



Universidad de Valladolid

**ESCUELA DE INGENIERÍA INFORMÁTICA
TRABAJO DE FIN DE GRADO**

**Grado en Ingeniería Informática
Mención en Computación**

**Procesos ETL y de anonimización
de interacciones en Moodle para su posterior análisis.**

Alumno: Iván López de Munain Quintana

Tutor: Carlos Enrique Vivaracho Pascual

Tutora: M^a Aránzazu Simón Hurtado

Índice general

Índice de figuras	7
Índice de tablas	9
Resumen	11
Abstract	13
1. Introducción	15
1.1. Entorno de aplicación	17
1.2. Objetivos del proyecto	18
1.2.1. Limitaciones	19
1.3. Estructura de la memoria	19
I Metodología y planificación	21
2. Metodología	23
2.1. Proceso de desarrollo	23
2.1.1. Scrum	23
2.1.2. Aplicación de Scrum al proyecto	25
2.2. Herramientas utilizadas	26
2.2.1. Moodle	26
2.2.2. Overleaf (L ^A T _E X)	26
2.2.3. Python	27
2.2.4. HTML	27
2.2.5. CSS	28
2.2.6. JavaScript	28
2.2.7. R (Rstudio)	28
3. Planificación del proyecto	31
3.1. Identificación y descripción del producto	31
3.2. Identificación de las actividades	33
3.2.1. Planificación y estimación temporal	34
3.2.2. Product Backlog	35
3.3. Identificación de riesgos	37

3.4. Presupuesto económico	40
3.4.1. Presupuesto final	41
4. Seguimiento del proyecto	43
4.1. Sprints	43
4.1.1. Sprint 1	43
4.1.2. Sprint 2	44
4.1.3. Sprint 3	44
4.1.4. Sprint 4	45
4.1.5. Sprint 5	45
4.1.6. Sprint 6	46
4.1.7. Sprint 7	47
4.1.8. Sprint 8	47
4.1.9. Sprint 9	48
4.1.10. Sprint 10	48
4.2. Resumen seguimiento	49
II Documentación técnica: Aplicación para la extracción de datos de Moodle	51
5. Introducción	53
6. Análisis	55
6.1. Requisitos funcionales	55
6.1.1. Requisitos funcionales de información	56
6.1.2. Requisitos funcionales de interacción	56
6.2. Requisitos no funcionales	60
6.3. Atributos de calidad	61
7. Diseño	63
7.1. Arquitectura de la aplicación	63
7.1.1. Patrón arquitectónico MVC	63
7.1.2. Endpoints de la aplicación	65
7.2. Diseño de datos	66
7.3. Diagramas UML	67
7.3.1. Diagrama de actividad	67
7.3.2. Diagrama de clases	68
7.3.3. Diagrama de secuencia CU-11.	69
7.3.4. Diagrama de secuencia CU-12.	69
7.3.5. Diagrama de secuencia CU-13.	69
7.4. Bocetos	71
8. Implementación	77
8.1. Lanzamiento aplicación	77
8.2. Extracción de datos	77
8.2.1. Servicios web de Moodle	77

8.2.2. Web scraping	80
8.3. Descarga de datos	81
8.4. Anonimización de datos	83
8.5. Integración con <i>App2-Dash</i>	84
9. Pruebas	87
9.1. P1 - Anonimización	87
9.2. P2 - Modo profesor	88
9.3. P3 - Modo investigador	88
III Documentación técnica: Aplicación dashboard	91
10.Introducción	93
10.1. Objetivos y motivación	93
11.Análisis	95
11.1. Requisitos funcionales	95
11.1.1. Requisitos funcionales de información	95
11.1.2. Requisitos funcionales de interacción	95
11.2. Requisitos no funcionales	96
11.3. Atributos de calidad	96
12.Planificación de la visualización	99
12.1. Función	99
12.2. Tono	99
12.3. Efecto deseado	99
13.Diseño	101
13.1. Descripción de los datos	101
13.2. Forma de representación y enfoque	101
14.Implementación	109
14.1. Implementación	109
14.1.1. Shiny	110
14.1.2. Plotly	111
IV Conclusiones y trabajos futuros	113
15.Conclusiones	115
16.Futuras investigaciones	117
Bibliografía	119

V	Apéndices	121
A.	Configuración Moodle	123
B.	Manual de la herramienta: Aplicación extracción de datos	127
B.1.	Manual de instalación	127
B.2.	Manual de usuario	128
B.2.1.	Manual de investigador	129
B.2.2.	Manual de profesor	136
C.	Manual de la herramienta: Aplicación dashboard	137
C.1.	Manual de instalación	137
C.2.	Manual de usuario	138

Índice de figuras

2.1. Proceso Scrum	25
3.1. Product Breakdown Structure	32
3.2. Work Breakdown Structure	33
7.1. Modelo entidad relación.	66
7.2. Diagrama de actividad del launcher.	67
7.3. Diagrama de actividad de la aplicación.	68
7.4. Diagrama de clases de la aplicación.	69
7.5. Diagramas de secuencia CU-11.	70
7.6. Diagramas de secuencia CU-12.	70
7.7. Diagramas de secuencia CU-13.	71
7.8. Página inicial de selección	72
7.9. Página de introducción de credenciales (1)	72
7.10. Página de introducción de credenciales (2)	73
7.11. Página de visualización de cursos	73
7.12. Página de visualización de información detallada de un curso	74
7.13. Página de selección y descarga de datos	75
13.1. Página inicial del dashboard.	102
13.2. Pestaña ‘Exploratory’ del dashboard.	103
13.3. Pestaña ‘Time series’ del dashboard.	104
13.4. Gráficos dentro de la pestaña ‘Time series’.	106
13.5. Pestaña ‘Events’ del dashboard.	107
B.1. Página de selección del modo de empleo.	128
B.2. Página de introducción de credenciales (1).	129
B.3. Página de introducción de credenciales (2).	130
B.4. Página de error en la conexión.	130
B.5. Página de selección de cursos.	131
B.6. Página de información detallada del curso.	132
B.7. Página de visualización/descarga de datos (1)	133
B.8. Página de visualización/descarga de datos (2)	134
B.9. Página de visualización/descarga de datos (3)	134
B.10. Página de visualización/descarga de datos (4)	135
B.11. Página de error en la búsqueda/descarga de datos.	135

C.1. Home page del dashboard.	139
C.2. Pestaña ‘Exploratory’ del dashboard.	140
C.3. Pestaña ‘Exploratory’ del dashboard (2).	140
C.4. Pestaña ‘Time series’ del dashboard.	141
C.5. Pestaña ‘Time series’ del dashboard (2).	141
C.6. Pestaña ‘Time series’ del dashboard (3).	142
C.7. Pestaña ‘Events’ del dashboard.	143
C.8. Pestaña ‘Events’ del dashboard (2).	143

Índice de tablas

2.1. Bibliotecas empleadas de Python	27
2.2. Paquetes empleados de R	29
3.1. Planificación de sprints	34
3.2. Historias de usuario priorizadas	35
3.3. Historias de usuario priorizadas	36
3.4. Significado de los clusters	38
3.5. Estimación de presupuesto económico	40
3.6. Horas invertidas en cada sprint	41
3.7. Coste final del proyecto	41
4.1. Resumen sprint 1	44
4.2. Resumen sprint 2	44
4.3. Resumen sprint 3	45
4.4. Resumen sprint 4	45
4.5. Resumen sprint 5	46
4.6. Resumen sprint 6	46
4.7. Resumen sprint 7	47
4.8. Resumen sprint 8	47
4.9. Resumen sprint 9	48
4.10. Resumen sprint 10	48
4.11. Horas invertidas por tarea	49
4.12. Calendarización historias de usuario.	50
6.1. CU-11. Introducción de credenciales	56
6.1. CU-11. Introducción de credenciales	57
6.2. CU-12. Selección de asignaturas	58
6.3. CU-13. Selección y descarga de datos	59
6.3. CU-13. Selección y descarga de datos	60
14.1. Widgets empleados.	110

Resumen

Este proyecto pretende desarrollar una aplicación web que permita extraer de forma sistematizada la máxima cantidad de datos posibles del sistema *e-learning* con mayor aceptación universitaria, *Moodle*. Esta aplicación tiene dos finalidades bien diferenciadas: la primera está asociada al ámbito de investigación, posibilitando la obtención de cantidades masivas de información anonimizada; mientras que la segunda se vincula a la ayuda de la práctica docente. Esta última, permitirá a profesores universitarios acceder a los datos de sus asignaturas y monitorizarlas, analizarlas e incluso mejorarlas gracias a la ayuda de un *dashboard* desplegado en una segunda aplicación web. Ambas aplicaciones son desarrolladas en lenguajes de programación distintos (*Python* y *R*) por lo que son integradas de forma conjunta.

Palabras clave: *Python, Flask, R, Shiny, HTML, JS, CSS, Moodle, sistemas e-learning, anonimización, big data, API, servicios web, web scraping, REST, dashboard, SCRUM, investigación, análisis del aprendizaje, aplicación web, interacciones, notas, procedimientos estadísticos, interfaces, técnicas ETL, base de datos de Moodle*

Abstract

This project aims to develop a web application that allows extract automatically the maximum amount of data possible from the *e-learning* system with the greatest university acceptance, *Moodle*. This application has two different purposes: the first is associated with the field of research, making it possible to obtain massive amounts of anonymized information; while the second is linked to help the university teaching. The last one will allow university professors to access the data of their subjects and monitor, analyze and improve them thanks to the aid of a *dashboard* deployed in a second web application. Both applications are developed in different programming languages (*Python* and *R*) so they are integrated in such a way that they are executed jointly.

Keywords: *Python, Flask, R, Shiny, HTML, JS, CSS, Moodle, systems e-learning, anonymization, big data, API, web services, web scraping, REST, dashboard, SCRUM, research, learning analytics, web application, logs, grades, statistical procedures, interfaces, ETL techniques, Moodle database*

Capítulo 1

Introducción

Desde tiempos pasados, se ha encontrado en la educación de la sociedad un factor clave para la prosperación de la misma. Por este motivo, la enseñanza de las nuevas generaciones se ha convertido en un aspecto categórico y primordial. Actualmente, gracias a los progresos tecnológicos, los procedimientos y metodologías empleados son cambiantes adoptando una tendencia claramente telemática.

Todo proceso educativo y sus recursos están centrados en el aprendizaje del alumno. Existe una amplia variedad en la literatura que versa sobre el tema, focalizándose en cómo mejorar y medir ese proceso de aprendizaje. En los últimos tiempos, la aparición de los medios digitales y su uso en la educación (e.g. los sistemas *e-learning*), ha proporcionado no solo una nueva herramienta de interacción con el estudiante, sino una posibilidad de obtener nuevos datos que nos proporcionen información acerca del proceso de aprendizaje del alumno. Es lo que se conoce como Análisis del Aprendizaje, usándose normalmente la expresión en inglés *Learning Analytics*.

Este es el campo en el que se engloba el presente trabajo. Más concretamente, forma parte de un proyecto más amplio que pretende intentar predecir o anticipar el rendimiento del alumno mediante las interacciones de éste con la plataforma digital de aprendizaje. El objetivo final no es predecir su calificación, sino detectar lo antes posible comportamientos que nos permitan anticipar un mal rendimiento y de esta manera poder hacer que el alumno reconduzca su aprendizaje. Aquí es donde se vislumbra la importancia que tiene analizar el proceso de aprendizaje y las ventajas que puede suponer para los procedimientos educativos actuales.

Como en todo análisis, se necesita información, y aquí es donde el presente Trabajo de Fin de Grado adquiere el protagonismo. Su principal objetivo es proporcionar los datos, de manera que sean representativos del problema a estudiar, pero que, además, cumplan con la normativa vigente en protección de datos.

En esta era tecnológica en la que vivimos, es impensable que los procedimientos de la enseñanza no estén asociados a las *Tecnologías de la Información y la Comunicación (TIC)*, sobre todo por el papel tan importante que cobran herramientas como los *Learning Management Systems (LMS)* ¹. Estas aplicaciones pueden ser tanto soluciones comerciales como *Blackboard/ WebCT*, o bien de código abierto como *Moodle*. Sin embargo, gran parte de la funcionalidad de estas plataformas no se explota en su total capacidad. [1]

Diversos estudios identifican a *Moodle* (*Modular Object-Oriented Dynamic Learning Environment*) como la herramienta *open-source* de gestión de aprendizaje con mayor uso y aceptación en entornos educativos de alto nivel [2] [3] [4], en los que los docentes y alumnos pueden interactuar en línea. Dicha plataforma integra diferentes módulos que dan soporte a funcionalidades como la creación, organización, colaboración o evaluación de actividades, entre otras, además de permitir la compartición de información entre usuarios tanto de forma síncrona (chats) como asíncrona (foros de discusión). [1]

La plataforma de *Moodle* proporciona un conjunto de funcionalidades que pueden agruparse en dos clases generales: módulos y recursos. Los recursos corresponden con el material educativo subido a la plataforma como pueden ser ficheros *PDF*, *PowerPoints*, *Word*, etcétera. Mientras que los módulos son componentes creados por *Moodle* que permiten interacciones profesor-alumno, profesor-profesor y alumno-alumno con el fin de conseguir la manipulación y transformación del contenido que almacena la plataforma. [1] [5]

Los módulos, atendiendo a la funcionalidad que ofrecen, pueden dividirse en seis clases genéricas: creación, organización, entrega, reusabilidad, comunicación, colaboración y evaluación [6]. A continuación, se muestran dichos módulos con su principal funcionalidad:

Para la versión de *Moodle* 3.8 los principales módulos son:

- | | |
|-------------------------------|--------------------------------------|
| ▪ Assignment: entregas | ▪ Workshop: entregas |
| ▪ Chat: comunicación | ▪ Forum: comunicación |
| ▪ Choice: evaluación | ▪ Feedback: evaluación |
| ▪ Quiz: evaluación | ▪ Survey: evaluación |
| ▪ Wiki: colaboración | ▪ Glossary: colaboración |
| ▪ IMS: reusabilidad | ▪ External tool: reusabilidad |

¹Normalmente para referirse a las aplicaciones informáticas en el ámbito educativo también se usan expresiones como *e-learning Systems*, *Course Management System (CMS)* o *Virtual Learning Environment (VLE)*.

- | | |
|-------------------------------|------------------------------|
| ▪ SCORM: reusabilidad | ▪ Database: creación |
| ▪ File: creación | ▪ Book: creación |
| ▪ Folder: organización | ▪ Label: organización |
| ▪ Lesson: organización | ▪ Page: organización |
| ▪ URL: organización | |

Hasta el momento, es clara la amplia funcionalidad que ofrece la herramienta *Moodle* y las ventajas que puede aportar en los procesos educativos. Sin embargo, ¿realmente se está explotando al máximo su potencial? Según un estudio realizado en la universidad portuguesa de Aveiro [1], en la mayoría de los casos la plataforma se emplea meramente como repositorio para realizar entregas o descargar material. Además, revela la necesidad del conocimiento en detalle sobre las herramientas disponibles, por parte del profesorado, para que tenga éxito la plataforma *e-learning*. También es importante destacar la diferencia significativa, en el uso de sistemas de gestión de aprendizaje, entre carreras de distinta naturaleza. Generalmente, los grados tecnológicos tienden a hacer un mayor uso de estas herramientas; aunque bien es cierto que puede haber grandes diferencias entre asignaturas del mismo grado. En definitiva, la eficiencia en el uso de sistemas *e-learning* se reduce básicamente a la tipología de la materia a enseñar, pero sobre todo, al profesor que la imparta.

Aún así, *Moodle* tiene un gran potencial que está poco trabajado ya que ni se explota a nivel educativo ni a nivel de Análisis de la Enseñanza. El principal problema es el difícil acceso a sus datos, además de los problemas legales asociados a esta cuestión. Dada la importancia de *Moodle* en el entorno educativo, el proyecto de *Learning Analytics*, en el que se engloba este TFG, pretende explotar el potencial de los datos que proporciona esta plataforma acerca del proceso de aprendizaje del estudiante.

1.1. Entorno de aplicación

Como bien se ha comentado anteriormente, el ámbito principal de aplicación es la educación, más concretamente, el Análisis de la Enseñanza y la investigación educativa en la universidad de Valladolid. Los procedimientos desarrollados a lo largo de este proyecto proporcionan una aplicación web tanto para profesores como investigadores. El uso aportado para los profesores está orientado al *Learning Analytics* para así monitorizar, mejorar y analizar sus propias asignaturas; significando esto que por su profesión ya gozan de acceso a dichos datos de los alumnos por lo que no tendrán que estar anonimizados. En cambio, para el enfoque de investigación, toda la información extraída deberá estar correctamente anonimizada acorde a la normativa vigente de

protección de datos.

La aplicación desarrollada, que da soporte a técnicas de extracción de datos *ETL* (*Extract, Transform, Load*), se ha diseñado con el objetivo de ejecutarla una única vez para obtener la mayor cantidad de información posible, sobre todo para la investigación. La principal ventaja respecto al uso directo de la herramienta *Moodle* reside en la obtención de los datos completamente procesados e integrados, además de anonimizados (en caso de disponerlos para investigación). Este hecho permite el estudio y análisis del impacto que tiene el uso de la plataforma *e-learning* en el sistema educativo, aparte de examinar la evolución de los alumnos y poder emplear este conocimiento en pos de una mejora en el procedimiento de aprendizaje.

1.2. Objetivos del proyecto

El proyecto desarrollado a continuación busca explotar y aprovechar la información procedente de los sistemas de gestión de aprendizaje, más concretamente de la plataforma empleada por la universidad de Valladolid, a saber, *Moodle*. Previamente, ya se ha puesto de manifiesto la existencia de grandes cantidades de datos que pasan desapercibidos si se restringe al uso de las plataformas mediante sus interfaces. Antes de avanzar a explicar los objetivos, es importante remarcar que el proyecto dispone de dos enfoques relacionados pero claramente diferenciados:

- **Investigación:** esta vertiente se caracteriza por tener el objetivo de extraer la mayor cantidad de información anonimizada posible de la plataforma *Moodle*, con el fin último de disponer los datos (procesados, integrados y anonimizados) en manos de investigadores para que realicen estudios más detallados en el ámbito educativo.
- **Learning analytics:** este enfoque busca extraer los datos más significativos, sin anonimizar, de la plataforma *Moodle*. En esta ocasión, el fin último reside en la monitorización, análisis y mejora de una asignatura por parte del profesor encargado de ella. Esto se logra mediante distintos procedimientos estadísticos y algoritmos que son mostrados en un *dashboard* realizado con los datos extraídos.

Tras haber explicado los fines generales de los distintos enfoques del proyecto, a continuación, se expresan los objetivos en términos más específicos:

- Analizar y estudiar la estructura de datos de *Moodle*, entendiendo cómo se organiza la base de datos de la herramienta.
- Explorar la *API* de *Moodle*, además de encontrar los servicios web y funcionalidades de la plataforma.
- Llevar acabo la aplicación de procesos *ETL* que permitan la extracción exhaustiva de datos procedentes del sistema *e-learning* y, así, poder explotar dicha información en posteriores estudios.
- Para permitir el fácil empleo de la información obtenida será necesario aplicar técnicas de procesamiento de datos como limpieza, transformación, eliminación de ruidos, etcétera.

- Desarrollar una aplicación para poder guardar los datos procesados en formato *csv* en una ruta específica del sistema. Para obtener dichos datos, la aplicación debe permitir acceder a los servicios web de *Moodle* y realizar *web scraping* sobre la página web.
- La aplicación desarrollada permitirá realizar una selección de los datos que se quiere descargar; en suma, para facilitar la selección, existirá la opción de previsualizar la información sin tener que descargarla.
- Desarrollar una segunda aplicación para el estudio de los datos extraídos. En esta ocasión, se trata de un *dashboard* en el que se realizarán distintos procedimientos exploratorios y de modelado. El principal fin será facilitar la monitorización de las prácticas de docencia por parte del profesorado y permitir mejorar las técnicas de enseñanza empleadas analizando las eficiencia de las mismas.
- Para el enfoque de *learning analytics* se deberá conseguir integrar ambas aplicaciones. Es decir, tras ejecutar la aplicación encargada de extraer los datos de *Moodle*, se deberá ofrecer la posibilidad de visualizar la segunda aplicación (el *dashboard* que está desarrollado en otro lenguaje) de tal forma que se garantice continuidad y dinamismo.

1.2.1. Limitaciones

La principal limitación que tiene la aplicación de extracción de datos es la actualización de versión de *Moodle*. Es decir, la *API* suele renovar algunos servicios web al cambiar de versión, ya sea modificando el nombre, los parámetros de entrada, la respuesta, etcétera. Esto provocaría que la aplicación quedase desfasada. Además, una parte de los datos se obtiene gracias a *web scraping* y, es ampliamente conocido, que las páginas web suelen ser cambiantes con una frecuencia considerable. Las restricciones y necesidades para emplear la parte de la aplicación que accede a la *API* de *Moodle* vienen descritos en el anexo [A-Configuración Moodle].

Por contra, la aplicación que da soporte al *dashboard* tan solo tiene la restricción de que los datos de entrada tengan el formato *csv* y su estructura sea igual a la explicada en el apartado [8.3.-Descarga de datos]. La principal limitación es la posibilidad de que alguno de los datos empleados para realizar los gráficos no disponga del formato adecuado, aunque se trata de una incompatibilidad de fácil solución.

1.3. Estructura de la memoria

Al inicio de la memoria, tras la portada y los índices, se encuentra tanto el resumen en inglés como en castellano, seguido de una introducción al problema donde se expone la motivación, el entorno de aplicación, los objetivos del proyecto y la descripción de la estructura de la memoria.

A partir de aquí, la estructura de la memoria consta de cinco partes bien diferenciadas. En la primera de ellas, se especifica la metodología y planificación del proyecto, donde se indica el proceso de desarrollo adoptado, su seguimiento y las tecnologías empleadas, además de identificar las actividades, productos y riesgos asociados al TFG. A continuación, se encuentra la parte sobre la documentación técnica de la primera aplicación, a saber, la encargada de extraer los datos de *Moodle*. En esta parte se encuentra la especificación del análisis, diseño, implementación

y pruebas asociadas a esta herramienta. La tercera parte, correspondiente a la documentación técnica de la aplicación que da soporte al *dashboard*, contiene información análoga a la expuesta en la parte previa. A continuación, se encuentra la parte de conclusiones y trabajos futuros.

Finalmente, tras la bibliografía, se encuentra la última parte correspondiente a los apéndices. Aquí se localiza información sobre las configuraciones necesarias que hay que realizar en *Moodle* para emplear los servicios web y los manuales de las aplicaciones desarrolladas (tanto el manual de instalación como el de uso).

Parte I

Metodología y planificación

Capítulo 2

Metodología

2.1. Proceso de desarrollo

Las metodologías ágiles fueron diseñadas para superar las desventajas de la implementación de los métodos clásicos pesados. Existen diferentes prácticas ágiles, entre las que destacan: *Kanban*, *Extreme Programming*, *Atern*, *Crystal Technologies* y *Scrum*. Todas tienen en común cuatro principios que se definen en el *Manifiesto Ágil* [7]:

- La interacción y las personas están por encima de los procesos y las metodologías.
- Trabajar conjunta y colaborativamente se encuentra por encima de sintetizar una documentación entendible.
- La colaboración con el cliente cobra más importancia que el propio contrato de negocio.
- Se rechaza el inmovilismo y se busca adaptarse a los cambios en lugar de seguir un plan.

2.1.1. Scrum

La metodología empleada durante el desarrollo de este proyecto es *Scrum*, caracterizándose por ser un framework ligero, iterativo (porque cada repetición permite analizar y construir) e incremental (ya que se puede incrementar la complejidad y/o funcionalidad constantemente) que permite gestionar tareas complejas. En definitiva, *Scrum* promueve los valores de trabajo en equipo en el que el trabajador forma parte de la ecuación principal.

Características principales [11] [12]:

- Como bien se ha comentado, es una estrategia incremental que está abierta al cambio en vez de adecuarse a la ejecución completa de un plan.
- La colaboración entre los integrantes del equipo se vuelve un aspecto principal. Al igual que la auto-organización y la auto-gestión de tal forma que así el equipo se marca sus ritmos y el desarrollo es progresivo.
- La priorización se convierte en una herramienta potente ya que permite establecer criterios para saber qué requisitos son más importantes y deben ser los primeros en desarrollarse.

- Al primarse el trabajo en equipo y valorar más a las personas, la calidad del resultado se basa en el conocimiento adquirido por los trabajadores por encima de los procedimientos ejecutados.
- El seguimiento de la construcción del producto es de forma incremental a través de iteraciones, ayudándose de una alta interacción con el cliente (quién será el que dé por finalizada la producción del producto).

Artefactos empleados [13]:

- **Product Backlog:** se trata de una lista de requisitos (en forma de *user storie*) ordenada en función de la importancia y prioridad de cada una de las historias de usuario. Se trata de una entidad dinámica que puede ir variando a lo largo del proyecto y nunca llega a estar completa. Es priorizado por el *Product Owner* y repriorizado el inicio de cada sprint.
- **Sprint Backlog:** corresponde con los ítems del *Product Backlog* seleccionados para desarrollar en el sprint en curso.
- **Incremento:** parte del producto terminado, en el sprint actual y en anteriores, en condiciones de ser usado.

Eventos [14] [7]:

La metodología *Scrum* se guía mediante sprints que pueden ir de una semana a un mes y se suceden sin interrupción para mantener la continuidad del proyecto. Durante cada sprint, el *Scrum Team* trabaja para completar uno o más incrementos de un producto mayor. Para cada sprint, se definen los siguientes eventos.

- **Sprint Planning:** el equipo determina los requisitos priorizados para el sprint, además de definir y estimar las tareas para cada requisito. Se genera el *Sprint Backlog* y el objetivo del sprint. Participa tanto el *Scrum Master* como el equipo.
- **Daily Scrum:** se trata de reuniones de menos de 15 minutos en las que participa el equipo y el responsable es el *Scrum Master*. No se resuelven problemas, tan solo se identifican y se pretende dar respuesta a preguntas como ‘¿qué hice desde la última reunión diaria?’, ‘¿qué voy a hacer hasta la próxima reunión?’ o ‘¿qué dificultades tengo para realizar mi tarea?’.
- **Sprint Review:** se trata de una reunión de 2-4 horas en la que participan todos (equipo, *Scrum Master* y *Product Owner*). La finalidad es presentar las nuevas funcionalidades implementadas de la demo del producto, además de generar feedback al respecto.
- **Sprint Retrospective:** se realiza al final de cada sprint y participan todos, sirve para reflexionar sobre el desarrollo del mismo. No debería durar más de una hora.

Participantes [15]:

- **Scrum Master:** se trata del líder y facilitador del equipo. Se encarga de eliminar los impedimentos y responsabilizarse del producto, además de promover los valores, principios y prácticas de *Scrum*. Tan solo hay uno por equipo y es la pieza que conecta al *Product Owner*, al *Scrum Team* y a la organización.

- **Scrum Team:** se trata del equipo (unos 7 $[+/-2]$ integrantes) en el que los roles son difusos y se busca que sea multifuncional e interdisciplinario. Trabajan a tiempo completo en cada sprint, caracterizándose por la auto-organización.
- **Product Owner:** representante de los clientes y *stakeholders*. Dispone de autoridad para cambiar el producto y aceptar/rechazar los resultados de un sprint. Solo hay uno por equipo y es el encargado de priorizar la lista de requisitos y asegurarse de la rentabilidad del producto.

Finalmente, en la Figura 2.1, se muestra un resumen gráfico del proceso llevado a cabo por la metodología ágil *Scrum*.

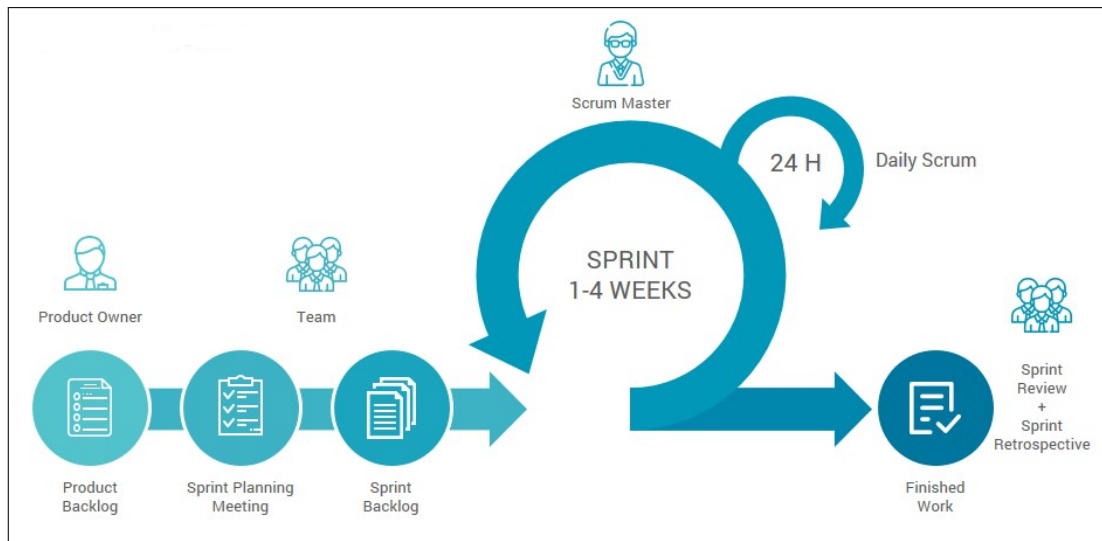


Figura 2.1: Proceso Scrum

2.1.2. Aplicación de Scrum al proyecto

Tras haber sido explicada la metodología *Scrum*, resulta evidente la necesidad de realizar una adaptación al contexto del proyecto aquí desarrollado. Dada la limitación del personal, el alumno, autor del TFG, debe cumplir con varios roles. Tendrá que ejercer el papel de todos los integrantes del *Scrum Team* y el del *Product Owner*, esto significa que se encargará de la identificación, priorización, implementación y gestión de historias de usuario. Mientras que los tutores representarán la figura del *Scrum Master*.

La duración de los sprints se ha establecido en dos semanas y, en vez de emplear *daylies*, se harán *weeklies* los lunes. De este modo, la revisión de los progresos serán realizados el segundo lunes de cada sprint en el *Sprint Review*. Acto seguido, tras el *Sprint Retrospective*, se llevará a cabo el *Sprint Planning* para el siguiente sprint.

La especificación del *Product Backlog* y los distintos *Sprints Backlogs* son mostrados en las secciones posteriores [3-Planificación del proyecto] y [4-Seguimiento del proyecto].

2.2. Herramientas utilizadas

A la hora de desarrollar el proyecto han sido necesarios distintos programas software, los cuales son descritos a continuación.

2.2.1. Moodle

Moodle es una herramienta de gestión de aprendizaje de distribución libre desarrollada en *PHP*. Esta herramienta permite construir comunidades educativas en línea, además de proporcionarle a educadores, administradores y estudiantes un sistema integrado único, robusto y seguro para crear ambientes de aprendizaje personalizados. Tras más de 10 años de desarrollo orientado por la pedagogía de constructivismo social, se trata del sistema *e-learning* más ampliamente utilizado en el mundo, llegando a alcanzar los 79 millones de usuarios a un nivel mundial. [20]

Durante el proyecto se ha trabajado con la versión de *Moodle* 3.8, siendo actualmente la más reciente, que se instaló en una máquina virtual para tener acceso a ella tanto los tutores (para introducir en dicho *Moodle* información anonimizada de sus asignaturas) como el alumno (para trabajar con esos datos). Se puede acceder a dicha máquina virtual vía *SSH* o vía web, siendo esta última la mayoritariamente empleada. Destacar que para acceder vía web se empleó la última versión del buscador *Mozilla Firefox* (75.0).

La obtención de los datos de este sistema *e-learning* se consigue mediante los servicios web de su *API* y a través de la aplicación de técnicas de *web scraping*. Los servicios web habilitan que otros sistemas puedan entrar a *Moodle* y realizar una gran variedad de operaciones mientras que *web scraping* es una técnica utilizada para extraer información de sitios web de forma automática a través de programas software o lenguajes de programación. Ambos métodos son explicados más detenidamente, indicando los detalles de la implementación en el proyecto, en los apartados [8.2.1.-Servicios web de Moodle] y [8.2.2.-Web scraping] respectivamente. Se han usado estos dos métodos distintos de extracción de datos puesto que permiten obtener diferentes tipos de información.

