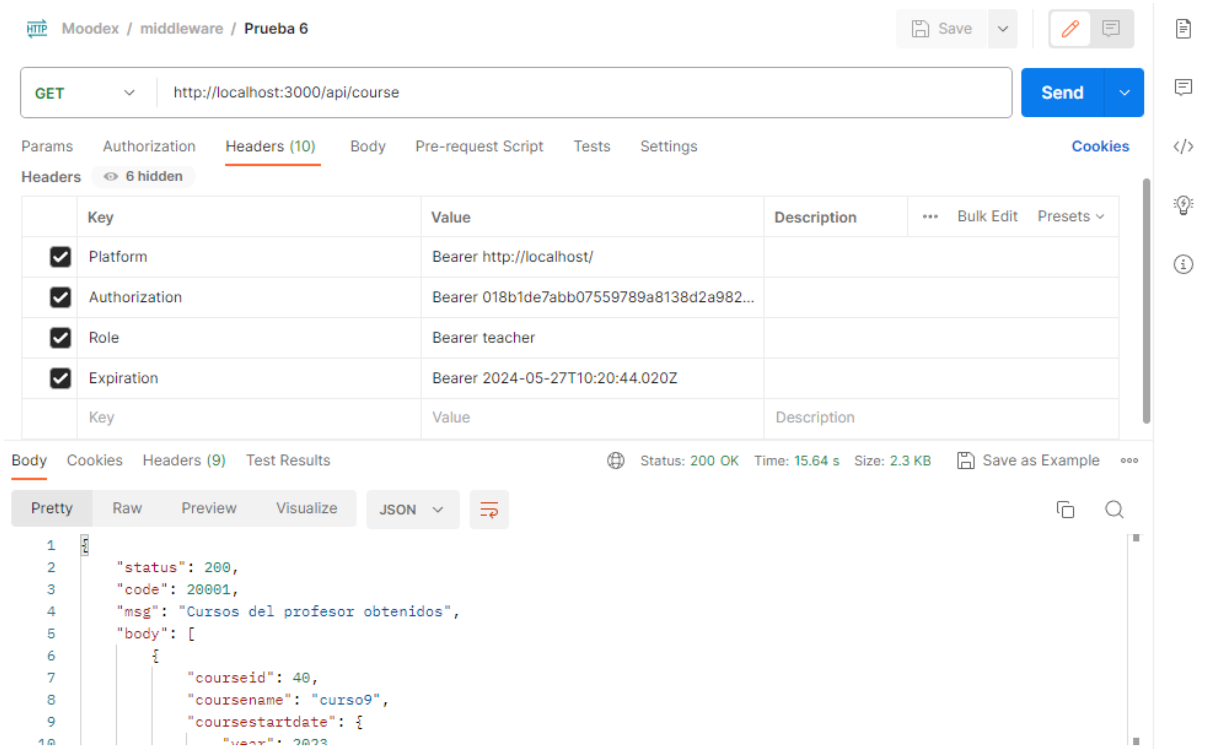


Fig. 28.- Ejemplo petición en Postman

Las siguientes tablas (Tabla 101, Tabla 102, Tabla 103, Tabla 104 y Tabla 105) detallan en qué consiste cada prueba, cuál es el resultado esperado de la misma, cuál es el resultado obtenido y si estos dos resultados coinciden. Que los dos resultados coincidan puede significar que todo funciona correctamente, en cambio, que no coincidan significa que hay algún tipo de error. En este último caso, se debe explicar cuál es el error y cómo se soluciona, además de repetir la prueba una vez se aplique la solución.

Tabla 101.- Pruebas del bloque Login (I)

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	Iniciar sesión sin proporcionar una plataforma, un nombre de usuario y una contraseña.	Mensaje de error notificando la falta de datos de la petición.	Mensaje de error notificando la falta de datos de la petición.	Sí	
2	Iniciar sesión con	Mensaje de	Mensaje de	Sí	

	un nombre de usuario y una contraseña que no corresponden a ningún usuario de la plataforma.	error notificando que los datos de la petición no son válidos.	error notificando que los datos de la petición no son válidos.		
3	Iniciar sesión con un nombre de usuario y una contraseña que corresponde a un usuario de la plataforma que no se encuentra inscrito en ningún curso.	Mensaje de error notificando que los datos de la petición no son válidos.	Mensaje de error notificando que los datos de la petición no son válidos.	NO	El rol de Usuario identificado de la plataforma Moodle no dispone de los permisos necesarios. La solución es simple, otorgarle los permisos.
3.1	Iniciar sesión con un nombre de usuario y una contraseña que corresponde a un usuario de la plataforma que no se encuentra inscrito en ningún curso.	Mensaje de error notificando que los datos de la petición no son válidos.	Mensaje de error notificando que los datos de la petición no son válidos.	Sí	
4	Iniciar sesión con un nombre de usuario y una contraseña que corresponde a un usuario de la plataforma que se encuentra inscrito al menos en un curso, pero ni lo administra ni lo imparte.	Mensaje de error notificando que los datos de la petición no son válidos.	Mensaje de error notificando que los datos de la petición no son válidos.	NO	Las condiciones del while que recorre los cursos del usuario son incorrectas. La solución es simple, corregirlas.
4.1	Iniciar sesión con un nombre de usuario y una contraseña que corresponde a un usuario de la plataforma que se encuentra inscrito al menos en un curso, pero ni lo administra ni lo imparte.	Mensaje de error notificando que los datos de la petición no son válidos.	Mensaje de error notificando que los datos de la petición no son válidos.	Sí	

5	Iniciar sesión con un nombre de usuario y una contraseña que corresponden a un administrador de la plataforma.	Mensaje notificando la validez de los datos de la petición, y el token, el rol de administrador y el nombre completo del usuario en el cuerpo de la respuesta.	Mensaje notificando la validez de los datos de la petición, y el token, el rol de administrador y el nombre completo del usuario en el cuerpo de la respuesta.	Sí	
6	Iniciar sesión con un nombre de usuario y una contraseña que corresponden a un docente de la plataforma.	Mensaje notificando la validez de los datos de la petición, y el token, el rol de profesor y el nombre completo del usuario en el cuerpo de la respuesta.	Mensaje notificando la validez de los datos de la petición, y el token, el rol de profesor y el nombre completo del usuario en el cuerpo de la respuesta.	Sí	

Tabla 102.- Pruebas del bloque Middleware

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	Llamar a algún servicio sin proporcionar las cabeceras de autorización necesarias (Platform, Authorization, Role y Expiration).	Mensaje de error notificando la falta de las cabeceras de autorización necesarias.	Mensaje de error notificando la falta de las cabeceras de autorización necesarias.	Sí	
2	Llamar a algún servicio sin proporcionar los datos de autorización necesarios (plataforma, token, rol y fecha	Mensaje de error notificando la falta de los datos de autorización necesarios.	Mensaje de error notificando la falta de los datos de autorización necesarios.	Sí	

	de caducidad del token).				
3	Llamar a algún servicio sin proporcionar una plataforma válida.	Mensaje de error notificando que la plataforma no es válida.	Mensaje de error notificando que la plataforma no es válida.	Sí	
4	Llamar a algún servicio sin proporcionar un token válido.	Mensaje de error notificando que el token no es válido.	Mensaje de error notificando que el token no es válido.	Sí	
5	Llamar a algún servicio sin proporcionar un rol válido.	Mensaje de error notificando que el rol no es válido.	Mensaje de error notificando que el rol no es válido.	Sí	
6	Llamar a algún servicio pasado el tiempo de validez del token.	Mensaje de error notificando que la sesión ha expirado y que el token ya no es válido.	Mensaje de error notificando que la sesión ha expirado y que el token ya no es válido.	Sí	
7	Llamar a algún servicio proporcionando todas las cabeceras y todos los datos de autorización necesarios, y todos estos siendo válidos.	Mensaje notificando la validez de los datos de la petición, y los datos correspondientes en el cuerpo de la respuesta.	Mensaje notificando la validez de los datos de la petición, y los datos correspondientes en el cuerpo de la respuesta.	Sí	