2.2.2. Overleaf (L^AT_EX)

Overleaf [21] es un editor de L^AT_EX en línea. Las principales ventajas frente a usar L^AT_EX en local es que no es necesario instalar ningún programa (tan solo es preciso crear una cuenta con un correo electrónico), dispone de un visualizador PDF para observar cómo se configura el documento, puedes compartir tus documentos y trabajar simultáneamente con otras personas. La única condición para poder utilizar esta herramienta es disponer de conexión a internet.

L^AT_EX es una herramienta empleada para editar textos con alta calidad tipográfica, habitualmente utilizado en artículos o textos científicos. Por este motivo, este es el sistema de composición de textos usado para realizar la documentación del proyecto.

2.2.3. Python

Python es un lenguaje de programación multiparadigma ya que soporta orientación a objetos, programación imperativa y funcional. Es interpretado, dinámico y multiplataforma. Durante el proyecto se emplea para la construcción de scripts que permitan la aplicación de técnicas *ETL* sobre los conjuntos de datos de *Moodle*. Además, se utiliza el *framework Flask* para el desarrollo de la aplicación web que integra a todos los servicios implementados. La versión de *Python* empleada es la 3.7.3.

Framework Flask:

→ Flask [22] es un *micro framework* escrito en *Python* creado para facilitar el desarrollo de aplicaciones web bajo el patrón arquitectónico MVC. Permite ir añadiendo nuevas funcionalidades a la aplicación mediante *plugins*. Este *framework* permite el empleo de páginas web dinámicas facilitando y abstrayendo su construcción. Los principales motivos por los que se ha decidido emplear esta herramienta son que permite desarrollar una app de forma ágil, incluye un servidor de desarrollo, es compatible con *Python3*, proporciona un fácil manejo de rutas y se trata de código abierto del que se dispone extensa documentación.

Bibliotecas empleadas:

→ Las bibliotecas de *Python* empleadas para llevar a cabo el proyecto se listan alfabéticamente en la Tabla 2.1.

Ordenadas alfabéticamente:		
1.- BeautifulSoup	7.- io	13.- path
2.- csv	8.- json	14.- random
3.- collections	9.- mechanize	15.- render_template
4.- datetime	10.- numpy	16.- request
5.- flask	11.- openpyxl	17.- shutil
6.- http.cookiejar	12.- os	18.- xmltodict

Tabla 2.1: Bibliotecas empleadas de Python

2.2.4. HTML

HTML (*HyperText Markup Language*) [23] es un lenguaje de marcado de hipertexto para la elaboración de páginas web. Define tanto una estructura básica como un código para la definición de contenido de una página web, ya sea texto, imágenes, vídeos, etcétera. El lenguaje se caracteriza por estar compuesto en su totalidad de elementos, que a su vez están constituidos por etiquetas, atributos (pares nombre-valor) y contenidos (lo que se encuentre dentro de una etiqueta).

HTML fundamentalmente se emplea para definir la estructura de la aplicación y en menor medida para modificar la apariencia (ya que esto se realiza principalmente mediante *CSS* y *JavaScript*). En el proyecto se emplea para el desarrollo de las vistas de las aplicaciones web, siendo la versión utilizada *HTML5*.

2.2.5. CSS

CSS (*Cascading Style Sheets*) [24] es un lenguaje de hojas de estilo en cascada que da soporte al diseño gráfico para permitir estilizar la presentación de un documento estructurado en un lenguaje de marcado como es *HTML*. *CSS* permite estilizar todas las características en un archivo diferente, creando el diseño por separado y después integrándolo sobre el marcado *HTML*, consiguiendo así que éste sea mucho más limpio y fácil de mantener. Además, *CSS* permite tener múltiples estilos en una página *HTML*. Durante el proyecto, para aportar un diseño gráfico a las vistas de las aplicaciones, se han utilizado tanto hojas de estilo personalizadas por el alumno como el *framework CSS Bootstrap 4*.

Bootstrap 4:

→ Se trata de un *framework front-end* de código abierto para el diseño de aplicaciones web. Contiene plantillas de diseño (basadas en *HTML* y *CSS*) con tipografía, formularios, botones, tablas, etcétera. Además de soportar extensiones de *JavaScript* adicionales.

2.2.6. JavaScript

JavaScript es un lenguaje de programación interpretado el cual se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Durante el proyecto se usa principalmente como parte de la aplicación web permitiendo mejoras en la interfaz de usuario, es por esto que se encuentra complementando a los *scripts HTML*. Gracias a este lenguaje se permite la interacción del usuario con la interfaz, creando así una página web funcional con contenido dinámico.

2.2.7. R (Rstudio)

R se trata de un entorno y lenguaje de programación enfocado al análisis estadístico. Debido a que es un software libre, se ha convertido en uno de los lenguajes más populares en *machine learning* y *minería de datos*. Además de que puede integrarse con distintas bases de datos y existen paquetes que facilitan su utilización desde lenguajes de programación interpretados como *Python*. Otro de sus puntos fuertes es su capacidad para la representación gráfica mediante paquetes como *plotly* o *ggplot2*. Mediante este software se desarrolla la aplicación que da soporte al *dashboard*, siendo fundamentales las bibliotecas *shiny* y *flexdashboard*. La versión de *R* empleada es la 3.5.0.

Paquete Shiny:

→ *R Shiny App* es un paquete de *R* cuya finalidad es el desarrollo de aplicaciones interactivas de visualización de datos. Da soporte a la instalación de una aplicación web en un servidor,

pudiendo personalizarse mediante sintaxis *HTML*, *CSS* o *Javascript*.

Paquetes empleados:

→ Los paquetes empleados de *R* para llevar a cabo el proyecto se listan alfabéticamente en la Tabla 2.2.

Ordenados alfabéticamente:	
1.- dplyr	5.- rmarkdown
2.- flexdashboard	6.- shiny
3.- highcharter	7.- tibble
4.- plotly	8.- treemap

Tabla 2.2: Paquetes empleados de R

Capítulo 3

Planificación del proyecto

La planificación del desarrollo del proyecto se convierte en un aspecto fundamental y determinante en el éxito o fracaso del mismo. A consecuencia de esto, resulta vital identificar y describir tanto los productos como las actividades necesarias para crearlos. Además, es importante determinar el orden y temporización de éstas, aparte de estimar la cantidad de esfuerzo y recursos necesarios para ejecutarlas. En suma, otro aspecto a tener en cuenta son los riesgos del proyecto. A continuación, en las siguientes subsecciones se abordarán cada uno de estos temas.

3.1. Identificación y descripción del producto

Durante este proyecto se han desarrollado dos productos claramente diferenciados: una aplicación que permite extraer información de la plataforma *Moodle* y otra aplicación que da soporte a un *dashboard* (el cual está creado mediante los datos obtenidos del sistema *e-learning*), además de la documentación al respecto. Para cada uno de estos productos, se documenta su descripción según los criterios fijados por *PRINCE2* [7].

Aplicación Moodle:

- **Identificador:** App1-Moodle.
- **Propósito:** extracción y anonimización (cuando sea necesario) de la información que almacena el sistema de aprendizaje *Moodle*.
- **Derivación:** basado en técnicas *ETL* y *web scraping*.
- **Composición:** compuesto por diferentes scripts que acceden a los servicios web de la *API* y a la plataforma web directamente.
- **Formato:** aplicación web desarrollada mediante el framework de *Python*, *Flask*.
- **Estándares relevantes:** las peticiones a los servicios web de la *API* se realizan mediante el protocolo *REST*.
- **Criterios de calidad:** eficiencia y rendimiento en extracción de datos; seguridad en la anonimización de los mismos.

Aplicación Dashboard:

- **Identificador:** App2-Dash.
- **Propósito:** realización de un *dashboard* que mediante algoritmos y procedimientos estadísticos permita el análisis de la información extraída de *Moodle*.
- **Derivación:** basado en procedimientos estadísticos, algoritmos y técnicas de aprendizaje automático.
- **Composición:** compuesto por scripts en *R* que leen los datos y realizan procedimientos estadísticos sobre los mismos.
- **Formato:** aplicación web desarrollada mediante el paquete *shiny* de *R*.
- **Criterios de calidad:** eficiencia en la ejecución de los procedimientos; fiabilidad de los resultados y conclusiones obtenidas.

No obstante, los productos son cualquier tipo de información producida por los desarrolladores del sistema. Estos productos pueden ser entregables, cuando se entregan al usuario al final del proyecto, o intermedios, cuando forman parte de un producto mayor. Por consiguiente, los productos forman una clara jerarquía. Los descriptos previamente conforman los entregables, pero, a continuación, se mostrará un *PBS* genérico (*Product Breakdown Structure*) para apreciar la descomposición de los mismos.

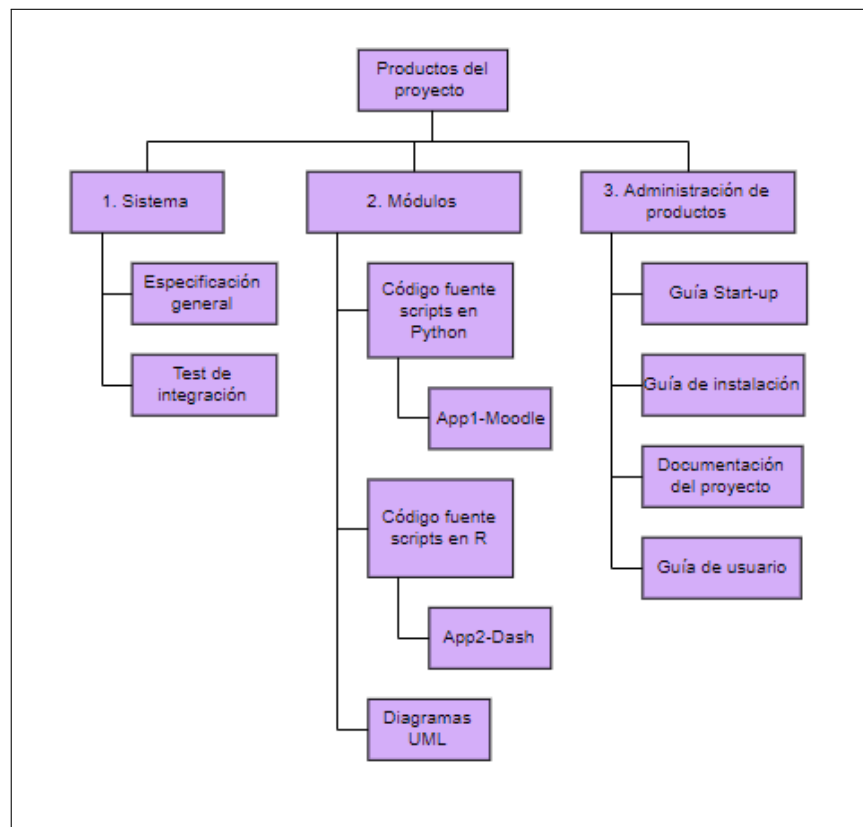


Figura 3.1: Product Breakdown Structure

3.2. Identificación de las actividades

Los productos presentados en el apartado anterior requieren de actividades que los creen, debido a esto resulta importante sintetizar una lista de tareas asociadas al proyecto. Aplicando este concepto, se emplea una aproximación basada en actividades [7]. Para facilitararlo, se va a realizar un *WBS* genérico (*Work Breakdown Structure*) en el que se muestra la descomposición de tareas de alto nivel, según las etapas básicas del ciclo de vida de desarrollo, hasta obtener un conjunto de actividades de bajo nivel (Figura 3.2).

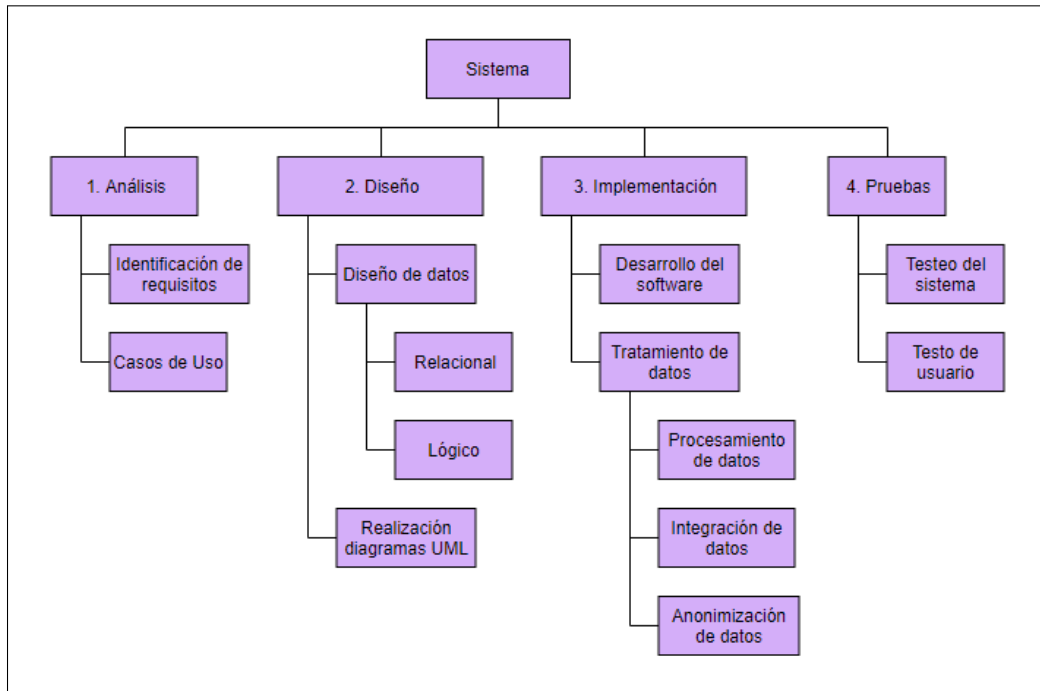


Figura 3.2: Work Breakdown Structure

Al estar aplicando una metodología ágil, tanto la estimación como la planificación del proyecto difieren a los que se aplicaría en un proyecto tradicional guiado por métodos pesados clásicos. Esto se debe al carácter adaptativo, en vez de predictivo, de los procedimientos ágiles [16]. En los apartados siguientes se expondrá tanto el *Product Backlog* como la planificación y estimación temporal del proyecto.

3.2.1. Planificación y estimación temporal

La planificación inicial por fechas de los sprints se muestra en la Tabla 3.1 y, de forma más genérica, la consecución de las historias de usuario en el diagrama de *Gantt* (Tabla 4.12). Lo que respecta a la estimación temporal, en *Scrum* normalmente se emplea la técnica *Planning Poker* que permite obtener una medida del tamaño relativo de todas las historias de usuario respecto a sí mismas [16]. A consecuencia de la adaptación de *Scrum* al contexto de este proyecto y debido a las características del mismo, no tiene sentido aplicar dicha técnica de estimación aquí¹. No obstante, a continuación se realizará una estimación global del esfuerzo necesario para desarrollar el proyecto.

Realizar una estimación global de cuántas horas serán necesarias para desarrollar este proyecto es complicado. Debido a esto, se utiliza el número de créditos correspondientes al Trabajo de Fin de Grado del Grado en Informática (12 ECTS) para realizar una aproximación del número total de horas necesarias. Cada crédito equivale a un trabajo por parte del estudiante de entre 25-30 horas [17], significando esto que el esfuerzo global se encontrará de 300 a 360 horas. Esto supone que cada sprint mostrado en la Tabla 3.1 requerirá un trabajo de aproximadamente 30-36 horas. Cabe destacar que esta estimación de esfuerzo representa las horas útiles y productivas necesarias para desarrollar el proyecto, no se muestran las horas en las que, por ejemplo, se puede estar detenido en un problema o dificultad.

Sprint	Fecha inicio	Fecha finalización
1	17/02/2020	02/03/2020
2	02/03/2020	16/03/2020
3	16/03/2020	30/03/2020
4	30/03/2020	13/04/2020
5	13/04/2020	27/04/2020
6	27/04/2020	11/05/2020
7	11/05/2020	25/05/2020
8	25/05/2020	08/06/2020
9	08/06/2020	22/06/2020
10	22/06/2020	6/07/2020

Tabla 3.1: Planificación de sprints

¹En caso de querer informarse acerca el procedimiento de *Planning Poker* pinche [aquí](#).

3.2.2. Product Backlog

A continuación, se muestran las historias de usuarios, ya priorizadas, en la Tabla 3.2 y la Tabla 3.3. La información proporcionada en este apartado expone el número de historia de usuario, su título, el valor que tiene para el cliente, el esfuerzo que supone desarrollarlo, el riesgo que puede implicar (Tabla 3.2) y una pequeña descripción (Tabla 3.3). Los requisitos serán descritos con mayor extensión en [II-Documentación técnica: Aplicación para la extracción de datos de Moodle] y [III-Documentación técnica: Aplicación dashboard].

Para simplificar la clasificación del valor, esfuerzo y riesgo asociado a una historia de usuario se ha decidido categorizarlos en tres clases: ‘Bajo’, ‘Medio’ y ‘Alto’.

Número	Título	Valor	Esfuerzo	Riesgo
001	BD de Moodle	Medio	Medio	Bajo
002	API y servicios web de Moodle	Alto	Alto	Bajo
003	Aplicar técnicas ETL	Alto	Alto	Bajo
004	Interfaz gráfica de usuario (<i>App1-Moodle</i>)	Bajo	Medio	Bajo
005	Seleccionar datos	Medio	Alto	Medio
006	Anonimizar datos	Alto	Medio	Alto
007	Descargar datos	Alto	Bajo	Medio
008	Interfaz gráfica de usuario (<i>App2-Dash</i>)	Medio	Medio	Bajo
009	Seleccionar datos a visualizar	Medio	Medio	Bajo
010	Seleccionar métodos visualización	Medio	Alto	Bajo
011	Navegar por el dashboard	Medio	Medio	Bajo
012	Integrar <i>App1-Moodle</i> y <i>App2-Dash</i>	Alto	Alto	Bajo

Tabla 3.2: Historias de usuario priorizadas

Número	Descripción
001	Como desarrollador se necesita conocer la base de datos de Moodle con el objetivo de explorar y extraer información asociada.
002	Como desarrollador, desde el enfoque de investigador, se necesita tener acceso a la API de Moodle y utilizar sus servicios web con el objetivo de obtener información.
003	Como desarrollador se necesita tener acceso a la página web de Moodle mediante web scraping con el objetivo de obtener un mayor número de datos.
004	Como usuario quiero visualizar una interfaz gráfica que me permita descargar datos procedentes de Moodle.
005	Como usuario quiero poder seleccionar los datos que quiero descargar y analizar.
006	Como usuario, desde el enfoque de investigador, se necesita que los datos mostrados se encuentren totalmente anonimizados.
007	Como usuario quiero que la aplicación me permita descargar los datos que previamente se han seleccionado.
008	Como usuario, desde el enfoque de <i>learning analytics</i> , quiero visualizar una interfaz gráfica que dé soporte a un dashboard.
009	Como usuario, desde el enfoque de <i>learning analytics</i> , quiero poder seleccionar los datos a representar y los filtros aplicables en el <i>dashboard</i> .
010	Como usuario, desde el enfoque de <i>learning analytics</i> , quiero poder seleccionar los gráficos a visualizar en el dashboard.
011	Como usuario, desde el enfoque de <i>learning analytics</i> , quiero poder navegar por el dashboard y los diferentes layouts.
012	Como usuario, desde el enfoque de <i>learning analytics</i> , quiero visualizar el dashboard a continuación de emplear <i>App1-Moodle</i> , de forma sencilla y dinámica.

Tabla 3.3: Historias de usuario priorizadas

3.3. Identificación de riesgos

Otro aspecto fundamental a tener en cuenta en la planificación de un proyecto es la identificación de riesgos. Éstos están relacionados con problemas futuros que pueden tener efectos adversos sobre los objetivos perseguidos a lo largo del desarrollo, cobrando especial importancia en este proyecto puesto que al disponer de un personal limitado (solo una persona) el impacto puede ser mucho mayor. A continuación, se muestra un framework básico para enfrentarse a los riesgos [7]:

1. Identificación de riesgos.
2. Análisis y priorización de riesgos.
3. Planning y monitorización de riesgos.

Identificación:

→ R.1.- Debido a las características del proyecto y al contexto académico en el que se está desarrollando tan solo existe un único riesgo crítico. Dicho riesgo reside en la posibilidad de desanonimizar los datos obtenidos de *Moodle* pudiendo así obtener información personal sobre alumnos concretos. Este hecho supondría poder conocer el progreso académico de alumnos de forma individual y personalizada.

→ R.2.- Otro riesgo, aunque de menor relevancia, sería de tipo tecnológico. El hecho de no tener experiencia previa en las tecnologías empleadas para realizar las dos aplicaciones (*Flask* y *shiny*) supone un peligro en la consecución de los objetivos.

→ R.3.- El tercer riesgo identificado es de tipo personal, pudiendo darse la situación de que un miembro del equipo enferme.

→ R.4.- Se encuentra el riesgo de la existencia de requisitos ambiguos y cambiantes. Ambos casos se tratan de riesgos de tipo contractual.

→ R.5.- Un riesgo latente es el retraso en las planificaciones realizadas, esto supondría la necesidad de un mayor número de horas para finalizar el proyecto. Se trata de un riesgo de tipo planificación.

→ R.6.- Un riesgo de tipo técnico sería tener problemas con los dispositivos electrónicos que se está trabajando y diera lugar a pérdida de trabajo realizado, traducándose esto en retrasos en la planificación.

Análisis y priorización:

Al identificar pocos riesgos, tanto el análisis como la priorización, se convierten en una tarea sencilla. Un método para estimar la relevancia de cada riesgo y así priorizarlos es emplear la siguiente fórmula:

$$\text{Exposición Riesgo} = (\text{daño potencial}) \cdot (\text{probabilidad de ocurrencia})$$

Normalmente es difícil dar cifras exactas de probabilidad y de daños, por lo que tanto el impacto potencial como la probabilidad de ocurrencia se van a categorizar en ‘Alto’, ‘Significativo’, ‘Moderado’ y ‘Bajo’. Significando cada uno de los clusters lo siguiente [7]:

Cluster	Probabilidad	Impacto ²
Alto	Más del 50 % prob. de ocurrir	Más de 30 % del gasto presupuestado
Significativo	30 %-50 % prob. de ocurrir	20 %-29 % del gasto presupuestado
Moderado	10 %-29 % prob. de ocurrir	10 %-19 % del gasto presupuestado
Bajo	Menos del 10 % prob. de ocurrir	Menos del 10 % del gasto presupuestado

Tabla 3.4: Significado de los clusters

A continuación, se va a mostrar el análisis de los riesgos y se listarán de mayor a menor prioridad (el número identificador del riesgo se mantendrá con respecto al apartado de identificación).

→ R.1.- El riesgo de la posibilidad de desanonimizar los datos de *Moodle* supondría una infracción de las políticas de privacidad personal, violando el derecho de conservación de la confidencialidad de datos personales. Esto supondría un impacto potencial ‘Alto’, principalmente porque supondría problemas legales con los estudiantes afectados (traduciéndose en posibles denuncias de alta cuantía), aunque la probabilidad de ocurrencia es menor que ‘Bajo’. Por estas características se convierte en el riesgo prioritario del proyecto.

→ R.6.- El riesgo de pérdidas de trabajo ya realizado a causa de problemas en el hardware tendría un impacto ‘Alto’ puesto que supondría retrasos y pérdidas de recursos, aunque la probabilidad es ‘Bajo’.

→ R.4.- El riesgo de requisitos cambiantes y ambiguos tendrían un impacto ‘Medio’ y una probabilidad ‘Medio’.

→ R.5.- La posibilidad de retrasos en la planificación supondrían un impacto ‘Medio’ puesto que pondrían en peligro la finalización en la fecha prevista, aunque la probabilidad es ‘Bajo’.

→ R.2.- El riesgo de que el equipo no esté suficientemente formado y carezca de experiencia con determinadas herramientas tendría un impacto ‘Medio’ ya que la curva de aprendizaje sería extensa, aunque la probabilidad es ‘Bajo’.

→ R.3.- Al contar con una única persona trabajando, el riesgo de enfermar cobra mayor relevancia que en otras situaciones. No obstante, el impacto solo sería ‘Bajo’ y la probabilidad ‘Media’.

²El gasto presupuestado para desarrollar este proyecto se encuentra en la subsección [3.4-Presupuesto económico].

Planning y monitorización:

Una vez identificados, analizados y priorizados los riesgos, es necesario decidir cómo afrontar cada uno de ellos: aceptación, reducción, mitigación/contingencia, transferencia (aunque en este contexto académico carece de sentido) o evitación.

→ R.1.- Ya que es el único riesgo que supondría problemas legales resulta evidente la necesidad de reducción de probabilidad de que suceda. Para ello, se adoptan todas las medidas posibles para conseguir la completa anonimización de los datos. Todos los procedimientos llevados a cabo para conseguir la total anonimización de los datos se encuentran explicados en el subapartado [8.4.-Anonimización de datos]. En suma, en la subsección [9.1. P1 - Anonimización], se describen pruebas en determinados supuestos en los que no se han podido desanonimizar los datos.

→ R.2.- La falta de experiencia con las herramientas necesarias se mitiga mediante la reserva de partes de horas del proyecto para alojar la curva de aprendizaje, mientras que el plan de contingencia se basa en apoyarse tanto en los tutores como en profesores universitarios que gozan de un mayor conocimiento.

→ R.3.- Enfermar es el tipo de riesgo que es natural que pueda ocurrir, por eso es mejor aceptarlo. No obstante, es posible mitigarlo dejando holgura entre la fecha de finalización de proyecto y la entrega del mismo. Las tareas retrasadas se realizarán en sprints posteriores.

→ R.4.- Para mitigar los requisitos ambiguos es preciso dar importancia al proceso de elaboración de historias de usuario, mientras que en el caso de requisitos cambiantes la contingencia sería adaptarse a los cambios aprovechándose del empleo de una metodología ágil.

→ R.5.- El riesgo de retraso general en la planificación se mitiga procurando la existencia de holgura entre el tiempo de entrega y de finalización del proyecto, de tal modo que hubiese espacio temporal para realizar algún sprint adicional.

→ R.6.- El riesgo de pérdida de información es fácilmente mitigable y reducible, basta con mantener actualizaciones periódicas en repositorios o discos duros para evitar la pérdida de trabajo realizado.

3.4. Presupuesto económico

En este proyecto, al tratarse de un contexto académico y carecer de necesidades económicas reales, es complicado realizar un presupuesto riguroso. Sin embargo, se va a mostrar el presupuesto económico hipotético que tendría que realizarse si este mismo proyecto se desarrollase en un ámbito empresarial con las personas y tecnologías necesarias.

El Boletín Oficial del Estado de 2020 [18] sitúa el sueldo medio de un Programador/Analista informático, correspondiente al grupo de profesión E de la sección ‘Administración e informática’, en 1.674,96 euros mensuales. Aproximando que al mes se trabajan 160 horas (40 horas semanales durante 4 semanas), el coste por hora sería igual a 10,46 euros. Por lo que teniendo en cuenta que el proyecto consta de 300 a 360 horas, el gasto en dicha persona estaría entre 3.140,55 y 3.768,66 euros. Además, habría que pagar los gastos de seguridad social que corresponden con el 30 % del salario del trabajador [19] (en nuestro caso entre 942,16 y 1.130,59 euros).

En un marco real, también habría que tener en cuenta el sueldo del responsable de zona o manager del proyecto. No obstante, debido a la dificultad para realizar el cálculo de horas que desempeñaría en este proyecto, se obviará.

El coste estimado alcanzaría el valor entre 4,082.71 y 4,899.25 euros, identificando como gasto presupuestado final la media aritmética de dichas cifras: 4,490.98 euros. Aun así, resulta considerable incrementar un 15 % más del presupuesto total (673.64 euros) para lidiar con riesgos que puedan ocurrir a lo largo del desarrollo. Finalmente, la estimación del gasto total se situaría igual a un valor de 5,164.62 euros quedando esta información reflejada en la Tabla 3.5.

	Coste (€)
Analista/Programador Informático	3,454.60
Seguridad social	1,036.37
Fondo auxiliar	673.64
Suma total	5,164.62

Tabla 3.5: Estimación de presupuesto económico

3.4.1. Presupuesto final

Tras terminar el proyecto, se observa que el número total de horas necesarias es igual a 372.5 en lugar de las 330 horas que habían sido estimadas. Dichas horas empleadas quedan repartidas entre los distintos sprints de la siguiente manera (Tabla 3.6):

Sprint	Número de horas
1	35
2	34
3	32
4	31
5	41
6	34.5
7	34
8	53
9	37
10	41
Total	372.5

Tabla 3.6: Horas invertidas en cada sprint

A causa de estas variaciones, los gastos asociados al proyecto se ven afectados. Esta información se muestra en la Tabla 3.7 en la que se expone el coste neto final del proyecto. En suma, se observa cómo fue necesario emplear casi la totalidad del fondo auxiliar por lo que la estimación realizada fue correcta y además ninguno de los riesgos se materializó.

	Coste (€)
Analista/Programador Informático	3,896.35
Seguridad social	1,168.90
Suma total	5,065.25
Fondo auxiliar empleado	574.27

Tabla 3.7: Coste final del proyecto

Capítulo 4

Seguimiento del proyecto

Este apartado pretende mostrar el seguimiento del desarrollo del proyecto, exponiendo la ejecución tanto de tareas asociadas a las historias de usuario como de tareas independientes pero necesarias para la consecución de los objetivos finales. Para cada uno de los sprints, se especificarán las tareas ejecutadas, el tipo asociado (relacionada a una historia de usuario o no), su estado al finalizar el sprint (finalizada o no), una descripción y el número de horas que se han dedicado a dicha actividad. Cabe resaltar que el tipo de una tarea puede ser:

- Relacionada a una historia de usuario. En este caso se representará mediante las siglas ‘US’ (User Story) seguido del identificador de la historia de usuario asociada.
- Independiente de las historias de usuario. Dada esta situación puede clasificarse de las siguientes maneras: ‘Trabajo’ (cuando son tareas necesarias) y ‘QA’ (cuando son tareas vinculadas con la comprobación de calidad y test).

Otro punto a destacar es que las tareas que no se completen en un sprint formarán parte del *Sprint Backlog* del sprint siguiente. Además, se puede concluir de este seguimiento que aquellas tareas que han requerido una cantidad igual o superior a 12-14 horas podrían haberse fragmentado en sub-tareas.

4.1. Sprints

4.1.1. Sprint 1

A lo largo del sprint 1 se ha comenzado la documentación del proyecto: incluyendo la introducción, identificación del alcance y objetivos, metodología a desarrollar, tecnologías a emplear, identificación y reflexión sobre las historias de usuario, etcétera. Además de iniciar el análisis y exploración de la BD de *Moodle*. Estas tareas suponen un total de 35 horas invertidas (Tabla 4.1).

Tarea	Tipo	Estado	Horas	Descripción
T001	Trabajo	Completado	5	Documentación introducción y objetivos
T002	Trabajo	Completado	6	Documentación metodología
T003	Trabajo	Completado	6	Documentación tecnologías
T004	Trabajo	Progreso	7	Documentación planificación
T005	Trabajo	Completado	4	Historias de usuario
T006	US001	Progreso	6	Exploración BD Moodle
T007	Trabajo	Progreso	1	Documentación seguimiento
Total invertido: 35 horas				

Tabla 4.1: Resumen sprint 1

4.1.2. Sprint 2

En el sprint 2 se ha continuado con la documentación del proyecto asociada a la planificación (identificación de productos, actividades, riesgos, presupuesto, etcétera). Además, se ha completado la exploración y análisis tanto de la BD de *Moodle* como de su *API*; tras configurar la plataforma se ha iniciado el desarrollo de scripts en *Python* que permitan acceder a la *API*. El cómputo global de este sprint es de 34 horas (Tabla 4.2).

Tarea	Tipo	Estado	Horas	Descripción
T004	Trabajo	Progreso	2	Documentación planificación
T006	US001	Completado	4	Exploración BD Moodle
T007	Trabajo	Progreso	1	Documentación seguimiento
T008	Trabajo	Completado	8	Exploración API Moodle
T009	Trabajo	Completado	4	Exploración servicios web Moodle
T010	Trabajo	Completado	10	Configuración Moodle
T011	US002	Progreso	5	Acceso a API con Python
Total invertido: 34 horas				

Tabla 4.2: Resumen sprint 2

4.1.3. Sprint 3

Durante el sprint 3 se ha finalizado con la documentación del proyecto asociada a la planificación. Además, se ha completado el acceso a la *API* de *Moodle* e implementado distintos servicios web para la extracción de datos. Finalmente, se ha comenzado la implementación de

web scraping sobre la página web de *Moodle*. El número de horas totales invertidas es igual a 32 (Tabla 4.3).