Tabla 103.- Pruebas del bloque Courses (I)

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	Solicitar los cursos del profesor sin disponer del rol	Mensaje de error notificando la falta de	Mensaje de error notificando la falta de	Sí	

	de profesor.	autorización.	autorización.		
2	Solicitar los cursos del profesor disponiendo del rol de profesor.	Mensaje notificando la autorización, y los cursos que imparte el profesor en el cuerpo de la respuesta.	Mensaje notificando la autorización, y los cursos que imparte el profesor en el cuerpo de la respuesta.	SÍ	

Tabla 104.- Pruebas del bloque Events

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	Solicitar los eventos de calendario de los alumnos del curso sin proporcionar el identificador, la fecha de inicio y la fecha de fin del curso.	Mensaje de error notificando la falta de datos.	Mensaje de error notificando la falta de datos.	SÍ	
2	Solicitar los eventos de calendario de los alumnos del curso sin disponer del rol de profesor.	Mensaje de error notificando la falta de autorización.	Mensaje de error notificando la falta de datos.	NO	Al comprobar si se obtiene el identificador, la fecha de inicio y la fecha de fin del curso el número 0 da problemas. La solución al problema consiste en comprobar que se obtienen sus cadenas.
2.1	Solicitar los eventos de calendario de los alumnos del curso sin disponer del rol de profesor.	Mensaje de error notificando la falta de autorización.	Mensaje de error notificando la falta de datos.	NO	Al comprobar si el identificador, la fecha de inicio y la fecha de fin del curso son números naturales el número 0 da problemas. Por

					lo que se considera un caso especial y se comprueba aparte.
2.2	Solicitar los eventos de calendario de los alumnos del curso sin disponer del rol de profesor.	Mensaje de error notificando la falta de autorización.	Mensaje de error notificando la falta de autorización.	Sí	
3	Solicitar los eventos de calendario de los alumnos de un curso sin alumnos.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y 0 alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y 0 alumnos en el cuerpo de la respuesta.	Sí	
4	Solicitar los eventos de calendario de los alumnos de un curso sin alumnos con eventos de calendario en alguno de sus cursos.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y el número total de alumnos en el cuerpo de la respuesta.	NO	El rol de Usuario identificado de la plataforma Moodle no dispone de los permisos necesarios. La solución es simple, otorgarle los permisos.
4.1	Solicitar los eventos de calendario de los alumnos de un curso sin alumnos con eventos de calendario en alguno de sus cursos.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y el número total de alumnos en el cuerpo de la respuesta.	Sí	
5	Solicitar los eventos de calendario de los alumnos de un curso sin alumnos con eventos de	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía	Sí	

	calendario de curso visibles en alguno de sus cursos.	y el número total de alumnos en el cuerpo de la respuesta.	y el número total de alumnos en el cuerpo de la respuesta.		
6	Solicitar los eventos de calendario de los alumnos de un curso con al menos un alumno con al menos un evento de calendario de curso visible en alguno de sus cursos. Estos eventos de calendario están establecidos únicamente en un día.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Sí	
7	Solicitar los eventos de calendario de los alumnos de un curso con al menos un alumno con al menos un evento de calendario de curso visible en alguno de sus cursos. Estos eventos de calendario están establecidos en varios días.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Sí	
8	Solicitar los eventos de calendario de los alumnos de un curso con al menos un alumno con varios eventos de calendario en la misma fecha.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Sí	
9	Solicitar los	Mensaje	Mensaje	Sí	

	eventos de calendario de los alumnos de un curso con al menos dos alumnos con un evento de calendario en la misma fecha.	notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.		
10	Solicitar los eventos de calendario de los alumnos de un curso con al menos un alumno con al menos un evento de calendario en un curso que no imparta el profesor del curso mencionado al principio.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista de eventos vacía y el número total de alumnos en el cuerpo de la respuesta.	NO	El rol de Usuario identificado de la plataforma Moodle no dispone de los permisos necesarios. La solución es simple, otorgarle los permisos.
10.1	Solicitar los eventos de calendario de los alumnos de un curso con al menos un alumno con al menos un evento de calendario en un curso que no imparta el profesor del curso mencionado al principio.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	Mensaje notificando la obtención de los eventos, y una lista con los eventos y el número total de alumnos en el cuerpo de la respuesta.	SÍ	

Tabla 105.- Pruebas del bloque NotFound (I)

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	Solicitar la información de una ruta desconocida.	Mensaje de error notificando la inexistencia de	Mensaje de error notificando la inexistencia	SÍ	

		la ruta.	de la ruta.		
2	Solicitar la información de una ruta conocida.	Mensaje que notifica la existencia de la ruta.	Mensaje que notifica la existencia de la ruta.	Sí	

Pruebas del frontend

Al igual que las pruebas del backend, las pruebas del frontend se distribuyen por temática o funcionalidad, formando así distintos bloques de pruebas y proporcionando cierto orden. Además, las siguientes tablas (Tabla 106, Tabla 107, Tabla 108, Tabla 109 y Tabla 110) también definen las pruebas que se realizan y exponen e interpretan los resultados que se obtienen.

Tabla 106.- Pruebas del bloque Login (II)

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	El usuario intenta iniciar sesión sin proporcionar una plataforma, un nombre de usuario y una contraseña.	La pantalla muestra una alerta que notifica que todos los campos del formulario son obligatorios.	La pantalla muestra una alerta que notifica que todos los campos del formulario son obligatorios.	Sí	
2	El usuario intenta iniciar sesión con un nombre de usuario y una contraseña que no corresponden a ningún usuario de la plataforma.	La pantalla muestra una alerta que notifica que el nombre de usuario y la contraseña no corresponden a ningún docente ni a ningún administrador de la plataforma.	La pantalla muestra una alerta que notifica que el nombre de usuario y la contraseña no corresponden a ningún docente ni a ningún administrador de la plataforma.	Sí	
3	El usuario intenta iniciar sesión con un nombre de	La pantalla muestra una alerta que	La pantalla muestra una alerta que	Sí	

	usuario y una contraseña que corresponde a un usuario de la plataforma que no se encuentra inscrito en ningún curso.	notifica que el nombre de usuario y la contraseña no corresponden a ningún docente ni a ningún administrador de la plataforma.	notifica que el nombre de usuario y la contraseña no corresponden a ningún docente ni a ningún administrador de la plataforma.		
4	El usuario intenta iniciar sesión con un nombre de usuario y una contraseña que corresponde a un usuario de la plataforma que se encuentra inscrito al menos en un curso, pero ni lo administra ni lo imparte.	La pantalla muestra una alerta que notifica que el nombre de usuario y la contraseña no corresponden a ningún docente ni a ningún administrador de la plataforma.	La pantalla muestra una alerta que notifica que el nombre de usuario y la contraseña no corresponden a ningún docente ni a ningún administrador de la plataforma.	Sí	
5	El usuario intenta iniciar sesión con un nombre de usuario y una contraseña que corresponden a un administrador de la plataforma.	Se almacena, la plataforma, el token, el rol y el nombre completo del usuario en el sessionStorage y se muestra la pantalla principal del administrador.	Se almacena, la plataforma, el token, el rol y el nombre completo del usuario en el sessionStorage y se muestra la pantalla principal del administrador.	Sí	
6	El usuario intenta iniciar sesión con un nombre de usuario y una contraseña que corresponden a un docente de la plataforma.	Se almacena, la plataforma, el token, el rol y el nombre completo del usuario en el sessionStorage y se muestra la pantalla principal del profesor.	Se almacena, la plataforma, el token, el rol y el nombre completo del usuario en el sessionStorage y se muestra la pantalla principal del profesor.	Sí	