Tarea	Tipo	Estado	Horas	Descripción
T004	Trabajo	Completado	2	Documentación planificación
T007	Trabajo	Progreso	1	Documentación seguimiento
T011	US002	Completado	7	Acceso a API con Python
T012	US002	Progreso ¹	18	Implementación servicios web con Python
T013	US003	Progreso	4	Implementación web scraping con Python
Total invertido: 32 horas				

Tabla 4.3: Resumen sprint 3

4.1.4. Sprint 4

En el transcurso del sprint 4 se ha finalizado la implementación de *web scraping* sobre la página web de *Moodle*. Además, se ha comenzado la documentación de la implementación de los scripts desarrollados en Python hasta el momento y se ha iniciado el desarrollo de la interfaz de *App1-Moodle* mediante *HTML*, *JavaScript* y *CSS*. El número de horas totales invertidas es igual a 31 (Tabla 4.4).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Progreso	1	Documentación seguimiento
T013	US003	Completado	12	Implementación web scraping con Python
T014	Trabajo	Progreso	3	Documentación implementación
T015	US004	Progreso	15	Interfaz App1-Moodle
Total invertido: 31 horas				

Tabla 4.4: Resumen sprint 4

4.1.5. Sprint 5

En el sprint 5 se continúa con la documentación tanto de la implementación como del seguimiento. Además, se ha completado el desarrollo de la interfaz *App1-Moodle* y los mecanismos para seleccionar los datos que se quieran descargar, quedando pendiente para el siguiente sprint

¹Se remarca que la tarea T012, que supone la implementación de servicios web para extraer datos, no se considera completada porque conforme se avanza en el proyecto se identifican nuevas necesidades de información proporcionadas por servicios no implementados.

la integración en *Flask* y la anonimización de la información. Se ha necesitado invertir 41 horas para alcanzar estos resultados (Tabla 4.5).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Progreso	1	Documentación seguimiento
T014	Trabajo	Progreso	2	Documentación implementación
T015	US004	Completado	12	Interfaz App1-Moodle
T016	Trabajo	Progreso	8	Integración scripts en Flask
T017	US005	Completado	14	Selección de datos deseados
T018	US006	Progreso	4	Anonimización de datos seleccionados
Total invertido: 41 horas				

Tabla 4.5: Resumen sprint 5

4.1.6. Sprint 6

En el sprint 6 se continúa, aunque en menor medida, con la documentación tanto de la implementación como del seguimiento. En suma, se ha completado la integración de los scripts y la anonimización de los datos. Se realiza por primera vez una prueba de verificación de calidad (QA) para comprobar la robustez de la anonimización aplicada; esto denota la importancia que tiene este aspecto en el proyecto. Finalmente también se implementa la posibilidad de descarga de datos en un formato y ruta específica. El cómputo global de horas invertido es igual a 34.5 (Tabla 4.6).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Progreso	1	Documentación seguimiento
T014	Trabajo	Progreso	0.5	Documentación implementación
T016	Trabajo	Completado	6	Integración scripts en Flask
T018	US006	Completado	10	Anonimización de datos seleccionados
T019	QA	Completado	8	Testeo anonimización
T020	US007	Completado	9	Descarga de datos seleccionados
Total invertido: 34.5 horas				

Tabla 4.6: Resumen sprint 6

4.1.7. Sprint 7

Durante el sprint 7 se ha continuado documentando y modificando la implementación realizada hasta el momento. Además, se ha finalizado el desarrollo de la interfaz gráfica de usuario de la aplicación *App2-Dash* y los mecanismos de selección y procesado de los datos que se visualizan en el *dashboard*. El cómputo global de horas invertido es igual a 34 (Tabla 4.7).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Progreso	1	Documentación seguimiento
T014	Trabajo	Progreso	5	Documentación implementación
T021	US008	Completado	23	Interfaz <i>App2-Dash</i>
T022	US009	Completado	5	Selección datos a representar
Total invertido: 34 horas				

Tabla 4.7: Resumen sprint 7

4.1.8. Sprint 8

A lo largo del sprint 8, se implementa en *R* las distintas visualizaciones que componen el *dashboard* con los correspondientes procedimientos estadísticos y la posibilidad de fácil navegación a través del mismo. Además, se realiza la prueba de correcto funcionamiento de la aplicación *App1-Moodle* en modo profesor en un entorno real, siendo monitorizado por los tutores del TFG. No obstante, la tarea se vuelve más ardua de lo esperado y se descubren errores que propician la necesidad de modificar la implementación de *App1-Moodle*. Tras realizar estas modificaciones finalmente se llevó a cabo la prueba de testeo con éxito. El cómputo global de horas invertido es igual a 53 (Tabla 4.8).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Progreso	1	Documentación seguimiento
T014	Trabajo	Progreso	3	Documentación implementación
T023	US010	Completado	18	Implementación métodos de visualización <i>dashboard</i>
T024	QA	Completado	8	Testeo <i>App1-Moodle</i> (modo profesor)
T025	Trabajo	Completado	17	Modificación <i>App1-Moodle</i>
T026	US011	Completado	6	Implementación navegación <i>dashboard</i>
Total invertido: 53 horas				

Tabla 4.8: Resumen sprint 8

4.1.9. Sprint 9

Durante el sprint 9 se intensifica la documentación de la implementación, puesto que se requerían actualizaciones en la misma. Además, se comienza la documentación de los manuales de usuario y de instalación de ambas aplicaciones. También se desarrolla la integración de *App1-Moodle* y *App2-Dash* obteniendo un resultado positivo en la prueba del mismo, aunque fueron necesarias algunas modificaciones en la implementación (Tabla 4.9).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Progreso	1	Documentación seguimiento
T014	Trabajo	Progreso	5	Documentación implementación
T027	Trabajo	Progreso	8	Documentación manuales
T028	US012	Completado	12	Integración aplicaciones
T029	QA	Completado	6	Test <i>App2-Dash</i>
T030	Trabajo	Completado	5	Modificación <i>App2-Dash</i>
Total invertido: 37 horas				

Tabla 4.9: Resumen sprint 9

4.1.10. Sprint 10

Durante el sprint 10, se completa y termina la documentación del proyecto (implementación, diseño, manuales, etcétera). Además, se prueba el modo investigador de la aplicación *App1-Moodle* que supone la necesidad de realizar ciertas modificaciones en la herramienta. No obstante, debido a la tardanza en probar dicha parte de la aplicación, no hubo tiempo de corregir los errores encontrados. Este proceso de prueba es explicado más detenidamente en [9.3. P3 - Modo investigador] (Tabla 4.10).

Tarea	Tipo	Estado	Horas	Descripción
T007	Trabajo	Completado	1	Documentación seguimiento
T014	Trabajo	Completado	7	Documentación implementación
T027	Trabajo	Completado	12	Documentación manuales
T031	QA	Completado	5	Test <i>App1-Moodle</i> (modo investigador)
T032	Trabajo	Completado	8	Modificación <i>App1-Moodle</i>
T033	Trabajo	Completado	8	Finalización memoria
Total invertido: 41 horas				

Tabla 4.10: Resumen sprint 10

4.2. Resumen seguimiento

Realizando un resumen, se muestra en la Tabla 4.11 el total de horas invertidas en cada una de las tareas y el total global de horas útiles/productivas trabajadas para lograr la consecución de los objetivos del proyecto.

Además, para finalizar el seguimiento, se mostrará de forma resumida y genérica, un diagrama de *Gantt* (Tabla 4.12) que muestre la organización y ejecución temporal de las historias de usuario asociadas al proyecto.

Tarea	Horas	Tarea	Horas	Tarea	Horas	Tarea	Horas
T001	5	T011	12	T021	23	T031	5
T002	6	T012	18	T022	5	T032	8
T003	6	T013	16	T023	18	T033	8
T004	11	T014	25.5	T024	8		
T005	4	T015	27	T025	17		
T006	10	T016	14	T026	6		
T007	10	T017	14	T027	20		
T008	8	T018	14	T028	12		
T009	4	T019	8	T029	6		
T010	10	T020	9	T030	5		
Total invertido: 372.5 horas							

Tabla 4.11: Horas invertidas por tarea

	Marzo	Abril	Mayo	Junio	Julio
US001					
US002					
US003					
US004					
US005					
US006					
US007					
US008					
US009					
US010					
US011					
US012					

Tabla 4.12: Calendarización historias de usuario.

Parte II

Documentación técnica: Aplicación para la extracción de datos de Moodle

Capítulo 5

Introducción

En esta parte se documenta toda la información técnica asociada a la primera aplicación desarrollada durante el proyecto, la cual permite la extracción de datos de la plataforma *Moodle*. Cabe destacar que esta aplicación consta de dos usos diferenciados por lo que disponen de distintas funcionalidades y requisitos, estas son: ‘investigador’ y ‘profesor’. Se recuerda que el identificador de dicho producto es *App1-Moodle*.

Antes de avanzar a la documentación técnica de *App1-Moodle*, cabe remarcar que para el modo de empleo ‘investigador’ es necesario hacer una serie de configuraciones en la plataforma de *Moodle* que son especificadas en el anexo **[A.-Configuración Moodle]**.

Capítulo 6

Análisis

En esta sección se van a describir los requisitos del sistema y los atributos de calidad de la aplicación desarrollada. A continuación, dentro de los requisitos, se describen tanto los funcionales (incluyendo los de información e interacción) como los no funcionales.

6.1. Requisitos funcionales

Los requisitos funcionales del sistema que permite la extracción de información de *Moodle* (*App1-Moodle*) son:

1. El sistema deberá permitir aplicar procedimientos *ETL* para la obtención de información procedente de *Moodle*.
2. El sistema deberá permitir extraer datos de la plataforma de aprendizaje *Moodle* a través de los servicios web de la *API* de dicha herramienta. Este proceso se basará en peticiones bajo el protocolo *REST*.
3. El sistema deberá permitir extraer información directamente de la web de *Moodle*. Este proceso se basará en técnicas de *web scraping* y en el análisis de la estructura jerárquica de la página *html*.
4. El sistema deberá permitir introducir las credenciales del usuario de *Moodle* (username y password), así como la URL de login y el token de acceso para los servicios web.
5. El sistema deberá permitir seleccionar el modo de empleo de la aplicación, a saber, investigación o *learning analytics*.
6. El sistema deberá permitir navegar entre las distintas vistas de la aplicación.
7. El sistema deberá permitir seleccionar las asignaturas de las que se quiera descargar datos, además de poder visualizar información acerca de ellas (e.g. número de alumnos, número y tipos de módulos y recursos, etcétera).
8. El sistema deberá permitir seleccionar la información que se quiere descargar, a saber, datos acerca de interacciones, logs, foros, etcétera.

9. El sistema deberá permitir visualizar los datos antes de su descarga, para, así, facilitar su selección.
10. El sistema deberá permitir descargar la información seleccionada, totalmente anonimizada para el modo investigador mientras que sin anonimizar para el modo profesor.
11. Para el modo investigación, el sistema deberá asegurar la completa anonimización de los datos que se quieran extraer, tanto en la previsualización de ellos como en su descarga.
12. En modo profesor, el sistema debe permitir visualizar, de forma continua, el *dashboard* correspondiente con la aplicación *App2-Dash*. Es decir, el sistema debe permitir lanzar la segunda aplicación.

6.1.1. Requisitos funcionales de información

Dentro de los requisitos funcionales, para la aplicación *App1-Moodle* se encuentran los siguientes requisitos de información:

1. El sistema almacenará información sobre las credenciales del usuario de *Moodle*, a saber, el nombre de usuario, contraseña y URL de login.
2. El sistema almacenará información sobre las credenciales de usuario necesarias para acceder a los servicios web de la *API* de *Moodle*, a saber, la URL y el token de acceso; además del protocolo para realizar las peticiones (REST).
3. El sistema almacenará las selecciones que vaya haciendo el usuario (e.g. asignatura deseada).
4. Para el modo profesor, el sistema almacenará las rutas y nombres de los ficheros de datos que se hayan descargado. El fin reside en ejecutar el *dashboard* de la aplicación *App2-Dash*.

6.1.2. Requisitos funcionales de interacción

Un tipo muy relevante de los requisitos funcionales son los de interacción, correspondiendo estos con los casos de uso. A continuación, se describen los distintos casos de uso asociados a la aplicación *App1-Moodle*:

Caso de uso 1.1: Introducción de credenciales

Nombre e ID del CU	CU-11. Introducción de credenciales
Actor	Usuario
Descripción	El usuario introduce sus credenciales en el sistema. Si ha accedido como profesor tan solo es necesario usuario, url de login y contraseña de <i>Moodle</i> ; mientras que para investigador se requiere además de lo nombrado el token de acceso y la url de la <i>API</i> , generando así una petición de entrada.

Tabla 6.1: CU-11. Introducción de credenciales

Precondiciones	<i>PRE:</i> <ol style="list-style-type: none"> 1. Aplicación ya lanzada y selección de modo. 2. El usuario dispone de las credenciales necesarias, siendo estas correctas y válidas. 3. <i>Moodle</i> se encuentra configurado adecuadamente. 4. <i>Moodle</i> se encuentra operativo.
Postcondiciones	<i>POST:</i> <ol style="list-style-type: none"> 1. La petición de entrada junto con las credenciales quedan almacenadas en el sistema.
Flujo normal	<i>Flujo:</i> <ol style="list-style-type: none"> 1. El actor introduce las credenciales necesarias según el modo con el que haya accedido al sistema e indica que quiere acceder a la plataforma. 2. El sistema comprueba las credenciales introducidas. 3. Si las credenciales son correctas, el sistema almacena dichos datos e informa al actor del resultado positivo y da paso a la vista de selección de asignaturas.
Flujo alternativo	<i>Flujo alternativo:</i> <p>F3. Si los datos son incorrectos se informa al usuario del error y no se procede a acceder a la plataforma.</p>
Excepciones	<i>Excepciones:</i> <p>E1. El usuario ha dejado campos requeridos sin rellenar.</p> <p>E2. Las credenciales son incorrectas o no existen.</p> <p>E3. El usuario está bloqueado, eliminado o suspendido.</p>
Prioridad	Alta
Otra info	Es relevante que el usuario disponga de los permisos necesarios para realizar estos procedimientos.

Tabla 6.1: CU-11. Introducción de credenciales

Caso de uso 1.2: Selección de asignaturas de las que se quieren descargar datos

Nombre e ID del CU	CU-12. Selección de asignaturas
Actor	Usuario
Descripción	El usuario busca las asignaturas de las que desea descargar datos, pudiendo usar información detallada sobre ellas para así facilitar su selección.
Precondiciones	<p><i>PRE:</i></p> <ol style="list-style-type: none"> 1. Aplicación lanzada y seleccionado modo investigador 2. Usuario ya está identificado y logueado en el sistema.
Postcondiciones	<p><i>POST:</i></p> <ol style="list-style-type: none"> 1. La selección de las asignaturas queda almacenada en el sistema.
Flujo normal	<p><i>Flujo:</i></p> <ol style="list-style-type: none"> 1. Realización del CU-11. 2. El actor selecciona las asignaturas de las que quiere descargar datos. 3. El sistema almacena la selección realizada por el usuario. 4. El usuario presiona el botón de selección de asignaturas. 5. El sistema mostrará que se ha realizado exitosamente y dará paso a la vista de selección de datos.
Flujo alternativo	<p><i>Flujo alternativo:</i></p> <p>F2. El actor puede ayudarse de un buscador y de una tabla de información detallada para realizar su selección.</p>
Excepciones	-
Prioridad	Alta
Otra info	-

Tabla 6.2: CU-12. Selección de asignaturas

Caso de uso 1.3: Selección y descarga/visualización de datos

Nombre e ID del CU	CU-13. Selección y descarga/visualización de datos
Actor	Usuario
Descripción	De las asignaturas previamente seleccionadas (en el modo investigación) o del total de asignaturas asociadas al usuario (modo profesor), el usuario busca los datos que desea y los descarga/visualiza.
Precondiciones	<p><i>PRE:</i></p> <ol style="list-style-type: none"> 1. Aplicación ya lanzada y selección de modo 2. Usuario ya está identificado y logueado en el sistema. 3. Hay asignaturas seleccionadas.
Postcondiciones	<p><i>POST:</i></p> <ol style="list-style-type: none"> 1. La selección de los datos deseados queda almacenada en el sistema y se descargan/visualizan.
Flujo normal	<p><i>Flujo:</i></p> <ol style="list-style-type: none"> 1. Realización del CU-11. 2. Realización del CU-12. 3. El actor selecciona los tipos de datos que desea descargar/visualizar. 4. El sistema almacena la selección realizada por el usuario y además ofrece una visualización previa de ellos en forma de tabla. 5. El usuario presiona el botón de descargar datos. 6. El sistema descargará los datos con un formato y ruta determinada. Finalmente mostrará que se ha realizado exitosamente y permitirá seguir con otras descargas.

Tabla 6.3: CU-13. Selección y descarga de datos

Flujo alternativo	<i>Flujo alternativo:</i> FA3. El actor puede ayudarse de <i>selectInputs</i> para introducir filtros, además de una visualización previa de los datos para facilitar la decisión de selección. FA5. El actor puede no pulsar el botón si solo desea visualizar los datos y no descargarlos. FA6. Si se ha accedido al sistema en modo investigador los datos se descargan anonimizados, mientras que si es en modo profesor se lanzará la aplicación <i>App2-Dash</i> .
Excepciones	<i>Excepciones:</i> E3. El usuario ha introducido filtros inválidos. E3'. El usuario intenta buscar datos que no existen.
Prioridad	Muy Alta
Otra info	Es indispensable que el usuario logueado tenga los permisos necesarios para realizar todas las operaciones.

Tabla 6.3: CU-13. Selección y descarga de datos

6.2. Requisitos no funcionales

Los requisitos no funcionales que el sistema *App1-Moodle* debe satisfacer son:

1. Los datos descargados se almacenarán en una determinada ruta en formato *csv* separado por punto y coma (;').
2. La interfaz desarrollada permitirá una fácil navegación entre las distintas vistas de la aplicación.
3. La introducción de todos los credenciales por parte del usuario podrá ser realizada en menos de 5 minutos de experiencia con el sistema.
4. La instalación del sistema podrá ser realizada en menos de media hora.
5. Las funcionalidades del sistema podrán ser aprendidas por un usuario sin previa experiencia en menos de una hora, alcanzando un nivel avanzado de empleo.
6. El tiempo de respuesta para la obtención de los datos de *Moodle* no sobrepasará el orden de cinco minutos.

7. El porcentaje de fallos será menor al 2 % de los usos del sistema por un usuario estándar.
8. El mantenimiento del sistema se podrá realizar de forma eficiente con tan solo haber leído la documentación aquí mostrada.
9. La información deseada podrá previsualizarse en una tabla. Dicha tabla deberá disponer de varias pestañas en el caso de que la petición de datos dé lugar a varias respuestas (e.g. petición de información sobre varias asignaturas; una pestaña por asignatura).
10. Las peticiones al servicio web de *Moodle* se realizarán bajo el protocolo REST.
11. Las selecciones que podrá realizar el usuario se basarán en entradas de *radio buttons*, *data-list*, *checkbox* y *selectInput*.

6.3. Atributos de calidad

En este apartado se examina los indicadores de calidad [25] asociados a la aplicación *App1-Moodle*, entre los que destacan los siguientes:

- **Rendimiento:** esta característica no funcional tiene que ver con los tiempos de espera y de ejecución, así como la gestión de la memoria y posibles problemas de concurrencia. Debido a la naturaleza de la aplicación, tan solo es remarcable la eficiencia en el uso de memoria y tiempo de espera para obtener los datos descargados.
- **Usabilidad y simplicidad:** es de gran importancia desarrollar una aplicación usable que sea sencilla de manejar para así facilitar y fomentar su uso. *App1-Moodle* destaca por su sencillez y apenas carecer de complejidades accidentales que surgen de malas decisiones de diseño.
- **Seguridad:** esta característica es la que mayor relevancia cobra en la aplicación. Eso es debido a que se comprueba la identificación de los usuarios que tratan de acceder a las funcionalidades, garantizando así la seguridad del sistema, y maximiza la prioridad de la privacidad de datos anonimizando toda la información descargada/visualizada que sea destinada a la investigación.
- **Extensibilidad:** una característica importante de *App1-Moodle* es la facilidad para agregar nuevos requisitos. Es decir, existe la posibilidad, de forma sencilla, de incluir nuevos servicios web a la aplicación que permitan descargar/visualizar nuevos tipos de datos que almacene la plataforma *Moodle*. O por ejemplo implementar otros protocolos para realizar las peticiones sobre la *API*.
- **Escalabilidad:** se trata de la capacidad del sistema para trabajar con diferentes cargas de trabajo. No es de las características más fuertes de la aplicación puesto que si se incrementase enormemente el flujo de datos a descargar/visualizar el rendimiento de la aplicación se vería afectado. Se ha testeado descargar cantidades de datos en un orden de 10^5 filas por 10 columnas tardando como mucho un minuto de tiempo. No obstante, lo más seguro es que la ejecución con un gran número de datos empeoraría el rendimiento.
- **Compleitud:** el sistema es capaz de realizar todas las operaciones definidas en los requisitos funcionales.

- **Correctitud y consistencia:** se consiguen mediante la ausencia de errores y la coherencia entre todas las operaciones que puede realizar el usuario.

Capítulo 7

Diseño

A lo largo de esta sección se expone la arquitectura de la aplicación (explicando el patrón arquitectónico empleado y su adaptación al contexto), el diseño de datos (incluyendo el diagrama entidad-relación) y los diagramas de clase/secuencia. Además de los bocetos asociados a la interfaz de la herramienta.

7.1. Arquitectura de la aplicación

7.1.1. Patrón arquitectónico MVC

La estructura de la aplicación *App1-Moodle* se basa en la utilización de un servidor web que soporta diversos *endpoints*, aportando cada uno de ellos una funcionalidad determinada al sistema. Estos *endpoints* corresponden con rutas específicas dentro de la página web que se encuentra alojada en la máquina local.

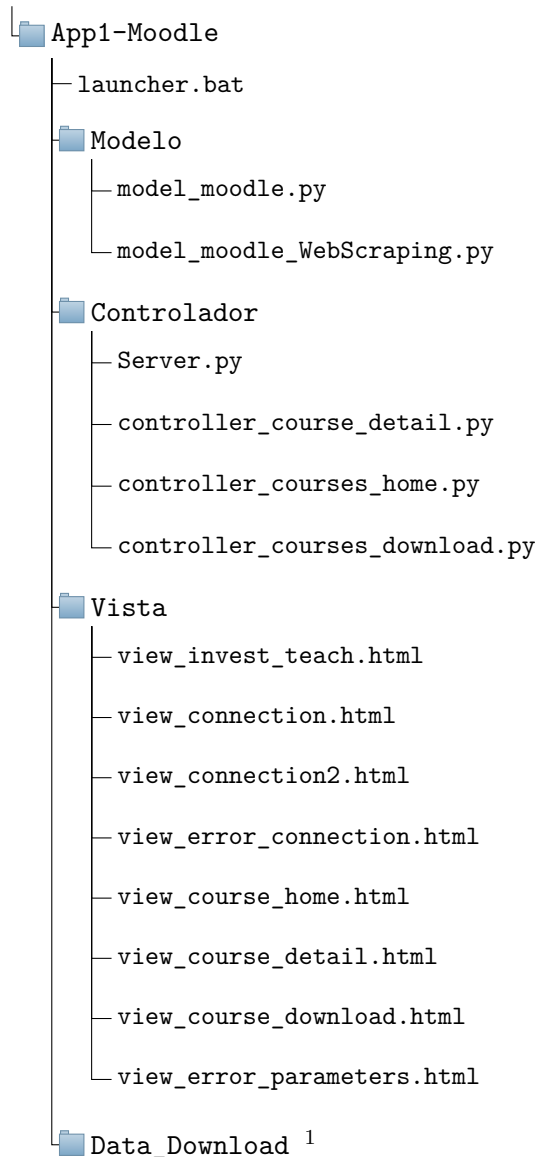
Dicha aplicación se ha desarrollado bajo un patrón arquitectónico *MVC* (Modelo-Vista-Controlador) pasivo, significando esto que el controlador manipula de forma exclusiva el modelo. El funcionamiento se reduce a que el controlador modifica el modelo y notifica a la vista de que éste ha cambiado, esto provoca que el modelo sea completamente independiente de la vista y del controlador. El motivo por el que se ha escogido este paradigma reside en la capacidad que tiene para soportar múltiples vistas en la interfaz de usuario sin dependencia directa entre el modelo y la vista. A continuación, se muestra la correspondencia con cada uno de los componentes:

- **Modelo:** se corresponde con cada una de las fuentes de datos empleadas, tanto la *API* de *Moodle* (servicios web) como la página web de *Moodle* (*web scraping*); se trata de la misma entidad como fuente de datos pero accedida de diferente manera (obteniendo distinta información). Estos componentes encapsulan el núcleo de datos y funcionalidad principal, independientemente del comportamiento de entrada y de salida.
- **Controlador:** se corresponde con el servidor (programado con el framework *Flask* de *Python*) y con los scripts (en *Python*) responsables de dar soporte a los servicios web y al *wrapper*. Sus principales responsabilidades residen en recibir las entradas de usuario como eventos y traducirlos como peticiones de servicio al modelo, además de recuperar datos de este último. Simplificando, gestiona las acciones que ocurren en la vista y la actualiza con los datos procedentes del modelo.

- **Vista:** se corresponde con los distintos documentos *HTML*, cuyo estilo está dotado gracias al uso de *CSS* y al framework *CSS Bootstrap 4*. Dichas vistas se encuentran renderizadas en la url que proporciona el servidor: 127.0.0.1:5000 (host: 127.0.0.1 [local-host], puerto: 5000). La funcionalidad más relevante reside en mostrar la información al usuario y actualizarse en función de la información proporcionada por el controlador.

Como bien se ha comentado, al tratarse de un servidor web, las vistas se alternan basándose en distintas rutas (*endpoints*) dentro de la propia página, detallándose esta cuestión en la siguiente subsección. Aunque antes se muestra cómo los componentes citados se distribuyen en el sistema de ficheros adoptando la siguiente forma:

Aplicación extracción datos Moodle



¹La carpeta *Data_Download* corresponde con un directorio dinámico en el que se almacenarán los datos que se vayan descargando. Su estructura es explicada con mayor detalle en [8.3.-Descarga de datos].

7.1.2. Endpoints de la aplicación

A continuación, se explican los distintos *endpoints* en los que se divide la aplicación web *App1-Moodle*. En este apartado se busca entender la estructura sobre la que se cimienta el sistema, no obstante, aspectos como la usabilidad, instalación o mecanismos de interacción quedan más detallados en la sección **[B-Manual de la herramienta: Aplicación extracción de datos]**. Los *endpoints* son los siguientes:

- **[-1] - /:**
Se trata del *endpoint* raíz (correspondiente con la vista *view_invest_teach*) donde el usuario debe elegir el modo de empleo de la aplicación mediante un *radio button*, ya sea el rol de investigador o el de profesor. Independientemente del rol que se escoga se accederá al *endpoint* [0] que se encuentra en la ruta '/Login'.
- **[0] - /Login:**
Se trata del *endpoint* correspondiente con la vista *view_connection*, donde el usuario tiene que introducir las credenciales de nombre de usuario y contraseña (además de URL de login) de *Moodle*, para realizarlo se dispone de distintos inputs en los que es posible escribir. Se comprobarán las credenciales y en caso de ser válidas, si se ha escogido el modo investigación, se avanzará a la vista *view_connection2*, pasando al *endpoint* [1]. En cambio, si se había elegido el rol de profesor se accederá directamente al *endpoint* [4].
- **[1] - /Auth:**
Este *endpoint* sirve para terminar de introducir el resto de credenciales necesarias para el rol de investigador, a saber, token de acceso, uri y protocolo con el que realizar las peticiones. En caso de ser válidas las credenciales se continuará hacia *view_courses_home*, en caso contrario a *view_error_connection* (en ambas ocasiones *endpoint* [2]).
- **[2] - /check_connection/<string:tok>/<string:uri>/<string:prot>:**
En caso de credenciales válidos, proporciona una tabla con todos los cursos asociados al usuario de *Moodle* con token de acceso *tok*, en la url *uri* y empleando el protocolo *prot*. Además, dispone de un buscador que permite encontrar asignaturas por nombre. Esta vista (*view_course_home*) se ayuda de un botón para acceder a información detallada de cada una de las asignaturas, que se encuentra en otro *endpoint* ([3]), alternándose así con la vista *view_course_detail*. Finalmente, la tabla de asignaturas constará de checkboxes para seleccionar aquellas que se quieran analizar, dando paso al último *endpoint* ([4]) con la vista *view_course_download*.
- **[3] - /check_connection/<string:tok>/<string:uri>/<string:prot>/<string:course_view>:**
Proporciona un menú de pestañas que permiten observar distintos datos, en forma de tabla, asociados a un curso en concreto (cuyo identificador es *course_view*). De esta vista, *view_course_detail*, tan solo es posible retroceder al *endpoint* [2].
- **[4] - /course_selection/<string:tok>/<string:uri>/<string:prot>/<string:list_course>:**
Finalmente, este *endpoint* proporciona unos mecanismos de entrada de información que permiten al usuario seleccionar los tipos de datos, asociados a una asignatura, que desea

descargar. El concepto de información asociada a una asignatura se consigue gracias a una lista de identificadores de cursos almacenada en el *endpoint* [2] (*list_course*). Además de lo expuesto, existe la posibilidad de visualizar la información de forma tabulada sin necesidad de descargarla. Para conseguir este propósito, se ayuda de *query strings* que variarán dependiendo de la información que se quiera buscar ². Este recurso se utiliza para realizar una petición GET sobre este mismo *endpoint* ([4]) pero con los parámetros de la consulta en cuestión. Si se ha llegado a este punto desde el modo investigador, tanto la visualización como la descarga de los datos estará totalmente anonimizada; mientras que si se ha accedido con el rol de profesor no se encontrará anonimizada y se permitirá el lanzamiento de la aplicación *App2-Dash* en la ruta 127.0.0.1:5001.

7.2. Diseño de datos

En la Figura 7.1, se muestra un modelo entidad-relación con el fin de enseñar las relaciones principales que asocian unas entidades con otras, además de identificar los atributos relevantes que poseen dichas entidades.

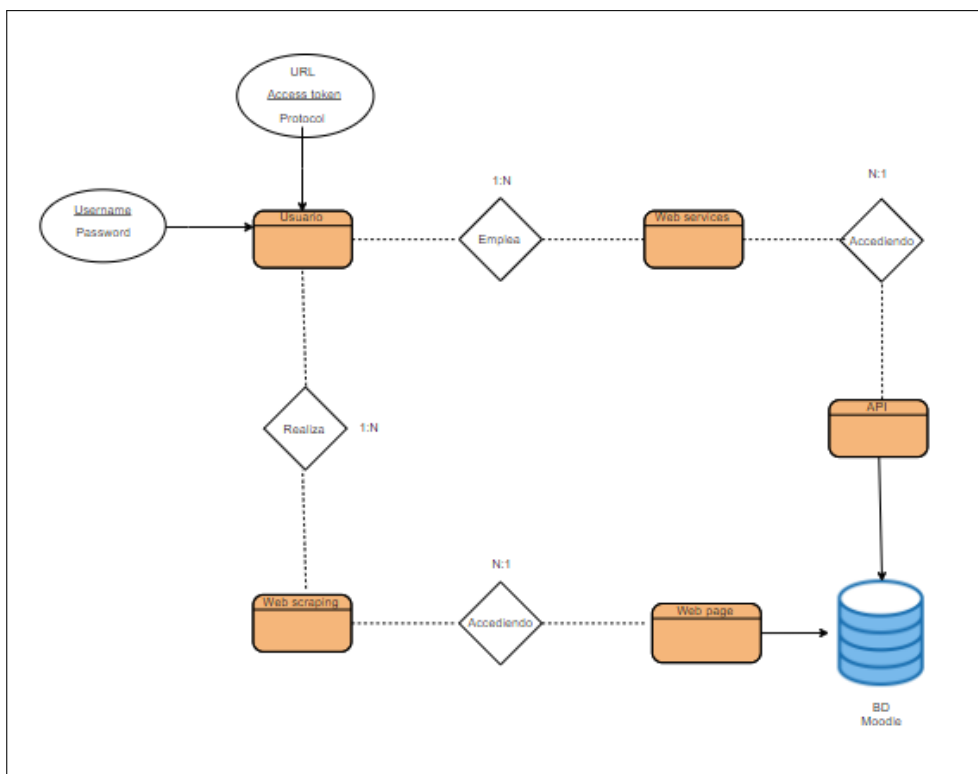


Figura 7.1: Modelo entidad relación.