Tabla 107.- Pruebas del bloque Session

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	El usuario intenta realizar alguna acción pasado el tiempo de validez del token.	Se cierra la sesión y la pantalla muestra una alerta que notifica que la sesión ha expirado.	Se cierra la sesión y la pantalla muestra una alerta que notifica que la sesión ha expirado.	Sí	
2	El usuario intenta cerrar sesión.	Se elimina la plataforma, el token, el rol, el nombre completo y la fecha de caducidad del token del usuario del sessionStorage y se muestra la pantalla de inicio de la aplicación.	Se elimina la plataforma, el token, el rol, el nombre completo y la fecha de caducidad del token del usuario del sessionStorage y se muestra la pantalla de inicio de la aplicación.		

Tabla 108.- Pruebas del bloque Courses (II)

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	El usuario, que imparte al menos un curso actualmente, logra iniciar sesión.	Se muestra la pantalla principal del profesor con los cursos que imparte. Al menos uno de ellos se encuentra entre los cursos en progreso.	Se muestra la pantalla principal del profesor con los cursos que imparte. Al menos uno de ellos se encuentra entre los cursos en progreso.	Sí	
2	El usuario, que impartirá al	Se muestra la pantalla	Se muestra la pantalla	Sí	

	menos un curso en el futuro, logra iniciar sesión.	principal del profesor con los cursos que imparte. Al menos uno de ellos se encuentra entre los cursos futuros.	principal del profesor con los cursos que imparte. Al menos uno de ellos se encuentra entre los cursos futuros.		
3	El usuario, que impartió al menos un curso en el pasado, logra iniciar sesión.	Se muestra la pantalla principal del profesor con los cursos que imparte. Al menos uno de ellos se encuentra entre los cursos pasados.	Se muestra la pantalla principal del profesor con los cursos que imparte. Al menos uno de ellos se encuentra entre los cursos pasados.	Sí	

Tabla 109.- Pruebas del bloque Calendar

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
1	El profesor pulsa sobre un curso sin alumnos.	La pantalla muestra un calendario sin eventos.	La pantalla muestra un calendario sin eventos.	Sí	
2	El profesor pulsa sobre un curso sin alumnos con eventos de calendario en alguno de sus cursos.	La pantalla muestra un calendario sin eventos.	La pantalla muestra un calendario sin eventos.	Sí	
3	El profesor pulsa sobre un curso sin alumnos con eventos de calendario de curso visibles en alguno de sus cursos.	La pantalla muestra un calendario sin eventos.	La pantalla muestra un calendario sin eventos.	Sí	

4	El profesor pulsa sobre un curso con al menos un alumno con al menos un evento de calendario de curso visible en alguno de sus cursos. Estos eventos de calendario están establecidos únicamente en un día.	La pantalla muestra un calendario con eventos.	La pantalla muestra un calendario con eventos.	Sí	
5	El profesor pulsa sobre un curso con al menos un alumno con al menos un evento de calendario de curso visible en alguno de sus cursos. Estos eventos de calendario están establecidos en varios días.	La pantalla muestra un calendario con eventos.	La pantalla muestra un calendario con eventos.	Sí	
6	El profesor pulsa sobre un curso con al menos un alumno con varios eventos de calendario en la misma fecha.	La pantalla muestra un calendario con eventos.	La pantalla muestra un calendario con eventos.	Sí	
7	El profesor pulsa sobre un curso con al menos dos alumnos con un evento de calendario en la misma fecha.	La pantalla muestra un calendario con eventos.	La pantalla muestra un calendario con eventos.	Sí	
8	El profesor pulsa sobre un curso con al menos un alumno con al menos un evento de calendario en un curso que no imparta el profesor del curso	La pantalla muestra un calendario con eventos.	La pantalla muestra un calendario con eventos.	Sí	

	mencionado al principio.				
9	El profesor pulsa sobre un curso que entre el 0 y el 25% de su alumnado tiene un evento de calendario en la misma fecha.	La pantalla muestra un calendario con al menos un día de color gris, es decir, ese día entre el 0 y el 25% del alumnado tiene asignada una tarea.	La pantalla muestra un calendario con al menos un día de color gris, es decir, ese día entre el 0 y el 25% del alumnado tiene asignada una tarea.	Sí	
10	El profesor pulsa sobre un curso que entre el 25 y el 50% de su alumnado tiene un evento de calendario en la misma fecha.	La pantalla muestra un calendario con al menos un día de color naranja, es decir, ese día entre el 25 y el 50% del alumnado tiene asignada una tarea.	La pantalla muestra un calendario con al menos un día de color naranja, es decir, ese día entre el 25 y el 50% del alumnado tiene asignada una tarea.	Sí	
11	El profesor pulsa sobre un curso que entre el 50 y el 100% de su alumnado tiene un evento de calendario en la misma fecha.	La pantalla muestra un calendario con al menos un día de color rojo, es decir, ese día entre el 50 y el 100% del alumnado tiene asignada una tarea.	La pantalla muestra un calendario con al menos un día de color rojo, es decir, ese día entre el 50 y el 100% del alumnado tiene asignada una tarea.	Sí	

Tabla 110.- Pruebas del bloque NotFound (II)

Código	Descripción	Resultado esperado	Resultado obtenido	¿OK?	Comentarios
--------	-------------	--------------------	--------------------	------	-------------

1	El usuario intenta solicitar la información de una ruta desconocida.	La pantalla muestra un mensaje de error que notifica la inexistencia de la ruta.	La pantalla muestra un mensaje de error que notifica la inexistencia de la ruta.	Sí	
2	El usuario intenta solicitar la información de una ruta conocida.	La pantalla muestra la información correspondiente.	La pantalla muestra la información correspondiente.	Sí	

CONCLUSIONES Y TRABAJO FUTURO

En este último apartado de la memoria se analiza todo lo realizado a lo largo del desarrollo del proyecto y se recogen las conclusiones. También se hace un análisis que indica la opinión del desarrollador sobre cómo debería evolucionar la aplicación.

Objetivos cumplidos

Al comenzar a documentar el trabajo, una de las primeras cosas que se hicieron fue definir los objetivos del proyecto. Estos se dividieron entre objetivos funcionales principales y secundarios y objetivos personales. En un principio el desarrollador tenía en mente cumplir todos ellos y ahora llega el momento de ver si ha sido así. Para ello, a continuación, se presentan nuevamente y se explica por qué se dan o no por cumplidos:

Se va a empezar por describir los objetivos principales del proyecto que son los objetivos mínimos que se pedían para dar el trabajo por realizado.

El primero de ellos decía lo siguiente: *“Desarrollar una herramienta que proporcione información al profesorado y que pueda servir de ayuda para planificar la docencia y la entrega de tareas. Esto le puede llegar a beneficiar al alumnado porque es más difícil que le coincidan entregas”*.

Este primer objetivo se puede dar por cumplido porque se ha logrado desarrollar una aplicación web que les proporciona la información que les puede servir de ayuda para planificar la docencia y la entrega de tareas al profesorado de las plataformas de docencia desarrolladas a través de Moodle.