²Los *query string* son un recurso habitual para enviar parámetros a una página web al realizar una petición GET sobre ella. La estructura general es de la forma: ‘?parameter1=value1¶meter2=value2...’. Por consiguiente, los *query strings* de *App1-Moodle* tan solo variarán según los parámetros y valores empleados.

La Figura 7.1 muestra las relaciones principales existentes en *App1-Moodle*. Como ya se ha comentado, el usuario puede acceder a la aplicación en modo profesor o investigador. La primera diferencia reside en los atributos de cada uno (credenciales necesarios). Esto se basa en que en el modo investigador se extraen datos mediante la *API* (requiere token de acceso, protocolo y URL) y mediante *web scraping* (requiere URL de login, usuario y contraseña de *Moodle*); mientras que para el rol de profesor tan solo se extrae información mediante *web scraping*, por lo que necesita menos credenciales.

7.3. Diagramas UML

7.3.1. Diagrama de actividad

Los diagramas de actividades muestran el flujo de trabajo desde el punto de inicio hasta el punto final detallando muchas de las rutas de decisiones que existen en el progreso de eventos contenidos en la actividad. Debido a que *App1-Moodle* dispone de dos modos de empleo distintos y que, además, se integra con la aplicación *App2-Dash*, un diagrama de actividad resulta fundamental para explicar las diferentes secuencias de actividades del sistema.

En la Figura 7.2 se aprecia el diagrama de actividades asociado al archivo *batch* encargado de lanzar la aplicación, estando explicada la implementación de dicho fichero en secciones posteriores. El procedimiento se reduce a ejecutar el *batch*, disponiendo en primera instancia de una entrada de usuario para saber si se tienen todas las bibliotecas necesarias instaladas. Aquí es donde se representa la bifurcación condicional: en caso negativo se instalarán las bibliotecas mientras que si la situación es que ya se disponen de los paquetes no se realizará nada y se continuará con la ejecución. Tras esta bifurcación, se observa cómo hay una división en el flujo de trabajo dando lugar a dos tareas concurrentes, cada una de ellas representando a un hilo. El hilo principal es el que ejecuta y lanza la aplicación *App1-Moodle*, mientras que el secundario entra en espera durante 10 segundos para posteriormente abrir el navegador *Chrome* en la ruta donde se despliega la aplicación web. Posteriormente dichas tareas se unen y se termina la actividad asociada al *launcher*.

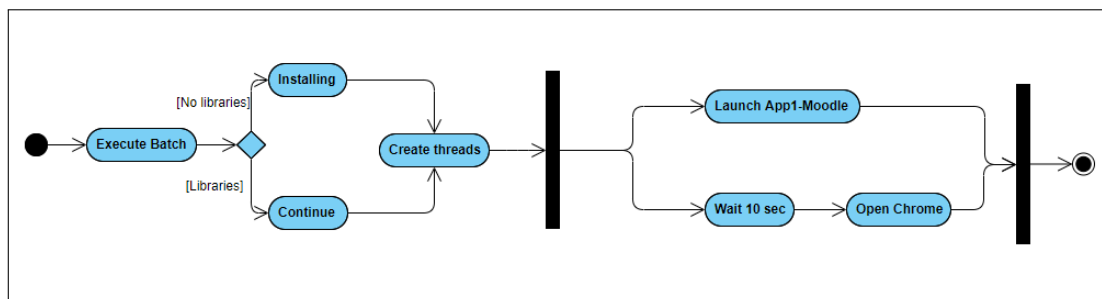


Figura 7.2: Diagrama de actividad del launcher.

A partir de este momento, ya está desplegada la aplicación *App1-Moodle* en el navegador *Chrome*. El diagrama de actividad asociado a este proceso viene expuesto en la Figura 7.3. El primer evento viene descrito por la selección de modo de uso de la aplicación, el cual condicionará el resto de la ejecución. En caso de elegir el modo de empleo ‘Profesor’, será necesario introducir

las credenciales de *Moodle* (nombre de usuario y contraseña) y la URL de login. Posteriormente, se avanza a la selección de los datos, de los cuales se pueden tan solo ver (y de ahí volver otra vez a la selección de datos) o descargar y a continuación lanzar la aplicación *App2-Dash* para visualizar el *dashboard*. Una vez se haya explorado todo el *dashboard*, se puede cerrar esta aplicación y volver a la selección de datos.

Mientras que si se selecciona el modo de uso ‘Investigador’, además de introducir los credenciales requeridos por el otro modo, también se requieren el token de acceso y el protocolo de petición. A partir de ahí, se avanza a la vista de selección de cursos (Courses Home) donde se permite ver información detallada sobre ellos (View details). Una vez realizada la selección de cursos de los que se quiere visualizar/descargar información, se accede a la selección de datos pudiéndolos solo ver (y de ahí volver después a la selección de datos) o descargar anonimizados (y volver otra vez a la selección).

En este diagrama no se ha incluido un nodo final de actividad porque se trata de una aplicación web que puede ser finalizada en cualquier momento. El principal objetivo del mismo es explicar el flujo interno de la herramienta para entender las diferencias entre ambos modos de empleo.

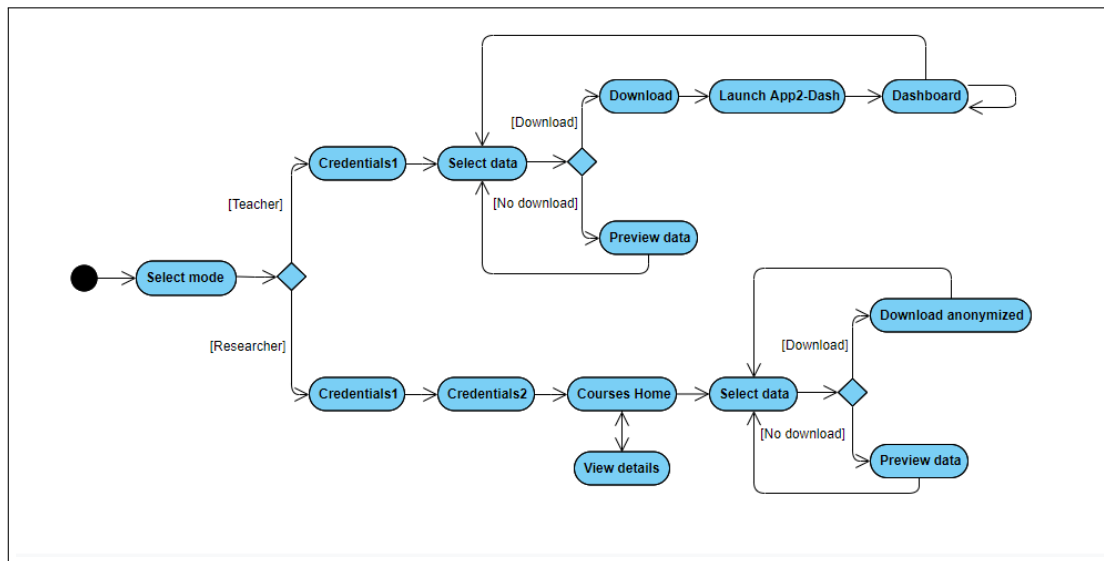


Figura 7.3: Diagrama de actividad de la aplicación.

7.3.2. Diagrama de clases

En el diagrama de clases de la Figura 7.4, se muestra de forma genérica los atributos y operaciones de los usuarios (ya sea en modo profesor o investigador). A la vez de cómo se relacionan con la forma de extracción de datos, ya sea mediante la técnica de *web scraping* o a través de los servicios web de la *API* de *Moodle*.

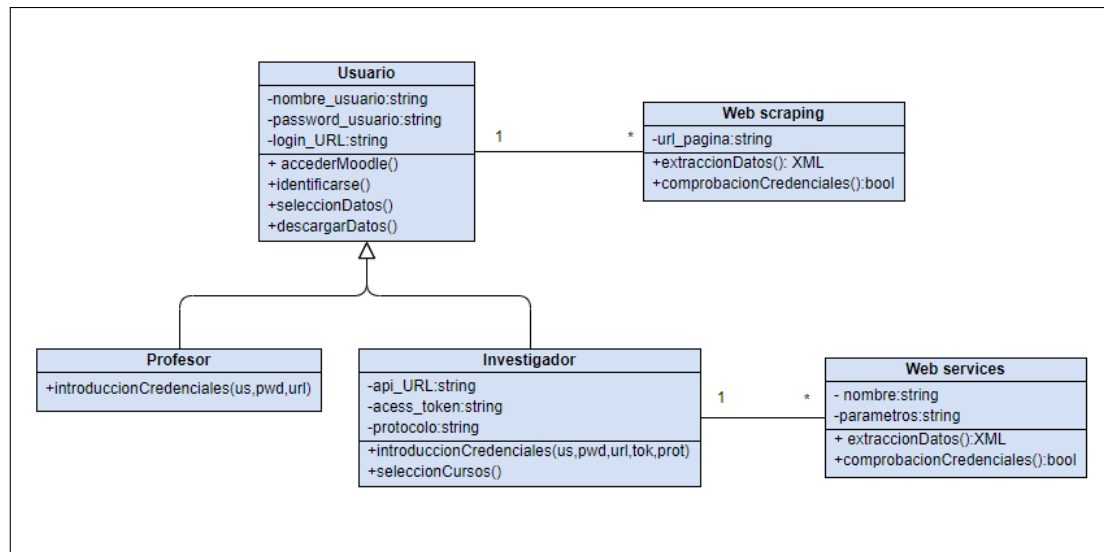


Figura 7.4: Diagrama de clases de la aplicación.

7.3.3. Diagrama de secuencia CU-11.

El diagrama de secuencia mostrado en la Figura 7.5 corresponde con la realización del Caso de Uso 1.1. Éste representa la acción de identificarse en el sistema introduciendo todas las credenciales posibles (correspondiente con el modo investigador). No obstante, se remarca que en caso de loguearse en modo profesor el procedimiento sería el mismo pero no haría falta el token de acceso ni el protocolo. Si las credenciales son correctas, se accede al sistema exitosamente mientras que si hay algún tipo de problema se lanza un mensaje de error.

7.3.4. Diagrama de secuencia CU-12.

El diagrama de secuencia mostrado en la Figura 7.6 corresponde con la realización del Caso de Uso 1.2. Éste representa la acción de seleccionar asignaturas de las que se quieren extraer datos. Para ello, requiere de la realización del Caso de Uso 1.1 suponiendo la necesidad de estar identificado correctamente en el sistema. Después señala las asignaturas de las que quiere obtener información y el sistema almacena la elección.

7.3.5. Diagrama de secuencia CU-13.

El diagrama de secuencia mostrado en la Figura 7.7 corresponde con la realización del Caso de Uso 1.3. Éste representa la acción de seleccionar y descargar/visualizar información deseada. Para ello, requiere de la realización del Caso de Uso 1.1 y del Caso de Uso 1.2 suponiendo la necesidad de estar identificado correctamente en el sistema y que haya asignaturas seleccionadas. Después el actor selecciona los datos deseados y los puede visualizar o descargar. La descarga de datos en modo investigador supone que se encuentran anonimizados, mientras que en modo profesor propicia el despliegue la aplicación *App2-Dash*.

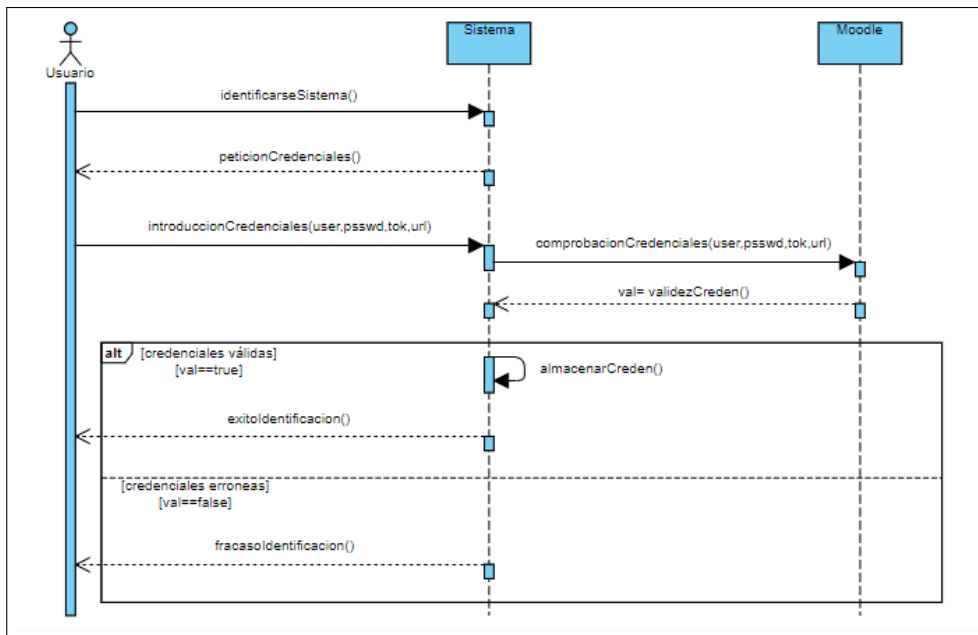


Figura 7.5: Diagramas de secuencia CU-11.

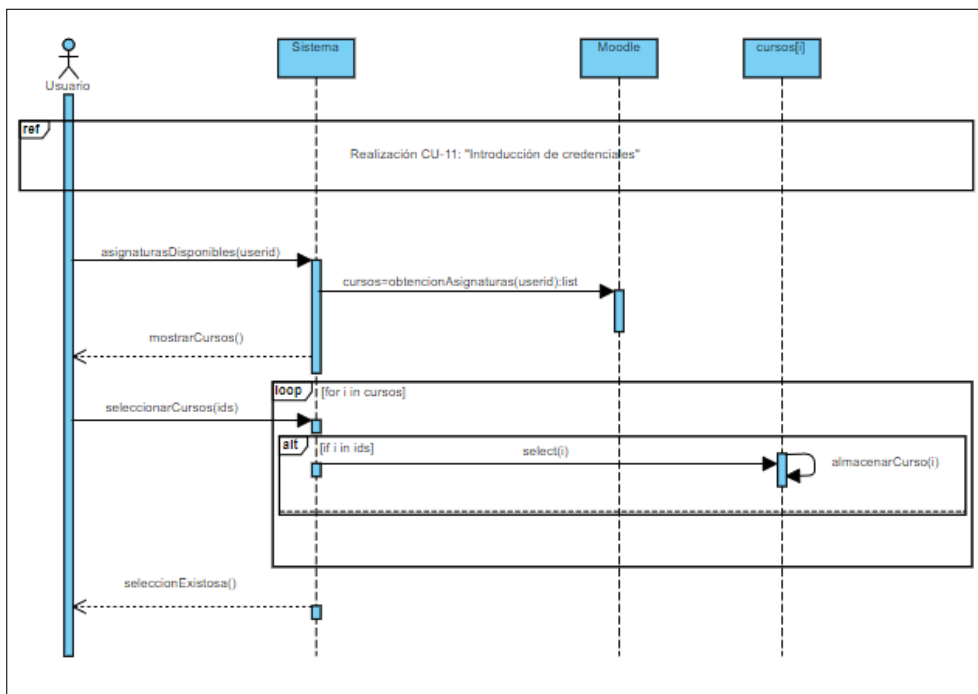


Figura 7.6: Diagramas de secuencia CU-12.

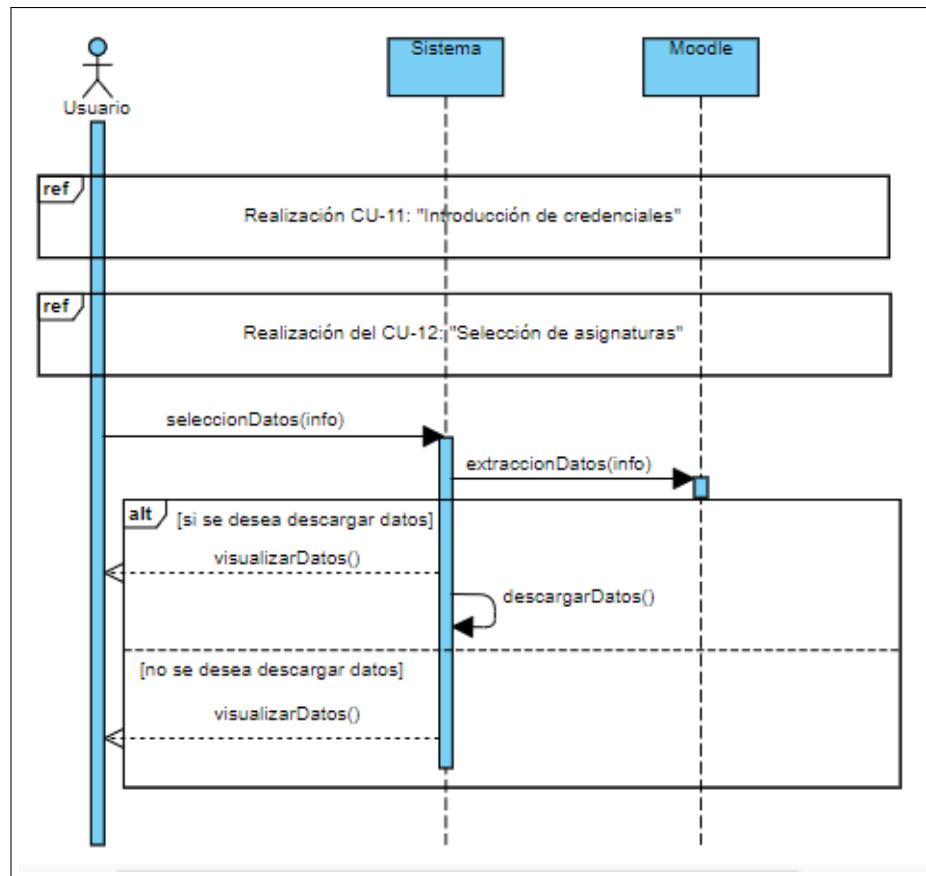


Figura 7.7: Diagramas de secuencia CU-13.

7.4. Bocetos

A continuación se exponen los bocetos de diseño de cómo se van a distribuir las distintas interfaces que componen la aplicación *App1-Moodle*. Aunque en algunos bocetos no se muestre implementada la posibilidad de retroceso (e.g. un botón 'Back'), la herramienta sí que posee de estos mecanismos. Además, tampoco se va a enseñar ningún boceto de las vistas a las que se accede cuando ocurre un fallo debido a su simplicidad y falta de funcionalidad, ya que tan solo informan al usuario de que ha ocurrido un error. No obstante, pese a las pequeñas variaciones que puede haber entre los bocetos y la interfaz real, las cuales tratan de especificarse también en esta sección, la interfaz final se encuentra en la parte [B.-Manual de la herramienta: Aplicación extracción de datos].

En la Figura 7.8 se muestra la distribución de la página inicial de la herramienta, aquí es donde se produce la selección de modo de empleo. La interfaz se reduce a dos partes bien diferenciadas, una para cada modo, en el que se explica en qué consiste cada una y se disponen dos *radio buttons* para realizar la selección. Una vez decidido, tan solo hay que clicar el botón 'Continue' para avanzar de ventana.

ACCESS MOODLE DATA

Learning analytics

☒ Access as a teacher

Research

☐ Access as a researcher

Continue

Figura 7.8: Página inicial de selección

Independientemente de la elección realizada, la siguiente interfaz se ilustra en la Figura 7.9. Ésta está constituida por unas entradas de texto donde se requiere el nombre de usuario y contraseña de *Moodle*. Aunque en este boceto no aparezca, en la herramienta final también se requiere la URL de login del sistema de aprendizaje. Una vez rellenados correctamente, pulsando el botón ‘Next’, se avanza a la Figura 7.10 si se ha accedido como investigador o a la Figura 7.13 en caso de acceder como profesor.

Introduce credentials...

User:	User1
Password:	*****

Next

Figura 7.9: Página de introducción de credenciales (1)

En la Figura 7.10 se exponen el resto de credenciales necesarios para un investigador que requiere el uso de los servicios web para extraer datos. Estos datos son el token de acceso, la URL de la *API* y el protocolo para realizar la petición. Todos se corresponden con entradas de texto mientras que el protocolo se selecciona en una lista. Una vez introducidos, se pulsa el botón ‘Next’ dando paso a la vista expresada en la Figura 7.11.

Introduce credentials...	
Access token:	0878XXXXXX
URI:	https://mymoodle.org
Protocol:	REST

Next

Figura 7.10: Página de introducción de credenciales (2)

La Figura 7.11 tan solo es accesible en modo investigador. Se aprecia cómo la interfaz se va a dividir en dos partes. La zona izquierda se corresponde con una sección explicativa de lo que se ilustra en esta vista (exponiendo que se muestran todas las asignaturas asociadas al usuario que se ha identificado en el sistema y que puede hacer una selección de las mismas). Esta parte también dispondrá de un buscador de ayuda para encontrar las asignaturas por el nombre que se desee y de un botón llamado ‘Select courses’ que servirá para acceder a la siguiente vista una vez se hayan seleccionado las asignaturas de las que se desee descargar/visualizar información. En cambio, en la parte derecha se muestra una tabla con el nombre e identificador de todas las asignaturas. Para cada una de ellas, se tiene la posibilidad de clicar el botón ‘View’ para ver información más detallada sobre la asignatura (Figura 7.12). Además se puede seleccionar un *checkbox* indicando así los cursos de los que se desea obtener datos.

Back

Courses associated

Help to search:

Select courses

Course	Info
<input type="checkbox"/> ID:1 Asignatura de prueba	View
<input checked="" type="checkbox"/> ID:2 Asignatura de prueba 2	View
<input type="checkbox"/> ID:3 Asignatura de prueba 3	View
...	...

Figura 7.11: Página de visualización de cursos

Si se clicaa el botón ‘View’ se accede a la Figura 7.12. Esta vista tan solo sirve para obtener información detallada de una asignatura en concreto, facilitando así la tarea de selección. Se muestra en forma tabulada datos sobre los módulos, recursos, etcétera, asociados a dicho curso. De aquí, pulsando el botón ‘Back’, se volvería a la Figura 7.11.

INFORMATION ABOUT THE COURSE

Details	Modules	Resources	Forums	Notes
---------	---------	-----------	--------	-------

Feature	Value

Back

Figura 7.12: Página de visualización de información detallada de un curso

En cambio, si desde la vista mostrada en la Figura 7.11 se clicase el botón ‘Select Courses’, se accedería a la ventana ilustrada en la Figura 7.13. Aquí es donde vuelven a converger los dos modos de uso de la aplicación, obviamente con algunas diferencias. La vista se divide en dos zonas bien diferenciadas estando la izquierda compuesta por una lista en donde se pueda elegir el tipo de datos que se quiere buscar (cambiarán según el modo escogido), una pequeña documentación seguida de entradas de texto y/o listas que representarán los parámetros (pudiendo ser necesarios o simplemente filtros), un *radio button* para indicar si se quieren descargar los datos y finalmente un botón llamado ‘Download/View data’ para descargar o visualizar los datos. Como ya se ha explicado, en caso de ser investigador se descargarán los datos anonimizados mientras que para un profesor se descargarán y se lanzará el *dashboard*. Si no se selecciona el *radio button*, tan solo se visualizarán los datos, en forma tabulada, en la parte derecha de la ventana (en la Figura 7.13 se representa de forma generalizada los nombres de las columnas: C1, C2, etcétera).

Back

Obtain data about:
Logs

Documentation and parameters:

Parameter 1
Parameter 2

Download data:

☐ Download data

Download/View data

C1	C2	...	C _{n-1}	C _n
~	~	~	~	~
~	~	~	~	~
~	~	~	~	~

Figura 7.13: Página de selección y descarga de datos

Capítulo 8

Implementación

Esta sección muestra los aspectos más relevantes de la implementación de la aplicación *App1-Moodle*, entre los que destacan cómo se lanza la herramienta, cómo se extraen los datos, cómo se anonimizan, cómo se descargan y cómo se integra esta aplicación con el *dashboard* de *App2-Dash*.

8.1. Lanzamiento aplicación

Se ha implementado un fichero *batch* para procurar el lanzamiento de *App1-Moodle* de la forma más cómoda y rápida posible. El archivo se llama *launcher.bat* y permite la interacción con el usuario para saber si es necesario o no instalar las bibliotecas, además, crea dos hilos de ejecución. Uno de ellos corresponde con el lanzamiento del navegador *Chrome* en la ruta 127.0.0.1:5000; éste se duerme 10 segundos para dar tiempo a que el otro hilo, el que corresponde con la ejecución de la aplicación, tenga tiempo suficiente para cargar y lanzar *App1-Moodle* en la ruta citada. Hay alguna ocasión en que 10 segundos se quedan cortos y el navegador se abre sin que esté la aplicación lanzada, en estos casos valdría con actualizar la página web pulsando F5 para que todo funcione correctamente. En este fichero también se consigue extraer la versión de *R* que se está empleando (la cual se pasa como argumento a *App1-Moodle*), esto será fundamental para permitir la integración con la aplicación *App2-Dash* explicada en la subsección [8.5.-Integración con *App2-Dash*].

8.2. Extracción de datos

La obtención de los datos se consigue de dos maneras distintas: *web scrapping* y servicios web de la *API*. El modo de empleo investigador extrae información mediante ambos métodos, mientras que el de profesor tan solo hace uso de la técnica de *web scrapping* debido a que para acceder a los servicios web de la *API* son necesarios ciertos requisitos que son explicados a continuación.

8.2.1. Servicios web de Moodle

Los servicios web habilitan que otros sistemas puedan entrar a *Moodle* y realizar una gran variedad de operaciones. Como ya se explica en el anexo [A.-Configuración Moodle], el uso de estos servicios suele requerir ciertos permisos que en la mayoría de los casos tan solo dispone

un usuario administrador de *Moodle*. A consecuencia de esto, este mecanismo de extracción de datos tan solo se emplea en el modo investigador (el cual está pensado para ser ejecutado por los Técnicos de la Escuela de Informática que poseen las credenciales de usuario administrador del *Moodle* asociado a dicha Escuela). Las credenciales necesarias para acceder a estos servicios son el token de acceso (vinculado a un usuario administrador) y la URI del *Moodle*.

Las peticiones de datos se puede realizar bajo distintos protocolos (e.g. *SOAP*, *REST*, etcétera) aunque finalmente tan solo se ha implementado *REST*. Éste se define por ser una interfaz para conectar varios sistemas basados en el protocolo *HTTP* y nos sirve para obtener y generar datos y operaciones, devolviendo esos datos en formatos muy específicos como *XML* y *JSON*. En esta ocasión se caracteriza por devolver la respuesta en un formato *XML*. A continuación, dicha respuesta se transforma a un diccionario complejo (normalmente compuesto por varios niveles y listas) y de ahí se obtienen finalmente los datos. Las bibliotecas de *Python* fundamentales para realizar estas peticiones y la conversión de *XML* a un diccionario son *requests* y *xmldict*, respectivamente.

A la hora de emplear los servicios web es fundamental analizar la documentación de la *API* de *Moodle* para conocer cómo debe construirse la URI de petición, los parámetros (posibles o incluso necesarios) y la estructura de la respuesta. Esta memoria no va a profundizar en estos aspectos para cada uno de los servicios web empleados puesto que ya se dispone de esa información en la propia documentación de la *API* (en el anexo [A.-Configuración Moodle] se indica cómo acceder a ella). No obstante, a continuación se listan todos los servicios web implementados seguidos de una breve descripción:

- **core_course_get_courses:**
Sirve para obtener información detallada sobre todas las asignaturas asociadas al usuario que ha realizado la petición.
- **core_course_create_courses:**
Este servicio no se usa en la aplicación, sin embargo, se encuentra implementado y sirve para crear una asignatura nueva. Se ha implementado para mostrar que las funcionalidades de los servicios web son bastante amplias permitiendo no solo extraer datos, sino crear cursos, usuarios, etcétera.
- **core_course_get_course_module:**
Permite obtener información sobre los módulos (e.g. foros, wikis, etcétera) pertenecientes a una asignatura determinada.
- **core_course_get_contents:**
Sirve para extraer información asociada al contenido, por ejemplo los recursos, de una asignatura en concreto.
- **core_user_get_users:**
Permite obtener todos los usuarios asociados al *Moodle* que se ha accedido, disponiendo de información detallada sobre los mismos.
- **core_enrol_get_enrolled_users:**
Permite obtener los usuarios, con información detallada de ellos, que están enrolados en una asignatura concreta.

- **core_user_create_users:**
Este servicio tampoco se emplea en la aplicación aunque sí está implementado, sirve para crear usuarios de forma remota sin acceder a la interfaz de *Moodle*.
- **mod_forum_get_forums_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los foros que contiene una asignatura en concreto, en caso de no disponer de foros la respuesta de la petición se encontrará vacía.
- **mod_label_get_labels_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los labels que contiene una asignatura en concreto, en caso de no disponer de labels la respuesta de la petición se encontrará vacía.
- **mod_workshop_get_workshops_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los workshops que contiene una asignatura en concreto, en caso de no disponer de workshops la respuesta de la petición se encontrará vacía.
- **mod_wiki_get_wikis_by_courses:**
Sirve para obtener información detallada asociada a cada una de las wikis que contiene una asignatura en concreto, en caso de no disponer de wikis la respuesta de la petición se encontrará vacía.
- **mod_choice_get_choices_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los choices que contiene una asignatura en concreto, en caso de no disponer de choices la respuesta de la petición se encontrará vacía.
- **mod_data_get_databases_by_courses:**
Sirve para obtener información detallada asociada a cada una de las bases de datos que contiene una asignatura en concreto.
- **mod_folder_get_folders_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los directorios que contiene una asignatura en concreto, en caso de no disponer de carpetas la respuesta de la petición se encontrará vacía.
- **mod_glossary_get_glossaries_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los glosarios que contiene una asignatura en concreto, en caso de no disponer de ninguno la respuesta de la petición se encontrará vacía.
- **mod_quiz_get_quizzes_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los quizz que contiene una asignatura en concreto, en caso de no disponer de quizz la respuesta de la petición se encontrará vacía.
- **mod_url_get_urls_by_courses:**
Sirve para obtener información detallada asociada a cada uno de los enlaces que contiene

una asignatura en concreto, en caso de no disponer de links la respuesta de la petición se encontrará vacía.

8.2.2. Web scraping

Web scraping es una técnica utilizada para extraer información de sitios web de forma automática a través de programas software o lenguajes de programación. Normalmente, estos programas simulan la navegación de un humano en la web utilizando el protocolo *HTTP*. Como bien se ha comentado, este mecanismo se emplea en ambos modos de uso de la aplicación. La idea es acceder a la plataforma web de *Moodle*, de forma automática, para extraer la información deseada.

La página web de *Moodle* se trata de un sistema en el que es necesario loguearse, por lo que se precisa del nombre de usuario y la contraseña para acceder a dicha plataforma, además de la URL donde se realiza la identificación con la introducción de credenciales. Una vez se dispongan de estas credenciales, mediante *web scraping*, tan solo queda explorar la estructura *HTML* analizando su jerarquía para acceder automáticamente a los datos que se quieren descargar.

Estudiando distintas versiones del sistema de aprendizaje *Moodle*, se descubre que las URLs de enlace son las que presentan un formato más robusto cuando se cambia de una versión a otra. Es decir, se modifica con más frecuencia la forma para acceder a una sección (cambiando la interfaz) que la URL destino a la que se va a acceder. Poniendo un ejemplo para hacerlo más entendible, es más habitual que, para una determinada versión, acceder a una zona (cuyo link es X) de la plataforma estuviese implementado mediante la acción de clicar un botón; mientras que para otra versión posterior el acceder a dicha zona (cuyo link sigue manteniéndose como X) ya no se implementa mediante un botón sino a través de otro mecanismo de entrada. Este hecho supone que en la exploración de las jerarquías *HTML* se buscase directamente las URL destino a las que se quería acceder; propiciando que finalmente se desempeñase un estudio exhaustivo de cómo se organiza la estructura de URLs a lo largo de la plataforma de *Moodle*. Tras conseguir acceder a la página web donde se encuentran los datos deseados, tan solo queda descargarlos y, en caso de modo investigador, anonimizarlos.

Mediante esta técnica se extrae información sobre los logs y las notas de los usuarios asociados a una asignatura determinada. A continuación, se muestran las distintas URLs que son exploradas y, por lo tanto, necesarias para el correcto funcionamiento de la aplicación:

- **<https://aulas.inf.uva.es/login/index.php>**

Se trata de la página de login para acceder a la plataforma de *Moodle*. A partir de este link, el cual se pide en el inicio de la aplicación, se construyen los siguientes. Aquí es preciso buscar los forms que contienen las entradas de texto para introducir el nombre de usuario y contraseña, y posteriormente clicar el botón submit para iniciar sesión.

- **<http://virtual.lab.inf.uva.es/my/>**

Corresponde con la página inicial de la plataforma. En caso de acceder en modo profesor se emplea este link para obtener los nombres de las distintas asignaturas a las que está vinculado el usuario. Por contra, en modo investigador, ya que se requieren credenciales de administrador, esto no es necesario, puesto que las asignaturas se muestran directamente en la URL donde se pueden ver los logs (explicada a continuación).

- **<http://virtual.lab.inf.uva.es/grade/report/grader/index.php?id=3>**
Se trata de la zona donde se pueden visualizar las notas de los alumnos de una asignatura cuyo identificador es el número 3. De aquí, se extrae información del nombre, rol e identificador de los alumnos enrolados en el curso.
- **<http://virtual.lab.inf.uva.es/grade/export/xml/index.php?id=3>**
Se trata de la URL donde se descargan los datos (en formato XML) de las notas obtenidas por los alumnos en la asignatura de identificador 3.
- **<http://virtual.lab.inf.uva.es/report/log/index.php?id=3>**
Este link sirve para acceder a la zona que permite ver las distintas interacciones realizadas en la asignatura con identificador 3. Aquí, mediante *query strings*, es posible aplicar filtros en la obtención de logs (e.g. por usuario, módulo afectado, día de la interacción, tipo de acción, etcétera). El *query string* tiene la siguiente forma, siendo el nombre de los parámetros constante y variando su valor:
`'chooselog=1&showusers=0&showcourses=0&id=3&user=&date=&modid=&modaction=&origin=&edulevel=-1&logreader=logstore_standard'`

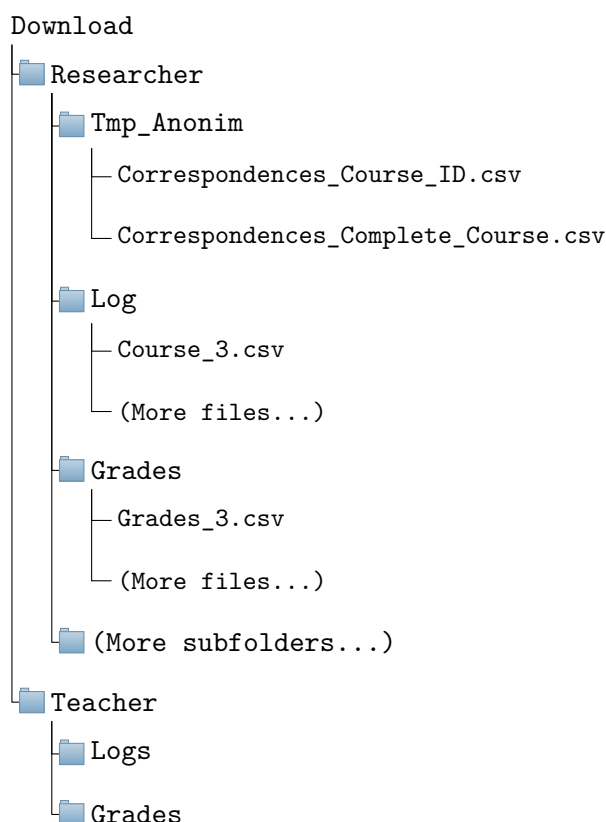
8.3. Descarga de datos

Salvando la diferencia de la anonimización, que se realiza solo para el modo asociado a la investigación, la cual se explica en la siguiente subsección, la descarga de datos se desarrolla de la misma forma para ambos modos de empleo. Consiste en un directorio dinámico llamado ‘Download’ que se creará, en caso de que no exista, en el directorio actual donde se encuentra la aplicación cuando se descargue información. Esta carpeta representa el nodo raíz, a partir de la cual se irán creando subcarpetas, en caso de que sea necesario, donde finalmente se almacenen los ficheros de datos. La primera bifurcación de carpetas corresponde con la separación entre modos de empleo, dando lugar a las carpetas ‘Researcher’ y ‘Teacher’. A partir de aquí se crean directorios según el tipo de datos que contengan, y toda la información almacenada se guarda con el mismo formato *csv*, siendo el separador el ‘;’.

Respecto a los ficheros *csv*, se les asigna un nombre en función de los datos que contengan y se almacenan dentro del directorio correspondiente (es decir, un fichero que contenga datos sobre un foro se encontrará dentro del directorio ‘Forum’). En la mayoría de los casos, los ficheros se nombran mediante una cadena que comienza con el tipo del módulo que se trata (e.g. foro, curso, etcétera) y continúa con el identificador y/o nombre del mismo o alguna característica distintiva. A modo de ejemplo se va a mostrar cómo puede llamarse un fichero que contenga información sobre los logs de una asignatura determinada, teniendo en cuenta que este tipo de contenido puede filtrarse en la aplicación por ID de usuario, ID de módulo, tipo de acción y tipo de evento. Entonces, siguiendo con el ejemplo, para un curso con ID *x*, filtrado solo para el usuario con ID *y*, cuyas interacciones afectasen al módulo *z* y se tratasen de acciones que supusiesen la *creación* de algo, el nombre del fichero sería así de explícito:

Logs_Course_x_User_y_Module_z_Action_Create.csv

Volviendo a la organización de directorios, el mapa del sistema de ficheros para las descargas de datos se muestra en el siguiente diagrama:



Del anterior diagrama se observa que ambas carpetas ('Teacher' y 'Researcher') tan solo coinciden en las subcarpetas de 'Logs' y 'Grades' (son las únicas existentes en 'Teacher') cuyos ficheros disponen además del mismo formato de nombre. El resto de la jerarquía difiere destacando el directorio 'Tmp_Anonim'. Este contiene las correspondencias necesarias para invertir el proceso de anonimización. El fichero 'Correspondences_Course_ID' contiene la asociación entre el identificador y el nombre de la asignatura; mientras que el archivo 'Correspondences_Complete_Course' almacena las correspondencias entre el ID de *Moodle*, el nombre de usuario, el email de usuario y el identificador totalmente anonimizado. Aunque no esté relacionado con este TFG, resulta categórico matizar que 'Tmp_Anonim' se trata de un directorio temporal ya que la información anonimizada extraída será integrada (mediante el email de la UVa) con otros datos aportados por la Universidad de Valladolid formando parte de un proyecto de investigación mayor; destruyéndose dicha carpeta temporal tras la unificación de los datos (haciendo imposible el proceso de desanonimización).

Otro aspecto a remarcar es que existen más subcarpetas que las mostradas para el directorio 'Researcher', como por ejemplo: 'Wikis' (almacenan información sobre las wikis de la asignatura), 'Forums' (guardan datos sobre los foros del curso) o 'Users' (esta carpeta tan solo representa un resguardo que contiene toda la información detallada sobre los usuarios por si se diese el ca-

so de tener problemas con los ficheros de ‘Tmp_Anonim’, por lo que esta carpeta también es temporal y se deberá borrar cuando llegue el momento), entre otras.

La mayoría de ficheros *csv* creados gozan de la misma composición de dos columnas ‘Característica’ y ‘Valor’. Sin embargo, la información asociada a las notas y a los logs disponen de una estructura distinta. En los ficheros que almacenan los logs, cada fila corresponde con una interacción, y las columnas almacenan datos asociados a dicha interacción, a saber:

- Fecha de realización de la interacción.
- Nombre del usuario responsable de la interacción; en caso de estar en modo investigador se dispondrá anonimizado.
- Usuario afectado por la interacción; también estará anonimizado si se está empleando el modo investigador.
- Contexto del evento: indica a qué afecta la interacción (e.g. al curso, a un fichero, etcétera).
- Componente vinculado a la interacción.
- Nombre del evento.
- Una breve descripción sobre la interacción realizada.
- Origen de la interacción, es decir, si se ha realizado vía web, servicios web, etcétera.
- Dirección IP de donde se ha realizado la interacción; en modo investigador es eliminada.
- Rol del usuario que ha realizado la interacción, pudiendo ser alumno o profesor.
- El ID proporcionado por *Moodle*, sirve para realizar correspondencias con otros datos. Tan solo se mantiene en el modo profesor, para investigador se elimina.

Los ficheros que almacenan información sobre las notas de una asignatura están estructurados correspondiendo cada fila a un alumno (cuyo nombre es anonimizado en modo investigador) y dispondrá de tantas columnas como exámenes/entregas tenga la asignatura, además de mostrar la nota final de la asignatura (en caso de que disponga de ella).

8.4. Anonimización de datos

El procedimiento de anonimización se basa en el análisis y estudio de los datos extraídos para encontrar aquellos que permitirían identificar a un alumno. Cabe destacar que esta herramienta será ejecutada por el Técnico responsable de la aplicación *Moodle* de la Escuela de Informática, Técnico que por su trabajo tiene acceso a esa información. Posteriormente, proporcionará los datos anonimizados a los investigadores que corresponda y, así, en los trabajos de investigación no se manejarán los datos originales y los individuos no serán identificables después del proceso de anonimización. Se anonimizarán los siguientes datos en el resultado final:

- **Curso:** no aparecerá el año del curso que se extrae la información. La asociación entre estos identificadores y el curso al que se corresponden se hará en la tabla temporal intermedia.

- **Identificador de usuario:** a todos los usuarios se les asigna un identificador nuevo que sea independiente del que ya poseían en *Moodle*. Este identificador se consigue partiendo de un número determinado (X) y a cada usuario se le asigna como ID dicho número más un valor aleatorio entre el 1 y el 10 (e.g. $X=X+\text{random}(1,10)$). Este valor se trata como un acumulador y, al ser sumas de cantidades aleatorias, en cada ejecución a los usuarios se les asigna un ID distinto, consiguiendo que ni el desarrollador conozca con certeza las correspondencias (éstas se encuentran almacenadas en la tabla temporal intermedia).
- **Calificaciones:** *Moodle* permite establecer la calificación máxima de cada ejercicio evaluable, pudiendo ser ésta diferente (e.g. sobre 100, sobre 10, sobre 5, etcétera). Por este motivo, se normalizarán todas las calificaciones a un rango común entre 0 y 1, dividiendo cada calificación por la máxima obtenida en esa actividad. Una vez anonimizada es imposible conocer la calificación real, ya que no es posible darle la vuelta atrás al proceso, al desconocer, una vez normalizada, la cantidad por la que se dividió cada calificación.
- **Identificador de asignatura:** se elimina el nombre de la asignatura de los datos y se deja un identificador anónimo; las correspondencias se guardarán en la tabla temporal intermedia.
- **Recursos y módulos:** los nombres de recursos y módulos, que en algunas ocasiones pueden llegar a ser bastante distintivos, simplemente serán eliminados de los datos. No se modificarán sus identificadores de *Moodle* por la imposibilidad de realizar una asociación.
- **Fechas:** un tipo de datos extraídos y de bastante relevancia son las fechas (por ejemplo de las interacciones y logs). Por este motivo, se anonimizan (puesto que pueden ser utilizadas para invertir el proceso de anonimización) tratando de perder la menor cantidad posible de información. El proceso se basa en disponer las fechas en orden cronológico y etiquetar la primera como ‘Día 1’, a partir de ahí se van etiquetando el resto de días. Una vez anonimizada es imposible revertir el proceso puesto que se desconoce el año/mes/día de cada una de las fechas y tampoco se sabe cuál es la primera fecha o la última. De esta forma se pierde conocimiento de los meses y años pero en un estudio se podría analizar perfectamente la existencia de estacionalidad. Además, en el caso de que la fecha vaya acompañada de hora (h/min/seg), se obviarán los minutos y segundos para imposibilitar la identificación de un registro ocurrido a una hora, minuto y segundo determinado.

8.5. Integración con *App2-Dash*

Tras haber ejecutado la aplicación en modo profesor, existe la posibilidad de visualizar un *dashboard* de los datos descargados. Este *dashboard* se encuentra en la aplicación *App2-Dash* cuya documentación técnica se muestra en [III.-Documentación técnica: Aplicación *dashboard*]. En este subapartado se explica cómo se ha implementado la integración de ambas aplicaciones consiguiendo así que ambas se puedan ejecutar de forma continua, sin necesidad de que el usuario realice ninguna acción adicional mejorando así la usabilidad y el dinamismo.

Para conseguir esto, desde los scripts en *Python* de la aplicación *App1-Moodle*, se ejecuta un comando en la terminal para que ejecute un archivo auxiliar en *R* (llamado *integration*). El comando en cuestión es el siguiente:

```
'C:/Route/Rscript.exe' -vanilla integration.R arg1 arg2
```

Del anterior comando, en la primera parte se indica la ruta al ejecutable 'Rscript'. En *Windows*, esta ruta es 'C:/Program Files/R/R-3.5.0/bin/Rscript.exe', donde lo único que puede variar es la versión de *R* que se esté empleado (3.5.0), aunque esto no es un problema pues esta información se obtiene previamente en el *launcher* que se ha explicado antes. A continuación, se emplea el argumento '-vanilla' que sirve para indicar que el fichero *R* que se va a ejecutar requiere argumentos, posteriormente se indica el nombre del archivo *R* y los argumentos que se le pasa (en este caso corresponden con las rutas de los datos de los logs y las notas).

A partir del fichero *integration.R*, se comprueba si los paquetes necesarios están instalados (instalándolos en caso negativo), y se lanza la aplicación *App2-Dash* en el navegador *Chrome* en la ruta 127.0.0.1:5001.

Capítulo 9

Pruebas

Esta sección muestra las distintas pruebas que se han realizado para demostrar el correcto funcionamiento de la aplicación *App1-Moodle* y los procedimientos desarrollados. En todos los casos se explica el proceso llevado a cabo indicando las dificultades que ha habido, cómo se han solucionado y si los resultados finales han sido positivos.

9.1. P1 - Anonimización

Como bien se ha remarcado a lo largo de esta documentación, el tema de la anonimización de los datos adquiere una relevancia importante. El motivo principal son los problemas legales que podría implicar la inversión de este proceso que supondría la revelación de datos personales de los alumnos. Antes de comprobar la eficiencia de la anonimización, es importante remarcar que se trata de datos académicos sin ningún tipo de valor económico, es decir, sería extraño que un ‘hacker’ dedicase sus esfuerzos en desanonimizar dicha información. No obstante, pese a que es evidente la falta de interés lucrativo que pueden tener estos datos, no se ha escatimado en el empleo de técnicas de anonimización y los esfuerzos asociados a aplicarlas.

En primer lugar, se va a suponer que se dispone de los datos descargados anonimizados. A partir de aquí, tan solo se conoce que esa información pertenece a la Universidad de Valladolid, más concretamente a una asignatura del Grado en Informática. Sin embargo, no se puede avanzar más en el conocimiento sobre los datos puesto que ni se conoce el nombre de la asignatura, ni se sabe el año en el que se cursó, tampoco aparecen los nombres de alumnos, profesores o recursos (ficheros, foros, etcétera) asociados al curso... Llevándolo a una situación extrema, supongamos que un hacker obtiene las credenciales de *Moodle* consiguiendo acceder a la plataforma y, por lo tanto, ver los datos sin anonimizar. Pese a que tuviese la información en su estado natural no tendría forma de asociarlo a los datos anonimizados puesto que incluso las fechas están modificadas para que no sea posible realizar esa conexión. Teniendo en cuenta la gran cantidad de años de la Escuela de Informática y el amplio número de asignaturas, averiguar el año y la asignatura mediante un algoritmo de fuerza bruta sería inmanejable.

9.2. P2 - Modo profesor

Esta segunda prueba realizada busca asegurar el correcto funcionamiento del modo de empleo ‘Profesor’ de la aplicación *App1-Moodle*. Llegados al punto de que la implementación funcionaba perfectamente con asignaturas de prueba en un *Moodle* instalado en una máquina virtual, el principal objetivo se convertía en probarlo en una de ámbito académico real. Para llevarlo a cabo, se necesitaron las credenciales de los tutores del TFG, por lo que se realizó la prueba en un ordenador distinto al que se había utilizado para desarrollar la aplicación, aunque con el mismo SO. Dicho ordenador carecía de *Python* y *R* instalados, por ende fue necesario instalar todas las bibliotecas y paquetes requeridos por la herramienta. Este hecho propició darse cuenta de la necesidad de un archivo *batch* que desempeñase de forma automática, si así se requería, este trabajo.

Fueron necesarias varias reuniones para encontrar pequeñas variaciones y fallos que impedían obtener resultados exitosos en las pruebas. No obstante, tras las modificaciones adecuadas, se realizaron con éxito las pruebas necesarias para garantizar el correcto funcionamiento de esta parte de la herramienta. Además, como la aplicación *App2-Dash* está asociada a este modo de empleo (profesor) y se ejecuta de forma continua, se aprovechó para probarla también. En esta ocasión, se obtuvieron de forma más rápida resultados satisfactorios garantizando así su correcto funcionamiento.

A lo largo de esta prueba se encontraron los típicos problemas de integración de datos, como por ejemplo, el almacenamiento de la misma información con distinto formato según el *Moodle* que se emplease para extraer datos. Esto se solucionó empleando los identificadores proporcionados por *Moodle* con el fin de integrar datos procedentes de distintas fuentes. Además, se identificó un error debido a que los tests realizados en la plataforma de prueba se hicieron bajo un usuario de *Moodle* que tenía el rol de administrador, sin caer en la cuenta de que la interfaz del sistema variaba al acceder con un rol de profesor. En suma, se encontró un *bug* de *Moodle* que consiste en que donde se almacena la información sobre los logs de un curso, se encuentran también las interacciones de toda la historia de la asignatura, es decir, no hay posibilidad de aplicar filtros para obtener solo los logs del curso actual sino que por defecto y única posibilidad existente es obtener todas las interacciones desde la creación de esa asignatura. Este hecho se solucionó aplicando el filtro dentro de la misma herramienta, pero dicho *bug* afecta significativamente a la eficiencia de la aplicación.

Respecto a la aplicación *App2-Dash*, se descubrió que para instalar paquetes de *R* era necesario ejecutar el script como administrador del sistema para disponer de permisos de escritura, en caso contrario daría lugar a un error. Otros fallos encontrados fueron el cambio de formatos, por ejemplo en fechas, o el cambio de idioma (inglés en el *Moodle* empleado en la máquina virtual y castellano en el *Moodle* de los tutores) que fueron solucionados con facilidad.

9.3. P3 - Modo investigador

La última prueba realizada corresponde con el modo investigador implementado en la aplicación *App1-Moodle*. Este modo, que extrae información mediante los servicios web de la API de *Moodle*, requiere permisos de administrador en dicha plataforma. Debido a esto, el hecho de

probar la herramienta en el *Moodle* real de la Escuela de Informática requiere la intervención de los técnicos de la misma, pues son quienes poseen las credenciales con los permisos necesarios. Este ha sido el principal inconveniente encontrado.

Naturalmente, los técnicos no disponen de horario y disponibilidad completo para atender este proyecto, haciéndose, por lo tanto, evidente la escasa eficiencia del modo en que había que probar la herramienta. En suma, debido a la situación actual provocada por el *Covid-19*, los técnicos han estado desbordados de trabajo hasta casi el final del curso, por lo que la primera vez que se quedó para probar la aplicación estaba ya muy próxima la fecha de entrega del proyecto. No obstante, todo el modo investigador funcionó correctamente a excepción de un único servicio web, llamado ‘core_user_get_users’, que aún no se ha encontrado el motivo del por qué daba una excepción de control de acceso. Sin haber tenido tiempo para examinar dicha excepción, es muy posible que se trate de un problema menor asociado a la modificación, al variar de versión de *Moodle*, de los parámetros o estructura de la petición REST.

Parte III

Documentación técnica: Aplicación dashboard

Capítulo 10

Introducción

En esta parte se documenta toda la información técnica asociada a la segunda aplicación desarrollada durante el proyecto, la cual da soporte a un *dashboard*. El identificador de dicho producto es *App2-Dash* y cabe recordar que tan solo está disponible para el modo profesor.

10.1. Objetivos y motivación

Como previamente se ha expuesto, la educación de la población representa una llave clave para el movimiento y prosperidad de una sociedad. Por este motivo, se pretende estudiar y analizar datos académicos que nos permitan aportar claridad o proponer mejoras a los actuales métodos y procesos de enseñanza.

El principal objetivo es ayudar a los profesores a monitorizar y analizar sus asignaturas, además de proporcionarles una herramienta que les facilite la docencia. En definitiva, el fin último es extraer conclusiones a partir de datos académicos universitarios, mediante algoritmos y procedimientos estadísticos, que ayuden tanto a docentes como alumnos en los distintos procesos de aprendizaje, tratando de conseguir que la educación sea óptima.

Capítulo 11

Análisis

En esta sección se van a describir los requisitos del sistema y la descripción de los datos usados por la aplicación desarrollada. A continuación, dentro de los requisitos, se describen tanto los funcionales (incluyendo los de información e interacción) como los no funcionales.

11.1. Requisitos funcionales

Los requisitos funcionales asociados a la aplicación *App2-Dash* son los siguientes:

1. El sistema deberá permitir representar distintos gráficos y procedimientos estadísticos.
2. El sistema deberá permitir introducir archivos de datos que contengan información sobre *Moodle* para, así, realizar análisis estadísticos.
3. El sistema deberá permitir navegar por la interfaz mediante *tabs* para observar los resultados.
4. El sistema deberá permitir al usuario realizar selecciones para obtener nuevos análisis (e.g. escoger los usuarios a ser estudiados, seleccionar entre profesores o alumnos, etcétera).

11.1.1. Requisitos funcionales de información

En lo que respecta a requisitos funcionales de información, para *App2-Dash*, el sistema tan solo tendrá que almacenar la información relativa a las selecciones que vaya realizando el usuario mediante *sliders*, *selectInput*...para así guiar los nuevos análisis estadísticos.

11.1.2. Requisitos funcionales de interacción

A diferencia de *App1-Moodle*, esta aplicación carece de casos de uso complejos. Esto es debido a que se trata de un conjunto de algoritmos y procedimientos estadísticos ejecutados en serie, en los que el usuario, como mucho, participa en la entrada de algún *button*, *slider* o *selectInput*. La participación por parte del usuario se limita a la navegación y exploración del *dashboard* a través de las distintas pestañas y layouts existentes. El usuario puede:

- Seleccionar el cuadro de mando (mediante pestañas).

- Seleccionar el rol de los usuarios (mediante *radio buttons*).
- Seleccionar un usuario en concreto (mediante un *selectInput*).
- Seleccionar un rango de notas (mediante un *slider*).
- Cerrar la aplicación (mediante un botón).

11.2. Requisitos no funcionales

App2-Dash presenta los siguientes requisitos no funcionales:

1. Los datos introducidos deberán corresponder con un fichero *csv* separado por punto y coma (‘;’).
2. La interfaz desarrollada permitirá una fácil navegación entre las distintas vistas de la aplicación.
3. La instalación del sistema podrá ser realizada en menos de diez minutos.
4. Las funcionalidades del sistema podrán ser aprendidas, por un usuario sin previa experiencia, en menos de veinte minutos, alcanzando un nivel avanzado de empleo.
5. El tiempo de respuesta para la obtención de las gráficas y procedimientos estadísticos no sobrepasará los 60 segundos.
6. El porcentaje de fallos será menor al 1 % de los usos del sistema por un usuario estándar.
7. El mantenimiento del sistema se podrá realizar de forma eficiente con tan solo haber leído la documentación aquí mostrada.
8. Las selecciones que podrá realizar el usuario se basarán en entradas de *radio buttons*, *sliders*, *buttons* y *selectInput*.
9. Los resultados se visualizarán en gráficos con diferentes layouts separados por pestañas.

11.3. Atributos de calidad

En este apartado se examinan los indicadores de calidad [25] asociados a la aplicación *App2-Dash*, entre los que destacan los siguientes:

- **Rendimiento:** al igual que en la anterior aplicación, es remarcable la eficiencia en el uso de memoria y en el breve tiempo de espera necesario para visualizar los gráficos y resultados. Además, incluso cuando interactúa el usuario, el tiempo de ejecución es mínimo.
- **Usabilidad y simplicidad:** *App2-Dash* destaca por su sencillez y usabilidad, en definitiva, resulta un aspecto fundamental en los *dashboard*. Es necesario que sea intuitivo y de fácil exploración.

- **Extensibilidad:** una característica importante de *App2-Dash* es la facilidad para agregar nuevos requisitos. Es decir, existe la posibilidad, de forma sencilla, de incluir nuevas pestañas con diferentes layouts y procedimientos estadísticos que permitan explorar y analizar en mayor profundidad el problema abordado.
- **Escalabilidad:** aunque no sea de las características más fuertes de la aplicación, sí permitiría aumentar considerablemente la carga de trabajo (e.g con mayores cantidades de datos a procesar) sin apenas afectar el rendimiento.
- **Robustez:** por la naturaleza del *dashboard*, la durabilidad del sistema funcionando correctamente es larga. Además, la fiabilidad y estabilidad ante posibles incidencias es alta, teniendo la capacidad de introducir modificaciones sin producir errores.
- **Completitud:** el sistema es capaz de realizar todas las operaciones definidas en los requisitos funcionales.
- **Correctitud y consistencia:** se consiguen mediante la ausencia de errores y la coherencia entre todas las operaciones que puede realizar el usuario.

Capítulo 12

Planificación de la visualización

Toda visualización requiere de una planificación en la que se especifique la función para la que se ha concebido (explicativo o exploratorio), el tono con el que quiere presentarse (pragmático o emotivo) y el efecto deseado que busca conseguir en el usuario.

12.1. Función

La función de este *dashboard* es principalmente explicativa puesto que la información expuesta es conocida a priori, es decir, los logs, notas... se pueden observar en la misma plataforma de *Moodle* (aunque no de una forma compacta que permita un análisis). No obstante, también tiene una gran parte exploratoria puesto que permite extraer conocimiento que se desconocía como por ejemplo la asociación entre la evolución académica del alumno y el número de interacciones, entre otras muchas conclusiones que se puedan obtener.

12.2. Tono

El *dashboard* dispone de un tono claramente pragmático puesto que su finalidad es analizar los datos académicos y extraer conclusiones al respecto. En ningún momento busca apelar a los sentimientos del usuario, sino más bien aportar claridad a la gran cantidad de información existente oculta tras los procedimientos educativos de la enseñanza universitaria.

12.3. Efecto deseado

El efecto deseado que se quiere conseguir con estas visualizaciones integradas en el *dashboard* se resume en los siguientes puntos:

- Reflexionar sobre la herramienta educativa *Moodle*.
- Generar preguntas sobre cómo dirigir los procesos educativos disponiendo de esta información.
- Generar preguntas sobre cómo administrar la inmensa cantidad de información que almacena *Moodle*.

- Extrapolar nuevas asociaciones y conclusiones que permitan replantearse nuevas direcciones en las técnicas de enseñanza.
- Extrapolar nuevas asociaciones y conclusiones que permitan mejorar el desempeño tanto de profesores como alumnos.

Capítulo 13

Diseño

A continuación, se describen los datos empleados para desarrollar *App2-Dash* y además se muestran los diseños iniciales de las distintas partes del *dashboard*. Esto supone que en esta sección se expondrán los bocetos de lo que será la configuración final de la visualización, aunque pueden existir variaciones entre estos esbozos y la disposición última, la cual se enseña en el apartado [C.2.-Manual de usuario].

13.1. Descripción de los datos

Los datos empleados para realizar el *dashboard* corresponden con la información obtenida mediante la aplicación *App1-Moodle*. Los ficheros de datos tienen un formato *csv* separado por ‘;’ y almacenan datos asociados a la experiencia académica de los alumnos en una asignatura determinada.

Más detalladamente, los ficheros contienen información sobre las interacciones de los usuarios en la plataforma *Moodle*, teniéndose conocimiento sobre la fecha de interacción, el identificador de usuario (completamente anonimizado), una descripción del log, el tipo de acción realizada, tipo de fichero afectado y el rol del usuario (alumno o profesor). Además, se conoce la evolución académica de cada uno de los alumnos, es decir, se dispone de las notas que han ido obteniendo en las distintas pruebas de la asignatura (notas normalizadas y categorizadas como bien se explica en el apartado [8.4-Anonimización de datos]).

13.2. Forma de representación y enfoque

El mecanismo principal de navegación a través del *dashboard* es la existencia de distintas pestañas (*Tabs*). Estas pestañas se encuentran en la parte superior de la visualización y se mantienen visibles de forma constante durante toda la interacción. En la Figura 13.1, se puede apreciar la página inicial del *dashboard*, mostrándose tanto el título de la visualización (*Dashboard*) como dichas pestañas en la parte superior. Se trata de la *home page* (pestaña *Home*) en la que hay un *sidebar* lateral donde se encuentra un resumen del cometido de la aplicación y un botón encargado de cerrar la herramienta. Además, en el cuerpo central de la página, se desarrolla un índice en el que para cada una de las pestañas disponibles se incluye una breve descripción y se enumeran los gráficos que contiene. De esta forma se pretende poner en contexto

y situar al usuario, aparte de facilitar su navegación por las distintas *tabs*. En esta página no existe información gráfica pero se dispone de la posibilidad de realizar scroll para observar todo el contenido del índice.

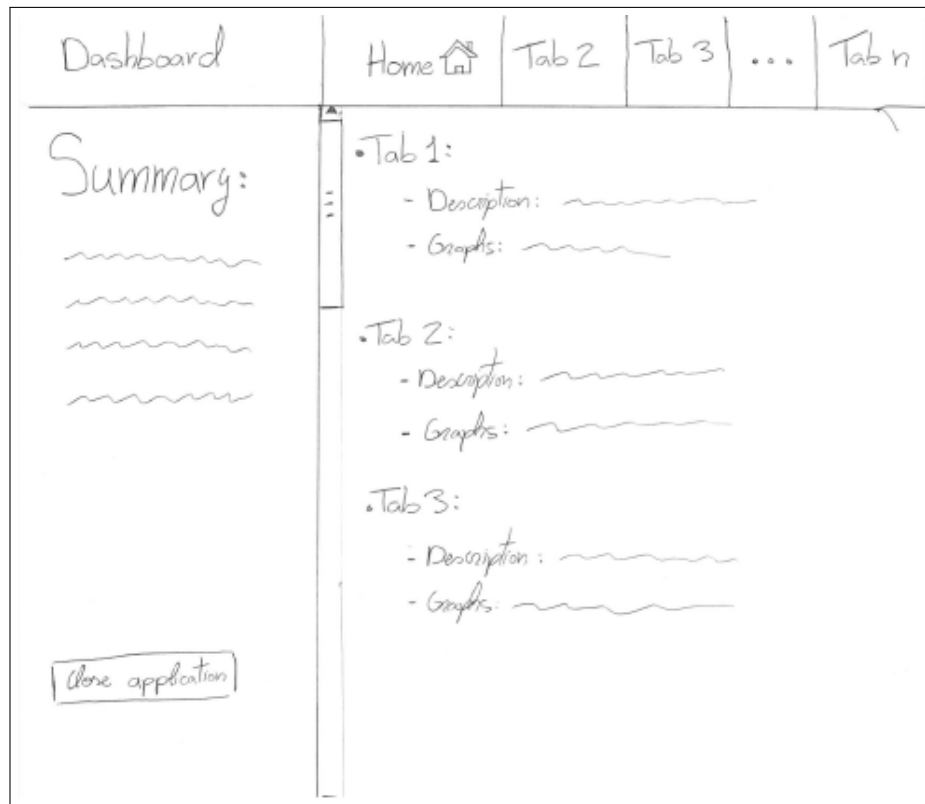


Figura 13.1: Página inicial del dashboard.

La Figura 13.2 muestra la pestaña ‘Exploratory’ en la que se pretende buscar asociaciones entre las notas obtenidas por los alumnos y el número de interacciones que han realizado en la plataforma *Moodle*. Además, se dispone de cuatro *value-box* (justo debajo de las pestañas) que muestran información general sobre la asignatura que se está analizando. Se muestra el número total de usuarios asociados a dicho curso, el total de interacciones realizadas durante el transcurso del mismo, el ratio de interacciones por usuario y la nota media de todos los alumnos. En el cuerpo principal de esta visualización se encuentra un gráfico *barchart* compuesto, en el que para cada usuario se muestra una barra mostrando su nota media y otra barra que indique el número de interacciones que ha realizado. Más detalladamente, se trata de un diagrama de barras en el que el eje X muestran los alumnos y, además, se dispone de dos ejes Y. El eje de ordenadas de la izquierda mantiene la escala de las notas mientras que el de la derecha corresponde con el número de interacciones. En suma, dicha representación se encuentra ordenada en función de las notas de los alumnos facilitando la extracción de conclusiones. Profundizando más, se aprecia en la parte izquierda un *sidebar* que contiene un *slider* como entrada con dos extremos móviles. Esto permite delimitar un rango de notas entre el 0 y el 10 para realizar el gráfico de barras y así solo mostrar aquellos usuarios cuya nota media se encuentre en dicho intervalo. También cabe

destacar que el gráfico se diseña con la posibilidad de pasar el ratón por encima de las barras y que aparezcan de manera interactiva los datos asociados al usuario (su ID, su nota media y número de interacciones realizadas).