El profesorado se puede conectar a la aplicación web introduciendo sus credenciales, entonces la aplicación se encarga de obtener y de mostrar los cursos que ha impartido, imparte o va a impartir en un futuro. Si el profesor pulsa sobre uno de los cursos puede visualizar un calendario con un código de colores que le marca los días apropiados para asignar una tarea a los alumnos. Los colores corresponden al porcentaje del alumnado del curso que tiene al menos una tarea asignada determinado día. Por lo que cuanto menor sea el porcentaje más recomendable es asignar una tarea.

El segundo trataba de *“Realizar un trabajo que, al menos, cumpla los requisitos y objetivos mínimos necesarios para aprobar el TFG y que cuente con el visto bueno del tutor y del desarrollador”*.

Este segundo también se da por cumplido dado que el tutor ha sido el que, desde un principio, ha establecido los requisitos y los objetivos mínimos del proyecto y en una de las últimas reuniones ha expresado que la aplicación web los ha alcanzado. Por lo que el trabajo realizado cuenta con el visto bueno del tutor. El desarrollador, por su parte, está

satisfecho porque ha conseguido desarrollar las funciones básicas de la aplicación, no obstante, le habría gustado terminar el proyecto por completo.

El tercer objetivo consistía en *“Aplicar los conocimientos obtenidos a lo largo de la carrera con el fin de plasmar sobre el trabajo todo lo aprendido”*.

Este objetivo se da por cumplido dado que no es posible desarrollar este proyecto sin aplicar ciertos conocimientos adquiridos a lo largo de la carrera. Por un lado, para llevar a cabo la aplicación web son necesarios conocimientos avanzados de programación y conocimientos básicos de programación de sistemas web. Por otro lado, en el transcurso de la carrera se ha ido adquiriendo experiencia sobre cómo gestionar proyectos y esta se ve reflejada tanto en la aplicación como en la memoria.

El cuarto y último objetivo principal decía lo siguiente: *“Aprender a utilizar herramientas novedosas para el desarrollador”*.

Este también se ha cumplido dado que el desarrollador ha utilizado determinadas herramientas a lo largo del desarrollo del proyecto que nunca antes había usado. Estas herramientas son las siguientes:

- **Express:** Express forma parte de las herramientas de desarrollo del proyecto y se trata del framework de Node.js con el que se ha trabajado para desarrollar el backend de la aplicación web.
- **Angular:** Angular forma parte de las herramientas de desarrollo del proyecto y se trata del framework con el que se ha desarrollado el frontend de la aplicación web.
- **TypeScript:** TypeScript forma parte de las herramientas de desarrollo del proyecto y se trata del lenguaje de programación que utiliza la lógica de los componentes que se generan en Angular.
- **Postman:** Postman forma parte de las herramientas de pruebas y se trata de la aplicación a través la cual se han llevado a cabo las distintas pruebas a los servicios de los que dispone el backend de la aplicación web.
- **Moodle:** Moodle forma parte de las herramientas de pruebas y se trata de la plataforma de docencia de la que obtiene la información el backend de la aplicación web, por lo que se ha administrado de tal manera que se prueben todos los casos y se tengan en cuenta todas las situaciones posibles.

En último lugar, se encuentran los objetivos secundarios: *“Como objetivo secundario se pretenden añadir funcionalidades extra, como pueden ser ampliar las estadísticas disponibles o permitir estadísticas más complejas y facilitar la comunicación entre eGela y la aplicación desarrollada para reducir lo máximo posible la carga de trabajo del profesorado evitándole tener que andar alternando las plataformas”*.

En otras palabras, se pretendía proporcionar una funcionalidad a los administradores de los cursos de las plataformas desarrolladas con Moodle, además de permitir planificar una tarea a los profesores directamente desde la interfaz que muestra el calendario de un determinado curso. La funcionalidad planteada consiste en que la administración pueda hacer informes del uso que le da cada tipo de usuario (estudiante y profesor) a los cursos.

Estos objetivos son secundarios y opcionales, y aunque el desarrollador en un principio pretendiese llevar dichas funciones a cabo, su decisión final ha sido no implementarlas. Ha preferido dar el trabajo por finalizado y tomarse un descanso.

Planificación final

La planificación temporal que se iba a seguir en un principio a lo largo del desarrollo del proyecto finalmente no se ha ajustado a la realidad.

Para empezar, ha habido días que no se ha cumplido con la carga de trabajo diaria de 4 horas dado que el desarrollador a partir de cierto momento se saturaba y no conseguía ser productivo.

Además, en un principio se tenía planteado trabajar los 7 días de la semana, algo imposible de llevar a la práctica. Ha habido días que el desarrollador necesitaba un descanso y también ha habido periodos de vacaciones de por medio.

No obstante, no todo el desfase de la planificación inicial ha sido debido a una disminución en la carga de trabajo semanal. También ha venido dado porque la estimación del tiempo que iba a llevar el desarrollo del proyecto se quedó corta. Cada uno de los paquetes de trabajo definidos en el apartado de Alcance han subestimado el tiempo necesario para llevar a cabo la tarea.

Por estas razones, el proyecto se ha acabado a finales de junio de 2023. Más de medio año más tarde de lo esperado.

La Tabla 111 recoge un resumen de la duración estimada y la duración real de los paquetes de trabajo con más peso del proyecto.

Tabla 111.- Resumen de la duración estimada y real de los paquetes de trabajo con más peso del proyecto

Paquete de trabajo	Duración estimada	Duración real	Diferencia
1 - Introducción	10 horas	15 horas	5 horas
2 - Planteamiento inicial	100 horas	150 horas	50 horas
3 - Antecedentes	10 horas	15 horas	5 horas
4 - Prototipo I - Iniciar sesión	72 horas	100 horas	28 horas
5 - Prototipo II - Listar cursos	80 horas	112 horas	32 horas

6 - Prototipo III - Mostrar calendario	96 horas	134 horas	38 horas
9 - Conclusiones y trabajo futuro	20 horas	20 horas	0 horas
10 - Resumen	10 horas	10 horas	0 horas
Moodex	398 horas	556 horas	158 horas

Como se puede observar en la tabla anterior y como ya se ha mencionado, el desarrollo de la aplicación ha terminado con los 3 primeros prototipos que son las funciones básicas que había que realizar sí o sí. En un principio se estimó que el desarrollo de estos prototipos, y de la documentación, sería de 398 horas, pero, finalmente, ha sido de 556 horas. 158 horas de diferencia, un 40% más de tiempo.

Sin duda alguna, los paquetes cuya estimación más diferencia tienen con la realidad son los prototipos. Esto puede ser debido a que obtener el conocimiento necesario para trabajar con nuevas herramientas le ha supuesto un gran esfuerzo y tiempo al desarrollador. Asimismo, surgen un sinnúmero de dudas y problemas que hay que resolver o de cambios en la implementación que exigen mayor dedicación, o lo que viene a ser lo mismo, más tiempo.

Riesgos y sus apariciones

A lo largo del desarrollo ha surgido alguno de los imprevistos que se recogieron en el apartado de Gestión de riesgos. Uno de los más destacados es haber realizado una planificación temporal incorrecta, lo que ha acarreado una gran desajuste en el tiempo dedicado a cada tarea y al proyecto en general, como se ha podido ver en la sección anterior. Este riesgo tenía una probabilidad muy alta de suceder y el impacto consecuente ha sido considerable. Sin embargo, se ha ido aplicando el plan de contingencia de vez en cuando para disminuir todo lo posible el retraso en la ejecución del proyecto. Este plan ha consistido en hacer un reajuste a la planificación.