Finalmente, además de lo mostrado por el boceto, también se implementó en el *sidebar* un *selectInput* que permita elegir un alumno determinado y se incorporó un nuevo gráfico (incluyendo una nueva pestaña con el gráfico ya existente) que mostrase las distintas notas de dicho alumno a lo largo del curso para poder analizar su evolución.

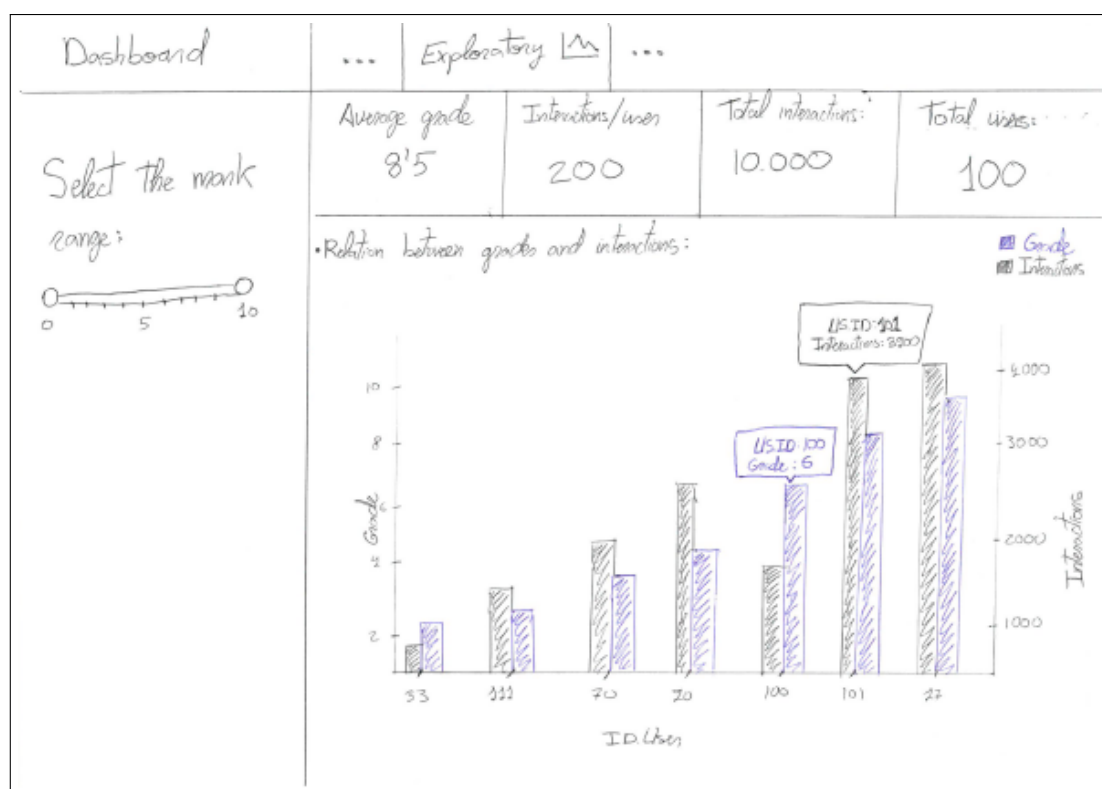


Figura 13.2: Pestaña 'Exploratory' del dashboard.

La siguiente pestaña, mostrada en la Figura 13.3, se denomina 'Time series' y sirve para visualizar la evolución temporal de las interacciones realizadas por los usuarios, tanto profesores como alumnos, en el sistema *Moodle*. Se observa cómo esta *tab* dispone de un layout que contiene a su vez tres pestañas ('General', 'By hour clustering' y 'Distribution hour clustering') mostrando cada una de ellas un gráfico distinto. Por defecto, se encuentra en la pestaña 'General' que ilustra la evolución de las interacciones mediante una serie temporal. El eje X corresponde con las fechas en las que se han realizado las interacciones y el eje Y representa el total de interacciones realizadas en un determinado día. También existe la posibilidad de deslizar el ratón por encima de los puntos para que aparezca una ventana con la información relevante asociada a dicho punto.

Además, hay un *sidebar* lateral que dispone de una entrada para seleccionar el usuario que se desea estudiar (por defecto se encuentra en la opción ‘All’). Este filtro se consigue mediante una lista que contiene todos los nombres de usuario. Otro filtro que puede ser relevante es la posibilidad de elegir el rol del usuario a ser estudiado (distinguiendo entre profesor y alumno), siendo la opción por defecto ‘All’. Este cometido se logra mediante el uso de *radio buttons*. Cabe remarcar que estas entradas afectan a las tres pestañas citadas anteriormente: ‘General’, ‘By hour clustering’ y ‘Distribution hour clustering’ que son mostradas en la Figura 13.4.

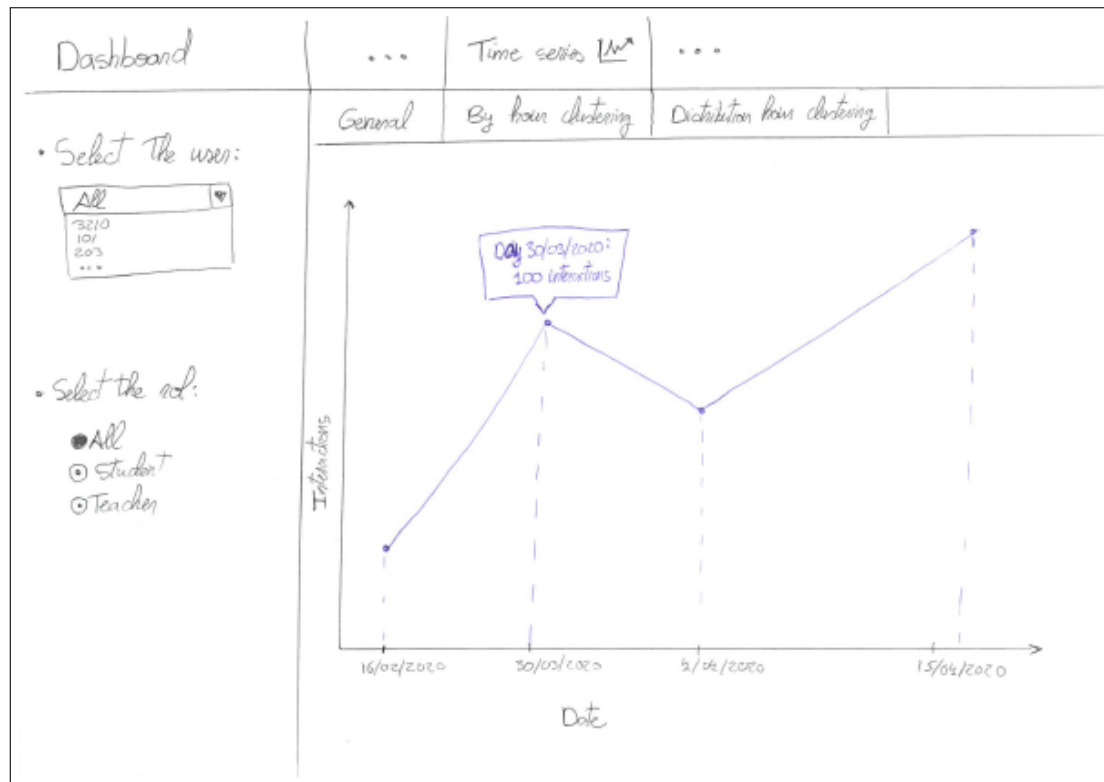


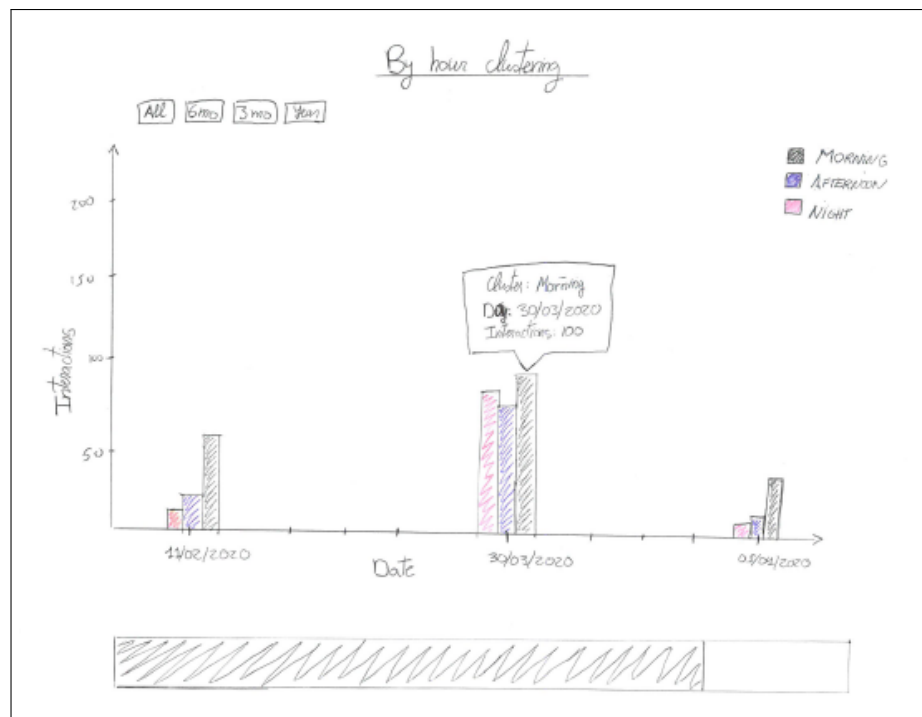
Figura 13.3: Pestaña ‘Time series’ del dashboard.

En la Figura 13.4 (a) se observa el gráfico correspondiente con la pestaña ‘By hour clustering’ en la que también se representa una serie temporal del número de interacciones. A diferencia de la representación anterior, se trata de una serie temporal dividida según clusters horarios (‘Morning’, ‘Afternoon’ y ‘Night’). Para cada día, existe una barra por cluster que representa el número de interacciones de dicho día en cada una de las franja horarias. Al deslizar el ratón por encima de la visualización aparecen ventanas con la información asociada.

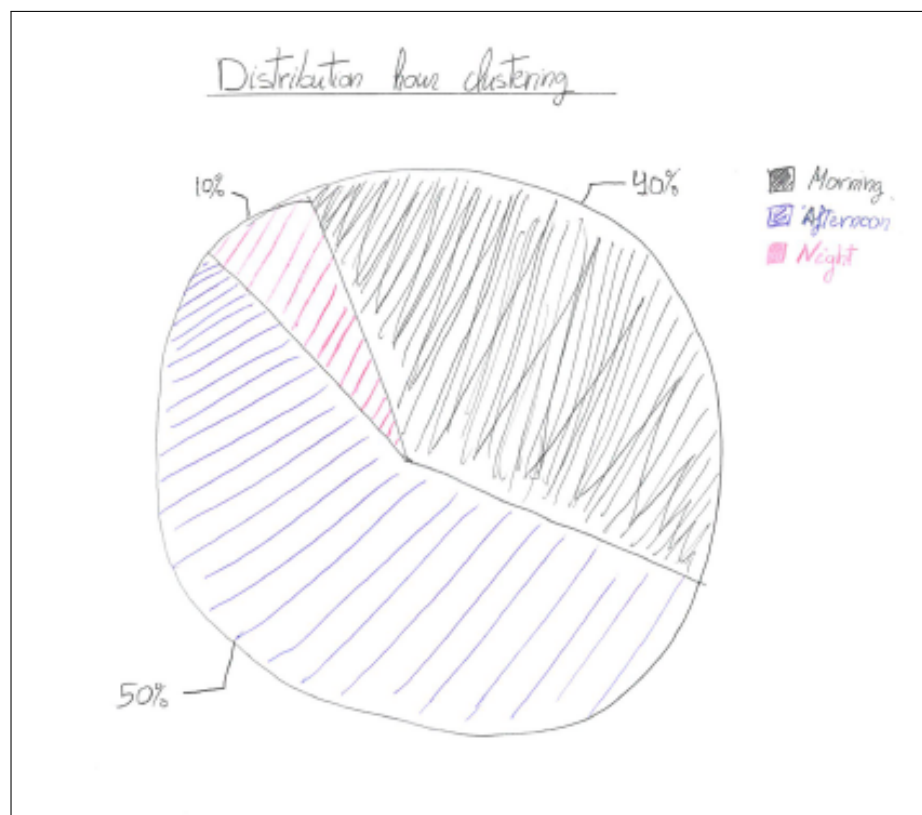
Además, se dispone de botones en la parte superior izquierda que permiten delimitar el tamaño de la ventana temporal a representar, siendo las posibilidades: toda la ventana temporal, un año, lo que se lleva de año, seis meses o tres meses. Complementando a esta funcionalidad, se encuentra un *slider* en la parte inferior que permite tanto delimitar el tamaño de la ventana temporal (de forma deliberada) como deslizar dicha ventana a lo largo de toda la serie temporal. Un ejemplo de uso sería clicar el botón de tres meses y así representar una ventana temporal de interacciones por cluster de longitud tres meses y después, mediante el *slider*, deslizar dicha ventana a lo largo de toda la secuencia temporal. Esto permite encontrar patrones temporales y horarios.

En la Figura 13.4 (b) se observa el gráfico correspondiente con la pestaña ‘Distribution hour clustering’. Inicialmente, esta representación iba a contener tan solo un *piechart* para poder apreciar, en porcentajes, cómo se distribuyen las interacciones entre los distintos clusters horarios y así observar qué franja horaria es la dominante. No obstante, finalmente se emplearon seis franjas horarias en vez de tres: ‘Early Morning’, ‘Morning’, ‘Midday’, ‘Afternoon’, ‘Evening’ y ‘Night’. Como consecuencia, el *piechart* se complementó con dos *barchart* situados cada uno en un lateral que representaban en valores absolutos el número de interacciones realizados en la primera mitad del día y en la segunda mitad. Esta diferencia de diseño puede apreciarse en la Figura C.6. Al igual que en los casos anteriores, existe la posibilidad de deslizar el ratón por encima de los gráficos para así ver ventanas con datos asociados a esa parte de la representación.

Resulta importante volver a recordar que los gráficos de las pestañas ‘General’, ‘By hour clustering’ y ‘Distribution hour clustering’ se ven afectados por los filtros de selección de usuario y rol de éste último. Gracias a todas estas herramientas citadas, se consigue analizar la existencia de patrones temporales estudiando si hay diferencia entre franjas horarias, roles o usuarios. Pudiendo relacionar estas conclusiones con la información mostrada en otras pestañas del *dashboard*.



(a) Por cluster horario



(b) Distribución de clusters

Figura 13.4: Gráficos dentro de la pestaña 'Time series'.

La Figura 13.5 representa el boceto de la pestaña ‘Events’. En esta ocasión, el layout de la página corresponde con dos partes diferenciadas: en la izquierda hay dos pestañas en las que se representa tanto un *treemap* como un *barchart* resumen del tipo de acciones realizadas en las interacciones. Mientras que en la parte de la derecha se expone un *donut chart* ilustrando los tipos de módulos/recursos que son afectados con las acciones.

El *treemap* pretende ilustrar la distribución, en valores absolutos, del tipo de acciones realizadas, siendo las categorías: ‘Download’, ‘Upload’, ‘Delete’, ‘Create’, ‘View’ y ‘Others’. Además de permitir navegar por subcategorías que muestren la acción concreta (e.g. ‘View a course’ que estaría dentro de la categoría ‘View’). En suma, se dispone de otra *tab* que almacena un diagrama de barras genérico que representa el total de acciones correspondientes a cada una de las categorías principales.

En cambio, la parte derecha del layout corresponde con un *donut chart* que representa el porcentaje de los tipos de módulos/recursos que se ven afectados por las acciones previamente expuestas (pudiendo ser ‘File’, ‘Forum’, ‘Course’, ‘Glossary’, ‘Wiki’, etcétera). En todos los gráficos se dispone de la interacción de deslizar el ratón por encima de los gráficos para obtener ventanas que expliciten la información asociada.

Finalmente, en esta pestaña también se incluyó un *sidebar* lateral con *radio buttons* para seleccionar el rol de los usuarios y un *selectInput* para poder escoger un usuario determinado (de la misma forma que se ha mostrado en la Figura 13.3).

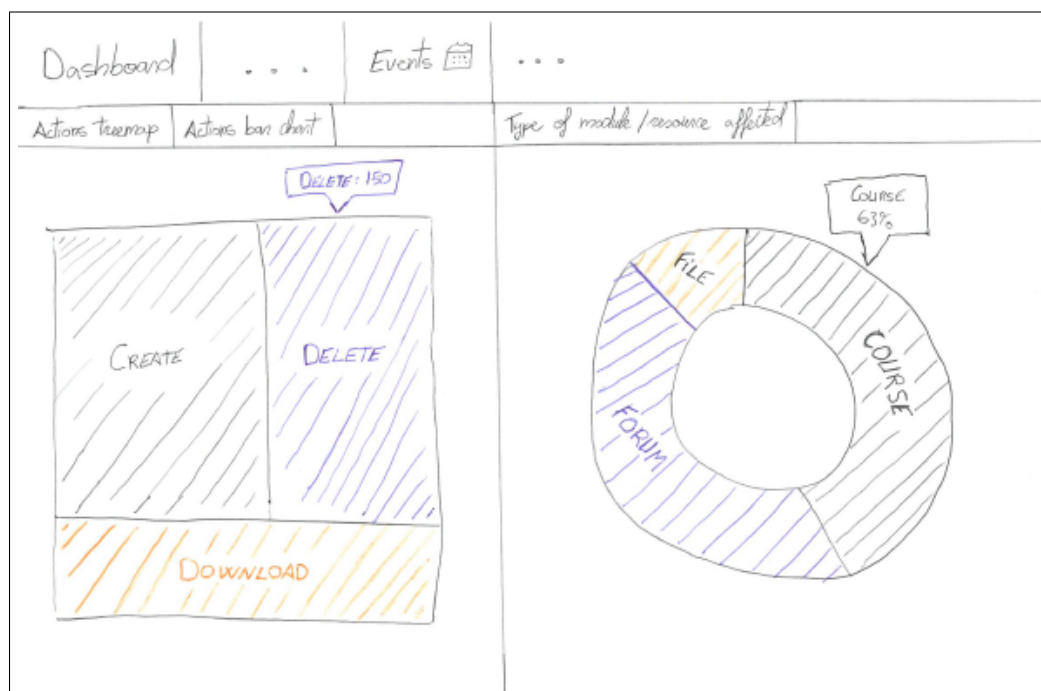


Figura 13.5: Pestaña ‘Events’ del dashboard.

Capítulo 14

Implementación

14.1. Implementación

El *dashboard* se realiza mediante el lenguaje de programación *R*, usando principalmente los paquetes *shiny* y *plotly* que facilitan el desarrollo de la aplicación. El código se divide en distintos *chunks* y la estructura se consigue mediante el uso de unas determinadas reglas de sintaxis mostradas a continuación.

- Cada una de las pestañas principales del *dashboard* (e.g. ‘Exploratory’) se definen y se distinguen unas de otras mediante la siguiente sintaxis:

Nombre pestaña {data-orientation=rows}
=====

En esta aplicación junto al nombre de la pestaña se incluye un emoticono de la página *Font Awesome* para hacer la interfaz más ilustrativa. Esta parte corresponde con lenguaje *markdown* por lo que para incluir los iconos es preciso ejecutar comandos de *R* en dicho entorno (es decir, fuera de los *chunks*). Además, en este mismo punto es donde se puede decidir el layout de la pestaña para determinar la posición de los gráficos (si la disposición tendrá una orientación por filas o por columnas). Cabe destacar que se puede introducir código *HTML* dentro del entorno de *markdown*.

- Una vez decidida la orientación, se crea la cuadrícula que contendrá los distintos gráficos. La separación de cada fila/columna se realiza de la siguiente forma:

Row/Column {alguna característica}

Aunque no se aprecie bien, se trata de guiones consecutivos y entre corchetes se puede especificar alguna característica de la fila/columna que se está definiendo (e.g. la anchura/altura de la fila/columna, si se trata de un *tabset*, etcétera). Para finalizar la implementación de la estructura, queda definir la segunda dimensión de la cuadrícula, es decir, hasta el momento se ha definido el número de filas o de columnas, falta determinar la componente restante. En el caso que la pestaña se rija mediante una orientación por filas,

para definir una columna dentro de una fila se hace mediante ‘### Nombre columna’. De este modo, el siguiente ejemplo constaría de una única pestaña con un layout de una fila y dos columnas:

```
Nombre pestaña {data-orientation=rows}
=====
Row {alguna característica}
-----

### Column 1
### Column 2
```

- En el ejemplo anterior, dentro de cada una de las columnas, se definirían los gráficos que se quisiese mostrar en dicha posición de la pestaña. El código se encuentra dentro de *chunks* y, en aquellas pestañas que se permita aplicar filtros por parte del usuario, cobran gran relevancia los *widgets* de *shiny* y el entorno reactivo.
- Además de lo citado, también se dispone de la opción de incluir en la aplicación dos pestañas por defecto en la parte superior derecha de la interfaz. Una de ellas sirve para mostrar el código con el que se ha realizado y la otra para poder compartir la aplicación a través de las redes sociales.

Una vez expuesta la implementación de la estructura general de la aplicación, se va a explicar en los siguientes subapartados el papel fundamental que adquieren cada uno de los paquetes citados.

14.1.1. Shiny

Shiny es una herramienta para crear aplicaciones web interactivas que permiten a los usuarios interactuar con los datos sin tener que manipular el código. Se basa en la programación reactiva que vincula los valores de entrada con los de salida. Además dispone de *widgets* ya definidos como *sliders*, *value-box*, *checkbox*, etcétera. Siendo estas herramientas las responsables de permitir al usuario introducir filtros a la hora de realizar los gráficos, como por ejemplo seleccionar un rol o un intervalo de notas determinado. Los *widgets* empleados son:

Widget	Función asociada
Slider	sliderInput()
Value box	valueBox()
Select list	selectInput()
Radio button	radioButtons()

Tabla 14.1: Widgets empleados.

Para trabajar con dichos *widgets* es necesario realizarlo en un entorno reactivo; para conseguirlo en *R* se utilizan las funciones *render**()¹.

14.1.2. Plotly

La librería *plotly* representa la herramienta principal para la representación de los gráficos que componen el *dashboard*. El principal motivo es que, a diferencia de otras librerías como *ggplot2*, *plotly* permite crear gráficos interactivos en vez de estáticos, consiguiendo así dinamismo en la aplicación. De esta forma se consigue explotar al máximo su potencial alcanzando un mayor grado de representación e ilustración de la cuestión que se quiere analizar.

Aquellos gráficos que se hayan implementado mediante *plotly* disponen de distintas posibilidades de interacción. En primer lugar, existe la posibilidad de ‘recortar’ directamente en el gráfico para poder focalizar la atención en una determinada parte de la representación. Además de esto, existe un menú en la parte superior derecha del gráfico en la que se proporcionan funcionalidades como el zoom, selección de una parte del gráfico, resetear y autoescalar a la situación inicial, disponer de un modo de ‘arrastre’ para desplazarse por la ilustración en caso de haber empleado el zoom, selección de variables (clicando sobre su nombre), etcétera. Existen más opciones de interacción que las aquí citadas pero todas son intuitivas y de fácil exploración por parte del usuario. Por último, cabe destacar que dependiendo del tipo de gráfico, las posibilidades de interacción variarán.

¹El asterisco significa que existen diversas funciones nombradas con el patrón *render* más el tipo de objeto que se quiere renderizar proporcionando así un contexto reactivo. Un ejemplo es *renderPlot* cuando se quiere renderizar un gráfico o *renderTable* cuando se pretende realizar lo propio con una tabla.

Parte IV

Conclusiones y trabajos futuros

Capítulo 15

Conclusiones

Al finalizar este trabajo, se puede decir que los objetivos planteados inicialmente han sido alcanzados, tanto en el enfoque de Investigación como en el de Análisis de la Enseñanza. Además de los objetivos técnicos, también se ha conseguido alcanzar las metas personales de aprendizaje de las herramientas empleadas.

A nivel personal, este proyecto me ha permitido mejorar mis habilidades y conocimientos técnicos sobre los lenguajes de programación usados (sobre todo *Python* y *R*), teniendo en cuenta que es la primera vez que desarrollaba aplicaciones web con sus respectivas interfaces.

A nivel funcional, la herramienta construida permite obtener y explotar la información contenida por el sistema *e-learning Moodle* con el objetivo final del análisis de la enseñanza. La principal ventaja que aporta es la integración, síntesis y unificación de los datos, además de la anonimización de los mismos en caso de querer disponerlos para el ámbito de investigación. Atendiendo a la investigación, *App1-Moodle* permite obtener grandes cantidades de datos procesados para ser analizados en estudios académicos. Este hecho proporcionará a los investigadores, de forma sencilla, la materia prima sobre lo que fundamentar sus análisis que, aunque sonando demasiado ambicioso, podrían desembocar en deducciones que ayuden a mejorar los procesos de enseñanza.

En cambio, una funcionalidad de la aplicación más fehaciente, es la posibilidad que se les da a los docentes universitarios para visualizar los datos de sus asignaturas. Se ha logrado desarrollar una aplicación, *App2-Dash*, que proporciona una herramienta que permite monitorizar y analizar sus asignaturas. Esto puede ayudar a los profesores en la toma de decisiones docentes, además de detectar posibles mejoras tanto en el enfoque como en los procedimientos llevados a cabo en sus asignaturas. También permite examinar qué partes de las mismas tienen mayor aceptación entre el alumnado, pudiendo inspeccionar qué recursos y módulos son los que tienen un mayor uso.

Ambas aplicaciones se han probado con las asignaturas de los tutores del TFG, por lo que, en definitiva, se trata de una herramienta extrapolable al mundo real académico. Esta aplicación busca ser usada y explotada por el resto de la comunidad educativa, pero sobre todo, pretende permitir el hecho de aumentar las funcionalidades ya implementadas. Resumiendo, el *dashboard* desarrollado, junto a la aplicación *App1-Moodle*, pueden conformar una herramienta muy potente que sirva de gran ayuda a las prácticas docentes y al análisis de la enseñanza.

En definitiva, todo el desarrollo aquí mostrado, creemos que aporta un pequeño y humilde grano de arena en pos de una mejora académica, buscando la adecuación a la progresiva telematización de los procesos educativos.

Capítulo 16

Futuras investigaciones

La aplicación *App1-Moodle* se trata de una herramienta fácilmente extensible, por lo que existen futuras líneas de trabajo. Como bien se ha comentado, *Moodle* goza de una gran cantidad de módulos que no son explotados por la comunidad docente, y por ende, existen sus respectivos servicios web que permiten extraer información de ellos. La aplicación aquí desarrollada extrae información de los logs y notas (a través de *web scraping*) y de usuarios, foros y wikis (mediante los servicios web). Complementando a esta información se podría implementar, fácilmente, la extracción de otros tipos de datos relacionados con los glosarios, quizzes, workshops, etcétera. Esto no se llegó a implementar en *App1-Moodle* porque el tiempo para desarrollar el TFG es limitado y además se quiso desarrollar también la aplicación *App2-Dash*.

La aplicación *App2-Dash* se trata de un *dashboard*, por lo que siempre se va a poder ampliar sus funcionalidades realizando representaciones o procedimientos estadísticos distintos. En definitiva, esta herramienta depende de los datos que se le proporcionen, es decir, en este contexto está supeditado a la información extraída por *App1-Moodle*. Esto supone que si se proporcionasen mayor cantidad de nuevos tipos de datos, podrían explorarse nuevas vías de representación; no obstante, incluso con los datos que se ya se emplean, se podrían desarrollar otros tipos de gráficos que aporten conocimientos distintos al docente.

Bibliografía

- [1] Carolina Costa, Helena Alvelosa, Leonor Teixeiraa. 2012. The use of Moodle e-learning platform: a study in a Portuguese University.
- [2] Paulsen, M., 2003, Experiences with Learning Management Systems in 113 European Institutions. Educational Technology & Society.
- [3] Alexander, B., 2006. Web 2.0: A new wave of innovation for teaching and learning? Educause Review.
- [4] Bremer, D., R. Bryant, 2005. A Comparison of two learning management Systems: Moodle vs Blackboard. in 18th Annual Conference of the National Advisory Committee on Computing Qualifications.
- [5] Blin, F., M. Munro, 2008. Why hasn't technology disrupted academics' teaching practices? Understanding resistance to change through the lens of activity theory. Comput. Educ., 50(2), p. 475-490.
- [6] Piotrowski, M., 2010. What is an e-learning platform?, in Learning management system technologies and software solutions for online teaching: tools and applications, I. Global, Editor.
- [7] Bob Hughes, Mike Cotterell, 2009. Software project Management, Fifth Edition. Chapter 4, 12; Chapter 3, p 61; Chapter 6, p 133; Chapter 7.
- [8] Documentación general sobre la versión 3.8 de Moodle.
Acceso: Marzo 2020.
- [9] Documentación sobre el uso de servicios web en Moodle, versión 3.8.
Acceso: Marzo 2020.
- [10] Web service FAQ: Moodle 3.8.
Acceso: Marzo 2020.
- [11] Conectart: Metodología Ágil Scrum
Acceso: Marzo 2020.
- [12] OpenWebinars: Principios de Scrum
Acceso: Marzo 2020.
- [13] Agileando: ¿Qué son los artefactos de Scrum?
Acceso: Marzo 2020.

- [14] Scrum Manager: Eventos
Acceso: Marzo 2020.
- [15] Softeng: Procesos y roles de Scrum
Acceso: Marzo 2020.
- [16] Proyectos ágiles.org: Introducción a la estimación y planificación ágil
Acceso: Marzo 2020.
- [17] Portal UNED: Estudiar en el Espacio Europeo de Educación Superior
Acceso: Marzo 2020.
- [18] Boletín Oficial del Estado: Ministerio de Trabajo y Economía Social.
Acceso: Marzo 2020.
- [19] Practicopedia: ¿Cuál es el coste real de tener un trabajador para una empresa?.
Acceso: Marzo 2020.
- [20] Docs Moodle: Acerca de Moodle.
Acceso: Marzo 2020.
- [21] Michelle Núñez Galindo, Diana Dueñas Chávez. Creación y edición de documentos con \LaTeX . Grupo de Ingeniería Lingüística Instituto de Ingeniería UNAM, 23 de octubre de 2017.
Acceso: Marzo 2020.
- [22] OpenWebinars: Qué es Flask
Acceso: Marzo 2020.
- [23] DevCode: ¿Qué es HTML?
Acceso: Marzo 2020.
- [24] Hostinger tutoriales: ¿Qué es CSS?
Acceso: Marzo 2020.
- [25] Uqbar: Atributos de calidad
Acceso: Marzo 2020.

Parte V

Apéndices

Apéndice A

Configuración Moodle

Antes de especificar las distintas configuraciones necesarias para acceder a la *API* de *Moodle* y emplear los servicios web, se remarca que esto tan solo es imprescindible para el empleo de la aplicación *App1-Moodle* en modo investigación, en caso de usar la aplicación mediante el enfoque de *learning analytics* (profesor) no es necesario.

Por lo tanto, para poder usar los servicios web de *Moodle* y llevar a cabo la extracción de una parte de los datos de la plataforma es necesario realizar una serie de configuraciones en el propio sistema. Siguiendo los pasos mostrados a continuación, se permite acceder a los servicios ya construidos por *Moodle* o a nuevos creados por el cliente y así explotar al máximo la funcionalidad de la plataforma. Toda esta información, aunque menos detallada y extensa, se puede encontrar en la página oficial de *Moodle* [8] [9]. Las configuraciones necesarias son las siguientes:

■ Habilitar servicios web

1. Administration → Site Administration → Advanced features
2. Clicar en ‘Enable web service’ que por defecto no está habilitado.
3. Clicar en ‘Save changes’.

■ Habilitar protocolos

1. Administration → Site Administration → Plugins → Web services → Manage protocols
2. Clicar en los protocolos que se quieran habilitar, en el caso aquí implementado, el protocolo *REST*.
3. Existe la posibilidad de habilitar la generación automática de la documentación de los servicios web. Esto es útil en aquellos casos en los que se crean servicios web clientes.
4. Clicar en ‘Save changes’.

■ Creación de un servicio

En el caso de que los servicios web ya definidos en *Moodle* no cubran las necesidades del cliente se pueden crear servicios web nuevos de la siguiente manera.

1. Administration → Site Administration → Plugins → Web services → External services.
2. Clicar en ‘Add new custom service’.
3. Llegados a este punto se debe indicar qué usuarios están autorizados para emplear dicho servicio web, si se requiere algún permiso en especial o si se pueden descargar o subir archivos.
4. Introducir un nombre para el servicio web y clicar en ‘Enabled’ para así habilitarlo.
5. Finalmente clicar en ‘Save changes’.

■ Añadir funciones al servicio

Una vez creado el servicio es necesario añadir funciones que aporten la funcionalidad buscada por la aplicación que se está construyendo.

1. Administration → Site Administration → Plugins → Web services → Add functions
2. En este punto, se observarán los distintos servicios que se hayan construido además del predefinido por la propia plataforma.
3. Habrá que clicar en ‘Functions’ del servicio externo al que se quiera añadir nuevas funcionalidades.
4. Esta visión muestra las funciones ya añadidas al servicio, su descripción y los permisos necesarios para utilizarlos.
5. Clicar en ‘Add functions’ y seleccionar la funcionalidad buscada.

■ Creación de un token de acceso

Después de realizar todos los pasos anteriores, es necesario crear un token de acceso vinculado a un usuario en específico para poder hacer uso de los servicios web, consiguiéndose de la siguiente forma.

1. Administration → Site Administration → Plugins → Web services → Manage tokens
2. Clicar en ‘Add’.
3. Seleccionar el usuario al que se quiere que esté relacionado el token.
4. Seleccionar el servicio al que se quiere acceder.
5. Se puede imponer restricciones de *IP* o fechas de vencimiento del token
6. Clicar en ‘Save changes’.

■ Habilitación de permisos

Dependiendo del rol de usuario al que se haya vinculado el token de acceso dispondrá de distintos permisos. Estos permisos pueden observarse en: Administration → Site Administration → Plugins → Web services → Check User Capability.

Las distintas funciones de los servicios web suelen requerir una serie de permisos para poder llamarlas y ejecutarlas. Es por esto que se aconseja vincular el token de acceso a un usuario administrador para no tener carencias de permisos. Aun así, el usuario debe satisfacer los siguientes requisitos mínimos:

- Disponer de la competencia para crear un token: *moodle/webservice:createtoken*
- Disponer de la competencia para entablar conexión según el protocolo escogido: *webservice/rest:use*, *webservice/soap:use*, *webservice/xmlrpc:use* o *webservice/amf:use*.
- Disponer de las competencias requeridas por las propias funciones del servicio web.

■ Obtención de la documentación de la *API* de *Moodle*

Este apartado no es indispensable para emplear los servicios web pero es bastante relevante analizar la documentación de la *API* ya que ahí se describen las funciones construidas por *Moodle*. Además, también se muestran los parámetros y formatos necesarios para las peticiones en función del protocolo que se utilice, se indica la estructura de la respuesta, etcétera. Dicha documentación se puede acceder mediante la siguiente ruta:

1. Administration → Plugins → Web service → API Documentation

■ Creación de un usuario y asignación de rol

Este apartado es útil en caso de querer crear un nuevo usuario con el que acceder a los servicios web de la plataforma.

1. Administration → Site Administration → Users → Add a new user
2. Rellenar toda la información correspondiente y en el campo ‘Choose an authentication method’ elegir la opción ‘Web service authentication’.
3. Como bien se ha comentado previamente, es aconsejable utilizar un usuario administrador para no tener problemas de permisos a la hora de acceder a los servicios web. Para asignar dicho rol, u otros que se requiera, se realiza en la siguiente ruta: Administration → Site Administration → Users → Permissions → Assign system roles
4. A partir de aquí, tan solo es necesario elegir el rol que se quiere asignar al usuario elegido.

Sin embargo, pese haber seguido los pasos aquí mostrados, pueden encontrarse errores a la hora de intentar establecer una conexión con la plataforma. Para obtener una mayor información acerca de ellos se puede cambiar el modo de *debugging* de *Moodle* a desarrollador, esto se encuentra en: Site Administration → Development → Debugging → Debug Messages. Normalmente, estos errores consisten en excepciones de control de acceso que, en la mayoría de los casos, se deben a alguna de las siguientes razones [10]:

- Usuario autorizado y creación del token: fallo de la restricción *IP* al autenticar al usuario o la fecha de validación ha expirado.
- El usuario no se encuentra en la lista autorizada.

- El usuario no goza de los permisos necesarios para emplear los servicios web.
- El sitio web se encuentra en modo de mantenimiento.
- El usuario está suspendido, eliminado o no confirmado.
- La autenticación del usuario está asignada a ‘nologin’ (tan solo sería editar su perfil).
- El servicio web se encuentra deshabilitado.
- El token de acceso no existe.
- La función que se quiere emplear no está incluida en el servicio web que se está llamando. Las funciones que tienen que estar incluidas vienen citadas en la sección posterior **[8.2.1.- Servicios web de Moodle]**.

Apéndice B

Manual de la herramienta: Aplicación extracción de datos

Este apartado se emplea para explicar tanto el manual de instalación de la aplicación como el manual de usuario que ayude a emplearla. El manual de instalación indica los requisitos técnicos, sistema operativo, etcétera, necesarios para instalar la aplicación, además de explicar todo lo necesario para conseguir e instalar el software de terceros que se precise. Al tratarse de una aplicación web, también se indica cómo se realiza el despliegue de la misma. Mientras que el manual de usuario contiene la información relevante, lo más completa y clara posible, dirigido a un usuario no avanzado sobre cómo utilizar la aplicación. Este manual equivale a la ayuda del programa.

B.1. Manual de instalación

El sistema operativo empleado para desarrollar la aplicación es *Windows 10* aunque, debido a que el lenguaje de programación empleado (*Python*) es multiplataforma, *App1-Moodle* se puede desplegar en otros SO como *Linux*. Sin embargo, se recomienda desplegarla en el SO *Windows* puesto que la ejecución de dicha aplicación no se ha probado en ninguna otra plataforma y se carece de conocimiento sobre alguna posible incompatibilidad al respecto.

Para poder ejecutar la aplicación es necesario disponer de *Python*, teniendo en cuenta que sea una versión compatible o igual a la aquí empleada (3.7.3). Además, se precisa tener descargadas e instaladas las bibliotecas empleadas durante el desarrollo, encontrándose nombradas en la sección **[2.2.- Herramientas utilizadas]**. De las bibliotecas citadas, la más importante es el *framework* *Flask* que permite realizar la aplicación web, propiciando este hecho la necesidad de disponer de los lenguajes *HTML*, *JS* y *CSS*.

Se dispone de un archivo *batch*, llamado *launcher*, que se emplea para lanzar la aplicación. Para ejecutar dicho fichero en *Windows*, basta con hacer doble clic en él y se abrirá directamente una terminal. Primero pide al usuario que indique con [Y/N] si ya dispone de todas las bibliotecas instaladas para en caso negativo instalarlas todas (por ejemplo si se diese el caso de que ha usado la aplicación en ocasiones anteriores no sería necesaria la instalación). Además de esto, en el fichero *.bat* se crean algunas variables de entorno y se ejecuta la aplicación desplegándola en la

ruta 127.0.0.1:5000, por lo que también se abre automáticamente el navegador *Chrome* (por lo que es necesario) en dicha dirección. Es posible que el navegador se abra más rápido que lo que tarda en desplegarse la aplicación, esto no supone ningún error puesto que tan solo es necesario refrescar la página pulsando F5 para que se muestre finalmente *App1-Moodle*.

En caso de que este fichero diese algún tipo de error, se recomienda instalar las bibliotecas manualmente mediante el comando `pip install1`; y otra forma de desplegar la aplicación se consigue abriendo una terminal en el mismo directorio donde se encuentra la app y ejecutando el comando `python app.py arg1`, correspondiendo *arg1* con la ruta del sistema al fichero *Rscript.exe*. Después de esto, la aplicación estará disponible en el navegador en la ruta 127.0.0.1:5000.

B.2. Manual de usuario

Al tratarse de una aplicación con vistas a ejecutarse pocas veces (ya que en cada ejecución se pretende obtener la mayor cantidad de información) provoca que los usuarios carezcan de excesiva experiencia con la interfaz. Por este motivo, el uso y la exploración de la misma es bastante sencillo.

La página principal, que se encuentra en la ruta 127.0.0.1:5000 equivalente al *endpoint* raíz ('/'), corresponde con la selección de modo de empleo de la aplicación. Esto se refleja en la Figura B.1, donde se puede escoger entre el rol de profesor o investigador. En función de esta acción, es adecuado focalizarse en la documentación mostrada en [B.2.1.-Manual de investigador] o en [B.2.2.-Manual de profesor].

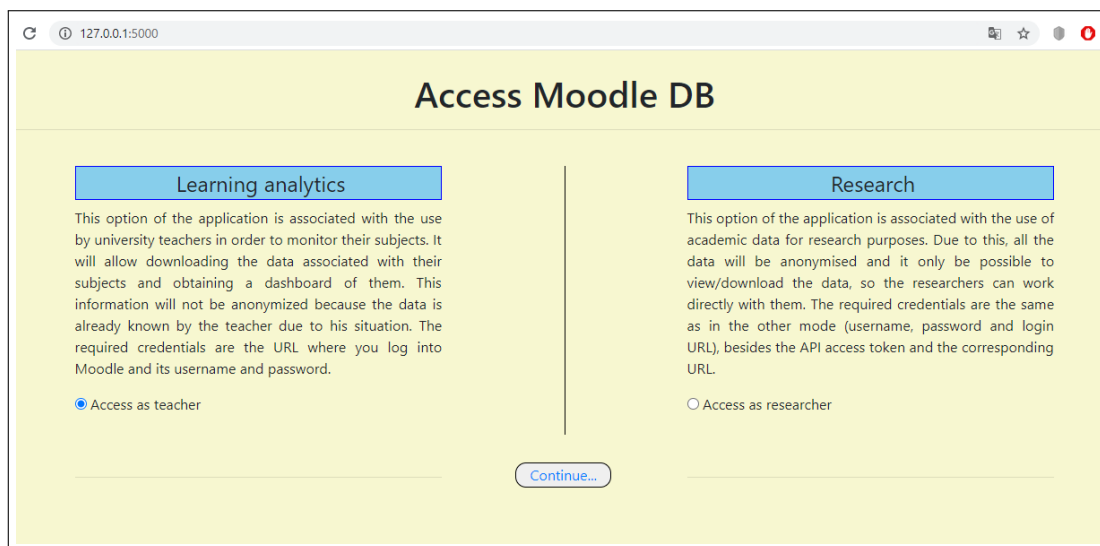
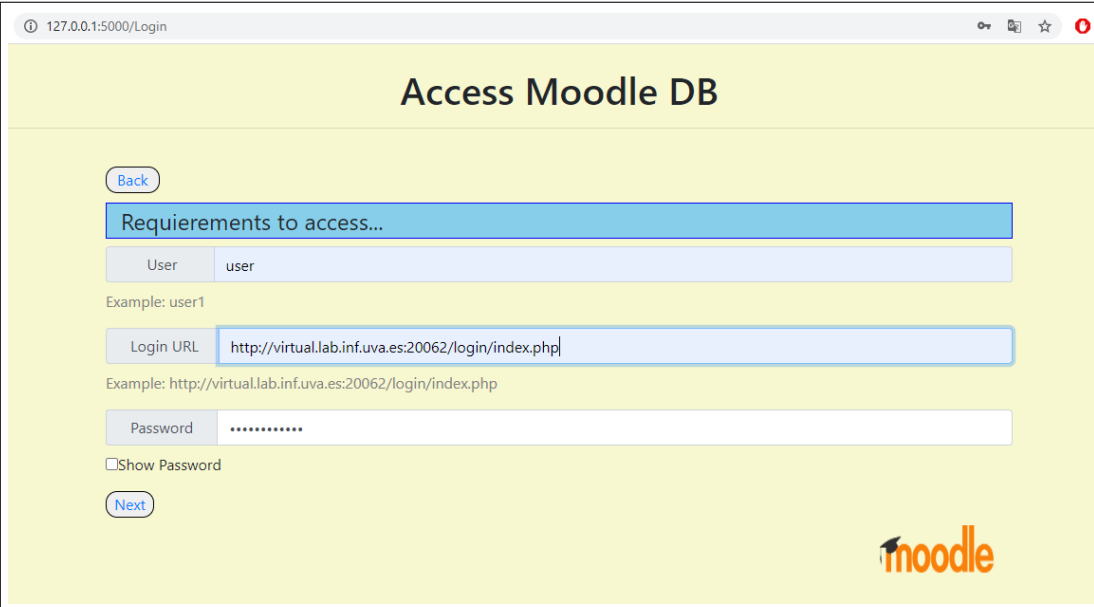


Figura B.1: Página de selección del modo de empleo.

¹Las bibliotecas necesarias se encuentran ordenadas alfabéticamente en la Tabla 2.1. Se recuerda que para instalarlas se precisa del comando `'pip install name_library'`.

B.2.1. Manual de investigador

La vista inicial corresponde con la introducción de los primeros credenciales (a saber, la URL de login, el usuario y contraseña de *Moodle*). Se trata de un formulario análogo a los habituales en las páginas web en el que también existe la posibilidad de visualizar la contraseña, se realiza así para que resulte familiar y, por lo tanto, más sencillo al usuario. La ruta ('/Login') se puede observar en la parte superior de la Figura B.2, mientras que las entradas corresponden con inputs de texto. Clicando en el botón 'Next' se avanzará a la siguiente pantalla de la interfaz.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/Login'. The main heading is 'Access Moodle DB'. Below this, there is a 'Back' button. A section titled 'Requirements to access...' contains three input fields: 'User' (containing 'user'), 'Login URL' (containing 'http://virtual.lab.inf.uva.es:20062/login/index.php|'), and 'Password' (containing masked characters). Below the password field is a checkbox labeled 'Show Password'. At the bottom of the form is a 'Next' button. The Moodle logo is visible in the bottom right corner.

Figura B.2: Página de introducción de credenciales (1).

La Figura B.3 muestra la etapa final de introducción de credenciales, pidiéndose aquí el token de acceso, la URI y el protocolo para realizar la petición. Las dos primeras corresponden con entradas de texto mientras que el protocolo se elige de una lista (destacar que tan solo se ha implementado el protocolo *REST* por lo que solo habrá una opción). Se puede observar en la parte superior cómo esta página corresponde con la ruta '/Auth' y que existe la posibilidad de volver atrás mediante el botón 'Back', o, tras haber rellenado todos los campos, clicar en el botón 'Connecting' para acceder a las bases de datos de *Moodle*. Cabe remarcar que tanto en esta pantalla como en la anterior se requiere que todos los campos estén rellenos para poder avanzar.

Una vez pulsado el botón 'Connecting' se realizarán las comprobaciones de las credenciales introducidas para ver si son válidas. En caso positivo se avanzará a la pantalla mostrada en la Figura B.5, mientras que en caso de credenciales erróneos se dará paso a la vista expuesta en la Figura B.4.

Esta última figura (Figura B.4) tan solo muestra al usuario que ha habido un problema al intentar acceder a las bases de datos de *Moodle* adornado con un gift. No especifica detallada-

mente cuál ha podido ser el error puesto que los motivos pueden ser muy variados y amplios. A consecuencia de esto, guía al usuario a leer la documentación de la aplicación (refiriéndose a esta memoria). En caso de tener problemas se recomienda al usuario que lea el apartado [A.-Configuración Moodle] donde se explicitan las configuraciones que hay que realizar en la plataforma *Moodle* y los errores que pueden ocurrir. De esta vista tan solo se puede volver a la página inicial (Figura B.2) mediante el botón ‘Back’.

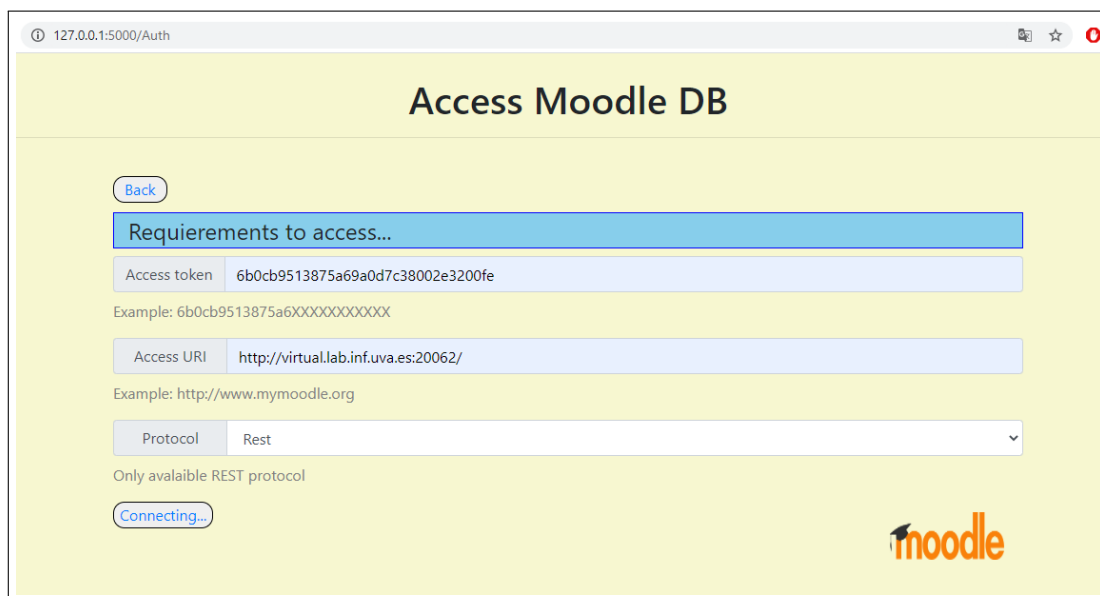


Figura B.3: Página de introducción de credenciales (2).

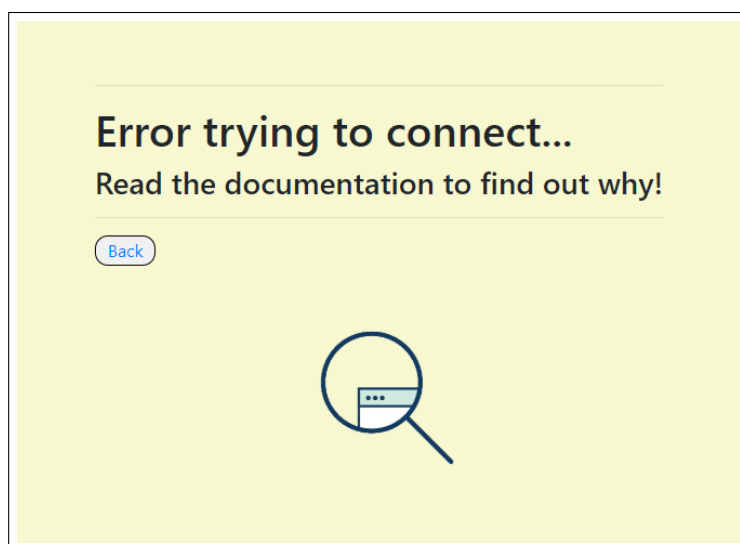


Figura B.4: Página de error en la conexión.

Como bien se ha comentado, en caso de que la configuración de *Moodle* y las credenciales sean válidas se da paso a la vista mostrada en la Figura B.5. Dicha página corresponde con la ruta ‘/check_connection/token/uri/protocolo’ y la interfaz se divide en un layout con dos partes claramente diferenciadas.

La parte derecha contiene una tabla con todas las asignaturas a las que está asociado el usuario que ha accedido al sistema. En dicha tabla se incluye tanto el identificador interno como el nombre del curso, además de haber un checkbox y un botón para cada fila. Los checkboxes sirven para seleccionar aquellas asignaturas de las que se quiere descargar datos. No obstante, si hay muchas asignaturas, la selección puede ser complicada, por lo que se dispone del botón ‘View’ para acceder a una nueva vista en la que se mostrará información detallada para la asignatura en cuestión.

La parte izquierda, además de disponer de un botón para volver a la introducción de credenciales, consta de una breve información sobre el contenido de la página. En suma, otro mecanismo para facilitar la selección de asignaturas, es la existencia de un buscador que permite encontrar cursos por su nombre. Finalmente, hay un botón llamado ‘Select courses’ que permite avanzar a la siguiente pantalla donde se podrá visualizar/descargar los datos asociados a las asignaturas seleccionadas mediante los checkboxes.

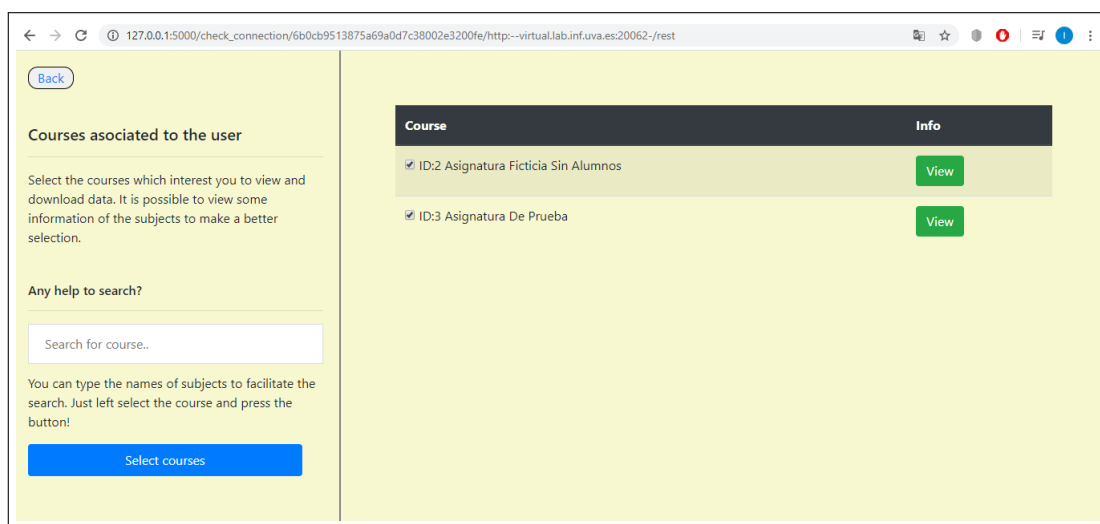
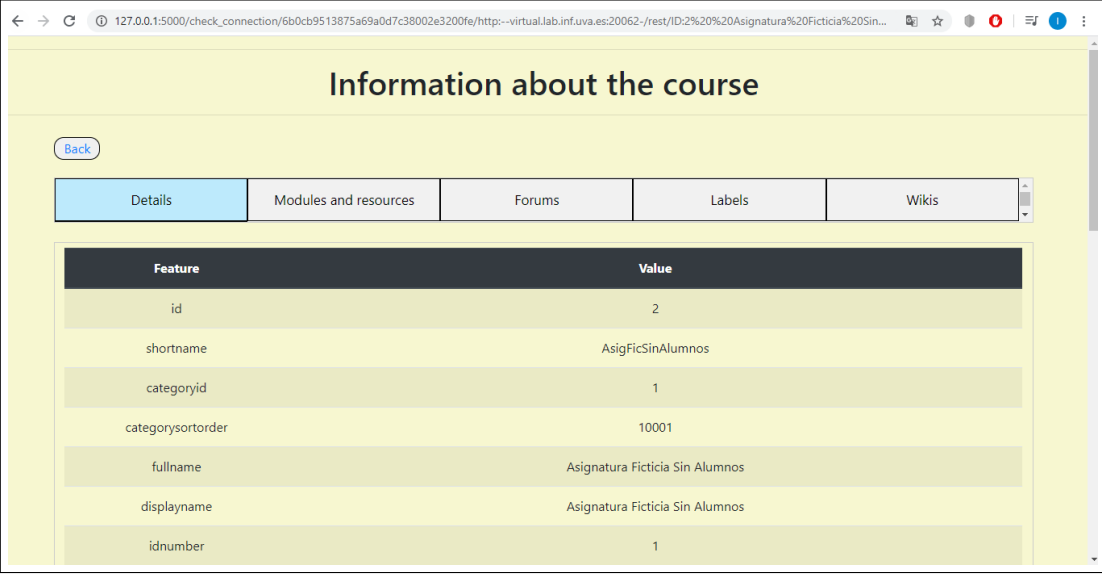


Figura B.5: Página de selección de cursos.

En caso de clicar en el botón ‘View’ enseñado en la Figura B.5, se mostraría la pantalla ilustrada en la Figura B.6. Dicha vista se encuentra en la ruta ‘/check_connection/token/uri/protocolo/id_course>’ y desde ahí solo es posible retroceder a la pantalla mostrada en la Figura B.5 mediante el botón ‘Back’.

Además, dispone de un conjunto de pestañas mostrando cada una de ellas información diferente en forma tabulada. Independientemente de la asignatura seleccionada, siempre va a haber información asociada a los detalles internos (‘Details’), a los usuarios relacionados (‘Users’) y

a los módulos/recursos de los que disponga ('Modules and resources'). A partir de ahí, la información de cada asignatura variará en función de su contenido, es decir, si consta de foros dispondrá de una pestaña que almacena información al respecto ('Forums'), lo mismo ocurrirá con las wikis ('Wikis'), etcétera. Esto se realiza con el fin de facilitar la decisión de selección de dicho curso. Se puede observar que el número de pestañas puede aumentar sin problemas ya que se dispone de un scroll para navegar sobre ellas.



Feature	Value
id	2
shortname	AsigFicSinAlumnos
categoryid	1
categorysortorder	10001
fullname	Asignatura Ficticia Sin Alumnos
displayname	Asignatura Ficticia Sin Alumnos
idnumber	1

Figura B.6: Página de información detallada del curso.

Si en la Figura B.5 se pulsa el botón 'Select courses', se avanza a la página donde ya se pueden visualizar/descargar los datos. Esta vista, cuya ruta es '/course_selection/token/uri/protocolo/ids_cursos', viene ilustrada en la Figura B.7 en la que se aprecia un layout que divide la interfaz en dos zonas diferenciadas.

La parte izquierda corresponde con el motor de búsqueda de información, siendo una entrada de texto para escribir algún carácter clave para encontrar los tipos de datos que se quieren obtener. Un ejemplo de uso sería buscar 'logs' o simplemente teclear la letra 'l' para encontrar la sección relativa a los logs; se ve cómo estas búsquedas se encuentran al final de la lista. El usuario también puede limitarse a desplegar la lista e ir buscando en ella el apartado que le interese. Como bien se ha comentado anteriormente, la aplicación está pensada para ser ejecutada pocas veces y descargarse la mayor cantidad de información posible, por esto hay una opción que permite descargar directamente todos los datos asociados. Además de esto, existe la posibilidad de retroceso a la vista de selección de cursos (Figura B.5) presionando el botón 'Back'.

En la parte derecha, inicialmente, no se muestra nada mas que un simple gift. Este espacio será destinado para permitir la visualización de los datos seleccionados. Es decir, esta vista en su conjunto, aunque localizándose en la misma ruta base, modificará su aspecto en función de las búsquedas de datos que se realicen (consiguiéndose mediante el uso de *query strings*). Estas variaciones en la interfaz son comentadas en las siguientes imágenes.

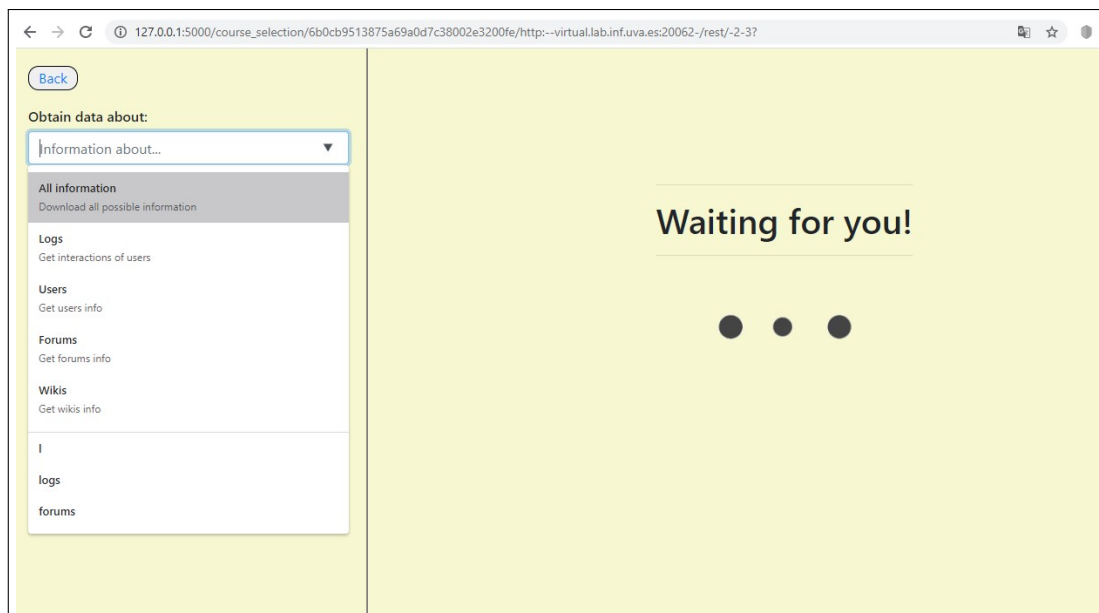
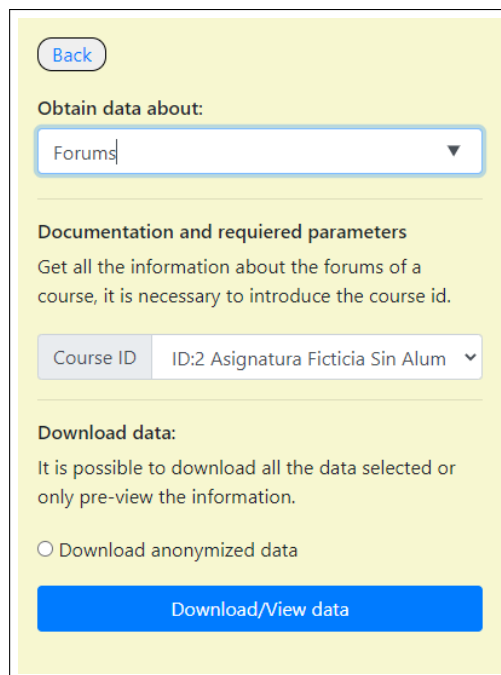


Figura B.7: Página de visualización/descarga de datos (1)

La Figura B.8 muestra cómo varía la interfaz de la zona del motor de búsqueda cuando, por ejemplo, se quiere descargar información sobre los foros. Se muestra una breve documentación y parámetros necesarios, en este caso, tan solo el ID del curso (el cual no es necesario saberlo, ya que se almacenan en una lista). Además, mediante un *radio button*, permite seleccionar si se quieren descargar los datos anonimizados (en caso negativo tan solo se visualizarán; en caso positivo se descargarán en formato *csv* separado por punto y coma en el directorio y con el nombre correspondientes). Tras realizar estas selecciones bastará con clicar el botón ‘Download/View data’ para descargar y/o visualizar la información anonimizada, siendo dicha visualización la mostrada en la Figura B.9.

En esta ocasión se aprecia cómo la asignatura ficticia consta de dos foros de los que se puede visualizar su información clicando en las respectivas pestañas. Cabe remarcar que para el resto de tipos de datos (e.g. logs, wikis, etcétera) el proceso y modo de empleo es análogo al mostrado con este ejemplo. Tan solo podría hacerse mención especial a los datos asociados a los logs puesto que son los que mayor utilidad tienen y mayor número de filtros pueden aplicarse, a saber: ID del curso, ID del usuario que ha realizado la interacción, ID del módulo afectado, tipo de acción cometida (estando las opciones almacenadas en una lista: borrado, creación, etcétera) o rol del usuario (almacenándose también en una lista: profesor, estudiante, etcétera). Esto se muestra en la Figura B.10 en la que se aprecia un scroll que permite seguir mostrando el resto de filtros y opcionalidades como el de descarga.



[Back](#)

Obtain data about:

Forums

Documentation and required parameters

Get all the information about the forums of a course, it is necessary to introduce the course id.