Evaluación económica final

Debido al incremento de horas de desarrollo respecto a la planificación inicial, los costes finales de desarrollo también se han visto incrementados. Todos los tipos de costes (coste de desarrollo, coste del hardware empleado y los costes indirectos) son dependientes del número de horas dedicado al desarrollo del proyecto, por lo que se puede estimar que el coste final es el coste total estimado en un principio multiplicado por la división entre el número de horas dedicadas realmente y el número de horas estimadas. En un principio se estimaron 398 horas y un coste total de 5420 € por realizar las funciones básicas, sin

embargo, finalmente se han dedicado 556 horas para ello. Por lo que, el nuevo coste asciende a 7570 €.

Cabe recordar que el propósito de este proyecto no es la obtención de beneficios económicos. Su finalidad, como ya se ha comentado anteriormente, es desarrollar una herramienta que proporcione información al profesorado y que pueda servir de ayuda para planificar la docencia y la entrega de tareas.

Líneas futuras

Ahora se va a hablar del futuro del proyecto. En caso de decidir seguir con el proyecto, en opinión del desarrollador, el próximo paso sería completarlo, es decir, cumplir los objetivos secundarios ya mencionados.

Para ello, debería implementar nuevos servicios en el backend que obtengan la información que requiere el administrador para saber el uso que le dan tanto los estudiantes como los profesores a los cursos.

A continuación, debería desarrollar una interfaz gráfica que le diese la oportunidad de escoger de qué curso o cursos quiere obtener esta información, entre qué fechas, el tipo de usuario al que analizar, etc. Esta misma interfaz, una vez escogida la información deseada, debería de mostrársela al administrador por pantalla.

Además de los objetivos secundarios, el desarrollador también tiene en mente varias mejoras y ampliaciones que le hubiera gustado implementar.

La primera ampliación que el desarrollador llevaría a cabo sería darle la opción al profesor de filtrar sus cursos por el estado de los mismos (en progreso, futuros y pasados). Por ejemplo, si el profesor solo quisiera ver sus cursos en progreso tendría un filtro a mano que le mostraría únicamente esos cursos, y lo mismo con el resto de estados.

De igual manera, si el profesor impartiera una gran cantidad de cursos y quisiese consultar uno en específico y no quisiese estar buscándolo entre todos uno a uno, tendría a su alcance un buscador en el que escribiría el nombre del curso, y la interfaz únicamente le mostraría ese curso.

El resto de mejoras son meros detalles que no son del gusto del desarrollador. Por un lado, cuando un profesor selecciona uno de sus cursos la pantalla le muestra el calendario del curso. Esto es correcto, pero no hay manera de quitar la selección y que el calendario desaparezca. Por otro lado, un profesor puede seleccionar una fecha del calendario de determinado curso, pero no puede quitar su selección. La mejora reside en poder quitar dichas selecciones sin tener que recargar la página.

Reflexión personal

Gracias al desarrollo de este proyecto he obtenido nuevos conocimientos y más experiencia en diferentes aspectos.

Para empezar, uno de los errores más notables a lo largo del proyecto ha sido hacer una mala planificación temporal por una subestimación del tiempo necesario para realizar cada tarea. No obstante, cometer este error me ha hecho darme cuenta que es un aspecto que debo mejorar. En futuras ocasiones le daré más importancia y ya tendré una referencia en la que apoyarme.

A lo largo del proyecto he utilizado tecnologías y herramientas que nunca antes había usado y he adquirido el manejo y la destreza necesaria para desenvolverme sin ningún tipo de problema. Ahora tengo el conocimiento suficiente para manejarme con soltura tanto en el frontend como en el backend de una aplicación y he mejorado mis dotes a la hora de realizar aplicaciones web. En el hipotético caso de que me mandasen desarrollar una aplicación web desde cero, no tendría duda alguna de que herramientas escogería y de qué debería hacer para comenzar.

Antes de empezar a implementar la aplicación, he tenido que diseñarla en base a los requisitos del cliente. Y una vez implementada, he tenido que definir y realizar una serie de pruebas. Por lo que he adquirido más experiencia de la que ya tenía en estas fases de desarrollo.

Tener que gestionar un proyecto de estas características de forma individual lleva más tiempo que haciéndolo en grupo, pero te ves recompensado con una mayor fortaleza a nivel técnico y a nivel personal. Sin embargo, debo decir que he echado en falta el reto que supone la toma de decisiones y los consensos a los que se tienen que llegar en grupo. Porque la realidad es que la mayoría de trabajos son en grupo.

El desarrollo del proyecto también me ha dado la oportunidad de ver con mayor claridad de entre todas las fases de desarrollo de una aplicación a la que me quiero dedicar. Por un lado, tengo bastante claro que documentar no es lo mío, se me hace muy tedioso y aburrido. Por otro lado, el trabajo de diseño y, sobre todo, el de pruebas no es mi devoción, lo que realmente me gusta es el trabajo de programación, más concretamente, el diseño de interfaces gráficas. Por lo que si en un trabajo me diesen a elegir la parte de la aplicación en la que trabajar, me decantaría por la parte cliente.

Quiero decir que estoy satisfecho del trabajo realizado, no obstante, me hubiese gustado acabar el proyecto por completo, entonces me podría sentir del todo orgulloso.

También siento gratitud hacía la universidad y, sobre todo, a mi tutor por darme la oportunidad de desarrollar una herramienta que les pueda ser de utilidad tanto al profesorado, de manera directa, como al alumnado, de manera indirecta. Nunca está de más ayudar.

Siento que he aprovechado el tiempo porque para desarrollar la aplicación he tenido que aplicar ciertos conocimientos que he adquirido a lo largo de estos 4 años de carrera y, una vez me adentre en el mundo laboral, también los voy a tener que usar.

En resumen, con este proyecto he adquirido nuevos conocimientos y más experiencia de la que ya tenía a la hora de desarrollar aplicaciones web y de gestionar todo lo que las rodea. Además, me he dado cuenta de mis preferencias a la hora de trabajar y de todo lo que ha implicado el proyecto a nivel personal.

BIBLIOGRAFÍA

[1] alredsa

Imagen del ciclo de vida incremental

<http://alredsa.blogspot.com/2017/07/ciclo-de-vida-de-proyectos-clasico.html> (accedido el 21/07/2023)

[2] felipemartinez

¿Qué es y para qué sirve Gloomaps?

<https://felipemartinez.es/herramienta-para-hacer-organigramas-gloomaps> (accedido el 31/07/2022)

[3] Capterra

¿Qué es y para qué sirve Visual Paradigm?

<https://www.capterra.pe/software/145716/visual-paradigm> (accedido el 13/02/2022)

[4] Visual Studio

¿Qué es y para qué sirve Visual Studio Code?

<https://visualstudio.microsoft.com/es/> (accedido el 18/10/2022)

[5] Drauta

¿Qué es y para qué sirve Node.js?

<https://www.drauta.com/que-es-nodejs-y-para-que-sirve> (accedido el 13/02/2022)

[6] MDN Web Docs

¿Qué es y para qué sirve Express?

https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction (accedido el 22/08/2022)

[7] Quality devs

¿Qué es y para qué sirve Angular?

<https://www.qualitydevs.com/2019/09/16/que-es-angular-y-para-que-sirve/> (accedido el 13/02/2022)

[8] Microsoft

¿Qué es y para qué sirve TypeScript?

<https://docs.microsoft.com/es-es/archive/msdn-magazine/2013/june/modern-apps-use-typescript-in-modern-apps> (accedido el 13/02/2022)

[9] encora

¿Qué es y para qué sirve Postman?

<https://www.encora.com/es/blog/como-realizar-pruebas-automatizadas-con-postman>

(accedido el 18/10/2022)

[10] Moodle

¿Qué es y para qué sirve Moodle?

https://docs.moodle.org/all/es/Acerca_de_Moodle (accedido el 13/02/2022)

[11] Axios

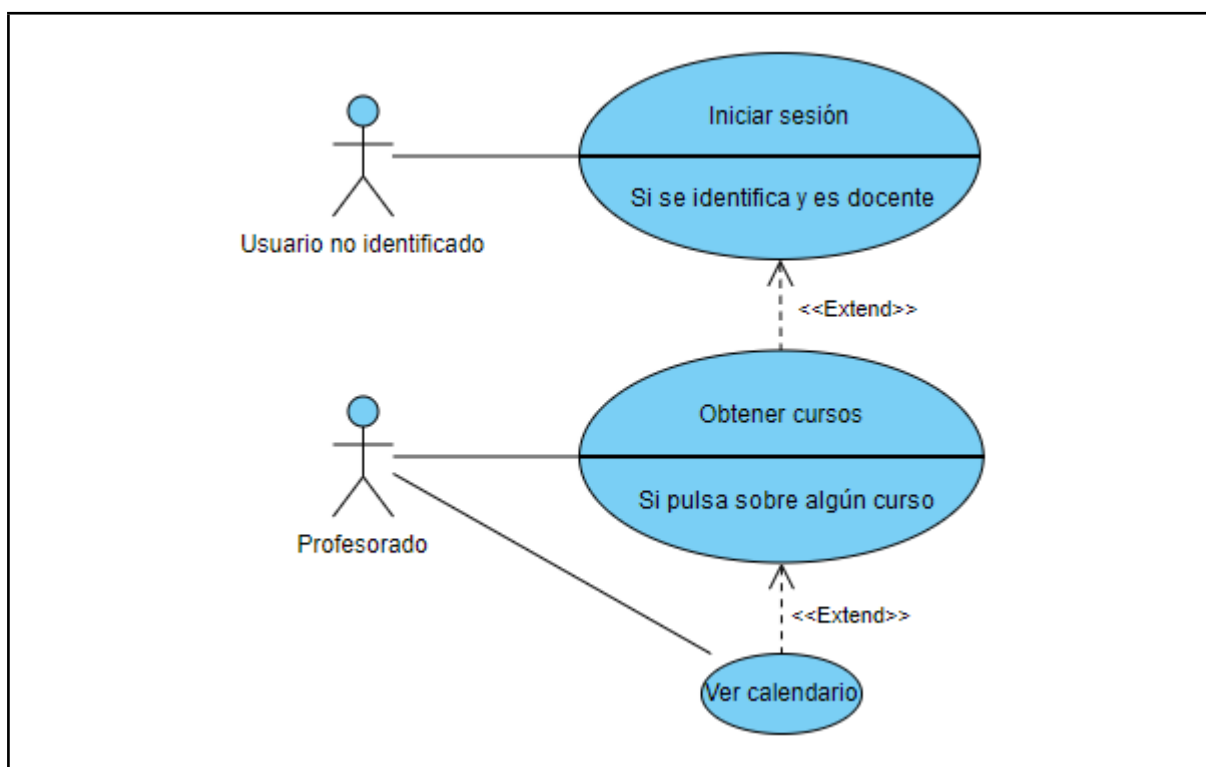
¿Qué es y para qué sirve Axios?

<https://www.arsys.es/blog/axios> (accedido el 01/07/2023)

ANEXO I.- CASOS DE USO EXTENDIDOS

En este anexo se presentan los casos de uso extendidos correspondientes a los casos de uso del modelo de casos de uso, es decir, se hace un análisis detallado de cada caso de uso mostrando de forma explícita toda la información relativa a su funcionamiento y a los roles que participan, además de los distintos tipos de decisiones lógicas que suceden y la forma de gestionarlas.

Iniciar sesión



Nombre	Iniciar sesión
Descripción	Permite identificarse introduciendo sus credenciales al profesorado y a la administración de las plataformas de docencia desarrolladas con Moodle.
Actores	Usuario no identificado.
Precondiciones	El usuario no debe haber iniciado sesión aún.
Requisitos no funcionales	Ninguno.
Flujo de eventos	<ol style="list-style-type: none"> 1. El usuario no identificado introduce la plataforma y sus credenciales y pulsa el botón Iniciar sesión (Fig. 29). <ol style="list-style-type: none"> a. Si el nombre de usuario y la contraseña no

	<p>corresponden a ningún docente o administrador de la plataforma, entonces se muestra un mensaje de error que informa de ello (Fig. 30).</p> <p>b. Si los datos introducidos son correctos, entonces se redirige a la pantalla principal de la aplicación correspondiente (Fig. 31 y Fig. 32).</p>
Postcondiciones	El usuario no identificado habrá iniciado sesión.
Interfaz gráfica	 <p>Fig. 29.- Pantalla de inicio</p>  <p>Fig. 30.- Mensaje de error</p>