Course ID ID:2 Asignatura Ficticia Sin Alum

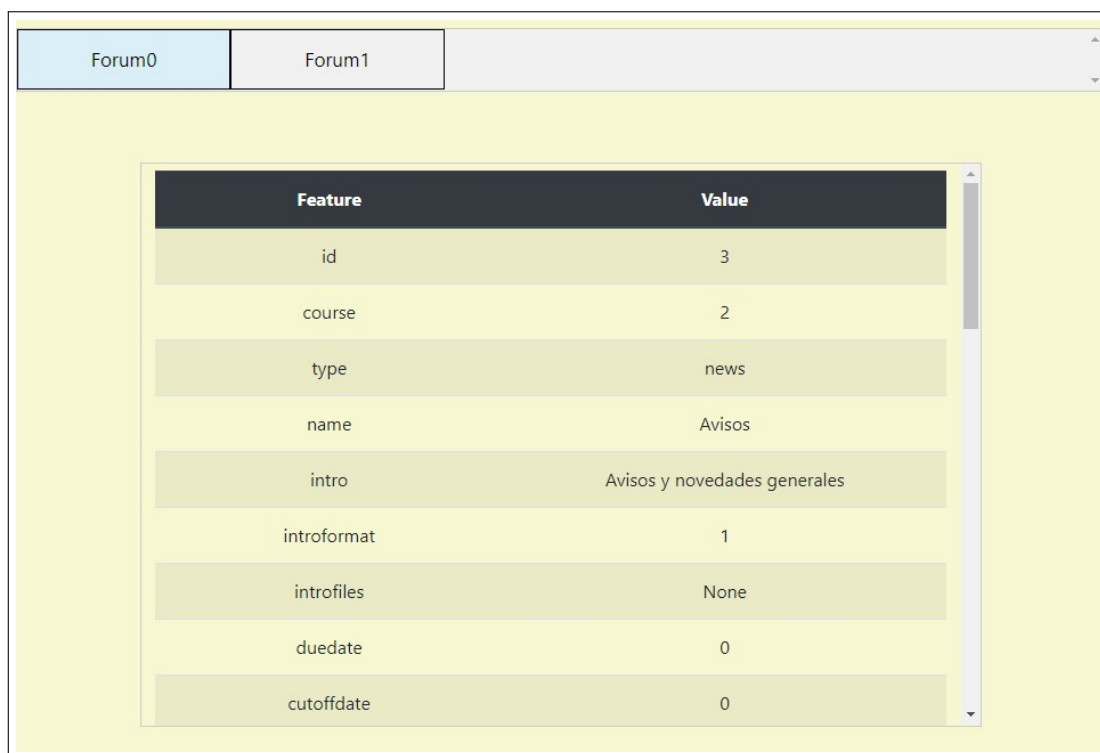
Download data:

It is possible to download all the data selected or only pre-view the information.

☐ Download anonymized data

Download/View data

Figura B.8: Página de visualización/descarga de datos (2)



Feature	Value
id	3
course	2
type	news
name	Avisos
intro	Avisos y novedades generales
introformat	1
introfiles	None
duedate	0
cutoffdate	0

Figura B.9: Página de visualización/descarga de datos (3)

Back

Obtain data about:

Logs and grades ▼

Documentation and required parameters

Get all the information about the logs which registered the activity of users in a course. It is possible to search specifying the userid, courseid, moduleid, type of action and event. For example, if the field of userid is empty, all users will be returned. Furthermore, you can get the marks of the users on that subject.

Course ID ID:2 Asignatura Ficticia Sin Alt ▼

User ID Type the user ID

Module ID Type the module ID

Actions All actions ▼

Figura B.10: Página de visualización/descarga de datos (4)

Finalmente, si en la búsqueda de los datos o en la aplicación de filtros ocurre algún problema como, por ejemplo, la inexistencia de dichos datos o la introducción de parámetros erróneos, se dará paso a la vista mostrada por la Figura B.11. El botón 'Back' permite el retroceso a la página de selección de información (Figura B.7).

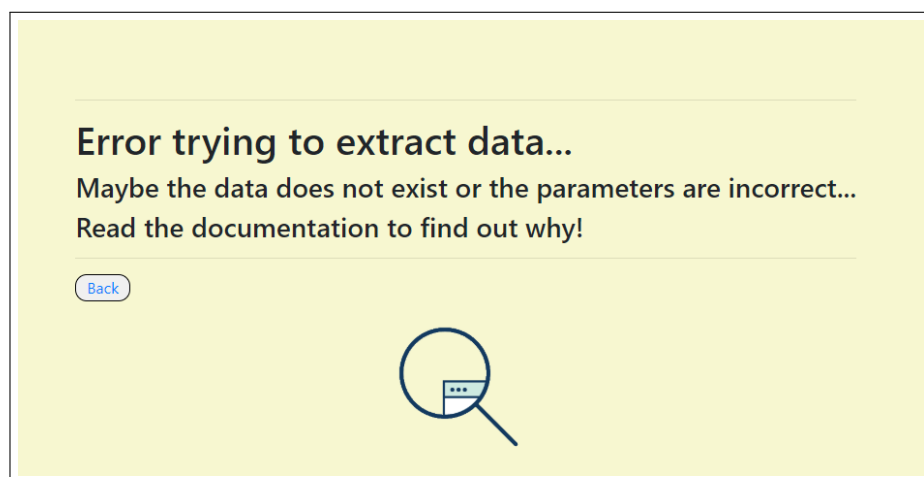


Figura B.11: Página de error en la búsqueda/descarga de datos.

B.2.2. Manual de profesor

Este manual es muy similar al anterior, la principal diferencia reside en que tan solo extrae información mediante *web scraping*. Por consiguiente, las credenciales necesarias son la URL de login, el usuario y la contraseña de *Moodle*.

Tras efectuar la selección de modo en el *endpoint* raíz mostrado en la Figura B.1, se avanza a la introducción de credenciales en la ruta *‘/Login’* reflejado en la Figura B.2. Hasta aquí el procedimiento es exactamente igual que para el modo de investigador, pero ahora, en vez de acceder a la introducción de más credenciales (Figura B.3), se accede directamente a la ruta *‘/course_selection/token/uri/protocolo/ids_cursos’* disponiendo de la misma interfaz que la expuesta en la Figura B.7. Aunque en este modo de empleo se dispone de distintos tipos de datos sobre los que buscar (tan solo logs y notas), la disposición gráfica tanto del motor de búsqueda como de la pre-visualización se mantiene equivalente a lo ya mostrado en las Figuras B.9 y B.10. La principal diferencia reside en que los datos descargados/visualizados no se encuentran anonimizados y en que se puede lanzar la aplicación *App2-Dash* (clicando un *radio button*) en la ruta 127.0.0.1:5001 donde se muestre el *dashboard* con la información que se ha descargado. En caso de haber algún error en la búsqueda de datos se redirigirá a la ventana mostrada en la Figura B.11.

Apéndice C

Manual de la herramienta: Aplicación dashboard

Al igual que antes, este apartado se emplea para explicar tanto el manual de instalación de la aplicación como el manual de usuario que ayude a emplearla. El manual de instalación indica los requisitos técnicos, sistema operativo, etcétera, necesarios para instalar la aplicación, además de explicar todo lo necesario para conseguir e instalar el software de terceros que se precise. Al tratarse de una aplicación web, también se indica cómo se realiza el despliegue de la misma. Mientras que el manual de usuario contiene la información relevante, lo más completa y clara posible, dirigido a un usuario no avanzado sobre cómo utilizar la aplicación. Este manual equivale a la ayuda del programa.

C.1. Manual de instalación

El sistema operativo empleado para desarrollar la aplicación es *Windows 10* aunque, debido a que el lenguaje de programación empleado (*R*) es multiplataforma, *App2-Dash* se puede desplegar en otros SO como *Linux*. Sin embargo, se recomienda desplegarla en el SO *Windows* puesto que la ejecución de dicha aplicación no se ha probado en ninguna otra plataforma y se carece de conocimiento sobre alguna posible incompatibilidad al respecto.

Para poder ejecutar la aplicación es necesario disponer de *R* descargado, teniendo en cuenta que sea una versión compatible o igual a la aquí empleada (3.5.0). Además, se precisa tener descargados e instalados los paquetes empleados durante el desarrollo, encontrándose nombrados en la sección **[2.2.- Herramientas utilizadas]**, más concretamente en la Tabla 2.2. De los paquetes citados, el más importante es *shiny* ya que permite desarrollar la aplicación web. Cabe remarcar que se ha implementado de tal forma que comprueba si el sistema dispone de los paquetes necesarios instalados y en caso negativo los instala, por lo que el usuario no debe preocuparse de nada. No obstante, en la situación de que esto diese algún tipo de error, se recomienda que se instalen los paquetes de forma manual accediendo a *R* y ejecutando el comando `install.packages("name_package")`. Además, es importante tener en cuenta que en *R* para poder instalar paquetes en la ruta especificada por el propio lenguaje de programación es necesario disponer de permisos de escritura (los cuales son disponibles por defecto por un administrador).

Una vez se tenga instalado todo lo citado, tan solo queda desplegar la aplicación arrancando el servidor. Esto se consigue ejecutando el script *Rmd*, ya sea mediante el IDE *Rstudio* o mediante línea de comandos. No obstante, teniendo en cuenta que esta aplicación se ha implementado de forma que se lance tras estar empleando *App1-Moodle* no es necesario utilizar ningún IDE o terminal, mejorándose así la usabilidad y facilitando un alto grado de dinamismo. Tan solo será necesario emplear esos métodos en caso de querer ejecutar esta aplicación independientemente de la otra.

Como la aplicación se concibe en conjunto con *App1-Moodle*, el usuario no tendrá ningún problema para desplegarla, es más, no tendrá que realizar nada para ejecutarla. Tan solo debe saber que tras indicar en una de las vistas de la interfaz de *App1-Moodle* (en modo profesor) que se quiere visualizar el *dashboard* se lanzará *App2-Dash* automáticamente en el buscador *Chrome* en la ruta 127.0.0.1:5001.

C.2. Manual de usuario

Al tratarse de un *dashboard* se ha desarrollado de tal forma que el uso y la exploración del mismo sea sencillo e intuitivo. La página principal, mostrada en la Figura C.1, se compone de un panel principal donde vienen resumidas las características de todas las pestañas principales que componen el *dashboard*. Esto pretende ayudar al usuario a entender y situarse en el contexto de lo que se está exponiendo, además de facilitar la búsqueda de lo que requiera. También se dispone de un *sidebar* en el lateral izquierdo donde consta un resumen general de la aplicación y un botón para cerrar la aplicación (una vez se pulse en él se terminará la ejecución y aparecerá una ventana en el navegador notificando al usuario que se ha finalizado la herramienta y que debe cerrar la pestaña del buscador). Este es el único método para finalizar la ejecución de la aplicación, es decir, si se cierra directamente la pestaña del navegador, la herramienta se quedará ejecutando en segundo plano.

De esta página principal también se puede apreciar la composición general de la herramienta, estando constituida por las pestañas ‘Home’ (ya explicada), ‘Exploratory’, ‘Time series’ y ‘Events’. Además, se puede observar cómo existe la posibilidad de visualizar el código de la aplicación y compartirlo por distintas vías de comunicación (estas opciones se encuentran en la parte superior derecha).

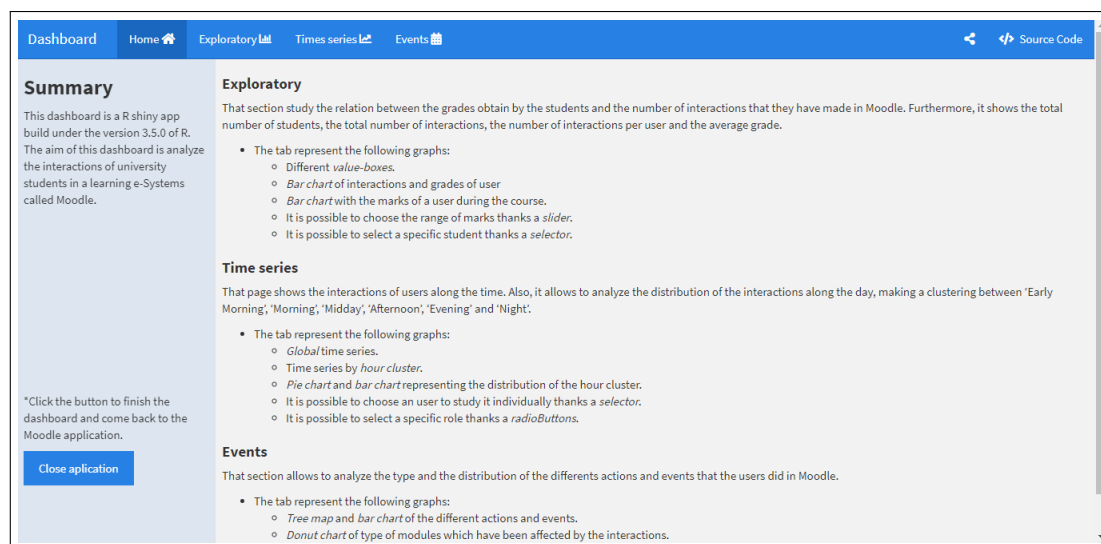


Figura C.1: Home page del dashboard.

Accediendo a la pestaña ‘Exploratory’ se observaría la composición de gráficos mostrados en la Figura C.2. Esta pestaña muestra, en diferentes *value-box*, el número total de alumnos de la asignatura, el número total de interacciones realizadas en la misma, el número medio de interacciones por usuario y la nota media del conjunto de la clase. Estos *value-box* no son interactivos, tan solo muestran la información de manera estática, a no ser que se modifique el *selectInput* del *sidebar* lateral y se elija a un alumno en concreto por lo que cambiaría los valores de dichos *value-box*.

Siguiendo con el contenido del *sidebar*, además de una pequeña explicación y de la entrada para seleccionar a un alumno determinado, también se encuentra un *slider* para indicar el rango de notas que se quiere mostrar en los gráficos. Estas entradas de usuario afectan a los gráficos mostrados en centro del layout. En dicha zona, se aprecian dos pestañas secundarias: ‘Relation between grades and interactions in Moodle’ y ‘Grades during the course’. La primera de ellas representa un *bar chart* compuesto interactivo donde se muestra el número de interacciones y la nota media por alumno (pudiéndose filtrar por rango de nota gracias al *slider* y por alumno mediante el *selectInput*). Las posibilidades de interacción, en este y casi en la mayoría de gráficos que componen el *dashboard*, han sido explicadas en el apartado [14.1.2.-Plotly] por lo que no se abordarán aquí. La segunda pestaña también se trata de un *bar chart* que ilustra la evolución de un alumno en concreto a lo largo del curso, es decir, muestra las notas obtenidas por dicho alumno en la asignatura. Para que este gráfico se muestre es necesario que en el *selectInput* haya sido escogido un alumno concreto en vez de estar la opción de todos los estudiantes. En caso de que esté esta opción, se expondrá un gráfico en blanco indicando que se precisa seleccionar un alumno en concreto para procesar el gráfico.

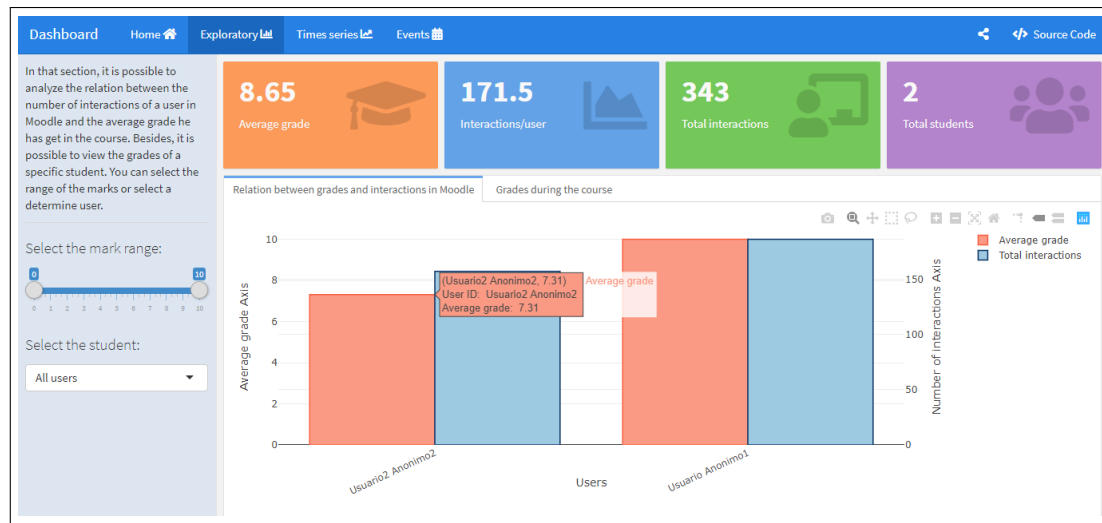


Figura C.2: Pestaña 'Exploratory' del dashboard.

En la Figura anterior se muestra la composición por defecto de esta pestaña, pero en caso de seleccionar la pestaña secundaria 'Grades during the course' y escoger un alumno determinado en el *selectInput*, el gráfico mostrado será el de la Figura C.3 donde se representa la evolución académica de dicho alumno a lo largo del curso (en esta ocasión al ser ficticio no hay nombre para las entregas/pruebas y simplemente son numeradas).

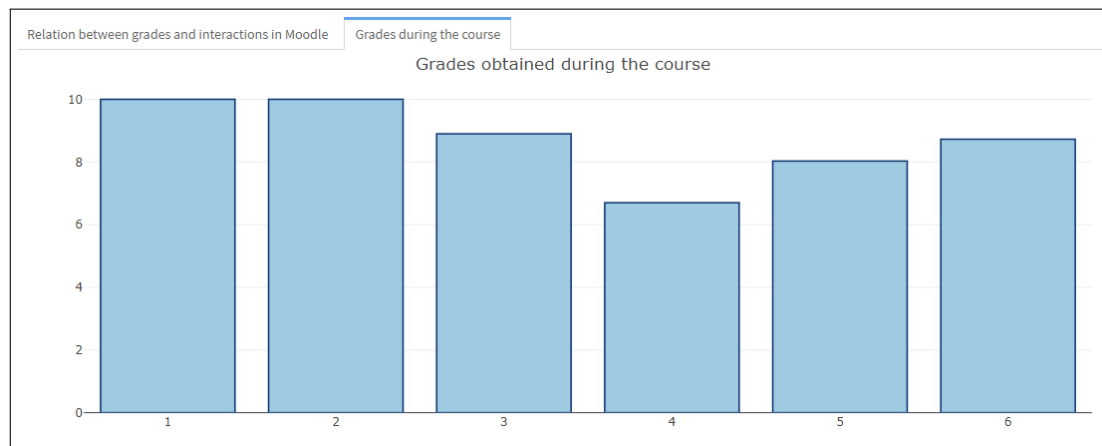


Figura C.3: Pestaña 'Exploratory' del dashboard (2).

La pestaña principal de 'Time series' corresponde con la totalidad de información temporal de los logs de la asignatura. Al igual que antes, dispone de un *sidebar* lateral donde se permite seleccionar un alumno determinado (*selectInput*) o un rol general (*radio button*) que sirvan de filtro para realizar los gráficos. La parte principal del layout consta de tres pestañas secundarias, estando todas afectadas por los filtros previamente citados: 'General', 'By hour clustering' y 'Distribution by hour clustering'. La primera de ellas, ilustrada junto con el resto de la composición en la Figura C.4, corresponde con una serie temporal que representa el número de interacciones

realizadas por día. Esto permite identificar posibles patrones de interacción o deducir qué meses son más laboriosos para los alumnos.

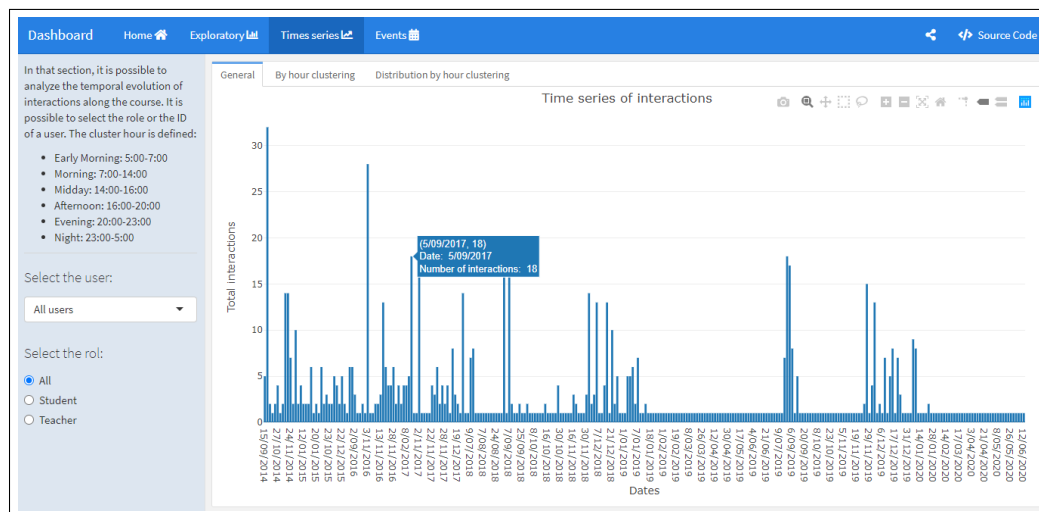


Figura C.4: Pestaña ‘Time series’ del dashboard.

El anterior gráfico se trata de una serie temporal bastante general que se complementa con la segunda pestaña secundaria, ‘By hour clustering’, que realiza la misma representación pero distinguiendo por cluster horario (Figura C.5). Los grupos creados son seis (Early Morning, Morning, Midday, Afternoon, Evening, Night) y vienen definidos y especificados en el *sidebar* lateral. Además, existe la posibilidad de navegar a través de la secuencia temporal mediante un *slider*, pudiendo indicar el tamaño de la ventana temporal mediante los botones que se encuentran en la parte superior izquierda del gráfico (indicando la longitud de 3 meses, 6 meses, 1 año, etcétera) o bien de forma manual clicando directamente en el gráfico o en el *slider*.

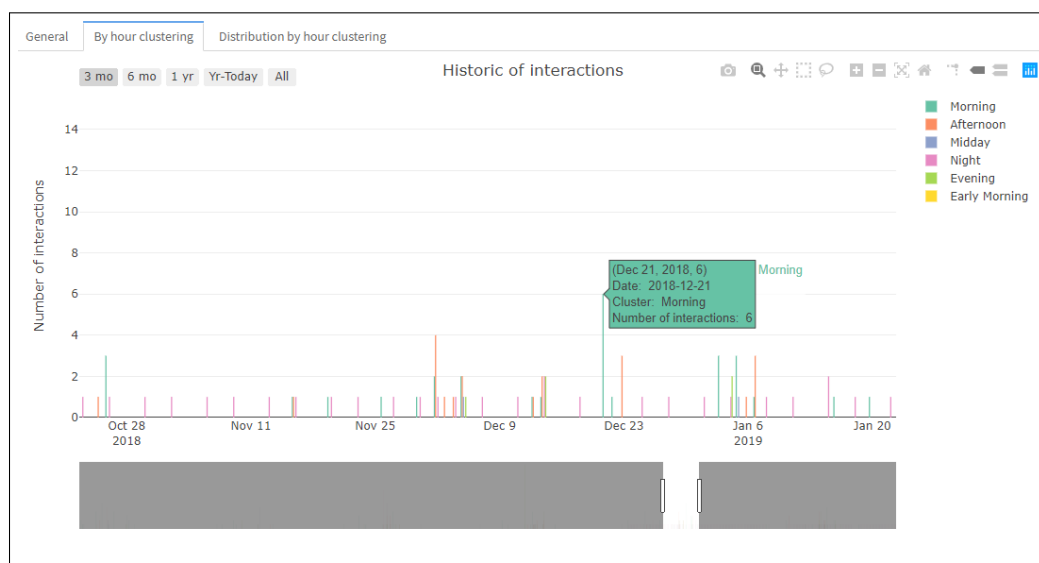


Figura C.5: Pestaña ‘Time series’ del dashboard (2).

Terminando de explicar esta pestaña principal, en la Figura C.6 se expone el gráfico correspondiente a la pestaña secundaria ‘Distribution by hour clustering’. Esta ilustración está formada por un gráfico compuesto de un *piechart* central y dos *barchart* laterales. Esto muestra la distribución de las interacciones en los clusters horarios tanto en números absolutos como en porcentajes, pudiendo extraer también conocimiento sobre el reparto de las acciones entre la primera parte del día y la segunda. Al igual que en los gráficos anteriores, se ve afectado por los filtros localizados en el *sidebar*.

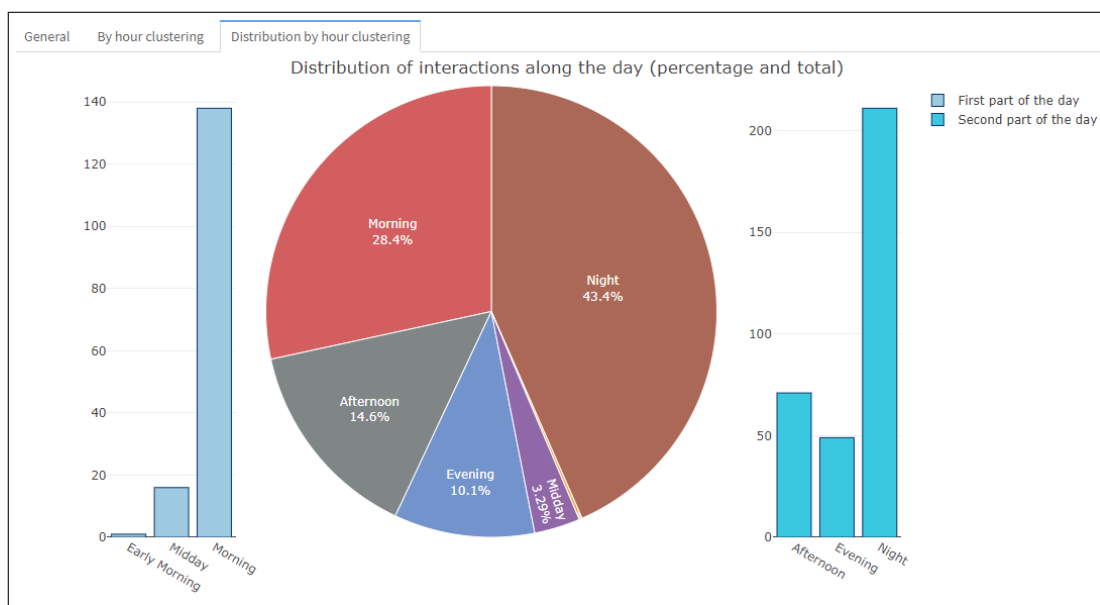


Figura C.6: Pestaña ‘Time series’ del dashboard (3).

Accediendo a la última pestaña mostrada en la Figura C.7, llamada ‘Events’, se observa cómo la composición es similar a las anteriores ya mostradas. Se desarrolla análogamente con el fin de que sea intuitiva y fácilmente recordable y usable por parte del usuario. El *sidebar* lateral muestra la posibilidad de los mismos filtros explicados en pestañas previas. Esta vez, el centro del layout está dividido en dos partes: la izquierda conformada por las pestañas secundarias ‘Actions treemap’ y ‘Actions barchart’ y la de la derecha formada por ‘Type of module/resource affected’ y ‘Especified module/resource affected’.

Gracias a la Figura C.7 se observan las pestañas secundarias ‘Actions treemap’ y ‘Type of module/resource affected’. La primera representa un *treemap* de los tipos de acciones realizadas por los usuarios en las interacciones con la asignatura. Se trata de un árbol interactivo con dos niveles de profundidad en el que se muestra el total de interacciones de un determinado tipo de acción. Mientras que ‘Type of module/resource affected’ expone mediante un *donut chart* la distribución del tipo de módulos y recursos afectados por las interacciones realizadas por los usuarios.

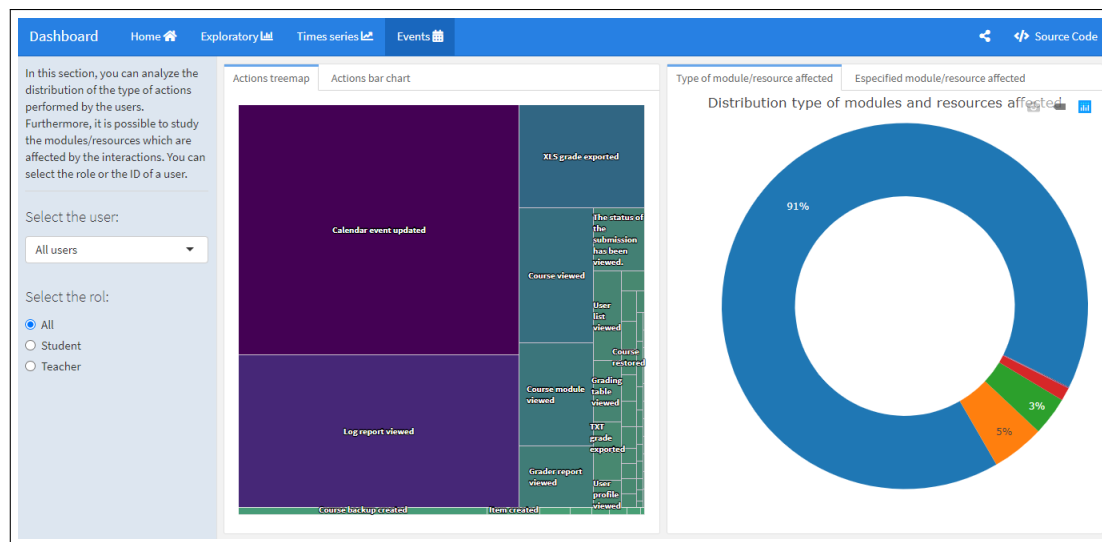


Figura C.7: Pestaña 'Events' del dashboard.

Finalizando así la guía de usuario del *dashboard*, la Figura C.8 expone las pestañas secundarias 'Actions barchart' y 'Type of module/resource affected'. La primera representa un sencillo *barchart* que indica el número total de cada uno de los tipos generales de acciones, complementando así al *treemap*. Mientras que la segunda pestaña representa una especificación del anterior *donut chart* en el que se expone las veces que un determinado módulo/recurso es afectado por una interacción. Este último gráfico es útil para que un profesor conozca qué recursos (PDFs, Words, etcétera) o qué módulos (entregas, foros, etcétera) son más utilizados por los alumnos.

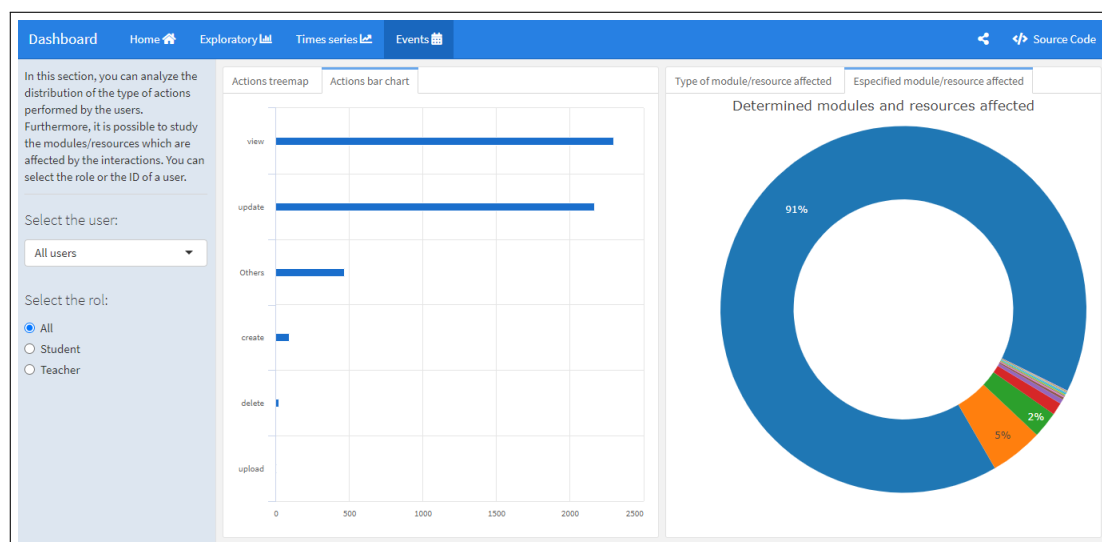


Figura C.8: Pestaña 'Events' del dashboard (2).