Página principal del administrador

Fig. 31.- Pantalla principal del administrador

Página principal del profesor

Cursos

En progreso

curso9

Futuros

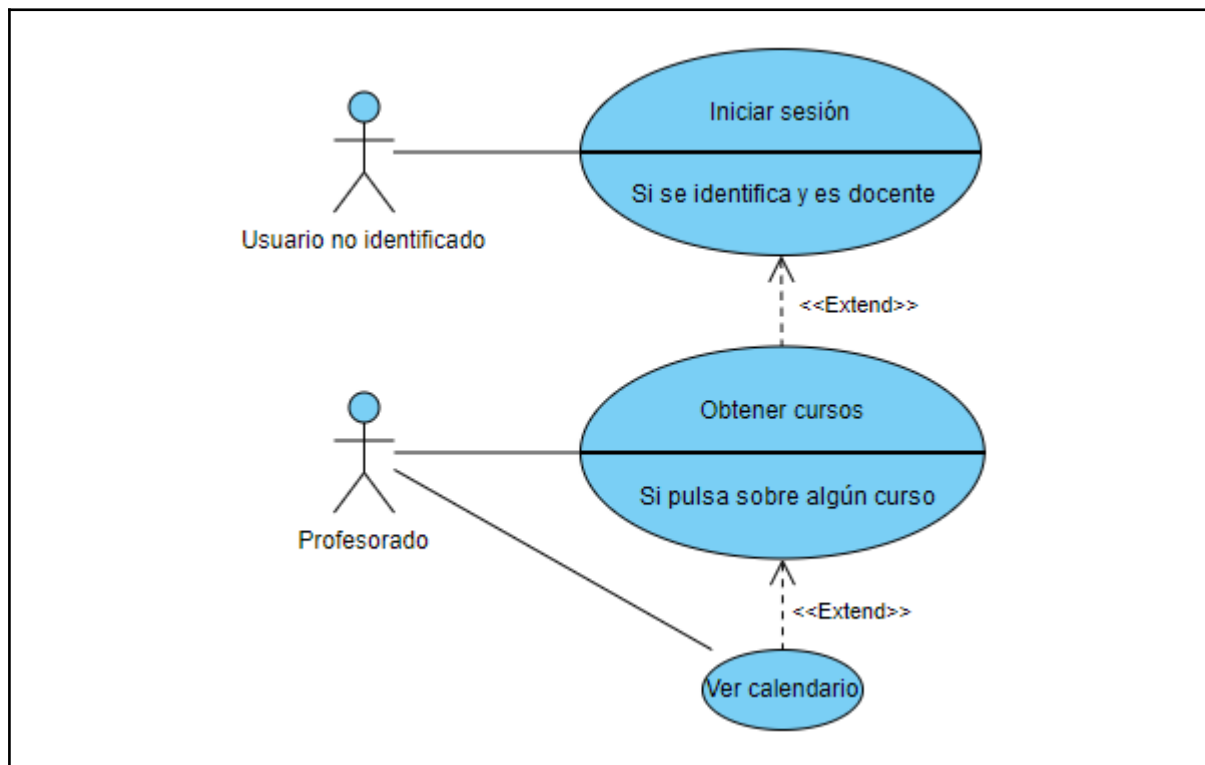
curso10

Pasados

curso11

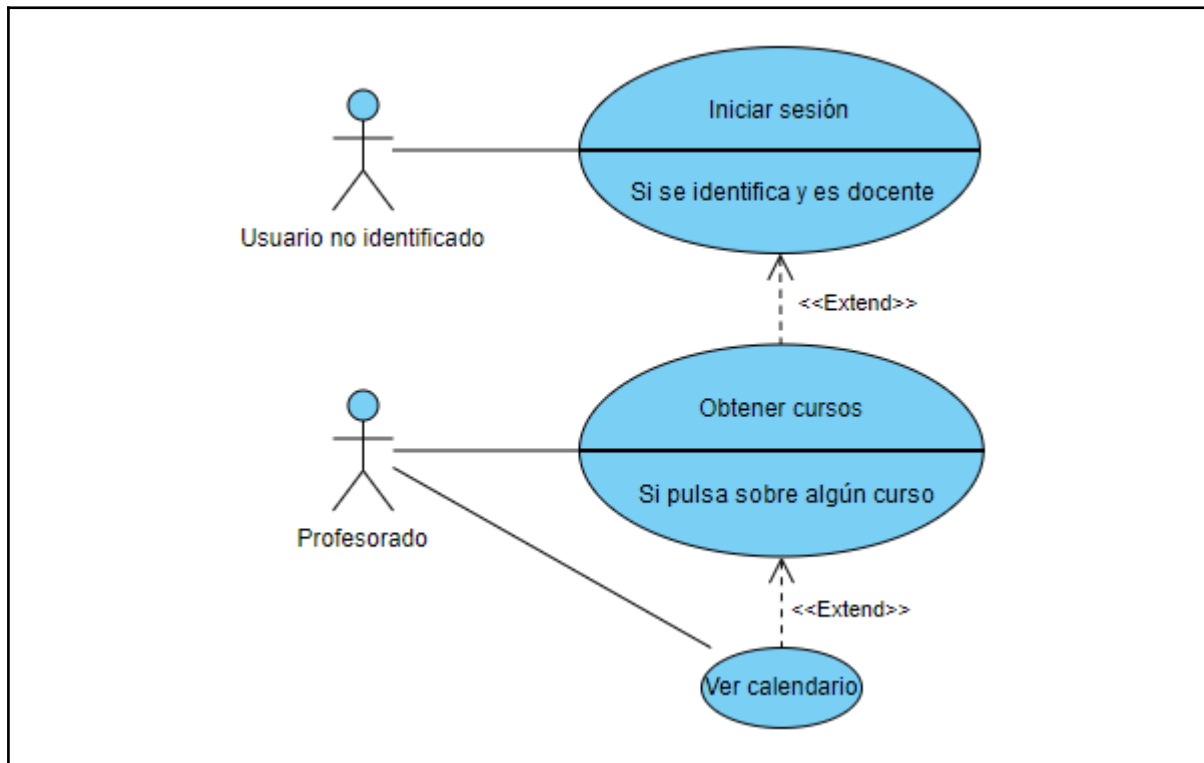
Fig. 32.- Pantalla principal del docente

Obtener cursos



Nombre	Obtener cursos
Descripción	Permite obtener los cursos que imparte al profesorado identificado.
Actores	Profesorado.
Precondiciones	El docente debe haber iniciado sesión.
Requisitos no funcionales	Ninguno.
Flujo de eventos	1. Nada más el docente logre iniciar sesión, el sistema le redirige a la pantalla principal del profesor y esta le muestra sus cursos (Fig. 32).
Postcondiciones	El sistema mostrará al docente identificado los cursos que imparte.

Ver calendario



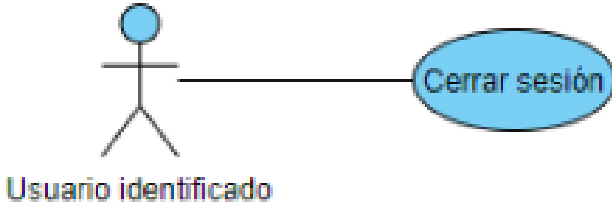
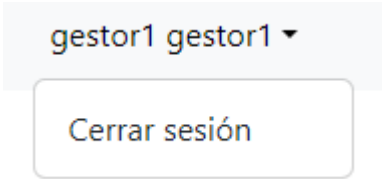
Nombre	Ver calendario
Descripción	Permite ver el calendario que le indica los días apropiados para asignar una tarea a los estudiantes de un determinado curso al profesorado identificado.
Actores	Profesorado.
Precondiciones	El docente debe haber iniciado sesión y debe haber seleccionado uno de los cursos que le muestra la pantalla principal.
Requisitos no funcionales	Ninguno.
Flujo de eventos	1. El docente selecciona uno de los cursos que le muestra la pantalla principal (Fig. 32), entonces la pantalla le muestra el calendario que le indica los días apropiados para asignar una tarea a los estudiantes del curso (Fig. 33).
Postcondiciones	El sistema mostrará al docente el calendario que le indica los días apropiados para asignar una tarea a los estudiantes del curso que ha escogido en la pantalla principal.

Interfaz gráfica



Fig. 33.- Calendario del curso

Cerrar sesión

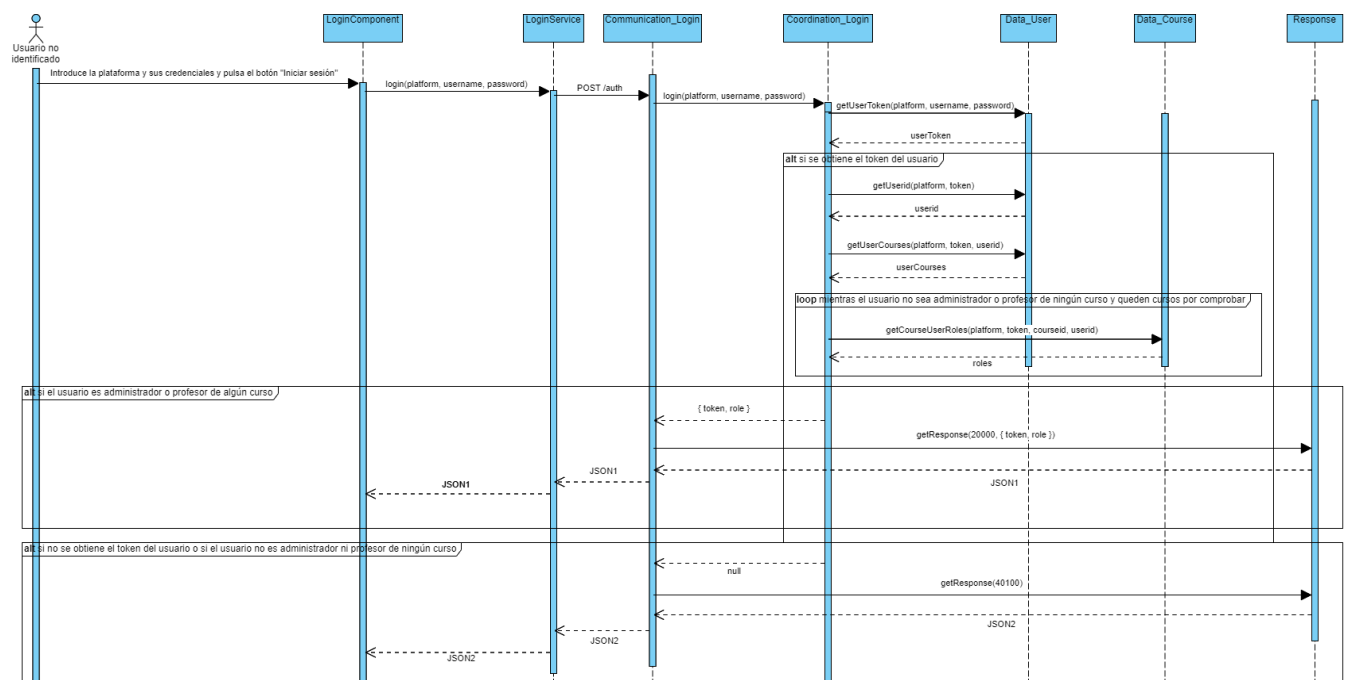
 <pre>graph LR Actor[Usuario identificado] --- UseCase((Cerrar sesión))</pre>	
Nombre	Cerrar sesión
Descripción	Permite cerrar sesión a los usuarios identificados.
Actores	Usuario identificado.
Precondiciones	El usuario debe haber iniciado sesión.
Requisitos no funcionales	Ninguno.
Flujo de eventos	1. El usuario identificado pulsa el botón Cerrar sesión en la cabecera de la web (Fig. 34), entonces el sistema le redirige a la pantalla de inicio (Fig. 29).
Postcondiciones	El usuario identificado habrá cerrado sesión.
Interfaz gráfica	<div data-bbox="791 1099 1173 1276"></div> <p data-bbox="782 1384 1182 1417">Fig. 34.- Botón Cerrar sesión</p>

ANEXO II.- DIAGRAMAS DE SECUENCIA

En este anexo se presentan los diagramas de secuencia correspondientes a los casos de uso extendidos del anexo anterior. Los diagramas de secuencia son una solución de modelado dinámico popular en UML porque se centran específicamente en líneas de vida o en los procesos y objetos que coexisten simultáneamente, y los mensajes intercambiados entre ellos para ejecutar una función antes de que la línea de vida termine.

Con el objetivo de simplificar los diagramas se ha decidido suponer que el token es válido en todos aquellos que requieren autenticación. En el caso contrario, como ya se ha mencionado anteriormente, la API REST devuelve el código de error estándar 401 UNAUTHORIZED de HTTP.

Iniciar sesión



El objeto JSON1 sigue el siguiente formato:

```
{
  "status": 200,
  "code": 20000,
  "msg": "Token y rol obtenidos",
  "body": { token, role }
}
```

El objeto JSON2 sigue el siguiente formato:

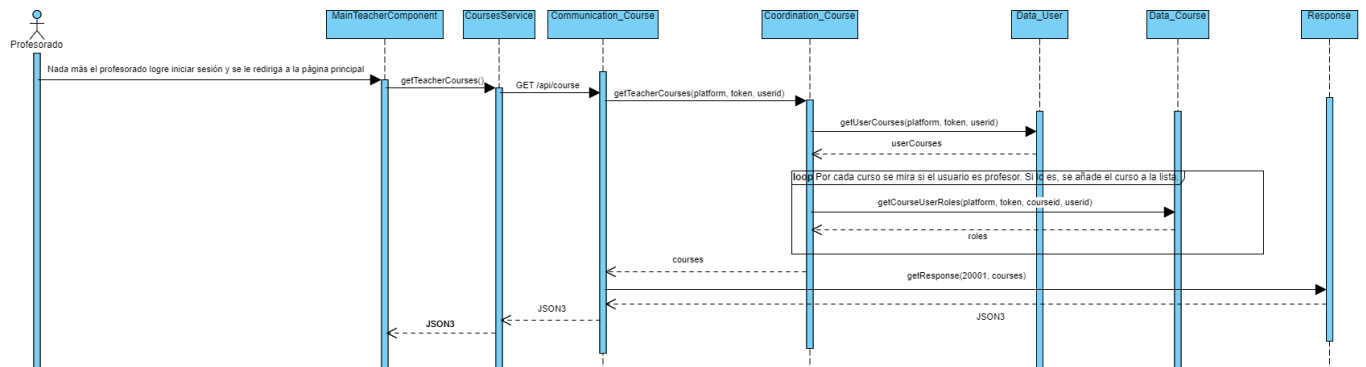
```
{
  "status": 401,
```

```

        "code": 40100,
        "msg": "¡No se han obtenido el token y el rol dado que el nombre de usuario y la contraseña no corresponden a ningún administrador ni a ningún docente de la plataforma!",
        "body": null
    }

```

Obtener cursos



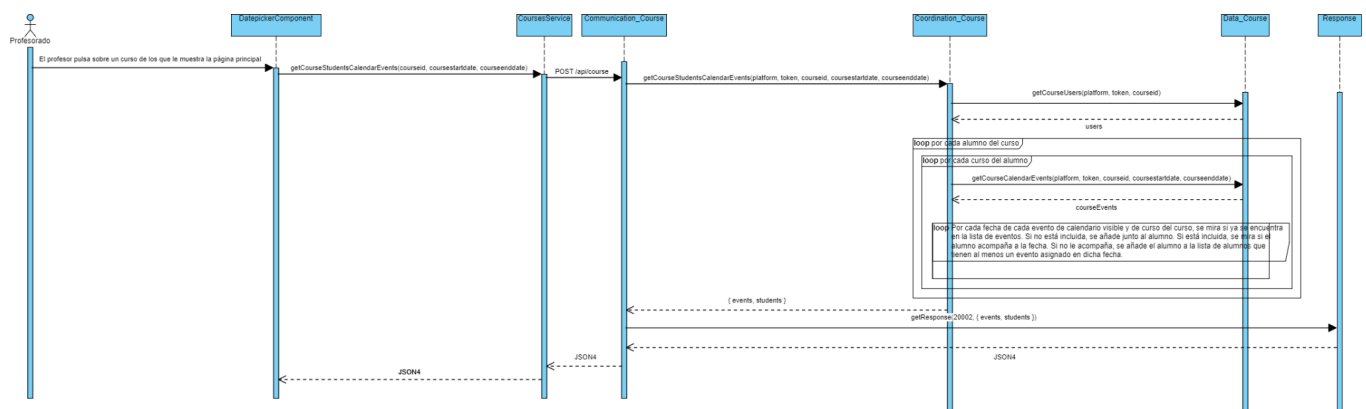
El objeto JSON3 sigue el siguiente formato:

```

{
    "status": 200,
    "code": 20001,
    "msg": "Cursos del profesor obtenidos",
    "body": courses
}

```

Ver calendario



El objeto JSON4 sigue el siguiente formato:

```
{  
  "status": 200,  
  "code": 20002,  
  "msg": "Eventos de calendario y número total de alumnos del curso  
obtenidos",  
  "body": { events, students }  
}
```

Cerrar sesión

